Faculdade de Engenharia da Universidade do Porto

FEUP

Rede domótica KNX sobre rede física CAN

Tiago Manuel da Silva Mendes

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

Orientador: Prof. Dr. Mário de Sousa
Co-orientador: Prof. Dr. Paulo Portugal

Agosto de 2010

Resumo

O KNX é um protocolo normalizado para uso em redes de domótica, que tem tido cada vez mais adesão no mercado (ABB, Siemens, etc.). Foi desenvolvido em Maio de 1999, numa parceria das associações EIBA, EHSA e BCI, com o objectivo de desenvolver um standard internacional para o controlo de habitações e edifícios. No entanto, a solução de automação de edifícios, utilizando o protocolo KNX, perde pelo seu elevado custo, sendo uma das suas razões, a interface à rede física. Pretende-se com este trabalho desenvolver uma forma de encapsular o protocolo KNX numa rede física de baixo custo, o CANbus. Este último, foi desenvolvido pela empresa alemã Bosch em meados dos anos 80, com o intuito de servir a área motorizada mas, devido à sua comprovada fiabilidade e robustez tem sido, também, utilizado em aplicações industriais e em sistemas que necessitam de controlo distribuído em tempo real.

Abstract

The KNX is a standard protocol for use in home automation networks, which has been increasing its acceptation in the market (ABB, Siemens, etc.)...It was developed in May 1999, a partnership of the following associations EIBA, EHSA, BCI, with the aim of developing an international standard for homes and buildings control. However, this solution loses by its high cost, one reason of that is the cost of the physical network interface. The aim of this work is basically reduce that cost by specifying a way to encapsulate the KNX protocol in a physical network of low cost, the CANbus, which was developed by the German company Bosch in the mid 80s, with the purpose to serve the motorized area but, because of its reliability and robustness also has been used in industrial applications and systems that need distributed real time control.

Agradecimentos

Queria agradecer a todos que tornaram este trabalho possível: A toda a comunidade da Faculdade da Engenharia da Universidade do Porto em especial ao meu Orientador, Professor Dr. Mário de Sousa. A toda a minha família por todo o apoio dado nesta minha caminhada. Aos meus grandes amigos André Leite e Luís Cardoso por toda a confiança, amizade e ajuda prestada. A todos aqueles que de uma forma directa ou indirecta contribuíram para a realização deste trabalho. E um especial obrigado à minha namorada Felisbela por toda a compreensão, ajuda e palavras de incentivo e confiança.

Índice

Lista de figuras

Lista de tabelas

Abreviaturas e Símbolos

Lista de abreviaturas

ABB – Empresa multinacional fundada pela união de três empresas: ASEA, Brown, Boveri & Cie

BCI - Batibus Club International

BCU – *"Bus Coupling Unit"*: Componente dos dispositivos EIB/KNX responsável pela ligação à rede

Bits/s - Bits por segundo.

CAN - Control Area Network

CCTV - Closed-circuit television

Dispositivo EIB/KNX - Elemento físico que comunica em EIB/KNX e que está ligado a uma rede EIB/KNX

EIS - EIB *Interworking Standards*

ETS - EIB *Tool Software*

EIBA - *European Installation Bus Association*

EHSA - European Home Systems Association

EIB/KNX - Protocolo da associação Konnex

LC - *"Line Coupler"*

OSI - Open Systems Interconnection

SMD - *Surface Mounted Components*

SMT - *Surface-mount technology*

Trama - Sequencia de octetos sempre com a notação little-endian

TP - Sigla de *Twisted pair*, Par entrançado

Capítulo 1

Introdução

Com o desígnio de melhorar a qualidade de vida, aumentar o bem-estar e a segurança, a Domótica, também conhecida como uma Ciência moderna de engenharia das instalações em edifícios, que pretende aplicar novas tecnologias tornando os edifícios inteligentes, é uma área que está a suscitar cada vez mais o interesse de clientes particulares, crescendo exponencialmente a sua procura por oferecer soluções que proporcionam maior conforto, e mesmo, a redução no gasto de energia eléctrica.

A Domótica incide sobre quatro vectores fundamentais:

- **Energia:** regulação de temperatura, gestão dos consumos de cada electrodoméstico e da potência contratada.
- **Segurança:** detecção de intrusão, incêndio, inundação, fuga de gás, etc.
- **Comunicação:** controlo remoto, telemetria, correio electrónico, Internet, etc.
- **Conforto:** Programações horárias, cenários luminosos, rega automática, etc.

Para conseguir tudo isto, a Domótica usa um conjunto de dispositivos que são distribuídos pela casa em função das necessidades dos proprietários. Estes dispositivos podem dividir-se em sensores, actuadores, controladores, interfaces e dispositivos específicos que, quando interligados, constituem o que se designa por Rede Domótica.

Os sensores capturam valores e informações do local, como: presença de pessoas, temperatura, falta de energia, fugas de água ou gás, incêndio, luminosidade, tempo, vento, humidade… Com estas informações, pode-se realizar o controlo de actuadores de elementos como - estores, portas, rega; funções - ligar, desligar - e, variar a iluminação ou o aquecimento, ventilação, sirenes de alarme… Para isso, será necessário o recurso de controladores que gerem a instalação de forma a receber as informações provenientes dos sensores e, assim, fazer operar os respectivos actuadores. Para que a informação seja transmitida, de e para, o utilizador são utilizadas interfaces como *Displays*, PCs, PDAs, *Webpads*, Consolas, Telefone, Telemóvel, Internet. Existem, ainda, outros elementos

necessários ao funcionamento do sistema, tais como, Modems, Routers ou Gateways, os quais permitem o envio de informação entre os diversos meios de transmissão onde viaja a mensagem.

Geralmente, é mais fácil e mais completo equipar uma casa durante a construção, devido à acessibilidade das paredes e à capacidade de fazer alterações de design, especificamente, para acomodar certas tecnologias. Os sistemas sem fio são geralmente utilizados em casas já construídas, pois evita a necessidade de fazer grandes mudanças estruturais. Estes comunicam via rádio ou sinais infravermelhos com um controlador central.

1.1 - Resumo de redes domótica

É grande a variedade de protocolos normalizados direccionados para a Domótica. Entre os mais usuais protocolos de comunicação, existentes actualmente no mercado, está o X-10, considerado o mais antigo e que teve muita aceitação, principalmente, por ser económico. Mas a sua grande desvantagem consiste na fraca robustez e no seu carácter rudimentar. Sistemas que utilizam este protocolo são facilmente encontrados pois não necessitam de instaladores profissionais e aproveitam a instalação eléctrica como meio de comunicação.

Em 2002 foi desenvolvido um novo protocolo de domótica, Powerline Comunication Bus (PLC-BUS), que utiliza a rede eléctrica também como suporte de comunicação. Este protocolo de domótica foi desenvolvido pela ATS na Holanda. A comunicação digital totalmente encriptada e a bi-direcionalidade são as principais características deste protocolo. O PLC-BUS, apresenta diversas vantagens concorrenciais quando comparada com outros protocolos de domótica por corrente portadora, nomeadamente, na fiabilidade e rapidez de execução das ordens executadas. Nos testes comparativos efectuados verificou-se uma rapidez muito superior aos sistemas por corrente portadora conhecidos, nomeadamente o X10 [7].

O protocolo, mais fiável e mais utilizado em sistemas domésticos existentes no mercado, é o EIB/KNX. O protocolo EIB/KNX tem sido desenvolvido dentro do contexto da União Europeia, com o fim de fazer frente aos produtos domóticos já existentes na América e Japão. Este protocolo permite a utilização de vários meios físicos, mas o mais utilizado é o par entrançado (TP). Este protocolo oferece muita robustez e flexibilidade, contudo, os produtos são de elevado custo. Para exemplificar, um interruptor de pressão quadruplo da Siemens custa aproximadamente 70,91€ [2].

Existem, ainda, outros protocolos de domótica mas, não são muito usuais e raramente se encontram no mercado [3].

Tabela 1.1 - Protocolos de Rede utilizados na Domótica [3].

Protocolos de Rede - Alianças e Grupos de Trabalhos		
Standard	**Meio Físico**	**Descrição**
BatiBUS	Par Entrançado	Sensores de união e actuadores para construir sistemas que controlem HVAC (Ar Condicionado), sistemas de segurança e acesso. Em convergência com EIB e EHS para KNX.
CEBus (Consumer Electronics Bus)	Todos	O Standard CEBus (EIA-600) é um protocolo criado pela Associação de Industrias Electrónicas (EIA) para ser possível a interligação e comunicação entre dispositivos electrónicos da casa.
EIB (European Installation Bus)	Par Entrançado	Sensores e actuadores para construir sistemas que controlem HVAC (Ar Condicionado), sistemas de segurança e acesso. Em convergência com EHS e BatiBus para KNX.
EHS (European Home System)	Todos	Uma colaboração entre indústrias e governos Europeus sobre Domótica. Entre alguma das suas missões a EHSA tem o objectivo de ser um standard na Europa de um BUS comum (EHS). Em convergência com EIB e BatiBus para KNX.
HomeRF (Home Radio Frequency Working Group)	RF	A missão do grupo de trabalho HomeRF é tornar possível uma vasta gama de produtos electrónicos de consumo que operem entre si, estabelecendo uma especificação aberta para comunicações digitais de RF (sem licença), para PC,s e produtos electrónicos de consumo em qualquer sitio e arredores da casa.
LonMark Interoperability Association	Todos	A associação LonMark tem a missão de integrar facilmente dispositivos baseados em redes LonWorks, fazendo uso de ferramentas e componentes standards.
ZIGBEE	RF	Pensa-se que este pode ser um dos standards que irá ser bastante utilizado no mundo da domótica. É uma versão melhorada do HomeRF e destinada a ambientes industriais.
Protocolos de Rede - Proprietários		
Lonworks Echelon Corp.	Todos	Redes de controlo comerciais e para a casa. Uma rede LonWorks é grupo de dispositivos trabalhando juntos para monitorizar, comunicar, e de algumas maneiras controlar. É muito parecido com uma LAN de PCs.
X-10	Corrente Eléctrica/RF	O protocolo mais utilizado na domótica, utiliza a rede eléctrica e facilita o controlo de dispositivos domóticos sem instalação de qualquer fio em casa, por utilizar a instalação de rede eléctrica já existente.
PLC-Bus	Corrente Eléctrica	O protocolo que utiliza a rede eléctrica e facilita o controlo de dispositivos domóticos sem instalação de qualquer fio em casa, por utilizar a instalação de rede eléctrica já existente.

1.2 - Domoitech

O projecto Domoitech foi realizado no ano lectivo de 2006/2007 com o propósito de criar uma configuração e equipamento para domótica compatível com soluções existentes no mercado, a um preço reduzido. Este projecto apresenta uma arquitectura descentralizada, constituída por dispositivos que permitem ligar várias entradas e saídas. É composto por uma placa principal, onde é efectuado o processamento do protocolo de domótica. É neste placa que é possível interligar sensores e actuadores com o recurso de placas secundárias de protecção e controlo de potência. Em cada placa principal é possível conectar até 7 placas secundárias. A Figura 1.1 representa a arquitectura de hardware da placa principal, e dos dois tipos de placas secundárias possíveis:

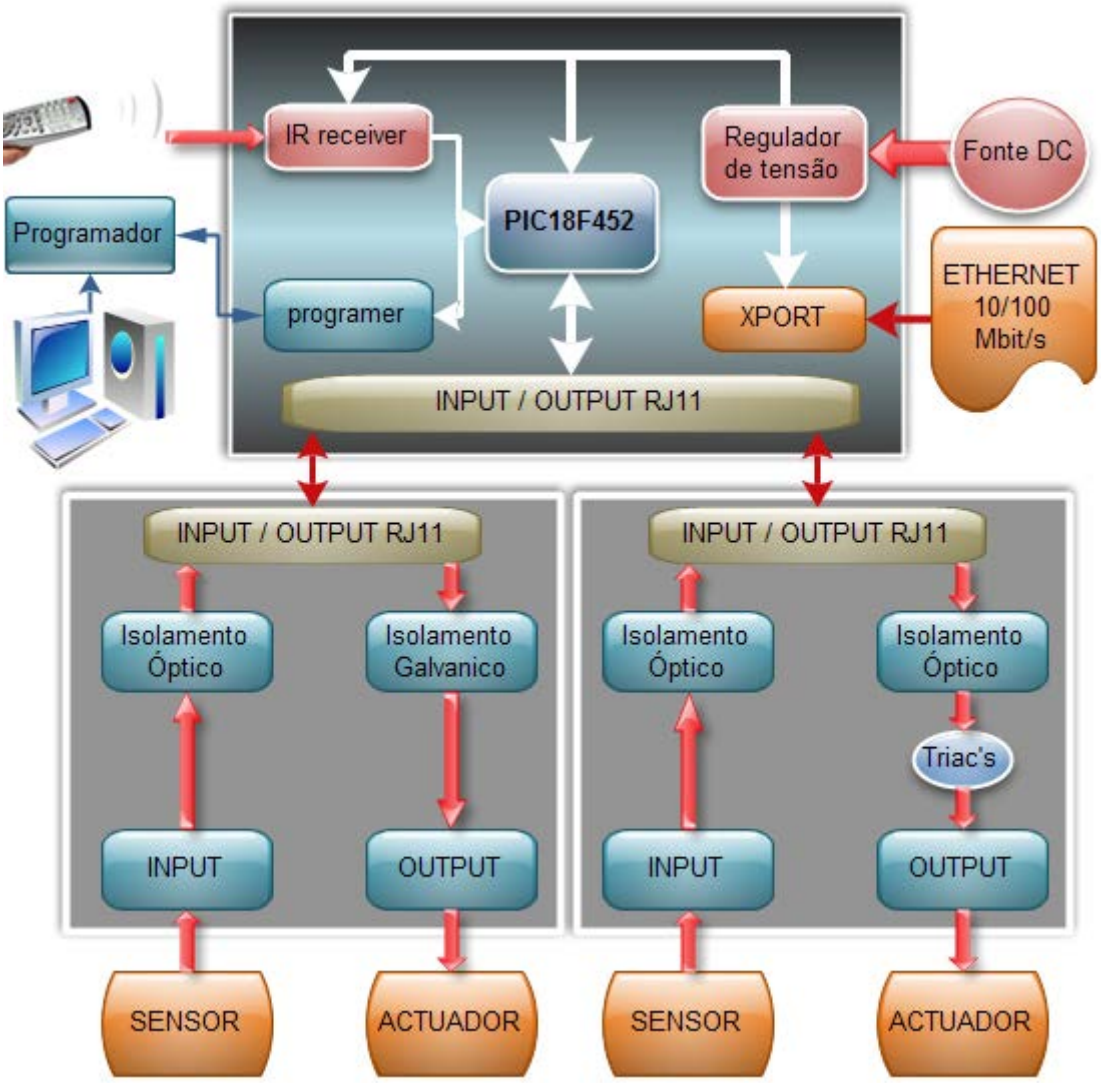

Figura 1.1 - Diagrama de blocos da estrutura de hardware Domoitech

Analisando a Figura 1.1, podemos observar um bloco que corresponde à placa principal representada pelo bloco superior que contém: um receptor de infravermelhos, reguladores de tensão, uma interface para o programador da placa, entradas/saídas RJ11 para ligar as placas secundárias, um microcontrolador PIC18F452 e uma XPORT. O microcontrolador é responsável pelo processamento que contém o protocolo de domótica e efectua leituras/escritas de dados, das placas secundárias e da XPORT. A XPORT efectua uma conversão bidireccional entre o protocolo RS232 e TCP/IP sob *Ethernet*. As placas secundárias podem ser de dois tipos, utilizando relés ou *triac's* para os actuadores. Estas placas secundárias contêm duas interfaces para entradas e duas para as saídas e os respectivos isolamentos ópticos e galvânicos.

Com a arquitectura descentralizada do Domoitech, é possível ter um computador central que poderá funcionar como servidor *web*, permitindo assim o acesso pelo exterior de forma a controlar remotamente os equipamentos, e realizar algumas tarefas de gestão da rede. Nesta solução não é possível configurar as placas principais sendo todas as configurações efectuadas no momento de programação do PIC18F452.

No projecto Domoitech, foi utilizado como meio físico o Ethernet e o protocolo de rede TCP/IP sob *Ethernet*. O KNX também suporta esse meio físico e tem uma norma denominada por KNXnet/IP para a utilização desse mesmo meio em redes IP.

Na altura em que o Domoitech foi desenvolvido, o protocolo utilizado era o EIB e, devido a este facto, o Domoitech não utilizou a norma KNXnet/IP e utilizou o *Ethernet* TCP/IP para simular um barramento *Twisted-Pair* do EIB. Isto resulta numa incompatibilidade com outros equipamentos KNX/EIB porque o acesso ao meio físico não respeita a norma. Apesar de o Domoitech codificar/descodificar tramas KNX/EIB, falta-lhe saber as regras para envio/recepção dessas mesmas tramas sob o protocolo de rede TCP/IP.

Para a comunicação por infra-vermelhos foi utilizado o protocolo RC5 desenvolvido pela Philips, que hoje o encontramos facilmente em diversos aparelhos electrónicos no nosso quotidiano. Trata-se de um protocolo relativamente simples, onde os dados referentes à tecla pressionada são enviados via infravermelho com uma modulação em frequência, sendo esta tipicamente 36 kHz. A codificação adoptada por este protocolo é do tipo Manchester, onde o bit 1 e bit 0 podem ser observados na Figura 1.2.

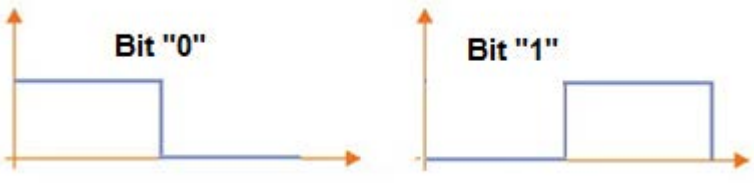

Figura 1.2 - Codificação dos bits no protocolo RC5

O tempo total dum bit no RC5 é tipicamente de 1779 µs e desta forma, se multiplicarmos por 14 bits obtemos o tempo correspondente a uma trama completa que é de

24.889ms. As tramas RC5 são enviadas com um período de 113.778ms que corresponde ao tempo que demoraria a enviar 64 bits.

A Tabela 1.2 apresenta o preço da solução Domoitech dando um exemplo de um total que consiste numa placa principal com as sete possíveis placas secundárias que no caso são duas placas de relés e cinco placas de triac's.

Tabela 1.2 - Custo das placas da solução Domoitech

Placa Principal	66 €
Placa secundária relés	11 €
Placa secundária triac's	8 €
Total (1+2+5)	128 €

1.3 - Motivação e Objectivos

A domótica oferece um elevado conjunto de benefícios aos seus utilizadores, como o conforto proveniente da automação de determinadas tarefas e actividades, bem como pelo aumento da segurança dos edifícios através de sistemas de vigilância, bastante utilizados actualmente, até a uma melhor gestão e consequente poupança energética.

O KONNEX, mais conhecido por KNX, promovido por três associações Europeias EIBA (European Installation Bus Association), BCI (Batibus Club International), EHSA (European Home Systems Association), teve como objectivo criar um único standard para a domótica e automação de edifícios. Actualmente o KNX é uma tecnologia comercial madura, bastante divulgada e com grande implantação no mercado, em virtude de ser uma solução normalizada internacionalmente, bastante completa e oferecendo um elevado nível funcional. Contudo, o seu ponto fraco e o qual despertou o interesse para o desenvolvimento deste projecto, incorre no facto desta solução se revelar bastante dispendiosa, no que se refere aos custos associados á sua implementação e, essencialmente, aos custos associados à sua interligação com a rede física. Daí a necessidade de desenvolver uma forma de encapsular esta tecnologia numa rede de baixo custo, propondo-se para essa função, o barramento CAN, pelas suas características de robustez e de fiabilidade, bem como, a sua aceitação no mercado.

1.4 - Arquitectura Geral da Solução Proposta

O sistema EIB/KNX (a ser detalhado no capítulo seguinte) está no mercado há mais de 20 anos e os seus componentes da primeira geração continuam compatíveis com os que estão a ser produzidos actualmente e a realizar todas as funções para que foram construídos. Um grande problema deste sistema, continua a ser o seu elevado custo, associado à interligação à rede física. Desta forma, pretende-se com este trabalho reduzir esse custo, encapsulando a informação EIB/KNX numa rede física CAN. Para a realização deste objectivo foi estruturado um sistema composto por dois módulos, onde um deles deu-se a designação de DOMO_GATMONITOR e o outro de DOMO_CAN. O DOMO_GATMONITOR apresenta entre outras, as seguintes funcionalidades:

- Monitorizar o sistema, poder observar o fluxo das tramas EIB/KNX que circulam no barramento CAN;
- Configurar o sistema de forma a alterar o que cada dispositivo controla ou o que fará actuar um dispositivo;
- Enviar e receber ficheiros de configuração.

O DOMO_CAN tem como funcionalidade interligar dispositivos simples como, interruptores ou lâmpadas, sem a necessidade de estes possuírem propriedades particulares e, sendo a comunicação entre os dispositivos (lâmpadas, interruptores…) efectuada e difundida no barramento CAN. Para descrever o sistema segue-se a Figura 1.3 - Diagrama de blocos que descreve o sistema.

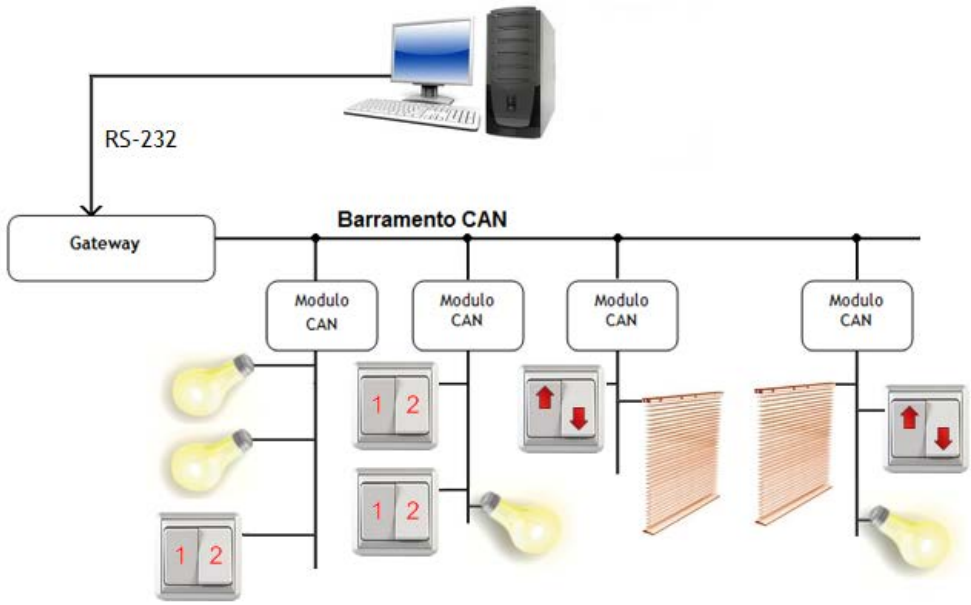

Figura 1.3 - Diagrama de blocos que descreve o sistema

Estes dois módulos serão descritos com detalhe a nível do hardware no Capítulo 4 e a nível de software implementado no Capítulo 5 deste documento.

Quando um dispositivo de um módulo DOMO_CAN gera informação necessária a ser difundida, este módulo cria tramas EIB/KNX e envia esta informação sobre a rede física CAN. Estas tramas são capturadas por todos os módulos presentes na rede para, de seguida, a informação ser processada e verificar-se se interessa ou não. Em caso afirmativo o módulo para o qual a trama EIB/KNX era destinada irá fazer actuar conforme a configuração que nele está presente.

Foi ainda desenvolvida uma aplicação em Java (para permitir ser utilizada em diferentes sistemas Operativos sendo necessário apenas ter instalada a *Java Virtual Machine)* que comunica com o módulo DOMO_GATEMONITOR e permite efectuar as opções pretendidas como as de configurar o sistema. Esta aplicação tem o nome de DOMO_CAN Monitor.

1.5 - Organização deste Documento

Este documento encontra-se dividido em seis capítulos distintos. Podemos encontrar no primeiro capítulo um estudo da arte, uma descrição do sistema a desenvolver e uma apresentação de um trabalho realizado no ano lectivo de 2006/2007 com o propósito de criar uma configuração e equipamento para domótica compatível com soluções existentes no mercado, a um preço reduzido, utilizado no desenvolvimento deste projecto.

No segundo capítulo é descrito em pormenor o protocolo KNX/EIB dando detalhe ao que se encontrava implementado no projecto Domoitech.

No capítulo três encontramos o protocolo utilizado para a camada física deste projecto, o protocolo CAN.

Os dois capítulos seguintes, ou seja, o capítulo quatro e cinco, referem-se a todo o trabalho desenvolvido neste projecto. A separação em dois capítulos distintos deve-se ao facto de no capítulo quatro se descrever todos os detalhes de hardware do projecto, ou seja todos os requisitos ao nível do hardware que foram tomados em conta, bem como, pesquisas e circuitos eléctricos desenvolvidos. E no capítulo cinco, detalha-se todo o software criado para os dois módulos e a aplicação criada em Java.

Por fim, no capítulo seis, são descritas as conclusões retiradas ao longo do desenvolvimento do projecto, comparações de preços finais do projecto e melhoramentos futuros possíveis para o trabalho realizado.

Capítulo 2

Protocolo EIB/KNX

O EIB é um sistema de automação desenvolvido por um conjunto de empresas líderes no Mercado Europeu do material eléctrico, com o objectivo notório de criar um sistema standard Europeu que permita a comunicação entre todos os dispositivos de uma instalação e constituir uma barreira às importações de produtos e sistemas semelhantes, produzidos nos mercados Japonês e dos Estados Unidos da América, onde estas tecnologias possuíam, um grau de maturidade superior ao produzido na Europa.

O EIB possui uma arquitectura descentralizada. Ele define uma relação elemento a elemento entre os dispositivos, o que permite distribuir a inteligência entre os sensores e actuadores instalados.

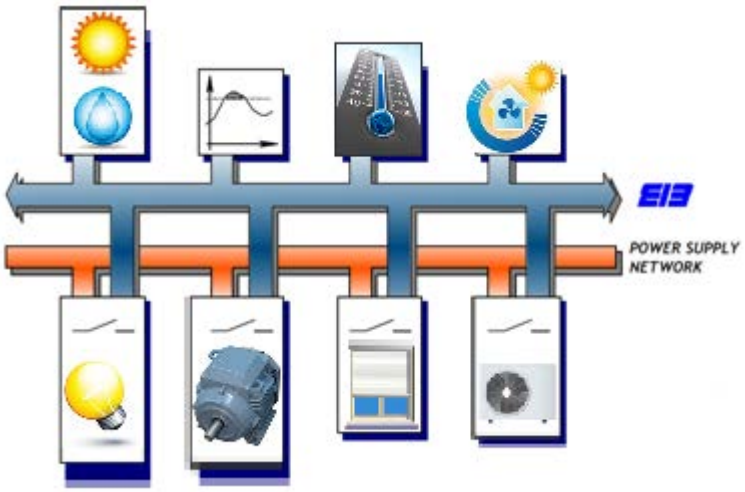

Figura 2.1 - Arquitectura descentralizada do EIB/KNX, baseada [20]

Segundo a EIBA, associação responsável por implantar o uso do sistema de instalação inteligente EIB, existem cerca de 10 milhões de dispositivos EIB instalados por todo o Mundo e

umas 70.000 instalações realizadas. Dispõe de uma gama de 5.000 produtos certificados e 70.000 instaladores qualificados [1].

Uma das principais áreas de actuação desta associação é a iniciativa de "convergência" KONNEX, mais conhecido por KNX, juntamente com o BCI (Batibus) e a EHSA (EHS). O KNX, como já referido anteriormente, foi promovido, com o objectivo de criar um único standard para a domótica e automação de edifícios que possa cobrir todas as necessidades e requisitos das instalações profissionais e residenciais no âmbito europeu. Tem em vista, melhorar as prestações dos diversos meios físicos de comunicação, sobretudo na tecnologia de radiofrequência, fundamental para a efectiva consolidação da domótica e para introduzir novos modos de funcionamento, que permitam aplicar uma filosofia Plug&Play a muitos dispositivos típicos de uma casa.

