Faculdade de Engenharia da Universidade do Porto

FEUP

Rede de sensores sem fios para veículo frigorífico e/ou indústria alimentar

André Soares Oliveira Leite

VERSÃO FINAL

Dissertação realizada(o) no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major de Telecomunicações

Orientador: Prof. Dr. Luís Almeida
Proponente: Eng. Vítor Silva (FoodInTech)

Junho de 2010

Resumo

Existe a necessidade por parte das indústrias alimentares da possibilidade de gerir o transporte dos seus produtos e garantir que estes não sofrem alterações no que diz respeito a qualidade para o consumo, devido a um transporte indevido. Para isso existe a norma NP EN 12 830, que homologa os sistemas de supervisão integrantes do transporte de produtos congelados, com a finalidade de se obter no final um relatório/prova das condições desse transporte, provando que os produtos transportados não estiverem expostos a qualquer tipo de situação que pusesse em causa a sua qualidade.

O principal problema por isso prende-se com a verificação periódica da temperatura dentro da zona de armazenamento de produtos, de maneira a que se verifique que esta não ultrapassa os limites aceitáveis. De igual forma é relevante também uma monitorização da abertura e fecho da porta da zona de armazenamento.

Desenvolveu-se por isso um sistema de aquisição de dados sem fios baseados numa rede de dispositivos, para actuar dentro de uma câmara de um veículo frigorífico. Nesta rede é monitorizada a temperatura em vários nós e a abertura e fecho da porta da zona de armazenamento dos produtos. Esta informação é enviada para o fora da zona de armazenamento dos produtos, mais propriamente para a cabine do veículo. Na cabine encontra-se uma solução computacional que é responsável por guardar as informações da temperatura e abertura de porta da câmara frigorífica ao longo da viagem, sendo ainda responsável pela geração de informação visual para o condutor bem como sinalização de aviso/alarme no caso de estes ocorrerem. No final da viagem o camião envia automaticamente para um computador de destino todo o registo (*logfile*) efectuado durante a viagem. O camião fica ainda a espera de receber um comando de configuração, comando esse que define como a rede de sensores do veículo se comportará ao longo da próxima viagem.

Com vista a concepção deste projecto optou-se pela utilização de uma rede de dispositivos *Zigbee* dentro da zona de armazenamento de produtos, a qual comunica com uma gateway que é responsável por enviar de forma fiável os dados da zona de armazenamento para a cabine do veículo através de CAN. A troca de informação da cabine com o computador de destino é feita através de Wi-Fi.

Abstract

There is a need by the food industry's to manage the transport of their products and ensure that these remain unchanged with respect to quality for consumption, due to improper transportation. For that there is a standard NP EN 12 830, which approves the supervisory of the transport of frozen products, with the aim of getting a report / proof of the conditions of carriage in the final, proving that the transported goods weren't exposed to any type of situation to undermine its quality.

The main problem therefore lies in the regular checking of temperature inside the storage area, so that it does not exceed the acceptable limits. Equally also relevant is the monitoring of the opening and closing of the door in the storage area.

Therefore we developed a system of data acquisition based on a wireless network device to operate within a chamber of a refrigerated vehicle. This network is responsible for monitor the temperature at various nodes and the opening and closing of the door in the storage area of the products. This information is sent to the area outside the storage of products, more specifically for the vehicle cabin. In the cabin we have a computer solution that is responsible for storing the information of temperature and open door of the refrigeration along the journey and is still responsible for generating visual information to the driver and signs warning / alarm in case of these occurs. At the end of the trip the truck must automatically sends to a target computer the entire record (logfile) made during the trip. After that the truck must still wait for receive a configuration command, command that defines how the sensor network inside the vehicle will behave over the next journey.

In order to design this project we chose to use Zigbee network devices inside the product storage area, which communicates with a gateway that is responsible for sending information from storage area to the vehicle cabin by CAN. Exchanging information with the destination computer is made by Wi-Fi.

Agradecimentos

Queria agradecer a todos os que tornaram este trabalho possível. A toda a comunidade da Faculdade de Engenharia da Universidade do Porto, especialmente ao meu orientador, professor Luís Almeida por toda a atenção, tempo dispendido, bem como suas recomendações e contribuições para o trabalho. Gostaria ainda de agradecer aos colegas Luís Oliveira, Pedro Silva, Luís Cardoso e Tiago Mendes por todo o apoio prestado. Por último queria ainda agradecer à minha família, sem a qual nunca chegaria a este ponto, e ainda à minha namorada, Carla, por toda a compreensão e palavras de incentivo.

Índice

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

Lista de abreviaturas (ordenadas por ordem alfabética)

ADC	*Analog to Digital Converter*
API	*Application Programming Interface*
BCD	Binary-*coded decimal*
CAN	*Controller-area network*
DDR	Double data rate
EIA	*Electronic Industries Alliance*
IEEE	*Institute of Electrical and Electronics Engineers*
IP	*Internet Protocol*
LAN	*Local Area Network*
LCD	*Liquid crystal display*
PCI	*Peripheral Component Interconnect*
PLC	*Power Line Communications*
RAM	*Random-access memory*
RS232	*Recommended Standard 232*
RS422	*Recommended Standard 422*
RS485	*Recommended Standard 485*
SATA	*Serial Advanced Technology Attachment*
SPI	*Serial Peripheral Interface*
TIA	*Telecommunications Industry Association*
UART	*Universal asynchronous receiver/transmitter*
USB	*Universal Serial Bus*
VGA	*Video Graphics Array*
WLAN	*Wireless Local Area Network*
WPAN	*Wireless Personal Area Network*

Lista de símbolos

V	Volt
m	Metro
ºC	Graus Centígrados
min	Minutos
h	Horas
Mbit/s	Mega bits por segundo
Kbit/s	Kilo bits por segundo
GHz	Giga Hertz
mW	mili Watts

Capítulo 1

Introdução

Neste capítulo pretende-se com base nas especificações iniciais do projecto começar por fazer um levantamento do problema em geral, dividi-lo em partes de forma a detalhar melhor o problema. Pretende-se ainda criar um fluxo de projecto o qual nos possibilite levar à concepção deste.

Uma vez que a elaboração deste projecto destina-se a fins comerciais, o sistema a ser projectado tem de seguir a norma Portuguesa, em particular a norma Portuguesa homologada para transporte e armazenagem de produtos congelados. Assim faremos também um pequeno levantamento dos principais requisitos impostos por esta norma.

1.1- Análise e descrição do sistema

Com a necessidade das indústrias da monitorização do transporte de produtos alimentares de forma a garantir a qualidade dos mesmos, pretende-se neste projecto chegar a um sistema que esteja em concordância com a norma portuguesa que homologa este tipo de sistemas. Pretende-se criar um produto capaz de fornecer dados ao cliente de forma clara, que seja de fácil instalação, que seja de alta interoperabilidade com outras aplicações e que apresente um baixo preço de acordo com todas as funcionalidades disponibilizadas.

O objectivo/propósito deste projecto consiste na realização de um sistema a integrar num veículo armazenador de produtos congelados que possibilite a:

- Monitorização do valor da temperatura por um operador (condutor) a partir do habitáculo do veículo;
- Monitorização da abertura/fecho da porta da zona de armazenamento pelo operador também a partir do habitáculo;
- Geração de alarmes no habitáculo em caso da temperatura estar fora de uma gama aceitável;

- Registo com uma determinada periodicidade de todos os eventos relacionados com a temperatura e abertura/fecho da zona de armazenamento ocorridos ao longo de toda a viagem, de forma a posteriormente ser enviado de forma autónoma para o destinatário.

O sistema deverá, para além destes requisitos funcionais enumerados, ter como característica principal, permitir uma simples integração no veículo, ou seja, o sistema deverá ser de fácil instalação no veículo e de maneira a que as alterações a nível estrutural neste sejam mínimas. Neste sentido, as tecnologias *wireless* constituem uma solução que nos possibilitara convergir num sistema com estas características, sendo por isso estudadas em capítulos mais adiante em particular as redes de sensores sem fios.

Tendo então em mente o envio e troca de informação via protocolo *wireless*, começou-se a projectar uma estrutura física global dos integrantes deste. Esta então seria composta por vários pontos de medição, distribuídos pela zona de armazenamento do veículo, sendo possível em cada um desses pontos a mensuração da temperatura e o envio via protocolo *wireless* para uma central, a qual estaria localizada na parte da frente do veículo (habitáculo). Esta central teria então a capacidade de armazenamento dos dados durante a viagem, e de fornecer uma monitorização visual ao condutor (através de um simples monitor), bem como ainda a geração de alarmes aquando de alguma anomalia detectada. De notar ainda que será preciso um ponto de mensuração, (tal como ilustrado na Figura 1.1 de "Ponto 1") que para além de realizar medições do valor da temperatura, teria uma ligação a um interruptor de forma a detectar a abertura e fecho da porta, para posterior envio também desta informação.

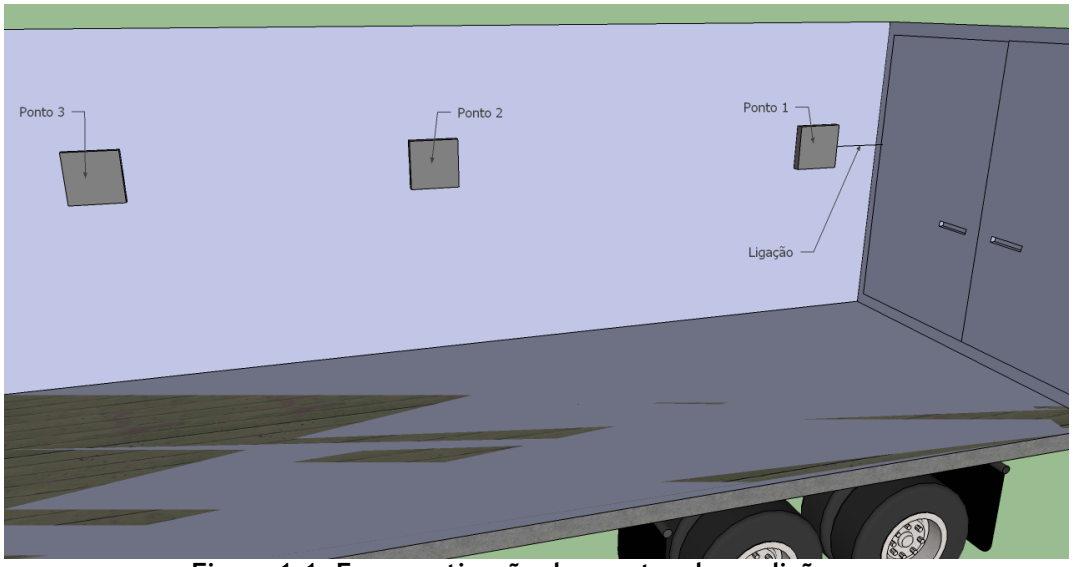


Figura 1.1- Esquematização dos pontos de medição

Após esta descrição global do funcionamento do sistema, surge um problema relacionado com a estrutura física deste tipo de veículos: as arcas frigoríficas utilizadas nestes para armazenamento de produtos congelados são completamente fechadas/isoladas, contendo na sua constituição elementos condutores. Ora isso constitui um problema para a nossa rede de dispositivos *wireless* sempre que estes pretendam enviar informação para a central, já que esses materiais condutores irão bloquear ou atenuar fortemente a passagem das ondas electromagnéticas para o exterior da arca.

Teve então por isso de ser pensado numa estratégia para conseguir resolver esse problema. De facto a única maneira de o resolver passa por o envio da informação do interior da arca frigorífica para o habitáculo via protocolo de comunicação com fios. Será por isso também estudado mais adiante vários tipos de protocolos com fios.

Adveniente então desta nossa limitação, o sistema de comunicação entre dispositivos que incorpora o veículo terá de ter a estrutura global apresentada na Figura 1.2.

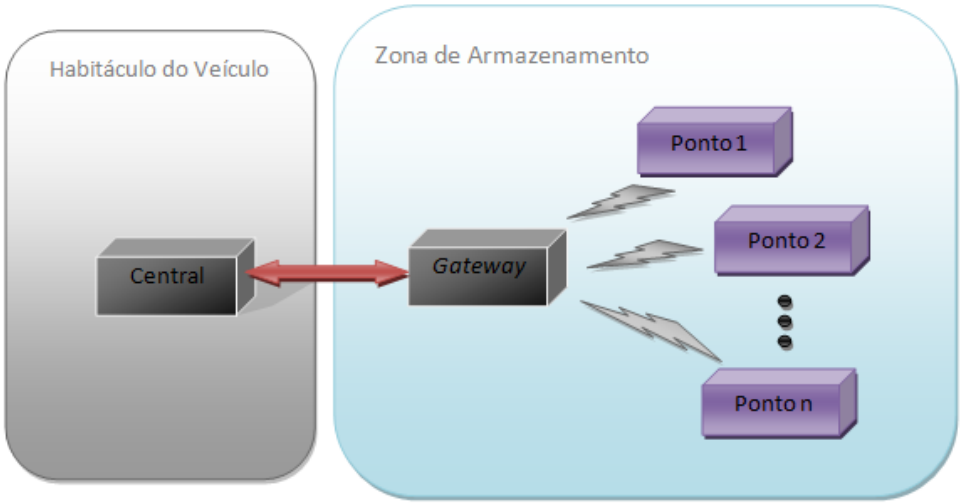


Figura 1.2- Esquema de comunicações no veículo transportador

É possível ter então na zona de armazenamento de produtos um número "n" de pontos de medição de temperatura que comunicam via protocolo *wireless* com uma *gateway* e que enviará as informações recebidas da rede sem fios para a central que se encontra dentro do habitáculo do veículo. A central por sua vez e como já referido acima, terá um monitor com informação diversa para ser disponibilizada ao operador (e.g. temperatura nos vários pontos, horas, alarmes de anomalias, etc.) de forma a este monitorizar todo o processo. A central será ainda responsável por enviar de forma autónoma todos os registos da viagem quando detectar que o veículo chegou ao destinatário da carga (Figura 1.3). Para isso o destinatário terá de dispor de um computador, onde estará ligado um receptor *wireless* do protocolo a ser utilizado no envio do *logfile*. Esse receptor *wireless* enviará então toda a informação recebida para uma aplicação a ser concebida em Java ou outra linguagem de

programação, a qual será responsável então por guardar a informação recebida em um ficheiro (*logfile*). Esse ficheiro deverá então conter:

- Informação registada pelo camião (temperatura e abertura/fecho de porta) e hora da sua ocorrência ao longo da viagem;
- Um campo identificador que visa identificar o camião que enviou a informação;
- Hora e data de quando foi recebida a informação (e criação do ficheiro).

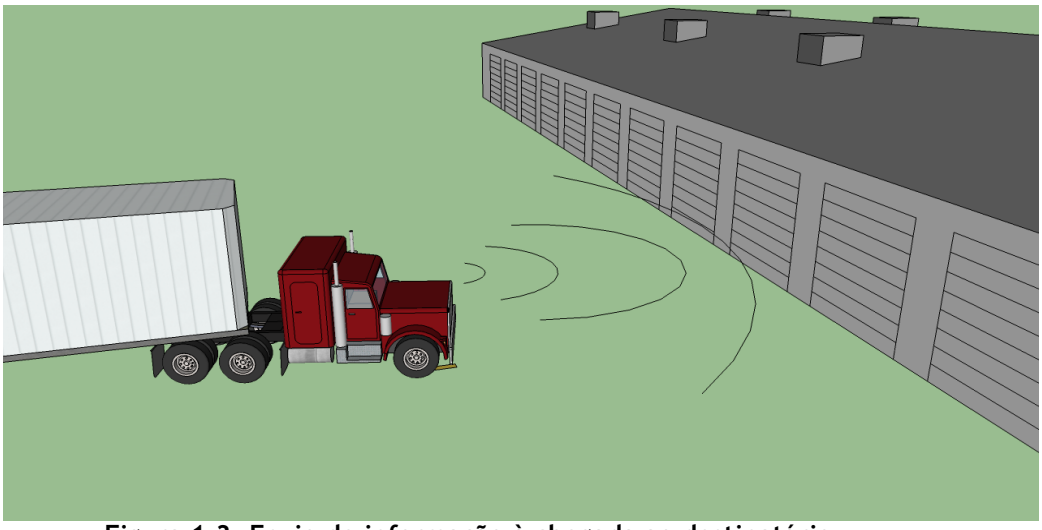


Figura 1.3- Envio de informação à chegada ao destinatário

Todo o projecto deverá ser realizado tendo em mente o cumprimento da norma NP EN 12830, a qual será estudada no próximo ponto, e que nos ajudará também a decidir na escolha dos equipamentos que constituirão o sistema.

1.2- Norma NP EN 12830

A norma NP EN 12830 é uma norma portuguesa para registadores de temperatura para transporte, armazenagem e distribuição de alimentos refrigerados, congelados, ultra-congelados e cremes gelados [1]. Uma vez que o sistema pretendido tem de ser regido por esta norma, esta deve ser objecto de estudo e análise à priori, com vista à obtenção das principais características do sistema a ser projectado. Neste ponto pretende-se elaborar um pequeno resumo das principais informações contidas no documento da norma NP EN 12830 a ter em conta no transporte de alimentos congelados.

Segundo a norma, a temperatura registada dentro da zona de armazenamento deverá estar em graus Célsius, e essa temperatura deverá ser registada com sensores de temperatura independentes, e não com os sensores de temperatura que são usados para o controlo do

sistema de refrigeração. Da informação que deve ser registada, é obrigatório conter a data e a hora do início do registo.

Esta norma especifica ainda as gamas de medição que o sensor de temperatura deverá conter. O limite inferior devera ser igual ou inferior a -25ºC, e como valor do limite superior está estabelecido ser igual ou superior a +15ºC. É especificado ainda que a amplitude de medição (diferença entre os limites superior e inferior) deverá ser igual ou superior a 50. Logo o sensor escolhido não poderá cobrir unicamente os limites mínimos estabelecidos pela norma que tem só 40 de amplitude de medição.

É definido pela norma duas classes deste tipo de sistemas, distinguidos pelos erros máximos, e a resolução da temperatura, apresentados na Tabela 1.1 abaixo.

Tabela 1.1- Classes do sistema

Classe	**1**	**2**
Erros máximos	±1ºC	±2ºC
Resolução	≤0.5ºC	≤1ºC

O intervalo máximo entre registos da temperatura para o transporte de alimentos é definido também pela norma e é dependente da duração da viagem. Quanto maior for a duração da viagem menor é a frequência de medição exigida pela norma, tal como ilustrado na Tabela 1.2

Tabela 1.2- Frequência de leituras a efectuar

Duração da viagem	**Intervalo máximo entre registos (min)**
Menor ou igual a 24h	5
Superior a 24h e menor ou igual a 7 dias	15
Superior a 7 dias	60

É de extrema importância o conhecimento das mínimas frequências de leitura da temperatura uma vez que os pontos de medição que virão a ser utilizados são alimentados com baterias. Assim dependendo da missão (duração da viagem) poder-se-á ajustar essa frequência de leitura de para um destes três valores, de modo a prolongar assim o tempo útil de funcionamento das baterias.

1.3- Estado da arte

Após alguma pesquisa sobre trabalhos e sistemas existentes com funcionalidades parecidas com este, concluímos que a oferta a nível do mercado é limitada, o que levou à proposta deste projecto.

De facto após uma pesquisa, apenas foi encontrado quatro produtos, todos da empresa "Grupo Vei" [26]. No documento [25] de apresentação destes sistemas, existe então a referência a dois deles (TranScan Sentinel e o TranScan 2) que possibilitam apenas a recolha de dados e posterior impressão em papel do relatório. No documento é especificado ainda que existe uma versão de cada um destes produtos para ser instalado no atrelado (compartimento de armazenamento) e outra para ser instalada no habitáculo (compartimento do rádio do veículo), não estando especificado neste último como é feita a ligação aos sensores da zona de armazenagem. A empresa apresenta depois ainda mais dois produtos, o TranScan XL e o TranScan Solo. O TranScan XL permite o descarregamento dos dados para o computador através de uma ligação sem fios a um dispositivo o DCU TranScan (que está conectado ao computador). O TranScan Solo permite uma ligação (não especificada) a um computador para configuração deste, permitindo a impressão do relatório em papel.

Destes sistemas apresentados pela empresa, o TranScan XL é aquele que se parece mais nas funcionalidades apresentadas com o sistema proposto para esta tese. Apesar de não haver muita informação a nível técnico sobre este produto, a instalação deste no veículo requer cablagem para alimentação e é realizada unicamente na zona de armazenagem, não sendo possível assim a monitorização a partir do habitáculo durante a viagem. Também é de salientar ainda que nenhum destes sistemas verifica e regista a abertura e fecho da porta da zona de armazenagem.

1.4- Estrutura da dissertação

Para além desta introdução, esta dissertação será composta por mais sete capítulos. No capítulo dois serão estudadas soluções para um protocolo com fios com vista a resolver o problema da impossibilidade de troca de dados via protocolo *wireless* para fora da zona de armazenagem dos produtos congelados.

No terceiro capítulo serão estudados soluções de protocolos *wireless* existentes tanto para a parte de troca de mensagens dentro da zona de armazenamento dos produtos, como também para a parte de envio e recepção de informação do veículo para o exterior.

O quarto capítulo visa, com base nos protocolos escolhidos para os vários subsistemas, especificar uma arquitectura do sistema e definir os integrantes deste com vista a montagem de um protótipo.

No quinto capítulo faz-se referência ao TinyOS, o qual foi utilizado como base na construção da aplicação concebida, bem como os principais algoritmos para toda a troca e concepção dos dados na rede de sensores *Zigbee*.

No sexto capítulo explica-se como se implementa nos microcontroladores PIC18F258 da Microchip o barramento CAN.

No sétimo capítulo demonstra-se o software elaborado para com vista a troca de dados entre o veículo e o exterior.

Por último as conclusões gerais do trabalho são feitas no oitavo capítulo.

