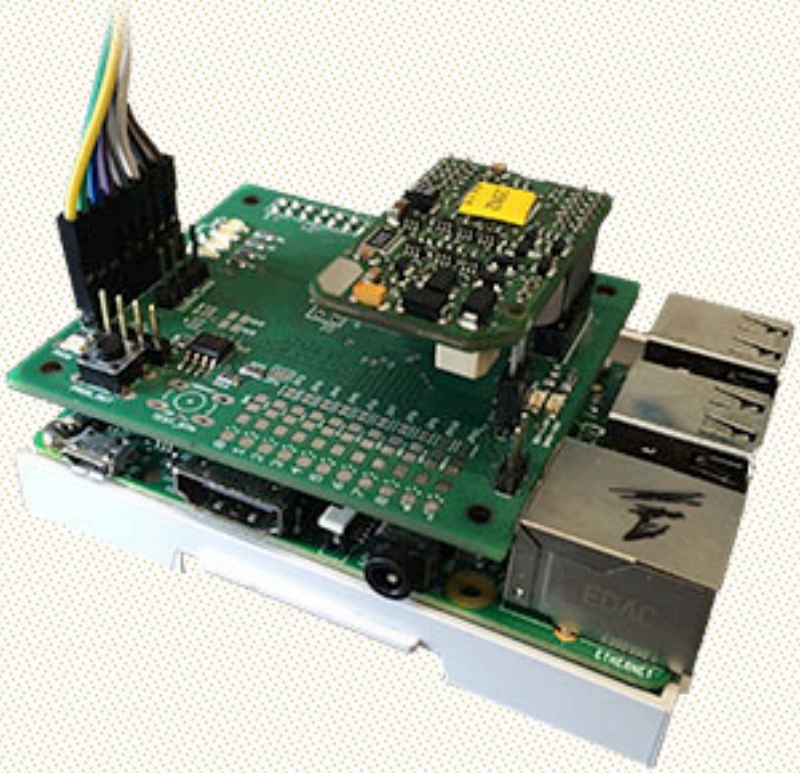

André Alexandre Gaspar da Silva

Desenvolvimento de uma Infraestrutura Eletrónica de Comunicação para o Controlo Remoto de "Casas Inteligentes" Usando KNX

Tese de Mestrado em Engenharia Eletrotécnica e de Computadores
09/2016

· U C ·

Universidade de Coimbra

Desenvolvimento de uma Infraestrutura Eletrónica de Comunicação para o Controlo Remoto de "Casas Inteligentes" Usando KNX

FCTUC

André Alexandre Gaspar da Silva

Orientador: Dr. João Filipe Ferreira

Co-Orientadores: Eng. Vitor Graveto e Eng. Vasco Gouveia

Departamento de Engenharia Eletrotécnica e de Computadores
Universidade de Coimbra

Dissertação submetida para obtenção do grau de
Mestre
Setembro 2016

Agradecimentos

A realização desta dissertação, inserida no plano curricular do Mestrado Integrado de Engenharia Eletrotécnica e de Computadores, só foi possível com o apoio de algumas pessoas a quem eu gostaria de agradecer.

E primeiro lugar, ao meu orientador, Dr. João Filipe Ferreira pela disponibilidade, orientação e confiança depositada na minha pessoa. Aos Co-Orientadores e também sócios gerentes da Wexcedo, Eng. Vitor Graveto e Eng. Vasco Gouveia, bem como ao meu colega Bruno Pereira pela disponibilidade, e ainda pelos recursos e condições de trabalho que me foram facultados para o boa concretização desta dissertação.

Também quero agradecer à *Fablab* de Penela e Coimbra pela cedência dos espaços e dos materiais para o desenvolvimento dos protótipos utilizados no projeto.

Por último, e não menos importante, aos meus pais, irmão e restante família, a oportunidade que me proporcionaram para prosseguir os meus estudos, assim como a motivação e apoio incondicional em todos os momentos.

A todos Muito Obrigado.

Sumário

A domótica tem vindo a aumentar a sua expressão ao longo dos anos. Atualmente, é uma realidade bem presente em muitos dos mais modernos edifícios, permitindo assim aos seus utilizadores um maior conforto, segurança, eficiência energética e ainda dotar as instalações de uma certa inteligência. Hoje em dia, a domótica, permite o controlo de diversas funcionalidades de uma habitação, como iluminação, climatização, segurança, estores, entre outras mais. Com o aumento das utilização da internet e de aplicações móveis surge a necessidade de criar equipamentos que permitam aos seus utilizadores de uma forma segura e simples, interagirem com as suas instalações.

Esta dissertação tem como objetivo colmatar uma lacuna recorrente no que se refere a conexões remotas seguras a instalações domésticas. Sendo o *KNX* uma norma universal e um dos protocolos mais utilizados, foi escolhido para ser a base deste projeto.

Desta forma surgiu o *Alfred*, uma infraestrutura que para além de permitir conexões remotas com segurança, permite interligação de instalações e ainda adicionar lógica condicional aumentado assim as possibilidades de configuração das instalações. O *Alfred* é uma infraestrutura autónoma e que permite aos instaladores uma fácil integração em sistemas novos ou já existentes utilizando uma conexão à internet, seja ela, por cabo ou WIFI. Deste modo, uma das utilizações possíveis do *Alfred* é permitir que, com a autorização do proprietário de uma instalação, um instalador se conecte remotamente e configure ou altere o modo de funcionamento da instalação, sem ter de e deslocar ao local e de forma segura ou mesmo, utilizando uma aplicação móvel, um utilizador possa interagir com a instalação criando uma maior ligação entre o utilizador e a sua habitação. Outra das funcionalidades é a interligação de instalações fazendo com que diversas instalações ajam como uma só, permitindo a passagem das mensagens por todas elas.

Para a concepção desta dissertação foi necessário conhecer a forma de funcionamento da norma *KNX* bem como todos os seus requisitos, uma vez que o *Alfred* terá de ser submetido a testes para poder ser um dispositivo certificado *KNX*. Desta forma foi necessário aprender a utilizar novos *softwares* específicos para o desenvolvimento de dispositivos e aplicações *KNX*, como é o caso do *BIM-Tools, Manufacturer Tool* e o *ETS*.

Uma vez que toda a infraestrutura seria produzida de raiz, foram abrangidas várias matérias

leccionadas no curso, desde criação de *software*, a produção e desenvolvimento de *hardware*, bem como testes de funcionalidade a todos os componentes.

Acrónimos e Abreviaturas

AM Aplication module

AP Application

BbC Backbone coupler

BCU Bus coupling units

BIM Bus Interface Module

DC Direct Current

DHCP Dynamic Host Configuration Protocol

DIP Dual In-Line Package

EEPROM Electrically-Erasable Programmable Read-Only Memory

ETS Engineering Tool Software

FTP File Transfer Protocol

GPIO General Purpose Input/Output

HTTP Hypertext Transfer Protocol

I/O Input/output

I2C Inter-Integrated Circuit

ICSP In Circuit Serial Programming

IDE Integrated Development Enviroment

IP Internet Protocol

ISO International Organization for Standardization

LC Line coupler

LED Light Emitting Diode

OSI Open Systems Interconnection

PCB Printed circuit board

PEI Physical External Interface

PHP Personal Home Page

PWM Pulse-Width Modulation

RAM Random Access Memory

RTC Real Time Clock

SCLK Serial Clock

SCP Secure Copy Protocol

SMD Surface-mount device

SMTP Simple Mail Transfer Protocol

SOC System On Chip

SRAM Static Random Access Memory

SSH Secure Shell

SSID Service Set Identifier

TCP Transmission Control Protocol

TP-UAR Twisted Pair - Universal Asynchronous Receiver/Transmitter

UART Universal Asynchronous Receiver/Transmitter

USB Universal Serial Bus

UDP User Datagram Protocol

UV Ultraviolet

Tabela de Conteúdos

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

1.1.1 Contexto

O mercado das casas inteligentes está em crescimento, como se pode verificar na Fig. 1.1.

No final de 2013, apenas na América do Norte já tinham sido instalados 5,5 milhões de sistemas inteligentes em casas americanas. Até 2019, estima-se que a essa inteligência chegue a 38,2 milhões de casas.

O número de casas inteligentes na Europa e na América do Norte chegou aos 17,9 milhões em 2015

Na Europa, o mercado das casas inteligentes encontra-se num estado mais embrionário, estando sensivelmente 3 anos atrás da América do Norte em termos de penetração e maturidade no mercado. Em 2013 existiam cerca de 1,45 milhões de casas inteligentes na Europa, mas a previsão desse valor para 2019 cresce exponencialmente, atingindo os 29,7 milhões.

68 milhões de casas na Europa e na América do Norte serão inteligentes até 2019

Com a contínua expansão do mercado de casas inteligentes, também cresce a oferta de soluções e protocolos. De entre os vários protocolos podemos destacar os seguintes:

- *X10*
- *Insteon*
- *UPB* (*Universal Powerline Bus*)
- *KNX*
- *ZigBee*
- *Wave*

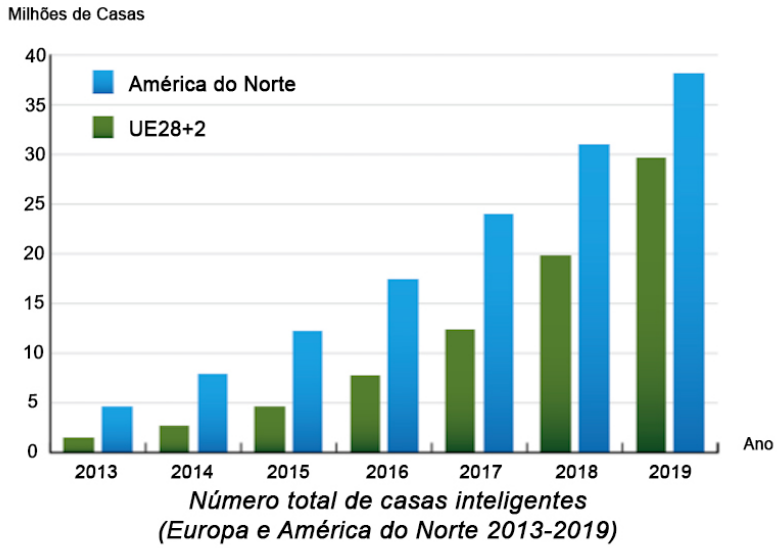

Figura 1.1 Estimativa de crescimento de casas inteligentes de 2013 a 2019 – adaptado de *Smart Homes and Home Automation* [7].

Um protocolo de automatização de casas é uma linguagem de comunicação de hardware que transmite instruções estre dispositivos sejam eles com ou sem fios.

Até recentemente, a única maneira de aceder a casas inteligentes era localmente, mas com os avanços nas tecnologias de acesso à internet, e a existência ubíqua da banda larga mundialmente com uma velocidade média de 4 *Mbit/s* [2] juntamente com a larga difusão de aplicações móveis [10], fazem com que seja possível o acesso remoto a estas instalações.

O *download* de aplicações móveis, em todo o mundo, foi de aproximadamente 2.52 biliões em 2009 e espera-se um enorme aumento para 268.69 biliões em 2017 [11], o que prova o constante crescimento da utilização das tecnologias móveis.

1.1.2 Porquê KNX?

KNX é mais que um protocolo, é uma norma internacional para automatização de edifícios que permite a integração e a programação de uma gama de mais de 7100 dispositivos, de mais de 400 fabricantes, usando apenas o *ETS* (software de gestão e comissionamento de redes *KNX*). [1]

Figura 1.2 *KNX* logo e exemplos de aplicações

Para garantir a qualidade do protocolo, todos os dispositivos são submetidos a rigorosos testes para obter uma certificação. Os dispositivos certificados contêm o logo do *KNX*. Também os integradores de *KNX* têm de frequentar e realizar cursos no "*KNX Certified Training Centre*".

KNX é o único sistema que cumpre os requisitos das normas europeias (*EN50090*) e internacional (*ISO/IEC 14543*). Tal conformidade, confirma a qualidade e o valor da tecnologia *KNX* e serve como símbolo de confiança para os proprietários. O *KNX* é livre, expansível e de fácil utilização.

1.2 Objectivos e contribuições

O principal objetivo deste projeto é o desenvolvimento de uma infraestrutura eletrónica para controlo remoto de casas inteligentes usando *KNX*. Mais especificamente, esta infraestrutura irá permitir o controlo remoto e a conexão de casas fornecendo para tal um canal de ligação entre dispositivos via *TCP-IP – KNX*. Utilizadores finais serão capazes de gerir e configurar as suas casas inteligentes, entre diversas formas, através de uma aplicação móvel ou web. Com o desenvolvimento desta infraestrutura, será possível uma mais fácil manipulação funcional e económica das habitações. A infraestrutura proposta permitirá o controlo de uma vasta gama de dispositivos eletrónicos desde sensores (interruptores, sensor de luz...) a

[1] A norma KNX será apresentada em detalhe na seção 2.2

atuadores. Este projeto foi executado em comum no âmbito do trabalho realizado na empresa WExcedo e para a Dissertação de Mestrado em Engenharia Electrotécnica e de Computadores da Universidade de Coimbra.

A infraestrutura desenvolvida tem como nome ***Alfred***.

Na Fig. 1.3 é possível ter uma visão geral do conceito por detrás do *Alfred*, e da sua integração numa instalação em edifício.

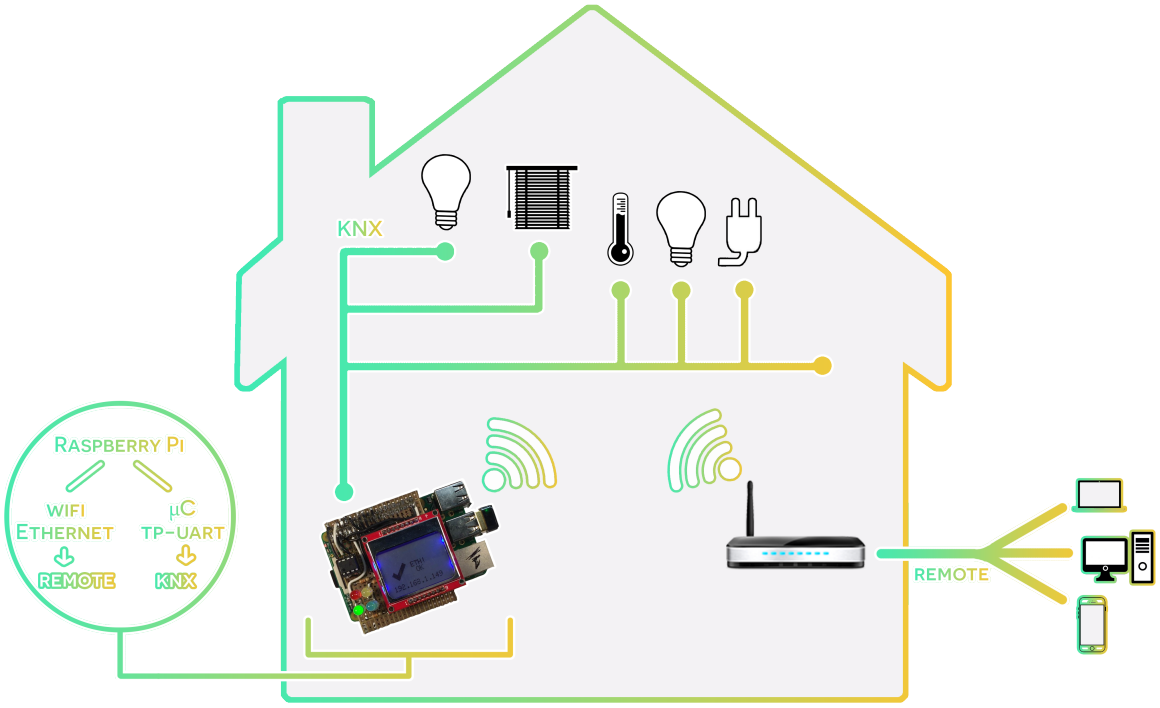

Figura 1.3 Conceito do projeto

A solução proposta oferece as seguintes vantagens em relação aos produtos já existentes no mercado:

- É concebida para funcionar juntamente com todos os dispositivos *KNX*;
- Não necessita de um computador para conectar à internet;
- Não necessita de configurações de *Router/Firewall*;
- Conecta automaticamente aos servidores *Wexcedo*;
- Possui funcionalidades embebidas para facilitar o comissionamento de instalações;
- Permite a utilização do *ETS* (aplicativo de comissionamento *KNX*) remotamente.

Existem produtos no mercado que já implementam estas funcionalidades, mas apenas de forma parcial. Por exemplo, o *Gira Home Server* da *Gira*[2] oferece maneira de aceder a instalações *KNX* através da internet, mas de forma não segura. *OpenRemote*[3] é um software que permite controlar remotamente instalações, mas necessita de configurações ao nível de routeamento de portos no router ou a criação de uma VPN, o que faz com que seja inseguro também. Ainda há outros como o *Amazon Echo*[4], *NEST*[5] ou *Apple HomeKit*[6] que são soluções fechadas e proprietárias para controlar edifícios remotamente, mas são muito caros e permitem pouca customização.

1.3 Estrutura da Dissertação

A presente dissertação encontra-se dividida em 4 capítulos. O primeiro capítulo abrange a introdução ao tema. No segundo capítulo consta a informação sobre o enquadramento no tema, nomeadamente a descrição da norma *KNX* e métodos utilizados no desenvolvimento da dissertação, ou seja, o *hardware* utilizado. No terceiro capítulo é descrita toda a implementação relativa ao projeto com informação detalhada de cada processo. Por fim, o quarto capítulo diz respeito às conclusões atingidas com a realização da dissertação e são apresentadas sugestões para trabalho futuro.