Sintetizando, partindo dos sistemas EIB, EHS e Batibus, trata-se de criar um único standard europeu que seja capaz de competir em qualidade, prestações e preços, com outros sistemas norte-americanos como o Lonworks ou CEBus.
Actualmente, o KNX contempla três modos de funcionamento:

1. **S-mode** (System mode): este modo de funcionamento usa a mesma filosofia que o EIB actual, isto é, os diversos dispositivos ou modos da nova instalação, são instalados e configurados por profissionais com ajuda de um software (ETS) especialmente concebido para este propósito. Este modo está pensado para o uso em instalações mais complexas que impliquem um elevado nível de integração e de funções a implementar. Os dispositivos S-mode só poderão ser comprados através de distribuidores especializados.
2. **E-mode** (*Easy mode*): na configuração do modo simples os dispositivos são programados em fábrica para realizar uma função concreta. Mesmo assim devem ser configurados alguns detalhes no local da instalação mediante o uso de um controlador (como uma porta residencial ou uma set-top-box) ou mediante micro-interruptores alojados nos dispositivos (semelhante ao X-10 ou outros dispositivos proprietários que há no mercado).
3. **A-mode** (*Automatic mode*): na configuração do modo automático, com uma filosofia Plug&Play, nem o instalador nem o utilizador final têm de configurar o dispositivo. Este modo será especialmente indicado para ser usado em electrodomésticos e equipamentos de entretenimento (consolas, set-top boxes, áudio e vídeo).

A Figura 2.2 apresentada a seguir, ilustra o que foi descrito um pouco durante o início deste capítulo 2, que, como podemos verificar, o sistema de base do KNX é o EIB e que para todos os modos de configuração acima descritos, existe um *software* criado pela KONNEX com

o nome ETS da abreviação de *Engineering Tool Software* para que se possa configurar ou reconfigurar os dispositivos KNX/EIB.

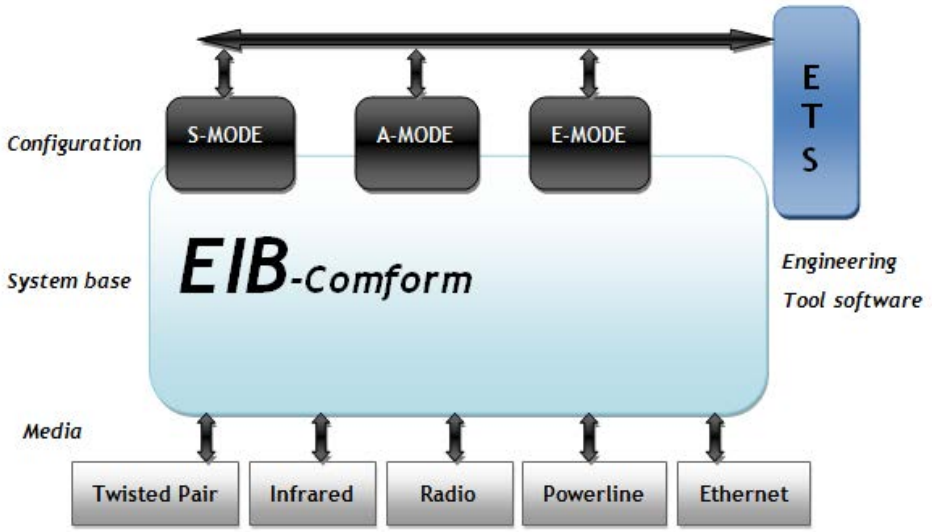

Figura 2.2 - Modelo do sistema KNX, baseada [3]

2.1- Componentes do Sistema EIB/KNX

Os componentes do sistema EIB/KNX são componentes que suportam as operações básicas do sistema, encontrando-se entre eles o Bus Coupling Units (BCU), os Line Couplers (LC), os Área Couplers, os Repeaters, etc.

O "repeater" é um repetidor de sinal para as redes par entrançado. A sua função é regenerar os sinais eléctricos permitindo assim criar redes maiores, necessárias a edifícios grandes.

O "Line Coupler" garante a filtragem e o encaminhamento dos pacotes de dados, da rede principal para o seu segmento e vice-versa, o que implica uma melhor gestão do tráfego global da rede, ou seja, este componente une ramificações da rede EIB/KNX à rede principal [15].

O "Area Coupler" o seu funcionamento é bastante semelhante ao componente "Line Coupler". Contudo, este componente é para ligar uma área à linha *Backbone* da rede a que pertence.

O Bus Coupling Units (BCU) tem como funcionalidade interligar dispositivos EIB/KNX à rede física. Estes podem ser dispositivos independentes através dos quais os módulos de aplicação se conectam à rede ou, em alternativa, podem estar integrados nestes dispositivos. Um BCU é composto por um módulo de transmissão e um módulo controlador (Communication Controler). O módulo de transmissão é responsável pelo envio e recepção de dados da rede.

2.2- Topologia

O EIB/KNX é uma rede distribuída que pode ter até 65536 dispositivos com endereços individuais de 16 bit podendo ser esses endereços de dois tipos: de grupo ou individual. Normalmente, a denotação dos endereços é feita da seguinte forma "x.x.xx", em que cada "x" representa um número decimal de 4bits. Deste modo, podemos construir endereços desde 0.0.00 até 15.15.255. O endereço de grupo não necessita de ser único, pelo que um dispositivo pode ter mais do que um endereço de grupo, por exemplo, um sensor pode fazer actuar vários dispositivos que estejam num grupo ou, apenas só um, através do endereço único do dispositivo a actuar.

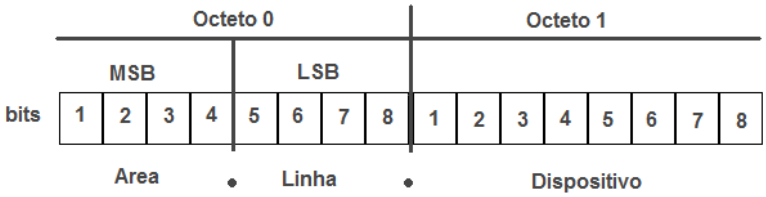

Figura 2.3 - Denotação dos endereços

Como referido, o endereço individual é único em cada dispositivo EIB/KNX e na denotação ("x.x.xx" em que cada x=4bits), o primeiro número decimal corresponde ao endereço da área, que são os 4 bits MSB do octeto 0 e o segundo número decimal, corresponde ao endereço da linha e são os 4 bits LSB do octeto 0. O octeto 1 é o que tem o endereço do dispositivo e corresponde ao terceiro número decimal na denotação dos endereços com dois pontos.

Como apenas 8bits representam o endereço do dispositivo, assim o número máximo de dispositivos é 256. A Figura 2.4 ilustra um exemplo de uma rede EIB/KNX com o *backbone* onde são ligadas várias áreas que contém as diversas sub-redes.

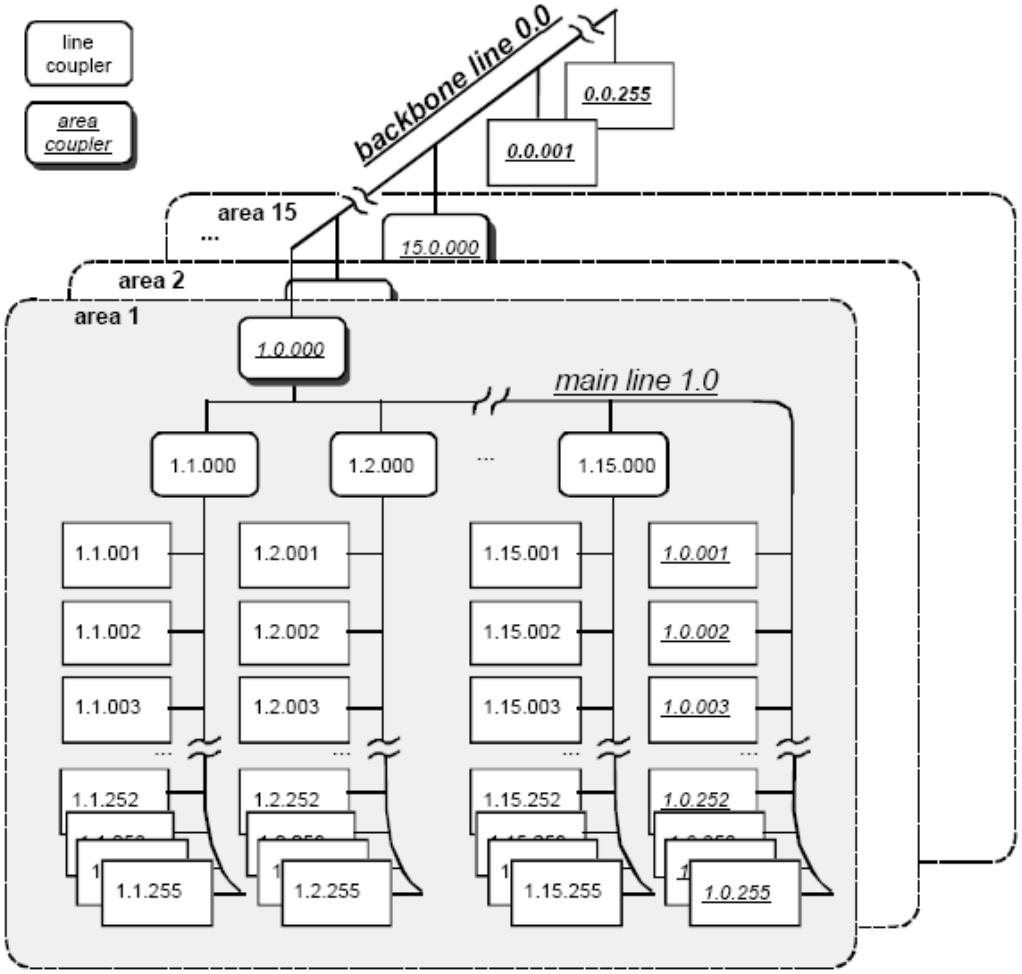

Figura 2.4 - Estrutura de uma rede [20]

Cada área tem uma linha principal (*main line*) onde vão ser ligadas as várias sub-redes que se denominam por linhas. Em cada área é possível ter até 15 linhas pois o seu endereçamento utiliza apenas 4bits e o endereço 0 é reservado para o acoplador entre a linha principal e o *backbone*.

Ligados ao *backbone* poderemos ter também no máximo 15 áreas pois o seu endereçamento utiliza também 4 bits e o endereço 0 é reservado para definir os dispositivos EIB/KNX que se ligam directamente ao *backbone*.

2.3- Formato da trama

Na Figura 2.5 podemos verificar a estrutura de uma trama normal EIB/KNX. As tramas estendidas (actualmente suportadas por poucos dispositivos), são semelhantes às tramas normais, ou seja, não estendidas, com a diferença de suportarem um tamanho máximo de 254 octetos, o que revela ser muito superior na quantidade de octetos suportados por cada trama, uma vez que as tramas do tipo não estendido, apenas suportam 23 octetos.

Figura 2.5 – Formato de uma trama EIB/KNX

O campo *de* controlo identifica a prioridade da trama bem como o seu tipo, isto é, se esta se trata de uma trama normal ou de uma trama estendida. Os campos de endereço de origem e de destino contêm como o próprio nome identifica os respectivos endereços dos dispositivos de origem e destino da trama. O endereço de origem é sempre um endereço físico mas, o endereço de destino pode ser um endereço físico ou um endereço de grupo. O campo tipo de endereço da trama irá fornecer a identificação do tipo de endereço utilizado, ou seja, se é um endereço de grupo ou individual. O campo *NPCI Length* contém o número máximo de saltos (*hops*) da trama, para que em cada router faça a decrementação deste número, até que este chegue a zero, sendo então a trama descartada.

O campo TPCI (*Transport Layer Protocol Control Information*) administra as relações de comunicação da camada de transporte, como por exemplo, para estabelecer e manter uma ligação ponto-a-ponto, e o campo APCI (*Application Layer Protocol Control Information*) controla os serviços da camada de aplicação (leitura, escrita, resposta…) que estão disponíveis entre os intervenientes da comunicação.

Os octetos que se seguem são referentes aos dados a enviar e dependendo, evidentemente, da função pretendida, podem ocupar até 14 octetos numa trama normal. Para os casos em que é necessário transferir dados superiores a 14 octetos, por exemplo em caso de o utilizador pretender carregar uma aplicação para um BCU, é necessário separar os dados em várias tramas, sendo essa operação do encargo da ferramenta de gestão. Por último, o campo *Frame Check* permite corroborar a solidez e fiabilidade da trama.

2.4- Meios físicos

Uma das grandes vantagens do EIB/KNX, é não estar aprisionado pelo meio físico a utilizar. A utilização do KNX/EIB noutros meios físicos que não estão referidos na norma é possível, caso o fabricante assim o pretenda. No entanto, é de notar que o software disponibilizado, apenas se encontra configurado para os meios físicos descritos. Para se poder utilizar um outro meio físico que não esteja descrito na norma e se poder usufruir do software ETS para configuração, é necessário solicitar a certificação desse novo meio, em questão, junto da KONNEX, para que se possa introduzir no software.

2.4.1- Par de condutores (TP1): aproveitando a norma EIB equivalente.

O par de condutores (TP1) funciona com uma taxa de transferência de 9600 bits/s. Os dados são transmitidos simetricamente através de par entrançado e a transmissão de sinais é feita por meio da diferença de tensão.

Cada octeto da trama é enviado para o barramento TP1 utilizando o modo série, ou seja, é enviado um bit de início, de seguida os oito bits do octeto, um bit de paridade para controlo de erros e, no final, um bit de identificação do fim.

O sinal transmitido no barramento é em corrente alternada com uma componente em corrente contínua. A tensão da componente corrente contínua pode tomar os valores de 21Vdc a 32 Vdc. O sinal lógico '1' representa o estado inactivo da linha, o que significa que quando se transmite um sinal lógico '1' é o mesmo que não transmitir nada. Para enviar um sinal lógico '1'' é necessário enviar um '0' que normalmente é o bit de início. O sinal lógico '0' é feito quando a linha fica activa durante 35 µs, e nos restantes 105 µs (até o final de transmissão do bit) a linha fica inactiva.

O acesso ao barramento é do tipo CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*), que significa que, sempre que um ou mais dispositivos tentam comunicar ao mesmo tempo, apenas o dispositivo que tiver maior prioridade transmite, enquanto os restantes concorrentes esperam para que o barramento esteja livre. Isto é conseguido porque, sempre que mais do que um, tenta comunicar, estes observam o que se encontra no barramento, e cada um verifica se esse valor foi o transmitido por ele e, em caso afirmativo, significa que este tem maior prioridade sobre a linha e continua a transmitir.

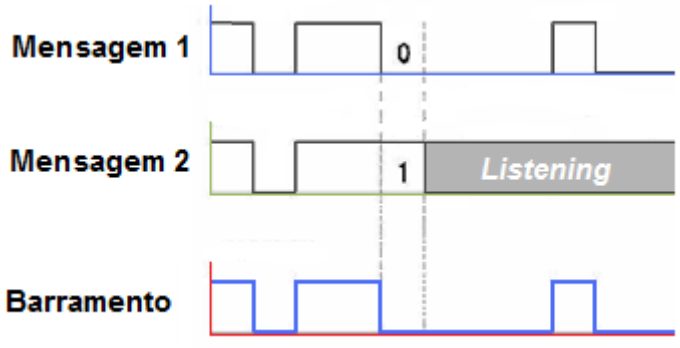

Figura 2.6 - Protocolo CSMA/CA

Se o dispositivo tem a mensagem de prioridade mais baixa então o dispositivo passa ao modo de escuta do barramento e espera que o mesmo seja desimpedido.

2.4.2- Infravermelhos: aproveitando a norma EIB: IR

Este meio de comunicação normalmente é utilizado nos comandos de infravermelhos que permitem controlar dispositivos EIB/KNX. A transmissão de dados por infravermelhos é assíncrona e pode ser unidireccional ou bidireccional mas, em *half-duplex,* como no caso dos

comandos de infravermelhos que são utilizados apenas para enviar comandos para dispositivos EIB/KNX. A frequência do sinal que é emitido pelo emissor de infravermelhos é de 447,5 kHz sendo a modulação em amplitude. A taxa de transmissão é de aproximadamente 7000 bits/s.

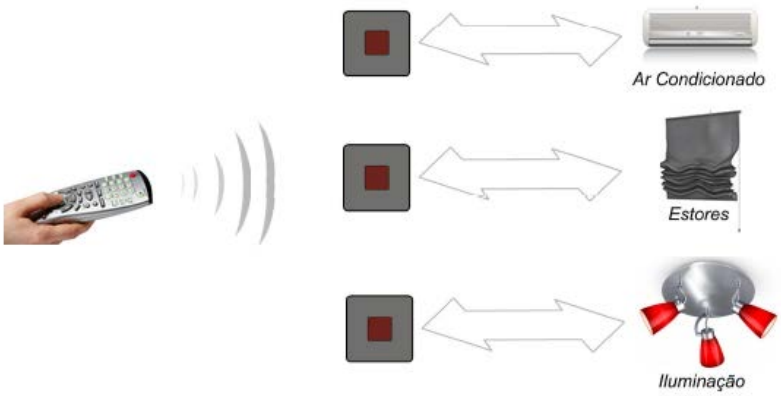

Figura 2.7 - Controlo por infravermelhos aproveitando a norma EIB: IR

2.4.3- Radiofrequência: aproveitando a norma EIB.RF

Estes dispositivos são conhecidos por dispositivos de curto alcance devido à sua potência máxima de emissão que é de 25mW, com uma taxa de transmissão de 16,384kBits/s e onde a banda de frequência é de 868MHz.

Utiliza uma modulação do tipo FSK (*Frequency-shift keying*) ou modulação por comutação de frequência, sendo estes sinais codificados por motivos de segurança e protecção dos dados numa codificação de Manchester.

Um dos problemas deste meio, como o anterior (infravermelhos), é que, como não existe delimitação da linha, ou seja, como é um meio físico aberto onde a única barreira do sinal é o alcance, se existir uma outra montagem com o mesmo meio e perto, é possível haver interacção entre dispositivos, onde isso não era desejado. Para evitar este problema é necessário alterar o domínio dos endereços EIB/KNX, para endereços mais longos, de forma a diminuir a probabilidade de efeitos indesejados acontecerem, por exemplo em duas habitações vizinhas.

2.4.4- Ethernet: aproveitando a norma EIB.net.

Para poder utilizar KNX sobre uma rede IP foi criado o protocolo KNXnet/IP. O KNXnet/IP é uma continuação do EIB.net que permite assim a utilização do EIB/KNX sob redes TCP/IP, estando esta nova actualização definida na norma EN 13321-2.

O KNXnet/IP situa-se na camada Física do modelo em camadas do KNX e define a integração do protocolo KNX sobre o protocolo de rede *Internet Protocol* (IP).

2.4.5- Correntes portadoras

Este meio físico tem muitas interferências e as taxas de transmissão são baixas, mas tem a vantagem de permitir a utilização da rede eléctrica já instalada. De facto, este meio físico é conhecido pela sua utilização na rede eléctrica.

Existem duas especificações no EIB/KNX para este meio físico: o PL110 e o PL132.

O PL110 tem uma taxa de transferência de 1200 bits/s e foi herdado do EIB mantendo a compatibilidade com os dispositivos EIB PL110. Podem, assim, comunicar dispositivos EIB PL110 com dispositivos KNX/EIB PL110.

O PL132 tem uma taxa de transferência de 2400 bits/s e foi herdado do EHS onde este ainda continua a ser utilizado. Contudo, ao contrário da norma PL110, neste caso não existe compatibilidade, ou seja, dispositivos EHS PL132 não conseguem comunicar com dispositivos EIB/KNX pois o protocolo de domótica é diferente.

É utilizada uma modulação do tipo *Spread frequency Shift Keying (SFSK*) para enviar dados pela rede eléctrica (230V@50Hz*)*, onde a frequência para o valor lógico '1' é 115,2 kHz e para o valor lógico '0' é de 105,6 kHz e a duração de um bit é de 833,33 µs, como se pode verificar na Figura 2.8.

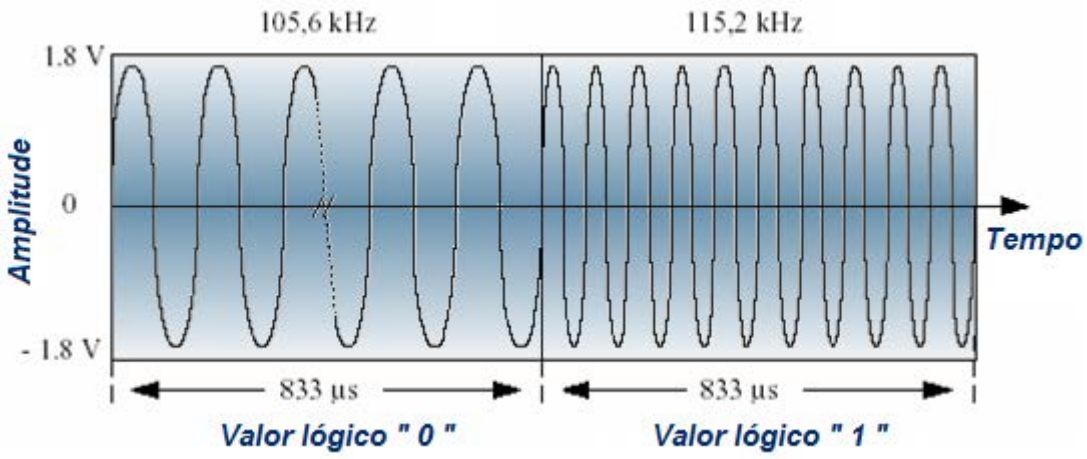

Figura 2.8 - Codificação do sinal utilizada na rede eléctrica

2.5- *Modelo das camadas* OSI do EIB/KNX

O sistema de comunicação do KNX/EIB tem uma estrutura constituída por uma pilha protocolar semelhante ao modelo de 7 camadas OSI, onde apenas são implementadas cinco camadas: Física, Ligação lógica, Rede, Transporte e, por último, a de Aplicação. A camada física, graças a um mecanismo que permite o EIB/KNX ser independente do meio físico faz com que esta camada seja independente das camadas superiores. As remanescentes camadas do modelo serão explicadas de seguida.

Figura 2.9 - Camadas OSI do sistema EIB/KNX

2.5.1- Camada de ligação lógica

Como já referido, o EIB/KNX tem um mecanismo que permite a este ser independente do meio físico utilizado. Isto é conseguido com esta camada de ligação lógica, pois esta é responsável pela ligação do nível físico com a camada de rede e permite que todas as camadas superiores sejam independentes do meio físico.

Neste nível, a trama tem a designação de LPDU e é aqui que são efectuadas as verificações do cabeçalho da trama, sendo verificado o formato da trama e o endereço, decidindo assim se esta trama lhe é destinada. Verifica também a consistência da mesma, ou seja, se esta contém erros ou mesmo se ocorreram erros de transmissão através do último octeto da trama que contém o *check octet* efectuado com um NOT XOR aos restantes octetos que constituem a trama.

2.5.2- *Camada* de rede

A trama resultante do nível anterior é enviada para este nível e é denominada por NPDU e corresponde a uma trama que tem encapsulado os dados que serão enviados às camadas superiores. O cabeçalho da trama NPDU tem a função de indicar que tipo de endereço de destino foi utilizado, isto é, se o endereço de destino representa um endereço de grupo, individual ou se é do tipo *broadcast*. Para isso são utilizados 3 bits situados no octeto 5. Nesta camada é efectuada a selecção do serviço de endereço a utilizar, mediante o endereço de destino presente. Se este é do tipo individual, a trama enviada para a camada superior utilizará os serviços do *N_Data*, se for de grupo utilizará os serviços *N_GroupData*.
Se o endereço EIB/KNX utilizado for igual a zero e se este for do tipo *broadcast*, a trama enviada para a camada de transporte utilizará os serviços do tipo *N_Broadcast*.

2.5.3- *Camada* de Transporte

A camada de transporte situada entre a camada de rede e a camada de aplicação realiza a interface entre estas duas camadas implementando os vários modos de comunicação:

- *Multicast*;
- *Broadcast;*
- Um-para-um sem conexão;
- Um-para-todos orientada à conexão.

A trama desta camada, é designada por TPDU e corresponde à trama NPDU mas reduzida, pois não tem o cabeçalho de controlo de rede. Esta trama contém os dados da camada de aplicação e o cabeçalho da camada de transporte, sendo este composto por um campo de controlo de 5 bits situados no octeto 5 da trama EIB/KNX. Este campo de controlo indica o código do serviço da camada de transporte.

2.5.4- Camada de aplicação

Esta camada corresponde à última camada do modelo OSI. Nesta camada os serviços disponíveis são referentes aos quatro modos de comunicação da camada de transporte: o modo *multicast*, *broadcast*, um-para-um sem conexão e um-para-todos orientada à conexão.
A Camada de aplicação utiliza dois bits do octeto 6 e o octeto 7 para identificar o serviço, O código do serviço que é transportado no cabeçalho da trama da camada de aplicação, pode ter até 10 bits e em alguns dos casos só são necessários 4 bits, como nos serviços do *multicast*. Assim os dados serão preenchidos a partir do bit 6 do octeto 7 e os restantes bits tomam o valor 0, como se pode verificar na Figura 2.10.

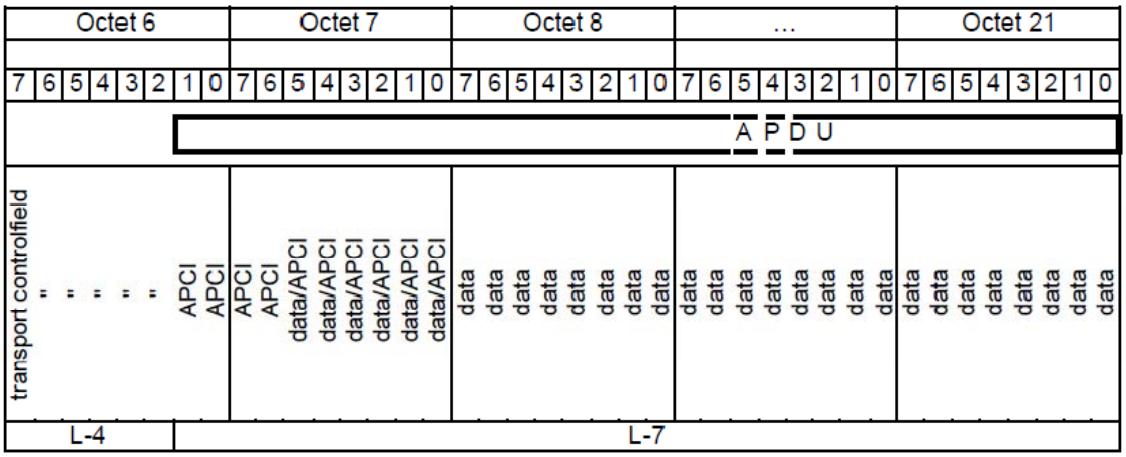

Figura 2.10 – Formato da trama APDU [20]

Os restantes octetos são para dados que no caso de a trama ser do tipo normal tem um limite de 14 octetos de dados pois o tamanho da trama é de 23 octetos apesar de na imagem se verificar que apenas contém 21 isto deve-se ao facto de a numeração na imagem começar no octeto zero e não estar representado o ultimo octeto que representa o *Frame Check*. A trama de aplicação é denominada por APDU e vai encapsulada na trama de transporte.

