UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE AUTOMAÇÃO E SISTEMAS

Gerson Fernando Budke

ABORDAGENS DE COMUNICAÇÃO PARA FUSÃO DE DADOS PARALELA EM REDES DE SENSORES SEM FIO IEEE 802.15.4 EM TOPOLOGIA ESTRELA

Florianópolis(SC)

2012

Gerson Fernando Budke

ABORDAGENS DE COMUNICAÇÃO PARA FUSÃO DE DADOS PARALELA EM REDES DE SENSORES SEM FIO IEEE 802.15.4 EM TOPOLOGIA ESTRELA

Dissertação submetido ao Programa de Pós–Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Engenharia de Automação e Sistemas, Área de Concentração em *Controle, Automação e Sistemas*.
Orientador: Carlos Barros Montez, Dr.
Coorientador: Ricardo Alexandre Reinaldo de Moraes, Dr.

Florianópolis(SC)

2012

Gerson Fernando Budke

ABORDAGENS DE COMUNICAÇÃO PARA FUSÃO DE DADOS PARALELA EM REDES DE SENSORES SEM FIO IEEE 802.15.4 EM TOPOLOGIA ESTRELA

Este Dissertação foi julgado aprovado para a obtenção do Título de "Mestre em Engenharia de Automação e Sistemas, Área de Concentração em *Controle, Automação e Sistemas*", e aprovado em sua forma final pelo Programa de Pós–Graduação em Engenharia de Automação e Sistemas da Universidade Federal de Santa Catarina.

Florianópolis(SC), 31 de Agosto 2012.

Jomi Fred Hübner, Dr.
Coordenador do Programa de Pós–Graduação em Engenharia de Automação e Sistemas

Banca Examinadora:

Carlos Barros Montez, Dr.
Orientador

Ricardo Alexandre Reinaldo de Moraes, Dr.
Coorientador

Frank Siqueira, Ph.D.

Patricia Della Méa Plentz, Dra.

Cristian Koliver, Dr.

A minha amada esposa que sempre me apoiou
e esteve ao meu lado...

AGRADECIMENTOS

Aos meus pais, Elibio e Salete, que me ensinarem a sempre persistir. A minha esposa, Vanessa, que sempre me ajudou e tornou possível realizar este trabalho. Ao meu irmão, Jean, por sempre me apoiar. Ao meu orientador, Carlos Barros Montez, por permitir que eu executasse este trabalho com liberdade e por suas incontáveis correções e sugestões. Ao meu coorientador, Ricardo Moraes, que colaborou com a análise de estatística do algoritmo do centro de fusão de dados e ajudou na elaboração deste documento. Ao PPGEAS e ao CNPQ que colaboraram na publicação deste trabalho. A todas as pessoas que de alguma forma tornaram este sonho possível e não foram citadas.

RESUMO

Neste trabalho é proposta uma abordagem de comunicação para controlar a probabilidade de envio dos nodos IEEE 802.15.4 visando a fusão de dados paralela em uma rede de sensores sem fio. O objetivo é ajustar o número de mensagens enviadas pelos sensores estabelecendo uma solução de compromisso entre reduzir o tráfego de mensagens e garantir que, em média, um número suficiente de mensagens alcance o centro de fusão de dados. Este número de mensagens deve garantir que o centro de fusão execute seu algoritmo com um determinado grau de confiabilidade. Técnicas convencionais como Dempster-Shafer ou Filtro de Kalman são exemplos de técnicas de fusão de dados que podem ser utilizadas em conjunto com esta abordagem. Para validar a proposta, uma solução de hardware e software foi desenvolvida, a fim de se obter um melhor desempenho energético, e permitir alcançar os objetivos. Os experimentos revelaram que o algoritmo proposto consegue elevar o tempo de vida da rede com a economia de energia alcançada. Os resultados obtidos mostraram a efetividade da proposta, principalmente para monitoramento em ambientes com informações redundantes, uma vez que mais nodos conseguem economizar energia.

Palavras-chave: Redes de Sensores sem Fio. IEEE 802.15.4. Fusão de dados.

ABSTRACT

In this research work is proposed a novel communication approach to control the sending probability of the IEEE 802.15.4 nodes to achieve parallel data fusion in Wireless Sensor Networks. This approach aims at adjusting the number of sent messages by the sensor nodes, establishing a tradeoff between the number of sent messages and the quality of data fusion. The number of messages received by the fusion center must guarantee a determined degree of reliability. Conventional data fusion techniques, such as Demspter-Shafer or Kalmann Filter, can be employed using the proposed approach. A prototype was implemented and experimental results show the effectiveness of this proposal.
Keywords: wireless sensor network. IEEE 802.15.4. data fusion.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

As redes de sensores sem fio (RSSF) ganham importância crescente a cada ano que passa. Essas redes foram propostas, originalmente, para aplicações de rastreamento de objetos (*tracking*) e, principalmente, para realizar monitoramento de sistemas. As grandezas monitoradas são as mais variadas e dependentes das aplicações; dentre elas destacam-se: monitoramento de gases, detecção de chamas, ruídos, vibrações e temperaturas em processos industriais e em transporte de líquidos por gasodutos, batimentos cardíacos e níveis de glicose em aplicações relacionadas à saúde humana (*health care*).

Uma RSSF é composta por nodos sensores com capacidade de comunicação sem fio (em Inglês esses dispositivos costumam ser denominados *motes*). Nessas redes geralmente há um nodo especial conhecido como coordenador, que é responsável pelo gerenciamento e manutenção da rede – entrada e saída de dispositivos. Assim, uma rede de sensores, na sua forma de organização mais simples, pode ser entendida como um sistema de monitoramento composto por um coordenador central e por vários outros nodos sensores, os quais se comunicam com o coordenador enviando os dados monitorados e recebendo os parâmetros de configuração. Esta forma de organização, na qual os nodos não possuem hierarquia entre si (com exceção do nodo coordenador), traz diversas vantagens e é usualmente conhecida como *topologia em estrela*.

Na grande maioria das situações, os dispositivos RSSF são alimentados por baterias ou fontes alternativas de energia. Nos casos onde baterias são utilizadas, o aumento no número de dispositivos na rede traz um problema de manutenção e custo. Além disso, devido à implantação em lugares de difícil acesso aliada a um grande número de nodos, a simples operação de substituição das baterias pode ter um custo proibitivo. Por conseguinte, nos dias de hoje, a preocupação com equipamentos ambientalmente corretos e que consomem cada vez menos energia também é cada vez maior na indústria.

A comunidade científica, identificando estas preocupações, busca soluções para tentar minimizar esses problemas. Uma delas foi desenvolvida pelo *Institute of Electrical and Electronics Engineers* (IEEE), que em 2003 publicou a norma IEEE 802.15.4 (IEEE-802.15.4, 2011), a qual padroniza a camada de nível I (camada Física) e a subcamada MAC do nível II, segundo o modelo OSI (*Open System Interconnection*), para dispositivos de redes pessoais sem fio (*wireless personal networks*). Os dois principais objetivos deste padrão são: baixo custo e baixo consumo energético. Em contrapartida, a taxa de transmissão obtida é baixa. Apesar desse padrão não ter sido projetado especificamente para RSSF, desde então essa tecnologia começou a ser

cada vez mais explorada nesse tipo de rede.

A *Zigbee*[1] *Alliance* é uma associação composta de membros interessados em desenvolver padrões com alto grau de liberdade e flexibilidade para permitir soluções com equipamentos mais inteligentes e mais sustentáveis. A primeira versão do padrão desta organização foi lançada em 2004, a qual adotou a camada física do padrão IEEE 802.15.4. O padrão Zigbee contempla um protocolo de comunicação que define perfis de aplicação para uso geral e aberto. Assim, dispositivos certificados de diferentes fabricantes podem interoperar e se comunicar. Os perfis existentes são nove, os quais cobrem desde aplicações de monitoramento residencial e predial, passando por aplicações comerciais com estrutura de pagamento on-line, terminando por aplicações de monitoramento de saúde. No ano de 2011, com o lançamento de uma nova especificação voltada para energia eficiente, um relatório foi emitido pela Zigbee Alliance apontando que no ano de 2010 a quantidade de venda dos dispositivos Zigbee dobrou e o acumulado dos últimos 5 anos cresceu 800%, o qual em 2010 representou 40% dos semicondutores fabricados com tecnologia IEEE 802.15.4 (ZIGBEEORG, 2011; WORLD'S, 2011).

Esta dissertação de mestrado propõe uma abordagem de comunicação para auxiliar no processo conhecido como fusão de dados paralela em RSSF, onde os dados coletados pelos nodos sensores são enviados, utilizando um canal de comunicação compartilhado e sem fio, para um nodo coordenador, que é responsável pelo recebimento dos dados e pelo processo de fusão.

1.1 MOTIVAÇÕES E CONTEXTUALIZAÇÃO

O grande aumento na adoção de equipamentos Zigbee é um exemplo de como a tecnologia de comunicação sem fio voltada para RSSF vem crescendo. Esse volume de RSSF apresenta muitas oportunidades de uso nas mais diversas aplicações, trazendo, como contrapartida, desafios a serem transpostos. Um dos principais desafios é a busca por soluções que economizem energia dos nodos e que, simultaneamente, aproveitem melhor as condições que a própria aplicação e o ambiente monitorado oferecem.

As soluções para economia energética podem ser divididas, de uma maneira geral, em soluções por projeto de hardware e soluções por software. No projeto de hardware é necessário decidir a respeito dos componentes utilizados, suas disposições e interligação. A escolha de um determinado componente é um desafio pois precisa levar em consideração sua durabilidade, consumo e tamanho. Estes fatores implicam diretamente no custo dos materiais e, dependendo das escolhas tomadas, o custo pode tornar o projeto

[1] `http://www.zigbee.org/`

inviável. Como exemplo, muitos dispositivos trabalham dentro da faixa de 3V e as baterias são ligadas diretamente aos componentes. Isso faz com que a carga das baterias se esgote rapidamente. Um projeto de uma fonte DC/DC pode aumentar o tempo de vida de um nodo em vários dias. Da mesma forma que as baterias, os projetos convencionais não se preocupam com o consumo de componentes como os LEDs, cujo consumo é por volta de 65 à 100mW. A simples escolha de LEDs de alta eficiência pode reduzir este consumo para menos de 10mW sem perda de desempenho. Finalmente, outra solução simples é ter disponível na interface de expansão uma *chave* para ligar/desligar o circuito externo e assim evitar o disperdício de energia. Os circuitos que podem se beneficiar dessa técnica são, por exemplo, as entradas e saídas analógicas.

Algumas soluções de software mais utilizadas para economia de energia nos nodos envolvem abordagens que ou desligam alguns circuitos dos nodos, colocando-os em inatividade durante um determinado tempo, ou ajustam o controle de potência dos rádios. No entanto, esta última abordagem geralmente é empregada em topologias onde os nodos se comunicam com múltiplos saltos, e estão fora do escopo desta dissertação, a qual aborda RSSF em topologias estrela.

Este trabalho está centrado nas RSSF baseadas no padrão IEEE 802.15.4 (IEEE-802.15.4, 2011), conforme vêm sendo propostas e adotadas em ambientes industriais (WILLIG, 2008). Nestes cenários os dados coletados pelos nodos sensores são disseminados na rede pelo canal sem fio – compartilhado por todos os dispositivos – para o nodo coordenador (uma estação base), o qual é responsável por recebê-los, tratá-los e, possivelmente, encaminhar os dados ou o resultado deste processamento para seu destino usando outro tipo de rede (por exemplo, uma rede cabeada).

Com relação às grandezas monitoradas, a maior parte desses sistemas são classificados como *homogêneos*, pois os sensores monitoram o mesmo fenômeno, no qual há redundância espacial com relação ao número de sensores implantados (PATIL; DAS, 2004; CHIUSO; SCHENATO, 2011). Desta forma, pode-se assumir que em um determinado intervalo de tempo, o nodo coordenador desta rede não necessita receber todas as mensagens de todos os nodos para alcançar um determinado nível de confiança sobre o fenômeno que está sendo monitorado no ambiente: apenas um subconjunto dessas mensagens é suficiente (NAKAMURA; LOUREIRO; FRERY, 2007; PATIL; DAS, 2004).

Devido ao baixo custo dos nodos, seus sensores são usualmente pouco confiáveis e, dessa forma, assume-se que os dados monitorados possuem um determinado grau de imprecisão. Como, neste trabalho, adotam-se abordagens que buscam colocar alguns nodos em intervalos de inatividade para eco-

nomizar energia, existe uma necessidade de se estabelecer uma solução de compromisso entre colocar o maior número possível de nodos em inatividade e, por outro lado, fazer com que o nodo coordenador receba pelo menos M mensagem de dados de diferentes sensores para alcançar um nível de confiança sobre o fenômeno monitorado, onde o valor M possa ser configurado pelo administrador da rede ou ajustado automaticamente em tempo de execução.

A técnica de fusão de dados é utilizada para aprimorar as informações extraídas dos dados recebidos pelo coordenador. Isso acaba por melhorar a tomada de decisão do sistema uma vez que a decisão é tomada com base na amostragem dos vários sensores combinados e não apenas com o único valor de um sensor (D'COSTA; SAYEED, 2003; LIU; MOURA, 2004; NICHOLSON, 2004; PATIL; DAS, 2004; PAI; HAN, 2006). A fusão de dados também permite que seja realizada a fusão de sensores homogêneos e/ou heterogêneos, como, por exemplo, GPS e Radar para se obter uma posição (x, y, z) no espaço tridimensional, e diversas aplicações de monitoramento utilizam diferentes tipo de sensores. Este trabalho de mestrado busca utilizar esta técnica para gerar uma estimativa do número de amostras necessárias para efetuar uma fusão de dados com um nível de qualidade. Com esta estimativa buscase minimizar o tráfego da rede e, consequentemente, economizar energia.

Importante ressaltar que as RSSF, normalmente, operam em um ambiente altamente dinâmico, onde o número de nodos sensores que compõe a rede pode mudar ao longo do tempo, seja por falhas nos enlaces de comunicação por ruídos aleatórios no meio ou pela mobilidade dos nodos. Nesse cenário, abordagens estáticas para escalonamento das atividades dos nodos, por exemplo baseadas em TDMA, não são adequadas para serem implementadas.

1.2 OBJETIVO GERAL

O principal objetivo desta dissertação de mestrado é propor uma abordagem de comunicação para fusão paralela de dados em RSSF.

1.2.1 Objetivos Específicos

Este trabalho possui os seguintes objetivos específicos:

(a) Apresentar os principais conceitos relacionados com RSSF e descrever o padrão 802.15.4;

(b) Apresentar os principais modelos de fusão de dados;

(c) Propor uma solução de hardware para implementar as abordagens propostas;

(d) Implementar as técnicas propostas no hardware desenvolvido;

(e) Avaliar experimentalmente a solução proposta.

1.3 METODOLOGIA

Para alcançar os objetivos deste trabalho, iniciou-se pelo estudo de artigos, normas e relatórios técnicos a respeito de: a) redes de sensores sem fio; b) padrão IEEE 802.15.4; c) fusão de dados e d) documentação do hardware disponível na universidade.

A partir dessas informações, será definida uma abordagem de comunicação que deverá consistir em uma solução tanto no nível de hardware quanto a nível de software, buscando melhorar os ganhos do sistema do ponto de vista energético. A abordagem será executada sobre uma RSSF, configurada em topologia estrela e equipados com hardware escolhido.

O coordenador da rede será um dispositivo que executará duas funções básicas: sincronizar a rede através de envio de beacons, conforme o padrão IEEE 802.15.4, e funcionar como um centro de fusão de dados, recebendo mensagens dos nodos e tomando decisões baseadas nestas informações.

Os nodos têm a função de coletar uma grandeza e transmitir ao centro de fusão (coordenador). O algoritmo dos nodos deverá apresentar mecanismos para controle do consumo de energia.

A solução de software (algoritmo de fusão de dados e economia com os nodos) deverá executar um algoritmo, distribuído, que permite uma economia média de energia sem comprometer a qualidade da estimação da rede que terá como base as grandezas monitoradas pelos nodos. No entanto, diferentemente de outros trabalhos sobre fusão de dados que foram encontrados na literatura, o enfoque desta dissertação não está na técnica de fusão de dados adotada, como (ex., estimação por filtro de Kalman (CHIUSO; SCHENATO, 2011), teoria Bayesiana, Dempster-Shafer
(NAKAMURA; LOUREIRO; FRERY, 2007) ou técnicas de votação (HOSEINNEZHAD; BAB-HADIASHAR, 2006)). O enfoque está na proposição de uma abordagem da comunicação para a disseminação de dados periódicos para o nodo coordenador, que é responsável pelo processo de fusão de dados. Esta abordagem busca aplicar técnicas de fusão de dados visando atender, simultaneamente, os seguintes objetivos: garantir que um número mínimo de

mensagens alcancem o coordenador da rede; maximizar o tempo de vida dos nodos; reagir dinamicamente a mudanças de configuração e ao número de nodos que compõe a rede. Esses objetivos, em geral, são conflitantes, o que torna um desafio atendê-los, ainda mais considerando o dinamismo da RSSF.

O trabalho será avaliado por meio de experimentos a serem realizados com os nodos reais, dividido em duas partes: a primeira visa garantir que o algoritmo executado nos nodos esteja funcionando corretamente. A segunda partes dos testes consiste em avaliar o comportamento da abordagem proposta.

1.4 ORGANIZAÇÃO DO TEXTO

O texto está organizado em sete capítulos. O Capítulo 2 fornece uma base de conhecimento a respeito de redes de sensores sem fio e a norma IEEE 802.15.4, apresentando as topologias de rede e descritas, de forma sucinta, partes mais importantes do padrão. O Capítulo 3 trata sobre Fusão de Dados, apresentando uma visão geral de fusão de dados e suas formas de classificação. No Capítulo 4 é descrito o projeto do hardware desenvolvido no decorrer desta dissertação, descrevendo uma visão geral dos softwares gerados para validar o hardware e do software básico. O Capítulo 5 apresenta a solução de software proposta e os algoritmos dos nodos sensores e do centro de fusão. Os resultados dos ensaios são apresentados e discutidos no Capítulo 6. Finalmente, no Capítulo 7 são apresentadas as conclusões finais do trabalho, comentários a respeito dos desafios encontrados e perspectivas futuras.

2 PADRÃO IEEE 802.15.4

2.1 INTRODUÇÃO

Uma rede de sensores sem fio (RSSF), em inglês *wireless sensor network (WSN)*, consiste diversos dispositivos distribuídos (também conhecido como nodos), com capacidade de monitorar grandezas do ambiente e encaminhar suas leituras para nodos especiais. Os dispositivos de uma RSSF são formados por microcontroladores simples, dotados de comunicação sem fio e sensores.

Em muitas RSSF – por exemplo, aquelas baseadas no padrão IEEE 802.15.4 – existe um tipo especial de nodo que pode desempenhar o papel de gerenciador de rede. Em algumas redes podem existir mais de um nodo que gerencia a rede. No entanto, usualmente há apenas um coordenador que recebe o nome de coordenador PAN (rede pessoal sem fio – *personal area network*). Esse nodo também pode ter a função de estação base podendo, em algumas configurações, conectar a PAN a outra rede, como a Internet. Os demais nodos têm a função de obter dados sobre o ambiente monitorado, tais como: temperatura, som, vibração, pressão, umidade, movimento de fluídos etc; e encaminhá-los na forma de mensagens para o coordenador PAN.

As características desejadas em um nodo de RSSF são: baixo consumo de energia, robustez e baixo custo. Essas características são necessárias pelo fato de que uma boa parte dos nodos geralmente é alimentada por baterias ou fontes especiais como energia solar. A robustez é fundamental, já que muitos dispositivos são utilizados para monitorar ambientes hostis ou de difícil acesso ao ser humano. O custo baixo é um fator determinante para se obter redes com milhares de dispositivos como, por exemplo, uma rede que monitora as marés no litoral, queimadas no campo, deslizamentos em encostas de morros etc.

O uso mais comum das RSSF é no monitoramento. Tubulações de gás e óleo, índices de poluição, queimadas no campo, temperaturas e erupções em vulcões são alguns dos exemplos de aplicações destas redes. Na indústria, a importância vem crescendo com o passar dos anos já que a manutenção e instalação são mais baratas que redes cabeadas e menos complexas.

O baixo custo destes dispositivos vem contribuindo para a adoção e uso dessa tecnologia. A área de aplicação que mais cresce é a da automação residencial e predial. Um dado fornecido pelo consórcio Zigbee Alliance mostrou que em 2010 os dispositivos com tecnologia 802.15.4/Zigbee dobrou e nos últimos 5 anos cresceu 800% (ZIGBEEORG, 2011). De acordo com a

publicação (CYNAPSIS, 2012; WORLD'S, 2011), o número de dispositivos comercializados sinaliza na direção que a norma IEEE 802.15.4 pode estar se tornando um padrão *de facto* para as RSSF.

Neste capítulo descreve-se o padrão IEEE 802.15.4, o qual é usado como infraestrutura de comunicação dessas redes.

2.2 IEEE 802.15.4

A norma IEEE 802.15.4 foi desenvolvida com a intenção de prover a padronização para as camadas física e enlace em nodos de redes de baixa velocidade que necessitam de baixo consumo energético (*low-rate wireless personal area networks* LR–WPANs). O principal objetivo de uma LR-WPAN é oferecer uma infraestrutura de comunicação de baixo custo para ser utilizada em aplicações de curto alcance e que necessitem de baixa taxa de dados.

A publicação original do IEEE 802.15.4 ocorreu em 2003 (IEEE-802.15.4, 2011) e, posteriormente, foi publicada uma nova atualização (IEEE-802.15.4, 2011) que adicionou duas novas camadas físicas para trabalhar em outras faixas de frequência, visando principalmente o mercado da Ásia. O MAC foi alterado, mas mantendo compatibilidade com a versão original. As modificações no quadro MAC foram feitas para indicar nova versão, novas formas de segurança e avanços como: suporte para uma base de tempo compartilhado e mecanismo de marcação de tempo, suporte a escalonamento de *beacon* e sincronização de mensagens *broadcast* em WPAN do tipo *beacon* ativado. Recentemente, em 2011, foi publicada uma nova revisão da norma (IEEE-802.15.4, 2011), a qual adiciona mais duas especificações físicas.

A camada física PHY (*Physical Layer*), em última análise, fornece o serviço de transmissão de dados, bem como a interface ao *physical layer management entity (MLE)*, o qual oferece acesso a cada função da camada e a tabela de informação relacionada a WPAN. Com isso, o PHY manipula a camada física do rádio realizando a seleção de canais e a verificação de energia e sinal. A norma IEEE (IEEE-802.15.4, 2011) especifica as seguintes faixas de frequências:

- 314-316 MHz e 430-434 MHz: China, CWPAN[1] (China WPAN) está fora do escopo da IEEE 802.15.4 (2011).
- 779-787 MHz: China, até 8 canais e usa a página 5 (desde 2006).
- 868.0-868.6 MHz: Europa, um canal na página 0 (2003). Estendido para as páginas 1 e 2 (2006). Total de 3 canais.

[1] CWPAN, publicações estão disponíveis em `http://www.cssn.net.cn`

- 902-928 MHz: América do Norte, até 10 canais, página 0 (2003). Estendido às páginas 1 e 2 (2006). Total de 30 canais.
- 950-956 MHz: Japão, até 22 canais na página 6 (desde 2011).
- 2400-2483.5 MHz: uso mundial, até 16 canais, página 0 (desde 2003).

O padrão define dois tipos de nodos: FFD (*full–function device*) – dispositivo com todas as funcionalidades e RFD (*reduced–function device*) – dispositivo com funcionalidades reduzidas. Um nodo RFD tipicamente é alimentado por baterias e opera com uma implementação mínima ou reduzida do protocolo MAC sendo voltado para funções simples. Um dispositivo FFD, por outro lado, pode atuar como coordenador PAN, o qual autentica os demais nodos. Esse tipo de nodo, operando como coordenador PAN, pode ainda transmitir periodicamente mensagens de *beacon* que atuam como um serviço de sincronização entre os dispositivos. A infraestrutura de rede IEEE 802.15.4 necessita de pelo menos um coordenador para gerenciar os demais nodos.

O padrão também define dois modos de acesso ao meio: modos com *beacon* ativado e sem *beacon*. Quando operando em modo *beacon* ativado, *beacons* são gerados pelo coordenador no primeiro *slot* de cada *superframe* obrigatoriamente. A Figura 1 ilustra a definição de *superframe*.