[2]http://www.gira.com/en/gebaeudetechnik/systeme/knx-eib_system/knx-produkte/server/homeserver.html
[3]http://www.openremote.org/display/HOME/OpenRemote
[4]http://www.amazon.com/Amazon-SK705DI-Echo/dp/B00X4WHP5E
[5]https://nest.com
[6]http://www.apple.com/ios/homekit/

Capítulo 2

Enquadramento e Métodos

Neste capítulo será explorado o tema da domótica. Uma vez que o *KNX* foi a tecnologia escolhida para a realização da tese, será feita uma análise mais profunda de modo a garantir que uma melhor percepção do tema.

2.1 Domótica

Domótica não é um conceito recente. As primeiras referências à domótica eram como sendo ficção cientifica. Mas acabou por se tornar realidade com o aparecimento do microcontrolador em 1970 [5].
O termo "domótica"resulta da junção da palavra latina "*domus*"(casa) com "robótica"(ligado ao ato de automatizar).
Com o aparecimento da domótica, pretendia-se controlar a iluminação, climatização e a segurança das habitações bem como a interligação destes elementos. No ano de 1975, foi desenvolvida, pela *Pico Electronics*, a primeira tecnologia de automação doméstica, o *X10* [9].
Atualmente os protocolos de comunicação aumentaram significativamente. Alguns dos protocolos mais vulgares são os seguintes : *X10*, *EIB/KNX*, *DALI*, *PLC/BUS*, *Konnex*, *Insteon*, *Zigbee* e *Zwave*. Tal como os protocolos de comunicação, também as aplicações de domótica aumentaram.

2.2 A norma KNX

2.2.1 História e Aplicações

O *KNX* resulta da convergência de 3 *standards* Europeus de Automação Predial e Residencial: *EIB*, *BatiBus* e *EHS*.

Em Maio de 1999 os membros das referidas associações juntaram-se para criar um novo sistema que garantisse um processo de convergência, de evolução e continuidade crescente no mercado internacional dando assim inicio ao *KNX*.
O *KNX* é o único *standard* mundial *ISO/IEC 14543* com protocolo aberto para a automação predial e residencial, habitualmente designada domótica.

A principal aplicação do *KNX* é em residências mas também pode ser aplicado em diversos tipos de edifícios, tais como espaços comerciais, bancos, hotéis, escolas e hospitais.

Uma das grandes vantagens do *KNX*, é o facto de ser um sistema distribuído e descentralizado, ou seja, caso falhe um dispositivo, o sistema mantém-se operacional, garantindo assim a gestão de diferentes instalações técnicas num edifício, com grande fiabilidade. Outra grande vantagem é também o facto de os equipamentos virem programados de fábrica e os integradores (técnicos que fazem a colocação em serviço da instalação) apenas realizarem alterações dos parâmetros de funcionamento intrínseco de cada equipamento e os parâmetros de comunicação. A comunicação é efetuada através de um único barramento e com a possibilidade de vários meios físicos.

Esta fiabilidade tem assegurando nos últimos 25 anos uma elevada continuidade de serviço.

A nível mundial existem mais de 400 fabricantes em 39 países que fabricam os mais de 7100 produtos *KNX* [4].

Mesmo sendo fabricantes diferentes, uma vez que os dispositivos respeitam a norma *KNX*, eles são capazes de comunicar entre si sem qualquer problema. Desta forma, um dono de uma instalação *KNX* tem total liberdade para procurar produtos de marcas diferentes de forma a corresponder aos seus requisitos tanto funcionais como financeiros.

Com a vasta gama de produtos certificados pela *KNX*, existem hoje em dia soluções para todo tipo de aplicações, desde o controlo de iluminação, controlo de persianas e estores, sistemas de segurança contra intrusão, controlo de sistemas de aquecimento, sistemas de rega, ventilação e ar condicionado, incêndio, fugas de gás, contagem e gestão de energia, entre muitas outras aplicações.

Uma das grandes vantagens do *KNX* é a simplicidade de instalação, mas a sua instalação, configuração, parametrização e colocação em serviço deve ser realizada por técnicos capacitados para o seu correto funcionamento, chamados de Instaladores. Estes técnicos são formados por centros de formação credenciados pela *KNX* e submetidos a um exame que lhes atribui a credenciação *KNX Partner* e o direito de estarem presentes numa lista disponibilizada pela *KNX* no seu site[1] de forma a que seja possível um dono de uma instalação escolher um Instalador credenciado.

Também para a parametrização dos equipamentos existe a grande vantagem de ser realizada por um software único (comercializado pela *KNX*) chamado *ETS*.

2.2.2 Arquitetura

A arquitetura KNX como adaptação da OSI da ISO

A Organização Internacional para a Normalização (*ISO*) decidiu criar uma arquitetura *standard* para redes de computadores. Os principais objetivos eram: [OSI]

- a criação de um modelo *standard* de comunicação de dados pelos quais os sistemas pudessem comunicar
- obter compatibilidades entre sistemas sem a necessidade de implementar o modelo completo em cada rede
- obter uma conexão aberta *(OSI)* entre todos os utilizadores de uma rede

Na Fig. 2.1, está ilustrado a estrutura do modelo *OSI*. A norma *KNX* baseia-se neste mesmo modelo, apenas não utilizando as camadas 5 e 6, correspondendo à camada de Sessão e Apresentação.

Camada de Aplicação: A camada de aplicação é onde residem as aplicações de rede e seus protocolos. A camada de aplicação da Internet inclui muitos protocolos, como o *HTTP* (que disponibiliza a solicitação de documentos *web*), *SMTP* (que disponibiliza a transferência de mensagens de correio electrónico) e *FTP* (que disponibiliza a transferência de ficheiros entre dois sistemas).

Camada de Transporte: Tal como o nome indica, a camada de transporte da Internet transporta mensagens. Na Internet existem dois protocolos de transporte, *TCP* e *UDP*,

[1] https://www.knx.org/knx-en/community/partners/list/

ambos podem transportar mensagens de camada de aplicação. O *TCP* fornece um serviço de conexão orientada, garantindo assim a entrega de mensagens da camada de aplicação para o destino. O *TCP* também parte mensagens longas em segmentos mais curtos e fornece um mecanismo de controlo de congestionamento. O protocolo *UDP* não oferece confiabilidade pois não contém controlo de fluxo, nem de congestionamento.

Camada de Rede: A camada de rede da Internet é responsável por mover pacotes da camada de rede, conhecidos como *datagramas*, de um *host* para outro.

Camada de Ligação de Dados: Esta camada rotas encaminha datagramas através de uma série de *routers* entre a origem e o destino.

Camada de Física: Enquanto o trabalho da camada de ligação é para mover *frames* inteiras a partir de um elemento de rede para um elemento adjacente, o trabalho da camada física é mover os *bits* individuais dentro da estrutura de um nó para o próximo. Existem muitos protocolos de camada física: um para o fio de cobre de par trançado, outro para o cabo coaxial e outro de fibra-ótica. [6]

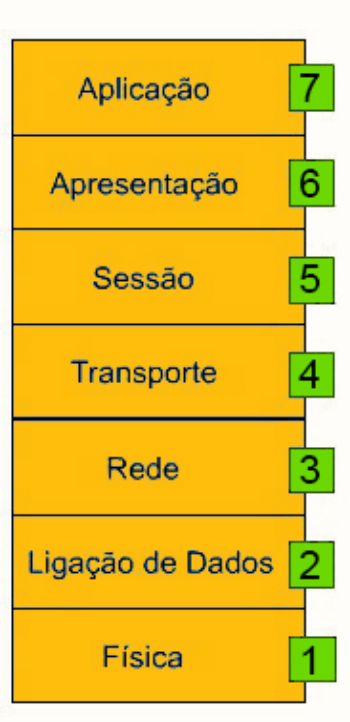

Figura 2.1 Estrutura do modelo *OSI*

Componentes KNX

Todos os dispositivos de uma instalação *KNX* se encontram ligados independentemente do meio-físico.

- Sensores (ex.: botões de pressão, sensores de temperatura, sensores de movimento)
- Atuadores (ex.: electroválvulas, relés, *displays*)
- Componentes do sistema *KNX* (ex.: *Line-Couplers*, *Backbone-Couplers*)

Conexões (Camada Física)

A norma *KNX* define vários meios físicos de comunicação:

- Cabo trançado (Twisted-Pair – herdado dos standards *BatiBUS* e *EIB Instabus*)
- *Powerline* (herdado dos standards *EIB* e *EHS*)
- *UPB* (Universal *Powerline Bus*)
- Rádio Frequência (*KNX-RF*)
- Ethernet (também conhecido como *EIBnet/IP* ou *KNXnet/IP*)

Topologia de instalações KNX

Uma instalação *KNX* pode ser dividida em linhas e áreas (ver Fig. 2.2):

- Uma **linha** é a unidade de instalação mais pequena. Cada linha comporta no máximo **64 dispositivos**.
- Uma **área** consiste num máximo de **15 linhas** conectadas à linha principal atravás de *Line Couplers (LC)*
- **Várias Áreas** podem ser obtidas com *Backbone Couplers (BbC)* juntando uma área à linha principal, até **15 áreas**.

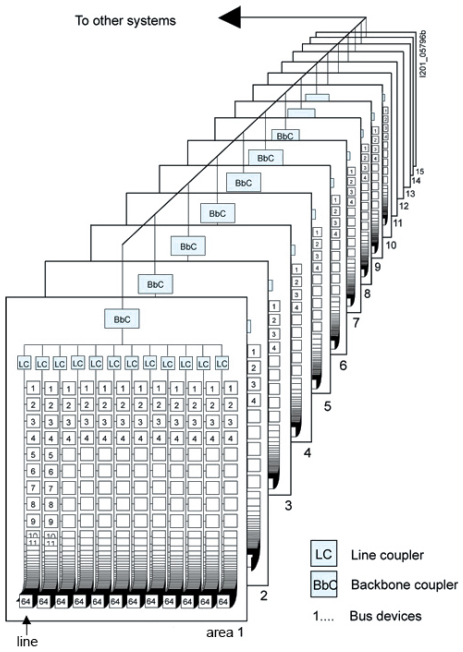

Figura 2.2 Topologia de uma instalação *KNX*

2.2.3 Estrutura do telegrama KNX

Todos os dispositivos num barramento *KNX* podem trocar mensagens entre eles usando telegramas. Um telegrama é dividido nos campos exibidos na Fig. 2.3.

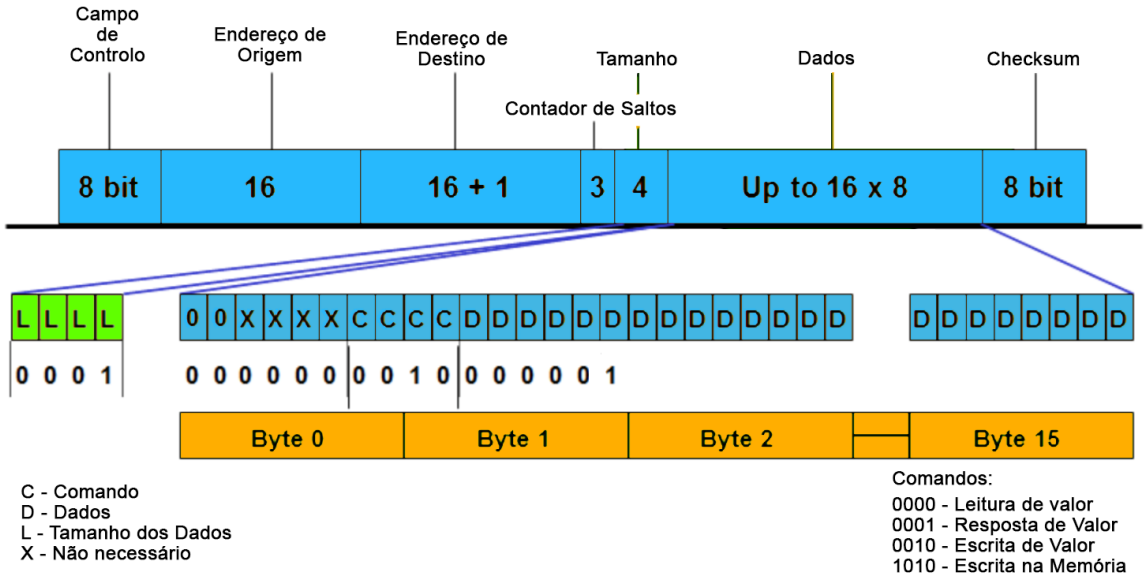

Figura 2.3 Estrutura de um telegrama *KNX*

2.2.4 Objetos de Grupo

Objetos de Grupo ou Objetos de Comunicação são localizações de memória nos dispositivos. Cada dispositivo pode ter um variado número de objetos consoante os parâmetros definidos na aplicação (ex.: um atuador de 6 saídas pode ser configurado para apenas usar uma saída, e desse modo, fica apenas com um objeto de entrada, que ao ser ativado faz atuar a sua saída). O tamanho dos objetos pode variar entre 1 *bit* e 14 *bytes* consoante a sua função. No caso de objetos de 1 bit, estes apenas têm dois estados (0 e 1). Usando Endereços de Grupo é possível juntar objetos entre dispositivos mas apenas objetos com o mesmo tamanho, ou seja, objetos de 1 *bit* apenas podem ser agrupados com objetos de 1 *bit*. Um objeto pode ser associado a vários Endereços de Grupo.

Na Fig. 2.4 encontra-se um exemplo usando um sensor (interruptor de duas teclas) e um atuador de duas saídas. É possível verificar que o Endereço Individual do sensor é o 1.1.1 e do atuador é o 1.1.2. É também de notar que ambos têm 2 Objetos de Grupo (Nr.0 e Nr.1) mas podia-se dar o caso de um ter mais que o outro. Ao Objecto de Grupo Nr.0 do sensor foi associado ao Endereço de Grupo 1/1/1 e uma vez que o Objecto de Grupo Nr.0 do atuador também pertence ao Endereço de Grupo 1/1/1, sempre que o *Left Rocker* for ativado, irá ativar o *Channel A* do actuador porque estão "ligados"pelo mesmo Endereço de Grupo. O valor de um objecto é enviado para o barramento *KNX* da seguinte forma:

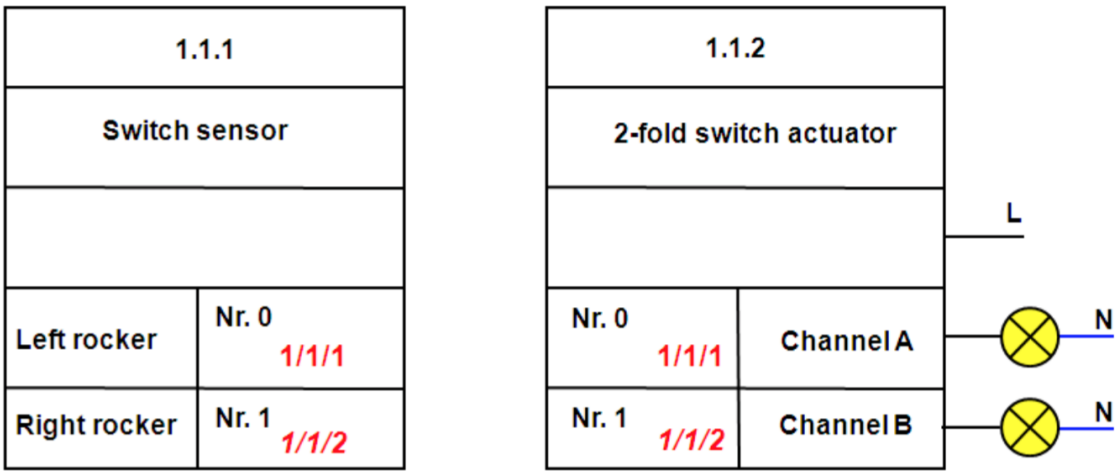

Figura 2.4 Exemplo de Objectos de Grupo e Endereços de Grupo

- Se o *Left Rocker* for pressionado, o sensor escreve "1"no seu Objecto de Grupo número 0. Este dispositivo transmite um telegrama com a informação "Endereço de Grupo 1/1/1, escreve valor, 1".

- Todos os dispositivos no barramento *KNX* que tenham um Endereço de Grupo 1/1/1, irão escrever "1"no seu Objecto de Grupo
- No exemplo, o "1"é escrito no Objecto 0
- A aplicação do atuador detecta que o valor do seu Objecto de Grupo foi alterado e executa a função no *Channel A*.

2.2.5 Endereço Individual

Um Endereço Físico, ou Endereço Individual, tal como o nome indica, tem de ser único numa instalação *KNX*. Este encontra-se dividido em 3 partes, os 4 *bits* mais significativos indicam a área, os seguintes 4 *bits*, a linha e o último *byte* indica o dispositivo. Premindo o Botão de Programação, presente em todos os dispositivos *KNX*, este fica preparado para receber o seu Endereço Individual através do software de configuração do *KNX*. O LED mantém-se aceso até à conclusão do processo de atribuição.

Exemplo de um Endereço Individual: 15.15.255 (pontos entre os números) Este endereço em específico é especial uma vez que é o atribuído a dispositivos novos (sem endereço atribuído).[2]

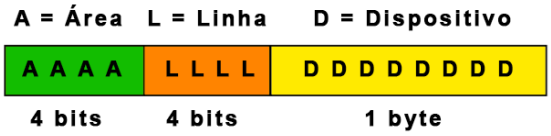

Figura 2.5 Estrutura de um Endereço Individual de dispositivos *KNX*

2.2.6 Endereços de Grupo

A comunicação entre dispositivos numa instalação *KNX* é feita através dos Endereços de Grupo. Serve como uma espécie de fio virtual entre Objetos de Grupo de dispositivos. Estes endereços são atribuídos no software de configuração do *KNX* e são compostos por 2 *bytes* podendo ter as seguintes configurações de níveis:

[2] **Importante:** O Endereço Individual não tem qualquer importância durante a operação normal da instalação. Só é necessário para poder carregar as definições e a aplicação.