Capítulo 2

Protocolos de comunicação com fios

Neste capítulo visa-se essencialmente a escolha de uma solução para envio dos dados recolhidos de dentro da câmara frigorífica para o habitáculo do veículo. Será para isso feito um levantamento dos principais protocolos de comunicações existentes hoje em dia, e dar-se-á prioridade a parâmetros como a simplicidade, popularidade e robustez protocolar para a decisão da opção a ser tomada.

2.1- RS232

O RS232 é um protocolo de comunicação de dados de interface série assíncrono [2], utilizado em muitos tipos de equipamentos, como computadores, modems, impressoras, microcontroladores, programadores de EPROM, e numa série de outros dispositivos. Foi introduzido pela *EIA* (*Electronics Industry Association*) em 1962 [3] com o objectivo de transportar informações entre dois dispositivos separados de curtas distâncias. A norma associada a este protocolo define as características do sinal eléctrico (e.g. níveis de tensão, comprimento do cabo), características mecânicas, conectores e configuração dos pinos [4]. Os níveis de tensão admissíveis variam entre 3 a 12V e -3 a -12V. O comprimento máximo do cabo é um dos pontos mais discutidos no mundo do RS232, estando este dependente da taxa de transferência de dados utilizada. Embora possa atingir maiores distâncias [2], este protocolo está classificado como sendo para distâncias pequenas, tipicamente até 15m.

Embora os cabos típicos de RS232 tenham 9 pinos (DB9S) ou 25 pinos (*D-type*) (Figura 2.1), são precisos apenas 3 para uma comunicação (um de transmissão, um de recepção e outro de referência ou massa) que permitem comunicação simultânea nos dois sentidos (*Full-Duplex)*. Os outros pinos são utilizados para funções especiais previstas pelo protocolo. RS-

232 é simples, universal, de baixo custo de implementação, bem compreendido e apoiado. Por essas razões, apesar de ser um protocolo que data desde a década de 60, mantém-se amplamente utilizado ainda pela indústria.

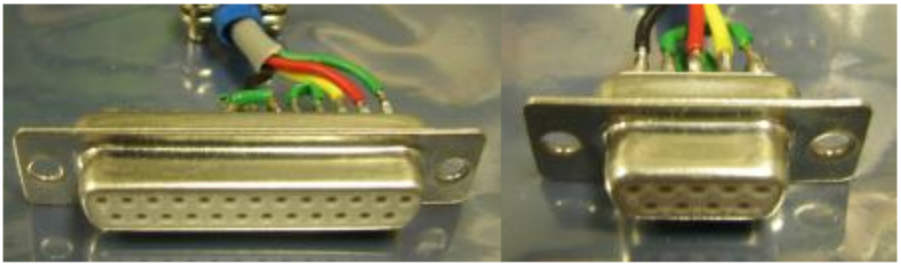


Figura 2.1- Cabos RS232 (D-type e DB9S)

2.2- RS422

O RS422 é muito conhecido como uma evolução do RS232. Este protocolo série destina-se a redes com ligação ponto a ponto (tal como o RS232) ou *Multi-drop* [8] (um emissor e vários receptores) usando condutores trançados entre si, transmitindo um sinal diferencial (como acontece no RS485) [9]. Com esta forma de transmissão de dados, este protocolo já consegue assim uma melhor imunidade ao ruído do que o RS232, e também uma transmissão a maior distância, permitindo atingir até 1200 m de distância. Neste protocolo é definido já tanto a camada física como a camada de dados, simplificando assim a utilização deste.

2.3- RS485

O RS485 é também conhecido como norma TIA/EIA-485. Foi publicado pela *Telecommunications Industry Association/Electronic Industries Alliance* (TIA/EIA). É um protocolo série que especifica as características eléctricas dos emissores e receptores para uso em comunicações multiponto. Assim este protocolo especifica somente a camada física, não estando a camada de sinalização/dados definida. O RS485 é caracterizado como um protocolo de transmissão de dados bidireccional *half-duplex* (quando é usada uma ligação com apenas um par de fios condutores trançados entre si), ou full-duplex (usando 4 fios condutores trançados). Este protocolo suporta até 32 dispositivos (nós) interligados entre si, podendo ir até 256 (dependendo da impedância associada aos dispositivos) [5]. Uma vez que utiliza uma linha diferencial de par trançado, o RS485 consegue transmitir a distâncias relativamente grandes (até 1200 metros) e é bastante robusto contra o ruído. É de referir ainda que este protocolo oferece velocidades relativamente altas de transmissão, dependendo do comprimento da linha (35 Mbit/s até 10 m e 100 kbit/s em 1200 m) [6].

2.4- CAN

CAN é acrónimo de "*Controller Area Network*", e é um protocolo de comunicação (ISO 11898) série síncrono que foi desenvolvido pela BOSCH em 1983 [5]. É um protocolo classificado como de grande eficiência e segurança, sendo muito usual no domínio da electrónica automóvel (controlo de motor, sensores, sistemas anti-derrapagem, entre outros.) [7]. Devido a sua comprovada fiabilidade e robustez também é utilizado em aplicações industriais em sistemas de controlo distribuído em tempo real. Para além disso, o protocolo permite ainda uma comunicação a grandes distâncias, como se pode constatar na tabela abaixo.

Tabela 2.1- Débito em função do comprimento da linha em *CAN*

Comprimento do barramento (m)	Taxa de transferência (kbit/s)
30	1000
100	500
250	250
500	125
1000	62.5

A camada física deste protocolo é, como no RS485, baseada num par de fios condutores que se encontram em par trançado diferencial (CANH e CANL), atenuando fortemente os efeitos causados por interferências electro-magnéticas. A camada de dados ao contrário do RS485, está completamente definida também, estando contemplado nesta a definição de prioridade de mensagens, detecção e sinalização de erros, distinção entre erros esporádicos e falhas permanentes na transmissão, flexibilidade de configurações entre outras funcionalidades.

2.5- Powerline DC

A tecnologia *Power Line* tem recebido uma crescente atenção nas últimas décadas devido aos benefícios inerentes, principalmente relacionados à redução do número de cabos e custos associados a estes. *Power Line Communication* (PLC) consiste em efectuar comunicação de dados através de cabos condutores de energia eléctrica já existentes. Esta tecnologia foi primeiro utilizada em empresas de energia, e desde os anos 80 também na domótica [10]. No entanto, a sua utilização em linhas de transmissão de energia eléctrica contínua (DC) de baixa tensão tem tido pouca relevância, havendo por isso muito pouca

informação ainda, e a ser preciso ser efectuado muito estudo, desenvolvimento e normalização nesta área de forma a poder torná-la numa possível solução para comunicação.

2.6- Protocolo Escolhido

Como foi dito no início deste capítulo, o protocolo a ser escolhido para resolver o problema da comunicação dos dados da zona de armazenagem dos produtos congelados para o habitáculo do veículo deveria ter como principais características a simplicidade, popularidade e a robustez protocolar. Neste sentido, de todas as possibilidades apresentadas faz sentido que se use o protocolo CAN, que foi criado muito com o propósito de um dos domínios em que este sistema se encaixa (domínio automóvel).

É de registar aqui que uma solução via *Powerline DC* também poderia ser bastante interessante de ser utilizada, já que como requisitos iniciais, este sistema pretendia ser de fácil instalação, recorrendo para isso ao uso de protocolos *wireless*, evitando com isso a cablagem. Esta solução contudo não será escolhida devido a este protocolo estar ainda em pleno desenvolvimento hoje em dia, mas aponta-se como uma solução interessante a ser estudada em separado, podendo num futuro vir a integrar este mesmo sistema.

2.7- Detalhes Técnicos do CAN

Como já foi dito, o CAN foi um protocolo concebido inicialmente para o uso na indústria automóvel. Apesar disso, actualmente é um dos protocolos mais utilizados em diversos outros tipos de sistemas para a troca de dados entre dispositivos. Isto é devido fundamentalmente a possibilidade de poder funcionar em ambiente electricamente hostis com uma elevada eficiência a um débito consideravelmente elevado.

2.7.1- Camadas OSI definidas

Neste protocolo estão definidos duas camadas, a camada física e a camada de enlace, como se apresenta na Figura abaixo.

Figura 2.1- Modelo de camadas OSI definidos em CAN

A camada física é responsável pelos aspectos físicos da transmissão dos bits no barramento, como por exemplo a codificação/descodificação de bits ou a sincronização entre nós. A camada de Enlace define o formato das mensagens, a prioridade do acesso ao barramento e disponibiliza mecanismos de detecção e prevenção de falhas.

2.7.2- Transmissão dos bits

O barramento CAN é baseado em dois fios, um o CANH e o CANL. A diferença de potencial nestes dois condutores é que dão o nível lógico alto e baixo no barramento, como se ilustra na Figura 2.2.

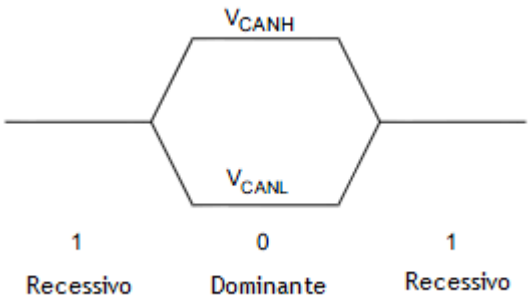


Figura 2.2- Representação dos níveis lógicos em CAN

A representação de um “0” bem como de “1” lógico tem uma denotação própria atribuída pelo protocolo CAN, denominando-se como bits recessivos e dominantes, respectivamente. Como se pode ver na Figura 2.2, quando o bit é recessivo significa que a diferença de potencial entre o sinal CANH e CANL é nula. Por outro lado quando estes apresentam uma diferença de potencial formam um bit dominante. É devido em parte a este

conceito a grande robustez do CAN as interferências, pois quando há interferências electromagnéticas, são sentidas em ambos os sinais, causando flutuações dos sinais para o mesmo sentido e com a mesma intensidade, permanecendo-se assim a diferença de potencial entre os condutores inalterada [32].

2.7.3- Tramas CAN

Na figura apresenta-se a estrutura da trama de dados (*Data Frame*) em CAN 2.0A (modo normal) e CAN 2.0B (modo identificador extendido).

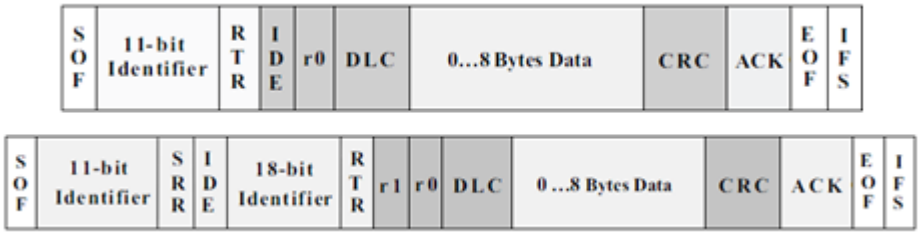


Figura 2.3- Formato das tramas CAN 2.0A (acima) e CAN 2.0B (abaixo).

O significado dos diversos campos destas tramas é os descritos:

- SOF (*Start of Frame)* - Representa a transmissão de um bit dominante para o barramento, indicando assim o início da transmissão da trama.
- *Identifier*- campo identificador do transmissor da trama. Para além disso estabelece prioridade das tramas no barramento. Quanto menor for o ID maior é a prioridade.
- RTR (*Single Remote Transmission*)- campo de um bit que quando preenchido com um bit dominante significa que a informação que está a ser enviada foi solicitada por outro nó.
- IDE (*Identifier Extended*)- bit que quando dominante simboliza que a trama enviada é do tipo CAN 2.0B.
- R0- bit reservado
- DLC (*Data Length Code*)- campo de 4 bits que indica a quantidade de dados (em bytes) enviados no campo de dados da trama.
- CRC (*Cyclic Redundancy Check*)- campo que apresenta um CRC baseado num polinómio de grau quinze para controlo de erros e ainda um campo delimitador.
- ACK- é um bit deixado a recessivo por parte de quem esta a enviar a mensagem e deve ser posto a dominante por todos os nós receptores caso tenham recebido a trama sem erros. O campo ACK é constituído por dois bits, um é o ACK e o outro é um delimitador.

- EOF (*End Of Frame*)- campo de 7 bits que indica o fim da trama e se existiram algum erro de *bit-stuffing*.
- IFS (*Interframe Space*)- campo de 7 bits que contem o tempo exacto tempo para o controlador mover a mensagem recebida para o buffer para ela destinado.
- R1- bit reservado tal como o r0.
- SRR (*Substitute Remote Request*)- bit que substitui o campo RTR nas tramas CAN2.0A, preenchendo esse campo com um bit recessivo.
- IDE (*Identifier Extension*)- campo de um bit que indica se a trama que esta a ser enviada tem campo de endereço extendido ou não.

Em CAN existem mais três tipos de tramas: a *Remote Frame*, a *Overload Frame* e a *Error Frame*. A *Remote Frame* é utilizada por um nó para solicitar a transmissão de dados a outro nó. A *Error Frame* é uma trama especial do CAN que é transmitida quando um nó detecta um erro numa trama, fazendo-se assim que o transmissor automaticamente retransmita a trama. Finalmente a *Overload Frame* é uma trama transmitida por um nó quando este está demasiadamente ocupado, dando assim os outros dispositivos mais tempo até a próxima transmissão [31].

2.7.4- Definição de prioridades em CAN

Como já foi dito acima, quanto menor for o valor do campo de identificação maior é a prioridade da mensagem. Assim um identificador constituído só por zeros é o que tem maior prioridade na rede. Quando dois dispositivos iniciam uma transmissão simultaneamente, o nó que enviar um bit dominante enquanto o outro está a enviar um bit recessivo ficará com o controlo do barramento, porque o bit dominante subscreve sob o bit recessivo. É por essa razão que os controladores CAN têm as saídas ligadas a sua própria entrada (para ver se quando esta a enviar perdeu o controlo do barramento).

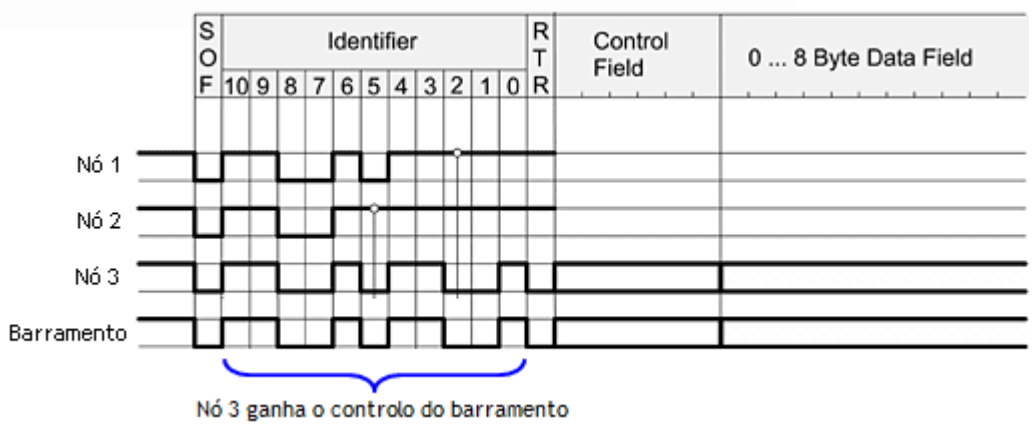


Figura 2.4- Prioridade de mensagens em CAN

2.7.5- Mecanismos de detecção de erros em CAN

Além da transmissão diferencial em CAN que confere quando imunidade as interferências, outras das características que faz este protocolo ser tão robusto é os mecanismos de detecção de erros que apresenta. Assim em CAN existem detecções de erros por:

- Monitorização ao nível do bit - em que cada dispositivo que comunicar esta constantemente a verificar se o bit que enviou é igual aquele que leu (a não ser quando esta a enviar o ID para assim se definir as prioridades).
- Verificação da trama - os campos CRC e ACK das tramas CAN apresentam um campo delimitador que são bits fixos de valor conhecido. Sempre que um receptor encontra um ou mais bits que não deveriam estar presentes entre estes campos é activado um erro.
- *Bit Stuffing* - em todos os campos das tramas CAN (excepto ACK e EOF) é realizado um *bit stuffing* por parte do emissor. Logo se um receptor detectar 6 bits consecutivos de mesmo valor lógico, este sabe que esta presente um erro.
- Verificação cíclica de redundância - O dispositivo que transmite calcula um valor originado por um polinómio a partir dos bits que constituem a trama até ao campo de dados e envia-o no campo logo a seguir (CRC). O dispositivo receptor calcula esse mesmo CRC com os bits recebidos e compara-o com o CRC enviado na trama. Caso sejam diferentes o receptor sabe que houve erro de transmissão.
- Erro de ACK- Este erro acontece sempre que o dispositivo que esta a enviar não recebe um bit dominante nesse campo. Quando o dispositivo emissor lê um bit dominante neste campo, sabe que pelo menos um nó recebeu a trama sem erros.

Em CAN é possível ainda distinguir falhas temporárias (devido por exemplo a uma interferência) de falhas permanentes (devido a um defeito por exemplo). Para isso os controladores contam com dois contadores de erros, o TEC (*Transmit Error Counter*) e o REC (*Receive Error Counter*), actuando segundo o diagrama da Figura 2.5.

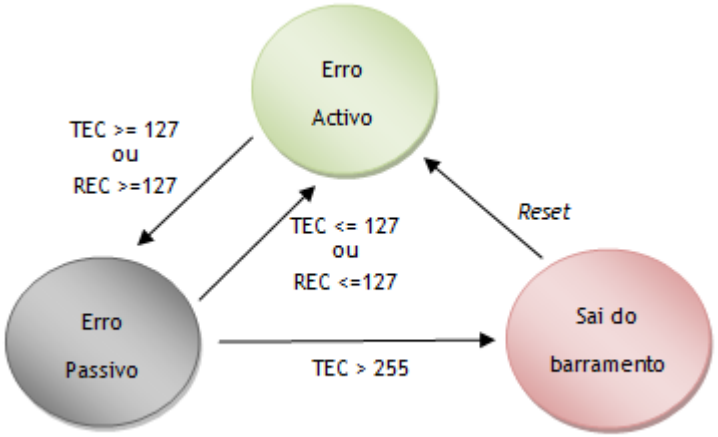


Figura 2.5- Diagrama de actuação de um dispositivo CAN à detecção de falhas.

Cada erro de transmissão ou de recepção faz com que seja enviado para o barramento uma trama de erro e também incrementar o respectivo contador (TEC ou REC). Se o dispositivo apresentar num desses contadores um número maior ou igual a 127, o nó entra no estado de Erro Passivo, estado esse onde deixa de ser enviado tramas de erro. Se o número de erros pelo contador ultrapassar os 255 o dispositivo entra no estado *Bus Off*, onde apenas consegue receber mensagens. Sempre que uma mensagem é correctamente enviada ou recebida, é descontado os valores acumulados nos respectivos contadores [32].

Capítulo 3

Protocolos de comunicação sem fios

Neste capítulo será abordado e comparadas algumas das soluções protocolares sem fios usadas hoje em dia. No final será escolhida uma solução para a troca de mensagens dentro da zona de armazenamento dos produtos congelados e para troca de dados do veículo transportador com o exterior.

3.1- Zigbee

Zigbee é um protocolo baseado na norma IEEE 802.15.4 que se enquadra numa especificação para áreas de redes pessoais sem fio (WPANs). Foi criado para ter um baixo débito (250 kbit/s) de transmissão de dados a curtas distâncias e um consumo reduzido [17]. A norma IEEE 802.15.4 em conjunto com o *Zigbee* define um protocolo de rede sem fios ideal para a implementação numa vasta gama de aplicações de baixo custo, baixo consumo, e de controlo e monitorização dentro de casa e ambiente industrial [17]. É por isso a par do Bluetooth um dos protocolos de comunicação sem fios de baixo consumo mais populares hoje um dia no mercado. Possui uma pilha protocolar de implementação bastante simplificada, conduzindo a interfaces de baixo custo. Tem ainda a possibilidade de suportar um elevado numero de nós por rede (máximo de 65535 dispositivos) quando comparado com o *Bluetooth* (máximo de 8) e de ainda apresentar consumos relativamente bem mais baixos quando comparado com este.

3.2- Bluetooth

Bluetooth é um protocolo de comunicação que define uma camada sobre a norma IEEE 802.15.1 [10], muito usual hoje em dia para troca de informações entre dispositivos como telemóveis, portáteis, computadores, impressoras, câmaras digitais através de uma frequência de rádio de curto alcance [12]. É por isso uma especificação para áreas de redes pessoais sem fio (WPANs). O *Bluetooth* foi um padrão de comunicação primariamente projectado com vista a aplicações de baixo consumo de energia, baixo alcance e que não necessitem de um débito considerável de transferência de dados (ver Tabela 3.1).

Tabela 3.1- Débitos atingidos no *Bluetooth*

Versão	Débito
Versão 1.2	1 Mbit/s
Versão 2.0	3 Mbit/s
Versão 3.0	24 Mbit/s

O *Bluetooth* está ainda normalizado em três classes (Tabela 3.2), que diferem uma das outras na potência que é irradiada, o que tem afectação directa não só sobre o alcance atingido mas como também na potência de transmissão e consumida pelos próprios equipamentos.

Tabela 3.2- Classe previstas pelo *Bluetooth*

Classe	Potência máxima	Alcance aproximado
Classe 1	100 mW (20 dBm)	Até 100 m
Classe 2	25 mW (4 dBm)	Até 10 m
Classe 3	1 mW (0 dBm)	Até 1 m

3.3- Wi-Fi

Wi-Fi é um dos mais populares padrões de comunicação sem fio no mercado. É uma norma internacional que descreve as características de uma rede de área local sem fios (WLAN). Wi-Fi é tecnicamente o nome da certificação dada pela Wi-Fi *Alliance* a compatibilidade entre dispositivos que utilizam a norma IEEE 802.11. No entanto, o termo tornou-se tão amplamente utilizado tendo permitido uma generalização. Na sua fase inicial, a tecnologia Wi-Fi era quase exclusivamente utilizado para ligar computadores portáteis sem fios à Internet através de redes locais (LANs), mas, graças à flexibilidade que a tecnologia proporciona, a tecnologia Wi-Fi é agora encontrada também numa série de dispositivos

electrónicos, como receptores de *home theater*, consolas de jogos, leitores de DVD, câmaras digitais e até dispositivos GPS.

