Universidade de Aveiro Departamento de Engenharia Mecânica
2016

Ricardo Jorge Gonçalves Duarte Pires

Desenvolvimento de módulos de controlo/aquisição local baseados em transdutores wi-fi

Universidade de Aveiro Departamento de Engenharia Mecânica
2016

Ricardo Jorge Gonçalves Duarte Pires

Desenvolvimento de módulos de controlo/aquisição local baseados em transdutores wi-fi

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica do Prof. Doutor José Paulo de Oliveira Santos, Prof. Auxiliar do Departamento de Engenharia Mecânica, da Universidade de Aveiro.

O júri / The jury

Presidente / President

Prof. Doutor Jorge Augusto Fernandes Ferreira
Prof. Auxiliar do Departamento de Engenharia Mecânica, da Universidade de Aveiro da Universidade de Aveiro.

Vogais / Committee

Prof. Doutor José Paulo de Oliveira Santos
Prof. Auxiliar do Departamento de Engenharia Mecânica, da Universidade de Aveiro (Orientador).

Prof. Doutor Arnaldo Silva Rodrigues de Oliveira
Prof. Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática, da Universidade de Aveiro (Arguente Principal).

Agradecimentos / Acknowledgements

Gostaria de começar por agradecer à Universidade de Aveiro, a excelente formação que me proporcionou, durante o meu percurso académico.

De seguida, gostaria de agradecer ao meu Orientador, Professor Doutor José Paulo de Oliveira Santos, o apoio demonstrado em especial durante a realização da presente dissertação e também durante as unidades curriculares que lecionou e que fazem parte do plano curricular do Mestrado Integrado em Engenharia Mecânica.

Gostaria também de agradecer à empresa T&T Multielétrica, Lda, a qual foi responsável pelo tema que escolhi para a minha dissertação.

Agradeço profundamente a todos os meus amigos, toda a amizade, cooperação, espírito de equipa e união, que largamente contribuíram para que tenha tido um percurso académico maravilhoso e que proporcionou aprendizagens únicas.

Finalmente, agradeço com muito amor à minha família, que em todos os momentos esteve sempre presente, demonstrando o seu apoio constante.

A TODOS, MUITO OBRIGADO!

Palavras-chave

Domótica, ESP8266, Automação, Comunicação RF, Serviços *web*, Sistema de monitorização e controlo.

Resumo

A domótica tem vindo a aumentar a sua presença nas habitações dos nossos dias, com o objetivo de melhorar o bem-estar dos seus utilizadores, contribuindo em simultâneo para a minimização dos consumos energéticos nos edifícios. Apesar da domótica ainda não estar implementada na maioria das residências, o número de casas inteligentes tem vindo a crescer.

A evolução das tecnologias, tem dado um forte contributo para o crescimento da domótica. No ramo da domótica, o uso de transdutores e módulos de aquisição/controlo local têm como função manter a operabilidade do edifício inteligente. Portanto, é no estudo destes dispositivos e na sua implementação, que o autor foca a sua atenção, na presente dissertação.

A solução e implementação propostas tem como dispositivo principal o ESP8266, que através da comunicação sem fios, IEEE 802.11, permite monitorizar e controlar remotamente os diversos dispositivos presentes na habitação.

A solução proposta e sua implementação recorrem a servidores WEB e a bases de dados MySQL, que são acedidas para fins de controlo dos dispositivos da habitação. A implementação dos nós de controlo recorre ao ESP8266, que por sua vez, incorpora os programas desenvolvidos em LUA, pelo autor. Para efeitos de controlo por parte do utilizador, foi também criado um programa em linguagem C#.

Keywords Home automation, ESP8266, Automation, RF Comunication, Web services, Monitoring and Control System.

Abstract

Domotics has been increasing its presence inside people's homes, with the goal of making their life's more comfortable and, at the same time, creating new ways to reduce the consumption of energy, inside buildings. In spite of domotics it's not yet implemented in the most residences, the number of smart homes has been growing.

The evolution of informatics, has been had an important role to the domotics evolution as well. In home automation, the use of transductors and local acquisition/control modules have the function to maintain the operability of the smart home. Thus, it's in the study of this devices, that the author aims his attention, in this master thesis.

The solution and the implementation have as main device the ESP8266, which through wireless network, IEEE 802.11, allow to monitoring and control the devices remotely, which control the habitation.

The proposed solution and its implementation use a WEB server and MySQL databases, which can be used to control home devices. The control nodes implementation use the ESP8266 modules, which are programmed in LUA, by the author. To establish control between the user and the building, was developed an application in C# language.

Conteúdo

Lista de Tabelas

Lista de Figuras

Glossário

Buffer

Região de memória usada para armazenar dados temporariamente.

Câmaras IP

Tipo de câmara que comunica com um sistema NVR (*Network Video Recorder*) por sinal digital. As câmaras estão associadas a uma *network* e são identificadas pelo seu próprio IP. Relativamente às câmaras de sinal analógico, não necessitam de conversão de sinal, permitem comunicação bidirecional e obter gravações com maior resolução.

Drop-in

Um dispositivo com compatibilidade *drop-in* é aquele que pode ser trocado pelo anterior sem necessidade de efetuar quaisquer alterações para que se torne compatível.

Handheld devices

Designação usada para referir dispositivos que foram desenhados para usar sobre a palma da mão.

Stack

Estrutura de dados que contém uma lista de elementos, organizados cronologicamente do fundo para o topo, em que o elemento do topo é o mais recente.

Lista de Acrónimos

A *Ampere.*

AM *Aplication Module.*

AP *Access Point.*

API *Application Programming Interface.*

AVAC Aquecimento Ventilação e Ar Condicionado.

BCI Batibus Club International.

BCU *Bus Coupler Unit.*

bps *Bits per second.*

CAL *Common Application Language.*

CN *Child Nodes.*

EHS *European Home Systems.*

EHSA *European Home Systems Association.*

EIB *European Instalation Bus.*

EIBA *European International Business Academy.*

ESB *Enhanced ShockBurst.*

FIFO *First In, First Out.*

GPIO *General Purpose Input Output.*

HM2M *Home Machine-To-Machine.*

HMMP *Home Machine-To-Machine Protocol.*

HNASC *Home Network Applications Service Center.*

HTML *HyperText Markup Language.*

Hz *Hertz.*

IAGW *Integrated Access Gateway.*

IP *Internet Protocol.*

IPS *Infrared Proximity Sensor.*

IR *Infrared.*

ISM *Industrial, Scientific and Medical.*

LVTTL *Low-voltage Transistor-transistor Logic.*

m Metro.

M2M *Machine-To-Machine.*

MAC *Media Access Control.*

P2P *Peer-to-peer.*

PC *Personal Computer.*

PHP *Hypertext Preprocessor.*

QoS *Quality of Service.*

RF *Radio Frequency.*

RJ45 *Registered Jack Interface Standard Number 45.*

SCR *Silicon Controlled Rectifier.*

SPI *Serial Peripheral Interface.*

TCP *Transmission Control Protocol.*

TRIAC *TRIode for Alternating Current.*

UART *Universal Asynchronous Receiver/Transmitter.*

USN *Ubiquitous Sensor Networks.*

V *Volt.*

WCS *Wireless Control Socket.*

Capítulo 1

Introdução

1.1 Enquadramento

Domótica é um termo derivado da união da palavra "domus" e da palavra "robótica", que remete para o conceito de "casa automatizada" [1]. Efetivamente, a domótica surgiu no início da década de 70, com vista a revolucionar as habitações existentes na altura, a torná-las mais cómodas para os seus habitantes, recorrendo a dispositivos eletrónicos, que lhes concederam um determinado grau de automatização, permitindo estabelecer a comunicação, por diversos meios, entre as mesmas e os seus utilizadores. Deste modo foi alcançado um novo domínio de tecnologia, que conseguiu combater as necessidades de maior conforto e racionalização energética dos edifícios. Por conseguinte, esta tecnologia veio abrir as portas a novas ideias da sua aplicabilidade, nomeadamente o aumento dos níveis de segurança das habitações, através da sua constante monitorização, quer no caso da segurança relacionada com o controlo de acessos, quer no caso do estado dos equipamentos integrados na casa, como por exemplo, a deteção de uma deficiência no sistema de climatização. Continuando a abordagem da domótica, esta abrange igualmente, a manipulação de sistemas de iluminação, sistemas de rega, aquecimento de águas de piscinas, abertura e fecho de estores e a integração de todos estes sistemas num único local, facilitando o controlo geral da habitação por parte do utilizador. Esta tecnologia tem crescido gradualmente em conformidade com a forte evolução tecnológica que se tem feito sentir ao longo dos anos o que acaba por contribuir para o aumento da celeridade da própria evolução da domótica e também alargá-la a novos horizontes, aumentando cada vez mais o número de novos conceitos, que se têm focado principalmente na comodidade dos habitantes.

1.2 Problema

Apesar da crescente evolução da domótica existem ainda alguns fatores, que fazem com que os consumidores optem por não automatizar as suas habitações. Um deles é o preço de aquisição dos dispositivos. As soluções existentes no mercado revelam-se satisfatórias do ponto de vista da qualidade, mas não do ponto de vista dos custos, facto inerente à exclusividade dos sistemas atuais, que não possibilitam a comunicação entre dispositivos de marcas diferentes e detêm assim um elevado poder de negociação perante os clientes. Para além disso, todos os serviços intrínsecos ao equipamento, em particular

a manutenção, são assegurados pelos mesmos, diminuindo assim a margem de manobra dos clientes, que se veem forçados a optar por uma determinada solução e ficarem ligados a ela eternamente, sujeitos aos seus termos e serviços.

O consumo energético dos próprios sistemas de domótica continua a ser alvo de estudo, com vista à sua diminuição, pois apesar do grande corte nos consumos energéticos que se apresentam nas casas convencionais, os tempos de crise são ainda um contratempo para investir em equipamentos dispendiosos. Portanto, é necessário evoluir no sentido do uso de equipamentos de baixo consumo e baixo preço de aquisição. Para além disso, o volume dos equipamentos existentes torna-se incómodo e esteticamente feio, sendo necessária uma vasta panóplia de acessórios, caso se pretenda usufruir de um controlo completo da casa.

Apesar do crescente tecnológico cada vez mais proeminente, ainda existe uma larga margem de manobra para inovar os sistemas contemporâneos.

1.3 Objetivos

Esta dissertação tem por objetivo desenvolver módulos de comunicacao wifi, que permitam adquirir os dados dos sensores de habitações e que tenham aplicação na área de domótica. Esses sensores devem cumprir os seguintes princípios: baixo custo de aquisição, versatilidade e compatibilidade com os sistemas já existentes, baixo impacto ambiental e discreto do ponto de vista estético, sem afetar a sua funcionalidade, sendo passível de aplicar a uma tecnologia de comunicação comum nos sistemas informáticos contemporâneos. Estes princípios são a base de todo o desenvolvimento deste projeto cuja solução proposta terá de satisfazer, para assegurar a sua integridade ao nível da sua aplicabilidade face aos sistemas atuais.

1.4 Estrutura da dissertação

A presente dissertação apresenta-se dividida em 5 capítulos acrescida de 2 anexos. Tendo iniciado com uma breve introdução ao conceito de domótica, na qual foi feito um enquadramento deste domínio tecnológico, seguido da apresentação de alguns problemas que ela pretende minimizar e também dos objetivos, que se pretendem alcançar no âmbito desta dissertação. No segundo capítulo, serão descritas, com o nível de detalhe necessário à sua compreensão, as tecnologias existentes no mercado atualmente, particularizando alguns detalhes específicos das mesmas, para no final identificar os problemas por elas levantados. No capítulo três serão abordadas as tecnologias que farão parte da solução proposta, que é referida no capítulo quatro, juntamente com a sua implementação, descrevendo o seu modo de funcionamento, por forma a validar a sua utilidade em aplicações no ramo da domótica. Finalmente, seguir-se-á uma conclusão que a analisará o desempenho da solução proposta para o problema apresentado.

Capítulo 2

Revisão do estado da arte

2.1 *Smart home*

As *smart homes* (casas inteligentes) nasceram em 1975 com o objetivo de revolucionar a vida quotidiana do cidadão. As casas inteligentes caracterizam-se por incorporarem uma rede de comunicação entre dispositivos, que a ela se liguem. Consoante o sistema com o qual se está a comunicar e o(s) protocolo(s) utilizado(s), esta rede pode usar como recursos, a própria rede elétrica, comunicações IR, ou ainda RF. Através das redes de domótica o utilizador pode controlar, quer localmente, quer remotamente, determinados elementos da casa, como por exemplo: sistemas de rega, iluminação, sistemas AVAC, segurança, os estores das janelas, entre outros dispositivos físicos.

2.2 Tecnologias existentes

Atualmente, existem inúmeras soluções comerciais, baseadas em vários protocolos criados no âmbito da domótica, para sistemas de automação de edifícios. Existem sistemas mais vocacionados para o mercado norte-americano, baseados nas tecnologias: X10, CEBus (já descontinuado), LonWorks, Smart House, e sistemas inicialmente desenvolvidos na Europa, como o Batibus, o EIB e o EHS. Existem também sistemas desenvolvidos em Portugal, tais como o VIVIMAT, Domus-int e o CARDIO, capazes de oferecer a monitorização e o controlo da instalação de domótica via *web*.

Neste capítulo, serão abordadas com maior destaque as tecnologias: X10, KNX, e LonWorks, uma vez que são aquelas que mais influência tiveram dentro do âmbito da domótica.

2.3 X10

O X10 é o protocolo mais antigo e usado nas aplicações de domótica. O seu desenvolvimento iniciou-se em 1975 pela empresa Pico Electonics, Ltd., em Glenrothes, na Escócia e em 1978 ficaram disponíveis os primeiros produtos com tecnologia X10 [2]. Esta tecnologia permite transmitir dados pela rede elétrica (110V nos EUA e 230V na Europa) com baixa cadência (60Hz nos EUA e 50Hz na Europa) e custos reduzidos.

2.3.1 Cablagem

O protocolo X10 comunica entre transmissores e recetores através do envio de sinais injetados diretamente na rede elétrica. Existem dispositivos que podem receber a informação por RF e convertê-la em informação X10, enviada através de um comando de mão, semelhante a um comando de TV, como se observa na figura 2.3, nomeadamente o modelo "8-in-1 universal remote control".

A vantagem da utilização da rede elétrica consiste em não ser necessária a colocação de novos cabos, focando-se na reutilização dos recursos já existentes evitando custos acrescidos e tornando-a assim como o principal meio de comunicação. Para além disso, a facilidade de implementação, aproveitando os meios já existentes e a possibilidade de utilizar uma arquitetura descentralizada constituem alguns dos aspetos que contribuíram para o seu sucesso. No entanto, é necessária a aquisição do equipamento de comunicação, representado nas figuras: 2.2, 2.3 e 2.5.

2.3.2 Equipamentos existentes

O sistema X10, suporta-se na rede elétrica já existente na habitação para injetar sinais de elevada frequência [3], especificamente uma sinusoide de 120kHz durante 1ms, quando a tensão da rede elétrica passa por 0 volts. Esses sinais são usados para comunicar entre os seus sistemas, sem interferir com os restantes equipamentos elétricos, que compõem a habitação.

Associados ao protocolo X10 estão diversos tipos de dispositivos, que comunicam por diferentes tipos de sinais. Os dispositivos X10 agrupam-se em 3 diferentes tipos: controladores, controladores sem fios e módulos. Os controladores têm como função enviar um endereço e um comando pela *powerline* para controlar os módulos. Os controladores sem fios têm como finalidade transmitir um sinal à central de receção, que por sua vez, transforma esse sinal em sinais X10 para serem transmitidos pela *powerline*. Os módulos controlam os equipamentos aos quais se ligam, ao receberem comandos vindos da *powerline* e dividem-se em 4 tipos diferentes: módulos *plug-in*, *built-in switches*, *micro modules*, e módulos para *DIN rail assembly* (figura 2.1).

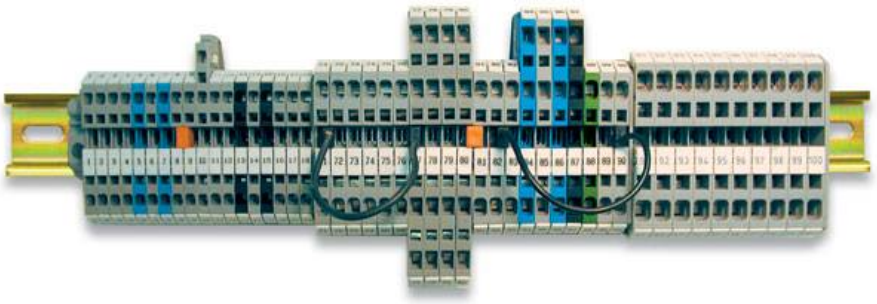


Figura 2.1: Exemplo de montagem *DIN rail* [4]

Considere-se que se quer controlar uma lâmpada através do LM12 e do MT10E (ver referências em anexo), representados na figura 2.2. O MT10E envia um comando pela *powerline* para o módulo LM12, que é descodificado pelo próprio LM12. Depois do LM12 verificar o endereço do comando contido na mensagem que lhe foi enviada executa a instrução.

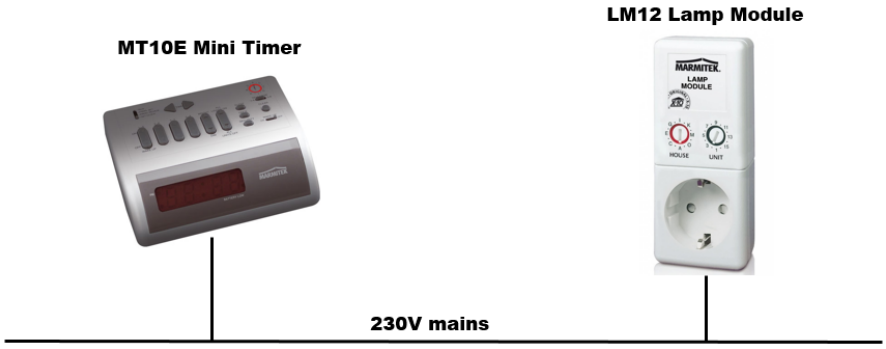


Figura 2.2: Ligação do *Mini Timer* ao *Lamp Module* (MT10E e LM12 são referências da Marmitek)

Este controlo também pode ser efetuado remotamente, através do uso do *8-in-1 Universal Remote Control*. No entanto, este transmissor requer uma conversão de sinal assegurada pelo TM13 (*Transceiver Module*). Como representado na figura 2.3, consoante o fim pretendido, os sistemas X10 possuem diversas soluções ao nível dos dispositivos anteriormente referidos. Caso se pretenda acrescentar a função de controlo de intensidade da luz das lâmpadas, o módulo LM12 pode ser substituído pelo LW11 e para sistemas exteriores, nomeadamente o controlo de bombas de água, o X10 oferece o AD10 DIN .

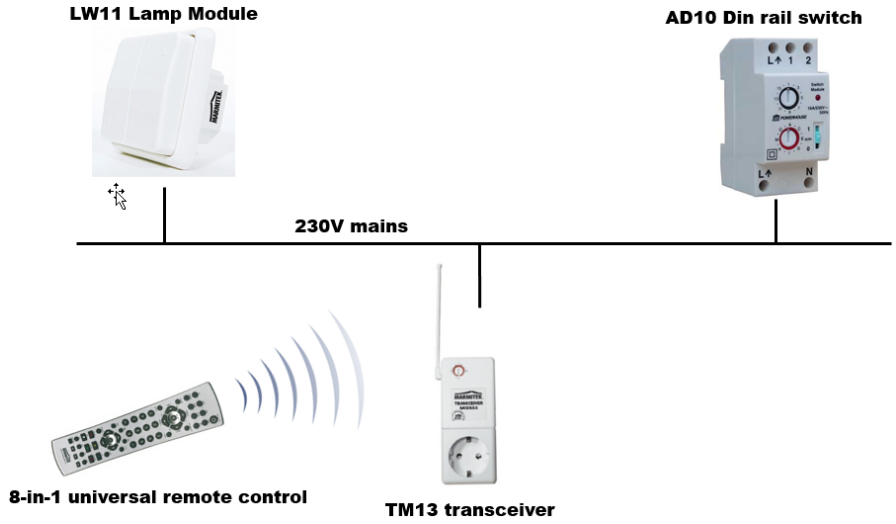


Figura 2.3: Ligação de vários módulos X10 com comunicação remota (TM13, LW11 e AD10 são referências da Marmitek)

A integração de um PC é uma outra propriedade deste protocolo, com a utilização do CM11 *(Computer Interface)* é permitido ao utilizador pré-programar ações a serem tomadas pelo sistema, por outras palavras, pode-se definir um botão de pressão, um horário, ou uma mensagem remota, que quando atuados executam a instrução pré-definida. Desta forma, o computador passa a ser um elemento crucial no fácil controlo por parte do utilizador, pois permite-lhe guardar uma série de instruções e criar programas de instruções que evitam a constante programação dos dispositivos um a um.

Por fim, como solução menos automatizada, o X10 também pode ser atuado pelos *standard switches*, dispositivos mais simples nos quais os utilizadores atuam diretamente de forma semelhante a um interruptor.

2.3.3 Tipo de sinais

Os sinais de alta frequência, que são injetados na rede elétrica, representam valores binários. A inclusão desses sinais é feita logo após a passagem da onda sinusoidal de 50 Hz pela origem, obtendo no máximo um atraso de 200 micros-segundos. Posteriormente, o sinal é enviado pela rede elétrica para os módulos recetores. Na figura 2.4 está ilustrada uma onda sinusoidal com a introdução de um sinal X10 [5].

De um modo sucinto, os controladores enviam códigos/sinais pela rede elétrica dirigidos aos módulos recetores que se pretendem controlar, sendo necessário definir os endereços quer dos controladores, quer nos módulos recetores.