2.5.4.1- Serviços

Dentro dos serviços disponíveis na camada de aplicação do EIB/KNX, os apresentados a seguir são alguns desses serviços com os seus respectivos códigos:

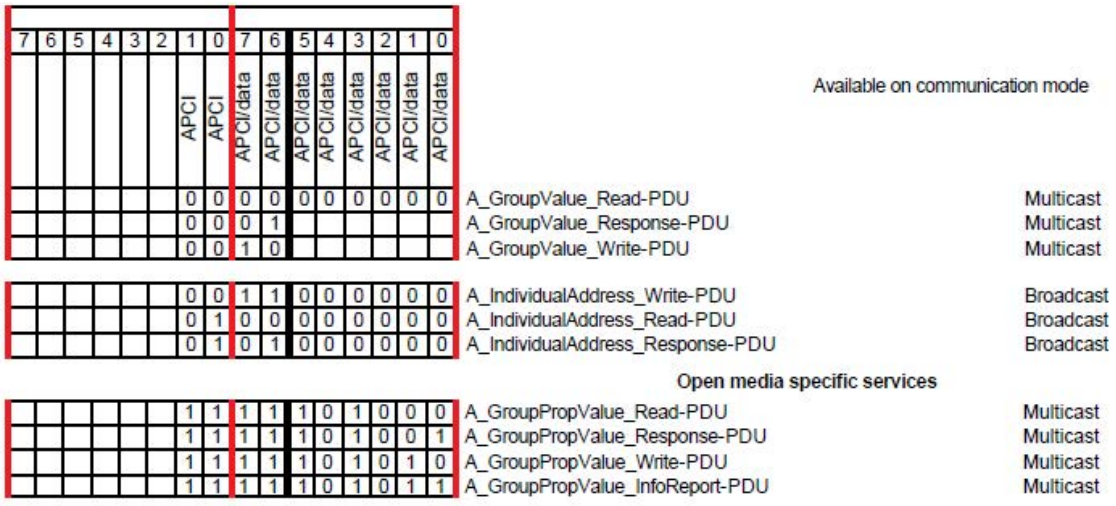

Figura 2.11 - Exemplo de serviços disponíveis no EIB/KNX [20]

Uma vez que este trabalho utiliza os serviços já implementados no projecto Domoitech, apenas iremos descrever esses serviços da camada de aplicação disponíveis.

Serviço *GroupValue_Read*

Este serviço é utilizado quando se pretende enviar tramas com pedidos de leitura de estados de variáveis, Entradas ou Saídas. O código deste serviço tem o valor decimal 0 como se pode verificar na Figura 2.12.

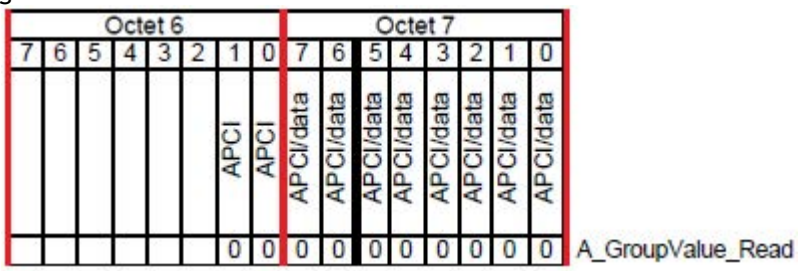

Figura 2.12 - código do serviço *GroupValue_Read* [20]

O grupo receptor desta trama de pedido de estados deverá, enviar uma resposta com os estados pedidos.

Serviço *GroupValue_Res*

Quando um Grupo recebe uma trama correspondente ao serviço de pedidos de leitura de estados de variáveis ou E/S, este responde com uma trama onde o serviço utilizado será o *GroupValue_Res*. Este serviço tem o valor decimal 1 quando retirados os bits 1,0 do octeto 6 e os bits 7 e 6 do octeto 7.

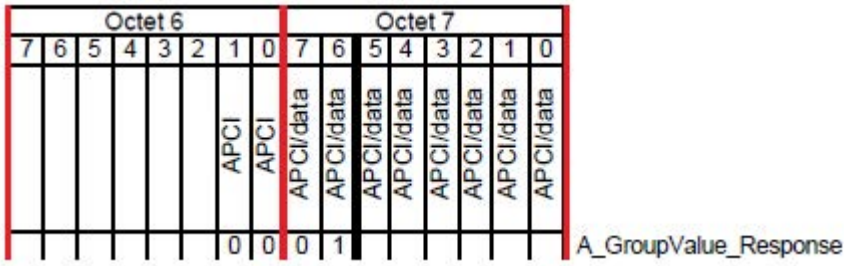

Figura 2.13 - Código do serviço GroupValue_Res [20]

Os estados das variáveis vão serão colocados imediatamente a seguir ao bit 6 do octeto 7 em caso de estes estados se referirem a variáveis booleanas isto é, se apenas é necessário um bit para representar esse estado, em caso afirmativo estes são colocados a seguir ao bit 6 do octeto 7, caso os estados não sejam deste tipo e for necessário mais espaço para o guardar então estes segue nos octetos seguintes.

Serviço GroupValue_Write

O código deste serviço tem o valor decimal 2, ou seja como já perceptível o bit 1 e 0 do octeto 6 têm o valor zero como o bit 6 do octeto 7 ficando assim o bit 7 do octeto 7 com o valor 1.

Este é o serviço que permite enviar tramas de escrita para um endereço de grupo fazendo alterar o valor da varivel de estado para o valor enviado na trama. O valor do estado

vai no octeto 7 para o caso de ser binário ou vai no octeto seguinte para o caso de ser um valor decimal de 8 bits.

2.5.4.2- *Datapoint Types*

Os dispositivos EIB/KNX têm de conseguir comunicar entre si, mesmo que sejam de diferentes fabricantes. Desta forma, para além da utilização do mesmo protocolo, necessitam de conseguir interpretar correctamente as mensagens enviadas por outros dispositivos. Isto é, se um interruptor envia por exemplo o comando desligar para uma lâmpada, essa tem de conhecer esse comando para atender a esse pedido. Para isso, existe definido um conjunto de *Datapoints types* que, anteriormente no protocolo EIB, se denominavam por EIS (EIB *Interworking Standards)*, (esta nova designação *datapoint types* é mais adequada porque os EIS são tipos de dados). Estes definem regras para a interoperabilidade de funções utilizadas pelos dispositivos.

Os *Datapoint types* são identificados por dois números de 16 bits separados por um ponto, por exemplo "1,007", que é do tipo Booleano e representa o objecto *Datapoint type Step* (DPT_Step). Os vários tipos de *Datapoint types* têm vários tipos de formatos de dados, ou seja, utilizando o exemplo anterior, o *Datapoint type step* é do tipo booleano pois utiliza apenas 1 bit, podendo assim ter dois estados possíveis, sendo eles, o mover gradualmente para cima ou para baixo, como exemplifica a Figura 2.14 onde estão apresentados os *Datapoint types* do tipo booleanos.

Datapoint Types				
ID:	Name:	Encoding: V = 0	V = 1	Usage:
1.001	DPT_Switch	Off	On	General
1.002	DPT_Bool	False	True	General
1.003	DPT_Enable	Disable	Enable	General
1.004	DPT_Ramp	No ramp	Ramp	FB only
1.005	DPT_Alarm	No alarm	Alarm	FB only
1.006	DPT_BinaryValue	Low	High	FB only
1.007	DPT_Step	Decrease	Increase	FB only
1.008	DPT_UpDown	Up	Down	General
1.009	DPT_OpenClose	Open	Close	General
1.010	DPT_Start	Stop	Start	General
1.011	DPT_State	Inactive	Active	FB only
1.012	DPT_Invert	Not inverted	Inverted	FB only
1.013	DPT_DimSendStyle	Start/stop	Cyclically	FB only
1.014	DPT_InputSource	Fixed	Calculated	FB only

Figura 2.14 - Data type "Boolean" [20]

Os tipos de Datapoints types que existem são os seguintes:

- Data Type “Boolean”
- Data Type “1-Bit Controlled”
- Data Type “3-Bit Controlled”
- Data Type “Character Set”
- Data Type “8-Bit Unsigned Value”

- Data Type “8-Bit Signed Value”
- Data Type “Status with Mode”
- Data Type “2-Octet Unsigned Value”
- Data Type “2-Octet Signed Value”
- Data Type “2-Octet Float Value”
- Data Type “Time”
- Data Type “Date”
- Data Type “4-Octet Unsigned Value”
- Data Type “4-Octet Signed Value”
- Data Type “4-Octet Float Value”
- Data Type “Access”
- Data Type “String”

O Domoitech reconhece apenas dois tipos de EIS: *SWITCHING e* DRIVE *CONTROL*. No KNX/EIB estes dois tipos de EIS resumem-se a apenas um *Datapoint Type*, o *boolean*. Como o Domoitech foi realizado com os EIS, assim serão explicados os dois EIS que ele implementa.

EIS 1 – *Switching*

Com este objecto é possível actuar numa carga do tipo ON/OFF ligada ao actuador, o que normalmente é utilizado para lâmpadas. No entanto as saídas das lâmpadas podem ser utilizadas para qualquer carga do mesmo tipo, em que a função requerida seja a mesma, ou seja onde é necessário ter apenas dois estados de entre os seguintes: (*on* / *off*), (*enable* / *disable*), (*alarm* / *no alarm*) e (*true* / *false*). O EIS 1, tem o valor de 1 bit e o código dele é 10 [21].

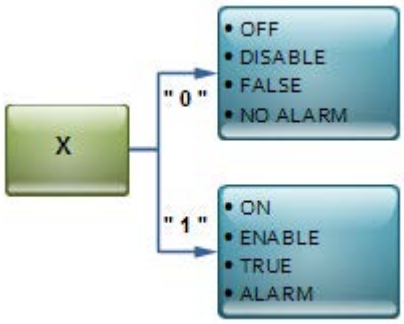

Figura 2.15 - *EIS Switching*

EIS 7 – Drive Control

Drive Control contém duas subfunções denominadas por “*move*” e “*step*”, em que a subfunção “*move*” é possível colocar o *drive* em movimento para um sentido ou para o outro, sendo utilizada para fechar ou abrir totalmente o estore ou seja com dois estados possíveis: mover para baixo ou mover para cima, em que o tempo de movimento é configurável pelo utilizador.

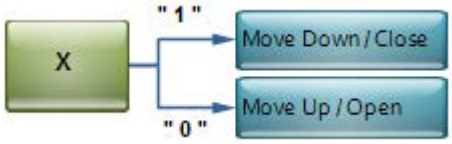

Figura 2.16 - *EIS Drive Control - Move*

Com a subfunção "*step*" é possível realizar movimentos graduais em ambos os sentidos. Este tempo de movimento gradual é também facilmente configurável, sendo também normalmente utilizada para fechar ou abrir gradualmente o estore e pode ter dois estados possíveis: mover para baixo ou mover para cima. A imagem seguinte ilustra os dois estados possíveis:

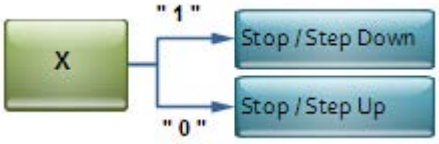

Figura 2.17 - *EIS Drive Control - Step*

A Figura 2.18 representa os quatro estados possíveis deste EIS: em movimento, parado, gradualmente para cima e gradualmente para baixo.

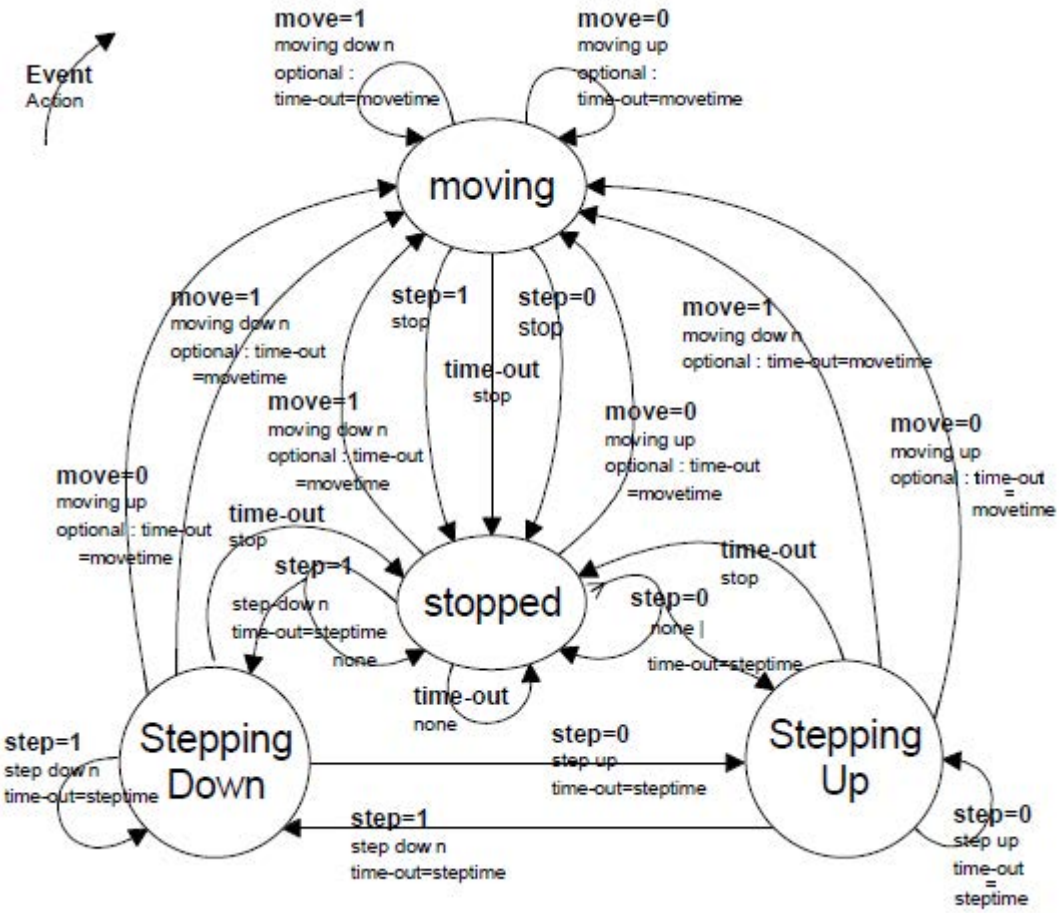

Figura 2.18 - Máquina de estados do *Drive Control* [21]

2.6- Software ETS

O ETS (*Engineering Tool Software)*, é uma ferramenta para engenharia de sistemas EIB/KNX, que tem várias utilizações, sendo que a mais utilizada é a de desenho e configuração de uma instalação ou projecto EIB/KNX. Com esta ferramenta é também possível fazer a gestão da rede EIB/KNX.

Este software contém uma base de dados de todos os produtos, que deve ser fornecida por cada fabricante. Nessa base de dados, encontram-se todas as informações sobre cada dispositivo EIB/KNX, o que permite saber quais as suas funções ou aplicações. É possível desenhar e configurar toda a rede no modo *offline* e posteriormente carregar toda a configuração para os dispositivos EIB/KNX. O ETS comporta-se como um Mestre de Configuração sempre que este se encontra ligado ao barramento.

Capítulo 3

Protocolo CAN

O protocolo CAN foi desenvolvido pela BOSCH na década de 1980 e tornou-se um padrão internacional (ISO 11898) em 1994. Foi especialmente desenvolvido para rápida troca de dados entre controladores electrónicos em veículos motorizados. Apesar de, originalmente ter sido desenvolvido para uso em automóveis, é cada vez mais utilizado em sistemas industriais, como por exemplo: um barramento interno de máquinas; interconexão de sistemas de medição distribuídos; funções de controlo e monitorização ou, apenas como um barramento para interligar sensores, actuadores e interfaces homem-máquina. Em ambos os casos, as principais características que levam à sua utilização são: baixo custo, a capacidade de funcionar em ambientes electricamente hostis, uma elevada capacidade para uso em sistemas de tempo real e a sua facilidade de utilização.

CAN é um protocolo de comunicação do tipo série síncrono de grande eficiência e segurança com capacidades *multi-master*, ou seja, é permitido utilizar mais de um *master* a controlar ou a obter informações dos dispositivos *slaves* interligados ao barramento CAN.

3.1- Modelo das camadas OSI do protocolo CAN

No modelo de camadas ISO/OSI, a especificação da rede CAN descreve apenas a camada física e a camada de ligação de dados. A camada física é responsável pelos aspectos da transmissão física dos dados (bits), como por exemplo a Codificação/descodificação de bits ou a sincronização entre nós. Os aspectos importantes desta camada são abordados nos dois subcapítulos seguintes (3.2 Lógica do barramento e 3.3 Requisitos temporais). A camada de ligação de dados define o formato das mensagens, a prioridade do acesso ao barramento e disponibiliza mecanismos de detecção e prevenção de falhas. Esta camada será abordada nos restantes subcapítulos deste capítulo. Apesar de se apresentar uma camada de aplicação no modelo OSI, não existe uma norma para definir a mesma. Contudo para facilitar a programação, e devido aos diferentes cenários de utilização deste protocolo, foram

desenvolvidas diversas camadas de *software* que se posicionam na camada de aplicação, definindo um perfil de comunicações e *frameworks* para a programação dos dispositivos, mas define também recomendações para as camadas física (cabos e fichas) e ligação de dados [18].

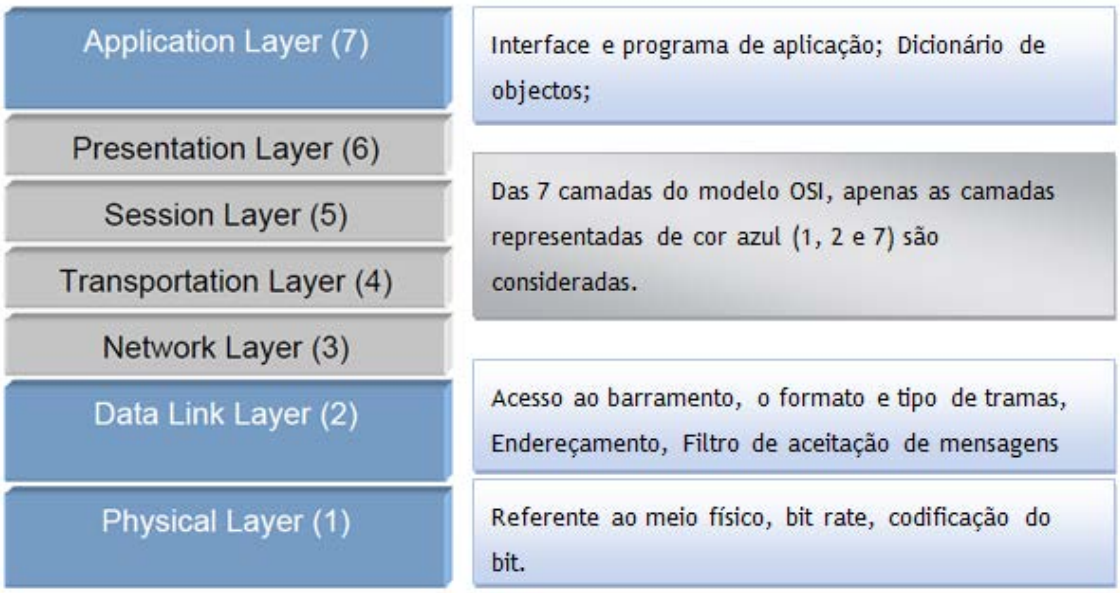

Figura 3.1 - Modelo de camadas ISO/OSI do CAN

3.2- Lógica do Barramento

A representação, tanto do estado lógico "0", como do estado lógico "1", no protocolo CAN tem uma denotação própria, onde estes são reconhecidos como bits dominantes ou recessivos, respectivamente. Como se pode verificar na Figura 3.2 a seguir apresentada, o bit dominante apresenta uma diferença de potencial, ao contrário do bit recessivo, em que esta é nula.

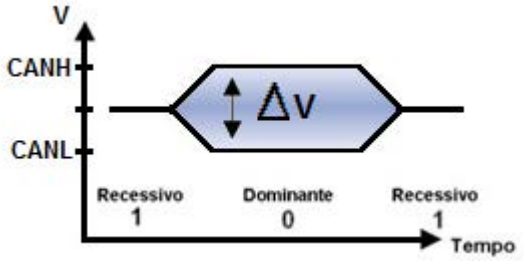

Figura 3.2 - Bit recessivo e Dominante

Os dados enviados através da rede são interpretados pela análise da diferença de potencial entre os condutores CAN_H e CAN_L, daí advir a classificação do barramento CAN

como Par Trançado Diferencial. Este conceito, atenua fortemente os efeitos causados por interferências electromagnéticas, uma vez que qualquer acção sobre um dos fios será sentida também pelo outro, causando flutuação em ambos os sinais para o mesmo sentido e com a mesma intensidade. Como a interpretação do sinal advém da medição da diferença de potencial entre os condutores CAN_H e CAN_L, que permanecerá inalterada, a comunicação não é prejudicada.

3.3- *Bit Timing*

Cada módulo CAN opera em sincronia com o oscilador que a cada conjunto de *n* períodos da frequência do oscilador dá-se o nome de *time-quantum* (Tq), sendo o tempo de transmissão de cada bit (*bit-time*) definido como múltiplo de Tq.

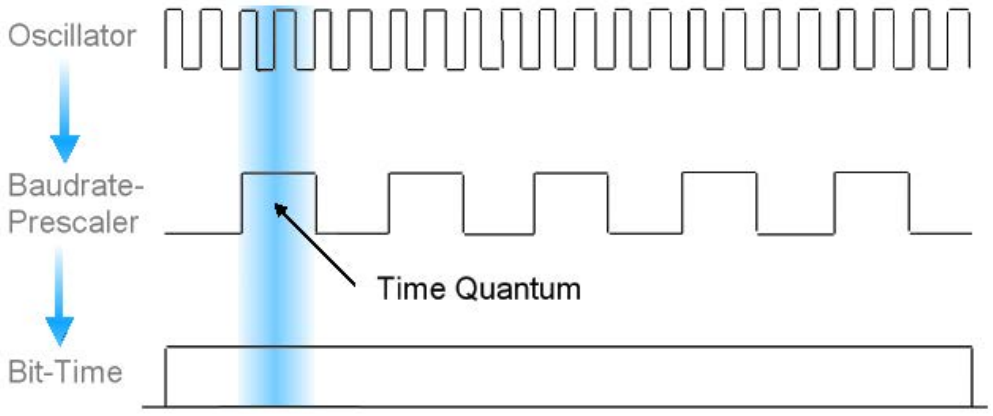

Figura 3.3 - *Time quantum* no CAN

A razão de ser definido como múltiplo de Tq deriva da subdivisão do bit time em quatro segmentos distintos, em que cada um deles tem a duração múltipla de Tq, o que faz com que a soma desses quatro segmentos origine um múltiplo de Tq, o Bit Time. Existe um segmento de sincronização com duração de 1 Tq, um segmento de propagação com 1 a 8 Tq e dois segmentos de fase com 1 a 8 Tq cada.

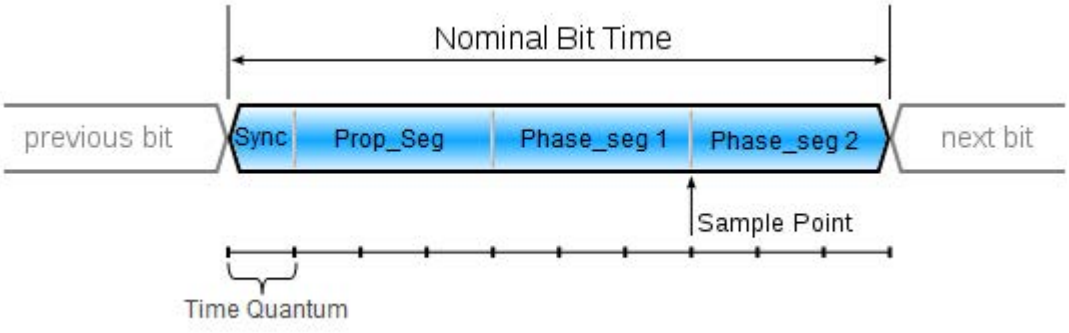

Figura 3.4 - Parcelas que definem o tempo de um bit em CAN

Quanto maior for o *bus*, mais significativos são os atrasos de propagação e maior tem que ser o *bit-time*, reduzindo a taxa de transmissão máxima. É ainda possível a utilização de *bridges* e repetidores para estender o comprimento máximo do barramento imposto.

A velocidade de transmissão dos dados é inversamente proporcional ao comprimento do barramento. A maior taxa de transmissão especificada é de 1Mbps, considerando-se um barramento de 40 metros. A Tabela 3.1 representa a relação entre o comprimento da rede (barramento) e a taxa de transmissão dos dados.

Tabela 3.1- Relação comprimento do cabo vs taxa de transferência

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

3.4- Modo standard e modo estendido

O protocolo de comunicação CAN é do tipo CSMA/CD+AMP (*Carrier Sense Multiple Access/Collision Detection with Non-Destructive Arbitration*) o que significa que todos os módulos verificam o estado do barramento, ou seja, se outro módulo está ou não a enviar mensagens com maior prioridade em caso de acontecer uma tentativa de múltiplo acesso ao barramento. O módulo cuja mensagem tiver menor prioridade pára a sua transmissão e entra no modo leitura, e o de maior prioridade continuará a enviar a sua mensagem deste ponto, sem ter que reiniciar a transmissão que se encontrava a enviar. A primeira versão do modo *standard CAN*, ISO11519 (low-speed CAN) é para aplicações até 125kbps, com o identificador de 11-bits. A segunda versão, ISO 11898 (1993), manteve os mesmos 11 bits identificadores e permite comunicações de 125kbps até 1Mbps. A mais recente actualização (1995) introduziu o modo estendido de 29 bits. O ISO11898 com a versão de 11-bits é frequentemente citado como *Standard CAN Version 2.0A*, enquanto a outra actualização de 29-bits de endereço é citada como *Extend CAN Version 2.0B*.

3.5- Formato e tipo de tramas

Existem 4 tipos de mensagens: a *Data Frame*, a *Remote Frame*, a *Overload Frame e* a *Error Frame*.

A *Data Frame* é uma trama cujo propósito é transportar informações correntes do sistema, ou seja, sempre que for necessário enviar dados. A trama *Remote Frame* é utilizada quando necessário solicitar informações do sistema. A *Overload Frame* consiste numa mensagem que serve para indicar que o dispositivo se encontra ocupado. Sempre que um dispositivo se encontra ocupado para atender mensagens provenientes de outros dispositivos, este responde com uma mensagem *Overload Frame*. A *Error Frame* é uma mensagem característica que serve para indicar que não foi possível enviar uma mensagem de controlo por ser recessiva.

Figura 3.5 - Formato de uma trama CAN 2.0A

O bit SFO indica o inicio da trama, seguido de um campo *indentifier* de 11bits que representa, como já referido anteriormente, a prioridade da mensagem a enviar. Como poderemos ver mais à frente, quanto menor for o valor lógico deste campo maior será a prioridade. O Campo RTR é usado quando um receptor pretende pedir uma informação a um transmissor. Como já foi referido, existem dois tipos de formatos de tramas (2.0A e 2.0B) onde essa identificação é feita através do campo IDE de um bit que, em caso de ser um bit dominante, indica que estamos na presença de uma trama do tipo CAN 2.0A, ou seja, que não serão utilizados bits de endereço extra (CAN 2.0B). Os restantes campos são de interpretação bastante fácil, onde R0 é apenas um bit reservado seguindo o campo DLC (4bits) e DATA (64bits), onde estes representam o tamanho da informação a transmitir e a informação em si, respectivamente. O campo CRC (teste de redundância cíclica) serve para detecção de erros de envio que é comum em vários tipos de protocolos. ACK representa o campo de validação da mensagem seguindo de EOF de 7bits indicadores de final da trama e de verificação do *"bit stuffing"*.

O formato da trama CAN 2.0B é de certa forma idêntica à 2.0A mas, contendo mais 18 bits para o campo de identificação. No local onde se encontrava o RTR agora encontra-se o bit SRR que apenas preenche o espaço com um bit recessivo. O Bit R1 tal como o R0 são bits reservados.