Tipicamente na norma 802.11 reserva as camadas de mais baixos níveis do modelo OSI para uma conexão sem fios que utiliza ondas electromagnéticas. A camada física (Physical *layer*) define a modulação das ondas de rádio e as características de sinalização para transmissão de dados, enquanto a camada de ligação de dados (*Data link layer*) define (as regras (protocolos) que permitem transferir unidades de informação usando os serviços da) camada física [14].

A especificação do padrão IEEE 802.11 é composta por mais de 20 padrões diferentes. Os padrões mais populares hoje em dia são 802.11b, 802.11g e 802.11n que são utilizados na maioria dos dispositivos Wi-Fi comercializados. Na tabela abaixo apresenta-se alguns desses padrões assim como algumas das suas principiais particularidades.

Tabela 3.3- Alguns dos padrões da norma 802.11

Standard	Frequência de Operação	Débito Máximo	Alcance
802.11a	5 GHz	50 Mbit/s	10 m
802.11b	2.4GHz	11 Mbit/s	30 m
802.11g	2.4GHz	54 Mbit/s	30 m
802.11n	2.4GHz	300 Mbit/s	50 m

Pode-se ver que o Wi-Fi pela sua popularidade, alcance e velocidade de transferência de dados é uma solução muito interessante para rede sem fios. No entanto há que ter em conta que a nível de consumo energético não é de longe das soluções de menor consumo encontradas no mercado [13] para este tipo de redes.

3.4- MiWi

MiWi é uma pilha protocolar proprietária, desenvolvida pela Microchip, para aplicações de curto alcance de redes sem fios, baseada no padrão IEEE 802.15.4. Enquadra-se portanto no âmbito de redes de área pessoal sem fio (WPAN's). A pilha protocolar MiWi foi criada com vista a fornecer uma pequena alternativa ao padrão Zigbee (também baseado na norma IEEE 802.15.4), apresentando em relação a este uma optimização no consumo, sendo também um protocolo de baixo débito, mas intitula-se como sendo uma solução relativamente mais barata [18].

3.5- Dash7

DASH7 é uma tecnologia de redes sem fios que evoluiu a partir da norma ISO 18000-7 (mesma que a tecnologia RFID). Este protocolo opera na banda não licenciada do espectro dos 433 MHz, e intitula-se como um protocolo de ultra-baixo consumo de energia, com consumos típicos de seis vezes inferior quando comparados com soluções equivalentes usando *Zigbee* [15], o que permite uma grande longevidade na vida da bateria destes dispositivos. Esta tecnologia de redes de sensores sem fios foi criada originalmente para uso militar [16] e só agora é que esta a começar a ser utilizada em aplicações para fins comerciais, havendo ainda muito pouca informação e dispositivos que integrem este protocolo de comunicações. De destacar ainda características como ter um alcance num raio de ate 2 km, transferência de dados a um debito de até 28 kbit/s, e uma grande capacidade de penetração através de obstáculos.

3.6- Protocolos escolhidos

Da análise destas soluções protocolares sem fios pretendia-se a escolha de um protocolo para a troca de mensagens dentro da zona de armazenamento dos produtos congelados e outro para troca de dados do veículo transportador com o exterior. Podemos ver na tabela abaixo um estudo comparativo de algumas características protocolares das normas aqui apresentadas.

Tabela 3.4- Análise de soluções *wireless*

Norma	Frequência	Encriptação	Consumo	Débito
Zigbee	868/915MHz, 2.4GHz	Sim	Baixo	250Kbits/s
MiWi	2.4GHz	Sim	Muito Baixo	-
Dash7	433MHz	Sim	Muito Baixo	28Kbits/s
WIFI (802.11g)	2.4GHz	Sim	Médio	54 Mbit/s
Bluetooth 2.0	2.4GHz	Sim	Baixo	3 Mbit/s

Para a zona de armazenamento a escolha recaiu sobre o *Zigbee*, devido a este ser um dos protocolos que consomem menos energia (nesta zona pretende-se um protocolo de pouco consumo energético para aumentar a longevidade das baterias) aquele que esta mais normalizado entre os dispositivos. A tecnologia MiWi tem características idênticas ao *Zigbee* e poderia ser uma escolha possível. No entanto por ser um protocolo privado da Microchip destina-se a ser única e exclusivamente usada por microcontroladores desta mesma empresa. Logo se quiséssemos usar esta tecnologia em outro microcontrolador teria que ser paga uma licença, o que torna a utilização desta tecnologia inviável.

Para a comunicação do veículo com o exterior a solução recai sobre o Wi-Fi, porque apesar de este não ser um protocolo económico a nível energético, considera-se que na zona do habitáculo do veículo é relativamente simples a ligação de um cabo de alimentação do veículo (ligado a bateria deste) a um dispositivo que integre Wi-Fi, aproveitando assim a vantagem de usarmos este protocolo tão familiarizado nos dias de hoje, e de consequentemente assim, só precisarmos no exterior de um computador com Wi-Fi para a troca de dados com este.

3.7- Zigbee/802.15.4 (Detalhes Técnicos)

Enquanto o IEEE 802.11 estava preocupado com características tais como grandes velocidades de transferências de dados, longas distâncias de alcance, possibilidade do usuário de uma rede obter conectividade em áreas fora da localidade geográfica onde está registado (*Roaming*) e reencaminhamento de mensagens, o IEEE 802.15 surgiu com vista a concessão de redes tendo em foco essencialmente o baixo custo, baixo consumo de potência e curto alcance. Desta norma surgiram então várias classes de WPAN's (*Wireless Personal Area Networks*) diferenciadas entre si pela taxa de transmissão de dados, consumo de potência e qualidade de serviço (QoS). O 802.15.4 foi uma das classes que surgiu, e que se destina a servir um conjunto de vários tipos de aplicações (industriais, residenciais e medicinais) que necessitassem de muito baixo consumo de potência e que não necessitem de grandes velocidades de transferência de dados bem como grande QoS. Foi então que mais tarde surgiu a Zigbee Alliance, que é uma aliança constituída por mais de 200 empresas, oriundas de mais de 20 países distintos e que desenvolveu a pilha protocolar do Zigbee por cima da pilha criada pelo IEEE 802.15.4, como se ilustra na Figura 3.1- Camadas cridas pelo IEEE 802.15.4.

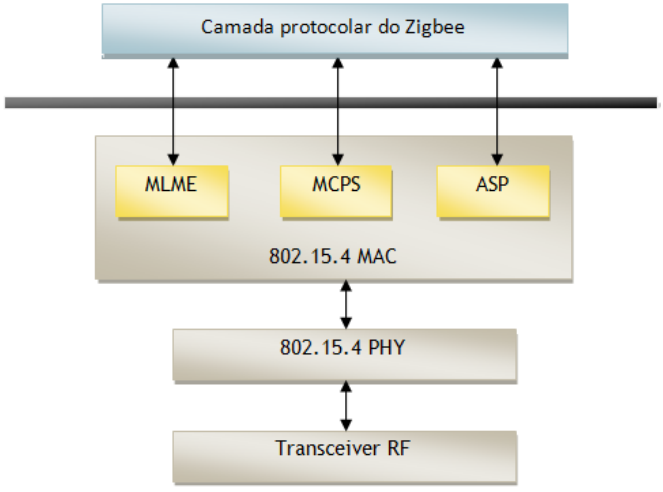


Figura 3.1- Camadas cridas pelo IEEE 802.15.4

3.7.1- IEE 802.15.4

Na norma IEEE 802.15.4 estão definidas duas camadas: a camada PHY e a camada MAC [35].

3.7.1.1- A camada Física

A camada Física (PHY) é responsável por funções como permitir a transmissão e recepção de mensagens através do canal físico RF. Das suas funções fazem parte também a activação e desactivação do transceiver, a detecção de energia (*ED – Receiver Energy Detection*), indicação da qualidade da ligação (*LQI – Link Quality Indication*) e selecção do canal de transmissão/recepção dos pacotes.

A detecção de energia (ED) é utilizada pelo algoritmo de selecção de canal, e retorna uma estimativa da potência do sinal recebido. A indicação da qualidade da ligação (LQI) é a caracterização da qualidade de recepção de um pacote.

É ainda na camada PHY que é especificado três bandas de frequências distintas de operação (Figura 3.2) [34]:

- Banda dos 868Mhz- opera contendo 1 canal apenas entre os 868Mhz e os 868.6Mhz, possibilitando uma transmissão a 20kb/s. Em termos de modulação, esta banda usa a BPSK (*Binary Phase Shift Keying*);
- Banda dos 915Mhz - opera contendo 10 canais entre os 902Mhz e os 928Mhz possibilitando uma transmissão a 40kb/s. Em termos de modulação, esta banda usa também a modulação BPSK;
- Banda dos 2.4Ghz - opera contendo 16 canais entre os 2400Mhz e os 2483.5Mhz possibilitando uma transmissão a 250kb/s. Em termos de modulação, esta banda usa a O-QPSK (*Offset Quadrature Phaseshift Keying*).

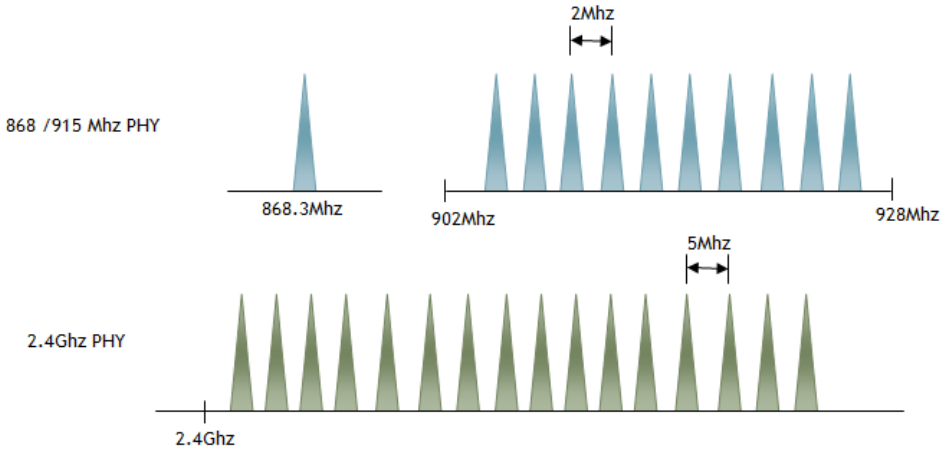


Figura 3.2- Diferentes bandas de frequências de operação

As tramas criadas e recebidas nesta camada, as PPDU (*Physical Protocol Data Unit*) apresentam o seguinte formato (Figura 3.3):

- SHR – contém um campo para sincronização de símbolo e de trama;
- PHR - contém informação do tamanho da trama
- PHY payload - que é a carga de comprimento variável que transporta a trama da camada MAC [33].

4 Octetos	1 Octeto	1 Octeto		Variável
Preâmbulo	SFD	Tamanho da trama (7 bits)	Reservado	PSDU
SHR		PHR		PHY payload

Figura 3.3 - Tramas formadas na camada física (PHY).

3.7.1.2- Camada MAC

À camada MAC cabe fundamentalmente o papel de controlar o acesso aos canais RF, utilizando mecanismos de prevenção de colisão CSMA-CA (*Carrier Sense Multiple Access - Collision Avoidance*), para o qual efectua comunicações com a camada inferior – camada PHY. Além disso, especifica o tipo de dispositivos permitidos na rede, a estrutura de tramas admissível e também está a seu encargo a sincronização e transmissão de tramas sinalização, permitindo a fiabilidade da operação. Das três subcamadas apresentadas na Figura 3.1, a MLME (*MAC Sublayer Management Entity*) é responsável pelos serviços e mecanismos de

gestão da camada, o MCPS (*MAC Common Part Sublayer*) é responsável por fornecer os serviços de dados básicos, como solicitações e envio de dados entre camadas, enquanto que o ASP (*Application Support Package*) é a interface utilizada para apoiar outras necessidades da camada da aplicação, como por exemplo, uma solicitação ao hardware para que entre num modo de baixa potência [37].
Na camada MAC são definidos quatro tipos de tramas (tramas MPDU):

- Uma trama de aviso, usada por um coordenador para transmitir avisos;
- Uma trama de dados, usada por todas as transferências de dados;
- Uma trama de reconhecimento (ACK), usada para confirmar a recepção de dados com sucesso;
- Uma trama de comando MAC, usada para trabalhar com controlo de transferências de todas as entidades MAC.

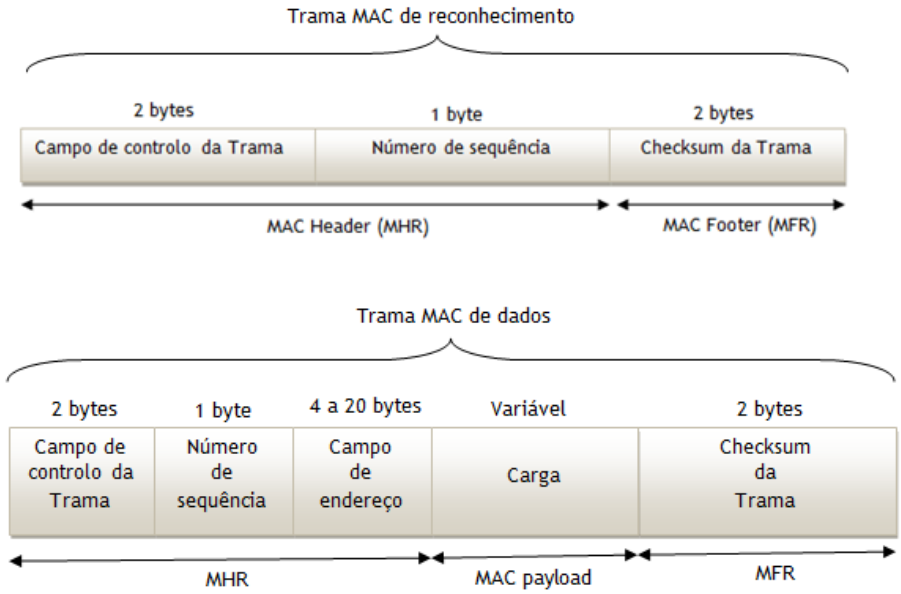


Figura 3.4- Exemplos de Tramas MAC

A trama MPDU consiste em um cabeçalho MAC (MHR), em uma carga MAC (MAC payload) e o fim de trama MAC (MFR). Estes campos têm informações específicas da camada MAC. O MHR é um campo que envolve um campo de controlo de trama, o número de sequência e o endereço de quem envia a informação. O campo MAC payload por sua vez contém os dados provenientes da camada acima, e é de tamanho variável. O MFR é um campo onde é apresentado um checksum de controlo de erros [33].

3.7.2- Protocolo Zigbee

Na Figura 3.5 ilustra-se a Camada de rede e a Camada de Aplicação definidas pelo Zigbee.

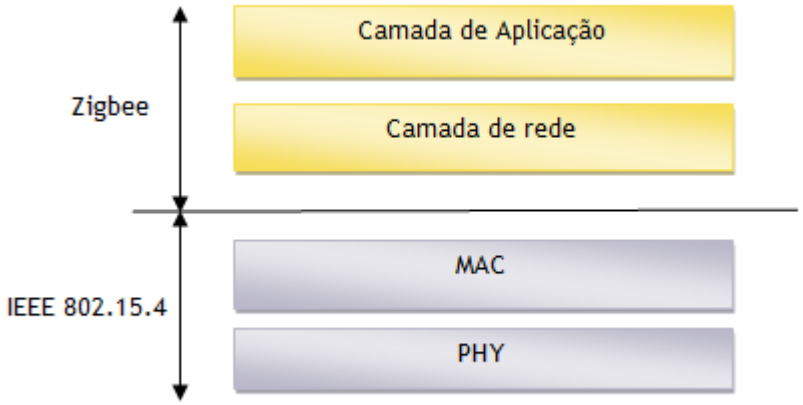


Figura 3.5- Camadas definidas no Zigbee

3.7.2.1- Camada de Rede

As principais funções da Camada de Rede é permitir o uso correcto da subcamada MAC e fornecer uma interface adequada para utilização pela próxima camada superior, ou seja, a camada de aplicação. Suas funcionalidades e estrutura são aquelas normalmente associadas às camadas de rede, ou seja, responsabilidade de início ou fim de ligação de um dispositivo à rede, descoberta de novos dispositivos nas vizinhanças (e o armazenamento de informações relativas aos mesmos), a atribuição de endereços (quando o dispositivo é o Coordenador). É ainda nesta camada que estão presentes os mecanismos de descoberta de rotas e encaminhamento da informação de acordo com a topologia actual da rede. Esse encaminhamento de informação é feito segundo o protocolo AODV (*Ad hoc On-Demand Distance Vector)*. Neste protocolo, com o objectivo de encontrar o dispositivo de destino, o dispositivo de origem transmite uma mensagem para todos os dispositivos vizinhos de solicitação de rota. Os vizinhos em seguida transmitem o pedido para os seus dispositivos vizinhos e assim sucessivamente até que o destinatário seja encontrado. Uma vez encontrado o dispositivo destinatário, este envia uma mensagem de resposta através do caminho mais curto de dispositivos até ao dispositivo remetente. Uma vez que recebida a resposta, o dispositivo remetente irá actualizar a sua tabela de encaminhamento para o endereço de destino com o caminho até chegar a esse dispositivo, associando um custo e esse caminho [38].

3.7.2.2- Camada de Aplicação

A Camada de Aplicação é a camada de mais alto nível definida pela especificação, e representa a interface final do protocolo ZigBee com o utilizador. Nesta camada estão contidas a maioria dos componentes adicionados pela especificação ZigBee. Esta contém as subcamadas:

- APS (*Aplication Support Sublayer*) que é responsável pelo reencaminhamento das mensagens na rede para os diferentes pedidos das aplicações em execução nos diversos nós. Esta inclui a actualização constante das tabelas de reencaminhamento das mensagens entre os dispositivos.
- ZDO (*ZigBee Device Object*) que pretende simplificar a gestão da rede efectuada pelas aplicações do utilizador. O ZDO contém o ZDP (Zigbee Device Profile) que pode ser considerada uma aplicação Zigbee projectada para a gestão da rede e não a troca de dados entre aplicações específicas. Esta aplicação fornece então comandos que permitem à aplicação do utilizador ter acesso a vários parâmetros de configuração da rede, como por exemplo o conteúdo das tabelas de reencaminhamento de mensagens, resultado da descoberta de dispositivos, entre outros. A subcamada ZDO fornece ainda controlo ao nível do nó local (sleep modes, controlo da potência de transmissão) e ao nível da rede (start / join, status da rede, etc) para qualquer tipo de nó.
- AF (*Application Framework*) que é responsável por fornecer uma interface aos objectos da aplicação, para que estes enviem e recebam dados. Os objectos da aplicação são definidos pelo utilizador/fabricante do dispositivo Zigbee. Cada objecto de aplicação é endereçado através do seu *endpoint*. Os *endpoints* variam entre o endereço 0 e o 255 sendo o 0 o endereço do ZDO e o 255 o de *broadcast* (comunicação com todos os dispositivos). A subcamada AF é a mais acima imediatamente antes da aplicação do utilizador, havendo até quem a considere como uma camada separada e não como subcamada da camada de aplicação.

Esta camada pretende assegurar uma correcta gestão e suporte para as diversas aplicações [38].

3.7.2.3- Topologias de redes

Numa rede Zigbee podem-se se encontrar dois tipos de dispositivos: Dispositivo com funções complexas (FFD) e dispositivos com funções elementares (RFD). Os dispositivos de funções complexas têm mais capacidade de memória e recursos para lidar com todo o tipo de tarefas do que os dispositivos de funções reduzidas. Estes dispositivos podem assumir o papel

de coordenador da rede (podendo assim iniciar a rede e controla-la). Suportam ainda o reencaminhamento de mensagens podendo trocar mensagens com routers, coordenadores e dispositivos das extremidades da rede (*end devices*) em contraste com os RFD que podem unicamente trocar mensagens com dispositivos routers ou com o coordenador.
Na Figura 3.6 ilustra-se as diferentes topologias de redes suportadas pelo IEEE 802.15.4.

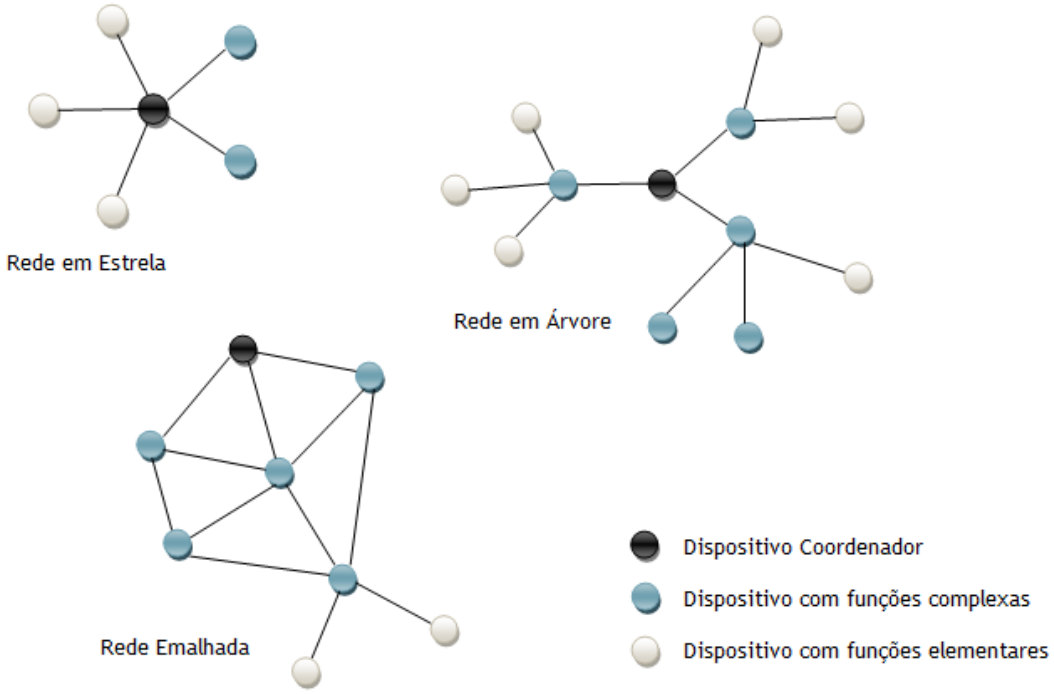


Figura 3.6- Topologias de rede suportadas pelo IEEE 802.15.4

Na topologia em estrela, a comunicação é estabelecida entre os dispositivos e um controlador central único, chamada de dispositivo coordenador da PAN. Na topologia de rede emalhada, existe também um coordenador PAN, mas em contraste com a topologia em estrela, qualquer dispositivo pode comunicar com outro qualquer já que este tipo de configuração permite que as mensagens sejam encaminhadas de dispositivo em dispositivo até chegar ao destinatário. A rede em árvore é um caso especial de uma rede emalhada em que grande parte dos dispositivos são de funções complexas e os que são de funções reduzidas estão conectados nas extremidades das ramificações. Qualquer dispositivo de funções complexas pode actuar como coordenador e disponibilizar serviços de sincronização a outros dispositivos e coordenadores. A vantagem desta estrutura em árvore é de se conseguir uma maior área de cobertura com o custo de um aumento da latência na recepção de mensagens [38].