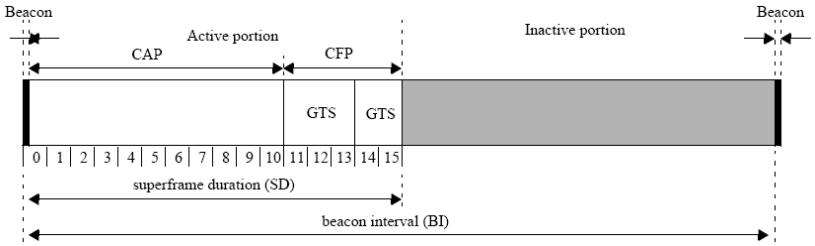


Figura 1: Estrutura de Superframe: Fonte (IEEE-802.15.4, 2011).

Cada *superframe* consiste de um período ativo e outro inativo. Na Figura 1 é mostrada a influência de dois parâmetros importantes: *Beacon Order* (BO) e *Superframe Order* (SO). O intervalo de tempo entre dois *beacons* consecutivos é chamado *Beacon Interval* (BI), e é definido por meio do parâmetro BO (na faixa de 2.4GHz o BI é igual a 15.36×2^{BO}ms, onde $0 \leq BO \leq 14$). O período ativo é denominado *Superframe Duration* (SD) e seu tamanho é definido pelo parâmetro SO ($SD = 15.36 \times 2^{SO}$ ms na faixa de 2.4 GHz, com $0 \leq SO \leq BO \leq 14$).

A Tabela 1 destaca os possíveis valores para o tamanho dos slots de acordo com o padrão IEEE 802.15.4. Observa-se que o período das aplicações pode variar de 15.36ms até 4.2 minutos, com o tamanho do slot variando de 960us até 15.7 segundos.

BO=SO	Period (ms)	Time Slot (ms)	Slot size (bytes)
0	15,4	0,96	30
1	30,7	1,92	60
2	61,4	3,84	120
3	122,9	7,68	240
4	245,8	15,36	480
5	491,5	30,72	960
6	983,1	61,44	1.920
7	1.966,1	122,88	3.840
8	3.932,2	245,76	7.680
9	7.864,3	491,52	15.360
10	15.728,6	983,04	30.720
11	31.457,3	1.966,08	61.440
12	62.914,6	3.932,16	122.880
13	125.829,1	7.864,32	245.760
14	251.658,2	15.728,64	491.520

Tabela 1: Períodos de superframe na IEEE 802.15.4 (assumindo que não existe período inativo)

Também pode ser observado que o SD é dividido em dois períodos: *Contention Access Period* (CAP) e *Collision Free Period* (CFP). O período CAP representa a faixa de *slots* com contenção de acesso e o algoritmo CSMA/CA deve ser utilizado para se obter acesso ao meio. O período CFP apresenta um número de *slots* com acesso garantido (*Guaranteed Time Slots* – GTS) via divisão de tempo (*Time-Division Multiple Access* – TDMA). Contudo, a maior desvantagem do CFP é que apenas sete GTS podem ser alocados. A abordagem de acesso garantido via GTS é, portanto, insuficiente para as aplicações que utilizam muitos nodos.

As topologias de rede suportadas pelo padrão IEEE (IEEE-802.15.4, 2011) são duas: estrela ou par-a-par. Devido à simplicidade, a topologia estrela vem sendo proposta em diversas aplicações, principalmente em monitoramento industrial (WILLIG, 2008). Na topologia estrela, o nodo central figura como coordenador PAN, e em muitas situações é o único nodo ligado a uma fonte de energia constante (IEEE-802.15.4, 2011), geralmente tendo maior capacidade de processamento que os outros nodos. Além da simplici-

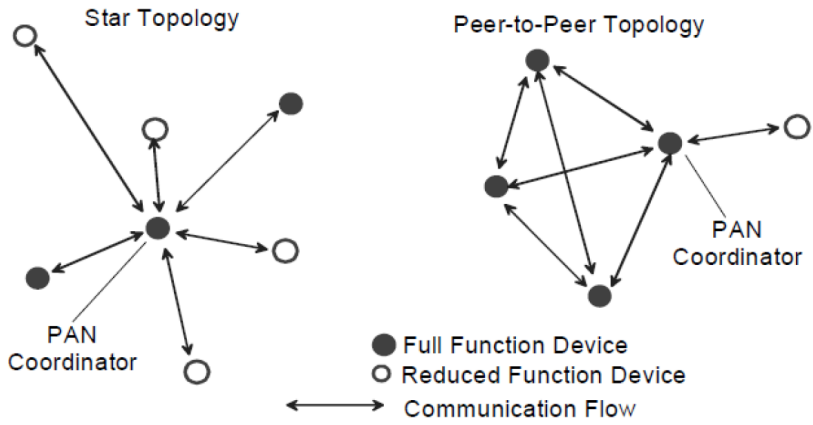


Figura 2: Topologias de rede: Fonte (IEEE-802.15.4, 2011).

dade, justifica-se a topologia estrela também pelo fato dos atuais transceivers de rádio-frequência (RF) poderem facilmente alcançar mais de 200m em condições de visada, o que permite a cobertura de grandes áreas.

Por outro lado, em uma rede par-a-par um dispositivo se comunica com outro diretamente, se estiver no alcance, ou indiretamente por meio de outros nodos. Cada nodo da rede precisa fazer as tarefas de comutação e roteamento de mensagens ao longo da rede. Essa topologia é muito utilizada atualmente para se criar redes de larga escala, formadas por centenas a milhares de nodos. Ressalta-se como vantagem o fato de permitir a implementação de rotas alternativas, oferecendo, geralmente, maior robustez se comparada à topologia estrela. Na Figura 2 é apresentado um esboço das topologias cobertas pela norma. Na Figura 3 é apresentado um exemplo de uma rede par-a-par montando uma estrutura complexa, conhecida como árvore com grupos (*cluster-tree*). Outras estruturas conhecidas como malha (*mesh*) também podem ser criadas usando topologia par-a-par.

Na Figura 4 é mostrado um diagrama com os modos operacionais do padrão IEEE 802.15.4. A sincronização da rede pode ser realizada de duas formas, com ou sem envio de *beacon* periódico. No caso mais simples, sem *beacon* periódico, o coordenador envia *beacons* apenas se solicitado para autenticação de rede. O envio de dados pelos nodos é realizado a qualquer momento utilizando CSMA-CA ou ALOHA. No caso de *beacon* periódico os nodos têm duas opções para realizar o envio de dados. No primeiro modo, *tracking beacon*, o nodo sempre sincroniza com o *beacon* e envia os dados

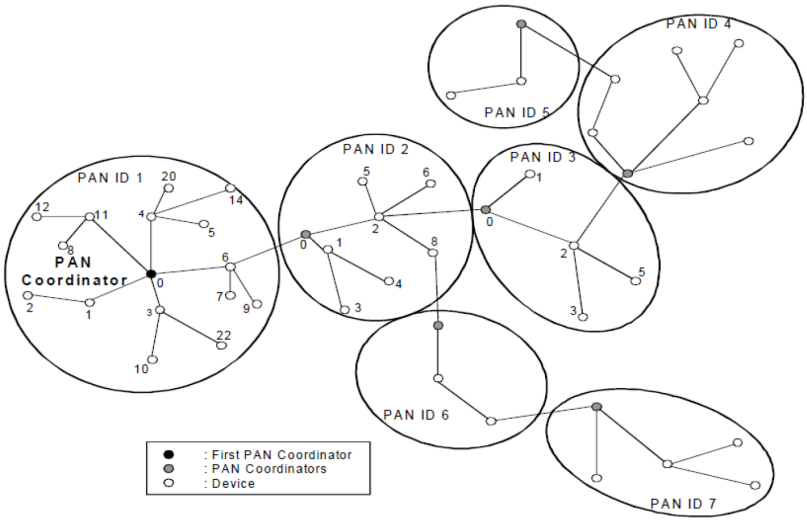


Figura 3: Rede Cluster Tree: Fonte (IEEE-802.15.4, 2011).

no momento oportuno respeitando o CAP e o CFP. No segundo modo, sem *tracking beacon*, o nodo pode "dormir" por longos períodos de tempo mas quando desejar transmitir uma informação, deve obrigatóriamente sincronizar com o primeiro *beacon* e enviar no momento oportuno. Isso é necessário para manter a integridade dos *slots* de GTS.

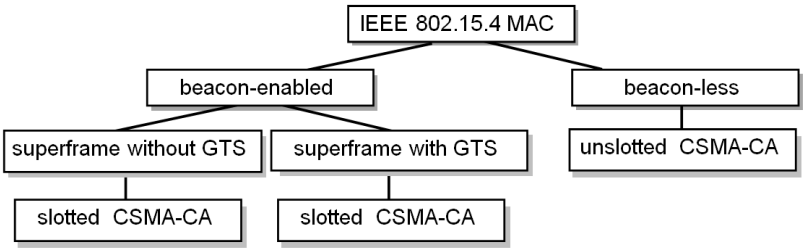


Figura 4: Modos operacionais do IEEE 802.15.4.

O padrão IEEE 802.15.4 de 2006 adiciona o escalonamento de *beacons*. Redes que utilizam essa abordagem geralmente são do tipo *cluster-tree*

ou *mesh*. Na Figura 5 é apresentado um diagrama de como ocorre esse processo. Quando existe mais de um dispositivo que gera *beacons* na rede, o coordenador inicia o envio do *beacon* e todos os dispositivos ao alcance recebem. Isso sinaliza o início de um intervalo entre *beacons* – BI. Outros dispositivos que enviam *beacon* devem aguardar o término do *SD*, conforme configurados, e quando o tempo indicado por $StartTime > SD$ for atingido, o dispositivo pode transmitir o seu *beacon*.

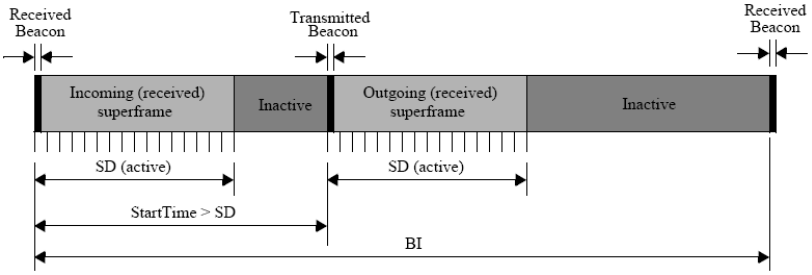


Figura 5: Relação entre chegada e saída de *beacon*: Fonte (IEEE-802.15.4, 2011).

O padrão IEEE 802.15.4 tem parâmetros para tratar da economia de energia, dependendo do tipo da aplicação e modos de uso. Por exemplo, o parâmetro de extensão da vida da bateria *Battery Life Extension* (BLE). Quando um nodo ativa o modo BLE, os valores do período exponencial de *backoff* do algoritimo CSMA/CA ficam limitados ao intervalo $[0, 2]$. Essa característica economiza energia mas pode aumentar o número de erros de transmissão devido a colisões quando existem muitos nodos na rede. É importante ressaltar que o coordenador também pode entrar em modo de economia de energia durante o período inativo.

A padrão IEEE 802.15.4-2011 oferece segurança por meio de criptografia, usando *Advanced Encryption Standard (AES)* com chave de 128 bits de comprimento (16 bytes). A maioria dos transceptores do mercado já implementa AES-128b no próprio hardware para possibilitar a utilização de segurança sem necessitar de processadores mais rápidos que poderiam aumentar o consumo de energia do nodo.

O algoritmo AES não é utilizado apenas para criptografar informações mas também para validar os dados a serem enviados. Este conceito é chamado de integridade de dados e é alcançado utilizando *Message Integrity Code (MIC)*, também conhecido como *Message Authentication Code (MAC)*, o qual é adicionado na mensagem. Ele é gerado criptografando partes do

quadro usando a “chave” da rede. Com isso, no caso de ser recebida uma mensagem de um nodo não confiável, a mensagem será descartada no caso do MIC gerado pelo emissor não corresponder a uma mensagem gerada utilizando a “chave” secreta. O MIC pode ter tamanhos diferentes: 32, 64 e 128 bits, contudo é sempre gerado usando o algoritmo AES-128b. Esse tamanho é o comprimento de bits adicionado a cada quadro. Quando maior o número de bits adicionado ao quadro, maior será a segurança e menos dados poderão ser trafegados na carga útil (*payload*) da mensagem. De acordo com as opções selecionadas para segurança é possível garantir tanto a confidencialidade quanto a autenticidade dos dados.

Para ser possível utilizar o recurso de segurança, cada nodo IEEE 802.15.4 deve ter capacidade de manipular uma lista de controle de “nodos confiáveis” junto com a regra de segurança adotada. Essa lista, denominada Lista de Controle de Acessso (*Access Control List – ACL*), armazena as seguintes informações:

- Endereço do nodo que se deseja comunicar.
- Tipo de Segurança, que é a regra de segurança a ser utilizada.
- Chave, que é a palavra de 128 bits para o algoritmo AES.
- Contador de envios, o qual é utilizado para evitar ataques de retransmissão.

Assim, quando um nodo deseja enviar uma mensagem para um nodo específico ou quando recebe um pacote, ele observa na ACL se o nodo que enviou a informação é confiável ou não. Caso seja, o nodo utiliza as informações contidas na ACL para aplicar a segurança. Se o nodo que enviou não estiver na lista, a mensagem é rejeitada ou inicia-se um processo de autenticação.

2.3 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram apresentados os principais conceitos relacionados com a norma IEEE 802.15.4, a qual foi elaborada com objetivo de padronizar as redes de baixo consumo energético. A compreensão deste padrão e seu modo de operação com *beacon*, em topologia estrela, é de grande importância para o entendimento da proposta deste trabalho.

3 FUSÃO DE DADOS

3.1 INTRODUÇÃO

A Fusão de Dados é um processo matemático que unifica informações de diversas fontes, homogêneas ou heterogêneas, para se obter um valor mais apurado. Um exemplo clássico pode ser encontrado em aeroportos com a combinação de dados de navegação GPS juntamente com as informações de radar (distância, altitude, direção e velocidade). Essas informações, se combinadas por meio de técnicas de fusão de dados, fornecem uma posição mais precisa de uma aeronave, quando comparada com a posição fornecida por apenas uma referência.

Técnicas de fusão de dados são aplicadas nas áreas civil e militar (FAOUZI; LEUNG; KURIAN, 2011). Na área civil, a fusão de dados é muito utilizada em máquinas de tomografia computadorizada (STEINER, 2006), onde são utilizadas uma imagem de ultrasom e a capacitância elétrica para gerar uma imagem de tomografia mais detalhada. Na área militar, geralmente a fusão de dados é utilizada para monitoramemto e vigilância, como mapeamento de margens de rios (GOLçALVES et al., 2009).

A literatura mostra diversas técnicas para se realizar uma fusão de dados, tanto para sensores homogêneos como para heterogêneos. Como a fusão de dados é utilizada em diversos cenários, de uma forma geral existe uma técnica mais utilizada para cada situação. Por exemplo, nas aplicações de sensoreamento remoto, segundo (AMARSAIKHANA; DOUGLAS, 2004), as técnicas mais utilizadas são estatísticas, como a teoria da evidência de Dempster-Shafer e redes neurais. Segundo os mesmos autores, IHS é a técnica mais utilizada para fusão de dados, onde H e S são componentes com informação espectral e I representa o componente da informação espacial.

Muitas abordagens para fusão de dados utilizam técnicas de Inteligência Artificial, reconhecimento de padrões ou estimação estatística. Como exemplos dessas técnicas, têm-se:

- **Técnicas estatísticas:** combinação de pesos, análise estatística multivariável e mi-neração de dados.
- **Técnicas probabilísticas:** modelos bayesianos como redes Bayesianas e Estado-Espaço, verossimilhança e filtros de Kalman.
- **Inteligência Artificial:** redes neurais, cognição artificial e algorimos genéticos. No geral, servem como ferramentas para derivar classificadores e estimadores.

3.2 MODELO DE FUSÃO DE DADOS

As aplicações das técnicas de Fusão de Dados para sistemas complexos não são novas. O modelo JDL (*Joint Directors of Laboratories*)[1] é da década de 1980 e foi revisado em 1999 (BLASCH; PLANO, 2002). O JDL é o modelo mais seguido para fusão de dados e em 2004 teve uma segunda revisão (LLINAS et al., 2004). Apesar disso, com o aumento das massas de dados armazenadas, imagens de satélites por exemplo, começa a surgir uma busca por técnicas para aumentar a precisão dos valores ou a qualidade dos dados apresentados. Outros exemplos que podem se beneficiar das técnicas de fusão de dados são os sistemas para controle de tráfego inteligente. Usando dados dos sensores instalados nos veículos e dados armazenados em *data centers* pode-se tentar prever o comportamento do fluxo do trânsito em tempo real e estimar a probabilidade de um congestionamento, baseado nos históricos armazenados.

A técnica de fusão de dados pode envolver vários níveis de processamento. O citado modelo JDL, por exemplo, modela os processos existentes em uma fusão de dados e é composto de cinco fases que podem ser visualizadas na Figura 6. Esse modelo fornece uma visão sistêmica da rede que executa fusão guiando o projetista pela identificação das principais soluções a serem incorporadas na rede, por exemplo, consultando uma base de dados.

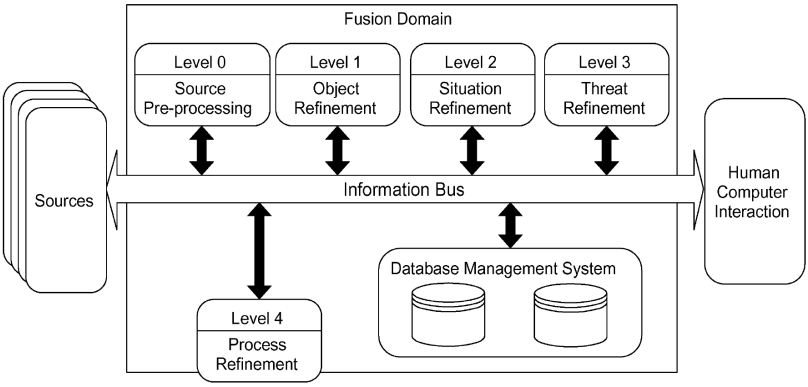


Figura 6: Modelo JDL. Fonte: (HALL; LLINAS, 1997).

1. **Nível 0:** Trata de pré-processar os dados, como: normalizar, formatar,

[1] JDL formava o *Data Fusion Subpanel*, o qual depois se tornou *Data Fusion Group*.

ordenar, comprimir dados etc. Em resumo, busca identificar os subobjetos ou requisitos nos dados para serem utilizados no próximo nível.

2. **Nível 1:** Escolhe os dados das fontes de informação apropriadas, tais como as redes de sensores sem fio, radares, satélites, ultrasom etc.

3. **Nível 2:** Identifica o provável estado atual do sistema, combinando os dados observados e eventos resultantes do nível anterior com outras fontes e bancos de dados. Estas fontes podem incluir informações de trânsito, informações de clima e tempo etc.

4. **Nível 3:** Avalia os padrões e dados relativos às ocorrências do evento ou fenômeno observado.

5. **Nível 4:** Busca aprimorar todo o processo de fusão de dados pelo contínuo refinamento das predições. Também avalia se são necessárias mais informações provenientes de outras fontes.

O modelo JDL foi proposto em âmbito de pesquisa militar, por isso sua terminologia e aplicação são direcionadas nesse sentido. Um outro inconveniente desse modelo é que ele não deixa explícita a interação entre os elementos processados. Na revisão de 2004, o nível 4 foi removido (LLINAS et al., 2004; ELMENREICH, 2002; NAKAMURA; LOUREIRO; FRERY, 2007).

3.3 CLASSIFICAÇÃO DA FUSÃO DE DADOS

Segundo (NAKAMURA; LOUREIRO; FRERY, 2007), a fusão de dados pode ser classificada de três formas: baseada na relação entre as fontes de informação, nos níveis de abstração ou na entrada e saída.

3.3.1 Classificação baseada na relação entre as fontes de informação

A classificação de uma fusão de dados baseada nas fontes de informação pode ser subdividida em três classes: redundante, complementar ou cooperativa.

Se duas ou mais partes independentes fornecem a mesma informação, estas partes podem ser fundidas para se obter um aumento na confiabilidade. Isso caracteriza uma fusão **redundante** (*competitive*). As fontes $S1$ e $S2$ da Figura 7 fornecem a mesma informação, a, na qual é realizada uma fusão obtendo-se um valor mais preciso, (A). Em RSSF, a fusão redundante pode

ser utilizada para aumentar a confiança e precisão da informação, ou para economizar energia, evitando que os nodos sensores transmitam informação desnecessária.

Quando as informações fornecidas pelas fontes representam diferentes partes de um contexto mais amplo, a fusão é classificada como **complementar** (*complementary*). Na Figura 7 as fontes $S2$ e $S3$ fornecem diferentes partes da informação, a e b, que são fundidas em uma nova peça de informação, mais ampla, denominada $(A+B)$ composta por partes não redundantes do ambiente. Pode-se exemplificar essa situação como sendo a informação de temperatura de uma região monitorada, composta pela temperatura do lado esquerdo somada à temperatura do lado direito.

Uma fusão é **cooperativa** (*cooperative*) quando duas ou mais fontes independentes são fundidas em uma nova informação, na maioria das vezes mais complexa. Essa informação, na perspectiva da aplicação, representa a realidade. As fontes $S4$ e $S5$ na Figura 7 fornecem diferentes informações, c e c', as quais são fundidas na informação (C) que descreve melhor o cenário quando comparado a c e c' individualmente.

O exemplo clássico da fusão cooperativa é a computação de um local alvo com base na informação do ângulo e distância. A fusão cooperativa deve ser aplicada com um certo cuidado, uma vez que os dados resultantes são sujeitos a imprecisão e imperfeição de todos os participantes da rede.

3.3.2 Classificação baseada nos níveis de abstração

A fusão de dados lida com três níveis de abstração de dados: medição, recurso e decisão. Então, de acordo com o nível de abstração dos dados manipulados a fusão de dados pode ser classificada em quatro categorias (NAKAMURA; LOUREIRO; FRERY, 2007):

- **Fusão de Baixo Nível (fusão de medição):** dados brutos são fornecidos como entrada e combinados em uma nova peça de dados que é mais precisa do que as entradas individuais.

- **Fusão de Médio Nível (fusão de recursos):** é uma fusão que utiliza atributos ou recursos (exemplo: formas, texturas, posição) para se obter um mapa de características que pode ser utilizado por outras tarefas (exemplo: segmentação ou detecção de objetos).

- **Fusão de Alto Nível (fusão de símbolos ou de decisão):** toma decisões ou símbolos como entrada e os combina para se obter uma decisão mais confiável ou global.

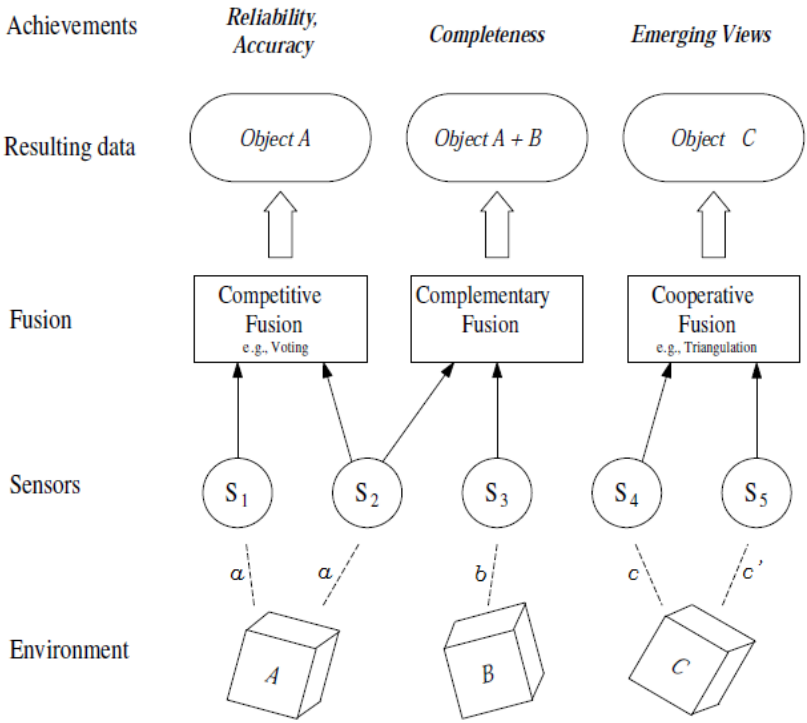


Figura 7: Fusão competitiva, complementar e cooperativa. Fonte: (ELMENREICH, 2002).

