Universidade do Minho
Escola de Engenharia

João Luís Neto Guimarães Pereira

Interface Gráfico e Módulo de Controlo de Dispositivos por Infravermelhos para Sistema de Domótica

Novembro de 2008

Universidade do Minho
Escola de Engenharia

João Luís Neto Guimarães Pereira

Interface Gráfico e Módulo de Controlo de Dispositivos por Infravermelhos para Sistema de Domótica

Dissertação para obtenção do grau de Mestre em:
Electrónica Industrial e Computadores

Trabalho efectuado sob a orientação de:
Professor Doutor António Fernando Macedo Ribeiro
Engenheiro Nino Sancho Sampaio Martins Pereira

Novembro de 2008

“O caminho da sabedoria é não ter medo de errar.”

Paulo Coelho

A Deus.
Aos meus pais: José e Ana Maria.
Aos meus irmãos: Paulo e Alberto Manuel.

Agradecimentos

Ao longo de toda esta jornada foram inúmeras as pessoas que de alguma forma contribuíram para o desenrolar deste projecto, mil desculpas para todos aqueles que eu me possa esquecer de mencionar.

Agradeço a Deus, por me mostrar a alegria e o entusiasmo nos momentos onde eu apenas senti desânimo e desilusão.

Aos meus orientadores, Professor Doutor Fernando Ribeiro e Engenheiro Nino Pereira, por todo o apoio e pelo facto de me mostrarem a realidade fora da universidade através do contacto com a empresa SAR (Soluções de Automação e Robótica).

Aos elementos da empresa SAR, por todo o apoio, paciência e material disponibilizado.

Aos técnicos das oficinas da Universidade do Minho, pela sua prontidão, flexibilidade e algumas dicas.

Aos meus amigos que acompanharam mais de perto este projecto, André Oliveira, Bruno Matos, Cristiano Santos, Luís Pacheco, Ricardo Carvalho, Sílvia Reis, pelo companheirismo, sugestões, brincadeiras e amizade.

Orgulhosamente, aos meus pais e irmãos, sem os quais toda esta caminhada não seria possível, por toda a confiança depositada bem como todo o apoio nos momentos mais difíceis.

Resumo

A domótica ou automação residencial é uma área com elevado potencial, quer a nível económico quer a nível de evolução tecnológica, contudo os preços elevados actualmente praticados dificultam a sua implantação no mercado.

A criação de um sistema com hardware baseado em autómatos programáveis permite uma redução de custos e ainda o aumento da integração do sistema. Hardware deste tipo associado a uma aplicação Flash permite a criação de sistemas altamente fiáveis e com interfaces gráficos amigáveis para o utilizador. Este projecto surge portanto num contexto de diminuição de custos e aumento de integração e robustez de um sistema de domótica, tendo em conta um interface utilizador-máquina simples e amigável. Seguindo este conceito, e com o intuito de aumentar o leque de dispositivos controláveis numa habitação, alargando o controlo a dispositivos tais como televisões por exemplo, surge também a necessidade do controlo de dispositivos por infravermelhos. A tese encontra-se assim dividida em duas fases, a implementação de software para interface com um sistema de domótica baseado em autómatos programáveis, e a implementação de um subsistema para o controlo de dispositivos por infravermelhos, com integração no sistema principal.

Numa primeira fase é apresentado o software desenvolvido para a configuração, monitorização e interacção com um sistema de domótica, são descritos todos os níveis de programação envolvidos, desde aplicação gráfica (Flash), interface com o hardware (PHP), base de dados e programação de autómatos (*Ladder*). Numa segunda fase é apresentado o subsistema desenvolvido, o qual tem a capacidade de ler, gravar e emitir comandos IR provenientes de diferentes protocolos, nesta fase é descrito todo o software e hardware envolvidos para a concretização desses processos.

Abstract

Domotics or Home Automation is an area with huge potential at economic or technological levels, although the prices actually in practice are too expensive and difficult its implantation on the market.

The creation of a system based on industrial programmable devices allows a cost reduction and also increases system integration. This type of hardware in association with a Flash application, leads to highly reliable systems and also very user friendly interfaces. This project appears in a context of price reduction, integration and performance increasing of a domotics system, keeping always in mind a simple and user friendly interface. Following this concept and in order to increase the number of different devices to control, expanding the control to devices like televisions for instance, appears the need of infrared communication. Thus this thesis is divided on two stages, the interfacing software for a domotics system based in programmable devices and the subsystem to interact with infrared controlled devices, with integration in the main system

In the first stage it is presented the developed software for configuration, monitoring and interaction with the domotics system, all the programming levels are described, such as graphic application (Flash), hardware interfacing (PHP), database and finally industrial device programming (Ladder). In the second stage it is presented the developed subsystem, which has the capability of reading, saving and sending infrared commands based on different protocols, at this stage is described all the software and hardware involved to accomplish these processes.

Índice

Índice de Figuras

Índice de Tabelas

Lista de Acrónimos

SAR	Soluções de Automação e Robótica
PLC	Programmable Logic Controller
PHP	"Personal Home Page" Hypertext Preprogramming
TV	Televisão
DVD	Digital Video Disc
IR	Infrared
PC	Personal Computer
RF	Radiofrequency
µC	Microcontrolador
PIC	Programmable Intelligent Computer
USB	Universal Serial Bus
MFC	Microsoft Foundation Class
RS-232	Recommended Standard 232
RS-485	Recommended Standard 485
EIB	European Installation Bus
EIBA	European Installation Bus Association
BCI	BatiBUS Club International
EHSA	European Home Systems Association
ETS	EIB Tool Software
EIB.TP	EIB Twisted Pair
EIB.PL	EIB Power Line
EIB.RF	EIB Radiofrequency
EIB.net2	EIB Ethernet
OSI	Open Systems Interconnection
CAN	Controller Area Network
GSM	Global System for Mobile Communications
GPRS	General Packet Radio Service
PDA	Personal digital assistants
ALICE	Automation Light Interface Control Environment
WAP	Wireless Access Protocol

PCB	Printed Circuit Board
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
DOS	Disk Operating System
ASF	Apache Software Foundation
API	Application Programming Interface
XAMPP	Cross Platform/Apache/MySQL/PHP/Perl
SGBD	Sistema Gestor de Base de Dados
SQL	Structured Query Language
NASA	National Aeronautics and Space Administration
HMI	Human-Machine Interface
I/O	Input/Output
TCP	Transmission Control Protocol
IP	Internet Protocol
SMS	Short Message Service
CDMA	Code Division Multiple Access
LSB	Low Significant Byte
MSB	Most Significant Byte
SCADA	Supervisory Control And Data Acquisition
CD	Compact Disc
LED	Light Emitting Diode
USART	Universal Synchronous Asynchronous Receiver Transmitter
UART	Universal Asynchronous Receiver Transmitter
SSOP	Shrink Small-Outline Package
TTL	Transistor-Transistor Logic
VID	Vendor ID
PID	Product ID
BPF	Band-pass Filter
ICSP	In-Circuit Serial Programming
MCU	Microcontroller
PDIP	Plastic Dual InLine Package
RAM	Random-acess Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
PWM	Pulse Width Modulation

CCP	Capture/Compare/PWM
I^2C	Inter-Integrated Circuit
ADC	Analogic-Digital Converter
IEEE	Institute of Electrical and Electronics Engineers
ISM	Industrial, Scientific and Medical
AES	Advanced Encryption Standard
DC	Direct Current

Capítulo I

1. Introdução

1.1 Definição de Domótica

O tema desta dissertação sendo a domótica, será pertinente começar justamente pela definição de domótica.

Começando pela origem da palavra “A Domótica é uma palavra que deriva do francês "Domotique" que podemos identificar com casa ("Domus") automática ("Imotique")” [1]. No fundo, domótica define-se como automação residencial.

A domótica é uma tecnologia relativamente recente no continente Europeu, implantou-se neste continente na 2ª metade da década de 80, mais propriamente em França [2]. Inicialmente pretendia-se controlar sistemas de iluminação, climatização e de segurança, sempre num contexto mais industrial ou até mesmo militar. Nos nossos dias, a ideia base é mais ou menos a mesma, contudo aplica-se a cada vez mais dispositivos e já num contexto mais doméstico [3].

As mudanças no quotidiano permitem que a domótica possa ser útil num vasto leque de novas áreas, tornando-se uma mais valia para cada vez mais pessoas. Como exemplo, poderá permitir despreocupação para pessoas que necessitam de viajar constantemente ou então mais autonomia para pessoas com necessidades especiais [3].

Esta tecnologia, pelo conforto e comodidade que pode proporcionar, apresenta um enorme potencial, quer a nível económico quer a nível de evolução tecnológica, porém enfrenta ainda alguns problemas de implantação no mercado devido aos preços elevados actualmente praticados. Neste sentido é importante torná-la acessível a cada vez mais pessoas o que implica uma redução nos custos [1] [2]. Na Fig. 1 pode ver-se o exemplo de uma habitação dotada de um sistema de domótica.

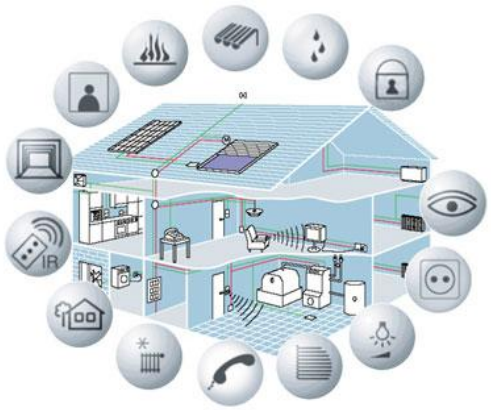


Fig. 1 Exemplo de sistema de domótica (controlo de vários dispositivos existentes na casa) [4]

1.2 Descrição do Trabalho

Este trabalho surge num contexto de diminuição de custos e aumento da robustez e integração de um sistema de domótica.

A utilização de hardware industrial como um simples PLC (*Programmable Logic Controller*) permite uma redução de custos. Com base em hardware deste tipo e com um interface gráfico baseado em Flash (quase um standard para internet) associado a um servidor *Web* (*Web Server*), é possível criar um ambiente de interface entre utilizador-sistema, simples, robusto e amigável para o utilizador. O objectivo é então o desenvolvimento de software que sirva de interface para um sistema de domótica constituído por PLCs da Unitronics. O software deverá ter as seguintes características:

- Programação em Flash/Actionscript para o interface gráfico
- Programação em PHP ("*Personal Home Page*" *Hypertext Preprogramming*) para fazer a "ponte" entre o software de controlo e o hardware
- Programação modular de forma a ser adaptada a vários tipos de hardware
- Capacidade de controlo de alguns dos dispositivos existentes numa habitação, tais como, lâmpadas, tomadas, estores, etc
- Monitorização do estado de cada dispositivo na habitação

Para que se possa ficar com uma melhor ideia do sistema de domótica em questão, na Fig. 2 encontra-se ilustrada a sua arquitectura.

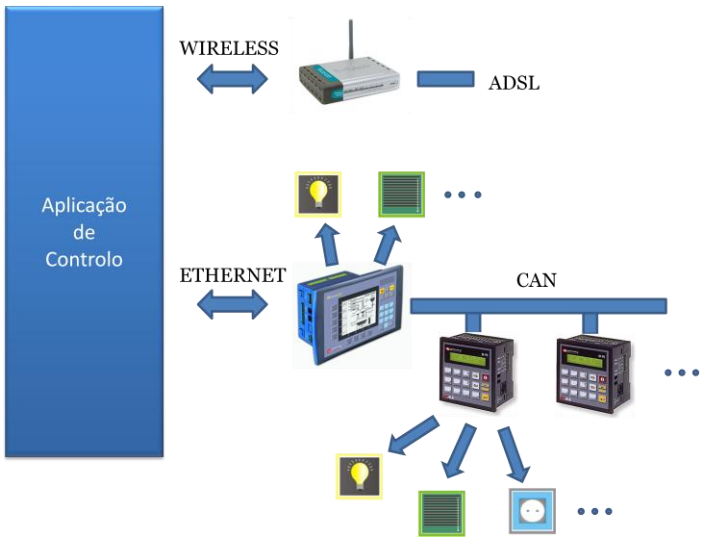


Fig. 2 Arquitectura do sistema de domótica

Com o intuito de aumentar o leque de dispositivos eléctricos controláveis numa habitação, surge a ideia de controlar dispositivos infravermelhos (Televisão (TV), DVD (*Digital Video Disc*), etc.) do inglês IR (*Infrared*). Neste sentido, uma segunda parte deste projecto, tem como objectivo o desenvolvimento de dois módulos, módulo de aquisição IR e módulo de emissão IR, para o controlo de dispositivos infravermelhos através de um computador (do inglês *Personal Computer* – PC), para integração no sistema de domótica principal.

O primeiro módulo, denominado módulo de aquisição IR, deverá ter as seguintes características:

- Leitura de comandos IR
- Comunicação com o PC para posterior gravação dos comandos IR
- Envio do comando IR em *broadcast* por radiofrequência (do inglês *radiofrequency* – RF) para uma rede de emissores IR

O segundo módulo, denominado módulo de emissão IR, deverá ter as seguintes características:

- Recepção do comando via RF e emissão desse comando por IR
- Capacidade de discernimento de informação, dado que a informação é transmitida em *broadcast*, este módulo deverá ser capaz de descartar toda aquela informação que não lhe diz respeito

Na figura seguinte é possível ver um esquema ilustrativo do subsistema de controlo de dispositivos por IR pretendido.

Fig. 3 Esquema ilustrativo do subsistema de controlo de dispositivos por IR

1.3 Solução Proposta

1.3.1 Software para Controlo do Sistema de Domótica

Começando pelo software de interface para o sistema de domótica, a solução proposta assentará numa arquitectura modular, como tal o software irá estar estruturado da seguinte forma: Módulo Cliente, Módulo Monitor e Módulo Auxiliar, desta forma o software de controlo poderá ser adaptado a qualquer tipo de hardware.

O "Módulo Cliente" será uma aplicação baseada em Flash, a qual será responsável por toda a interacção com o utilizador, desde todo o processo de configuração até à actuação no sistema. Através deste módulo será possível emitir ordens de escrita e leitura numa base de dados baseada em MySQL, bem como todas as ordens de actuação no hardware.

O "Módulo Monitor" será também uma aplicação baseada em Flash e será responsável pela leitura do estado dos dispositivos conectados aos PLCs, invocando para esse fim um *script* PHP para leitura, de X em X tempo. Dessa forma será possível obter informação actualizada acerca do estado de cada dispositivo no sistema.

O "Módulo Auxiliar" será responsável pela concretização de todos os pedidos efectuados tanto pelo "Módulo Cliente" como pelo "Módulo Monitor", desde manipulação da base de dados até à interacção com o hardware, contendo por isso todos os *scripts* PHP necessários aos vários acessos. Na Fig. 4 é possível ver toda esta estrutura do software aqui apresentada.

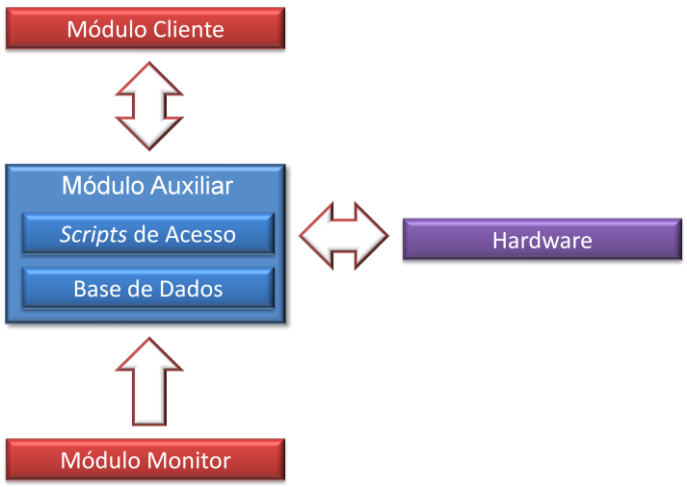


Fig. 4 Estrutura do software de controlo do sistema

1.3.2 Subsistema de Controlo de Dispositivos por IR

1.3.2.1 Hardware

Para a implementação da segunda parte do projecto será necessário desenvolver hardware que efectue um determinado tratamento do sinal IR (módulo de aquisição) e que reproduza esse sinal IR nas melhores condições (módulo de emissão). Os módulos comunicarão entre si através do protocolo *ZigBee*.

No módulo de aquisição (Fig. 5) é possível notar duas fases distintas de aquisição da informação IR. Numa primeira fase será adquirida a trama IR e posteriormente, numa segunda fase será adquirida a frequência da portadora em cuja trama se encontra modulada. Desta forma é possível garantir a leitura de tramas IR de diferentes protocolos. Toda esta informação será interpretada por um microcontrolador (μC) PIC (*Programmable Intelligent Computer*) e posteriormente enviada para o PC, através da porta USB (*Universal Serial Bus*), onde se encontrará em execução a aplicação de controlo. A informação será armazenada na base de dados e posteriormente poderá ser enviada ao módulo de emissão respectivo, via RF.

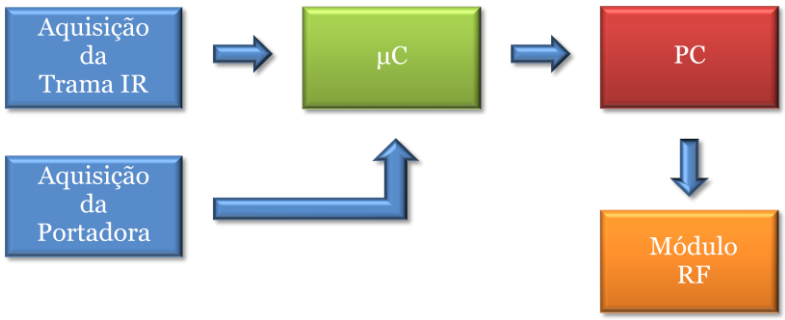


Fig. 5 Arquitectura do módulo de aquisição

O módulo de emissão será um pouco mais simples que o anterior, como é possível ver pelo diagrama da Fig. 6. A informação IR será recebida através do módulo RF e posteriormente interpretada pelo microcontrolador. O módulo terá um endereço associado, dessa forma terá a capacidade de identificar se a informação recebida por RF lhe diz ou não respeito. Por fim, depois de todos os parâmetros terem sido interpretados, a informação será modulada e enviada para o dispositivo IR em questão.

Fig. 6 Arquitectura do Módulo de Emissão

1.3.2.2 **Software**

A aplicação de controlo do módulo de aquisição estará integrada no "Módulo Cliente" do software de domótica principal. Contudo, que seja possível estabelecer uma comunicação com o microcontrolador bem como o envio de informação via RF, a aplicação principal irá invocar, por intermédio de um *script* PHP, uma aplicação baseada em C/C++ com suporte MFC (*Microsoft Foundation Class*). Assim estará estabelecida uma "ponte" entre o software de domótica e o hardware do subsistema de controlo. Na Fig. 7 é possível ver um diagrama ilustrativo da interacção do software com o hardware do módulo de aquisição.

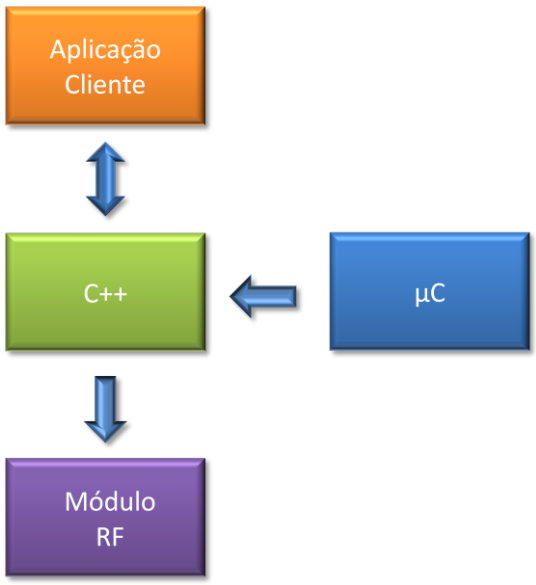


Fig. 7 Interacção do software de controlo com o hardware de aquisição

Capítulo II

2. Estado da Arte

2.1 Visão Geral de Sistemas Existentes no Mercado

Actualmente, existe no mercado uma grande variedade de sistemas de domótica. Existem sistemas cujo controlo é efectuado através de uma simples consola dedicada, localizada num ponto específico da habitação (Fig. 8 (a)), sistemas mais complexos, que permitem o controlo através de um simples televisor como se de uma mudança de canal se tratasse, dotados de um módulo de interligação com o televisor (Fig. 8 (b)), ou até mesmo sistemas baseados em *Web Servers* e portanto associados a um comum computador (Fig. 8 (c)).

Na Fig. 8 é possível ver alguns exemplos de sistemas de domótica já existentes no mercado, contudo no decorrer desta secção será dada mais ênfase a sistemas que melhor se enquadram na linha deste projecto, e portanto que surgem associados a um *Web Server*. Os sistemas serão apresentados numa perspectiva geral pois nem sempre estão disponíveis todos os detalhes técnicos uma vez que na sua grande maioria são sistemas já em comercialização. Será feita também uma abordagem a algumas das tecnologias usadas em sistemas de domótica, no sentido de contextualizar um pouco melhor a apresentação de cada um dos sistemas.

(a) (b) (c)

Fig. 8 (a) Sistema "Domus" da empresa portuguesa JG Domótica [5], (b) "Windows Media Center" associado ao sistema "Home Control" da empresa norte americana HAI (Home Automation) [6], (c) Sistema "iDom" da empresa portuguesa Domática [7]

2.2 Abordagem de Algumas Tecnologias Usadas em Sistemas de Domótica

2.2.1 X10

O protocolo X10 foi desenvolvido em 1975 pela Pico Electronics (empresa de engenharia Escocesa) com o objectivo de controlar dispositivos domésticos através de controlo remoto, para além do usual controlo através de interruptores [8]. Apesar de ter sido o primeiro protocolo a ser implementado em ambientes domésticos continua a ser bastante divulgado actualmente, tendo contudo um maior impacto no mercado norte-americano [9]. Fig. 9 é possível ver um exemplo de funcionamento de um sistema X10.

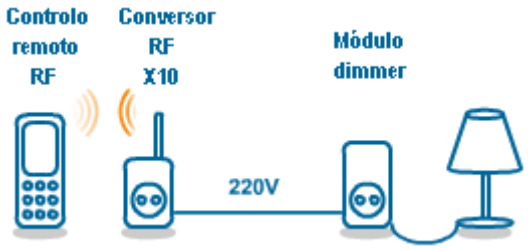


Fig. 9 Exemplo ilustrativo do funcionamento de um sistema X10 [9]

Este protocolo usa a rede eléctrica como meio de transmissão e permite enviar, para vários dispositivos receptores espalhados pela casa, comandos tais como: "On/Off", "Dim", "Bright", etc a uma taxa de 50 bps (*bits* por segundo) ou 60 bps, consoante a frequência da rede eléctrica [10]. Esta forma de comunicação aliada à simples montagem e configuração dos dispositivos X10 são os aspectos chave desta tecnologia e sem dúvida a sua grande vantagem em relação às demais tecnologias de domótica, porém a velocidade de transmissão é baixa em relação a outras tecnologias [9].

Os comandos são enviados para a rede eléctrica em *broadcast*, daí terá de existir uma forma de endereçar cada mensagem. A solução implementada por esta tecnologia face a esse problema é baseada na atribuição de um de 16 códigos de casa (A a P) e um de 16 códigos de aparelho (1 a 16) a cada dispositivo, desta forma é possível endereçar 256 dispositivos. Esta gama de endereços relativamente pequena torna esta tecnologia inadequada para grandes edifícios, onde possa ser necessário o controlo individual de mais de 256 dispositivos, o que é uma desvantagem em relação a outras tecnologias existentes. A atribuição de cada um dos códigos é feita manualmente em cada um dos

dispositivos, cabe portanto ao instalador garantir a não repetição do código em diferentes dispositivos [9]. Na Fig. 10 é possível ver o aspecto de um receptor X10 bem como a configuração do seu endereço.

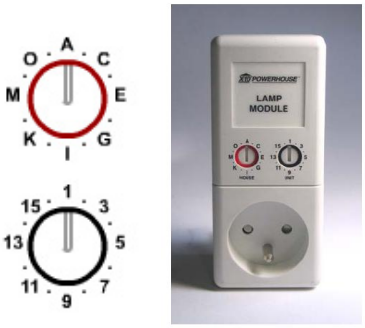


Fig. 10 Módulo *dimmer* X10 para lâmpadas incandescentes (configuração do endereço) [9] [11]

Esta tecnologia foi evoluindo e actualmente para além da comunicação por controlo remoto é também possível um controlo por computador, através de um controlador específico dotado de um interface RS-232 (*Recommended Standard 232*) [9].

A tecnologia aqui apresentada apresenta um grande sucesso dada a sua facilidade de instalação e adaptação em habitações mais antigas visto tirar partido da comunicação pela rede eléctrica, não sendo portanto necessária nova cablagem para o sistema de domótica. Contudo, dadas as suas limitações quer a nível de taxa de transmissão quer a nível de endereçamento não é adequada para grandes edifícios.

2.2.2 **EIB**

O EIB (*European Installation Bus*) é um protocolo criado pela EIBA (*European Installation Bus Association*) no final dos anos 80 com o intuito de dar resposta às novas necessidades de controlo em casas e edifícios [12]. Em 1999 a EIBA fundiu-se com as associações BCI (*BatiBUS Club International)* e EHSA (*European Home Systems Associatio*n) e deu origem à *KNX Association*, daí que frequentemente o EIB surja como EIB/KNX [13].

O aumento do uso de sistemas de vigilância e as novas exigências a nível de segurança e conforto conduziram a uma maior complexidade, morosidade e aumento de custos na instalação dos devidos equipamentos. Estes entraves, em especial a complexidade da instalação (aumento excessivo de cablagem) foram o mote para a

procura de um tipo de instalação mais simples e de maior confiança. A tecnologia EIB solucionou este problema introduzindo uma estrutura baseada num *bus* de duas linhas, para comando e alimentação, sendo assim capaz de transmitir toda a informação sem qualquer restrição e também permitindo a redução do número de cabos e consequentemente a redução dos riscos de incêndio. A topologia é baseada numa distribuição ponto a ponto, onde cada dispositivo pode comunicar directamente com os restantes, num total de 65536 dispositivos, ou seja, um espaço de endereçamento de 16 bits. Num sistema EIB os componentes são designados como sensores e actuadores. Estes componentes encontram-se ligados à linha de *bus* e é possível que qualquer sensor troque informações com qualquer actuador. A informação obtida pelos sensores será armazenada no chamado "telegrama" e posteriormente enviada para o(s) respectivo(s) actuador(es). Para que isto seja possível terá de ser feita uma programação prévia (através do ETS (*EIB Tool Software*)) de maneira a configurar o endereço de cada sensor e actuador, bem como a correspondência de cada um, desta forma será possível a criação de funções individuais ou funções de grupo [14]. Na Fig. 11 é possível ver a arquitectura de um sistema baseado no protocolo EIB, o Instabus EIB.

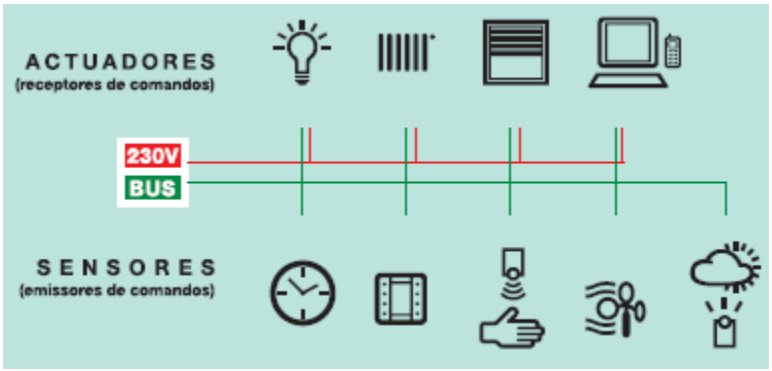


Fig. 11 Arquitectura do sistema Instabus EIB [15]

O meio físico de transmissão poderá ser de diferentes tipos, desde par entrançado (EIB.TP - *Twisted Pair*), rede eléctrica (EIB.PL – *Power Line*), radiofrequência (EIB.RF – *Radio Frequency*), *Ethernet* (EIB.net2), entre outros. A velocidade de transmissão difere consoante o meio físico de transmissão, contudo o meio vulgarmente utilizado por esta tecnologia é o par entrançado, o qual permite velocidades de transmissão de até 9600 bps (cabos até 1000 m (incluindo ramificações)) [14].

A fiabilidade na comunicação é assegurada segundo as camadas do modelo OSI (*Open Systems Interconnection*), o que torna esta tecnologia bastante confiável [14].

A tecnologia EIB sendo relativamente recente assume já um papel de elevada importância no mundo da domótica dada a sua elevada fiabilidade, segurança e flexibilidade. A vasta capacidade de endereçamento contribui para um maior leque de aplicações o que torna esta tecnologia apropriada para qualquer tipo de edifícios, desde pequenas habitações até hotéis ou escolas.

As desvantagens desta tecnologia prendem-se com o custo elevado dos seus dispositivos bem como pelo facto de necessitar de mão-de-obra especializada na sua instalação.

2.2.3 **CAN**

O protocolo CAN (*Controller Area Network*) foi desenvolvido pela BOSH em 1983. Esta tecnologia surgiu para aplicação em automóveis, visto não existir no mercado uma tecnologia que fosse totalmente satisfatória para uma aplicação no ramo automóvel, nomeadamente que possibilitasse uma redução na cablagem [16].

A tecnologia foi evoluindo e actualmente existem controladores CAN disponíveis a um custo relativamente baixo e dada a sua elevada fiabilidade é possível encontrar a tecnologia CAN aplicada também em áreas como a medicina, automação industrial, etc [17]. Em termos de automação residencial, a sua aplicação é relativamente recente, contudo já se têm dado passos importantes numa implementação mais consistente desta tecnologia em habitações [19] .

O protocolo CAN é um protocolo de comunicação série para aplicações em tempo real. A informação é emitida em *broadcast* e a transmissão é orientada à mensagem, isto é, a cada uma das estações não se encontra associado um endereço como acontece em redes *Ethernet* por exemplo, numa rede CAN a própria mensagem a enviar contém um identificador, o qual identifica o conteúdo e a prioridade da mensagem (aspecto chave para a comunicação em tempo real), este identificador é único em toda a rede, cabe então ao receptor o discernimento da informação. Através deste método é possível obter um sistema modular dotado de uma comunicação bastante fiável, o que torna esta tecnologia ideal para sistemas em tempo real [18]. Na Fig. 12 é possível ver um exemplo de funcionamento do sistema CAN. Em relação ao número máximo de dispositivos conectados em rede, teoricamente esse valor é de 2032 dispositivos, contudo devido às limitações dos *transceivers* utilizados pelo controlador CAN o valor é bastante mais baixo. Para um *transceiver* Philips (82C250) o valor máximo é de 110 dispositivos [20].

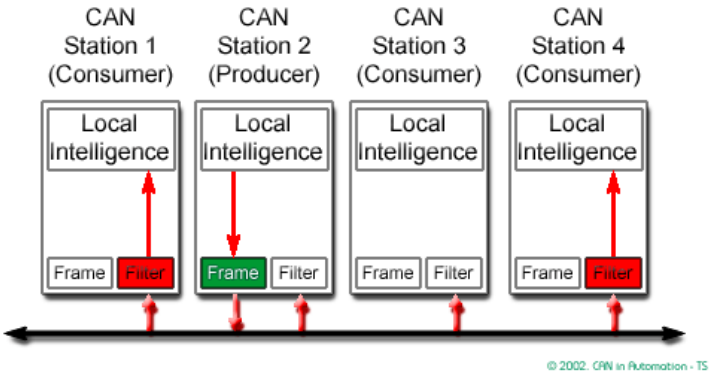


Fig. 12 Exemplo de funcionamento de um sistema CAN [18]

O meio de transmissão mais comum é baseado numa ligação a 2 fios (operação diferencial) em par entrançado sendo possível obter velocidades de transmissão de até 1 Mbps – megabits por segundo (cabos até 40 metros (m)) [21].

Em suma, o protocolo CAN embora tenha surgido para aplicação no ramo automóvel começa já a aparecer num contexto de automação residencial e dada a sua flexibilidade e fiabilidade apresenta fortes probabilidades de singrar neste mercado. A desvantagem do CAN reflecte-se um pouco a nível da instalação, pois será necessário pessoal especializado, um pouco à semelhança do EIB anteriormente apresentado. A velocidade de transmissão é também uma desvantagem uma vez que existem protocolos, nomeadamente o *Ethernet*, que permitem maiores débitos.

2.2.4 **RS-485**

O RS-485 (*Recommended Standard 485*) é um padrão de comunicação criado pela EIA (*Electronics Industry Association*) em 1983 [22]. Este padrão não especifica nem recomenda nenhum protocolo de comunicação, apenas especifica características eléctricas e modos de operação na rede, ao contrário das tecnologias anteriormente aqui apresentadas. Esta característica é ideal para empresas que desejem criar o seu próprio protocolo, garantindo assim exclusividade num determinado produto [23].

O meio de transmissão é baseado numa ligação a 2 fios em par entrançado (operação diferencial) o que permite uma comunicação bastante mais imune a interferências, à semelhança do que é utilizado no protocolo CAN por exemplo. A nível de velocidades de transmissão, tem-se como referência que para distâncias até 12 m é possível um débito máximo de 35 Mbps e para o máximo da distância permitida pelo padrão (1200 m) é possível um débito na ordem dos 100 kbps [24].

Com base no padrão RS-485 é possível conectar até 32 dispositivos numa rede multiponto com comunicação *half-duplex*, contudo é possível expandir este valor até 256 se forem utilizados módulos RS-485 com entradas de elevada impedância [24]. Apesar da possibilidade de expansão, este valor acaba por ser baixo se o objectivo for a aplicação num edifico de grandes dimensões, um hotel por exemplo. Na Fig. 13 é possível ver a topologia de uma rede RS-485.

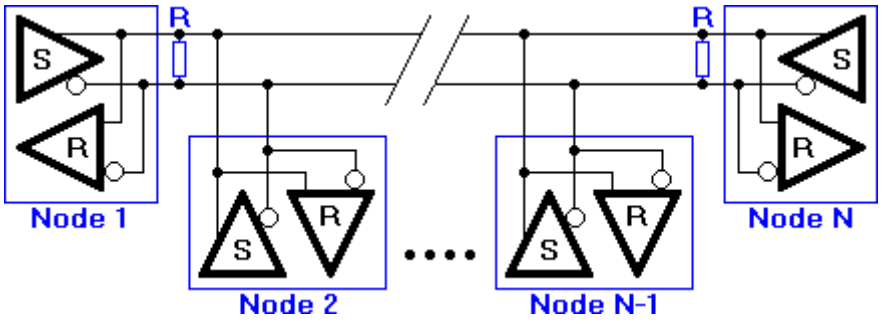


Fig. 13 Topologia de uma rede RS-485 [24]

A nível de funcionamento podem ser previstos dois casos. O primeiro é o modo *master/slave* em que existe um *master* a enviar informação para o(s) *slave(s)* através do *bus*. Neste modo não existem problemas de colisão. O segundo modo é o *multi-master*, neste modo vários *nodes* podem iniciar uma comunicação, neste caso poderão existir problemas de colisão e portanto será necessária uma detecção de erros por parte do protocolo a usar [24]. Em ambos os modos a comunicação é feita por endereçamento [23].

O RS-485 tem tido alguma aceitação no mundo da domótica. A implementação de uma rede de baixo custo comparativamente a uma rede *Ethernet* por exemplo, a possibilidade da definição de um protocolo próprio, garantindo mais flexibilidade e exclusividade e uma relação velocidade x distância de transmissão bastante boa, são razões que conduzem a esse sucesso. Contudo, o leque de aplicações está um pouco limitado a habitações mais pequenas, dada a sua baixa capacidade de endereçamento [1] [23].

2.3 Sistemas e Software de Domótica

2.3.1 iDom

O sistema "iDom" foi desenvolvido pela empresa portuguesa Domática - Electrónica e Informática Lda [25].

Este sistema apresenta uma característica distribuída, no qual existe um módulo principal responsável pela interacção com o exterior bem como pela gestão e configuração de vários módulos (módulos secundários) aos quais se encontram conectados os vários dispositivos a controlar ou a monitorizar (caso dos sensores). Os módulos secundários encontram-se conectados ao módulo principal e entre si através de CAN, o que só por si é já uma garantia de elevada fiabilidade. "A eficiência do iDom, deve-se em parte ao seu fiável sistema de comunicações, assente em tecnologia geralmente utilizada em sistemas críticos, nomeadamente na área da indústria automóvel, médica e espacial" [25].

O sistema é constituído essencialmente pelos seguintes módulos: "iDom Manager", "iDom Collector", "iDom Dimmer", "iDom Relay", "iDom Tristate" e o "iDom Sensor" [25].

O "iDom Manager" não é essencial para o funcionamento do sistema, uma vez que cada um dos módulos secundários pode estar programado para executar uma determinada tarefa consoante um determinado evento, contudo, permite dotar o sistema de mais funcionalidades. Através deste módulo é possível obter comunicação com o exterior, seja através de *Ethernet*, GSM (*Global System for Mobile Communications*) / GPRS (*General Packet Radio Service*) ou até mesmo via *Web*, o que permite um controlo e gestão do sistema à simples distância de um "click", não estando portanto limitado à simples interacção através de interruptores [25].

O "iDom Collector" é o módulo de entradas, sendo responsável por toda a leitura das entradas do sistema (interruptores, sensores, etc) com o objectivo de manter o sistema actualizado de maneira a executar as funções pré-programadas [25].

O "iDom Sensor" é um módulo sensorial, integrando vários sensores, nomeadamente de movimento, temperatura, luminosidade e receptor e emissor de infravermelhos [25].

Os restantes módulos são responsáveis pela actuação, o "iDom Dimmer" permite o controlo da iluminação com possibilidade de regulação luminosa, o "iDom Relay"

permite o controlo de tomadas e dispositivos de maior potência (máx. 16A/230VAC) e por fim o "iDom Tristate" permite o controlo de dispositivos com três estados, como é o caso dos estores e portões por exemplo. Na Fig. 14 (a) é possível visualizar a arquitectura de um sistema "iDom" [25].

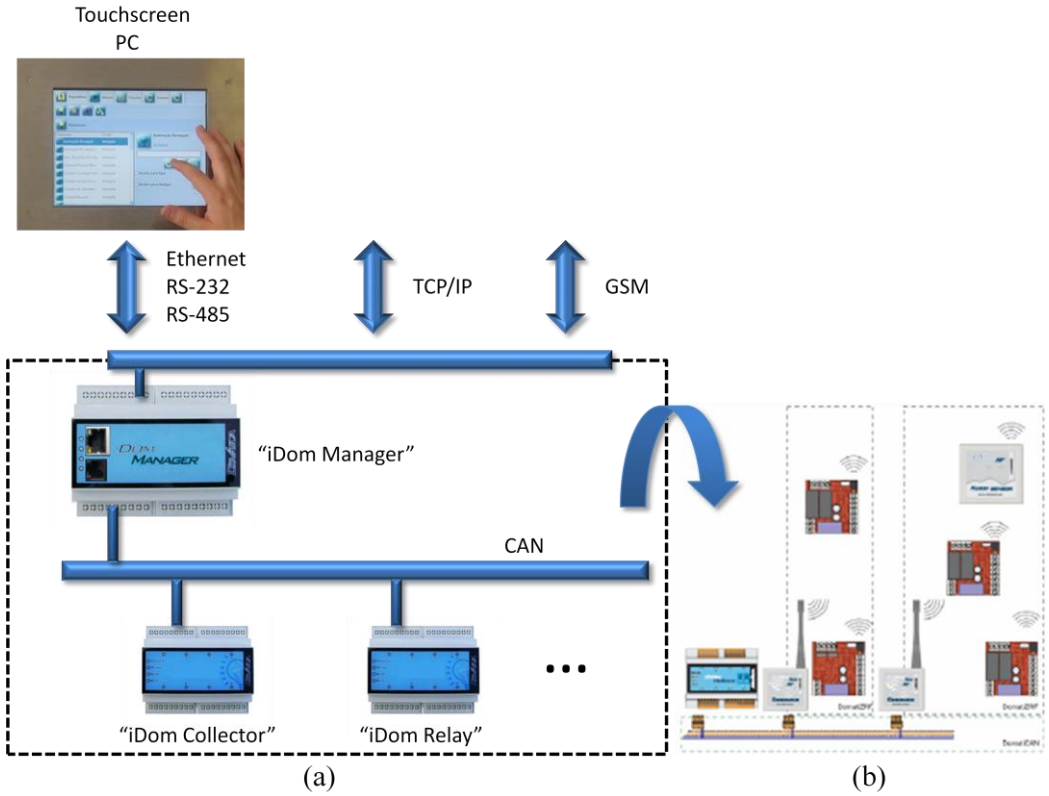


Fig. 14 Exemplo ilustrativo da arquitectura do sistema "iDom". ***(a) Sistema baseado em comunicação CAN (b) Sistema baseado em comunicação RF***

A característica modular deste sistema traz grandes vantagens a nível de instalação e configuração.

O sistema "iDom" tem ainda a capacidade de ser adaptado a habitações mais antigas, deixando portanto de existir a necessidade de uma pré-instalação de cablagem, isto é possível através da substituição de toda a rede CAN por uma rede RF, como é possível ver na Fig. 14 (b). Esta característica permite ao sistema abranger uma maior quota de mercado [25] [26].

O software de controlo é bastante simples e amigável para o utilizador. A partir de um ambiente gráfico bastante intuitivo o utilizador poderá configurar todas as funções do sistema bem como monitorizar todas as ocorrências. Um aspecto a realçar é o facto do controlo dos dispositivos ser feito através da interacção com a planta da casa, o que é interessante pois o utilizador pode ter uma visualização espacial do dispositivo a

controlar. Na Fig. 15 é possível ver duma maneira geral a aparência do interface gráfico deste sistema [25].

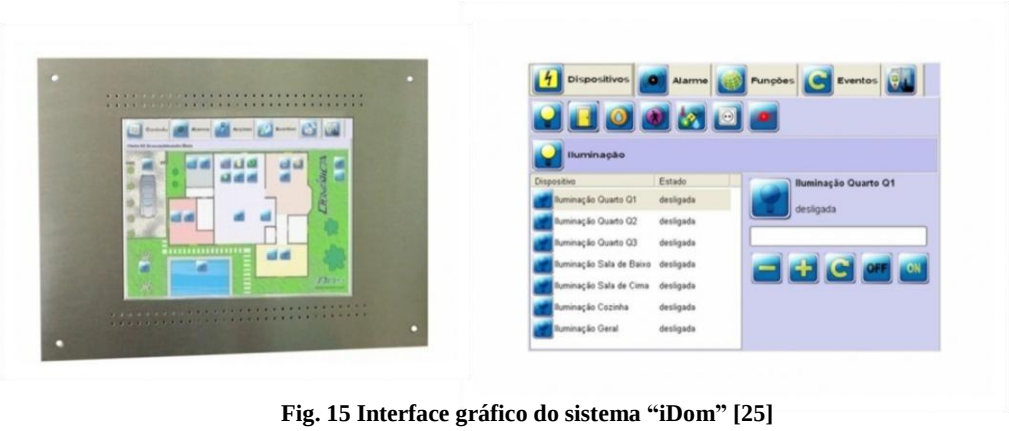


Fig. 15 Interface gráfico do sistema "iDom" [25]

O sistema "iDom" apesar de toda a sua flexibilidade apresenta algumas desvantagens. A primeira é um pouco comum a todos os sistemas de domótica, prende-se com o investimento inicial elevado. Outro aspecto está relacionado com a capacidade modular do sistema, que apesar de ser boa acaba por ser inferior à capacidade modular de um PLC, o que se pode tornar uma desvantagem na aplicação em edifícios de grandes dimensões. Por fim, a instalação e programação do sistema requer mão-de-obra especializada.

2.3.2 **Mordomus**

O sistema "Mordomus" foi também desenvolvido em Portugal e é o resultado da cooperação das empresas IVV Automação Lda e IOLine – Research & Development Lab [27].

Este sistema é centralizado, isto significa que existe uma unidade de controlo à qual se encontram conectados diversos módulos de actuação e aquisição. A rede é baseada no padrão RS-485, o que é uma desvantagem pois o sistema é dependente do fabricante, uma vez que o protocolo utilizado não é um standard [28].

O "Mordomus" ao contrário do sistema "iDom" não tem "inteligência" local, ou seja, todos os eventos são processados na unidade de controlo central e só depois será efectuada a respectiva actuação num determinado módulo actuador [28].

Os módulos principais constituintes do sistema são os seguintes: "Módulo Central", "Módulo de Comunicações", "Módulo de Entradas Digitais", "Módulo de Estores e Cortinas", "Módulo de Saídas ON/OFF", "Módulo *Dimmer*" e "Módulo de

Sonorização Ambiente". Existem ainda outros módulos, nomeadamente de vídeo vigilância, os quais podem também ser adaptados ao sistema, porém não são módulos desenvolvidos pelas empresas envolvidas no "Mordomus" [28]. Na Fig. 16 é possível ver um exemplo da arquitectura deste sistema.