Existe também a possibilidade de vários módulos recetores partilharem o mesmo endereço, constituindo uma zona, ou seja, atuam em simultâneo à mesma mensagem proveniente do módulo emissor.

Inicialmente é transmitida uma mensagem, que pede uma resposta aos dispositivos que contenham o código casa nela registado, permitindo saber quais os dispositivos que se encontram à escuta. Assim, é atribuído um código casa diferente, no caso de existir uma resposta de um comando anterior, *hail acknowlegde*. No caso particular das instruções predefinidas, para a instrução que baixa a luminosidade, o *bit* D8 representa o *bit* mais significativo, enquanto que os restantes (H1, H2, H4 e H8) representam os menos significativos. Os restantes dados seguem em *bytes* (8 bits), que definem os dados analógicos, depois de ser feita a conversão analógica/digital. Note-se que não devem existir intervalos entre os dados seguintes e os atuais, nem entre os *bytes* de dados, pois caso isso aconteça, as mensagens podem ser mal interpretadas, causando o mau funcionamento dos sistemas X10 [5].

O envio de dados é sincronizado nos pontos de cruzamento das sinusoides com uma linha de referência (*AC power line*), ou seja, através dos chamados *zero crossing points* [3], observável na figura 2.4. Para representar os valores binários 1 e 0, no caso do primeiro, é injetado um sinal para o ponto de cruzamento com o zero, cuja frequência é de 120kHz e duração de um milissegundo, ao passo que um 0 se representa pela ausência dos 120kHz. Os envios de 1ms devem ser igualmente transmitidos três vezes, de forma a coincidir com o *zero crossing point* das três fases. Segundo a figura 2.4, essas ocorrências registam-se aos 0ms, 2.778ms e 5.556ms, retomando-se aos 8.333ms no segundo ciclo da primeira fase. Assim se conclui, que existe um intervalo constante de tempo de 2.778ms bem definido entre fases.

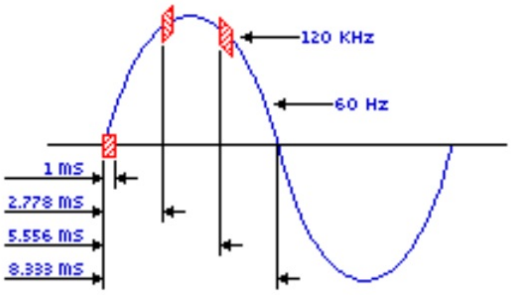


Figura 2.4: Onda de transmissão de dados em sistemas X10 [5]

2.3.4 Estrutura da mensagem

Para explicar a estrutura das mensagens X10 é necessário, numa primeira fase, compreender o significado de cada bloco, que elas contêm. Um dos blocos contém o chamado *House Code* (código casa), que é atribuído a um controlador e tem até um máximo de 16 endereços, correspondente ao comando de 16 grupos dos módulos recetores. Por seu lado, os recetores além do *House Code*, são endereçados pelo *Unit Code* (código de unidade) que corresponde ao número da zona de que o módulo faz parte [5].

Os módulos da figura 2.5 apresentam um conjunto de 16 letras (A-P) e 16 números por letra, no que resulta um total de 256 combinações possíveis.

Figura 2.5: Módulo de configuração dos endereços X10

A transmissão completa do código subdivide-se em 11 ciclos da linha de potência representados na figura 2.6.

Os primeiros dois ciclos representam o *start code*, os 4 seguintes o *house code* e os restantes podem representar, ou o *number code*, ou o *function code*, consoante o dispositivo a quem se destina a mensagem. À exceção do controlo da luminosidade (brilho e escurecimento), que é feito de forma contínua, as restantes operações são transmitidas em grupos de 2, com 3 ciclos de potência entre cada grupo de 2 códigos, como é mostrado na figura 2.7.

Relativamente aos códigos que seguem na mensagem, apresentados na tabela 2.1, cada um deles (código de 4 ou 5 bits) é transmitido em *half cycles* da linha de potência. Isto significa que na primeira metade do ciclo, caso seja transmitido um *bit* a 1, o *bit* da segunda metade do ciclo será obrigatoriamente 0, o seu complemento. A figura 2.6 tem como objetivo facilitar a compreensão desta regra. Todavia, no caso do código casa,

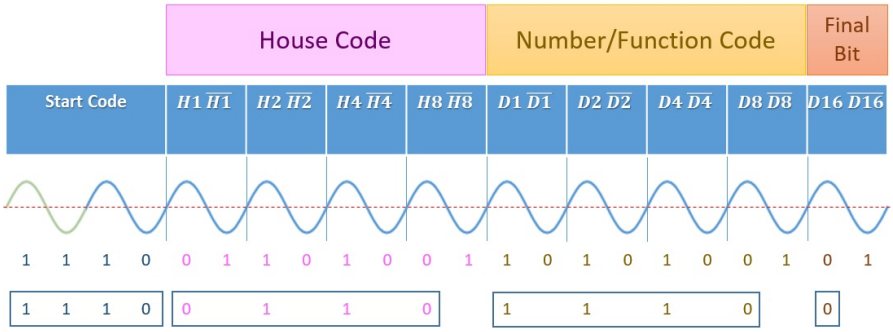


Figura 2.6: Estrutura da mensagem X10

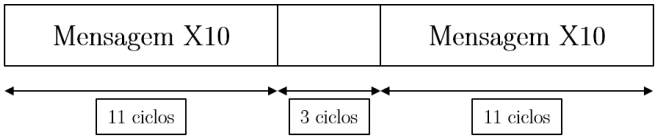


Figura 2.7: Envio de mensagem X10

este permanece constante com o valor "1110", tratando-se de um código único e que por conseguinte, não obedece à regra anterior.

Tabela 2.1: Tabela de códigos do sistema X10

	HOUSE CODES					KEY CODES				
	H1	H2	H4	H8		D1	D2	D4	D8	D16
A	0	1	1	0	1	0	1	1	0	0
B	1	1	1	0	2	1	1	1	0	0
C	0	0	1	0	3	0	0	1	0	0
D	1	0	1	0	4	1	0	1	0	0
E	0	0	0	1	5	0	0	0	1	0
F	1	0	0	1	6	1	0	0	1	0
G	0	1	0	1	7	0	1	0	1	0
H	1	1	0	1	8	1	1	0	1	0
I	0	1	1	1	9	0	1	1	1	0
J	1	1	1	1	10	1	1	1	1	0
K	0	0	1	1	11	0	0	1	1	0
L	1	0	1	1	12	1	0	1	1	0
M	0	0	0	0	13	0	0	0	0	0
N	1	0	0	0	14	1	0	0	0	0
O	0	1	0	0	15	0	1	0	0	0
P	1	1	0	0	16	1	1	0	0	0

Para além dos códigos atrás mencionados existem outros códigos de instrução que servem para enviar ordens específicas aos dispositivos, como expressa a tabela 2.2.

Tabela 2.2: Códigos de instrução [5]

Instrução	**Código Casa**				
Todas as unidades desligadas	0	0	0	0	1
Todas as unidades ligadas	0	0	0	1	1
Ligado (ON)	0	0	1	0	1
Desligado (OFF)	0	0	1	1	1
Escurecer (Dim)	0	1	0	0	1
Iluminar (Bright)	0	1	0	1	1
Todas as luzes desligadas	0	1	1	0	1
Extended Code	0	1	1	1	1
Hail Request	1	0	0	0	1
Hail Acknowledge	1	0	0	1	1
Pre-Set Dim	1	0	1	X	1
Extended Data (Analog)	1	1	0	0	1
Status = ON	1	1	0	1	1
Status = OFF	1	1	1	0	1
Status Request	1	1	1	1	1

2.3.5 Vantagens e Desvantagens

A principal desvantagem deste protocolo relaciona-se com a baixa velocidade de comunicação (relativamente aos concorrentes) e com o facto de não haver confirmação da chegada da mensagem, por parte dos dispositivos recetores. O ruído é também um senão desta tecnologia, que tem vindo a ser solucionado com a evolução dos protocolos A10 e S10. Para além destas, os sistemas X10 apresentam um leque de funções limitadas, que se restringem ao ligar/desligar e controlo da luminosidade. Não obstante, graças ao amadurecimento (mais de 30 anos de mercado) e à tecnologia implementada, os produtos X10 têm um preço muito competitivo no mercado residencial norte-americano. Esta tecnologia está de tal maneira enraizada que as suas instalações podem ser realizadas por eletricistas sem conhecimento na área de automação ou de informática, ou até pelos próprios utilizadores. Pode ser aplicado a habitações já construídas.

2.4 KNX

O protocolo KNX surgiu em maio de 1999, quando membros da EIBA, EHSA, e BCI, se reuniram para fundar a KNX Association citepknx, cujo logótipo se representa na figura 2.8. Esta empresa é líder mundial no que respeita a sistemas de controlo de

edifícios e casos inseridos na área da domótica. A KNX está associada a 44000 parceiros distribuídos por 125 países em todo o mundo.

Figura 2.8: Logótipo da KNX Association [6]

Este protocolo foi criado com o objetivo de substituir as soluções que existiam, desenvolvendo um padrão europeu, que permitisse a comunicação entre todos os dispositivos de uma habitação, ou de um edifício de serviços.

O sistema KNX é composto por uma unidade de acoplamento ao barramento, BCU, e por um módulo de aplicação, AM. A BCU encarrega-se de armazenar os endereços físicos e os parâmetros dos nós, comunicando diretamente com a rede, ao passo que a AM se incumbe da interface com o utilizador recorrendo a: botões de pressão, sensores de presença, sensores de temperatura, entre outros (figura 2.9).

Figura 2.9: BCU à esquerda e à direita um sensor de presença usado em sistemas KNX [7]

2.4.1 Cablagem e topologia

A rede elétrica do edifício é usada para alimentar os componentes físicos (par entrelaçado e linha de potência) e o *gateway* é usado para comunicar via RF com o barramento de comunicação. Associado ao KNX está um *software* desenvolvido especificamente para plataformas *Windows*.

KNX/EIB

Uma rede KNX/EIB pode albergar até 65536 dispositivos com endereços individuais de 16 bits. Esses endereços subdividem-se em endereços de grupo e endereços individuais. A denotação é feita com dois pontos e números decimais (X.X.X), onde X representa os números decimais separados por ponto (dois no total). O primeiro número decimal define o endereço de área e contém 4 bits, os mesmos que o segundo, que representa o endereço da linha ou sub-rede, enquanto que o último número decimal, agora com 8 bits, caracteriza o endereço do dispositivo. Desta forma, é possível obter no final 2^8 combinações de endereços, portanto 256 [6].

Como se observa na figura 2.10 existem 15 áreas ao todo, cada uma com uma *main line*, à qual se ligam as diversas linhas. Cada linha pode ter 256 endereços e para definir uma linha sabe-se que são necessários 4 bits, portanto têm-se no total $2^4 - 1$ linhas, uma vez que o "0" é reservado para o acoplador entre a linha principal e a *backbone*. Sem os endereços reservados para os acopladores, existem então 255 endereços × 16 linhas × 15 áreas, perfazendo um total de 61200 dispositivos passíveis de ligação à rede KNX [6].

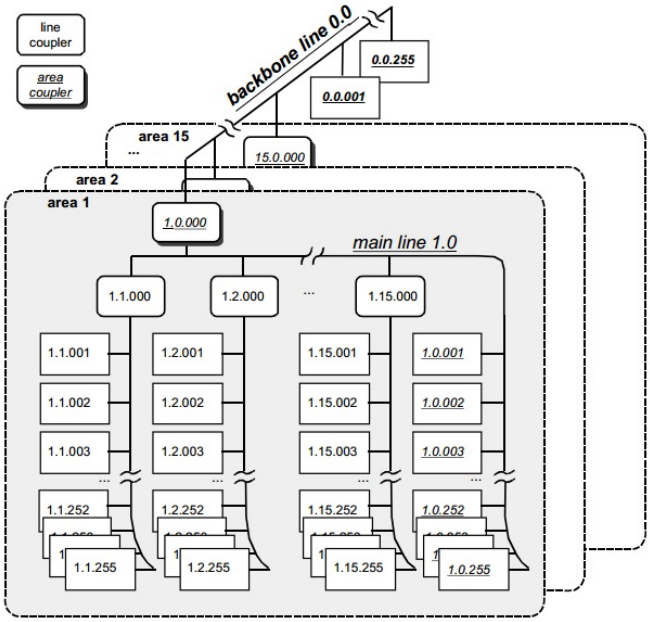


Figura 2.10: Estrutura de uma rede KNX [8]

Modos de comunicação [6]

Dentro dos sistemas KNX existem 6 formas de comunicação:

- TP-0
- TP-1
- PL-110
- PL-132
- RF
- Ethernet

Para facilitar a sua compreensão sem alargar muito em texto, as suas características foram resumidas na tabela 2.3.

Tabela 2.3: Meio de comunicação no KNX

Designação	Tipo	Bitrate	Origem	Troca de informação
TP-0	Par entrelaçado, tipo 0	4800 bits/s	BatiBUS	Opera na mesma linha de barramento do BatiBUS, mas não permite troca de informação entre os seus componentes
TP-1	Par entrelaçado, tipo 1	9600 bits/s	EIB	Opera na mesma linha de barramento do BatiBUS comunica entre KNX TP1 e EIB
PL−110	Powerline, 110 kHz	1200 bits/s	EIB	Opera na mesma rede elétrica e comunica entre KNX PL110 e EIB
PL−132	Powerline, 132 kHZ	2400 bits/s	EHS	KNX PL132 e EHS operam em conjunto, mas não comunicão entre si, sem um protocolo de conversão
RF	Radiofrequência a 868 MHz	38.4 kbits/s		Opera dentro da própria "framework" do padrão KNX
Ethernet	KNX através de IP			"Tunneling" entre "frames" do KNX encapsuladas nas "frame" IP

2.4.2 Um exemplo de implementação KNX

Na figura 2.11, encontra-se esquematizada uma implementação de um sistema baseado em KNX. Descrevendo resumidamente esse sistema, dele fazem parte unidades UP para serem atuadas como interruptor, atuadores de persianas (*blind actuators*) e componentes KNX RF para controlar: a iluminação, sistemas de aquecimento e persianas; sensores de presença para atuar sobre a iluminação dos compartimentos e que, em simultâneo, informam acerca da presença de indivíduos nos mesmos. O sistema possui um controlador da temperatura que pode ser atuado por um dispositivo móvel ou manualmente. Para funcionar, o sistema requer *routers* IP, fontes de alimentação, *gateways* e WLAN *nodes*. Como se pode observar na figura 2.11, existe uma linha comum a todos os sistemas, aquela que contém o IP da rede (a preto). A linha verde representa o barramento KNX, ao qual se ligam todos os dispositivos da rede elétrica, cujas mensagens são transferidas para a linha IP recorrendo ao KNX USB *interface*, representado na figura 2.12. Esses dispositivos comunicam posteriormente via RF, com um *gateway* KNX/KNX RF, que está inserido numa WLAN, cuja função passa por comunicar com os *handheld devices*. O CIBEK *Mini-Server* liga os interruptores instalados e atuadores do controlo da casa, através deste *link* permite registar a informação dos atuadores embutidos , tais como persianas ou luz. Assim o CIBEK *Mini Server* também assume a avaliação das atividades no edifício [9].

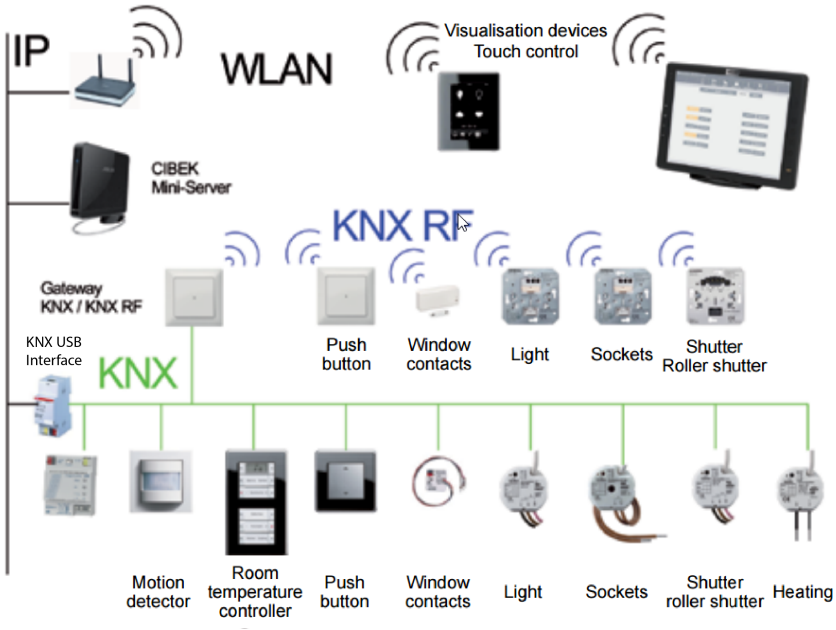


Figura 2.11: Exemplo de uma implementação KNX com dispositivos que comunicam via RF dentro de uma WLAN [10]

Figura 2.12: KNX USB *interface* [11]

Nas soluções convencionais, uma rede IP liga-se a inúmeros *gateways* KNX/IP para aceder aos subsistemas KNX, ativando a comunicação entre diferentes linhas KNX pela *backbone*. Existem vários dispositivos que apesar de se ligarem por *ethernet*, não conseguem comunicar com protocolo KNX portanto, para os integrar, ao invés de implementar um novo barramento KNX/EIB, as comunicações são feitas via IP. Resumindo, a estrutura consiste num dispositivo capaz de estabelecer comunicações sem se ligar ao sistema físico KNX, chamado de "Virtual KNX/EIB Device". O dispositivo deve atuar de forma semelhante a um *router*, que tem um dispositivo ligado à componente KNX. No entanto, esta solução requer um ambiente de configuração do meio, o ETS, *Engineering Tool Software*, introduzida em 1993. Esta ferramenta desenvolvida para sistemas operativos *Windows*, que neste momento já se encontra na sua 5ª versão, tem a função de configurar casas inteligentes e instalações de controlo. Existem para além do ETS, outros softwa-

res que utilizam igualmente KNX como meio físico para se incorporarem, no entanto, o ETS é aquele que oferece a maior compatibilidade com o padrão KNX [10], permite a importação de todas as bases de dados de produtos certificados de todos os fabricantes KNX, garante compatibilidade com versões anteriores de produtos ETS (até ao ETS 2) e é o mais usado pelos profissionais.

Modos KNX

O KNX oferece 3 modos de operação: "E", "S" e "A" (figura 2.13).

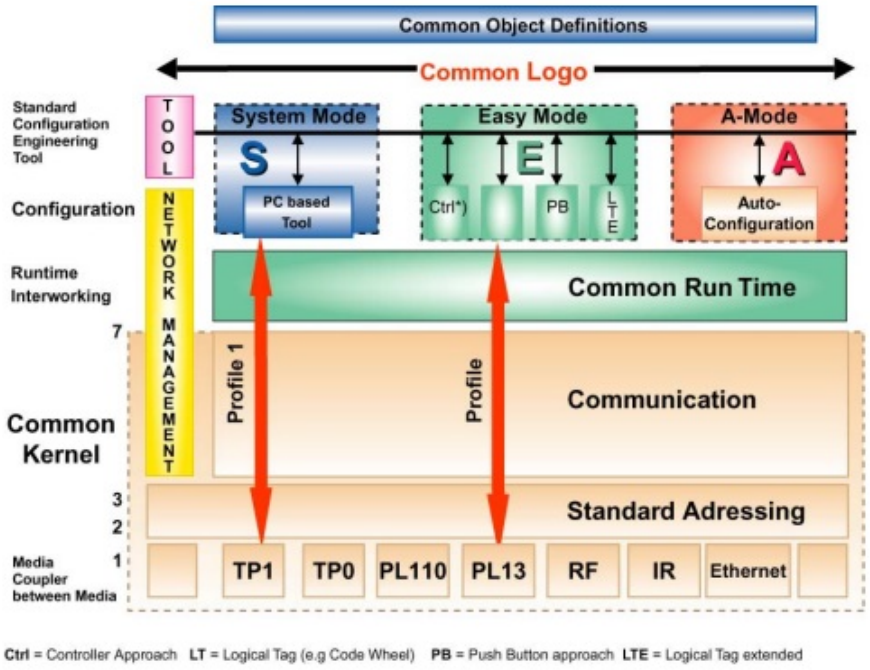


Figura 2.13: Arquitetura KNX [6]

O modo S, *System*, é direcionado para técnicos de instalação especializados, que entendem as funções de controlo do edifício. Neste modo os componentes são endereçados pelo ETS, baseando-se nas bases de dados do produto oferecidas pelo fabricante. Com o ETS, cada componente pode ser integralmente programável de acordo com os requisitos específicos. Este modo é também aquele que oferece o maior grau de flexibilidade em funcionalidades e comunicação. O modo E, *Easy*, é semelhante ao modo S, mas foi criado para agilizar a sua aprendizagem. Isto significa que neste modo, os seus componentes já estão pré-programados e carregados com um conjunto de parâmetros *standard*, utilizando um configurador básico, para reconfigurar parcialmente esses parâmetros, bem como as comunicações. Esta configuração pode ser feita recorrendo também ao ETS, representado na figura 2.14.