- **Fusão Multi-Nível:** quando uma fusão engloba dados (tanto na entrada quanto na saída da fusão) de diferentes níveis de abstração.

3.3.3 Classificação baseada na entrada e saída

Levando-se em consideração as entradas e saídas da informação, pode-se classificar a fusão de dados em cinco categorias (NAKAMURA; LOUREIRO; FRERY, 2007):

- **Dados entram – Dados saem (DAI-DAO):** nesta classe, a fusão de dados trata com dados brutos e o resultado também é na forma de dados brutos, possivelmente mais precisos.

- **Dados entram – Características saem (DAI-FEO):** a fusão de dados utiliza dados brutos como fonte para extrair características ou atributos que descrevem uma entidade. Uma entidade significa qualquer objeto, situação ou abstração.

- **Características entram – Características saem (FEI-FEO):** essa fusão trabalha com conjuntos de características para aprimorar, refinar ou extrair novas características.

- **Características entram – Decisões saem (FEI-DEO):** nesta classe, a fusão de dados pega um conjunto de características de uma entidade e gera uma representação simbólica ou uma decisão.

- **Decisões entram – Decisões saem (DEI-DEO):** Decisões podem ser fundidas para se obter novas decisões ou dar mais ênfase em alguma decisão anteriormente tomada ou gerada.

3.4 TIPOS DE SOLUÇÕES

Segundo (PINTO, 2010) existem três tipos de soluções para fusão de dados em redes sem fio: paralela, serial e híbrida. Nas soluções paralelas, os dados dos sensores são encaminhados para o centro de fusão sem que haja combinação dos dados ao longo do caminho da comunicação. A solução serial combina a informação de outro(s) sensor(es) com a sua informação para, somente então, executar o reencaminhamento de dados. Um solução híbrida combina o uso das duas soluções, podendo ser usada em redes hierárquicas, tais como redes com topologia *cluster-tree*. Na Figura 8 são exemplificadas cada uma das soluções.

3.5 TRABALHOS RELACIONADOS A FUSÃO DE DADOS

Os trabalhos na literatura sobre fusão de dados envolvem níveis de abstração diferentes, compreendendo desde modelos matemáticos e técnicas para processamento de sinais até trabalhos sobre aplicações de fusão de dados, onde se investiga como abordagens de fusão de dados podem ser usadas para o benefício da sociedade. Considerando a grande abragência desse assunto, os trabalhos relacionados abordados nesta seção são principalmente aqueles que influenciaram o modelo desenvolvido neste trabalho.

Em (CHIUSO; SCHENATO, 2011) são discutidos estimadores para estratégias de fusão de dados em RSSF sujeitos a ruído e perda de pacotes.

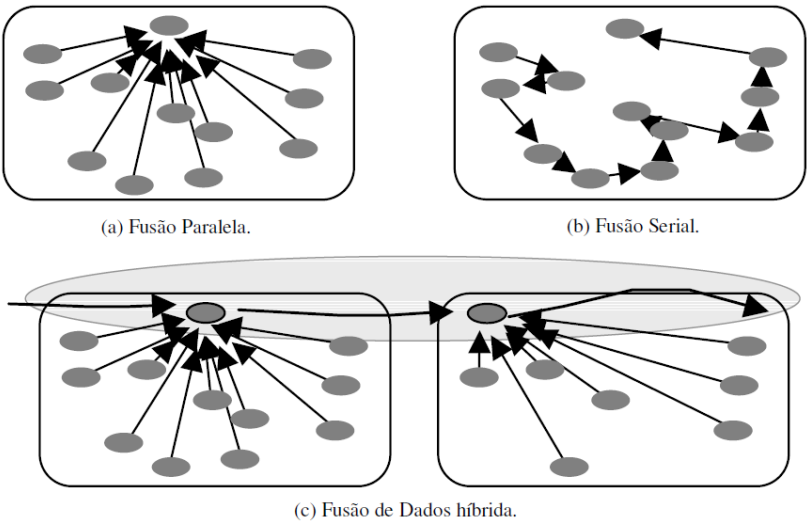


Figura 8: Tipos de soluções para fusão de dados. Fonte: (PINTO, 2010).

Os estimadores são baseados na ideia que os nodos sensores enviam suas estimativas locais para um centro de fusão, o qual estima o valor de uma variável monitorada de acordo com as medidas que foram recebidas. No entanto, assume-se que o meio de comunicação é sujeito a erros, atrasos e perdas de mensagens. Por conseguinte, o interesse maior do trabalho é a proposição de estimadores que funcionem apesar de descartes e perdas de mensagens enviadas pelos sensores. A Figura 9 ilustra um cenário de recebimento de amostras dos sensores do ponto de vista do centro de fusão. Cada *i–ésima* linha e *t–ésima* coluna de informação da matriz representa o valor recebido pelo centro de fusão de um nodo i no instante de tempo t.

Dois classificadores sub-ótimos são apresentados em (D'COSTA; SAYEED, 2003). O primeiro é um classificador baseado na média dos dados, o qual é utilizado quando os fenômenos são correlacionados. O outro classificador trata todas as leituras como sendo independentes, caracterizando uma fusão de decisão.

Em (PATIL; DAS, 2004) é explorado o problema de fusão de dados serial, na qual os dados sofrem um processo de fusão de dados local em cada nodo. Os dados produzidos a cada fusão atravessam a RSSF até alcançar o nodo responsável pela tomada de decisão relacionado à informação monitorada. Com objetivo de economizar energia, a fusão não é obrigatóriamente

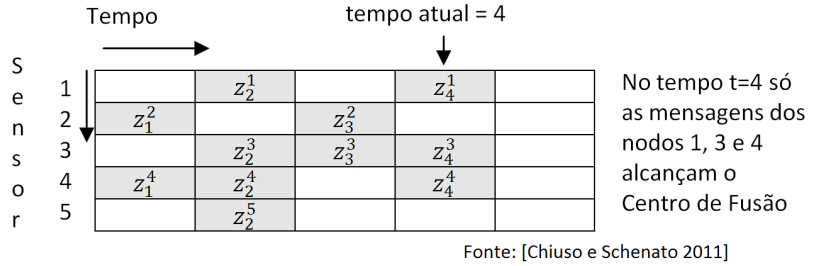


Figura 9: O valor da i–ésima linha e t–ésima coluna é a informação do nodo *i* recebida na Central de Fusão no tempo *t*.

feita em cada nodo. A abordagem adotada considera que a fusão deixa de ocorrer quando, ao alcançar um nodo qualquer, o valor de razão de verossimilhança (*Log-Likelihood Ratio*) é alcançado.

Um esquema de fusão de dados baseado em árvore é considerado em (YUAN; KRISHNAMURTHY; TRIPATHI, 2003). O objetivo principal é sincronizar multiplos níveis de fusão de dados. Assim, cada nodo decide a respeito de quando iniciar o processo de fusão e quanto tempo vai aguardar para determinar o fim do processo de fusão.

Em (HE et al., 2006), é proposta uma arquitetura hierárquica para agregação de dados que pode ser utilizada em aplicações de rastreamento usando RSSF. Para conseguir um bom equilíbrio entre eficiência energética, latência e disponibilidade, a arquitetura foi implementada em quatro camadas: sensor, nodo, grupo e nível de base. A comunicação entre os nodos é realizada primeiro no nível de grupo, onde o líder recebe o conteúdo de seus vizinhos; posteriormente cada líder envia os resultados locais para o coordenador. Foi proposto um parâmetro configurável chamado DOA (*Degree of Aggregation*), o qual representa o número de vezes que o conteúdo deve ser recebido antes de realizar a operação de agregação local e enviar os resultados para a estação base. Os experimentos mostraram que um valor da DOA muito pequeno leva a freqüentes falsos positivos. Por outro lado, um alto valor para DOA não é desejado, uma vez que ele introduz longos atrasos e alto consumo de energia.

Uma abordagem baseada em votação é apresentada por (AYERS; LIANG, 2011), que adota uma topologia em estrela onde o coordenador deve receber um certo número de mensagens, definido como *voto sim*, antes de tomar uma decisão. Uma função probabilista de recompensa é assumida no nodo central. Esta função, geralmente em forma de sino, recebe seu máximo

valor quando o número de votos sim é exatamente o valor desejado. O prêmio atual P é propagado em cada nodo, e eles tomam decisões a respeito de manter o estado atual (transmitir, não transmitir, ou modo baixa energia) com base em uma máquina de estados estocástica. Um dos principais inconvenientes desta abordagem é que a definição da função de recompensa não é uma tarefa trivial. No entanto, traz como vantagem uma abordagem autonômica e distribuída, onde cada nodo decide de forma independente se irá transmitir, ou não, sua mensagem.

Em (CHEN et al., 2011), foi proposto um protocolo de agregação de dados simples chamado *Lightweight Data Aggregation Protocol* (LDAP), o qual visa reduzir o consumo de energia dos aparelhos que ficam monitorando condições ambientais em sítios arquiológicos ou áreas de preservação histórica permanente. O LDAP atua com base nas mudanças que ocorrem no ambiente monitorado realizando fusão de dados. A fusão de dados é alimentada com os dados brutos coletados localmente e, se as informações forem relevantes, são transmitidas. Este processo diminui consideravelmente os dados transmitidos na RSSF, aumentando o tempo de vida da rede.

Em (PINTO; MONTEZ, 2010) é apresentado um modelo de comunicação com abordagem que incorpora propriedades autonômicas (auto-organização) na rede. Cada nodo possui autonomia para decidir se irá transmitir ou não sua mensagem, buscando a redução do consumo de energia nos nodos (evitando-se colisões de mensagens na rede) mas, simultaneamente fazendo com que um número mínimo de mensagens alcance a estação base. A abordagem adotada busca otimizar duas métricas (*Qualidade da Fusão* - QoF e *Eficiência* - Ef) utilizando um classificador baseado em algoritmo genético.

3.5.1 Comparação com trabalhos relacionados

O modelo e as métricas utilizadas em (PINTO; MONTEZ, 2010) são similares ao utilizado nesta dissertação. A maior diferença entre os trabalhos é que o proposto por (PINTO; MONTEZ, 2010) usa algoritmo genético buscando otimizar simultaneamente QoF e Ef; enquanto este trabalho desconsidera a métrica Ef mas leva em consideração a energia de cada nodo. A abordagem utilizada neste trabalho também possui um custo computacional bem menor do que o proposto por (PINTO; MONTEZ, 2010), e o comportamento do algoritmo também é mais distribuído, pois cada nodo toma decisões baseadas com a energia de suas baterias.

Com base no primeiro classificador apresentado em (D'COSTA; SAYEED, 2003), este trabalho de dissertação faz uso da fusão de

dados e submete o valor de saída a uma análise estatística para determinar se esse valor é adequado para o sistema ou são necessárias mais amostras. Esse processo acarreta na obtenção do valor da *QoF* do sistema e serve como guia para os nodos no processo de auto-ajuste de economia de energia.

Apesar do trabalho em (PATIL; DAS, 2004) ter como foco a fusão de dados serial, ao contrário desta proposta que aborda a fusão paralela, os objetivos de ambos são fortemente relacionados. Ambos abordam o problema de usar os dados lidos pelos sensores de forma colaborativa, com o objetivo de minimizar o consumo de energia na rede, implementando um esquema de detecção que busca envolver somente o número mínimo de sensores necessários para alcançar um critério específico desejado. Patil considera que a fusão deixa de ocorrer quando, ao alcançar um nodo qualquer, o valor de *Log-Likelihood Ratio* é alcançado. Neste trabalho, os nodos descidem quando vão enviar as mensagens para a RSSF com base no valor da Qualidade da Fusão (*QoF*).

No trabalho de (HE et al., 2006), quando se considera a comunicação no nível de grupo, o equilíbrio alcançado no artigo é semelhante ao objetivo principal deste trabalho e o parâmetro DOA assemelha-se à métrica de *QoF* adotada neste trabalho.

3.6 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo apresentou-se a visão geral de fusão de dados, as principais classificações e definições. Durante as seções, foi-se aprofundando no assunto com base nas classificações do modelo JDL, modelo mais tradicional, e reforçando que a fusão de dados não é uma técnica nova. Alguns exemplos foram apresentados para o melhor entendimento e para dar visibilidade sobre o conceito de fusão de dados para posteriormente localizar a contribuição da técnica deste trabalho de mestrado.

4 PROJETO DE HARDWARE E SOFTWARE BÁSICO

4.1 INTRODUÇÃO

Neste trabalho de mestrado são propostas abordagens de comunicação adequadas para implementar técnicas de fusão de dados em RSSF em topologia estrela compatíveis com a norma IEEE 802.15.4. O trabalho envolveu não somente a proposição das abordagens de software pois, no seu decorrer, houve o desenvolvimento de nodos sensores compatíveis com a norma IEEE 802.15.4 da RSSF, os quais foram usados nos experimentos. Neste capítulo são apresentadas informações do projeto de hardware e do software básico para os nodos. O projeto de hardware foi precedido por um levantamento de informações para construção de dois kits: um para trabalhos de campo e outro para pesquisas em laboratório. O software básico descrito neste trabalho e relacionado aos nodos se refere a todo aparato necessário para o devido funcionamento dos dispositivos, bem como as ferramentas que foram usadas como auxílio aos experimentos com os nodos.

4.2 PROJETO DO HARDWARE

O projeto de hardware foi elaborado a partir de três necessidades: uso do IEEE 802.15.4 no modo com *beacon*, alcance de rádio com longa distância e capacidade de *data logging* (gravação da massa de dados coletados em campo). Essas necessidades surgiram em decorrência da aplicação alvo inicialmente concebida para este trabalho, a qual envolvia o monitoramento contínuo de área agrícola.

Houve, inicialmente, um estudo no sentido de se utilizar hardware existente. Os nodos mais utilizados em experimentos científicos atualmente são os do modelo Mica-Z[1] encontrados em *kits* Crossbow[2]. O uso desses nodos seria conveniente pelo fato da disponibilidade imediata em nosso laboratório. Esses nodos têm como vantagem uma implementação da norma IEEE 802.15.4-2003/2006 com código aberto e bem conhecida pela comunidade científica. Essa norma é implementada através do software OpenZB[3] sobre o sistema operacional TinyOS[4].

No entanto, foram detectados alguns problemas na adoção deste hard-

[1] `http://bullseye.xbow.com:81/Products/productdetails.aspx?sid=164`
[2] `http://www.xbow.com`
[3] `http://www.open-zb.net`
[4] `http://www.tinyos.net`

ware. Um dos problemas é que ele não está totalmente em conformidade com padrão IEEE 802.15.4 no que tange à utilização do modo com *beacon*[5]. Esse fato é decorrente do hardware do Mica-Z não ter um relógio que satisfaça os requisitos de $\pm 40 ppm$ na frequência de 2.4 GHz. Observando-se que o projeto do Mica-Z possui praticamente 10 anos e o fato de ainda existirem problemas sérios de compatibilidade com o padrão a serem corrigidos, houve uma grande insegurança na adoção desses nodos. Ademais, após pesquisa no mercado, verificou-se que o hardware do Mica-Z não possui a venda uma expansão que permita a utilização de *data logging*. Finalmente, quando foram feitos testes em campo, constatou-se que esses nodos suportam comunicação em visada direta de apenas 30m de distância na média.

Devido a essas deficiências, optou-se por buscar outra alternativa de hardware que resolvesse esses problemas. Nesse sentido, um novo hardware foi especificado com objetivo de atender às seguintes características:

- Permitir o uso em campo, ainda que com vegetação densa como obstáculo, com alcance de comunicação superior a 30m.
- Ter alcance de comunicação superior a 100m quando os nodos tiverem visada direta.
- Permitir a utilização interna em prédios, possibilitando a comunicação entre nodos separados por pelo menos por um andar.
- Ter recursos para economizar energia nos módulos não utilizados. Se possível, permitir desligar unidades externas.
- Ter relógio de tempo real para realizar operação de *data logging* permitindo longos experimentos.
- Ter *slot* para uSD card para *data logging*.
- Usar transceptor compatível com IEEE 802.15.4-2006.
- Ter MAC confiável para utilizar modo *beacon* ativo com segurança.
- Possibilidade de alterar o código MAC, permitindo pesquisa básica sobre alternativas para o padrão IEEE 802.15.4.

[5] `http://code.google.com/p/tinyos-main/source/browse/trunk/tos/lib/mac/tkn154/README.txt`

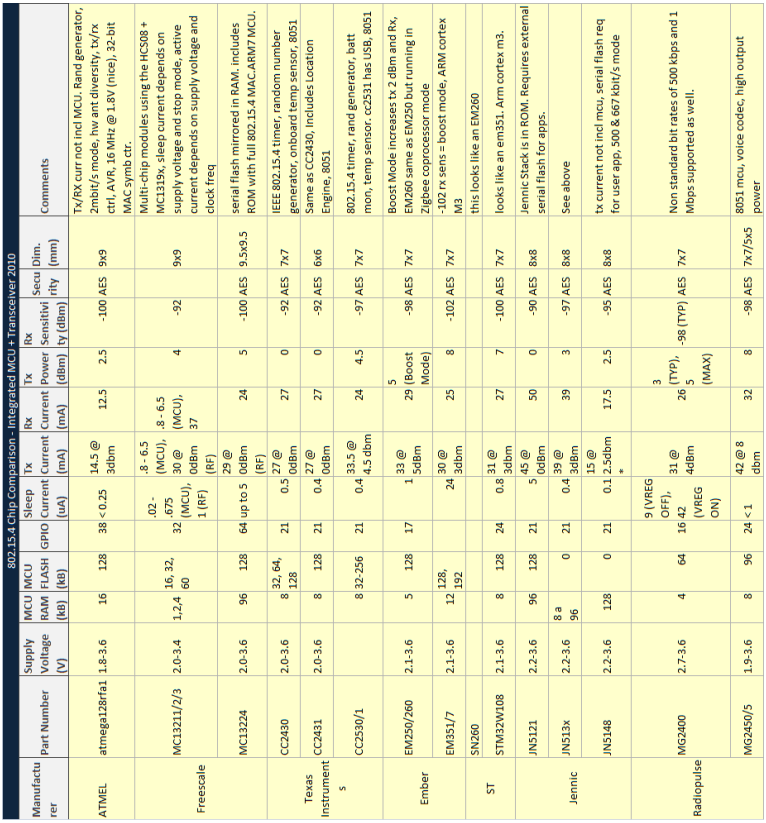
802.15.4 Chip Comparison - Transceiver 2010										
Manufacturer	Part Number	Supply Voltage (V)	Sleep Current (uA)	Tx Current (mA)	Rx Current (mA)	Tx Power (dBm)	Rx Sensitivity (dBm)	Security	Dim. (mm)	Comments
ATMEL	AT86RF230	1.8-3.6	0.02	16.5 @ 3 dBm	15.5	3	-101		5x5	no security
	AT86RF231	1.8-3.6	0.02	14.3 @ 3 dBm	13.2	3	-101	AES	5x5	Antenna diversity up to 2Mbits/s
	AT86RF212	1.8-3.6	0.2	17 @ 5 dBm	9.2	10	-110	AES	5x5	780/868/915 Mhz up to 1Mbit/s
Freescale	MC13201	2.0-3.4	1	30 @ 0dBm	37	4	-91		5x5	Non-zigbee/802.15.4 apps, PA and LNA output pins for ext amps, Tx/Rx Switch
	MC13202/3	2.0-3.4	1	30 @ 0dBm	37	4	-91		5x5	MC13202 = 802.15.4 only, MC1303 = 802.15.4 + Zigbee stack
	MC13191/2/3	2.0-3.4	1	30 @ 0dBm	37	4	-92		5x5	MC13191 = Non 802.15.4/Zigbee, MC13192 = 802.15.4 only, MC13193 = 802.15.4 + Zigbee
Texas Instruments	CC2420	2.1-3.6 (VREG), 1.6-3.6 (IO)	20	25.8 @ 0dBm	18.8	0	-95	CTR, CCM, AES	7x7	one of the first 802.15.4 chips out
	CC2520	1.8-3.8	0.03(LPM1), 175(LPM2)	33.6 @ 5dBm	22.3 (Normal), 18.8 (Low Current)	5(Typ), 7(Max)	-98	AES	5x5	
UBEC	UZ2400	2.4-3.6	2	23 @ 0dBm	19	0	-95	AES, CTR, CCM	6x6	Check errata on company website for issues with security engine and Zigbee 2006 spec.
Microchip	MRF24J40	2.4-3.6	2	22	18	0	-91	AES, CTR, CCM	6x6	Looks like same chip as UZ2400. Pin compatible and from Microchip's stack, looks like register compatible too.
RADIOPULSE	MG2410	1.8-3.6	<1	44 @ 10dbm	19.5	10	-100	AES	5x5	data rate from 31.2K to 4 Mbps, msk & oqpsk modulation high tx output, temp sensor

Figura 10: Comparação dos transceptores comerciais disponíveis no ano de 2010.

4.2.1 Levantamento de informações e escolha de componentes

A decisão tomada foi desenvolver uma plataforma de hardware que suportasse as características enumeradas. Com base em indicadores levantados durante uma pesquisa preliminar, em 2010 foi elaborada uma tabela comparativa (Figura 10) a fim de auxiliar na decisão do rádio a ser usado no projeto. Na figura são apresentados os resultados da pesquisa de rádios com microprocessador embutido conhecidos como SoC (System on Chip).

Na metade de 2012 foi realizada a mesma pesquisa a fim de verificar uma possível mudança do mercado. Nas tabelas são apresentadas apenas a evolução dos dispositivos, com objetivo de resumir a informação, mostrando



802.15.4 Chip Comparison - Integrated MCU + Transceiver 2010														
Manufacturer	Part Number	Supply Voltage (V)	MCU RAM (kB)	MCU FLASH (kB)	GPIO	Sleep Current (uA)	Tx Current (mA)	Rx Current (mA)	Tx Power (dBm)	Rx Sensitivity (dBm)	Security	Dim. (mm)	Comments
ATMEL	atmega128rfa1	1.8-3.6	16	128	38	< 0.25	14.5 @ 3dbm	12.5	2.5	-100	AES	9x9	Tx/Rx curr not incl MCU. Rand generator, 2mbit/s mode, hw ant diversity, tx/rx ctrl. AVR, 16 MHz @ 1.8V (nice), 32-bit MAC symb ctr.
Freescale	MC13211/2/3	2.0-3.4	1,2,4	16, 32, 60	32	.02 - .675 (MCU), 1 (RF)	.8 - 6.5 (MCU), 30 @ 0dBm (RF)	.8 - 6.5 (MCU), 37	4	-92		9x9	Multi-chip modules using the HCS08 + MC1319x, sleep current depends on supply voltage and stop mode, active current depends on supply voltage and clock freq
	MC13224	2.0-3.6	96	128	64	up to 5	29 @ 0dBm (RF)	24	5	-100	AES	9.5x9.5	serial flash mirrored in RAM. includes ROM with full 802.15.4 MAC.ARM7 MCU.
Texas Instruments	CC2430	2.0-3.6	8	32, 64, 128	21	0.5	27 @ 0dBm	27	0	-92	AES	7x7	IEEE 802.15.4 timer, random number generator, onboard temp sensor. 8051
	CC2431	2.0-3.6	8	128	21	0.4	27 @ 0dBm	27	0	-92	AES	6x6	Same as CC2430, Includes Location Engine, 8051
	CC2530/1		8	32-256	21	0.4	33.5 @ 4.5 dbm	24	4.5	-97	AES	7x7	802.15.4 timer, rand generator, batt mon, temp sensor. cc2531 has USB, 8051
Ember	EM250/260	2.1-3.6	5	128	17	1	33 @ 5dBm	29 (Boost Mode)	5	-98	AES	7x7	Boost Mode increases tx 2 dBm and Rx, EM260 same as EM250 but running in Zigbee coprocessor mode
	EM351/7	2.1-3.6	12	128, 192	24		30 @ 3dbm	25	8	-102	AES	7x7	-102 rx sens = boost mode, ARM cortex M3
ST	SN260	2.1-3.6	8	128	24	0.8	31 @ 3dbm	27	7	-100	AES	7x7	this looks like an EM260
	STM32W108												looks like an em351. Arm cortex m3.
Jennic	JN5121	2.2-3.6	96	128	21	5	45 @ 0dBm	50	0	-90	AES	8x8	Jennic Stack is in ROM. Requires external serial flash for apps.
	JN513x	2.2-3.6	8 a 96	0	21	0.4	39 @ 3dBm	39	3	-97	AES	8x8	See above
	JN5148	2.2-3.6	128	0	21	0.1	15 @ 2.5dbm *	17.5	2.5	-95	AES	8x8	tx current not incl mcu. serial flash req for user app, 500 & 667 kbit/s mode
Radiopulse	MG2400	2.7-3.6	4	64	16	9 (VREG OFF), 42 (VREG ON)	31 @ 4dBm	26	3 (TYP), 5 (MAX)	-98 (TYP)	AES	7x7	Non standard bit rates of 500 kbps and 1 Mbps supported as well.
	MG2450/5	1.9-3.6	8	96	24	< 1	42 @ 8 dbm	32	8	-98	AES	7x7/5x5	8051 mcu, voice codec, high output power

Figura 11: Comparação de SoC MCU e transceptor no ano de 2010.