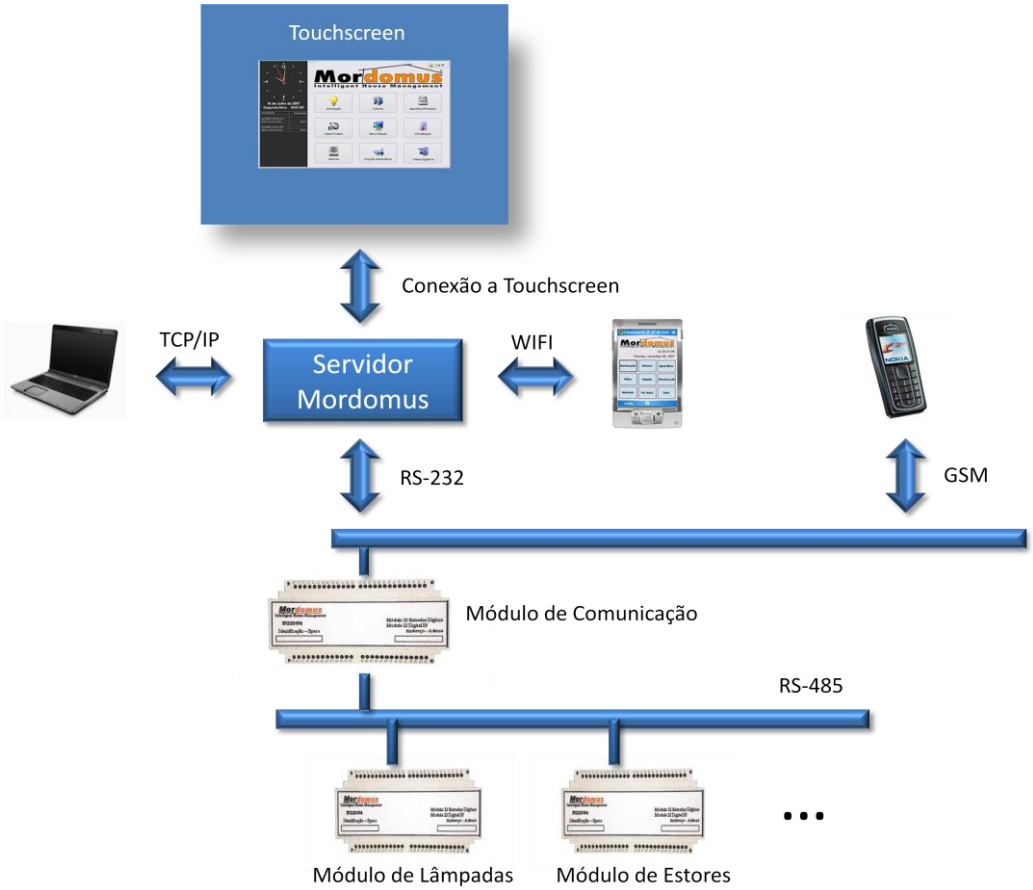


Fig. 16 Exemplo ilustrativo da arquitectura do sistema "Mordomus"

O "Módulo Central" é um simples PC dotado de um ecrã táctil, o que permite ao utilizador gerir, configurar e actuar no sistema através de um interface gráfico bastante simples e flexível. Este PC implementa um *Web Server* tornando assim também possível o controlo remoto do sistema via *Web* ou *Wifi* [28].

O "Módulo de Comunicação" é o responsável pelo interface entre os diferentes módulos do sistema e o "Módulo Central", tornando assim possível a interacção através do software "Mordomus". Com base neste módulo é também possível estabelecer uma conexão via GSM permitindo assim o controlo do sistema através de um telemóvel [28].

Os restantes módulos são responsáveis pela actuação e monitorização ("Módulo de Entradas Digitais"). É possível o controlo de estores e cortinas (3 saídas por módulo), dispositivos ON/OFF (máx. 8 saídas por módulo), lâmpadas, com

possibilidade de controlo de luminosidade (máximo 8 saídas (*dimmer*) ou 8 saídas (ON/OFF)) e por fim o controlo de sonorização ambiente, o que pode ser utilizado não só a nível de som ambiente mas também para a reprodução de sinais de aviso [28].

O software de controlo à semelhança do "iDom", visto anteriormente, apresenta também um interface bastante simples, contudo no "Mordomus" não é possível uma visualização da planta. Apesar de não ser possível uma visualização espacial dos dispositivos, o método utilizado torna-se também intuitivo pois é possível aceder ao controlo de dispositivos através de uma pré-selecção do piso e compartimento. O software permite ainda várias funcionalidades, desde a definição de funções automáticas, gestão de alarmes e toda a configuração do sistema a nível de hardware. Na Fig. 17 é possível ver umas imagens exemplo do software "Mordomus" [29].

Fig. 17 Interface gráfico do sistema "Mordomus" [29]

Em suma, o sistema "Mordomus" é um sistema proprietário, isto significa que não adopta nenhum protocolo standard para domótica, por essa razão o cliente estará sempre dependente das empresas em questão, seja para manutenção ou aquisição de novos módulos com novas funcionalidades. Outra desvantagem prende-se com o facto de não existir uma "inteligência" distribuída, ou seja, a unidade central processa todos os eventos o que significa que se existir uma avaria nesta unidade o sistema fica suspenso. O facto de serem utilizados padrões de comunicação mais antigos, nomeadamente RS-232 e RS-485 permite uma redução no custo do hardware, o que é uma vantagem uma vez que o custo dos sistemas de domótica é um dos principais entraves neste momento.

2.3.3 ***Harmony 2007***

O "Harmony 2007" é um aplicação .NET para controlo de sistemas de domótica, foi desenvolvido pela empresa inglesa Harmony Systems [30].

Este software tem a capacidade de interagir com diferentes tecnologias usadas em sistemas de domótica, nomeadamente X10, Dupline, Z-Wave, etc, ou seja, permite o uso de tecnologias não-proprietário, ao contrário do que se passa com o sistema "Mordomus", anteriormente apresentado, no qual o software é exclusivamente dedicado à tecnologia em questão, não podendo portanto ser utilizado com outro tipo de tecnologias existentes [31].

O "Harmony 2007" é baseado numa aplicação cliente-servidor, constituído por um PC central que implementa o *Web Server* ao qual se encontra associado o hardware respectivo de controlo da habitação, permitindo assim o acesso de vários clientes para controlo do sistema. Esta característica é semelhante à do sistema "Mordomus", anteriormente apresentado [31].

O software apresenta várias funcionalidades, desde o simples controlo ON/OFF, suporte para programação de funções automáticas e eventos temporizados, controlo de câmaras de vigilância etc. Com base no servidor implementado é possível interagir com o sistema de diversas formas, seja localmente ou remotamente. Localmente, através de uma aplicação baseada em *Windows Media Center* ou através de um *touchscreen*. Remotamente, via *Wifi* através de um PDA **(**Personal digital assistants**)** ou via *Web*. Na Fig. 18 é possível ver o exemplo da arquitectura de um sistema dotado com software "Harmony 2007" no controlo de tecnologia X10 (Módulos Reino Unido). Será de salientar, que no controlo dos dispositivos do sistema, será sempre necessário um módulo intermédio que faça a "ponte" entre o software e o hardware, neste caso, tendo em conta que a tecnologia usada é a X10, o módulo responsável será o CM11, como se poderá verificar na figura seguinte [31].

O facto do software "Harmony 2007" permitir o interface com diferentes tecnologias proporciona uma grande flexibilidade, dando assim uma certa liberdade de escolha ao cliente, permitindo assim a escolha de uma tecnologia que melhor se adapte às suas necessidades. Apesar de ser compatível com uma grande variedade de tecnologias, não é compatível com o EIB/KNX, é pena que um software com tanto potencial não seja compatível com um protocolo de domótica de grande prestígio e já bastante implantado no mercado [31].

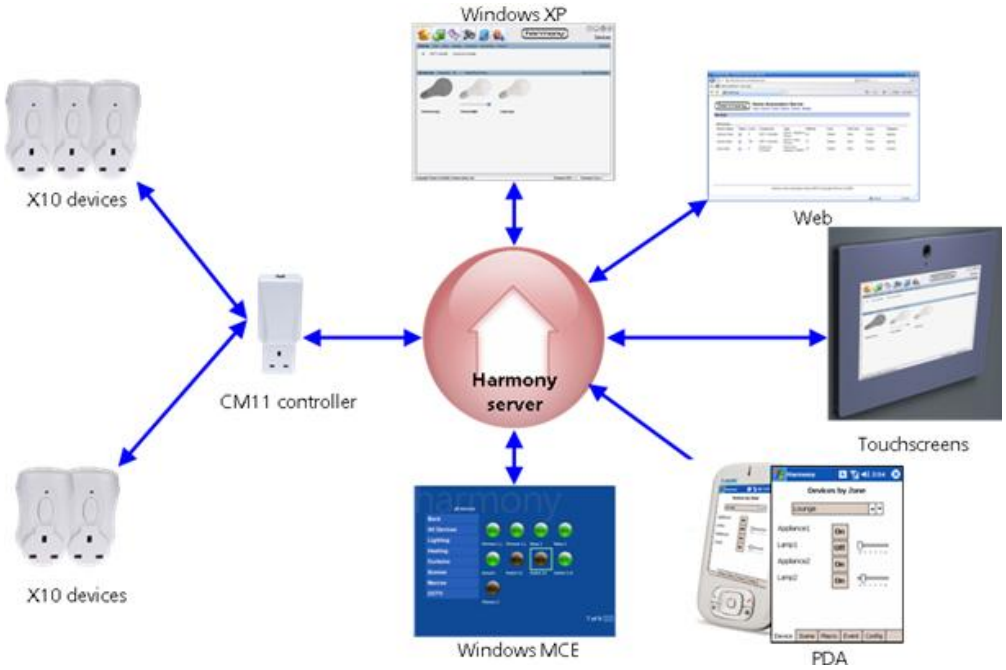


Fig. 18 Arquitectura de um sistema baseado no software "Harmony 2007" [32]

2.3.4 **ALICE**

O "ALICE" (*Automation Light Interface Control Environment*) é um software de controlo exclusivamente para sistemas dotados com tecnologia X10 e contrariamente ao que foi apresentado até agora é ainda um projecto em desenvolvimento e portanto não se encontra no mercado, contudo dadas as suas características e a possibilidade de acesso a todo o código fonte é pertinente apresentá-lo neste estado da arte. Esta aplicação apesar de já se encontrar implementada continua em constante desenvolvimento no sentido de expansão de funcionalidades e também correcção de alguns *bugs*. O desenvolvimento encontra-se a cargo de uma equipa constituída por três elementos, Walter Bogaardt, Calvin Yu e Tarah Hofmann [33].

O software "ALICE" é inteiramente baseado em Java e apresenta funcionalidades tais como: o usual controlo ON/OFF, configuração de cada um dos módulos, programação e monitorização de eventos, definição de macros e configurações de comunicação. O interface com o sistema X10 é feito, à semelhança do exemplo apresentado com o "Harmony 2007", através do módulo CM11. Este módulo detém uma memória capaz de armazenar macros, como tal essa potencialidade é explorada por este software o que permite que para determinadas acções seja dispensável o controlo de eventos através do PC. Na Fig. 19 é possível ver o aspecto gráfico deste software de controlo [33].

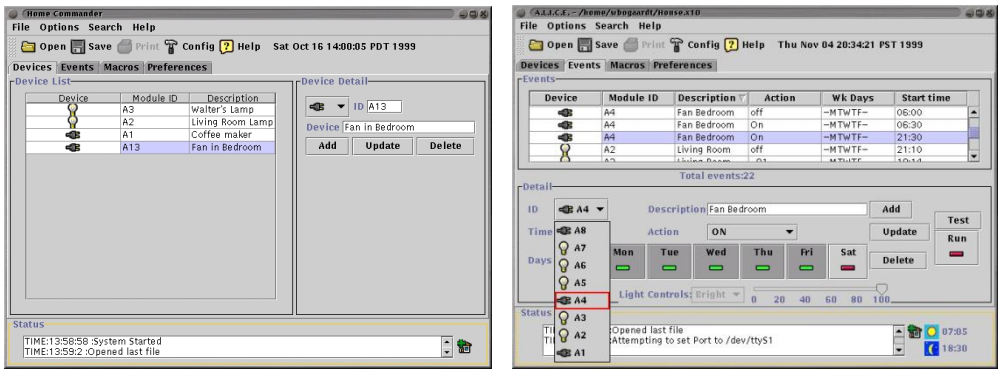


Fig. 19 Interface gráfico do "ALICE" [33]

A nível de monitorização é de realçar um aspecto interessante e bastante inovador em termos de aplicação domótica, é o facto do software permitir a reprodução vocal sintetizada de mensagens pré-definidas associadas a um determinado dispositivo, as quais serão reproduzidas sempre que este mude o seu estado. Isto torna-se possível graças ao FReeTTS [34], uma aplicação *open source* desenvolvida também em Java e que se encontra associada ao "ALICE" [33].

O "ALICE", para além do interface local, permite também interface *Web* e WAP (*Wireless Access Protocol*) o que torna possível um controlo remoto através de um PC ou telemóvel conectados à internet, disponibilizando grande parte das funcionalidades do acesso local, excepto monitorização e gravação de macros. Na Fig.20 é possível ver a arquitectura de um sistema baseado no software "ALICE" [33].

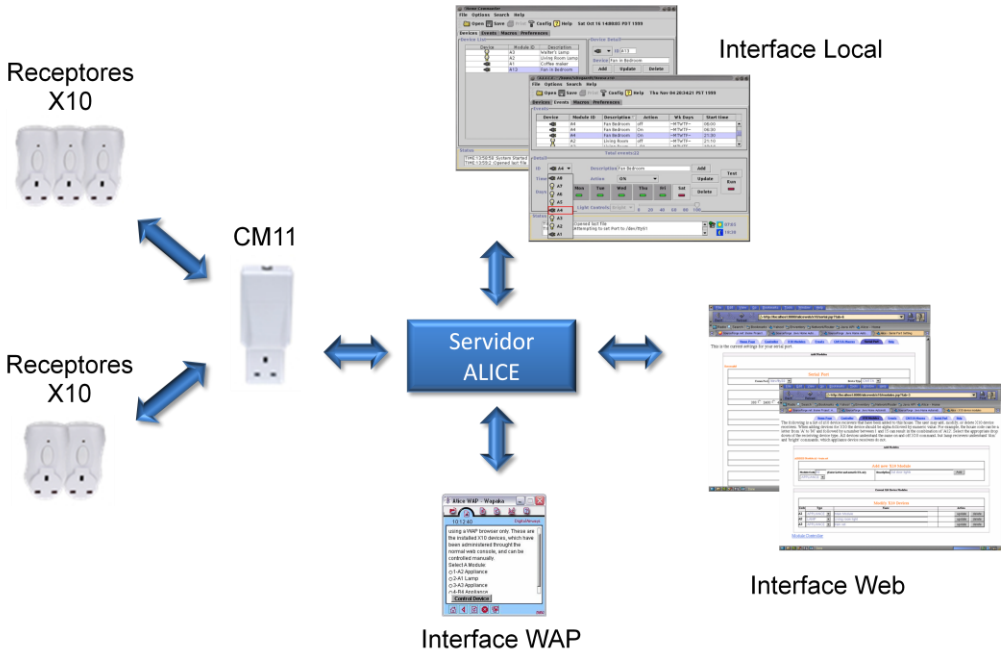


Fig. 20 Arquitectura de um sistema baseado no software "ALICE"

O software aqui apresentado, apesar de não ser comercializado e estar ainda em desenvolvimento, apresenta já um elevado potencial. O facto de ser compatível apenas com a tecnologia X10 torna-o um pouco limitado, contudo uma vez que é um software *open source* quem sabe possa surgir a possibilidade da sua compatibilidade com outras tecnologias existentes, nomeadamente o EIB/KNX ou *Ethernet.*

2.3.5 **Conclusões e Análise Comparativa**

Os sistemas apresentados, à excepção do sistema baseado no software "ALICE", encontram-se todos eles no mercado, e portanto possuem características que o sistema em apresentação nesta tese não possui, tais como, vídeo vigilância ou controlo de climatização, por exemplo. No entanto não significa que não estejam criadas as condições para se proceder a essa evolução. O objectivo deste projecto não é tanto proporcionar uma grande diversidade de dispositivos a controlar, mas sim mostrar, através do controlo de alguns dos dispositivos mais comuns presentes num habitação, a possibilidade de criação de um sistema de domótica dotado de software fiável, de simples interacção e interface amigável e hardware normalizado e de custo mais reduzido, juntos criando condições para a constante evolução do sistema e não envergando pelo caminho dos sistemas proprietário.

Na Tabela 1 é feita uma comparação entre os diferentes sistemas, incluindo o sistema em apresentação nesta tese. Em jeito de resumo, são apresentadas características tais como: se o sistema é constituído por hardware e software desenvolvidos pela própria empresa ou se apenas o software é desenvolvido e aplicado num determinado hardware, se o sistema é proprietário, se é necessária mão-de-obra especializada na sua instalação, quais as tecnologias existentes em termos de comunicação com o respectivo hardware, se existe a possibilidade de acesso ao sistema remotamente, e por fim se existe necessidade de uma pré-instalação de cabos.

	iDom	Mordomus	Harmony 2007	ALICE	Projecto
Hardware	√	√	-	-	-
Software	√	√	√	√	√
Proprietário	-	√	-	-	-
Instalação	Especializada	Especializada	Utilizador	Utilizador	Especializada
Tecnologias	CAN/RF *Ethernet*	RS-485 RS-232	X10 Dupline outras..	X10 RS-232	*Ethernet* CAN USB outras..
Acesso Remoto	√	√	√	√	√
Pré-Cablagem	√	√	[1]	-	√

Tabela 1 Comparação entre os diferentes sistemas/softwares de domótica

[1] A necessidade ou não de uma pré-cablagem estará relacionada com o tipo de hardware utilizado. Por exemplo o uso de um hardware Dupline implica uma pré-cablagem, no caso do X10 por exemplo, dadas as suas características já não será necessário.

2.4 Módulos para Controlo de Dispositivos por IR

Os módulos para controlo de dispositivos IR numa habitação são normalmente módulos opcionais, daí se ter optado pela criação de uma secção exclusivamente dedicada a estes subsistemas, e portanto não serem incluídos na secção 2.3.

Os módulos em apresentação são ligeiramente diferentes dos módulos desenvolvidos neste projecto na medida que diferem em algumas características, nomeadamente no meio de transmissão ou nas características de aquisição IR, inclusive alguns dos módulos em apresentação apenas permitem aquisição IR, contudo a sua apresentação acaba por ser pertinente dadas as suas características de aquisição.

2.4.1 Conversor X10-IR

Este módulo IR é exclusivamente dedicado a sistemas que utilizam a tecnologia X10. O conversor permite o controlo de dispositivos IR através de comandos X10 [35].

O "conversor X10-IR" tem dois modos distintos, o modo de aprendizagem e o modo de envio.

O modo aprendizagem permite ao utilizador memorizar até 32 comandos IR, contudo este processo só funciona correctamente para dispositivos com uma portadora de 38 kHz (kilohertz). Cada comando IR memorizado terá de estar sempre associado a um comando X10 pré-configurado.

O modo envio permite accionar um determinado dispositivo IR anteriormente programado, bastará portanto enviar o comando X10 associado. Este módulo torna assim possível o controlo de diferentes dispositivos IR espalhados pela habitação, tais como televisões, hi-fi, ar condicionado, etc. Na Fig. 21 é possível ver o conversor X10-IR em questão [35].

Fig. 21 Conversor X10-IR [35]

O conversor é bastante compacto e de fácil configuração, contudo encontra-se um pouco limitado a nível de aprendizagem, uma vez que existem diversos dispositivos IR com portadoras na ordem dos 36 kHz e 40 kHz. O facto do módulo ser baseado na tecnologia X10 acarreta todas a vantagens e desvantagens desta tecnologia, contudo é de salientar a liberdade de colocação do módulo, o qual não está dependente de uma pré-instalação de cablagem.

2.4.2 **Unzap**

O "Unzap" é um módulo desenvolvido por Alexander Neumann. Este módulo tem a capacidade de interpretar, armazenar e emitir comandos IR, pode portanto ser denominado por IR *transceiver* [36].

O módulo já se encontra disponível para venda, contudo todo o firmware e hardware (esquemático e *layout* PCB – *Printed Circuit Board*) é facultado no fórum de Alexander [36].

O módulo é constituído essencialmente por um microcontrolador Atmega 168, uma memória *flash* ATMEL de 2MB, um conector USB, quatro botões de selecção, um detector e um desmodulador IR e por fim emissores IR. Os comandos IR são interpretados pelo microcontrolador e posteriormente armazenados na memória *flash*. A aprendizagem engloba também a leitura da frequência da portadora, o que garante um vasto leque de aquisição de comandos, não estando portanto limitado a uma determinada portadora. Na Fig. 22 é possível ver uma imagem do "Unzap" [36].

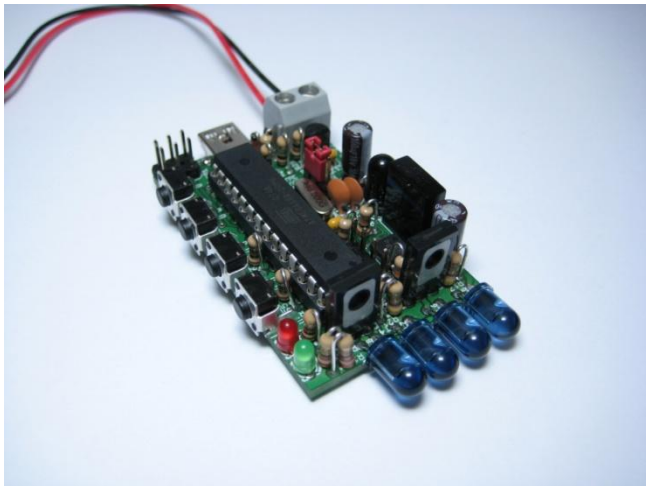


Fig. 22 Módulo "Unzap" [36]

O “Unzap” é capaz de interpretar e armazenar tramas IR provenientes de diferentes protocolos, contudo o comando a armazenar deverá ser sempre o mesmo, por exemplo “TV OFF”, isto porque o módulo envia consecutivamente todos os comandos armazenados na memória *flash*. Através dos botões de pressão é possível controlar esta emissão IR, sendo possível pará-la a qualquer momento ou até mesmo iniciar o modo *step-by-step* (envio trama-a-trama). O modo USB para *update* de firmware é também seleccionado a partir destes botões [36].

Este módulo poderá ser considerado um *gadget*, podendo ser útil para desligar vários aparelhos diferentes num determinado espaço, por exemplo, contudo poderia perfeitamente ser adaptado a um sistema de domótica, desde que fosse dotado de um módulo RF, por exemplo, que lhe permitisse receber ordens de comando provenientes de um PC [36].

2.4.3 **IR Widget**

O “IR Widget” é apenas um módulo de aquisição de comandos IR. Este módulo comunica directamente com o PC, no qual se encontrará em execução uma aplicação capaz de interpretar os dados IR e reproduzi-los de uma forma gráfica (“IR Scope”), permitindo assim o reconhecimento, análise ou gravação dos comandos. Este módulo não é comercializado [37].

O módulo é muito simples, sendo constituído essencialmente por um microcontrolador PIC 12F629, um conector RS-232, um detector e um desmodulador IR.

À semelhança do “Unzap”, este módulo permite também a leitura da portadora. Na Fig. 23 encontra-se ilustrado o “IR Widget” dotado de comunicação RS-232, contudo já existe uma versão mais recente dotada de comunicação USB, porém não foi ainda apresentada na forma de PCB [37].

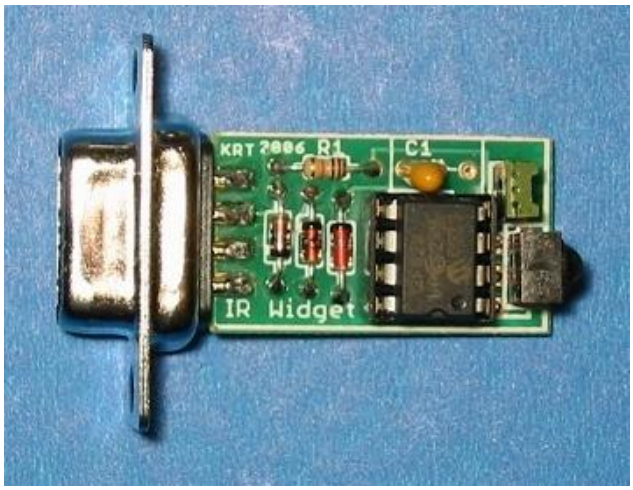


Fig. 23 Módulo "IR Widget" [37]

O software de interpretação dos dados associado a este módulo, o "IR Scope", permite duma forma bastante simples a visualização da trama IR adquirida, fornecendo toda a informação acerca da frequência da portadora e tempos ON e OFF de cada *bit*. Os dados poderão inclusive ser armazenados num ficheiro de texto. Na Fig. 24 é possível visualizar um exemplo da informação obtida numa aquisição IR [37].

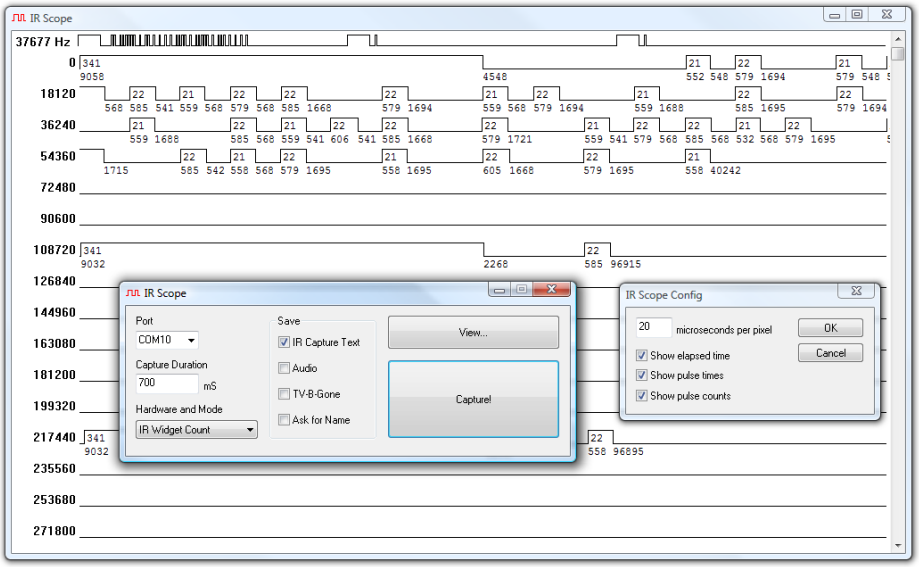


Fig. 24 Aplicação "IR Scope" [37]

O "IR Widget" mesmo não sendo comercializado, tendo em conta as suas características, acaba por ser pertinente a sua apresentação. Para que pudesse ser aplicado num sistema de domótica teria de ter a capacidade de emissão IR, contudo a nível de armazenamento de informação, uma vez que já permite gravação de comandos num ficheiro de texto, bastaria ser adaptado para a gravação numa base de dados

MySQL, por exemplo. Todo o firmware, software e hardware (esquemático e *layout* PCB) são facultados no *site* do projecto e portanto a evolução de funcionalidades deste módulo poderá estar bastante simplificada [37].

2.4.4 **Conclusões e Análise Comparativa**

Os módulos aqui apresentados, à excepção do "conversor X10-IR", não são ainda aplicados em sistemas de domótica, contudo se forem feitas as devidas alterações poderão perfeitamente tornar-se soluções aplicáveis no "mundo" da domótica.

A Tabela 2 apresenta as principais características a salientar num módulo de controlo de dispositivos IR. São essas, a forma de interacção para a emissão IR (local ou remota), o tipo de tecnologia usada na transmissão de informação (sempre que existe controlo remoto), a capacidade de emissão e aquisição IR e por fim a detecção de diferentes tipos de portadoras. É feita uma comparação entre os diferentes módulos apresentados, incluindo a solução IR proposta neste projecto.

	Conversor X10-IR	Unzap	IR Widget	IR Projecto
Controlo	Remoto	Local	-	Remoto
Transmissão	X10	-	-	RF
Emissão IR	√	√	-	√
Aquisição IR	√	√	√	√
Portadora	38kHz	Qualquer	Qualquer	Qualquer

Tabela 2 Comparação entre os diversos módulos

Como se pode verificar, através da análise da Tabela 2, dos módulos apresentados neste estado da arte apenas o "conversor X10-IR" é dotado de capacidade de transmissão. Apesar desse meio de transmissão não ser via RF, este módulo não necessita também de uma pré-cablagem, o que o torna portanto bastante semelhante à solução proposta neste projecto. Ao longo da pesquisa efectuada foram encontrados diversos produtos com capacidade de transmissão via RF, porém eram na sua grande maioria conversores IR-RF, para aumento do alcance IR, ora essa característica "foge" um pouco à solução proposta daí não ter sido feita nenhuma referência a esses produtos.

O “Unzap” e o “IR Widget” podem não apresentar a totalidade das funcionalidades pretendidas neste projecto, no entanto apresentam características importantes na aprendizagem da trama IR, ponto de maior dificuldade no que toca ao controlo de dispositivos IR.

Capítulo III

3. Introdução ao Projecto Desenvolvido

3.1 Cooperação

O projecto teve o apoio da empresa SAR - Soluções de Automação e Robótica que forneceu todo o material para o desenvolvimento deste projecto.

A SAR é uma empresa portuguesa fundada em 2006 por um grupo de jovens engenheiros formados na Universidade do Minho com especial interesse em áreas como a informática, electrónica, automação e robótica. Em 2007 a empresa adquire o estatuto de *spin-off* da Universidade do Minho devido aos interesses comuns em investigação e desenvolvimento. Actualmente, a SAR desenvolve protótipos que incluem desde robôs, máquinas industriais, aplicações de software, entre outros, tendo-se tornado referência em termos de robótica e sistemas móveis devido ao desenvolvimento de protótipos como a cadeira de rodas omnidireccional, o robô para recolha de bolas de golfe, etc. A SAR é portanto uma empresa virada para o futuro, inovando e concretizando as suas ideias.

Para mais informações ou futuros contactos basta seguir o *link* http://www.sarobotica.pt/.

3.2 Introdução

O projecto apresentado nesta tese poderá ser muito resumidamente caracterizado como um sistema de domótica constituído em parte por automação industrial associada a um controlo por intermédio de um PC.

O controlo do sistema é praticamente todo ele baseado em linguagens de programação para desenvolvimento *Web*, com excepção ao uso de C/C++. O interface gráfico é desenvolvido em Flash, o tratamento e armazenamento de dados é baseado em MySQL e a "ponte" entre o software e hardware é feita com base em PHP. Para que todos estes "ingredientes" se juntem, e como tal, possam interagir entre si, é utilizado o "Apache HTTP (*Hypertext Transfer Protocol*) Server", atribuindo assim ao PC características de *Web Server*. Todas estas ferramentas, à excepção do Flash, são livres, o que mostra também a linha que este projecto pretende seguir.

O hardware encontra-se subdividido, quer isto dizer que a actuação em dispositivos tais como tomadas, lâmpadas ou estores etc, é feita por intermédio de autómatos programáveis (PLCs) a passo que nos dispositivos controlados por IR, como é o caso das televisões, por exemplo, já é feita por intermédio do denominado subsistema de controlo por IR. Todo este hardware será apresentado no decorrer desta tese.

3.3 *Web Server*

3.3.1 **Apache HTTP Server**

O "Apache HTTP Server" é um servidor *Web* (*Web Server*) livre criado em 1995 por Rob McCool, porém foi só a partir de Abril de 1996 que começou a tornar-se popular, reputação essa que mantém até a data, inclusive num estudo feito em Dezembro de 2007 foi constatado que a sua utilização representa 47,20% dos servidores activos em todo o mundo [38][39]. Este servidor *Web* actualmente é suportado por uma organização sem fins lucrativos, a ASF (*Apache Software Foundation*).

A sua filosofia assenta em critérios tais como segurança, eficiência e modularidade, tornando-o um servidor capaz de proporcionar serviços HTTP totalmente compatíveis com os standards actuais [39].

O "Apache" é constituído por diferentes módulos, os quais garantem todas as suas funcionalidades e compatibilidade com o HTTP versão 1.1. Um utilizador com conhecimentos de programação poderá inclusive criar os seus próprios módulos através da API (*Application Programming Interface*) do "Apache" [39].

A segurança como já foi dito é um ponto importante e como tal o "Apache" dispõe de um módulo, o "mod_ssl.so", que torna o servidor compatível com o protocolo HTTPS (*Hypertext Transfer Protoco Securel*) e portanto permitindo também a troca encriptada de dados entre cliente e servidor [39].

A configuração do servidor é feita através do ficheiro "httpd.conf", editável num comum editor de texto. Neste ficheiro serão introduzidas directivas, denominadas directivas do "Apache", com base nas quais se poderá configurar todos os parâmetros que dizem respeito ao servidor, desde controlo de acessos, adição de novos módulos, etc. Todo este processo de configuração poderá ser bastante simplificado através do uso da ferramenta XAMPP (X – *Cross Platform*, A – Apache, M – MySQL, P – PHP, P –

Perl), a qual instala e configura de uma forma transparente para o utilizador, o "Apache HTTP Server" e para além disso inclui todo o restante conteúdo para que seja possível transformar um comum PC num *Web Server* com capacidade para armazenamento de dados e conteúdo dinâmico. O processo de configuração ficará assim bastante simplificado. A configuração "avançada" continuará disponível para qualquer modificação [39]. O XAMPP será apresentado em maior pormenor na secção 3.3.5.

O "Apache" pode ser obtido gratuitamente em http://httpd.apache.org/.

3.3.2 **PHP**

O PHP ("*Personal Home Page*" *Hypertext Preprogramming*) é uma linguagem de programação orientada a objectos *server-side*, ou seja, é executada do lado do servidor justamente antes da página ser enviada para o cliente. Os *scripts* que se executam no lado do servidor normalmente têm funcionalidades tais como o acesso a base de dados, acesso a redes, entre outros, é portanto aqui que o PHP se torna útil. Pode-se então dizer que esta linguagem funciona em "*background*" [41].

A linguagem foi criada por Rasmus Lerdof, tendo surgido por volta de 1994 com o nome de "*Personal Home Page Tools*", foi mais tarde rebaptizada. Inicialmente teria a função de substituir um conjunto de *scripts* Perl presentes na página pessoal deste senhor, contudo, dada a sua política *open source*, ao longo dos anos a linguagem foi tendo contribuições de vários desenvolvedores, os quais lhe foram conferindo cada vez mais funcionalidades. Actualmente, é muito mais do que um simples substituto, tendo-se tornado uma linguagem orientada a objectos extremamente robusta, estruturada e portável, portanto ideal para criar soluções *Web* fiáveis [42]. O PHP encontra-se já na versão 5.3, embora esta seja ainda uma versão "alpha1" [42].

O PHP apresenta semelhanças com o C/C++ quer a nível de sintaxe, tipos de dados ou até mesmo na criação de funções, contudo é uma linguagem interpretada ao contrário do C/C++ que são linguagens compiladas [42].

Em termos de aplicação, foram aqui apresentadas algumas funcionalidades desta linguagem, contudo no caso concreto deste projecto, o PHP desempenha duas funções primordiais. São essas, o acesso para escrita e leitura na base de dados e a "ponte" para comunicação com o hardware, seja com o subsistema IR ou com o PLC principal.

O PHP pode ser obtido gratuitamente em http://www.php.net/downloads.php.

3.3.3 **MySQL**

O MySQL é um sistema gestor de base de dados (SGBD) baseado na linguagem SQL (*Structured Query Language*). Este sistema foi criado na década de 80 pelos suecos David Axmark e Allan Larsson e o finlandês Michael "Monty" Widenius. Actualmente, o MySQL é propriedade da Sun Microsystems [44].

O sistema MySQL apresenta um grande sucesso, sendo o sistema eleito por empresas tais como a NASA (***N**ational **A**eronautics and **S**pace **A**dministration*), Texas Instruments, Motorola, entre outras. "É atualmente um dos bancos de dados mais populares, com mais de 10 milhões de instalações pelo mundo" [44]. O segredo deste sucesso é derivado de características tais como a sua portabilidade, compatibilidade com diversas linguagens de programação, suporte para vários tipos de tabelas (como "InnoDB" por exemplo), elevada estabilidade e desempenho, política de *open source*, possibilidade do uso de *Triggers*, entre outras [44]. O MySQL encontra-se já na versão 5.0.67 [44].

A aplicação do MySQL neste projecto passa pelo armazenamento de toda a informação do sistema de domótica, desde utilizadores, pisos, compartimentos e dispositivos inseridos, estado de cada dispositivo, informação relativa aos comandos IR, etc. Todos estes pontos serão apresentados em pormenor no capítulo 5.

O MySQL pode ser obtido gratuitamente em http://dev.mysql.com/downloads/mysql/5.0.html#downloads.

3.3.4 **Flash**

O Flash é um software para criação de animações interactivas que funcionam embutidas num navegador *Web*, não significando que apenas possa ser aplicado a navegadores *Web*, tendo outras aplicações. O software suporta imagens *bitmaps* e vídeos, contudo é considerado essencialmente de gráfico vectorial. Esta tecnologia foi

desenvolvida por Jonathan Gay que trabalhou no sentido de desenvolver programas dotados de interacção e animação. Em 1995 fundou a empresa Futurewave, continuando assim a evoluir a tecnologia de animação gráfica. Mais tarde, em 1996, a Macromedia (empresa de desenvolvimento de software gráfico e *Web*) adquiriu a Futurewave, foi então que o Flash ganhou novas e poderosas funcionalidades. A partir desta altura o Flash adoptou também uma linguagem de programação própria, o Actionscript, que lhe conferiu uma elevada liberdade na criação de animações. Quando se fala de Flash é usual falar também de Actionscript daí por vezes se denominar Flash/Actionscript. Actualmente, a Macromedia pertence à Adobe Systems, tendo sido adquirida em 1996 [46][47].

Este software surge portanto associado a gráficos vectoriais, aqui fica uma pequena noção teórica. Existem dois tipos de gráficos, os *bitmaps* e os vectoriais. Os *bitmaps*, ou mapas de *bits*, são aqueles constituídos por *pixels* nos quais é definida uma determinada cor. O que acontece então é que a informação nestes gráficos é guardada individualmente para cada pixel (coordenada e cor de cada *pixel*), ora isto manifesta logo uma dependência com o tamanho e resolução, e portanto conduz a uma perda de qualidade ao serem feitas alterações sucessivas à imagem, como é o caso de uma animação por exemplo. Os gráficos vectoriais são ligeiramente diferentes, neste caso a imagem é representada por vectores (linhas) que possuem algumas características, tais como cor, espessura, etc. Este gráficos não são dependentes do tamanho ou resolução e portanto nunca perdem a sua qualidade, para qualquer tamanho. Normalmente são utilizados em logótipos ou imagens mais simples, porém têm também a capacidade de criação de imagens bastante complexas. Em suma, cada tipo de gráfico tem a sua aplicação e como tal nenhum substitui o outro [47].

O Flash proporciona a criação de interfaces gráficos de elevada qualidade e fiabilidade, aliada a estas características surge também a possibilidade de comunicação com o exterior, como tal, adapta-se perfeitamente a este projecto. Neste caso em concreto a sua utilização está presente em toda a configuração, monitorização e actuação no sistema de domótica. Falar de Flash implica também falar de uma interacção baseada em botões e *Movie Clips*, os quais são fundamentais na aplicação desenvolvida.

A versão utilizada é a Macromedia Flash Professional versão 8/Actionscript 2.0. A versão *trial* do Adobe Flash CS4 Professional/Actionscript 3.0 (versão mais recente) poderá ser adquirida em http://www.adobe.com/downloads/.

3.3.5 **XAMPP**

O XAMPP é a ferramenta que possibilita, de uma forma bastante simples, a interacção entre as diferentes tecnologias apresentadas anteriormente, é por assim dizer o "coordenador" do *Web Server*. Dadas as dificuldades que existem em configurar um servidor *Web* "Apache" com suporte para funcionalidades tais como o PHP e MySQL, esta ferramenta veio simplificar essa árdua tarefa. O XAMPP encontra-se integrado no projecto "Apache Friends", um projecto sem fins lucrativos iniciado em 2002 por Kai Oswald Seidler e Kay Vogelgesang. No fundo, o objectivo é promover o servidor *Web* "Apache" tornando a sua configuração bastante mais simples em diferentes sistemas operativos [49].

A ferramenta XAMPP instala automaticamente o "Apache Web Server", PHP e MySQL. O Perl será uma extensão ao pacote XAMPP base. São instaladas ainda diversas aplicações, das quais é de salientar o phpMyAdmin que se mostrou bastante útil no teste e criação da base de dados deste projecto, que permite facilmente a manipulação de qualquer base de dados MySQL [49].

O sistema operativo utilizado foi o Windows XP e portanto para integrar quer todo o processo de instalação do XAMPP quer a posterior colocação de todos os ficheiros necessários no servidor, foi criado um ficheiro de instalação (*installshield*). Com base neste ficheiro o utilizador terá apenas de especificar a localização para o directório que suportará todo o sistema. Todo este processo de instalação, bem como alguns pontos específicos de configuração extra, nomeadamente instalação de *drivers*, encontram-se descritos no manual, presente em anexo. A versão do XAMPP utilizada foi a 1.6.7. Esta ferramenta poderá ser adquirida gratuitamente em http://www.apachefriends.org/en/xampp.html.

No site indicado encontra-se disponível o XAMPP (versão completa) e o XAMPP lite. Por experiência própria é aconselhada a instalação da versão completa, tendo surgido alguns problemas com a versão lite, nomeadamente no uso de *Triggers* e referências.

3.4 Arquitectura

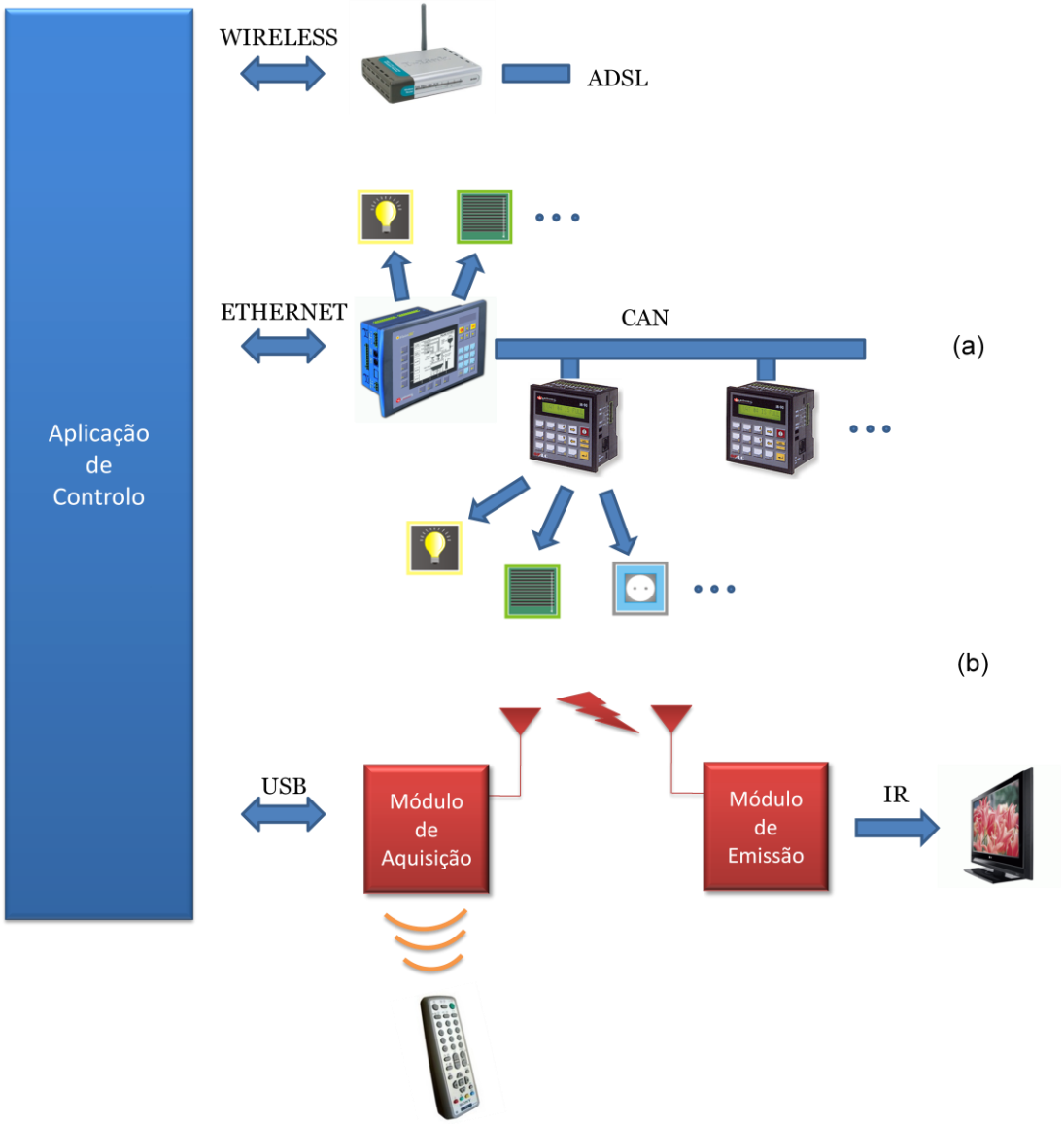


Fig. 25 Arquitectura geral do sistema de domótica

A Fig. 25 ilustra a arquitectura geral do sistema. (a) Rede de PLCs responsável pelo controlo dos diversos dispositivos espalhados pela casa. (b) Subsistema para controlo específico de dispositivos IR.

O objectivo deste trabalho como já foi dito é controlar o sistema através de uma aplicação Flash, contudo é também possível o controlo em paralelo através de botões de pressão, conectados às entradas dos respectivos PLCs. O acesso remoto ao sistema é igualmente possível, contudo esta funcionalidade encontra-se ainda um pouco limitada, uma vez que para já só é possível aceder via ambiente de trabalho remoto.

Capítulo IV

4. Autómatos Programáveis

4.1 Introdução

A actuação nos diversos dispositivos existentes no sistema, à excepção dos dispositivos controlados por IR, é feita por intermédio de autómatos programáveis, utilizados normalmente em ambientes industriais, os PLCs. Nesta secção serão apresentadas as principais características dos autómatos usados, indicando também as principais vantagens do uso desta tecnologia e o porquê do seu uso em ambientes residenciais.

