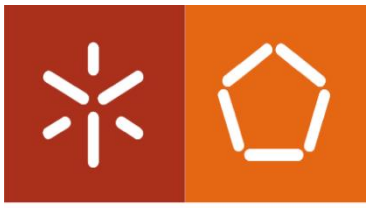


Universidade do Minho
Escola de Engenharia

Manuel Carlos de Araújo Vieira

Laboratório remoto para o ensino de Automação

Dissertação de Mestrado

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

Trabalho efetuado sob a orientação de

Professora Doutora Filomena Soares

Outubro de 2018

DECLARAÇÃO

Nome: Manuel Carlos de Araújo Vieira

Endereço eletrónico: mamuelvieira603@gmail.com Telefone: 962830876 (alternativo)

Bilhete de Identidade/Cartão do Cidadão: 5704387

Título da dissertação: Laboratório Remoto para o Ensino de Automação

Orientador: Professora Doutora Filomena Maria da Rocha Menezes de Oliveira Soares

Ano de conclusão: 2018

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

Universidade do Minho, 29/10/2018

Assinatura

AGRADECIMENTOS

A realização desta dissertação em Mestrado Integrado em Eletrónica Industrial e Computadores, não teria sido possível sem o apoio e incentivos de muitas pessoas ao qual ficarei eternamente grato.

À Professora Doutora Filomena Soares, pelo total apoio, pelas suas opiniões e críticas, pelo incentivo, dedicação e disponibilidade na ajuda durante esta fase, nas aulas lecionadas e também pelo saber transmitido.

Ao Professor Doutor João Sena, que de igual modo se mostrou incansável e sempre disponível, pelo apoio disponibilizado, críticas, opiniões e conhecimento transmitido.

Ao Professor Caetano, responsável pelas oficinas de Mecânica que me disponibilizou incondicionalmente as oficinas, para a realização dos trabalhos de construção das plataformas, pela ajuda e opiniões as quais registo para sempre com agrado e amizade.

Ao Vítor, coordenador das oficinas de Mecânica, pela sua incansável ajuda e dedicação na elaboração e construção das plataformas físicas.

À minha família, em especial à minha esposa Rosa e ao meu filho João Pedro, pela ausência e falta de atenção que por vezes não foi possível dar. À minha filha Ana Catarina que embora longe sempre me soube transmitir a sua força e apoio. Aos meus pais, Fernando e Dalila, em especial à minha mãe pelo facto de não estar entre nós e que sempre terá um lugar especial no meu coração. Aos restantes familiares, tios, sobrinhos, cunhados pelo incentivo e apoio.

Para os meus amigos, colegas de trabalho, entre outros que se cruzaram neste meu trajeto e que me apoiaram o meu agradecimento total e incondicional.

RESUMO

Como forma de se tornarem competitivas, as empresas têm cada vez maior necessidade de automatizar os processos de fabrico, o que requer mão-de-obra qualificada na área da automação industrial. Existem neste momento muitas instituições que têm por objetivo formar jovens para a entrada no mercado de trabalho, dando-lhes a preparação suficiente para derrubar barreiras relativas à sua contratação, especialmente na indústria. Mas, devido a dificuldades financeiras, é frequente que não disponham das ferramentas adequadas à formação dos jovens na área da automação. O objetivo principal desta dissertação é o desenvolvimento de instrumentos que permitam ultrapassar essa situação.

O trabalho realizado focou-se no projeto e no desenvolvimento de uma plataforma acessível através de uma ligação via *Internet* que permita a realização remota de experiências com redes de autómatos industriais. Os resultados dessas experiências poderão ser visualizados com o auxílio de uma camara de Protocolo de Internet. Tirou-se partido de um sistema de autómatos ligados em rede, já existente na Universidade do Minho, produto de projetos anteriores.

Em suma, pretendeu-se desenvolver um laboratório remoto para o ensino e a aprendizagem de utilização de sistemas de automação industrial, onde alunos universitários e técnicos de empresas, entre outros, possam ter um primeiro contacto com a programação de autómatos industriais ou então melhorar os conhecimentos já adquiridos nessa área.

Palavras-chave: Automação, redes de autómatos, laboratórios remotos.

ABSTRACT

In order to be competitive, companies are increasingly in need of automating manufacturing processes, which requires skilled labor in the area of industrial automation. There are now many institutions that aim to train young people to enter the labor market, giving them enough preparation to break down barriers to their hiring, especially in industry. However, due to financial difficulties, they often lack the appropriate tools to train young people in automation area. The main objective of this dissertation is the development of tools to overcome this situation.

The work carried out will focused on the design and development of an accessible platform through an Internet connection that allows the remote realization of experiences with networks of industrial automata. The results of these experiments can be viewed with the help of an Internet Protocol camera. It will take advantage of a system of networked automata, already existing in the University of Minho, product of previous projects.

In short, it was intended to develop a remote laboratory for the teaching and learning off the use industrial automation systems, where university students and company technicians, among others, can have a first contact with the programming of industrial automata or improve the knowledge already acquired in this area.

Keywords: Automation, automation networks, remote laboratories

Indices

Agradecimentos v
Resumo vii
Abstract ix
Lista de figuras xv
Lista de tabelas xix
Acrónimos xxi
1 Introdução 1
1.1. Motivação 4
1.2. Enquadramento 4
1.3. Objetivos 4
1.4. Estrutura da dissertação 5
2 Estado da arte 7
2.1. *WebLab-Deusto* – Universidade de *Deusto* 9
2.2. *The FPGA Laboratory* 10
2.3. *The Aquarium Laboratory* 11
2.4. *The Ud-Demo-PLD* 11
2.5. Espetroscópio motorizado – Projeto PEARL, *Open University UK* 12
2.6. *Laboratório de visão por computador - Projeto PEARL, Trinity College Dublin* 13
2.7. Laboratório Remoto de Automação Industrial – ISTP 14
2.8. WALC-AI: Laboratório para Aprendizagem do Controlo e Automação Industrial 15
2.9. *Compilação de várias configurações sobre Laboratórios Remotos* 18
3 Autómatos Programáveis 23
3.1 Autómato Omron CP1L-M30DT-D 26
3.1.1 Entradas 27
3.1.2 Saídas (*sinking* transístor) 27
3.1.3 *Memory Area* 29
3.1.4 *Work Area (W)* 31
3.1.5 *Holding Area* 31

3.1.6 *Auxiliary Area (A)* 32
3.1.7 *Temporary Relay Area (TR)* 33
3.1.8 *Data Memory Area (D)* 33
3.1.9 *Timer Area (T)* 33
3.1.10 *Counter Area(C)* 34
3.1.11 *Condition* Flags 34
3.1.12 *Task Flags Area (TK)* 34
3.1.13 *Index Registers (IR)* 34
3.1.14 *Data Registers (DR)* 34
3.2 CX-Programmer 35
3.3 Linguagens de Programação 36
3.4 GRAFCET 38
3.5 Ciclo Scan 39
3.6 Norma 6149 39
3.7 SCADA – Controlo de Supervisão e aquisição de dados 40
4 Protocolo de comunicação 43
4.1 *Ethernet* 45
4.2 TCP/IP 45
4.3 *Modbus* 46
4.3.1 *Modbus* ASCII 47
4.3.2 *Modbus* RTU 47
4.3.3 *Modbus* TCP/IP 47
4.4 *Wireless Modbus* 50
4.5 Modelo de dados Modbus TCP/IP 51
4.5.1 *Read Coil Status (01)* 55
4.5.2 *Read Holding Registers (03)* 57
4.5.3 *Read Input Registers (04)* 57
4.5.4 *Force Single Coil (05)* 58
4.5.5 *Preset Single Register (06)* 59
4.5.6 *Force Multiple Coils (15)* 60