Capítulo 4

Especificações e definição da Arquitectura

Pretende-se neste capítulo, após especificação protocolar da comunicação dos subsistemas, definir a arquitectura do sistema e especificar melhor o projecto, tanto a nível de funcionalidades bem como a nível dos dispositivos que virão a integrar este.

4.1- Arquitectura global do sistema

Depois de ter sido feito um levantamento de tecnologias de comunicações existentes, tanto baseadas em fios como sem fios, chegamos ao seguinte esquema para a arquitectura global do sistema (Figura 4.1).

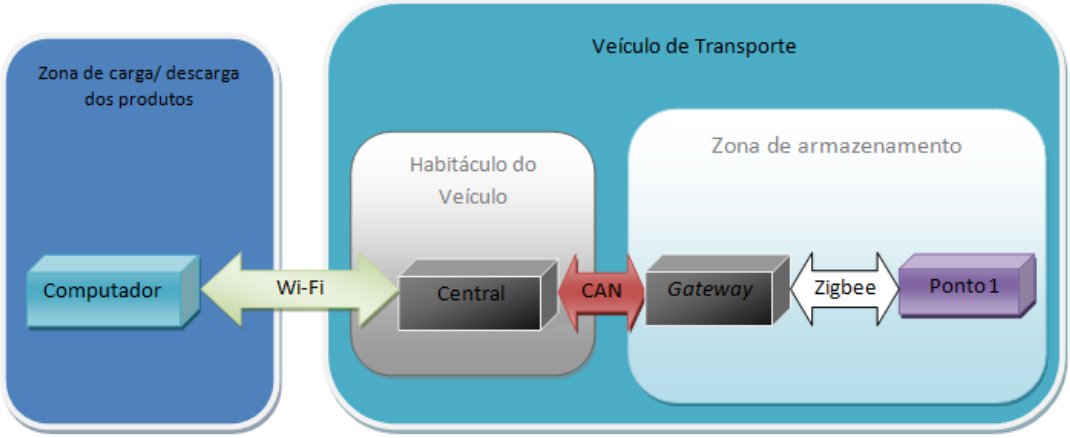


Figura 4.1- Arquitectura global do sistema

Como já tinha sido referido, dentro da zona de armazenagem dos produtos congelados, os pontos de medição poderão trocar informações com uma *gateway* através do protocolo *Zigbee*. A *gateway* por sua vez comunicará com o habitáculo do veículo, via protocolo CAN. As comunicações do veículo com o exterior (Zona de cargas e descargas dos produtos) ficarão então asseguradas pelo protocolo Wi-Fi.

4.2- Modos de Operação do Sistema

Foi já referido no primeiro capítulo algumas das funcionalidades do sistema, funcionalidades essas que nos permitiram chegar então a esta arquitectura global. Existe agora então a necessidade de uma melhor especificação dos modos de funcionamento do sistema.

Comecemos então por especificar três modos de operação em que o sistema pode-se encontrar:

- Modo de Carregamento dos Produtos
- Modo de transporte
- Modo de Descarga dos Produtos

O modo de Carregamento do Produtos corresponde exactamente a quando o sistema está a ser preparado para começar a sua missão de transporte. O modo de transporte corresponde a quando o sistema está em andamento na estrada. O modo de Descarga dos Produtos corresponde ao momento em que o sistema chega ao destinatário. Vai-se agora então definir que operações têm de ser feitas aquando destes modos de operação.

4.2.1- Modo de Carregamento dos Produtos

Como já foi dito anteriormente este modo de operação consiste em quando o veículo está a ser carregado com os produtos para iniciar o seu transporte. Aquando deste modo de operação, deverá ser possível através do computador (e de um *software* para o efeito):

- Escolher o ID do camião que se pretende configurar;
- Definir o período entre leituras de temperatura dentro da zona de armazenagem dos produtos de maneira a que se obtenha um ficheiro *log* que obedeça à norma;

4.2.2- Modo de Transporte

Neste modo de operação, o sistema deverá começar a efectuar a medição da temperatura com uma periodicidade igual a aquela a qual foi programado no Modo de Carregamento do Produtos. O valor da temperatura é então enviado logo após a medição ter sido efectuada para a *gateway* via *Zigbee*, sendo esta responsável por enviar via protocolo CAN o valor da medição recebida para a central. A central, que se encontra no habitáculo do veículo, é que deverá então registar o valor da temperatura e a hora em que ela ocorreu. Deverá ainda de acordo com o valor lido, sinalizar ao condutor por meio de informação visual (através de um monitor), a ocorrência de alarme caso esse valor de temperatura esteja fora dos valores normais. Não esquecer também que todas as aberturas/fechos da zona de armazenagem terão de ser registadas. Terá que se ter então um botão de pressão ligado a um dos nós de medição como já foi referido. Esse interruptor estará ligado a uma interrupção externa de maneira a que esse nó sempre que se abra e/ou feche a porta da zona de armazenamento de produtos, seja enviado essa informação para a central a registar.

4.2.3- Modo de Descarga dos Produtos

O sistema entra então neste modo sempre que chegar ao destinatário da carga. A central então aqui é responsável pela detecção da rede Wi-Fi associada ao cliente a que a carga se destina. Detectada então essa rede, a central deverá enviar o relatório de todas as ocorrências durante a viagem.

4.3- Escolha dos dispositivos integrantes do sistema

Neste ponto pretende-se então efectuar um levantamento dos dispositivos necessários a concepção de um protótipo, protótipo este que esteja de acordo com a arquitectura global a que chegamos para o sistema.

4.3.1- *Gateway* e Pontos de medição

Por simplicidade e uma vez que o tamanho do sistema é considerável, optou-se por plataformas de hardware existentes no mercado que já integrassem microprocessador, baterias, módulos de comunicações tudo num dispositivo. Encontrou-se como possíveis soluções dois nós autónomos:

- MicaZ da empresa Crossbow
- FireFly WSN Platform

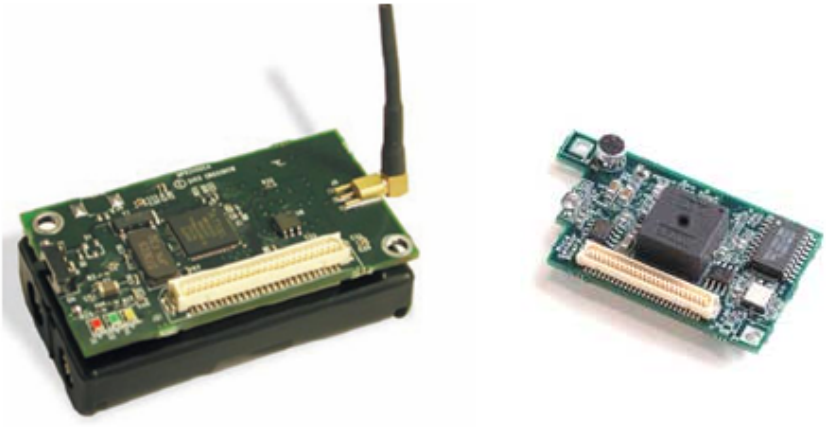


Figura 4.2- MicaZ e sua placa de sensores (MTS310)

Figura 4.3- FireFly WSN Platform com board expansão de sensores [20]

Tanto o FireFly WSN Platform (Figura 4.3) como o MicaZ (Figura 4.2) constituem plataformas *wireless* (IEEE 802.15.4) para integrar redes de sensores de baixo custo e de baixo consumo.

Estes dois dispositivos a nível de hardware e arquitectura não apresentam grandes diferenças (tamanhos semelhantes, ambos com microcontroladores da Atmel *low power*, ambos com uma duração de baterias que pode passar um ano de duração). Ora por razões de económicas (por já existir na Faculdade), a *gateway* e o Ponto 1 ao Ponto n, serão constituídos por módulos Micaz.

Os módulos Micaz utilizam um microcontrolador Atmel ATmega 128L 8-bit e um transceiver de IEEE 802.15.4 que garante as comunicações sem fios [21]. Esta plataforma possui ainda a vantagem de ter várias placas de expansão. Por exemplo a *board* MTS310

adiciona ao módulo vários tipos de sensores, entre eles um sensor de temperatura [22]. Após consulta da folha de características desse sensor de temperatura, verificou-se que a precisão deste era de 0.2 ºC e que consegue medir valores de temperatura compreendidos entre os -40 ºC e os 70 ºC, o que faz com que este seja uma escolha que cumpre com os requisitos da norma. Será então utilizado o Micaz com essa placa de expansão, placa esta que contém o sensor de temperatura que será usado para efectuar as medições.

4.3.2- Comunicações por CAN

Com vista a comunicação da *gateway* contida dentro da zona de armazenamento com a central (contida no habitáculo do veículo) via CAN, utilizou-se duas placas já existentes na faculdade, as DETPIC's (Figura 4.4), já que a *gateway* (constituída pelo Micaz), não dispunha de um controlador CAN incorporado.

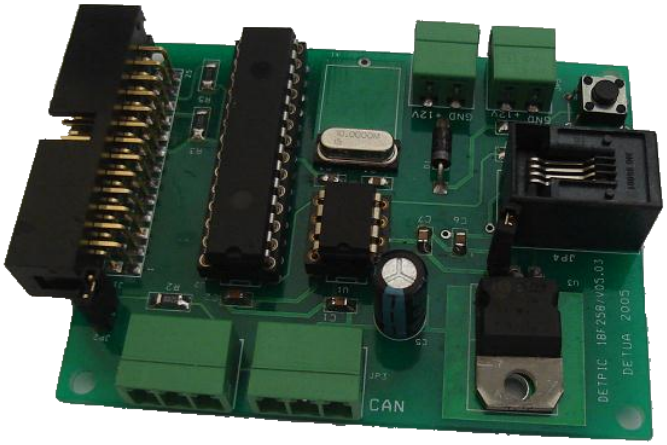


Figura 4.4- DETPIC

Estas placas são constituídas por um PIC18F258 (PIC que contém um módulo CAN integrado) e um controlador PCA82C250 para acesso ao barramento. Esta placa contém ainda uma ficha de 26 pinos para acesso directo a todos os pinos do PIC, onde se extraiu os sinais Rxd, Txd e GND para comunicação via RS232 com o MicaZ. Assim, e de acordo com o esquema de arquitectura do sistema apresentado na Figura 4.1, este é um dos elementos integrantes tanto da *gateway* como da central, ligando-se a ambas via RS232.

4.3.3- Central

Para a composição da central, foi escolhido uma solução computacional mais complexa, já que esta como requisitos mínimos impunha possuir Wi-Fi (802.11 (b/g/n)), uma porta RS232 e uma saída VGA para monitor. Teria de se ter em conta também que o orçamento para esta central protótipo não era muito grande (até €200). Começou-se então por procurar *motherboards*, de preferência já com placa gráfica e processador embutidos. Encontrou-se então a *motherboard* AT5NM10-I da ASUS (Figura 4.5).

Figura 4.5- *Motherboard* Asus AT5NM10-I

Esta *motherboad* apresentava como principais especificações:

- CPU Intel® Atom™ processor D510 (Integrado);
- 1 slots para memória RAM DDR-2 800/667;
- 1 slot PCI;
- Placa gráfica Integrated Intel Graphics Atom D510 com saída VGA;
- 2 interfaces SATA para disco;
- 2 portas RS232;
- 8 portas USB.

Comprada esta *motherboard*, a preocupação a seguir era de dotá-la de comunicações via 802.11. Para isso após uma pesquisa encontrou-se um adaptador para comunicações *wireless*, um Asus PCI-G31 (Figura 4.6), o qual dispunha de interface PCI e *drivers* para Windows, MacOS e Linux, o que não causaria problemas para ser usado na nossa *motherboard*.

Figura 4.6- Asus PCI-G31

A seguir comprou-se então um disco rígido para a *motherboard* (Samsung 2,5" SATA 160Gb 5400rpm), bem como uma memória RAM (Kingstom KVR800D2N6/1GB) (Figura 4.7).

Figura 4.7- RAM Kingstom e disco rígido Samsung

Só faltava agora mesmo uma fonte de alimentação para a nossa central. Como ela era para actuar dentro de um veículo não se poderia utilizar uma fonte convencional 230V AC. Pesquisou-se então por fontes 12/24V para o efeito e encontrou-se então o M1-ATX para operar juntamente com a caixa VoomPC-2 ™ car PC enclosure (Figura 4.8).

Figura 4.8- M1-ATX e VoomPC-2

Com isto já teríamos o nosso protótipo da central concluído. O problema foi que só o M1-ATX e a caixa VoomPC-2 sozinhos quase passavam o nosso orçamento. Achou-se então que

a não angariação destes dispositivos não seria crítico para no final comprovarmos a funcionalidade do nosso protótipo, pois estes não tinham qualquer tipo de interferência nos objectivos para este trabalho. Comprou-se por isso uma fonte tradicional, já que mesmo assim conseguiríamos simular os resultados pretendidos para o projecto. A fonte comprada foi uma 2HIX JET 450W ATX Power Suply (Figura 4.9).

Figura 4.9- Fonte de alimentação 2HIX JET 450W

4.4- Conclusão

Foi definido até aqui a arquitectura do sistema a ser implementada e as tecnologias de comunicação integrantes deste. Com base nessa arquitectura e nas especificações iniciais do sistema foram também definidos modos de operação do sistema, com o intuito de se definir a partida os vários estados em que o sistema se pode encontrar, bem como de que forma deverá interagir de maneira a que os resultados no final sejam os pretendidos, e que estejam em conformidade com a especificação inicial para este. Foi elaborado ainda neste capítulo um levantamento de todos os dispositivos que deverão ser adquiridos de modo a realização do protótipo.

Capítulo 5

Rede de sensores MicaZ

Neste capítulo descrevemos todos os algoritmos utilizados na concepção da rede de sensores MicaZ. Faz-se uma breve descrição dos princípios gerais do sistema operativo TinyOS, o qual foi utilizado como base ao desenvolvimento das aplicações a serem executadas por estes.

5.1- TinyOs

TinyOS é um sistema operativo que foi projectado especificamente para sensores sem fios de baixa potência. Ele difere da maioria dos sistemas operativos porque é feito para ser aplicado em pequenos microcontroladores normalmente alimentado por baterias, e por isso foca o seu design para operações de muito baixo consumo energético. Além disso torna a construção de aplicações em redes mais fáceis, fornecendo para isso um conjunto de serviços e abstracções, tais como sensores, comunicações, armazenamento e temporizadores. Ele define um modelo de execução de tarefas concorrente, para que os utilizadores possam construir aplicações focando-se separadamente em cada um das funcionalidades que têm de ser suportadas. As aplicações do TinyOS bem como o próprio sistema operativo são escritos em nesC. O nesC é um dialecto da linguagem C criado com o objectivo de, por um lado reduzir o tamanho de RAM a ser utilizada nos dispositivos (bem como o tamanho do código), e por outro ajudar a evitar *bugs* nas camadas de níveis mais baixo como *race conditions*.

De uma maneira geral, o TinyOS oferece três características que tornam a escrita de aplicações mais fácil:

- Apresenta um modelo ao nível do componente, ou seja, que permite a reutilização de pedaços de código conseguindo maiores abstracções;
- Um modelo de execução concorrente, que define como os componentes disputam os recursos entre si, bem como a forma de como executam as interrupções;

- API's (*Application Programming Interfaces*), serviços, bibliotecas de componentes e uma estrutura do componente global que simplifica a escrita de novas aplicações e serviços.

Em TinyOS existem dois tipos de componentes: módulos e configurações. Estes dois componentes diferem entre si na sua implementação. A implementação de um módulo consiste em secções de escrita código nesC. Um módulo pode declarar variáveis e funções, chamar funções e compilar para código *assembly*. A implementação de um componente de configuração consiste num segmento de código nesC que permite ligar componentes entre si. Os componentes são interligados pelas interfaces. Um módulo pode usar uma interface disponibilizada por outro módulo ou fornecer uma interface a fim de se ligar a outro módulo que a use [27].

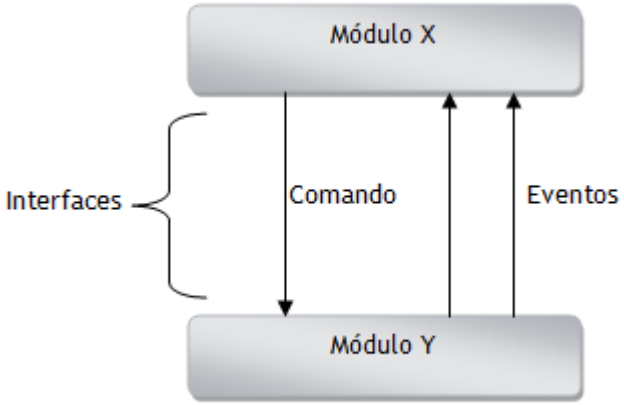


Figura 5.1- Ilustração da ligação de dois componentes entre si.

Na Figura 5.1 pretende-se ilustrar a ligação de dois módulos. O módulo X pode usar os comandos fornecidos pelo módulo Y. O módulo Y por sua vez tem de sinalizar os eventos gerados para o módulo X.

Actualmente existem várias versões do TinyOS, sendo que para este projecto foi utilizado a versão 1.x.

5.2- Desenvolvimento nos MicaZ

Com vista a desenvolver os segmentos de código a serem executados nos módulos MicaZ, foi revisto mais uma vez os requisitos do projecto. Este teria então de por um lado medir a temperatura com uma das três periodicidades apresentadas na secção 1.2, sendo que deveria de ser possível definir essa periodicidade através de comandos advenientes da central. Por outro lado haveria um ponto de medição que deveria conter na sua interface um

botão de interrupção de maneira a que esse módulo enviasse essa informação para a central. A pedido do orientador, acrescentou-se ainda um outro modo de funcionamento ao sistema, no qual este deveria apenas transmitir o valor da temperatura dos pontos de medição para a central se a temperatura medida fosse maior que um determinado valor definido (poupando assim energia associada às transmissões). O sistema quando neste modo, deveria transmitir um sinal de sinalização periódico para a central (*heartbeat*), de modo a indicar que se encontra em funcionamento (caso o valor da temperatura nunca atingisse o valor definido). Acrescentou-se ainda aos requisitos uma monitorização do estado das baterias de todos os nós MicaZ, que achou-se ser uma funcionalidade interessante a conter no sistema. Apesar de tanto a *gateway* como os pontos de medição serem constituídos por módulos MicaZ, o fluxo de projecto de cada um destes elementos foram diferentes devido às funcionalidades associadas a cada um.

5.3.1- *Gateway*

Já sabíamos então que a *gateway* teria de receber e enviar instruções por RS232 para o PIC, instruções essas que eram enviadas e recebidas pela central. Definiu-se por isso como ponto de partida começar a montar um circuito para interligar a *gateway* ao PIC. Na Figura 5.2 representa-se então o circuito desenvolvido. Ele é constituído basicamente por dois MAX3232 que é um driver projectado para funcionar com a norma TIA/EIA-232 (RS232) para dispositivos que operem entre os 3V e os 5.5V de tensão de alimentação. A alimentação do MAX3232 alimentado a 3.3V foi retirada directamente da ficha de expansão de 51 pinos do MicaZ (a ficha DF9B-51P-1V), enquanto a alimentação do MAX3232 alimentado a 5V foi retirada da ficha de 26 pinos da DETPIC.