O PLC começou a ser desenvolvido em 1968, tendo sido pela primeira vez aplicado na indústria em 1969. Este dispositivo foi criado no sentido de substituir os sistemas automáticos existentes na indústria automóvel e desde então tem vindo a adquirir cada vez mais funcionalidades. Actualmente é utilizado em diversos sectores da indústria e não só no ramo automóvel [50].

O controlo baseado em autómatos programáveis apresenta diversas vantagens em relação a outros tipos de controlo. Estes dispositivos são especialmente desenvolvidos para serem utilizados em terreno "hostil" para os dispositivos eléctricos, como é o caso de ambientes húmidos, ruidosos, etc., daí estarem "enraizados" em praticamente todos os sectores da indústria. São dispositivos fáceis de programar, pois usam uma linguagem muito acessível, linguagem de contactos ou *Ladder*. A sua estrutura modular simplifica bastante a adição de novos módulos e portanto novas funcionalidades. A distribuição de entradas/saídas já embutidas no dispositivo torna a instalação mais simples. Existem também algumas desvantagens, como é lógico. Apesar de ser um dispositivo simples de instalar e programar, requer sempre mão-de-obra qualificada, especialmente na detecção de erros, que por vezes se pode tornar complicada. Apesar de já existirem interfaces gráficos HMI (*Human-Machine Interface*), o interface utilizador-PLC é bastante inferior a um interface utilizador-PC por exemplo, o que poderá ser relevante, como é o caso deste projecto [54].

Face portanto ao elevado desempenho que o PLC tem comprovado em ambientes industriais é seguramente uma tecnologia aplicável num sistema de automação residencial, surgindo assim associado a um interface via PC.

O sistema presente neste projecto é constituído por PLCs da Unitronics. O PLC principal é o VISION 280 e todos os restantes são os M90. Estes foram os autómatos

fornecidos para o desenvolvimento do projecto, contudo dada a estrutura modular do software, é possível usar qualquer tipo de hardware, desde que sejam feitas as devidas alterações ao nível do PHP, nomeadamente nos *scripts* de comunicação.

4.2 Autómatos do Sistema

4.2.1 VISION 280

- **Características Principais** [52]
  - Suporte até 171 I/Os (*Inputs/Outputs*) através de módulos de expansão ou de acoplamento directo
  - Tipos de *Inputs*: Digital e Analógico
  - Tipos de *Outputs*: Digital, Analógico e Relé
  - Memória de Código *Ladder*: 1000 k
  - Memória de dados: 120 k (RAM) e 192 k (*Flash*)
  - Interface gráfico HMI Táctil
  - Comunicações: RS-232/RS-485, *Ethernet* (módulo opcional), suporte para GSM/SMS (*Short Message Service*), MODBUS, CANbus e acesso remoto via GPRS, GSM, CDMA (*Code Division Multiple Access*) ou modem
  - Montagem em painel

Algumas das funcionalidades deste PLC não foram usadas, nomeadamente o interface táctil HMI, uma vez que todo o interface gráfico é baseado em Flash, e portanto é feito através do PC.

Este PLC é a "cabeça" do sistema, por assim dizer, uma vez que será o responsável pela comunicação entre o software de controlo e todo o sistema, bem como

pelo encaminhamento de informação e monitorização da rede CAN composta pelos M90. Este PLC poderá também ter dispositivos a ele conectados.

O PC comunica com o VISION via *Ethernet* através do protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol*), desta forma é possível controlar/monitorizar qualquer dispositivo conectado não só ao VISION mas também a qualquer M90. No fundo sempre que se pretende actuar num determinado dispositivo, na realidade está a fazer-se o SET (‘1’) / RESET (‘0’) num determinado *bit* de memória do PLC. Caso esse *bit* esteja associado ao VISION, este será modificado imediatamente, caso contrário, se estiver associado a um M90, será encaminhada uma ordem de actuação via CAN para o *bit* de memória no M90 respectivo. O processo de leitura (READ) é semelhante, neste caso existirá um pedido de leitura de um determinado *bit*. Este encaminhamento de informação, quer para escrita ou leitura, foi uma solução desenvolvida por Marco Teixeira num projecto de estágio na SAR. Com base nesta solução, ao nível da aplicação de controlo a única preocupação a ter será saber quais os *bits* associados aos respectivos dispositivos, e portanto todo o processo de controlo/monitorização passará pelo SET/RESET/READ de *bits* de memória definidos no PLC VISION 280.

4.2.2 **M90**

- **Características Principais** [53]
  - Suporte até 38 I/Os através de acoplamento directo
  - Tipos de *Inputs* (acoplamento directo): Digital e Analógico
  - Tipos de *Outputs* (acoplamento directo): Digital e Relé
  - Suporte até 96 I/Os através de módulos de expansão
  - Tipos de *Inputs* (módulos de expansão): Digital e Analógico

- Tipos de *Outputs* (módulos de expansão): Digital
- Memória de Código *Ladder*: 24 – 36 k
- Memória de dados: 1 k
- Interface HMI LCD 16x2
- Comunicações: RS-232/RS-485, suporte para GSM/SMS, MODBUS e CANbus e acesso remoto via GPRS, GSM, CDMA ou modem
- Montagem em calha DIN ou painel

O M90 é o PLC responsável pela conexão da maior parte dos dispositivos existentes no sistema. O objectivo é conectar diversos M90 através de uma rede CAN, de forma a controlar os diversos dispositivos espalhados pela habitação. Os M90 poderão comunicar entre si e também com o VISION 280, formando assim parte do sistema de domótica. Com base nesta rede CAN é possível conectar até 63 PLCs, um número perfeitamente aceitável.

Os diversos PLCs, tanto os M90 como o VISION, estarão dotados de "inteligência" local, isto é, cada um deles terá o *Ladder* respectivo para o controlo local (através de botões de pressão/sensores) dos diversos dispositivos, contudo, uma vez que este sistema não foi implementado numa habitação real, será feito um *case study* no qual será simulada a interacção/monitorização de todo o sistema através da manipulação de bits de memória do VISION 280, não existindo portanto a ligação física da rede CAN. Neste *case study* será apresentada toda a estrutura a nível de *Ladder* e posteriormente será explicado todo o processo a nível de software, para o controlo genérico de dispositivos.

4.3 Comunicação via *Ethernet*

A aplicação de controlo, como já foi dito anteriormente, actua sobre *bits* de memória do PLC, seja para escrita (SET/RESET) ou leitura (READ). Para se proceder a essa manipulação de *bits* será necessário saber quais os comandos a enviar ao PLC, no sentido de efectuar a acção pretendida. Tendo em conta que a comunicação será feita via *Ethernet*, será também necessário saber quais os parâmetros necessários para esse tipo de comunicação.

O VISION permite dois tipos de formato para a troca de informação com o PC, o formato ASCII e o BINÁRIO. Neste caso foi utilizado o formato ASCII uma vez que o comando será sempre no sentido de actuar (SB – *Set Memory Bits* a '0' ou '1') e ler (RB – *Read Memory Bits*) *Memory Bits* (MB). Se fosse necessário ler diferentes tipos de memória no mesmo pedido, por exemplo RB e RE (*Read Inputs*) era imperativo o uso do formato BINÁRIO.

4.3.1 **Formato ASCII**

O formato genérico das tramas PC→VISION é o seguinte:

<STX>	**<UnitID>**	**<CommandCode>**	**<CommandParameters>**	**<CheckSum>**	**<ETX>**

<STX> Início da transmissão ('/' – ASCII 47);
<UnitID> Identificação do VISION (definido como 01) (2 caracteres);
<CommandCode> Tipo de acesso (SB ou RB, por exemplo);
<CommandParameters> Endereço inicial (4 caracteres) + SET/RESET de todos os *bits* a actuar ('0' ou '1') / Número de bits a ler;
<CheckSum> Verificação da integridade do conteúdo da trama (CRC) (2 caracteres), o cálculo do CRC é indicado em 4.3.2;
<ETX> Fim da transmissão ('CR' (*Carriage Return*) – ASCII 13).

O formato genérico das tramas VISION→PC é o seguinte:

<STX1>	**<UnitID>**	**<CommandCode>**	**<CommandParameters>**	**<CheckSum>**	**<ETX>**

<STX1> Início da transmissão ('/A');
<UnitID> Identificação do VISION (2 caracteres);
<CommandCode> Tipo de acesso (SB ou RB, por exemplo);
<CommandParameters> Ignorado (caso do SB por exemplo) / Estado do(s) *bit(s)* lidos (RB por exemplo);
<CheckSum> CRC enviado pelo PLC;
<ETX> Fim da transmissão ('CR').

Nota: Os valores introduzidos na criação das tramas a enviar ao PLC terão de estar representados em hexadecimal (HEX). Por exemplo, se for pretendido actuar no

endereço 32 (DEC), o valor a preencher no campo endereço será (0020 (HEX)). É importante salientar este aspecto para uma correcta percepção de toda a implementação.

- Exemplo concreto do SET de um *Memory Bit* (SB)

PC→PLC (Envio)

<STX>	<UnitID>	<CommandCode>	<Address>	<Length>	<Values>	<Checksum>	<ETX>
'/'	01	SB	0000	01	1	CRC	'CR'

PLC→PC (Resposta)

<STX1>	<UnitID>	<CommandCode>	<Checksum>	<ETX>
'/A'	01	SB	CRC	'CR'

Fig. 26 Exemplo do SET de um *Memory Bit*

A trama para escrita em *Memory Bits* é representada por SB.

Como se pode ver, no envio, o campo <CommandParameters> é desdobrado nos campos <Address>, <Length> e <Values>.

<Address> Endereço Inicial a partir do qual se pretende alterar o valor de *bits* de memória. No caso particular deste projecto, apenas interessa actuar *bit* a *bit*, como tal este campo representará o endereço a actuar, 0000 (HEX);
<Length> Número de bits a actuar, neste caso será preenchido com 01 (HEX) pois apenas será actuado 1 *bit*;
<Values> Valor pretendido (SET – '1', RESET – '0').

No caso da resposta, a trama enviada pelo PLC poderá ser ignorada, não contendo portanto informação relevante.

- Exemplo concreto da leitura de um *Memory Bit* (RB)

PC→PLC (Envio)

<STX>	<UnitID>	<CommandCode>	<Address>	<Length>	<Checksum>	<ETX>
'/'	01	RB	0000	05	CRC	'CR'

PLC→PC (Resposta)

<STX1>	<UnitID>	<CommandCode>	<Values>	<Checksum>	<ETX>
'/A'	01	RB	10101	CRC	'CR'

Fig. 27 Exemplo da leitura de um *Memory Bit* (RB)

A trama para pedido de leitura de *Memory bits* é representada por RB.

A trama RB é ligeiramente diferente da SB. Neste caso o campo <CommandParameters> é desdobrado apenas nos campos <Address> e <Length>.

<Address> Neste caso representará o endereço inicial para a leitura de *bits*, uma vez que numa só trama será feito o pedido de leitura de vários *bits*. Este campo é composto por 4 caracteres, no exemplo 0000 (HEX)**;**

<Length> Indica o número de *Memory bits* a ler (será definido pelo utilizador no monitor de sistema). Só se poderá ler um máximo de 256 *bits* a cada pedido de leitura e portanto este campo é constituído por 2 caracteres, no exemplo é feita a leitura de 5 *bits* (05 (HEX)) a partir do endereço 0000 (HEX).

Em relação à trama de resposta, o único campo importante é o campo <Values>, sendo ignorados todos os restantes. Neste campo aparecerá o estado dos *bits* pretendidos. No exemplo 10101, correspondente ao estado de cada um dos *bits* desde o endereço 0000 (HEX) até 0004 (HEX).

4.3.2 ***Checksum* Formato ASCII**

O *Checksum* é obtido através da seguinte expressão:

$$T_{ASCII} MOD\ 256 \quad (1)$$

Onde T_{ASCII} representa a soma dos valores ASCII de toda a trama.

No fundo o CRC será o resto da divisão entre T_{ASCII}e 256, em hexadecimal.

Considerando a trama SB do exemplo anterior: "/01SB0000011" + <CR>.

O CRC é calculado sem o STX ('/') e ETX ('CR'). O cálculo será feito da seguinte forma:

ASCII	**0**	**1**	**S**	**B**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
DEC	48	49	83	66	48	48	48	48	48	49	49

Fig. 28 Exemplo do valor DECIMAL de caracteres ASCII

$$T_{ASCII} = 48 + 49 + 83 + 66 + 48 + 48 + 48 + 48 + 48 + 49 + 49 = 584 \text{ (DEC)}$$

$$T_{ASCII} MOD\ 256 = 72 \text{ (DEC)} = 48 \text{ (HEX)}$$

O CRC será então 48 (HEX).

A trama SB obtida será "/01SB000001148" + <CR>.

4.3.3 **Cabeçalho *Ethernet***

A comunicação PC-PLC é feita via *Ethernet*, como tal para que seja possível comunicar desta forma, é necessário adicionar a cada trama ASCII um cabeçalho *Ethernet*, constituído por 6 bytes. O seu formato é o seguinte:

Bytes	**0**	**1**	**2**	**3**	**4**	**5**
	LSB	MSB	Protocol	Reserved	LSB	MSB
	Transaction Identifier		ASCII/BINARY	(0)	Length of Transaction	

Fig. 29 Constituição do cabeçalho *Ethernet*

"Transaction Identifier" – Este campo é definido pelo utilizador, neste caso foi definido o valor 1234 (DEC - Decimal). Os campos LSB (*Low Significant Byte*) e MSB (*Most Significant Byte*) são preenchidos com 210 (DEC) e 4 (DEC), respectivamente;

"Protocol" – Identificação do formato da trama a usar (101 (DEC) → ASCII, 102 (DEC) → BINARY);

"Reserved" – Este campo é reservado e como tal deve ser preenchido com zero;

"Length of Transaction" – Tamanho da trama (*bytes*) a enviar para o PLC, incluindo <STX> e <ETX>.

Seguindo o exemplo do SB, apresentado na secção 4.3.1, o cabeçalho *Ethernet* seria preenchido da seguinte forma:

"Transaction Identifier" – Definido no projecto como 1234 (DEC): LSB – 210 (DEC), MSB – 4 (DEC);

"Protocol" – Trama no formato ASCII logo a sua identificação será 101 (DEC);

"Reserved" – Campo reservado, preenchido com zero;

“Length of Transaction” – A trama a enviar é a seguinte: “/01SB000001148”+<CR>, o que dá um total de 15 caracteres (0000 1111 (BIN)), logo LSB – 15 (DEC), MSB – 0 (DEC).

Preenchendo os respectivos campos o cabeçalho ficará da seguinte forma:

BIN	1101 0010	0000 0100	0110 0101	0000 0000	0000 1111	0000 0000
DEC	210	4	101	0	15	0
	LSB	MSB	Protocol	Reserved	LSB	MSB
	Transaction Identifier		ASCII/BINARY	(0)	Length of Transaction	

Fig. 30 Cabeçalho *Ethernet* correspondente à trama em exemplo

4.3.4 **Trama Final**

Concluídos todos os passos, a trama final estará pronta a ser enviada para o VISION via *Ethernet*, faltando apenas identificar o IP e a Porta para a comunicação.

Seguindo mais uma vez o exemplo do SB, a trama final será a seguinte:

DEC						**ASCII**													
210	4	101	0	15	0	/	0	1	S	B	0	0	0	0	0	1	1	4	8	CR
Cabeçalho *Ethernet*						**<STX>**	**<UnitID>**		**<CC>**		**<Address>**				**<Length>**		**<Value>**	**CRC**		**CR**

Fig. 31 Apresentação final para a trama em exemplo

4.3.5 **Implementação em PHP**

A comunicação PC→PLC é executada a partir do servidor, como tal a sua implementação é feita em PHP. Para a criação e envio das tramas foram definidas três funções, **“gera_escritaMB”**, **“gera_leituraMB”** e **“envia_comando_plc”**. Estas funções estão presentes nos *scripts* PHP de envio e monitorização. Os *scripts* serão apresentados no capítulo 6, nesta secção será apresentada apenas a implementação destas funções de comunicação com o PLC.

- **gera_escritaMB($endereco,$valor)**

Esta função é responsável pela criação da trama para a escrita em *Memory Bits* (*Set Memory Bits*). Recebe como parâmetros o endereço do *bit* a actuar e o valor dessa actuação. Retorna uma trama com todos os campos correctamente definidos (cabeçalho *Ethernet* + comando completo) e portanto pronta para ser enviada ao PLC via *Ethernet*.

No fundo esta função traduz os passos anteriormente apresentados para a criação da trama a enviar para o PLC. Basicamente, a sequência de código é a seguinte:

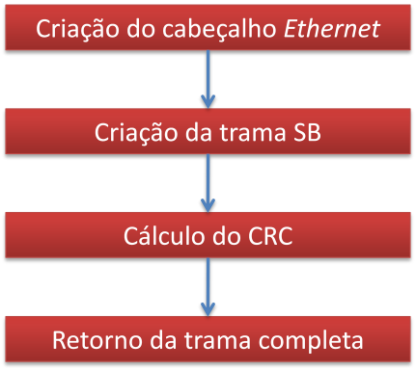


Fig. 32 Sequência de implementação da função "gera_escritaMB"

<PHP>

```
<?php

function gera_escritaMB($endereco, $valor)                    //2 argumentos: endereço do bit a actuar, valor ('0' ou '1')
{
        $init_string=chr(210) . chr(4) . chr(101) . chr(0) . chr(14) . chr(0) . chr(47);   //cabeçalho Ethernet + '/'
                                                                                            //(campo 4 temporário)
        $comando=sprintf("01SB%04X01%01X", $endereco, $valor);                              //comando a enviar (SB)
        $comprimento=strlen($comando) + 4;    //ajuste no comprimento, mais 4 bytes à trama actual ('/','CRC','CR')
        $init_string[4]=chr($comprimento);                                                  //actualizar campo 4 – "LSB do Length of Transaction"

        $soma=0;                                                                            //inicializar variável (soma dos valores ASCII da trama)

        for ( $i = 0; $i < strlen($comando); $i++)
                $soma = $soma + ord($comando[$i]);                                          //resultado da soma dos valores ASCII
        $checksum=sprintf("%02X", $soma % 256);                                             //cálculo do CRC: $soma MOD 256

        return $init_string . $comando . $checksum . chr(13);                              //devolver trama completa
                                                                                            //(cabeçalho Ethernet + comando completo)
}

?>
```

- **gera_leituraMB($endereco,$quantidade)**

A função "gera_leituraMB" é muito semelhante à "gera_escritaMB", a diferença está na criação do comando a enviar, cujos parâmetros serão diferentes, uma vez que se pretende fazer a leitura de um dado número de *Memory Bits*. Esta função recebe como parâmetros o endereço inicial e a quantidade de *Memory Bits* que se pretende ler (a partir do endereço inicial, inclusive).

<PHP>

```
<?php

function gera_leituraMB($endereco, $quantidade)     //2 argumentos: endereço inicial, quantidade de endereços a
                                                    //monitorizar (a partir do endereço inicial)
{
        $init_string = chr(210) . chr(4) . chr(101) . chr(0) . chr(14) . chr(0) . chr(47);

        $comando =sprintf("01RB%04X%02X", $endereco, $quantidade);          //comando a enviar (RB)
        $comprimento=strlen($comando) + 4;
        $init_string[4]=chr($comprimento);

        $soma=0;

        for ( $i = 0; $i < strlen($comando); $i++)
                $soma = $soma + ord($comando[$i]);
        $checksum=sprintf("%02X", $soma % 256);

        return $init_string . $comando . $checksum . chr(13);
}

?>
```

- **envia_comando_plc($comando)**

Esta função é responsável pela comunicação com o PLC, enviando as tramas criadas e recebendo a resposta do PLC. No caso da trama SB, a resposta do PLC será ignorada. Esta função recebe como argumento a trama completa criada por uma das funções anteriores.

A comunicação será estabelecida através de *sockets* e portanto é necessário definir o IP e a Porta de comunicação do PLC. A sequência do código é a seguinte:

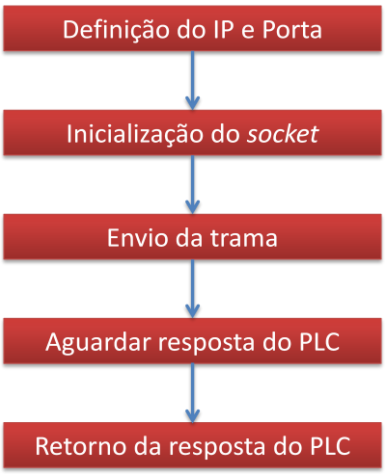


Fig. 33 Sequência de implementação da comunicação com o PLC

<PHP>

```
<?php

function envia_comando_plc($trama)          //1 argumento: trama a enviar (cabeçalho Ethernet + comando)
{
        $IP="192.168.0.112";                                    //ip associado ao PLC
        $port="502";                                            //porta associada ao PLC
        $resposta="";                                           //trama enviada pelo PLC

                $fp = fsockopen($IP, $port, $errno);            //abrir socket para conexão ao PLC
                if ($fp)
                {
                        fputs($fp, $trama);                     //enviar informação
                        do
                        {
                                $characters = fgetc($fp);
                                $resposta = $resposta . $characters;    //guardar trama enviada pelo PLC
                        }
                         while($characters != chr(13));         //esperar pela recepção de todos os caracteres
                                                                //(final da trama <CR>)

                }
        return $resposta;                                       //devolver resposta do PLC
}

?>
```

4.4 *Ladder (Case Study)*

A comunicação implementada em PHP não servirá de nada se à partida não estiverem definidos os *bits* de memória nos quais se pretende actuar, aos quais estarão associados os diferentes dispositivos (dispositivos não IR).

Para que seja possível mostrar o funcionamento do sistema, foi criado um diagrama de escada (*Ladder*) para o controlo de alguns dispositivos, nomeadamente lâmpadas, estores e tomadas. O controlo poderá ser feito localmente através de botões de pressão conectados às entradas do VISION 280 (simulação com as teclas do painel deste PLC) e através da aplicação Flash. Este controlo paralelo é possível porque quando se pretende actuar num determinado dispositivo não se está a actuar directamente no endereço físico associado a esse dispositivo, mas sim num MB (*Memory Bit*) intermédio. Esse MB será o responsável pelo SET/RESET no endereço físico respectivo. Poderá então distinguir-se dois tipos de endereços, o endereço de memória (MB) e o endereço físico. A análise da Fig. 34 ajuda a clarificar esta ideia de controlo paralelo. Como se pode verificar o botão de pressão (SB 41 – botão 1 no painel do PLC) actua no MB 0 (SET (S)/RESET (R)) que por sua vez actuará na saída do PLC, neste caso O 0. A figura ilustra a actuação na(s) lâmpada(s) conectada(s) à saída 0 do VISION.

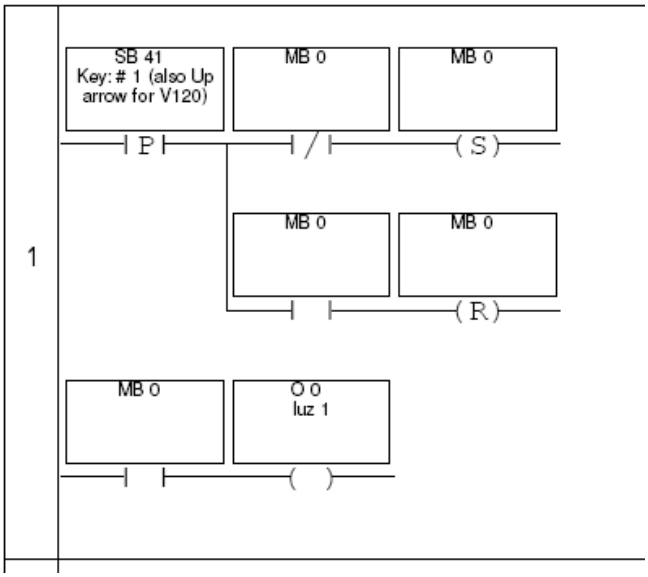


Fig. 34 Princípio de actuação do sistema

Todo o *Ladder* foi desenvolvido através do software da Unitronics, o "VisiLogic Ladder & HMI" que poderá ser obtido gratuitamente em http://www.unitronics.com/Content.aspx?page=Downloads&CatId=4.

O *Ladder* desenvolvido encontra-se disponível em anexo, como tal, nesta secção serão apresentados apenas alguns pontos relevantes, nomeadamente a configuração da comunicação via *Ethernet* e o controlo de estores.

4.4.1 **Configuração *Ethernet***

A comunicação PC→PLC via *Ethernet* só será possível após uma determinada configuração através de *Ladder*. Esta configuração assemelha-se à configuração *master-slave* de um sistema SCADA (*Supervisory Control And Data Acquisition*), no qual o PC representa o *master* e o PLC o *slave*. Para tal será necessário adicionar alguns blocos na rotina principal:

- "PLC NAME" – Identificador do PLC
- "TCP/IP CARD INIT" – IP associado ao PLC
- "MODBUS IP CONFIG" – configuração do modo MODBUS/IP:
  Encapsulamento dos dados em frames TCP/IP para que seja possível a transmissão via *Ethernet*
- "Ethernet Card Initialized", "Ethernet Socket 2 Connected" e "MODBUS IP" – activação da comunicação PC→PLC

A Fig. 35 ilustra todos estes blocos responsáveis pela comunicação via *Ethernet* através de MODBUS/IP.

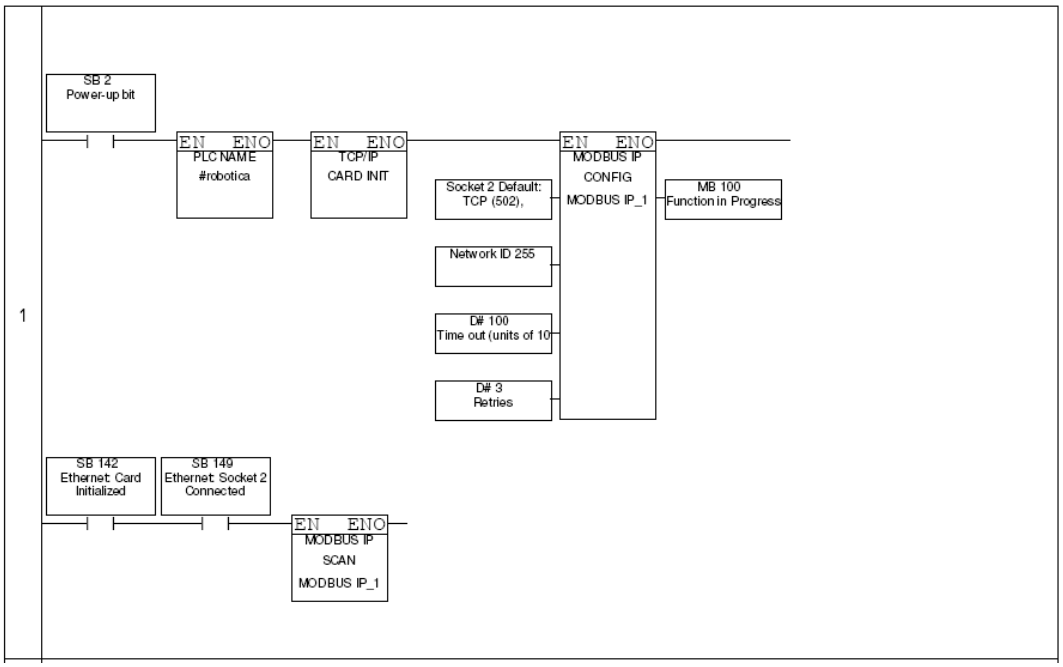


Fig. 35 Configuração da comunicação via *Ethernet*

4.4.2 **Controlo de Estores**

O controlo de estores é ligeiramente diferente do controlo dos restantes dispositivos ON/OFF (ilustrado na Fig. 34), uma vez que no estore serão necessárias duas saídas físicas (subir e descer estore). Os estores são controlados localmente por dois botões de pressão, subir estore e descer estore. Na Fig. 36 é possível ver o exemplo do controlo de subida do estore (O 3), o mesmo acontece para a descida, porém em saídas (O 4) e MB diferentes (MB 4).

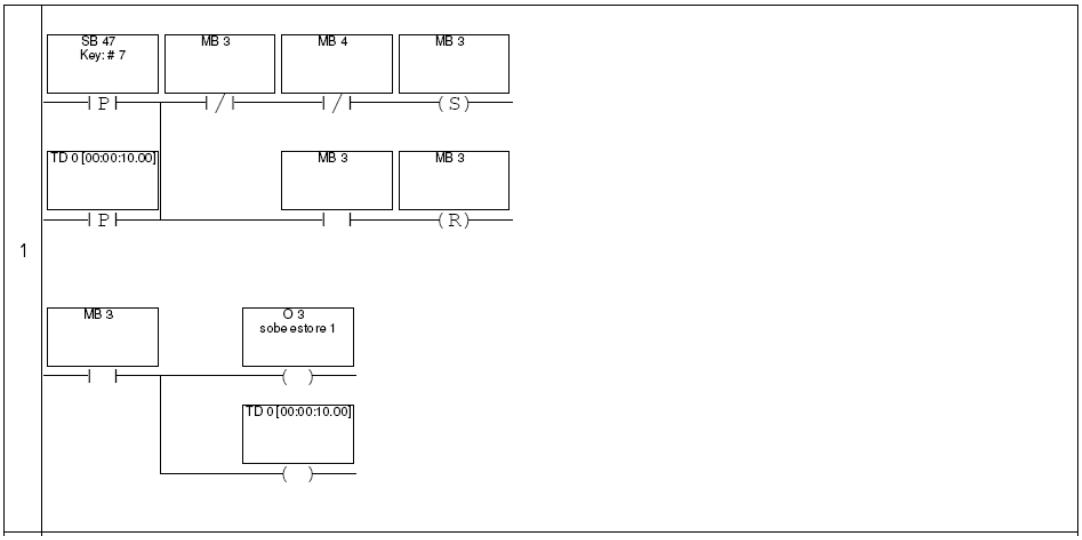


Fig. 36 Actuação de estores

No desenvolvimento do *Ladder* para o controlo deste tipo de dispositivos foram levados em conta alguns critérios, nomeadamente a impossibilidade de premir o botão de descida enquanto o estore estiver a subir e vice-versa. Foi também introduzido um *timer* (TD 0 na figura) cujo objectivo será fazer o RESET da saída activa (subir/descer estore) ao fim de X tempo, isto para que o estore esteja sempre no chamado estado de "repouso". Este tempo representa o tempo que o estore demora a abrir/fechar totalmente a partir do estado de "repouso", encontra-se definido em 1 minuto para efeitos de teste. Na Fig. 36 é possível verificar a aplicação destes critérios.

O *case study* apresentado neste projecto apresenta apenas o controlo de 2 lâmpadas (saídas independentes), 1 tomada e 1 estore. No capítulo 7, com a apresentação do subsistema de controlo de dispositivos por IR, será também adicionada ao sistema a possibilidade de controlar televisões (exemplo usado).

Capítulo V

5. Software de Controlo

5.1 Introdução

O software de controlo foi desenvolvido tendo em conta alguns critérios, nomeadamente a criação de um ambiente gráfico simples, intuitivo e amigável para o utilizador. Para além destes critérios, existiu também uma forte aposta na criação de uma estrutura modular, no sentido de simplificar qualquer modificação que seja necessária no código da aplicação. Num eventual trabalho futuro, esta estrutura permite também uma distribuição do trabalho numa equipa, possibilitando assim um desenvolvimento em paralelo.

Tendo sempre presentes os critérios apresentados e analisando também diversos tipos de software existente no mercado, surgiu a ideia de criar um software que permitisse ao utilizador visualizar a planta da sua habitação com a possibilidade de *zoom* de compartimentos e visualização espacial dos dispositivos, possibilitando assim uma interacção mais atractiva e selectiva. O mecanismo de configuração não foi deixado de parte, tendo sido criada uma área de configuração na qual o utilizador poderá de uma forma simples configurar a sua habitação, desde a simples introdução da(s) planta(s), posicionamento dos compartimentos e dispositivos dentro dos compartimentos, configuração de dispositivos IR, etc. Na Fig. 37 é possível ver de uma maneira geral o aspecto gráfico desta aplicação de controlo.

Fig. 37 *Layout* do software de controlo

5.2 Estrutura da Aplicação

A aplicação de controlo é composta por sete módulos. O "acesso", "intro", "menu", "configuracao", "sistema", "hardware" e "monitor". Todos estes módulos foram desenvolvidos em Flash e portanto têm a extensão .swf (*Shockwave Flash File*). O módulo "monitor" é um módulo auxiliar executado no arranque da aplicação, foi criado para possibilitar a monitorização do estado dos diversos dispositivos conectados ao PLC, não estando directamente associado à interacção com o utilizador.

A interacção com a aplicação de controlo apresenta uma sequência lógica, a qual se encontra ilustrada no diagrama da Fig. 38. Duma maneira geral, esta sequência traduz-se da seguinte forma: o acesso ao sistema é condicionado por um pedido de *login* e *password*, parâmetros esses que deverão ser introduzidos no primeiro acesso. Uma vez feito o *login*, o utilizador será brindado com uma breve animação, a qual poderá ser ignorada. O menu principal aparecerá, e o utilizador terá então ao seu dispor várias opções através de três módulos distintos: configuração do sistema, alteração de dados pessoais ("configuracao.swf"), controlo de dispositivos ("sistema.swf") e gestão de hardware ("hardware.swf"). Todos estes módulos, assim como o módulo de monitorização (paralelo ao interface de controlo), serão apresentados em maior pormenor no capítulo 6.

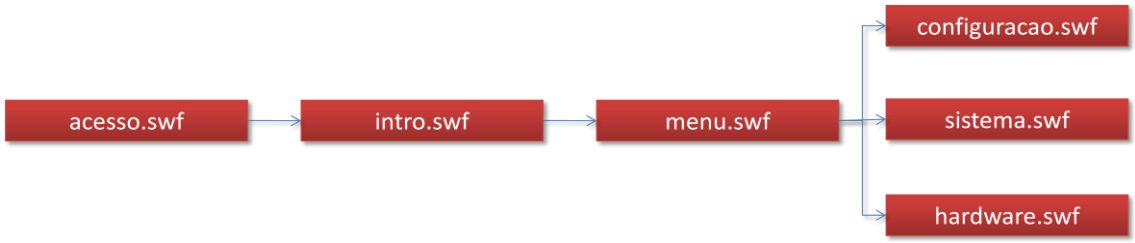


Fig. 38 Diagrama de interacção com a aplicação de controlo

No fundo esta sequência pode ser traduzida em Acesso, Configuração, Interacção e intrinsecamente em Monitorização.

5.3 Base de dados

Um sistema de controlo e monitorização necessita inevitavelmente de armazenar os seus dados. Ora este sistema não é excepção.

O armazenamento de dados, como já foi dito anteriormente, encontra-se a cargo de uma base de dados baseada em MySQL. Esta base de dados deverá acolher informação de diferentes tipos, a qual poderá ser subdividida da seguinte forma:

- Informação relativa ao utilizador: *login*, *password* e dados pessoais
- ao piso: nome e identificação da imagem associada
- aos compartimentos: piso em que se encontra, nome, localização na área de trabalho (x,y), orientação vertical ou horizontal e a área associada ao quadrado delineado ((xi,yi),(xf,yf))
- aos dispositivos: piso e compartimento onde se encontra, tipo de dispositivo, endereço físico e de memória associado, *clip* correspondente (relevante à sua identificação na aplicação) e localização na área do compartimento (x,y)
- aos estores (caso pontual): endereços físicos e de memória (subida e descida) e todos os restantes dados relativos aos dispositivos
- aos dispositivos IR: toda a informação relativa aos dispositivos à excepção dos endereços físico e de memória (campos "NULL"), endereço do módulo de emissão IR respectivo, tempos da portadora a ON e a OFF e o nome (relevante à sua configuração na aplicação)
- às tramas IR: identificação do dispositivo IR associado, nome do comando, pulsos a ON e tempos a OFF (específico da aquisição)
- à monitorização do sistema: identificação e estado dos dispositivos ("LIGADO" ou "DESLIGADO"), identificação do(s) dispositivo(s) activado(s) recentemente (para minimizar o tráfego de informação por parte da aplicação)
- à comunicação com o módulo de aquisição IR: identificação da porta COM associada

A base de dados criada, "domotica", apresenta dez tabelas. Na Fig. 39 encontra-se representado o esquema de dependências entre tabelas bem como os seus atributos.

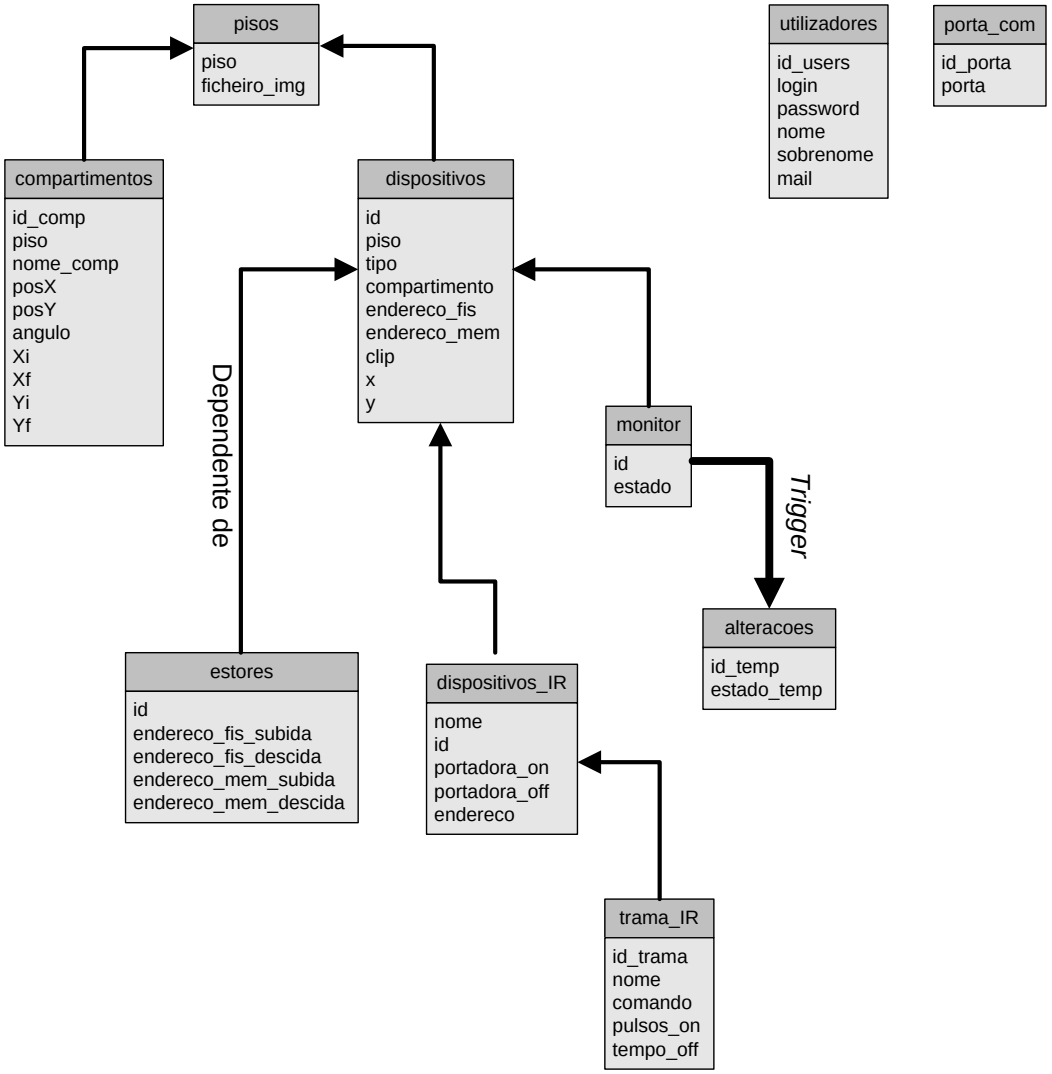


Fig. 39 Base de dados do sistema

A dependência entre tabelas foi usada para que fosse possível criar um processo de eliminação em cascata, isto é, ao existir esta dependência sempre que for eliminado um dispositivo (“id” 0 na tabela “dispositivos”, por exemplo) será desencadeado um processo de eliminação deste elemento em todas as tabelas associadas por este “id”. No caso de ser eliminado um piso, todos os compartimentos e dispositivos existentes nesse piso serão eliminados. A grande vantagem deste processo é a simplificação dos *scripts* PHP para manipulação da base de dados.

As tabelas “utilizadores” e “porta_com” são tabelas sem dependências. São usadas para armazenar os dados dos utilizadores e a identificação da porta COM associada à comunicação, respectivamente. O facto de ser eliminado um elemento na tabela “utilizadores” não afectará de forma alguma os restantes dados, a informação aqui contida diz respeito apenas a esta tabela. A tabela “porta_com”, por sua vez terá apenas

uma linha de dados, a qual nunca será eliminada sendo apenas actualizada, não afectando também o restante conteúdo da base de dados.

5.3.1 **Implementação**

A base de dados foi criada através da ferramenta phpMyAdmin, simplificando bastante todo o processo de criação, que de outra forma teria de ser feito em ambiente DOS (*Disk Operating System*). A título de exemplo, a criação da tabela "utilizadores" foi implementada da seguinte forma:

<MySQL>

```
CREATE TABLE utilizadores                                          #criação da tabela "utilizadores"
(
        id_users        SMALLINT        UNSIGNED        NOT NULL        AUTO_INCREMENT,     #campo
                                                                #"id_users", tamanho 16 bits, unsigned não nulo
                                                                #auto-incrementado a cada inserção
        login           CHAR(20)        NOT NULL,               #campo "login" máx. 20 caracteres, não nulo
        password        CHAR(20)        NOT NULL,               #campo "password" máx. 20 caracteres, não nulo
        nome            CHAR(20)        NOT NULL,               #campo "nome" máx. 20 caracteres, não nulo
        sobrenome       CHAR(20)        NOT NULL,               #campo "sobrenome" máx. 20 caracteres, não nulo
        mail            CHAR(30)        NOT NULL,               #campo "mail" máx. 20 caracteres, não nulo

        PRIMARY KEY (id_users)                                  #identificação da chave primária "id_users"

)TYPE=InnoDB;                                                   #tabela do tipo "InnoDB"
```

5.3.2 ***Trigger***

A aplicação de controlo permite a monitorização do estado de cada dispositivo, ora para que isto seja possível será necessário um acesso à base de dados com alguma frequência. Dada esta situação a aplicação teria de aceder constantemente à tabela que contém o estado de todos os dispositivos (tabela "monitor") e ler o estado dos dispositivos de um determinado compartimento. Dependendo da quantidade de dispositivos este processo pode tornar-se mais ou menos moroso, para além disso muitos dos dispositivos poderão ter o seu estado inalterado durante bastante tempo (caso das tomadas, por exemplo), ou seja, muitas vezes iria estar a ser lida informação irrelevante. Para resolver este problema foi usada uma tabela que armazena apenas o estado dos dispositivos que foram manipulados recentemente, a tabela "alteracoes" e portanto apenas será lida a informação dos dispositivos presentes nessa tabela.

A tabela “alteracoes” será preenchida a cada actualização no estado dos dispositivos da tabela “monitor”, ou seja, sempre que um dispositivo transite de “LIGADO” para “DESLIGADO” ou vice-versa. A tabela irá conter o “id” e o novo estado desse dispositivo. Esta acção é executada automaticamente através do chamado *Trigger*. A implementação em MySQL de um *Trigger* que satisfaça estas condições é feita da seguinte forma:

<MySQL>

```
CREATE TRIGGER modificou                    #criação do trigger denominado “modificou”
 AFTER UPDATE ON monitor                    #”disparar” trigger após a actualização do “estado” na tabela “monitor”
 FOR EACH ROW                               #executar trigger para alterações
                                            #verificadas em cada um das linhas
  INSERT INTO alteracoes VALUES(OLD.id,NEW.estado);   #acção: inserir o id e o novo valor
                                            #de estado na tabela “alteracoes”
```

Todo o código de implementação da base de dados está disponível em anexo.

5.4 **Princípios de Interacção**

A implementação da aplicação de controlo baseia-se essencialmente em Flash/Actionscript, sendo que a linguagem PHP representa basicamente uma forma de contacto com o exterior, seja com a base de dados seja com os dispositivos. Por esta razão, esta implementação será abordada em duas fases, numa primeira fase uma introdução à comunicação Flash←→PHP e numa segunda fase a implementação de cada um dos pontos fulcrais do sistema: Acesso, Configuração, Interacção e Monitorização. Para que seja possível obter uma melhor percepção será sempre apresentado o algoritmo e principais funções e *scripts* associados (Actionscript + PHP). Em termos de funções em Actionscript, é de salientar o facto de poderem ser distinguidos dois tipos, as funções desencadeadas pelo "click" em botões/*Movie Clips* (locais) e as funções globais.

O código de implementação encontra-se disponível na íntegra em anexo.

5.4.1 **Comunicação Flash←→PHP**

A comunicação da aplicação Flash com PHP é feita por intermédio da classe "LoadVars". Com base na criação de objectos desta classe é possível interagir com o *script* PHP de uma forma relativamente simples. O envio é feito pelo método "POST". A implementação, duma forma genérica, é feita da seguinte forma:

<Actionscript>

```
var enviar_dados:LoadVars = new LoadVars();          //criação dos objectos do tipo "LoadVars" para o
                                                                                    //envio e recepção da informação
var receber_dados:LoadVars = new LoadVars();


        enviar_dados.<var_flash> = <dados_flash>;      // criação da variável <var_flash> para envio de informação

        enviar_dados.sendAndLoad("URL do script PHP ",receber_dados,"POST");      //envio da <var_flash>
                                                                                    //para o script PHP respectivo, através do método POST

                receber_dados.onLoad = function() {                  //executada quando for recebida a informação
                                                                                    //enviada pelo PHP

                        trace(this.<variável enviada pelo script PHP>);          //mostra mensagem "ola do
PHP"

                }
```

<PHP>

```
<?php

$variavel = $_POST['<var_flash>'];          //recepção da variável enviada pela aplicação Flash pelo método "POST"

/*
EXECUÇÃO
*/

echo '&var_php=' . "ola do PHP";                    //envio de dados para a aplicação Flash
                                                    //(formato de envio "&<nome_da_var>=<dados>"
?>
```

5.4.2 **Acesso à Base de Dados**

A base de dados é acedida através de scripts PHP invocados pela aplicação Flash. Estes acessos são baseados nos métodos apresentados anteriormente na comunicação Flash←→PHP.