4.5.7 *Preset Multiple Registers (16)* 61
4.5.8 *Report Slave ID (17)* 63
5 Plataforma de testes 65
5.1 Plataforma A 67
5.2 Plataforma B 71
5.3 Plataforma A + Plataforma B 75
5.4 Testes de funcionamento 77
6 Plataforma web 85
6.1 Página web inicial 87
6.2 Página *About* 88
6.3 Página de registo 89
6.4 Página de Login 90
6.5 Página do Administrador 92
6.6 Página de espera 96
6.7 Página da plataforma 97
6.8 *Web Server* 100
6.9 C-Panel 101
6.10 Aplicação de Comando 104
6.10.1 Aplicação em modo de simulação 105
7 Conclusões e Trabalho Futuro 109
Referências 113

LISTA DE FIGURAS

Figura 1 - Robô do Laboratório Robot Lab [1]. 10
Figura 2 - Laboratório FPGA [1]. 10
Figura 3 - Laboratório do Aquário [1]. 11
Figura 4 - Modo demo do laboratório de Ud-Demo-PLD [2]. 12
Figura 5 - Espetroscópio Motorizado [4]. 12
Figura 6 - Laboratório de visão por computador [4]. 13
Figura 7 - Laboratório remoto de Automação Industrial [5]. 14
Figura 8 - Painel de monitorização remoto da "Casa Inteligente" e respetivo painel de controlo [7]. ... 15
Figura 9 - Painel de monitorização remota da "Casa Inteligente" através de vídeo câmara [7]. 16
Figura 10 - Página web com visualização do painel 1 [7]. 16
Figura 11 - Página web com visualização do painel 2 [7]. 17
Figura 12 - Compilação de diferentes tipos de arquitetura [8]. 20
Figura 13 - MHSA como estrutura complementar para desenvolvimento de laboratórios remotos [8].. 20
Figura 14 - Configuração operacional da arquitetura do laboratório remoto [8]. 21
Figura 15 - PLC aplicado à Indústria [12]. 26
Figura 16 - Autómato CP1L-M30DT-D [11]. 26
Figura 17 - Entradas [11]. 27
Figura 18 - Saídas [11]. 28
Figura 19 - Saídas 100.00 a 100.03 [11]. 28
Figura 20 - Saídas 100.04 a 101.03 [11]. 29
Figura 21 - Área CIO [11]. 29
Figura 22 - Área de ligação série [11]. 30
Figura 23 - Ligação série entre PLCs [11]. 31
Figura 24 - Área de trabalho [11]. 31
Figura 25 - Área de espera [11]. 32
Figura 26 - Área auxiliar [11]. 32
Figura 27 - Área de memória de dados [11]. 33
Figura 28 - Software CX-Programmer da Omron 35
Figura 29 - Linguagem LADDER [9]. 36
Figura 30 - Diagrama de blocos [9]. 37

Figura 31 - Texto estruturado [12]. 37
Figura 32 - Lista de instruções [13]. 38
Figura 33 - GRAFCET [9]. 38
Figura 34 - Interface gráfica HMI [15]. 41
Figura 35 - Despacho de Lisboa [15]. 41
Figura 36 - Construção da trama Modbus TCP [17]. 49
Figura 37 - Construção do pacote de dados Ethernet-TCP/IP [17]. 52
Figura 38 - Plataforma A. 67
Figura 39 - Encoder [18]. 68
Figura 40 - Adaptador CP1W-EIP61. 69
Figura 41 - Adaptador CP1W-CIF11. 69
Figura 42 - Switch Industrial de Ethernet de 8 Portas. 70
Figura 43 - Plataforma A (cablagem). 71
Figura 44 - Plataforma A. 72
Figura 45 - Plataforma B. 73
Figura 46 - Plataforma B (cablagem). 73
Figura 47 - Plataforma B (montagem final). 74
Figura 48 - Plataforma A + Plataforma B. 75
Figura 49 - Conexão entre as plataformas A e B. 76
Figura 50 - CX-Programmer. 77
Figura 51 - Teste 1. 78
Figura 52 - Teste 2. 79
Figura 53 - Teste 3. 80
Figura 54 - Teste 4. 81
Figura 55 - Testes via Ethernet. 82
Figura 56 - Teste 1 (CX-Programmer). 83
Figura 57 - Teste 2 (Plataforma A). 83
Figura 58 - Teste 3 (Plataforma B). 84
Figura 59 - Página web. 87
Figura 60 - Página de informação sobre o projeto. 88
Figura 61 - Página de Registo. 89
Figura 62 - Página de Login. 90

Figura 63 - Dados incorretos. 91
Figura 64 - Painel do Administrador. 92
Figura 65 - Painel do Administrador. 93
Figura 66 - Edição de dados. 94
Figura 67 - Update de dados. 95
Figura 68 - Update com dados inválidos. 95
Figura 69 - Página de Espera. 96
Figura 70 - Página do Laboratório. 97
Figura 71 - Timeout. 98
Figura 72 - Camara IP. 99
Figura 73 - C-Panel. 101
Figura 74 - Base de dados phpMyAdmin. 102
Figura 75 - Painel Administrador de Ficheiros. 103
Figura 76 - Aplicação de comando. 104
Figura 77 - CMD. 105
Figura 78 - Simulador Modbus. 105
Figura 79 - Leitura de dados. 106
Figura 80 - Simulação com Holding Registers. 106
Figura 81 - Leitura de dados (Gauge). 107

LISTA DE TABELAS

Tabela 1 - Modelo OSI para a pilha de comunicação Modbus TCP/IP [17]. 51
Tabela 2 - Registos Modbus [17]. 53
Tabela 3 - Funções e registos Modbus [17]. 54
Tabela 4 - Exemplo duma Header ADU [17]. 55
Tabela 5 - Resposta à leitura do estado da bobina [17]. 56
Tabela 6 – Exemplo de leitura do estado da bobina [17]. 56
Tabela 7 - Leitura do registo de espera [17]. 57
Tabela 8 - Leitura dos registos de entrada [17]. 58
Tabela 9 - Bobina em estado forçado [17]. 59
Tabela 10 - Registo predefinido [17]. 59
Tabela 11 - Múltiplas bobinas em estado forçado [17]. 60
Tabela 12 - Resposta ao estado forçado de múltiplas bobinas [17]. 61
Tabela 13 - Resposta ao estado forçado de múltiplas bobinas [17]. 61
Tabela 14 - Consulta de vários registos predefinidos [17]. 62
Tabela 15 - Resposta à predefinição de múltiplos registos [17]. 62
Tabela 16 - Informação à consulta para identificação do escravo[17]. 63
Tabela 17 - Resultado resposta à identificação do escravo [17]. 63

ACRÓNIMOS

ADU	*Application Data Unit*
CIO	*Chanel Input Output*
CPU	*Central Processing Unit*
LED	*Light Emitting Diode*
LSB	*Least Significant Bit*
MIEEIC	Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores
MBAP	*Modbus Application Protocol*
MOODLE	*Modular Object-Oriented Dynamic Learning Environment*
MSB	*Most Significant Bit*
NC	*Normally Closed*
NO	*Normally Open*
OSI	*Open Systems Interconnection*
OUT	*Output*
PLC	*Programmable Logic Controller*
RLL	*Relay Ladder Logic*
RTU	*Remote Terminal Unit*
SCADA	*Supervisory Control and Data Acquisition*

1 INTRODUÇÃO

Resumo

Neste capítulo apresentam-se a motivação, enquadramento e objetivos do trabalho desenvolvido. Também se faz um breve resumo sobre a estrutura desta dissertação.

1.1. Motivação

1.2. Enquadramento

1.3. Objetivos

1.4. Estrutura da Dissertação

Introdução

Como forma de se tornarem competitivas, as empresas têm cada vez maior necessidade de automatizar os processos de fabrico, o que requer mão-de-obra qualificada na área da automação industrial. Existem neste momento muitas instituições que têm por objetivo formar jovens para a entrada no mercado de trabalho, dando-lhes a preparação suficiente para derrubar barreiras relativas à sua contratação, especialmente na indústria. Mas, devido a dificuldades financeiras, é frequente que não disponham das ferramentas adequadas à formação dos jovens na área da automação. O objetivo principal desta dissertação é o desenvolvimento de instrumentos que permitam ultrapassar essa situação.