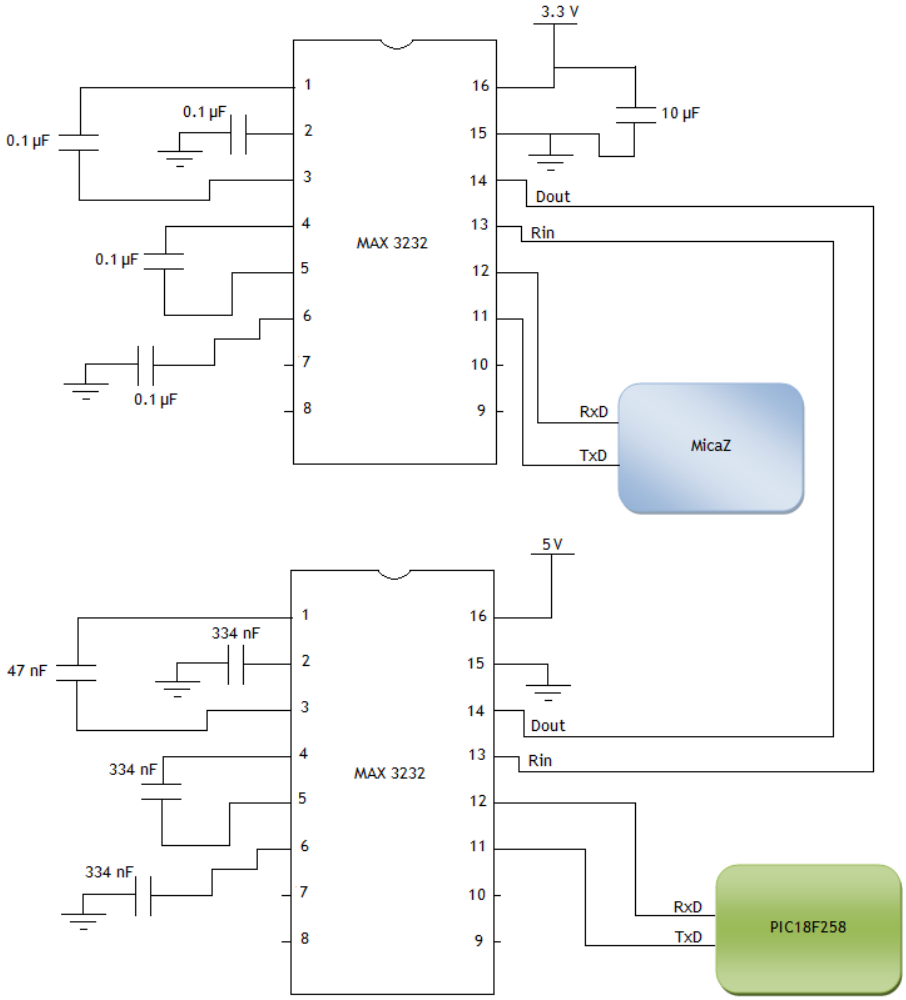


Figura 5.2- Circuito de Ligação por RS232 do MicaZ ao PIC

Tendo então agora todo o *hardware* montado para a *gateway*, procedeu-se então ao desenvolvimento do segmento de código a ser executado no MicaZ. Perante isso veio logo a necessidade então de se definir os comandos que este enviaria e receberia via RS232. Definiu-se então que os comandos seriam constituídos por tramas de oito *bytes*, as quais chegariam para a definição de todos os comandos pretendidos. Na tabela seguinte apresenta-se então os formatos das tramas associados a cada comando.

Tabela 5.1- Comandos trocados entre a central e a *gateway*

Comando	Trama
Configuração de leitura da temperatura de 5/5 minutos	10000000
Configuração de leitura da temperatura de 15/15 minutos	20000000
Configuração de leitura da temperatura de 1/1 hora	30000000
Porta Fechada	40000000

Porta aberta	50000000
Modo transmissão se temperatura máxima atingida	6TTTT000
Heartbeat do sistema	70000000
Carga de bateria fraca num nó	800000NN
Valor da Temperatura	9TTTT0NN
Parar Monitorização da Temperatura	A0000000

Os quatro bytes com um "T" associados ao comando de "Modo transmissão se temperatura máxima atingida" apresentado na Tabela 5.1, representam o parâmetro temperatura a partir da qual o sistema deverá começar a transmitir se esta for atingida. Este parâmetro é enviado da central, e é um número compreendido entre 0 e 4095 em codificação BCD (*Binary-coded decimal*), valor que corresponde à leitura directa da ADC de 12 bits do MicaZ. Utilizou-se esta espécie de mapeamento para evitar a transmissão de um byte referente ao sinal (e outro possivelmente para a vírgula) e porque assim também o valor enviado é comparado directamente no MicaZ logo após a leitura da sua ADC (não sendo necessário assim este realizar uma conversão, poupando-se recursos neste). Assim os valores recebidos e enviados pela central serão sempre em valores de temperatura de ADC, sendo está então encarregue das respectivas conversões para graus Célsius.

Os quatro bytes com um "T" associados ao comando "Valor da Temperatura" apresentado na Tabela 5.1 representam também a temperatura que a *gateway* recebeu de um dos nós na rede e que é enviada para a central, sendo essa temperatura enviada também em BCD referente ao valor lido directamente pela ADC. Os dois bytes sinalizados com um "N" neste comando referem-se a informação do Nó da rede que enviou esse valor de temperatura.

Os dois bytes sinalizados com um "N" no comando de "Carga de bateria fraca num nó", referem-se ao nó em que as baterias estão fracas. Este comando é enviado sempre que a leitura da tensão das baterias for inferior a 2500mV.

É de notar que os comandos de configuração do período da leitura da temperatura, "Modo de transmissão se temperatura máxima atingida" e "parar Monitorização da Temperatura" são comandos enviados da central para a *gateway*, sendo que os outros comandos são enviados da *gateway* para a central.

Tendo os comandos já definidos, podia-se então definir os segmentos de código para serem executados na *gateway*. Assim quando a *gateway* recebe uma trama de comando vinda da central via UART verifica qual o comando que recebeu, para assim construir o pacote da mensagem a enviar para todos os nós da rede por *Zigbee*. O mesmo se aplica quando recebe uma mensagem por *Zigbee* vinda dos outros nós da rede. Nesse caso, a *gateway* deve então

verificar que tipo de comando se trata de maneira a formar a trama correspondente a ser enviada por RS232.

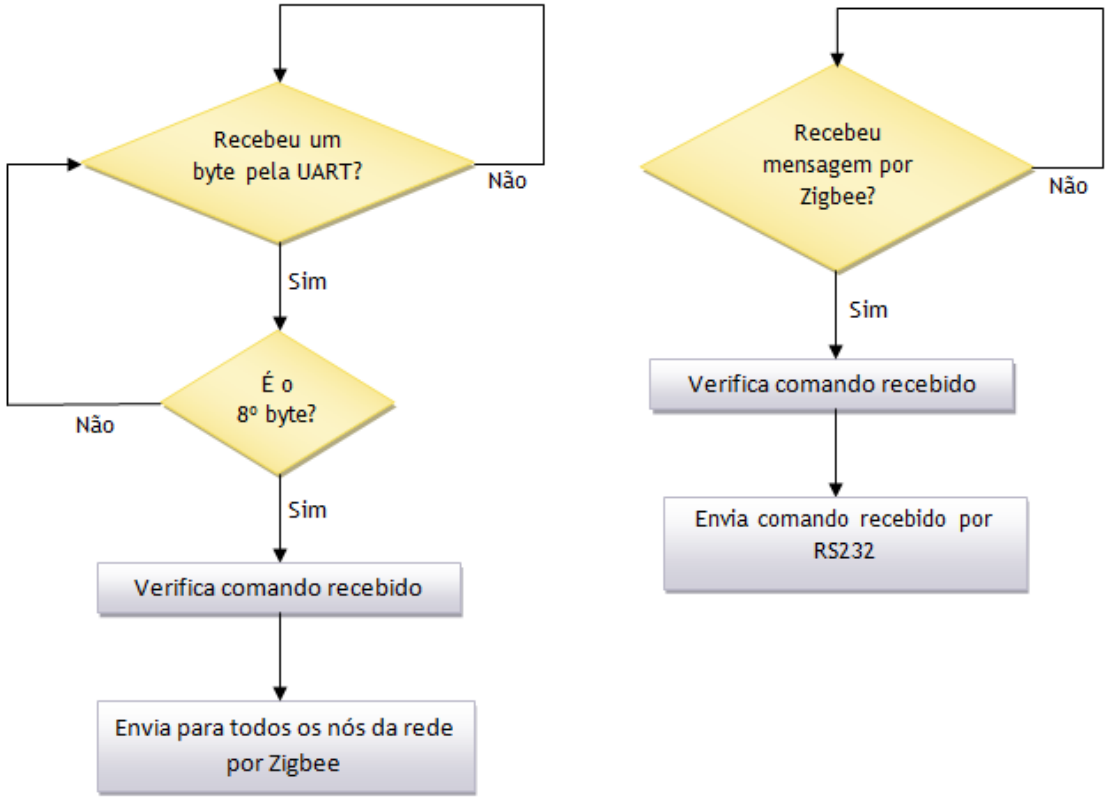


Figura 5.3- Algoritmo de recepção e envio de mensagens via RS232-*Zigbee* e *Zigbee*-RS232.

Paralelamente a estes processos, na *gateway* é verificado também com uma determinada periodicidade (de hora em hora) a tensão das baterias desta, de modo a que se estas apresentarem um valor de tensão abaixo dos 2500mV, seja enviado para a central a respectiva notificação (Figura 5.4).

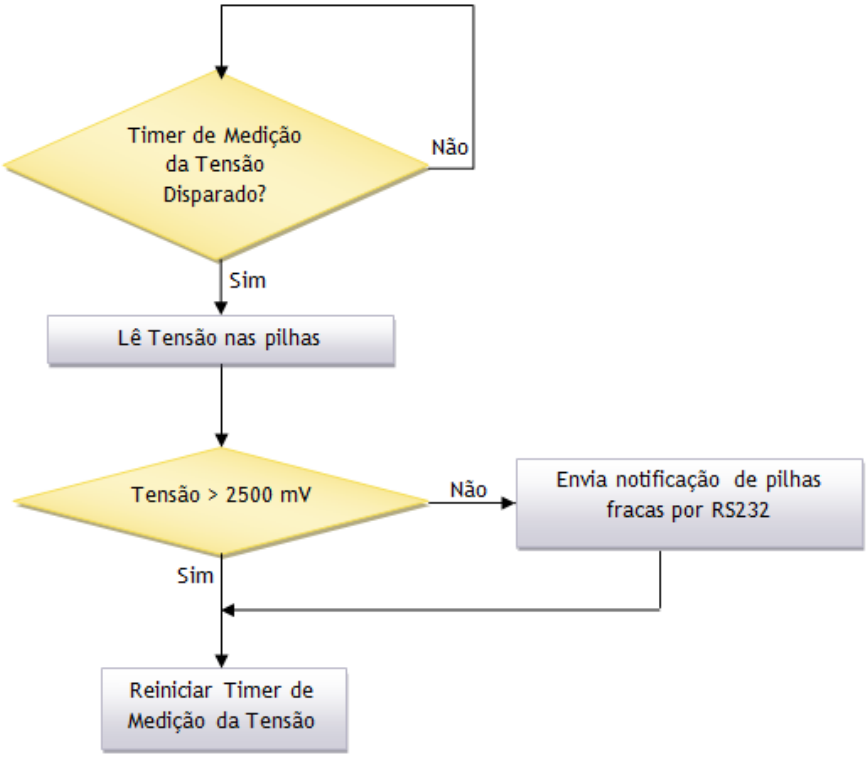


Figura 5.4- Fluxograma de leitura do estado das pilhas

5.3.2- Pontos de medição

Ao contrário da *gateway*, nos pontos de medição não foi preciso adicionar nenhum circuito, a não ser no ponto de medição que para além de ler a temperatura, monitoriza a abertura e fecho da porta da zona de armazenamento de produtos. Para esse ponto de medição desenhou-se uma placa de circuito impresso a ser conectada na ficha de expansão de 51 pinos do MicaZ, placa essa onde foi projectado um botão para activar a interrupção 3 do MicaZ (Figura 5.5).

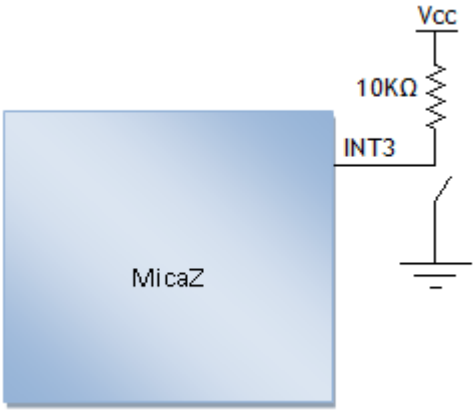


Figura 5.5- Botão de interrupção para monitorização da abertura e fecho da porta

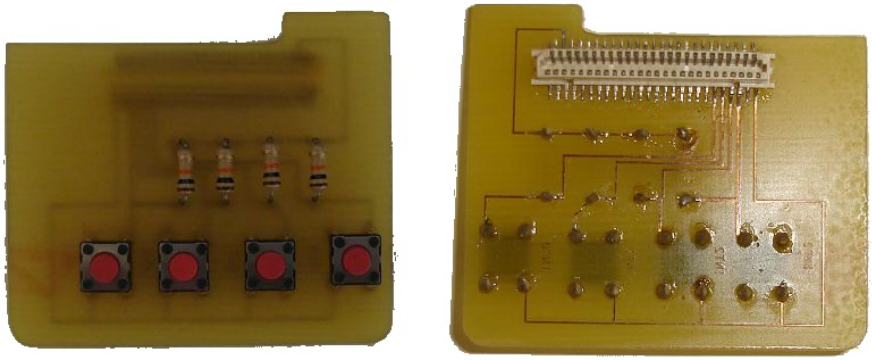
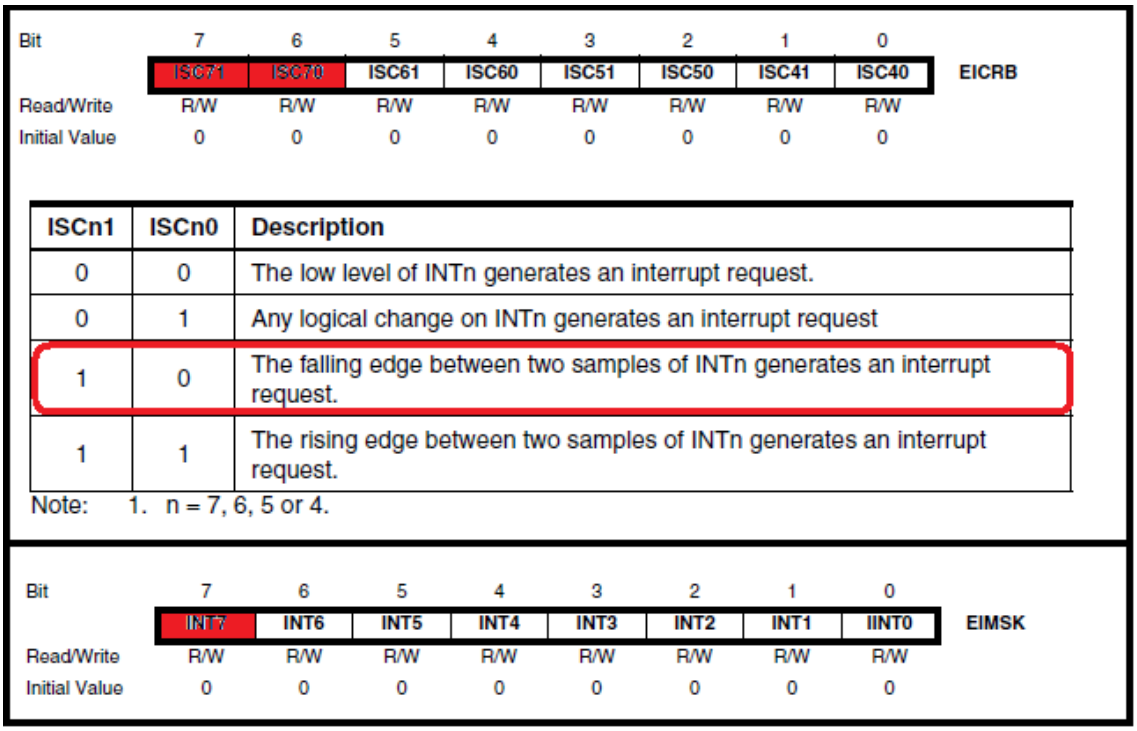


Figura 5.6- Placa de circuito Impresso com botão de interrupção

A folha de características do MicaZ apresenta 4 interrupções do ATmega128 que podem ser acedidas através da ficha de expansão de 51 pinos do MicaZ (INT0, INT1, INT2 e INT3). Qualquer uma destas interrupções pode ser utilizada menos a Interrupção 2 porque esta é usada pelo *transceiver Zigbee* do MicaZ para envio/recepção das mensagens e também para a leitura de temperatura do sensor MTS310. Optou-se por esse motivo por usar a Interrupção 3.

Com a concretização deste circuito deu-se inicio à realização do software a ser executado nos pontos de medição. Houve logo o problema de o TinyOS não fornecer qualquer interface com os pinos de interrupção, o que obrigou a usar-se instruções de baixo nível em C juntamente com o módulo que continha a aplicação, para acedermos directamente aos registos do ATmega128. Houve também outro problema relacionado com o mapeamento das interrupções por parte do MicaZ. Estranhamente, o pino de Interrupção 3 do MicaZ corresponde à Interrupção Externa 7 do ATmega128. Da consulta da folha de características do ATmega128 chegou-se a conclusão que para a interrupção externa 7 do ATmega128 ficasse sensível a qualquer transição de nível lógico, tinha-se que no registo EICRB (*External Interrupt Control Register B*) colocar o bit 7 (ISC70) a nível lógico alto e colocar um “0” no

bit 8 (ISC71). Por fim, para habilitar a interrupção 7 teve-se ainda de colocar um "1" no oitavo bit (INT7) do registo EIMSK (External Interrupt Mask Register).



Bit	7	6	5	4	3	2	1	0	
	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	EICRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ISCn1	ISCn0	Description
0	0	The low level of INTn generates an interrupt request.
0	1	Any logical change on INTn generates an interrupt request
1	0	The falling edge between two samples of INTn generates an interrupt request.
1	1	The rising edge between two samples of INTn generates an interrupt request.

Note: 1. n = 7, 6, 5 or 4.

Bit	7	6	5	4	3	2	1	0	
	INT7	INT6	INT5	INT4	INT3	INT2	INT1	IINT0	EIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Figura 5.7- Registos para activação de interrupção externa 7 do ATmega128 [30]

O segmento de código que escrito para inicialização da interrupção 7 da ATmega128 foi:

```
command result_t StdControl.start()
{
  sbi(EICRB,ISC70);  // botao de interrupcao 3
  cbi(EICRB,ISC71);
  sbi(EIMSK , INT7);  //enable int3 do micaz...

        (...)
```

Figura 5.8- Inicialização da interrupção externa 3 do MicaZ

Com o pino de interrupção 3 do MicaZ sensível a mudança de estado, para com isto monitorizarmos o estado da porta, criou-se uma variável que contém sempre o estado em que a porta se encontra. Inicializou-se essa variável com o estado do sistema inicial (considera-se que a porta inicialmente se encontra aberta). Depois sempre que o programa transita para a rotina de atendimento à interrupção 7, verifica-se nesta o estado anterior em que o sistema se encontrava (porta fechada ou aberta), sabendo-se assim se o botão foi pressionado (porta

fechou) ou largado (porta abriu), enviando-se essa informação para a *gateway* e actualizando-se o novo estado do sistema, como se apresenta na Figura 5.9.

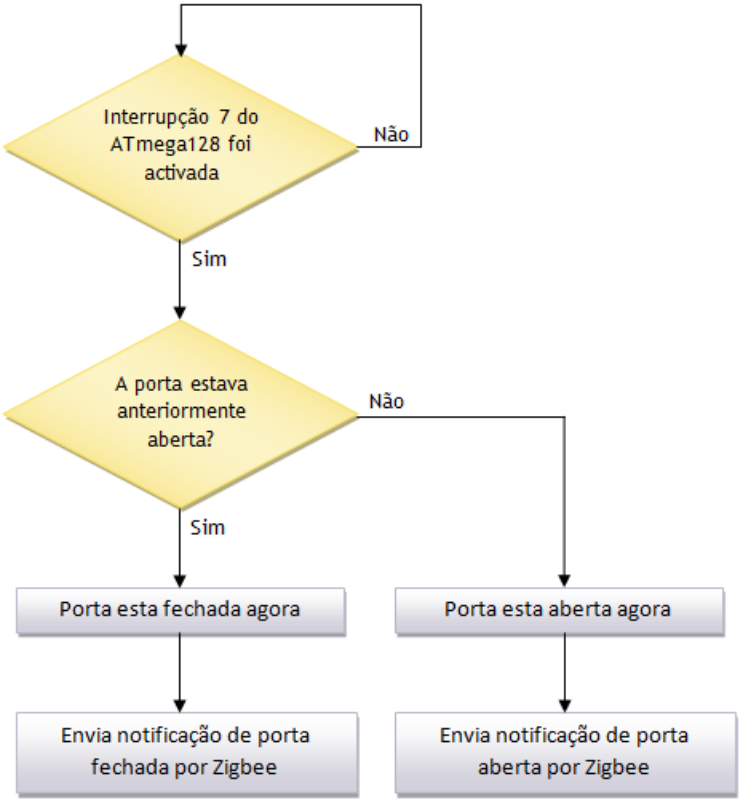


Figura 5.9- Fluxograma de monitorização do estado da porta

Faltava agora só implementar então as funcionalidades de medição da temperatura nos nós de medição e recepção de configurações por parte destes. Considerou-se então que de início não é realizada qualquer tipo de medição de temperatura, ficando o sistema a espera de receber por *Zigbee* uma mensagem de configuração para entrar em funcionamento. Conforme a configuração que o sistema recebe, este inicia um temporizador em função "TIMER_REPEAT" que é um tipo de temporizador fornecido pelo TinyOS, e que activa uma interrupção de *timer* a uma cadência definida repetitivamente (Figura 5.10 e Figura 5.11).

No modo de parar monitorização da temperatura a única coisa que o sistema faz é parar o temporizador que estiver activo, não sendo assim efectuadas medições.