O acesso à base de dados encontra-se condicionado por *password*, sendo esta definida pelo utilizador na configuração do *Web Server* (Ver Manual).

Na implementação do acesso à base de dados podem ser distinguidas quatro ou três execuções, dependendo se é ou não necessário "passar" variáveis ao *script* PHP. A sequência de execuções é a seguinte:

- Leitura da(s) variável(eis) enviadas pela aplicação Flash (passagem de variáveis via "POST")

<PHP>

```
<?php

$variavel = $_POST['<var_flash>'];          //recepção da variável enviada pela aplicação Flash pelo método "POST"
…
…
```

<continua>

- Leitura do ficheiro que contém a *password* de acesso à base de dados ("mysqlrootpasswd.txt")

<anterior>

```
$fp = fopen("/DOMOTICA/security/mysqlrootpasswd.txt","r");          //leitura do ficheiro que contém
                                                                    //a password definida pelo utilizador na configuração do xampp

        fseek($fp,36)                                               //posiciona o cursor do ficheiro no inicio da password
        fscanf($fp,"%s",$pass);                                     //variável $pass assume o valor da password

fclose($fp);
```

<continua>

- Conexão à base de dados “domotica”

<anterior>

```
mysql_connect("localhost", "root", $pass) or                        //conexão ao servidor
MySQL
    die("Não foi possível conectar: " . mysql_error());             //para efeitos de debug caso não seja possível conectar
                                                                    //à base de dados

mysql_select_db("domotica");                                        //selecção da base de dados “domotica”
```

<continua>

- Pedidos de acesso para leitura ou escrita

<anterior>

```
/*acesso para leitura*/
/*exemplo da leitura da tabela “utilizadores”*/

$dados = '';                                    // Variável responsável por armazenar todos os dados do query
$nLoop = 0;                                     // Variável responsável por armazenar o número total de linhas

$qr = mysql_query("SELECT * FROM utilizadores");        //”query”: ler todos os dados de todos os campos
da
                                                        //tabela utilizadores

        while($n = mysql_fetch_array($qr)) {            //enquanto existirem dados a ler, executa
                $dados .= '&login' . $nLoop . '=' . $n['login'];        //armazenar campo “login” da linha
“$n”
                $dados .= '&pass' . $nLoop . '=' . $n['password'];      //armazenar campo “password”
                                                                        //da linha “$n”
                $dados .= '&nome' . $nLoop . '=' . $n['nome'];          //armazenar campo “nome” da linha “$n”
                $dados .= '&sobrenome' . $nLoop . '=' . $n['sobrenome'];        //armazenar campo “sobrenome”
                                                                                //da linha “$n”
                $dados .= '&mail' . $nLoop . '=' . $n['mail'];          //armazenar campo “mail” da linha “$n”
        $nLoop++;                       //total de linhas lidas (para posterior informação à aplicação Flash)
        }
        echo utf8_encode($dados) . '&nLoop=' . $nLoop;          //envia dados para a aplicação Flash
```

<anterior>

```
/*acesso para escrita*/
/*exemplo da introdução de um utilizador na tabela “utilizadores”*/

$qr = mysql_query("INSERT INTO utilizadores ( id_users, login, password, nome, sobrenome, mail )
                        VALUES ( NULL , '$login', '$pass', '$nome', '$sobrenome', '$mail')");   //registo de novo
                                                                    //utilizador, dados enviados pelo flash(login,pass,nome,sobrenome,mail)

                if(mysql_affected_rows() == 1) {      // teste para confirmação de dados introduzidos com sucesso
                        echo '&sucesso=' . "ok";                              //envio de confirmação à aplicação
Flash
                }

mysql_close();          //desligar conexão ao servidor MySQL
?>
```

No capítulo seguinte serão apresentados os módulos principais constituintes da aplicação de controlo. A configuração de dispositivos IR será abordada apenas no capítulo 7, de maneira a tornar mais fácil a sua percepção dado que nesse capítulo será feita uma introdução teórica aos infravermelhos.

Capítulo VI

6. Módulos Principais

Este capítulo apresenta os módulos principais que constituem o software de controlo. Será apresentado o princípio de execução bem como as principais funções e scripts associados a cada um deles. Para uma melhor percepção, a leitura deste capítulo poderá ser acompanhada pelo manual presente em anexo, com especial atenção ao ponto **Configuração Da Aplicação "DOMÓTICA".**

6.1 Acesso ("acesso.swf")

O acesso ao sistema é condicionado pelo pedido de *login* e *password*. Estes dados encontram-se armazenados na tabela "utilizadores". Na Fig. 40 é possível ver o aspecto gráfico da janela de introdução de dados para o *login* no sistema.

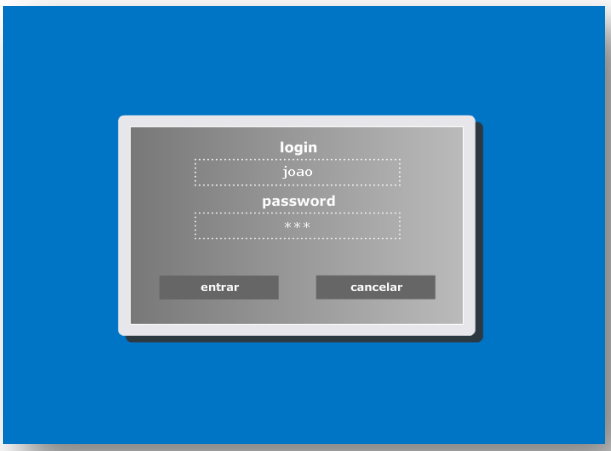


Fig. 40 Acesso ao sistema (*login*)

- **Princípio de Execução**

O primeiro passo na execução do acesso ao sistema será identificar a existência de utilizadores e portanto será feito um acesso para leitura da tabela utilizadores. Consoante a existência ou não de utilizadores será executada a janela de *login* (Fig. 40) ou de registo, respectivamente. Desta forma sempre que a aplicação é executada pela primeira vez será sempre executada a janela de registo do utilizador.

Os botões associados a cada uma das janelas (*login* ou registo) invocarão os *scripts* PHP para verificação de dados (botão "entrar") ou registo de utilizadores (botão "registar").

Após o *login* no sistema será executada uma breve animação de introdução ("intro.swf"), seguida pelo menu principal ("menu.swf").

A maioria das funções são desencadeadas com base em "clicks" feitos pelo utilizador, estas funções serão executadas mais frequentemente do que as globais como se poderá ver no seguimento deste capítulo. Este tipo de funções, designadas locais, são implementadas genericamente da seguinte forma:

<Actionscript>

```
/*botão/Movie Clip */

on(release) {                          //função executada depois de um click  de rato ou pressão de uma tecla
                                                //(neste caso encontra-se associada ao click sobre um
                                              //botão/Movie Clip (a mesma funcionalidade que a anterior)
/*
EXECUÇÃO DA FUNÇÃO
*/
}

/* ou  */

<instância>.onRelease = function() {      //função executada depois do botão/Movie Clip ser premido, instância
                                            //corresponde ao nome associado ao botão/Movie Clip, este tipo de
                                            //interacção apesar de ser equivalente à anterior, é mais frequente
/*                                                                                  //na interacção de Movie Clips
EXECUÇÃO DA FUNÇÃO
*/
}
```

O princípio de execução do módulo aqui apresentado poderá ser visualizado através do fluxograma da Fig. 41. A invocação do *script* PHP através da aplicação Flash é representada pelo processo que contém as duas barras verticais. É também de salientar que no caso dos *scripts* PHP de acesso à base de dados já está implícita a aquisição da *password* e a conexão à base de dados.

Os hexágonos correspondem a execuções iniciadas automaticamente, dentro da aplicação Flash.

- **Principais Funções Actionscript**
  - **Globais**

verificar_users(){}

É iniciada automaticamente no arranque da aplicação "acesso.swf". Tem como objectivo a verificação da existência de utilizadores, determinando assim o tipo de janela a ser executada. Essa verificação é feita através da invocação do *script* "listar.php".

  - **Locais**

Botão entrar → on(release){}

Responsável pela entrada no sistema, para esse fim invoca o *script* "listar.php" de modo a fazer a verificação dos dados inseridos na janela de *login*.

Botão registar → on(release){}

Responsável pelo registo do primeiro utilizador com acesso ao sistema, para esse fim invoca o *script* "salvar_user.php".

Botão cancelar → on(release){}

Sempre que executada permite ao utilizador sair das janelas de *login* ou registo.

- ***Scripts* PHP**

"listar.php"

Apresenta duas funcionalidades. A primeira é verificar o número de utilizadores, a segunda é fazer a leitura dos dados dessa mesma tabela, para que seja possível uma posterior comparação com os dados introduzidos na janela de *login*.

"salvar_user.php"

Apresenta duas funcionalidades. A primeira é verificar se o registo introduzido já se encontra presente na tabela "utilizadores", a segunda é armazenar os dados enviados pela aplicação Flash nessa mesma tabela. Neste caso só será usada a segunda funcionalidade uma vez que a janela de registo só é executada no primeiro acesso à aplicação e como tal ainda não existem utilizadores no sistema. No caso da "configuracao.swf" será explorada também a primeira funcionalidade.

Flash/Actionscript

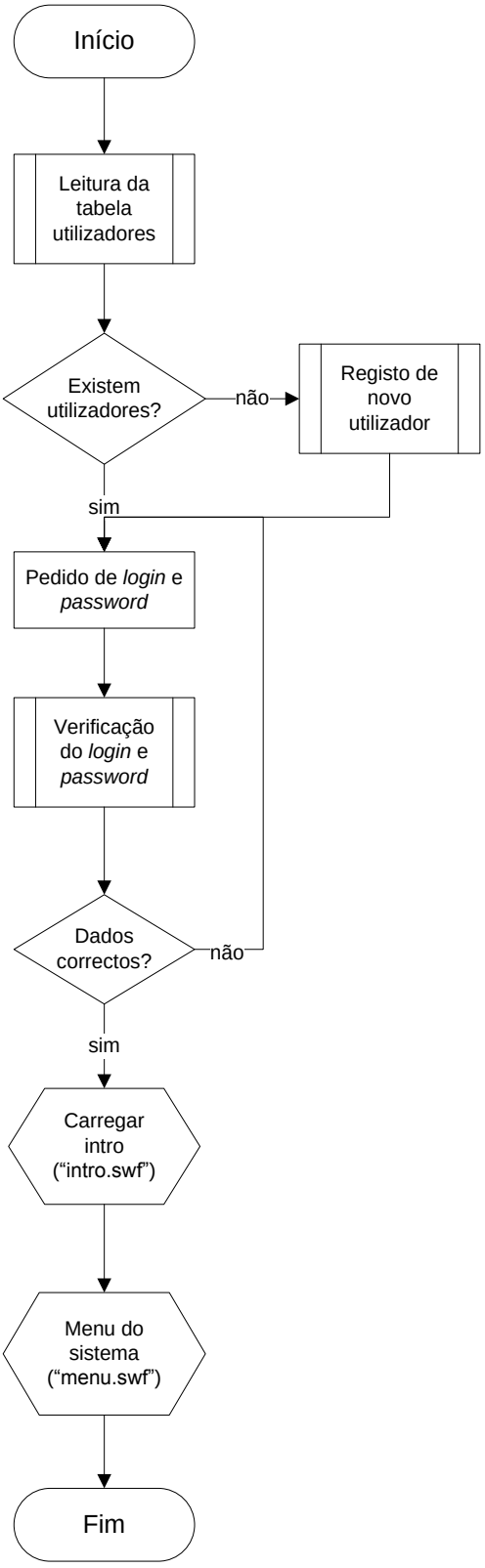


Fig. 41 Acesso ao sistema – Flash/Actionscript

6.2 Configuração (“configuracao.swf”)

A configuração do sistema encontra-se subdividida em quatro partes: gestão de utilizadores, introdução de pisos/compartimentos, introdução de dispositivos e configuração de dispositivos IR. A configuração de dispositivos IR será apresentada apenas no capítulo 7 como já foi dito.

6.2.1 Início do Módulo

O módulo de configuração necessita de fazer a leitura de alguns dados no seu arranque, para verificar a existência dos dados relativos a pisos e compartimentos. Essa leitura é necessária para um posterior acesso à configuração da planta.

A leitura dos dados é feita por intermédio da função global “ler_info()” a qual invocará o *script* “ler_info_pisos.php”. O algoritmo usado na execução deste *script* encontra-se representado pelo fluxograma seguinte.

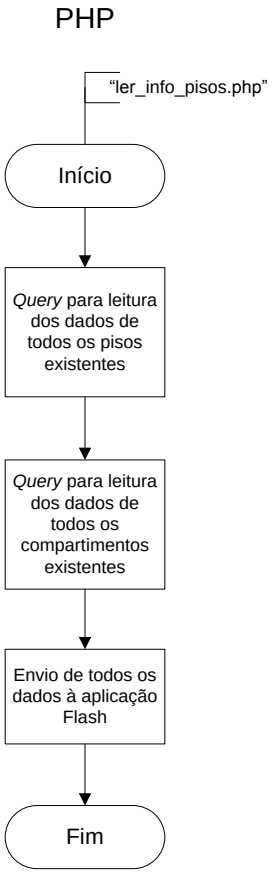


Fig. 42 Arranque do módulo de configuração – *script* PHP

6.2.2 **Gestão de Utilizadores**

A área de gestão de utilizadores permite a modificação dos dados pessoais de todos os utilizadores existentes no sistema, bem como a criação de novos utilizadores. Na Fig. 43 é possível ver o aspecto gráfico desta área.

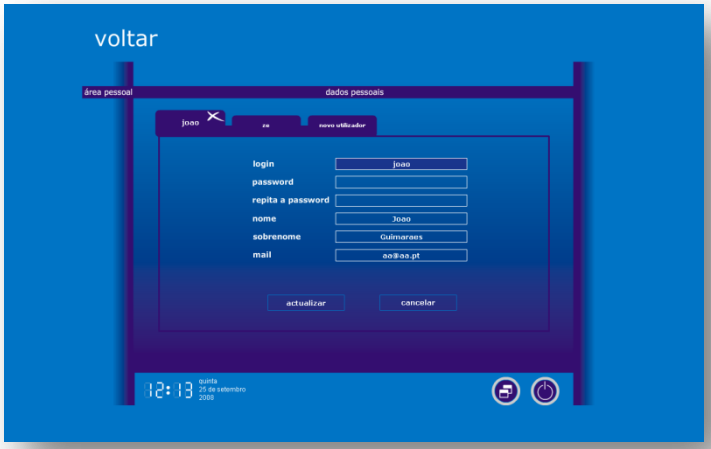


Fig. 43 Gestão de utilizadores

- **Princípio de Execução**

A fim de apurar quais os utilizadores existentes e quais os seus dados, terá de ser feito um acesso à tabela "utilizadores", só então será possível iniciar a apresentação dos utilizadores. Uma vez dentro da área de gestão, consoante a escolha do utilizador serão invocados os *scripts* PHP respectivos (alteração, registo ou eliminação de utilizadores). A eliminação de utilizadores só será possível se existir mais do que um utilizador, garantindo assim sempre a existência de um administrador. Através da análise do fluxograma da Fig. 44 poderá visualizar-se melhor toda esta sequência de execuções consoante as ordens do utilizador.

- **Principais Funções Actionscript**
  - **Globais**

ler_users(login:Array,nome:Array,sobrenome:Array,mail:Array,nr:Array){}

Responsável pela leitura inicial da tabela "utilizadores", invocando o *script* "listar.php", e pela criação de uma espécie de arquivo de utilizadores, com base nos dados lidos. Serão adquiridos todos os dados de todos os utilizadores (a *password* não

será recebida pela aplicação Flash). Os dados são armazenados nas variáveis passadas como argumento.

- **Locais**

Botão utilizador → this.onRelease = function(){}

Associada aos botões de identificação dos utilizadores existentes. Responsável por apresentar os dados do utilizador respectivo, dados esses, lidos no arranque da área de gestão de utilizadores. Este "botão" é tratado como um *Movie Clip* daí a interacção ser apresentada de forma ligeiramente diferente.

Botão registar → on(release){}

Responsável pelo registo de utilizadores, para esse fim invoca o *script* "salvar_user.php".

Botão eliminar → on(release){}

Responsável pela eliminação de utilizadores após um pedido de confirmação, para esse fim invoca o *script* "remover_user.php".

Botão cancelar → on(release){}

Responsável pelo cancelamento de registo/alteração de utilizador, "transportando" o utilizador para o menu anterior.

Botão actualizar → on(release){}

Responsável pela actualização da informação de um determinado utilizador, para esse fim invoca o *script* "alterar_user.php".

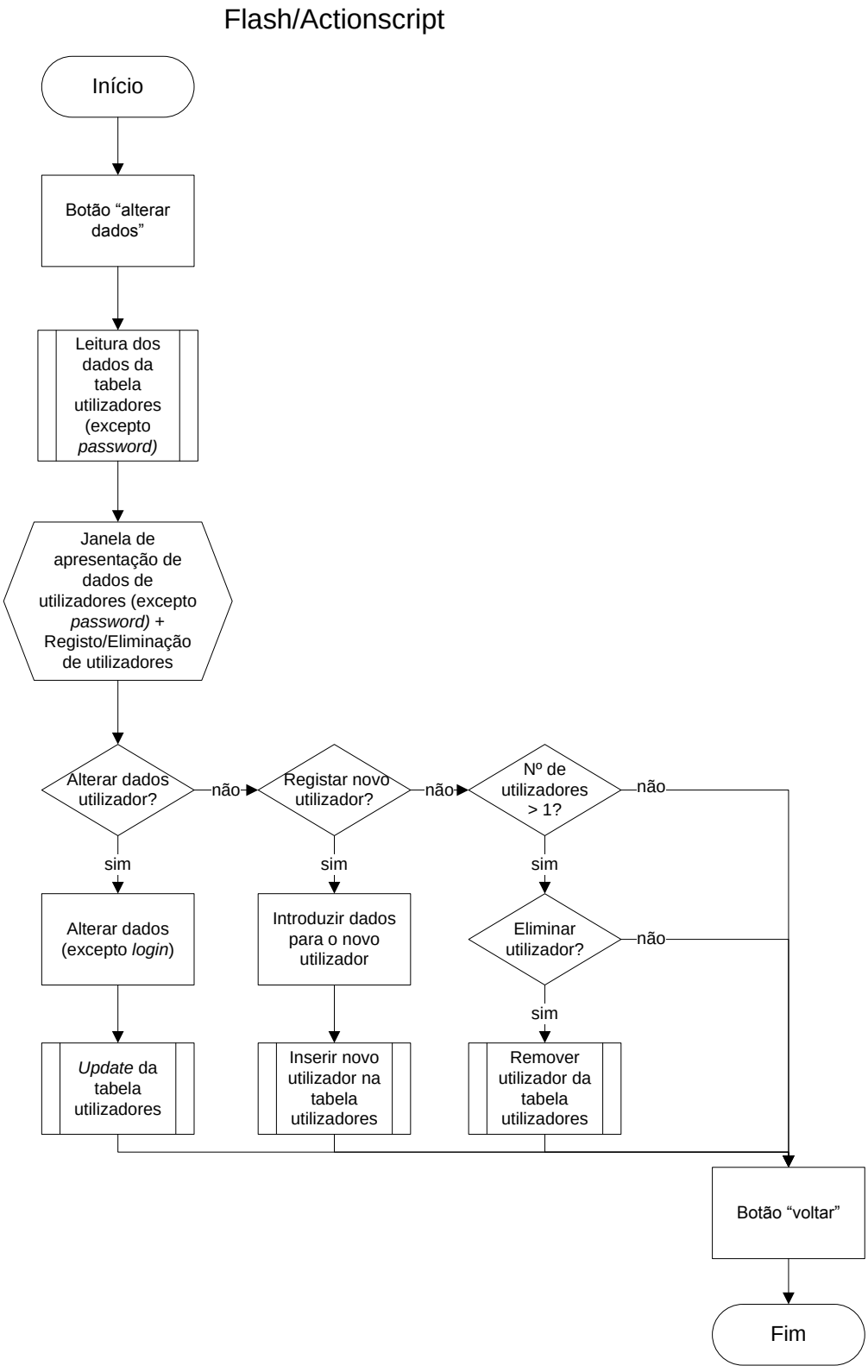


Fig. 44 Gestão de utilizadores – Flash/Actionscript

- ***Scripts* PHP**

listar.php

Script usado anteriormente (“acesso.swf”). Neste caso é usada a segunda funcionalidade, a leitura dos dados, para que seja possível apresentar a informação de cada utilizador existente.

alterar_user.php

Responsável pela actualização dos dados de um determinado utilizador na tabela “utilizadores”.

salvar_user.php

Script usado anteriormente (“acesso.swf”). Neste caso será usada a funcionalidade de verificação de registo existente, seguindo-se a introdução de um novo utilizador na tabela “utilizadores”.

remover_user.php

Responsável pela eliminação de um determinado utilizador da tabela “utilizadores".

6.2.3 Introdução de pisos/compartimentos

A área de introdução de pisos/compartimentos é a base do conceito de interacção da aplicação de controlo. A partir desta área o utilizador poderá introduzir e configurar todos os pisos e respectivamente todos os compartimentos da habitação, no fundo é a partir daqui que o sistema começa a ganhar a sua forma. Na figura seguinte é possível ver o aspecto gráfico desta área.

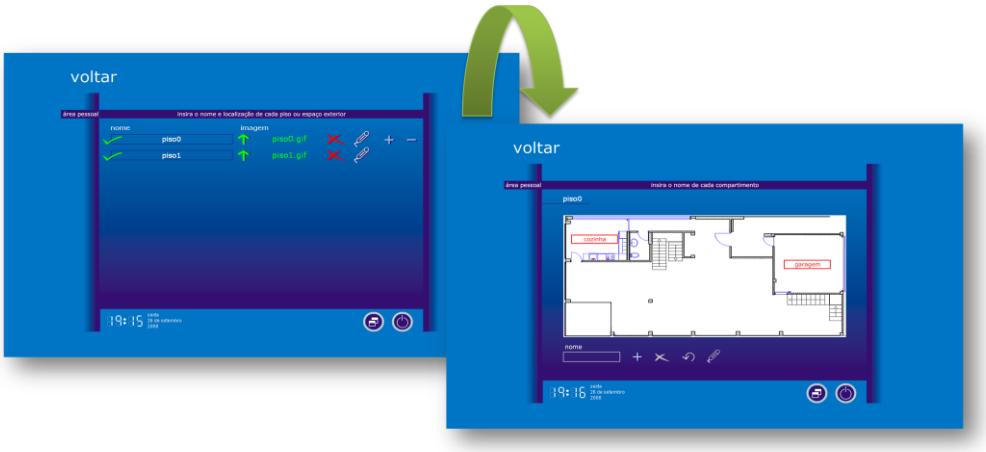


Fig. 45 Introdução/Configuração de pisos/compartimentos

- **Princípio de Execução**

Os dados relativos aos pisos e compartimentos são lidos no arranque deste módulo, como já foi dito em 6.2.1. Consoante a informação lida, esta área terá execuções diferentes. Supondo que é o primeiro acesso ao sistema, o utilizador terá de introduzir todos os pisos e respectivos compartimentos. Cada piso terá um nome e uma imagem associada, a qual será armazenada no servidor. Em relação aos compartimentos, estes também terão um nome associado, para além disso será necessário ainda posicioná-los e delinear a sua área. O posicionamento encontra-se limitado apenas pela área de trabalho (planta do piso).

A área de configuração envolve uma forte presença de *scripts* PHP, desde o *upload* de imagens e gravação/alteração/remoção de pisos até à introdução/alteração/remoção de compartimentos. O fluxograma da Fig. 46 ilustra todo o processo de introdução/configuração existente nesta área.

- **Principais Funções Actionscript**
  - **Globais**

pisos_configurados(){}

Responsável pela execução da apresentação dos pisos existentes no sistema. Baseado na informação lida no arranque do módulo.

inicializar_piso(){}

Responsável pelo carregamento da imagem associada ao piso, bem como de todos os compartimentos nele existentes. Baseado na informação lida no arranque do módulo.

guardar_piso(old_nomeP:String, piso_nome:String, dir_piso:String, actualizou:Number){}

Responsável pelo armazenamento/actualização de todos os dados do piso introduzido. O argumento "old_nomeP" é usado na identificação dos dados antigos relativos ao piso, presentes na tabela "pisos", de modo a possibilitar uma actualização sempre que pretendido. Invoca o *script* "guardar_pisos.php" para fazer o armazenamento ou actualização na tabela "pisos".

func_mouse(nome_comp:String, x_clip:Number, y_clip:Number, clip_angulo:Number){}

Responsável pela delineação da área do compartimento e posterior gravação/actualização dos dados relativos ao compartimento. O armazenamento na base de dados é feito invocando o *script* "guardar_alterar_comp.php".

- **Locais**

- Botões presentes na barra de introdução de pisos

Botão upload → on(release){}

Responsável pela selecção da imagem da planta de cada piso para posterior *upload*. O *upload* é feito por intermédio do *script* "upload.php".

Botão eliminar → on(release){}

Responsável pela eliminação do piso respectivo. Após uma confirmação para eliminação é invocado o *script* "eliminar_pisos.php".

Botão configurar → on(release){}

Responsável pelo acesso à configuração do piso bem como pelo armazenamento dos seus dados (nome e imagem). O armazenamento é feito através da execução da função "guardar_piso".

- Botões presentes na barra de introdução de compartimentos

Botão adicionar → on(release){}

Responsável pela introdução do compartimento na área de trabalho.

Botão eliminar → on(release){}

Responsável pela eliminação de um compartimento previamente seleccionado. Para esse fim é invocado o *script* "eliminar_compart.php" (não existe confirmação para eliminação).

Botão delinear área → on(release){}

Responsável pela execução da função "func_mouse" para a delineação da área do compartimento e armazenamento dos seus dados.

Botão rodar compartimento → on(release){}

Responsável pela rotação do compartimento seleccionado de forma a tornar possível um posicionamento mais adequado na área de trabalho.

- Botão compartimento

this.onPress = function(){}

Responsável pela selecção do compartimento e início do seu arrastamento para posicionamento.

this.onRelease = function(){}

Responsável pela paragem do arrastamento do compartimento.

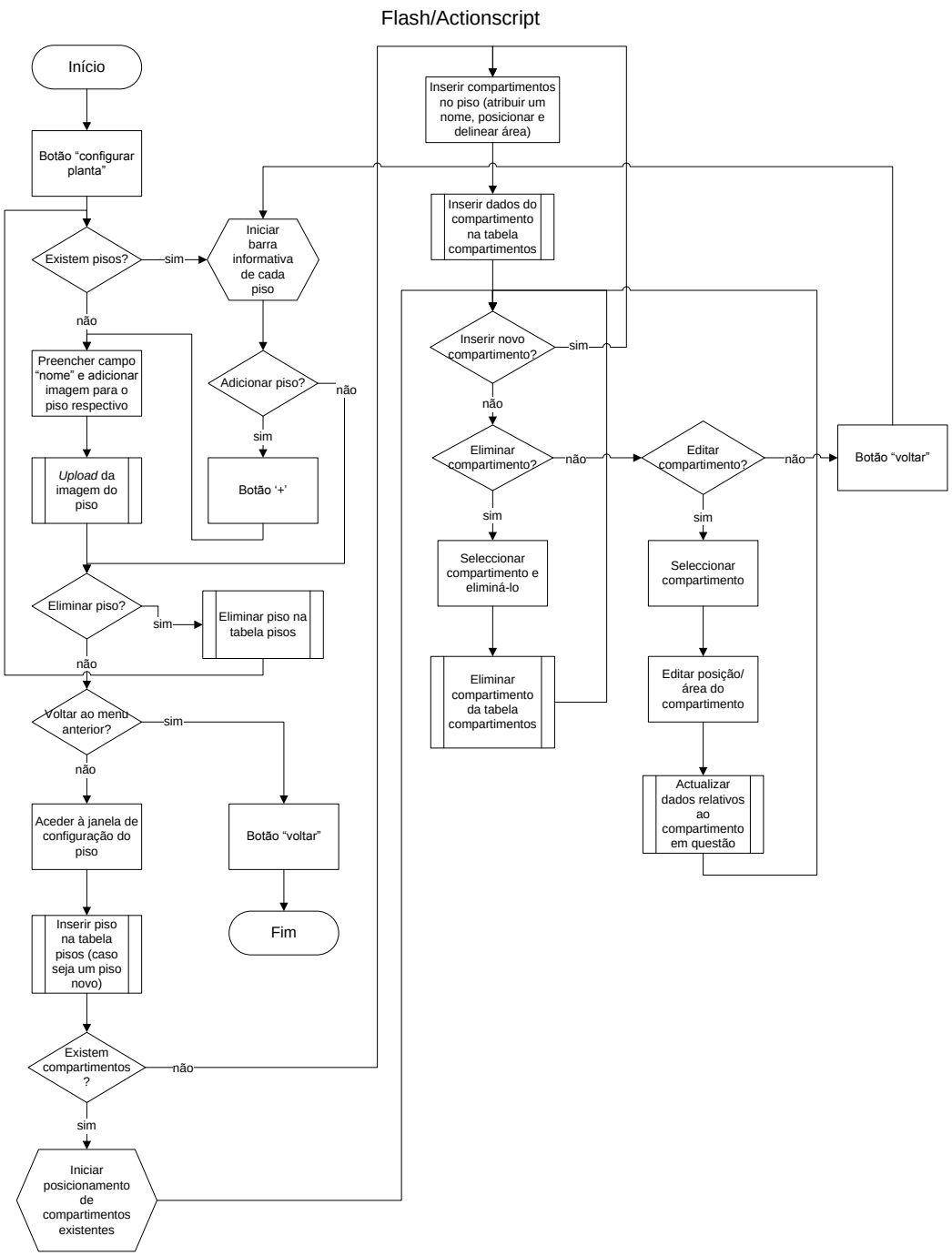


Fig. 46 Introdução/Configuração de pisos/compartimentos – Flash/Actionscript

- ***Scripts* PHP**

upload.php

Responsável pelo *upload* no servidor, da imagem da planta do piso seleccionada pelo utilizador.

guardar_pisos.php

Responsável pelo armazenamento dos dados do piso introduzido. Armazenamento na tabela "pisos".

eliminar_pisos.php

Responsável pela eliminação de um determinado piso na tabela "pisos".

guardar_alterar_comp.php

Responsável pela introdução e actualização dos dados de cada compartimento na tabela "compartimentos".

eliminar_compart.php

Responsável pela eliminação de um determinado compartimento na tabela "compartimentos".

6.2.4 Introdução de Dispositivos

A área de introdução/configuração de dispositivos permite ao utilizador configurar todos os dispositivos existentes num determinado compartimento. Nesta área o utilizador poderá posicionar livremente qualquer dispositivo bem como atribuir o endereço associado a cada um. Os dispositivos IR são um caso pontual pois para além do endereço é também necessário introduzir um nome. Na figura seguinte é possível ver o aspecto gráfico desta área.

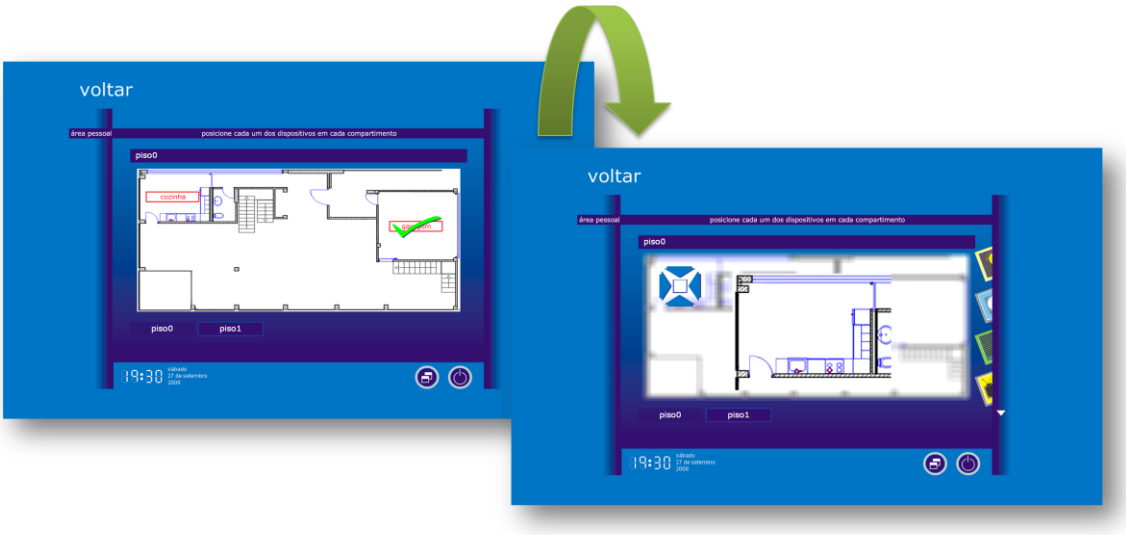


Fig. 47 Introdução/Configuração de dispositivos

- **Princípio de Execução**

A introdução de dispositivos é feita individualmente para casa piso e respectivamente para cada um dos compartimentos. O acesso ao piso deverá ser antecedido de uma verificação a fim de identificar os compartimentos que já se encontram configurados. Essa informação será adquirida através da leitura da tabela "dispositivos", associando o piso e os compartimentos existentes.

A leitura da informação dos dispositivos (tipo, posição e clip) será efectuada localmente para um determinado compartimento sempre que se aceda a esse compartimento. A cada introdução de dispositivos terão de ser introduzidos os respectivos endereços físico e de memória, para além do devido posicionamento na área do compartimento. No caso dos dispositivos IR só será introduzido um endereço, adicionalmente será também necessário associar um nome único ao dispositivo, para posterior configuração.

Os *scripts* PHP envolvidos são essencialmente usados na manipulação dos dispositivos (gravação/eliminação/actualização/leitura) à excepção do *script* que faz a leitura de compartimentos já configurados. O fluxograma da Fig. 48 ilustra todo este processo de introdução/configuração de dispositivos.

- **Principais Funções Actionscript**
  - **Globais**

listar_pisos_pos(){}

Responsável pela listagem dos pisos inseridos. Baseado na informação lida no arranque do módulo.

manipular_imagem(destino:MovieClip, imagem:String, filtro:Boolean){}

Responsável por aplicar/remover o efeito de desfocagem à imagem da planta (efeito *blur*). Usado na execução do *zoom* de compartimentos. Funciona em conjunto com a função "isolar".

ler_bd(piso:String, compartimento:String, tipo:Array, X:Array, Y:Array, nr:Array, clip:Array){}

Responsável pela leitura da informação dos dispositivos existentes num determinado compartimento. Os dados relevantes são armazenados nas variáveis passadas como argumento, à excepção das variáveis "piso" e "compartimento" que são determinantes na selecção do compartimento em questão.

disp_lidos(){}

Responsável pela apresentação, na área do compartimento, de todos os dispositivos existentes. Funciona em conjunto com a função "isolar".

f_compartimento(clips:Number,piso:Number){}

Responsável por adicionar à área de trabalho todos os compartimentos existentes. Funciona em conjunto com a função "isolar".

isolar(destino:MovieClip, mascara:MovieClip, imagem, Xinicial:Number, Yinicial:Number, ampliar:Number, centro_x:Number, centro_y:Number, zoom_in:Number){}

Responsável por todos os passos de execução do *zoom in*/*out* de compartimentos, executando diversas funções auxiliares. Desde manipulação da imagem (função "manipular_imagem"), isolamento do compartimento, apresentação dos dispositivos (função "disp_lidos") e todo o processo inverso (*zoom out*) até ao estado de apresentação inicial dos compartimentos (função "f_compartimento").

- **Locais**

Botão piso → on(release){}

Responsável pelo carregamento de todos os dados associados aos seus compartimentos, incluindo a verificação dos compartimentos que já contêm dispositivos, invocando o *script* "ler_comp.php".

Botão adicionar → on(release){}

Responsável pela introdução do dispositivo seleccionado na área do compartimento.

- Botão dispositivo (Semelhante à função do botão compartimento – 6.2.3)

this.onPress = function(){}

Responsável pela selecção do dispositivo e início do seu arrastamento para posicionamento.

this.onRelease = function(){}

Responsável pela paragem do arrastamento do dispositivo.

Botão compartimento → on(release){}

Responsável pela execução da função "isolar" de maneira a iniciar o *zoom in* do compartimento.

Botão *zoom out* → on(release){}

Responsável pela execução da função "isolar" de maneira a iniciar o *zoom out*, conduzindo o utilizador à planta principal.

Botão gravar → on(release){}

Responsável pela confirmação dos parâmetros introduzidos para o dispositivo e posterior invocação do *script* "guardar_disp.php" para gravação dos seus dados. Os dados de posicionamento serão temporariamente armazenados como "NULL".

Botão cancelar → on(release){}

Responsável pelo cancelamento da introdução de parâmetros do dispositivo e consequente eliminação desse dispositivo da área de trabalho.

Botão confirmar → on(release){}

Responsável pela invocação do *script* "modificar_disp.php" a fim de guardar a posição final de todos os dispositivos existentes no compartimento.

Botão eliminar → on(release){}

Responsável pela eliminação do dispositivo seleccionado, invocando assim o *script* "eliminar_disp.php".

Botão piso rodapé → on(release){}

Responsável pelo alternar entre pisos durante a introdução de dispositivos.

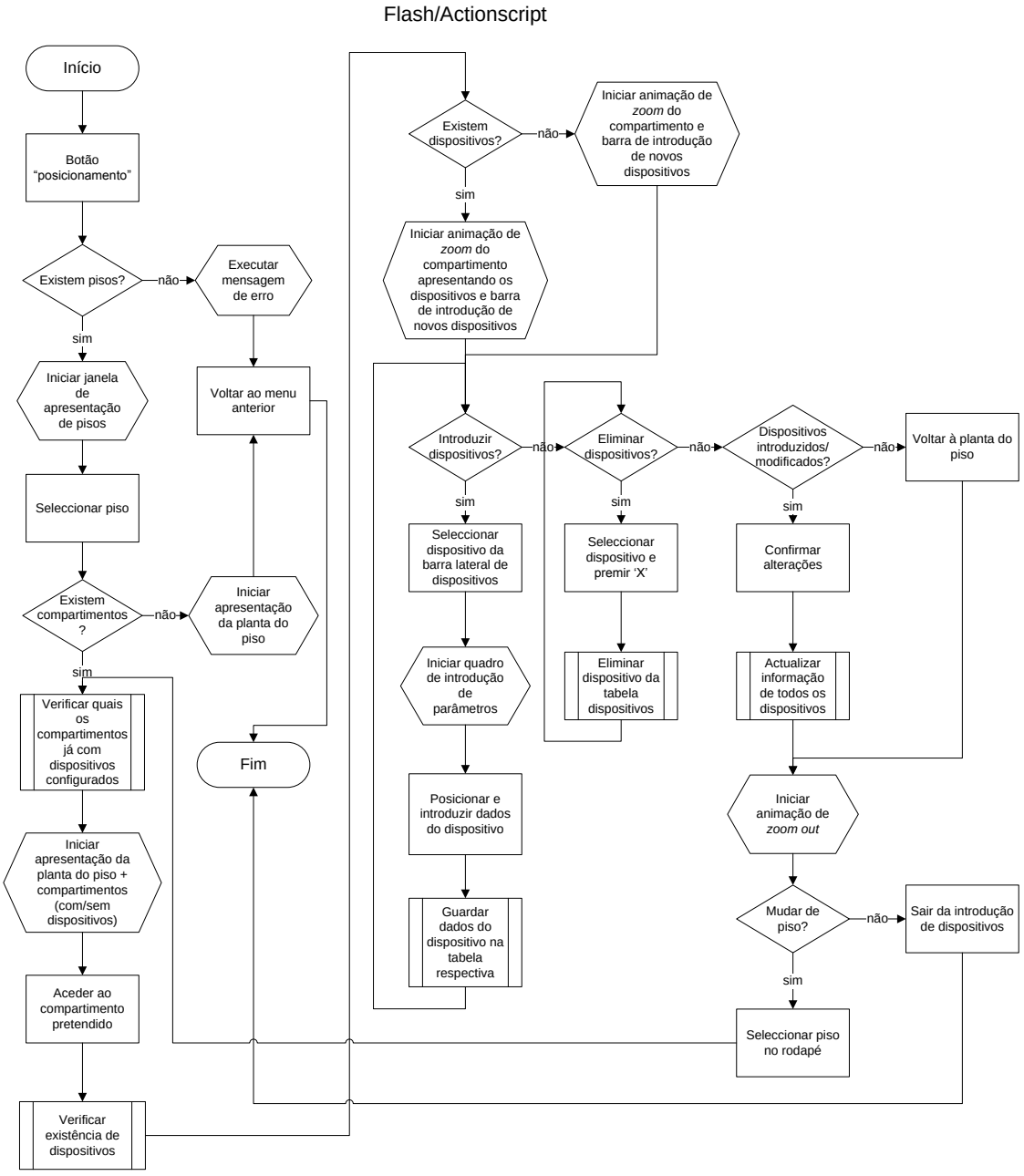


Fig. 48 Introdução/Configuração de dispositivos – Flash/Actionscript

- ***Scripts* PHP**

ler_comp.php

Responsável pela leitura dos compartimentos presentes na tabela "dispositivos", para que seja possível identificar os compartimentos que contêm dispositivos.

ler_disp.php

Este *script* apresenta duas funcionalidades. A primeira é a leitura de todos os dados relativos aos dispositivos existentes num determinado piso e respectivo compartimento, incluindo o seu estado. A segunda é a identificação do tipo de dispositivos existentes, funcionalidade que será explorada no módulo "sistema.swf".

guardar_disp.php

Responsável pelo armazenamento dos dados do dispositivo na tabela respectiva, uma vez que será tido em conta o tipo de dispositivo, já que os estores e dispositivos IR, dadas as suas características, para além da tabela "dispositivos" encontram-se armazenados também na tabela "estores" e "dispositivos_IR", respectivamente. Os Campos de posição, x e y (tabela "dispositivos"), nesta fase serão preenchidos com "NULL".

modificar_disp.php

Responsável pela actualização dos campos de posição do dispositivo. No caso de dispositivos introduzidos recentemente o primeiro acesso será sempre para substituição do campo "NULL" pelo valor respectivo.

eliminar_disp.php

Responsável pela eliminação de um determinado dispositivo da tabela "dispositivos" e leitura dessa mesma tabela para renovação da informação presente na aplicação Flash.

6.3 Interacção com o Sistema (“sistema.swf”)

A área de controlo de dispositivos é a área na qual o utilizador poderá actuar em todos os dispositivos introduzidos. A actuação no dispositivo escolhido irá reflectir-se em tramas *Ethernet* ou tramas IR, consoante o tipo de dispositivo a actuar. O interface é muito semelhante ao do módulo “configuracao.swf” e como tal grande parte das funções anteriores foram “aproveitadas”, essencialmente as funções de *zoom* e acesso aos pisos. A abordagem deste módulo será um pouco diferente no sentido de ser mais focado todo um processo de interacção com o hardware, tendo em conta que na secção anterior foi já apresentada a lógica principal de execução a nível da aplicação Flash. Na figura seguinte é possível visualizar o aspecto gráfico desta área de controlo.

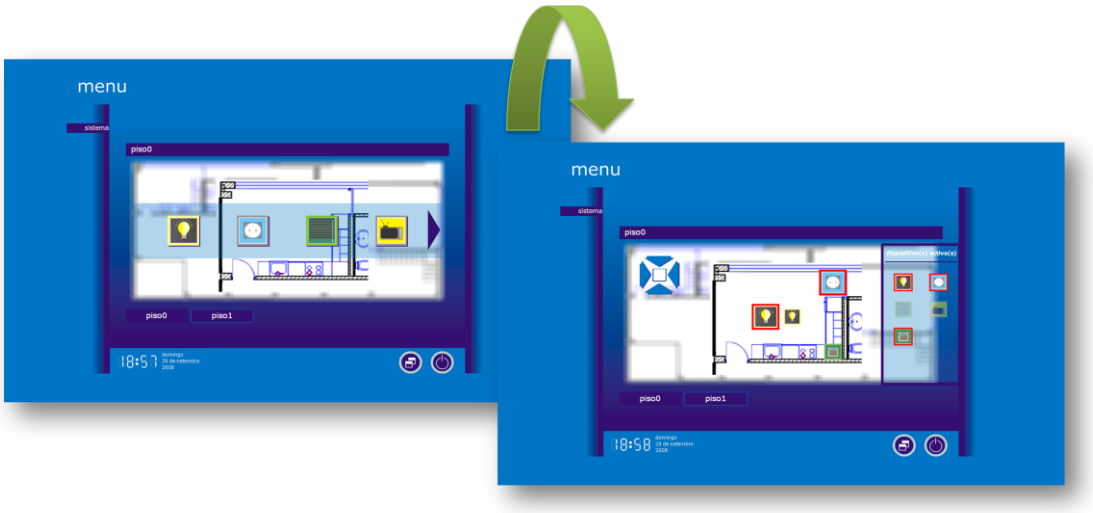


Fig. 49 Controlo de dispositivos

- **Princípio de Execução**

O conceito de interacção sendo semelhante ao do módulo de configuração, significa que o processo continua a ser baseado num primeiro acesso ao piso e só depois um acesso ao compartimento. Continua portanto a ser necessária a leitura inicial de todos os compartimentos e pisos existentes. A novidade reside numa janela de pré-selecção do tipo de dispositivo a controlar, permitindo uma diminuição da quantidade de dispositivos que aparecem inicialmente na área de trabalho.