O presente trabalho insere-se na identificação da necessidade de colmatar algumas lacunas existentes no sistema de ensino e na oportunidade em criar meios importantes com a finalidade de as minimizar. As instituições de ensino devem fornecer/disponibilizar meios práticos complementares aos teóricos, no caso especial na área da automação industrial.

A preparação de base na área da automação industrial, no que respeita à parte prática, é muitas vezes, uma barreira senão intransponível, verdadeiramente difícil de ultrapassar, aquando da entrada no mercado de trabalho.

Identificadas as necessidades e na posse de ferramentas e meios para dar um contributo na formação de futuros estudantes, este projeto no âmbito da dissertação pretende contribuir de forma direta na melhoria das competências adquiridas durante a formação dos jovens estudantes.

1.1. Motivação

Como motivação especial salienta-se o reconhecimento da necessidade deste tipo de projetos criar valor para todos os alunos que se revejam na área de automação, bem como e não menos importante, para aquelas pessoas que já estejam no mercado de trabalho e queiram melhorar o seu conhecimento sobre redes de autómatos.

Reconhecida que é a constante evolução tecnológica e a necessidade das empresas, fruto da forte concorrência, terem uma crescente necessidade de pessoas qualificadas capazes de entrarem no mercado de trabalho já com alguma experiência no manuseamento e programação de autómatos, faz com que projetos deste tipo sejam motivo de interesse pois a grande dificuldade das universidades, institutos, escolas de formação profissional, especialmente por motivos económicos, reside na escassez de recursos para aprendizagem prática.

1.2. Enquadramento

No âmbito do desenvolvimento da dissertação de mestrado do 5º ano do Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores (MIEEIC), foi criada uma bancada de testes conectada a um autómato que por sua vez está ligada via internet a uma plataforma *online* e que pode ser utilizada por alunos/estudantes na área de automação

1.3. Objetivos

O objetivo deste trabalho é o projeto e o desenvolvimento de uma plataforma acessível através de uma ligação via Internet que permita a realização remota de experiências com redes de autómatos industriais. Em primeiro lugar procedeu-se à ativação do sistema, rede de autómatos industriais e plataforma de testes, que deverá funcionar local e remotamente. Esta última funcionalidade deve permitir o acesso e a comunicação via *web* e um sistema de registo e agendamento de acessos reservados. Seguidamente procedeu-se à instalação de um circuito de vídeo composto por camaras que estarão conectadas via *web* para visualização das experiências a efetuar.

Finalizada a implementação da rede de comunicação procedeu-se à identificação e desenvolvimento de projetos ligados à indústria, passiveis de integrar a plataforma entretanto desenvolvida.

1.4. Estrutura da dissertação

A presente dissertação encontra-se dividida em 7 capítulos:

- **Capítulo II – Estado da Arte**, como forma de estabelecer uma visão mais abrangente sobre o que existe de oferta, e o quanto se poderá acrescentar de novo.

- **Capítulo III – Autómatos,** onde é abordada a sua referência histórica, autómato , linguagens de programação e sistemas de supervisão, capazes de serem utilizadas no projeto em curso.

- **Capítulo IV – Protocolo de Comunicação**, onde foi estudado e desenvolvido o meio de comunicação entre o meio físico (autómatos, bancadas) e a plataforma *web*.

- **Capítulo V – Plataformas de testes**, onde é projetada e construída uma bancada de ensaio, por forma a facilitar o uso de autómatos por alunos em laboratório e salas de aula mas também através do laboratório remoto de automação.

- **Capítulo VI – Plataforma Web**, desenvolvida em HTML, com recurso a uma base de dados, capaz de funcionar como interface *web* para o uso do laboratório remoto de automação. Tem como principais funcionalidades, o registo de utilizadores, gestão e agendamento do uso remoto e interface de utilização do laboratório.

- **Capítulo VII – Conclusões e Trabalho Futuro,** onde se pretende criar uma ponte entre o que foi realizado e ideias de projeto(s) futuro(s).

2 ESTADO DA ARTE

Resumo

O desenvolvimento de novas soluções inicia-se impreterivelmente pela pesquisa e estudo de soluções já existentes. A pesquisa cujos resultados se apresentam neste capítulo centrou-se em laboratórios remotos e/ou virtuais focados no ensino/aprendizagem na área da Automação Industrial.

2.1. *WebLab-Deuto* – Universidade de *Deusto*

2.2. *The FPGA Laboratory*

2.3. *The Aquarium Laboratory*

2.4. *The Ud-Demo-PLD*

2.5. Espetroscópio motorizado – Projeto *PEARL, Open University UK*

2.6. Laboratório de visão por computador - Projeto *PEARL, Trinity College Dublin*

2.7. Laboratório Remoto de Automação Industrial - Instituto Superior Técnico de Lisboa

2.8. WALC-AI: Laboratório para Aprendizagem do Controlo e Automação Industrial

2.9. Compilação de várias configurações sobre Laboratórios Remotos

Estado da arte

Com o decorrer da pesquisa efetuada foram sendo encontrados vários laboratórios especialmente em institutos e universidades em Portugal e em muitos outros países. A diversidade destes laboratórios é imensa focando projetos orientados para a área da física, medicina, ciências, tecnologias.

Consoante a área de interesse, é comum encontrarem-se laboratórios virtuais ou laboratórios remotos sendo que a escolha recai normalmente para um destes tipos.

Os laboratórios virtuais, funcionam essencialmente por emulação de *software*, como por exemplo o *Matlab* ou o *Labview* e não há uma interação física com dispositivos físicos, por exemplo máquinas e sensores.

Já no que se refere aos laboratórios remotos, pese embora sejam controlados à distância por uma ligação tipicamente via *web*, existe uma ação física entre os materiais (sensores, atuadores, motores, autómatos) mesmo sendo estes controlados por aplicações de *software* e com visualização por camaras.

Descrevem-se de seguida alguns dos laboratórios disponíveis.

2.1. *WebLab-Deusto* – Universidade de *Deusto*

A universidade de *Deusto* disponibiliza vários laboratórios de acesso remoto dos quais se destacam os seguintes [1]:

The Robot Lab

O *Robot Lab* possibilita a programação de um robô identificado na Figura 1 com o *software* que o utilizador entender e verificar remotamente o seu funcionamento. O *software* é carregado para um microprocessador controlador de interrupção programável (microprocessador PIC). De ressalvar que entre outras coisas este robô está equipado com sensores infravermelhos aos quais o utilizador tem acesso. Para ver os resultados, é fornecida uma *Webcam*.

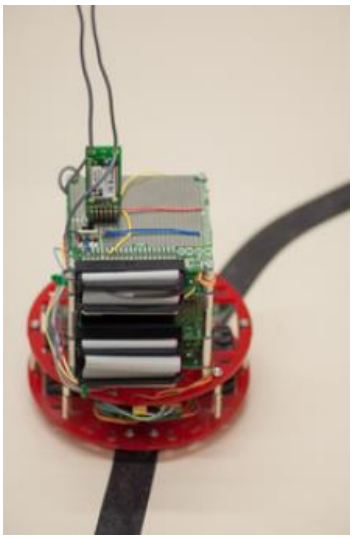


Figura 1 - Robô do Laboratório Robot Lab [1].

2.2. *The FPGA Laboratory*

O laboratório FPGA *(Field Programmable Gate Array)* permite praticar remotamente com uma Matriz de Porta Programável de Campo (FPGA). Através do software *Xilinx*, poder-se-á escrever um programa FPGA localmente. Uma vez que o programa é compilado e pronto para ser testado, basta simplesmente carregar o arquivo binário ". *bit*" através da experiência. Para ver os resultados, é fornecida uma *Webcam*. Pode-se também interagir com a placa remotamente, usando os *widgets* fornecidos.

Figura 2 - Laboratório FPGA [1].

2.3. *The Aquarium Laboratory*

O laboratório de aquário cria um acesso a um verdadeiro aquário localizado na Universidade de *Deusto*. Nela, é possível alimentar o peixe, ligar e desligar as luzes e, se o submarino estiver na água e estiver carregado, controlar o submarino. Com relação à alimentação dos peixes, pode parecer perigoso, mas não é. O sistema alimenta-os automaticamente três vezes ao dia, a cada 8 horas. Se um utilizador os alimentar, então não permite que nenhum outro utilizador os alimente antes da próxima mudança, garantindo que eles só são alimentados três vezes.