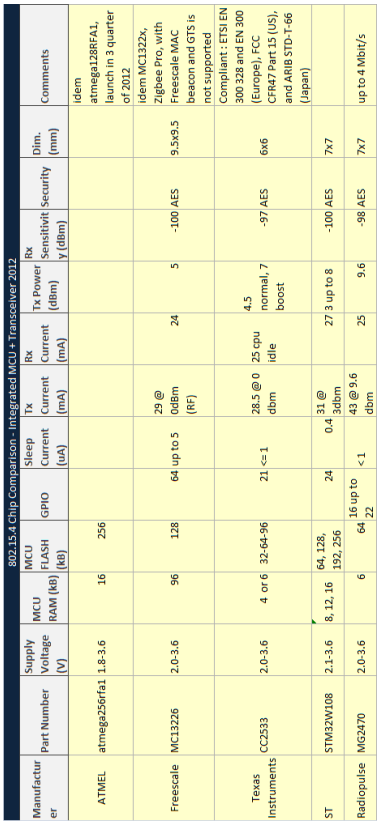
802.15.4 Chip Comparison - Transceiver 2012										
Manufacturer	Part Number	Supply Voltage (V)	Sleep Current (uA)	Tx Current (mA)	Rx Current (mA)	Tx Power (dBm)	Rx Sensitivity (dBm)	Security	Dim. (mm)	Comments
ATMEL	AT86RF232	1.8-3.6	0.4	13.8	11.8	3	-100	AES	5x5	Compliant to IEEE 802.15.4-2011, EN 300 328/440, FCC-CFR-47 Part 15, ARIB STD-66, RSS-210
	AT86RF233	1.8-3.6	0.02	13.8	11.8 / 6 Smart Receiving	4	-101	AES	5x5	Fully integrated, fast settling PLL to support Frequency Hopping, Supports 500kHz channel spacing
Freescale	MC13242_RF	1.8-3.6	< 1	15 @ 0 dBm	15 @ -102 dBm	10	-102	AES	5x5	Compliant to IEEE 802.15.4-2006
UBEC										acquired by Microchip
JENNIC										acquired by NXP
EMBER	EM260	2.1-3.6	1	36mA @ 2.5 (dBm) Normal, 42mA @ 4.5 (dBm) Boost	36mA Normal, 38mA Boost	2.5 normal, 4.5 boost	normal -99, -100 boost	AES	6x6	Stack is in chip that communicates by SPI
RADIOPULSE	CH0900	1.8-3.6	<1	28 @ 13dbm	13	13	1Mbit @ -90 dBm, -110 dBm @ 40k	AES		data rate from 31.2K to 4 Mbps, msk & oqpsk modulation high tx output, temp sensor

Figura 12: Comparação dos Transceptores no ano de 2012 – evolução.

como o mercado se alterou neste intervalo de tempo. A Figura 12 compara novamente apenas os rádios e na Figura 13 são apresentadas as comparações para os SoC.

Nas tabelas, os indicadores principais de economia de energia levantados estão na coluna *sleep* e *Tx/Rx current*. Quanto menores forem estes indicadores, maior será a economia de energia sem perda de funcionalidade (distância de alcance). Na maioria dos casos, os transceptores da ATMEL consomem menos da metade dos concorrentes incluindo o período de evolução. No caso dos SoC, o resultado foi semelhante ao dos transceptores. No pior dos casos, o SoC da ATMEL gasta muito menos energia que o dos concorrentes, colocando uma maior potência na saída.

Os principais indicadores de qualidade para transmissão e recepção de dados são *Tx Power* e *Rx Sensitivity*. Quando maior for a potência, maior será o alcance de envio. Contudo, de nada adianta uma potência alta se não existir boa sensibilidade para recepção. Para um transceptor poder receber os dados, quanto menor for a sensibilidade, menor será a potência necessária. Neste sentido, todos os hardwares foram relativamente parecidos. Tanto os transceptores quantos os SoC da ATMEL têm sensibilidade igual ou superior a *-100 dBm*, enquanto nem todos os concorrentes conseguem alcançar este valor. Com relação à potência, os transceptores da ATMEL ficaram no meio da escala para frequências de 2.4GHz. No caso de frequências sub-gigahertz,



802.15.4 Chip Comparison - Integrated MCU + Transceiver 2012													
Manufacturer	Part Number	Supply Voltage (V)	MCU RAM (kB)	MCU FLASH (kB)	GPIO	Sleep Current (uA)	Tx Current (mA)	Rx Current (mA)	Tx Power (dBm)	Rx Sensitivity (dBm)	Security	Dim. (mm)	Comments
ATMEL	atmega256rfa1	1.8-3.6	16	256									idem atmega128RFA1, launch in 3 quarter of 2012
Freescale	MC13226	2.0-3.6	96	128	64	up to 5	29 @ 0dBm (RF)	24	5	-100	AES	9.5x9.5	idem MC1322x, ZigBee Pro, with Freescale MAC beacon and GTS is not supported
Texas Instruments	CC2533	2.0-3.6	4 or 6	32-64-96	21	<=1	28.5 @ 0 dbm	25 cpu idle	4.5 normal, 7 boost	-97	AES	6x6	Compliant : ETSI EN 300 328 and EN 300 328 (Europe), FCC CFR47 Part 15 (US), and ARIB STD-T-66 (Japan)
ST	STM32W108	2.1-3.6	8, 12, 16	64, 128, 192, 256	24	0.4	31 @ 3dbm	27	3 up to 8	-100	AES	7x7	
Radiopulse	MG2470	2.0-3.6	6	64	16 up to 22	<1	43 @ 9.6 dbm	25	9.6	-98	AES	7x7	up to 4 Mbit/s

Figura 13: Comparação de SoC MCU e transceptor no ano de 2012 – evolução.

não existia concorrente até a comparação de 2012 onde apareceu apenas um com qualidade inferior. Para os SoC, a relação de potência ficou na média dos demais.

As tabelas foram criadas de acordo com os valores encontrados nos próprios *datasheets* dos dispositivos, comparando indicadores que determinavam a qualidade e economia de energia do dispositivo. Embora existissem outros dispositivos disponíveis no mercado, apenas foram levados em conta aqueles que atendessem ao padrão IEEE 802.15.4.

O projeto considerado mais promissor naquele momento foi o da ATMEL[6], o qual oferecia três formas para adquirir os rádios: apenas transceptor, rádio com com microcontrolador integrado, e módulo SoC com rádio e microcontrolador (MCU – *Microcontroller Unit*). A grande vantagem do SoC integrado, quando comparado com um rádio com transceptor integrado, é que neste último o consumo energético é maior. Outro fator determinante foi avaliação entre os projetos de como a pilha IEEE 802.15.4 é disponibilizada. A pilha da ATMEL é escrita em linguagem C, incluindo o modo com *beacon* ativo, e com muitos exemplos disponíveis.

A opção adotada foi a do rádio transceptor AT86RF212[7], o qual é compatível com a IEEE-802.15.4-2009c. A escolha do dispositivo sub-gigahertz teve como premissa o maior alcance, já que as frequências mais baixas permitem um maior alcance. A potência de sinal nas faixas abaixo de 1 GHz também é maior, o que motiva ainda mais a escolha. Outro fator interessante é que o *chip* permitiria a exploração do recurso para a banda de 780Mhz japonesa. Infelizmente nesta configuração não existe ainda um SoC disponível.

A melhor opção na época para o microprocessador também era da ATMEL com uma nova linha de processadores de 8/16 bits, XMEGA[8]. Essa linha é um avanço na atual linha de microcontroladores de 8 bits, conhecidos AVR 8, pois trabalha internamente com 16 bits e suporta frequências de até 32 Mhz oferecendo um melhor desempenho. Outro fator muito importante é que ele permite o controle de energia de todos os periféricos. O microprocessador escolhido foi o ATxmega192A3 e as principais características dessa família são:

- 4 canais de DMA (Direct Memory Access), só existente em processadores de 32 bits.
- Sistemas de eventos, dispensa o processador para iniciar e executar tarefas, tais como: iniciar conversão analógica digital, ativar/desativar

[6] `www.atmel.com`
[7] `http://www.atmel.com/devices/AT86RF212.aspx`
[8] `http://www.atmel.com/products/microcontrollers/avr/AVR_XMEGA.aspx`

timers e disparar canal DMA.

- Acelerador de criptografia AES-128/DES-64.
- multi-nível de interrupção.
- uso de bibliotecas para função de teclas capacitivas o que permite o desenvolvimento de equipamentos para uso externo e condições intemperes.

Um vez escolhidos os principais dispositivos a serem utilizados na plataforma, foi elaborado um projeto de hardware para economizar o máximo de energia possível. O microcontrolador adotado oferece um relógio de tempo real de 16 bits, mas de tempos em tempos é necessário um tratamento de estados do MAC, o que é indesejável. Com o uso de um relógio de tempo real (RTC) externo resolve-se o problema de acordar o microcontrolador em curtos intervalos. Foi escolhido um de alta precisão da Seiko[9] por ter excelente eficiência e ser de baixo custo. Uma característica desse RTC é que ele permite a correção de tempo, pois se este adiantar ou atrasar é possível realizar ajustes nos seus valores.

Outro componente importante no hardware seria a comunicação com o computador. A interface de comunicação mais simples é um SoC USB FTDI[10] que cria uma porta serial (CDC-COM) no PC. O problema é o alto custo relacionado com esse dispositivo. Em seu lugar foi adotado mais um microcontrolador Cypress PSoC[11] com interface USB de baixo custo. A grande vantagem de se colocar outro microcontrolador na placa é que ele pode ser utilizado por demanda, economizando energia, pois estará em modo baixa energia.

Outro fator levado em consideração foram os diodos emissores de luz (LED). Os LEDs mais usados em projetos de hardware consomem em torno de 100mW para fornecer boa luminosidade. Este é um consumo excessivo para sistemas RSSF. Quando se compara o consumo desses LEDs com o consumo do rádio (Figuras 10, 12, 11 e 13), não é dificil perceber que um LED pode consumir mais que o dobro da corrente do rádio da solução escolhida. A fim de minimizar este consumo, foram selecionados LEDs de alta eficiência luminosa, restringido seu consumo máximo à 10mW no projeto do hardware.

Por último, é necessário ter uma boa fonte que suporte o uso com o computador e por baterias. Foi realizada uma breve pesquisa a fim de levantar qual seria o melhor uso e quais são as melhores opções. Em busca de alternativas para aplicações de baixo consumo verificou-se a tendência recente na

[9] `http://www.eea.epson.com/portal/page/portal/home`
[10] `http://www.ftdichip.com`
[11] `http://www.cypress.com/?id=1353`

direção de se adotar soluções *Energy Harvest (EH)*[12]. Dispositivos com características EH são aqueles que funcionam com uso de recursos externos por meio de fontes de energia alternativa. O hardware foi desenvolvido para suportar dois modos. No modo 1, o hardware foi equipado com uma fonte que permite a utilização via USB ou 5 a 30Vdc. No modo 2, o hardware permite o uso de baterias conectadas diretamente ao circuito via fonte DC/DC. O uso de baterias ligadas diretamente ao circuito principal facilita os estudos feitos no decorrer desta dissertação que avaliam o consumo de energia através de baterias. No caso, quando o conversor DC/DC é ativado, ele transforma a energia da entrada (baterias ou fonte EH) que esteja na faixa de 0.5 até 3.2Vdc para 3.3Vdc. Se a fonte for desativada, as baterias estarão conectadas diretamente ao circuito.

4.2.2 Desenvolvimento do hardware

O projeto foi chamado de ATxMEGAv2 e elaborado na ferramenta CAD (*Computer-Aided Design*) Altium Designer[13]. A escolha dessa ferramenta deve-se principalmente a ela permitir integração com softwares de modelagem de produtos 3D. A versão final do projeto pode ser visualizada na Figura 14. O sistema final conta com pilha IEEE 802.15.4 compatível com o *addendum 2009c*, Sistema FAT32 para uso com cartões SDHC com alimentação controlada, microprocessador de 32Mhz, coprocessador de comunicação USB 2.0 *Full Speed (12 Mbit/s)*, 3 conectores de expansão: I2C, 8 pinos de I/O com porta serial, porta analógica e I/O, RTC com calendário compatível até 2099.

Os resultados alcançados com a placa desenvolvida cumpriram os requisitos colocados por este trabalho. O alcance da comunicação foi de 200m com visada direta e antena de 2dBi executando a 915 Mhz e modulação O-QPSK (mesmas características de símbolos e velocidade da faixa 2.4 GHz). O consumo médio do sistema foi de, no máximo, 130mW sem *data logging*. O *data logging* necessitou de picos de até 500mW. O sistema funcionou com a pilha da ATMEL e nos testes de *throughput* alcançou 234 kbps sem o protocolo CSMA/CA, sem mensagens de reconhecimento (sem ACKs) e com pacotes de 120 bytes. Este resultado está de acordo com a taxa de bits máxima teórica que é de de 250 kbps.

Como o resultado obtido foi satisfatório, um outro projeto foi elaborado a fim de se desenvolver um kit para estudo e desenvolvimento de abordagens relacionadas com a norma IEEE 802.15.4. Este projeto que foi chamado

[12] `http://www.energyharvesting.net`
[13] `www.altium.com`

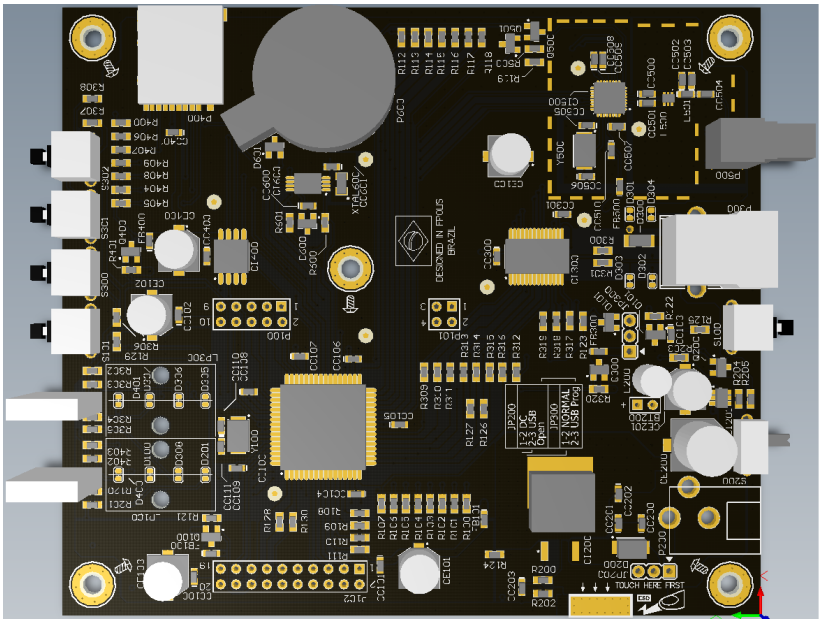


Figura 14: Projeto CAD do hardware desenvolvido para ser utilizado.

de KITRFA1 teve como foco o desenvolvimento da placa com um rádio e microprocessador na mesma pastilha, ou seja, componente integrado SoC. O microprocessador utilizado foi o AVR ATMEGA128RFA1[14]. Foram adicionados memória EEPROM com sensor de temperatura, sensor de presença e sensor de luminosidade. A comunicação passou a ter uma porta RS-485 *full-duplex* e duas portas de comunicação CDC-COM. Foram acrescentadas também memória FLASH 256k bytes e possibilidade de uso do rádio com antena SMA ou cerâmica. Portas de expansão foram adicionadas nos mesmos padrões existentes dos kits da própria ATMEL para facilitar a conexão com outros dispositivos, e estas suportam todas as funcionalidades do microprocessador em número de I/O.

Estes rádios suportam o uso de outras pilhas como a Zigbee[15], ISA-100[16] e WirelessHART[17]. Como objeto de estudo foi realizado a portabili-

[14] `http://www.atmel.com/products/microcontrollers/Wireless/single-chip_solutions.aspx`

[15] `http://www.zigbee.org`

[16] `http://www.isa.org`

[17] `http://www.hartcomm.org/protocol/wihart/wireless_technology.html`

dade da pilha Zigbee para os dois protótipos e os resultados foram satisfatórios. Nesse estudo foi verificado que as duas plataformas podem servir como base para testes e comparações com outros modelos de comunicação.

4.3 PROJETO DE SOFTWARE BÁSICO RELACIONADO

Devido às várias características desejadas no projeto, foi necessário dividir em várias partes a construção do software básico, sendo elas: testes de hardware, portabilidade da pilha IEEE 802.15.4, *drivers* do microprocessador AVR, *drivers* do microprocessador PSoC, software para gravação da placa, *sniffer* e sua integração com Wireshark[18], gerador de ruídos para o IEEE 802.15.4 e, finalmente, *drivers* RTC e portabilidade para uso de FAT 32.

4.3.1 Testes de hardware

Os testes de hardware tiveram por objetivo encontrar problemas com projeto e montagem do hardware. Uma aplicação foi desenvolvida para efetuar esse diagnóstico e permitir buscar problemas com a utilização de uma ferramenta de depuração de software. Os testes compreenderam as seguintes atividades:

1. Verificar se o processador funciona com a ferramenta de depuração.

2. Testar cada um dos LEDs da placa visualmente.

3. Testar cada uma das portas de I/O com ajuda de osciloscópio.

4. Ligar/Desligar o conversor DC/DC e avaliar a alimentação da placa utilizando um multímetro.

Os testes foram executados em uma das placas desenvolvidas para verificar a existência de inconsistências. Os resultados não constataram qualquer problema. Com essa etapa completa foi possível avançar para a próxima: efetuar o porte da pilha 802.15.4 para o hardware.

[18] `http://www.wireshark.org`

4.3.2 Portabilidade da pilha IEEE 802.15.4

O porte da pilha IEEE 802.15.4 foi executado com ajuda de um manual elaborado pela ATMEL. O manual orienta detalhadamente sobre como realizar as alterações necessárias, as quais consistem em informar como o microcontrolador irá se comunicar com o rádio, configurando todas as portas de I/O, e reescrevendo o arquivo *makefile* para a nova placa.

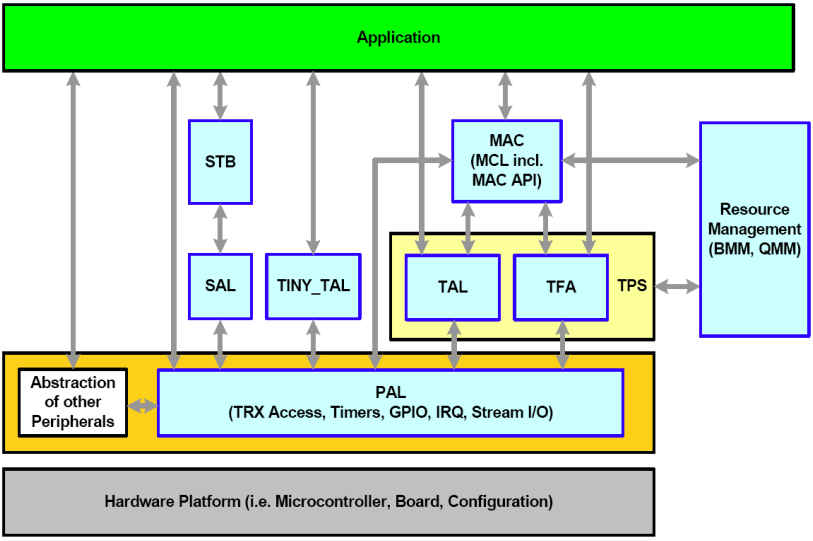


Figura 15: Arquitetura do MAC ATMEL (ATMEL, 2011).

A arquitetura do MAC da ATMEL pode ser visualizada na Figura 15. Essa arquitetura consiste de várias camadas com funções bem específicas, facilitando todo o desenvolvimento, sendo elas (ATMEL, 2011):

- **Camada de Abstração da Plataform (PAL):** É a camada do microcontrolador, sendo responsável por todo acesso ao hardware como: comunicação, LED, timers etc.

- **Camada de Abstração do Transceptor (TAL):** Contém toda a parte de software responsável pelo gerenciamento do rádio de acordo com a norma IEEE 802.15.4, exemplo: CSMA/CA, LQI, máquina de estado do rádio etc.

- **TinyTAL:** É uma versão mais leve da TAL não compatível com IEEE 802.15.4. Sua finalidade é criar aplicações mais simples.

- **Acesso aos Recursos do Transceptor (TFA):** Nesta camada fica toda a parte de acesso a recursos do transceptor que não faz parte da IEEE 802.15.4, como: leitura da bateria, transmissão contínua, medição de temperatura etc.

- **Camada MAC:** Consiste na implementação da pilha IEEE 802.15.4-2006.

- **Camada de Abstração de Segurança (SAL):** Fornece uma interface de programação de aplicação (API) para acessar os mecanismos de criptografia AES-128 e DES-64.

- **Software Toolbox (STB):** API escrita sobre a SAL para fácil acesso à parte de segurança.

- **Gerenciamento de Recursos (BMM/QMM):** Responsável pelo gerenciamento de *buffers* e *fila* para envio e recepção de dados/comandos.

Na sequência, a próxima etapa consistiu em alterar os arquivos de *makefiles* para compilação de aplicações de demonstração encontradas no próprio MAC da Atmel, a fim de executar testes com o hardware. Nos testes de hardware que haviam sido feitos, não era totalmente possível determinar se o projeto e montagem estavam corretos com respeito à utilização do rádio. Para avaliar o rádio, a placa foi testada com uma aplicação de exemplo a qual executa um teste de desempenho (*throughput*) do rádio. Esta é uma aplicação simples e não requer o uso do MAC, apenas os *drivers* para comunicar com o rádio. Na Figura 16 é possível visualizar a estrutura da pilha de software. O exemplo de testes de *throughput* utiliza a camada TAL. O trabalho de adaptar o exemplo para a placa é uma tarefa complexa, havendo necessidade de se acertar as configurações para que a MCU possa comunicar-se adequadamente com o rádio. Os resultados finais obtidos foram satisfatórios.

Como a MCU e rádio estavam se comunicando, foi realizado mais um porte, agora para uma aplicação que utiliza o MAC 802.15.4. O software consiste em um FFD configurado como coordenador e um RFD. A aplicação estabelece uma rede e o dispositivo, uma vez autenticado, envia 8 bytes aleatórios para o coordenador. O resultado do teste foi positivo, sendo que a etapa de portabilidade da pilha IEEE 802.15.4 foi considerada como concluída.

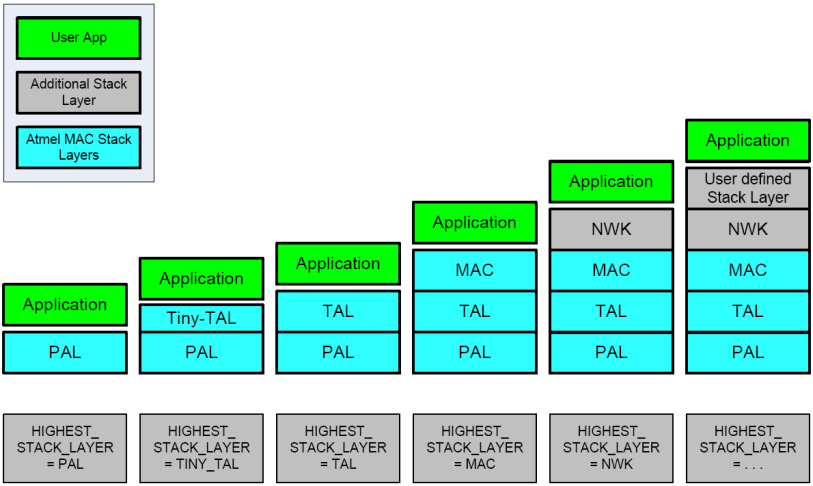


Figura 16: Formas de uso da pilha de Software (ATMEL, 2011).