O “click” sobre o dispositivo ou sobre o seu painel de controlo traduz-se em duas execuções distintas, a actuação no hardware respectivo e a alteração do seu estado na tabela “monitor”. Os dispositivos IR não sofrem alteração de estado na tabela monitor.

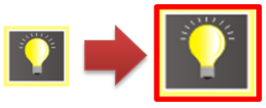
O módulo permite não só o controlo de dispositivos mas também a sua monitorização, à excepção dos dispositivos IR, como já se sabe.

6.3.1 Controlo

- **Principais Funções Actionscript**

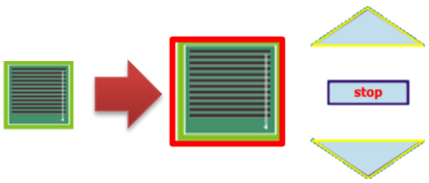
No controlo dos dispositivos são distinguidas três categorias: controlo de dispositivos ON/OFF genéricos, estores e dispositivos IR. A interacção com o hardware e base de dados será diferente em cada uma destas categorias, como tal, foram implementados três tipos de funções. O fluxograma da Fig. 50 ilustra o processo de controlo para um determinado dispositivo.

- **Globais**



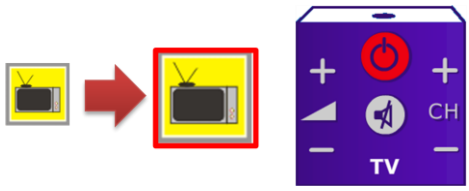
guardar_estado(piso:String, compartimento:String, clip:String, estado:String){}

Responsável pela invocação do *script* "guardar_estado.php" para a actuação no endereço respectivo do PLC e posterior actualização do estado do dispositivo na tabela "monitor". A opção passada no argumento "estado" irá traduzir-se no SET/RESET do respectivo endereço. Os restantes argumentos identificam univocamente um dispositivo na tabela "dispositivos".



comando_estore(piso:String, compartimento:String, clip:String, estado:Number, endereco:Number, acesso:String, estore_id:Number){}

Responsável pela invocação do *script* "on_off_estore.php" para a manipulação dos estores. Com base no argumento "acesso" é possível definir o tipo de acesso ao *script*.



enviar_comandoIR(comando:String, clip:Number, piso:String, compartimento:String){}

Responsável pela invocação do *script* "enviar_IR.php" para o controlo dos dispositivos IR. O argumento "comando" corresponde ao comando IR a enviar, cujos parâmetros serão posteriormente lidos da base de dados.

- **Locais**

Botão dispositivo → this.onRelease = function() {}

Responsável pela animação gráfica de activação/desactivação do dispositivo e consequentemente a execução das funções respectivas de acordo com o tipo de dispositivo a controlar. Para além disso sempre que se aplique, adiciona ao dispositivo o respectivo painel de controlo (estores e televisão).

Botão de selecção → this.onRelease = function() {}

Responsável pela selecção dos dispositivos visíveis na área de trabalho.

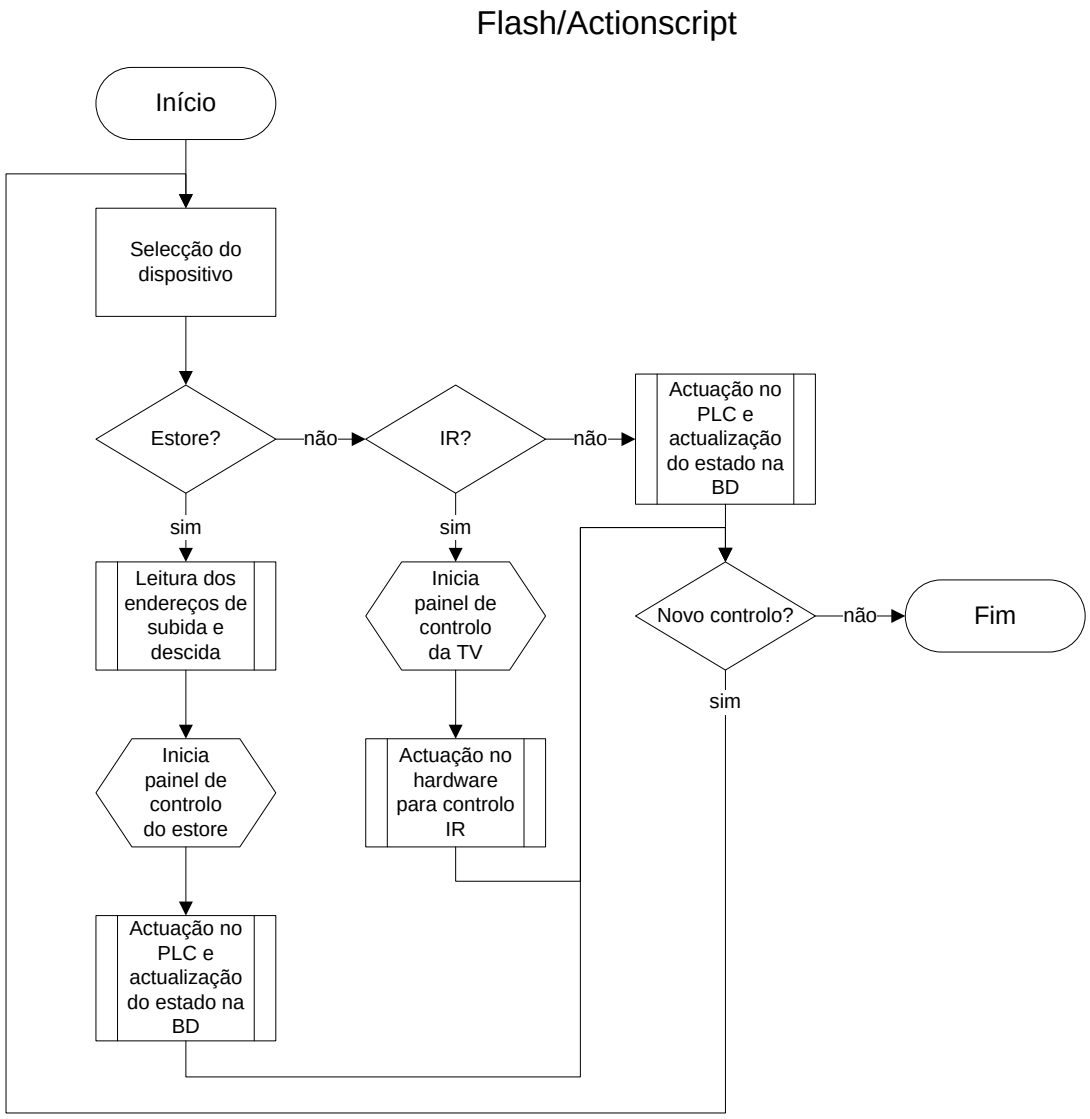


Fig. 50 Controlo de dispositivos – Flash/Actionscript

- ***Scripts* PHP**

guardar_estado.php

Responsável pela criação e envio da trama *Ethernet* para a actuação no *bit* de memória do PLC respectivo, e também pela actualização do estado do dispositivo na tabela "monitor".

on_off_estore.php

Este *script* apresenta duas funcionalidades. A primeira é a leitura de todos os dados relevantes ao controlo do estore. A segunda é equivalente à funcionalidade do *script* "guardar_estado.php". As funcionalidades são seleccionadas consoante o

argumento “acesso” da função “comando_estore”. Tendo em conta as características do controlo de estores foi criada a primeira funcionalidade, desta forma será feito um único acesso para leitura dos endereços do estore, evitando assim uma leitura individual para cada uma das execuções (subida e descida).

enviar_IR.php

Responsável pela leitura da tabela “trama_IR” de forma a adquirir todos os parâmetros necessários para o envio do comando IR em questão. Este *script* será abordado novamente no capítulo 8, quando for apresentado em maior pormenor todo o processo de controlo de dispositivos IR.

6.3.2 **Monitorização**

O estado de um dispositivo é alterado através de um “click” sobre o mesmo, e como se sabe irá ser executada uma mudança gráfica indicando o seu estado. O mesmo deverá acontecer sempre que se actue no dispositivo através de um interruptor local (simulado através das teclas do painel do PLC), e como tal terá de ser dado o mesmo *feedback* ao utilizador. Para que isto aconteça a aplicação terá de monitorizar periodicamente o estado dos dispositivos, esta monitorização será baseada na leitura da tabela “alteracoes”, a qual contém a identificação e o estado dos dispositivos recentemente controlados. Este processo é efectuado através da execução da função “ler_estado”, a qual é executada automaticamente de 2 em 2 segundos. Na Fig. 51 é possível visualizar a sequência de monitorização levada a cabo pela aplicação.

A monitorização só é possível graças ao módulo “monitor.swf” que é executado em *background*. Este módulo verifica periodicamente o estado dos dispositivos conectados ao PLC actualizando a tabela “monitor”. O módulo será apresentado em seguida na secção 6.4.

- **Principais Funções Actionscript**
  - **Globais**

ler_estado(){}

Responsável pela invocação do *script* “monitorizar.php” para actualização periódica do estado dos dispositivos na aplicação.

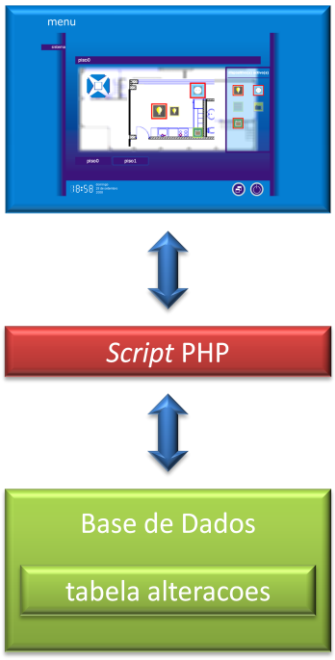


Fig. 51 Monitorização no controlo de dispositivos

- ***Scripts* PHP**

monitorizar.php

Responsável pela leitura de todo o conteúdo da tabela "alteracoes" e posterior envio da informação à aplicação Flash.

6.4 Monitorização ("monitor.swf")

O módulo de monitorização é iniciado no arranque da aplicação de controlo ficando posteriormente a ser executado em *background*. O objectivo é monitorizar periodicamente o estado dos *bits* de memória do PLC associados aos dispositivos existentes no sistema, e actualizar esse estado na tabela "monitor". Com base neste processo é possível obter um *feedback* do estado de todos os dispositivos (excepto dispositivos IR) mesmo quando a actuação não é feita via aplicação local mas sim através de interruptores. O número de *bits* para leitura será definido pelo utilizador no arranque do módulo, ficando posteriormente registado num ficheiro de texto. Na figura seguinte é possível ver o aspecto gráfico da janela "Monitor" onde a título de exemplo será feita a monitorização de dez *Memory Bits* (0-9). A leitura será feita com intervalos de 3 segundos.

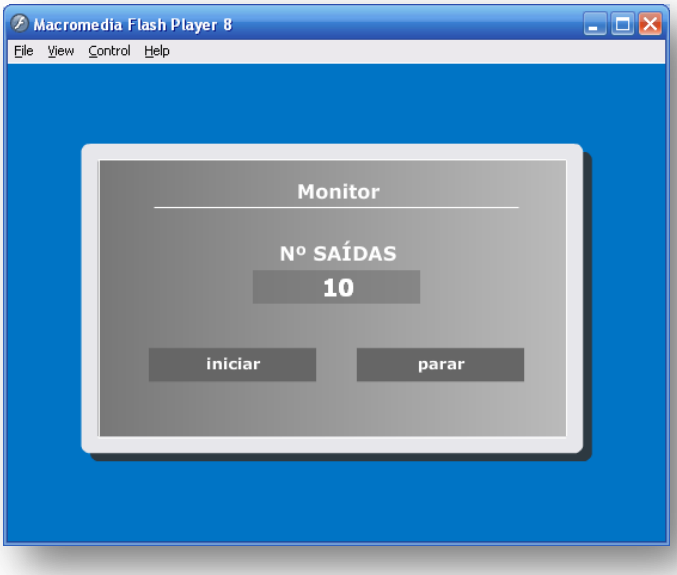


Fig. 52 Janela "Monitor"

- **Princípio de Execução**

O acesso à aplicação principal será antecedido pela janela "Monitor" a fim de se fazer a introdução do número de *bits* a monitorizar e posteriormente se iniciar o processo de monitorização. O botão "iniciar" dará inicio à monitorização e execução do módulo de acesso ao sistema ("acesso.swf").

Em termos de implementação, a sequência será basicamente a seguinte:

- Aquisição do número de *Memory bits* para leitura
- Criação e envio da trama *Ethernet* para a leitura dos *Memory Bits*
- Interpretação da informação proveniente do PLC isolando os dados relevantes (campo de valores)
- Correspondência de cada endereço com o "id" respectivo
- Tradução dos valores em estados, '0' (DESLIGADO) e '1' (LIGADO)
- Armazenamento temporário do "id" e respectivo estado
- Actualização da tabela "monitor" com os dados armazenados temporariamente
- Limpeza da informação da tabela "alteracoes" ao fim de 1 minuto, com vista a impedir o armazenamento excessivo de informação

A Fig. 53 ilustra todo o processo de interacções para o apuramento e armazenamento do estado de cada dispositivo conectado ao PLC.

- **Principais Funções Actionscript**
  - **Globais**

ler_estado(){}

Responsável pela invocação do *script* "ler_registos.php" a fim de se iniciar todo o processo de monitorização. Esta função é executada automaticamente de 3 em 3 segundos. Como se pode verificar o intervalo de leitura para monitorização usado no módulo "sistema.swf" é ligeiramente diferente, se o intervalo fosse igual poderia correr-se o risco de ler informação desactualizada, uma vez que a tabela "alteracoes" poderia ainda não ter sido actualizada, pois todo o processo que leva à actualização da tabela "alteracoes" é ligeiramente mais moroso do que o acesso levado a cabo pelo *script* "monitorizar.php".

  - **Locais**

Botão "iniciar" → on(release){}

Responsável pelo inicio do ciclo de execução da função "ler_estado".

Botão "parar" → on(release){}

Responsável pelo cancelamento da monitorização.

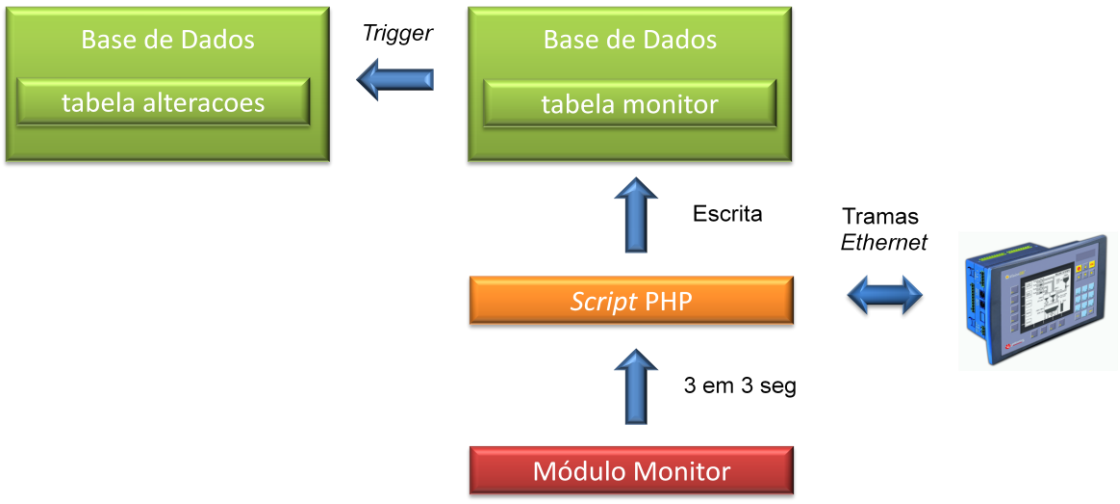


Fig. 53 Processo de monitorização do sistema

- ***Scripts* PHP**

ler_registos.php

Responsável por toda a interacção, seja com o PLC seja com a base de dados. A sequência apresentada no início da explicação deste módulo é implementada por este *script*.

6.5 Configuração de Hardware ("hardware.swf")

A área denominada, configuração de hardware, destina-se de certa forma à listagem de toda a informação relevante relativa aos dispositivos existentes no sistema, tornando possível, de uma forma simples, a alteração do endereço/eliminação de dispositivos. No caso dos dispositivos IR a informação relativa ao endereço não pode ser alterada, neste caso não existe interesse numa alteração porque o máximo que pode acontecer é uma mudança de aparelho IR num determinado compartimento, logo, será necessária uma nova configuração IR e não propriamente uma alteração de endereço. No caso dos dispositivos conectados ao PLC as coisas são diferentes, no sentido que poderá ser necessário fazer uma alteração no *Ladder*, o que poderá implicar a modificação de algum endereço, e portanto, o endereço que antes estava associado a uma tomada agora poderia estar associado a uma lâmpada, por exemplo.

A comunicação com o subsistema IR também poderá ser aqui configurada, nomeadamente a alteração da porta COM a utilizar.

Na figura seguinte encontra-se ilustrada esta área de configuração de hardware.

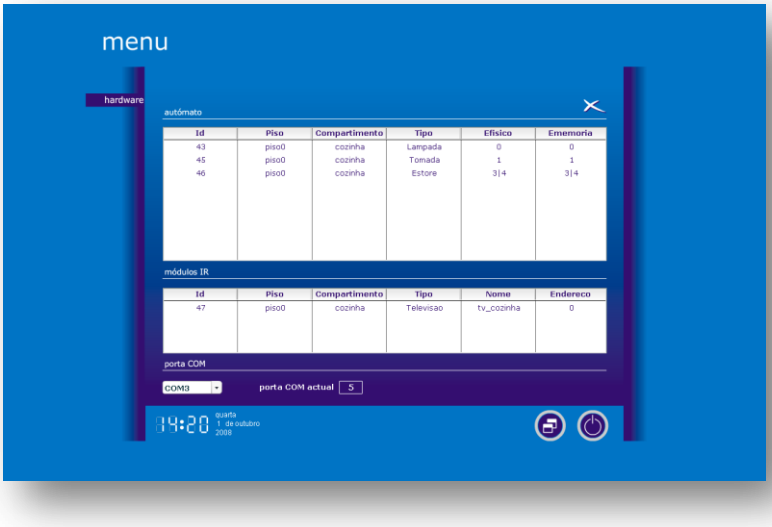


Fig. 54 Configuração de hardware

- **Princípio de Execução**

A listagem de informação é feita após a leitura de toda a informação existente na base de dados, relativa aos dispositivos. Esta listagem será executada por intermédio de uma *DataGrid*, a qual permitirá a posterior manipulação de cada um dos dispositivos.

A nível de configuração da comunicação com o subsistema IR terá de ser obtida informação acerca da(s) porta(s) COM existente(s) no PC, para esse fim é usada uma aplicação C/C++ (invocada a partir de um *script* PHP), que inclusive apresenta outras funcionalidades, nomeadamente na interacção com o módulo de aquisição do subsistema IR, como se poderá ver no capítulo 7. A informação da(s) porta(s) COM é apresentada numa *ComboBox*. O fluxograma da Fig. 55 ilustra toda o processo de interacção levado a cabo pelo módulo "hardware.swf" no sentido de efectuar a operação pretendida pelo utilizador.

- **Principais Funções Actionscript**
  - **Globais**

ler_COM(){}

Responsável pela leitura da(s) porta(s) COM existente(s) no PC e posterior apresentação na *ComboBox*. A porta em utilização é igualmente lida. Para esse fim é invocado o *script* "com.php".

ler_hardware(){}

Função executada automaticamente no arranque do módulo. Numa primeira fase, é responsável pela leitura dos dados relevantes de todos os dispositivos existentes no sistema, para posterior apresentação na *DataGrid*, numa segunda fase é responsável pela execução da função "ler_COM". Na leitura dos dados, é invocado o *script* "ler_hardware.php".

update_hardware(id:Number, endereco:String, tipo:String, coluna:Number){}

Responsável pela actualização do endereço do dispositivo, célula seleccionada pelo utilizador na *DataGrid*. Esta função é iniciada a partir do momento em que a célula deixa de estar seleccionada. A actualização é feita através da invocação do *script* "update_hardware.php".

eliminar_hardware(id:Number){}

Responsável pela eliminação de um determinado dispositivo, para esse fim invoca o *script* "eliminar_hardware.php".

gravar_COM(com:Number){}

Responsável pela actualização da porta COM. Sempre que exista uma alteração na *ComboBox* esta função invoca o *script* "guardar_com.php".

- **Locais**

- Manipulação da *DataGrid*

editar_listener.cellEdit = function(evt_obj:Object){}

Responsável pela alteração dos campos de endereço dos dispositivos conectados ao PLC. Executa a função "update_hardware".

clicar_listener1.cellPress = function(evt_obj:Object){}

Responsável pela selecção de elementos da *DataGrid* onde se encontram os dispositivos conectados ao PLC.

clicar_listener2.cellPress = function(evt_obj:Object){}

Responsável pela selecção de elementos da *DataGrid* onde se encontram os dispositivos IR. A selecção dos dispositivos IR e não IR será identificada por cores diferentes, daí a existência de funções diferentes.

- Manipulação da *ComboBox*

cbListener.change = function(evt_obj:Object){}

Responsável pela execução da função "gravar_COM", sempre que existe a selecção de um valor na *ComboBox*, de forma a ser feita a actualização da porta COM.

Botão eliminar → on(release){}

Responsável pela eliminação, após confirmação, do dispositivo previamente seleccionado da *DataGrid*. Para esse fim é executada a função "eliminar_hardware".

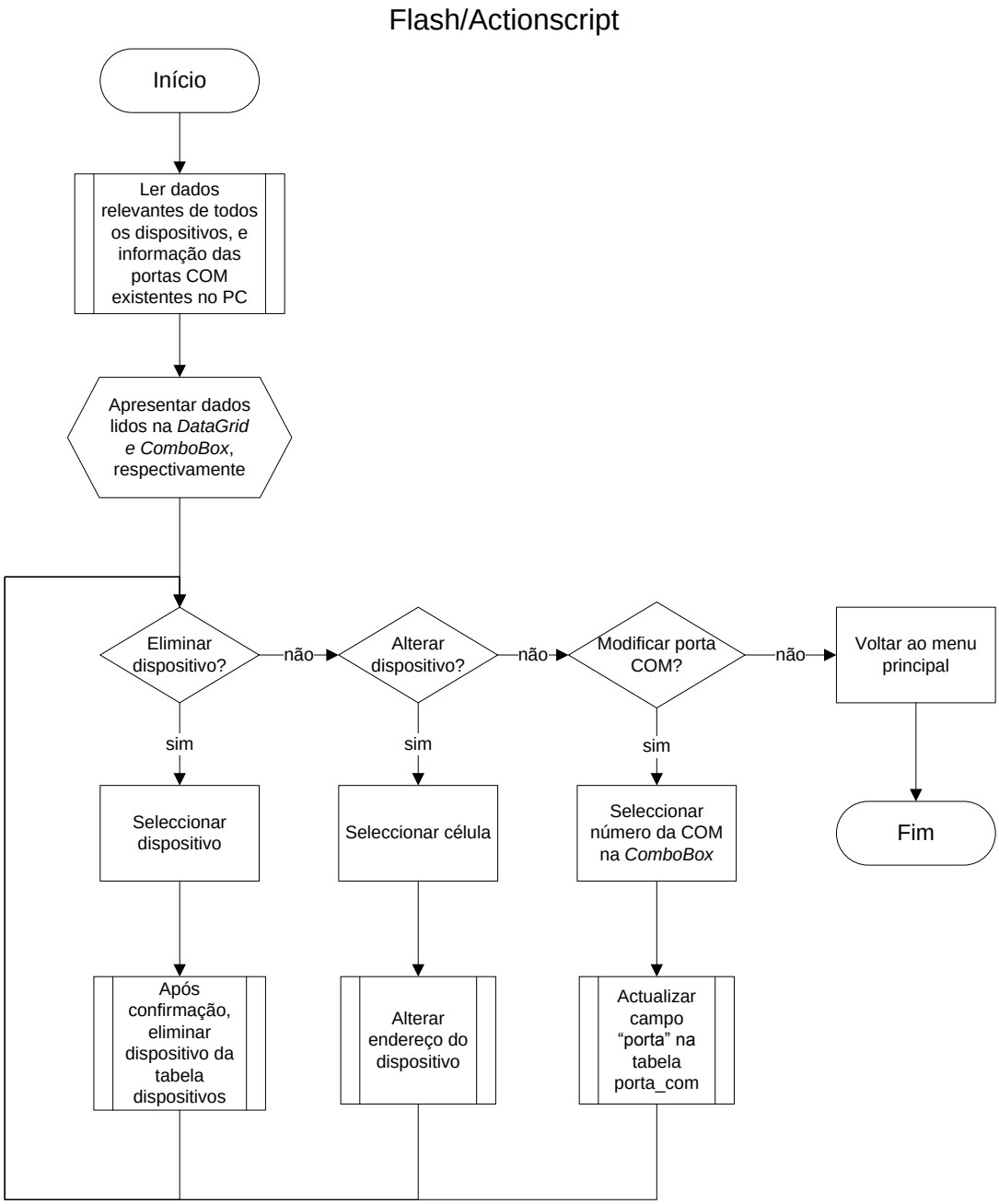


Fig. 55 Configuração do hardware - Flash/Actionscript

- ***Scripts* PHP**

ler_hardware.php

Responsável pela leitura dos dados de todos os dispositivos existentes, nomeadamente a leitura das tabelas "dispositivos", "estores" e "dispositivos_IR".

eliminar_hardware.php

Responsável pela eliminação de dispositivos da tabela "dispositivos". Basta eliminar o "id" presente nesta tabela pois, dadas as características da base de dados, os "id" presentes nas tabelas associadas serão também eliminados.

update_hardware.php

Responsável pela actualização do endereço na tabela "dispositivos" ou "estores", consoante o tipo de endereço e dispositivo em questão.

guardar_com.php

Este *script* apresenta duas funcionalidades. A primeira é a gravação da porta COM associada à comunicação com o subsistema IR, a segunda é a actualização dessa porta COM. A primeira funcionalidade é usada apenas quando o dispositivo IR é configurado, caso não exista ainda uma porta COM associada (em apresentação no capítulo 8). A segunda funcionalidade é usada sempre que se verifique que a porta COM existente é diferente da seleccionada, e como tal será feita a sua actualização. A tabela "porta_com" retém o valor da porta COM em utilização.

Capítulo VII

7. Subsistema de Controlo de Dispositivos por IR

7.1 Introdução

O subsistema apresentado neste capítulo visa essencialmente a interpretação e emissão de sinais infravermelhos, tendo em conta a sua aplicação no controlo remoto de aparelhos domésticos. Para se perceber um pouco melhor este tipo de sinais bem como a sua aplicação em electrónica é fundamental fazer uma introdução à optoelectrónica.

A optoelectrónica poderá ser descrita como uma área científica que associa a electrónica e a óptica com o intuito de desenvolver dispositivos capazes de converter sinais eléctricos em sinais ópticos ou vice-versa e portanto, qualquer dispositivo que opere como um transdutor nestas condições, é considerado um dispositivo optoelectrónico. O desenvolvimento nesta área começou essencialmente após a invenção do laser em 1958, e desde então, têm-se dado passos importantes, não só a nível de evolução dos dispositivos optoelectrónicos mas também na sua aplicação em áreas como a transmissão de dados. Tal desenvolvimento conduziu portanto a um grande impacto no quotidiano, actualmente a optoelectrónica está presente nos mais diversos sistemas e dispositivos, desde transmissão de sinais de TV por cabo feita por intermédio de fibras ópticas, unidades de leitura de CD (*Compact Disc*), controlo remoto de diversos aparelhos através de infravermelhos (amplamente divulgado), tais como televisões, ar condicionado, etc, entre outras aplicações [54][55].

Um sistema dito optoelectrónico é essencialmente constituído por uma fonte emissora de luz, um detector de luz e o respectivo meio de transmissão, tal como indicado na Fig. 56. A luz pode ser de diferentes tipos, dependendo da aplicação. Neste caso em concreto está-se a lidar com luz infravermelha proveniente do telecomando IR do aparelho em questão. Este tipo de luz é invisível ao olho humano, dado que se situa fora da gama espectral compreendida entre os 400 nm (nanometros) e os 700 nm [55].

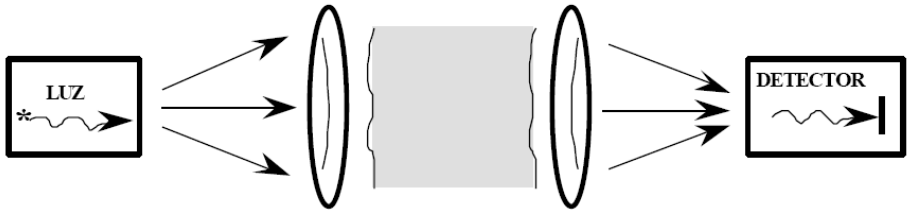


Fig. 56 Exemplo genérico de um sistema optoelectrónico [55]

Para que se possa ficar com uma melhor ideia acerca do comprimento de onda dos infravermelhos em relação aos diversos tipos de radiação, na figura seguinte é apresentado o espectro electromagnético, salientando o espectro visível ao olho humano.

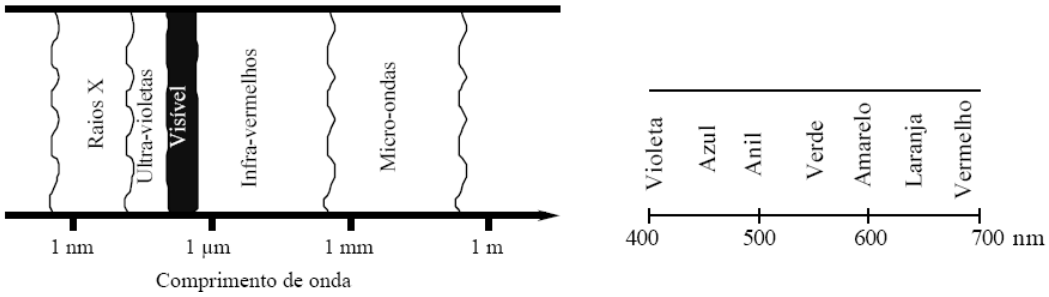


Fig. 57 Espectro electromagnético [55]

O facto da radiação infravermelha ser invisível é sem dúvida um dos aspectos chave, permitindo aumentar bastante o seu leque de aplicações. Este aspecto associado à imunidade a interferências electromagnéticas determina a sua grande utilidade, nomeadamente no controlo remoto de dispositivos [55].

7.2 Emissão e Recepção de Infravermelhos

A transmissão de sinais infravermelhos é bastante útil, como se sabe, contudo a transmissão deste tipo de sinais tem de ser feita segundo determinados critérios, devido à existência de diversas fontes exteriores emissoras de infravermelhos, basicamente tudo o que liberta calor liberta infravermelhos, e portanto é necessário distinguir um sinal IR pertinente daquele que representa apenas ruído. A solução passa pela modulação do sinal IR, desta forma o receptor facilmente distingue um sinal pertinente de um ruído. Existem inúmeras técnicas de modulação e codificação de sinais IR, sendo que a mais usada é a modulação em amplitude, tendo em conta a sua simplicidade de implementação. Basicamente, neste tipo de modulação os sinais IR formam grupos de impulsos a uma determinada frequência (frequência da portadora) e são delimitados por espaços no qual nenhum sinal é gerado. Em termos de codificação, esta varia de protocolo para protocolo como será apresentado na próxima secção [56][57].

O circuito de transmissão (telecomando IR) é constituído essencialmente por uma unidade de controlo e uma unidade de emissão IR. A unidade de controlo é constituída por um microcontrolador responsável pela modulação e codificação do sinal. O sinal é

emitido por intermédio de um ou mais LEDs (*Light Emitting Diodes*) IR (unidade de emissão IR). O circuito de transmissão é caracterizado pelo seu baixo consumo e elevado alcance na emissão do sinal infravermelho [56].

O circuito de recepção é bastante mais complexo do que o circuito de transmissão, na medida que terá de desempenhar diversas funções, nomeadamente todo o processo de condicionamento até à obtenção de um sinal eléctrico interpretável, para que seja possível identificar e executar o comando em questão [56].

Na figura seguinte encontra-se ilustrado de uma forma geral todo o processo que decorre desde a transmissão do sinal IR até à sua recepção e interpretação. A unidade de controlo de transmissão, a título de exemplo, encontra-se representada pela família de microcontroladores MC9S08RC/RD/RE/RG. Em relação à unidade de controlo usada na interpretação do sinal, esta depende do aparelho em questão ("RECEIVER'S MCU").

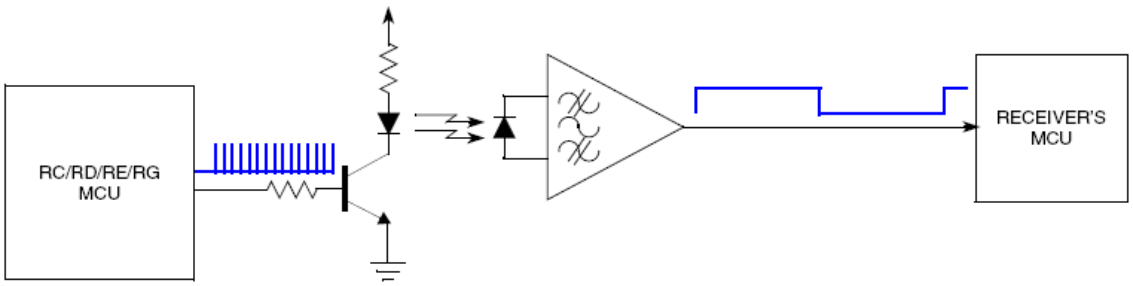


Fig. 58 Esquema genérico do processo de transmissão e recepção IR [57]

7.3 **Protocolos IR**

Actualmente existe no mercado uma grande variedade de aparelhos controláveis por infravermelhos, e como tal existe uma grande diversidade de fabricantes. O que acontece, é que muitos dos fabricantes adoptaram o seu próprio protocolo IR, e portanto não se pode dizer que existe um protocolo standard, o que traz alguns problemas quando o objectivo é por exemplo desenvolver um emissor IR universal, dado que existem algumas diferenças entre os diferentes protocolos. Ora, neste trabalho, não foi desenvolvido um subsistema capaz de interpretar todos os protocolos IR existentes, o que seria uma tarefa bastante difícil, mas sim um subsistema capaz de interpretar alguns dos protocolos mais usados, como por exemplo o protocolo Philips RC-5 e o Sony SIRC, protocolos que se encontram na gama dos 36-40 kHz de frequência de portadora.

Para que seja possível interpretar comandos IR de diferentes protocolos é necessário primeiro conhecer as principais características dos protocolos para uma posterior identificação de diferenças e semelhanças entre eles no sentido de criar um algoritmo. Será tido como referência alguns dos protocolos presentes em [56].

- **Frequência da portadora**

O valor da frequência da portadora (*carrier frequency*) pode estar compreendido entre os 30 kHz-60 kHz, contudo uma grande parte dos protocolos usa frequências na gama dos 36-40 kHz. Este é portanto um critério importante e que terá de ser tido em conta na leitura da trama IR.

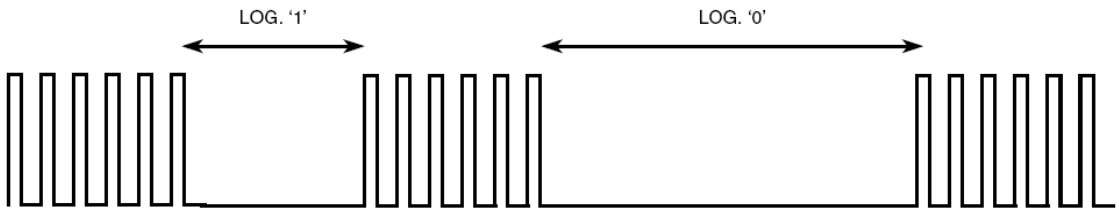
- ***Duty Cycle* da portadora**

O *duty cycle* da portadora é mais ou menos comum entre os diferentes protocolos, este valor é na ordem de 1/4, 1/3 do período. Esta característica será tida em conta na emissão IR.

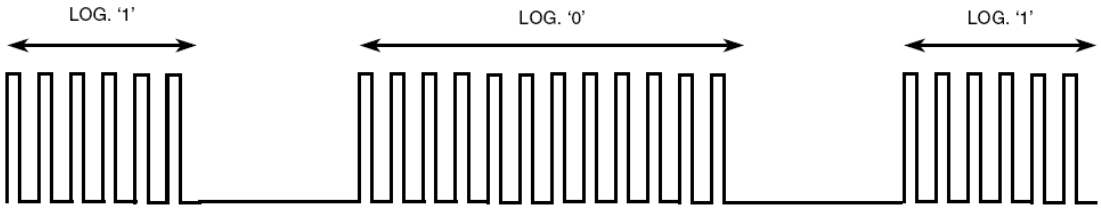
- **Modulação e codificação**

Como foi já dito existem diversas técnicas de modulação, porém a técnica de modulação em amplitude é sem dúvida a mais comum. Em termos de codificação, com base nesta técnica, podem ser usados quatro tipos de codificações, dependendo do protocolo em questão: distância de impulso (*pulse distance*), largura de impulso (*pulse*

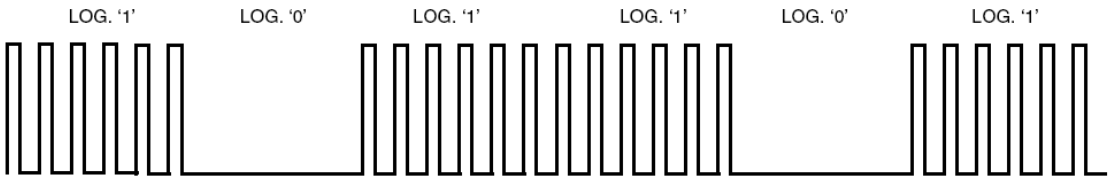
width), posição de impulso (*pulse position*) e *Manchester*. A figura seguinte ilustra a representação dos níveis lógicos ‘0’ e ‘1’ em cada uma das codificações, desta forma será fácil perceber as diferenças entre estas [57].



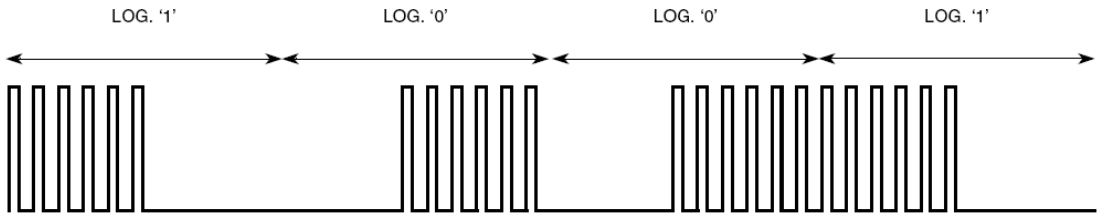
(a) Codificação por distância de impulso



(b) Codificação por largura de impulso



(c) Codificação por posição de impulso



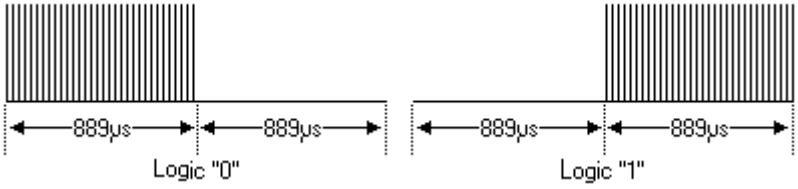
(d) Codificação de *Manchester*

Fig. 59 Representação dos vários tipos de codificação [57]

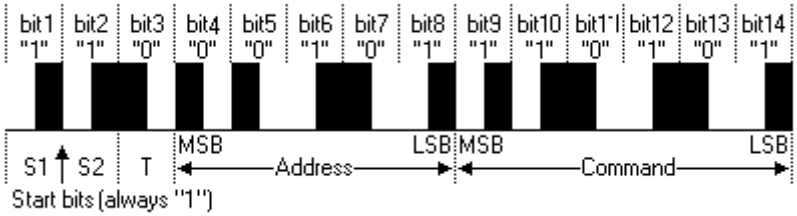
- **Constituição da trama IR**

A constituição da trama IR é também uma variável, essencialmente no número de *bits* que a constituem, contudo, em termos de estrutura, basicamente uma trama IR é constituída pelos campos “Start Bit(s)”, “Address” e “Command”, podendo existir excepcionalmente outros campos, porém estes são os básicos. O intervalo de tempo entre tramas é geralmente bastante grande, este intervalo indica ao receptor o final de cada trama. Em suma, dadas as suas características, pode dizer-se que a transmissão IR se assemelha à transmissão série assíncrona [56].

A título de exemplo, será apresentada na figura seguinte uma trama baseada no protocolo Philips RC-5, um protocolo com bastante sucesso a nível mundial. Com base na figura poderá assim perceber-se melhor a constituição de uma trama IR.



(a) Representação dos valores lógicos '0' e '1'



(b) Exemplo de uma trama Philips RC-5

Fig. 60 Protocolo Philips RC-5 [56]

O protocolo Philips RC-5 usa uma modulação de *Manchester* com uma frequência de portadora de 36 kHz (*duty cycle* de 1/4 ou 1/3). Os *bits* '0' e '1' ocupam o mesmo tempo de execução (1,778 milisegundos (ms)), nos quais metade do tempo é preenchido com um trem de impulsos a uma frequência de 36 kHz e outra metade onde existe um espaço de tempo "morto", onde nenhum sinal é emitido [56].

Em termos de informação enviada, a trama é constituída por dois "Start Bits", um *bit* "Toogle" e os respectivos campos "Address" e "Command". O *bit* "Toogle" é invertido sempre que uma tecla é pressionada, é usado para distinguir uma tecla que se mantém pressionada de uma que foi pressionada pela segunda vez consecutiva. O campo "Address" indica o tipo de aparelho que se pretende controlar (5 *bits*) e finalmente o campo "Command" representa o comando a enviar (6 *bits*) [56].

7.4 Constituição

O subsistema desenvolvido tem a capacidade de interpretação, armazenamento e emissão de comandos IR na gama de frequências de portadora dos 36-40 kHz, tendo em conta o uso da modulação em amplitude.

As tramas IR são adquiridas por intermédio de um circuito de leitura baseado num microcontrolador, e enviadas ao PC via USB para posterior armazenamento na base de dados. Depois de armazenadas, as tramas IR poderão ser enviadas via RF para um circuito de emissão IR. Como se pode verificar distinguem-se três processos distintos, aquisição, transmissão e emissão IR. Estes processos encontram-se distribuídos por dois módulos, os módulos aquisição e emissão IR.

7.5 Módulo de Aquisição IR

O módulo de aquisição é o responsável por todo o processo de aquisição, interpretação, armazenamento e transmissão IR. Cada um destes processos é desempenhado por blocos distintos, os quais serão apresentados em seguida. A figura seguinte apresenta os blocos constituintes deste módulo.

Fig. 61 Blocos constituintes do módulo de aquisição IR

7.5.1 Comunicação com o PC

A comunicação entre o PC e o módulo de aquisição é necessária por duas razões. A primeira devido à necessidade de armazenamento de toda a informação interpretada pelo microcontrolador na base de dados, a segunda devido à transmissão dos dados armazenados para o módulo de emissão.

A comunicação é feita através de USB. A escolha deste tipo de conexão deve-se essencialmente à sua presença em qualquer computador actual e ao facto de permitir a alimentação do circuito em questão, não sendo portanto necessário o uso de uma fonte de alimentação externa, permitindo correntes até 500 mA (miliamper)

O microcontrolador possui apenas interface USART (*Universal Synchronous Asynchronous Receiver Transmitter*), não implementando portanto o protocolo USB, e como tal não é possível conectá-lo directamente à porta USB. Tendo em conta esta situação será necessário um dispositivo intermédio que faça uma conversão USB-UART (*Universal Asynchronous Receiver Transmitter*) e vice-versa. O dispositivo responsável por esta conversão é o FTDI FT232RL. Este dispositivo possibilita assim uma comunicação com o microcontrolador via USB tendo em conta os critérios presentes numa comunicação série assíncrona, estando a porta USB associada a uma porta COM virtual (VCP – *Virtual COM Port*). A criação da COM virtual requer a instalação de um *driver*. Todos os passos para a instalação e configuração dessa COM encontram-se explicados no manual do sistema, presente em anexo.

- **FTDI FT232RL**

A Texas Instruments também disponibiliza conversores deste tipo, contudo é necessário adicionar alguns componentes externos para o seu funcionamento, nomeadamente um cristal, daí a escolha ter recaído sobre o FTDI. Na figura seguinte é possível ver a imagem do FTDI com o encapsulamento usado (SSOP - *Shrink Small-Outline Package*) e a sua distribuição de pinos.