Figura 3 - Laboratório do Aquário [1].

2.4. *The Ud-Demo-PLD*

O *Ud-Demo-PLD* permite que o utilizador pratique remotamente com um Dispositivo Lógico Programável (PLD).

Como experiência o *Programmable Logic Device* (PLD), *Logic Device* (LD) padrão, através do software *Xilinix* este pode escrever um programa PLD localmente. Uma vez que o programa é compilado e pronto para ser testado, pode carregar o arquivo binário". *jed*", será programado na placa física e executado.

Na figura 4 pode ver-se a imagem da página *web* onde foi efetuada uma experiência embora que limitada pela versão demo. Para utilizar na sua versão integral o utilizador terá de se registar e agendar o acesso.

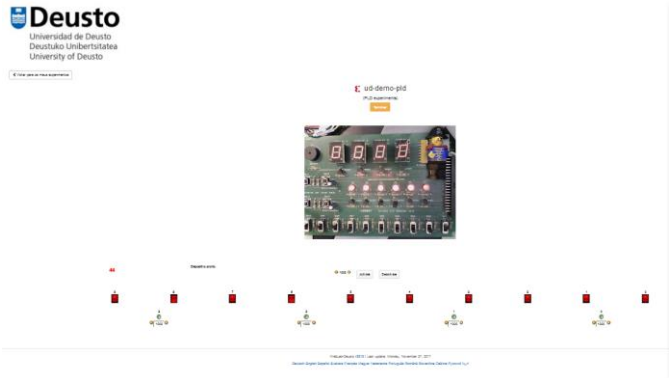


Figura 4 - Modo demo do laboratório de Ud-Demo-PLD [2].

2.5. Espetroscópio motorizado – Projeto PEARL, *Open University UK*

Direcionado para a familiarização dos espetros de luz e nas diferenças quantitativas e qualitativas dos espetros dos vários tipos de luz. O projeto baseia-se num espetroscópio motorizado que é controlado remotamente. Embora não tenha sido possível implementar uma solução única do ponto de vista técnico, o projeto foi considerado bem-sucedido. Quanto à aprendizagem verificou-se que os alunos atingiram os principais objetivos geralmente cumpridos nas experiências de laboratório presenciais [3].

Figura 5 - Espetroscópio Motorizado [4].

2.6. *Laboratório de visão por computador - Projeto PEARL, Trinity College Dublin*

Consiste numa mesa com uma placa de circuito impresso, numa câmara, e luzes sendo todos estes elementos controlados remotamente. O laboratório é constituído por um servidor de laboratório e uma outra máquina servidor *web*.

O servidor *web* providencia a *interface* com o utilizador, sendo a comunicação efetuada por HTTP de forma a evitar as *firewalls*. A comunicação vídeo é efetuada por *Streaming* direto, usando *Real Time Streaming Protocol* (RTSP) do servidor de laboratório para o servidor *web*, sendo posteriormente enviados para o cliente usando Java. O controlo da mesa e da câmara foi implementado usando C++ e o controlo das luzes é efetuado usando tecnologia X-10. Para a *interface* com o utilizador são utilizadas *applets Java* [4].

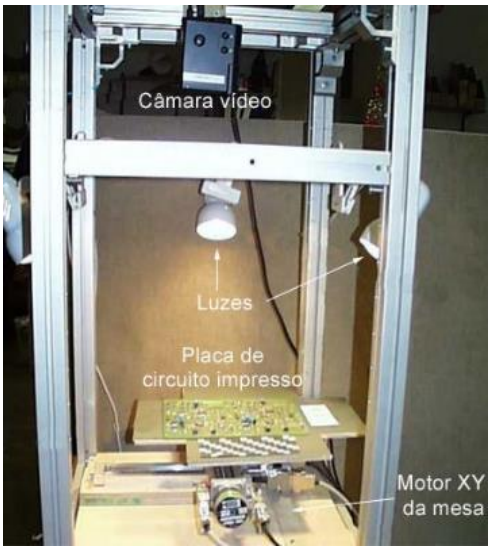


Figura 6 - Laboratório de visão por computador [4].

2.7. Laboratório Remoto de Automação Industrial – ISTP

O Laboratório Remoto de Automação Industrial encontra-se fisicamente instalado no Laboratório de Automação e Robótica da Área Científica de Controlo, Automação e Informática Industrial, Departamento de Engenharia Mecânica do Instituto Superior Técnico. É composto por 8 postos independentes, cada um com o seu autómato. Quatro deles estão associados a equipamento que podem controlar constituindo assim um laboratório remoto. Todos eles têm alojadas páginas *web* que simulam equipamento, constituindo assim laboratórios virtuais [5].

O Laboratório Remoto de Automação Industrial do Instituto Superior Técnico da Universidade Técnica de Lisboa está mais dedicado ao ensino da electropneumática. O aluno com base no *software* dos autómatos SAIA utilizados têm acesso a uma *interface* que lhes permita interagir com o painel demonstrado na figura 7. A ligação dos alunos ao laboratório é feita via VPN por uma questão de facilidade e proteção. As tecnologias utilizadas do lado do utilizador são os softwares da SAIA juntamente com o navegador de internet [6].

As tecnologias do laboratório são todas com recurso uma vez mais aos *softwares* SAIA. Por navegador de internet os alunos têm acesso a uma interface com vários botões que devidamente programados permitem ter uma interação com o autómato. Permite ainda ter uma perceção dos movimentos das hastes através de pequenas animações com o desenho dos cilindros (Ribeiro 2012) [6].

Figura 7 - Laboratório remoto de Automação Industrial [5].

2.8. WALC-AI: Laboratório para Aprendizagem do Controlo e Automação Industrial

O laboratório que a seguir será apresentado foi desenvolvido como parte de uma dissertação do Mestrado Integrado do Engenharia Eletrónica Industrial e Computadores, na Universidade do Minho [7].

O trabalho consistiu na programação de um autómato por forma a controlar uma casa em miniatura com diversas funcionalidades, tais como, acionamento de interruptores para ligação da luz interior e exterior, abertura e fecho de portas, temperatura, refrigeração, detetores de intrusão, entre outros, por forma a simular uma casa real. De seguida e com recurso ao software *LabView* foi desenvolvido um sistema remoto de controlo e monitorização da referida casa.

A janela apresentada na Figura 8 apresenta a interface de controlo e monitorização da "Casa Inteligente". Na monitorização da casa são utilizados diversos sinalizadores que indicam os sensores que estão a ser atuados e as lâmpadas que estão a ser controladas. Ao lado da foto da casa está colocado um painel virtual com um conjunto de botões e os respetivos sinalizadores que tem como função simular os botões e leds existentes na parte de baixo casa [7].

Neste projeto também foi utilizado uma câmara com endereço IP, que para além de disponibilizar a imagem real da casa, também substitui a fotografia utilizada na janela do teste anterior. A Figura 9 apresenta a janela com a imagem da casa obtida através da câmara e o painel lateral com os botões e sinalizadores responsáveis pela monitorização e controlo da casa [7].

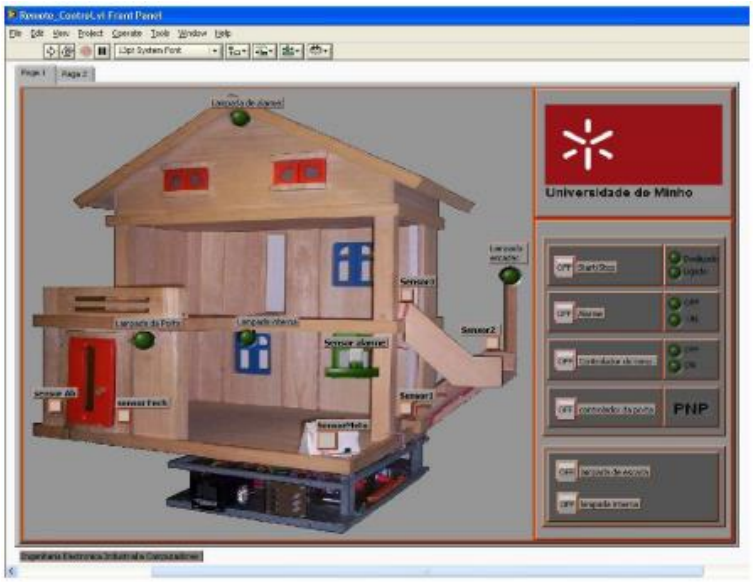


Figura 8 - Painel de monitorização remoto da "Casa Inteligente" e respetivo painel de controlo [7].