4.3.3 *Drivers* microcontrolador AVR e PSoC

A etapa de construção de *drivers* para o microcontrolador AVR foi necessária para melhorar a forma de desenvolvimento e criação de uma base para o restante das etapas. O projeto do MAC da ATMEL foi concebido para ser facilmente compreendido, mas isso não significa necessáriamente que é simples de desenvolver usando este MAC. Nesta etapa, as tarefas desempenhadas foram:

1. Criação de um *makefile* para cada parte do projeto (MAC, Rádio, MCU, *drivers* e serviços).

2. Criação de um *makefile* simplificado para o projeto.

3. Criação de um projeto para uso de *drivers* já existentes para o processador.

A tarefa de construir os *drivers* para o PSoC foi dividida em duas etapas: construir a comunicação serial com a MCU AVR e desenvolver a comunicação USB com computador. A parte da serial e USB foi simples de cumprir. A criação de softwares simples com PSoC é muito produtivo e rápido pelo fato de quase não existir código a ser escrito. Basta, praticamente,

configurar e utilizar. Um detalhe importante foi a necessidade que se teve de alterar manualmente a parte de *drivers* para o Windows[19] x64, uma vez que o o arquivo $*.inf$ não é o mesmo.

4.3.4 Software para gravação da placa

O desenvolvimento de uma nova placa requer um trabalho extra para embarcar o software na placa sem ajuda de um aparelho programador. Para que isso seja possível, a MCU deve autogravar os dados enviados por uma porta de comunicação. Para realizar esta tarefa, um software especifico (*bootloader*) para cada MCU e placa deve ser desenvolvido, e é necessário uma aplicação no PC para efetuar a comunicação e transferência do arquivo.

Como foram desenvolvidas duas placas, dois softwares de *bootloader* foram implementados. Como os processadores AVR mega e AVR xmega são diferentes em sua arquitetura, os softwares de *bootloader* são completamente diferentes. Contudo, utiizam o mesmo protocolo de comunicação com o computador.

O *bootloader* é um aplicativo executado apenas no momento da inicialização da placa. Isso acontece após a placa ser ligada ou reinicializada. Esse software é gravado em uma região especial da memória de programa e somente ele tem acesso à leitura e à gravação da memória de programa. O software deve aguardar uma sinalização para operar, e geralmente um botão tem essa função. Assim, quando a placa é ligada com o botão pressionado, o *bootloader* entra em ação.

A aplicação do lado do computador deve se comunicar com o *bootloader* da placa para poder repassar a nova versão do software. Essa aplicação deve ler o arquivo gerado pelo compilador e repassar, por meio de um protocolo seguro, os dados para o *bootloader* gravar na memória de programa.

O software do computador foi chamado AVRxPROG sendo uma versão melhorada de uma nota de aplicação da própria ATMEL. A versão do AVRxPROG executa em Windows 32/64 bit e utiliza os arquivos XML do próprio ATMEL Studio 6 para se auto configurar. O software suporta o uso de até 256 portas seriais incluindo as CDC-COM USB. Ele permite a gravação de qualquer familia AVR (tiny, mega ou xmega) de qualquer tamanho de memória de programa (4-384k bytes). Tanto o AVRxPROG como os *bootloader* suportam a transferência por blocos de dados, garantindo uma gravação de 128k bytes de programa em 15 segundos. Na Figura 17 pode ser visualizado um teste de conexão com o KITRFA1.

[19] `www.microsoft.com`

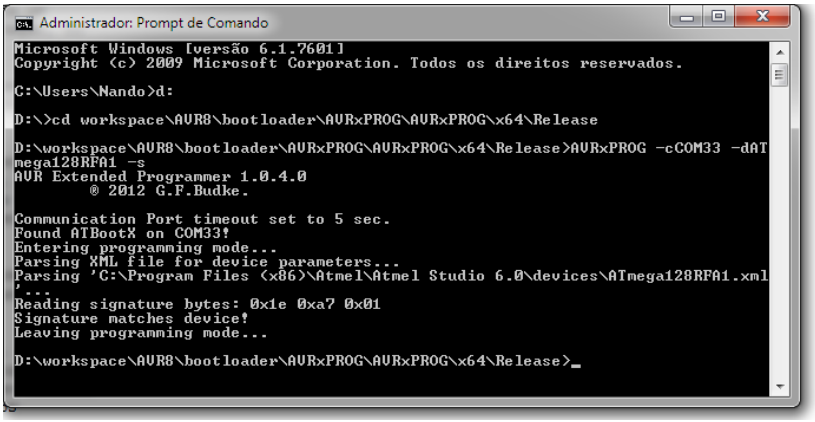


Figura 17: AVRxPROG versão x64 com KITRFA1.

4.3.5 *Sniffer* e integração com Wireshark

A criação de um hardware de *sniffer* e a integração com o software Wireshark teve como objetivo melhorar o entendimento da pilha IEEE 802.15.4. O Wireshark é um analizador de protocolos gratuito e de código aberto. Ele é utilizado para descobrir problemas de rede, análises, desenvolvimento de sofwtare de comunicação de dados e educação. Originalmente, antes de 2006, esse software era conhecido como *Ethereal*. O *sniffer* foi baseado numa aplicação de demonstração da ATMEL, utilizando o recurso *modo promíscuo*. Esse recurso permite que todas as informações trafegadas sejam visualizadas independentemente do endereço de rede. O único requisito é que no hardware devem ser configurados a página e o canal de comunicação.

A Figura 18 apresenta um exemplo de comunicação no Wireshark. Pode ser observada na figura uma aplicação desenvolvida a qual funciona como ponte de dados para o Wireshark. Essa aplicação é necessária pois o Wireshark utiliza as placas de rede do computador ou algum canal *pipe nomeado* de comunicação para receber os dados a serem analizados. A aplicação ponte foi chamada de *Wireshark Bridge (wsbridge)*. Ela abre uma porta serial existente no PC e cria uma interface *pipe* local, no PC, chamada *Local* : $\backslash\backslash.pipe\backslash wireshark$. Com isso é possível abrir essa interface no Wireshark e receber os dados brutos da IEEE 802.15.4.

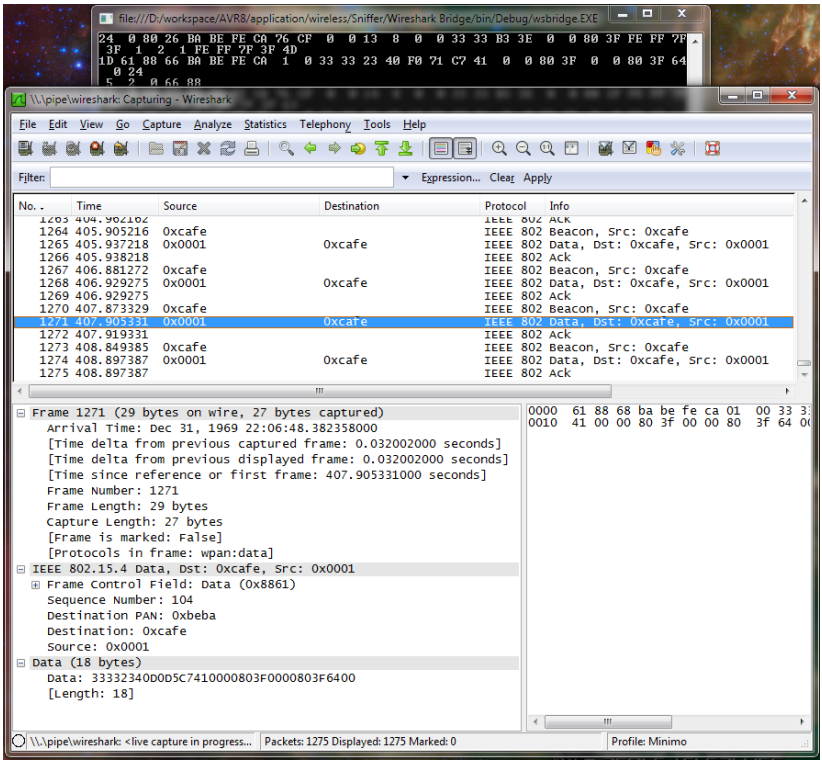


Figura 18: Wireshark executando em conjunto com *Sniffer*.

4.3.6 Gerador de ruídos para IEEE 802.15.4

Outro recurso interessante desenvolvido foi um gerador de ruído (ou gerador de interferências). O objetivo dessa ferramenta foi o de possibilitar a criação de dois tipos de interferências controladas no meio:

- A primeira busca manter o meio ocupado. Dessa forma, quando qualquer rádio naquele canal tentar acessar o meio com CSMA/CA, via de regra irá encontrar o meio ocupado.

- A segunda busca produzir uma interferência, a qual permita que os dispositivos acessem o meio e quando eles estiverem transmitindo, corromper o pacote, induzindo “ruídos” no meio de forma aleatória.

O modelo adotado de interferência tem por base o elaborado por (WILLIG, 2008). Foram desenvolvidos dois polinômios para efetuar os cálculos de tempo de interferência e tempo inativo do algoritmo. Os polinômios foram implementados para gerar uma interferência que corresponde à curva *Log-Normal*(WILLIG, 2008) com 30% de uso do meio. Ao executar o gerador de ruídos é possível configurar o tipo de interferência sendo meio ocupado e/ou interferência eletromagnética e informar se a interferência será contínua ou segundo a distribuição *Log-Normal*.

Os testes foram aplicados utilizando um *sniffer* para constatar a atuação do ruído e uma aplicação que ficava enviando *broadcast* a cada segundo de uma variável utilizada como contador, assim em cada pacote a variável era incrementada. Com a aplicação enviando informações é possível visualizar todos os pacotes enviados. Ao ligar o gerador simulando meio ocupado contínuo, os pacotes pararam de ser recebidos. Ao desligar o gerador novamente os dados passaram a ser recebidos. Ao aplicar o teste eletromagnético contínuo, os pacotes que eram recebidos sempre estavam corrompidos. O último teste a ser realizado foi o da distribuição em *Log-Normal* de meio ocupado e eletromagnético. Os resultados foram satisfatórios onde alguns pacotes se perdiam, enquanto outros eram recebidos porém estavam corrompidos.

4.3.7 *Drivers* RTC e portabilidade para uso de FAT 32

A última parte desenvolvida foram os *drivers* de relógio de tempo real e o porte do sistema de arquivos FAT32 conhecido como FatFS[20] e o sistema FAT32 de Roland-Riegel[21].

A comunicação com RTC dependia da adaptação dos *drivers* I2C. os quais ainda não haviam sido portados para as plataformas. Eles foram escritos com base nas notas de aplicação da ATMEL para a comunicação I2C (Inter Integrated Circuit). A abstração do RTC foi implementada após validação da comunicação I2C com uso de osciloscópio. O teste consistiu em colocar a bateria para funcionamento do RTC sem fontes externas e configurar o relógio para a data atual, desligar a placa, consultar o horário e verificar se estava correto. O teste foi bem sucedido e atualmente o relógio continua em funcionamento. Notou-se que, após certo tempo, o relógio começou a adiantar. Segundo a especificação, isso acontece pela escolha dos componentes e detalhes mecânicos da placa de circuito impresso, os quais devem ser compensados em laboratório. No entanto, esses ajustes não foram realizados. Também não foi desenvolvida a aplicação de ajuste de desvio do relógio o

[20] `http://elm-chan.org/fsw/ff/00index_e.html`
[21] `http://www.roland-riegel.de/sd-reader`

qual utiliza os registradores do RTC para essa configuração.

O sistema de arquivos FAT32 permite que as placas funcionem como um gravador, do inglês *data logger*. Com essa funcionalidade é possível armazenar até 4 Gbytes de informação em um único arquivo e 256 Gbytes de informação total. Isso permite que sejam coletadas milhares de informações para posterior análise. A implementação do sistema de arquivos FAT32 foi desenvolvida para utilizar cartões de memória *micro-sd card* – os cartões de memória utilizados em aparelhos celulares modernos. Estes cartões têm dois modelos, os *Standard* e os *SDHC* (Secure Digital High Capacity). Os cartões *standard*, ou padrão, não armazenam mais que 4 Gbytes de informação total. Inicialmente, estes cartões eram produzidos para armazenar 128 Mbytes e, com o avanço da tecnologia e necessidade de maiores velocidades, o padrão *SDHC* gradualmente vem substituindo os cartões *standard*. As duas implementações testadas do sistema FAT32 operam com esses cartões. Em relação a funcionalidades disponíveis, a implementação FatFS é mais completa e a de Roland-Riegel é mais otimizada. A utilização de uma implementação ou outra depende realmente da aplicação e espaço disponível para colocar o software.

No teste realizado, que foi gravar um arquivo com 16 Mbytes e tentar abrir no PC com cartões de 128 Mbytes e 2 Gbytes, as duas implementações foram bem sucedidas. No entanto, houve uma grande dificuldade inicial para colocar as implementações para funcionar. Descobriu-se, finalmente, que o cartão de memória utilizado já estava formatado, mas precisava ser novamente formatado com o Windows, o que impedia o correto funcionamento do sistema FAT32. O problema encontrado foi que a implementação dos sistemas FAT utilizadas apenas trabalham com setor de disco no tamanho de 512 bytes e o cartão estava formatado com 4096 bytes por setor. Após a formatação com a opção de 512 bytes por setor o sistema funcionou corretamemte. Nos teste a média de gravação se manteve por volta de 25 kbytes/s, o que é mais do que suficiente para as aplicações de RSSF.

4.4 CONSIDERAÇÕES DO CAPÍTULO

Este capítulo descreveu o projeto de hardware e do software básico para os nodos utilizados para o desenvolvimento e avaliação das abordagens propostas neste trabalho. A fase de levantamento de requisitos e projeto de hardware foi importante para o desenvolvimento das abordagens de software que serão descritas no próximo capítulo. Além de todo o projeto do hardware, foi estudado um modelo de interferências para compreender melhor como o software, com ajuda do hardware, deve ser protegido. Nesse sentido, o pró-

prio hardware desenvolvido, através da configuração de seu rádio, foi empregado como um gerador de interferências. Para se verificar os efeitos das interferências no meio junto aos quadros de enlace, um *sniffer* foi viabilizado através da integração do Wireshark com a plataforma de software básico desenvolvida. Com a utilização de um gravador como *data logging* foi possível desenvolver um sistema que pode armazenar seus dados e, quando possível, economizar energia pela ativação do cartão apenas quando necessário, com uso de alimentação controlada.

5 COMUNICAÇÃO DINÂMICA PARA FUSÃO DE DADOS

5.1 INTRODUÇÃO

As aplicações executadas em RSSF devem lidar com as restrições de recursos computacionais dos nodos e também com a capacidade de energia limitada (assumindo nodos alimentados por bateria). Neste trabalho é proposta uma abordagem de comunicação baseada na arquitetura mestre e escravo. O principal objetivo é realizar a fusão de dado, em um sistema onde o nodo coordenador (MN) implementa o centro de fusão e há N nodos sensores (SN) que periodicamente transmitem os dados coletados. Neste capítulo os termos Coordenador e Nodo Mestre (MN) são utilizados de forma intercambiável.

Nesta abordagem, o coordenador sinaliza para os SN quando ele deseja informação de todos os nodos. Isso significa que os SN vão competir, entre si, pelo acesso ao meio para enviar seus dados para o coordenador utilizando o algoritmo CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) do IEEE 802.15.4. Durante a sinalização, o coordenador repassa duas informações: $TargetQoF$ que é o número de mensagens necessárias para se realizar uma fusão de dados com confiança e F, que é o número de mensagens que o coordenador recebeu na sinalização anterior. Os SN, de posse desta informação, executam de forma descentralizada um algoritmo que ajusta a probabilidade individual, que se denominou probabilidade local (*LOCAL_PROB*), a qual representa uma parcela de participação de cada SN na fusão de dados. A Figura 19 esboça graficamente como é esse ciclo de iteração entre o coordenador e os nodos.

Na Figura 20 é ilustrado o modelo proposto para uma rede IEEE 802.15.4 com modo *beacon* ativado. Um *SN* decide dinamicamente se vai ou não coletar e transmitir os dados em cada período de *beacon* (*round*) – de acordo com a sua probabilidade local. A mensagem é transmitida para o *MN*, durante o Período de Contenção de Acesso (*CAP*), é executando o algoritmo *slotted CSMA-CA* para se obter o acesso ao meio. O *MN* tem o papel de centro de fusão.

A mensagem de *beacon* é utilizada para repassar informações aos *SN* e também tem a função de sincronização. O *BI* também corresponde à periodicidade da tarefa de fusão de dados. Em cada *BI*, um *SN* pode decidir transmitir nenhuma ou apenas uma mensagem ao *MN*. Cada mensagem tem um *deadline* absoluto D. Se o *deadline* não for cumprido, a informação não será mais útil para a tarefa de fusão de dados no *MN* (i.e. *firm deadline*). Este *deadline* absoluto é o mesmo para todos os *SN* e corresponde ao período de

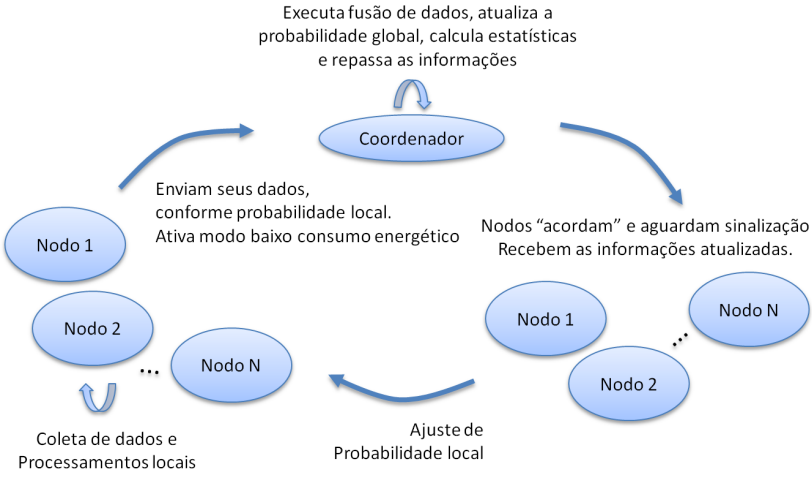


Figura 19: Ciclo do modelo de execução.

início do próximo intervalo de *beacon* ($D_i = B_i$).

O *MN* calcula métricas de execução antes de enviar a mensagem de *beacon*, para realizar um ajuste no processo de fusão. No modelo proposto, a principal métrica global considerada é a Qualidade da Fusão (QoF). A idéia básica da métrica QoF é representar a qualidade da informação da fusão de dados. Um número grande de mensagens para a execução da tarefa de fusão de dados resulta numa informação mais confiável, ou seja, a QoF é uma métrica relacionada com o número de mensagens recebidas. Contudo, o consumo energético da rede é muito maior. Como exemplo, usando a Figura 9 como referência, no tempo $t = 4$ o número de mensagens recebidas resulta na estimação da QoF igual a 3.

No restante deste capítulo apresentam-se em detalhes o modelo e os algoritmos propostos.

5.2 ABORDAGEM PROPOSTA

5.2.1 Considerações Iniciais

Neste trabalho considera-se um sistema homogêneo, onde todos os sensores medem o mesmo fenômeno físico e existe redundância espacial

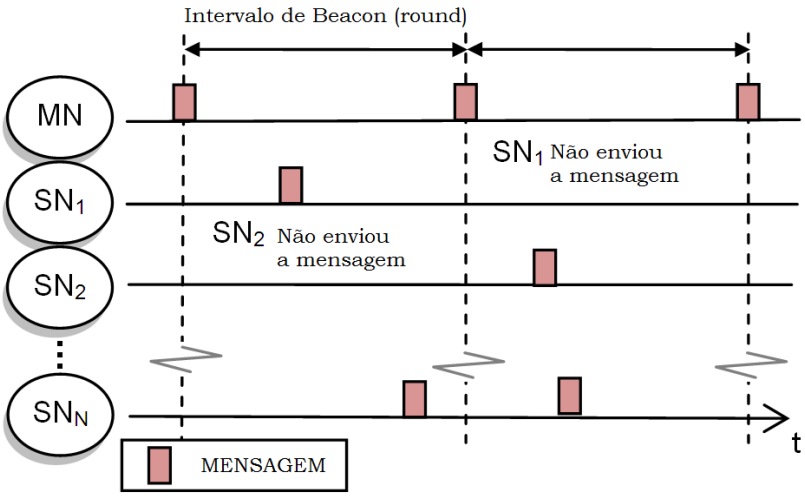


Figura 20: Exemplo do modelo proposto.

no número de sensores (PATIL; DAS, 2004; CHIUSO; SCHENATO, 2011). Neste contexto, não é necessário que sejam transmitidas todas as mensagens ao coordenador em cada período de comunicação (*round*).

Adicionalmente foram considerados os seguintes pressupostos para a rede e aplicação:

A1. A rede é composta por vários nodos sensores, que periodicamente podem enviar informações para o coordenador;

A2. Os dados transmitidos pelos nodos sensores tem um *deadline*;

A3. Os sensores não são confiáveis e os dados monitorados têm um certo grau de imprecisão;

A4. Em cada período de comunicação (*round*), o coordenador necessita receber, em média, M mensagens para cumprir o nível de confiança do fenômeno monitorado;

A5. A rede é formada por N nodos sensores, tal que $N >> M$;

A6. O número de sensores pode variar ao longo do tempo;

Os presupostos A1, A2 e A3 são comuns em várias aplicações de RSSF, tais como a monitoração de objeto móvel ou rastreamento de intruso

– onde os dados coletados perdem a validade se não forem entregues no *deadline* especificado. No modelo de tarefas adotado para os nodos sensores, este tempo limite pode ser representado como um *deadline* relativo (*firm deadline*) cujo valor coincide com o período de monitoramento, ou seja, o *deadline* é igual ao período entre *beacons*.

A justificativa para o afirmação A4 vem da definição adotada de fusão de dados. Seguindo esta definição, o processo de fusão de dados é considerado como o responsável por combinar dados de diferentes sensores com o objetivo de alcançar uma melhor precisão e compreensão do fenômeno observado, em relação ao uso de apenas um sensor (CHIUSO; SCHENATO, 2011).

O pressuposto A5 foi elaborado pelo fato de que, normalmente, as RSSF são implantadas com um grande número de nodos e existe uma redundância espacial entre eles. A afirmação A6 depende da aplicação e também pode ocorrer se os nodos são móveis (por exemplo, nós sensores instalados em robôs móveis ou nós deslizantes implantados em montanha), um ruído aleatório no ambiente (por exemplo máquinas de solda em ambientes industriais ou interferência de outras redes), os obstáculos temporários que se destacam entre os nós (por exemplo, um veículo automatizado guiado em chão de fábrica), ou simplesmente devido ao esgotamento da bateria de nós.

O trabalho leva em consideração os pressupostos anteriormente citados e assume o mesmo modelo base proposto em (PINTO; MONTEZ, 2010), onde o sistema é semelhante a uma arquitetura mestre e escravo. Existe, então, um coordenador PAN (*MN*) e *N* nodos sensores (*SN*) que periodicamente transmitem os dados coletados. No sistema mestre escravo, o mestre tem o papel de buscar, ou requisitar, a informação em apenas um escravo por vez, caracterizando um sistema cíclico com execução bem definida.

Também assume-se que de acordo com o valor da QoF e o intervalo de confiança calculado no centro de fusão é possível ajustar o número de mensagens ($TargetQoF$) que devem alcançar o centro de fusão. Para tal, é proposto um algoritmo que divide dinamicamente a probabilidade de envio em cada nodo sensor (SN).

Cada *SN* toma a decisão de transmitir de forma descentralizada e leva em conta um parâmetro de probabilidade de envio, definida neste trabalho como a probabilidade local, que guia o número de mensagens enviadas por cada *SN* em cada *round*. Como o sinal é considerado homogêneo na área de monitoramento, uma probabilidade de envio bem configurada economiza energia na rede, reduzindo o número de mensagens na RSSF. Como um exemplo, no primeiro *round* mostrado na Figura 20, o *SN* 2 decide, com base no histórico e probabilidade local, não coletar os dados e não transmitir informação. A execução desse algoritmo ocorre, na sua maior parte, de forma

distribuída, nos nodos que realizam a coleta de dados. A parte restante do algoritmo é executada pelo nodo coordenador. O algoritmo deve, então, se adaptar em cada nodo de tal forma que o número de mensagens que alcança o centro de fusão, na média, se mantenha acima de uma meta indicada pelo nodo coordenador ($TargetQoF$).

Nesta dissertação foram desenvolvidas duas versões de algoritmos para avaliar a proposta. A primeira versão ignora os valores da fusão dos dados e envia um valor fixo para os SN. Na segunda versão o coordenador processa as informações recebidas dos SN e, com base nestes valores e aplicando técnicas estatísticas, são gerados novos valores de referências, em tempo de execução, para serem enviados aos nodos.

Nas próximas seções apresenta-se a abordagem proposta para os *SN* e para *MN*.