Por último, o modo A, *Automatic*, como o próprio nome indica, trata-se de uma aplicação *end-user*, por outras palavras, vocaciona-se diretamente aos utilizadores sem experiência, sendo o mais *user friendly* dos três, destacando-se no uso em eletrodomésticos ou como um *addon* de instalação para o consumidor. Este modo possui mecanismos automáticos de configuração, adaptando de forma automática as suas comunicações aos

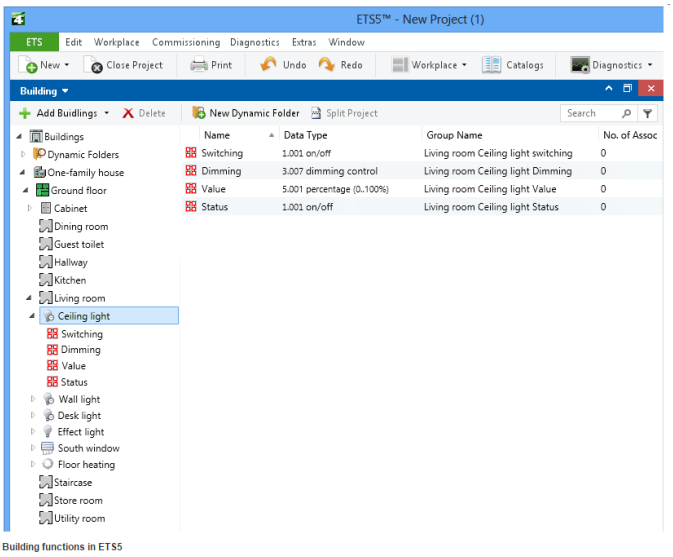


Figura 2.14: Aspeto de uma janela do *software* ETS [10]

restantes componentes que se encontrem no mesmo (modo A) e na mesma rede. Cada componente contém uma definição fixa de parâmetros e uma biblioteca com instruções de como comunicar com os restantes componentes [6]. Através da figura 2.15, torna-se compreensível a relação entre a dificuldade de operar sobre o sistema e o poder que o utilizador sobre ele detém.

Note-se que a partir da 3ª versão do ETS, passou a ser possível usar mais do que um dos modos anteriores na mesma rede.

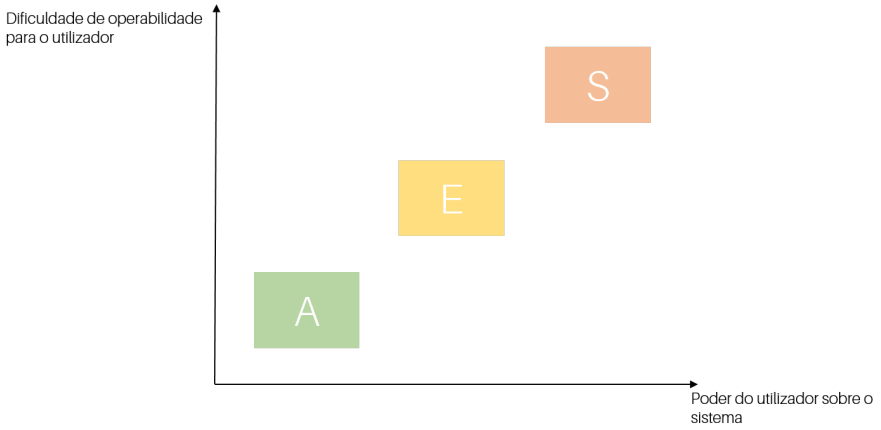


Figura 2.15: O utilizador perante os três modos KNX

2.4.3 Estrutura da mensagem

Como ilustrado na tabela 2.4, a estrutura da mensagem KNX inclui 7 campos. No campo de controlo são atribuídas as prioridades aos dados a transmitir, através de 8 bits. Seguem-se os campos do endereço de origem e do endereço de destino respetivamente,

ambos compostos por 16 bits, se bem que, no caso do endereço de destino, é acrescentado um último bit para distinguir se se trata de um endereço físico de um dispositivo, ou de um grupo. Estes 16 bits subdividem-se em 4 bits para representar o endereço físico, mais 4 bits para definir a sua linha no barramento e ainda em outros 8 bits para a caracterização do próprio dispositivo. A estrutura continua com 3 bits para serem utilizados como um contador e 4 bits para o campo de comprimento da mensagem. Este último, contém o número de *bytes* que o campo útil de informação alberga, definindo assim o tamanho da mensagem, que segue fisicamente no sexto campo (informação útil). Por fim, o último campo tem como função a validação da mensagem, por forma a detetar possíveis erros que ela contenha. Sempre que a mensagem esteja correta, é enviado um aviso ao emissor para dizer que a receção foi concluída com sucesso.

Tabela 2.4: Estrutura de uma mensagem KNK

Controlo	End. Origem	End. Destino	Contador	Comprimento	Info. Útil	Confirmação
8 bits	16 bits	16 + 1 bit	3 bits	4 bits	até 16 x 8 bits	8 bits

2.4.4 Breve discussão dos pontos positivos e negativos do protocolo KNX

No que concerne às características do KNX, pode-se destacar a clara flexibilidade deste sistema, que permite modificar e estender instalações já existentes. Para além disto, não necessita de um controlador central e não está sujeito a qualquer hierarquia de funcionamento, sendo totalmente independente de sistemas de terceiros e neles pode ser integrado, traduzindo-se assim a flexibilidade anteriormente falada. Em relação à qualidade deste produto, a associação *Konnex* requer um alto nível de controlo de qualidade durante todas as etapas da vida do equipamento. Assim, todos os produtos desenvolvidos pelos membros da Konnex, que assumem a marca KNX, têm de demonstrar compatibilidade com a norma ISO 9001 [12], que representa a referência internacional para a Certificação de Sistemas de Gestão da Qualidade. Os pontos mais desfavoráveis que se podem destacar desta tecnologia, são a relativa complexidade das instalações e o custo relativamente elevado do equipamento.

2.5 LonWorks

A LonWorks ou simplesmente LON, *Local Operating Network*, foi criada em 1992, pela Echelon Corporation [12]. Desde então, tem sido usada por várias empresas do ramo da domótica, não obstante, contrariamente às soluções vistas até então, esta é mais vocacionada para hotéis, indústrias e edifícios administrativos porque, devido aos custos elevados, não é passível de ser aplicada em habitações particulares. Para além disso, a LonWorks não é vendida como produto final, mas sim como uma tecnologia que se pode aplicar em sistemas de domótica.

2.5.1 Meios de comunicação e de tipos de sinal

A LonWorks usa como meios físicos de comunicação: a rede elétrica, par entrelaçado, radiofrequência, infravermelhos, cabo coaxial e fibra ótica. Para além destes, o LonWorks

também pode ser ligado via Intra ou Internet.

A arquitetura consiste em utilizar um único barramento para interligar os dispositivos da instalação, comunicando ponto a ponto, P2P. Na figura 2.16 é mostrado o percurso de um pacote, que é transmitido entre dois nós e na figura 2.17 é observável como se ligam os dispositivos.

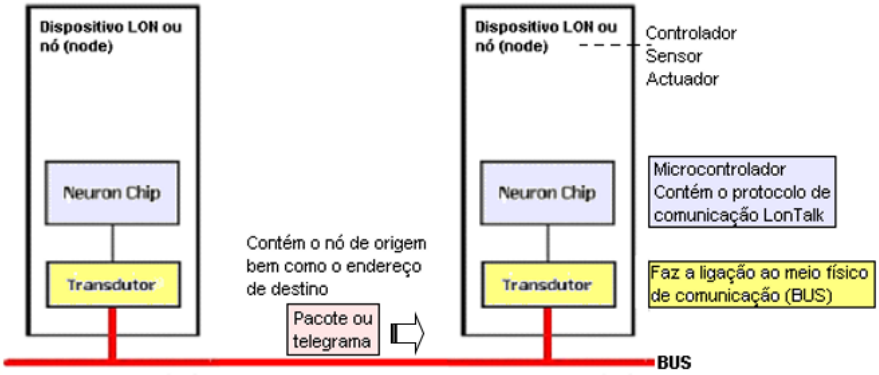


Figura 2.16: Topologia LonWorks [13]

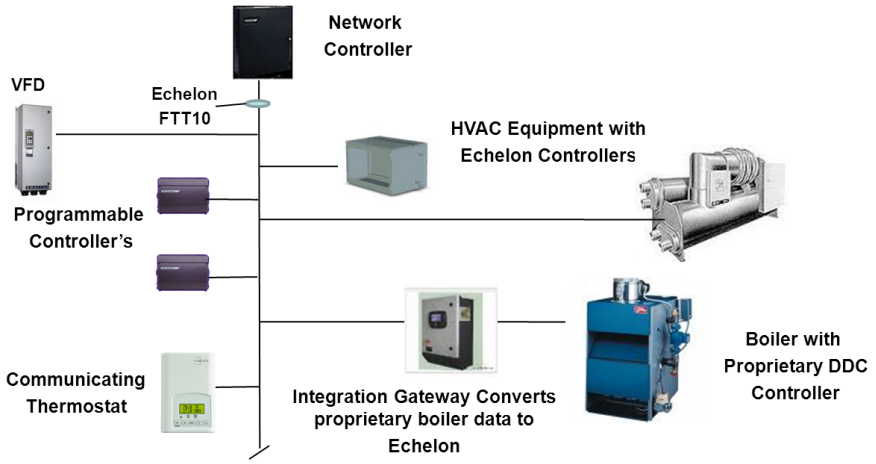


Figura 2.17: Ligação de dispositivos recorrendo a LonWorks [14]

2.5.2 Transmissão de mensagens

Existem duas formas de enviar mensagens entre dispositivos. A primeira passa por utilizar diretamente o destinatário da mensagem e o seu formato, isto implica reduzir a generalização da aplicação, pois os endereços dos destinatários estão previamente definidos pelo programa. No segundo caso, o método recorre à troca de mensagens de forma implícita, permitindo desenvolver a aplicação na sua íntegra. Estas mensagens denominam-se mensagens de rede, *Network Variables* (NV) e caso se tratem de estruturas de dados que obedeçam ao padrão especificado pela LonMark, passam a designar-se de variáveis de rede padrões, *Standard Network Variables Type* (SNVT).

Depois de verificar os tipos de mensagens que podem ser trocados pela rede, é necessário saber como é feita a troca dessas mesmas mensagens. Segundo o protocolo LonTalk (será abordado ainda na presente secção) existem 4 tipos diferentes de serviços de entrega, que podem ser selecionados durante o seu desenvolvimento, ou configurados através de mensagens de rede.

No tipo AKCD (*acknowledged*), o receptor deve enviar de volta para o emissor da mensagem, um reconhecimento que confirme que a mensagem foi recebida de forma correta, caso contrário, deverá retransmitir a mensagem a quantidade de vezes especificada, até atingir um limite de tempo. No caso de se tratar do envio para um grupo de dispositivos, o recetor deve receber a mensagem de reconhecimento de cada dispositivo, se isso não acontecer, a mensagem é reenviada, com identificadores de transições, que se encarregam de evitar o envio da mensagem em duplicado.

No caso do tipo REQUEST, o dispositivo faz uma requisição para um ou mais dispositivos e espera pela resposta dos mesmos. Desta forma, a aplicação é responsável por tratar da requisição recebida antes de ser enviada uma resposta, sendo que essa resposta deve ser igualmente analisada pela aplicação. Tal como no serviço ACKD, também no REQUEST se têm *timeout's* de envio e quantidade máxima de retransmissões, que podem ser configurados durante a programação, ou usando uma ferramenta de gestão de rede.

Relativamente ao serviço UNACKD (unacknowledged), a mensagem é igualmente enviada para um ou mais serviços, mas sem que seja esperada resposta. Por este motivo, este serviço é muito utilizado quando se pretende enviar uma mensagem para um número elevado de dispositivos, garantindo que a rede não é sobrecarregada pelos serviços de escuta, aquando a transferência das mensagens.

O último serviço é semelhante ao serviço UNAKCD no seu funcionamento, a diferença está no envio da mensagem, que é retransmitida o número de vezes que foi previamente configurado pela ferramenta de gestão de rede. Com a repetição do envio da informação, a probabilidade da mensagem não ser recebida pelo destinatário vê-se reduzida. Este serviço tem o nome de UNACKD_RPT (unacknowledged repeated) [12].

Serviço por camadas

A tecnologia LonWorks subdivide-se em camadas, responsáveis por tarefas bem definidas.

- Camada física: todos os recetores reconhecem um *bit*, que seja enviado na rede por outros dispositivos.
- Camada de conexão: define a forma de comunicação do dispositivo, de um modo sucinto, estabelece prioridades e ordens de transmissão e rege o reconhecimento dos dados por arte dos dispositivos.
- Camada de rede: zona de endereçamento. Define os endereços dos dispositivos e como devem atuar consoante o canal de comunicação.
- Camada de transporte: é responsável pela entrega dos dados, sua validação e deteção de mensagens em duplicado.
- Camada de sessão: controla as trocas de dados.

- Camada de apresentação: estrutura os dados que estão a ser trocados pelas camadas mais baixas, por forma a possibilitar diferentes codificações (variáveis de rede e mensagens de aplicações).

O protocolo LonTalk

O protocolo utilizado pela LonWorks denomina-se LonTalk, e pode ser implementado em qualquer microcontrolador, ou microprocessador. O NeuronChip (figura 2.18), desenvolvido pela Motorola e o L-Core pela LOYTEC, são exemplos de dois chips que possuem *stack's* LonTalk.

A implementação das camadas anteriormente descritas é feita à custa do Neuron, um *chip* registado pela Echelon [12].

Figura 2.18: Chip Neuron da série 5000 [15]

As especificações do Neuron5000 encontram-se sintetizadas na tabela 2.5.

Tabela 2.5: Espicificações do chip Neuron 5000 [15]

Características	**Descrição**
Voltagem de operação	3.3V
Temperatura de operação	-40^{o}C até 85^{o}C
Dimensões	Pacote QFN 7 mm × 7 mm 48-pin
Processador	Clock interno com frequências até 80 MHz.
Memória	Interface de memória serie para EEPROM's e memória flash para memórias não voláteis.
Capacidade	Suporta até 254 variáveis de rede e 127 identidades
Comunicações	Comunicações com portas de 5 pinos e drive de 3.3V com tolerância a pinos de 5V
Interface	12 pinos I/O com 35 modelos I/O padrão programáveis
Armazenamento	Suporta até 42 kB de código para a aplicação
Memória dinâmica (RAM)	64kB (44 kB acessíveis ao utilizador)
Memória de leitura (ROM)	16 kB
Outros	48-bit exclusivos para identificação do chip
	Aceita dispositivos de baixo preço
	Interruptores dedicados para melhorar a resposta a eventos externos
	Inclui hardware UART com 16-byte de recepção e transmissão FIFOs

Este processador inclui no seu interior 3 processadores lógicos independentes de 8 bits, para gerir a camada física do controlo de acesso de dados (MAC), a rede e a aplicação do utilizador. Estes elementos estão ilustrados na figura 2.19.

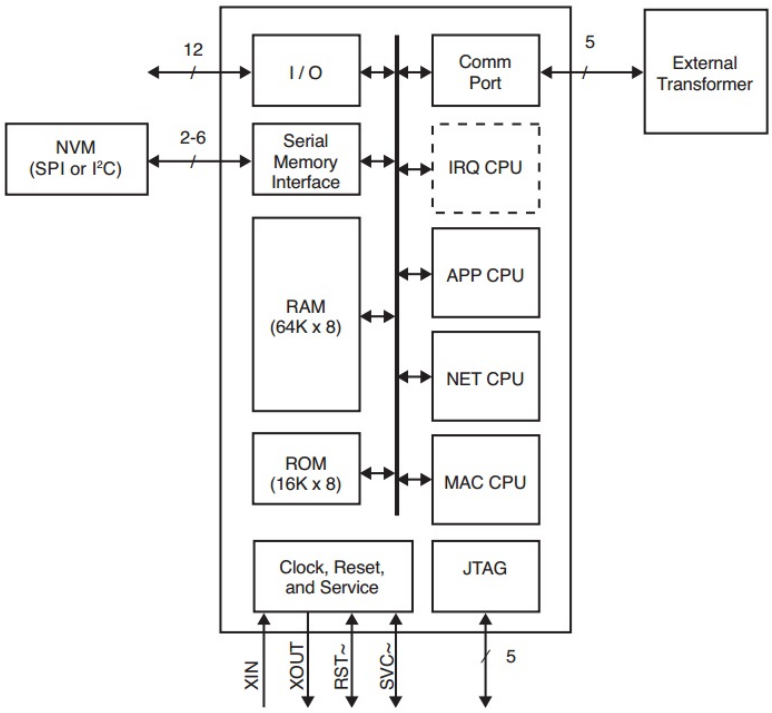


Figura 2.19: Estrutura interna do processador do chip Neuron 5000 [15]

Descrição dos pinos do processador Neuron 5000

Todas as entradas do Neuron5000 são de baixa voltagem e compatíveis com o tipo de ligação transístor-transístor (LVTTL), caracterizando-se pelo seu baixo nível de perdas e é tolerável a 5V. Relativamente às saídas, todas as digitais são do tipo *slew-rate*, tecnologia que permite reduzir a interferência eletromagnética.

O LonTalk pode ser implementado em meios físicos diferentes (rede elétrica, fibra ótica, radiofrequência, entre outros) e a ele pode ser acoplado um número infindável de dispositivos, mais precisamente $2^{48} \cdot 32385$, sendo que 32385 são os dispositivos permitidos por domínio e 2^{48} são os diferentes domínios permitidos. Através da figura 2.20, podem-se observar as funções correspondentes de cada pino do processador do Neuron 5000.

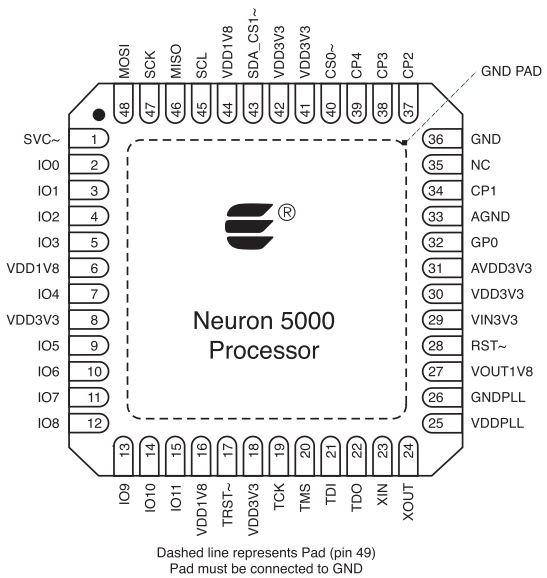


Figura 2.20: Disposição dos pinos do processador do chip Neuron 5000 [15]

2.5.3 Considerações finais

Trata-se de uma tecnologia aberta, por outras palavras, permite o uso de novos dispositivos, eliminando a obrigatoriedade de optar por um fornecedor exclusivo. Desta forma, gastos monetários, tempos de manutenção e tempos de espera relacionados com a entrega de material são evitados.

Para além da vantagem de ser uma tecnologia aberta, a LonWorks é também uma tecnologia do tipo descentralizado. Por esta razão, não existe a necessidade da existência de um controlador central, que coordene todas as ordens a ser processadas. Por conseguinte, o número de ligações diminui e a instalação torna-se muito menos complexa.

2.6 CEBus

O sistema CEBus (Consumer Electronic Bus) foi normalizado pela EIA (Electronic Industries Association) em Setembro de 1992, com o objetivo de desenvolver uma rede de comunicação para ambiente residencial [12].

2.6.1 Cablagem e topologia

O protocolo CEBus utilizava um modelo de comunicação de dispositivo a dispositivo sem hierarquias nem restrições, usando um barramento comum. O CEBus definia dois tipos de canais de comunicação: um canal de controlo para mensagens entre dispositivos, e um conjunto de canais de dados para distribuição de áudio, vídeo ou outro qualquer sinal de banda larga. A estrutura física do protocolo pode ser acompanhada pela figura 2.21.

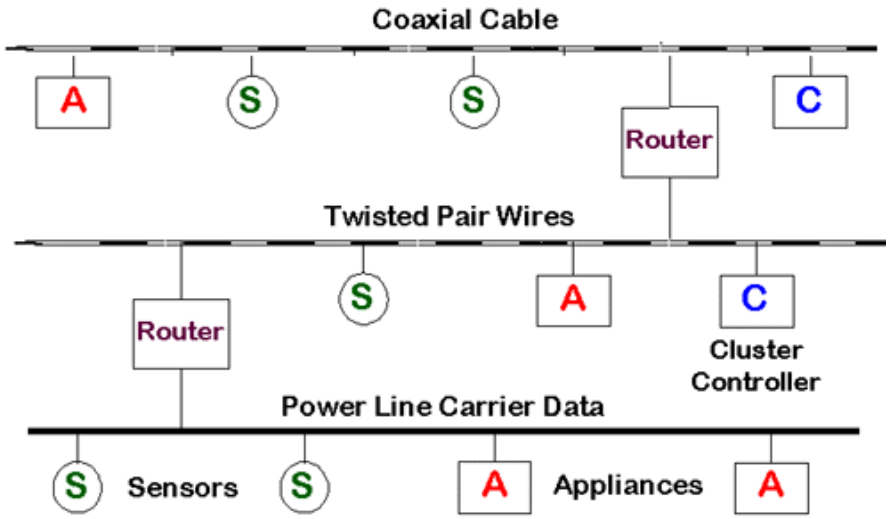


Figura 2.21: Estrutura física em CEBus [16]

2.6.2 Controlo de acesso ao meio

A norma CEBus disponibilizava vários meios físicos de transmissão:

- Linha de potência elétrica (PL);
- Cabo de par entrançado (TP);
- Cabo Coaxial (CX), preferencialmente para dispositivos que utilizaram sinais de áudio e vídeo;
- Radiofrequência (RF);
- Infravermelhos (IR).

Com os diversos meios físicos de transmissão podia-se encontrar nesta tecnologia a melhor solução para as diferentes implementações, existindo apenas a necessidade de implementar um dispositivo capaz de comunicar entre os dois meios de transmissão, um *router*, por exemplo. Cada nó, onde estava inserido o dispositivo CEBus, podia ser dividido em três partes. A parte designada por PROTOCOLO era igual em todos os dispositivos, sendo esta a responsável pela receção e transmissão das mensagens pelo meio físico. O dispositivo interpretava as mensagens CAL provenientes do barramento, através da camada CAL. Por último, a parte designada de APLICAÇÃO representava a aplicação do nó e era constituída pelo *hardware* e *software* que definiam a operação do dispositivo.