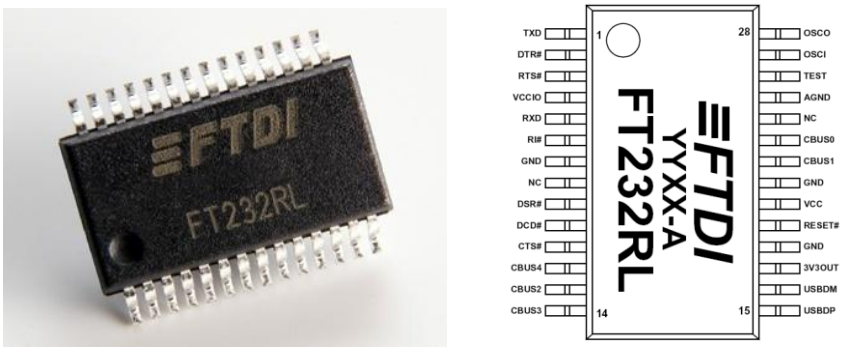


Fig. 62 FTDI FT232RL (SSOP) [58]

As características mais relevantes deste dispositivo são as seguintes (extraídas do *datasheet* [58])

- Conversão USB-UART e vice-versa, num único integrado
- UART com suporte para 7 ou 8 bits de dados, 1 ou 2 stop bits e paridade par/ímpar/marca/espaço ou nenhuma

- Taxa de transferência entre 300Mbps-3Mbps (níveis TTL - *Transistor-Transistor Logic*)
- Compatibilidade com USB 1.0 e 2.0
- 4 Pinos configuráveis pelo utilizador (conexão a LEDs para informação de transmissão/recepção por exemplo)
- EEPROM integrada para armazenamento de alguns parâmetros, nomeadamente o USB VID/PID (*Vendor/Product ID*)
- Suporte para alimentação via USB (*Bus Powered*)
- Clock gerado internamente
- Baixo consumo

O FTDI não necessita de componentes externos para o seu funcionamento, contudo para a diminuição de interferências externas é importante adicionar alguns componentes, nomeadamente ferrite. Para além disso, tendo em conta que se pretende uma alimentação de todo o circuito a partir do *bus* USB, o esquema de ligações deverá ser o seguinte.

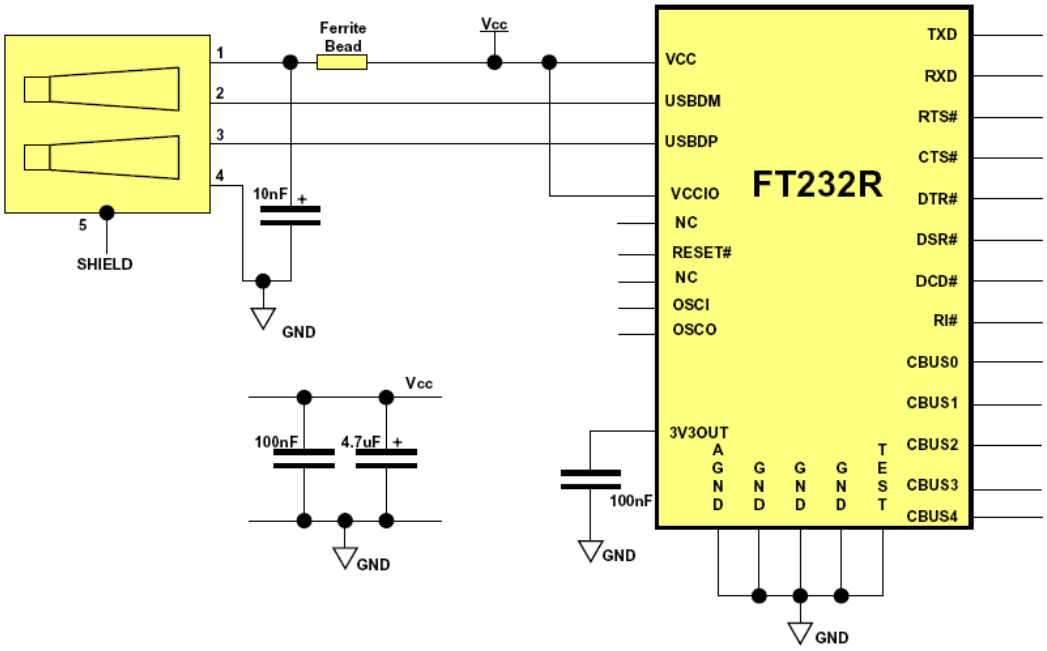


Fig. 63 Esquema de ligações do FTDI (*Bus Powered)* [58]

A nível de pinos de interface, são essencialmente utilizados o "TXD" e "RXD", para comunicação com o microcontrolador e módulo XBee. Os pinos "DTR#" e

“RTS#” são utilizados excepcionalmente para que seja possível a actualização de firmware do módulo XBee.

7.5.2 **Aquisição dos Sinais IR**

Após a análise dos vários parâmetros característicos de uma trama IR chegou-se à conclusão que para ser possível reproduzir sinais IR de diferentes protocolos é necessário conhecer todos os tempos de transmissão de cada *bit* constituinte da trama e também a sua frequência de portadora, só assim será possível contornar o facto de existirem diferentes codificações e diferentes frequências de portadora. O *duty cycle* da portadora é semelhante nos vários protocolos analisados, podendo fixar-se o valor em 1/4 do período.

O ponto de partida está dado, contudo a trama IR só poderá ser interpretada se o sinal IR for reconhecido, isto significa que é necessário em primeiro lugar traduzir o sinal IR num sinal eléctrico. Esta tradução será feita individualmente para a aquisição da trama e da portadora.

7.5.2.1 **Aquisição da Trama IR**

A trama IR é adquirida através do receptor IR Sharp GP1UX511QS. Este receptor faz todo o condicionamento do sinal IR apresentando na sua saída um sinal TTL de acordo com a trama recebida (Fig. 64). Inicialmente foi desenvolvido um circuito com vários estágios para o condicionamento do sinal IR, contudo, depois de efectuados alguns testes verificou-se que não seria uma solução muito fiável e portanto optou-se pelo uso deste receptor. O problema do uso deste receptor é o facto de ser necessária uma aquisição individual da portadora com base num circuito auxiliar, o que não era necessário no circuito inicialmente desenvolvido.

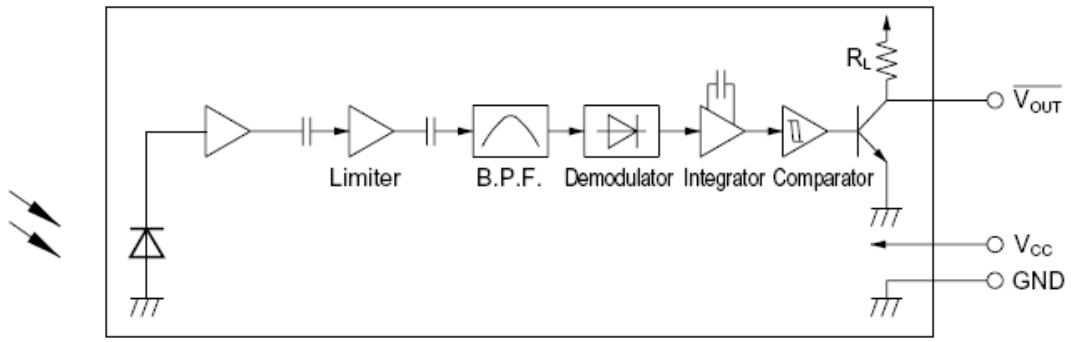


Fig. 64 Diagrama interno do receptor Sharp série GP1UX51QS [59]

O sinal IR é adquirido através de um fotodíodo sendo posteriormente amplificado e limitado ("Limiter"). A limitação é necessária para garantir um sinal de amplitude constante e como tal, independente da distância de emissão. O andar seguinte, o filtro passa banda ("B.P.F" do inglês *Band-pass Filter*), está sintonizado para uma determinada frequência (variável consoante o modelo de receptor, neste caso 38 kHz) e portanto só passam ao andar seguinte sinais com frequências próximas da frequência sintonizada (largura de banda de aproximadamente 4 kHz). Os andares seguintes, "Demodulator", "Integrator" e "Comparator" são responsáveis pela desmodulação do sinal e apresentação do mesmo na forma TTL. É importante ter em atenção que a presença de um sinal modulado é traduzida na saída como um nível lógico '0'. Na figura seguinte é apresentado o exemplo de um sinal IR modulado e a respectiva forma de onda à saída do receptor [60].

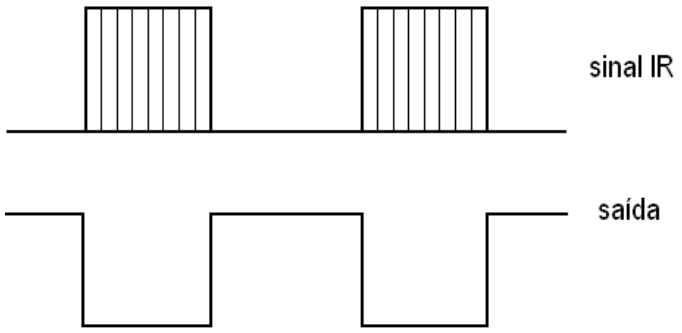


Fig. 65 Exemplo do sinal de saída para uma determinada trama IR

- **Sharp GP1UX511QS**

Fig. 66 Sharp GP1UX511QS [61]

O receptor Sharp pode ser perfeitamente usado na leitura de tramas de diversos protocolos baseados numa modulação em amplitude. Na tabela seguinte são apresentadas as principais características de alguns dos protocolos existentes, nomeadamente a nível de tempos de transmissão e frequência da portadora. Com base nessas características será feita uma análise do receptor Sharp no sentido de mostrar a sua compatibilidade.

Informação			**Modulação**		**Ausência de sinal**	
Protocolo	Portadora (kHz)	Codificação	tmín (µs)	tmáx (µs)	tmín (µs)	tmáx (µs)
Philips RC-5	36	*Manchester*	889	889	889	889
Sony SIRC	40	Largura de impulso	600	1200	600	600
Nokia NRC17	38	*Manchester*	500	500	500	500

Tabela 3 Características principais de alguns protocolos

O modelo escolhido está optimizado para receber sinais IR com portadoras na ordem dos 38 kHz, contudo dada a sua largura de banda de aproximadamente 4 kHz, permitirá uma recepção sem problemas de frequências na gama dos 36-40 kHz. É portanto compatível com os protocolos indicados [60].

Em relação ao tipo de codificação usado não existe problema, o importante é o tipo de modulação, pois este receptor só permite a recepção de sinais IR modulados em amplitude. Dado que o tipo de modulação mais usado é em amplitude este aspecto acaba por não ser muito relevante [60].

Os tempos de transmissão, ou seja tempo de modulação do sinal e tempo em que não existe sinal, são também critérios a ter em conta. Como se pode analisar na Tabela 3 estes valores variam bastante entre os protocolos analisados. O receptor Sharp impõe

algumas condições no que diz respeito a tempos de transmissão, essas condições são as seguintes [59]:

- O tempo de modulação mínimo admitido é de 200 µs (microsegundos)
- A razão entre o tempo de modulação e o tempo total de transmissão de uma trama (D_t) terá de ser menor ou igual que 40%. Esta razão é dada pela seguinte expressão:

$$D_t = (\sum_{N=1}^{n} t_N / T) \times 100\ (\%) \quad (2)$$

A figura seguinte proporciona um melhor entendimento das variáveis envolvidas no cálculo da razão, D_t.

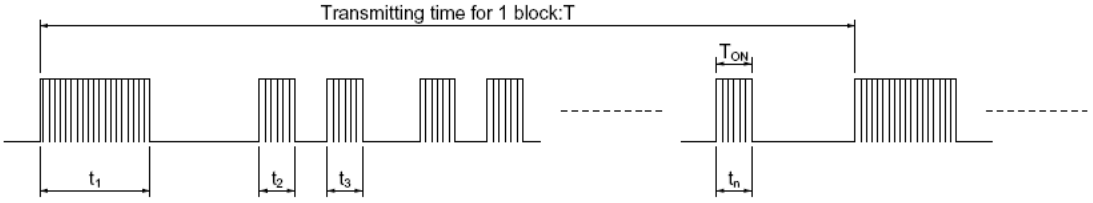


Fig. 67 Parâmetros envolvidos no cálculo da razão D_t [59]

Analisando a Tabela 3 verifica-se que todos os protocolos apresentam tempos de modulação superiores a 200 µs e portanto o primeiro critério verifica-se. Em relação à verificação do segundo critério são necessários alguns cálculos. Tomando como exemplo uma trama do protocolo Sony SIRC presente na figura seguinte, obtém-se:

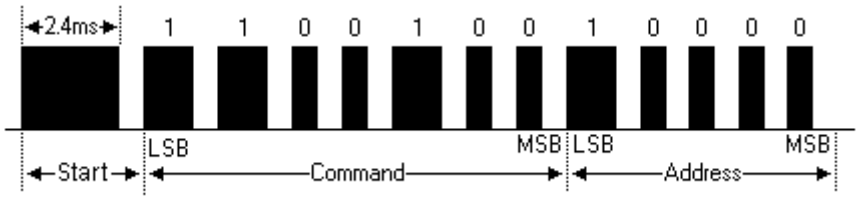


Fig. 68 Exemplo de uma trama do protocolo Sony SIRC [56]

$$tempo\ de\ emissão\ (ms) = t_{start}on + t_{Command}on + t_{Address}on \quad (3)$$

$$tempo\ "morto"\ (ms) = t_{Start}off + t_{Command}off + t_{Address}off + t_{Repeat} \quad (4)$$

$$tempo\ da\ trama\ (ms) = tempo\ de\ emissão + tempo\ "morto" \quad (5)$$

Sabendo que,

$t_{start}on\ (ms) = 2{,}4$ e $t_{start}off\ (ms) = 0{,}6$

$t_{,1},on\ (ms) = 1{,}2$ e $t_{,1},off\ (ms) = 0{,}6$

$t_{,0},on\ (ms) = t_{,0},off\ (ms) = 0{,}6$

$t_{Repeat}\ (ms) = 50$

Feitas as contas obtém-se, $D_t \approx 19\%$, como tal um sinal IR baseado neste protocolo poderá ser adquirido pelo receptor Sharp, verificando-se assim o segundo critério. Os restantes protocolos presentes na tabela são também compatíveis com este receptor. Com certeza que existem muitos mais, contudo estes foram tidos como referência.

Em termos de conexão, este receptor é ligado directamente à unidade de interpretação da trama IR (PIC), contudo para melhorar o seu desempenho é necessário aplicar um filtro RC na sua linha de alimentação, devido a ruídos que possam ser introduzidos. Na Fig. 69 encontra-se representada a conexão do filtro RC a aplicar, com valores de $C_0 = 47\mu F$ e $R_1 = 47\ \Omega$.

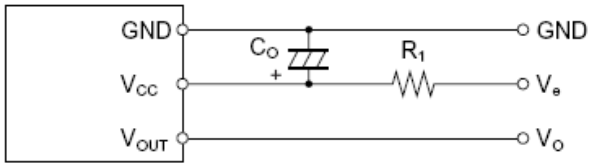


Fig. 69 Filtro RC aplicado externamente ao receptor IR [59]

7.5.2.2 **Aquisição da Portadora**

O uso do receptor IR da Sharp, como se sabe, torna necessária a utilização de um circuito auxiliar para detecção da portadora. Este circuito encontra-se dividido em três estágios: detecção IR, amplificação e limitação e finalmente ajuste do sinal (Fig. 70). Basicamente este processo consiste na implementação de todos os estágios até ao filtro passa banda, presentes no diagrama interno do receptor Sharp (Fig. 64). O filtro não foi implementado uma vez que o processo de gravação de comandos IR será feito com relativa proximidade do módulo de aquisição e como tal não existirá grande problema com interferências exteriores, como se poderá verificar nos resultados obtidos.

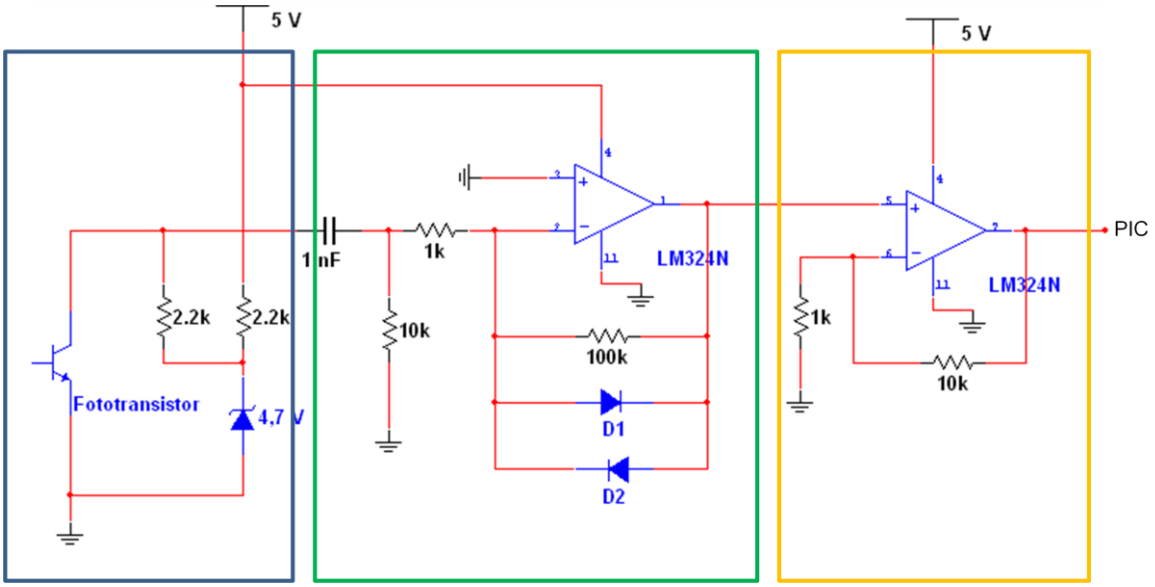


Fig. 70 Circuito de aquisição da portadora

- **Detecção do sinal IR**

O estágio de detecção consiste na tradução da luz infravermelha numa corrente eléctrica, denominado estágio de conversão "luz-corrente". Esta conversão é feita através de um fototransístor do tipo NPN da Vishay. Inicialmente foi usado um fotodíodo comum, contudo surgiram alguns problemas, nomeadamente na qualidade do sinal obtido e portanto após alguns testes acabou por optar-se por um fototransístor. Este estágio permite a detecção de qualquer frequência de portadora, contudo no seu todo, o circuito de aquisição está limitado à gama de 36-40 kHz, devido ao receptor Sharp.

Fig. 71 Fototransístor NPN da Vishay [62]

- **Amplificação e limitação do sinal**

Este estágio tem como objectivo a conversão da corrente em tensão, de forma a obter-se um sinal aproximado do sinal modulado enviado pelo emissor IR. O sinal é amplificado e limitado. A limitação é feita pela mesma razão explicada anteriormente

no receptor Sharp, ou seja, independência da amplitude em relação à proximidade do receptor IR, neste caso do fototransístor.

- **Ajuste do sinal**

O estágio final tem como objectivo a obtenção de um sinal interpretável pelo microcontrolador, e como tal é necessário ajustar o sinal para níveis de tensão admissíveis. A informação segue então para o microcontrolador PIC.

7.5.3 Leitura dos sinais IR

7.5.3.1 Microcontrolador

A interpretação dos sinais IR encontra-se a cargo de um microcontrolador, este era já um facto desde o inicio do projecto. O microcontrolador a escolher, para desempenhar a função pretendida, deveria reunir as seguintes características básicas:

- Unidade USART para comunicação com o exterior
- Portas I/O (*Input/Output*) disponíveis para aquisição do sinal IR, conexão a LEDs, botões de pressão, etc
- *Timer* de 16 bits para leitura dos tempos envolvidos no sinal IR
- Possibilidade de ser programado no próprio circuito (ICSP – *In-Circuit Serial Programming*), para posteriores actualizações de firmware se necessário
- Baixo consumo, dado que a alimentação seria por USB
- Memória de dados suficiente para o armazenamento temporário de comandos IR
- Relativamente compacto

Reunidas todas estas características, foi escolhido o microcontrolador PIC da Microchip, mais propriamente o modelo 16F876A, da série 16 MCU (Microcontroller).

Actualmente existem diversos fabricantes de microcontroladores e portanto a liberdade de escolha é muito grande. A escolha do fabricante, Microchip, deve-se essencialmente à existência de ferramentas de programação livres, qualidade da

documentação disponibilizada e devido ao envio de amostras grátis dos seus componentes para estudantes e empresas. Em relação à escolha da família, o factor eliminatório foi a existência de pelo menos uma unidade USART, tendo-se verificado que esta unidade está disponível apenas a partir da série 16 MCU, inclusive. Dentro desta série e tendo em conta os restantes critérios, foi escolhido o 16F876A. Na figura seguinte é possível ver o encapsulamento usado (PDIP – *Plastic Dual InLine Package*), bem como a disposição dos seus pinos. Na figura encontra-se apresentado o PIC 16F873A, dado que a disposição de pinos é exactamente igual. A diferença é apenas em termos de memória.

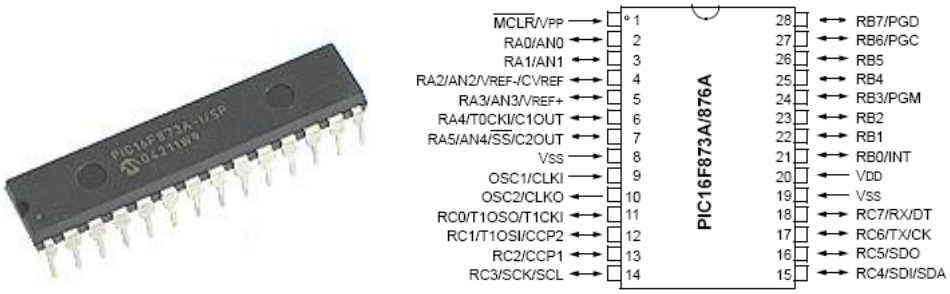


Fig. 72 Microchip PIC 16F873A [63]

As principais características do PIC 16F876A são as seguintes (extraídas do *datasheet* [63])

- Microcontrolador de 8 bits
- Arquitectura *Harvard*
- Operação até 20 MHz
- 14 k bytes de memória de código
- 368 bytes de memória RAM (*Random-acess Memory*)
- 256 bytes de memória EEPROM (*Electrically Erasable Programmable Read-Only Memory*)
- 22 pinos I/O
- 1 *Timer* de 16 bits e 2 *Timers* de 8 bits
- 2 Módulos PWM (*Pulse Width Modulation*): CCP – *Capture/Compare/PWM*
- Periféricos de comunicação USART e I^2C (*Inter-Integrated Circuit*)
- 5 ADC (*Analogic-Digital Converter*) de 10 bits
- ICSP

- Tecnologia de baixo consumo

Como se pode verificar, este microcontrolador adapta-se perfeitamente às necessidades deste projecto.

7.5.3.2 **Processo de Leitura**

Os sinais IR em aquisição não podem ser interpretados simultaneamente, como tal é necessário seleccionar cada um deles individualmente. O primeiro sinal a ser interpretado será o sinal da trama seguindo-se o sinal da portadora. Esta selecção é comandada pelo microcontrolador e é baseada em dois transístores bipolares que funcionam como interruptores, tal como indicado na figura seguinte.

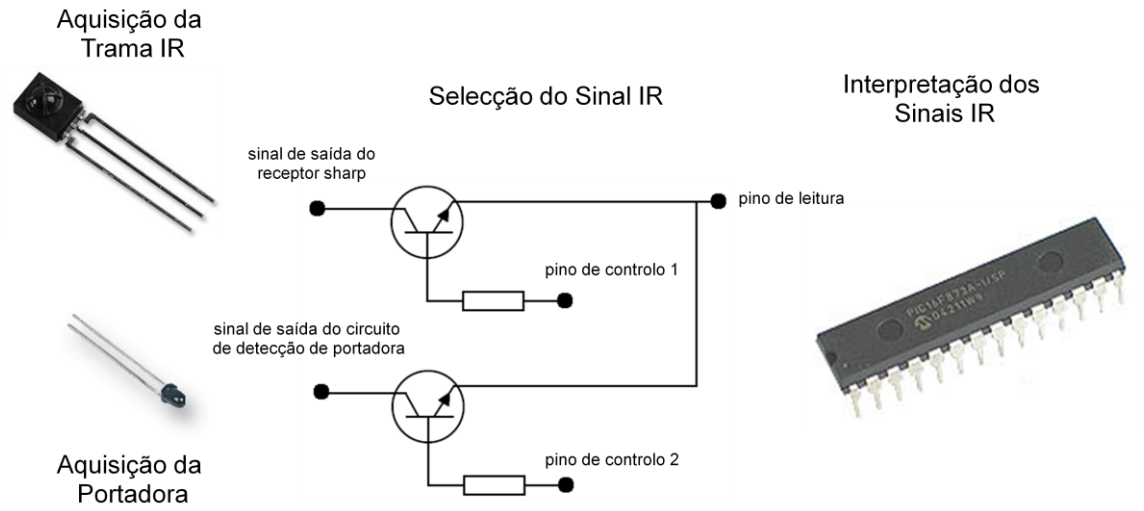


Fig. 73 Selecção dos sinais IR

Uma vez seleccionado o sinal será então feita a sua interpretação. Para se perceber melhor o processo de interpretação dos sinais será tido como referência um sinal IR genérico modulado em amplitude, com uma determinada portadora. O sinal encontra-se representado na figura seguinte.

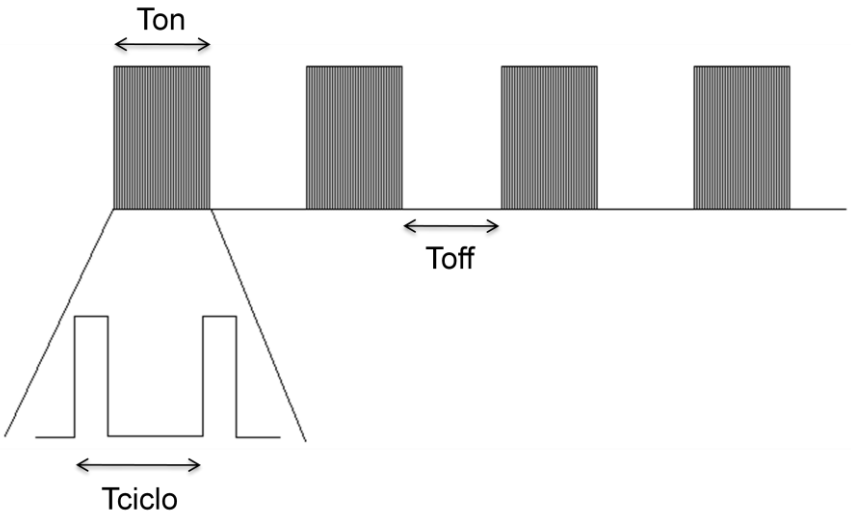


Fig. 74 Exemplo de uma trama IR

A leitura da trama consiste basicamente na leitura de cada um dos intervalos de tempo, onde existe modulação (Ton) e onde não existe sinal (Toff). É necessário ter em atenção que o sinal da trama aparece agora invertido em relação ao original (devido ao receptor Sharp), tal como indica a Fig. 75. Neste exemplo não existe problema, contudo os Ton e Toff nem sempre são iguais. No fundo o que é feito é capturar os instantes em que ocorrem as transições, ou seja, flancos descendentes (C1) e flancos ascendentes (C2). Imaginando que cada um dos instantes, C1 e C2, corresponde a um instante de tempo obtém-se:

$$T_{on} = C2_1 - C1_1 \quad (6)$$

$$T_{off} = C1_2 - C2_1 \quad (7)$$

Na implementação no microcontrolador estes instantes correspondem a um valor do *timer* e portanto para se obter o instante de tempo associado é necessário efectuar cálculos tendo em conta a frequência do cristal, como se poderá ver mais adiante no algoritmo implementado.

Neste processo de leitura são armazenados 36 pontos de captura. Os intervalos de tempo são posteriormente calculados e enviados ao PC, cabe depois à aplicação C/C++ desenvolvida detectar o tamanho da trama IR. Desta forma a leitura adapta-se a tramas IR de comprimentos diferentes. Esse processo é feito com base no intervalo de tempo existente entre emissão de tramas, o qual é bastante elevado comparativamente ao intervalo de tempo total da trama emitida e como tal facilmente se detecta o final da trama.

O processo de leitura da portadora é muito semelhante. A frequência da portadora é detectada com base no período do sinal, como tal, basta fazer a captura do instante num dos flancos (foi escolhido o flanco ascendente). Tendo em conta que C2 e C2' são também instantes de tempo, obtém-se:

$$T_{ciclo} = C2_2 - C2_1 \tag{8}$$

Na leitura da portadora são armazenados 16 pontos de captura, de maneira a garantir uma leitura mais credível. Os tempos de ciclo são posteriormente calculados e enviados ao PC, à semelhança do processo de leitura anterior. Esses tempos de ciclo são posteriormente analisados e comparados através da aplicação C/C++ obtendo-se assim o período efectivo do sinal.

Os instantes de transição associados aos sinais IR, são capturados por intermédio dos módulos CCP. A existência deste tipo de módulos não era um requisito obrigatório no microcontrolador a escolher, contudo tendo em conta que um sinal modulado em amplitude é equivalente a um sinal PWM, o uso destes módulos simplificou bastante todo o processo de leitura.

Na Fig. 75 encontra-se representada uma simulação dos sinais IR provenientes do receptor Sharp (em cima) e do circuito de detecção da portadora (em baixo). Com base nesta figura é possível perceber melhor o método usado no cálculo de cada um dos intervalos de tempo.

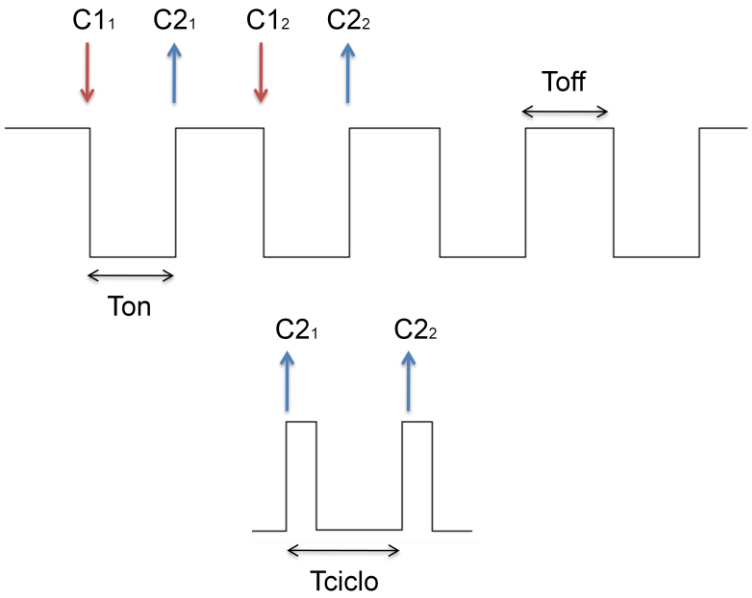


Fig. 75 Exemplo de sinais IR obtidos na aquisição da trama IR e portadora

Toda a informação IR interpretada é armazenada temporariamente na memória de dados do microcontrolador, sendo posteriormente enviada ao PC via USB, para uma última interpretação e futuro armazenamento na base de dados.

- **Algoritmo**

A implementação de todo o processo de leitura dos sinais IR e posterior comunicação com o PC encontra-se representada no fluxograma da Fig. 78. Todo o código de implementação foi desenvolvido na linguagem C e encontra-se disponível na íntegra em anexo.

A comunicação é feita por intermédio da unidade USART configurada no modo assíncrono. O pacote de dados é transmitido a um *baud rate* de 19200 bps e é do tipo 8-N-1, ou seja, 8 *bits* de dados, ausência de *bit* de paridade e 1 *stop bit*. O *start bit* já está implícito. Na figura seguinte é possível ver o exemplo de transmissão do caracter 'A'.

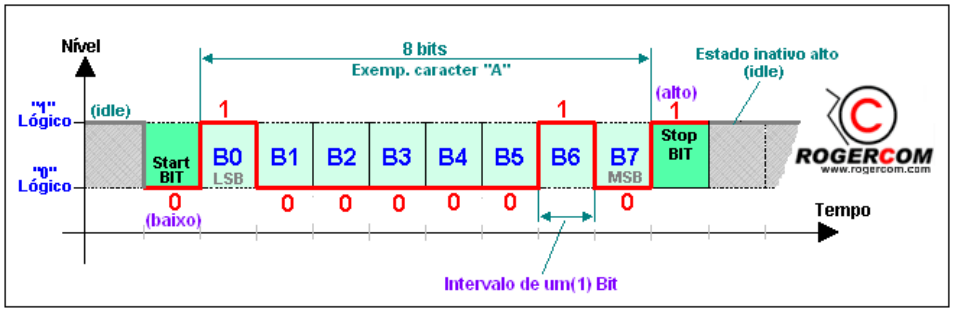


Fig. 76 Exemplo de transmissão do caracter 'A' [64]

Em termos de leitura, o processo é feito por intermédio dos módulos CCP configurados para a captura de sinais PWM. Cada um dos módulos dispõe de um registo de 16 *bits*, o qual dependendo do flanco para o qual é configurado, irá armazenar o instante associado à transição respectiva. O "CCP_1" está configurado para transições descendentes e o "CCP_2" para transições ascendentes. As transições ascendentes irão desencadear uma interrupção, na qual serão armazenados em *arrays* os instantes associados a cada um dos CCPs, dado que o seu valor irá ser actualizado nas próximas transições. Após a leitura de todos os instantes é necessário calcular os intervalos de tempo associados, e como tal será tido em conta o cristal. O cristal usado é de 12 MHz e sabendo que o *clock* interno do microcontrolador funciona a 1/4 da frequência do cristal, o tempo associado a um valor N presente no *timer* é calculado da seguinte forma:

$$T(\mu s) = N \times \frac{1}{\frac{f_{clock}}{4}} \quad (9)$$

Neste ponto encontra-se reunida toda a informação respectiva ao sinal IR adquirido, resta apenas fazer uma última análise antes dos dados serem enviados à aplicação de controlo (Flash). Essa análise é feita na aplicação C/C++ (aplicação "ponte"). A informação lida pelo PIC é então enviada ao PC no seguinte formato:

$\mathbf{T_{ciclo1}P}$	$\mathbf{T_{ciclo2}P}$	…	$\mathbf{T_{on1},}$	$\mathbf{T_{off1}.}$	$\mathbf{T_{on2},}$	$\mathbf{T_{off2}.}$	…	**F**

Fig. 77 Formato da informação enviada ao PC

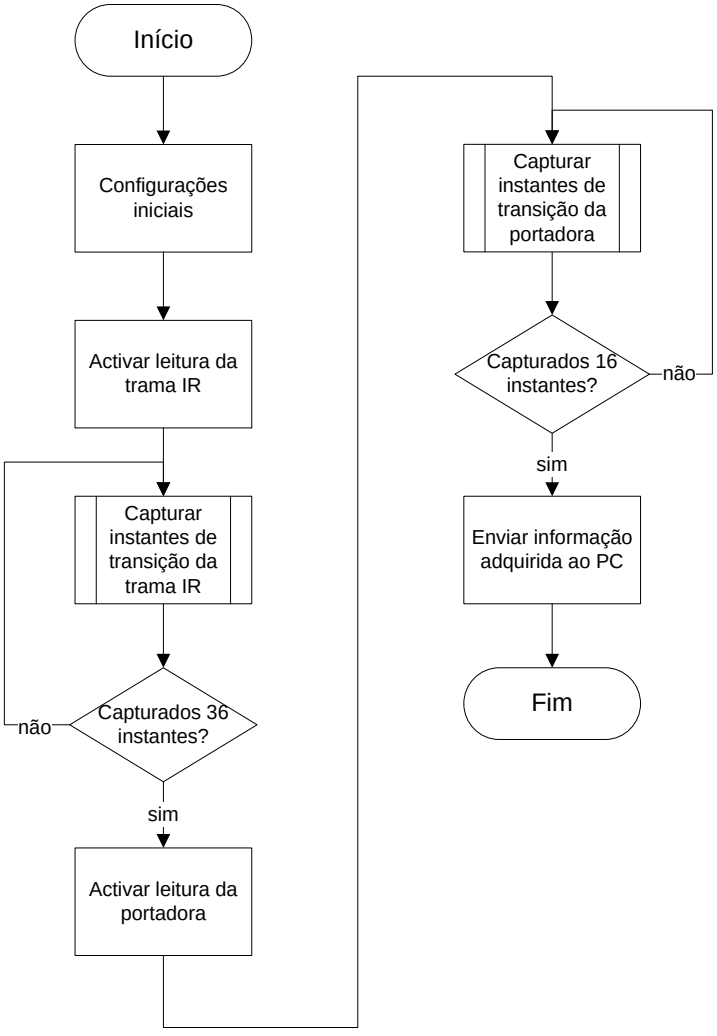


Fig. 78 Processo de leitura dos sinais IR

7.5.4 **Transmissão RF**

A transmissão de informação entre o módulo de aquisição IR e o módulo de emissão IR, é feita via radiofrequência, como já foi dito. Para executar este tipo de transmissão foram usados os módulos XBee da Maxstream, os quais implementam o protocolo ZigBee. A título de curiosidade, será feita uma breve apresentação a este protocolo, não entrando em muito detalhe, dado que estes módulos permitem uma total transparência ao utilizador, não sendo necessário ser um perito em ZigBee para trabalhar sobre esse protocolo, essa é uma das vantagens da utilização destes módulos.

- **Protocolo ZigBee**

O protocolo ZigBee é extremamente recente, tendo surgido em Dezembro de 2004. Este protocolo foi desenvolvido pela ZigBee Alliance, um consórcio entre diferentes entidades, nomeadamente fabricantes de semicondutores, provedores de tecnologia, etc, com o intuito de dar suporte a rede de sensores sem fios [65].

A política adoptada pela tecnologia ZigBee é essencialmente a criação de redes sem fios via RF com baixo consumo de energia e de elevada performance, encontrando-se em conformidade com a norma IEEE (*Institute of Electrical and Electronics Engineers*) 802.15.4 [65].

O ZigBee apresenta características únicas que o destacam dos restantes protocolos RF e que justificam o seu sucesso. Esta tecnologia foi projectada objectivando a criação de redes sem fios com as seguintes características [65] :

- Criação simples
- Baixo consumo de energia
- Interfaces de baixo custo, dado que o ZigBee apresenta uma pilha protocolar de implementação simplificada
- Diferentes tipos de topologias de rede, tais como: estrela, ponto-a-ponto e árvore
- Elevada fiabilidade
- Encriptação de dados até 128 bits
- Bandas de operação ISM (*Industrial, Scientific and Medical*), não requerendo licenças específicas

Tendo em conta todas estas características será fácil imaginar as inúmeras aplicações desta tecnologia. O ZigBee é portanto a solução ideal para a transmissão de dados via RF pretendida neste projecto.

- **Módulos XBee da Maxstream**

Os módulos ZigBee escolhidos, XBee da Maxstream, já tinham sido anteriormente testados pela empresa SAR e como tal, dado o seu *feedback* positivo optou-se pela sua utilização em detrimento de outros módulos com a mesma tecnologia.

Os módulos XBee estão disponíveis nas versões XBee e XBee PRO. Existe uma total compatibilidade entre ambas as versões, a diferença é essencialmente em termos de alcance RF. O XBee PRO apresenta um alcance superior.

A versão usada foi a XBee, tendo em conta que existe uma total compatibilidade entre ambas se for necessário obter um alcance superior bastará fazer a substituição pela versão PRO, fazendo as devidas configurações. Na figura seguinte encontram-se representados os módulos XBee com os diferentes tipos de antena disponíveis. Tendo em conta a aplicação em questão foi utilizada a "Chip Antenna".

Fig. 79 Módulos Xbee com diferentes tipos de antena [64]

As principais características do módulo XBee são as seguintes (extraídas do *datasheet* [66])

- Tensão de alimentação entre 2,4 V (Volts) e 3,4 V
- Potência máxima de transmissão de 1 mW (miliWatt)
- Alcance de 30 m em ambientes internos e 100 m em ambientes externos
- Frequência de operação ISM 2,4 GHz (gigahertz)
- 16 Canais
- Taxa de dados (interface) máxima de 115200 bps
- Taxa de transmissão RF de 250 kbps

- Topologias de rede do tipo estrela, ponto-a-ponto e malha
- Encriptação de 128 *bits* AES (*Advanced Encryption Standard*)
- Interface UART
- 7 Entradas Analógicas e 9 Entradas/Saídas Digitais
- 2 PWM
- Simples configuração de parâmetros, através do software X-CTU

- **Alimentação e interface com o PC**

A tensão de alimentação dos módulos XBee encontra-se na gama dos 2,4 V e 3,4 V, como se pode verificar existem diferenças de tensão em relação ao *bus* USB e portanto é necessário limitar a tensão de alguma forma. O FTDI utilizado apresenta um regulador interno de 3,3 V disponível a partir do pino 17, este regulador pode fornecer uma corrente até 50 mA, um valor suficiente para o módulo XBee, porém se no futuro se optar por uma versão PRO esse valor já não é suficiente. Por essa razão foi utilizado o regulador MAX604 da Maxim, capaz de fornecer uma corrente até 500 mA e uma tensão de saída de 3,3 V. Na figura seguinte encontra-se representado o esquema de ligações usado com este tipo de conversor (esquema típico de funcionamento presente no *datasheet* [67]).

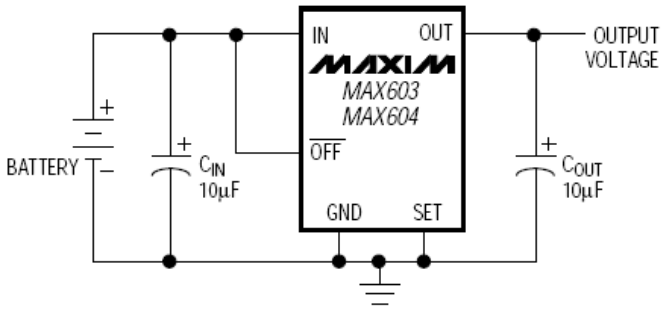


Fig. 80 Esquema típico de funcionamento com o MAX604 [67]

Em termos de alimentação o assunto está resolvido, contudo é necessário limitar também os sinais de entrada no XBee, nomeadamente nos pinos "DIN" (2), "DTR" (9) e "RTS" (16). Neste caso, a solução passa pelo uso de um simples divisor de tensão, indicado na figura seguinte.

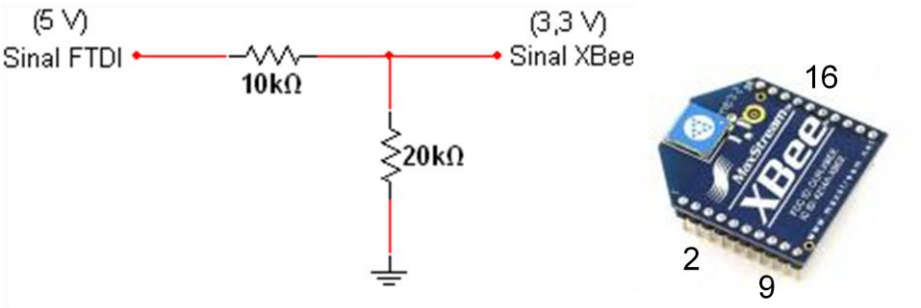


Fig. 81 Condicionamento de sinais de entrada no XBee

- **Configuração dos módulos**

O primeiro passo para a transmissão sem fios via ZigBee é a configuração dos módulos, de maneira a definir os parâmetros da rede pretendida. Deverá então fazer-se uma primeira análise sobre as características da rede, essencialmente o número de dispositivos que a compõem, topologia, modo de operação, taxas de dados (interface), modo de transmissão (*broadcast* ou *unicast*) e encriptação.

Características da rede pretendida:

- Modo de operação transparente
- Máximo de 256 dispositivos
- Topologia em estrela
- Taxa de dados aceitável
- Transmissão em *broadcast*
- Sem encriptação

O subsistema criado é apenas para simular o controlo de dispositivos IR e portanto é constituído apenas por dois módulos XBee, presentes no módulo de aquisição e emissão IR. No futuro, a ideia é ter um módulo de emissão IR em cada compartimento da habitação onde existam dispositivos IR, como tal a transmissão a partir do módulo de aquisição será feita em *broadcast*. A topologia de rede será em estrela, tal como indicado na figura seguinte.

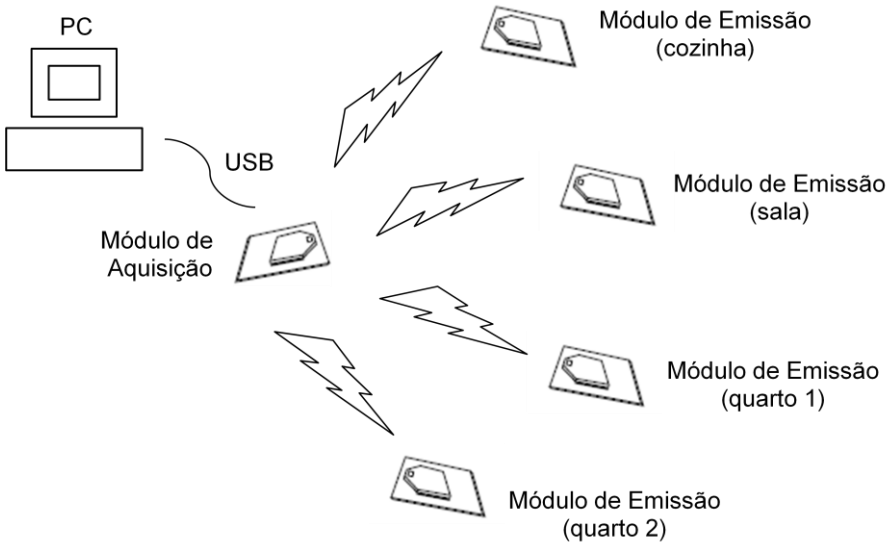


Fig. 82 Topologia da rede

A transmissão em *broadcast* apesar de ser bastante útil nesta situação apresenta algumas diferenças em relação à transmissão em *unicast*, nomeadamente pelo facto de não permitir ACKs (*Acknowledgements*) nem reenvios de informação. Os testes efectuados comprovaram que este detalhe não se revela muito importante dado que a transmissão decorre sem problemas. A nível de taxa de dados optou-se por um *baud rate* de 19200 bps, uma taxa aceitável e que praticamente não provoca erros de comunicação. Em termos de endereçamento dos dispositivos é utilizado um endereçamento de 16 *bits*, o mínimo permitido pelo módulo XBee, um valor mais do que suficiente. A encriptação de dados não foi utilizada uma vez que não tem muito interesse nesta situação, podendo ser activada e configurada sempre que necessário.