5.3 ALGORITMO *BUFFER PROBABILÍSTICO* (NODOS)

O Algoritmo *Buffer Probabilístico* será descrito de acordo com as partes distribuídas (algoritmo dos *SN*) e a parte que cabe ao coordenador. O algoritmo completo dos nodos pode ser visualizado em Algoritmo 1. Este algoritmo será dividido nas seguintes partes funcionais para facilitar a compreensão: *flood_buffer*, *sort_rand*, *check_last_frame* e *buffer_adjust_probability*. O Algoritmo 6 descreve de forma completa as funcionalidades do coordenador. Ambos os algoritmos são descritos no decorrer deste capítulo.

A primeira parte do algoritmo (linhas 1–5) está relacionada com o processo de inicialização, onde é definido o valor da probabilidade local na variável *LOCAL_PROB*. Essa probabilidade é então espalhada em um buffer de probabilidades, onde cada posição do buffer representa um *round* e o valor 1 indica que a mensagem deve ser enviada, e o valor 0 caso não haja envio. O valor inicial da probabilidade local é dado pela energia da bateria. No Algoritmo, o buffer é representado por $buffer[COUNT]$ onde $COUNT$ marca a posição atual do *round*.

A equação da bateria, Equação 5.1, foi obtida em testes experimentais, levando em conta a faixa de trabalho dos nodos quando alimentados por duas baterias: de 1800 até $3000mV$, onde MIN_BAT é o menor valor de funcionamento dos nodos, no caso $1.8V$. A Equação consiste em pegar o valor de tensão da bateria, dividi-lo por 1000, e recuperar o valor normalizado em uma escala de 0 a 100%. O valor retornado é um número real no intervalo $[0-1]$. Nestes testes, foi observado que quando a voltagem fica abaixo de $2.0V$, o nodo para de transmitir rapidamente; parando totalmente quando a voltagem fica abaixo de $1.8V$. Portanto, a equação busca distribuir uniformemente o

valor da probabilidade local em um buffer circular. Ao iniciar o sistema, a probabilidade local é igual à probabilidade da bateria e é associada ao buffer de probabilidades pela execução do método *flood_buffer*. A Figura 21 mostra que os valores 1 e 0 representam, respectivamente, que o SN deve tentar transmitir ou permanecer em modo de economia de energia (sleep) em cada *round* de comunicação.

$$battery\_probability \leftarrow min\left(1.0, max\left(0, \left(\frac{battery\_voltage}{1000}\right) - MIN\_BAT\right)\right) \tag{5.1}$$

Após o período de inicialização, o algoritmo aguarda pelo início de cada *round* que é determinado pelo *MN* ao enviar a mensagem de *beacon*. Nesta mensagem são repassados: o número de quadros (F) recebidos pelo

Algoritmo 1 : probability_buffer (executa em cada nodo sensor).

```
1: init network
2: LOCAL_PROB ← max(0, (battery_voltage/1000) − MIN_BAT)
3: LOCAL_PROB ← min(1.0, LOCAL_PROB)
4: flood_buffer(LOCAL_PROB)
5: COUNT ← 0
6: while true do
7:    wait next Beacon Message
8:    if network is active then
9:       (TargetQoF, F) ← beacon_payload
10:      check_last_frame()
11:      if F > TargetQoF and BUFFER_PROB > LOCAL_PROB then
12:         sort_rand(0)
13:      else
14:         sort_rand(1)
15:      end if
16:      if buffer[COUNT] = 1 then
17:         data ← sensor_data_aquisition()
18:         nwk_send(data)
19:      end if
20:      COUNT = COUNT + 1
21:      if COUNT mod LOCAL_PROB_ADJ == 0 then
22:         adjust_local_probability()
23:      end if
24:      COUNT ← COUNT mod BUFFER_LEN
25:   end if
26: end while
```

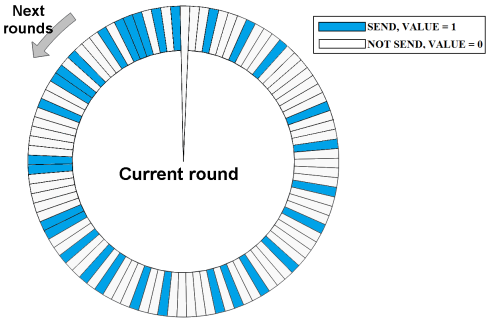


Figura 21: Buffer circular, cada entrada representa um *round*.

MN no *round* anterior e o número de mensagens que o *MN* deseja receber ($TargetQoF$). Estes valores são atualizados por todos os SN (ver linha 9).

O algoritmo é composto por mais 3 métodos principais: *check_last_frame*, *sort_rand* e *adjust_local_probability*. O método *check_last_frame* (linha 10) ajusta a probabilidade no buffer. Ele é responsável por verificar se o número mínimo de mensagens está sendo alcançado para o *round* de comunicação anterior. Quando isso não acontece, é necessário um meio de alavancar rapidamente a probabilidade do buffer. O método executa testes para saber se deve aumentar a probabilidade no buffer comparando $TargetQoF$, F e a posição anterior do buffer ($COUNT - 1$). Na sequência, independentemente da atuação de *check_last_frame*, o algoritmo avalia se deve diminuir ou aumentar a probabilidade no buffer (linha 11) verificando se o número de *frames* está dentro do esperado e se a probabilidade no buffer é maior que a probabilidade local. Neste momento, o algoritmo faz um ajuste, aumentando ou diminuindo a probabilidade no buffer (linhas 11 – 15) por meio da execução do método *sort_rand*.

Quando o método *sort_rand(0)* é executado, por exemplo, no momento que o nodo descobre que deve economizar energia (i.e. $F > TargetQoF$ e $buffer\_probability() > LOCAL\_PROB$ (linha 11)) uma posição no buffer é selecionada aleatoriamente e se o valor da posição tiver $valor = 1$, o $valor = 0$ é inserido, caso contrário um valor é buscado sequencialmente até encontrar uma posição com $valor = 1$ para alterá-la para $valor = 0$ (Figura 22). É importante notar que se $F <= TargetQoF$ a probabilidade associada com o buffer deve aumentar, independentemente da probabilidade local e da energia da bateria (linha 11).

A variável $BUFFER\_PROB$ armazena a probabilidade de envio que

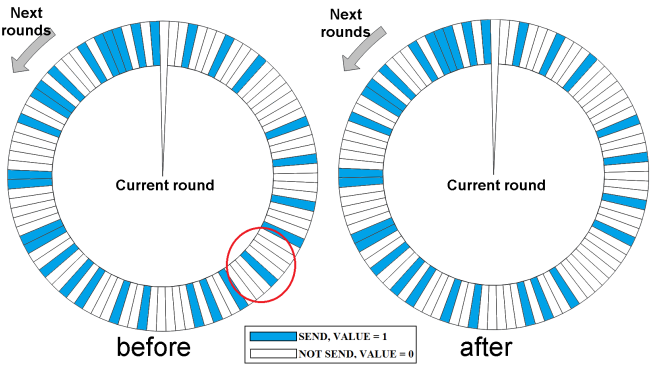


Figura 22: Exemplo de execução do método *sort_rand(0)*.

está associada ao buffer (i.e. a razão das entradas com $valor = 1$ em relação ao tamanho do buffer). As variáveis $BUFFER\_PROB$ e *LOCAL_PROB* são utilizadas para determinar a geração de curvas de convergência com base na diferença de potencial (voltagem) de cada conjunto de baterias. As curvas de convergência são diferentes em cada nodo pois eles dificilmente apresentam a mesma voltagem de bateria e raramente são ligados ao mesmo tempo. Assim, quando existe a execução do método *adjust_local_probability*, em algum nodo, a probabilidade local é alterada instantaneamente e passa a ser a nova referência para aquele nodo por *LOCAL_PROB_ADJ rounds*. Durante este tempo, a probabilidade do buffer converge para a probabilidade local a cada *round* até que $BUFFER\_PROB$ seja equivalente a *LOCAL_PROB*; neste momento $BUFFER\_PROB$ fica oscilando ao redor de *LOCAL_PROB*.

O processamento local dos sensores e o envio dos dados são determinados pelo valor da posição atual do buffer (linhas 16–19). Assim, somente se $buffer[COUNT]$ for igual a 1, o processamento local seguido do envio dos dados é realizado.

Na parte final, o algoritmo verifica se deve realizar um ajuste na probabilidade local (linhas 20–24), onde foi adotado um conceito de sessão. Uma sessão consiste no intervalo de tempo determinado por *LOCAL_PROB_ADJ rounds* de comprimento *BI*. O ajuste da probabilidade local $adjust\_local\_probability()$ é executado sempre que o resto da divisão de $(COUNT/LOCAL\_PROB\_ADJ)$ for zero. Em seguida a variável $(COUNT)$ é normalizada em relação ao tamanho do buffer $(BUFFER\_LEN)$.

A seguir, descrevem-se as principais partes funcionais do Algoritmo 1.

5.3.1 *flood_buffer*

O método *flood_buffer* é utilizado para inicializar o *buffer* de probabilidades (Algoritmo 1 – linha 4). Esse método tem como missão distribuir uniformemente um valor de probabilidade associado à tensão da bateria no *buffer* de probabilidades. A distribuição é realizada com base na tensão da bateria.

Algoritmo 2 : flood_buffer

```
 1: count ← 0
 2: while count < BUFFER_LENGTH do
 3:     buffer[count] ← 1
 4:     count ← count + 1
 5: end while
 6: count ← 0
 7: BUFFER_PROB ← 0.0
 8: buffer_count ← 0
 9: while count < BUFFER_LENGTH do
10:     if BUFFER_PROB ≤ LOCAL_PROB then
11:         count ← count + 1
12:         buffer_count ← buffer_count + 1
13:         BUFFER_PROB ← buffer_count/count
14:     else
15:         sort_rand(0)
16:     end if
17: end while
```

O primeiro passo do método *flood_buffer* consiste em preencher todo o buffer de probabilidades com $valor = 1$. Isso é feito para ter um estado inicial consistente antes de começar a distribuir a probabilidade. Em seguida, o método busca descobrir o número de posições no buffer que representam o valor da probabilidade local *LOCAL_PROB* (linhas 10–13). Após esse passo, o método distribui o restante da probabilidade no buffer, executando $sort\_rand(0)$, enquanto a variável *count* for menor que o tamanho do buffer ($BUFFER\_LENGTH$).

5.3.2 *sort_rand*

O método *sort rand* é responsável por inserir um valor 0 ou 1 em posição aleatória no *buffer* de probabilidades (Algoritmo 1 – linhas 12 e 14). O método gera um índice aleatório e verifica se a posição no *buffer* é diferente

da passada como parâmetro. Em caso dessa condição não ocorrer, o método busca de forma sequencial, a partir do índice aleatório, a primeira posição com valor válido. Quando encontrada uma posição válida, o valor passado como parâmetro é gravado e a probabilidade do buffer é recalculada. No caso de não existir uma posição com o valor desejado, o *buffer* é mantido em seu estado atual. O método *sort rand* é apresentado no Algoritmo 3.

Algoritmo 3 : sort_rand

```
Require: value{0,1}
 1: lrand ← (rand() mod BUFFER_LENGTH − 2) + 1
 2: count ← 0
 3: while count < BUFFER_LENGTH do
 4:    if buffer[lrand] <> value then
 5:       buffer[lrand] ← value
 6:       if value then
 7:          buffer_count ← buffer_count + 1
 8:       else
 9:          buffer_count ← buffer_count − 1
10:       end if
11:       BUFFER_PROB ← buffer_count/BUFFER_LENGTH
12:       return
13:    else
14:       lrand ← lrand + 1
15:       lrand ← lrand mod BUFFER_LENGTH
16:    end if
17:    count ← count + 1
18: end while
```

O primeiro passo do método é determinar o valor do índice aleatório *lrand* para acessar o buffer e inicializar a variável *count*, que determina uma busca completa no buffer. Em seguida, o método começa a verificar se a posição indexada por *lrand* possui um valor contrário ao parâmetro passado (linha 4). Se a condição for verdadeira, o método finaliza alterando o valor na posição indexada por *lrand* e recalculando a probabilidade do buffer (linha 11). Caso contrário, o método gera um novo índice sequencialmente (linhas 14–15) enquanto a variável *count* não ultrapassar o tamanho total do buffer (linha 3). Caso uma posição não for encontrada, isso significa que o buffer já se encontra todo preenchido com o valor passado como parâmetro.

5.3.3 *check_old_frame*

Em vários momentos o número de mensagens recebidas pelo coordenador (F) pode ser menor que o desejável (*TargetQoF*) devido aos ajustes das probabilidades locais nos nodos, devido a menor energia na bateria de alguns nodos ou devido a ajustes de redução de energia. Quando isso ocorre é necessário um recurso para alavancar as probabilidades locais de forma rápida a fim de manter a *QoF* do sistema. O método *check_old_frame* tem esse propósito. Os passos executados podem ser visualizados no Algoritmo 4.

Algoritmo 4 : check_old_frame.

```
 1: if F < TargetQoF then
 2:     old_count ← (count − 1) mod BUFFER_LENGTH
 3:     if buffer[old_count] = 0 then
 4:         buffer[old_count] ← 1
 5:         if count_buffer < BUFFER_LENGTH then
 6:             count_buffer ← count_buffer + 1
 7:             BUFFER_PROB ← buffer_count/BUFFER_LENGTH
 8:         else
 9:             count_buffer ← BUFFER_LENGTH
10:             BUFFER_PROB ← 1.0
11:         end if
12:     end if
13: end if
14: if F == TargetQoF then
15:     old_count ← (count − 1) mod BUFFER_LENGTH
16:     if buffer[old_count] = 0 then
17:         if (genrand_int32() mod 2) = 0 then
18:             buffer[old_count] ← 1
19:             if count_buffer < BUFFER_LENGTH then
20:                 count_buffer ← count_buffer + 1
21:                 BUFFER_PROB ← buffer_count/BUFFER_LENGTH
22:             else
23:                 count_buffer ← BUFFER_LENGTH
24:                 BUFFER_PROB ← 1.0
25:             end if
26:         end if
27:     end if
28: end if
```

O funcionamento do *check old frame* consiste em verificar se o número de dispositivos mínimo para a fusão foi alcançado no ciclo anterior. Isso é feito comparando os valores de F e $TargetQoF$ recebidos no quadro

de *beacon*. Quando o valor de F for inferior a $TargetQoF$, o método aumenta a probabilidade inserindo $valor = 1$ na posição anterior do *buffer* de probabilidades.

Em um segundo momento, o método faz outra verificação para decidir se aumenta a probabilidade do buffer quando F e $TargetQoF$ são iguais. Esta é uma decisão a qual torna o nodo colaborativo, antecipando uma redução do valor de F e obrigando a $TargetQOF$ a subir. Para tal, o método verifica se a posição anterior do buffer apresenta $valor = 0$. Se sim, um número aleatório é gerado e, se for par, a probabilidade é aumentada inserindo $valor = 1$ na posição anterior do *buffer* de probabilidades.

5.3.4 *buffer_adjust_probability*

O valor da probabilidade local (*LOCAL_PROB*) é uma função da tensão da bateria e a quantidade de mensagens ($TargetQoF$) que devem ser enviadas para o coordenador a fim de estimar a QoF. Este ajuste ocorre a cada *LOCAL_PROB_ADJ rounds* (linha 2). A probabilidade da bateria é recuperada na variável *aux*, a fim de se obter o valor de correção na probabilidade local (linha 3).

Algoritmo 5 : Método **buffer_adjust_probability**.

```
1:  COUNT = COUNT + 1
2:  if (COUNT mod LOCAL_PROB_ADJ then
3:      aux ← min(1.0, max(0, (battery_voltage/1000) − MIN_BAT))
4:      if F > TargetQoF then
5:          if genrand_int32() is even then
6:              LOCAL_PROB ← LOCAL_PROB − ((1.0 − aux) ∗ 5%)
7:          end if
8:      else
9:          LOCAL_PROB ← min(1.0, LOCAL_PROB + (aux ∗ 5%))
10:     end if
11: end if
12: COUNT ← COUNT mod BUFFER_LENGTH
```

O valor do ajuste da probabilidade local foi estipulado para ser no máximo $\pm 5\%$ relativo à energia da bateria. O valor de 5% foi determinado a partir de testes realizados. Isso significa que quando a bateria é nova, o dispositivo pode contribuir mais, aumentando rapidamente sua probabilidade. Assim, quando a energia da bateria está por volta de 50% o valor de ajuste da probabilidade não deve passar de $\pm 2.5\%$.

Em seguida, é verificado se o sistema está precisando de mais ou me-

nos contribuições (linha 4). Se o valor da probabilidade local pode descer e o nodo passar a economizar mais energia, o nodo executa um teste (linha 5) que determina se ele vai contribuir menos com o sistema ou continuar igual. Se o nodo optar por economizar energia, ele executa o ajuste da linha 6. Caso contrário, o ajuste da linha 9 é executado. A fórmula de ajuste é diferente para incremento e decremento, fazendo com que dispositivos com mais energia na bateria contribuam mais.

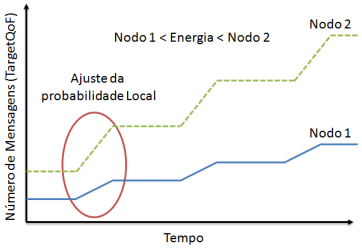


Figura 23: Aumento da probabilidade Local

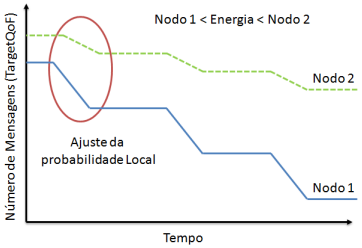


Figura 24: Diminuição da probabilidade Local

O resultado da execução do algoritmo *buffer adjust probability* gera um comportamento para economia de energia e uma obrigação de envio de mensagens. As figuras 23 e 24 ilustram o comportamento ao longo do tempo de dois nodos. Para analisar o gráfico, deve-se levar em consideração que o dispositivo 1 tem menos energia que o dispositivo 2 e que as retas representam as probabilidades de cada nodo. Com o passar do tempo pode-se observar que existem duas configurações: no gráfico da Figura 23 os dispositivos aumentam a taxa de transmissão pois a quantidade de mensagens recebidas pelo coordenador está baixa e os dispositivos devem aumentar a sua colaboração com o sistema. No gráfico da Figura 24 os dispositivos começam a enviar menos (economizando energia). Na primeira configuração, o dispositivo 2 aumenta sua probabilidade muito mais que o dispositivo 1 (e consequentemente a transmissão de pacotes) por ter mais energia na bateria. Na segunda configuração, o dispositivo 2 apresenta uma redução menos acentuada em relação ao dispositivo 1. Isso acontece por ele possuir mais energia, forçando-o a contribuir mais com o sistema.

Essa geração de curvas de probabilidade geram grupos de dispositivos. Quando os dispositivos têm a mesma energia da bateria eles tendem a se dividir em grupos onde uns colaboram mais que os outros. Contudo, com o passar do tempo, os nodos acabam migrando de um grupo para o outro, o que é um comportamento desejável, uma vez que o consumo de energia é redistribuída entre os nodos.

5.4 ALGORITMO *CENTRO DE FUSÃO*

O nodo mestre (MN) ou coordenador é responsável pelo recebimento dos dados dos SN, onde cada valor individual é "fundido" a outros, gerando informações sobre o fenômeno gerado. Portanto, este nodo também tem a função de um Centro de Fusão e é responsável por enviar referências aos SN a cada intervalo de *beacon*.

O nodo coordenador tem um *timer* de sistema utilizado para sincronizar as operações de fusão de dados e, consequentemente, alterar o conteúdo do *beacon payload* com as informações a serem repassadas aos nodos. No momento em que a contagem do *timer* expirar, o algoritmo de fusão é executado, realizando caculos estatísticos e atualizando o *beacon payload*. Estas informações são repassadas para os nodos no próximo *beacon*. Para executar essas funcionalidades implementou-se no coordenador os Algoritmos 6–9 que são comentados a seguir.

O Algoritmo 6 inicia as variáveis locais e realiza o processamento, que consiste em contar o número de *frames* e os valores monitorados recebidos (neste trabalho considerou-se testes relacionados com temperatura), ver linhas 9 – 10. Em seguida obtém-se a média das temperaturas (linha 15) se o MN recebeu algum *frame*. O próximo passo é realizar o cálculo da variância e marcar os quadros recebidos como já processados (linhas 20 – 21).

Uma vez que a variância foi determinada o valor da $TargetQoF$ pode ser calculado. No Algoritmo o valor de $TargetQoF$ é representado pela variável *target_nodes*. O cálculo é representado como uma equação de filtro dividida em três partes: na primeira parte é calculado um valor base que considera 50% do valor anterior da $TargetQoF$. Na segunda parte é realizado o cálculo da quantidade de nodos que devem enviar dados, de acordo com a variância atual, o intervalo de confiança desejável e com a tolerância de erro especificada. Este valor é obtido pela Equação 5.2 abaixo:

$$target\_nodes = \left(\frac{Z * Sx}{E} \right)^2 \tag{5.2}$$

onde Z representa o intervalo de confiança, Sx o desvio padrão e E é a estimativa do erro. Para um intervalo de confiança de 95% o valor de Z é igual a $1,96$. Em seguida observa-se que a função teto é utilizada. O objetivo é retornar o próximo número inteiro, que representa a quantidade de mensagens desejável pelo MN no próximo round (linha 29).

A temperatura de fusão é cálculada com um filtro de 50% da temperatura atual somado a 50% da temperatura existente da fusão anterior (linha 31). Se não for recebido nenhum *frame*, a temperatura da fusão permanece

Algoritmo 6 : Fusion_center (coordenador).

```
 1: index ← 0
 2: actual_temp ← 0.0
 3: actual_nodes ← 0.0
 4: frames_received ← 0
 5: variancia ← 0.0
 6: desvio_padrao ← 0.0
 7: for index < MAX_NUMBER_OF_DEVICES do
 8:     if device_list[index].framerec > device_list[index].frameproc then
 9:         frames_received ← frames_received + 1
10:         actual_temp ← actual_temp + device_list[index].temperature
11:     end if
12:     index ← index + 1
13: end for
14: if frames_received > 0 then
15:     actual_temp ← actual_temp/frames_received
16: end if
17: index ← 0
18: for index < MAX_NUMBER_OF_DEVICES do
19:     if device_list[index].framerec > device_list[index].frameproc then
20:         variancia ← variancia + (device_list[index].temperature − actual_temp)^2
21:         device_list[index].frameproc ← device_list[index].frameproc + 1
22:     end if
23:     index ← index + 1
24: end for
25: if frames_received > 0 then
26:     if frames_received > 2 then
27:         variancia ← variancia/(frames_received − 1)
28:         desvio_padrao ← √variancia
29:         target_nodes ← ceil((target_nodes ∗ 0.50) + (1.95 ∗ desvio_padrao ∗ 0.50))
30:     end if
31:     fusion_temp ← (fusion_temp ∗ 0.50) + (actual_temp ∗ 0.50)
32: end if
33: if target_nodes < 2 then
34:     target_nodes ← 2
35: end if
36: if no_of_assoc_devices > 0 then
37:     actual_fusion ← frames_received/no_of_assoc_devices
38: else
39:     actual_fusion ← 0.0
40: end if
41: media_fusion ← (media_fusion ∗ 0.75) + (actual_fusion ∗ 0.25)
42: media_nodes ← (media_nodes ∗ 0.75) + (frames_received ∗ 0.25)
43: cicle_fusion ← cicle_fusion + 1
44: serialize_beacon_payload()
```

com o valor do *round* anterior. Da mesma forma, se o número de *frames* recebidos for apenas um (1), a variância e o desvio padrão são zero e os valores de *target_nodes* permanece o mesmo do *round* anterior (linhas 26 à 30). Isso

acontece por ser necessário pelo menos duas amostras para se obter um desvio padrão, uma verificação é realizada na linha 33 para forçar o sistema a manter um valor mínimo de operação para $TargetQoF$ com $valor = 2$.

Em seguida, o índice da fusão pode ser calculado como o número de *frames* recebidos dividido pelo número mínimo de dispositivos associados na rede (linhas 36–40). O índice médio da fusão para ser enviado aos nodos é calculado como uma função de amortecimento, considerando 75% para o valor anterior e o restante para o calculado no *round* atual. A média de transmissões é calculada da mesma forma (linhas 41 e 42). O indicador de operação $cicle\_fusion$ é incrementado para informar que no próximo *beacon* os nodos devem executar o algoritmo local (linha 43). Por fim, o *beacon payload* é alterado com as novas informações para o próximo ciclo de cálculos.