2.6.3 Estrutura da mensagem

Em relação ao modelo de mensagens CEBus, representado na figura 2.22, estas continham um *start code*, um campo de controlo, um endereço de destino, um endereço de origem, uma mensagem CAL e um campo chamado de *checksum*. O *start code* era constituído por um valor aleatório de 8 bits que tinha como função efetuar a deteção e

resolução de colisões durante o acesso ao canal. O campo de controlo identificava o tipo de serviços da ligação de dados (DLL - Data Link Level) que o pacote continha, sendo a sua dimensão de 8 bits. O campo de destino e o de origem tinham uma dimensão de 32 bits, em que 16 bits correspondiam ao endereço de sistema e os restantes 16 bits correspondiam ao endereço do nó. A mensagem CAL possuía o tamanho máximo de 32 bytes. O último campo da mensagem continha um tamanho variável de 8 bits no máximo e servia para determinar a integridade do conteúdo da mesma.

Start Code	Campo de Controlo	Endereço de Destino	Endereço de Origem	Mensagem CAL	Checksum
8 bit	8 bit	16 bit	16 bit	até 32 bytes	8 bit

Figura 2.22: Estrutura da mensagem em CEBus

2.6.4 CEBus, uma tecnologia extinta

Relativamente ao CEBus, conclui-se que apesar de ser um sistema que cumpria com as funções para o qual foi criado, não conseguiu permanecer no mercado, sendo superado por outros que envolveram tecnologias mais recentes e de menor custo. No entanto, foi um sistema que teve algum destaque enquanto esteve em vigor, justificando a sua breve abordagem nesta dissertação.

2.7 Soluções de domótica recentemente desenvolvidas

Durante as secções: 2.3, 2.4, 2.5 e 2.6, foram abordadas as soluções comerciais: X10, KNX, LonWorks e CEBus. Na secção 2.7 serão abordadas 3 soluções académicas provenientes de estudos no âmbito de dissertações e uma proveniente de um artigo científico.

2.7.1 Solução integradora [1]

Apesar da existência de sistemas bastante evoluídos, a sua acessibilidade era bastante reduzida. Fatores como custo de aquisição, compatibilidade e escolha reduzida, faziam parte da génese deste problema.

Como tal, surgiu a ideia de criar um sistema integrador na área da domótica para divulgar esta área cada vez mais procurada pelos mercados. Com essa interligação seria possível ter dois ou mais sistemas a funcionar num edifício ou habitação, permitindo ao utilizador optar pelos dispositivos de mais baixo preço e aproveitar instalações mais antigas, de forma a interligá-las com sistemas recentes. Assim, a concorrência na área da domótica seria maior. Desenvolveu-se uma solução para o controlo e monitorização de habitações e de grandes edifícios, capaz de utilizar diferentes sistemas, nomeadamente, o sistema X10, o sistema EIB, autómatos industriais e dispositivos de radiofrequência (RF). Os sistemas físicos que foram sujeitos a controlo foram: a iluminação, sistema AVAC, sistema de rega e controlo de acessos. Para além disto, também se desenvolveu um programa para ambiente *Windows* capaz de oferecer um controlo dinâmico da habitação através de uma interface gráfica. A solução desenvolvida integrou três diferentes tipos de *hardware*, dispositivos RF, um autómato industrial e um computador, permitindo assim controlar e monitorizar vários parâmetros da habitação. Através dos dispositivos RF

controlou-se a iluminação e os sistemas de AVAC, com o autómato industrial o sistema de rega e a abertura/fecho de portões e portas de acesso e o registo dos acontecimentos de todas as ações tomadas na habitação ficou ao encargo de um computador.

Arquitetura da solução

Na figura 2.23 é observável que todos os periféricos comunicam com o equipamento central. Da arquitetura fazem partes os sistemas EIB e X10, um autómato industrial, dispositivos RF e um computador, para registar todos os eventos.

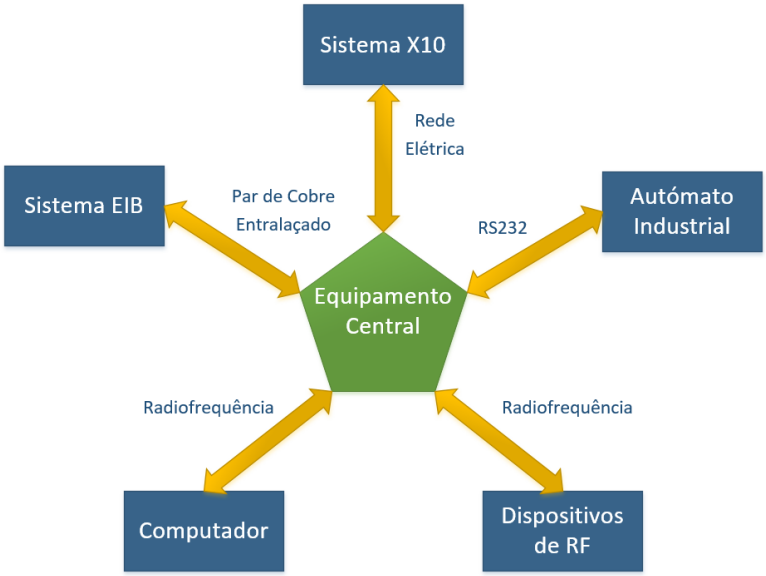


Figura 2.23: Arquitetura da solução integradora [1]

Comunicação

A comunicação foi o factor chave para o desenvolvimento da solução integradora, porque era necessário assegurar que esta chegava ao sistema pretendido. Na tabela 2.6 seguem as comunicações efetuadas nesta solução.

Tabela 2.6: As diferentes comunicações existentes entre os vários dispositivos

Dispositivo RF	$\Longleftrightarrow$	Dispositivo RF
Dispositivo RF	$\Longleftrightarrow$	Computador
Dispositivo RF	$\Longleftrightarrow$	Autómato
Dispositivo RF	$\Longleftrightarrow$	Sistema X10
Dispositivo RF	$\Longleftrightarrow$	Sistema EIB
Sistema EIB	$\Longleftrightarrow$	Sistema X10
Computador	$\Longleftrightarrow$	Equipamento central

Síntese do problema

Contudo, apesar da solução ser efetivamente uma mais valia na integração das soluções de domótica, os dispositivos usados estão neste momento ultrapassados. Não existe comunicação com os sistemas EIB nem com os dispositivos X10, por forma a completar a arquitetura apresentada na figura 2.23. Para além destes aspetos seria importante usar dispositivos *wi-fi* para substituir as comunicações.

2.7.2 Desenvolvimento de uma infraestrutura de comando multifunções EIB-KNX para smartphone [17]

O desenvolvimento de uma infraestrutura que permita comandar uma instalação EIB-KNX a partir de um *smartphone* é uma proposta interessante, especialmente pelo facto do utilizador possuir o controlo da sua casa na palma da mão. Nesta solução é importante a segurança no encaminhamento de dados em redes móveis sem fios de baixos recursos, eficiência energética e conectividade entre elementos heterogéneos, por forma a evitar que utilizadores indesejados acedam ao controlo da habitação.

Desenvolvimento da solução

O desenvolvimento foi focado em aspetos como a versatilidade e facilidade de utilização. Neste contexto, o trabalho foi suportado por um conjunto de tecnologia bastante diversificada, quer ao nível das comunicações quer de plataformas. Esteve em destaque o recurso a comunicações sem fios através de redes 802.11 *wi-fi*, comunicações sobre TCP/IP, nomeadamente MODBUS/TCP e protocolos no domínio da domótica como o KNX e KNX-IP, representados na figura 2.24. A nível de plataformas, o trabalho foi desenvolvido sobre o sistema operativo *Android*, tendo a um nível mais baixo sido suportado por controlador KNX/IP e router KNX da Wago. O *core* do desenvolvimento foi focado na constituição de uma base de dados que relaciona a estrutura do edifício com a funcionalidade da instalação de domótica. Foi igualmente focada no desenvolvimento de uma aplicação que, suportada na informação da base de dados, é capaz de gerar

dinamicamente uma interface gráfica que representa a constituição e funcionalidade do edifício, assim como, a capacidade de mapear essas funcionalidades em serviços de comunicações que interagem com a instalação de domótica. Para o utilizador o sistema deve apresentar-se como uma aplicação que corre no telemóvel, capaz de obter a configuração de uma instalação elétrica, disponibilizando uma interface gráfica para o seu comando. Comando à distância, supervisão e configuração de sistemas associados à domótica foram os fins práticos resultantes desta dissertação.

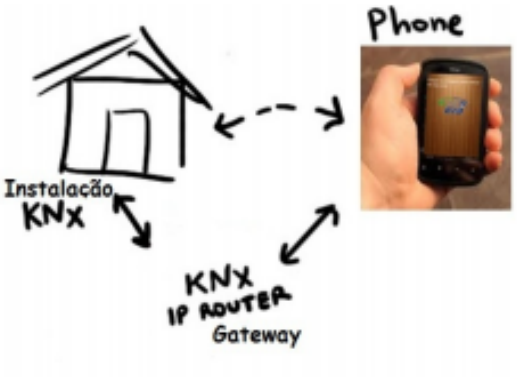


Figura 2.24: Arquitetura KNX/IP [17]

Um dos pontos fortes desta tese foi a mobilidade, como tal, necessitou de um cuidado especial. Numa fase inicial recolheram-se as tecnologias existentes no mercado, relativas às redes mais adequadas (redes IEEE 802.1x.) sendo as candidatas: wifi IEEE 802.11, Bluetooth IEEE 802.15.1 e ZigBee IEEE 802.15.4. *Wi-fi* foi a solução encontrada, dada a versatilidade que oferece, possibilitando a inter-conectividade entre um elevado número de dispositivos móveis e a comunicação via *web*. Para assegurar a conectividade entre a componente móvel e a infraestrutura cabelada da rede de domótica foi necessária a existência de um *gateway* que permitisse a comunicação da infraestrutura móvel para a rede de domótica e desta para a infraestrutura móvel. Desta forma todos os componentes da instalação passaram a garantir a comunicação.

Síntese do problema

Esta solução apresenta-se de facto como uma evolução significativa no que concerne ao poder do utilizador sobre os dispositivos, que assim pode ser feita à distância através de um dispositivo que persegue o ser humano para todo o lado, o que remete para o controlo permanente sobre o edifício. No entanto, esta solução vê-se limitada pelo facto de estar dependente de um *gateway*.

2.7.3 Um sistema inteligente de controlo e monitorização [18]

Um dos aspetos nos quais a domótica carece de algum desenvolvimento é em sistemas para monitorização e controlo de uma forma integrada em tempo real. Por conseguinte, foi criada uma rede de domótica que combinou a comunicação do meio de par entrelaçado com o meio sem fios por radiofrequência. O sistema incluiu microcontroladores que desempenharam funções de monitorização e atuação, os quais comunicavam por MODBUS. Com auxílio de *software* SCADA foi criada uma solução que permitiu acesso remoto ao

sistema e a possibilidade de prestar serviços pela *web* e integrar dispositivos de fabricantes diferentes.

A solução desenvolvida

A solução proposta, ilustrada na figura 2.25, engloba: módulos RF, o módulo central (router), uma plataforma central (movicon), plataformas secundárias, par entrelaçado, módulos 485 e comandos IR.

O sistema partiu de uma estrutura centralizada e dividiu-se em diversos subsistemas modulares. A plataforma central foi desenvolvida com recurso ao *Movicon Building Automation* que deu suporte a diversas plataformas secundárias, através de um servidor local, e ao armazenamento de dados devido à compatibilidade com diversas bases de dados. Foi através destas plataformas que o utilizador recebeu a interface gráfica de acesso aos serviços de monitorização e de controlo. Esta parte do sistema está representada pelos blocos a vermelho. Ligado à plataforma central, através de uma interface *serial*, esteve um microcontrolador, representado pelo bloco a cinzento, que fez o reencaminhamento das mensagens no sistema. Este microcontrolador recebia todos os pedidos enviados pela plataforma central e conseguia, através do endereço de destino, determinar em que meio estava localizada a estação de destino. Este microcontrolador, através de dois transcetores, tinha uma *interface* em cada um dos meios de transmissão e na sua essência funcionou como um *router*.

Arquitetura

Os blocos a amarelo representam os subsistemas modulares que, em função do meio onde estão conectados, designam-se por módulos RS-485 (barramento RS-485) ou por módulos RF (rede sobre radiofrequência). Estes módulos podiam ser subsistemas convencionais criados pelo autor, que tinham por base um microcontrolador e um transcetor alusivo ao meio em que estavam inseridos, ou outros dispositivos desde que compatíveis com o barramento RS-485 e cujo protocolo de comunicação fosse suportado pelo *Movicon5*. Os módulos convencionais puderam também ser controlados através de um comando de infravermelhos.

A implementação do 485 foi justificada como sendo uma solução de baixo custo, que garantia um elevado nível de segurança no meio e também permitia eliminar com sucesso o ruído da comunicação. No entanto, quando comparada com sistemas sem fios, chegou-se à conclusão que apresentava maiores requisitos de instalação e limitação na localização dos dispositivos na rede. Portanto, para dar suporte a uma comunicação sem fios, foi proposta também uma solução com comunicação via RF.

No meio RF foi adotada uma topologia em estrela, onde o módulo *master* funcionou como nó central, garantindo a comunicação. Ambos os meios foram definidos como *half-duplex* e funcionaram segundo um modelo *master/slave*. A definição dos meios como *half-duplex* deveu-se, no caso do barramento RS-485, à necessidade de apenas 1 par de fios entrançados e da compatibilidade de dispositivos *full-duplex* com um meio *half-duplex* (o contrário já não se verificou) e no caso do meio RF, à maioria dos transcetores existentes no mercado serem *half-duplex*, relativamente aos transcetores *full-duplex*, que por serem mais complexos, são significativamente mais caros [18].

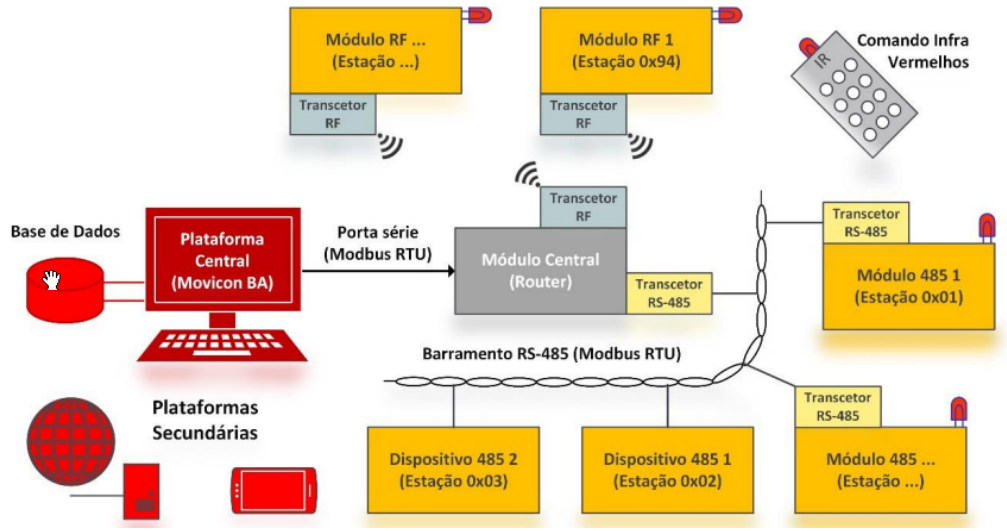


Figura 2.25: Arquitetura do sistema de monitorização e controlo [18]

Conversão com transcetores

Para criar um meio cablado, sob o protocolo RS-485, com o microcontrolador foi necessário adicionar um transcetor para converter os níveis TTL para níveis diferenciais. O mesmo valeu para criar o meio sem fios por radiofrequência com o microcontrolador selecionado, para que fosse permitido o envio de mensagens sobre a forma de sinais eletromagnéticos propagáveis através do ar.

Desenvolvimento de um protótipo

Para generalizar a solução foi desenvolvida uma placa protótipo (figura 2.26) para comunicações RF e do tipo 485. Nela foram inseridos um microcontrolador PIC18F26J50, um recetor recetor IR TSOP34136, relés, reguladores de tensão e também entradas compatíveis com placas MODRS485 e transcetores ADM3483ARZ.

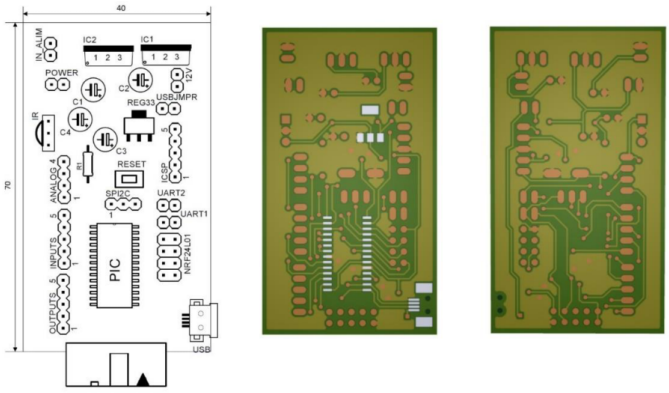


Figura 2.26: Placa protótipo [18]

Síntese do problema

À semelhança da solução integradora, também esta não apresenta uma solução desenvolvida para funcionar por *wi-fi*, recorre a uma vastidão muito complexa de equipamentos e para além disso, alguns dos equipamentos necessitam de espaços alojados para serem colocados.

2.7.4 A Smart Gateway Architecture for Improving Efficiency of Home Network Applications [19]

Na domótica continuava a coexistir um problema entre as diversas plataformas existentes, o uso de um *gateway* para garantir a funcionalidade da implementação. Nesse sentido, em Novembro de 2015, foi apresentado o IAGW, com o objetivo de criar um *gateway* dedicado ao uso em controlo de habitações. Este *gateway* suporta várias aplicações desenvolvidas para ambientes domésticos, com possibilidade de configuração local para o nó e serviço de acesso. O modo de trabalho deste *gateway* e o seus parâmetros podem ser definidos à custa de uma consola *web*.

Visão geral do sistema e sua arquitetura

A hierarquia da arquitetura divide-se em 3 camadas, figura 2.27, a camada USN, a camada de rede e a camada de serviço. Nos CN's, incluídos na camada USN, estão incluídos os serviços de qualidade do ar, WCS, IPS, câmara de vigilância, e diversos sensores. Todos os CN's, à exceção da câmara de vídeo estão integrados com módulos *ZigBee*. A câmara liga-se ao IAGW por *wifi* ou por uma ligação por cabo com tomada RJ45. O HNASC recebe os dados pelo IAGW, armazena-os em tempo real e faz a sua autenticação.

Como arquitetura funcional proposta para o IAGW, ilustrado na figura 2.27, são observáveis vários módulos no seu interior. O módulo *"Node Mngr"* reponsabiliza-se por gerir os CN's, o *"Packet Mngr"* tem como função transformar num pacote de dados *standard*, a informação oriunda dos CN's; o módulo de segurança inclui serviços de desencriptação, autenticação e autorização de acesso aos dados, o QoS é responsável pelo serviço de qualidade e por fim, o *Remote Maintenance* encarrega-se do suporte remoto do sistema.

O acesso pode ser feito através de dois meios: o adaptativo e o M2M. O IAGW divide-se em dois tipos de interfaces de comunicação: HMMP-S e HMMP-A. O primeiro relaciona-se com o acesso aos terminais, adquire os atributos da rede doméstica, bem como todos os tipos de serviços através do IAGW, com a sua comunicação suportada à custa de dispositivos *ZigBee*. O segundo relaciona-se com a interface de transmissão de dados entre o *ZigBee* e a plataforma HM2M.

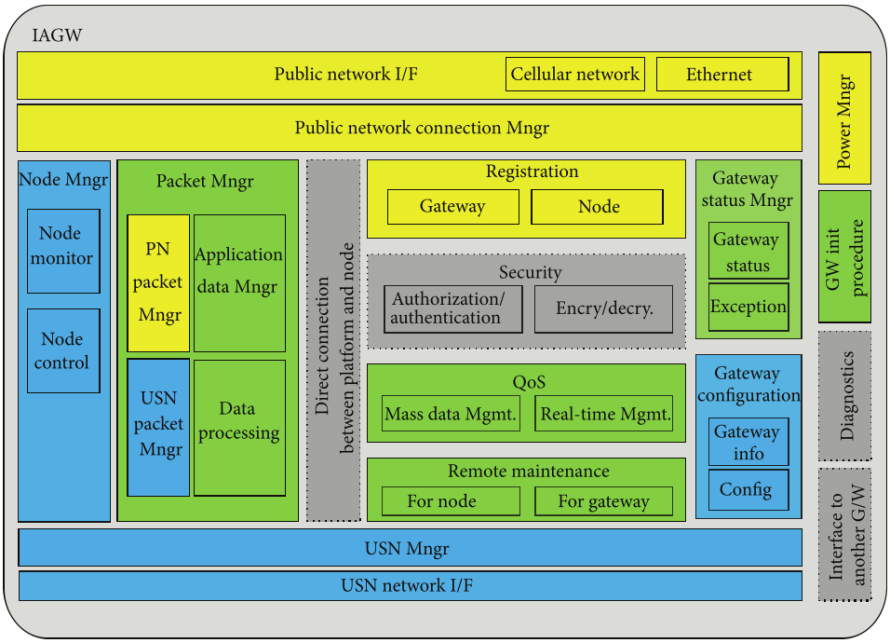


Figura 2.27: Disposição dos módulos no interior do IAGW [19]

Considerações finais

De um modo sucinto, a mais valia desta implementação passa pela integração de vários tipos de dispositivos, com diferentes tipos de comunicação, numa solução sem fios. No entanto, o uso de módulos *ZigBee* afeta a sua *performance*, quer ao nível da velocidade (Riço, 2014), quer ao nível da sua compatibilidade [17].