- Livre - a atribuição dos 16 *bits* é livre podendo o endereço ir de 0 a 65535
- 2 níveis - o endereço é dividido em 2 partes, os primeiros 5 *bits* indicam o grupo principal (de 0 a 31) e os restantes 11 *bits* indicam o subgrupo (de 0 a 2047)
- 3 níveis - o endereço é dividido em 3 partes, os primeiros 5 *bits* indicam o grupo principal (de 0 a 31), os seguintes 3 *bits* indicam o grupo intermédio (de 0 a 7) e os restantes 8 *bits* indicam o subgrupo (de 0 a 255)

As configurações de níveis podem ser mudadas nas propriedades de cada projeto. O Endereço de Grupo 0/0/0 é reservado para mensagens de *broadcast* (mensagens para todos os dispositivos do barramento *KNX*).

Exemplo de um Endereço de Grupo: 15/15/255 (barras entre os números)

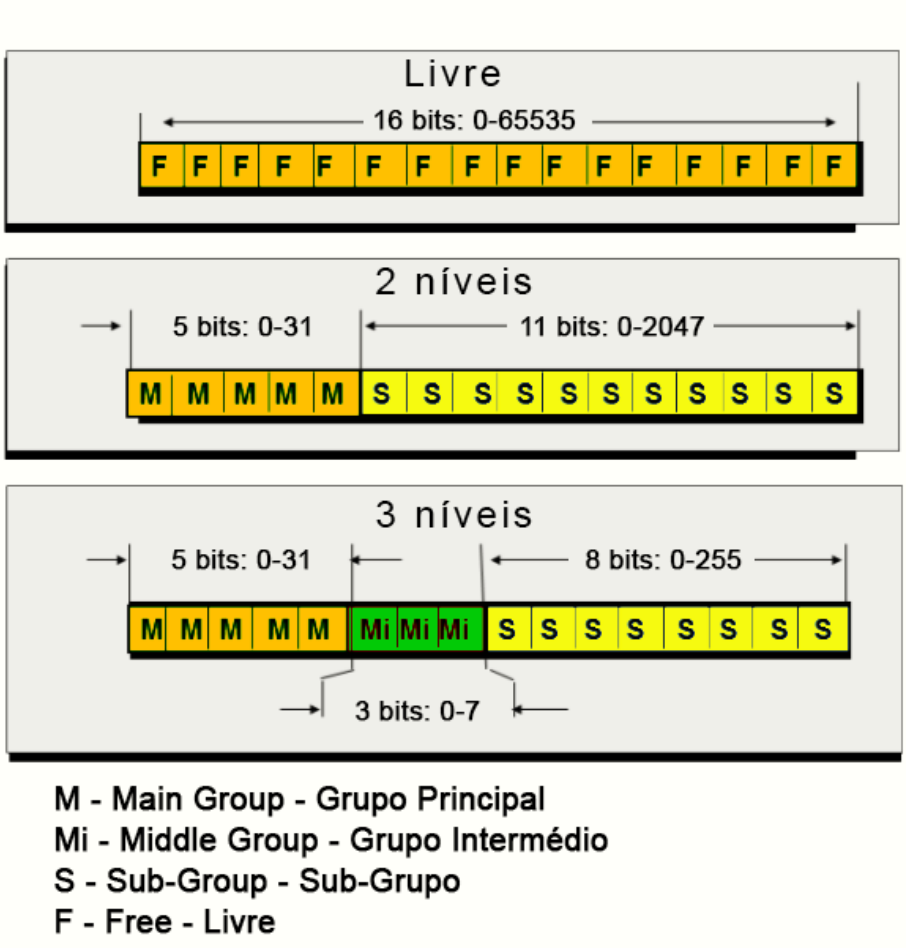

Figura 2.6 Estrutura de um Endereços de Grupo de dispositivos *KNX*

O instalador encarregado pelo projeto pode decidir a forma como os níveis são utilizados. Ex.:

- Grupo principal = piso
- Grupo intermédio = tipo de função
- Subgrupo = função na instalação (ex.: cozinha luz *on/off*, quarto janela *on/off*...).

Os atuadores podem estar à escuta de vários Endereços de Grupo. Já os sensores apenas podem enviar um Endereço de Grupo por telegrama. Os Endereços de Grupo são atribuídos aos Objetos de Grupo do respectivo sensor ou atuador com a ajuda do software de configuração do *KNX* (*S-mode*) ou automaticamente e invisível ao utilizador (*E-mode* ou *A-mode*).

2.2.7 ETS

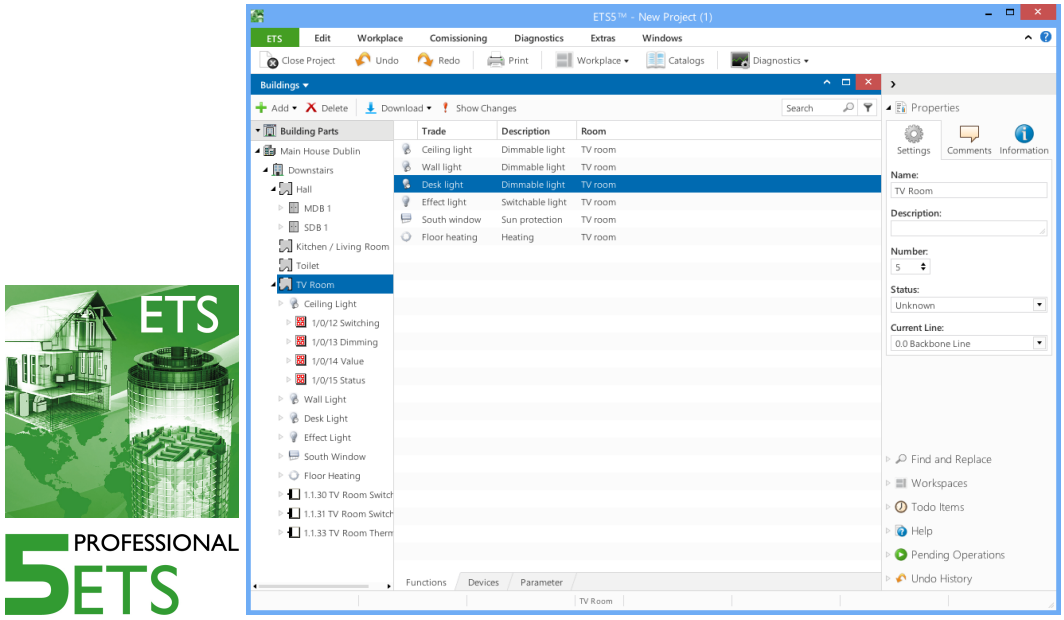

Figura 2.7 Logótipo da aplicação *ETS* e Aspecto gráfico da aplicação

A programação e o comissionamento de instalações *KNX* é feita a partir de um software para *Windows*, o *ETS* (Engineering Tool Software), criado pela *KONNEX* e atualmente na versão 5.5.2. O *ETS* contém uma base de dados de produtos chamada de Catálogo onde é possível encontrar a aplicação correspondente a cada produto.

Para a criação de uma instalação *KNX* são necessários os seguintes passos:

- Atribuir um Endereço Individual a cada dispositivo (necessita de ter o LED de Programação ligado)
- Importar Catálogos e escolher a Aplicação correspondente para cada dispositivo
- Configurar a Aplicação caso seja configurável
- Criar e atribuir Endereços de Grupo (lâmpadas, estores) aos dispositivos
- Fazer *Download* da informação

O *ETS* inclui também a capacidade de monitorizar o barramento para resolver problemas.

2.2.8 Modos de Programação de Dispositivos

A-Mode

A-Mode (Automatic Mode) corresponde a um modo de operação em que os dispositivos se configuram automaticamente permitindo assim que o utilizador os possa instalar. A ideia deste modo de configuração é o dispositivo detetar automaticamente o seu parceiro e estabelecer a conexão entre ambos. Este modo é adequado a pequenos projetos onde o tipo de dispositivo apenas exista uma vez na instalação. Devido ao tipo de configuração, cada dispositivo requer uma certa inteligência para efetuar a instalação.

E-Mode

E-Mode (Easy Mode) corresponde a um modo de operação em que os dispositivos são pré-programados e requerem algum conhecimento básico para instalar. Alguns parâmetros necessitam de ser configurados para satisfazer os requisitos do utilizador. Normalmente, dispositivos deste modo de configuração, são vendidos já com a Aplicação carregada. Este modo de configuração é adequado para pequenos projetos e faz com que o *KNX* seja acessível a instaladores com menos conhecimento.

S-Mode

S-mode (System Mode) corresponde a um modo de operação em que os dispositivos têm de ser programados e instalados por técnicos especializados. Todos os passos de configuração têm de ser efectuados manualmente pelo instalador, requerendo assim instaladores especializados, como é o caso do *Alfred*.

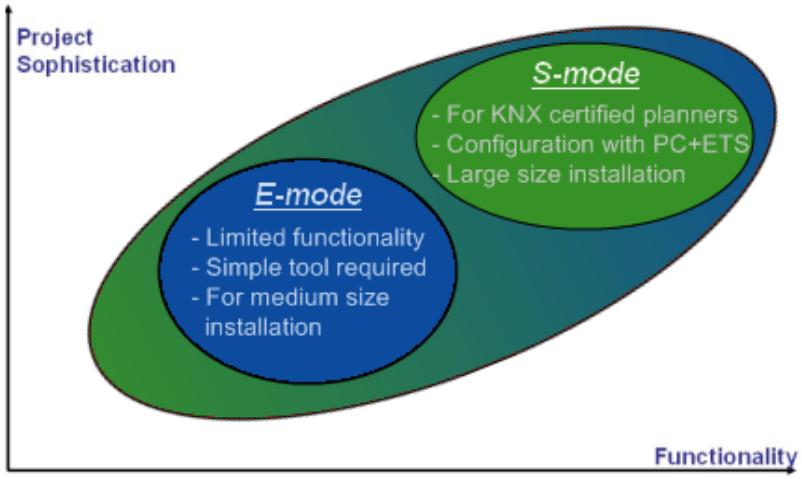

Figura 2.8 Diferenças entre o *E-Mode* e o *S-Mode*

2.3 Hardware de Desenvolvimento

2.3.1 Raspberry Pi

O *Alfred* tem por base um *Raspberry Pi 2* modelo B+. O *Raspberry Pi* é um pequeno computador baseado em um *System on a Chip* (*SoC*). Tem um processador *ARM Cortex-A7 Quad-Core* 900MHz, 1GB de memória RAM e para além de portas USB e porta Ethernet, tem 40 *GPIO's* que faz com que o computador possa interagir com o mundo exterior. A escolha do *Raspberry Pi* deveu-se ao facto de ser necessário correr *Linux* uma vez que teria de ser possível estabelecer uma ligação à internet, seja por cabo ou por *WIFI*, teria de ser possível a criação de um *Hotspot* bem como alojar uma página web, teria de disponibilizar *GPIO's* para utilizar LED's e conter portas *UART* e I^2C.

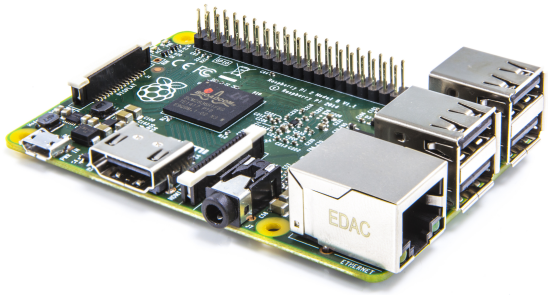

Figura 2.9 Raspberry Pi 2 B+

2.3.2 Arduino Uno

O *Arduino Uno* é placa fabricada na Itália utilizada como plataforma de prototipagem eletrónica a um preço acessível a todos. Tem como base um microcontrolador de 8 bits, o *ATmega328* que pertence à família *AVR* da *Atmel*. Tem 14 pinos digitais de *I/O* (do pino 0 ao 13), em que os pinos 3, 5, 6, 9, 10 e 11 podem servir de saída *PWM*, tem 6 entradas analógicas (pino A0 a A5), um *clock* de 16 MHz graças a um cristal externo, uma conexão USB para alimentação e comunicação, um conector de alimentação externa, um conector *ICSP*, e um botão de *reset*. Para carregar código ou gravar um novo *bootloader* basta conectá-lo a um computador através um cabo USB.

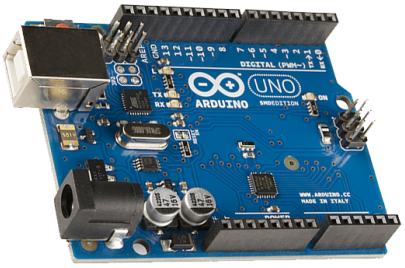

Figura 2.10 Arduino Uno

Características

- Microcontrolador: *Atmega328*
- Tensão de operação: 5V
- Tensão de entrada (recomendada): 7-12V
- Tensão de entrada (limite): 6-20V
- Entradas/Saídas: 14 (6 pinos *PWM*)
- Pinos Analógicos (Entrada): 6
- Memória *Flash*: 32 KB (*Atmega328*), 500 bytes usado pelo *bootloader*
- *SRAM*: 2 KB (*Atmega328*)
- *EEPROM*: 1 KB (*Atmega328*)
- Velocidade do *Clock*: 16 MHz

IDE

O *Arduino IDE* é uma aplicação multiplataforma escrita em *Java*. Atualmente encontra-se na versão 1.6.11 e é possível descarregar gratuitamente através do site oficial do *Arduino* [3]. É possível criar código para o *Arduino* no seu editor de código. Este contém recursos como realce de sintaxe e parênteses correspondentes. Com apenas um clique é possível compilar e carregar programas para a placa. O código é criado em *C/C++*, permitindo assim criar com facilidade muitas operações de entrada e saída, tendo que definir apenas duas funções no pedido para fazer um programa funcional:

setup() – Função que corre apenas uma vez no arranque do microcontrolador e onde podem ser definidas configurações de arranque do código.

loop() – Função que corre em ciclo infinito após o *setup()* até que o microcontrolador seja desligado da corrente.

Figura 2.11 Arduino IDE

[3] www.arduino.cc

2.3.3 O interface com o KNX – BIM

O processo de interligação do *Alfred* com o *KNX* teve inicio com o estudo da melhor abordagem quanto ao *hardware* a utilizar, uma vez que essa escolha tinha influência no custo, tempo de desenvolvimento e produção do projeto. O meio físico de comunicação definido para o *Alfred* foi o cabo trançado uma vez que é o mais comum em instalações já existentes, o que vai permitir uma maior abrangência de mercado. Desta forma o *hardware* necessário para comunicação via cabo trançado é um *TP-UART* e um microcontrolador que contenha a *stack* (*software* de sistema para dispositivos *KNX*) do protocolo.

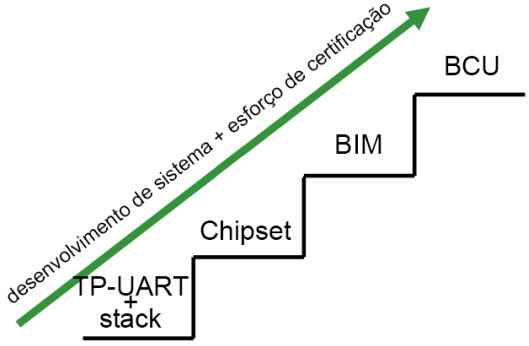

Figura 2.12 Esforço de certificação

Para obter a certificação para um dispositivo *KNX*, é necessário tanto o *software* como *hardware* o sejam. Desta forma, existem diversas abordagens possíveis para o desenvolvimento de um dispositivo, conforme ilustrado na Fig. 2.12. Para um menor custo de certificação, a opção é utilizar dispositivos previamente certificados, mas que têm um custo de aquisição mais alto, como é o caso do *BCU* e do *BIM*. Desta forma, optou-se pelo *BIM* uma vez que pretendíamos colocá-lo dentro do *Alfred*.

TP-UART

O *TP-UART* é um integrado que contém acopulamento físico entre o *KNX* (a componente fisica do *KNX* neste caso é o *Twisted Pair - TP*) e uma comunicação *UART* [8]. Desta forma, utilizando o *TP-UART*, evita-se que o microcontrolador tenha que fazer a codificação e descodificação dos *bits*, pois o *TP-UART* envia as os telegramas já completos através de *UART* para o microcontrolador.

UART abreviatura de *Universal Asynchronous Receiver/Transmitter* consiste numa comunicação série assíncrona.

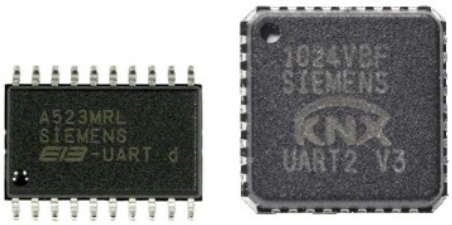

Figura 2.13 *TP-UART* à esquerda e *TP-UART2* à direita

Para o desenvolvimento de *hardware KNX* existem soluções já previamente certificadas facilitando o desenvolvimento e custos de certificação. Como tal, existe o *BCU (Bus Coupling Unit)* e o *BIM (Bus Interface Module)* da *Siemens*. Existem no entanto outras soluções de outras marcas, como a *Tapko* e a *Weinzierl*, mas devido aos custos, foram colocadas de parte.

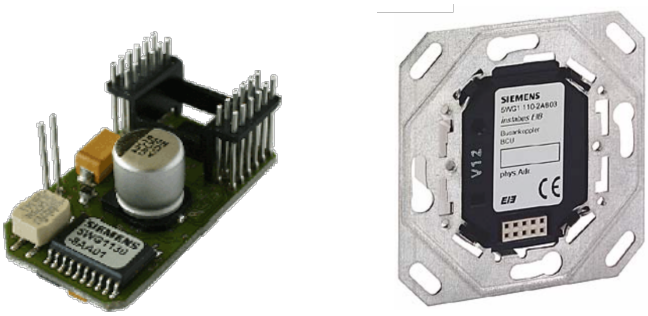

Figura 2.14 *BIM* à esquerda e *BCU* à direita

O interior de um BCU é igual a um *BIM* mas apenas disponibiliza para o exterior um conjunto normalizado de pinos, chamado de *PEI* (*Phisical External Interface*) através dos quais é possível colocar o dispositivo desenvolvido, enquanto o *BIM* disponibiliza todos os pinos. A diferença entre o *BIM* e o *BCU* é o facto do *BCU* ser um produto pronto a usar bastando juntar o *AM (Application Module)* enquanto o *BIM* serve para colocar dentro de um dispositivo.