5.4.1 Estrutura associated_device_t e constantes de estatística

A estrutura de dados *associated_device_t* (Estrutura 7) armazena as informações de endereço de rede e dados dos nodos para processamento no Algoritmo 6. As informações são: endereço de 64 e 16 bit, número de *frames* recebidos e processados, tamanho do *buffer* de probabilidades no nodo, qualidade do sinal, tensão da bateria, temperatura e probabilidade local do nodo. Estas informações são preenchidas durante a recepção dos dados, que é realizado pelo Algoritmo 8.

Estrutura 7 : associated_device_t.

```
 1: structassociated_device_t
 2: {
 3:   uint64_t long_address
 4:   uint16_t short_address
 5:   uint32_t framerec
 6:   uint32_t frameproc
 7:   uint8_t bufferlength
 8:   uint8_t lqi
 9:   double vbat
10:   double temperature
11:   double probability
12: }
```

5.4.2 Algoritmo usr_mcps_data_ind()

O algoritmo *usr_mcps_data_ind* (Algoritmo 8) tem como papel desempenhar o *parsing* de informações do *payload* de dados. O algoritmo busca, na lista de dispositivos conectados, a referência da estrutura de dados e a preenche com as novas informações.

Algoritmo 8 : usr_mcps_data_ind().

Require: $wpan\_addr\_spec\_t * SrcAddrSpec$
Require: $wpan\_addr\_spec\_t * DstAddrSpec$
Require: $uint8\_t\ msduLength$
Require: $uint8\_t * msdu$
Require: $uint8\_t\ mpduLinkQuality$
1: $associated\_device\_t\ adt = find\_device(SrcAddrSpec)$
2: **if** $adt = NULL$ **then**
3: **return**
4: **end if**
5: $adt.framerec \leftarrow adt.framerec + 1$
6: $adt.lqi \leftarrow getLQI(msdu)$
7: $adt.vbat \leftarrow getVBat(msdu)$
8: $adt.temperature \leftarrow getTemperature(msdu)$
9: $adt.probability \leftarrow getProbability(msdu)$
10: $adt.bufferlength \leftarrow getBufferLength(msdu)$

5.4.3 Algoritmo serialize_beacon_payload()

No algoritmo, *serialize_beacon_payload* (Algoritmo 9) é o lugar onde são inseridas as referências para os nodos. As informações são: indicador de operação, número de nodos desejado, *frames* recebidos e média de participação dos nodos. Uma vez adicionadas, é executada uma requisição para alterar o conteúdo do *beacon payload* que fica armazenado na estrutura de parâmetros do *MAC* (*PIB*).

A principal diferença entre as versões do Algoritmo do Coordenador é realizada pela troca do valor $target\_nodes$, calculado por um valor fixo de referência. Com essa alteração os nodos passam a receber uma referência fixa e é possível testar o comportamento do algoritmo distribuído de forma controlada.

Algoritmo 9 : serialize_beacon_payload().

1: $beacon\_payload \leftarrow \oslash$
2: $beacon\_payload \leftarrow cicle\_fusion$
3: $beacon\_payload \leftarrow target\_nodes(TargetQoF)$
4: $beacon\_payload \leftarrow frames\_received(F)$
5: $wpan\_mlme\_set\_req(macBeaconPayload, beacon\_payload)$

5.5 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram apresentados o modelo e os algoritmos propostos para melhorar a utilização da energia em RSSF com topologia estrela. Um algoritmo distribuído para ajuste de consumo de energia foi proposto em conjunto com um algoritmo de fusão de dados. O objetivo global foi propor uma abordagem de comunicação para RSSF com economia de energia nos nodos sem degradar a eficiência do sistema. Foram detalhados os algoritmos dos SN e do MN. No próximo capítulo são apresentados os testes realizados e os resultados experimentais do modelo proposto.

6 RESULTADOS

6.1 INTRODUÇÃO

Neste capítulo são apresentados os resultados dos experimentos realizados para o modelo proposto no capítulo anterior resultantes dos algoritmos propostos executando no hardware elaborado. Inicialmente será apresentado o ambiente dos ensaios e os materiais utilizados. Em seguida, serão apresentados os resultados dos testes com *QoF* fixa e variável e o capítulo é fechado com algumas considerações.

6.2 AMBIENTE EXPERIMENTAL

Com a finalidade de avaliar o comportamento da abordagem proposta, diversos experimentos foram realizados. Em todos os cenários utilizou-se os kits ATxMEGAv2 cujo hardware está equipado com um rádio AT86RF212 configurado para operar na faixa de 900MHz, seguindo o padrão IEEE 802.15.4. A pilha de protocolos de rede utilizada foi a versão 2.7.1 da ATMEL. O hardware está também equipado com um relógio de tempo real e porta de comunicação serial via USB-CDC. O kit também suporta uso de cartões uSD para serem utilizados como *datalogger* e fonte DC/DC para maior otimização do uso das baterias.

O algoritmo proposto está implementado diretamente sobre o MAC, evitando a necessidade de uso de um sistema operacional mais complexo, por exemplo *TinyOS*[1] ou *freeRTOS*[2]. Como um dos objetivos do estudo era investigar o comportamento do tempo de duração das baterias nos nodos SN, a fonte de alimentação DC/DC não foi ativada e os nodos trabalham no modo *passthru*. Os demais componentes de hardware, como relógio de tempo real, foram deixados inalterados e configurados conforme os valores padrão após o *reset* do hardware.

Com o objetivo de economizar energia, a opção *Battery Life Extension* do MAC 802.15.4 foi ativada nos SN e, durante a espera do *beacon*, o micro-controlador se encontra em *sleep*. Algumas otimizações a nível de software também foram realizadas. Por exemplo, a função *rand()* da *stdlib* foi substituída pela *Mersenne twister* de *Takuji Nishimura and Makoto Matsumoto*(NISHIMURA; MATSUMOTO, 2010). Esta função é baseada em uma

[1] `www.tinyos.net`
[2] `http://www.freertos.org`

matriz de recorrência sobre um campo binário finito F_2. Ela fornece a geração rápida de números pseudo-aleatórios de alta qualidade, tendo sido projetada especificamente para corrigir defeitos usuais em algoritmos convencionais. Seu nome deriva do fato de que o comprimento do período é escolhido para ser um número primo de *Mersenne*.

A rede foi configurada para ser utilizada em modo com *beacon* ativado, onde os pacotes na rede são sinalizados com *acknowledged*, e o valor do *Beacon Order (BO)* é $= 6$, e *Superframe Order (SO)* $= 5$ (portanto, BI corresponde a aproximadamente 1s). Para os testes foram utilizados 10 dispositivos (SN), os quais, foram dispostos fisicamente conforme apresentado na Figura 25, em um ambiente fechado. O coordenador (MN) está localizado em outra sala separados por uma parede de alvenaria. O software utilizado para gerar os arquivos de informações para posterior análise foi o *RS232 Data Logger* da *Eltima Software*, que fica instalado em um PC, onde o MN se encontra conectado.

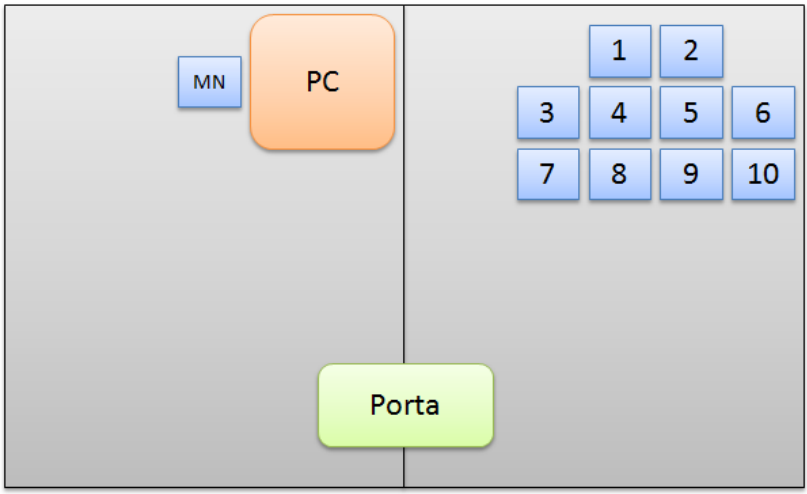


Figura 25: Disposição dos nodos para os experimentos.

Os experimentos foram conduzidos em dois cenários distintos: o primeiro cenário teve como principal objetivo validar o algoritmo nos nodos e verificar o comportamento do algoritmo distribuído. No segundo cenário, os experimentos foram conduzidos para validação do algoritmo de fusão de dados e controle do número desejável de mensagens ($TargetQoF$) que devem atingir o nodo coordenador. Em seguida, foram conduzidos expe-

rimentos para avaliar o algoritmo de fusão de dados, controlando a economia de energia e com o valor de *TargetQoF* variável. O período da sessão (*LOCAL_PROB_ADJ*) e o tamanho do buffer de probabilidades foram configurados para o tamanho de 100 *rounds*.

6.2.1 Cenário 1 - *TargetQoF* fixa

Neste cenário, cinco experimentos foram conduzidos. Para todos os experimentos, a configuração da rede consistiu de 10 SN conectados ao MN. O valor da *TargetQoF* foi fixado com valor igual a 2 em cada *round*.

O primeiro experimento teve como objetivo verificar a capacidade do algoritmo de se auto ajustar às condições dinâmicas da rede. O segundo experimento avalia o número de transmissões *versus* a voltagem da bateria dos SN. Com isso foi criado um gráfico de dispersão conforme apresentado na Figura 27. No terceiro experimento, foi avaliada a capacidade do algoritmo se auto adaptar às condições dinâmicas da rede (Figura 28). Neste caso, alguns SN foram configurados para entrar na rede e sair da rede em tempos específicos.

Os outros dois últimos experimentos foram realizados para examinar o comportamento individual dos nodos (Figuras 29 e 30). Nos resultados, para cada nodo foi apresentado o número de transmissões ao longo do tempo. Para melhorar a legibilidade dos gráficos, somente os valores de três nodos são exibidos em cada figura. Na Figura 30, por exemplo, os nodos 1, 2, 3 e 5 foram escondidos, já que eles têm o comportamento similar ao nodo 4. Os nodos 8 e 9 também são escondidos por apresentarem o mesmo comportamento do nodo 6.

6.2.2 Cenário 2 - *TargetQoF* variável

Nos experimentos de *TargetQoF* variável, a configuração de disposição e número de nodos não foram alterados. Como o algoritmo dos SN já foi avaliado no cenário 1, foram conduzidos dois experimentos neste cenário. Os testes com *TargetQoF* variável fazem uso do monitoramento da temperatura para validar o algoritmo no centro de fusão de dados. Para isso, foi considerado um erro de $1^{\circ}C$ na variação da temperatura, ou seja, a *TargetQoF* deve ser tal a fim de garantir essa precisão. Os sensores de temperatura utilizados são os encontrados de forma embarcada no próprio micro-controlador e não têm grande precisão. Para todas as figuras do cenário 2 a cor azul representa *TargetQoF* variável, a cor vermelha representa o número de *frames* recebidos

e a cor verde representa a temperatura monitorada.

O primeiro experimento com *TargetQoF* variável foi elaborado para certificar que a operação de fusão está funcionando corretamente e verificar como os nodos se comportam. O teste consiste em substituir o valor de temperatura lida nos nodos por valores fixos da seguinte forma: o primeiro nodo a se conectar na rede sempre transmite a temperatura de 35°C, os demais 30°C. Nesta configuração, a temperatura média é de 30.5°C quando o coordenador receber as informações dos 10 nodos. Esta temperatura deve ser maior nos casos em que um número menor de nodos transmitir e o primeiro nodo também o fizer. A Figura 31 exibe o gráfico com resultado desse experimento.

O segundo experimento foi realizado de forma completa, englobando os algoritmos dos SN e do centro de fusão, permitindo gerar a estimativa real da *TargetQoF*. Este experimento é dividido em três partes: a primeira é uma análise geral de comportamento. Na segunda parte, é realizada uma análise aprofundada em um ponto específico do monitoramento. Por fim, na terceira parte é conduzida uma análise a respeito do comportamento dos nodos.

6.3 RESULTADOS COM *TARGETQOF* FIXA

Para os experimentos um e dois, as Figuras 26 e 28 mostram o número de mensagens recebidas durante o período de tempo (s). A linha clara representa o valor instantâneo e a linha escura representa uma média de mensagens a cada 10 amostras.

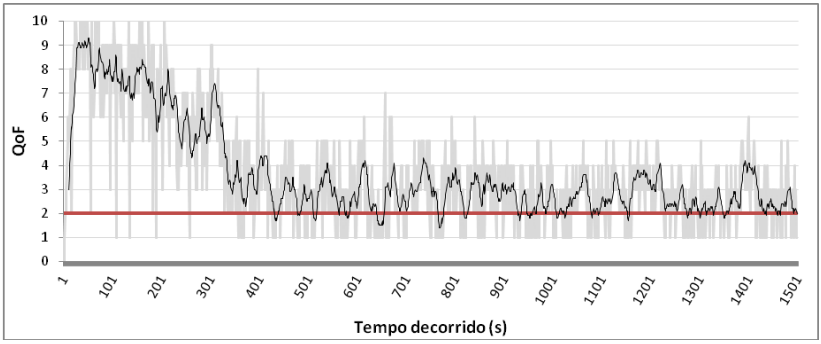


Figura 26: Inicio do experimento com *TargetQoF* fixa e convergência ao longo do tempo.

A Figura 26 mostra a situação inicial de um experimento, sendo possí-

vel observar que no início o número de mensagens que alcançam MN é muito maior que o valor da $TargetQoF$. O tempo de ajuste da rede fica em torno de 400 *rounds*. Este tempo esta vinculado ao período da sessão o qual tem valor 100, ou seja, após 4 execuções de probabilidade local a rede se auto ajusta, reduzindo o número de transmissões, tendo como objetivo alcançar o valor especificado da $TargetQoF = 2$, e por consequentemente, maximizando o tempo de vida da rede.

Como o algoritmo é dinâmico, projetado para lidar com mudanças nas condições da rede, é esperado uma variação ao redor da objetivo ($TargetQoF$). Contudo, foi observado em vários outros experimentos, com outros valores de $TargetQoF$, que a variação é maior quando o valor da $TargetQoF$ é muito menor em relação ao número total de nodos. Isto ocorre devido ao fato que a probabilidade local nos nodos é continuamente reduzida ao longo do tempo com a premissa de economizar energia. Assim, quando aumentam as chances do algoritmo executar o método *adjust_local_probability* para baixo, o sistema irá tentar alavancar a probabilidade local para tentar garantir que no próximo *round* os nodos SN vão transmitir e cumprir com o objetivo da $TargetQoF$.

A Figura 27 mostra o gráfico de dispersão dos nodos do primeiro experimento após 15 execuções, caracterizando o segundo experimento. É possível notar que os nodos com menos energia transmitem abaixo de 10% do tempo. Por outro lado, nodos com mais energia transmitem acima de 50%

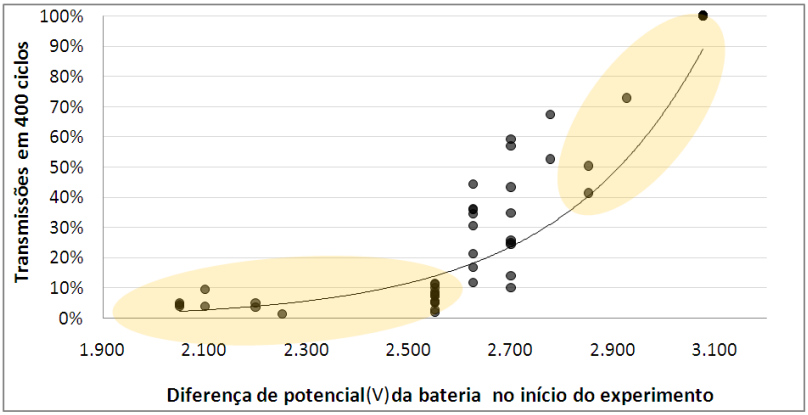


Figura 27: Energia inicial dos nodos frente ao número de transmissões efetuadas por eles.

do tempo e nodos alimentados com uma fonte externa de energia transmitem em todos os *rounds* 100% do tempo. No gráfico é possível ver três regiões: além das regiões com maior e menor do que a energia, existe uma região intermediária com nodos apresentando voltagem da bateria por volta de $2.7V$, e a probabilidade local flutua entre os valores intermediários. A variação acontece devido aos nodos com a mesma tensão de bateria ficarem constantemente alternando a probabilidade local. Esse fenômeno pode ser notado também nas demais faixas, por exemplo, $2.55V$ e $2.65V$.

A Figura 28 mostra o experimento número três, onde a configuração inicial da rede consiste de 5 SN conectados ao MN e, no instante igual a 1700s, mais 5 SN juntam-se à rede. O valor $TargetQoF$ permanece igual a 2 mensagens por *round*. O resultado mostra que quando mais nodos se juntam a rede, maior o número de mensagens recebidas. Isto ocorre porque os nodos iniciam geralmente com alta probabilidade local (*LOCAL_PROB*). O tempo gasto pela rede para se auto ajustar é mais curto que no primeiro experimento (Figura 26), uma vez que no tempo $t = 1700$ metade da rede já estava de acordo com o valor da $TargetQoF$.

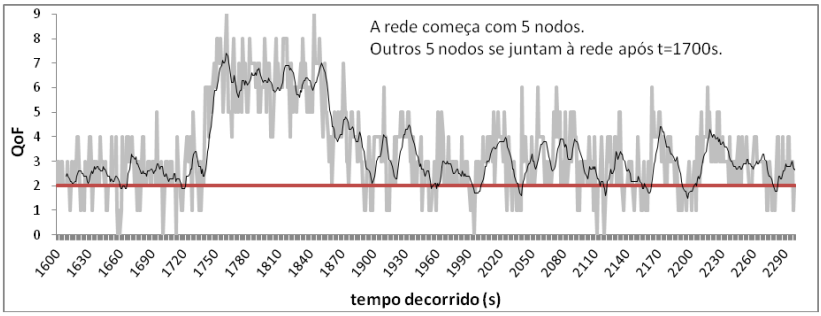


Figura 28: Rede executando com 5 SN e outros 5 SN juntam-se à rede no tempo t=1700. Sistema fica convergindo por 200 *rounds* (2 ajustes de probabilidade local).

Na Figura 29, o nodo 5 é o dispositivo com menor energia em seu conjunto de baterias, com voltagem inicial de 2.35V. O nodo 6 é alimentado por uma fonte de energia externa de 3V. Todos os outros nodos – representados pelo nodo 4 na Figura – têm diferença de potencial da bateria igual a 2.55V. É possível observar que o nodo alimentado com fonte externa coopera com compromisso de alcançar o valor de $TargetQoF$ e assume a responsabilidade de enviar uma mensagem a cada *round*. Por outro lado, o nodo que tem o menor valor de energia transmite menos mensagens do que o restante dos no-

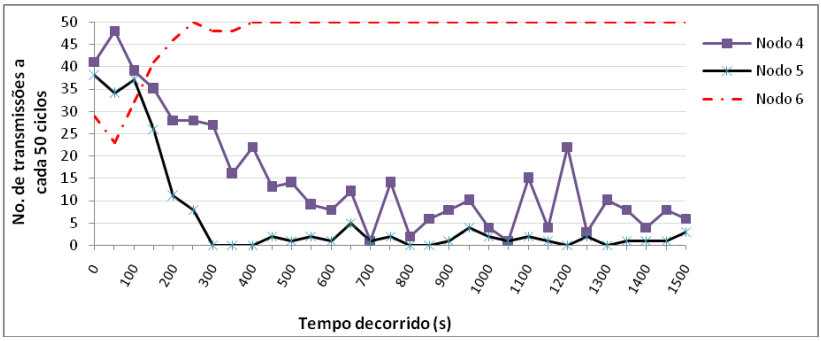


Figura 29: Nodos com diferentes níveis de energia nas baterias. O nodo 6 é alimentado permanentemente por fonte externa de 3V e nodo 5 possui menos energia na bateria.

dos. Contudo, este nodo nem sempre fica sem transmitir, por esse não ser um comportamento apropriado. O propósito do algoritmo é reduzir a probabilidade de envio deste nodo, em relação aos demais, mantendo o compromisso de ajudar a realizar os objetivos do centro de fusão, o que acarreta que eventualmente ele tem que transmitir.

Na Figura 30 não há nodos alimentados por fontes externas. No entanto, o nodo 7 possui tensão de 2.3V, diferentemente de todos os outros nodos que possuem tensão de 2.55V. Neste experimento, os nodos 6 e 7 se juntam à rede por volta do tempo t=1650. Como esperado, o nodo 7 reduz

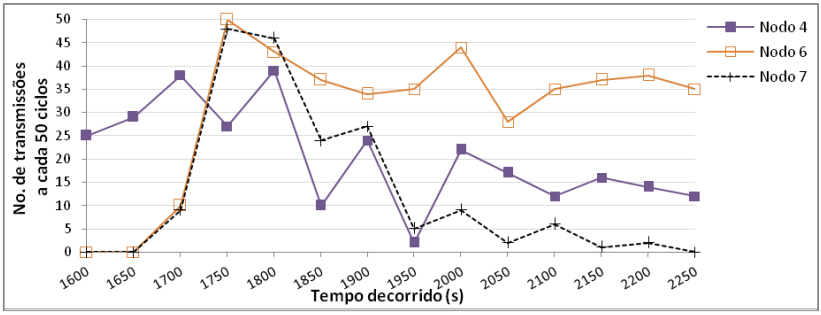


Figura 30: Nodo 7 possui menos energia em suas baterias. Nodos 6 e 7 começam a transmitir somente no tempo t=1650.

sua probabilidade de transmissão (completando mais posições do buffer com $valor = 0$). Outro detalhe interessante é que os dispositivos que estavam transmitindo há mais tempo na rede (e.g. nodo 4) se aproveitam do momento de transição e baixam suas probabilidades locais. Esse comportamento não é definitivo, pois com o passar do tempo suas probabilidades locais de envio acabam subindo novamente, equilibrando-se com os nodos que possuem a mesma tensão. O interessante é que após o período de ajuste, que gira em torno de 200 *rounds*, o nodo com menor energia (nodo 7) é o que apresenta a menor probabilidade local seguido dos nodos 4 e 6. Isso mostra que o algoritmo dos SN está funcionando conforme o esperado.

Os experimentos com *QoF* fixa foram importantes para realizar ajustes nos algoritmos dos SN. Com estes ajustes, o sistema passou a ter um maior grau de confiabilidade e pode ser submetido a uma *QoF* variável. Assim, o algoritmo dos SN foi validado e seu funcionamento está de acordo com o esperado, ou seja, os nodos devem economizar energia, sempre que possível, sem comprometer o centro de fusão. A ordem de economia de energia deve obedecer os níveis de energia do conjunto de baterias de cada nodo.

6.4 RESULTADOS COM *TARGET QOF* VARIÁVEL

O primeiro experimento com *Target QoF* variável, Figura 31, foi conduzido de tal forma que até o tempo 300 no máximo 5 nodos deveriam estar conectados e transmitindo, sendo que os nodos são ligados aos poucos. Quando os primeiros nodos se conectam a temperatura média chega a 35°C. Conforme mais nodos vão se conectando (tempo 50) a temperatura média cai. Próximo do tempo 100, existem 2 nodos que ficam transmitindo con-

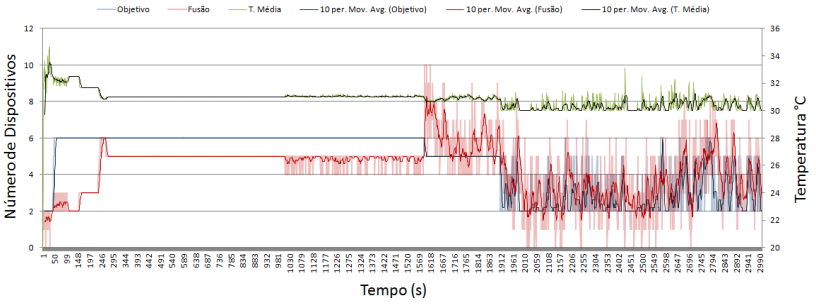


Figura 31: Teste da *QoF* com valores de temperatura pré-definidos.

tinuamente e um terceiro está tentando se conectar. Quando este consegue transmitir, o sistema fica com três nodos transmitindo continuamente. Mais três nodos são ligados, porém, apenas dois serão mantidos e o terceiro será desativado. Assim, pouco antes do tempo 300s, 5 nodos estão continuamente transmitindo.