2.8 nRF

2.8.1 Características e aplicabilidade

O nRF24L01+, figura 2.28, é um *chip* que usa radiofrequência para enviar sinais. Trata-se de um dispositivos usado em aplicações de baixo consumo, mais especificamente, o nRF24L01+ apresenta picos de corrente inferiores a 14mA e é alimentado por tensões entre os 1.9V e os 3.6V, reguladas por um regulador de tensão interno, habilitando a tecnologia "advanced power management", o que permite que seja alimentado por pequenas baterias substituíveis após longos períodos de tempo, devido à elevada autonomia do *chip*. O nRF24L01+ trabalha a uma frequência de 2.4GHz e transfere dados a uma taxa de 2Mbps ao longo de um raio de 200m sem obstáculos em seu redor. Este *chip* tem um limite máximo de 32 bytes para o envio de mensagens e a configuração inicial pode ser aplicada até um limite máximo de seis módulos. Para controlar as aplicações, através do suporte oferecido pela tecnologia *Enhanced Schock Burst*, um acelerador do protocolo do *hardware*, este dispositivo usa uma interface SPI de alta velocidade, que pode transferir dados até uma taxa de 10Mbps e consegue operar entre temperaturas negativas de 40 graus celsius até 80 graus celsius positivos [20]. Este dispositivo não requer filtros de ciclo, ressonadores, ou díodos, apenas necessita de um cristal, uma antena e da combi-

nação dos circuitos internos. Na tabela 2.7 estão representadas as funções dos pinos do nRF.

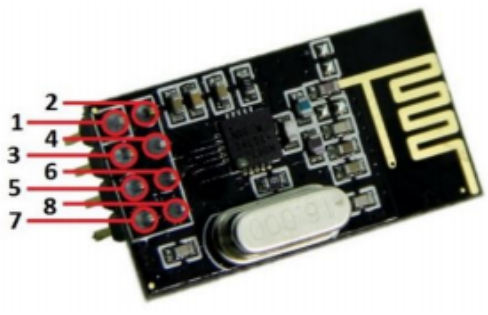


Figura 2.28: Transcetor de radiofrequência nRF24L01+ [18]

Tabela 2.7: Descrição dos pinos do transcetor nRF24L01+ [18]

Pino	Abreviatura	Designação	Descrição
1	GND	*Ground*	Massa - 0V
2	VCC	*Voltage from a Power Supply*	Alimentação de 1.9 a 3.6V
3	CE	*Chip Enable*	Entrada digital que permite ativar o chip para receção/transmissão
4	CSN	*Chip Select NOT*	Entrada digital que permite selecionar o chip na comunicação SPI
5	SCK	*Serial Clock*	Entrada do sinal de relógio
6	MOSI	*Master Output Slave Input*	Entrada de dados SPI
7	MISO	*Master Input Slave Output*	Saída de dados SPI
8	IRQ	*Interrupt Request*	Saída digital que sinaliza as interrupções (na transição de 1 para 0)

De uma forma mais resumida, abaixo encontra-se a lista das características principais do nRF24L01+.

- Baixo custo, *single-chip 2.4GHz GFSK RF transceiver IC*
- Livre de licenças para usar a banda de operação de 2.4GHz da banda ISM
- Opções de 250kbps, 1Mbps e 2Mbps de velocidade de propagação no ar
- Protocolo de aceleração de *hardware* melhorado
- Consumo muito baixo com vida da bateria estimada para além de 12 meses
- Compatibilidade *drop-in* com o transmissor da geração anterior do Nordic nRF24L01
- Compatibilidade com todos os dispositivos Nordic das séries nRF24L, nRF24E e nRF240.

2.8.2 O protocolo *Enhanced Shockburst*

Como já referido, o nRF24L01+ utiliza a radiofrequência para comunicar. O protocolo inerente a essa comunicação foi desenvolvido pela Nordic, sendo um protocolo que suporta o tipo de comunicação bidirecional, ou seja, permite que os pacotes de dados circulem do dispositivo 1 para o dispositivo 2 e *vice-versa*. Este protocolo inclui um pacote

de *buffer*, um pacote de reconhecimento e ainda retransmissão automática de pacotes perdidos [21].

Este protocolo encontra-se embebido no *hardware* da família nRF24Lxx e permite manusear e efetuar a transição automática de pacotes para fácil implementação de uma comunicação bidirecional fiável. A transição é feita à custa de dois transmissores, sendo um definido como "primary receiver" (PRX) e o outro como "primary transmitter"(PTX). Na figura 2.29 é mostrada a forma como é feita a comunicação e o seu sentido (bidirecional).

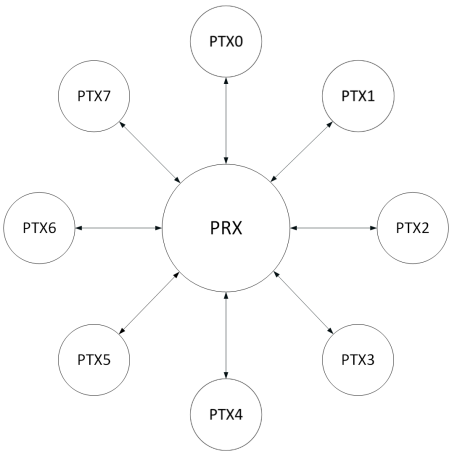


Figura 2.29: Rede do protocolo ESB [21]

2.8.3 Características do ESB

- Topologia de rede em estrela
- 1 até 32 bytes de comprimento para a carga de transmissão
- Comunicação bidirecional entre cada PRX e PTX
- Reconhecimento de pacotes e sua retransmissão automática, para diminuir a quantidade de dados perdidos.
- TX e RX individuais tipo FIFO para cada canal.
- Compatibilidade com versões anteriores de dispositivos da família nRF24Lxx *Enhanced ShockBurst*

2.8.4 Considerações finais sobre o nRF

Concluindo, o nRF24L01+ tem a vantagem de comunicar sem recurso a cablagem e possui um consumo de energia reduzido, alcançado à custa da tecnologia *advanced power management*. Contudo, este dispositivo utiliza a frequência de 2.4GHz, usada já para comunicações *wireless* do *router* doméstico com os dispositivos pessoais que se ligam à *internet*, podendo causar interferências. Para além disso, a sua implementação vê-se limitada a um uso máximo de 6 nRF24L01+.

Capítulo 3

Arquitetura e especificação da solução proposta

No presente capítulo será apresentada a arquitetura da solução proposta, bem como as suas especificações. Esta solução tem como objetivo satisfazer a interação homem/casa e para atingir esse fim serão usadas formas de comunicação sem fios, entre dispositivos que o homem usa no seu dia a dia (*smartphones, tablets*, e/ou computadores pessoais) e os atuadores/sensores que comunicam com os dispositivos da casa (estores, sistemas de iluminação, sistemas AVAC, controlo de acessos e sistemas de rega).

3.1 Arquitetura

A arquitetura do sistema, representada na figura 3.1, consiste num servidor, num cliente e no meio de comunicação usado. Os servidores, vulgarmente conhecidos por *hosts*, são aqueles que fornecem serviços e informações na rede, enquanto que os clientes são quem acede a tais serviços e informações. Quanto ao meio de comunicação, este pode ser feito à custa de meios cablados ou sem fios. Para esta dissertação será usada a norma IEEE 802.11 para comunicação sem fios e a comunicação cablada será feita via GPIO.

A figura 3.2 ilustra a forma como se pretende implementar a solução. Nela representam-se os diversos dispositivos físicos que se pretendem controlar, à custa de um ESP8266, daí o facto de ele estar sempre ilustrado juntamente a cada um desses dispositivos. Trata-se de uma solução simples, que usa apenas três tipos de dispositivos para comunicar entre o utilizador e o *hardware* que se pretende controlar. O ESP8266, o *router* da rede doméstica e os dispositivos pessoais (*smartphones*, *tablets*, e/ou computadores pessoais).

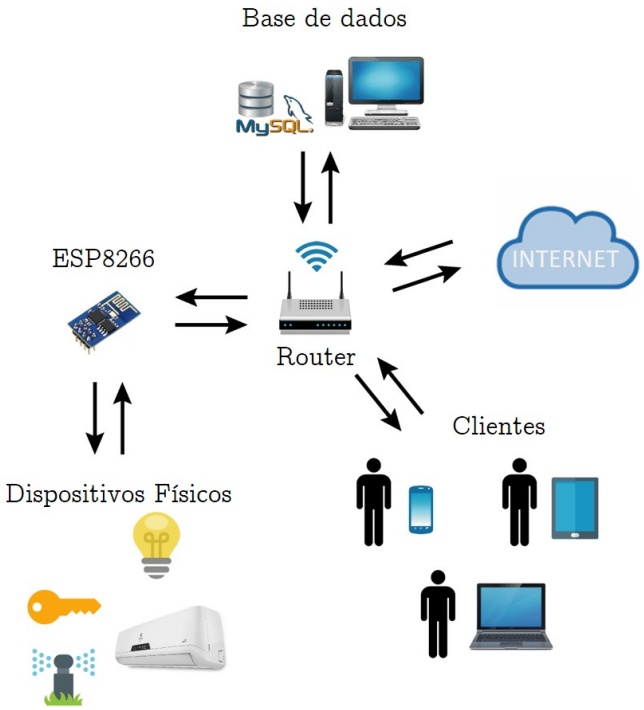


Figura 3.1: Arquitetura da solução

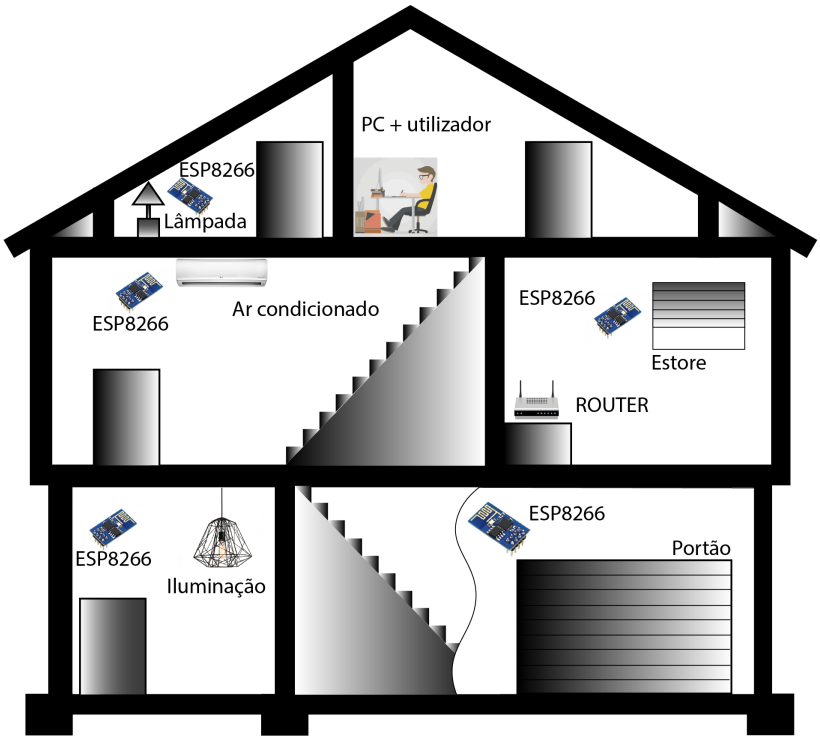


Figura 3.2: Esquema da proposta aplicada a um edifício

Considere-se novamente o modelo representado na figura 3.2. No andar de cima, está representado o utilizador, a trabalhar no seu computador pessoal, onde pode controlar e monitorizar os dispositivos presentes na sua habitação, nomeadamente a iluminação, um estore, um portão e uma unidade interior de ar condicionado. No caso da iluminação, o ESP8266 pode funcionar como um interruptor ou ainda controlar o fluxo luminoso.

3.1.1 Especificação da solução

Enquadrando este modelo na comunicação cliente/servidor, os ESP8266 representam os clientes e o computador pessoal representa o servidor da base de dados. O servidor baseia-se nos recursos disponibilizados por um servidor de base de dados local, ao passo que o cliente consiste, no caso dos *tablet's* e *smartphones*, não numa aplicação dedicada, mas numa página acedida através de um *browser* e no caso dos computadores, numa aplicação desenvolvida para o efeito de controlo e monitorização de um edifício. No que respeita à comunicação cliente/servidor, esta é feita recorrendo à norma IEEE 802.11, ao cargo de dois protocolos de comunicação. O TCP *(Transmission Control Protocol)*, e o HTTP *(Hypertext Transfer Protocol)*. Quanto ao dispositivo físico usado para transferir a informação entre eles, é utilizado o *router* doméstico, ou seja, existe uma LAN *(Local Area Network)* comum às duas entidades (cliente e servidor).

Servidor

Como representado na figura 3.3, o servidor é constituído por 4 tipos de elementos. O ESP8266 é o elemento central do servidor, funcionando como um *access point* da rede local, sendo o responsável por processar e reencaminhar a informação recebida pelos sensores e pela base de dados para a LAN. Os sensores serão responsáveis por adquirir os dados de temperatura e humidade do ar e convertê-los em sinais elétricos, enquanto que a base de dados, cujo banco de informação se aloja na memória de um computador, é responsável pelo armazenamento e fornecimento dos dados lidos pelos sensores.

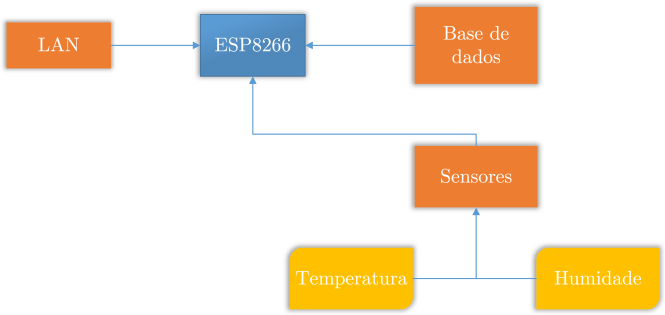


Figura 3.3: Estrutura do servidor

Cliente

O cliente é o responsável por permitir ao utilizador interagir com o edifício. Da sua estrutura (representada na figura 3.4) fazem parte dispositivos móveis (*tablet's* e *smartphones*) e computadores pessoais, que se ligam à área local, efetuando assim a comunicação com o servidor. No caso dos dispositivos móveis são utilizadas páginas

HTML como *interface* gráfica para o utilizador, enquanto que no caso do computador pessoal é usada uma aplicação desenvolvida em linguagem C#, com o auxílio do *software* Visual Studio 2012 e de um compilador da *Microsoft*, o .NET.

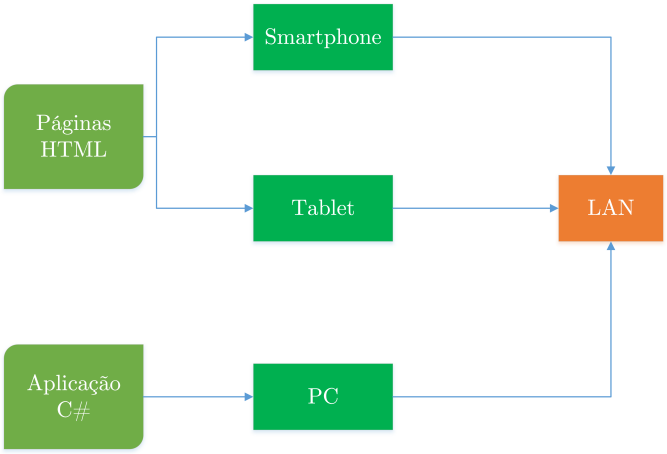


Figura 3.4: Estrutura do cliente

Capítulo 4

Implementação dos nodos ligados aos sensores/atuadores

Neste capítulo será apresentada a forma como se estabelecem as ligações entre os dispositivos constituintes da rede aos sensores e atuadores. Numa primeira abordagem será referido o *hardware* utilizado, em particular os componentes, as suas especificações e as suas funções. Posteriormente será explicado o *software* desenvolvido, quais as linguagens usadas e de que forma é feita a comunicação.

4.1 *Hardware* utilizado

O *hardware* utilizado para desenvolver a solução consistiu no ESP8266, embutido na placa de desenvolvimento da NodeMCU (NodeMCU ESP8266 Development Board) e no sensor de temperatura e humidade, DHT22. Relés, resistências, condensadores, motores DC, LED's, optoacopladores, reguladores de tensão e a placa branca foram também equipamentos auxiliares no desenvolvimento da solução.

4.1.1 ESP8266

O primeiro modelo do ESP8266 (ESP8266-01), criado pela AI-Thinker, surgiu em agosto de 2014 e tem vindo a ser produzido pela Espressif Systems, um fabricante Chinês sediado em Shanghai. Hoje em dia é comum encontrar o ESP8266 em diversas variantes que se encontram representadas na figura 4.1, onde os algarismos caracterizam cada variante da placa. Este dispositivo foi criado com o intuito de minimizar o consumo de energia, garantindo em simultâneo a fiabilidade de operação, de forma incorporada numa única placa muito compacta (figura 4.2). Esta placa possui não só a capacidade de operar de forma independente (*standalone*), mas também como *slave* de um *host* de um microcontrolador, por exemplo, para fazer a transmissão de dados desse microcontrolador para um ou mais dispositivos [22].

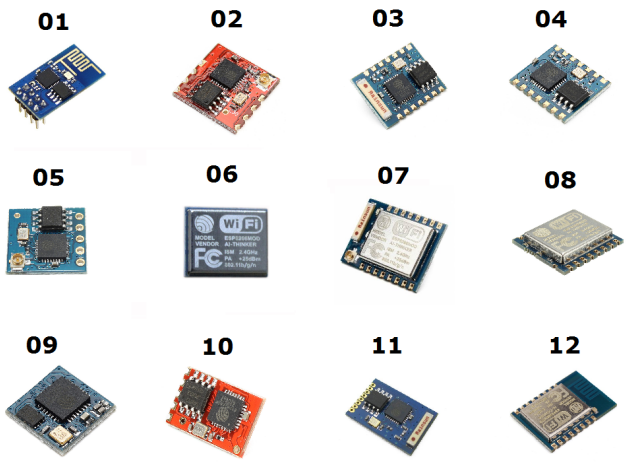


Figura 4.1: Vários modelos do ESP8266

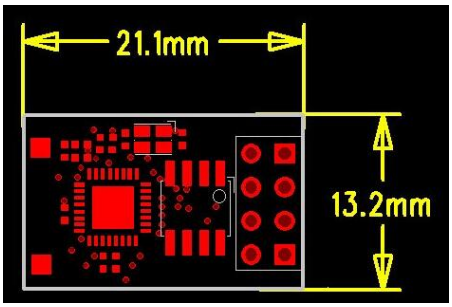


Figura 4.2: Dimensões externas da placa ESP8266-01 [23]

4.1.2 Especificações

- Protocolo *802.11 b/g/n*
- *wifi Direct* (P2P), *soft*-AP
- Pilha com protocolo TCP/IP integrado
- CPU de 32-bit e baixa voltagem incorporado
- SDIO 2.0, SPI, UART

4.1.3 Aplicações do ESP8266

- Domótica
- Electrodomésticos
- Iluminação
- Rede de Infraestruturas Locais

- Controlo Industrial Remoto
- Alarmes
- Câmaras IP
- Sensores de Rede
- Dispositivos de Localização Remota
- Etiquetas de Identificação
- Sistemas de Posicionamento *wifi*

4.1.4 NodeMCU

Para efeitos de teste recorreu-se à placa ilustrada na figura 4.3, que incorpora o modelo 12 do ESP8266, escolhida para agilizar o processo de desenvolvimento do *software* inerente à solução a desenvolver. No início do desenvolvimento da solução, foi usado o modelo 01 do ESP8266, no entanto, devido à disposição dos seus pinos não ser favorável à sua utilização na placa branca (foi necessário efetuar um suporte compatível com a placa branca) e ao facto das ligações cabladas muitas vezes não garantirem a corrente necessária ao dispositivo, optou-se por recorrer à NodeMCU ESP8266 Development Board. Ao usar o modelo 01 foi necessário usar dois botões para fazer *reset*, cortando momentaneamente a alimentação, para que o dispositivo fosse reconhecido pelo *software* "Esplorer", figura 4.4, usado para desenvolvimento do código Lua usado no projeto.