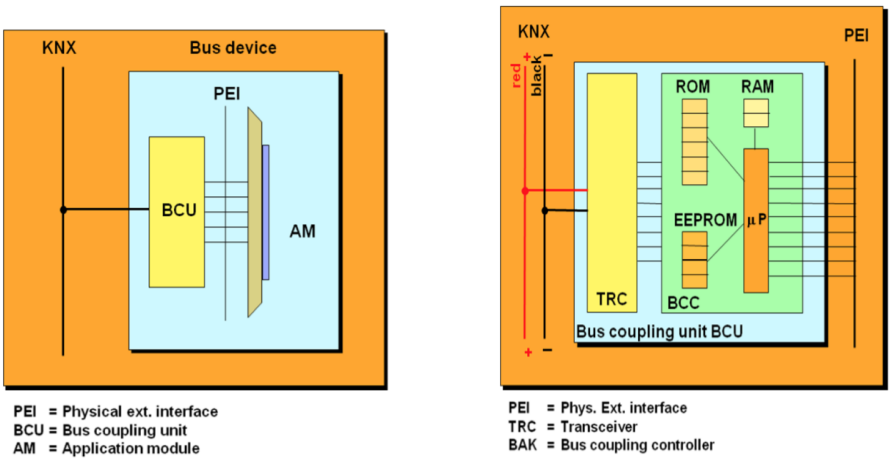

Figura 2.15 Estrutura de um dispositivo *KNX*

O *BIM* é composto por um *TP-UART* e um microcontrolador que implemente a *stack KNX*.

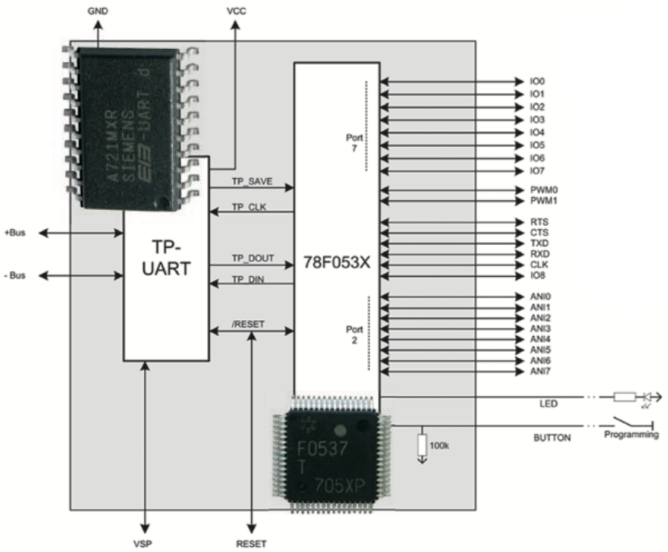

Figura 2.16 Estrutura de um *BIM*

Existe um LED e um botão de programação que permitem colocar o dispositivo em modo programação para, por exemplo, carregar o seu Endereço Individual.

PEI

O *PEI* é composto por 10 ou 12 pinos em que:

- 0 V (pino 1 e 10), +5 V DC (pino 5) e +24 V DC (pino 8)
- *PEI-type* (pino 6)
- I/O 1, 2, 3, 4 e 5 (pinos 3, 2, 4, 7 e 9) ou alternativamente *SCLK, RxD/RDI, TxD/TDO, CTS* e *RTS*
- *PLMB* e C7 (pinos 5a e 6a) no caso do conector de 12 pinos

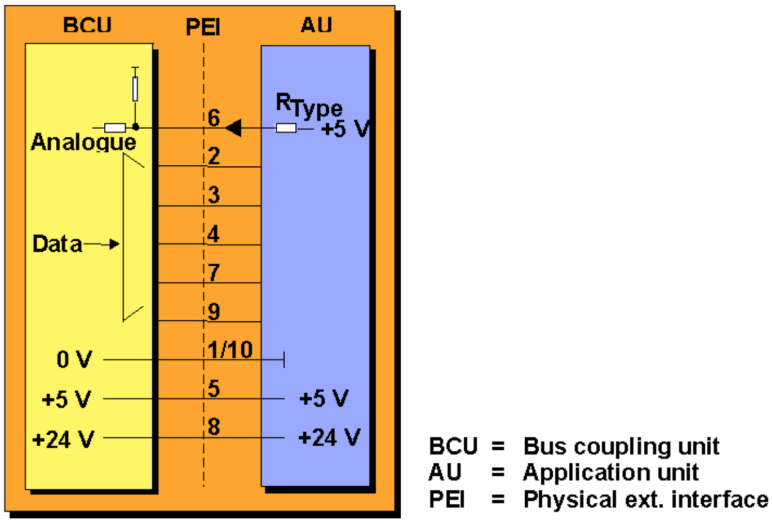

Figura 2.17 *PEI*

O *PEI-type* é definido através de uma resistência *R-type* ligada entre o pino 6 e 5V. O *BCU* é capaz de detetar através do *PEI-type* se o *Application Module* corresponde com o *Application Program*. Se o *PEI-type* não corresponder, o *BCU* pára automaticamente o *Application Program*.

Na tabela 2.1 é possível ver os *PEI-types* existentes:

Tipo	Tensão	Função
0	0.00	Sem AM conectado
2	0.50	4 Entradas Binárias (analógicas), 1 Saída Binária
4	1.00	2 Entradas Binárias (analógicas), 2+1 Saídas Binárias
6	1.50	3 Entradas Binárias (analógicas), 1+1 Saídas Binárias
12	3.00	Série Síncrono
14	3.50	Série Síncrono com tamanho fixo
16	4.00	Série Assíncrono
19	4.75	4+1 Saídas Binárias
20	5.00	Download

Tabela 2.1 *PEI-types*

Tipos de BIM

Existem 4 tipos de *BIM's* que podem ser escolhidos para o desenvolvimento de dispositivos. As diferenças entre eles são nomeadamente o tamanho e também numa versão varia a gama de temperaturas admitidas.

	Flash	RAM	Temperatura
BIM M130	8 KByte	200 bytes	-5°C...+45°C
BIM M131	16 KByte	1.2 KBytes	-5°C...+45°C
BIM M133	48 KByte	5.2 KBytes	-5°C...+45°C
BIM M135	8 KByte	200 bytes	-25°C...+70°C

Tabela 2.2 Tipos de *BIM*

2.3.4 BIM Evaluation Board

A maneira de começar o desenvolvimento de uma aplicação para o *BIM* é através da *BIM Evaluation Board*. Esta placa consiste num *BIM M133* em que todos os pinos do microcontrolador se encontram disponíveis, ao contrário de um normal *BIM* que apenas tem o *PEI* disponível. Isto permite um total controlo do microcontrolador inclusivé, a capacidade de o programar diretamente a partir do software sem tem de fazer todos os passos descritos na seção 3.5. Para tal é utilizado um programador da *Renesas*, o *E1*, que é configurado no *IAR Embedded Workbench*™ (seção 3.4.2) de modo a carregar o código. Desta forma é possível fazer *debug* do código enquanto se encontra a correr.

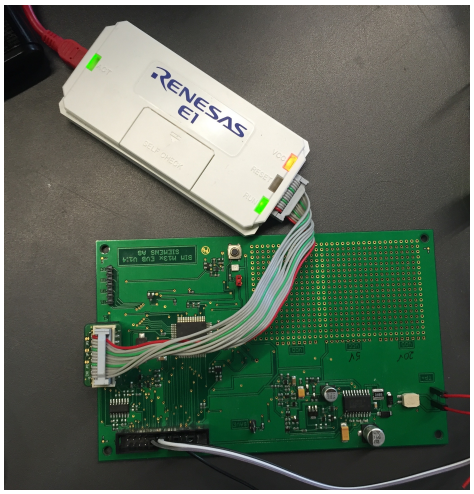

Figura 2.18 *BIM Evaluation Board* conectada ao programador *E1*

2.4 Criação de PCBs

A criação de *PCBs* (*Printed Circut Boards* ou Placas de Circuito Impresso) são um passo importante na criação de protótipos. Com a necessidade de passar o *Alfred* a protótipo, foram utilizadas e/ou testadas as seguintes técnicas.

2.4.1 Passagem do desenho para PCB

Insuladora

Este processo consiste na gravação das pistas através de luz *UV* numa Placa de Circuito Impresso Pré Sensibilizada, sendo necessário a passagem da placa em Soda Cáustica para o desenho ficar visível.

Nota: Insuladora cedida pela *Fablab Coimbra* para a realização dos testes.

Figura 2.19 Imagem da insuladora utilizada

Gravadora Laser

Este processo foi testado sem garantias do seu sucesso. Após várias tentativas de diversas potências e tempos foi possível chegar a resultados bastante impressionantes. A maior desvantagem é o custo associado à utilização de uma Gravadora Laser.

Nota: Gravadora cedida pela *Fablab Coimbra* para a realização dos testes.

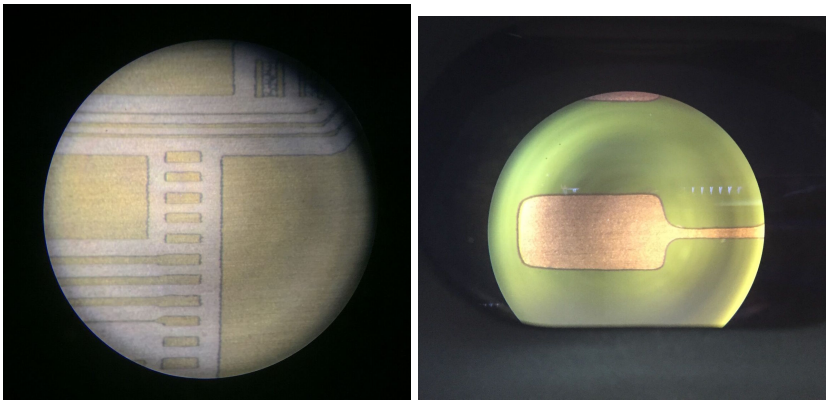

Figura 2.20 Vista ao microscópio de uma gravação a laser. À esquerda, gravação laser. À direita, revelação após gravação a laser

2.4.2 Revelação da PCB

Fresa

A fresagem de *PCBs* é um processo relativamente simples e de baixo custo. Consiste numa fresadora ou *CNC* que grava a Placa de Circuito Impresso deixando apenas o cobre das pistas e dos *pads*. As desvantagens devem-se ao facto de que se ter de utilizar velocidades baixas de modo a que não se partam as brocas, uma vez que são bastante finas. Para obter placas com distâncias mais curtas usam-se brocas de pequeno diâmetro, obrigando a velocidades baixas, aumentando assim o tempo de produção. Em contra partida, para encurtar o tempo, tem de se utilizar brocas maiores, fazendo com que não seja possível gravar certas pistas e em especial, *pads* para componentes *SMD*.

Nota: Gravadora cedida pela *Fablab Penela* para a realização do protótipo.

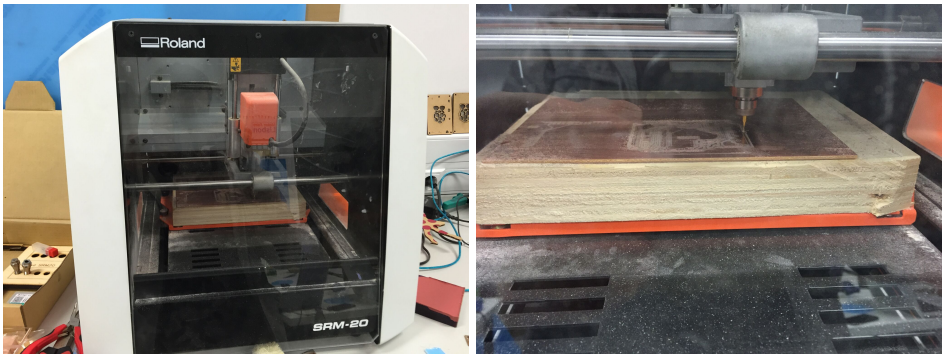

Figura 2.21 Revelação com Fresa

Percloreto de Ferro

Ao colocar no Percloreto de Ferro, idealmente aquecido, uma placa previamente preparada pelos processos 2.4.1, o cobre é corroído, deixando apenas as pistas. As desvantagens deste processo devem-se ao facto de não ser muito preciso, sendo necessário alguma prática e conhecimento de tempos e quantidades adequadas. Além disso, o Percloreto de Ferro corrói os materiais.

Nota: Percloreto e materiais cedidos pela *Fablab Coimbra* para a realização dos protótipos.

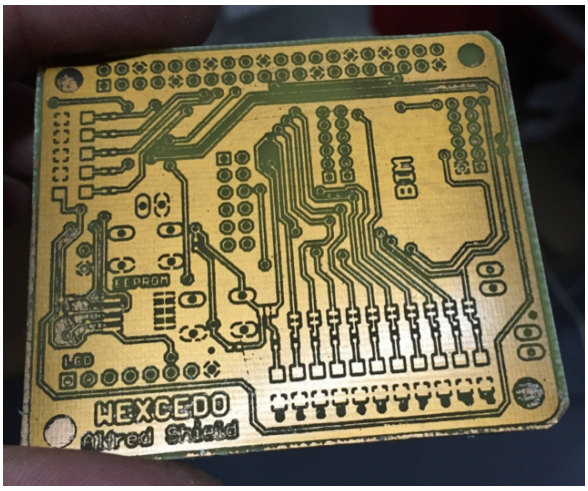

Figura 2.22 Revelação com Percloreto de Ferro

Ácido Muriático e Peróxido de Hidrogénio

Como alternativa ao Percloreto de Ferro, testou-se a solução de Ácido Muriático e Peróxido de Hidrogénio (vulgarmente conhecido como Água Oxigenada 130 vol.). Esta solução é altamente tóxica e deve ser preparada com muito cuidado e de preferência numa *hotte*. As desvantagens deste processo devem-se ao facto de ser necessário muitos cuidados na manipulação destes reagentes.

Nota: *Hotte* e materiais cedidos pela *Fablab Coimbra* para a realização dos protótipos.

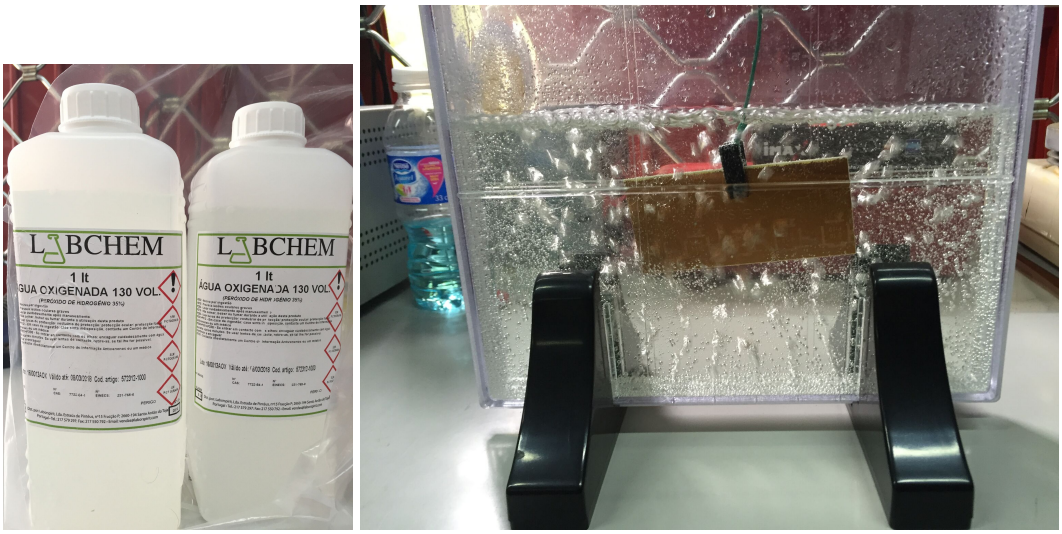

Figura 2.23 Revelação com Ácido Muriático e Peróxido de Hidrogénio

Capítulo 3

Implementação e Resultados

3.1 Infraestrutura desenvolvida - O Alfred

3.1.1 Estrutura física

O *Alfred* é composto por três módulos, sendo estes, o *Raspberry Pi*, a *shield* e um *BIM*. Nas figuras seguintes é possível visualizar o protótipo e respetiva caixa *DIN*.

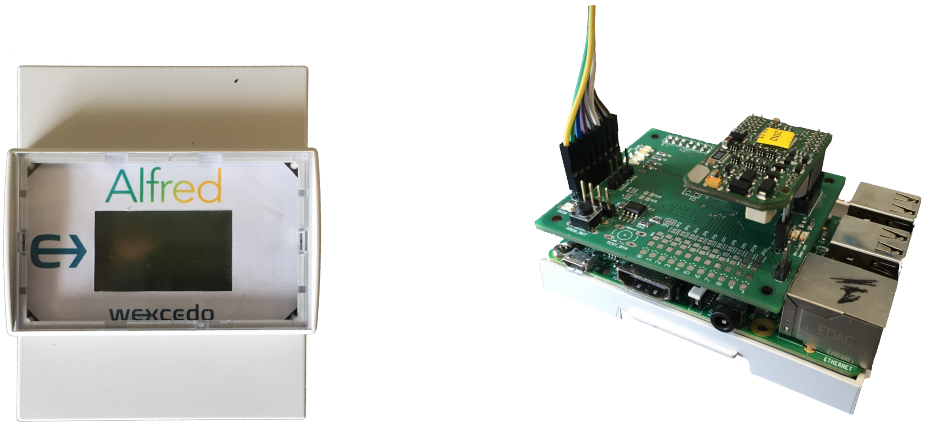

Figura 3.1 Aspecto físico do *Alfred*. À esquerda, protótipo com caixa *DIN*. À direita, protótipo.