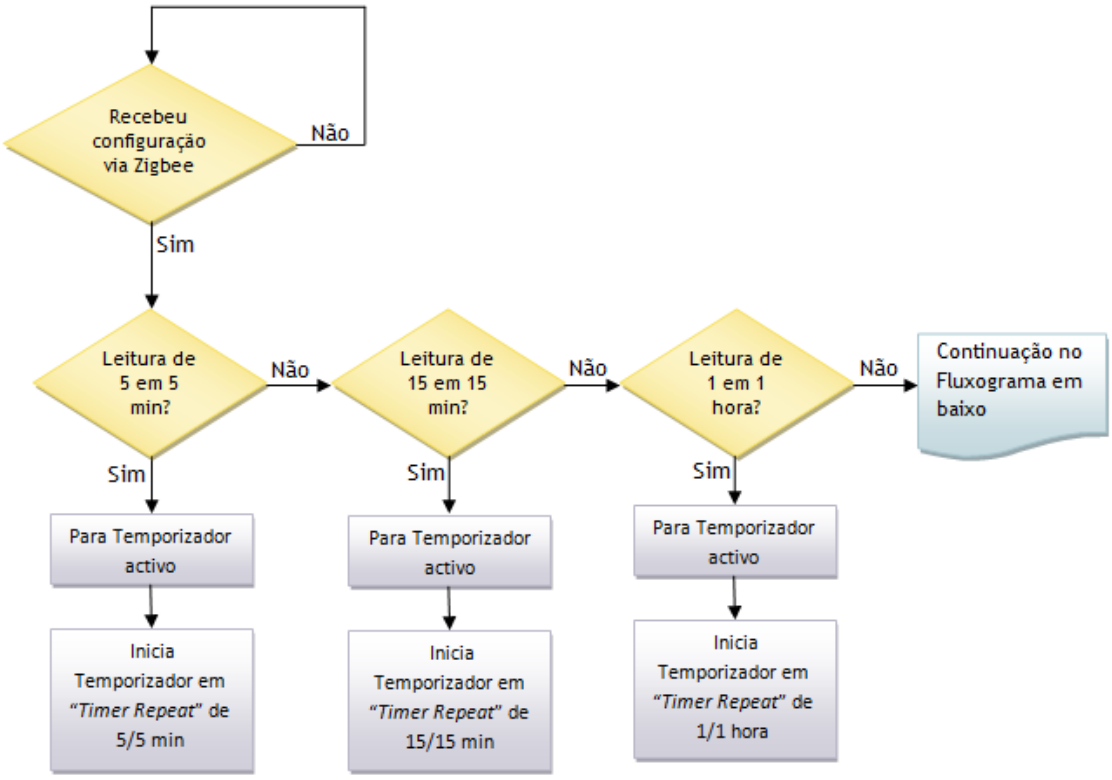


Figura 5.10- Configuração dos nós de medição

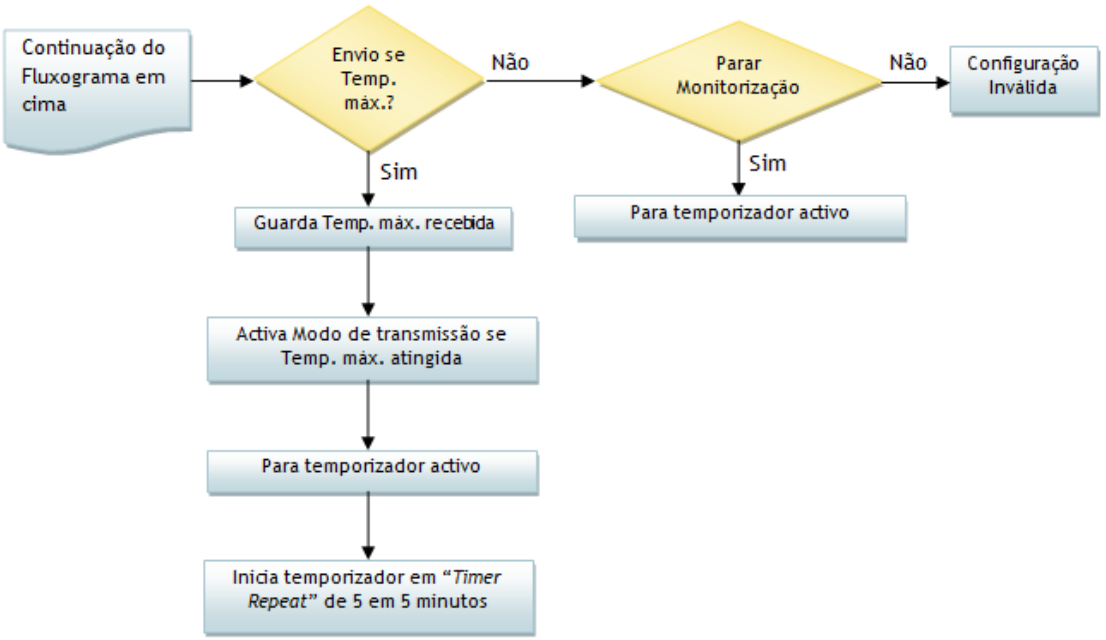


Figura 5.11- Continuação do Fluxograma da Figura 5.15

Quando o temporizador do sistema é disparado, este lê o valor da Temperatura actual da ADC. Depois verifica se o sistema se encontra no modo "Modo transmissão se temperatura máxima atingida". Se não estiver, o nó de medição mensura a temperatura e envia a informação da temperatura para a *gateway*. Por outro lado se o sistema se encontrar no modo "Modo transmissão se temperatura máxima atingida", este incrementa um contador que tem como função de informar o nó de quando tem de enviar o sinal de *heartbeat*. Depois de o incrementar, verifica se esse contador chegou a 24. Se tiver chegado envia o sinal de *heartbeat* por Zigbee, colocando o contador a zero. Assim o sinal de *heartbeat* é enviado sempre de duas em duas horas caso a temperatura nesse intervalo de tempo não atinja o valor máximo admissível, como se esquematiza na Figura 5.13.

Por último, e a semelhança do que se implementou para a *gateway*, em cada ponto de medição é realizado também uma verificação da tensão de baterias com uma dada periodicidade. Também aqui se o valor lido da tensão das baterias for inferior a 2500 mV é enviado uma sinalização de bateria fraca, mas neste caso, a sinalização é enviada por *Zigbee* para a *gateway* (Figura 5.12), sendo este responsável pelo reencaminhamento da mensagem para a central.

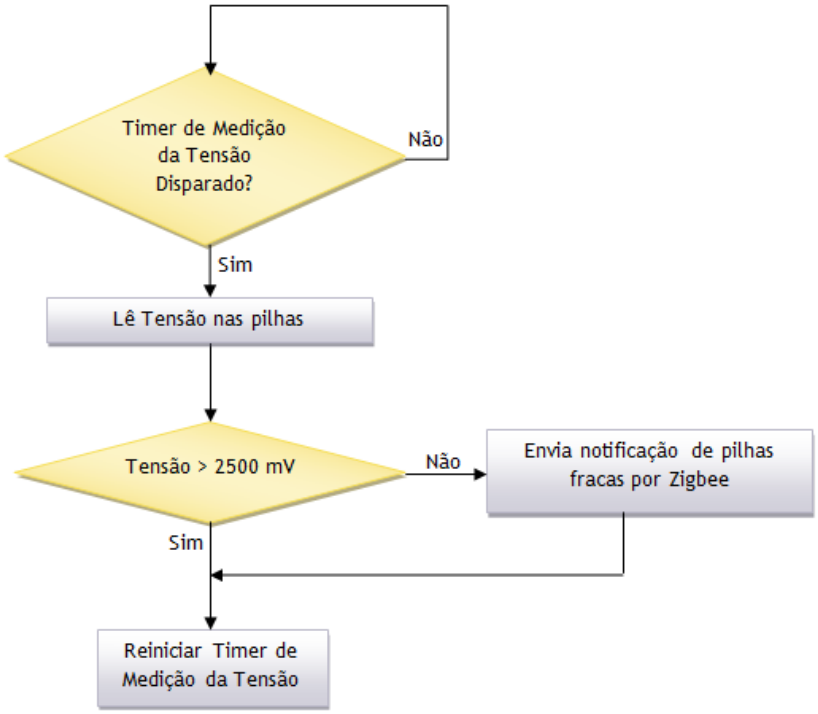


Figura 5.12- Verificação da tensão das baterias nos pontos de medição

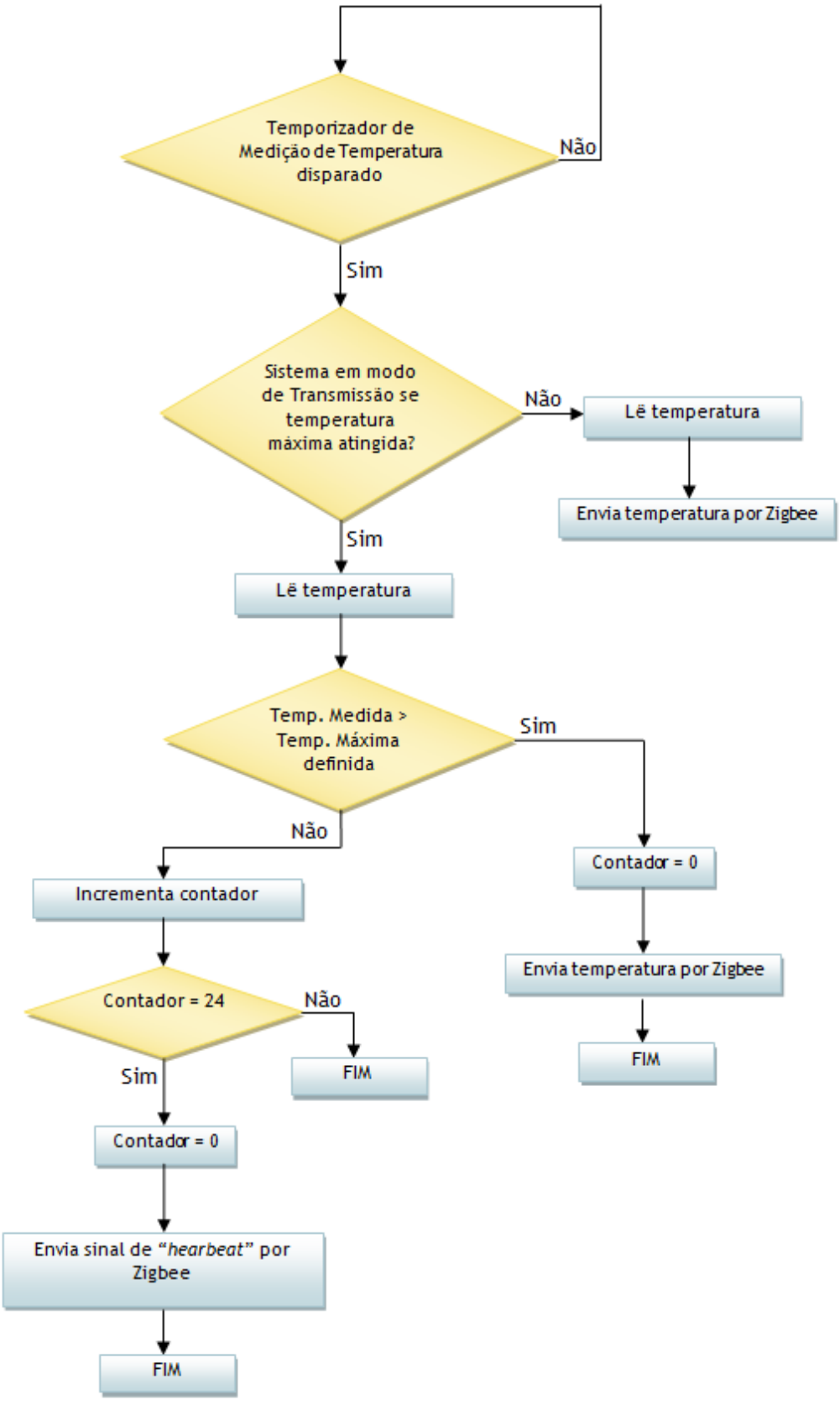


Figura 5.13- Ciclo de leitura da temperatura

5.3- Conclusão

Neste capítulo definiu-se uma rede de troca de dados via protocolo *Zigbee* a partir de nós MicaZ. Essa rede tem como interface a *gateway*, a qual permite alterar o modo de funcionamento desta, injectando-se tramas via RS232. Consegue-se ainda através dessa interface receber toda a informação que os pontos de medição enviam para a *gateway*. Com isto temos então concluído um subsistema do nosso projecto, o qual conseguimos controlar e supervisionar através de tramas RS232 por nós criadas.

Capítulo 6

Subsistema CAN

Neste capítulo pretende-se descrever como foi implementado o barramento CAN a partir de dois PIC18F258.

6.1- CAN no PIC 18F258

Nos pontos a seguir pretende-se descrever o processo inicialização do barramento CAN, bem como o envio e recepção de mensagens CAN, através do controlador CAN que o PIC18F258 possuí.

6.2.1- Inicialização do barramento

Para inicializar o barramento CAN, primeiro coloca-se o bit REQOP2 (bit mais significativo dos três bits que constituem o *Request CAN Operation Mode bits*) do registo CANCON (*Can Control Register*) a nível lógico alto. A seguir espera-se até que o bit OPMODE2 do registo CANSTAT mude para nível lógico alto, indicando que o barramento CAN se encontra no modo de configuração. Uma vez o controlador CAN em modo de configuração, configura-se os registos BRGCON1, BRGCON2, BRGCON3 (*Baud Rate Control Registers*), registos estes que controlam os tempos de bit, os quais geram o *baud rate* do barramento CAN. O *baud rate* gerado foi de 167Kbit/s com os valores apresentados na Figura 6.1. Depois define-se uma filtragem de mensagens no barramento CAN, filtragem essa para receber-se apenas as mensagens com identificador de tamanho padrão (11 bits) nos *buffers* de recepção 0 e 1 (registos RXF0SIDL e RXF1SIDL). Por fim, ainda no modo de configuração do controlador, define-se se se pretende usar máscaras nos *buffers* de recepção com o intuito de filtrar gamas de endereços. No nosso caso não foram usadas máscaras já que a quantidade de mensagens que circulam no barramento CAN não é significativa. Com vista agora a colocar o

barramento CAN no modo normal, coloca-se a seguir os bits do registo CANCON a nível lógico baixo. Depois de este estar então já no modo normal, limpam-se as *flags* de recepção e de transmissão, as *flags* indicadoras de *buffers* de recepção cheios e habilita-se a que o controlador quando o *buffer* 0 de recepção estiver cheio passe directamente a receber no *buffer* 1 (bit RXB0DBEN do registo RXB0CON). Depois por último, configura-se os registos RXB0CON e RXB1CON de modo a que só sejam recebidos nos *buffers* de recepção apenas mensagens CAN válidas.

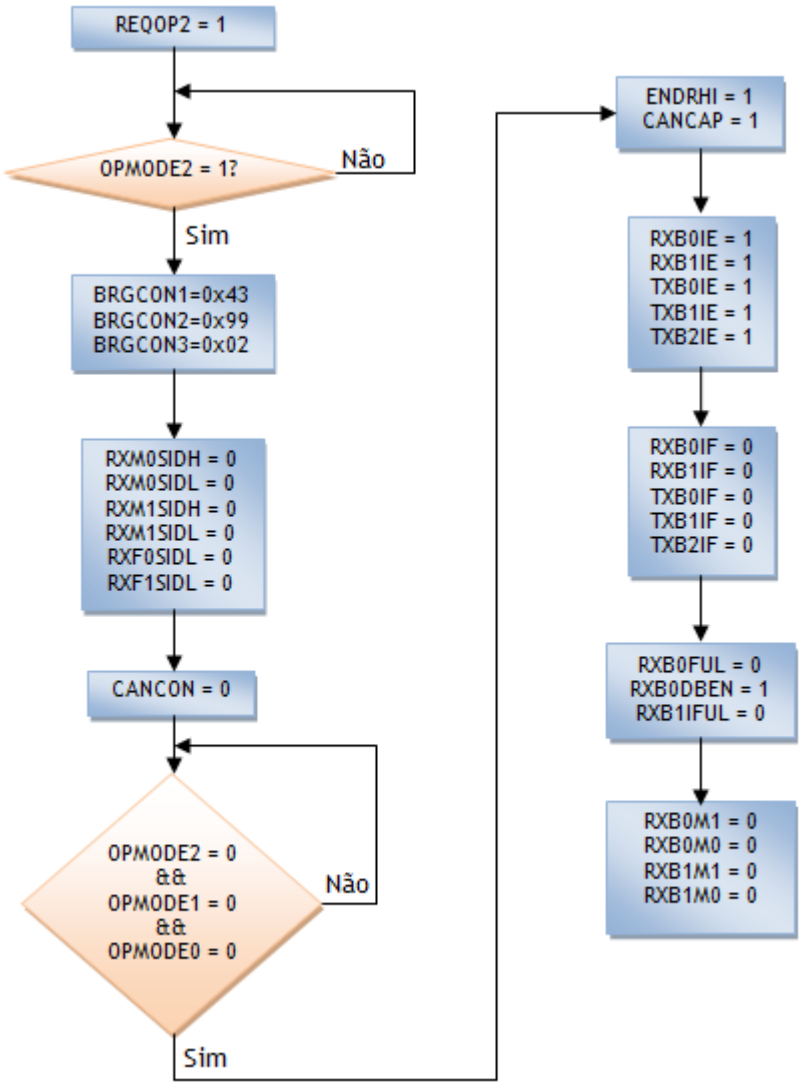


Figura 6.1- Inicialização barramento CAN.

6.2.2- Envio de mensagens pelo barramento

Depois de inicializado o controlador CAN, o envio de mensagens é feito como se ilustra na Figura 6.2. Primeiro é seleccionado um dos três *buffers* de envio existentes, verificando-se se nesse *buffer* está a ser transmitido alguma mensagem. Quando o *buffer* não

estiver a transmitir dados, escreve-se o ID de 11 bits nos registos TXBnSIDH e TXBnSIDL (os três bits menos significativos do ID nos três bits mais significativos do registo TXBnSIDL e os restantes oito bits do ID no registo TXBnSIDH). De seguida escreve-se o tamanho dos dados da trama CAN a ser enviada (número de bytes) no respectivo registo TXBnDLC, carrega-se os bytes a ser enviados nos registos TXBnD0 a TXBnD7 e por fim envia-se os dados pondo o respectivo TXREQ a nível lógico alto.

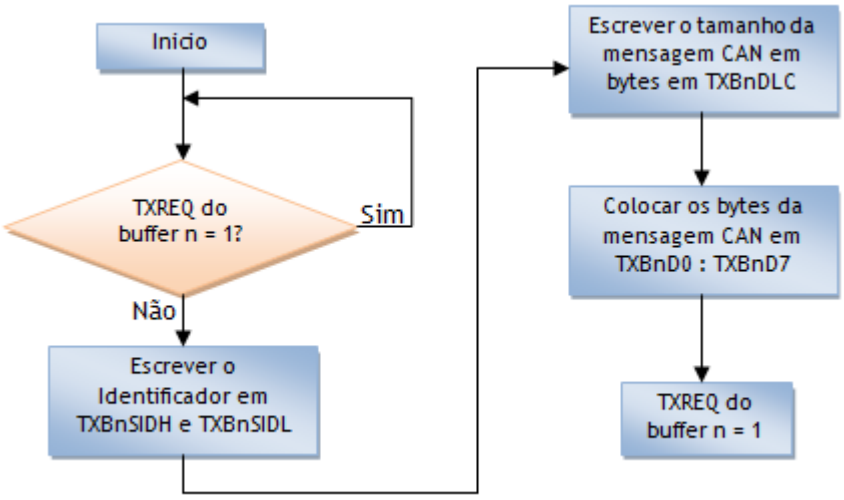


Figura 6.2- Envio de uma mensagem por CAN.

6.2.3- Recepção de mensagens do barramento

Quando uma mensagem CAN é recebida, a respectiva *flag* RXB0IF ou RXB1IF é activada, notificando-se que foi recebida uma mensagem CAN no buffer de recepção 0 ou 1 respectivamente. Para procedermos a recepção da mensagem CAN temos primeiro de limpar a *flag* de recepção (RXB0IF ou RXB1IF) e a respectiva *flag* indicadora de buffer cheio (RXB0FUL ou RXB1FUL). De seguida lê-se o endereço CAN remetente da mensagem nos registos RXBnSIDL e RXBnSIDH (os três bits menos significativos do identificador nos três bits mais significativos do registo RXBnSIDL e os restantes oito bits do identificador no registo RXBnSIDH), lê-se o tamanho da mensagem recebida nos quatro bits menos significativos do registo RXBnDLC e por fim lê-se os bytes CAN recebidos nos registos RXBnD0 ao RXBnD7.

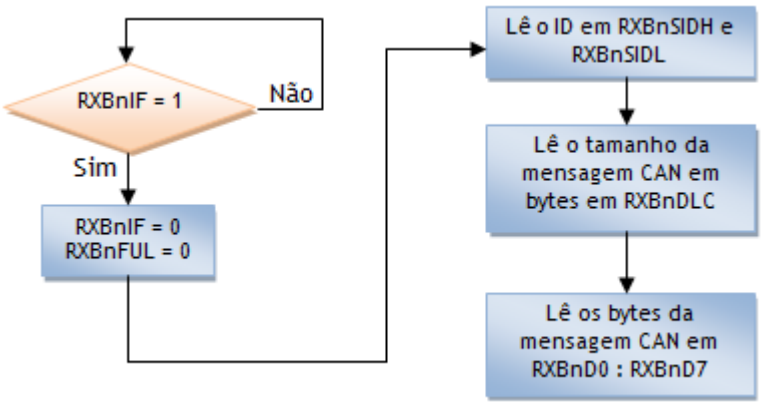


Figura 6.3- Recepção de mensagem CAN.

6.2- Software desenvolvido

O objectivo do barramento CAN era de passar, como já tinha sido mencionado, os dados de dentro da zona de armazenamento dos produtos para a cabine onde se encontra o condutor. Estes dados a serem enviados de um lado para o outro, eram tramas constituídas de oito bytes, tramas que foram definidas na secção 5.3.1. Assim, estes microcontroladores PIC deveriam comportar-se como simples intermediários no envio das tramas da *gateway* para a central, como da central para a *gateway*, tal como se esquematiza na Figura 6.4.

Figura 6.4- Esquema de ligação dos PIC's.

O algoritmo implementado para a implementação da funcionalidade pretendida foi o seguinte: sempre que o PIC recebe oito bytes de dados por RS232 envia esses mesmos bytes numa trama CAN para o outro PIC. Quando o mesmo PIC recebe uma mensagem CAN, este tinha de enviar cada um dos oito bytes recebidos via RS232 (Figura 6.5). Assim como as funcionalidades dos microcontroladores eram idênticas, os segmentos de código desenvolvidos são idênticos para os dois microcontroladores. Alterou-se apenas os endereços CAN definidos para cada um.