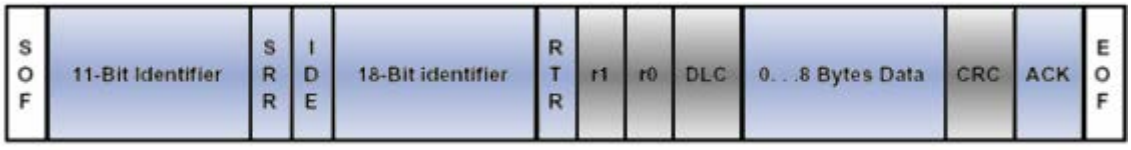

Figura 3.6 - Formato de uma Trama CAN 2.0B

3.6- Método de selecção da prioridade

Um transmissor envia uma mensagem para o barramento onde cada dispositivo decide, com base no endereço recebido, se deve processar a mensagem ou não, como ilustra a Figura 3.7.

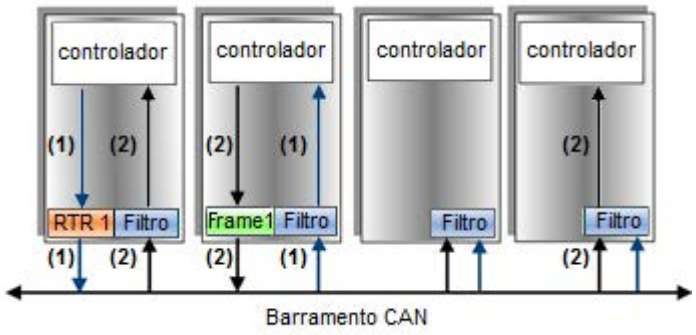

Figura 3.7 - Exemplo de envio de uma trama, baseada [11]

Como se pode verificar na Figura 3.7, o primeiro dispositivo envia uma trama com o pedido de envio de informação (RTR-*remote transmission request*), ou seja, requerendo um certo tipo de informação que será entregue por quem a contiver, mensagem esta representada por (1), que de seguida é aceite apenas pelo dispositivo dois, por ser o nó com a informação pedida, onde este responde com uma trama de dados, contendo a informação pedida, representada por (2) e esta é aceite pelos dispositivos um e quatro.

Como referido anteriormente, o campo *identifier* da trama determina a prioridade que a mensagem tem, na concorrência ao acesso ao barramento CAN, dado que mais do que um dispositivo pode tentar comunicar ao mesmo tempo e, para isso, o CAN utiliza um método para seleccionar quem terá prioridade na mensagem a enviar. Essa prioridade é feita da forma indicada na Figura 3.8.

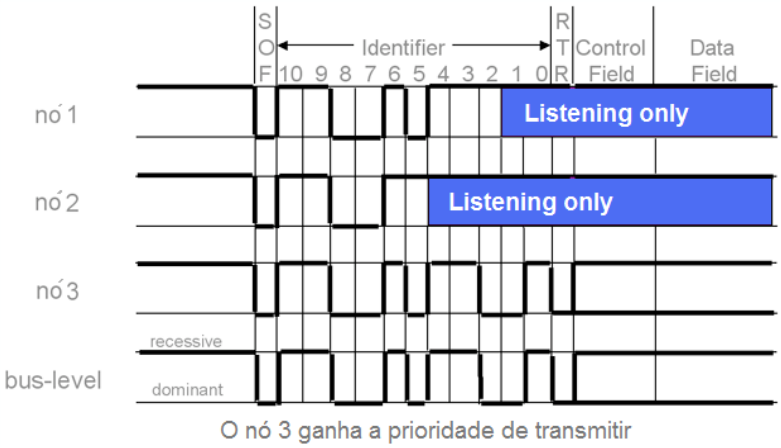

Figura 3.8 - Método de selecção de prioridade, baseada [11]

Assim, é evitada a colisão de mensagens quando mais do que um nó inicia a transmissão ao mesmo tempo, onde cada nó envia os bits do seu endereço (campo *identifier*)

de mensagem e monitoriza o nível do barramento para verificar se o que se encontra no barramento é o que foi enviado por este. Enquanto os bits de todos os transmissores forem iguais não acontece nada, caso contrário, tem prioridade o bit dominante.

3.7- Mecanismos de detecção

Por motivos de fiabilidade e para garantir que dados corrompidos de origem do barramento não sejam válidos, o protocolo CAN apresenta os seguintes mecanismos que possibilitam essa detecção.

1. **Monitorização ao nível do bit**

Cada dispositivo presente no barramento monitoriza o bit que se encontra no barramento e verifica se o valor desse bit é realmente igual ao que este transmitiu. O erro ocorre quando isto não acontece ou seja quando o valor do bit no barramento é diferente do valor enviado. Existe uma excepção que se verifica quando os dispositivos se encontram na fase de selecção de prioridade, que como descrito neste capítulo, quando o dispositivo verifica que o valor do barramento é diferente do enviado, automaticamente pára de transmitir, sem ocorrer erro.

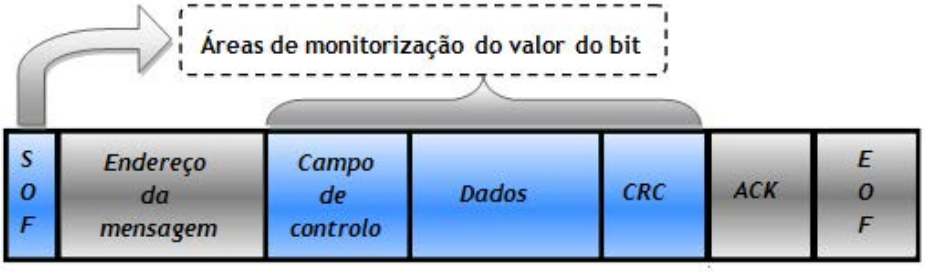

Figura 3.9 - Áreas de da trama CAN sensíveis a monitorização do valor do bit

2. **Verificação da trama**

Numa trama CAN existem bits fixos cujo seu valor é conhecido como o bit delimitador do campo CRC ou o bit delimitador do campo ACK. Se um receptor encontrar um ou mais bits que não correspondem ao que deveria estar presente, um erro de forma é activado, indicando assim uma trama mal recebida ou esta não foi bem enviada.

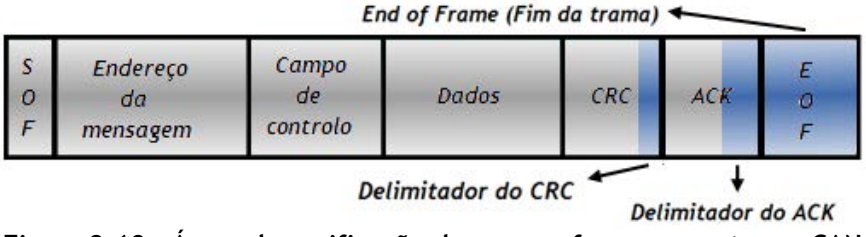

Figura 3.10 - Áreas de verificação de erro na forma numa trama CAN

3. **Bit Stuffing**

O erro de *bit stuff* é detectado sempre que aparecem 6 bits consecutivos iguais, durante a área que corresponde a parte da mensagem onde foi efectuado o processo de *bit stuffing*. Em caso de detecção deste erro é gerada uma mensagem de erro (*ERROR FRAME*).

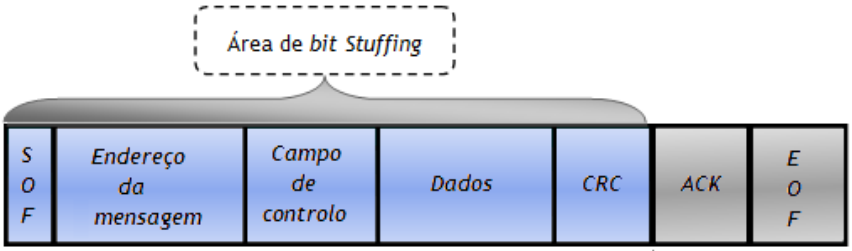

Figura 3.11 - Área associada ao processo de *bit Stuffing*

4. **Verificação cíclica de redundância *CRC***

O transmissor calcula um valor originado por um polinómio gerador com os valores desdo do inicio da trama até ao fim do campo de dados e esse valor é guardado no campo CRC. O mesmo cálculo é efectuado assim que a mensagem chega ao receptor que irá comparar esse valor resultante com o valor contido na trama no campo CRC, se estes não coincidirem significa que a mensagem apresenta erros de transmissão. Caso acontece este erro a mensagem é descartada pelo receptor e emite uma trama de erro para pedir a retransmissão da trama errónea.

5. **Erro de *Acknowledge***

O erro de *Acknowledge* acontece sempre que o dispositivo que envia uma trama para o barramento, não recebe um bit dominante no campo ACK da trama, o que significa que nenhum dispositivo recebeu a trama correctamente. No caso de este detectar um bit dominante, apenas se tem conhecimento de que pelo menos um dispositivo receptor recebeu a trama correctamente.

3.8- Falhas temporárias e falhas permanentes

Para distinguir entre falhas temporárias que possam acontecer no barramento como interferências, e falhas permanentes como um módulo que apresenta defeitos ou mesmo uma ligação defeituosa ao módulo, são utilizados os contadores de erro (*Transmit Error Counter* (TEC) e *Receive Error Counter* (REC)) que controlam o estado em que o módulo se encontra.

Existem três estados possíveis, e todos os módulos CAN iniciam a actividade no estado activo conforme ilustrado na Figura 3.12 apresentada a seguir.

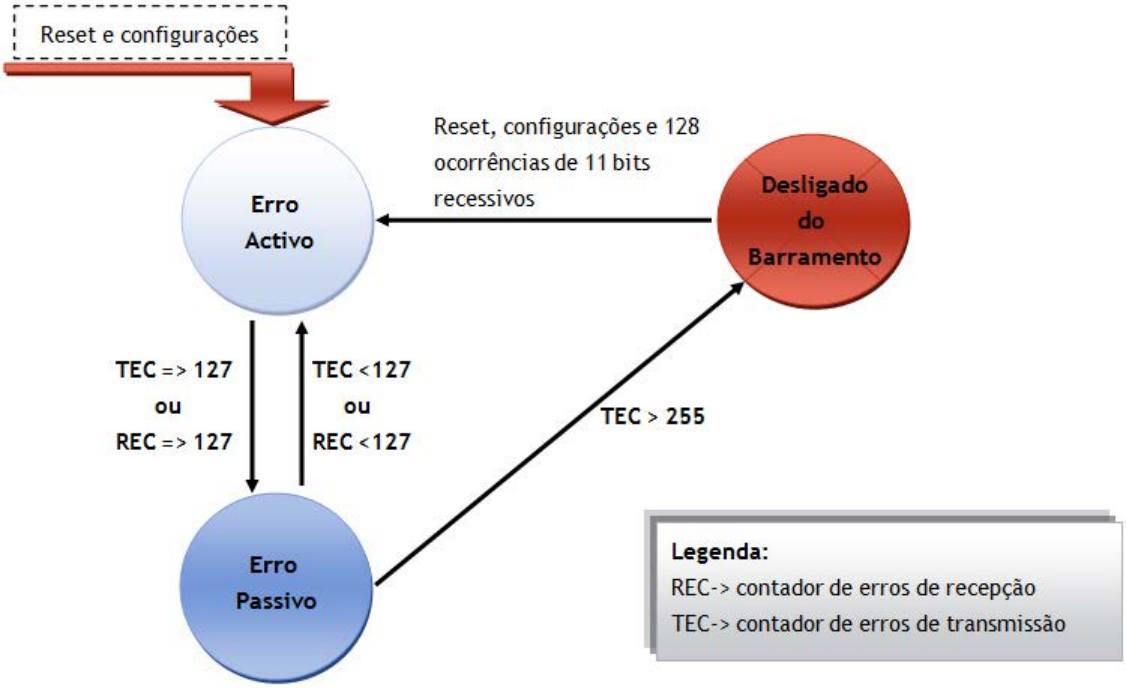

Figura 3.12 - Diagrama de estados de um nó CAN

Estes contadores de erros são do tipo *up/down counter,* ou seja estes podem ser incrementados em caso de erros ou decrementados em casso de sucesso, o contador *TEC* que é responsável por detectar erros de transmissão e o REC para detectar erros de recepção.

Por definição, cada contagem equivale a um ponto. Ao detectar um erro é enviado uma *flag* de erro activo e o contador aumenta a pontuação para este nó. Se a próxima mensagem enviada por este nó não apresentar a *flag* de erro os pontos dados anteriormente são descontados.

O estado erro activo é válido enquanto ambos os contadores têm um valor menor ou igual a 127. Se o contador de erros de recepção ou o contador de erros de transmissão apresentarem uma contagem maior do que 127 pontos o nó entra no estado erro passivo, onde é possível ainda enviar mensagens, mas a coerência dos dados deixa de ser garantida. Quando ocorre uma falha na ligação ao *barramento,* um defeito no módulo CAN ou até no caso de acumulação extrema de erros, se o contador de erros associados à transmissão tiver um valor superior a 255, levará este módulo a entrar no estado de *bus off* (desligado do barramento) em que apenas conseguem receber mensagens. Para sair deste estado, é preciso impor um *reset* e receber uma sequência de 128 × 11 bits recessivos;

Capítulo 4

Projecto de Hardware

Tendo em conta os requisitos do sistema, foram efectuadas pesquisas de hardware que poderíamos ou não utilizar. Numa fase inicial, foi decidido utilizar um microcontrolador da Microchip [6], devido à vasta gama de produtos que apresenta. Contudo, não foi posta de parte, muito pelo contrário, a possibilidade da utilização de microcontroladores de outras marcas, principalmente a marca NXP [14] por apresentar dispositivos bastante atractivos, quer pelo seu preço, quer pelas suas características, que iremos documentar e apresentar de seguida no ponto 4.2-Pesquisas.

Previamente, é necessário um estudo de como será a estrutura do sistema.

4.1- Estrutura

Como já referido no decorrer deste projecto, existem dois módulos a desenvolver: o DOMO_CAN e o DOMO_GATMONITOR. Na Figura 4.1 está representada a estrutura do DOMO_GATMONITOR. O DOMO_GATMONITOR é constituído por uma interface CAN para poder comunicar com a rede CAN dos DOMO_CAN e uma interface RS232 para poder comunicar com um computador para poder configurar os DOMO_CANs e poder observar as tramas enviadas por estes e até mesmo enviar comandos, como por exemplo efectuar um reset à rede domótica. Para além das interfaces mencionadas o DOMO_GATMONITOR contém ainda uma expansão que permite obter a temperatura ambiente, luminosidade, hora, data e mesmo uma possibilidade de integrar uma memória EEPROM extra.

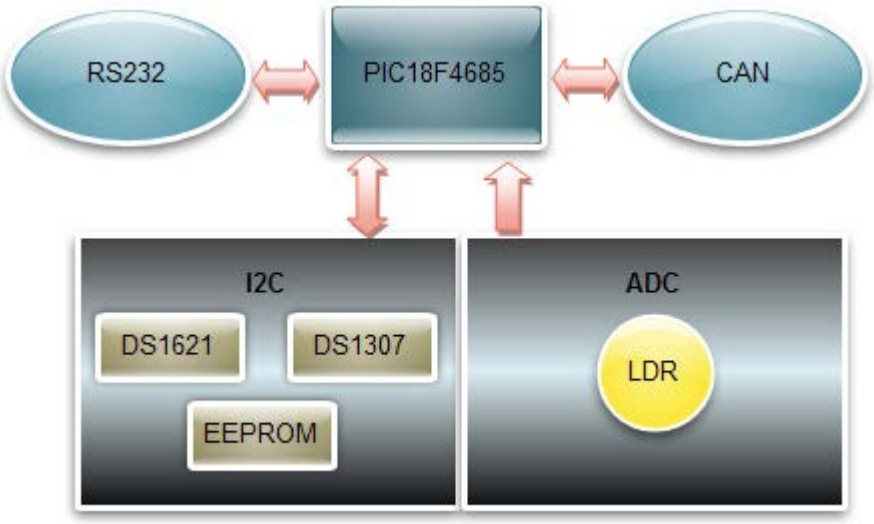

Figura 4.1 - Estrutura do DOMO_GATMONITOR

A LDR (*Light Dependent Resistor*) encontra-se ligada a uma entrada do ADC do PIC18F através de um divisor de tensão, sendo possível assim ler a queda de tensão na LDR que varia com a luz. No barramento I2C tem interligado dois componentes, um RTC (DS1307 - *Real Time Clock*) e um sensor de temperatura (DS1621).

A arquitectura pretendida para o DOMO_CAN incorre na existência de uma placa principal, a qual contém todo o processamento necessário e os vários módulos de interface com os sistemas exteriores, bem como na existência de dois tipos de placas secundárias, saídas em relés e saídas em triac's, que nos permitirá fazer a interface com os equipamentos que pretendemos controlar. Esta arquitectura é semelhante à arquitectura utilizada no Domoitech uma vez que o módulo DOMO_CAN será o módulo que utilizará parte do trabalho de software desenvolvido no Domoitech. Apesar da arquitectura ser semelhante, várias alterações foram introduzidas não só pelo facto de o meio físico ser diferente, pois a introdução de um selector de endereço teve como principal objectivo remover a necessidade de ser necessário desenvolver vários projectos de sofware devido a cada módulo ter um endereço KNX/EIB diferente. Assim com um selector de endereço é possível escolher o endereço do módulo tanto para o CAN como para o KNX/EIB. A Figura 4.2 representa um diagrama de blocos desta arquitectura a desenvolver.

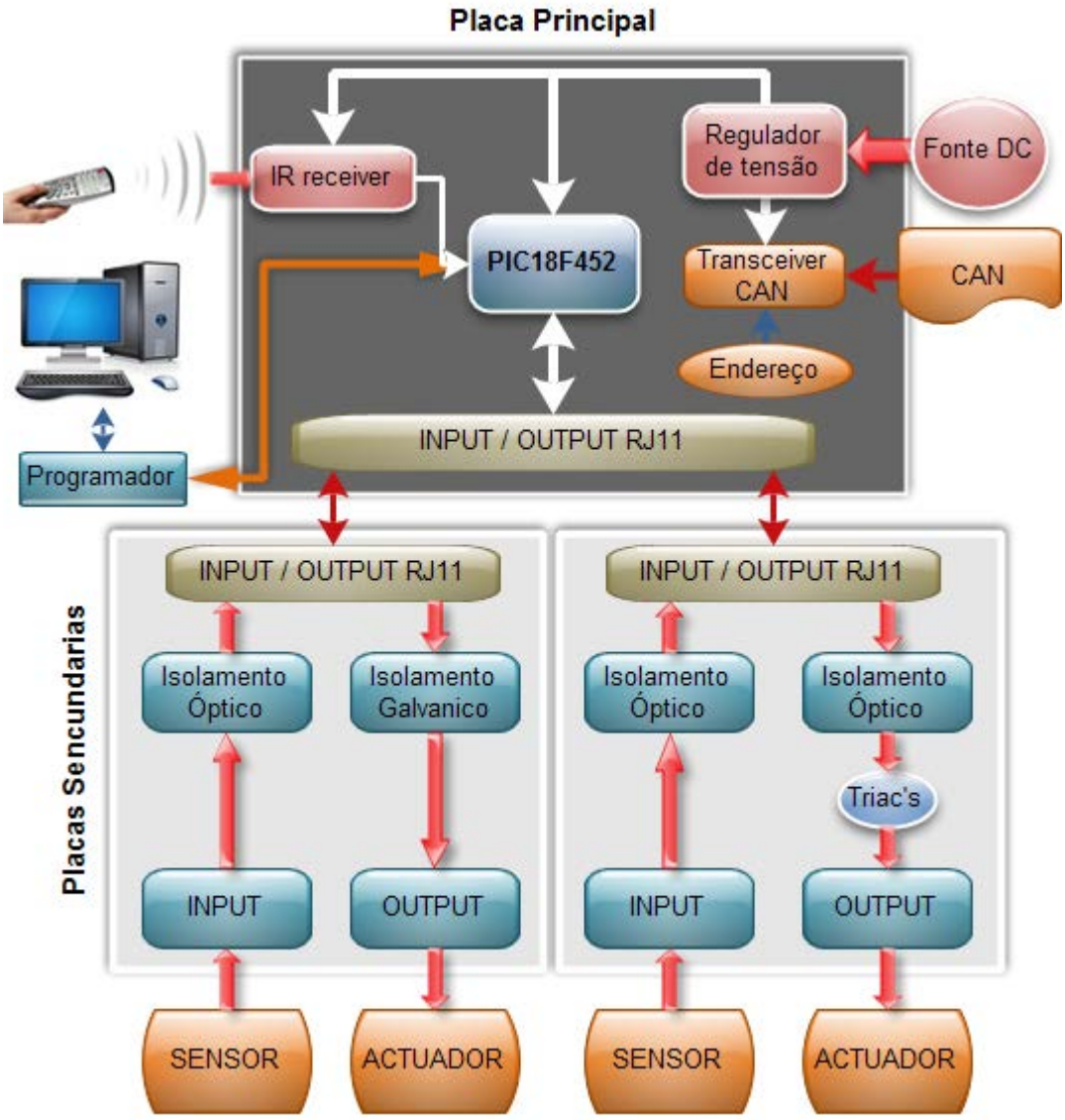

Figura 4.2- Esquema de blocos do sistema a desenvolver do DOMO_CAN

Este módulo para além de comunicar utilizando o barramento CAN tornará possível a recepção de comandos via infravermelhos, de modo a podermos controlar os dispositivos ligados na rede, tal como acontecia no Domoitech.

4.2- Pesquisas

Uma vez que este projecto tem como principal objectivo reduzir custos associados à utilização de redes KNX, nomeadamente, à interface à rede física, encapsulando este em CAN, será necessário que o microcontrolador a utilizar possua interface do tipo CAN. Sendo assim, toda a pesquisa incidiu sobre esta particularidade que o microcontrolador deveria possuir.

Foram pesquisados outros microcontroladores tais como os NXP, por serem dotados de alto processamento, interface Ethernet, USB e o seu preço ser menor ou igual aos seus concorrentes directos. O facto de estes possuírem interface *Ethernet* causou uma grande incitação para o seu uso pois, poderíamos com o mesmo projecto, desenvolver não só KNX sobre uma interface CAN, mas igualmente sobre *Ethernet*. Tal, se afiguraria uma característica bastante atractiva para este projecto.

Para a utilização destes microcontroladores era necessário a utilização de um sistema embutido como Linux ou uClinux [12] e drivers para se poder aceder à interface Ethernet. Foi feita uma pesquisa para se tentar saber quais os microcontroladores que eram compatíveis [13], ou para os quais haveria estes recursos de software disponíveis. O que se verificou foi que, para os microcontroladores seleccionados não se encontrou estes recursos. Estes recursos estavam sim disponíveis para uma gama de microcontroladores que não dispunham de interface CAN, ou apresentavam uma área que excedia o tamanho pretendido neste projecto. Sendo assim, optamos por escolher então microcontroladores da família PIC.

A nossa primeira escolha recaiu sobre a família PIC24H, que continha a interface desejada e possuía um tamanho razoável de *flahs memory* (superior a 32kbytes).

Device	Pins	Program Flash Memory (Kbyte)	RAM (Kbyte)(1)	Remappable Peripheral							External Interrupts(3)	RTCC	I²C™	CRC Generator	10-bit/12-bit ADC (Channels)	Analog Comparator (2 Channels/Voltage Regulator)	8-bit Parallel Master Port (Address Lines)	I/O Pins	Packages
				Remappable Pins	16-bit Timer(2)	Input Capture	Output Compare Standard PWM	UART	SPI	ECAN™									
PIC24HJ128GP504	44	128	8	26	5	4	4	2	2	1	3	1	1	1	13	1/1	11	35	QFN TQFP
PIC24HJ128GP502	28	128	8	16	5	4	4	2	2	1	3	1	1	1	10	1/0	2	21	SDIP SOIC QFN-S
PIC24HJ128GP204	44	128	8	26	5	4	4	2	2	0	3	1	1	1	13	1/1	11	35	QFN TQFP
PIC24HJ128GP202	28	128	8	16	5	4	4	2	2	0	3	1	1	1	10	1/0	2	21	SDIP SOIC QFN-S
PIC24HJ64GP504	44	64	8	26	5	4	4	2	2	1	3	1	1	1	13	1/1	11	35	QFN TQFP
PIC24HJ64GP502	28	64	8	16	5	4	4	2	2	1	3	1	1	1	10	1/0	2	21	SDIP SOIC QFN-S
PIC24HJ64GP204	44	64	8	26	5	4	4	2	2	0	3	1	1	1	13	1/1	11	35	QFN TQFP
PIC24HJ64GP202	28	64	8	16	5	4	4	2	2	0	3	1	1	1	10	1/0	2	21	SDIP SOIC QFN-S
PIC24HJ32GP304	44	32	4	26	5	4	4	2	2	0	3	1	1	1	13	1/1	11	35	QFN TQFP
PIC24HJ32GP302	28	32	4	16	5	4	4	2	2	0	3	1	1	1	10	1/0	2	21	SDIP SOIC QFN-S

Figura 4.3- PIC24H e os dispositivos seleccionados [6].

Contudo, além da particularidade do microcontrolador ter de incorporar uma interface CAN, a necessidade de outras particularidades foi posta em questão, à medida que a pesquisa era efectuada, como por exemplo, se deveria ou não possuir *EPPROM*, ou mesmo qual o tamanho da *flash memory* necessária. Chegou-se, entretanto, à conclusão que, para além da particularidade CAN, o microcontrolador deveria possuir EPPROM, pois seria necessária para guardar as configurações efectuadas por uma aplicação criada em java. Esta aplicação será descrita no Capítulo 5, no ponto 5.3-DOMO_CAN Monitor. As características pretendidas indicaram a família PIC18F como a mais adequada pois satisfazia todos estes detalhes de especificação almejados. Optou-se então por utilizar o microcontrolador PIC18F4685 tanto para o desenvolvimento do DOMO_CAN como para o DOMO_GATMONITOR pois, ambos necessitariam das mesmas especificações iniciais pretendidas.

4.3- Microchip PIC18F4685

Este microcontrolador é dotado de interface ECAN e não CAN mas o que não trará qualquer tipo de problema ou dificuldade pois ECAN é apenas um melhoramento do protocolo CAN. O ECAN tem três modos de funcionamento:

- Modo 0 – *Fully backward compatible Legacy mode*;
- Modo 1 – *Enhanced Legacy mode;*
- Modo 2 – *Hardware FIFO mode*.

Destes modos, o Modo 0 é completamente compatível com os modelos anteriores (CAN).

Descrição técnica:

- 8Bits
- Flash Memory Size: 96KB
- EEPROM Memory Size: 1024Byte
- RAM Memory Size: 3328Byte
- No. of I/O Lines: 36
- No. of ADC Inputs: 11
- No. of Timers: 4
- No. of PWM Channels: 5
- Clock Frequency: 40MHz
- Interface Type: ECAN, EUSART, I2C, PSP, SPI
- Min Supply Voltage: 4.2V
- Max Supply Voltage: 5.5V
- Termination Type: SMD
- Case Style: TQFP
- No. of Pins: 44
- Microprocessor/Controller Features: Self-Programming, ECAN, LIN, USART

- Number of bits in ADC:10
- Packaging Type: Peel Pack
- Digital IC Case Style: TQFP

Este microcontrolador tem um custo de 7,82€ para encomendas entre 10 e 99 unidades na distribuidora Farnell [5]. É dotado de todos os aspectos/características detalhadas como necessárias ao desenvolvimento do projecto porém, apresenta ainda outra característica que é importante analisar, que consiste no facto do seu encapsulamento ser do tipo TQFP, um tipo de encapsulamento SMD, ou seja tecnologia SMT (*Surface-mount technology*).

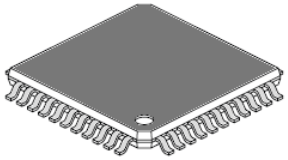

Figura 4.4 - Aspecto físico do encapsulamento [6].

Esta característica deve-se simplesmente ao facto de se terem delimitado as buscas conforme o mercado que dispomos. Uma vez que não temos nenhum tipo de problema em utilizar este tipo de tecnologia, a nossa escolha recaiu sobre este microcontrolador. Na Figura 4.5, apresentada a seguir, encontra-se o diagrama de pinos.