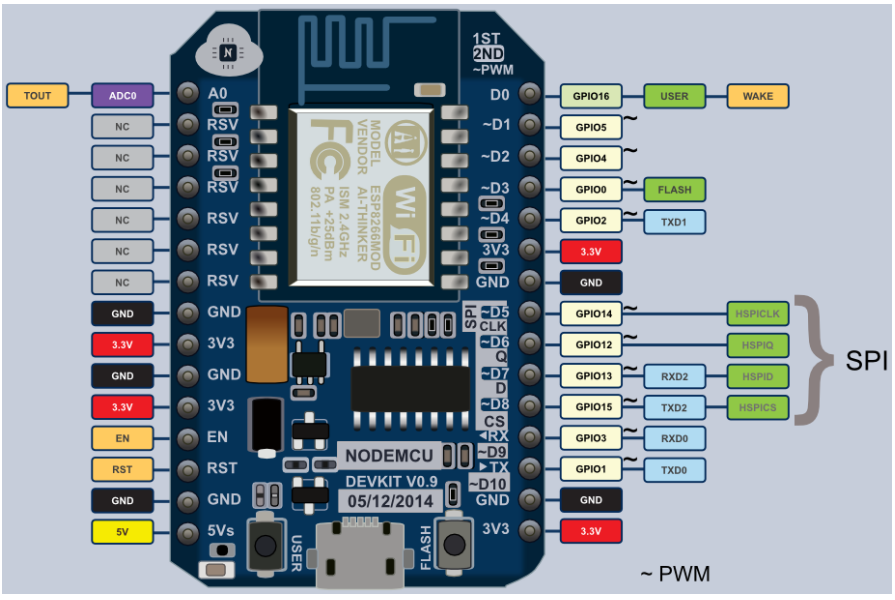


Figura 4.3: Disposição dos pinos da placa de desenvolvimento da NODEMCU [24]

A escolha pelo *software* "Esplorer" não foi uma decisão difícil de tomar, porque é um *software* inteiramente dedicado ao desenvolvimento de soluções com o ESP8266. Não só

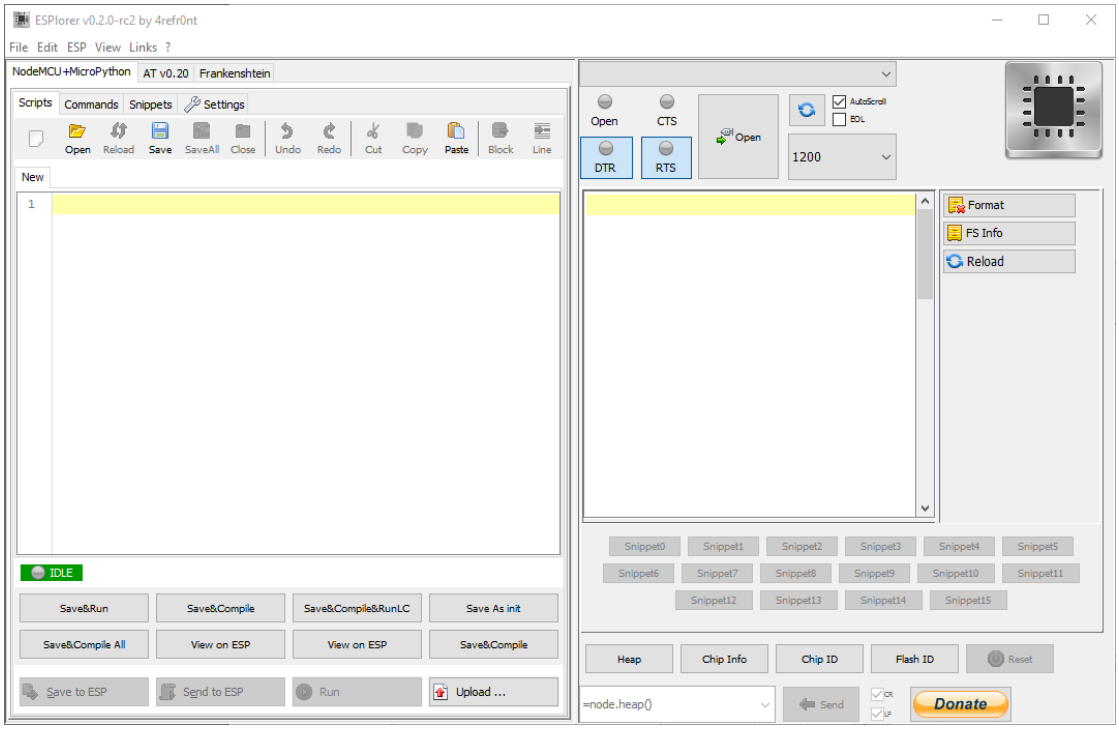


Figura 4.4: Aspeto do ambiente de desenvolvimento do código Lua para o ESP

possui o básico editor de texto e a consola de monitorização, como já contém comandos gravados à disposição do programador. O *Esplorer* permite usar tanto o *firmware* NODEMCU, como comandos AT.

As outras opções recaíam sobre os *software's* "LuaLoader" e "Arduino IDE". O primeiro dedicado a *scripts* escritos em linguagem Lua, foi entendido como um ambiente bastante limitado e com algumas falhas. O segundo, muito popular entre os utilizadores de Arduino, era uma solução a ter em conta, uma vez que contava com a existência de bibliotecas dedicadas para o uso do ESP8266 e muita informação à volta desta matéria. Todavia, revelou-se menos agradável à sua utilização relativamente ao Esplorer, pois não continha botões dedicados às funcionalidades do ESP8266, com as respetivas instruções predefinidas. De um modo sucinto, foi utilizada a placa NodeMCU ESP8266 Development Board, programada com linguagem Lua, através do ambiente de desenvolvimento *Esplorer*.

4.1.5 DHT22

O DHT22 (figura 4.5) é um sensor, cujas características vêm descritas na tabela 4.1, que é utilizado para medir a humidade relativa e temperatura do ar. Este sensor tem um sinal digital calibrado à saída, cujo elemento sensível se encontra ligado a um *chip* de 8 bits. Tamanho reduzido, baixo consumo energético e transmissão a longo alcance (20 metros), são particularidades que o tornam um dispositivo de fácil implementação em diversos sistemas de monitorização.

Este transdutor recorre a dois sensores para medir os valores físicos convertendo-os em grandezas elétricas, que possam ler lidas pelo ESP8266. O princípio de funcionamento

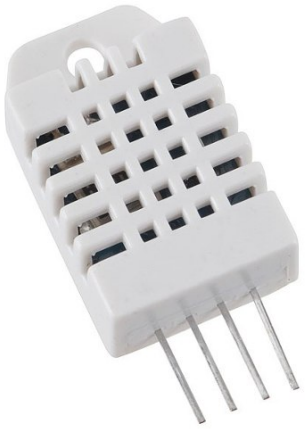


Figura 4.5: Sensor DHT22

Tabela 4.1: Características do DHT22 [25]

Modelo	**DHT22**
Fonte de energia	3.3 a 6V DC
Sinal de saída	Sinal digital através de barramento simples
Elemento sensível	Condensador polimérico
Humidade de Operação	0 a 100% RH
Temperatura de operação	-40 a 80 graus Celsius
Erros de medição	Humidade ±2% RH(Max ±5% RH)
	Temperatura < ±0.5 graus Celsius
Resolução/ sensibilidade	Humidade 0.1% RH
	Temperatura 0.1 graus Celsius
Repetibilidade	Humidade ±1% RH
	Temperatura ±0.2 graus Celsius
Histerese de humidade	±0.3% RH
Estabilidade	±0.5% RH/ano
Período sensível médio	2 segundos
Dimensões	Tamanho pequeno: 14 × 18 × 5.5 mm
	Tamanho grande: 22 × 28 x × mm

do DHT22 baseia-se na alteração de resistências. Para medir a humidade é usado um sensor constituído por dois elétrodos que envolvem um substrato, como mostra a figura 4.6.

À medida que a humidade se altera, a condutividade do substrato e a resistência entre os elétrodos é também alterada. Este efeito é então traduzido pelo circuito integrado do DHT22, seguindo posteriormente a informação para o ESP8266. No caso da temperatura é usado um termístor, um sensor que à medida que a sua temperatura é alterada, muda

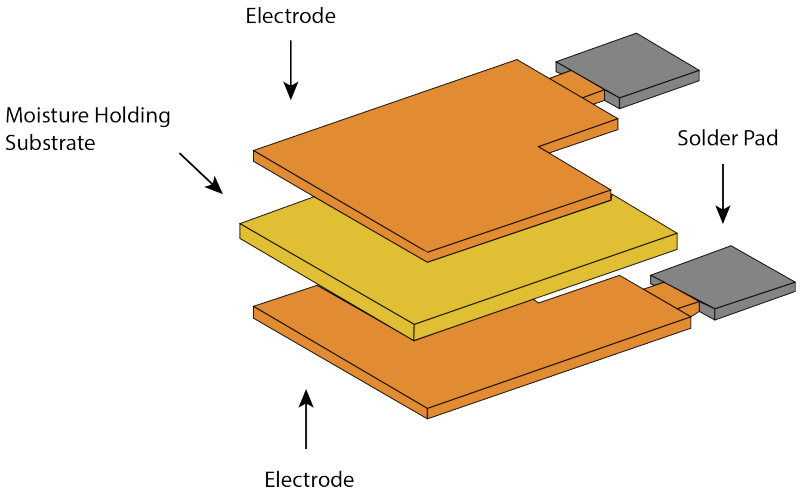


Figura 4.6: Sensor de humidade do DHT22 [26]

o valor da sua resistência. Desta forma o valor das grandezas físicas é transformado em sinais elétricos, para serem lidos pelo ESP8266 [26].

Os dados transmitidos são compostos por uma parte inteira e uma parte decimal, seguidas da fórmula para os dados. Em primeiro lugar, o DHT22 envia um sinal a 1, depois, seguem-se: 8 bits para a parte inteira de RH, 8 bits para a parte decimal de RH, 8 bits da parte integral para a temperatura, 8 bits da parte decimal para a temperatura e 8 bits para a checksum. Se os dados transmitidos forem os correctos, a checksum deve corresponder aos últimos 8 bits, seguindo a estrutura acima referida. Quando a MCU envia o sinal de início, o DHT22 muda do modo de poupança de energia para o modo de execução. Quando a MCU termina o envio do sinal de início, o DHT22 responde com um sinal de 40 bits de dados que devolvem a humidade relativa e a temperatura para a MCU [25].

Relativamente aos tempos de transmissão, nomeadamente aos tempos em que o sinal deve manter o estado, para garantir que o sensor consegue detetar o sinal da MCU, encontra-se informação mais detalhada no *datasheet* do dispositivo, mencionado nas referências bibliográficas.

4.1.6 Optoacopladores

Um optoacoplador, também chamado de optoisolador é um isolador ótico que, à semelhança dos transformadores, conecta dois circuitos elétricos que fisicamente se encontram separados.

Apesar de se tratar de um dispositivo simples, é extremamente vantajoso para fazer a ponte entre dois circuitos com características muito distintas, particularmente ao nível da diferença de potencial. O design mais básico de um optoacoplador, consiste num LED que produz um feixe luminoso e um dispositivo semicondutor fotossensível, cuja função passa por detetar o raio emitido. Estes componentes vendem-se envoltos numa caixa preta, com os respetivos contactos metálicos colocados para o exterior, como ilustrado na figura 4.7.

Existem diversos tipos de optoacopladores adequados a diferentes finalidades. Toda-

Figura 4.7: Optoacoplador [27]

via, todos eles utilizam um emissor de luz e um componente que lhe é sensível, sendo que esse componente pode ser: um foto-díodo, um foto-transístor, um foto-SCR, ou um foto-TRIAC. Para circuitos DC são vulgarmente utilizados os foto-transistor e para os circuitos AC os foto-TRIAC. Existem no entanto, outros tipos de configurações para além daquelas que são ilustradas na figura 4.8.

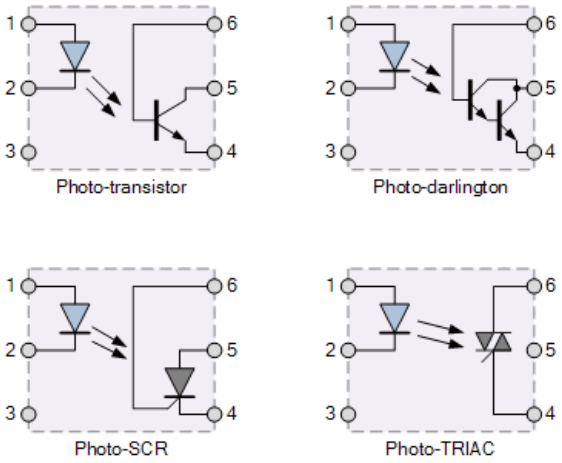


Figura 4.8: Diversos tipos de optoacopladores [27]

Tomando por exemplo a figura 4.9, a corrente da fonte passa através do LED de entrada, o qual emite o feixe luminoso, cuja intensidade é proporcional ao sinal elétrico. Esta luz incide na base do foto-transistor, alterando o seu estado e por conseguinte, passa a ser um meio condutor. Quando o feixe de luz do LED é interrompido, o foto-transistor deixa novamente de conduzir corrente elétrica.

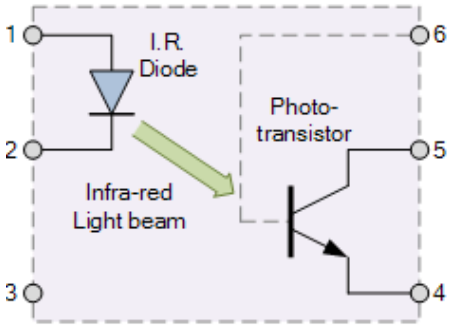


Figura 4.9: Interior de um optoacoplador (Electronic Tutorials, 2016)

A grande vantagem destes dispositivos passa por evitar que picos de tensão possam danificar os circuitos do sistema. Neste caso concreto, a finalidade do optoacoplador é proteger os dispositivos integrados, particularmente os ESP8266.

4.1.7 Localização dos dispositivos no interior da habitação

Na habitação existem 4 tipos de equipamento ou dispositivos que se pretendem controlar: sistemas de iluminação, sistemas AVAC, sistemas de rega e controlo de acessos. Para ligar/desligar a iluminação ou regular a sua luminosidade, o ESP8266 pode ser incorporado no interior do candeeiro, ou no circuito da rede de lâmpadas que ilumina uma divisão. Olhando para o caso do estore e do portão, ambos se baseiam no controlo de um motor elétrico, embora no caso do portão seja um motor com diferentes características, devido ao valor de binário necessário para fazer mover o portão, que se revela superior relativamente ao do estore. A função do ESP8266 será acionar o motor, fazendo com que gire no sentido pretendido, consoante se pretenda abrir, ou fechar o estore. Relativamente à aplicação do ESP8266 em unidades de controlo de ar condicionado, seria necessário efetuar um estudo dos circuitos dessas unidades para colocar o ESP8266 sem que os dois fossem danificados, ou cuja operabilidade ficasse posta em causa. Contudo, a ideia é passar os valores que estão a ser lidos por essa unidade para os dispositivos de controlo. Uma solução mais económica é usar apenas um sensor de temperatura que comunique com o ESP8266 e assim envie o valor da temperatura para o utilizador. Finalmente, no caso dos dispositivos que não possam integrar um ESP8266, há a possibilidade de usar o ESP8266 incorporado nas tomadas da rede elétrica, para cortar a sua iluminação, substituindo as tomadas inteligentes, como a que se faz representar na figura 4.10. Neste caso, é necessário recorrer a um transístor ligado a uma saída digital do ESP8266, que por sua vez faz atuar um relé, que se encontra ligado ao aparelho a controlar e que é alimentado pela rede elétrica. Os sistemas de rega são um tema que não será abordado, mas poderia ser implementado. O ESP8266 poderia, juntamente com relés, atuar os aspersores de rega, quando o utilizador assim o desejasse.

Para controlar dispositivos externos, ligados às tomadas da casa, a solução passa por incorporar na parte interior da tomada (no interior da parede) um ESP8266. Esse ESP vai usar uma das suas saídas digitais para poder assim atuar sobre um transístor, que por sua vez se encontra ligado a um relé, sendo o relé o responsável pela ativação da tomada. A necessidade do uso do relé baseia-se no facto de se estar a atuar sobre uma tomada que fornecerá 220V, através de sinais digitais com apenas 3.3V. De um modo

Figura 4.10: Tomada inteligente [28]

sucinto, dentro da tomada existe um ESP8266 alimentado externamente com 3.3V para o manter em funcionamento (consultar anexos). Ligado à saída digital a controlar, está a base do transístor, alimentado pela mesma fonte de energia que alimenta o ESP8266. Por sua vez, o emissor desse transístor liga-se a um relé que está alimentado pela rede elétrica e assim garante os 220V à saída da tomada.

4.2 *Software* desenvolvido

Nesta secção é descrito o *software* desenvolvido para a programação do ESP8266. Numa primeira parte é exposta a forma de como preparar o ESP8266 para funcionar com a linguagem de programação Lua, depois será abordada a comunicação com o sensor DHT22 e por fim, a forma como através do ESP8266, os dispositivos do edifício são atuados.

4.2.1 Linguagem Lua

A linguagem Lua foi criada em 1993, na Pontífica Universidade Católica do Rio de Janeiro, onde prossegue até aos dias de hoje o seu desenvolvimento. Trata-se de uma linguagem leve e eficaz, caracterizada pela rápida velocidade de execução que oferece aos programas nos quais se insere. A sua aplicação em sistemas embutidos tem sido uma mais valia para quem deseja adaptar código escrito noutras linguagens, como por exemplo: C, C++, Fortran, Java, C#, *etc.* Pode inclusivamente ser adaptada em *scripts* com as linguagens Pearl e Ruby. Esta facilidade surge no seguimento das suas API's, simples e práticas. Lua é facilmente extensível com bibliotecas escritas noutras linguagens e, apesar de não ser uma linguagem orientada a objetos, fornece mecanismos para implementar classes, mais especificamente, Lua oferece mecanismos para implementar construções ao invés de oferecer um número enorme de construções diretamente na linguagem. O pacote Lua é tão pequeno, que descompactado, ocupa apenas 1.1MB, isto incluindo já o código fonte e respetiva documentação.

Esta linguagem tem a vantagem de ser *opensource* e de ter uma forte compatibilidade com sistemas baseados em *Unix* e *Windows*, para além das plataformas para dispositivos móveis, casos da linguagem *Android*, IOS, BREW e *Windows Phone*, e ainda de microprocessadores embutidos, em particular com arquitetura ARM.

Para agilizar ainda mais as aplicações escritas nesta linguagem, existe uma variante do Lua, LuaJIT, que recorre a um compilador *just-in-time*.

A linguagem Lua apresenta uma interpretação do tipo dinâmico executando trechos de código, criados dinamicamente, no mesmo ambiente de execução do programa. Para além disso, faz verificação dos tipos de variável durante a execução do programa, sem que seja aplicada uma operação a um tipo incorreto. Outra característica é a função *garbage collection*, não sendo necessário ter o cuidado de gerir a memória, ou seja, o uso de comandos para libertar espaço é evitado.

De uma forma geral, as linguagens dinâmicas não são compiladas pelo código nativo da máquina, são apenas interpretadas, tratando-se de um caso de implementação dessas linguagens, e não das linguagens em si, desta forma, a implementação via interpretador é favorecida em detrimento da construção de compiladores [29].

4.2.2 Preparação do ESP8266 para utilização da linguagem Lua

Por defeito, o ESP8266 vem preparado para funcionar à custa de comandos AT, uma forma de comunicação desenvolvida por Dennis Hayes durante a década de oitenta, atualmente uma forma obsoleta para efetuar comunicações neste tipo de dispositivos e que requerem uma grande quantidade de instruções para realizar operações simples. Para usar a programação Lua com o ESP8266 foi necessário usar o *software ESP8266Flasher* (figura 4.11) para carregar o *firmware* da NodeMCU no ESP8266. Ao usar a NodeMCU ESP8266 Development Board esta acção viu-se facilitada, porque a placa já continha um módulo que efetuava a conversão de dados de USB, barramento universal do tipo *serial* usado nos computadores, para as *interfaces serial* usadas pelo ESP8266, ao passo que inicialmente foi necessário usar um conversor externo para o modelo 01 do ESP8266. Depois de carregado o *firmware*, o ESP8266 estava pronto a ser programado com o auxílio do *Esplorer*, já referido na subsecção 4.1.4.

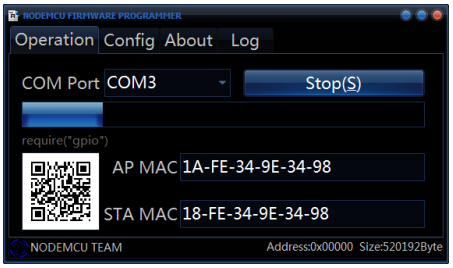


Figura 4.11: *Software ESP8266flasher*

Comunicação com o sensor DHT22

Para comunicar com o sensor DHT22 foram utilizados 3 *scripts* com código Lua, e dois com código PHP. Os códigos Lua foram transferidos para o ESP8266, que os executa de forma cíclica. O *script* "init.Lua" tem como função unicamente "chamar" o *script* "readdht22.Lua", que tem como função executar uma série de instruções baseadas no terceiro *script* ("dht22.Lua").

De um modo sucinto, o programa inicia com o *script* "init.Lua", onde inicialmente é feita a ligação ao *router*, como ilustrado na figura 4.12.

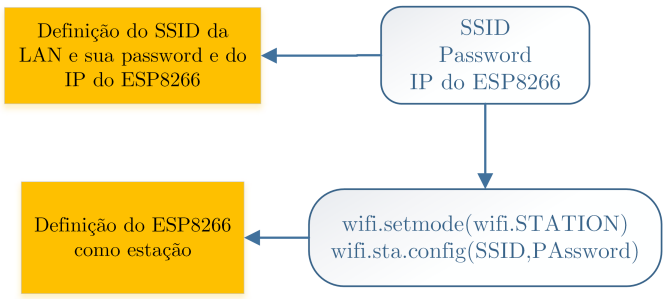


Figura 4.12: Ligação ao *router*

De seguida é utilizada uma função "sendData()", que numa primeira fase faz a chamada para o terceiro *script*. O dht22.Lua é um *script* que foi criado por uma comunidade aberta, a qual permite a sua partilha e utilização. Este código é dedicado especificamente ao sensor DHT22 e tem como função a leitura dos sinais vindos do sensor, recorrendo a ciclos *for* para interpretar as mensagens *bit* a *bit*. A figura 4.13 representa os estados dos 40 *bits* da mensagem, que ao nível da programação são representados pelo fluxograma da figura 4.14.