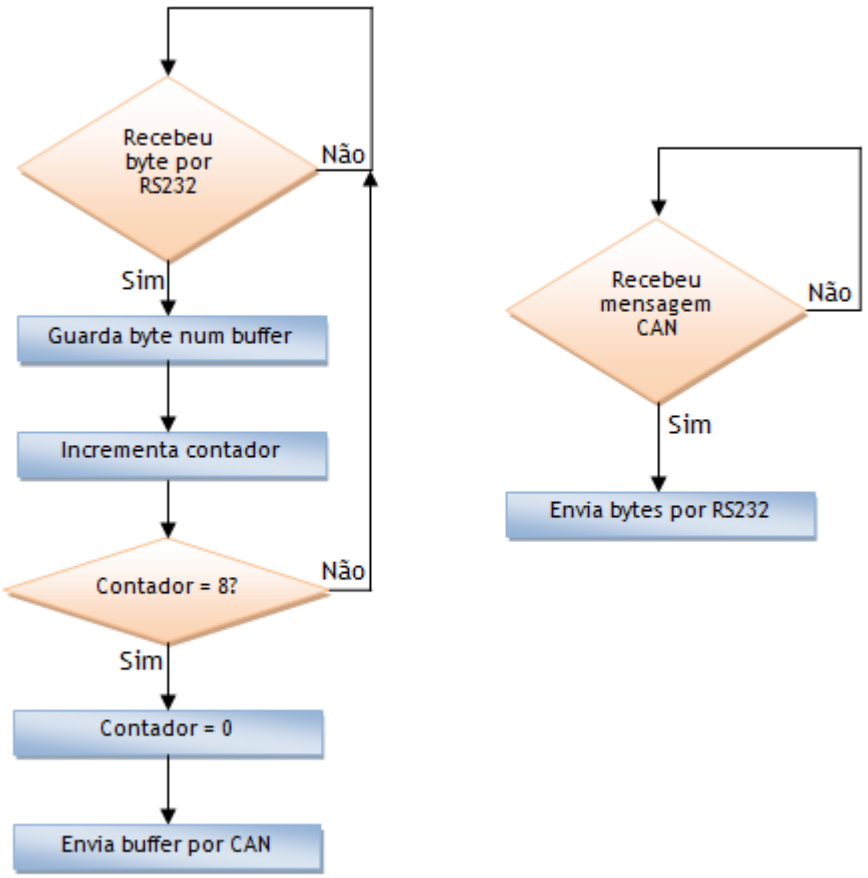


Figura 6.5- Fluxograma ilustrativo dos processos executados nos PIC's.

É de salientar que devido ao RS232 ser um protocolo de comunicação do tipo *Full-Duplex*, não existe o perigo de conflitos de dados devido ao envio e recepção num mesmo instante de tempo. Por outro lado embora o CAN utilize um barramento partilhado onde cada nó transmite à vez, os conflitos de acesso são resolvidos por um método de arbitragem que é determinístico e baseado em prioridades fixas (os ID's das mensagens).

6.3- Conclusão

Neste capítulo descreveu-se todo o processo envolvido na construção de um barramento CAN utilizando dois microcontroladores PIC18F258 da Microchip. Descreveu-se igualmente os algoritmos utilizados com vista à interligação do microcontrolador que interage (via RS232) tanto com a *gateway*, assim como o que interage com a central do nosso sistema.

Capítulo 7

Ligação à central de gestão

Neste capítulo pretende-se descrever a solução computacional que foi elaborada de maneira a que fosse sido registados os vários eventos ao longo da viagem. Pretende-se também ainda descrever, a solução a que se chegou para que quando o camião chegasse no final da viagem, fosse descarregado o ficheiro *log* para o computador de destino, bem como a forma de como é configurado o modo de operação no camião.

7.1- Arquitectura Global

O esquema de rede a que se chegou para a troca de dados entre a central do camião e outro computador via Wi-Fi foi o apresentado na Figura 7.1.

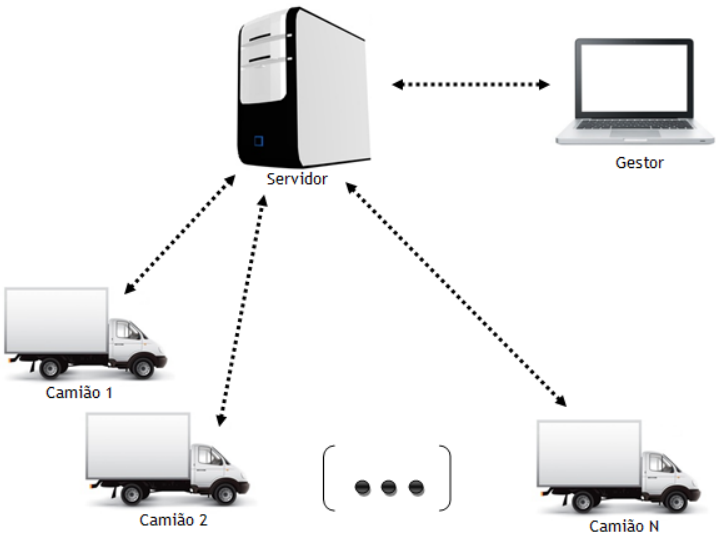


Figura 7.1- Arquitectura da Rede

Esta é composta por três tipos de componentes: camião, um servidor e um gestor. A rede criada é baseada em *sockets* Java TCP/IP, e parte do pressuposto que o sistema operativo que está a ser executado na central do camião já conhece a rede Wi-Fi onde o servidor está ligado. Parte-se do pressuposto também que quando o sistema operativo detecta essa rede liga-se automaticamente a ela (que é o que normalmente acontece quando um computador hoje em dia detecta uma rede *wireless*). Após ligação a essa rede, a central do camião tenta estabelecer uma ligação com o servidor através da criação de um *socket* TCP, *socket* esse criado com o IP e porta que o servidor se encontra a usar. O servidor ao detectar esse *socket* aceita a ligação pedida, e atribui-lhe um *thread, thread* com o qual o camião efectua a troca de mensagens com o servidor a partir daí, possibilitando assim ao servidor ficar a espera da entrada de novos camiões no sistema. O servidor depois nesse *thread* efectua uma espécie de autentificação, enviando o comando "Chave" para o camião, que ao recebê-lo envia a chave definida para o sistema. Este mecanismo é realizado para prevenir outros *sockets* (provenientes de outras possíveis aplicações) de se conectar ao nosso servidor. Autenticado o camião, é pedido por parte do servidor a identificação do camião conectado. Quando o servidor recebe essa identificação, envia-a para o gestor para que este saiba que o camião se encontra online. O Gestor ao receber essa informação faz imediatamente um pedido de *log* ao servidor. O camião ao receber esse pedido envia o *log* para o servidor, sendo este reencaminhado para o gestor. É possível no final através do gestor definir uma configuração para o camião, que quando a recebe se desconecta-se automaticamente do sistema.

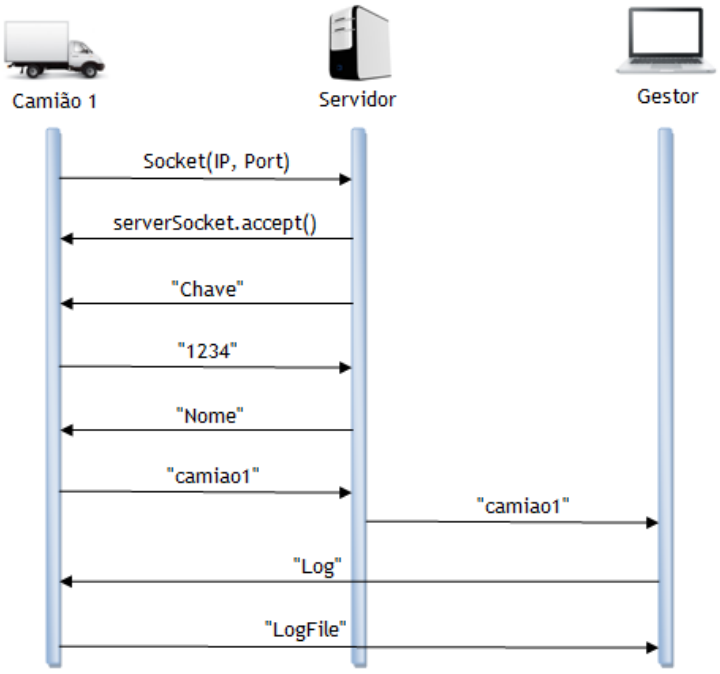


Figura 7.2- Troca de mensagens entre Camião, Servidor e Gestor

7.2- Software da Central

Como já referido até aqui, todos os eventos que ocorrem durante a viagem do camião são enviados pelos pontos de medições, passando por uma série de dispositivos até serem armazenados na solução computacional (central). Ora o nosso problema fixava-se então inicialmente em construir uma aplicação que recebesse as tramas enviadas pelo PIC através de RS232, e que as descodificasse de maneira a gerar informação visual para o condutor, bem como fosse registando todos os eventos ocorridos de maneira a gerar o ficheiro *log* da viagem.

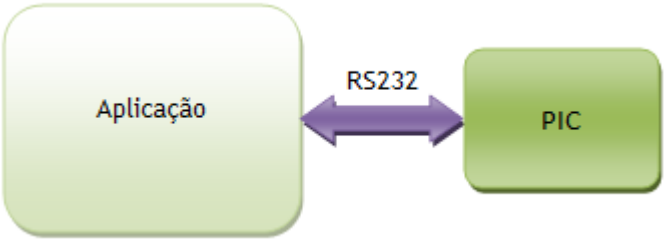


Figura 7.3- Troca de Dados entra Aplicação da Central e PIC via RS232

A aplicação que trata de todas as funcionalidades atribuídas à central foi realizada em java, de maneira a que pudesse ser executada em qualquer sistema operativo que possua o *java runtime environment* instalado. A aplicação a que se chegou possuí o aspecto apresentado abaixo (Figura 7.4).

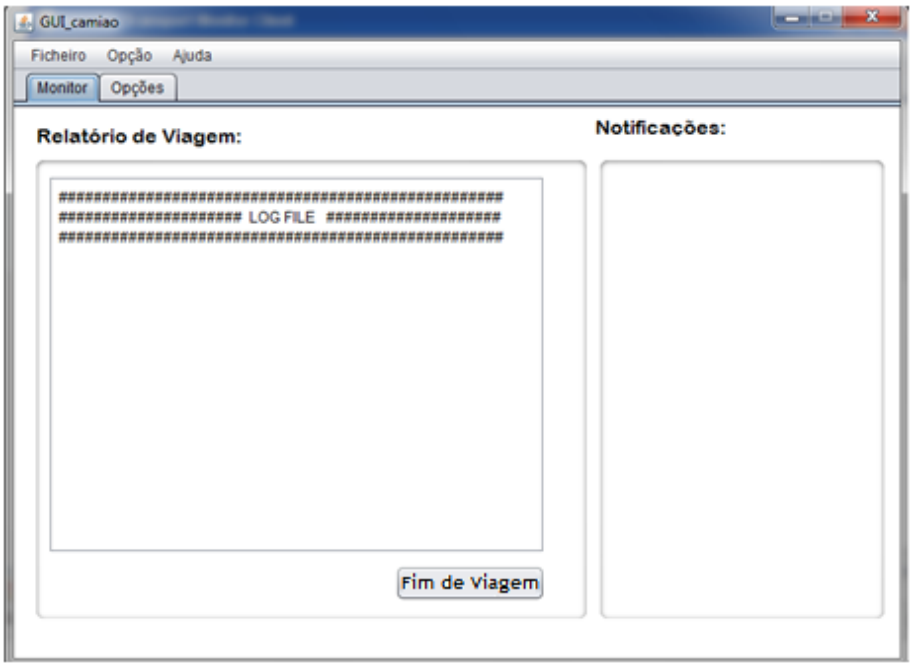


Figura 7.4- Página inicial da aplicação do camião

Pode ver-se de entre outras coisas na interface desta aplicação, o botão “Fim de Viagem”. Esse botão foi criado para que o sistema mesmo que o camião já tenha sido configurado, não comece imediatamente a fazer a monitorização de eventos. Assim mesmo configurado o camião só inicia a monitorização e a procura do servidor a conectar-se apenas após esse botão ter sido pressionado. Pode ver-se ainda nesta aplicação duas grandes áreas: a área de Relatório de Viagem que é a zona onde começa a aparecer o registo dos eventos, e a zona das Notificações, onde aparece mensagens visuais de alertas.

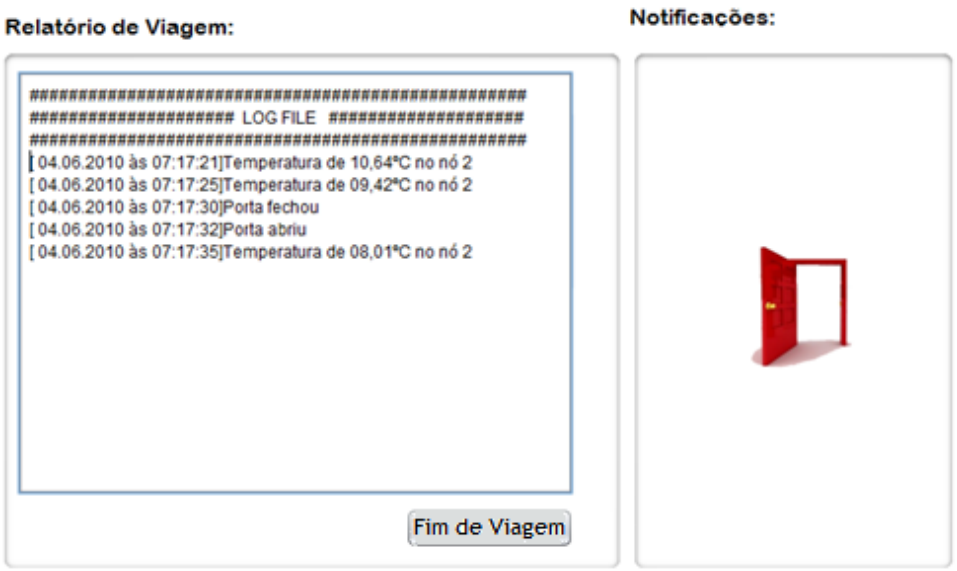


Figura 7.5- Registo dos eventos e alerta de porta aberta

Como se pode ver na Figura 7.5, o registo dos eventos é acompanhado sempre da hora e data de quando estes ocorreram, bem como também da informação do nó que o registou (caso de trate de um evento de medida de temperatura). Os alertas que o sistema notifica são basicamente três:

- Alerta de temperatura – O utilizador pode definir este alerta no separador de “Opções” (Figura 7.7), e esta temperatura então definida é a temperatura a partir do qual o sistema emite este alerta caso a temperatura de medição em algum nó exceda este valor;
- Alerta de Porta Aberta – Serve para notificar o condutor que a porta da zona de armazenamento dos produtos se encontra aberta;
- Alerta de baterias fracas – Este alerta serve para informar qual ou quais os dispositivos da rede sem fios apresentam valor de tensão das baterias abaixo dos 2500 mV.

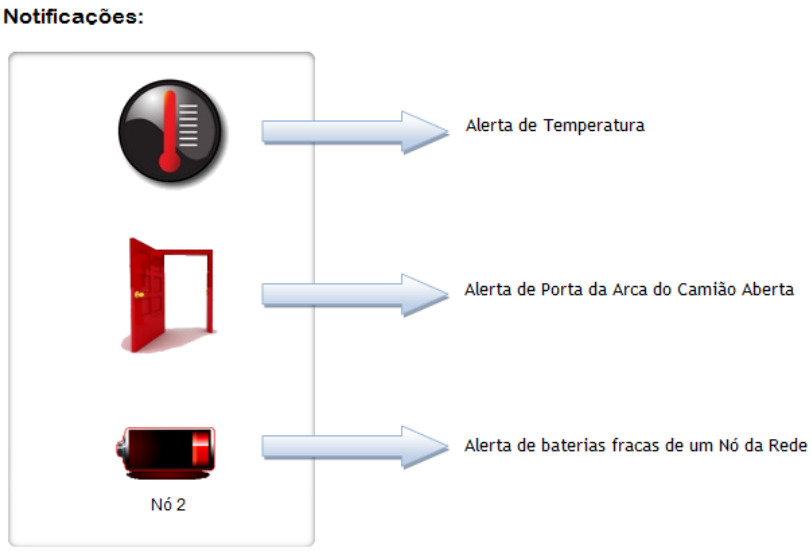


Figura 7.6- Alertas ocorrentes no sistema

O separador com o menu "Opções" é exibido na Figura 7.7. Este menu permite ao utilizador configurar as conexões do sistema (Nome do veículo na rede, porta COM utilizada para comunicação, IP e porta para comunicação com o servidor) e a temperatura a partir do qual o sistema deve exibir o alerta de temperatura. De notar que todas as temperaturas que esta aplicação recebe provenientes dos Pontos de Medição, são recebidas em valores referentes a leitura directa da ADC dos MicaZ, sendo então esta aplicação responsável pela conversão do valor recebido para graus Célsius.

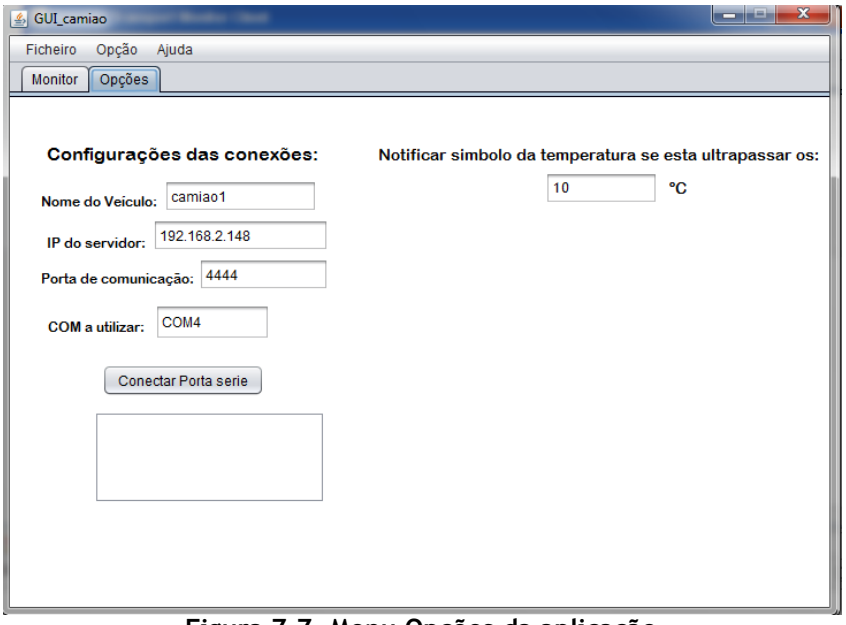


Figura 7.7- Menu Opções da aplicação

De uma perspectiva de alto nível, o software da central comporta-se como a sequência de estados representadas na Figura 7.8.

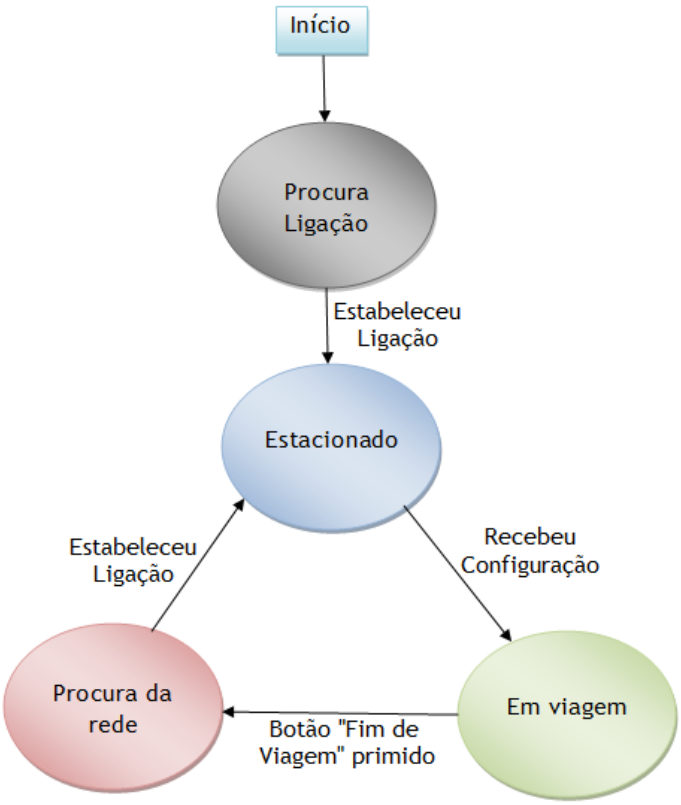


Figura 7.8- Estados da central

No início quando o software é inicializado, este começa por procurar ligação ao servidor. Quando se liga a este passa então para o estado "Estacionado" onde espera por receber a configuração que será utilizada durante a viagem. Quando é recebida uma configuração a central entra no modo "Em viagem" onde começa a monitorizar o processo. Após esta fase se for pressionado o botão "Fim de Viagem" a central começa a tentar conectar-se ao servidor, e quando o consegue envia-lhe o histórico gerado passando outra vez para o estado "Estacionado".

7.3- Gestor

O gestor como já foi dito é responsável por um lado por receber os *log's* provenientes dos camiões (e guardá-los) como também configurar o modo de funcionamento destes. Para isso ele tem de estar previamente ligado ao servidor. Na Figura 7.9 apresenta-se o interface do gestor realizado. O utilizador quando abre esta aplicação é-lhe pedido uma autentificação.