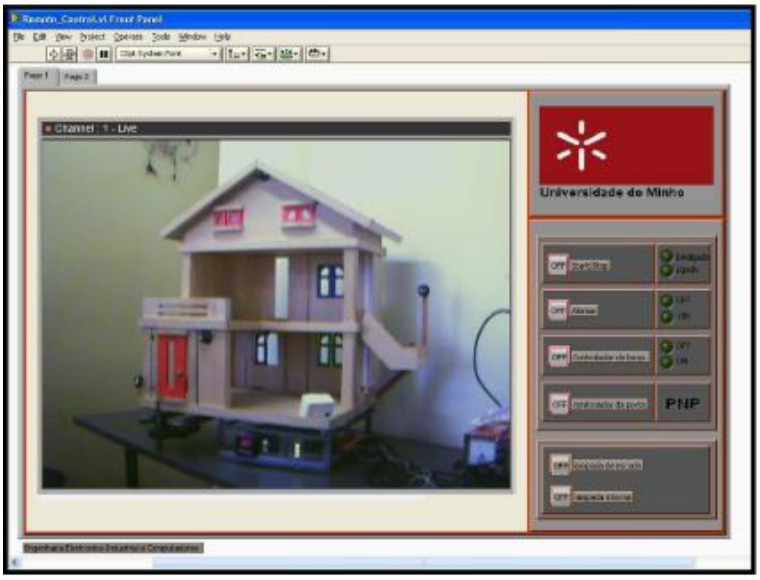


Figura 9 - Painel de monitorização remota da "Casa Inteligente" através de vídeo câmara [7].

O primeiro teste foi realizado com o protocolo *Host Link*. Como este protocolo só funciona com o autómato ligado diretamente ao PC (através da comunicação série), então para controlar a casa via *Web* foi necessário usar a capacidade que o *LabView* possui em gerar páginas *Web* automaticamente [7].

A Figura 10 mostra a página gerada pelo *LabView* onde se pode ver o painel de controlo com a foto da casa (painel frontal da Figura 8).

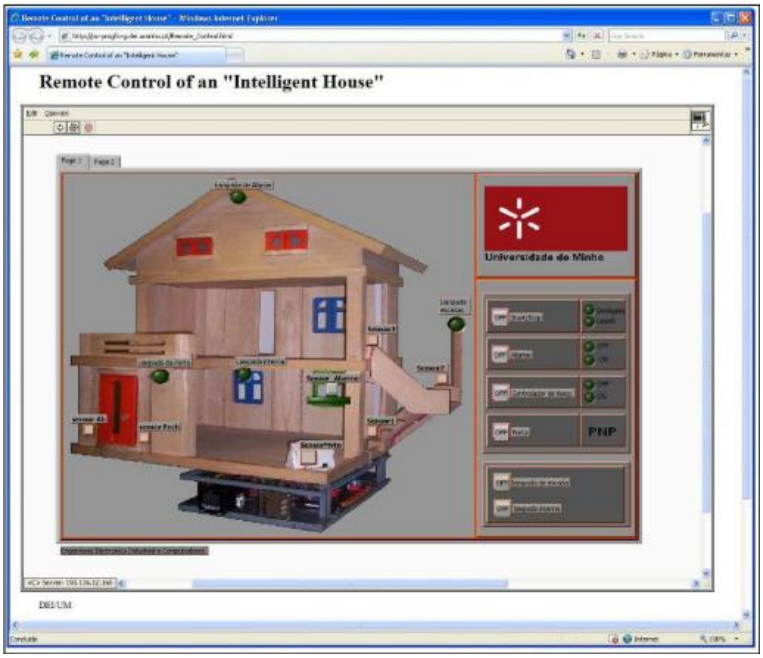


Figura 10 - Página web com visualização do painel 1 [7].

Na Figura 11 mostra a janela com a imagem da casa dada através da câmara na página gerada pelo *LabView*.

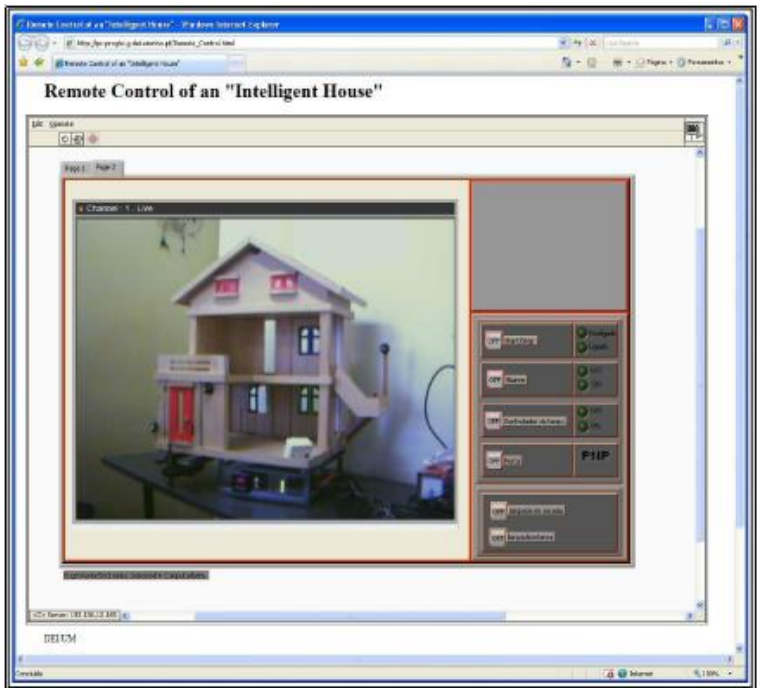


Figura 11 - Página web com visualização do painel 2 [7].

2.9. *Compilação de várias configurações sobre Laboratórios Remotos*

Num artigo publicado no *IEEE Journal, sob o título Transactions on Learning Technologies*, numa publicação conjunta da autoria de Martin Kaluz Javier Garcıa-Zubıa, Miroslav Fikar e Lubos Cirka, podem ser visualizadas algumas abordagens mais comuns sobre Laboratórios remotos. De seguida, os autores, apresentam uma nova tipologia, baseada na utilização de Routers de Rede Industriais.

As diferentes configurações identificadas pelos autores, na pesquisa por estes efetuada, como sendo as mais recorrentemente utilizadas em Laboratórios Remotos e apresentada na Figura 12, são as seguintes:

1. **Cliente – PC servidor com controlo por software – dispositivo experimental.**

   Esta arquitetura, bastante utilizada, usa software de controlo com conexão direta entre o PC (servidor) e o dispositivo a controlar normalmente por porta série ou USB. Tem como particularidade a sua baixa dificuldade de implementação e custo. Como software de controlo destacaram-se dois tipos de abordagens, uma que usa um aplicativo cliente *Java* e a tecnologia do servidor baseada em *Java/Matlab* e uma outra que utiliza o software Labview (Figura 12 – caso 1) [8].

2. **Cliente – PC servidor com controlo por software – dispositivo DAQ - dispositivo experimental.**

   A configuração seguinte (Figura 12 – caso 2) é bastante semelhante à anterior, com uma única diferença que é o dispositivo de aquisição de dados (geralmente cartão DAQ). Esta arquitetura é normalmente utilizada quando não é possível conectar diretamente o PC do lado do servidor. Como principal particularidade, este tipo de configuração de *hardware* geralmente reduz os requisitos do *software* de processamento de sinal incluído no PC de controlo, mas também aumentam o preço [8].