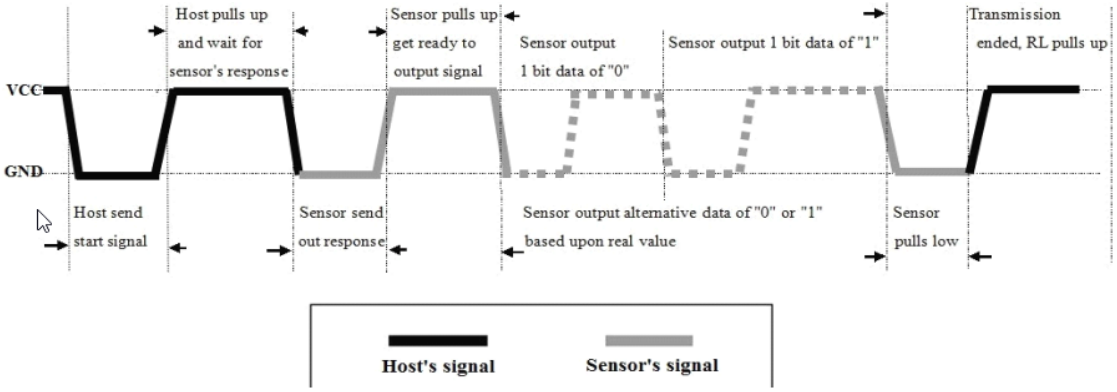


Figura 4.13: Estado dos *bits* durante a transmissão da mensagem no sensor DHT22 [25]

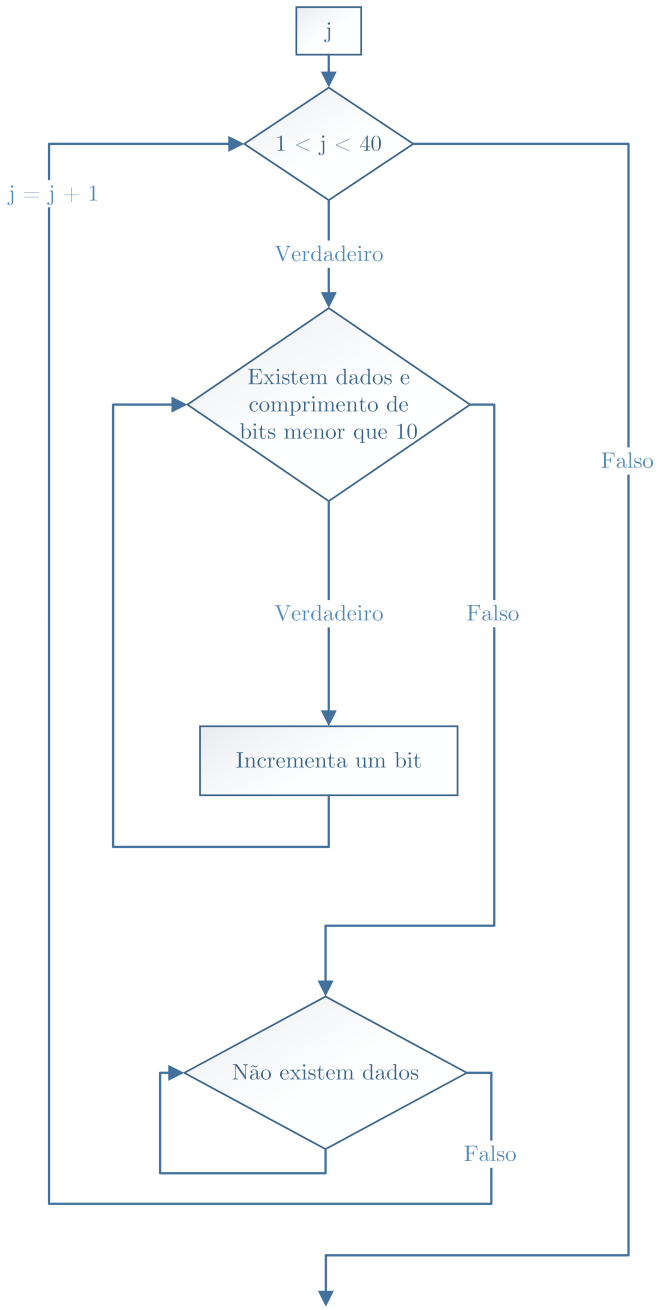


Figura 4.14: Ciclo *for* responsável por ler os 40 bits da mensagem do DHT22.

No final são retornados dois valores através da variável "M", como se pode verificar na figura 4.15.

```
--Retorna o valor da temperatura que vai ser chamado pelo readdht22.lua
function M.getTemperature()
  return temperature
end

--Retorna o valor da humidade que vai ser chamado pelo readdht22.lua
function M.getHumidity()
  return humidity
end

return M
```

Figura 4.15: Retorno dos sinais da temperatura e da humidade.

Retornados os valores, o programa volta à execução do readdht22.Lua, onde vai ler e converter os valores da variável "M" para serem interpretados nas unidades requeridas (excerto de código da figura 4.16).

```
t = dht22.getTemperature()
h = dht22.getHumidity()
humidade=(h/10).."."..(h%10)
temperatura=(t/10).."."..(t%10)
```

Figura 4.16: Leitura e conversão dos valores de temperatura e humidade.

Nesta fase, falta comunicar com a base de dados por forma a enviar os dados pretendidos. Para tal, considerando o excerto de código da figura 4.17, é observável que o código é em tudo semelhante àquele que é colocado na barra de endereços de um *browser*. A sua função é carregar o ficheiro "inserirValores.php", que por sua vez, se encontra alojado no servidor Apache.

```
conn:send("GET /inserirValores.php?temp="..temperatura.."&humi="..humidade.." HTTP/1.1\r\n")
conn:send("Host: IP\r\r")
conn:send("Accept: */*\r\n")
conn:send("User-Agent: Mozilla/4.0 (compatible; esp8266 Lua; Windows NT 5.1)\r\n")
conn:send("\r\n")
```

Figura 4.17: Mensagem HTTP a ser enviada ao *browser*.

Código para acionamento de dispositivos externos

Comparativamente ao código desenvolvido para o sensor DHT22, o código para o acionamento de dispositivos externos foi muito mais simples. Este código consiste simplesmente no envio de instruções ON/OFF para a *web*, usando o ESP8266 como meio de comunicação, onde se encontra um *script* com o código Lua. No interior deste código, é utilizada uma zona de memória (*buffer*), que contém uma sequência de caracteres (*string*), que é responsável pela criação de uma página *web* em HTML, conforme ilustrado na figura 4.18.

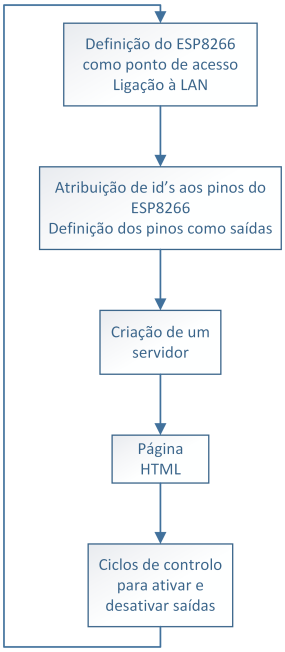


Figura 4.18: Incorporação de código HTML na execução do código Lua

4.2.3 Análise geral da solução

Como resultado da interligação de todos os dispositivos, o sistema final resultou como esquematizado na figura 4.19.

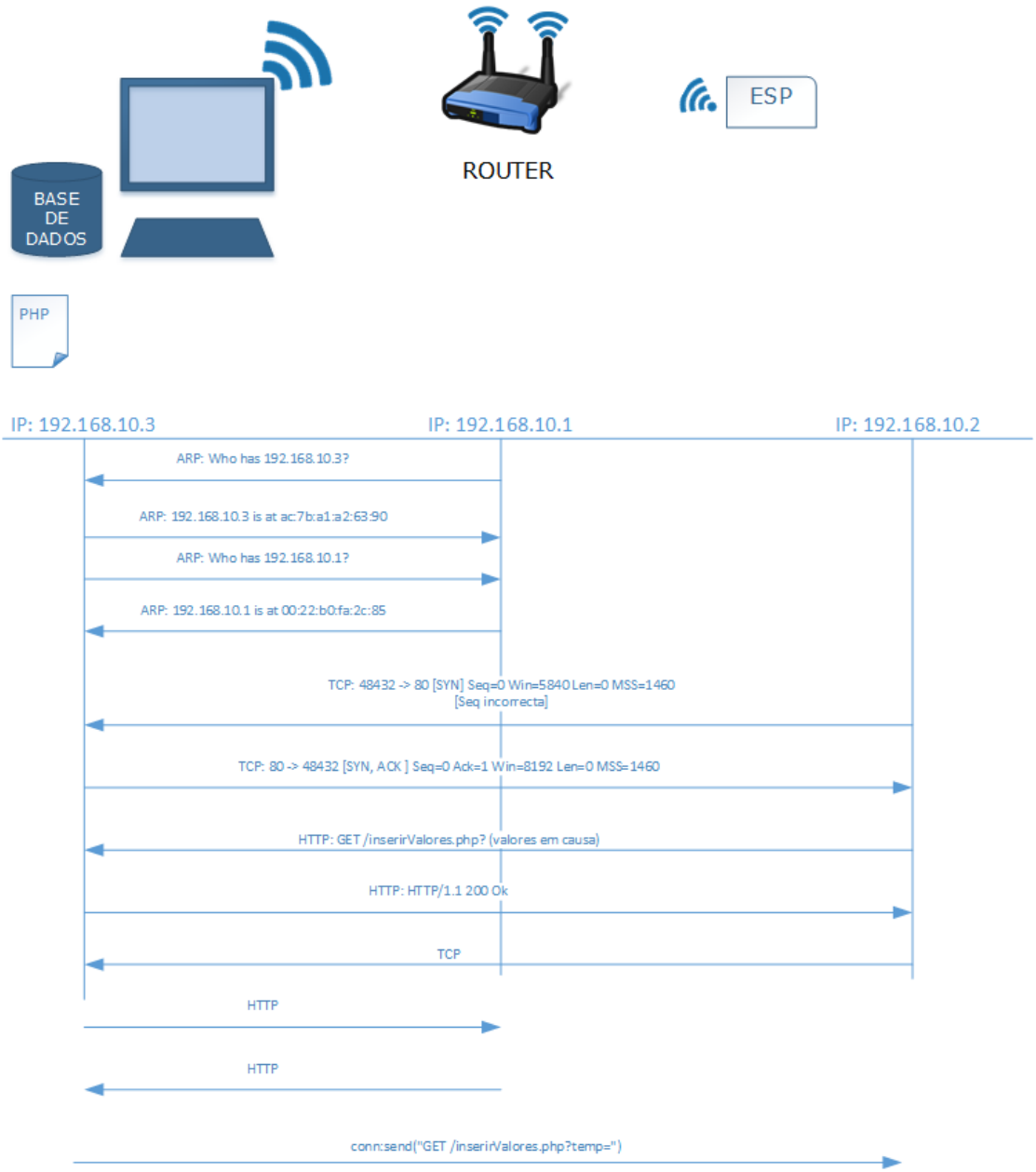


Figura 4.19: Linhas de comunicação

Em suma, a implementação da solução recorre a 5 diferentes plataformas. Em primeiro lugar, como centro de toda a comunicação reside uma base de dados MySQL, que pode ser utilizada sem qualquer custo financeiro. É através desta base de dados, que todos os registos inerentes ao sistema físico circulam, sendo esta a forma encontrada para comunicar entre as diversas plataformas da presente solução. No que concerne ao

ESP8266, a sua programação é feita ao abrigo do *firmware* NODEMCU, que funciona sobre linguagem Lua e a qual é usada igualmente para os *scripts* que fazem parte dos ficheiros a usar pelo ESP8266, ficheiros esses que chamam outros ficheiros do tipo PHP, usados para enviar *queries* MySQL. Com a aplicação desenvolvida em C#, juntamente com instruções HTML, o utilizador passa a controlar de uma forma cómoda todo o sistema, sem a necessidade de saber todas as instruções que correm em *background*. Uma vantagem desta aplicação passa pelo uso da plataforma *windows*, a mais familiar à maior parte dos cidadãos. A compatibilidade com a base de dados MySQL é também uma mais valia, pois não só é a forma de comunicar entre as diferentes plataformas da aplicação, bem como é uma forma de o utilizador guardar os dados ao longo do tempo.

Capítulo 5

Implementação do sistema de gestão e *interface* com o utilizador

5.1 Ativação de saídas

O sistema permite ao utilizador comandar o seu sistema de domótica através de um computador pessoal com plataforma *windows*. Para ativar uma saída, por exemplo, acionar um motor para levantar um estore, o utilizador prime o botão "Ligar" presente num *software* desenvolvido em linguagem C#. Entretanto, do lado do ESP8266, que está em permanente escuta e ligado à rede *wifi* do edifício, é lançado um servidor TCP, que alberga uma página HTML incorporada no código Lua, no interior desse ESP8266. Primido o botão, é chamada uma função integrada no código do programa que está a correr em *windows*, que por sua vez simula a chamada de uma página *web*, cujo IP corresponde ao IP do ESP8266 e onde vem inserida a instrução correspondente ao botão que o utilizador premiu. Através de dessa instrução é ativada uma *tag*, que não é nada mais do que uma referência, que na continuação da sequência do programa Lua vai será usada dentro de uma estrutura de controlo *else/if*, que vai ativar a saída correspondente. Em simultâneo, através de funções contidas no código C#, nomeadamente à custa da função "MySqlWrite", que será abordada na secção 5.3.1.

5.2 Leitura dos dados da base de dados

Uma das propriedades do programa, para além dos estados das saídas é a monitorização da temperatura ambiente e humidade relativa. Para tal, existe um DHT22 ligado aos pinos do ESP8266, que faz a leitura desses dados. Esse ESP8266 contém três programas desenvolvidos em código Lua. O primeiro, intitulado de "dht22.Lua" que na verdade se trata de uma biblioteca desenvolvida por terceiros e é responsável por traduzir os dados que são enviados do DHT22 para o ESP8266; o segundo, com o nome "readdht22.Lua" que contém duas funções, "sendData" e "getData", responsáveis por enviar os dados do sensor para a base de dados e receber os dados, nessa mesma base, inseridos no ciclo anterior. Para comunicar com a base de dados, estas duas funções recorrem a páginas PHP localizadas no interior do servidor local, dentro da pasta "htdocs", localizada no interior da instalação do XAMPP; por último, o *script* "init.Lua", cuja função passa por chamar ciclicamente o ficheiro "readdht22.Lua".

5.3 Aplicação gráfica em C#

Como forma de oferecer uma solução para uso em computadores pessoais foi criada uma aplicação, baseada na linguagem C# com recurso ao *software* "Visual Studio 2012 Premium", e ao compilador .NET.

A solução consiste num programa desenvolvido, figura 5.1, que permite ao utilizador comunicar com o ESP8266 e com a base de dados do sistema, localizada num servidor, que para teste se trata de um servidor local, mas que provou funcionar igualmente com servidores remotos.

A janela principal possui dois painéis horizontais. O painel superior encontra-se dividido em três colunas. A da esquerda, é responsável por conectar o programa à base de dados e verificar se a ligação foi feita com sucesso, o painel do centro, devolve os últimos valores lidos da temperatura do ambiente e da sua humidade relativa, finalmente, o painel da direita permite ao utilizador controlar um motor elétrico, que pode estar acoplado, por exemplo a um estore, permitindo controlar os seus movimentos ascendente e descendente. Quanto ao painel inferior, este remonta para o controlo dos botões "On/Off" relativos à iluminação. Existe também um menu básico, onde o utilizador pode consultar informações relacionadas com o programa e uma breve ajuda do como o operar.

Figura 5.1: Aspeto da aplicação para controlo e monitorização de um sistema de domótica

Iniciado o programa, o utilizador deve estabelecer a conexão à base de dados, caso contrário, os botões apesar de ativos, devolverão uma mensagem de erro relacionado com a conexão à base de dados. Efetuada a conexão, aparece a mensagem "conectado com sucesso" e o símbolo de "tick" a fundo verde. Nesta fase, são devolvidos os valores que

se encontram nas tabelas da base de dados (figura 5.2), bem como a data da última atualização dos valores presentes na base de dados, representada abaixo da versão do programa.

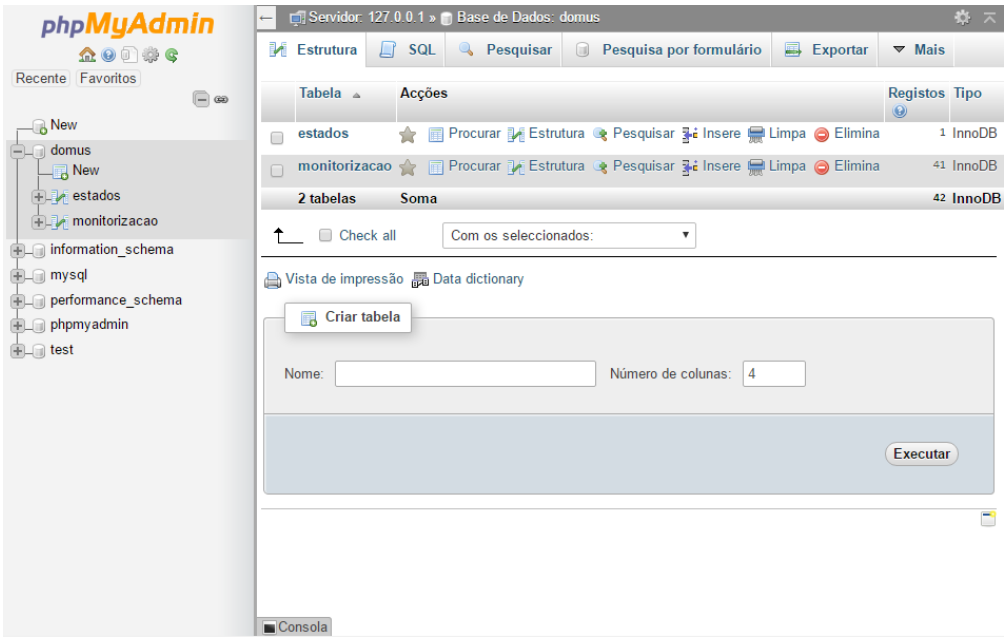


Figura 5.2: Aspeto da janela MySQL

5.3.1 Descrição do funcionamento do programa

O código do programa encontra-se dividido em 3 janelas: a principal, que foi abordada no início da presente secção e mais duas auxiliares, uma que contém informações sobre o programa e a que indica ao utilizador como trabalhar com o programa.

As janelas auxiliares resumem-se a caixas de texto com informações e não requerem qualquer explicação. Em relação ao código da janela principal, este estrutura-se em 16 funções: 8 para eventos de clique em botões, 3 para o menu e 4 funções criadas para simplificar o código, por forma a o torná-lo mais compacto.

A função principal resume-se a inicializar a janela principal e as funções de menu a inicializar as janelas auxiliares, não carecendo de qualquer explicação.

As 8 funções para os eventos de clique subdividem-se em dois tipos. As funções relacionadas com os botões de "On/Off" são seis, pois existem dois botões por *led*, o que liga e o que desliga.

Considere-se o evento "ligar *led* verde", nesse caso ao clicar no botão ON para o *led* verde é chamada uma função, que atribui um valor a uma *string*, que fica guardada em memória para ser utilizada mais tarde e que fica com a informação de que o botão foi premido. Caso a conexão à base de dados tenha sido estabelecida com sucesso, é executada uma função de escrita, que será explicada mais à frente. De seguida, é chamada uma página HTML com o IP do ESP8266, cuja função é ligar o *led* verde, e é imposta na mesma *string* do IP, figura 5.3, mas que o programa prevê que não seja aberta no *browser*, uma vez que não há essa necessidade e o *led* acende. Caso não seja respeitada a

condição de que a conexão à base de dados foi estabelecida é lançada uma mensagem de alerta para o utilizador. O código para as restantes 5 funções relacionadas com o ligar e o desligar dos *leds* funcionam da mesma forma, com a diferença da instrução que segue ao IP do ESP8266, fator que diferencia os diferentes objetivos dessas 6 funções.

```
string content = client.DownloadString("http://192.168.1.73/?pin=ON3");
```

Figura 5.3: *String* que contém o IP do ESP8266 e a instrução para a função pretendida

Atente-se agora na função, que tem como objetivo ligar e desligar um motor. A sua estrutura assemelha-se às funções utilizadas nos botões dos *leds*, com a diferença de que utiliza apenas um botão para ligar e desligar, utilizando como referência a inscrição do próprio botão. Inclui igualmente o alerta, caso a ligação à base de dados não tenha sido estabelecida.

Relativamente ao botão para estabelecer a conexão à base de dados, ao ser clicado é executada uma função, que no caso do seu texto conter a legenda "Conectar" executa as inscrições para iniciar a conexão. Especificamente executa duas funções auxiliares. Ambas serão explicadas mais à frente. Caso o texto do botão de conexão contenha a palavra "Desconectar", a mesma função encarregue da ligação à base de dados, vai neste caso desligar essaa ligação e devolve os valores originais do programa, sem afetar as informações da base de dados, através de uma outra função.

Passando agora à função de escrita na base de dados, a sua estrutura é definida pelo código responsável pela ligação à base de dados, por um ciclo *switch case* e pela verificação dos últimos estados das variáveis. Para efetuar a ligação à base MySQL é necessário instalar um *software* que permita a conexão com a mesma, que neste caso concreto se tratou do "MySQL Connector Net 6.9.9". Depois é necessário ativar as referências "MySQL" dentro do *Visual Studio*, usar a instrução "using MySql.Data.MysqlClient" e as funções MySQL necessárias ficam ativas, como mostram as palavras destacadas a azul claro na figura 5.4. Estabelecida a ligação, o código segue para a função *switch case*, onde é utilizada a *string* "botão" para diferenciar as instruções requeridas pelo utilizador. Consoante essa *string*, o programa coloca o valor "1" para designar "led ligado" e "0" para designar "led desligado", na cor do *led* correspondente. Apenas um valor é colocado por linha na tabela, permitindo ao utilizador saber a data em que esse *led* foi ativado ou desativado. Terminada a estrutura de controlo *switch case*, o programa fecha a conexão à base de dados e corre uma função que verifica os últimos estados das variáveis.