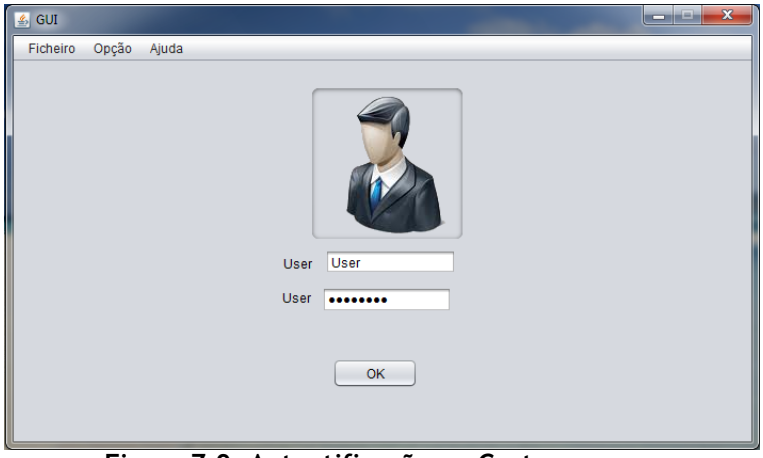


Figura 7.9- Autentificação no Gestor

Depois de o utilizador ser correctamente validado, este depara-se com a interface apresentada na Figura 7.10. Esta é composta por três menus contidos em separadores: o menu de "Configuração", o menu "Histórico" e o menu "Opções".

No menu "Configuração" o operador pode configurar cada um dos camiões que estejam conectados ao servidor com uma das quatro configurações disponíveis (Figura 7.10).

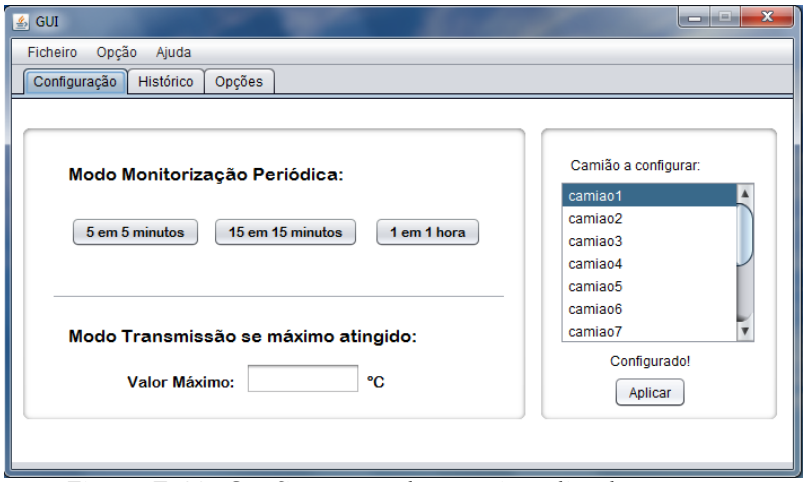


Figura 7.10- Configuração dos camiões ligados

No menu "Histórico" é possível ter acesso a todos os eventos que acontecem na rede, como por exemplo se chegou um camião (e por isso se encontra ligado ao servidor a espera de receber configuração), bem como que camiões já foram configurados, e que por isso já não se encontram ligados ao servidor (Figura 7.11).

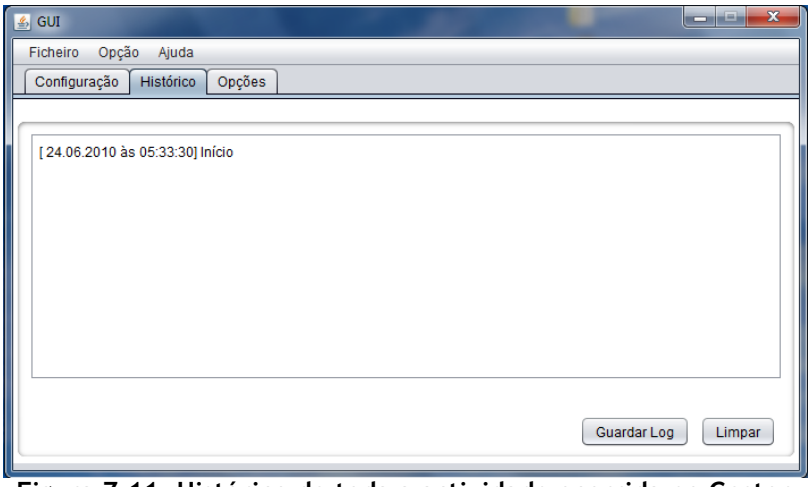


Figura 7.11- Histórico de toda a actividade ocorrida no Gestor

No menu "Opções" o utilizador tem acesso aos parâmetros de ligação ao servidor (IP e porta que este utiliza). O utilizador para começar a monitorização de todo o processo com o gestor deve começar por ligá-lo primeiramente ao servidor usando este separador.

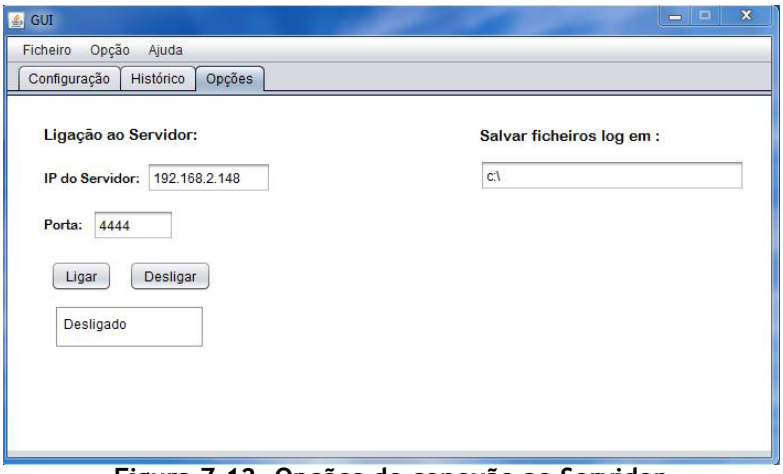


Figura 7.12- Opções de conexão ao Servidor

É possível ainda definir neste menu o local para onde os ficheiros *log's* enviados pelos camiões são guardados no computador. É de salientar que os ficheiros *log's* criados são do tipo *.txt e que apresentam como nome o nome do camião e a data e a hora que o entregou.

Capítulo 8

Testes realizados

8.1- Teste da rede de sensores

Os testes ao funcionamento de cada subsistema foram desenvolvidos gradualmente a medida que estes subsistemas eram realizados. Assim o primeiro subsistema testado foi a rede de sensores, rede constituída pelo micaZ responsável por receber os eventos dos pontos de medições, bem como enviar alterar as configurações destes. Este subsistema se ligaria ao próximo subsistema via RS232. Assim ligamos este nosso subsistema a porta RS232 de um PC e testou-se todas as funcionalidades que este deveria ter, como se ilustra na Figura 8.1.

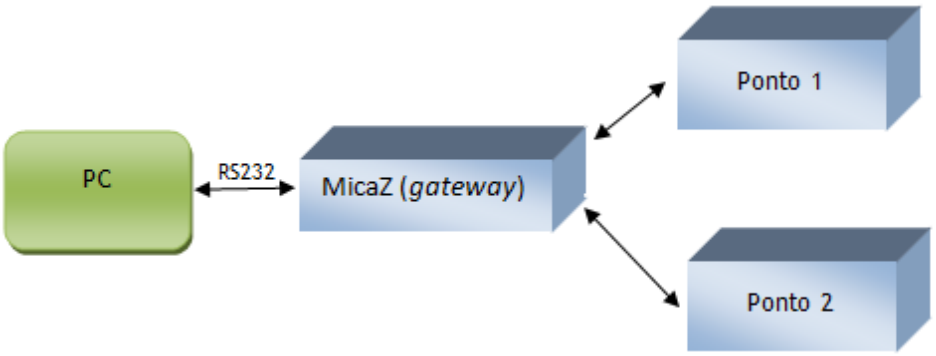


Figura 8.1- Esquema para teste da rede Zigbee

No final deste teste verificou-se então com recurso a um programa de acesso directo a porta RS232 do PC o correcto funcionamento deste subsistema, verificando que este era reconfigurado correctamente de acordo com o envio para este de uma trama especificada, bem como recebíamos as tramas indicadoras de porta aberta e fechada bem como os valores das medições da temperatura nos tempos configurados. Os testes realizados sobre os modos de monitorização da temperatura de cinco em cinco minutos, de quinze em quinze minutos e

de uma em uma hora foram realizados com valores de tempos entre medições mais baixos, por razões óbvias. Utilizou-se em substituição a estas medições de cinco em cinco segundos, de dez em dez segundos e de quinze em quinze segundos respectivamente.

8.2- Teste do subsistema CAN

Após o teste do subsistema constituído pela rede de sensores *Zigbee*, procedeu-se ao teste do próximo subsistema que era composto pelos dois microcontroladores PIC ligados via protocolo CAN. Recorreu-se para a realização deste teste à montagem especificada na Figura 8.2.

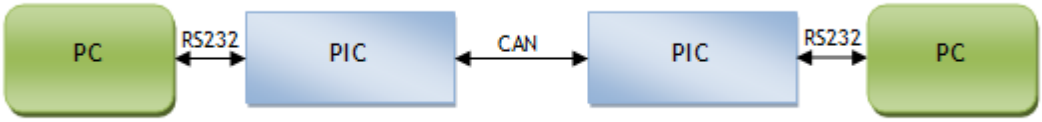


Figura 8.2- Montagem para teste do barramento CAN

Após montagem deste subsistema, outra vez com o auxílio de um programa para monitorização da porta série do PC, verificou-se se o envio de oito bytes consecutivos em um dos PC's, fariam aparecer os mesmos oito bytes no outro PC e vice-versa, que era esta a função que este subsistema deveria realizar.

8.3- Teste de integração

Depois de se ter testado o subsistema CAN, testou-se a aplicação que é executada pela central e a aplicação executada pelo gestor (bem ainda como o servidor). Para este teste ligou-se todos os restantes subsistemas já testados. Ligou-se então primeiramente o servidor, de seguida o gestor e por fim a aplicação da central. Verificou-se no software gestor a entrada na rede da central, devido ao aparecimento desta na lista de camiões (com o nome do camião igual ao designado no software da central). A partir então do gestor enviou-se um dos comandos de configuração, verificando de seguida então a construção do ficheiro *log* com os eventos associados a temperatura medidas bem como os eventos associados ao abrir e fechar da porta e ainda a sinalização de alarmes, tendo se estimulado o sistema para que estes ocorressem. Desligou-se a rede *wireless* da central e carregou-se no botão "Iniciar Viagem" para a simulação da detecção de rede. De seguida conectou-se a rede *wireless* verificando que a central conectava-se ao servidor (através da notificação no gestor) e entregando o log gerado após ter sido configurada. Repetiu-se depois este processo de

configuração para os restantes modos de maneira a verificar os seus correctos funcionamentos e verificando os estados possíveis associados a central exibidos na Figura 7.8.

Assim com estes testes para além de verificar o funcionamento do sistema computacional, verificou-se também ao mesmo tempo o correcto funcionamento da interligação dos vários subsistemas. No entanto há que referir que o sistema não chegou mesmo a ser testado e instalado num veículo, porque como foi dito na secção 4.3.3, não foi comprado nenhuma fonte DC que possibilitasse ligar a central a este. Também não foram testadas situações anómalas de funcionamento bem como não foram injectados erros para análise de resposta/recuperação do sistema. Mesmo assim com os testes realizados pode-se aferir que os conceitos por detrás deste projecto foram comprovados.

Capítulo 9

Conclusão

Neste trabalho de dissertação foi proposto a realização de um sistema que possibilitasse a monitorização da temperatura ocorrente dentro da zona de armazenamento de produtos de um veículo frigorífico, bem como a monitorização da abertura e fecho da porta deste ao longo da viagem. Esse sistema deveria ser composto por nós de medição distribuídos por essa zona de armazenamento para que se conseguisse assim quantificar essa grandeza em diversos pontos. Esses nós deveriam usar tecnologia *wireless* para que não fossem utilizados fios na transmissão de informação, tornando assim o sistema de fácil instalação em qualquer veículo. Pretendia-se no final que o sistema descarregasse todo o registos efectuados ao longo da viagem para um computador, de maneira a gerar-se um registo de toda a viagem no final.

A primeira abordagem ao problema foi de tentar perceber se havia algumas limitações para a concepção deste. Cedo deparou-se com o problema da transmissão ou recepção de informação por um qualquer tipo de protocolo de comunicação sem fios entre a zona interior e exterior à zona de armazenagem dos produtos. Devido a esta ser constituída por materiais condutores, as ondas electromagnéticas ao passar por esta seriam fortemente atenuadas. Chegou-se então a conclusão que seria mesmo preciso usar um protocolo de comunicação com fios para que se conseguisse enviar a informação para o exterior dessa zona.

Já tendo em conta então estas nossas limitações, avançou-se então para uma arquitectura global dos dispositivos integrantes do nosso sistema. Chegou-se a conclusão que o camião seria composto por três tipos de dispositivos: pontos de medições (que estariam espalhados pela zona de armazenagem de produtos, e que seriam responsáveis tanto por efectuar a mensuração da temperatura com uma dada periodicidade, como também um deles pela verificação do estado da porta da zona de armazenagem dos produtos), uma *gateway*

(que seria responsável por receber toda a informação proveniente dos pontos de medição e envia-la via protocolo com "fios" para fora da zona de armazenamento dos produtos) e uma central (cuja responsabilidade assentava tanto por receber dados provenientes da *gateway*, como de enviar no fim da viagem todos os eventos recebidos para um computador).

Após isto constituiu-se como próximo passo estudar os diferentes protocolos de comunicação que seriam usados nas comunicações entre os vários dispositivos. Do estudo realizado no Capítulo 3, concluiu-se que a melhor solução protocolar *wireless* para troca de informações dentro da zona de armazenagem dos produtos congelados era o *Zigbee*, devido a este ser um protocolo de baixo consumo de potência, o que seria uma mais-valia, tendo em conta que esses pontos de medição seriam alimentados a baterias. Neste mesmo capítulo estudou-se também qual o protocolo que seria mais aconselhado para envio dos eventos armazenados na central para um computador exterior no final da viagem. A escolha aí recaiu sobre o Wi-Fi (802.11 b/g/n), porque considerou-se que nesta zona o consumo de potência não era um aspecto crítico, já que nesta zona facilmente se retiraria uma alimentação proveniente do veículo, e ainda devido ao Wi-Fi ser um protocolo de alto alcance.

Paralelamente a isto estudou-se uma solução protocolar com fios, que servisse para passar os dados da *gateway* para a central com alguma fiabilidade tendo em conta o meio. Achou-se então consistente e também uma mais-valia para o projecto a integração do protocolo CAN nesse ponto, por este ter sido criado essencialmente para uso no domínio automóvel, e responder assim as condições adversas que esta parte do nosso sistema estaria sujeita.

Com os protocolos de comunicação a serem utilizados definidos, começou-se então a procura dos dispositivos que apresentassem características que nos permitissem integrá-los no nosso sistema. Daí surgiu então os módulos MicaZ da Crossbow juntamente com a placa de expansão com sensores MTS310 para constituírem os pontos de medição. Para o ponto de medição que estaria encarregado da monitorização da abertura e fecho da porta da zona de armazenamento, construiu-se uma placa de circuito impresso que era composta por um botão ligado a interrupção 3 do MicaZ, conseguindo-se assim com a alteração do estado desse botão a monitorização do estado da porta. Para a *gateway* a solução adoptada foi baseada num MicaZ juntamente com um microcontrolador PIC18F258 ligado via RS232, o qual seria responsável por transportar os dados até a central via protocolo CAN. Por último para a central optou-se por uma solução computacional mais complexa, uma vez que o protocolo Wi-Fi assim o exigia e para também assim possibilitar uma monitorização dos acontecimentos e possíveis situações de alarmes para o condutor de uma forma mais intuitiva. Sugeriu-se então como protótipo uma solução baseada numa *mini-ITX* da Asus ligada via RS232 a um outro microcontrolador PIC18F258, de maneira a essa ter assim acesso as mensagens CAN provenientes da *gateway*. Ligou-se ainda a essa *mini-ITX* uma placa Wi-Fi, possibilitando assim a troca de dados via 802.11 com outro computador.

Com todos os dispositivos integrantes do sistema escolhidos, passou-se então ao desenvolvimento de todo o software necessário para o correcto funcionamento deste. É de salientar que para a central, desenvolveu-se uma aplicação em Java criada para por um lado receber toda a informação gerada pela rede de dispositivos Zigbee, tratando-a de maneira á que fosse gerado um histórico dos eventos ocorridos durante a viagem (bem como alertas de forma visual no caso do acontecimento destes), como também por outro lado conectar-se a um servidor baseado em *sockets* Java TCP para o qual era enviado esse ficheiro *log*, ficando no final a espera de receber uma próxima configuração. Essa configuração era então gerada e enviada através do servidor por outra aplicação desenvolvida (gestor), que através da conexão ao servidor, consegue receber e guardar os ficheiros log recebidos, como também configurar todos os veículos que se encontrem na rede á espera de receber a sua próxima configuração.

É de referir que mesmo sendo esta solução destinada a fins comerciais, a escolha dos componentes integrantes deste sistema não teve em conta (tirando a solução computacional para a central) os preços destes, já que o nosso orçamento era algo restrito e também devido ao facto de por exemplo nos casos dos MicaZ, o sensor MTS310 bem ainda como as DETPIC's já existirem na Faculdade. Assim, uma possível evolução deste projecto deverá ter em conta os custos efectivos de todos os componentes e sua integração.

Referências

[1] "Norma Portuguesa NP EN 12830", Instituto Português da Qualidade, Fevereiro de 2009.

[2] *RS232 Specifications and standard*. Disponível em http://www.lammertbies.nl/comm/info/RS-232_specs.html. Acesso em 25/Janeiro/2010.

[3] *Overview of RS-232*. Disponível em http://hw-server.com/rs232-overview-rs232-standard. Acesso em 25/Janeiro/2010.

[4] *RS-232 serial interface pinout*. Disponível em http://pinouts.ru/SerialPorts/RS232_pinout.shtml. Acesso em 25/Janeiro/2010

[5] "*Comparing Bus Solutions*", Texas Instruments, Fevereiro de 2004.

[6] *RS-485 Bus*. Disponível em http://www.interfacebus.com/Design_Connector_RS485.html. Acesso em 27/Janeiro/2010.

[7] CAN BUS. Disponível em http://www.pcs.usp.br/~laa/Grupos/EEM/CAN_Bus_Parte_2.html. Acesso em 28/Janeiro/2010.

[8] RS-422. Disponível em http://www.interfacebus.com/Design_Connector_RS422.html. Acesso em 3/Fevereiro/2010.

[9] Sangoma. Disponível em http://www.sangoma.com/support/tutorials/rs232_rs422_and_v35.html. Acesso 3/Fevereiro/2010.

[10] Pedro Silva, Luís Almeida, Daniele Caprini, Tullio Facchinetti, Francesco Benzi, Thomas Nolte, "Experiments on timing aspects of DC-Powerline communications", 2009.

[11] IEEE 802.15.1. Disponível em http://www.ieee802.org/15/pub/TG1.html. Acesso em 3/Fevereiro/2010.

[12] Bluetooth Tutorial – Specifications. Disponível em http://www.palowireless.com/infotooth/tutorial.asp. Acesso em 3/Fevereiro/2010.

[13] DATAWEEK. Disponível em http://www.dataweek.co.za/news.aspx?pklNewsId=26651&pklCategoryID=42. Acesso em 3/Fevereiro/2010.

[14] *Introduction to Wi-Fi*. Disponível em http://en.kioskea.net/contents/wifi/wifiintro.php3. Acesso em 3/Fevereiro/2010.

[15] *Dash7 Alliance*. Disponível em http://www.dash7.org/. Acesso em 3/Fevereiro/2010.

[16] DASH7. Disponível em http://en.wikipedia.org/wiki/DASH7. Acesso em 3/Fevereiro/2010.

[17] Zigbee Alliance. Disponível em www.zigbee.org/. Acesso em 3/Fevereiro/2010.

[18] EETimes. Disponível em http://www.eetimes.com/showArticle.jhtml?articleID=194300580. Acesso em 3/Fevereiro/2010.
[19] Microchip. Disponível em http://www.microchip.com/. Acesso em 3/Fevereiro/2010.
[20] Nano-RK. Disponível em http://www.nanork.org/wiki/FireFly. Acesso em 4/Fevereiro/2010.
[21] Crossbow, Micaz, Wireless Measurement System. San Jose, California.
[22] Crossbow, MTS/MDA Sensor Board Users Manual. Junho, 2007. http://www.nanork.org/wiki/firefly-sensor-basic. Acesso em 4/Fevereiro/2010.
[23] ATMEL. Disponível em http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4427. Acesso em 4/Fevereiro/2010.
[24] ATMEL. Disponível em http://atmel.com/dyn/products/tools_card_mcu.asp?tool_id=4659&source=redirect. Acesso em 4/Fevereiro/2010.
[25] "Sistemas de monitorização de temperatura", TranScan.
[26] Grupo Vei. Disponível em http://www.grupovei.pt/. Acesso em 6/Fevereiro/2010.
[27] P. Levis, D. Gay, *TinyOS Programming*. Cambridge University Press, 2009, pp. 4-9.
[28] D. Ibrahim, *Advanced PIC Microcontroller Projects in C*. British Library Cataloguing, 2008, capítulo 9.
[29] HI-TECH Software, *HI-TECH PICC-18 Compiler*, HI-TECH Software, 2006.
[30] Atmel, 8-bit AVR Microcontroller with 128K Bytes In-System Programmable Flash ATmega128, Atmel Corporation, 2009.
[31]Texas Instruments, Introduction to the Controller Area Network (CAN), Texas Instruments Incorporated, Julho 2008.
[32]Bosh, Can Specification Version 2.0, ROBERT BOSCH GmbH, Março de 1997.
[33]S.Ergen, ZigBee/IEEE 802.15.4 Summary, Setembro de 2004.
[34]IEEE 802.15.4, Chun-Yi Chen, Setembro de 2007.
[35]ZigBee Alliance, ZigBee and Wireless Radio Frequency Coexistence, 2007.
[36] Freescale, 802.15.4 MAC PHY Software, Março de 2010.
[37]Lukas Krammer, IEEE 802.15.4 MAC API, Maio de 2008.
[38] Relatório de Zigbee. Disponível em www.fe.up.pt/~ee02055/**Relatorio_ZigBee_Hardware**.pdf. *Acesso em 6 de Junho de 2010.*