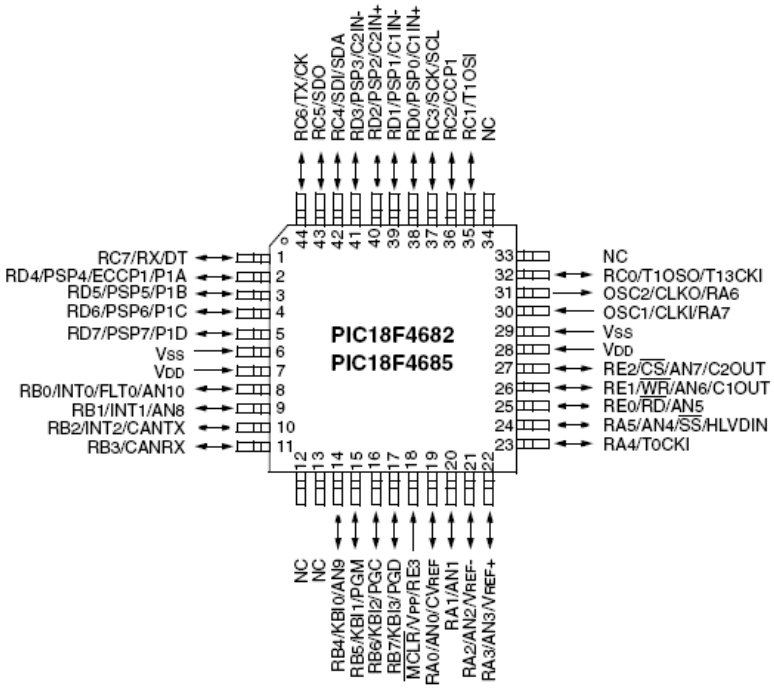

Figura 4.5 - Diagrama de pinos do PIC18F4685 [6].

Para além disso, é necessário utilizar um transceiver (driver de linha) ligado ao microcontrolador para podermos ter a interface CAN (CANH e CANL), uma vez que o

microcontrolador apenas implementa o protocolo a nível de software e não a nível da lógica do barramento CAN.

Isto deve-se ao facto de se querer proteger o micro, uma vez que, assim, em caso de avaria ou falha grave no barramento, que pudesse por em risco o mesmo, apenas se danifica este driver, ficando o micro protegido.

O *transceiver* que iremos utilizar será o MCP2551 da *microchip* [6] com um custo associado de 0,88€ para quantias entre 10 e 99 unidades encomendadas na distribuidora Farnel [5].

4.4- Esquemas Eléctricos

Nesta secção, iremos apresentar os esquemas eléctricos a desenvolver, onde encontramos os dois principais módulos, o modulo CAN e o DOMO_GATMONITOR. Foram, ainda, estudados esquemas de pequenos módulos suplementares e auxiliares que são necessários para a elaboração do projecto.

4.4.1- DOMO_CAN

Para o DOMO_CAN, como já referido anteriormente, iremos desenvolver duas placas de pequenas dimensões para a interligação dos dispositivos, de forma a se efectuar um isolamento da parte de potência.

4.4.1.1- Saídas em relés para o DOMO_CAN

Os circuitos eléctricos destas placas secundárias foram aproveitados do trabalho Domoitech uma vez que o DOMO_CAN apresenta uma estrutura idêntica.

Nesta placa o isolamento dos dispositivos de potência é efectuado por relés. Esta placa conta com duas entradas para dois sensores (interruptores) e duas saídas para dois actuadores. Desta forma os relés efectuam o isolamento da parte de potências entre as saídas de controlo e as entradas do microcontrolador incorporado no DOMO_CAN.

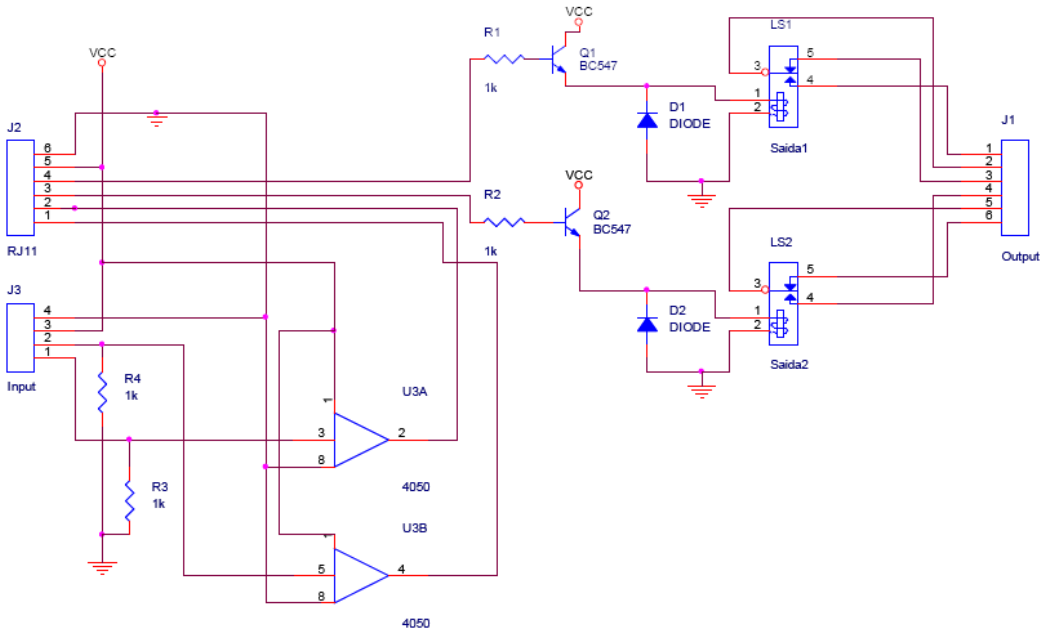

Figura 4.6 - Esquema da placa de saídas em Relés [3].

4.4.1.2- Saídas em Triac's para o DOMO_CAN

Tal como a placa anterior, esta permite controlar dois dispositivos e possui também duas entradas. Nesta opção a comutação é feita com o recurso a triac's por terem menores dimensões que os relés, bem como, o seu inferior preço.

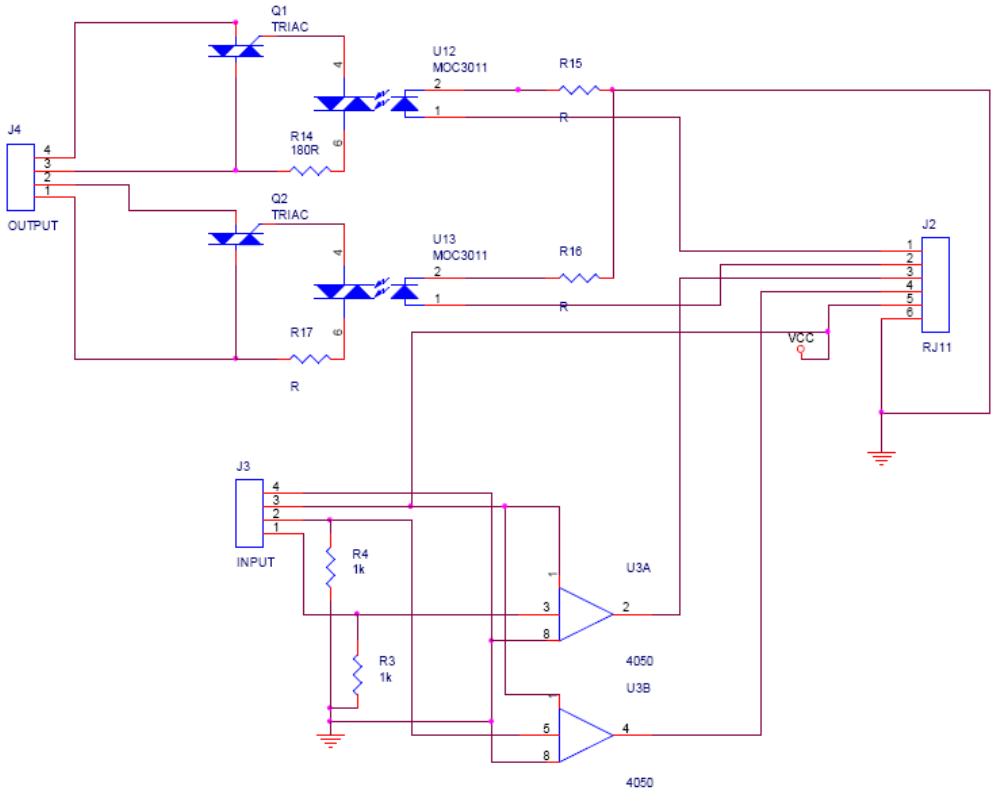

Figura 4.7 - Esquema da placa de saídas em Triac's [3].

4.4.1.3- DOMO_CAN

Esta é a placa principal do DOMO_CAN, por isso, a que contém todo o tipo de hardware de processamentos, bem como, as interligações com as placas de isolamento e com o Barramento CAN. Contém ainda um DIP Switch de quatro vias para se poder seleccionar o endereço de cada DOMO_CAN.

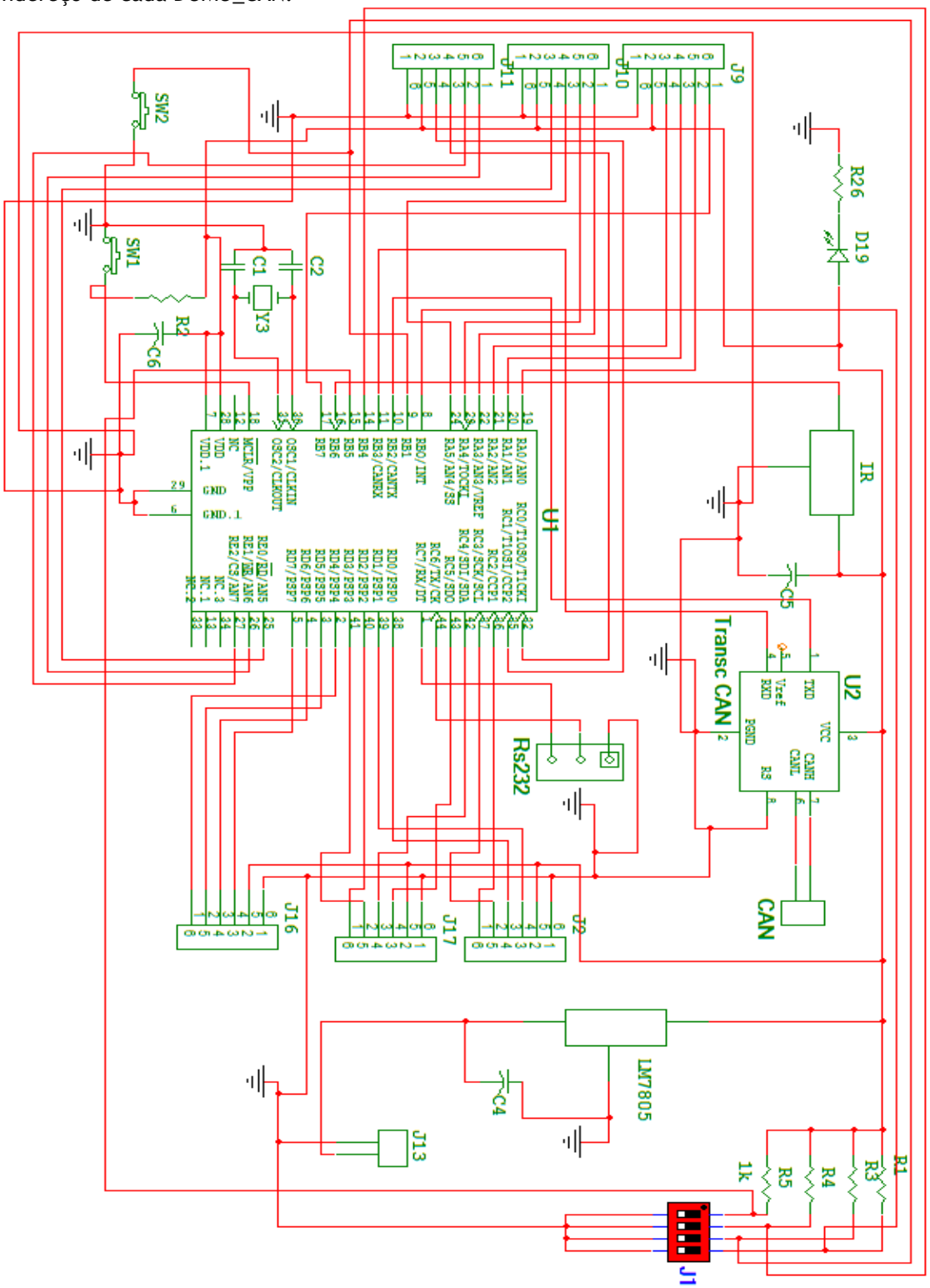

Figura 4.8 - Esquema da placa principal a desenvolver

4.4.2- DOMO_GATMONITOR

Este é um esquema do DOMO_GATMONITOR que se construiu. Este módulo tem a particularidade de permitir seleccionar qual o dispositivo de porta série se pretende utilizar, isto é, todo o funcionamento do sistema está construído para que, mais tarde, se for necessário ou desejável, por exemplo, utilizar uma XPORT para que toda a rede CAN possa ser interligada a uma rede Ethernet, seja possível com apenas a necessidade de se efectuar umas alterações no código de programação, sem ser necessário alterar o hardware, a não ser incluir a XPORT.

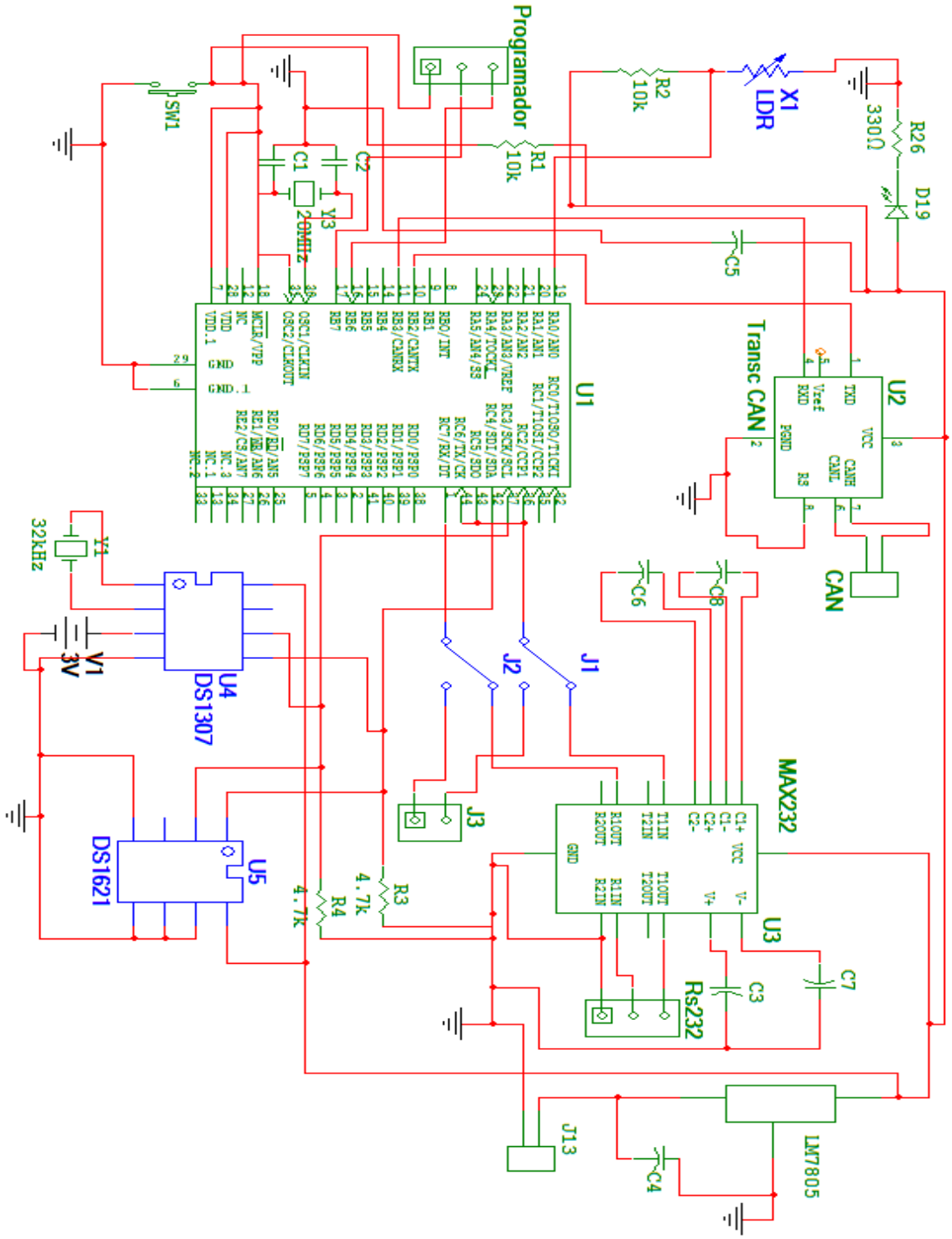

Figura 4.9 - Esquema eléctrico do DOMO_GATMONITOR

4.5- Aspecto final do Hardware desenvolvido

Após o estudo da estrutura e do circuito eléctrico procedeu-se ao desenvolvimento das placas em si. Para a construção do layout dos PCBs (*printed circuit board*)l, foi utilizado o Software *NI Circuit Design Suite* com a excepção do DOMO_GATEMONITOR em que não se chegou a efectuar nenhum layout, devido a questões relacionadas com a gestão de tempo (é necessário esperar algum tempo quando se manda fazer uma PCB).

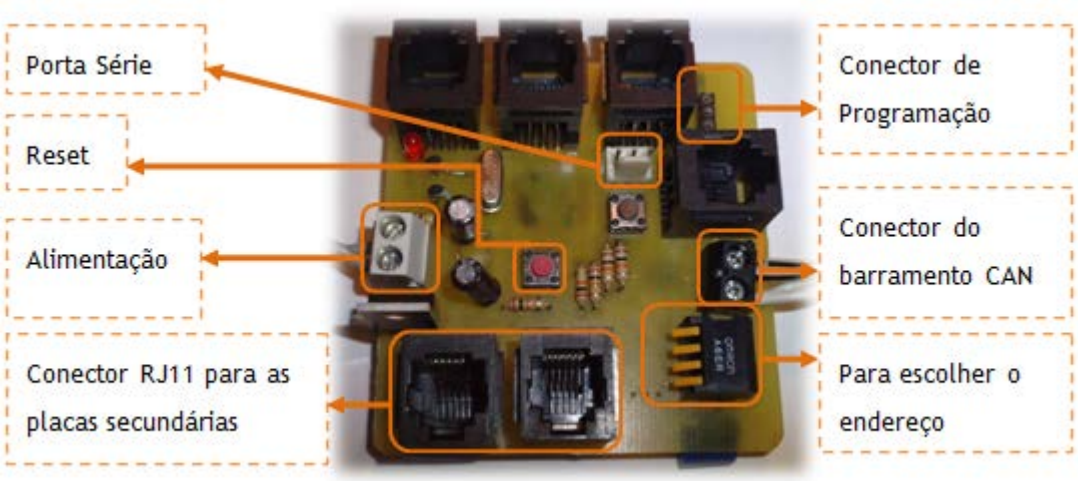

Figura 4.10 - Placa do DOMO_CAN

Foram desenvolvidas duas placas DOMO_CAN para efectuar a simulação da rede CAN. Para as placas de potência foram utilizadas as placas anteriormente desenvolvidas no trabalho Domoitech devido a uma questão de gestão do tempo, pois como já referido, estas são idênticas ao que seria necessário desenvolver.

Uma vez que não se efectuou nenhum *layout* devido ao tempo necessário para fazer uma PCB, optou-se por construir um protótipo em placas PCB pré-furadas para a construção deste módulo. Este módulo é constituído por duas placas distintas, sendo uma delas a placa que contém o hardware necessário ao seu funcionamento e a restante a placa que contém todos os sensores utilizados. Desta forma, é possível remover em caso de não ser necessário.

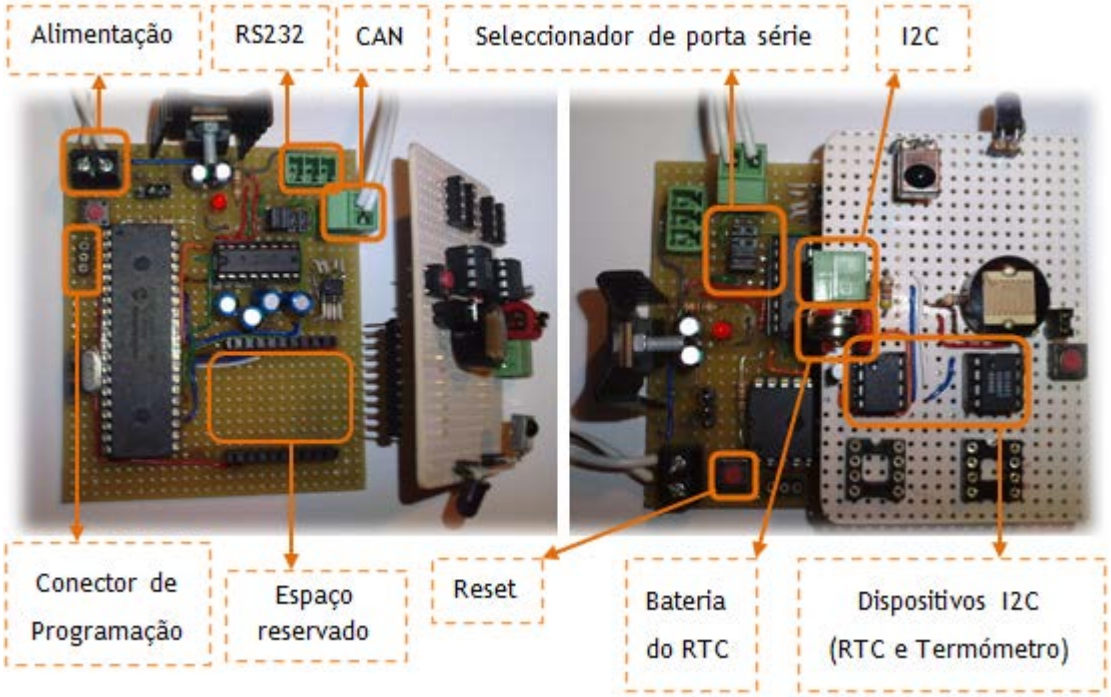

Figura 4.11 - Placa do DOMO_GATMONITOR

O DOMO_GATEMONITOR tem a possibilidade de desacoplar a placa responsável pelos sensores que este módulo incorpora. Mesmo com esta placa dos sensores colocada é possível colocar uma XPORT no espaço reservado pois foi pensado para essa situação. Através do seleccionador da porta série pode-se escolher qual o dispositivo MAX232 ou XPORT (em caso de se colocar) ao qual o microcontrolador está ligado.

Capítulo 5

Projecto de Software

Depois do desenvolvimento do Hardware, avançou-se para uma fase de implementação de software dos módulos que, como já referido, são 2 grandes módulos: DOMO_CAN e DOMO_GATMONITOR. Estes dois módulos vão ser descritos neste capítulo quanto ao seu funcionamento e a forma como foram desenvolvidas as suas camadas de software que implementam os requisitos. Foi desenvolvida em conjunto uma Aplicação Java com o nome de DOMO_CAN Monitor que será também descrita neste capítulo.

5.1- DOMO_CAN

Dado que este módulo teve como base de software o projecto Domoitech, foi necessário um estudo do mesmo. Este projecto Domoitech encontra-se dividido em quatro camadas, que correspondem às camadas definidas pelo protocolo KNX/EIB com exclusão da camada de rede, pois trata-se de uma camada pequena e foi integrada juntamente com a de transporte.[3] A primeira camada, Physical Layer, é responsável pela interligação entre o software e o hardware, controla a recepção e envio de tramas KNX/EIB, faz a leitura das entradas digitais e implementa a descodificação do protocolo RC5. Na segunda camada, Transport Layer, é feita a descodificação e a codificação das tramas EIB, faz-se o tratamento dos endereços físicos e endereços de grupo, do check octeto e do control field. A camada seguinte do software trata dos dados que dizem respeito ao campo de transporte da trama EIB.

Por último, a camada da aplicação que trata os serviços EIB e os EIS's. Esta camada será tratada com maior detalhe mais à frente devido à sua complexidade.

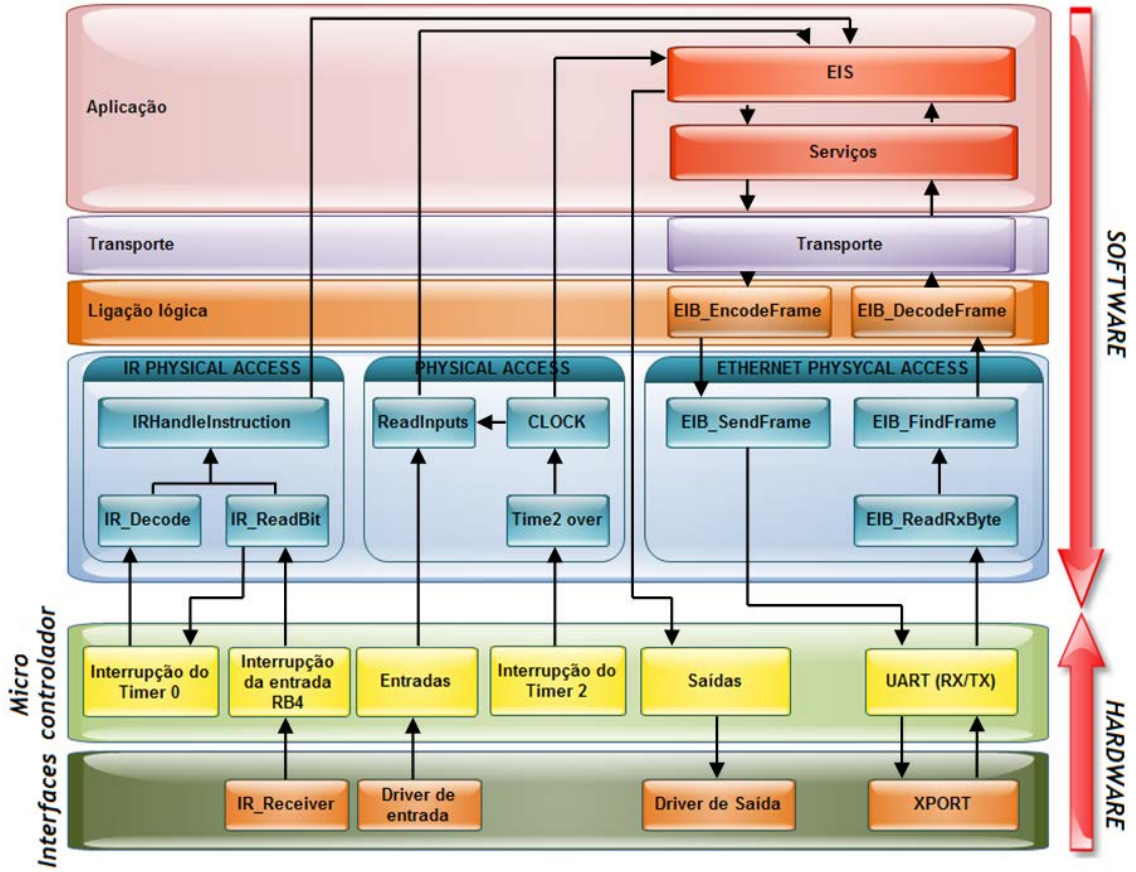

Figura 5.1- Diagrama de blocos das camadas do Domoitech

A Figura 5.1 representa as quatro camadas referidas em que o projecto Domoitech está construído. As setas representadas indicam o sentido do fluxo de informação entre processos e módulos ou entre processos e o nível de hardware. Neste sistema os processos, que estão na camada física, acedem directamente ao hardware. Estes processos têm as seguintes funções:

- **EIB_ReadRxByte:** Ao receber um byte pela porta série este processo guarda-o num vector circular. Ao oitavo byte activa o processo EIB_FindFrame, pois significa que já recebeu o número mínimo de bytes de uma trama EIB. A comunicação entre estes dois processos é feita através do vector Rxbufer.[3]
- **EIB_FindFrame:** Procura no vector Rxbufer por uma trama EIB válida e sempre que esta é encontrada o processo EIB_DecodeFrame é activado.
- **EIB_SendFrame:** Envia a trama EIB guardada numa variável global TxBuf. É este vector que serve para fazer a comunicação entre este processo e todos aqueles que necessitam de enviar uma trama EIB pela porta série.
- **IR_ReadBit:** sempre que ocorra uma interrupção na porta B, mudança de flanco do pino RB4, o que significa que está a receber um comando IV, este processo é activo. Descodifica qual o bit recebido e guarda-o na variável RC5_FrameBuf. Se recebeu 14 bits e se ocorreu interrupção do timer0 então activa o processo IR_Decode.