A escolha do modo de operação da rede é importante pois o interface de comunicação com o XBee é um interface série com a possibilidade de operação em dois modos distintos, o modo transparente e o modo API. No modo transparente os dados são enviados e recebidos da mesma forma que na comunicação série assíncrona segundo o padrão RS-232, por exemplo. No modo API as coisas passam-se de forma ligeiramente diferente, pois os dados transmitidos e recebidos estão contidos em frames, sendo possível definir várias operações ou eventos. Este modo de operação permite assim diversas funcionalidades, em especial a capacidade de identificação do endereço fonte de cada pacote recebido e o estado (sucesso/falha) de cada pacote transmitido [66]. Para a aplicação em questão o modo transparente adapta-se perfeitamente.

Após a definição das características da rede é necessário então fazer as devidas configurações nos módulos em questão. A configuração é feita através do software X-

CTU via USB, com o respectivo XBee presente no módulo de aquisição. Para mais detalhes basta consultar o manual do sistema, presente em anexo.

Os parâmetros essenciais a configurar são os seguintes: endereçamentos de origem e destino (16 *bits*), identificação do módulo, canal e taxa de transmissão. Depois existem parâmetros opcionais, nomeadamente o uso de encriptação. Os módulos XBee presentes nos módulos de aquisição e emissão IR serão denominados de "BASE" e "REMOTO", respectivamente. A tabela seguinte apresenta os parâmetros essenciais a configurar.

	MY	DL	DH	CH	NI	BD (bps)
BASE	0	0xFFFF	0	0x0C (CH 2)	"BASE"	19200
REMOTO	1	0	0	0x0C (CH 2)	"REMOTO"	19200

Tabela 4 Parâmetros a configurar nos módulos XBee

MY – Endereço associado ao módulo origem (*Source Address*)

DL – Endereço associado ao módulo destino, LSB (*Destination Address Low*)

DH – Endereço associado ao módulo destino, MSB (*Destination Address High*)

CH – Canal de transmissão (*Channel*)

NI – Nome identificativo do módulo (*Node Identifier*), campo opcional

BD – Taxa de dados (*Baud Rate*)

Na configuração de novos módulos de emissão IR bastará alterar os campos "MY" e "NI" do módulo XBee associado, os restantes campos mantêm-se inalterados.

7.6 Aplicação de Interface entre o Módulo de Aquisição e o PC

7.6.1 Estrutura

A aplicação desenvolvida é baseada em programação por classes em C/C++ com suporte MFC. Esta aplicação basicamente efectua a "ponte" entre a aplicação principal (Flash) e o módulo de aquisição IR, apresentando determinadas funcionalidades. Inicialmente tentou-se estabelecer esta "ponte" directamente através de uma extensão PHP para funcionamento com a porta série, contudo não se conseguiu obter os resultados esperados e portanto optou-se por usar esta aplicação como intermediária.

As funcionalidades são as seguintes: identificação das portas COM existentes no PC, recepção dos dados IR lidos pelo microcontrolador e transmissão de dados via RF para o módulo de emissão, seja para gravação do endereço na sua EEPROM ou emissão da trama IR. Cada uma destas funcionalidades é seleccionada consoante o comando passado como argumento através da linha de comandos, comando esse enviado pela aplicação Flash.

A aplicação é constituída essencialmente por duas classes. A classe "CSerialPort" e a "CSerial_Control". A "CSerialPort" é uma classe MFC livre para a manipulação da porta série com base no sistema operativo Windows. A "CSerial_Control" é a classe que contém todas as funções necessárias à implementação das funcionalidades anteriormente apresentadas. Esta classe será utilizada na função *main* da aplicação. Na figura seguinte é possível visualizar a estrutura da aplicação bem como a interacção com o exterior, como por exemplo com a aplicação Flash. A figura ilustra assim todas as interacções possíveis de executar pela aplicação C/C++.

O código de implementação encontra-se disponível em anexo.

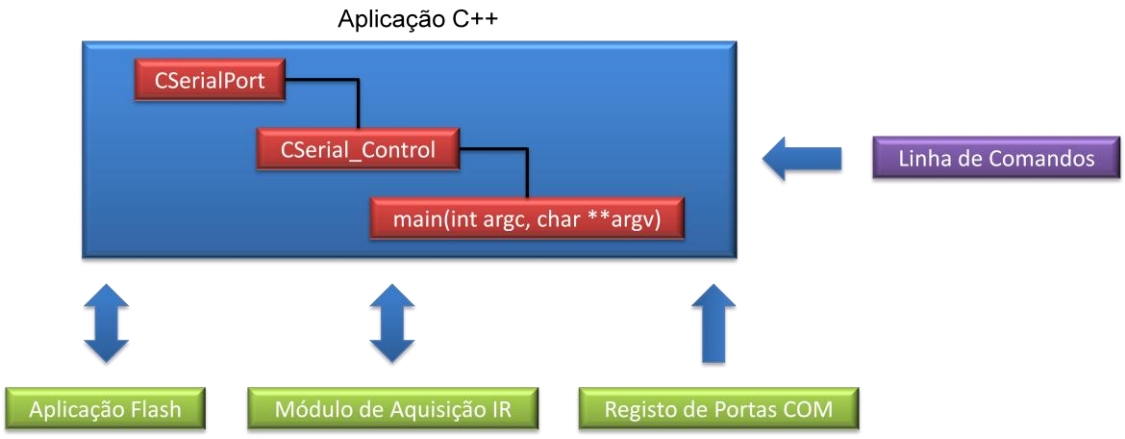


Fig. 83 Diagrama de interacção da aplicação C/C++

7.6.2 Classe CSerialPort

A classe "CSerialPort" é no fundo uma ferramenta para a manipulação da porta série, e como tal não será apresentada em pormenor. Basicamente esta classe disponibiliza diversas funções para que se possa trabalhar com a porta série de uma forma transparente e intuitiva, "escondendo" assim toda a complexidade existente. Para mais informações poderá consultar-se o seguinte *site* http://www.codeproject.com/KB/system/cserialport.aspx.

7.6.3 **Classe CSerial_Control**

A classe “CSerialControl” como já foi dito, implementa as funções necessárias à execução das diversas funcionalidades associadas ao subsistema IR. Esta classe tira partido da classe “CSerialPort” sempre que é necessária uma interacção com a porta série. As funções implementadas são as seguintes:

void abrir(int porta)

Responsável pela abertura da porta COM cuja identificação é passada como argumento. Executa a função “Open” da classe “CSerialPort” com os restantes parâmetros de comunicação já definidos, isto é: 19200 bps, 8-N-1 sem controlo de fluxo.

void fechar()

Responsável pelo fecho da porta COM activa. Executa a função “Close” da classe “CSerialPort”.

void enviar(char *dados)

Responsável pelo envio dos dados para a porta série. Recebe como argumento um apontador para o endereço do *buffer* que contém os dados a enviar. Executa a função “Write” da classe “CSerialPort”.

void ler(CString *on, CString *off, CString *portadora)

Responsável pela interpretação da informação enviada pelo microcontrolador. Os campos são interpretados e armazenados no *array* respectivo (passado como argumento).

void nr_porta_serie()

Responsável pela leitura do registo do Windows que suporta a informação acerca da(s) porta(s) COM existente(s) no computador (COM físicas ou virtuais). Esta função imprime os dados lidos na linha de comandos.

7.6.4 **Função *main***

A função *main* é a responsável pela execução de cada uma das funcionalidades da aplicação consoante os parâmetros passados através da linha de comandos. Esta função serve-se da classe "CSerial_Control".

Os parâmetros de controlo inseridos na linha de comandos podem apresentar quatro prefixos diferentes: "receber", "enviar", "EEPROM" e "ler_com", cada um deles é seguido de uma mensagem constituída por N elementos, de acordo com o tipo de prefixo.

7.6.4.1 **Prefixo "receber"**

A mensagem é constituída da seguinte forma:

Argumentos	**argv[1]**		**argv[2]**
Campo	<Prefixo>	<espaço>	<COM>

Fig. 84 Constituição da mensagem "receber"

Esta mensagem activa o modo de leitura da informação enviada pelo microcontrolador. Este processo envolve quatro passos essenciais: interpretação da informação enviada pelo microcontrolador, detecção do comprimento da trama, detecção do período efectivo da portadora, agrupamento dos dados da trama e portadora e envio para a linha de comandos.

- **Interpretação da informação enviada pelo microcontrolador**

A interpretação é feita através da execução da função "ler" presente na classe "CSerial_Control". Este processo de interpretação encontra-se representado no fluxograma da figura seguinte.

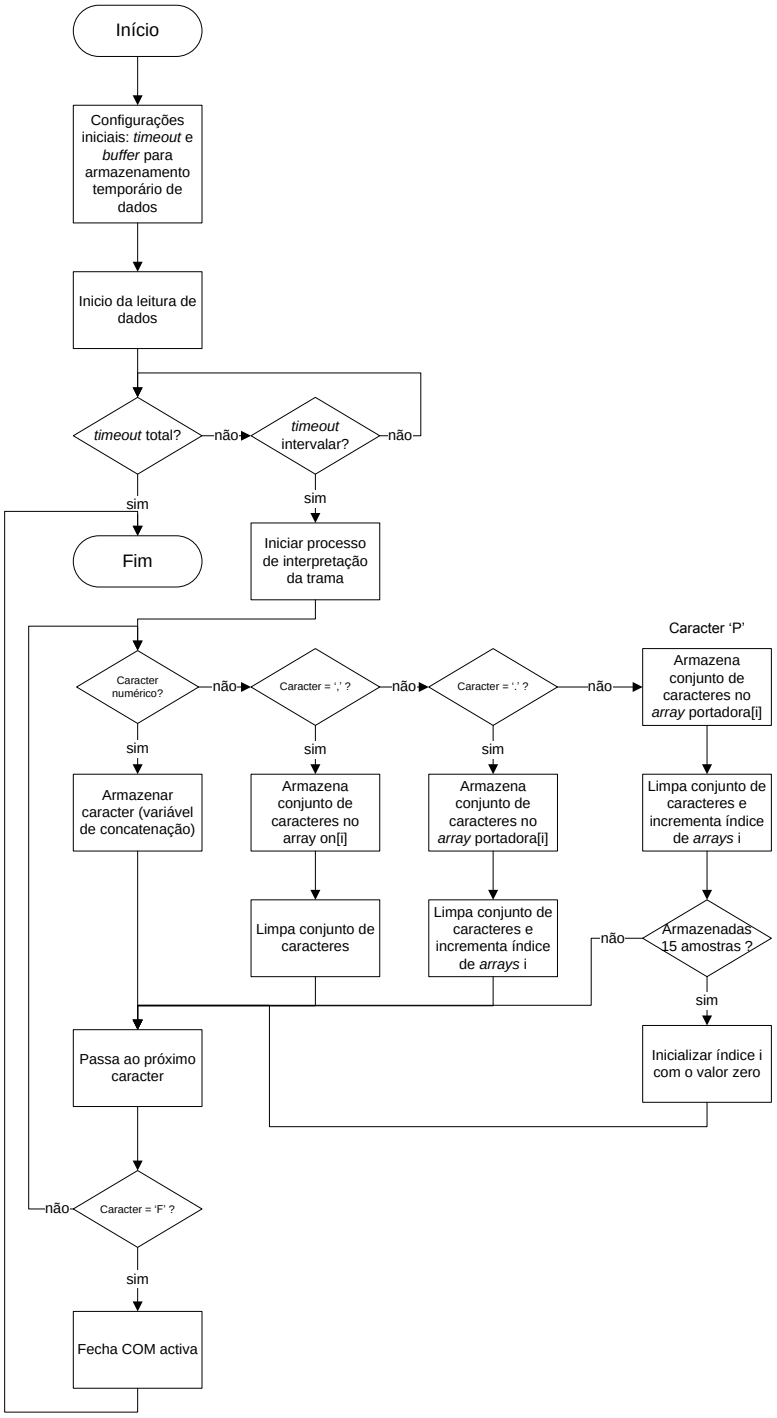


Fig. 85 Execução da função "ler"

- **Detecção do comprimento da trama**

Depois de todos os parâmetros IR terem sido armazenados nos respectivos *arrays* será agora efectuada a verificação acerca do comprimento da trama, no sentido de identificar o número de pares (on e off) relevantes. Pela análise das características de alguns protocolos verificou-se que o tempo "morto" associado à transmissão de um *bit* é

geralmente inferior a 2 ms, à excepção do tempo associado ao "start bit", como tal, tomando como referência esse valor sabe-se que ao existir um tempo superior (conteúdo do *array* "off") significa que se está perante o inicio de uma retransmissão e como tal detecta-se o final da trama. O tempo "morto" associado ao "start bit" é considerado uma excepção e não entra na verificação. No fluxograma da figura seguinte é possível analisar este processo de detecção do comprimento da trama. O valor final do índice *i* representará o número de pares (on e off), relevantes.

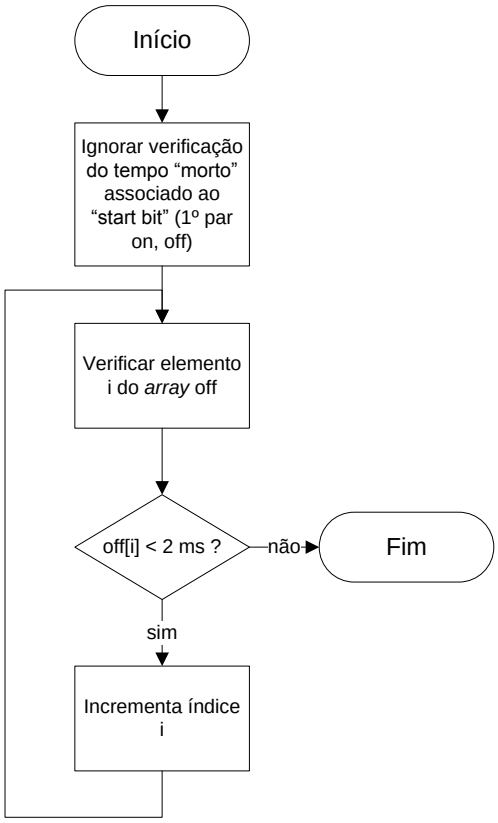


Fig. 86 Detecção do comprimento da trama

- **Detecção do período efectivo da portadora**

O microcontrolador adquire 15 amostras do período da portadora, como tal, é agora necessário chegar a um valor efectivo com base nessas amostras. Este processo é feito através da verificação da "moda" do conjunto de amostras, o valor mais frequente será então o valor final do período da portadora. O fluxograma da figura seguinte ilustra todo o processo de execução até se encontrar o valor efectivo do período.

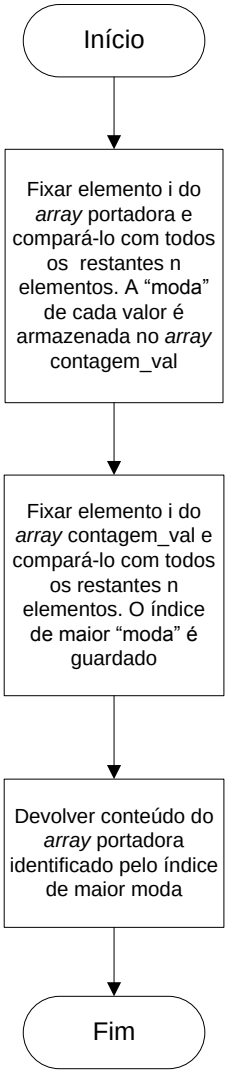


Fig. 87 Detecção do período efectivo da portadora

- **Envio dos dados agrupados para a linha de comandos**

A informação depois de correctamente agrupada será enviada para a linha de comandos, para posteriormente ser lida pela aplicação Flash. A informação é enviada com o seguinte formato:

$$\mathbf{Nº.\,pares\ on,off}\,|\quad \mathbf{Nº.\,ciclos_1}\,|\quad \mathbf{T_{off1}}\,|\quad \dots\quad |\quad \mathbf{T_{portadora}}|$$

Fig. 88 Formato da informação enviada para a linha de comandos

Esta informação será interpretada por um *script* PHP e só depois será enviada à aplicação Flash. O campo "Nº. pares on, off" é meramente auxiliar, é usado pelo *script* PHP no tratamento dos restantes dados.

A informação relativa ao tempo no qual existe modulação é enviada na forma de "Nº. ciclos". Optou-se por este formato de modo a ser mais fácil a reprodução do sinal por parte do módulo de emissão, afastando a necessidade de cálculos extra. Desta forma sabe-se que é necessário enviar X ciclos a uma determinada frequência. O cálculo para a obtenção do número de ciclos é feito da seguinte forma:

$$Nº.ciclos_N = \frac{T_{on\,N}}{T_{portadora}} \quad (9)$$

O valor do cálculo é arredondado e apresentado como um inteiro.

A informação relativa ao tempo "morto" é enviada directamente com o valor correspondente no *array* off.

Em relação à portadora é enviada a informação acerca do seu período. Esse valor será depois usado pela aplicação Flash nos cálculos respectivos, nomeadamente na obtenção dos 25% e 75% do período da portadora, para armazenamento na base de dados.

7.6.4.2 **Prefixo "enviar"**

A mensagem é constituída da seguinte forma:

Argumentos	**argv[1]**		**argv[**2**]**		**argv[**3**]**		**argv[**4**]**
Campo	<Prefixo>	<espaço>	<COM>	<espaço>	<Endereço>	<espaço>	<Dados>

Fig. 89 Constituição da mensagem "enviar"

Esta mensagem activa o modo de envio via RF da informação lida da base de dados. O campo <Endereço> corresponde ao endereço do módulo de emissão IR para o qual se pretende enviar os dados. Este campo é constituído por 3 caracteres. O campo <Dados> corresponde aos dados lidos da base de dados, agrupados segundo um formato próprio. Os restantes campos são os básicos, já conhecidos.

O formato da informação presente no campo <Dados> é o seguinte:

$\frac{1}{4}T_{portadora}$	**P**	$\frac{3}{4}T_{portadora}$	**P**	$\mathbf{N^{o}.ciclos_1}$,	$\mathbf{T_{off1}}$.	…	**F**

Fig. 90 Formato da informação presente no campo<Dados>

A informação é estruturada desta forma por intermédio de um script PHP, invocado pela aplicação Flash.

7.6.4.3 **Prefixo "EEPROM"**

A mensagem é constituída da seguinte forma:

Argumentos	**argv[1]**		**argv[**2**]**		**argv[**3**]**
Campo	<Prefixo>	<espaço>	<COM>	<espaço>	'W'<Endereço>'/'

Fig. 91 Constituição da mensagem "EEPROM"

Esta mensagem activa o modo de gravação do endereço na EEPROM do módulo de emissão em configuração. O campo <Endereço> é enviado com um determinado formato, esta é a forma de se distinguir o envio de um endereço de um envio de informação relativa a uma trama. O módulo de emissão sabe então que a partir do momento em que recebe o caracter 'W' toda a restante informação até ao caracter '/' diz respeito ao endereço a armazenar na sua EEPROM.

7.6.4.4 **Prefixo "ler_com"**

A mensagem é constituída da seguinte forma:

Argumentos	**argv[1]**
Campo	<Prefixo>

Fig. 92 Constituição da mensagem "ler_com"

Esta mensagem activa a leitura da(s) porta(s) COM existente(s) no computador. Este processo é feito através da função "nr_porta_serie" presente na classe "CSerial_Control". Basicamente, o que esta função faz é um acesso ao registo do

Windows que contém toda essa informação, mais propriamente o registo "HKEY_LOCAL_MACHINE\HARDWARE\ DEVICEMAP\SERIALCOMM".

7.7 Módulo de Emissão IR

O módulo de emissão IR é o responsável pela emissão de um determinado comando IR, para tal, desempenha duas funções essenciais: interpretação da trama recebida via RF com a informação relativa ao comando a emitir e a reprodução do sinal IR com base nos valores obtidos. Estes processos são executados por intermédio de um microcontrolador PIC. Para que o comando IR seja emitido, o endereço presente na trama de transmissão RF, deverá corresponder ao endereço do módulo e portanto numa primeira fase é feita essa verificação. Numa segunda fase, após ter sido feita a interpretação de toda a informação é então emitido o comando. Os blocos associados a este módulo são os seguintes.

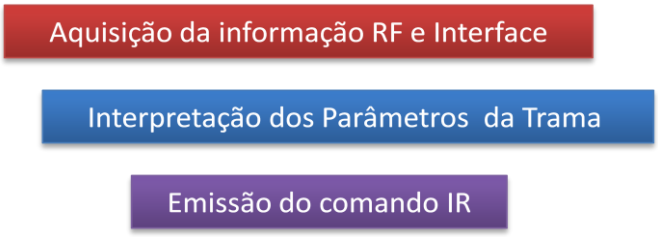


Fig. 93 Blocos constituintes do módulo de emissão IR

7.7.1 Aquisição da Informação RF e Interface

A informação RF é recebida através do módulo XBee. Em termos de interface com o microcontrolador não existiria nenhuma limitação em termos de conexão, uma vez que o microcontrolador só recebe informação enviada pelo XBee, contudo, a pensar um pouco no futuro foi introduzida a possibilidade de envio de dados ao PC via RF. Esta nova funcionalidade requer uma limitação do sinal que chega ao pino "DIN" do XBee e portanto o interface é idêntico ao da Fig. 81, porém o sinal de entrada não será o "Sinal FTDI" mas sim o sinal "TX PIC". A figura seguinte ilustra de uma maneira geral este interface entre o microcontrolador e o módulo XBee.

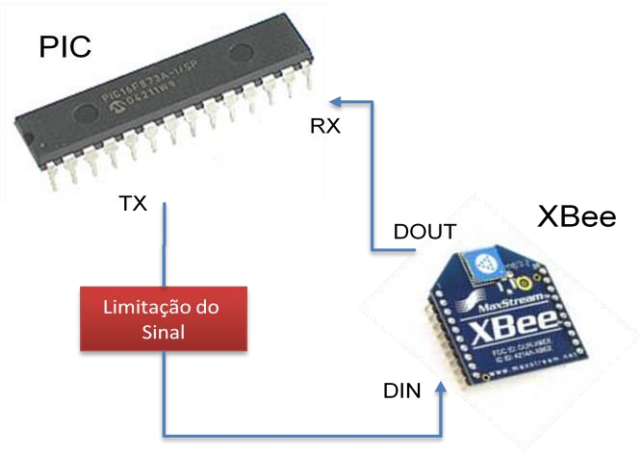


Fig. 94 Interface entre o microcontrolador e o módulo XBee

A alimentação do módulo é feita através de uma fonte externa (máx. 12 V DC (*Direct Current*)). São necessários dois níveis de tensão distintos, 3,3 V DC para a alimentação do módulo XBee e 5 V DC para a alimentação do microcontrolador. O nível de tensão de 3,3 V é obtido através do MAX604, à semelhança do módulo de aquisição. O nível de tensão de 5 V por sua vez é obtido através de um circuito integrado equivalente, o MAX603. A diferença entre estes dois integrados reside apenas no valor da tensão de saída, sendo que os componentes externos a adicionar são exactamente iguais em ambos. O esquema de ligações foi apresentado anteriormente, encontra-se ilustrado na Fig. 80.

7.7.2 Interpretação dos Parâmetros da Trama

O microcontrolador responsável pela interpretação da trama recebida via RF é o mesmo usado no módulo de aquisição, o PIC 16F876A.

A informação recebida poderá ser de dois tipos: endereço, para a gravação na EEPROM ou trama, para a emissão IR. A recepção de cada caracter constituinte da trama desencadeia uma interrupção. É com base na função associada a essa interrupção que todos os parâmetros serão interpretados. Este processo de interpretação encontra-se ilustrado no fluxograma da figura seguinte.

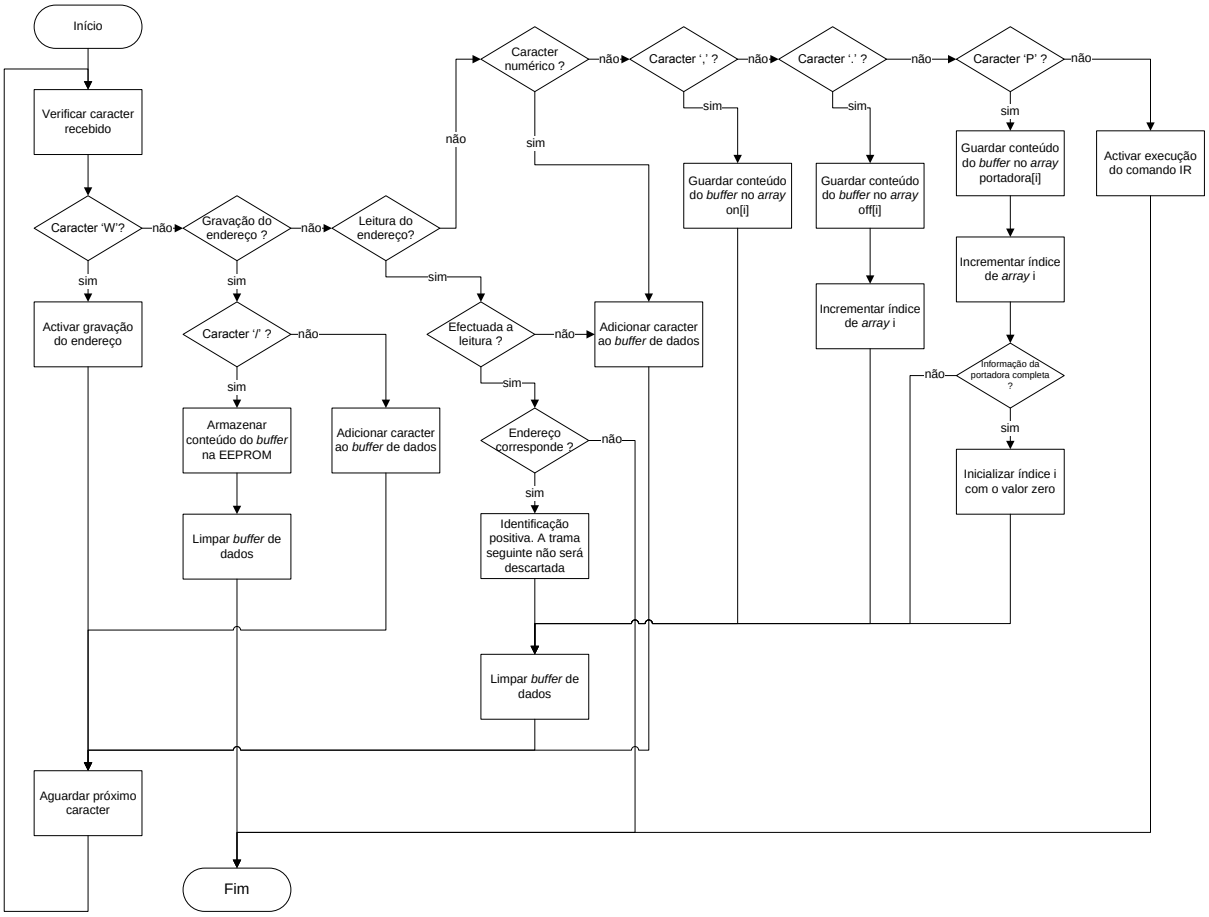


Fig. 95 Interpretação dos parâmetros recebidos via RF

7.7.3 **Emissão do Comando IR**

Após a interpretação da trama recebida, estão reunidos todos os parâmetros essenciais: período da portadora (25% *duty cycle* + 75% tempo "morto"), número de ciclos correspondentes ao intervalo de tempo no qual existe modulação e também os intervalos de tempo onde não existe sinal (tempo "morto"). O próximo passo será então a emissão do comando IR correspondente. Este processo é executado pela função *main*. Na Fig. 96 é possível analisar o fluxograma de execução deste processo. É de salientar que cada comando IR é emitido mais do que uma vez a cada execução, isto porque, após a realização de alguns testes verificou-se que a emissão de apenas uma trama não resultava numa actuação positiva nos aparelhos IR testados, como tal, optou-se por um ciclo de execução com repetições (5 emissões). Chegou-se então à conclusão que ao premir o botão de um telecomando IR é emitido um conjunto de tramas com um determinado ciclo de repetição. O valor escolhido é suficiente e não permite que exista uma execução dupla do comando em questão.

O sinal de saída é gerado num pino do microcontrolador, ao qual se encontra associado um circuito de emissão, tal como indicado na Fig. 97.

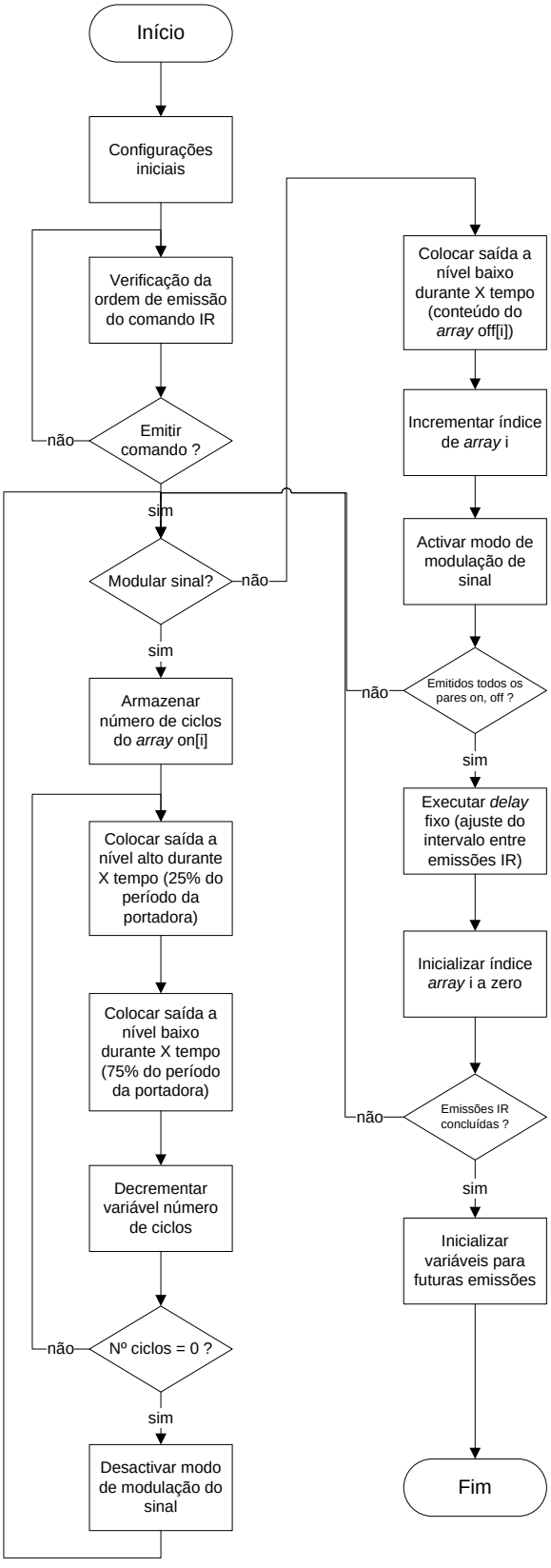


Fig. 96 Processo de emissão do comando IR

- **Circuito de Emissão IR**

O circuito de emissão é controlado pelo microcontrolador. O seu objectivo é fazer a conversão de um sinal eléctrico num sinal infravermelho para que seja possível controlar o respectivo aparelho IR.

O circuito é bastante simples como se pode ver na Fig. 97. Basicamente, é constituído por um transístor bipolar 2N2222 e um LED IR usual. O transístor funciona como um auxiliar do pino de controlo do microcontrolador, tornando possível a drenagem de uma corrente suficiente de modo a possibilitar um alcance razoável do sinal IR emitido, isto porque o alcance do sinal está directamente relacionado com a corrente directa que percorre o emissor IR. O LED IR tem um funcionamento semelhante a um LED normal, porém em vez de emitir luz visível emite luz infravermelha. Foi usado um LED IR da marca Kingbright, com um ângulo de emissão de 30º.

Regra geral os LEDs IR apresentam um valor médio de corrente máximo admissível muito baixo, neste caso na ordem dos 50 mA, o que conduz a uma emissão pouco eficiente, por outro lado o valor de pico máximo é elevado, neste caso na ordem dos 1,2 A (Ampere) (tendo em conta um curto período de tempo: *duty cycle* de 100 µs como referência), proporcionando assim um aumento do alcance de emissão. Como se sabe os sinais emitidos serão modulados a uma determinada frequência com um *duty cycle* fixo de 25%, ora tendo em conta que a frequência mais baixa será de 36 kHz, isto significa que na pior das hipóteses tem-se um tempo de "LED ON" na ordem dos 7 µs e portanto não existe qualquer problema face ao tempo imposto na emissão de uma corrente de pico elevada. Mais informações acerca deste LED IR podem ser encontradas no *datasheet* presente em [68].

O circuito foi dimensionado para uma corrente de pico na ordem dos 350 mA, um valor muito aquém dos limites máximos impostos quer pelo transístor quer pelo LED IR, contudo é um valor suficiente, permitindo já distâncias de emissão bastante satisfatórias e aumentado também o tempo de vida do LED.

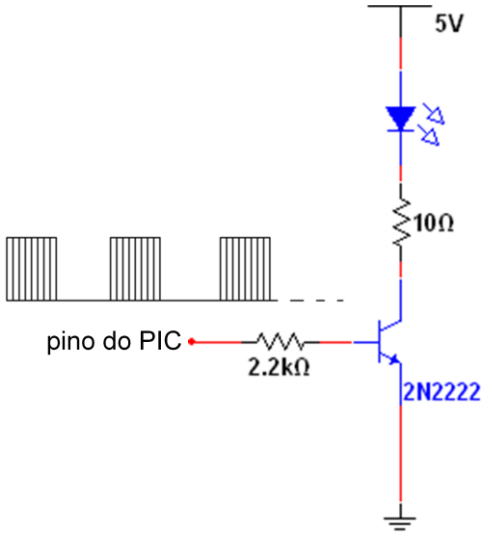


Fig. 97 Circuito de emissão IR

Capítulo VIII

8. Integração no Sistema de Domótica Principal

8.1 Introdução

O hardware de controlo de dispositivos IR apesar de ser totalmente independente do sistema de PLCs não significa que o seu controlo tenha de ser também independente, como tal é importante integrar esse controlo na aplicação Flash. A grande diferença de um dispositivo IR em relação a qualquer outro dispositivo reside essencialmente na sua configuração, existindo uma área especialmente dedicada para esse efeito (integrada na área de configuração global). Em termos de interacção para actuação segue-se a mesma linha, no fundo o que muda é naturalmente a consola de interacção e essencialmente o processo de comunicação existente em *background*, o qual é ligeiramente mais complexo. No decorrer deste capítulo será apresentado todo o processo quer de configuração quer de actuação dos dispositivos IR. Para mais informações acerca da configuração de dispositivos IR poderá ser consultado o manual presente em anexo.

8.2 Configuração de Dispositivos IR (integrado em "configuracao.swf")

A configuração dos dispositivos IR é dividida em duas fases. A primeira fase consiste no envio do endereço via RF para gravação na EEPROM do módulo de emissão em configuração. A segunda fase consiste na listagem e gravação dos comandos IR associados ao dispositivo em questão. Todo este processo de configuração é efectuado na "área de configuração" no submenu "IR config". Na figura seguinte é possível ver o aspecto gráfico deste submenu de configuração.

Fig. 98 Configuração de dispositivos IR

- **Princípio de Execução**

A configuração IR pressupõe uma introdução prévia do dispositivo no respectivo compartimento, como tal, o primeiro passo será obter uma listagem de todos os dispositivos IR existentes no sistema, identificando aqueles que já foram configurados (no caso de não ser a primeira execução). Considera-se que um dispositivo já foi configurado quando os seus dados de portadora são diferentes de "NULL" (valor por defeito), dado que esse é o último campo a ser preenchido quando este é configurado. Uma vez obtida a listagem de dispositivos é necessário obter também a identificação de todas a portas COM existentes no PC, para que o utilizador possa seleccionar a porta correcta para comunicação com o hardware (módulo de aquisição). Tanto as listagens de dispositivos como de portas COM são apresentadas por intermédio de *ComboBox*. Posto isto poderá seleccionar-se o dispositivo pretendido e iniciar a sua configuração. Os processos envolvidos na configuração de um novo dispositivo IR encontram-se ilustrados no fluxograma da Fig. 101. As funções e *script*s PHP envolvidos nestes processos serão apresentados em seguida.

8.2.1 Listagem dos dispositivos e identificação da(s) porta(s) COM

- **Principais Funções Actionscript**
  - **Globais**

ler_disp_IR_COM() {}

Responsável pela listagem dos dispositivos IR existentes no sistema, identificando aqueles já configurados, e também listagem da(s) porta(s) COM existente(s). Para esse fim é invocado o *script* "ler_disp_IR.php". Os dados lidos da base de dados são inseridos na *ComboBox* respectiva.

gravar_COM(com:Number) {}

Responsável pela gravação da porta COM seleccionada pelo utilizador através da *ComboBox* (identificação passada como argumento). Para esse fim é invocado o *script* "guardar_com.php", usado também no módulo "hardware.swf", como foi visto no capítulo 6 (secção 6.5). O parâmetro passado como argumento poderá conduzir a duas situações: gravação ou actualização na tabela "porta_com".

- **Locais**

- Manipulação da *ComboBox*

cbListener1.change = function(evt_obj:Object) {}

Responsável pelo início da configuração do dispositivo seleccionado pelo utilizador através da *ComboBox*. Caso o dispositivo já tenha sido configurado irá aparecer um "visto" informativo.

cbListener2.change = function(evt_obj:Object) {}

Responsável pela execução da função "gravar_COM", sempre que existe a selecção de um valor na *ComboBox*.

8.2.2 Envio do endereço via RF

A gravação do endereço no módulo de emissão em questão é iniciada por intermédio do botão ilustrado na figura seguinte.

Fig. 99 Ordem de envio para a gravação do endereço na EEPROM

- **Principais Funções Actionscript**
  - **Locais**

Botão "EEPROM" → this.onRelease = function() {}

Responsável pela ordem de envio para gravação do endereço no módulo de emissão em configuração. Para esse fim invoca o *script* "gravar_EEPROM.php".

8.2.3 Listagem e gravação dos comandos IR

A ordem para a leitura do comando IR de cada dispositivo é dada através do clique num telecomando de televisão virtual, o qual apresenta os botões essenciais (Fig. 100). O facto desta interacção ter como base uma televisão não significa que não seja possível gravar comandos provenientes de um telecomando de qualquer outro aparelho, contudo como a ideia é mostrar a possibilidade de controlo de alguns dispositivos IR, optou-se por utilizar como exemplo a televisão, no entanto para que se possa dar a evolução para o controlo de outros aparelhos basta criar o telecomando virtual respectivo, tendo em conta claro que o aparelho se encontra dentro dos critérios impostos pelo hardware de leitura.

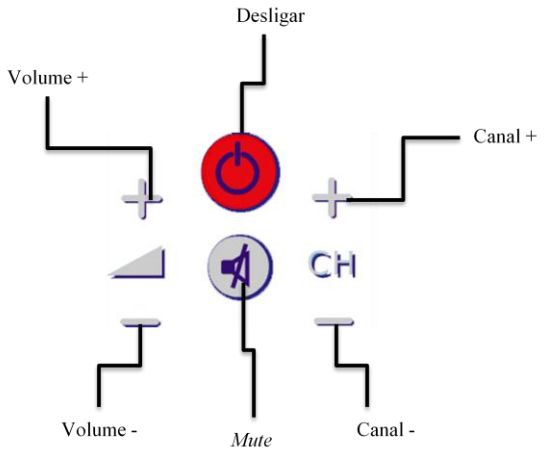


Fig. 100 Telecomando virtual para gravação de comandos IR (Televisão)

Sempre que existe a leitura de um comando será feita uma listagem dos dados recebidos. Esta listagem é meramente informativa, para que seja possível ter uma ideia dos tempos envolvidos num determinado comando. Os dados a apresentar são os tempos nos quais existe modulação, os tempos "mortos" e a frequência da portadora. Como se sabe, a informação associada aos tempos de modulação bem como à portadora,

é enviada à aplicação Flash na forma de "nº de ciclos" e período, respectivamente, como tal, terão de ser executados cálculos auxiliares.

- **Cálculo do tempo associado à modulação**

O tempo de cada modulação é calculado com base no período da portadora, esse cálculo é feito da seguinte forma:

$$T_N(\mu s) = N^{\underline{o}}.ciclos_N \times T_{portadora}(\mu s) \quad (10)$$

- **Cálculo da portadora**

O processo envolvido no cálculo da frequência da portadora é ligeiramente mais complexo, não devido ao cálculo em si, mas sim pelo facto do valor obtido não corresponder a um standard. Como tal é necessário efectuar um processo de comparação de modo a obter um valor de frequência standard. Tendo em conta as limitações do hardware sabe-se que apenas existem três valores standard, 36 kHz, 38 kHz e 40 kHz.

O primeiro passo é o cálculo da frequência, o qual é feito da seguinte forma:

$$f(kHz) = \frac{1}{T_{portadora}(\mu s)} \times 1000 \quad (11)$$

Uma vez obtido o valor da frequência é necessário então verificar se esse valor corresponde a um standard. Sabe-se que o valor $T_{portadora}(\mu s)$ presente no cálculo da frequência é um valor arredondado. Por exemplo para uma frequência de 36 kHz o valor do período equivale a aproximadamente 27,78 µs. Após o arredondamento obtém-se um número inteiro igual ou inferior ao valor especificado, neste caso ficaria então 27 µs, ou seja, $36\ kHz \rightarrow T_{portadora} = 27\ \mu s$. Posto isto, ao ser efectuado o cálculo da frequência será obtido um valor superior a 36 kHz, contudo esse valor é ainda inferior a 38 kHz, desta forma o termo de comparação para a obtenção de um valor standard é feito da seguinte forma:

$$\begin{cases} 36\ kHz \leq f < 38\ kHz\,, & f = 36\ kHz \\ 38\ kHz \leq f < 40\ kHz\,, & f = 38\ kHz \\ f \geq 40\ kHz\,, & f = 40\ kHz \end{cases} \quad (12)$$

- **Principais Funções Actionscript**
  - **Globais**

ler_trama() {}

Responsável pela execução do processo de leitura e posterior listagem de toda a informação relacionada com o comando em questão. Com base nos dados adquiridos efectua os cálculos respectivos. O processo de leitura é iniciado por intermédio do *script* "receber_IR.php".

guardar_portadora(portadora:Number, nome:String,) {}

Responsável pelos cálculos associados à portadora passada como argumento (*duty cycle* 25% + 75 % tempo "morto") e gravação desses dados na base de dados. Esta função é invocada pela função "guardar_trama" quando for feito o armazenamento do último comando IR. A gravação da portadora é feita por intermédio do *script* "guardar_portadora.php". O parâmetro nome identifica univocamente o dispositivo IR na tabela a manipular.

guardar_trama(nome:String,comando:String,impulsos:Array,
tmp_off:Array) {}

Responsável pelo armazenamento na base de dados de toda a informação relativa ao comando IR adquirido. Para esse fim invoca o *script* "guardar_trama.php". Uma vez armazenados todos os dados será então invocada a função "guardar_portadora".

eliminar_incompleto(nome:String) {}

Responsável por eliminar os dados de um dispositivo IR configurado incorrectamente. É executada sempre que se inicie uma nova configuração sem que uma anterior tivesse sido completada correctamente. Esta função evita assim a redundância de informação relativamente a tramas já gravadas na base de dados. Esta acção é feita por intermédio do *script* "eliminar_incompleto.php".

  - **Locais**

Botão "visto" → visto_IR.onRelease = function() {}

Responsável pela ordem de gravação de dados na base de dados. Para esse fim executa a função "guardar_trama".

Botões telecomando virtual → <instância>.onRelease = function() {}

Responsável pela ordem de inicialização de todo o processo envolvido na leitura do comando IR respectivo. Para esse fim invoca a função "ler_trama".