3.1.2 Arquitetura de alto nível

A arquitetura de alto nível do *Alfred* pode ser visualizada na figura seguinte. De uma forma geral, estão representados os componentes que o integram. Quanto ao *Raspberry Pi*, é utilizada a porta Ethernet, uma placa WIFI ligada numa porta USB e 4 *GPIO's* conectados a LED's de estado da *shield*, indicando os modos de operação do *Alfred*: Off, Configuração, Ethernet e Wifi. São ainda usados os *GPIO's* referentes à comunicação série e I^2C. A comunicação I^2C serve para conectar a *EEPROM* ao *Raspberry Pi* e a comunicação série é utilizada entre o *Raspberry Pi* e o microcontrolador do *BIM*. Na *shield* estão ainda disponíveis o botão programação e o LED de programação do *BIM*. Foram adicionados LED's aos *GPIO's* do *BIM* para facilitar o debug do código. A *shield* encontra-se preparada para desativar os LED's de *debug* caso não sejam necessários.

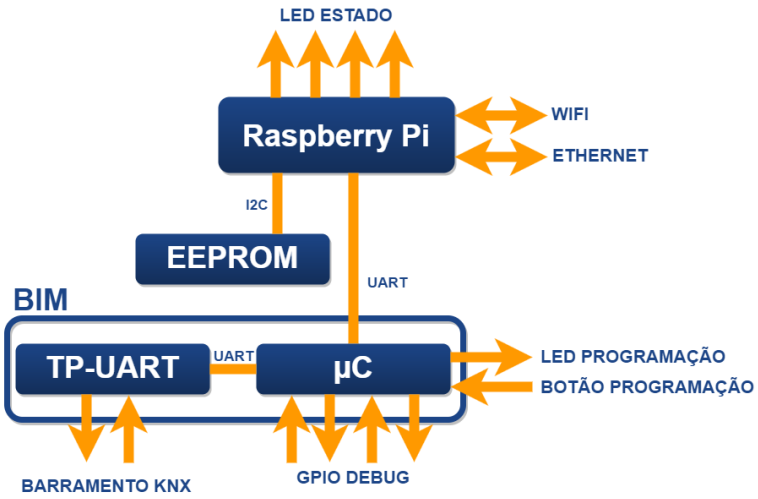

Figura 3.2 Arquitetura a alto nível do *Alfred*

3.1.3 Aplicações do Alfred

O *Alfred* é um dispositivo capaz de revolucionar uma instalação *KNX*. Estando apenas conectado ao barramento *KNX* o *Alfred* consegue efetuar operações lógicas com os GO's (*Group Objects*) existentes. Imaginando a necessidade de acender uma lâmpada quando premido um botão, mas apenas quando for de noite, é possível usando a lógica existente no *Alfred* para manipular os valores dos sensores (botão e sensor de luz) de modo a que a luz acenda quando ambos forem verdadeiros. Ligando o *Alfred* à internet, é possível aceder remotamente, e com segurança, à instalação onde se encontra. Utilizando vários *Alfred* é possível interligar instalações de modo a que passem a agir como uma só.

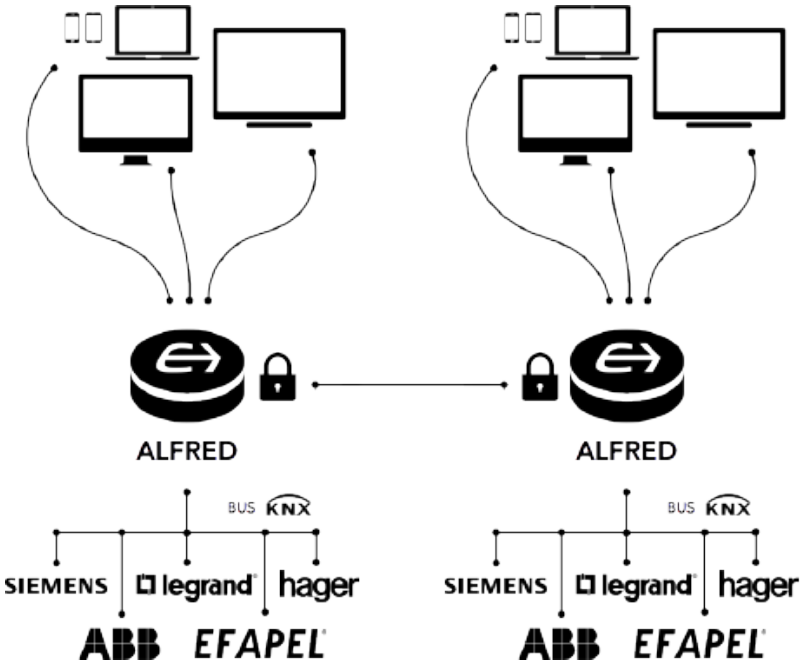

Figura 3.3 Aplicações do *Alfred* (conexão remota e interligação de dispositivos)

3.2 Fases de Desenvolvimento do Alfred

Os primeiros passos no desenvolvimento do *Alfred*, nomeadamente o trabalho de teste, foram efetuados numa *breadboard*. Os testes foram efetuados utilizando um *Raspberry Pi B*, um LCD 20x4 (uma vez que o de 16x2 não era suficiente), 4 LED's e um botão de pressão tal como é possível verificar na Fig. 3.4.

Esta montagem servia como base ao desenvolvimento do *Hotspot* referido na secção 3.3.2.

Os LED's têm o seguinte propósito (da esquerda para a direita):

- Vermelho - Indica que o *Alfred* se encontra em modo de configuração
- Amarelo - Indica que o *Alfred* se encontra em modo de *Hotspot*
- Verde (1) - Indica que o *Alfred* se encontra conectado à rede via cabo *Ethernet*
- Verde (2) - Indica que o *Alfred* se encontra conectado à rede via *WIFI*

No LCD era possível verificar os processos a correr. O botão efetuava *reset* ao *Alfred*.

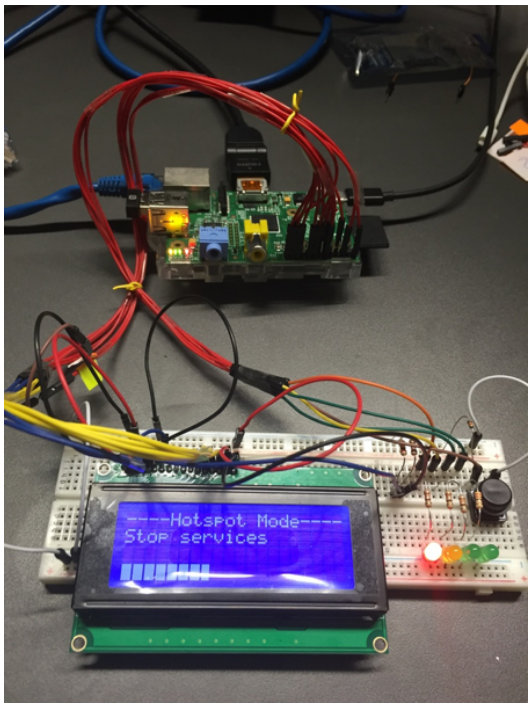

Figura 3.4 Alfred v1

Na fase seguinte, houve um *upgrade* de modelo e de quantidade de *Raspberry Pi*, passando para o modelo 2 B+ - ver Fig. 3.5. O LCD também sofreu um *upgrade* passando a ser gráfico. Foi acrescentado também a funcionalidade que informa o *MAC address* no caso da rede *WIFI* esteja protegida com filtro *MAC address*.

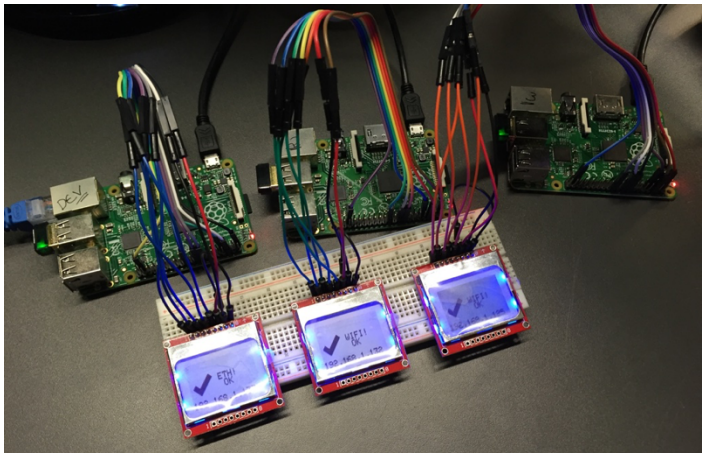

Figura 3.5 Alfred v2

Na Fig. 3.6 consta o primeiro protótipo assemblado. Para tal recorreu-se a uma placa perfurada como suporte. Neste passo também foi adicionada a *EEPROM* que contém o número de série, referida na secção 3.3.5.

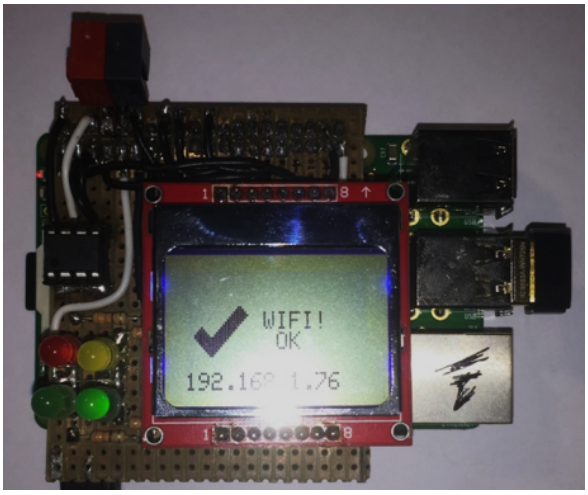

Figura 3.6 Alfred v3

Na Fig. 3.7 está presente a primeira placa que serviria como base ao *Alfred*. Ao nível de *hardware* é igual à versão anterior, apenas mudando o suporte, passando para uma Placa de Circuito Impresso (*PCB*) e os componentes passaram a ser do tipo *SMD*. Foi utilizado o processo referido na secção 2.4.2. Após a assemblagem foram encontrados alguns erros que viriam a ser corrigidos na versão seguinte.

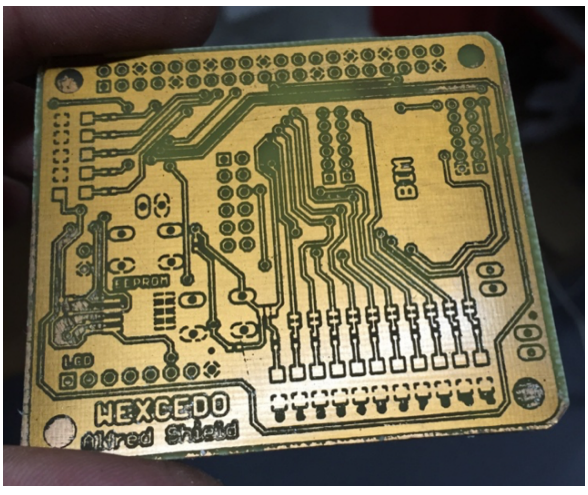

Figura 3.7 Alfred v4

A versão mais atual da *PCB* do *Alfred* encontra-se na Fig. 3.8. Após a correção dos erros da versão anterior, procedeu-se à produção da Placa de Circuito Impresso, desta vez ao encargo da empresa *Euro Circuits* [1].

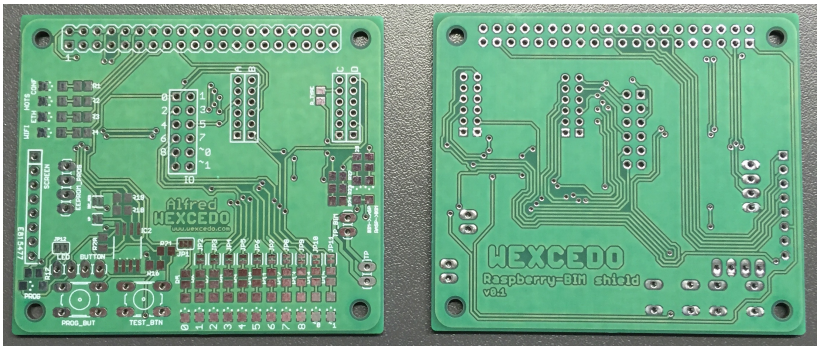

Figura 3.8 Alfred v5

3.3 Detalhes do Alfred

3.3.1 Desenvolvimento, compilação e execução de código

O desenvolvimento do *Alfred* começou com uma primeira ambientação a uma nova forma de utilizar um sistema operativo sem utilizar um ambiente gráfico. Para facilitar a criação e o desenvolvimento de código foi necessário conhecer dois comandos do *Linux*, o *SSH* e o *SCP*. O comando *SSH (Secure Shell)* serve para iniciar sessão e executar comandos num dispositivo remoto de forma segura e encriptada. O *SCP (secure copy)* utiliza o *SSH* para enviar ficheiros entre dispositivos. Desta forma, com estes comandos, pode-se desenvolver código num dispositivo externo e, através do comando *SCP*, enviá-lo para o *Raspberry Pi*, compilar e executar esse código num terminal remoto através de uma ligação via *SSH*.

3.3.2 Hotspot

O primeiro objetivo do *Alfred* como dispositivo é a fácil instalação e integração num lar. Sendo a base do *Alfred* um *Raspberry Pi* a correr *Linux*, seria necessário ligar periféricos como rato e/ou teclado para efetuar a ligação e autenticação a uma rede *WIFI*. Para contornar essa necessidade surgiu a ideia da criação de um *Hotspot Wifi*, para quando não existisse uma rede *Ethernet* (não necessita de credenciais (conecta automaticamente)), fosse possível

[1] *www.eurocircuits.com*

conectar um dispositivo móvel a essa rede, de modo a que fossem exibidas todas as redes disponíveis e possibilitasse a sua autenticação. Uma vez que o modelo do *Raspberry Pi* utilizado não tem placa *WIFI* nativa, foi necessária a utilização de uma *Pen Wifi USB*. Tendo já todos os periféricos necessários o passo seguinte é a alteração de ficheiros do sistema de modo a que seja possível a configuração do *Hotspot*. Os passos importantes a reter são, a atribuição de um *IP* estático de modo a que seja esse o *IP* que será inserido para aceder à página Web, a criação de um servidor de *DHCP*, para atribuir endereços *IP* aos dispositivos conectados ao *Hotspot* e por fim a configuração do *hostapd (Access Point daemon)* onde se define o *SSID* e palavra-passe da rede. Estando o *Hotspot* devidamente configurado é possível estabelecer ligação através de um dispositivo móvel à rede *WIFI* do *Raspberry Pi*. Desta forma, quando o *Alfred* é colocado num novo lar, e quando não existe rede *Ethernet*, este cria um *Hotspot* e indica que o *Hotspot* se encontra criado através de um LED amarelo e o IP da página no seu LCD. Caso a rede local esteja protegida por filtro de *MAC address*, o *Alfred* disponibiliza também o seu *MAC address* no LCD para este poder ser adicionado aos filtros de *MAC* da rede.
Na seção B.1 encontram-se todos os passos referentes à criação do *Hotspot*.

Para proceder à criação da placa, foi utilizado o processo de Fresagem referido na seção 2.4.2.

3.3.3 Página Web

Para ser possível mostrar ao utilizador uma página *Web* quando se conectado ao *Hotspot*, é necessária a instalação de um servidor *Web*. Neste caso, o servidor *web* escolhido foi o *Apache*. Uma vez instalado basta simplesmente colocar o site na diretoria */var/www/html/*. Através de um dispositivo ligado ao *Hotspot* é possível visualizar a página acedendo ao IP estático definido como *IP* do *Hotspot*, neste caso, 192.168.2.1. Para a criação da página foi utilizado o *Bootstrap* como base. Utilizando *PHP* foi possível fazer a interligação entre os ficheiros presentes no *Raspberry Pi* e a página Web e correr comandos no *Raspberry Pi*, como por exemplo, de modo a exibir as redes disponíveis, através de código em *PHP* pode-se correr comandos de consola, neste caso o *"iwlist wlan0 scan"* que devolve informação sobre todas as redes disponíveis. Na página *web* é possível observar todas as redes existentes bem como a qualidade de sinal e se se encontra protegida ou não. Nesta versão ainda não é possível conectar a redes protegidas por *WPA2 Empresarial*. Após escolhida a rede e inserida a palavra passe, o *Alfred* termina o *Hotspot* e conecta-se à rede selecionada e liga o LED correspondente à conexão por *WIFI* e indica no *LCD* que foi conectado com sucesso e o seu *IP* na rede. Caso a conexão não seja valida, por exemplo, devido à introdução de uma

palavra-passe errada, é novamente criado o *Hotspot* de modo a que seja possível inserir de novo a palavra-passe.