3. **Cliente – servidor *proxy* – PC com controlo por software – dispositivos experimentais.**

   Na proposta seguinte, representado na Figura 12 – caso 3, utiliza nodos de laboratório (eventualmente formado por um switch para ramificação de vários outros laboratórios), fornece uma maior capacidade do que qualquer outra para gerir as conexões de diferentes arquiteturas.

De referir que a utilização de *switches* reduz a possibilidade de colisão entre os pacotes de dados em transito, aumentando assim a fiabilidade. Cada nó geralmente consiste num computador local (servidor de laboratório) com *software* e *hardware* apropriado para aquisição de dados. A principal vantagem desta arquitetura está na ampla conectividade entre os diversos nós que podem estar localizados em diferentes locais (salas, cidades, países), enquanto os acessos dos utilizadores são geridos por um servidor *proxy* [8].

4. **Cliente – PC servidor com sistema SCADA – unidade de controlo – dispositivo experimental.**

   Na representação apresentada na Figura 12 – caso 4, a arquitetura utilizada é cliente – servidor com sistema de supervisão e aquisição de dados (SCADA) - unidade de controlo - dispositivo experimental. A combinação entre o sistema SCADA e uma unidade de controlo (PLC ou outro controlador de *hardware*) é uma das abordagens industriais mais usadas. Entre os recursos comuns, como aquisição de dados, a maioria dos sistemas SCADA também fornece Interfaces Homem-Máquina (HMIs), que podem ser usadas para implementação no lado cliente de laboratórios remotos. Abordagens nesta categoria são caraterizadas por seu alto custo decorrente do uso de *software* e *hardware* comerciais, mas também pela redução significativa do esforço e do tempo de implementação necessários [8].

5. **Cliente – PC servidor/microcomputador - placa eletrónica programável – dispositivo experimental.**

   Por último, a arquitetura seguinte (Figura 12 – caso 5), através da utilização de aplicações muito populares nos últimos anos, baseados em placas programáveis como *Field Programmable Gate Array (FPGA),* microcontroladores Arduíno e alternativas baratas para computadores padrão (por exemplo, *boards* baseadas na arquitetura ARM como *Raspberry Pi, Pandaboard* e outros). Devido ao amplo espetro de vários componentes de hardware, esta categoria não pode ser generalizada como um tipo específico de arquitetura e difere de outros tipos apenas no uso de componentes baratos. Essa abordagem pode ser descrita como a alternativa oposta à categoria 4, especialmente do ponto de vista de preço e implementação. Embora o preço de compra do *hardware* seja incomparavelmente menor do que o preço dos sistemas de controlo industrial, a dificuldade de implementação e o esforço necessário para o desenvolvimento de *software*

aumentam rapidamente, porque os desenvolvedores precisam trabalhar com componentes brutos [8].

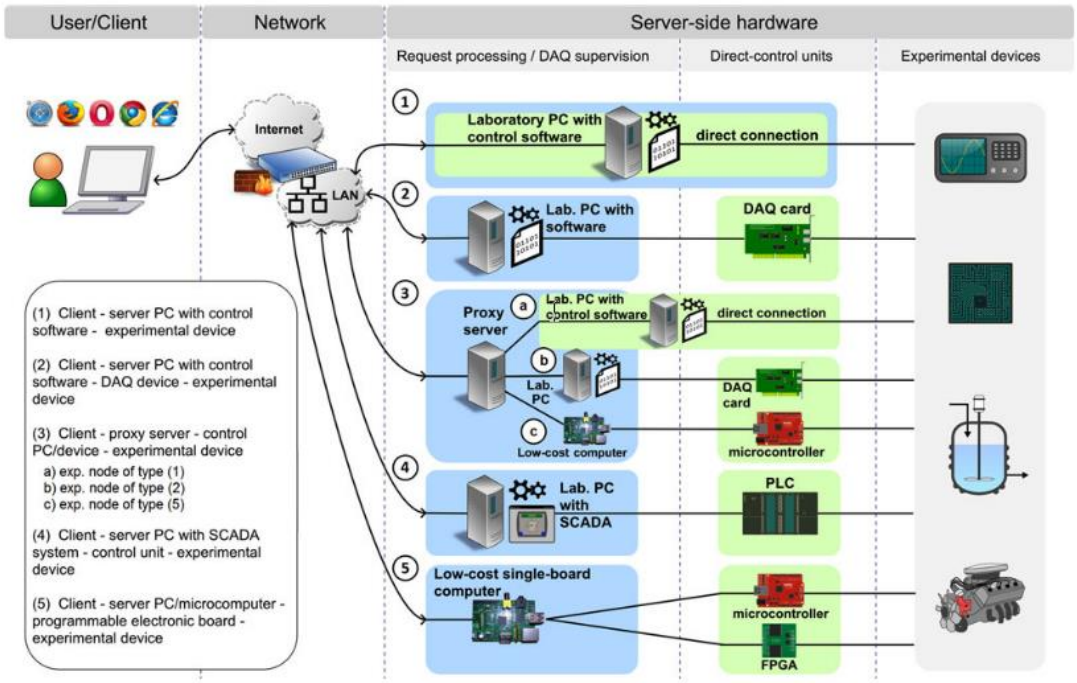


Figura 12 - Compilação de diferentes tipos de arquitetura [8].

Como proposta, pelos referidos autores, são de seguida apresentados os argumentos e uma nova tipologia de arquitetura, baseada na utilização de routers de redes industriais (na sigla inglesa INR – *Industrial Network Routers*).

Neste trabalho, um novo tipo de estrutura física, Arquitetura de *Software* e *Hardware* de Múltiplos Propósitos (MHSA), é proposto como a camada complementar entre L1 e L2 de sistemas de desenvolvimento baseados em API (Figura 13) [8].

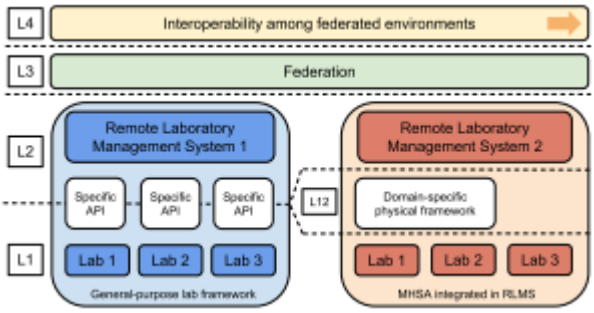


Figura 13 - MHSA como estrutura complementar para desenvolvimento de laboratórios remotos [8].

A arquitetura de *hardware* proposta pelos autores consiste em dois dispositivos principais de controlo: o INR e o PLC. Ao contrário dos tipos tradicionais de arquiteturas, o MHSA substitui o computador servidor por um INR, que é projetado principalmente para supervisionar os controladores industriais conectados à rede local. O INR representa a camada de supervisão da configuração operacional (Figura 14). A camada operacional inferior consiste em nós de PLCs e dispositivos experimentais. Dependendo da conectividade dum PLC específico usado na arquitetura, fornecido pelos módulos de E / S do PLC, e do número de sinais necessários para controlar o dispositivo experimental, um PLC pode manipular várias experiências num nó de laboratório [8].

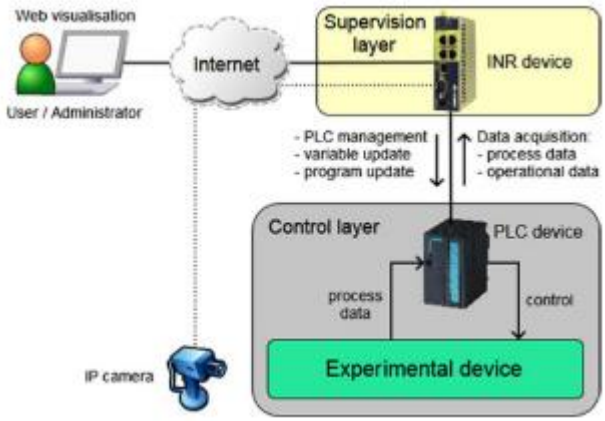


Figura 14 - Configuração operacional da arquitetura do laboratório remoto [8].