- **IR_Decode:** Descodifica qual o comando RC5 recebido e activa o processo IRHandleInstruction. Para a comunicação entre estes dois processos é utilizada a variável RC5_Instruction.
- **IRHandleInstruction:** Este processo chama o serviço GroupValueW_REQ para procurar qual o endereço de grupo correspondente ao comando RC5 recebido e constrói a trama. Após a trama construída activa o processo EIB_SendFrame para este enviar a trama.
- **Timer2_Over:** Processo que é activo sempre que uma interrupção do timer2 acontece, o que fará com que seja incrementada a variável de contagem contido no módulo clock.
- **Clock:** Implementa dois contadores responsáveis pela activação de dois processos com períodos de execução diferentes.
- **ReadInputs:** Este processo verifica o estado das entradas e caso haja alguma entrada activa, testa o intervalo de tempo durante o qual esta foi activa, para detectar impulsos de pequena e longa duração. Este método é usado para implementar dois interruptores lógicos existindo apenas um único interruptor físico. Caso haja mudança de estado no interruptor lógico, chama o serviço GroupValueW_REQ, o qual procura o endereço de grupo correspondente, na tabela de associação, e constrói a trama para ser enviada pelo processo EIB_SendFrame sendo por isso este processo activado, após a construção da trama. Este processo é activado ciclicamente com um tempo de refrescamento de 20ms.

Após a análise do trabalho Domoitech identificou-se quais os processos a serem alterados e de que forma seriam construídos. Estes novos processos construídos são totalmente compatíveis com o restante trabalho Domoitech para assegurar que o trabalho continuaria a funcionar após a remoção dos processos que não interessavam e da introdução dos novos. A Figura 5.2 apresentada a seguir representa a área de intervenção no projecto Domoitech por parte deste novo DOMO_CAN que foi desenvolvido.

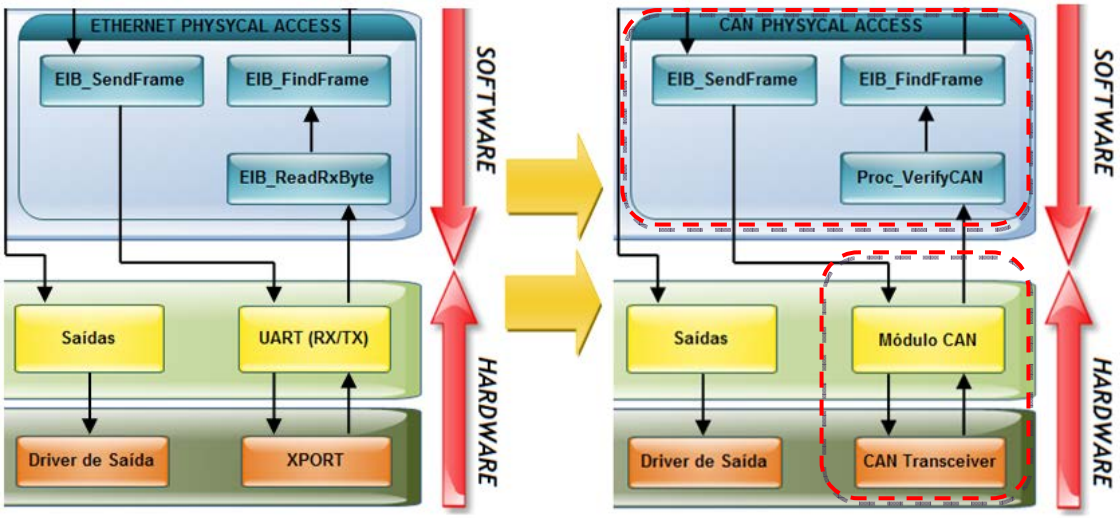

Figura 5.2- Alterações efectuadas na camada física.

Como se pode verificar na Figura 5.2, foi necessário efectuar alterações. Essas alterações efectuaram-se a nível do hardware, que em vez de uma XPORT ligada à porta UART do microcontrolador, temos um transceiver de CAN que permite fazer o acoplamento da linha ao módulo CAN. Outras duas alterações que foram necessárias efectuar deveram-se a esta anterior, pois como a camada física foi alterada foi necessário alterar o processo EIB_ReadRxByte para um novo que, em vez de receber os dados pela porta serie do micro recebe pela porta CAN, o que altera a estrutura de recepção e a ligação entre o processo EIB_FindFrame. Foi também necessário alterar o processo EIB_SendFrame, que se manteve com o mesmo nome pois, apesar de ter sido alterado para continuar a exercer a função de interligação entre a camada superior e o envio da trama KNX/EIB sobre CAN, a estrutura manteve-se, sendo alterado apenas o algoritmo de envio de dados, que será explicado neste subcapítulo.

5.1.1- Inicialização do módulo de CAN

Para se poder usar o módulo CAN incorporado no microcontrolador foi necessário efectuar uma inicialização do módulo CAN, o que obrigou a um conhecimento dos registos de controlo da configuração do módulo CAN presente no PIC18F4685 utilizado.

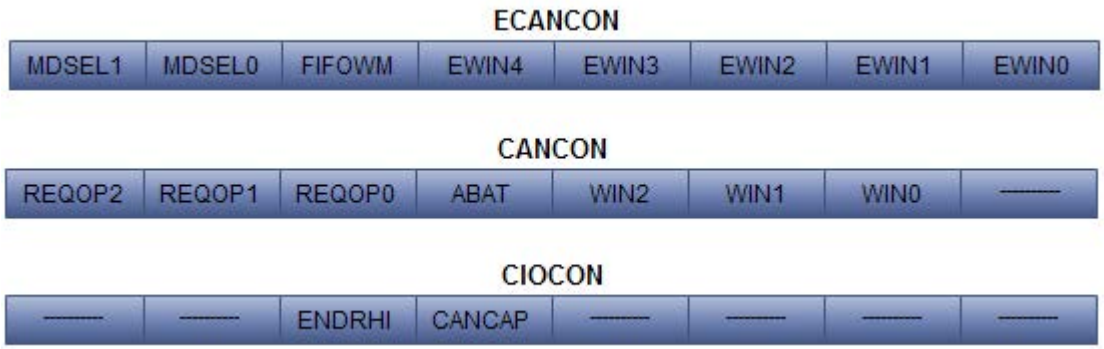

Figura 5.3- Registos para definirem o modo de funcionamento do módulo CAN

O registo ECANCON permite definir entre três modos de funcionamento: o modo *Legacy mode* (Modo 0, por defeito), *Enhanced Legacy mode* (Modo 1), Enhanced FIFO mode (Modo 2). O modo escolhido foi o *Legacy mode,* que se encontra atribuído por defeito, por ser o modo que permite que este módulo seja compatível com outras placas que utilizem CAN com um módulo diferente deste tipo. Como já referido atrás, este trata-se de um módulo com capacidades acrescidas mas, se forem activas torna-o incompatível com módulos que não as tenham. O registo CANCON é utilizado para se escolher o modo pretendido:

- 1xx = *Request Configuration mode*, em que xx significa que o valor não interessa;
- 011 = *Request Listen Only mode;*
- 010 = *Request Loopback mode;*
- 001 = *Request Disable mode;*
- 000 = *Request Normal mode;*

Para escolher um destes modos são utilizados os bits 5,6 e 7. Esta escolha tem a particularidade do seu efeito não ser imediato, daí o nome *Request* antes de cada opção, o que significa que apenas está pedido para se mudar para o modo seleccionado, o que obriga a uma verificação para se saber se o pedido já se realizou, como podemos observar na Figura 5.4 que representa o fluxo de código para a inicialização do módulo CAN.

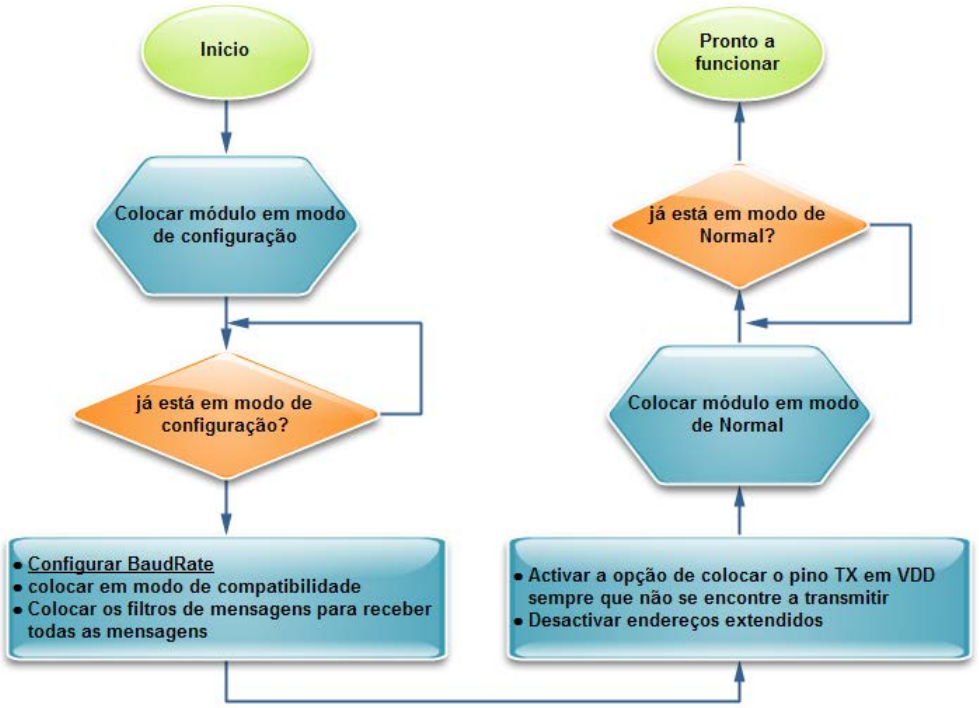

Figura 5.4- Fluxograma da estrutura utilizada para a inicialização do módulo CAN

Outro aspecto importante na configuração inicial é a determinação do *Baudrate* utilizado, o qual deve ser igual em todos os módulos na rede para que estes possam estar sincronizados e conseguirem comunicar. Neste projecto foi utilizado um cristal de 20MHz e foi definido um *Baudrate* de 125kBit/s.

Como explicado no Capítulo 3, no subcapítulo 3.3-*Bit Timing*, existem parâmetros que quando definidos, determinam o *Baudrate* utilizado. Esses parâmetros encontram-se em registos especiais do módulo CAN.

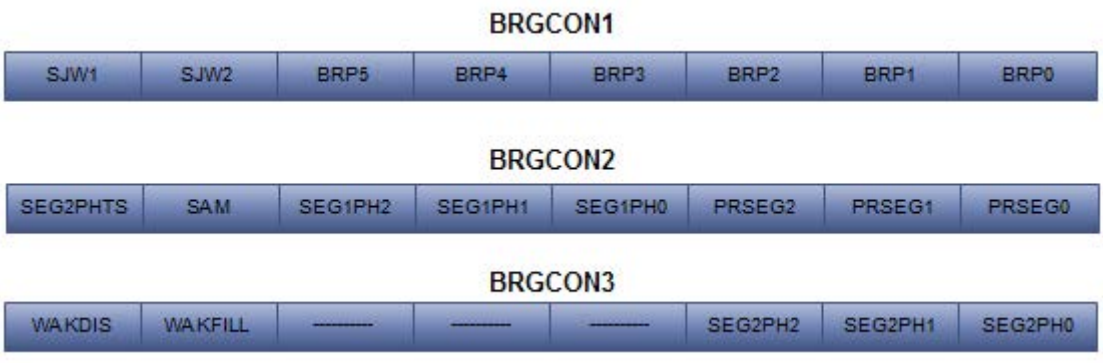

Figura 5.5- Registos para definirem o *Baudrate* do CAN no PIC18f4685

Na Figura 5.5 podemos observar o registo BRGCON1 onde é efectuada a configuração do valor do parâmetro *Baud Rate Prescaler*, que se situa nos seis primeiros bits como se pode verificar, e do parâmetro *Synchronized Jump Width bits* que são os últimos dois bits mais

significativos. Neste registo foi colocado para o parâmetro *Baud Rate Prescaler* o valor quatro e para o parâmetro *Synchronized Jump Width* bits o valor zero. O registo BRGCON2 define os parâmetros *Propagation Time Select* (bits 0,1,2) com o valor dois, *Phase Segment 1* (bits 3,4,5) com o valor cinco, *Sample of the CAN bus Line* que é apenas um bit, o sexto bit deste registo e foi definido com o valor um, *Phase Segment 2 Time Select bit* com valor igual a zero. Por último, o registo BRGCON3 que, como se pode verificar, contém três bits que não são utilizados. Os três primeiros referem-se ao parâmetro *Phase Segment 2 Time Select* que foi definido com o valor cinco, os restantes dois bits referem-se a uma opção de *Wake-up* e foram ambos definidos com o valor zero, o que indica que a opção está activa mas que não são utilizados filtros para a opção de *wake_up*. Assim, com os valores definidos podemos verificar que valor de *baudrate* foi utilizado. Sabendo que TQ (*Time Quantum*) é igual a duas vezes o valor de Baud Rate Prescaler mais um, sobre o valor do cristal utilizado que foi igual a 20MHz, temos um TQ de 0,0000005 segundos. A partir deste valor podemos somar as parcelas que correspondem ao tempo nominal de um bit. Consultando a *datasheet* do PIC18F4685 verificou-se que as parcelas a somar tinham os seguintes valores, com base nos parâmetros definidos, (que não é mais que o valor definido mais uma unidade):

- *Synchronization jump width time* (valor definido-0) -> uma unidade TQ
- *Phase Segment 1 time* (valor definido-5) -> seis unidades TQ
- *Propagation time* (valor definido-2) -> três unidades TQ
- *Phase Segment 2 time* (valor definido-5) -> 6 unidades TQ

Fazendo então a soma das parcelas que representam o tempo de bit, chegamos ao valor de 0,000008 segundos. Como o *Baudrate* é o inverso deste tempo obtemos finalmente o valor de 125000bit/s, o que mostra que os valores de configuração foram bem definidos.

Para se definir o endereço CAN de cada DOMO_CAN é utilizado o *Dipswitch* de quatro vias em hardware. Como apenas é possível alterar quatro bits obtêm-se dezasseis combinações, o que permite ligar dezasseis DOMO_CANs. Os endereços CAN iniciam no valor 16 até ao valor 31. Os endereços de zero a dezasseis são utilizados para, as tramas anteriormente definidas, por exemplo. O Endereço KNX/EIB físico de cada DOMO_CAN está definido como sendo igual ao endereço utilizado para o endereço CAN. Assim, os endereços KNX/EIB ficam definidos como valores entre 0.1.16 até 0.1.31 (valores apresentados na denotação usada em KNX/EIB).

5.1.2- Camada Física

Depois da comunicação a funcionar, foi então criado o processo que atendia à recepção de tramas CAN disponíveis que, como referido, intitula-se de Proc_VerifyCAN e foi efectuada a alteração nos processos EIB_SendFrame e no CLOCK. Foi necessário rever a

interrupção do timer2 que o processo CLOCK utilizava, para introduzir uma variável de contagem referente ao tempo de activação do processo Proc_VerifyCAN, sendo esse tempo dado então pelo número de vezes que a interrupção do timer2 é accionada, ou seja, se a variável de contagem for igual a dois então o processo Proc_VerifyCAN é activado de duas em duas interrupções do timer2.

Primeiramente, iremos explicar as alterações efectuadas no processo EIB_SendFrame, como se configurou o módulo CAN contido no PIC18F e, por fim, o processo Proc_VerifyCAN.

5.1.2.1- EIB_SendFrame

Este processo é responsável por enviar a trama KNX/EIB para o barramento CAN. Contudo, como já referido no Capítulo 3 - Protocolo CAN, no subcapítulo 3.5-Formato e tipo de tramas, uma trama CAN apenas contém um campo de dados que consegue armazenar até oito bytes e, ainda, como referido no Capítulo 2 - Protocolo EIB/KNX, no subcapítulo 2.3-Formato da trama, uma trama KNX/EIB pode conter até um máximo de vinte e três bytes. Devido a esta diferença, foi necessário dividir as tramas KNX/EIB para serem enviadas em tramas mais pequenas de apenas oito bytes.

Para o envio, é assim criado um buffer que contém a trama KNX/EIB a ser enviada. Desse buffer são retirados os primeiros oito bytes (não é necessário verificar se o buffer contém pelo menos oito, pois uma trama KNX/EIB tem um tamanho mínimo de oito bytes que representa uma trama com o campo de dados vazio) e enviados por CAN, retirando-se seguidamente outra parcela de oito bytes ou o número de bytes restantes do buffer, para tornar a enviar por CAN. Este Processo tem a particularidade de que, uma vez activo este desactiva as interrupções, voltando a activa-las apenas quando concluído o processo de envio.

5.1.2.2- Proc_VerifyCAN

Este novo processo Proc_VerifyCAN é responsável pela verificação da existência e recepção de dados disponíveis no barramento CAN. Este Processo é activo ciclicamente com um tempo de refrescamento de 13ms, associado a interrupção do timer2 onde é incrementada uma variável de contagem contida no módulo CLOCK que define o número de interrupções necessárias para o processo Proc_VerifyCAN ser activado. Para isto foi

necessário fazer uma alteração no módulo CLOCK para que este fizesse esta função como faz igualmente para o processo ReadInputs.

Como uma trama KNX/EIB pode ter até vinte e três bytes foi necessário dividir a trama para que esta pudesse ser colocada em pequenas tramas de oito bytes, que é o tamanho de uma trama CAN. Desta forma apareceu um outro problema:quando um módulo DOMO_CAN estiver a transmitir uma trama KNX/EIB e um outro módulo tentar enviar uma outra trama KNX/EIBe, no caso deste segundo ter um endereço que lhe permita ter uma prioridade superior ao que anteriormente se encontrava a transmitir, este iniciará a sua transmissão entre as tramas CAN que correspondiam a uma trama KNX/EIB do primeiro módulo DOMO_CAN. A Figura 5.6 demonstra este problema.

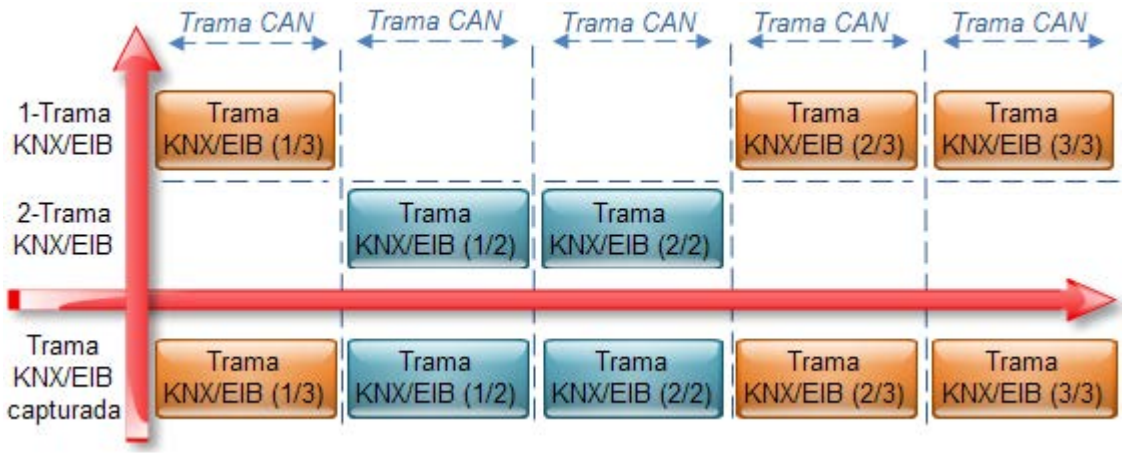

Figura 5.6- Representação do problema na recepção de uma trama KNX/EIB dividida em várias tramas CAN.

Para resolver este problema utilizou-se um mecanismo que recorre a vários vectores de tamanho superior ao tamanho máximo de uma trama KNX/EIB. Como o tamanho máximo de uma trama KNX/EIB é de vinte e três bytes então foram utilizados vectores de vinte e quatro bytes. Cada vector tem associado um campo que identifica o endereço CAN para o qual está associado, isto é, sempre que uma trama CAN chega e, se esta é uma parte de uma trama KNX/EIB (pois existem tramas CAN definidas que são utilizadas para processo de configuração que serão descritas mais á frente), verifica-se de quem é a mensagem pelo endereço de CAN e procura-se nos campos associados aos vectores se existe um que não esteja ocupado (campo ID=0) e guarda-se a mensagem no vector, incrementando um outro campo associado a esse vector, chamado Idade.

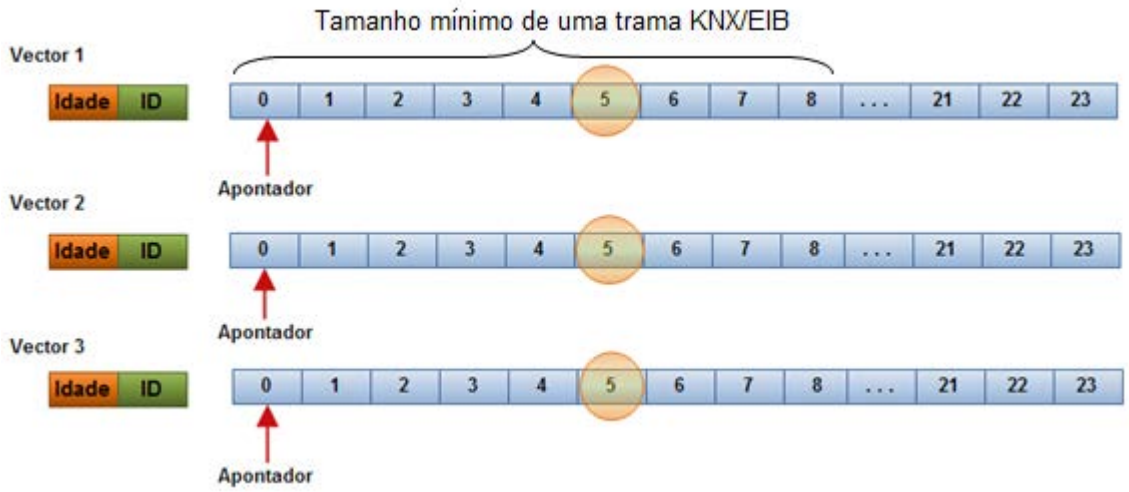

Figura 5.7- Mecanismo de recepção de tramas

Assim, quando chegar outra trama CAN do mesmo endereço, é procurado se esse endereço se encontra atribuído. Em caso afirmativo a trama CAN é adicionada ao vector ao qual esse endereço está associado. A trama KNX/EIB é detectada como completa através do índice número 5 do vector pois este contém o número de dados KNX/EIB presentes na trama e, assim, é possível saber quantas tramas CAN foram necessárias para refazer a trama KNX/EIB. Quando uma trama é detectada esta é enviada para o processo EIB_FindFrame e o vector é inicializado, bem como os campos a ele associados (Idade e ID).

No caso de todos os vectores estarem ocupados a receber tramas de endereços CAN e se houver uma outra trama para receber que não seja de nenhum endereço presente, nesse caso é feita uma pesquisa aos campos de Idade dos vectores para saber qual o vector que é mais antigo e este é inicializado, sendo a nova trama guardada nesse vector. Desta forma, permite que, no caso de um possível erro e de um vector estar ocupado com dados que não interessam, se for necessário um vector para guardar novas tramas CAN, os dados erróneos presentes são descartados.

A Figura 5.8 apresentada a seguir representa o fluxograma do método enunciado.

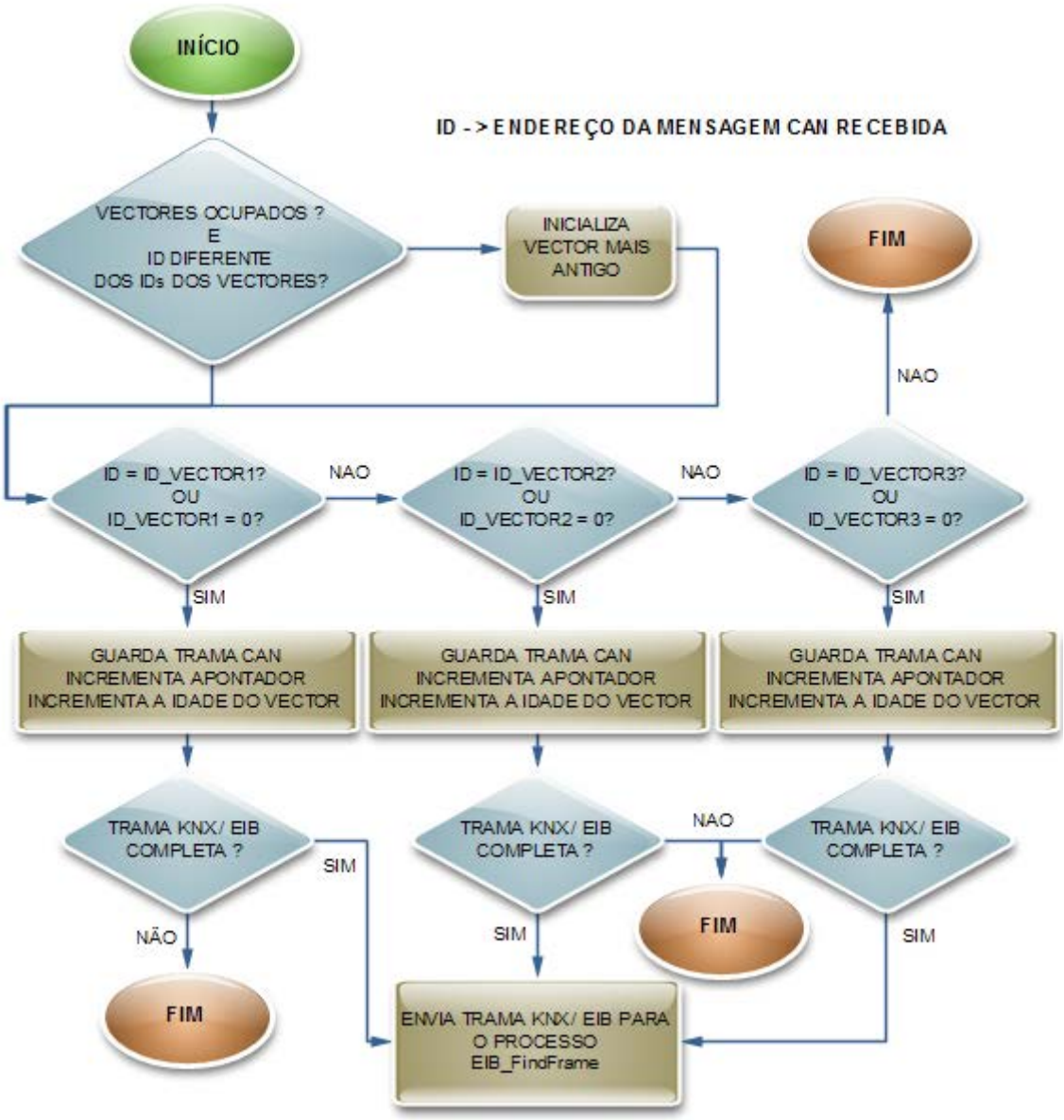

Figura 5.8- Fluxograma do Método de recepção das tramas KNX/EIB

Na Figura 5.8 é apresentado apenas o método de recepção das tramas CAN que formam uma trama KNX/EIB. Contudo, neste processo, é ainda possível receber outro tipo de informação proveniente do DOMO_GATMONITOR. Este processo permite, assim, verificar se a trama CAN disponível, se trata de uma trama CAN de configuração, de comando ou se é uma trama referente a uma trama KNX/EIB.