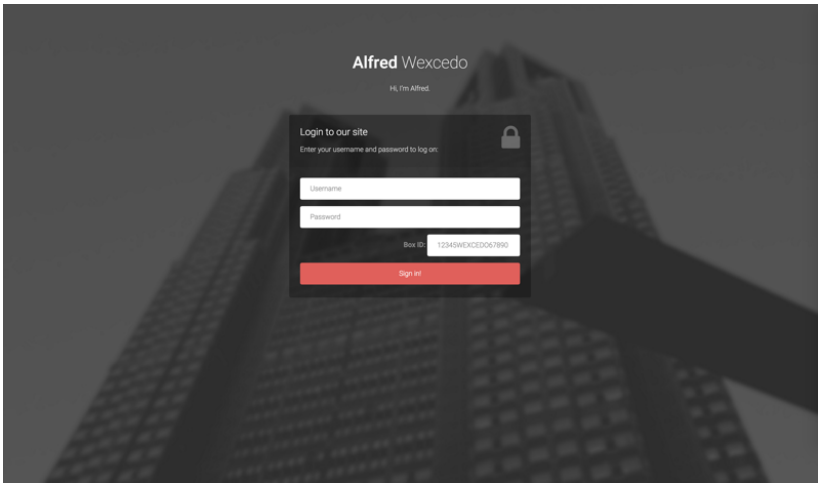

Figura 3.9 Primeira fase de design da página do *Alfred*

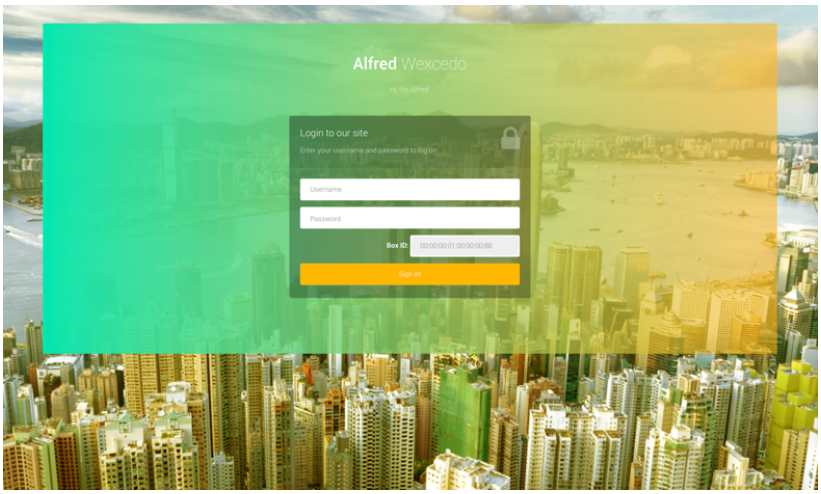

Figura 3.10 Design actual da página do *Alfred*

3.3.4 Port-Forwarding

Para ser possível estabelecer conexão entre os servidores *Wexcedo* e o *Alfred* é necessário que o router tenha portos do exterior redirecionados para o *Alfred*. Para evitar que o utilizador tenha que o fazer manualmente, caso o router tenha o protocolo *UPnP* ativo, os portos são redirecionados automaticamente.

3.3.5 Utilização e programação da EEPROM

Após iniciar autenticar com sucesso na rede local, é atribuído um *IP* ao *Alfred*, *IP* este que é exibido no LCD. Acedendo a esse *IP* através de um *browser*, é exibida a página de autenticação do *Alfred*, onde se encontra os campos de *login* e o número de série. Primeiramente o número de série era guardado num ficheiro no *Alfred*, mas rapidamente se chegou à conclusão que não seria viável por motivos de segurança e dificuldade de incrementação na produção. Dessa forma, a alternativa foi colocar o número de série, bem como a data de produção numa *EEPROM*, que é gravada automaticamente aquando a produção do *Alfred*. A *EEPROM* conecta-se ao *Alfred* através de um barramento I^2C.

Com o intuito de facilitar o processo de gravação de *EEPROM's* foi desenvolvido um dispositivo que o fizesse automaticamente. A base do gravador é um *Atmega328p* anteriormente referido como sendo o microcontrolador do *Arduino*. Uma vez que o microcontrolador tem uma *EEPROM* interna, esta é utilizada para guardar o número de série atual de modo a que, mesmo desligando o gravador, na vez seguinte, seja possível incrementá-lo. Estão disponíveis dois modos de gravação, sendo um, colocando uma *EEPROM* no *socket* e o outro, conectando a uma *EEPROM* externa através dos pinos disponíveis. Quando utilizado o *socket*, existe um botão para efetuar a gravação. Caso seja através dos pinos, existe um pino de *Input* que se encontra em *Pull-Up* pois o *Alfred* contém um pino a *Ground* no *header* de gravação fazendo com que ao ligar o conector, a gravação se efetue automaticamente.

Para além do número se série, é gravado também o lote e a data de produção. Para tal existe um *RTC (Real Time Clock)* de forma a que a data seja preservada mesmo com o gravador desligado.

Tanto o *RTC* como as *EEPROM's* conectadas comunicam através de I^2C, sendo as *EEPROM's* o endereço 0x50 e o *RTC* o endereço 0x68.

No LCD é possível visualizar o número de série a ser gravado e indica a conclusão da gravação. Existe ainda um LED que indica o processo de gravação piscando.

Características

- Microcontrolador: *Atmega328p*
- Tensão de operação: 5V
- Tensão de entrada (recomendada): 7-25V
- *Socket DIP* de 8 pinos para *EEPROM*
- Pinos para conexão a *EEPROM*

- LCD alfanumérico 16x2
- Contaste do LCD regulável
- Pinos de programação
- *Real Time Clock*

Na Fig. 3.11 é possível visualizar o aspecto visual do gravador.

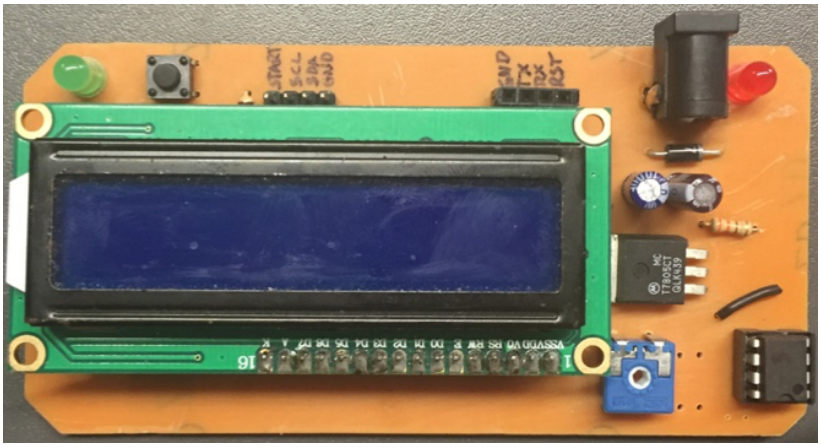

Figura 3.11 Gravador de *EEPROM's*

Nas seções A.3 e A.4 é possível visualizar o esquemático e o desenho da *PCB*, respectivamente. A placa foi desenhada com o *software EAGLE PCB Design* da empresa *CadSoft Computer GmbH*.

Para proceder à criação da placa, foi utilizado o processo de Fresagem referido na seção 2.4.2.

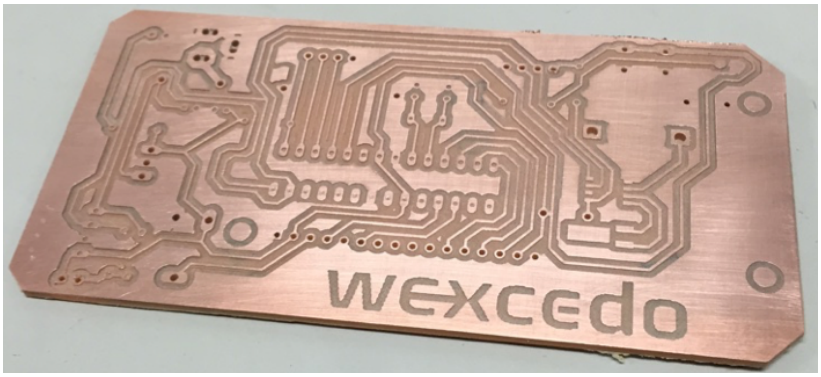

Figura 3.12 *PCB* do gravador de *EEPROM's*

3.3.6 Hardware do Alfred

Na Fig. 3.13 são demonstradas as características e funções principais do *Alfred*. Algumas das funcionalidades são apenas de teste, e por esse motivo, apenas se encontram assembladas em placas de desenvolvimento.

Características

- LEDs de Estado
- Pinos para gravação da *EEPROM* através do Gravador da secção 3.3.5
- *EEPROM* com as seguintes funcionalidades:

  Jumper de proteção de escrita na *EEPROM*

  Jumper de resistências de *Pull-Up* de I^2C
- LED de Programação
- Botão de Programação
- Botão de Teste (convertível no LED de teste 0 através de *jumper*)
- 10 LEDs de teste de *GPIO's* sendo os 2 últimos capazes de *PWM*
- Conversão do protocolo Série (5V para 3.3V) com LED's de *debug*
- Pinos de teste para *GPIO's* do *BIM*

Nas seções A.1 e A.2 é possível visualizar o esquemático e o desenho da *PCB*, respectivamente. A placa foi desenhada com o *software EAGLE PCB Design* da empresa *CadSoft Computer GmbH*.

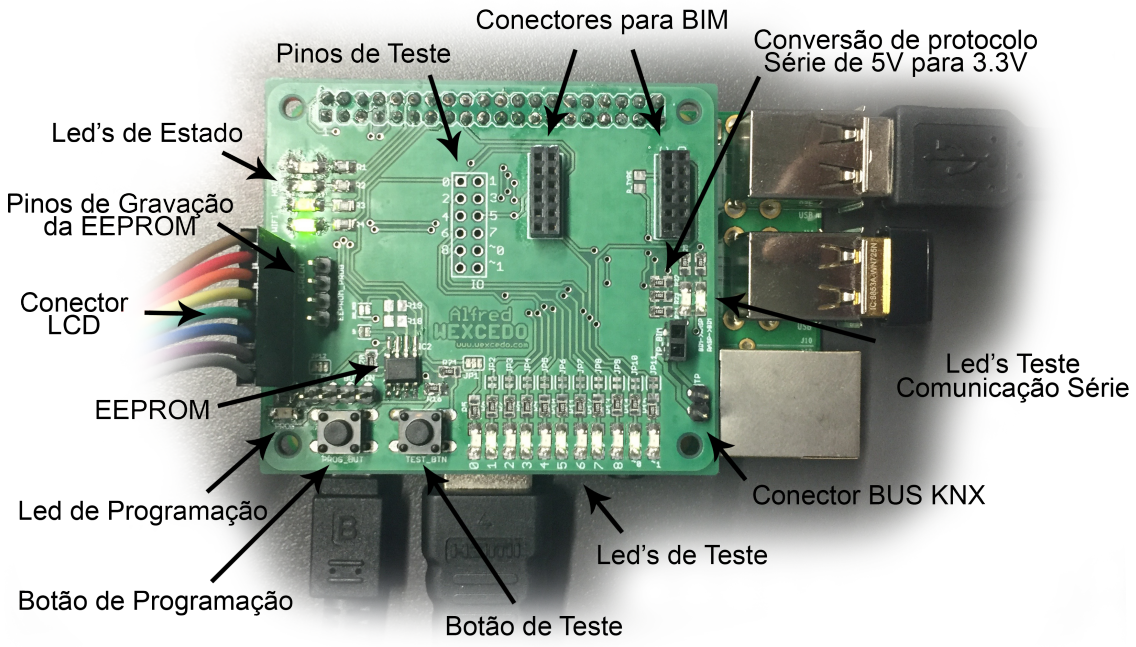

Figura 3.13 Características da PCB do *Alfred*

3.4 Processo de implementação do Alfred - KNX

Para o desenvolvimento de dispositivos *KNX* é necessário uma serie de procedimentos envolvendo vários *softwares* desde a programação, a integração e a configuração do dispositivo.

3.4.1 BIM-Tools

Como anteriormente referido, o desenvolvimento do *Alfred* no que toca à componente *KNX* tem por base um *BIM*. Dessa forma, é necessário software para facilitar o começo do desenvolvimento do código necessário para o microcontrolador do *BIM*. É através deste código que irão ser definidos no *BIM* todas as variáveis referentes a Objectos e a Parâmetros a que os instaladores irão ter acesso. Esse software é o ***BIM-Tools*** da *Siemens* e foi utilizada a versão 1.1.54.23007.

O primeiro passo é iniciar o *BIM-Wizard* onde são criados os Objetos, a Tabela de Endereçamento e Tabela de Associações.

Para o início do desenvolvimento, foi tido como base o documento 2.5.1 do **KNX Cookbook** da norma *KNX* [3].

Tabela de Endereçamento, de Objetos e de Associações

Estas tabelas são a base da gestão dos Objetos e dos Endereços de Grupo que lhes serão atribuídos.

À chegada de uma mensagem *KNX*, a *stack* faz o seguinte:

- Analisa o Endereço de Grupo da mensagem
- Determina o Objecto associado consultando:

  A Tabela de Endereços de Grupo onde listam os Endereços de Grupo

  A Tabela de Associações, que associa os Endereços de Grupo com os Objetos

  A Tabela de Objetos onde listam os Objetos e as suas descrições
- Atualiza o valor do Objecto no código
- Atualiza a *RAMFlag* do Objecto indicando que foi alterado o valor.

De seguida a aplicação:

- Verifica que houve algum *Update* nas *RAMFlags*
- Executar o eventual código que use o Objecto
- Faz *Reset* às *RAMFlags*

Na Fig. 3.14 é possível verificar todo o processo da chegada da mensagem ao *BIM*. Na Fig. 3.15 é possível verificar o ficheiro *tables.c* criado pelo *BIM-Wizard* onde estão presentes os Objetos a Tabela de Endereçamento e Tabela de Associações.

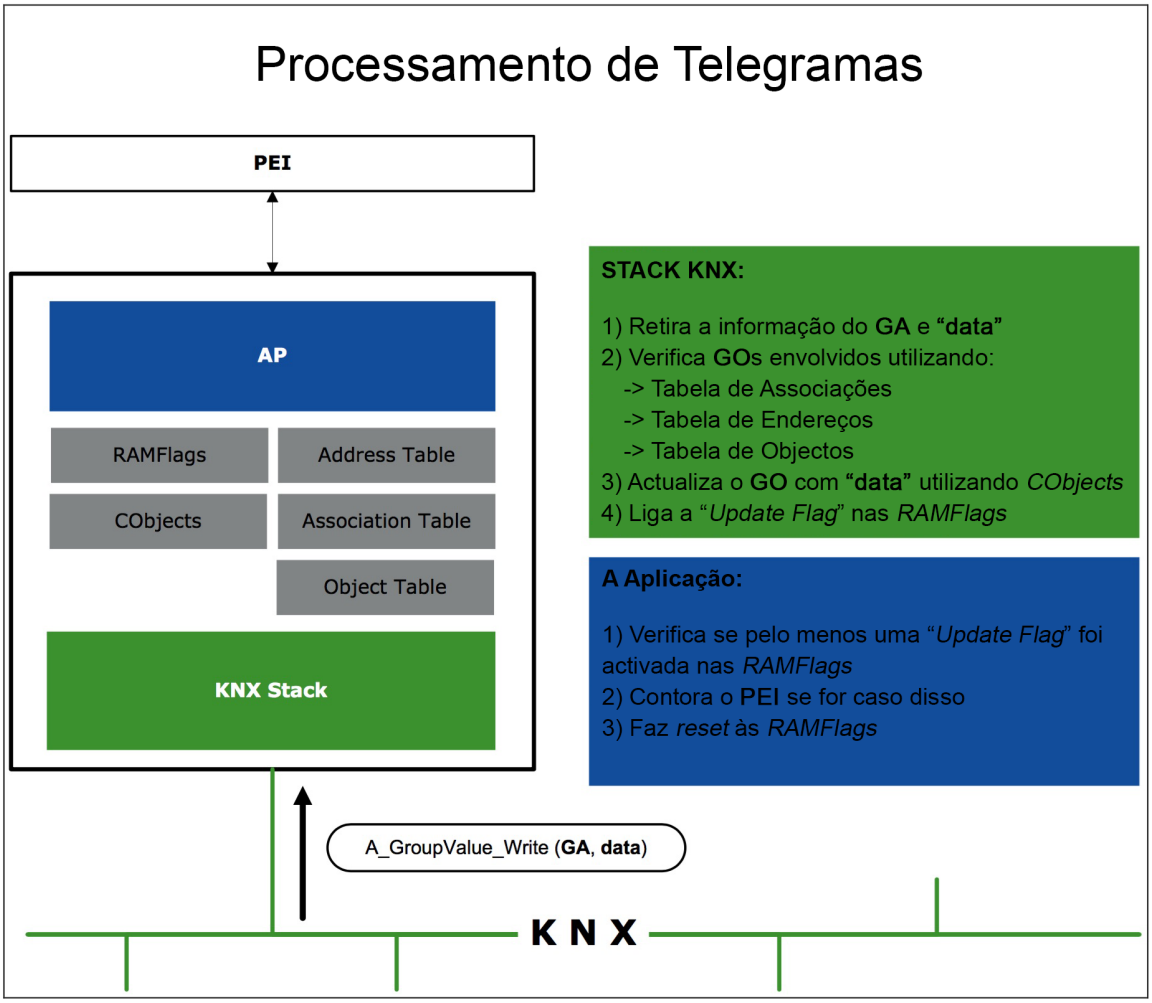

Figura 3.14 Exemplo do processo de um Telegrama KNX - adaptado de: Norma KNX 2.5.1 [3]

Analisando o ficheiro verifica-se que a tabela de Endereços (`AdrTab[]`) se encontra no segmento *ADDRTAB* e tem como primeiro parâmetro, o número total de Objetos de Comunicação mais 1, de seguida encontra-se o Endereço Físico, por fim, encontram-se os valores dos Endereços de Grupo.
Da mesma maneira, verifica-se que a Tabela de Associação se encontra no segmento de memória *ASSOCTAB* e tem como parâmetros o número total de Objetos de Comunicação e de seguida aparece a associação do *index* do Endereço de Grupo ao *index* de Objetos de Grupo. Por fim encontra-se a tabela de Objetos de Comunicação ou Objetos de Grupo no segmento *COMTAB*. Os parâmetros dessa tabela são o número total de Objetos de Comunicação e de seguida as *Flags* para cada objecto.