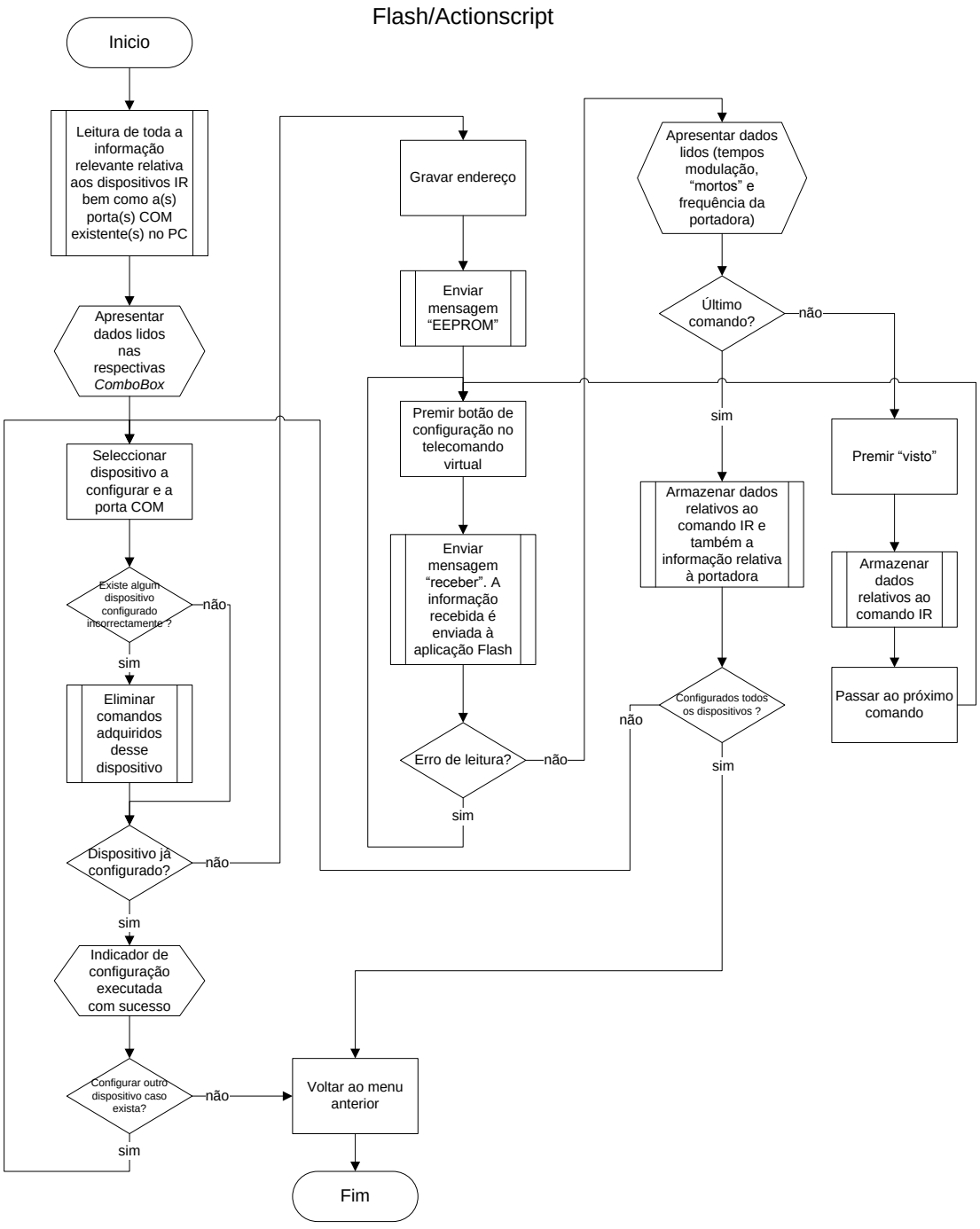


Fig. 101 Processo de configuração de dispositivos IR

8.2.4 **Interface com a Aplicação C/C++**

O interface com a aplicação C/C++ é feito por intermédio da linha de comandos. O acesso à linha de comandos é feito através de PHP com base na função "shell_exec", a qual torna possível tanto o envio como a recepção de informação através da linha de comandos. A implementação desta interacção PHP←→Aplicação C/C++ é feita da seguinte forma:

<PHP>

```
<?php

$acesso = "receber";          //definição do tipo de acesso à aplicação, neste caso acesso para leitura de um comando IR
$com = '5';                   //definição da porta COM para posterior comunicação com o hardware

$shell = explode("|", shell_exec("porta_serie.exe $acesso $com") );   //execução da aplicação c/c++ com
                                                                      //passagem de parâmetros para dar inicio ao acesso respectivo
                                                                      //a função "explode" é usada para a separação dos dados, os quais estão separados por ´|´

echo $shell[0];               //recepção da informação enviada pela aplicação, imprime "dado1"
echo $shell[1];               //imprime "dado2"
…
?>
```

<C/C++>

```
…
void main(int argc, char **argv) {                                    //função main

        if(!strcmp(argv[1],"receber"))        //interpretação do parâmetro "acesso" passado como argumento
                printf("dado1|dado2…");       //imprime informação na linha de comandos com um
                                              //determinado formato para posterior separação de dados
…
}
```

8.2.5 **Scripts PHP**

ler_disp_IR.php

Responsável pela leitura de toda a informação relevante da tabela "dispositivos_IR". Para além disso executa também a aplicação C/C++ para leitura da(s) porta(s) COM.

guardar_com.php

Este *script* é comum ao módulo "hardware.swf".

gravar_EEPROM.php

Responsável pelo envio do endereço via RF para gravação na EEPROM do módulo de emissão em configuração. Executa a mensagem com prefixo "EEPROM".

receber_IR.php

Responsável pelo execução da mensagem com prefixo "receber" enviando posteriormente os dados recebidos à aplicação Flash.

guardar_portadora.php

Responsável pelo armazenamento da informação relativa à portadora na tabela "dispositivos_IR".

guardar_trama.php

Responsável pelo armazenamento na tabela "trama_IR" de todos os dados relativos ao comando IR adquirido.

eliminar_incompleto.php

Responsável por eliminar os dados presentes na tabela "trama_IR" relativos a um dispositivo IR configurado incorrectamente.

8.3 Actuação em Dispositivos IR ("sistema.swf")

O processo de interacção para o controlo de dispositivos IR é bastante semelhante ao processo dos demais dispositivos, a diferença é essencialmente em termos de consola de interacção e comunicação com o dispositivo, como já foi apresentado no capítulo 6 (secção 6.3.1). Nesta secção será feita uma abordagem no sentido de mostrar um pouco melhor todos os passos de execução, dado que este processo de comunicação envolve várias etapas.

- **Ordem de Envio**

A ordem de envio é dada pela consola de interacção com o dispositivo IR. O clique num dos botões disponíveis invoca um script PHP ("enviar_IR.php"), o qual terá a responsabilidade de aceder à base de dados de forma a adquirir toda a informação relevante para o envio da trama associada a esse comando. A informação adquirida será

organizada segundo o formato apresentado na Fig. 90 e anexada à mensagem com o prefixo “enviar” (Fig. 89). Na figura seguinte encontra-se ilustrado este primeiro passo.

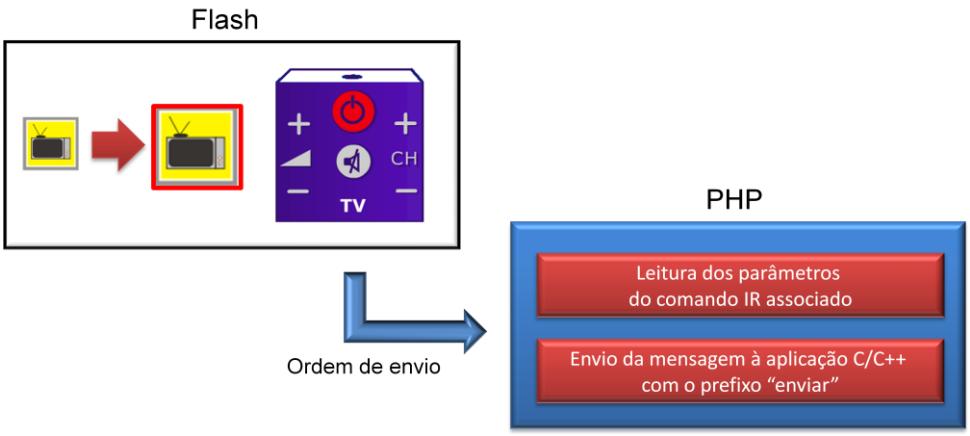


Fig. 102 Ordem de envio

- **Transmissão via RF**

A informação enviada através da linha de comandos por intermédio do *script* PHP é agora lida pela aplicação C/C++. A informação relevante a enviar ao módulo de emissão é apenas a informação presente nos campos <Endereço> e <Dados> (Fig. 89). O campo <Prefixo> indica o processo a implementar e o campo <COM> por sua vez indica qual a porta a usar na comunicação série para interface com o módulo XBee (BASE). O módulo XBee (BASE) fará posteriormente o envio da informação via RF. Na figura seguinte encontra-se ilustrado este segundo passo.

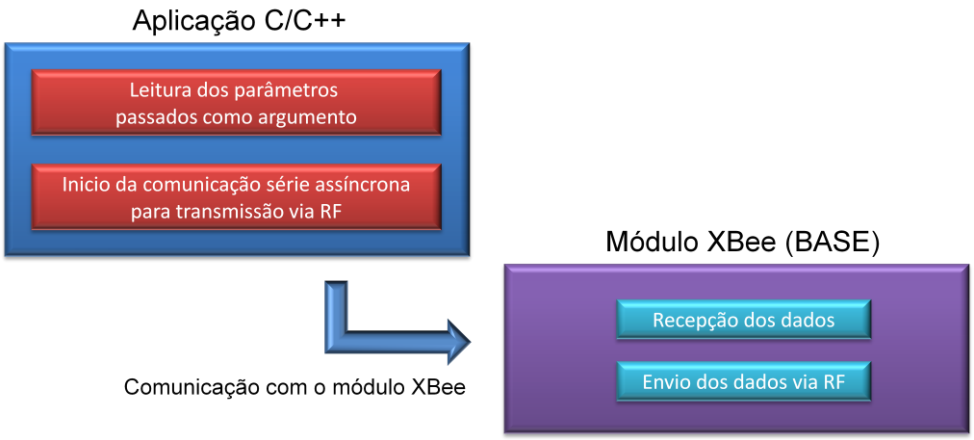


Fig. 103 Comunicação com o módulo XBee para o envio via RF

- **Emissão IR**

A informação transmitida via RF será recebida pelo módulo XBee (REMOTO) e enviada ao PIC para interpretação, no sentido de se recriar o comando IR a emitir. Partindo do princípio que a verificação do endereço foi positiva, será então iniciado o ciclo de emissão, estando portanto concluído todo o processo de emissão de um determinado comando IR. Na figura seguinte encontra-se ilustrado este último passo envolvido no controlo de dispositivos IR.

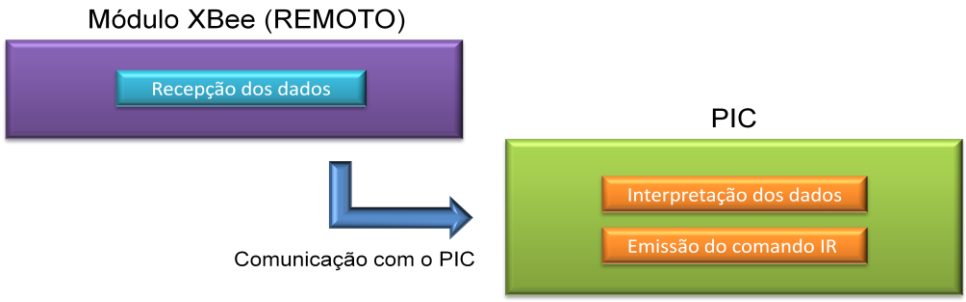


Fig. 104 Emissão IR

Capítulo IX

9. Resultados

O sistema não foi implementado numa habitação real, contudo para verificar a eficácia e funcionalidade do software de controlo, foram efectuados testes tendo em conta a configuração, monitorização e actuação em dispositivos tais como, lâmpadas, tomadas e estores. A nível do controlo de dispositivos por infravermelhos foi efectuado o teste em dispositivos baseados em três protocolos distintos, o Philips RC-5, o Sony SIRC e o Panasonic. Os resultados obtidos mostraram-se bastante satisfatórios como se poderá verificar.

9.1 Aplicação de Controlo e Monitorização

Para verificar o correcto funcionamento do software de controlo foram feitos os seguintes testes:

- Configuração
- Actuação
- Monitorização
- Fiabilidade
- Interacção utilizador-sistema

Todos estes testes foram realizados tendo em conta o funcionamento com dispositivos como lâmpadas, tomadas e estores tendo sido criado o *Ladder* necessário, integrando um controlo local simulado através dos botões presentes na consola do VISION 280 (tal como indicado no capítulo 4, secção 4.4 – *Case Study*). Este *Ladder* encontra-se dividido em controlo de dispositivos ON/OFF usuais, tais como lâmpadas e tomadas, e dispositivos especiais, caso dos estores. Foram configuradas duas saídas para lâmpadas, uma para tomadas e outra para estores. A tomada é simulada através do ON/OFF do relé do PLC. Os endereços correspondentes são os seguintes:

Dispositivo	Controlo Local	Endereço físico	Endereço de Memória
Lâmpada 1	Botão ‘1’	0	0
Lâmpada 2	Botão ‘2’	1	1
Tomada	Botão ‘4’	2	2
Estore	Botão ‘7’ (subir)/ ‘8’ (descer)	3 (subida) / 4 (descida)	3 / 4

Tabela 5 Endereços associados aos dispositivos testados

Com base nestes dados simulou-se a configuração de uma habitação de raiz, tendo em conta uma planta específica.

Ao longo de todo o desenvolvimento foram corrigidos alguns *bugs* encontrados, essencialmente a nível de transições entre objectos Flash, e portanto nesta fase o objectivo não seria tanto a detecção de *bugs* mas sim uma análise global do funcionamento do software de controlo. Não quer isto dizer que todos os *bugs* tenham sido encontrados, dado que é muito difícil verificar todas as combinações de execução de uma aplicação, porém, a nível de comunicação nunca existiram problemas de maior.

Em termos de área de configuração é dada uma grande liberdade ao utilizador em todo o processo de personalização da sua planta, desde livre posicionamento do identificador de compartimentos e dispositivos. O facto de ser necessário delinear a área de um compartimento para posterior *zoom* poderá ser um processo um pouco aborrecido, porém foi a melhor solução encontrada para implementar essa funcionalidade, a qual será bastante útil na actuação de dispositivos. A nível de gestão de utilizadores é possível executar todas as funções pertinentes, alteração/eliminação/introdução de utilizadores. Apesar desta possibilidade de múltiplos utilizadores, a execução será sempre igual para todos os utilizadores não existindo portanto execuções personalizadas.

A actuação nos dispositivos é feita com o simples “click” no dispositivo ou na consola do dispositivo em questão, obtendo-se um *feedback* imediato dessa actuação. O processo de actuação é sempre feito para um piso e respectivo compartimento, para além disso será sempre feita uma pré-selecção do dispositivo a controlar, com isto consegue-se um controlo mais específico e ao mesmo tempo obtém-se uma noção espacial do dispositivo a controlar, com base no *zoom in* do compartimento. Inicialmente implementou-se nesta área uma função de identificação dos compartimentos com dispositivos activos, seria bastante interessante, contudo com a introdução dos dispositivos IR acabou por ser retirada dado que não existia maneira de

verificar o estado de uma televisão, por exemplo, caso esta tivesse sido controlada localmente, e como tal esta funcionalidade já não faria sentido.

A monitorização desempenha um papel importante neste sistema, encontrando-se também em correcto funcionamento. O tempo estabelecido para o intervalo de leitura dos *Memory Bits* é suficiente, podendo inclusive ser aumentado.

Em termos de fiabilidade o teste centrou-se essencialmente na comunicação com o PLC, seja para leitura ou escrita de *Memory Bits*. O facto do hardware ser baseado em PLCs e a comunicação ser via *Ethernet* eleva bastante o grau de fiabilidade de todo o sistema, dado que, não se verificaram problemas quer a nível de actuação, quer a nível de leitura.

Para o teste de interacção utilizador-sistema nada melhor do que pedir a algumas pessoas para executar uma configuração de raiz com base nos dados anteriores, esperando pelo seu *feedback*. Posto isto, em termos de *login*/registo e controlo de dispositivos não se verificaram dificuldades, no entanto a nível de configuração verificou-se que o contacto com a área de configuração não era tão intuitivo, nomeadamente a nível de delineação da área do compartimento, o que por um lado é normal tendo em conta que essa área é ligeiramente mais complexa e é necessário percorrer alguns passos até configurar correctamente todos os parâmetros envolvidos. Esta área é mais dedicada ao configurador, contudo uma análise mais atenta do manual poderá ajudar o utilizador a ultrapassar esse problema. Duma maneira geral o *feedback* foi positivo, verificando-se que a parte mais relevante para o utilizador, o controlo de dispositivos, é bastante intuitiva.

Seria interessante analisar também a interacção remota com o sistema, com base em múltiplos clientes, contudo essa funcionalidade não foi implementada, estando apenas disponível o acesso via ambiente de trabalho remoto. Na figura seguinte é possível ver os dispositivos presentes no teste deste sistema.

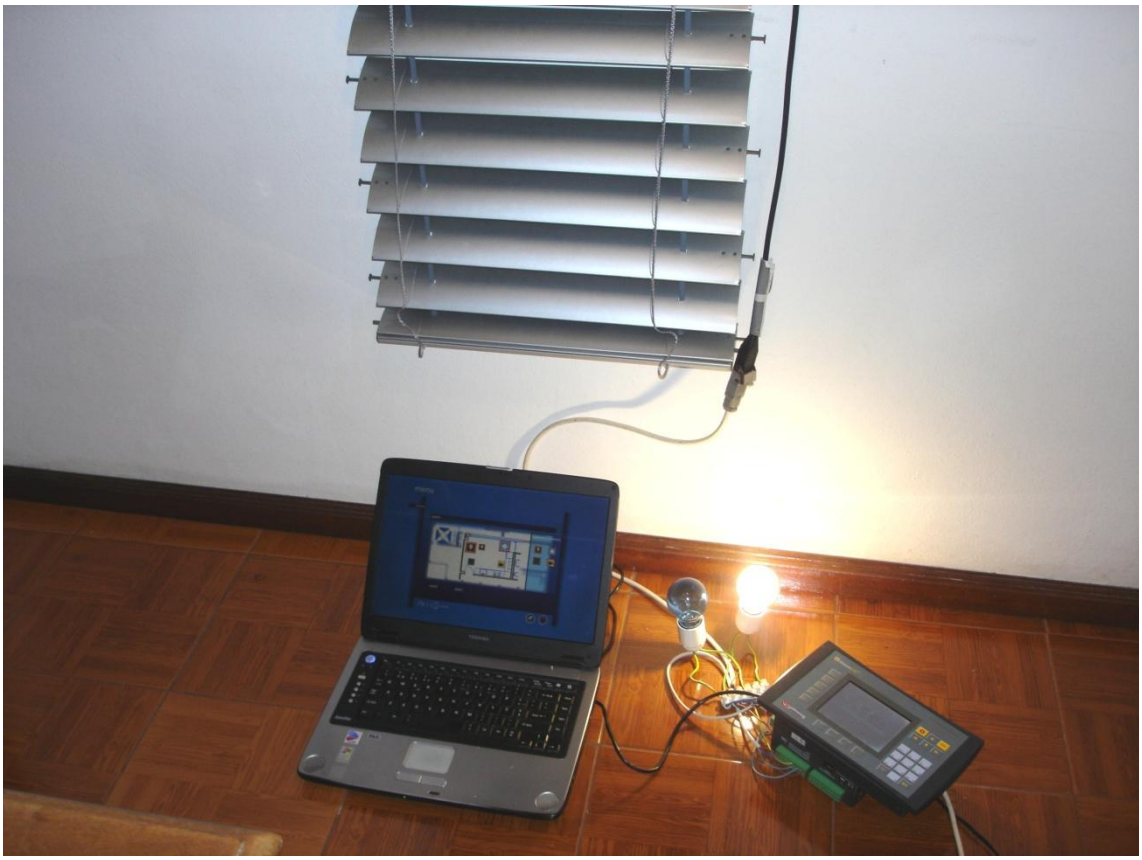
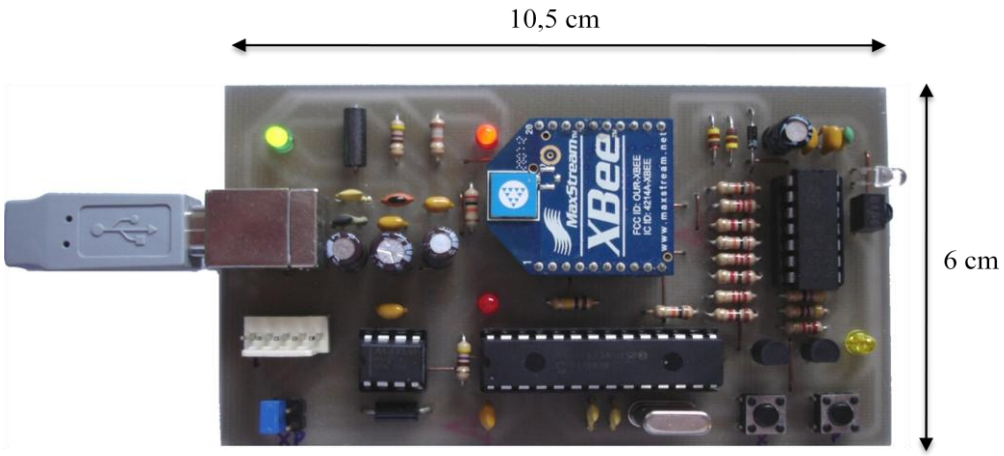


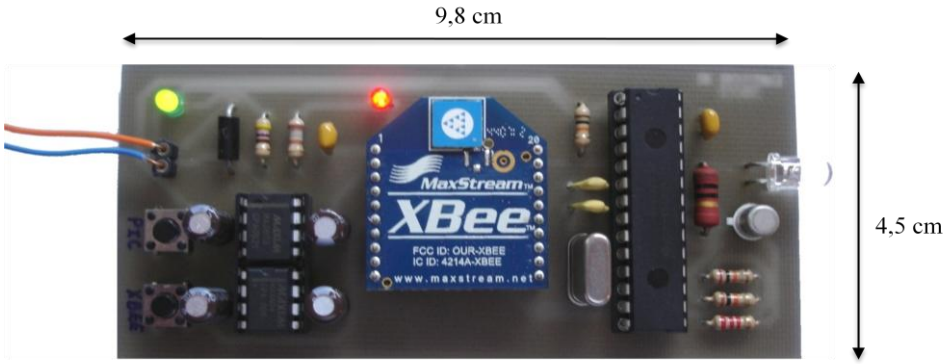
Fig. 105 Teste do sistema (PLC)

9.2 Subsistema de Controlo de Dispositivos por IR

O desenvolvimento de todo o processo de interpretação, gravação e emissão de comandos IR, associado aos módulos de aquisição e emissão IR, foi concretizado em duas PCBs, tal como ilustrado na Fig. 106. No sentido de verificar a eficácia deste subsistema foram efectuados também alguns testes, nomeadamente a nível da capacidade de leitura e emissão de comandos IR, identificando as suas limitações. A transmissão da informação via RF foi também foco de atenção, sendo analisada a possibilidade de erros de transmissão, dado que a transmissão em *broadcast* traz algumas limitações a nível de funcionamento do módulo XBee, tal como foi apresentado no capítulo 7, secção 7.5.4 (Transmissão RF).



(a) PCB do módulo de aquisição IR



(b) PCB do módulo de emissão IR

Fig. 106 PCBs para o controlo de dispositivos por infravermelhos

As PCBs implementam todas as funcionalidades descritas ao longo da tese, bem como algumas funcionalidades extra, nomeadamente a actualização do firmware do PIC por exemplo. O esquemático e *layout* das placas encontra-se disponível em anexo.

- **PCB do módulo de aquisição IR**

Esta PCB apresenta as seguintes características:

  - Comunicação com o PC via USB
  - Transmissão de dados via RF
  - Leitura de tramas IR de protocolos com portadoras compreendidas entre os 36-40 kHz
  - Configuração de parâmetros e actualização de firmware dos módulos XBee
  - Actualização de firmware do PIC
  - Alimentação via *bus* USB

- **PCB do módulo de emissão IR**

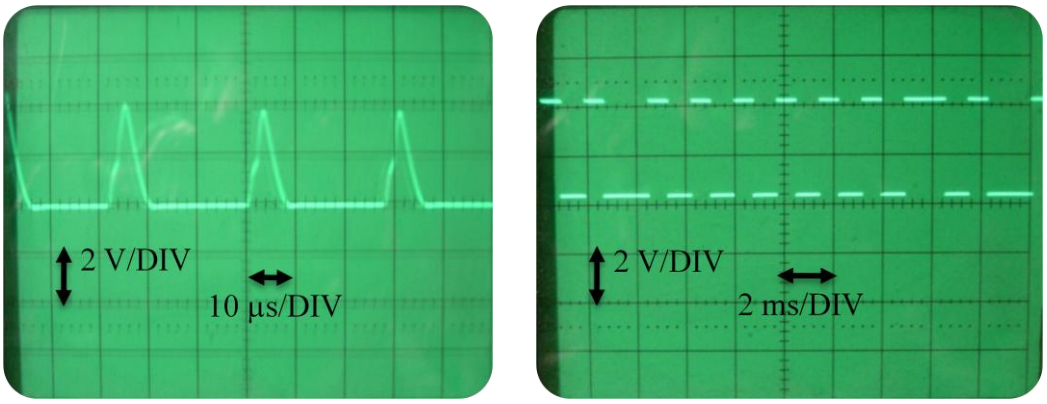
Esta PCB apresenta as seguintes características:

  - Recepção e interpretação da informação enviada via RF
  - Discernimento de informação
  - Emissão IR
  - Alimentação via fonte externa (máx. 12 V DC)
  - Protecção contra troca de polaridade

Com vista a verificar as funcionalidades acima mencionadas, essencialmente a nível de leitura e transmissão de informação entre módulos, foram feitos então os devidos testes com os protocolos Philips RC-5, Sony SIRC e Panasonic.

9.2.1 **Leitura de Comandos IR**

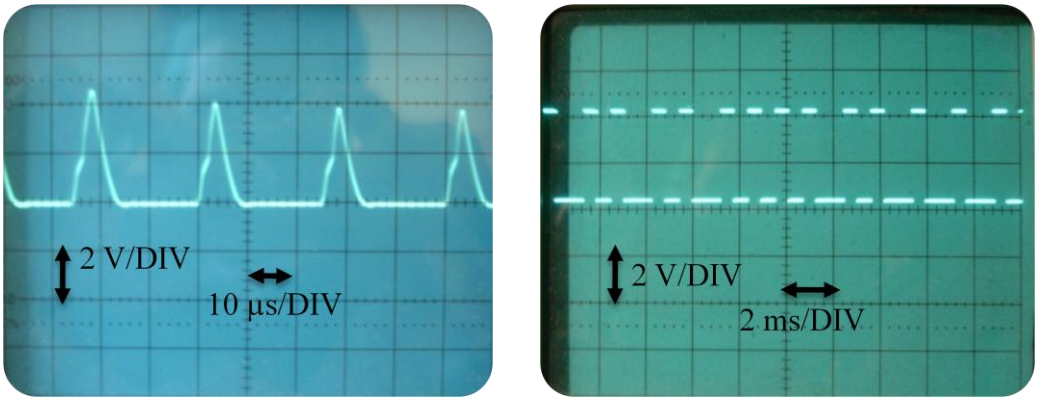
- **Protocolo Philips RC-5**



(a) Sinal da Portadora **(b) Sinal da Trama**

Fig. 107 Sinais obtidos na leitura de um comando IR Philips RC-5

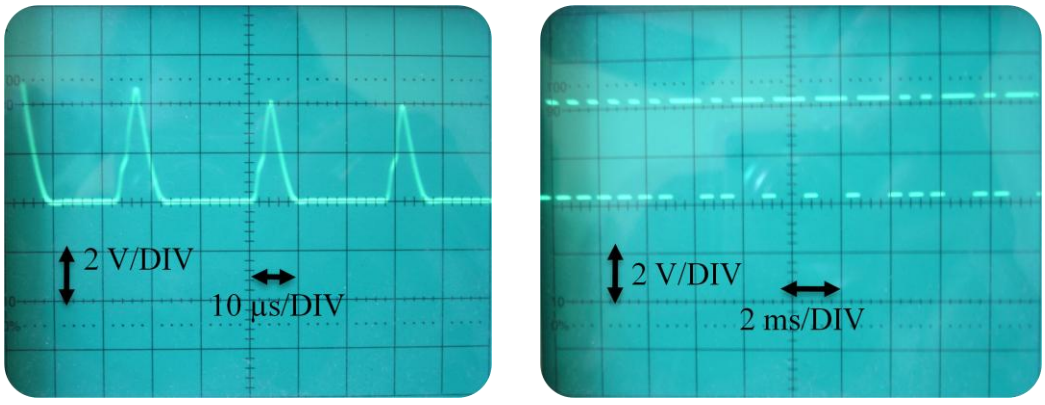
- **Protocolo Sony SIRC**



(b) Sinal da Portadora **(b) Sinal da Trama**

Fig. 108 Sinais obtidos na leitura de um comando IR Sony SIRC

- **Protocolo Panasonic**



(c) Sinal da Portadora **(b) Sinal da Trama**

Fig. 109 Sinais obtidos na leitura de um comando IR Panasonic

Os sinais apresentados comprovam a eficácia do sistema de aquisição (sinais compreendido entre 36-40 kHz), contudo em termos de interpretação surgem algumas limitações. O algoritmo de interpretação mostra-se eficaz, porém o problema reside na falta de memória disponível pelo microcontrolador no que diz respeito à memorização temporária de capturas da trama. No máximo poderão ser interpretados 35 pares ON/OFF, um valor suficiente para uma trama RC-5 ou SIRC, contudo, a nível de tramas baseadas no protocolo usado pela Panasonic verificou-se que não era suficiente.

Os testes que foram sendo feitos ao longo de todo o desenvolvimento do subsistema foram baseados no protocolo RC-5, os restantes protocolos apresentados foram avaliados teoricamente com base na informação presente em [56] (Sony SIRC) e em [69] ("Old" Panasonic), estando dentro dos critérios admissíveis pelo software e hardware. Após os testes físicos verificou-se que o Sony SIRC era interpretável, porém o protocolo Panasonic não, dado que a versão testada não correspondia à versão "Old" avaliada teoricamente, apresentando diferenças em termos de portadora (ainda compatível) e em termos de número de pares ON/OFF constituintes (agora incompatível). Não foram feitas alterações no sentido de resolver esse problema pois implicava uma mudança no microcontrolador usado, dado que a memória está aproveitada ao máximo.

Os tempos obtidos na leitura de um comando baseado nos protocolos anteriores (interpretáveis) foram os seguintes:

Philips RC-5 ("Desligar")	
ON (µs)	**OFF (µs)**
864	873
1782	862
864	887
864	887
864	888
864	888
864	888
864	888
864	1774
864	888
1755	888
864	repetição

Tabela 6 Dados obtidos na aquisição de um comando do protocolo Philips RC-5

Sony SIRC (“Desligar/Ligar”)			
ON (µs)	**OFF (µs)**	**ON (µs)**	**OFF (µs)**
2350	606	1200	592
1200	568	1200	592
600	599	1200	592
1200	591	575	591
600	599	600	600
1200	591	1200	600
600	600	600	591
600	599	575	600
600	600	1200	600
1200	591	600	repetição
600	600		

Tabela 7 Dados obtidos na aquisição de um comando do protocolo Sony SIRC

Os valores obtidos apresentados sofreram uma pequena calibração por software, no sentido de se obter uma representação mais fiel do tempo associado ao protocolo. Esse ajuste foi baseado no protocolo Philips RC-5. Estes valores resultaram numa actuação positiva no aparelho respectivo.

Em termos de distância máxima admitida ao posicionamento do telecomando IR para a aquisição do sinal IR, existe uma limitação imposta pelo circuito de recepção da portadora. O máximo é de 30 cm com um ângulo de recepção de cerca de 50º. Esta distância julga-se suficiente tendo em conta que todo o processo de gravação de comandos IR é feito junto ao PC de controlo, e portanto não será muito importante uma distância maior.

A ausência de um filtro passa banda no circuito de detecção da portadora não se revelou muito importante, como se pode comprovar pelos gráficos obtidos.

As restantes funcionalidades também funcionam correctamente, nomeadamente a comunicação com o PC, actualizações de firmware, configuração dos módulos XBee e transmissão RF.

9.2.2 **Transmissão e Emissão de Comandos IR**

O teste de transmissão RF foi feito após a certeza de que os comandos IR emitidos tinham resultados de actuação positivos no receptor em questão, e portanto inicialmente

foi feito um teste com transmissão local de dados, a fim de se comprovar o correcto funcionamento de composição e emissão do comando por parte do módulo de emissão IR. Este teste passava por todo o processo desde a ordem de actuação em Flash até à emissão IR. A transmissão local era baseada numa comunicação via RS-232, nesta fase o módulo de emissão encontrava-se ainda em *breadboard*. Com base neste teste verificou-se também a distância de emissão do sinal IR, obtendo-se um alcance máximo de 10 m. Após a obtenção destes resultados positivos passou-se então ao teste de transmissão via RF.

A transmissão RF via *broadcast* poderia apresentar alguns problemas, dadas as limitações, contudo nos testes efectuados verificou-se uma transmissão sem problemas uma vez que as ordens de actuação eram concretizadas em actuações positivas. Concluiu-se assim com sucesso o teste à parte de transmissão e emissão deste subsistema de controlo de dispositivos por IR.

Capítulo X

10. Conclusões e Sugestões para Trabalho Futuro

10.1 Conclusões

O projecto apresentado nesta tese demonstra que a criação de sistemas de automação residencial baseados em software Flash no controlo de hardware industrial, proporcionam um conjunto de interacção no qual a simplicidade e a performance caminham lado a lado. Na pesquisa efectuada não se encontrou no mercado um sistema com interface gráfico inteiramente baseado em Flash ou que fosse constituído por hardware industrial (PLCs), e portanto estes são de certa forma os aspectos chave deste projecto. Em termos de funcionalidades, nesta solução não existem tantas quantas as que estão presentes em alguns dos sistemas apresentados no estado da arte, nomeadamente a nível de climatização, regulação de luminosidade, vídeo vigilância, funções automáticas, etc., no entanto estão criadas as condições para que se possa dar essa evolução, sendo necessário continuar o desenvolvimento da aplicação de controlo. Todo o controlo do sistema encontra-se em correcto funcionamento, desde a manipulação da base de dados, comunicação (leitura/escrita MB) com o PLC, configurações, no entanto poderão surgir alguns *bugs*, essencialmente a nível de validação de objectos Flash, uma vez que, apesar de todos os testes efectuados, poderão não ter sido verificadas todas as combinações possíveis.

O subsistema de controlo de dispositivos por infravermelhos teve uma perfeita integração no sistema principal, tendo-se assim conseguido dotar o sistema de comunicação IR. Verificou-se que a comunicação via RF através dos módulos XBee foi uma boa opção, quer a nível de fiabilidade, quer a nível de integração. A nível de aquisição de comandos IR, mesmo limitando o universo de protocolos com portadoras compreendidas entre 36-40 kHz, não se pode afirmar que é possível a leitura de todos os protocolos, até porque existem bastante mais para além daqueles que foram testados, contudo comprovou-se que alguns dos mais usados são interpretáveis, como é o caso do Philips RC-5 e Sony SIRC. Em termos de mercado, tendo em conta a leitura, emissão e transmissão (via RF) de comandos IR, foram encontrados módulos que permitem uma ou outra funcionalidade, porém no conjunto e na forma como estão organizadas neste projecto, não se encontraram sistemas que tivessem um funcionamento semelhante.

10.2 Sugestões para Trabalho Futuro

Um sistema de domótica encontra-se em constante desenvolvimento, no sentido de se integrar cada vez mais funcionalidades e ao mesmo tempo aumentar ainda mais a sua performance. Ora este sistema não é excepção, como tal aqui ficam algumas sugestões de funcionalidades que não foram implementadas ou aspectos que podem ser modificados no sentido de melhorar a qualidade do sistema.

Aplicação de Controlo

- Adicionar novas funcionalidades, em especial regulação de luminosidade, climatização, videovigilância e gestão de rega
- Funções automáticas, tais como a simulação de presença de pessoas na habitação
- Funções personalizadas, como por exemplo a regulação de luminosidade num dado compartimento ao gosto de um utilizador específico
- Acesso remoto de vários utilizadores simultaneamente

Subsistema de Controlo de Dispositivos por IR

- Criação de uma extensão PHP baseada na aplicação C/C++ desenvolvida para interface com o módulo de aquisição IR, libertando assim a necessidade de uma aplicação intermédia
- Poderá ser substituído o microcontrolador usado no módulo de aquisição IR por um com mais memória de dados, no sentido de se aumentar o número de capturas da trama IR
- Dotar o módulo de emissão IR de capacidades sensoriais, por exemplo sensores de temperatura, presença e luminosidade

Referências

[1] "A Casa Inteligente" – Introdução à domótica e alguns aspectos técnicos http://acasainteligente.blog.com/

[2] "Domótica – Presente e Futuro", Apresentação, Renato Nunes, Instituto Superior Técnico / INESC-ID www.enei.net/enei2006/documentos/apresentacoes/**Dia2_Prof_Renato**%20Nunes_Domotica.pdf

[3] Domótica – Introdução http://ambiente.dec.uc.pt/~carrilho/Domotica.doc

[4] "Domótica – Edifícios Inteligentes" (imagem) http://engenium.wordpress.com/2007/02/02/domotica-edificios-inteligentes/

[5] Sistema de domótica da empresa JG Domótica (imagem) http://www.jgdomotica.com/jgd-f-por.htm

[6] Software para domótica da empresa HAI (imagem) http://www.homeauto.com/Products/Software/MCE.asp

[7] Sistema de domótica da empresa Domática (imagem) http://www.domatica.pt/?lop=conteudo&op=65b9eea6e1cc6bb9f0cd2a47751a186f&id=3ef815416f775098fe977004015c6193

[8] X10 - Wikipedia http://en.wikipedia.org/wiki/X10_(industry_standard)

[9] "O que é o X-10?" – euroX10 http://www.eurox10.com/Content/X10Information.htm

[10] "Teoria de transmissão de sinais X-10" – euroX10 http://www.eurox10.com/Content/X10SignalTheory.htm

[11] Módulos X10 – X10 Europe (imagem) http://www.x10europe.com/ha/receivers.htm

[12] "VDI/VDE-IT – to bring High-tech to success", Apresentação http://www.vdivde-it.de/smarthome/ws2/vortraege/strese.pdf

[13] Casas Inteligentes, José Augusto Alves e José Mota, Edições Centro Atlântico 2003, Capítulo 9 – Principais Sistemas, Pág.101

[14] "Sistema EIB" em Dissertação "Aplicações Domóticas para Cidadãos com Paralisia Cerebral", Biblioteca Digital http://portal.ua.pt/bibliotecad/default.asp?H1=2&H2=1&H3=2&H4=0&H5=0&num=668

[15] Instabus EIB, Catálogo da empresa Merten (imagem)
www.tev.pt/downloads/12.pdf

[16] "CAN history"
http://www.can-cia.org/index.php?id=161

[17] "CAN – Background Information"
http://www.mjschofield.com/history.htm

[18] "CAN protocol"
http://www.can-cia.org/index.php?id=518

[19] Artigo de jornal online, "CAN em Automação Residencial"
http://www.uai.com.br/UAI/html/sessao_8/2007/05/31/em_noticia_interna,id_sessao=8&id_noticia=13552/em_noticia_interna.shtml

[20] "CAN information"
http://hem.bredband.net/stafni/developer/CAN.htm

[21] "CAN physical layer"
http://www.can-cia.org/index.php?id=517

[22] "EIA Standard RS-485"
http://www.tbos.net/download/sumRS485.htm

[23] Informação padrão RS-485
http://www.escience.unicamp.br/angra/admin/publicacoes/documentos/publicacao_616_RS485.pdf

[24] "Introduction to RS-485"
http://www.lammertbies.nl/comm/info/RS-485.html

[25] Informação do sistema "iDom" da empresa Domática
http://www.domatica.pt/?lop=conteudo&op=f7177163c833dff4b38fc8d2872f1ec6

[26] Seminário Automação de Processos Industriais, IST, Julho de 2007
http://users.isr.ist.utl.pt/~pjcro/cadeiras/api0607/files_aux/seminario_14.pdf

[27] Funcionalidades sistema "Mordomus" das empresas IVV e IOLine
http://www.mordomus.com/funcionalidades.html

[28] IVV Automação – Módulos do sistema "Mordomus"
http://www.ivv-aut.com/index.php?option=catalogo&Itemid=99&func=index.php&cPath=7

[29] CD de apresentação do sistema "Mordomus" (imagens)

[30] Software "Harmony 2007" – Informação geral
http://www.automatedhome.co.uk/Software/Harmony-2007-Home-Automation-Software-Launched.html

[31] Software "Harmony 2007" – Informação e Compatibilidade a nível de hardware
http://www.domialifestyle.com/Harmony2.asp

[32] Estrutura de um sistema baseado no software "Harmony 2007" (imagem)
http://www.domia.eu/index.php?option=com_content&view=article&id=185&Itemid=106

[33] Software de controlo "ALICE" – "Java Home Automation"
http://jhome.sourceforge.net/

[34] "FreeTTS" - Software para sintetização vocal
http://freetts.sourceforge.net/docs/index.php#what_is_freetts

[35] Conversor X10-IR
http://www.centralcasa.com/prodetalhes.asp?familia=ModulosX10

[36] Projecto "Unzap"
http://www.lochraster.org/unzap/?en

[37] "IR Widget"
http://www.compendiumarcana.com/irwidget/

[38] "Servidor Apache" - Wikipedia
http://pt.wikipedia.org/wiki/Servidor_Apache

[39] "Apache" *Web Site*
http://httpd.apache.org/

[40] "Apache" - Configuração
http://httpd.apache.org/docs/2.0/configuring.html

[41] "O que é o PHP?"
http://www.criarweb.com/artigos/202.php

[42] "PHP" - Wikipedia
http://pt.wikipedia.org/wiki/Php

[43] "PHP" *Web Site - Download*
http://www.php.net/

[44] "MySQL" - Wikipedia
http://pt.wikipedia.org/wiki/MySQL

[45] "MySQL" *Web Site - Donwload*
http://dev.mysql.com/downloads/mysql/5.0.html#downloads

[46] Documentário – "A História do Flash"
http://www.tomcarvalho.com/blog/?p=7#respond/

[47] "Adobe Flash" - Wikipedia
http://pt.wikipedia.org/wiki/Adobe_flash

[48] "O que é o Flash?"
http://www.criarweb.com/artigos/282.php

[49] "XAMPP" - Introdução
http://www.apachefriends.org/pt_br/index.html

[50] "PLC History"
http://www.plcmanual.com/plc-history
[51] "Introduction to PLCs"
http://www.plcmanual.com/
[52] Especificações VISION 280
http://www.unitronics.com/Series.aspx?page=vision280#Tab=Spec
[53] Especificações M90/M91
http://www.unitronics.com/Series.aspx?page=M90#Tab=Spec
[54] Optoeletronics- past, present, and future, C. Kumar N. Patel, AT & T Bell Laboratories
http://www.springerlink.com/content/7680x2j577u94123/fulltext.pdf?page=1
[55] Comunicação Digital por Feixe de Infravermelhos, Laboratórios Integrados II
[56] "IR Remote Control Theory"
http://www.sbprojects.com/knowledge/ir/ir.htm
[57] *Application Note,* "Infrared Remote Control Techniques on MC9S08RC/RD/RE/RG Family"
www.freescale.com/files/microcontrollers/doc/app_note/**AN3053**.pdf
[58] *Datasheet*, Conversor USB-UART (FTDI)
http://www.ftdichip.com/Documents/DataSheets/DS_FT232R.pdf
[59] *Datasheet*, Receptor IR Sharp GP1UX511QS
http://sharp-world.com/products/device/lineup/data/pdf/datasheet/gp1ux51qs_e.pdf
[60] Application Note, "Use of IR Detecting Units"
http://joule.bu.edu/~hazen/DataSheets/Sharp/Inf-appnote_Use_of_IR_Detecting_Units.pdf
[61] Receptor IR Sharp GP1UX511QS (imagem)
http://pt.farnell.com/sharp/gp1ux511qs/photodiode-ir-detector/dp/1243868
[62] Fototransístor da Vishay (imagem)
http://pt.farnell.com/vishay/bpw85/phototransistor-t1/dp/1045380?_requestid=203193
[63] *Datasheet* PIC 16F87XA
http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf
[64] Transmissão série assíncrona (imagem)
http://www.rogercom.com/
[65] "Uma breve abordagem ao protocolo ZigBee"
http://paginas.fe.up.pt/~ee02055/info_zigbee.pdf
[66] *Datasheet* módulos XBee
http://ftp1.digi.com/support/documentation/90000982_A.pdf
[67] *Datasheet* reguladores de tensão MAX603-MAX604
http://datasheets.maxim-ic.com/en/ds/MAX603-MAX604.pdf

[68] *Datasheet* LED IR Kingbright
http://www.kemt.fei.tuke.sk/Predmety/KEMT411_ESM/_web/wwwfiles/pouzita%20literatura/%5B17%5D%20L53F3C.pdf

[69] "Panasonic's old infrared remote protocol"
http://users.telenet.be/davshomepage/panacode.htm

Bibliografia

Internet

Manual VISION 280 – “Vision230-260-280 User Guide”
http://support.elmark.com.pl/unitronics/PDF/Vision230-260-280_User%20Guide.pdf

Comunicações VISION – “Vision Communication”
http://www.unitronics.com/data/uploads/software%20utilities/VisionCommunication.pdf

Manual do software “Visilogic”
“Getting Started”
http://support.elmark.com.pl/unitronics/PDF/VisiLogic_Software_Manual-Getting_Started.pdf
“Ladder”
http://support.elmark.com.pl/unitronics/PDF/VisiLogic_Software_Manual-Ladder.pdf

Manual do PHP
http://pt.php.net/get/php_manual_pt_BR.chm/from/a/mirror

Manual do XBee
http://ssdl.stanford.edu/ssdl/images/stories/AA236/0708A/Lab/Rover/Parts/xbeeproproductmanual.pdf

Manual do PIC 16F876A
http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010240

Manual do FTDI 232RL
http://www.ftdichip.com/Documents/DataSheets/DS_FT232R.pdf

Funcionamento do controlo remoto via infravermelhos
http://www.sbprojects.com/knowledge/ir/ir.htm

Livros

"Casas Inteligentes", José Augusto Alves e José Mota, Edições Centro Atlântico 2003

Guia "Aprendendo Action Script 2.0 no Flash", Jen deHaan, Peter deHaan, Joey Lott, 1ª Edição, 2005

"SQL", Luís Damas, FCA Editora, 7ª Edição

"MosFets e Amplificadores Operacionais- Teoria e Aplicações", J. G. Rocha, netmove Comunicação Global Editora, 2005

Artigos

"Controle Remoto com o Microcontrolador 87LPC76X", Newton C. Braga, Revista Saber Electrónica, Nº. 384, Janeiro 2005

"ZigBee Transceiver – Xbee in practice", Fabrice André, Revista Elektor, Nº. 363, Março 2007

Manuais Académicos

"Comunicação Digital por Feixe de Infravermelhos", disciplina de Laboratórios Integrados II, Universidade do Minho