Uma trama de configuração é proveniente do DOMO_GATMONITOR e tem como objectivo configurar o DOMO_CAN para o qual a trama é destinada. Uma trama de comando também é uma trama proveniente do DOMO_GATMONITOR e permite enviar comandos como, apagar configurações guardadas ou efectuar um reset a todos os DOMO_CAN presentes. Estas duas tramas especiais (configuração e comando) têm endereços específicos conhecidos por todos na rede CAN, sendo efectuada desta forma a distinção entre todos os tipos de tramas.

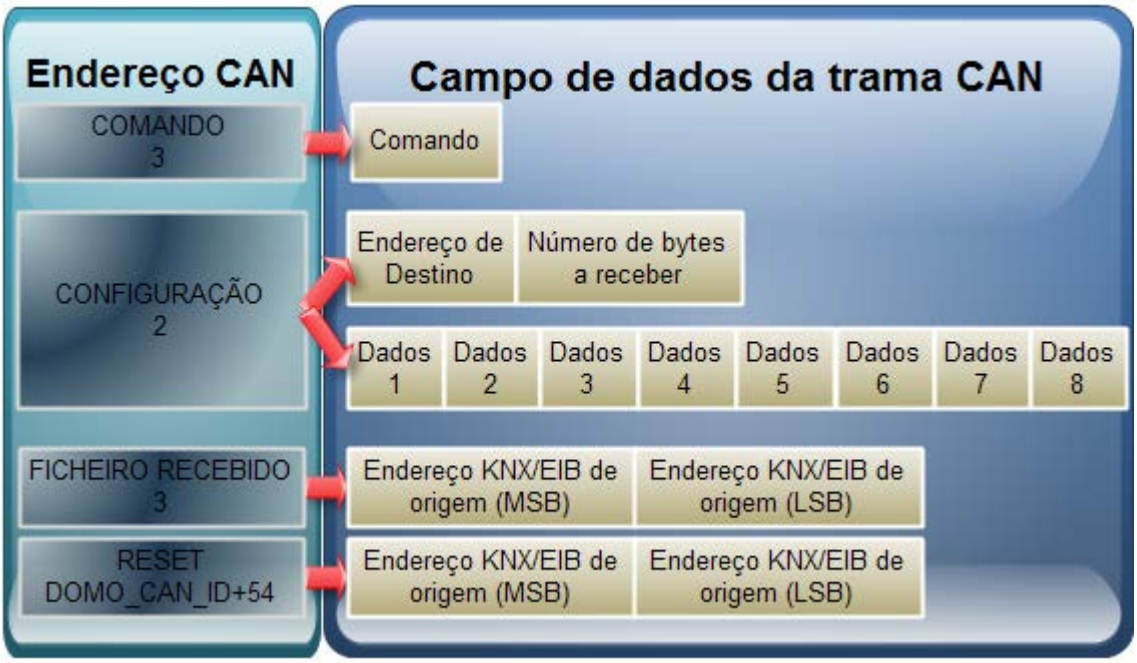

Figura 5.9- Tramas CAN criadas

A trama de comando apenas utiliza um byte de dados dos oito bytes disponíveis numa trama CAN. Já a trama de configuração utiliza os oito bytes em caso de serem precisos, pois os dados de configuração são provenientes de um ficheiro de configuração enviado pelo DOMO_GATMONITOR. Como se pode verificar pela Figura 5.9 a trama de configuração subdivide-se em duas tramas interdependentes, na medida em que é necessário enviar uma primeira trama, sempre que o DOMO_GATMONITOR pretende enviar um ficheiro de configuração para um módulo DOMO_CAN; de seguida, são enviadas tramas de configuração com os bytes contidos no ficheiro de configuração. A primeira trama utilizada, contém apenas 2 bytes de dados, um referente ao endereço do módulo DOMO_CAN para o qual o ficheiro é destinado e, outro com o número de bytes do ficheiro de configuração.

O DOMO_CAN contém ainda uma trama CAN predefinida que é enviada sempre que é efectuado um reset ou sempre que se liga o DOMO_CAN. Esta trama tem a funcionalidade de indicar ao DOMO_GATMONITOR que este reiniciou e se encontra operacional. O Endereço desta trama tem o nome de ID_RESET mas, difere de DOMO_CAN para DOMO_CAN devido ao facto de entrarem em conflito de acesso ao barramento. Se ligarmos o sistema, todos os DOMO_CANs irão responder com esta trama e se o endereço fosse igual teríamos um problema de colisão de dados. Desta forma, os endereços de reset são calculados somando apenas uma constante de valor igual a cinquenta e quatro, para garantir que estes se encontram fora da gama dos endereços atribuídos aos DOMO_CANs. Por consequência, também ficam fora dos endereços de configuração e de comando pois estes estão definidos com os valores dois e

três, respectivamente. Assim os endereços de reset ficam definidos entre setenta e oitenta e cinco.

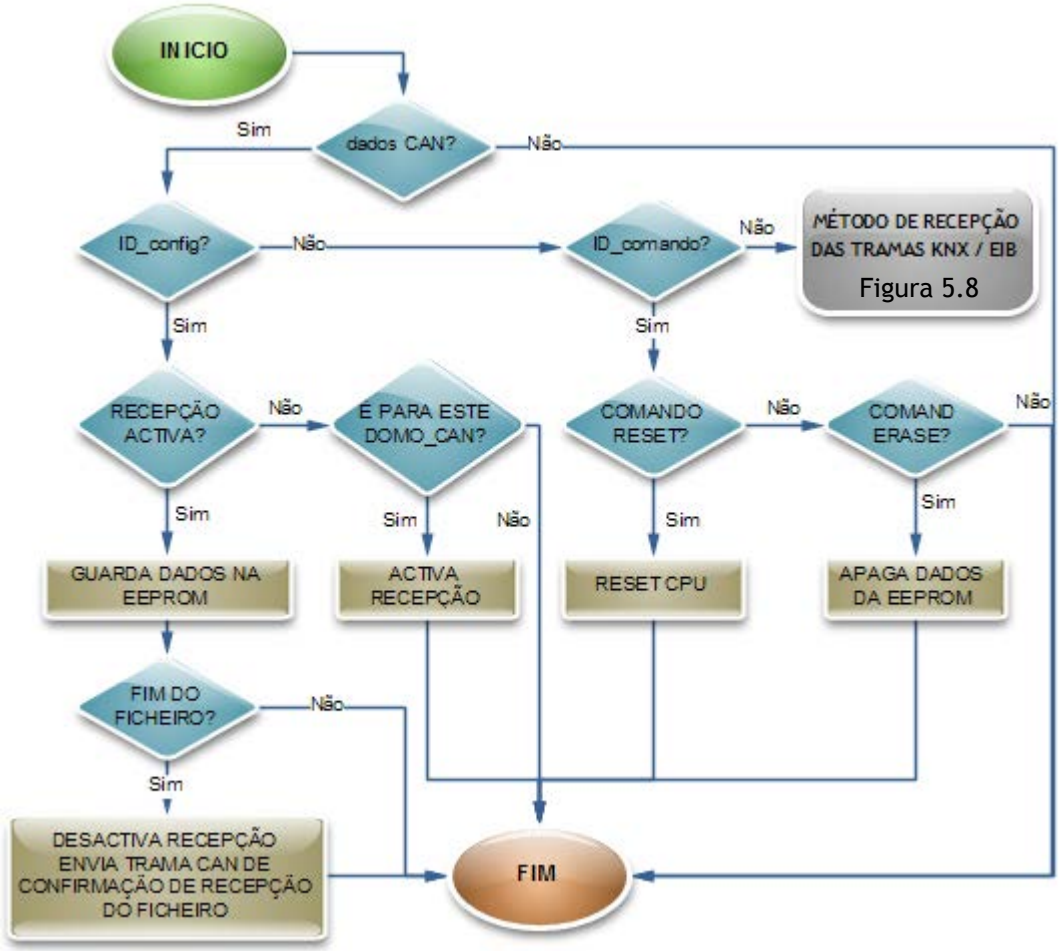

Figura 5.10- Fluxograma da estrutura do processo Proc_VerifyCAN

A Figura 5.10 representa o fluxograma do processo Proc_VerifyCAN. Uma vez activo este processo, verifica-se se algum buffer do módulo CAN se encontra com dados para serem lidos. Em caso de não existirem dados, o processo termina. Em caso afirmativo, é verificado se o endereço da mensagem CAN é um endereço de mensagem de configuração ou se é um endereço de uma mensagem de comando. De notar que os comandos são efectuados por todos os módulos, pois uma trama de mensagem referente a um comando não especifica um endereço de destino, como se pode verificar na Figura 5.9. Se o endereço não se referir a nenhum dos dois endereços mencionados então é porque se trata de uma parte de uma trama KNX/EIB.

Quando um endereço do tipo mensagem de configuração é recebido, verifica-se se uma *flag*, que indica que se está a receber configurações, está activa e, em caso afirmativo, os dados da trama são guardados num vector. Assim que o número de bytes recebidos atingir

o número de bytes a receber, os valores desse vector são copiados para a memória EEPROM do microcontrolador.

No entanto, se esta *flag* não se encontrar activa, é verificado se o ficheiro de configuração se destina a este DOMO_CAN. Em caso afirmativo, a *flag* de recepção de configuração é activada e, em caso negativo não. Assim que o ficheiro de configurações é enviado, o DOMO_CAN que recebeu o ficheiro responde com uma trama CAN, com um endereço específico designado FILE_DONE e, com um corpo de dados de apenas dois bytes que se referem ao seu endereço KNX/EIB. Desta forma, é possível verificar se o processo foi concluído com sucesso, ficando o DOMO_CAN com a nova configuração guardada em EEPROM.

5.1.3- Camada de Aplicação

Na Camada de Aplicação do projecto foi adicionada a capacidade de se poder efectuar uma configuração. Esta configuração não foi implementada segundo o protocolo KNX/EIB devido ao tempo de desenvolvimento deste projecto ser reduzido. Desta forma foi criado um protocolo de configuração e de comandos que foi descrito no ponto 5.1.2.2- Proc_VerifyCAN.

5.1.3.1- Configuração

A capacidade do DOMO_CAN se poder configurar advém do facto de este poder receber um ficheiro de configuração que lhe é destinado. Uma vez guardado na memória EEPROM, essa configuração passa apenas pela construção de uma tabela de associação que em vez de conter constantes predefinidas, contém ligações a endereços de memória onde estão guardados os valores que formam assim a tabela de associação.

A tabela de associação é uma tabela que permite associar saídas e entradas. Essa associação é efectuada atribuindo aos dispositivos (entradas e saídas) um endereço de grupo e a respectiva função de cada dispositivo.

Tabela 5.1 - Exemplo da tabela de associação com constantes predefinidas

Endereço	Dispositivo	Função	
2	1	0	Associa o dispositivo 1, o dispositivo 10 e
2	10	0	o dispositivo 23 com a função 0
2	23	0	
5	5	1	Associa o dispositivo 5, o dispositivo 12
5	12	1	com a função de 1

Na Tabela 5.1 é apresentada uma tabela de associação de dispositivos, com constantes predefinidas em código sem a possibilidade de estes valores poderem ser alterados. Em substituição a esta tabela anterior, foi construída uma outra que contém apontadores para a memória EEPROM, em que Read_eeprom é apenas uma função que retorna o valor para o qual o parâmetro de entrada da função aponta. CONF é apenas o nome do valor do apontador inicial.

Tabela 5.2 - Tabela de associação criada com apontadores para memória EEPROM

Endereço	Dispositivo	Função
Read_eeprom (CONF)	Read_eeprom (CONF+1)	Read_eeprom (CONF+2)
Read_eeprom (CONF+3)	Read_eeprom CONF+4)	Read_eeprom (CONF+5)
Read_eeprom (CONF+6)	Read_eeprom (CONF+7)	Read_eeprom (CONF+8)
Read_eeprom (CONF+9)	Read_eeprom (CONF+10)	Read_eeprom (CONF+11)
Read_eeprom (CONF+12)	Read_eeprom (CONF+13)	Read_eeprom (CONF+14)

5.2- DOMO_GATMONITOR

Este módulo do sistema denominado como DOMO_GATMONITOR permite efectuar uma gestão da rede, configurar os módulos DOMO_CAN, monitorizar a rede e até mesmo efectuar alterações em dispositivos que se encontram na rede.

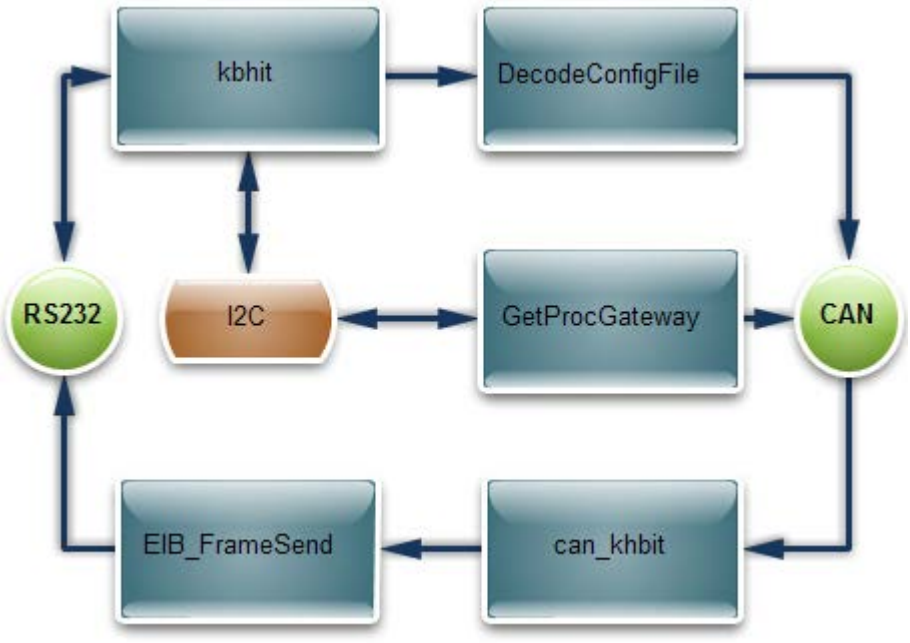

Figura 5.11 - Diagrama de blocos da estrutura dos processos

5.2.1- khbit

Analisando a Figura 5.11 verifica-se que existem dois processos que utilizam a interface com a porta RS232, o EIB_FrameSend e o khbit. O khbit é o segmento de código responsável pela interface com a aplicação desenvolvida em Java (descrita no subcapítulo 5.3-DOMO_CAN Monitor) através de tramas construídas e apresentadas na tabela seguinte.

Tabela 5.3 - Tramas construídas para comunicação via porta serie (RS232)

Tipo	Comando PC	Exemplo da trama enviada pelo PIC
Temperatura	1	X temp/t23,5
Luminosidade	2	X lumix/t
Hora	3	X horas/t12:34
Data	4	X data/t22:05:10
Receber ficheiro	10	X file/treceiving
	12493245803589942/	
		X file/tdone
Pedido de ficheiro	11	X file/tsendingdone/t124932480884202/
Activar Inspector	5	X inspector/tactivado
Desactivar Inspector	6	X inspector/tdesactivado
Reset todos DOMO_CANs	7	X resetall/treseting
Reset GATMONITOR	8	X resetgatw/treseting
Em que X corresponde ao tamanho da trama em bytes por exemplo 10temp/t23,5		

A tabela apresentada em cima representa o tipo de pedidos que a aplicação DOMO_CAN Monitor pode efectuar ao DOMO_GATMONITOR apresentados na coluna **"Comando PC"** e a resposta por parte do DOMO_GATMONITOR na coluna **"Exemplo da trama enviada pelo PIC"**. Por exemplo, sempre que é efectuado um pedido de temperatura é recebido o byte de valor 1 que dará origem a uma trama de resposta do tipo apresentada na Figura 5.12.

tamanho da trama em bytes	comando	Dados
10	temp	23,5

Figura 5.12 - Trama de resposta ao comando 1 (temperatura)

Os caracteres “/t” servem para que se possa separar os comandos dos dados, e esse tratamento é efectuado na Aplicação Java.

Um comando de grande importância implementado é o “receber ficheiro”. Este comando serve para enviar um ficheiro de configuração para o DOMO_GATMONITOR que, após a recepção deste, reenvia para o DOMO_CAN a qual é destinado. Quando o DOMO_GATMONITOR está a receber um ficheiro uma flag é activada, sendo desactivada assim que o ficheiro seja totalmente recebido. Se durante a recepção algum processo for activo, terá de esperar que o ficheiro seja recebido para poder ser efectuado.

Sempre que o DOMO_GATMONITOR recebe o comando com valor 10 este responde com uma trama 15file/treceiving, o que indica que está pronto para a recepção do ficheiro. A aplicação, ao receber esta notificação envia o ficheiro finalizando-o com “/” e, o DOMO_GATMONITOR envia uma trama a indicar que recebeu o ficheiro com sucesso 10file/tdone.

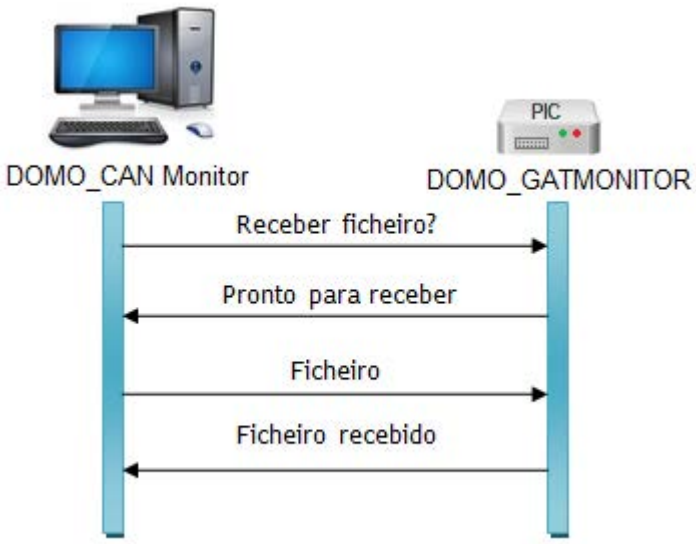

Figura 5.13 - Pedido de "receber ficheiro"

5.2.2- DecodeConfigFile

Após a recepção do ficheiro de configuração é necessário identificar a que DOMO_CAN este é destinado. Isso é conseguido pois o primeiro byte indica o endereço de destino e em caso de este conter o valor zero significa que a configuração é para o DOMO_GATMONITOR.

Em caso deste ficheiro se destinar para o DOMO_GATMONITOR, este é guardado em memória EEPROM e o processo GetProcGateway é activado. Mas, se o ficheiro contiver um endereço, ou seja, um valor diferente de zero, então este prepara uma trama CAN com o endereço de mensagem igual a dois que significa ID_CAN_CONFIG, reconhecido em toda a

rede CAN e com dois bytes no campo de dados. O primeiro byte representa o endereço e o segundo o número de bytes que o ficheiro contém.

Endereço	Dados	
ID_CAN_CONFIG=2	Endereço de destino	Tamanho do ficheiro

Figura 5.14 - Trama CAN de aviso de envio de ficheiro de configuração para a rede CAN

Os módulos DOMO_CAN ao receberem esta trama tratam a informação e a quem o endereço coincidir activa o processo de recepção do ficheiro, como demonstra o fluxograma Figura 5.10, anteriormente explicado. De seguida, este módulo DOMO_GATMONITOR começa o envio de ficheiro para o barramento CAN.

5.2.3- can_khbit

Sempre que uma trama CAN é enviada por parte dos módulos DOMO_CAN, esta é capturada. Como referido no ponto 5.1.2.2-Proc_VerifyCAN, os módulos DOMO_CAN para além de enviarem tramas que são partes de uma trama KNX/EIB, também enviam tramas de resposta ao módulo DOMO_GATMONITOR.

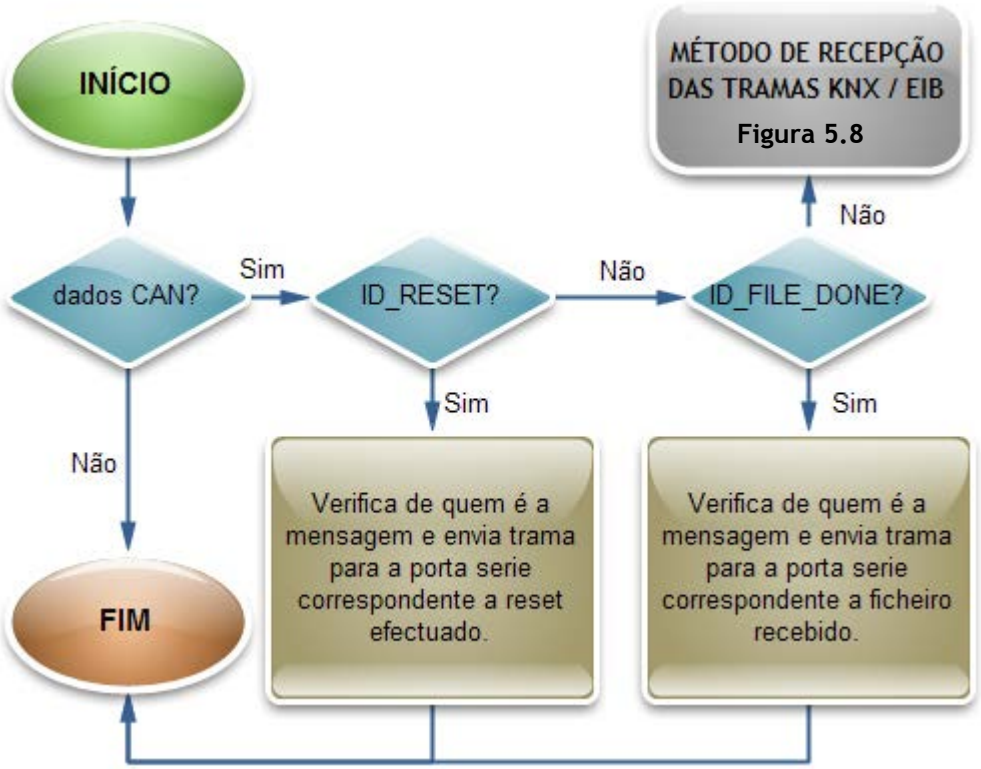

Figura 5.15 - Fluxograma da estrutura do código do processo can_khbit

Para a recepção das tramas que correspondem a uma trama KNX/EIB é utilizado o mesmo método que foi utilizado nos DOMO_CAN como demonstra a Figura 5.15, com uma

pequena diferença: sempre que uma trama KNX/EIB completa é recebida, em vez de a trama ser enviada para o processo EIB_FindFrame, no DOMO_GATMONITOR a trama é enviada para o processo EIB_FrameSend, sendo este activado.

Relativamente às tramas enviadas para a porta série, no que respeita ao "reset efectuado", este tem a forma, 22resetall/tdone/t0.1.08, em que 0.1.18 representa o endereço do módulo que reiniciou em notação KNX/EIB. A trama referente a "ficheiro recebido" tem a forma 21file/tcandone/t0.1.08.

5.2.4- EIB_FrameSend

Este processo permite enviar as tramas completas KNX/EIB pela porta serie, mas apenas é enviada a trama se anteriormente a função de inspector tiver sido activada pelo utilizador, através do envio do comando de valor 5 usando a aplicação DOMO_CAN Monitor como já referido. Esta função faz com que, num endereço da memória EEPROM definido como INSPECTOR_ADDR com o valor 3, seja guardado o valor 1, para que, se o sistema for reiniciado o estado da função esteja ainda definido. A trama KNX/EIB é enviada para aplicação usando a trama 36canframe/tBC|00|17|00|10|F1|00|81|25 em que a trama tem os valores dos campos divididos pelo carácter "|" e estão em hexadecimal.

5.2.5- GetProcGateway

Quando este processo é activo mediante o envio de um ficheiro de configuração destinado ao módulo DOMO_GATMONITOR, este verifica que funções/valores estão definidos, pois é possível activar as seguintes funções:

- Enviar trama KNX/EIB para um endereço de grupo definido e com um valor de dados igual a zero em caso da temperatura ser inferior a um limite definido ou com o valor um em caso da temperatura ser superior a um determinado limite, igualmente definido.
- Enviar trama KNX/EIB para um endereço de grupo definido e com um valor de dados igual a um, em caso da luminosidade ser inferior a um limite definido ou com o valor zero em caso da temperatura ser superior a um determinado limite igualmente definido.
- Enviar trama KNX/EIB para um endereço de grupo definido e com um valor de dados igual a zero ou igual a um, dependendo da opção escolhida a uma determinada hora definida.

Estas definições estão guardadas na EEPROM, e foram guardadas aquando da recepção do ficheiro de configuração, pois é este que define estas opções.

Uma vez que os ficheiros de configuração são criados pela aplicação, apenas serão descritos com mais pormenor a nível de construção no subcapítulo 5.3-DOMO_CAN Monitor.

5.2.6-I²C

O protocolo I²C será descrito apenas de forma breve, uma vez que, apenas foi implementado neste projecto como um extra de configuração, não fazendo parte dos requisitos do trabalho. Mas uma vez utilizado será especificado em que consiste e que dispositivos I²C foram utilizados.

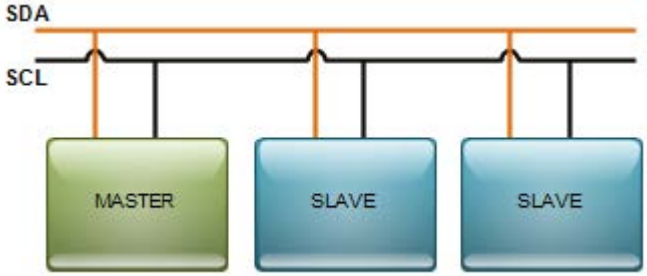

Figura 5.16 - Topologia da rede I²C

O I²C utiliza duas linhas, uma de dados chamadas SDA (data bits) e outra SCL (sinal de relógio). As mensagens enviadas pelo I²C contêm os endereços que definem qual o dispositivo que deve responder (cada dispositivo tem um endereço único)

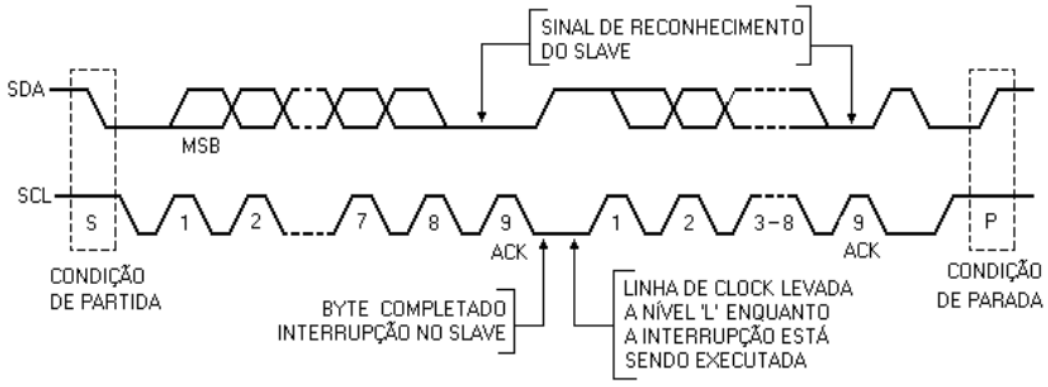

Figura 5.17 - Exemplo de uma transmissão de dados em I²C [19]

Para iniciar uma comunicação é necessário efectuar uma condição de *Start* e de seguida enviar o endereço para o dispositivo *slave*, onde este é constituído por sete bits e após o envio do endereço é enviado um bit que define o tipo de operação que se deseja efectuar, isto é, se queremos efectuar uma leitura ou se pretendemos escrever no *slave*. Após estes oito bits o *slave* responde com um bit de *acknowledge*. Após o envio de um byte o receptor deve responder sempre com este bit de *acknowledge*.