A estrutura física é projetada para remover quaisquer requisitos de programação para novas instâncias de laboratório e para reduzir o procedimento de desenvolvimento para algumas etapas triviais de conexão e configuração. Esta característica é alcançada pelo novo conceito multiuso de projeto de arquitetura de *hardware* e *software*. Como resultado dessa abordagem, economias significativas podem ser alcançadas em termos de tempo e esforço para o desenvolvimento de novos laboratórios. Por outro lado, a construção física da arquitetura limita o seu uso a uma faixa específica (mas ampla) de aplicativos. Em geral, a arquitetura pode atender a todos os tipos de experiências com PLCs, bem como a qualquer tipo de laboratório ou aparelho industrial que possa ser conectado e controlado por PLCs. Isso torna a arquitetura geralmente adequada para laboratórios remotos de controlo automático, onde processos tecnológicos reais são usados [8].

3 AUTÓMATOS PROGRAMÁVEIS

Resumo

Neste capítulo, depois de uma introdução sobre autómatos programáveis, apresentam-se as principais características do autómato utilizado na plataforma de testes (Plataforma A), a aplicação de software CX-Programmer, linguagens de programação mais utilizadas e sistema de supervisão (SCADA).

3.1. Autómato Omron CP1L-M30DT-D

3.2. CX-Programmer

3.3. Linguagens de Programação

3.4. GRAFCET

3.5. Ciclo Scan

3.6. Norma 61499

3.7. SCADA

Autómatos Programáveis

De acordo com a NEMA (National Electrical Manufactures Assossiation), um Autómato Programável ou Controlador Lógico Programável (CPL ou PLC na sigla inglesa) é um *"Aparelho eletrónico digital que utiliza uma memória programável para o armazenamento interno de instruções para implementações específicas, tais como lógica, sequenciamento, temporização, contagem e aritmética, para controlar, através de módulos de entradas e saídas, vários tipos de máquinas ou processos"* [9]. Um PLC é um pequeno computador com um sistema operativo (OS) integrado. Este sistema operativo é altamente especializado e otimizado para lidar com eventos recebidos em tempo real, ou seja, no momento de sua ocorrência.

Os PLCs são atualmente a tecnologia de controlo de processo industrial amplamente utilizada. O autómato programável eliminou uma grande parte do *hardwiring* associado aos circuitos de controlo de relés convencionais fazendo com que alterações de automatismo não originassem alterações de cablagem, passando estas a serem efetuadas ao nível do programa. Também, a redução de componentes mecânicos possibilitou um incremento na fiabilidade dos circuitos. Outra grande vantagem é a possibilidade de o desenvolvimento do programa poder ser efetuado em paralelo com a instalação dos equipamentos.

Como a estrutura de um PLC é baseada nos mesmos princípios usados na arquitetura de computadores, ele é capaz não apenas de executar as tarefas de comutação de relés, mas também de executar outras funções, tais como temporização, contagem e comparação bem como processamento de sinais analógicos. Ao contrário de um computador pessoal, um PLC é projetado para operar no ambiente industrial, sendo equipado com interfaces especiais de entrada / saída e uma linguagem de programação de controlo [10].

A Figura 15 mostra algumas das funções aplicadas na indústria.

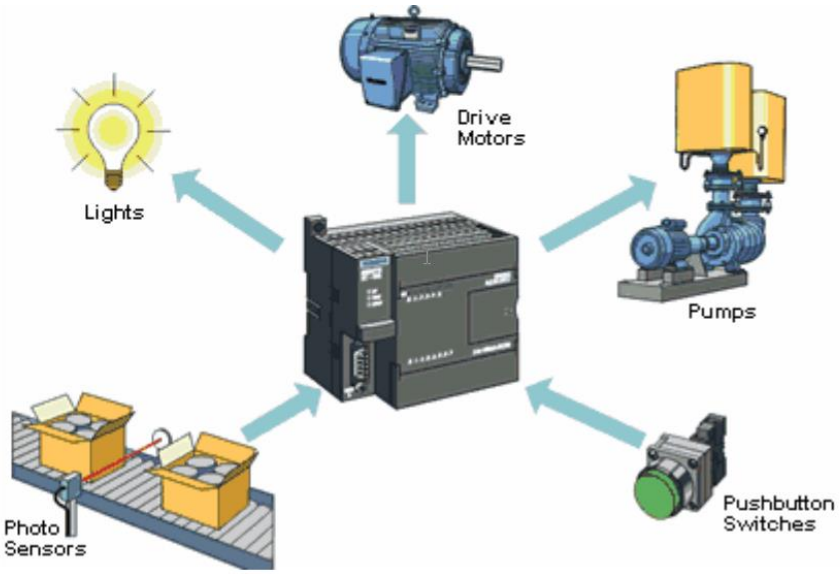


Figura 15 - PLC aplicado à Indústria [12].

3.1 Autómato Omron CP1L-M30DT-D

O autómato utilizado na plataforma A, que de seguida abordaremos com maior detalhe, é da Marca *Omron*, modelo CP1L-M30DT-D.

Na Figura 16, é possível ver de forma esquematizada a sua configuração.

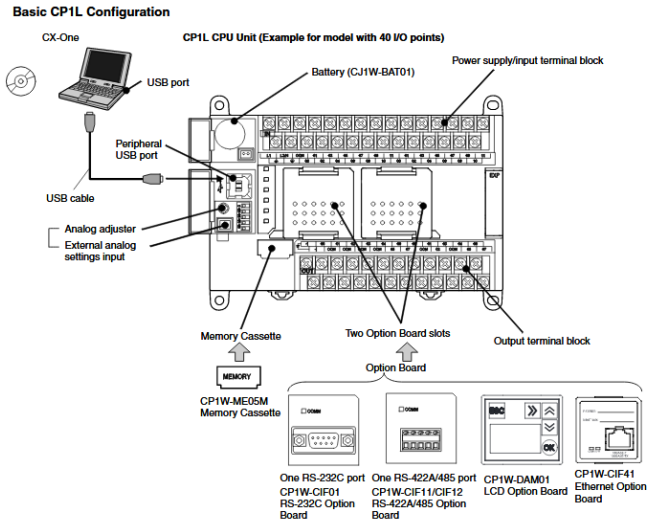


Figura 16 - Autómato CP1L-M30DT-D [11].

3.1.1 Entradas

O circuito de entradas deste autómato, possui 18 pontos comuns que podem ser conectados diretamente à saída do transformador de 24V DC, ou então, como opção, ligadas ao ponto neutro desse mesmo transformador.

Como opção, e para o projeto em curso, optou-se por considerar que os pontos comuns seriam conectados ao neutro (0 V). Assim, e para este caso em concreto, o positivo proveniente do transformador foi ligado ao ponto positivo (+) do autómato e o negativo do transformador ao ponto negativo (-) do autómato, aliás como seria de esperar, no entanto, e por opção, como já atrás referido, o ponto comum (COM) foi ligado ao ponto negativo (-). Caso se optasse por ter o ponto comum (COM) como positivo (+), então este teria de ser ligado ao ponto positivo (+).

A Figura 17 apresenta a configuração das referidas entradas.

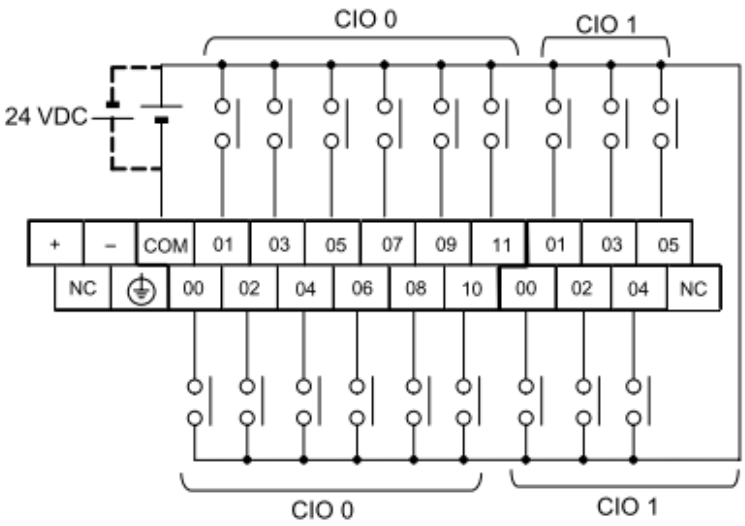


Figura 17 - Entradas [11].