Ex.: Este dispositivo tem 2 Objetos, em que o *Obj0* é acessível pelo Endereço de Grupo 0/0/4 (0x00 0x04) e o *Obj1* pelo Endereço de Grupo 0/0/2 (0x00 0x02) pois se encontra associado pela Tabela de Associação que o primeiro Endereço de Grupo da Tabela de Endereços (0x00 0x04) está associado ao Objecto 0x00 (*Obj0*). Da mesma maneira a Tabela de Associação indica que o segundo Endereço de Grupo da Tabela de Endereços (0x00 0x02) está associado ao Objeto 0x01 (*Obj1*).

Nota: Todos estes valores são para efeitos de *debug* caso carregado para a *Evaluation Board*, pois numa normal utilização do *BIM* estes valores seriam carregados pelo o *ETS*. O Endereço Fisico de *debug* é ignorado e assumido 15.15.255 até ser atribuído um endereço válido.

```
/**************************************************************************
*
* File: tables.c
*
**************************************************************************/
#include "BIM_M13x.h"

#pragma constseg = ADDRTAB
__root const BYTE AdrTab[] =
{     NUM_OF_COM_OBJ + 1,    // for debug: one group address per communication object
      0x11, 0xFD,            // physical address 1.1.253
      // group addresses
      0x00, 0x04,
      0x00, 0x02,
      0x00                   // padding
};
#pragma constseg = ASSOCTAB
__root const BYTE AssTab[] =
{     NUM_OF_COM_OBJ + 0,    // for debug: one association per communication object
//    grp.addr.,  comobj. num.,
           0x01,          0x00,
           0x02,          0x01,
           0x00                  // padding
};
#pragma constseg = COMTAB
__root const BYTE EE_CommsTab[] =
{     NUM_OF_COM_OBJ,        // number of communication objects,
                0x00,        // pointer to RAMFlags is no longer used
                             // but must be specified for compatibility
/*Obj0*/ NULL, (COMM_ENABLE + WRITE_ENABLE + TRANSMIT_ENABLE + READ_ENABLE + TX_LOW), UINT1,
/*Obj1*/ NULL, (COMM_ENABLE + WRITE_ENABLE + TRANSMIT_ENABLE + READ_ENABLE + TX_LOW), UINT8

};
#pragma constseg = default
```

Figura 3.15 Exemplo da inicialização de Tabelas pelo BIM-Wizard

3.4.2 IAR Embedded Workbench™

Para editar os ficheiros criados pelo *BIM-Wizard* usa-se o *software IAR Embedded Workbench for Renesas 78k*, tendo-se recorrido para este projeto a versão 4.80.2.

Na Fig. 3.16 é possível ver a hierarquia de ficheiros do projeto. Os ficheiros mais utilizados no desenvolvimento são os seguintes:

- BIM_M13x.h - ficheiro que contém os definições e as estruturas definidas na norma *KNX*
- main.c - Ficheiro onde será escrito o código
- tables.c - Ficheiro que contém as definições dos Objetos e as Tabelas de Endereçamento e de Associações

O código a correr no *BIM* é escrito no ficheiro *main.c* dentro de um ciclo infinito.

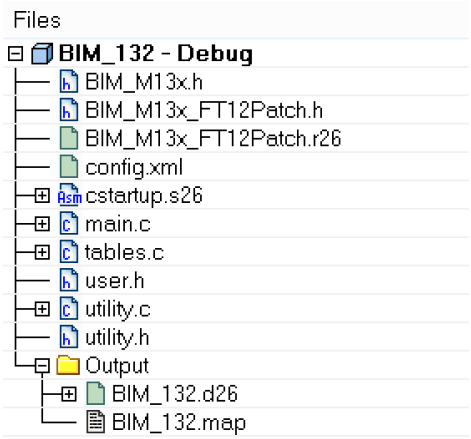

Figura 3.16 Ficheiros criados pelo BIM-Wizard abertos no IAR

Após desenvolvido o código é necessário passar o compilador de modo *Debug* para modo *Release* e compilar de modo a ser gerado um ficheiro com a extensão ".s19". Este ficheiro não é utilizável no *Manufacturer Tool*, tendo que ser modificado através de uma das seguintes alternativas:

- No *IAR Embedded Workbench*™ adicionando o comando referido na Pág. 25 do documento 2.5.1 da Norma *KNX* ao *Post Buid*
- Com o *BIM Tools* executar a funcionalidade *"Modify s19"*

3.5 Processo de implementação do Alfred - Produto ETS

Para o *Alfred* ser um poduto *KNX*, tem que fazer parte do Catálogo de Produtos do *ETS*. A partir do produto, será possível configurar o *Alfred*, configurando o número de Objectos que terá, consoante o número de Regras Lógicas que possuir.

Desta forma, após concluída toda a programação ao nível do microcontrolador, é necessário fazer a interligação com o que será o produto final, o produto *Alfred* no *ETS*. Para tal, é necessário conectar todas as seções de memória referentes a Objectos e Parâmetros com o Produto. Esse processo é efectudo com o **KNX Manufacturer Tool**.

3.5.1 KNX Manufacturer Tool

O *KNX Manufacturer Tool* é o principal *software* na criação de um dispositivo *KNX*. Este permite criar e testar uma nova entrada para o dispositivo na Tabela de Produtos do *ETS*. A versão utilizada foi a v5.5.0.

Para começar o desenvolvimento do Produto, é necessário criar os ficheiros referentes a:

- *Application Program* - onde é feito o mapeamento de toda a memória
- *Hardware* - onde são definidos: o número de série, número de *Manufacturer*...
- *Catalog* - onde são criadas as entradas dos dispositivos que irão aparecer no *ETS*

É no ficheiro do *Application Program* que é feito grande parte da configuração. Existem duas grandes seções dentro do ficheiro: a seção estática e a seção dinâmica.

Seção Estática

Nesta seção são configurados:

- Segmentos de Código - onde são definidas as seções de memória
- Parâmetros
- Objetos
- Procedimentos para o carregamento da aplicação - onde são definidos todos os passos a ser efetuados pelo *ETS* aquando o carregamento
- Dados binários - onde é carregado o ficheiro *.s19* modificado

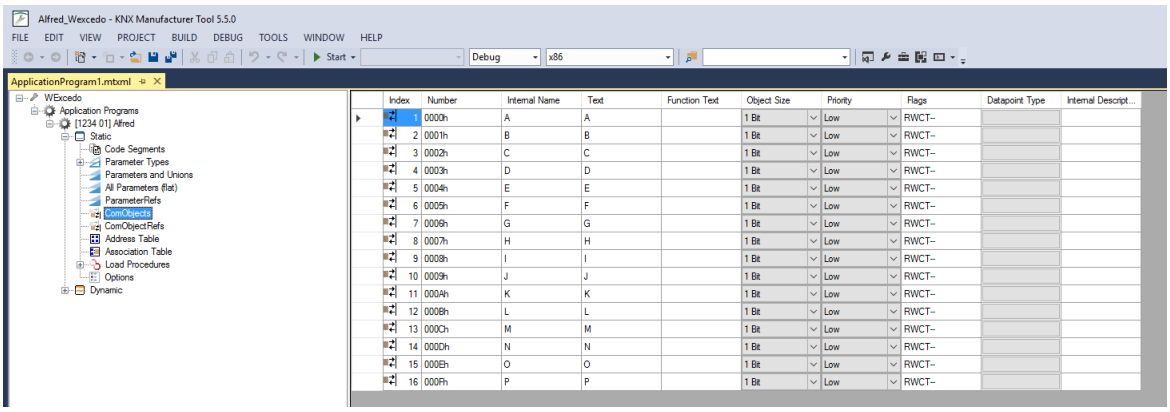

Figura 3.17 Edição da seção estática do BIM - Objectos

Na Fig. 3.17 é possível ver a configuração dos Objetos utilizados no *Alfred*. Todos os Objetos presentes (do A ao P) são referentes às Regras Lógicas do mesmo.

Na Fig. 3.18 verifica-se que foram criados 4 Tipos de Parâmetros. É importante a criação de Tipos de Parâmetros pois quando forem utilizados na seção Dinâmica, o *ETS* saberá automaticamente o aspecto gráfico dos campos. Por exemplo, foi criado um Tipo *Restriction* ao qual foi dado o nome *Lógica*, o *ETS* já sabe que todos os Parâmetros que forem da categoria *Lógica*, serão *Restrictions*, logo aparecerão como *ComboBox*. Se fossem da categoria *Text* apareceriam como *TextBox*.

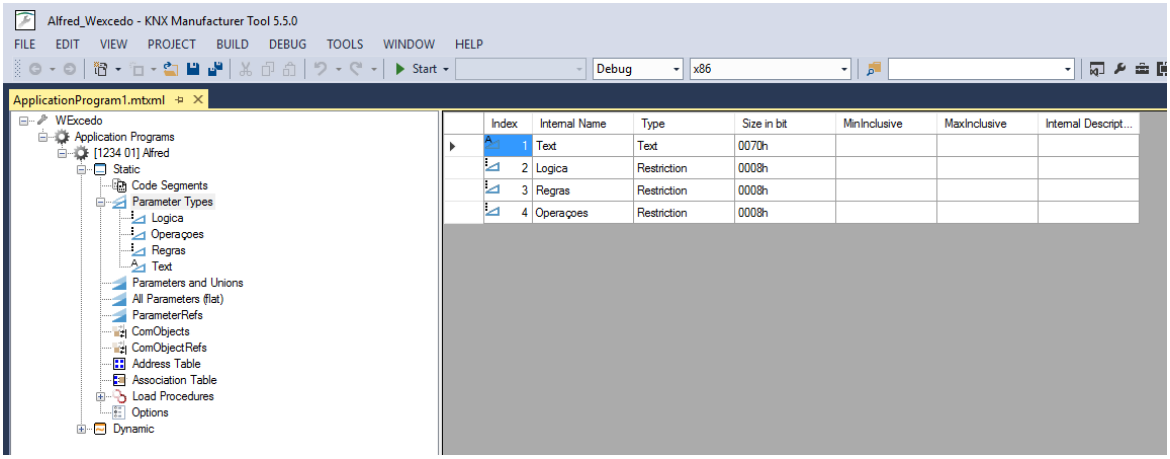

Figura 3.18 Edição da seção estática do BIM - Tipos de Parâmetros

Na Fig. 3.19 encontra-se a definição de alguns dos Parâmetros do *Alfred*. Estes Parâmetros aparecerão no Produto *Alfred* do *ETS*. Nesta seção apenas se definem e atribuem os tipos definidos anteriormente. Só na seção Dinâmica é que são utilizados de modo a compor a

interface gráfica. É possível verificar a existência de um Parâmetro dedicado ao *DNS* do tipo texto. Os restantes são referentes às Regras Lógicas, dois *inputs* por regra, uma operação e um *output*.

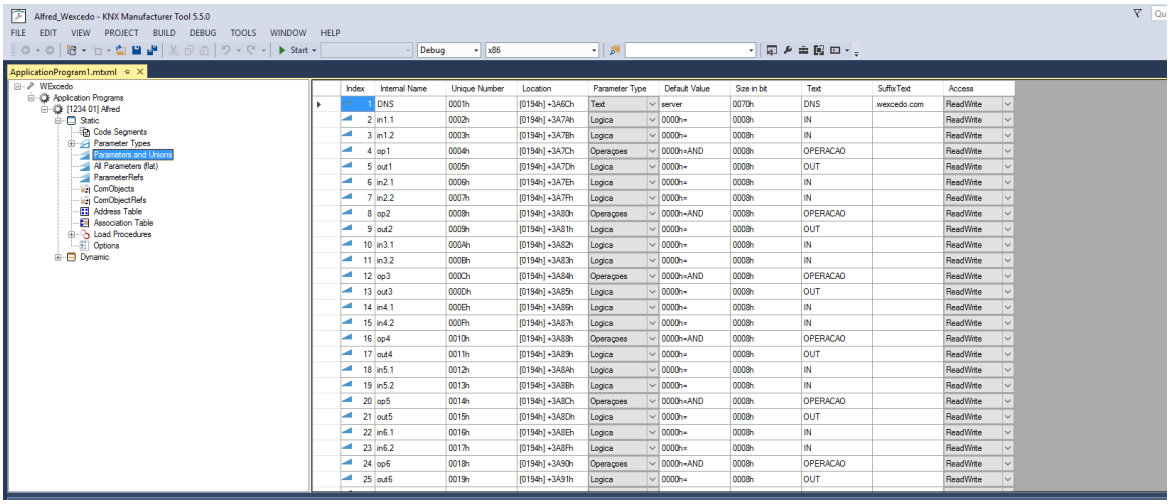

Figura 3.19 Edição da seção estática do BIM - Parâmetros

Seção Dinâmica

Nesta seção é definida toda a estrutura que aparecerá no *ETS* (Fig. 3.20) bem como as condições dos Parâmetros pelos quais os Objetos aparecerão no dispositivo. Também se indicam todos os Parâmetros a ser exibidos, que também podem variar consoante condições aplicadas a outros Parâmetros.

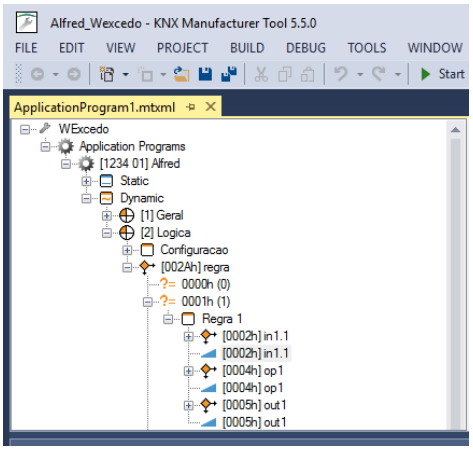

Figura 3.20 Edição da seção dinâmica do BIM

3.5.2 Resultado no ETS

O resultado obtido no *ETS* é o exibido na Fig. 3.21. É possível verificar a existência da hierarquia definida na Fig. 3.20 e que os Parâmetros são os definidos na Fig. 3.19.
Resta apenas, após a configuração, carregar toda a informação através do barramento.

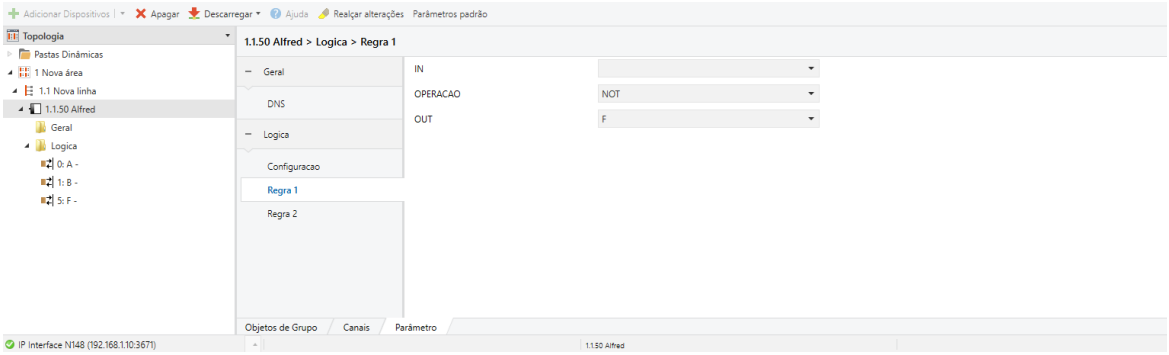

Figura 3.21 Exemplo de edição de uma Regra de Lógica do *Alfred* no *ETS*

3.6 Resultados Experimentais

3.6.1 Instalação do Alfred num Alojamento Rural

Como primeira experiência fora da bancada de testes, o *Alfred* foi implementado numa situação real onde o objetivo é uma conexão remota. Este teste está a servir para identificar possíveis melhorias e encontrar eventuais erros.

O local da instalação é um Alojamento Rural com diversos quartos em que era pretendido que cada arrendatário apenas obtivesse acesso a certas divisões, incluindo a porta principal.

A implementação consta numa instalação de diversas fechaduras elétricas, um atuador com múltiplas saídas e um *Alfred*. O atuador é o dispositivo responsável pela abertura das fechaduras. Assim, o *Alfred* serve como ligação remota entre a aplicação do proprietário e a instalação *KNX* do Alojamento Rural. Deste modo, os utilizadores serão capazes de abrir remotamente as divisões a que têm direito e o proprietário gerir remotamente as permissões dos utilizadores.

Até ao momento, a instalação encontra-se a funcionar como pretendido, não se tendo sido encontrado até então qualquer erro que afetasse o seu desempenho expectado.

3.7 Expositor Alfred

Para poder ser apresentado em exposições, foi criado um expositor onde consta um *Alfred* e respectiva *PCB*. Deste modo, é possível mostrar o produto final em funcionamento, bem como a sua constituição.
Este expositor já foi apresentado em diversas feiras em vários países - ver Fig. 3.22.

Figura 3.22 Expositor do *Alfred*

Capítulo 4

Conclusão e Trabalho Futuro

4.1 Conclusão

O trabalho realizado nesta dissertação focou-se no desenvolvimento de uma infraestrutura que permitisse o controlo remoto de Casas Inteligentes. Todas as etapas estabelecidas inicialmente relativas ao desenvolvimento durante a dissertação, foram realizadas com sucesso, tendo inclusivé já sido implementadas e testadas numa situação real. Deste modo, o *Alfred* é capaz de se integrar numa instalação, conectar à internet, estabelecer conexões com o exterior e atuar numa instalação *KNX*.