Dispositivos

Foram utilizados dois dispositivos I^2C que permitem saber as horas e a data actual bem como a temperatura ambiente onde o DOMO_GATMONITOR se encontra.

Para detectar a temperatura foi utilizado um DS1621 que tem uma precisão de 0.5 graus Célsius. A resposta ao comando de leitura de temperatura é dada em dois bytes, em que o primeiro representa a parte inteira da temperatura e, o segundo, a parte decimal. Esta última apenas tem duas possibilidades de valores: uma representa o 0.5 e a outra representa o 0.0. Se este segundo byte (da parte decimal) for igual a 128, então existe o meio grau; no caso de ser igual a zero, não existe meio grau.

O outro dispositivo é um RTC (*Real Time Clock*) que após definida a hora e a data actual, mantém a contagem mesmo que se desligue a alimentação do DOMO_GATMONITOR, através de um cristal e de uma alimentação extra.

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00–59
01h	0	10 Minutes			Minutes				Minutes	00–59
02h	0	12	10 Hour	10 Hour	Hours				Hours	1–12 +AM/PM 00–23
		24	PM/ AM							
03h	0	0	0	0	0	DAY			Day	01–07
04h	0	0	10 Date		Date				Date	01–31
05h	0	0	0	10 Month	Month				Month	01–12
06h	10 Year				Year				Year	00–99
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h–3Fh									RAM 56 x 8	00h–FFh

0 = Always reads back as 0.

Figura 5.18 - Registos do RTC

Como se pode verificar pela figura, as respostas do dispositivo têm a codificação BCD. Para ler um determinado registo, apenas temos de enviar uma trama de leitura com o endereço do registo pretendido.

5.3- DOMO_CAN Monitor

Por último foi desenvolvida uma aplicação java, DOMO_CAN Monitor que comunica pela porta série de um computador com o DOMO_GATMONITOR. Esta aplicação é a responsável por criar os ficheiros de configuração e de monitorizar a rede de domótica. Possibilita, assim, a configuração de todos os módulos através dos ficheiros por esta criados.

Esta aplicação contém um menu de apresentação que é mostrado sempre que se inicia a mesma ou, se o utilizador efectuar um “logout”, que serve para bloquear a sua sessão, não permitindo que alguém possa efectuar alterações ou mesmo desconfigurar o sistema. Neste menu apenas é possível efectuar login ou sair da aplicação, pois as outras opções estão bloqueadas. A Figura 5.19 representa este menu de apresentação.

Figura 5.19 - Menu de apresentação inicial do DOMO_CAN Monitor

Para se iniciar as configurações ou mesmo a monitorização, é necessário efectuar um login, que na aplicação vem, por defeito, o utilizador igual a "User" e a palavra-chave igual a "Password". Mas é possível alterar estes parâmetros bem como a porta COM a utilizar para a comunicação.

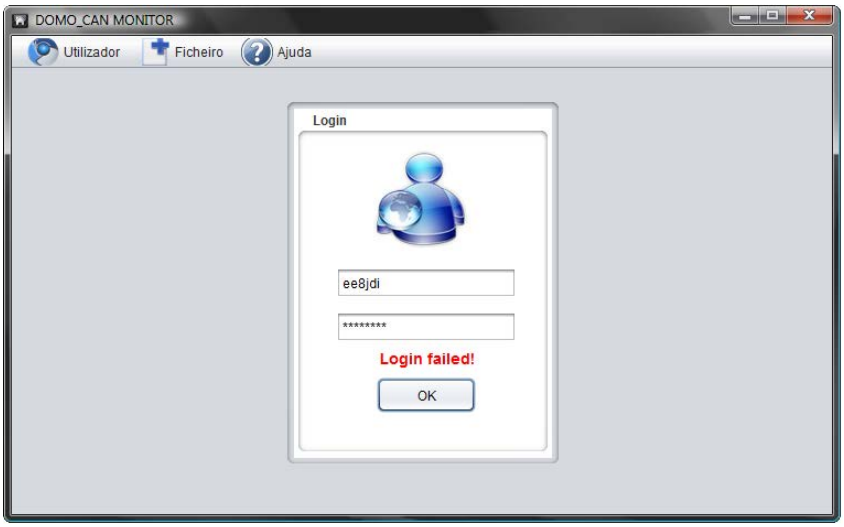

Figura 5.20 - Menu Login do DOMO_CAN Monitor

Uma vez efectuado o login, o utilizador passa a ter acesso a todas as opções da barra de ferramentas apresentada na Figura 5.21.

Figura 5.21 - Barra de ferramentas do DOMO_CAN Monitor

Para a alteração dessas variáveis, ou para as conseguir armazenar, foi utilizada uma *API de Preferences* do java que permite às aplicações a possibilidade de armazenar e recuperar as preferências do utilizador, do sistema e, dados de configuração. Ou seja, permite diferenciar preferências de utilizador de preferências do sistema, permitindo facilmente separar as preferências de um utilizador específico das do sistema (aplicadas a todos os utilizadores).

No menu de preferências podemos facilmente mudar a porta COM a utilizar para comunicação seleccionando o *tab* "Comunicação", ou alterar preferências a nível de utilizador como o nome do utilizador e palavra-chave a utilizar no *tab* "Utilizador". Apesar de estar presente um campo para definir o nome que terão os ficheiros de configuração que são criados pela aplicação ou os que são recebidos do DOMO_GATMONITOR, esta função não está a ser utilizada pois o método utilizado para os nomes dos ficheiros é automático, dependendo da data e hora a que são criados. Assim, se se criar um ficheiro de configuração para um DOMO_CAN, será necessário escolher apenas a directoria pois o nome será construído da seguinte forma, "Config_file_23.05.2010 AS 14.45.txt". Desta forma, permite ao utilizador uma melhor gestão dos ficheiros de configuração.

Figura 5.22 - Menu preferências do DOMO_CAN Monitor

Após estas alterações nestes campos, o utilizador tem a opção de "aplicar" as preferências, que apenas as guarda mas a aplicação continua no menu das preferências, e as opções de "cancelar" ou mesmo "Ok", em que ambas retornam para o menu normal da aplicação, com a diferença que a opção "cancelar" não guarda e a opção "Ok" guarda.

Ao alterar a palavra-chave é necessário saber a palavra-chave antiga. Ao efectuar a opção "Ok" ou a opção "Aplicar", se a palavra-chave antiga introduzida estiver correcta

aparece uma mensagem de alerta a notificar que a palavra-chave foi alterada com sucesso. Por outro lado, se a palavra-chave introduzida estiver incorrecta, não aparecerá nada, o que significa que a palavra-chave não foi alterada ou não houve a tentativa de alterar.

O menu normal da aplicação que aparece após o login efectuado é o apresentado na Figura 5.23, onde é possível escolher entre quatro *tabs* possíveis. O *tab* terminal permite ao utilizador enviar comandos ao DOMO_GATMONITOR, como saber a temperatura, a luminosidade, as horas e a data, efectuar reset ao DOMO_GATMONITOR ou a todos os DOMO_CANs ou mesmo activar e desactivar a função de inspector. Ao activar a função de inspector, como já referido, permite observar as tramas KNX/EIB que circulam no barramento CAN.

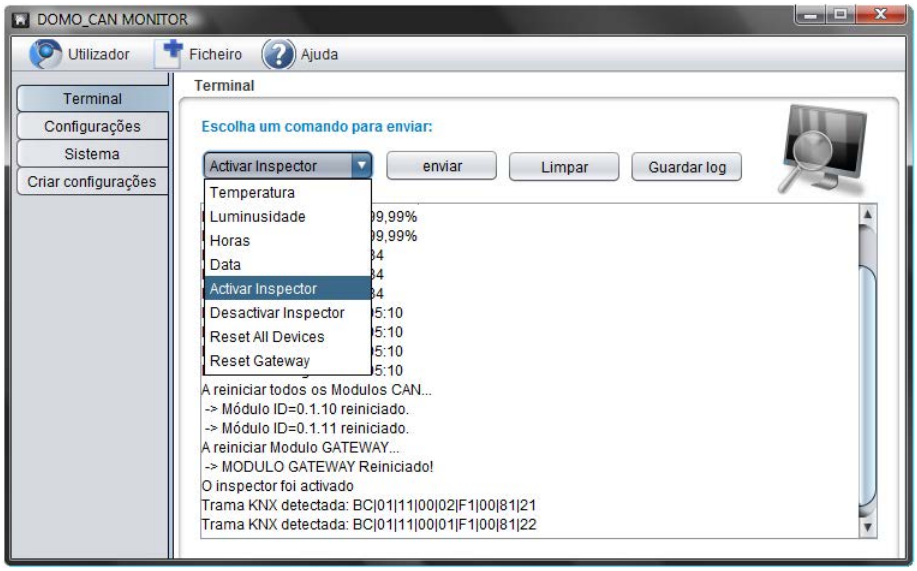

Figura 5.23 - Menu normal – *tab* terminal - do DOMO_CAN Monitor

Neste *tab* terminal é possível guardar um *logfile* (*lists of actions occurred*) dos acontecimentos registados na consola presente neste *tab*, onde é impresso as tramas KNX/EIB, ou as respostas aos comandos, bem como, qualquer alteração do sistema, como por exemplo, a configuração de um módulo ou até um reset de um DOMO_CAN.

Neste menu normal é possível escolher outros *tabs* como o de configurações, o de Sistema ou o de Criar as configurações. No menu configurações, é dada a possibilidade ao utilizador de enviar os ficheiros de configuração ou de receber um ficheiro de configuração. A função de enviar ficheiros permite tanto enviar para o DOMO_GATMONITOR como para qualquer módulo DOMO_CAN presente na rede CAN. Por outro lado, na função de receber apenas foi implementada a função de receber o ficheiro de configurações do DOMO_GATMONITOR.

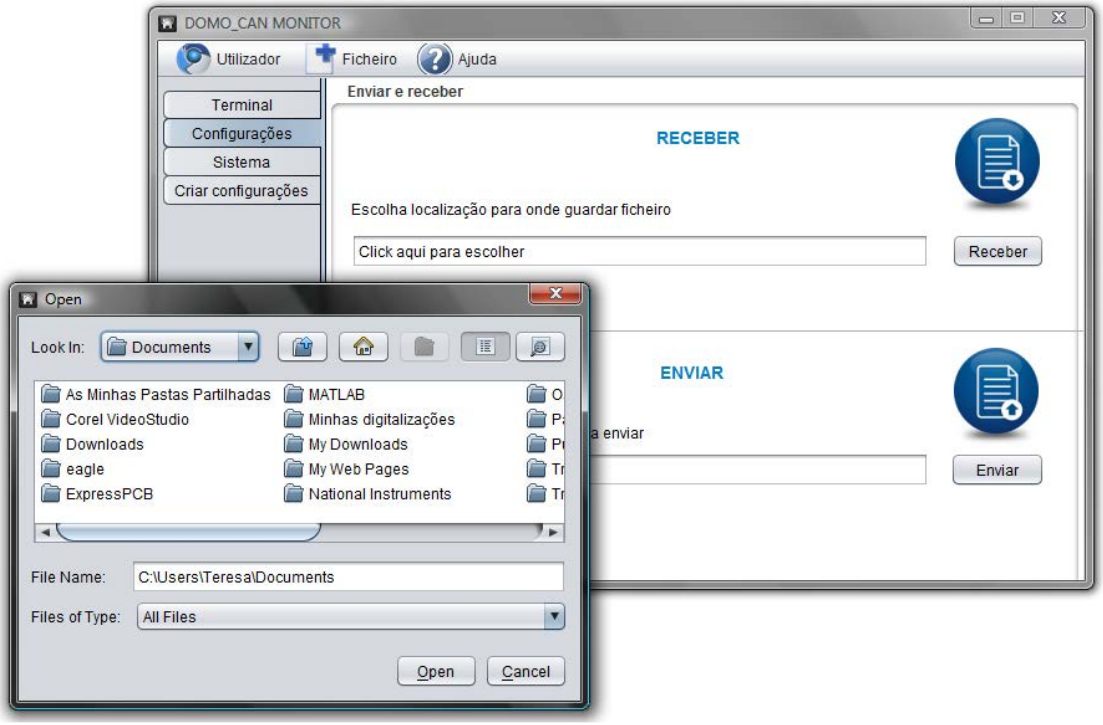

Figura 5.24 - Menu normal – *tab* configurações - do DOMO_CAN Monitor

Para enviar ou receber um ficheiro de configuração o utilizador terá de escolher inicialmente uma directoria onde guardar o ficheiro que pretende enviar. Em caso de não escolher a directoria ou o ficheiro, aparecerá uma mensagem de aviso em forma de notificação, indicando o erro que o utilizador está a cometer. Este tratamento de erros está efectuado para quase todas as situações, como por exemplo, se a ligação já estiver efectuada e o utilizador escolher a opção ligar ou mesmo se a porta definida (COM17) não se encontrar definida. Existe um conjunto vasto de erros que foram sinalizados para que o utilizador tenha conhecimento em caso de efectuar algum. A caixa de mensagem tem o aspecto apresentado na Figura 5.25, que representa a mensagem que aparece, caso o utilizador escolha na barra de ferramentas a opção de "Update". Neste caso, a mensagem não é de erro mas sim de informação.

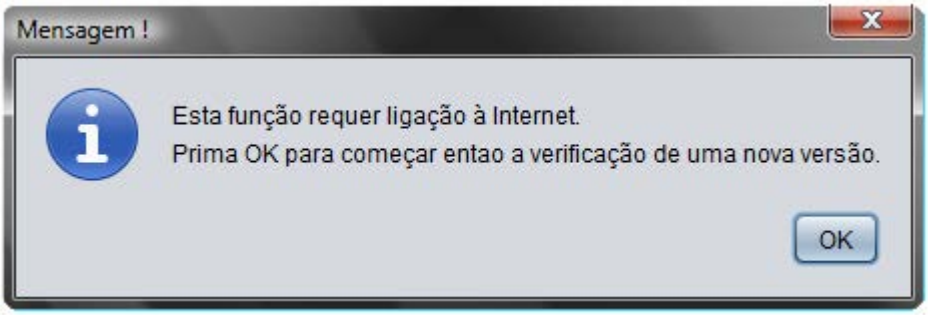

Figura 5.25 - Mensagem de informação da opção de "Update"

Uma das grandes funcionalidades desta aplicação é a possibilidade de criar os ficheiros de configuração. Esta funcionalidade é obtida ao escolher o *tab* "Criar Configurações". Neste *tab* o utilizador pode criar ficheiros de configuração para os módulos DOMO_CAN escolhendo o endereço para o qual se destina o ficheiro. Também é possível criar um ficheiro de configuração para o módulo DOMO_GATMONITOR, no qual o utilizador escolhe, para uma das três variáveis (temperatura, luminosidade e hora/data), um grupo e uma determinada função a efectuar, como ON, OFF, MOVE Up ou MOVE Down.

Figura 5.26 - Menu Criar configurações do DOMO_CAN Monitor

Após criado o ficheiro de configuração e escolhida a directoria onde guardar é possível abrir esse ficheiro utilizando qualquer editor de texto e até efectuar alterações. O aspecto após se criar o ficheiro é apresentado na Figura 5.27.

```
1  ;#################################################################
2  ; Config file Criado em 23.06.2010 AS 10.39 para o Módulo 0.1.16
3  ;#################################################################
4  16,3,1,0
5  3,12,0
6  3,3,0
7  /end
8  ;#################################################################
9  ;#################################################################
10 ;#################################################################
11 ;
```

Figura 5.27 - Exemplo de um ficheiro criado para um DOMO_CAN com o endereço 0.1.16

Estes ficheiros permitem adicionar linhas de comentários, desde que o início da linha comece com ";". O ficheiro de configuração acaba com "/end".

O *tab* sistema não foi concluído no que diz respeito à iteração com o sistema, estando apenas a funcionar toda a interface gráfica. Neste *tab* seria possível escolher qualquer módulo DOMO_CAN presente na rede e efectuar alterações nesse módulo. Uma vez que este *tab* não se encontra a funcionar, se o utilizador o seleccionar, aparecerá uma mensagem com essa informação, indicando ao utilizador que efectue a opção de "Update" para verificar a existência de uma nova versão.

Figura 5.28 - Menu normal - *tab* Sistema (não concluído) - do DOMO_CAN Monitor

Capítulo 6

Conclusão

Esta dissertação teve como objectivo principal desenvolver uma rede de domótica que utilizasse o protocolo KNX/EIB em que o meio físico a utilizar fosse o CAN, devido ao facto de os sistemas KNX/EIB terem valores elevados de preço e a razão desses valores elevados estarem associados à interligação ao meio físico.

Para o desenvolvimento deste sistema recorreu-se ao trabalho Domoitech que foi realizado na Faculdade de Engenharia da Universidade do Porto no âmbito do projecto final de curso em Julho de 2007. O uso deste trabalho deveu-se à sua estrutura definida de um sistema de domótica semelhante ao que era necessário desenvolver, ou seja, ter as camadas superiores do KNX/EIB definidas, tais como a camada de aplicação, transporte e rede onde estas se encontravam a funcionar correctamente [3]. Contudo, foi necessário desenvolver novo hardware para completar a arquitectura da solução proposta. A solução passou por definir dois módulos intitulados de DOMO_CAN e DOMO_GATMONITOR. O primeiro realizava e implementava o protocolo KNX/EIB reutilizando o trabalho realizado no Domoitech, com alterações a nível da camada física, pois neste projecto a camada física escolhida foi o CAN. E o segundo realizava apenas a função de gateway, que permitiu uma gestão e configuração do sistema. O projecto de hardware passou por várias etapas, uma delas consistiu estudo dos componentes a utilizar, dando relevância a factores como: onde seria possível adquirir, o preço e o tamanho associado aos componentes, dado que um dos grandes objectivos deste projecto era obter um preço reduzido e que o sistema pudesse ser montado em caixas eléctricas já existentes numa habitação. Após o desenvolvimento da placa de PCB, foram efectuados testes eléctricos para saber se todos os pontos estavam bem soldados ou se não continham qualquer tipo de defeito associado ao processo de fabrico das PCBs. Uma vez concluída esta etapa, avançou-se para a fase do projecto de software onde o primeiro passo foi utilizar um pequeno segmento de código para apenas testar o sistema no que dizia respeito ao bloco de processamento (PIC18F). Esta etapa foi de pequena duração pois o sistema mostrou estar a funcionar correctamente, fazendo o que tinha sido programado. A

fase seguinte consistiu em implementar a comunicação utilizando o protocolo CAN. O facto de este módulo CAN ser na realidade ECAN não trouxe problemas pois foi configurado para funcionar em modo de compatibilidade para que fosse possível adicionar outras placas ao sistema domótico sem que estas possuíssem módulos ECAN. Após o módulo CAN estar completamente configurado procedeu-se a testes de comunicação, que não tiveram o resultado esperado, pois nada acontecia e não havia comunicação. Para se resolver o problema seguiram-se vários testes, como verificar os níveis de tensão das saídas do transceiver CAN, que tinha os valores correctos de 2,5Volts. Outro teste efectuado foi activar as interrupções associadas ao CAN e verificar se alguma ocorria sempre que se tentava comunicar. Com este teste verificou-se que ocorria sempre uma interrupção de erro de comunicação, que incitou a um novo teste de conectividade entre os pinos CANTX e CANRX do microcontrolador e com os pinos do transceiver e verificou-se que havia uma soldadura que não estava correcta. Após a correcção desta soldadura a comunicação estabeleceu-se sem ocorrência de erros.

Apenas quando a comunicação foi concluída se integrou o meio físico CAN no protocolo KNX/EIB que levantou problemas, tais como, o simples facto de uma trama KNX/EIB ter um tamanho máximo de 23bytes e uma trama CAN apenas ter um campo de dados de tamanho máximo igual a oito bytes. Para resolver o problema foi necessário dividir a trama KNX/EIB em tramas de oito bytes para que estas pudessem ser enviadas por CAN, o que levantou um outro problema, o de voltar a reagrupar as pequenas tramas dando origem à trama correcta KNX/EIB. Numa primeira fase de teste, e como apenas tínhamos dois módulos DOMO_CAN, o método usado foi: sempre que um DOMO_CAN enviava uma trama KNX/EIB, utilizava o seu endereço CAN para enviar a primeira trama e, as restantes, enviava com endereço zero para que estas ganhassem a prioridade em caso de um outro DOMO_CAN tentar aceder ao barramento. Esta forma de resolver era ineficiente pois não garantia que, no período de espera em que um DOMO_CAN preparava a segunda trama CAN com endereço zero para enviar, um outro tentasse enviar a sua trama CAN. Para isso, foi criado um método de recepção de tramas CAN (apenas para tramas que se referissem a partes de uma trama KNX/EIB, pois para as restantes não foi necessário) que utiliza três vectores (com possibilidade de aumentar o numero de vectores facilmente em código) identificados com parâmetros que permitem assim reconstruir as tramas KNX/EIB.

O facto de se ter desenvolvido o DOMO_GATMONITOR que funciona como uma gateway de CAN para RS232, permitiu observar as tramas CAN que eram enviadas para o barramento, o que ajudou numa fase de testes e de desenvolvimento deste método de recepção, assim como de todo o projecto. O sistema foi pensado para ser de instalação fácil e configurável, daí a necessidade de desenvolver uma aplicação java que permitisse implementar funções como a função simples de efectuar reset ao sistema, que nos dá de seguida a informação de quantos módulos DOMO_CAN estão interligados e quais os seus

endereços, ou para saber se existe algum problema com algum dos DOMO_CAN. Esta função foi criada para ajudar na instalação pois ao efectuar um reset manual num DOMO_CAN também fará com que apareça o seu endereço na aplicação, o que é necessário saber para o desenvolvimento dos ficheiros de configuração. De facto, pensou-se fazer uma integração do sistema com o software ETS disponibilizado pela KNX [4]. Esta solução foi descartada devido ao tempo limitado para o desenvolvimento deste projecto. Desta forma, foi escolhida a opção de desenvolver uma aplicação em java, por ser multiplataforma, na qual o utilizador apenas necessita de ter instalado no seu computador uma JVM (*Java Virtual Machine*). No desenvolvimento desta aplicação foi necessário ter o cuidado de, ao utilizar funções que pudessem depender da plataforma, verificar qual o sistema operativo utilizado pelo utilizador.

Numa fase final, foram apenas efectuados testes de funcionamento onde se verificou que o sistema estava a implementar correctamente os serviços e os objectos utilizados no Domoitech, bem como as configurações dos módulos DOMO_CAN e DOMO_GATMONITOR. Para estes testes foram criados vários ficheiros de configuração com a aplicação, bem como manualmente.

6.1- Custo final do projecto

Como um dos objectivos deste trabalho era reduzir o custo de uma implementação de uma rede domótica utilizando o protocolo KNX/EIB, é apresentada na Tabela 6.1 o preço de cada placa desenvolvida neste trabalho.

Tabela 6.1 – Preço final do Projecto

Descrição	Preço total (€)
Placa principal do DOMO_CAN	19
Placas secundárias para o DOMO_CAN (saídas em Triac's)	8
Placas secundárias para o DOMO_CAN (saídas em Relés)	11
Placa DOMO_GATMONITOR (Principal+sensores I^2C)	18+14

É possível observar que o custo alcançado é reduzido em comparação com a solução proposta no projecto Domoitech apresentada na Tabela 1.2 - Custo das placas da solução Domoitech. Em comparação com uma solução utilizando componentes KNX/EIB, na qual, por exemplo, o preço de um acoplador de linha ou área tem um preço de 360 euros [16], esta solução desenvolvida tem um preço extremamente reduzido. Contudo, perde em comparação com a capacidade vasta de interligação de equipamentos, como o ar condicionado, e serviços

possíveis, numa montagem com componentes KNX/EIB. De facto o preço é mais reduzido mas não foram exploradas todas as potencialidades do KNX/EIB.

6.2- Melhoramentos futuros

Como qualquer projecto desenvolvido existe uma quantidade vasta de progressos a serem efectuados com vista ao sistema poder ser comercializado, tais como:

- A simples realização de caixas e de melhores PCBs para os módulos desenvolvidos;
- Desenvolver a capacidade do sistema poder ser configurado através do software ETS;
- Colocar em todos os DOMO_CANs a capacidade de realizar as funções do DOMO_GATMONITOR ou seja de funcionar como gateway, eliminando assim a necessidade do uso do DOMO_GATMONITOR para realizar apenas esta função;
- Implementar um meio físico Wireless utilizando por exemplo o protocolo zigbee para situações em que seja difícil a instalação de cabos;
- Implementar os restantes serviços do KNX/EIB;
- Realizar testes do sistema com dispositivos KNX/EIB de outros fabricantes;
- Certificar o produto junto da Konnex.

Referências

[1] "A casa inteligente". Disponível em http://www.acasainteligente.com/. Acesso em 7 de Novembro de 2009.

[2] MONTEIRO, José, Montagem de um sistema didáctico utilizando a tecnologia Instabus/EIB da Siemens, FEUP, Dezembro 2004.

[3] DOMOITECH, Domoitech – Domótica com protocolo EIB, Duarte Pinto e Diana Palma, FEUP – DEEC. Julho 2007.

[4] "KNX". Disponível em http://www.knx.org/. Acesso em Outubro de 2009.

[5] "FARNELL". Disponível em www.farnell.com. Acesso em Outubro de 2009.

[6] "MICROCHIP". Disponível em www.microchip.com. Acesso em Outubro de 2009.

[7] "A MKTi - Domótica e Iluminação". Disponível em http://www.mkti.pt/. Acesso em Novembro de 2009.

[8] Bosch, Robert GmbH, "CAN Specification version 2.0", 1991

[9] CAN BUS BARRAMENTO CONTROLLER AREA NETWORK "CONCEITUAÇÃO". Disponível em http://www.pcs.usp.br/~laa/Grupos/EEM/CAN_Bus_Parte_2.html. Acesso em 16 de Novembro de 2009.

[10] Göhner, Prof. Dr.-Ing. Dr. h.c. P. Göhner, Theory: CAN-Bus, Universität Stuttgart, Institute of Industrial Automation and Software Engineering, 27/04/2006.

[11] "CAN bus (Controller Area Network), an overview". Disponível em http://www.softing.com/home/en/industrial-automation/products/can-bus/more-can-bus/index.php?navanchor=3010320. Acesso em 16 de Novembro de 2009.

[12] "Embedded Linux/Microcontroller Project". Disponível em http://www.uclinux.org/. Acesso em 17 de Novembro de 2009.

[13] "NXP Operating Systems (RTOS / OS)". Disponível em http://www.standardics.nxp.com/literature/other/microcontrollers/pdf/arm.mcu.rtos.pdf. Acesso em 18 de Outubro de 2009.

[14] "NXP". Disponível em http://www.nxp.com/. Acesso em Dezembro de 2009.

[15] The EIB Association: The EIBA Handbook Series – Release 3.0, Brussels, 1998.

[16] "KNX shop". Disponível em http://www.knxshop.co.uk/. Acesso em 15 de Janeiro de 2010.

[17] "Opternus-components". Disponível em http://www.opternus.com/en/opternus-components.html. Acesso em 16 Janeiro de 2010.

[18] “CAN in Automation”. Disponível em http://www.can-cia.org/. acesso em 29 Maio de 2010.

[19] “I2C Bus”. Disponível em http://jeronimomachado.vilabol.uol.com.br/I2C.htm.A cesso em 4 de Junho de 2010.

[20] KNX HANDBOOK, KNX Specification, Konnex Association, Bruxelas, Fevereiro de 2004.

[21] EIBA Handbook Series, EIB, Volume 3: System Specifications, Parte 7: Interworking, 24 de Março de 1999.