Por volta do tempo 1000s, um dos nodos perde alguns pacotes por estar operando no limite da bateria 1.8V. Esta informação não fica evidente apenas observando o gráfico e é significativa para explicar o motivo da oscilação até o tempo 1570s. Independentemente disso, o nodo se manteve ativo na rede durante todo o experimento e realizou contribuições.

Em seguida, no período 1570s outros 5 nodos entram na rede. Nesse momento, a *TargetQoF* cai quase que instantaneamente para 5. Em seguida os nodos começam a convergir para a nova referência. Após um período de tempo, a fusão de dados tem um valor mais apurado e novamente a *TargetQoF* cai, indo para o valor 2 (tempo 1900). A partir desse momento, os SN e o centro de fusão MN atualizam-se constantemente para alcançarem seus objetivos. Os nodos tentam economizar energia e, quando o fazem em demasia, a saída da fusão tem um desvio padrão relevante e um novo valor de *TargetQoF* é gerado. Este processo permanece ao longo do tempo.

O experimento mostra que a temperatura média alcançada no centro de fusão é adequada. Por volta do tempo 1912s o sistema está com uma média abaixo de 30.5°C e a *TargetQoF* passa a variar entre 2 e 5 nodos, com média a cada 10 amostras de 3 nodos. A temperatura média varia entre [30.0:30.5)°C com um intervalo de confiança para 95%, o que é melhor do que uma média aritmética dos 10 nodos transmitindo continuamente. A partir deste momento, pode-se afirmar que: o valor médio da temperatura no centro de fusão se encontra melhor que uma média aritmética de 10 nodos. Esse valor é alcançado com menos de 4 transmissões em média, o que representa uma economia de energia em torno de 60% nas transmissões.

O segundo experimento conduzido apresenta os resultados de todos os algoritmos trabalhando em conjunto, Figura 32. Este experimento foi iniciado no momento em que alguns dos SN estavam expostos a raios solares. Isso permitiu avaliar o sistema com temperaturas bem diferentes durante um determinado tempo no início do experimento, e com o passar do tempo eles deixaram de receber essa energia. Esse experimento foi propositalmente realizado para mostrar alguns detalhes da execução dos algoritmos. Os detalhes podem ser visualizados na Figura 33 que é a seleção na cor laranja da Figura 32. Esta seleção representa a segunda análise conduzida.

O experimento foi iniciado com apenas 5 nodos para existir um alto desvio padrão. Dois nodos recebiam diretamente raios solares e chegaram a apresentar uma temperatura de 43.2 e 44.74°C, enquanto os demais apre-

sentavam 31.0, 26.2, 23.4 °C (tempo 100). Devido a um alto desvio padrão apresentado, o centro de fusão determinava uma $TargetQoF = 21$, obrigando os nodos a transmitirem continuamente. Por volta do tempo 750s outros 5 nodos foram conectados e o sistema melhorou consideravelmente. Após o período de tempo de 100s a $TargetQoF$ já tinha se auto ajustado para um valor em torno de 10 nodos, e continuava caindo. Após esse marco, o algoritmo de buffer junto com a fusão de dados vai gradativamente melhorando o resultado monitorado. Por volta do tempo 2100s, nenhum nodo estava sob influência de raios solares, o que acarretou em uma melhora significativa no recurso monitorado. O experimento é conduzido neste estado até o tempo 6000s onde iniciou-se o desligamento gradual dos nodos. Devido ao desligamento dos nodos, a temperatura começou a ser deslocada para cima, ficando menos apurada. A partir do tempo 7300s foram deixados apenas 4 nodos ligados e estes passaram a transmitir dados coletados a cada *round*.

A parte mais relevante deste experimento está nos resultados do tempo 6000s até o tempo 7300s. No tempo 6000s existem 10 SN transmitindo informações, chegando ao centro de fusão em torno de 5 mensagens. Quando restaram apenas 4 nodos, o resultado da fusão tinha se posicionado por volta de 1°C de diferença. Em termos de consumo energético, o valor em 6000s é equivalente ao de 7300s, contudo, o valor do fenômeno monitorado era consideravelmente diferente.

A segunda parte da análise é conduzida na seleção na cor laranja da Figura 32. Esta seleção, Figura 33, busca uma análise para mostrar em mais detalhes o comportamento do centro de fusão e o algoritmo de buffer. Apesar dela estar numerada de 1 a 500, para facilitar a leitura, ela compreende a faixa de tempo de 2000 a 2500s da Figura 32. A $TargetQoF$ é um reflexo do des-

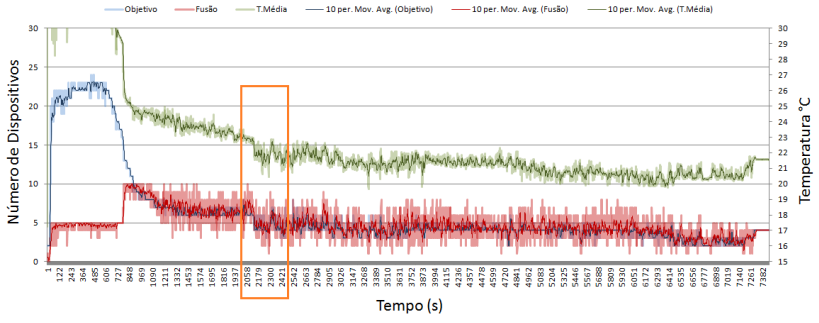


Figura 32: Experimento real com fusão de dados e QoF variável (10 nodos).

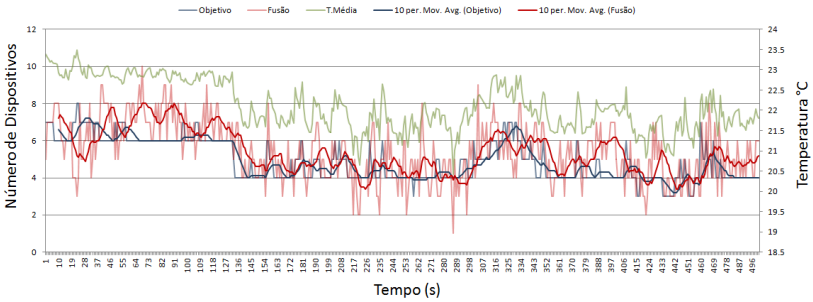


Figura 33: Período de fusão de dados com dados reais de 10 nodos.

vio padrão das amostras de temperaturas recebidas. Entre o tempo 10 e 28 o número de amostras cai devido aos nodos economizarem muita energia. Com isso o desvio padrão destas amostras aumentou a ponto de fazer a *TargetQoF* subir de 6 para 7 e em seguida 8. Uma vez que os nodos detectaram esse aumento na *TargetQoF* pelo *feedback* enviado pelo centro de fusão, os nodos passam a transmitir mais, restaurando o dado monitorado a uma faixa aceitável. Como o centro de fusão aumentou por um determinado tempo o valor da *TargetQoF*, há um reflexo nos nodos ao longo do tempo (64 a 100s). Esse aumento no número de transmissões ajudou a temperatura média a se estabelecer dentro de um valor mais correto e fez com que o desvio padrão diminuisse (tempo 130s). Uma vez que o erro cai, o valor da *TargetQoF* baixa. Essa diminuição é sentida pelos nodos e eles aproveitam para realizar ajustes nas probabilidades locais, buscando novamente a economia de energia. Se o ajuste não prejudica o valor medido, o sistema permanece economizando energia até o desvio padrão se alterar novamente (tempo 290s).

Esse distúrbio no desvio padrão pode ocorrer por diversos fatores, como raios solares incidindo diretamente sobre os sensores ou a posição dos sensores no espaço. O importante é que o centro de fusão detecta e repassa esta informação para os nodos na forma de *TargetQoF*. Os nodos acabam transmitindo mais para compensar um valor que pode estar gerando o alto desvio padrão gerando um comportamento cíclico. Um exemplo desse comportamento pode ser observado no tempo (300 a 400s).

Na Figura 34 é apresentado o gráfico de transmissão acumulada dos SN do mesmo período utilizado na Figura 33. O intuito é mostrar o comportamento local e/ou individual neste mesmo período para os nodos. Os nodos 1 e 7 estavam com 1.75 e 2.4V de bateria, os demais apresentavam 2.55V. Pode-se notar que os nodos 2, 3, 8 e 10 passam a transmitir menos a partir do

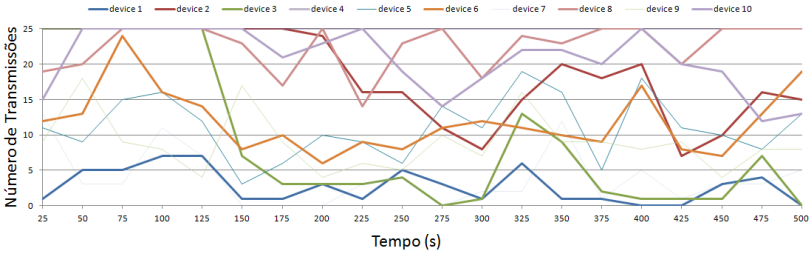


Figura 34: Transmissões acumuladas de 10 nodos da seleção da Figura 32

tempo 100. Estes nodos são os que passaram a economizar energia fazendo a curva de fusão cair na Figura 33, para o mesmo período. Quando a taxa de transmissão começa a cair muito, estes mesmos nodos voltaram a aumentar a sua taxa de transmissão. Este fato fica mais evidente com os nodos 2, 3 e 10 a partir do tempo 300s. Os dispositivos 1, 5, 7 e 9 apesar de sofrerem alterações, em alguns casos consideráveis, no final estavam praticamente com a mesma probabilidade. Isto mostra que uma vez que um nodo chega a uma certa probabilidade ele tende a manter o valor da frequência de transmissão se não pode economizar mais energia. Por via de regra, ele experimenta economizar energia e a mantém somente se o sistema não sofre com isso. Este fato fica evidente com o nodo 3, que estava sempre transmitindo e passou a quase não transmitir. Em seguida aumenta sua probabilidade local e volta a reduzir porque outros assumiram a responsabilidade. O nodo 6 é um exemplo que assume responsabilidade no final. Ele tenta assumir logo no início do período, volta a economizar energia e, no final do período, ele passa a ter uma grande contribuição novamente.

6.5 CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram apresentados os resultados de vários experimentos executados dos algoritmos propostos implementados no hardware desenvolvido. Inicialmente ensaios foram conduzidos para testar o comportamento dos nodos e desenvolver as métricas para o sistema distribuído. Em seguida, as regras de economia de energia foram verificadas para validar o modelo e algoritmos dos nodos.

Os próximos ensaios foram conduzidos para validar o algoritmo do centro de fusão de dados executando com valores de $TargetQoF$ fixa. Com os dados coletados, uma análise foi conduzida para verificar se o algoritmo

de buffer probabilístico é capaz de economizar energia sem comprometer a taxa de mensagens necessárias para a fusão de dados.

Uma vez validado o algoritmo nos SN, outros experimentos foram conduzidos com *TargetQoF* variável. Com os dados de ensaios reais foi possível evidenciar que o algoritmo de fusão de dados consegue, em média, as informações precisas se utilizado com o algoritmo de buffer probabilístico e com economia de energia. Isso fica evidente em uma comparação que é realizada sobre 10 nodos transmitindo 40% do tempo e outros 4 nodos transmitindo informações constantemente. O resultado foi que para o mesmo consumo energético o resultado com 10 nodos estava com qualidade superior.

Neste capítulo foram mostrados ainda como o algoritmo refina os dados com o passar do tempo, economiza energia e ajusta as taxas de *TargetQoF* sem comprometer os resultados. A análise é conduzida de forma a explicar o ciclo de trabalho do algoritmo de fusão de dados em conjunto com o algoritmo de buffer probabilístico, com evidências reais. O resultado apontou que os valores são consistentes e que os nodos economizam energia somente se não comprometem o resultado do centro de fusão.

7 CONCLUSÕES

7.1 REVISÃO DAS MOTIVAÇÕES E OBJETIVOS

As redes de sensores sem fio (RSSF) ganham importância crescente a cada ano que passa. A adoção de equipamentos Zigbee (cuja norma adota como camada física o padrão IEEE 802.15.4) é um exemplo. Esta tecnologia de comunicação sem fio voltada para RSSF cresceu na ordem de 800% nos últimos 5 anos. Na grande maioria das situações, os dispositivos RSSF são alimentados por baterias ou fontes alternativas de energia. Nos casos onde baterias são utilizadas, o aumento no número de dispositivos na rede traz vários problemas. Entre os problemas, a complexidade, condições dinâmicas e a imposibilidade de administração humana das RSSF levantam a necessidade de sistemas com características de auto-gestão, multi-objetivo e que dinamicamente ajustam-se às condições da rede.

Os nodos de RSSF têm, de forma geral, baixo custo e seus sensores são usualmente pouco confiáveis. Dessa forma, os projetos de RSSF assumem que os dados monitorados possuem um determinado grau de imprecisão. Uma das técnicas utilizadas para melhorar a confiabilidade dos dados é a fusão de dados. Ela aprimora as informações extraídas dos dados lidos pelos sensores para ajudar na tomada de decisão do sistema e fornecer valores mais confiáveis. Técnicas de fusão de dados podem ser realizadas de três formas: serial, paralela e híbida; sendo a fusão de dados paralela a mais adequada para redes em topologia estrela, a qual é assumida como premissa deste trabalho.

O principal objetivo desta dissertação de mestrado foi propor uma abordagem de comunicação para fusão paralela de dados em RSSF. A solução de software adotada executa um algoritmo que busca economizar energia, na média, sem sacrificar a qualidade da estimação da rede baseada nas grandezas monitoradas pelos nodos. Para tal, foram estudados os principais conceitos relacionados com RSSF e sobre o padrão 802.15.4 e os principais modelos de fusão de dados a fim de encontrar o melhor para o modelo proposto. Como o trabalho tem foco voltado à economia de energia, foi realizado um estudo e proposta uma solução de hardware que utiliza o padrão 802.15.4. De posse do hardware, foram implementadas as técnicas propostas e a validação foi conduzida e experimentalmente avaliada.

7.2 CONTRIBUIÇÃO E ESCOPO DO TRABALHO

A principal contribuição deste trabalho foi uma abordagem de comunicação utilizando a técnica de fusão paralela de dados em RSSF IEEE 802.15.4. A abordagem é composta de uma solução de hardware e software. O hardware foi desenvolvido com intuito de economizar energia utilizando os recursos do padrão IEEE 802.15.4 e algumas otimizações a nível de projeto. O software é composto por dois algoritmos. O primeiro algoritmo é executado de forma distribuída nos SN. Este software permite que cada nodo se auto-ajuste e controle sua participação no envio de mensagens na RSSF de forma a não comprometer o resultado monitorado. O segundo algoritmo é executado no coordenador, realizando uma fusão paralela de dados. Nesta operação, de posse da métrica *TargetQoF* e do número de mensagens recebidas no *round* atual, o MN envia estas informações para os SN se auto-ajustarem.

A abordagem proposta permite que técnicas convencionais para aplicações de fusão de dados, como Dempster-Shafer ou Filtro de Kalman (NAKAMURA; LOUREIRO; FRERY, 2007), ou estratégias simples, como MF (Measurement Fusion) ou OPEF (Open-loop Partial Estimate Fusion) (CHIUSO; SCHENATO, 2011) possam ser implementadas. As abordagem hierárquicas ou com base em árvore também podem tirar vantagem, incorporando a abordagem proposta em fusão de nível de grupo ou *cluster*.

Com uma melhoria desenvolvida no algoritmo, a qual permite o ajuste automático da *TargetQoF*, foi possível economizar energia sem comprometer os resultados. Os experimentos com *TargetQoF* variável mostraram que o algoritmo consegue melhores resultados, pois explora uma maior diversidade de possibilidades. Os resultados são melhores quando comparados com abordagens mais simples, que transmitem de forma determinística e com menos redundância de informações; ou seja, em termos energéticos se consegue iguais ou melhores resultados.

O algoritmo dos SN é simples, sendo implementado em nodos reais, o que torna possível alcançar os objetivos traçados neste trabalho. A implementação pode ser feita em qualquer nodo com recursos limitados em termos de uso de memória, energia e capacidade de processamento. Em termos de *overhead* introduzido nas mensagens de controle, o algoritmo apenas acrescenta três bytes para controlar uma rede com até 256 nodos.

7.3 TRABALHOS PUBLICADOS

Até o presente momento, o presente trabalho foi submetido na forma de dois artigos para simpósios (um nacional e outro internacional), tendo sido

aceito para publicação como trabalho completo em ambos. Os primeiros resultados foram publicados no *Simpósio Brasileiro de Engenharia de Sistemas Computacionais (SBESC)*, realizado na cidade de Florianópolis nos dias 7 a 11 de novembro de 2011 (BUDKE et al., 2011).

Com os resultados consolidados do algoritmo distribuído para os SN o trabalho foi aceito no congresso *ETFA 2012 – IEEE International Conference on Emerging Technology & Factory Automation*, realizado em Cracóvia, Polônia, nos dias 17 a 21 de Setembro de 2012 (BUDKE et al., 2012).

7.4 PERSPECTIVAS FUTURAS

Como proposta de trabalhos futuros, pretende-se fazer experimentos como um número maior de nodos e explorar questões relacionadas com mobilidade. Provavelmente, essa tarefa envolverá o emprego de um simulador.

No algoritmo de buffer probabilístico pretende-se verificar a sensibilidade frente a diferentes tamanhos no *buffer*, uma vez que este tamanho interfere no período de inicialização. Além disso, pretende-se verificar outras alternativas no algoritmo para reagir mais rapidamente a mudanças na topologia da rede. Outro trabalho futuro importante que pode ser desenvolvido é estudar o comportamento em redes do tipo *mesh*, com e sem *clusters*.

REFERÊNCIAS BIBLIOGRÁFICAS

AMARSAIKHANA, D.; DOUGLAS, T. Data fusion and multisource image classification. *International Journal of Remote Sensing*, v. 25, p. 2529–2539, 2004. <http://dx.doi.org/10.1080/0143116031000115111>.

ATMEL. *Atmel AVR2025: IEEE 802.15.4 MAC Software Package - User Guide*. 2011.

AYERS, M.; LIANG, Y. Gureen game: An energy-efficient qos control scheme for wireless sensor networks. In: *Proceedings of Green Computing Conference and Workshops*. [S.l.: s.n.], 2011. p. 1–8.

BLASCH, E. P.; PLANO, S. Jdl level 5 fusion model "user refinement"issues and applications in group tracking. *SPIE: International Society for Pptics and Photonics - Aerosense*, v. 4729, p. 270–279, 2002.

BUDKE, G. F. et al. A communication approach for parallel data fusion in ieee 802.15.4 wireless sensor networks. *Computing System Engineering (SBESC), 2011 Brazilian Symposium on*, p. 60–65, November 2011. http://dx.doi.org/10.1109/SBESC.2011.37.

BUDKE, G. F. et al. A dynamic communication approach for data fusion in ieee 802.15.4 wireless sensor networks. *ETFA 2012 – IEEE International Conference on Emerging Technology & Factory Automation*, September 2012.

CHEN, X. et al. A lightweight data aggregation protocol in wireless sensor networks for the protection of ancient sites. *Journal of Sensor Technology*, v. 1, p. 91–98, December 2011.

CHIUSO, A.; SCHENATO, L. Information fusion strategies and performance bounds in packet-drop networks. *Automatica*, v. 47, n. 7, p. 1304–1316, 2011.

CYNAPSIS. *ZigBee: A Growing Technology for Smart Green Energy*. 2012. http://www.cynapsys.de/en/articles/zigbee-a-growing-technology-for-smart-green-energy.html.

D'COSTA, A.; SAYEED, A. M. Data versus decision fusion for distributed classification in sensor networks. In: *Proceedings of the 2003 IEEE conference on Military communications - Volume I*. Washington, DC, USA: IEEE Computer Society, 2003. (MILCOM'03), p. 585–590. ISBN 0-7803-8140-8. <http://dl.acm.org/citation.cfm?id=1950503.1950625>.

ELMENREICH, W. *Sensor Fusion in Time-Triggered Systems*. Tese (Dissertation) — University of Viena - Austria, 2002.

FAOUZI, N.-E. E.; LEUNG, H.; KURIAN, A. Data fusion in intelligent transportation systems: Progress and challenges - a survey. *Information Fusion*, v. 12, n. 1, p. 4–10, 2011. ISSN 1566-2535. Special Issue on Intelligent Transportation Systems. <http://www.sciencedirect.com/science/article/pii/S1566253510000643>.

GOLçALVES, F. D. et al. Fusão de dados multisensor para a identificação e o mapeamento de ambientes flúvio-estuarinos da amazônia. *Revista Brasileira de Geofísica*, v. 27, p. 47–67, 2009. <http://dx.doi.org/10.1590/S0102-261X2009000500005>.

HALL, D. L.; LLINAS, J. An introduction to multisensor data fusion. In: *Proceedings of the IEEE - JANUARY 1997*. [S.l.: s.n.], 1997. v. 85, n. 1.

HE, T. et al. An overview of data aggregation architecture for real-time tracking with sensor networks. In: *IPDPS 2006 – 20th Int. Parallel and Distributed Processing Symp.* [S.l.: s.n.], 2006. p. 1–8.

HOSEINNEZHAD, R.; BAB-HADIASHAR, A. Fusion of redundant information in brake-by-wire systems using a fuzzy voter. *Journal of Advances in Information Fusion*, v. 1, n. 1, p. 52–62, 2006.

IEEE-802.15.4. Part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area network (LR-WPAN). *in IEEE-SA Standards Board 802.15.4-2011*, 2011.

LIU, E. C.; MOURA, J. M. F. Fusion in sensor networks: convergence study. In: *In Proceedings of the International Conference on Acoustic, Speech, and Signal Processing*. [S.l.: s.n.], 2004. p. 865–868.

LLINAS, J. et al. Revisiting the jdl data fusion model ii. In: *In P. Svensson and J. Schubert (Eds.), Proceedings of the Seventh International Conference on Information Fusion (FUSION 2004*. [S.l.: s.n.], 2004. p. 1218–1230.

NAKAMURA, E. F.; LOUREIRO, A. A. F.; FRERY, A. C. Information fusion for wireless sensor networks: Methods, models, and classifications. *ACM Computing Surveys*, v. 39, n. 3, p. 1304–1316, 2007.

NICHOLSON, D. An automatic method for eliminating spurious data from sensor networks. In: *IEEE Target Tracking 2004: Algorithms and Applications*. [S.l.: s.n.], 2004. p. 57–61.

NISHIMURA, T.; MATSUMOTO, M. *A Mersenne Twister pseudorandom number generator with period* $2^{19937} - 1$. 2010. `www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html`.

PAI, H.-T.; HAN, Y. S. Power-efficient data fusion assurance using direct voting mechanism in wireless sensor networks. *Sensor Networks, Ubiquitous, and Trustworthy Computing, International Conference on*, IEEE Computer Society, Los Alamitos, CA, USA, v. 1, p. 368–375, 2006.

PATIL, S.; DAS, S. R. Serial data fusion using space-filling curves in wireless sensor networks. In: *Proc. of IEEE Int. Conf. on Sensor and Ad Hoc Communications and Networks SECON'04*. [S.l.: s.n.], 2004. p. 182–190.

PINTO, A.; MONTEZ, C. Autonomic approaches for enhancing communication qos in dense wireless sensor networks with real time requirements. In: *Proc. of IEEE Int. Test Conference ITC'2010*. [S.l.: s.n.], 2010. p. 1–10.

PINTO, A. S. R. *Abordagens Autonômicas para Qualidade de Serviço da Comunicação em Redes de Sensores sem Fio Densas com Requisitos Temporais*. Tese (Tese) — UFSC - Universidade Federal de Santa Catarina, 2010.

STEINER, G. Sequential fusion of ultrasound and eletrical capacitance tomography. *International Journal of Information and System Sciences*, v. 2, n. 4, p. 484–497, 2006.

WILLIG, A. *Recent and Emerging Topics in Wireless Industrial Communications: A Selection*. 2008.

WORLD'S, O. *ZigBee/802.15.4 Module Revenues to Approach $1.7 Billion in 2015*. 2011. `http://www.onworld.com/news/newszigbeenetofthings.html`.

YUAN, W.; KRISHNAMURTHY, S. V.; TRIPATHI, S. K. Synchronization of multiple levels of data fusion in wireless sensor networks. In: *Global Telecommunications Conference*. [S.l.]: GLOBECOM '03. IEEE, 2003. p. 221–225.

ZIGBEEORG. *Zigbee Alliance Releases New Smart Energy Standard*. 2011. `http://goingzigbee.com/zigbee-alliance-releases-new-smart-energy-standard`.