4.2 Trabalho Futuro

O trabalho desenvolvido no âmbito desta dissertação resultou num primeiro protótipo. Existem ainda alguns passos a ser concretizados, nomeadamente:

Hardware

- Desenvolvimento de uma nova *shield* que inclua o *BIM*, ou seja, desta forma o *BIM* deixará de ser um dispositivo independente e passará a fazer parte da *shield*
- Implementação de um multiplexador de comunicação série entre o *TP-UART* e o microcontrolador do *BIM* fazendo com que seja possível multiplexar as mensagens vindas do *TP-UART* para o microcontrolador e para o *Raspberry Pi* ou vice-versa. Isto deve-se ao facto do microcontrolador do *BIM* rejeitar certas mensagens, mas uma vez que chegam ao *TP-UART*, estas serão enviadas diretamente para o *Raspberry Pi*.

- Desenvolvimento de um dispositivo completo utilizando o *Raspberry Compute Module*, fazendo com que de deixe de utilizar um *Raspberry Pi* comum, aproximando-se de um produto final

Software

- Melhoramentos ao nível da conexão WIFI a partir da página *web*, nomeadamente, a adição da capacidade de conexão a redes protegidas por *WPA2 Empresarial*
- Permitir a detecção da conexão de um cabo de Rede de forma a que, mesmo conectado a uma rede WIFI, o *Alfred* seja capaz de comutar para a conexão por cabo autonomamente

Bibliografia

[OSI] *Serial Data Transmission and KNX Protocol*. KNX Association.

[2] Akamai Technologies (2014). Second Quarter 2014 'State Of The Internet' Report – Highlights. Online.

[3] Association, K. (2013). *KNX Cookbook - KNX development on basis of existing system components*, volume 2.5.1. KNX Association.

[4] Association, K. (2016). KNX the worldwide STANDARD for home and building control. Online.

[5] Buckingham, A. (2015). The history of home automation from the beginning. Online.

[6] e Keith W. Ross, J. F. K. (2007). *Computer Networking - A Top-Down Approach*. Pearson Education.

[7] Kurkinen, L. (2014). Smart Homes and Home Automation – 68 million homes in Europe and North America will be smart by 2019. Online, M2M Research Series.

[8] Opternus (2015). Opternus Components. Online.

[9] R F Boehm, H. Y. e. J. Y. (2015). *Handbook of clean energy systems*, volume 5º volume. Chichester, West Sussex.

[10] Rudolph, S. (2015). Mobile Apps Usage – Statistics and Trends. Online.

[11] Statista (2014). Number of mobile app downloads worldwide from 2009 to 2017 (in millions). Online.

Anexo A

Esquemáticos

A.1 Esquemático Alfred

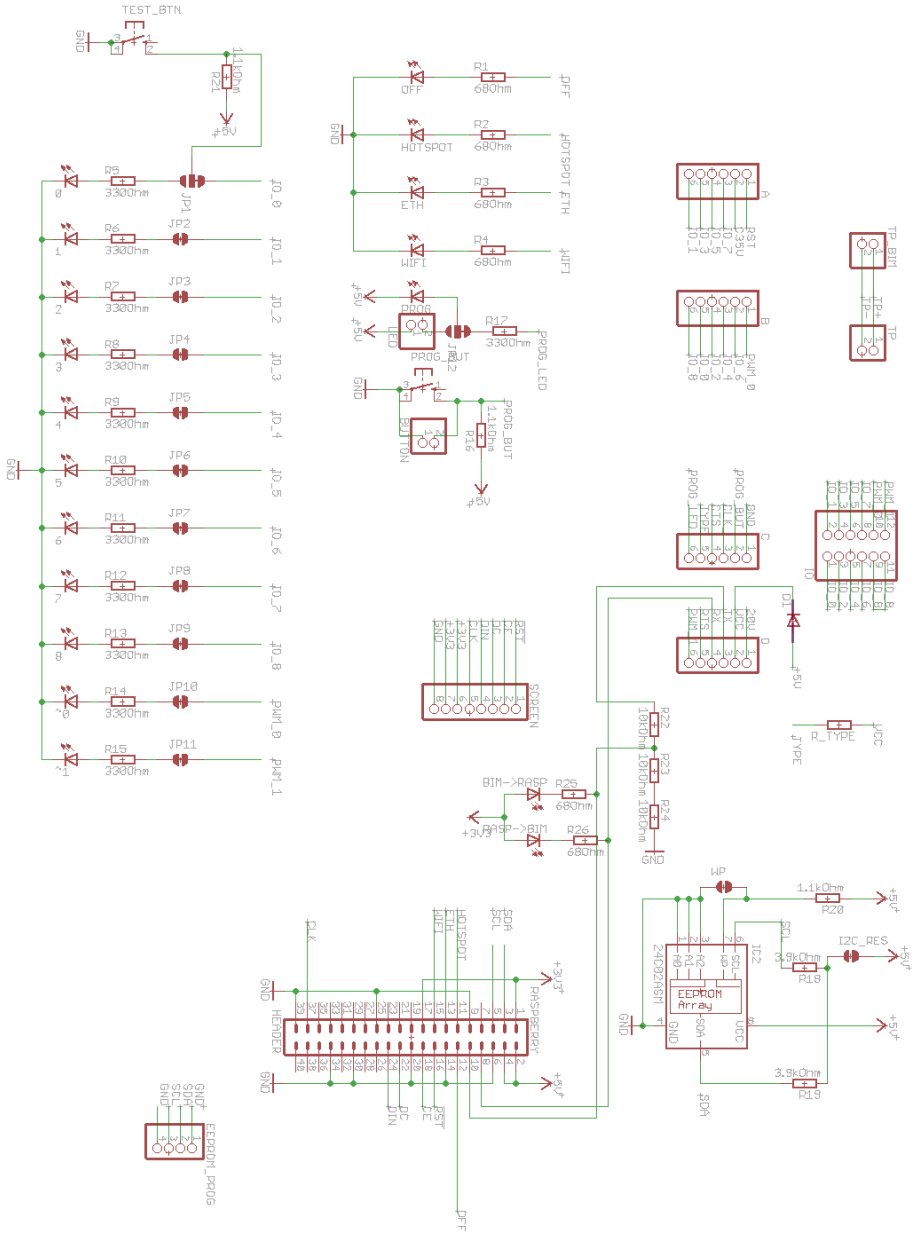

Figura A.1 Esquemático Alfred

A.2 PCB Alfred

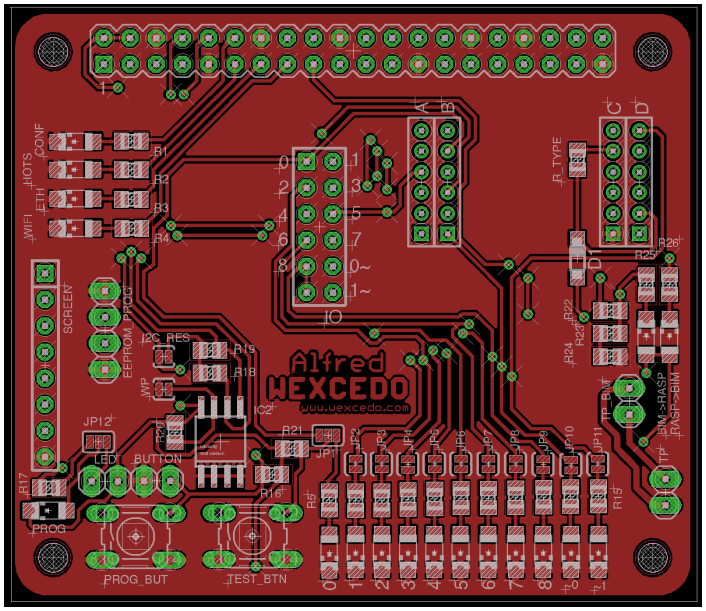

Figura A.2 PCB Frente - Alfred

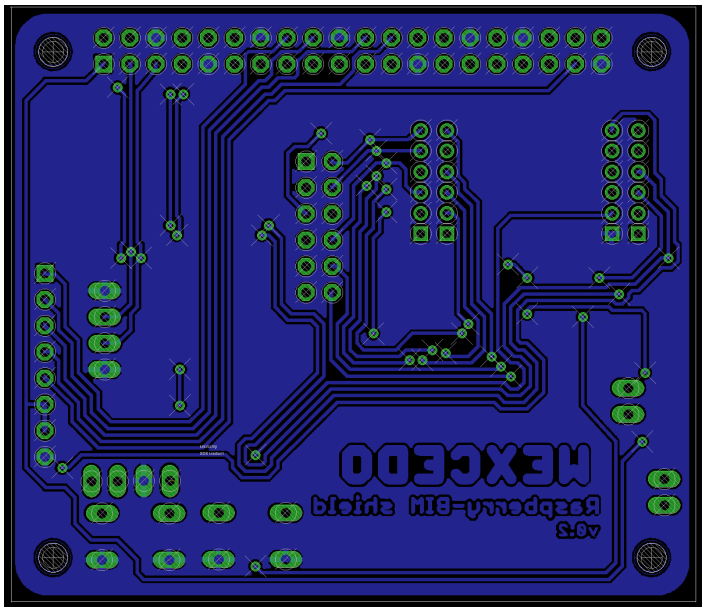

Figura A.3 PCB Trás - Alfred

A.3 Esquemático Gravador EEPROM

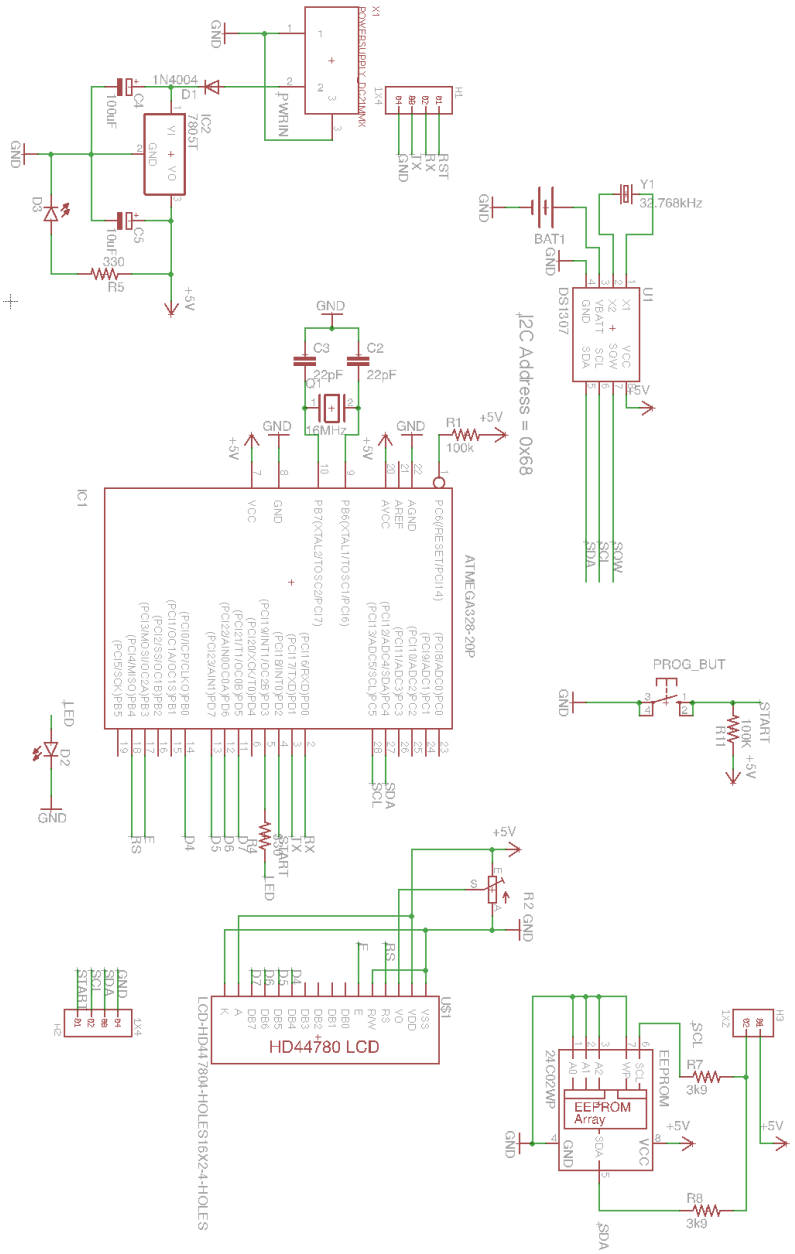

Figura A.4 Esquemático Gravador EEPROM

A.4 PCB Gravador EEPROM

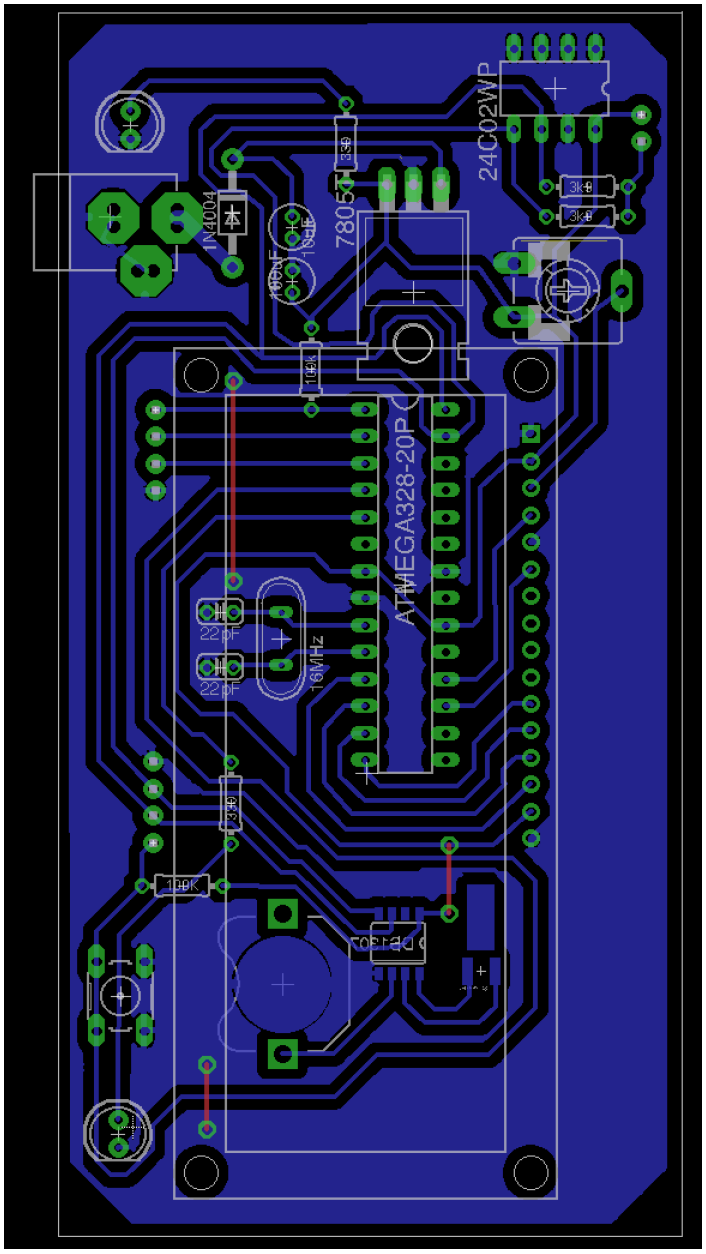

Figura A.5 PCB Frente - Gravador EEPROM

Anexo B

Tutoriais

B.1 Raspberry Pi - Configuração do Hotspot

B.1.1 Ficheiros modificados:

- /etc/network/interfaces
- /etc/dhcp/dhcpd.conf
- /etc/default/isc-dhcp-server
- /etc/hostapd/hostapd.conf
- /etc/default/hostapd
- /etc/default/ifplugd

B.1.2 Passos

Instalações

Instalação dos *drivers* da pen WIFI:

```
wget http://www.daveconroy.com/wp3/wp-content/uploads/2013/07
/hostapd.zip
unzip hostapd.zip
sudo mv /usr/sbin/hostapd /usr/sbin/hostapd.bak
sudo mv hostapd /usr/sbin/hostapd.edimax
sudo ln -sf /usr/sbin/hostapd.edimax /usr/sbin/hostapd
sudo chown root.root /usr/sbin/hostapd
```

```
sudo chmod 755 /usr/sbin/hostapd
```

De seguida, instalar servidor de *DHCP* e serviço de *Hotspot*:

```
sudo apt-get install hostapd isc-dhcp-server
```

Definir IP estático na placa WIFI (wlan0)

Editar o ficheiro */etc/network/interfaces* (usando o *nano* por exemplo):

```
auto wlan0
allow-hotplug wlan0

iface wlan0 inet static
address 192.168.42.1
netmask 255.255.255.0
gateway 192.168.42.1
```

Configurar o servidor de *DHCP*

Editar o ficheiro */etc/dhcp/dhcpd.conf* e alterar:

```
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
#authoritative;
```

para:

```
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;
authoritative;
```

e adicionar:

```
DHCPDARGS=wlan0;

subnet 192.168.42.0 netmask 255.255.255.0 {
range 192.168.42.10 192.168.42.50;
option broadcast-address 192.168.42.255;
option routers 192.168.42.1;
default-lease-time 600;
```

```
max-lease-time 7200;
option domain-name "local";
option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

Editar o ficheiro */etc/default/isc-dhcp-server* e alterar:

```
INTERFACES=""
```

para:

```
INTERFACES="wlan0"
```

B.1.3 Configurar o Hotspot

Editar o ficheiro */etc/hostapd/hostapd.conf* e alterar:

```
interface=wlan0
driver=rtl871xdrv
#bridge=br0
ssid=WexcedoAP
channel=1
wmm_enabled=0
wpa=1
wpa_passphrase=wexcedo
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
auth_algs=1
macaddr_acl=0
```

Editar o ficheiro */etc/default/hostapd* e alterar:

```
#DAEMON_CONF=""
```

para:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

B.1.4 Testar o Hotspot

```
sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf
```

B.1.5 Correr como serviço

```
sudo service hostapd start
sudo service isc-dhcp-server start
```

B.1.6 Correr no arranque

```
sudo update-rc.d hostapd enable
sudo update-rc.d isc-dhcp-server enable
```