A função de verificação de estados estabelece igualmente a ligação à base de dados e guarda numa *string* o número de linhas atuais da tabela "estados" da base de dados, figura 5.5. De seguida, passa à leitura dos dados existentes na última linha da tabela "estados". Depois, através das condições de controlo: *if/else, while e do/while* é verificado se existe valor "0", ou "1" e caso não exista passa para a linha anterior até encontrar um valor. O objetivo da leitura se efetuar nesta sequência e com estas condições de controlo serve para a base de dados retornar o último estado válido, considerando inválido a inexistência de dados, ou valores diferentes de "0" ou "1", o que neste último caso nunca se verificaria. A inexistência de dados deve-se ao facto de que cada vez que o utilizador altera o estado de uma variável, apenas a coluna correspondente a essa variável escreve os valores "0", ou "1", daí a necessidade de executar o código da forma supracitada. Note-se que a *string* que contém o valor total das colunas é de extrema importância para ser utilizada nos contadores, após a conversão para o tipo "inteiro", permitindo dinamicamente aceder às

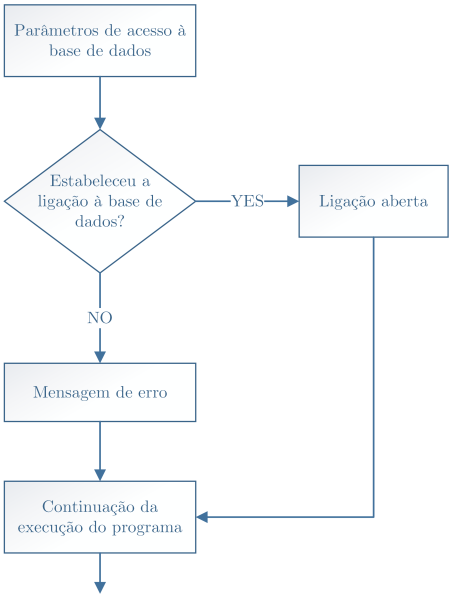


Figura 5.4: Fluxograma relativo à ligação à base de dados

linhas desejadas, independente do número total de linhas que a tabela contém, uma vez que o seu número é sempre alterado à medida que o utilizador altera o estado de uma variável. Outra nota a referir é a utilização de uma *string*, que carece de conversão e a não utilização de uma variável do tipo inteiro diretamente. Esta consequência justifica-se pela forma como as *queries* MySQL são enviadas, de forma sucinta, cada instrução para a base de dados segue numa *query*, cujo formato é precisamente uma *string*.

```
string numberOfRows = "SELECT COUNT(*) FROM estados";
```

Figura 5.5: *Query* que conta o número total de linhas da tabela "estados"

Para terminar, é também de referir que consoante os estados na tabela, também a aparência da janela se altera, para permitir ao utilizador uma fácil monitorização do sistema. Essas alterações são processadas simplesmente através de estruturas de controlo *if/else* e comandos inerentes à linguagem C#.

Finalmente, resta explicar agora a função de leitura da tabelas da base de dados. Esta função contempla igualmente as instruções de ligação à base de dados, mas no entanto é utilizada para ler os dados da tabela "monitorização". Através das instruções do ciclo *while*, representado no fluxograma da figura 5.6, são retornados os valores existentes na tabela, correspondentes à temperatura ambiente, humidade relativa e data de escrita desses mesmos dados. Esses valores são convertidos para o tipo *string* e utilizados para alterar os valores na janela, para sua consulta.

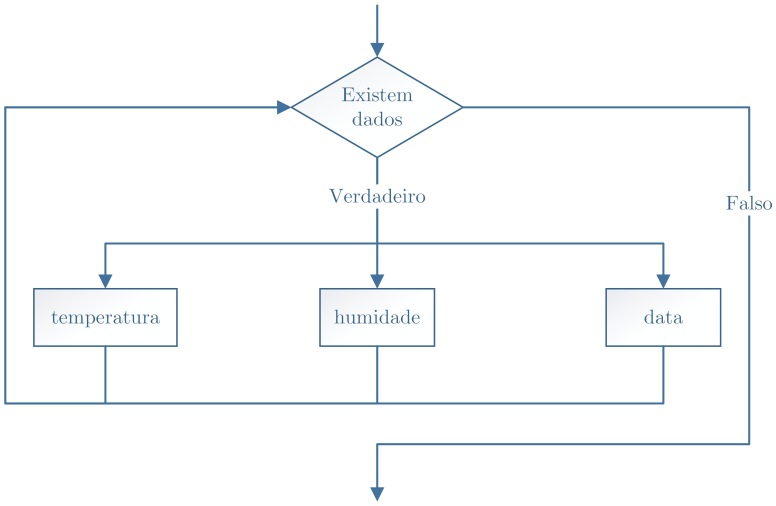


Figura 5.6: Leitura dos dados da tabela "monitorizacao" e alteração dos valores na janela

O restante código resume-se à alteração da *interface* gráfica consoante os dados que vão sendo monitorizados pelo ESP8266 e vão chegando às tabelas da base de dados, ou que através desta mesma *interface*, constituinte da aplicação em C# são controlados.

Capítulo 6

Resultados

Neste capítulo serão demonstrados os resultados obtidos. De forma resumida foi desenvolvida uma forma de comunicação entre o ESP8266 e uma base de dados MySQL, que pode ser acedida através de uma aplicação em C#, também desenvolvida pelo autor.

6.1 Análise da solução desenvolvida

Para validar a solução foi criado um protótipo, cuja configuração se representa na figura 6.1. Como se observa nessa figura, no protótipo físico são controlados os estados de 3 LED's e de um motor. Para efeitos de monitorização foi utilizado o sensor DHT22, que monitoriza a temperatura e a humidade.

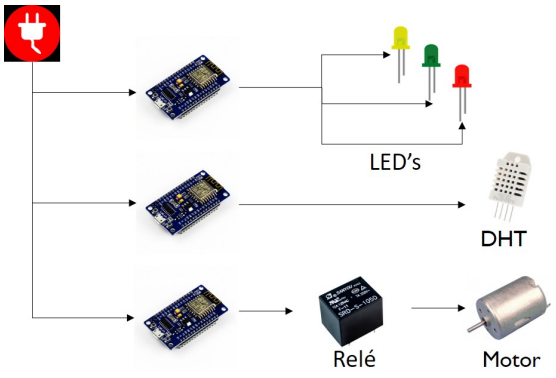


Figura 6.1: Esquema do protótipo

Para efeitos de controlo, criando interação com o utilizador, foi usada a aplicação desenvolvida em C#, que apresenta o aspeto da figura 6.2, quando a ligação à base de dados não está ativa. É de notar que os indicadores dos LED's não têm qualquer significado nesta fase. Para além disso, a data da última atualização dos dados, o valor da temperatura e o valor da humidade são representados por três ponte de interrogação.

Figura 6.2: Ligação à base de dados não efetuada

Se o utilizador tentar atuar sobre os LED's, ou sobre o motor com a ligação à base de dados desligada, aparecerá o alerta da figura 6.3.

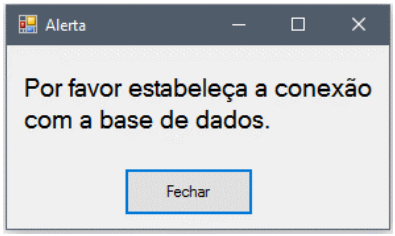


Figura 6.3: Janela de alerta

Caso o utilizador aceda à base de dados, aparecerá a janela da figura 6.4.

Figura 6.4: Ligação à base de dados ativa

Nesta janela, já é visível a mensagem "conectado com sucesso", os valores de tempe-

ratura e humidade e a data da última atualização. Como se pode ver ainda na figura 6.4, o motor está desligado (caso contrário, o botão que contém o texto "ligar", conteria a palavra "desligar") e os LED's verde e amarelo estão acesos. Portanto, está-se perante o estado representado no protótipo físico (figura 6.5) e na tabela *mysql* da figura 6.7. Chama-se a atenção para a luminosidade dos LED's verde e amarelo e também para os estados a "1" envolvidos pelos retângulos a verde, respetivamente nessas figuras.

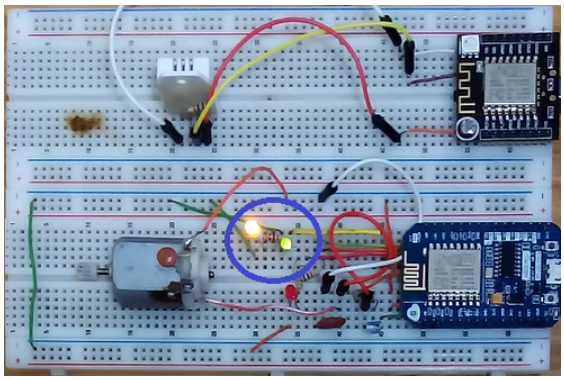


Figura 6.5: Protótipo físico em funcionamento

	id	Vermelho	Amarelo	Verde	Motor
Edita Copiar Apagar	730	0	0	0	0
Edita Copiar Apagar	731			1	
Edita Copiar Apagar	732		1		
Edita Copiar Apagar	733	1			
Edita Copiar Apagar	734				1
Edita Copiar Apagar	735				0
Edita Copiar Apagar	736	0			
Edita Copiar Apagar	737		0		
Edita Copiar Apagar	738			0	

Figura 6.6: Janela de monitorização da temperatura e humidade

Para terminar, na figura 6.6 são apresentados os valores monitorizados da temperatura e da humidade, com intervalos de 10 segundos entre si. Note-se que esses valores não foram tirados na mesma altura do momento da figura 6.4, facto verificável pelos valores diferentes.

	Data	Temperatura	Humidade
Edita Copiar Apagar	2016-11-24 17:33:27	21.3	42.2
Edita Copiar Apagar	2016-12-17 18:41:47	19.1	55.4
Edita Copiar Apagar	2016-12-17 18:41:57	19.1	56.1
Edita Copiar Apagar	2016-12-17 18:42:07	19.1	56.0
Edita Copiar Apagar	2016-12-17 18:42:17	19.1	55.9
Edita Copiar Apagar	2016-12-17 18:42:27	19.1	55.8

Figura 6.7: Janela dos estados dos dispositivos

Capítulo 7

Conclusão e trabalhos futuros

7.1 Conclusões

Em suma, pode afirmar-se que os objetivos propostos foram atingidos com sucesso. Com a solução proposta foi possível criar uma solução económica, sem afetar a estética do edifício devido às suas dimensões e posicionamento, com um meio de propagação de rede usual, a fim de não acrescer custos extra ao consumidor e evitar a aquisição de um leque alargado de novos materiais de um fornecedor exclusivo. Com base no estudo efetuado no capítulo 2, as soluções de domótica atuais apresentavam aspetos negativos que se refletiam, maioritariamente, em custos acrescidos para o consumidor. Para além disso, obrigavam o cliente a optar por uma determinada marca, fruto da exclusividade das ofertas. Para solucionar esse facto, a solução proposta permite usufruir de equipamentos já adquiridos, implementando os novos dispositivos em soluções já existentes sem grandes custos de aquisição. No que concerne ao meio de transmissão de dados, ao invés de usar protocolos embebidos nos dispositivos criados pelas empresas, a solução encontrada foi o recurso à rede *wifi 802.11* existente na maioria das habitações contemporâneas.

7.2 Trabalhos futuros

Relativamente a trabalhos futuros, seria interessante criar uma *interface* gráfica, que permitisse ao utilizador facilmente monitorizar e controlar os dispositivos da sua habitação, edifício de uma companhia de serviços, ou outro qualquer edifício que tivesse o sistema implementado, para dispositivos móveis (*tablets e smartphones*). Para além disso, criar uma biblioteca definitiva para utilizar a programação Lua embebida nas novas versões de PHP e também criar uma forma de comunicar com uma base de dados MySQL, através da linguagem Lua, inserida no ESP8266. Um outro ponto importante, que pode ser promovido também a trabalho futuro, é a segurança das redes que operam no sistema proposto.

Bibliografia

[1] K. Castro, "Domótica: desenvolvimento de uma solução integradora," dissertação de mestrado, Departamento de Engenharia Mecânica, Universidade de Aveiro, 2012.

[2] Allen S., Kuramochi, Y., *et al.*, *Easy X10 Projects for Creating a Smart Home*. 2005.

[3] J. Burroughs, *X-10 Home Automation Using the PIC16F877A*. Microchip Technology Inc., 2002.

[4] Altech, "Din rail terminal blocks." `http://www.altechcorp.com/HTML/DIN_Rail_Blocks-A.html`, 2014. [Em linha; acedido a 19-Agosto-2016].

[5] S. H. USA, "How X10 Works." `http://www.smarthomeusa.com/how-x10-works/`. [Em linha; acedido a 2-Março-2016].

[6] KNX, *System-architecture: The-One-Single-Standard*, vol. 32. 2014.

[7] Berker, "Catalogue 2014." `http://hagerv7.interactivdoc.fr/viewer/14gb_berker_catalogue_pdf_294/`, 2014. [Em linha; acedido a 3-Março-2016].

[8] KNX Association, *KNX Solutions*, vol. 5174 of *Lecture Notes in Computer Science*. Indianapolis, USA: Sams Technical Publishing, 2012.

[9] P. Mein, "mein Paul." `https://www.knx.org/knx-en/knx/association/introduction/index.php`, 2014. [Em linha; acedido a 2-Fevereiro-2016].

[10] KNX, "Ets5 features." `https://www.knx.org/ie/software/ets/features/index.php`, 2015. [Em linha; acedido a 3-Março-2016].

[11] Newron System, "KNX USB Interface - Newron System - Innovative Solutions for BMS Architectures." `http://www.newron-system.com/KNX-USB-310?lang=en`, 2012. [Em linha; acedido a 6-Março-2016].

[12] T. F. Silva, "Desenvolvimento de um sistema domótico: monitorização e controlo das infraestruturas eletrónicas de uma habitação," dissertação de mestrado, Escola Superior de Tecnologia de Tomar, Instituto Politécnico de Tomar, 2013.

[13] D. Mata, "Domótica: Protocolo de comunicação lonworks." `http://www.prof2000.pt/users/lpa`, 2014. [Em linha; acedido a 15-Fevereiro-2016].

[14] CS3, "Common protocols in HVAC." `http://www.cs3.com/index.cfm/mainpages/2/sitepages/show/77`, 2010. [Em linha; acedido a 3-Março-2016].

[15] Echelon, "Neuron 5000 processor." http://www.echelon.com/assets/blt893b4cd-a48d3cda3/Neuron-5000-Processor-datasheet.pdf, 2014. [Em linha; acedido a 10-Março-2016].

[16] P. Kenneth, "Introduction to cebus comunications protocol." http://www.hometoys.com/content.php?url=/htinews/aug97/articles/kwacks/kwacks.htm, 1997. [Em linha; acedido a 3-Março-2016].

[17] A. Ferreira, "Desenvolvimento de infra-estrutura de comando multifunções EIB-KNX para smartphone," relatório final de projeto, Instituto Politécnico de Bragança, Bragança, Portugal, Rel. 1, Bragança, 2012.

[18] A. Gama, "Domótica - Sistema Inteligente de Controlo e Monotorização," dissertação de mestrado, Departamento de Engenharia Mecânica, Universidade de Aveiro, 2014.

[19] F. Ding, A. Song, E. Tong, and J. Li, "A Smart Gateway Architecture for Improving Efficiency of Home Network Applications," *Journal of Sensors*, vol. 2016, pp. 1–10, Set 2015.

[20] Nordic, "nrf24l01+ overview." http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P/(language)/eng-GB, 2013. [Em linha; acedido a 20-Julho-2016].

[21] Nordic, "Enhanced shockburst user guide." https://devzone.nordicsemi.com/documentation/nrf51/4.3.0/html/-groupesbusersguide.html, 2013. [Em linha; acedido a 20-Julho-2016].

[22] S. Espressif, "ESP8266 Technical Reference." https://espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf, 2016. [Em linha; acedido a 30-Agosto-2016].

[23] Anibit, "WiFi Serial Transceiver Module w/ ESP8266, ESP-01 edition | Anibit Technology." https://anibit.com/product/wls01001. [Em linha; acedido a 15-Março-2016].

[24] F. Core, "mein Paul." http://www.meinpaul.de/index.php/paul-persoen-/licher-assistent-technische-details, 2016. [Em linha; acedido a 6 Março-2016].

[25] T. Liu, *Digital-output relative humidity & temperature sensor/module*. Aosong Electronics Co.,Ltd, 2014.

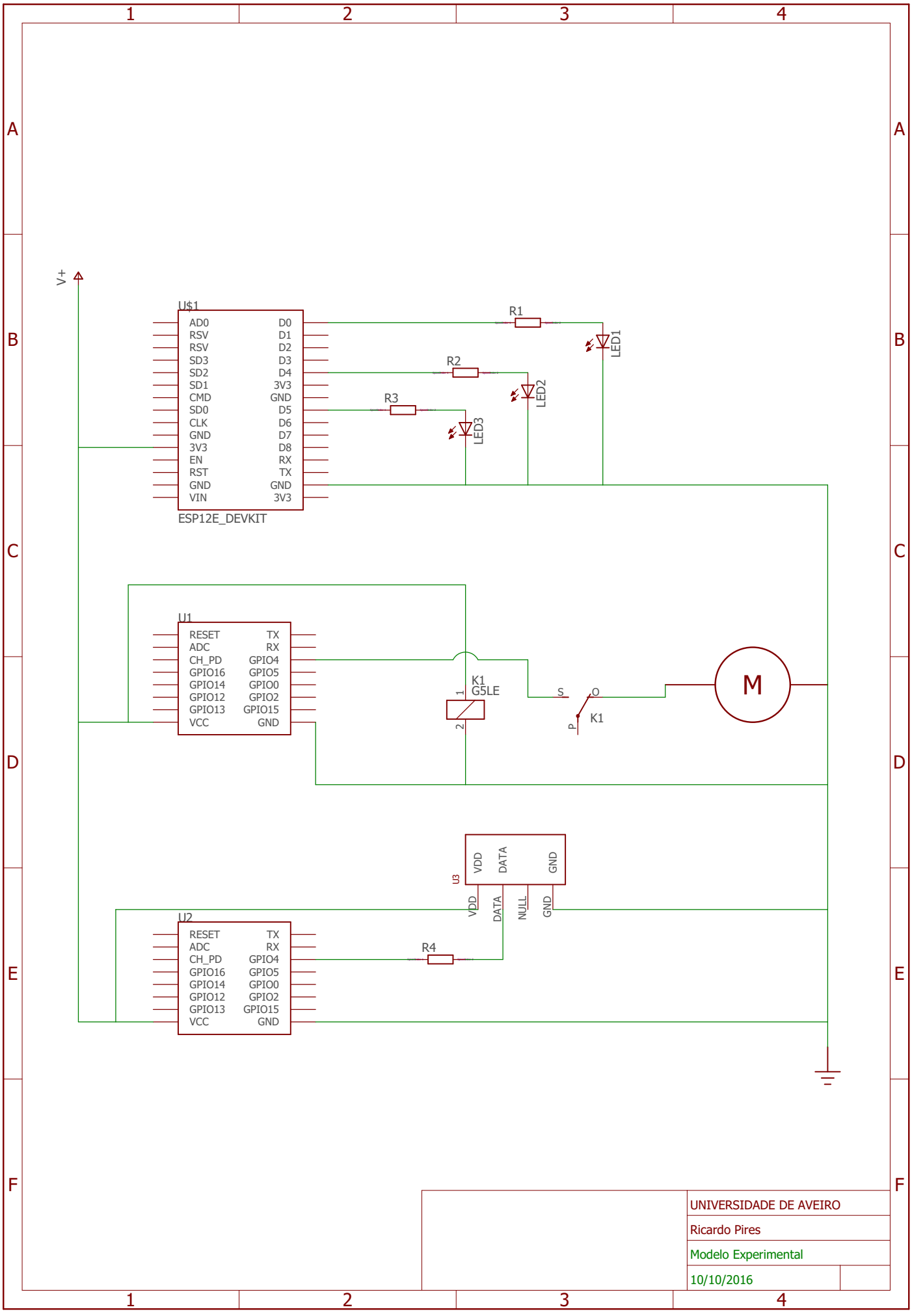
[26] D. Nedelkovski, "Dht11 & dht22 sensors temperature and humidity tutorial using arduino." http://howtomechatronics.com/tutorials/arduino/dht11-dht22-sensorstemperature-and-humidity-tutorial-using-arduino/, 2016. [Em linha; acedido a 3-Agosto-2016].

[27] E. Tutorials, "Optocoupler tutorial." http://www.electronics-tutorials.ws/blog/optocoupler.html, 2016. [Em linha; acedido a 28-Julho-2016].

[28] DLink, "Smart home starter kit dch-100kt." http://www.dlink.com/pt/pt/home-solutions/mydlink-home/smartplugs/dch-100kt-smart-home-hd-starter-kit, 2016. [Em linha; acedido a 20-Junho-2016].

[29] PUC-Rio, "Lua 5.2 reference manual." http://www.lua.org/manual/5.2/, 2015. [Em linha; acedido a 12-Maio-2016].

ANEXOS


V+
U$1
AD0
RSV
RSV
SD3
SD2
SD1
CMD
SD0
CLK
GND
3V3
EN
RST
GND
VIN
D0
D1
D2
D3
D4
3V3
GND
D5
D6
D7
D8
RX
TX
GND
3V3
ESP12E_DEVKIT
R1
LED1
R2
LED2
R3
LED3
U1
RESET
ADC
CH_PD
GPIO16
GPIO14
GPIO12
GPIO13
VCC
TX
RX
GPIO4
GPIO5
GPIO0
GPIO2
GPIO15
GND
K1
G5LE
1
2
S
O
P
K1
M
U3
VDD
DATA
GND
VDD
DATA
NULL
GND
U2
R4
UNIVERSIDADE DE AVEIRO
Ricardo Pires
Modelo Experimental
10/10/2016