3.1.2 Saídas (*sinking* transístor)

As saídas deste autómato são efetuadas através de transístores em configuração tipo *sinking.* Esta configuração (tipo *sinking*), garante que a carga será "afundada" à terra (ou neutro) através do transístor de saída e pressupõe que a carga será ligada em modo *sourcing* (fonte), ou seja, ao coletor do transístor de saída.

Na figura 18 está representada o esquema de ligação das saídas.

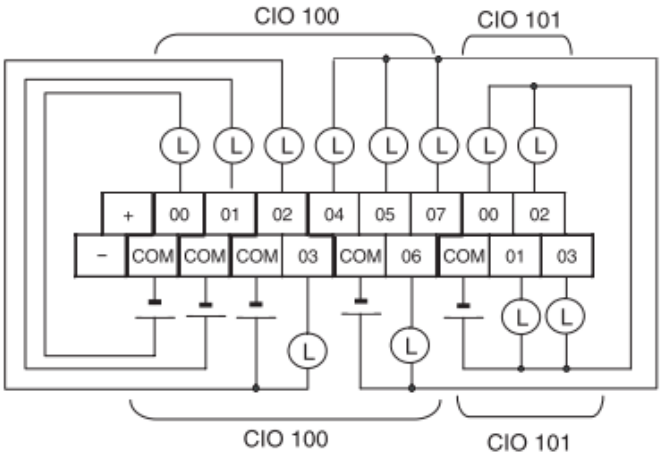


Figura 18 - Saídas [11].

Pese embora todas as 12 saídas sejam do tipo *sinking*, as 4 primeiras (100.00 a 100.03), diferem das restantes 8 (100.04 a 101.03). Isto deve-se apenas a uma opção tomada pelo fabricante, neste caso a *Omron*, para facilitar o uso de determinadas saídas em funções com diferentes configurações.

As seguintes Figuras (19 e 20), apresentam o diagrama esquemático simplificado das referidas configurações.

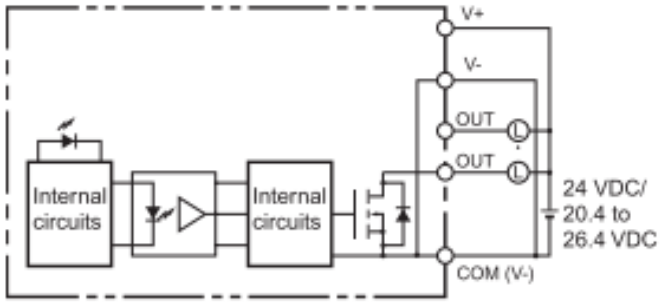


Figura 19 - Saídas 100.00 a 100.03 [11].

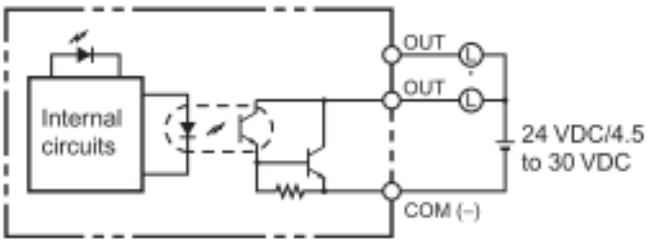


Figura 20 - Saídas 100.04 a 101.03 [11].

3.1.3 *Memory Area*

Esta região de memória (E/S) contém as áreas de dados que podem ser acedidas como operandos de instrução. Esta área de memoria está, no entanto, ainda subdividida em várias regiões pré alocadas a *CIO Area, Work Area*, *Holding Area, Auxiliary Area*, *DM Area, Timer Area, Counter Area, Task Flag Area*, *Data Registers, Index Registers, Condition Flag Area and Clock Pulse* Area como podemos ver na Figura 21.

.

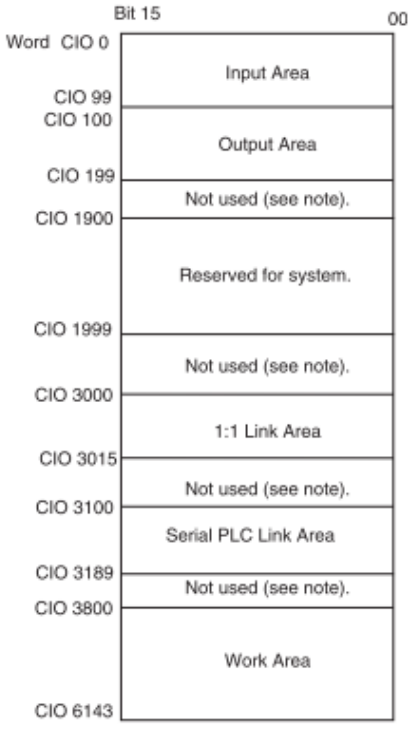


Figura 21 - Área CIO [11].

3.1.3.1 *I/O Area*

A área CIO tem definida para uso 6144 *words*. As primeiras 200 *words* (CIO 0 a CIO 199) estão alocadas às entradas e saídas embebidas, sendo que as 100 primeiras estão alocadas às entradas e as restantes 100 às saídas. Assim, estão disponíveis para uso tanto das entradas como das saídas 100 *words* cada que correspondem a $100\ Words\ x\ 2\ Bytes = 200\ Bytes$ ou $200\ Bytes * 8\ Bits = \ 1600\ Bits$

3.1.3.2 *1:1 Link Area*

Na área compreendida entre CIO 3000 e CIO 3015, os 256 *bits* ($16\ Words * 2\ Bytes * 8\ Bits = 256\ Bits$) podem ser utilizados em ligações mestre / escravo numa relação 1:1 entre unidades CPU do tipo CP1L e CPM2. Este tipo de ligação é utilizado para ligação em série entre dois PLC'S via RS-232C (Figura 22).

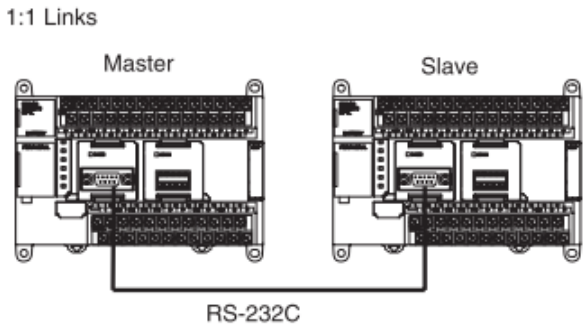


Figura 22 - Área de ligação série [11].

3.1.3.3 *Serial PLC Link Area*

As *Words* alocadas nesta área correspondem à área compreendida entre a CIO 3100 e CIO 3189. Corresponde a um espaço de 90 *Words* e estão alocadas ao uso de ligações em serie *(Serial PLC Links)* entre PLC'S com CPU do tipo CP1L oi CP1H. No entanto, o espaço não utilizado por estas ligações pode ser utilizado na programação. Este tipo de ligação (série via RS-232C) possibilita a troca de dados entre os CPU'S sem necessidade de programação especial (Figura 23).

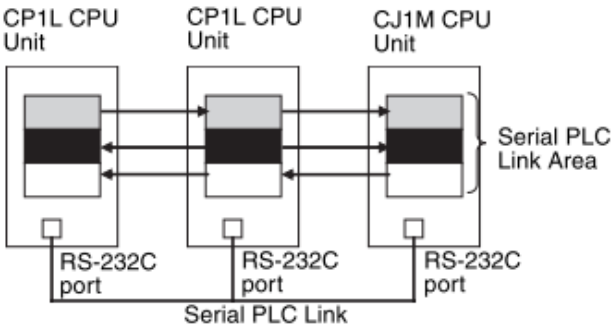


Figura 23 - Ligação série entre PLCs [11].

3.1.4 *Work Area (W)*

Words da área de trabalho *(Work Area)* podem ser utilizadas na programação e não podem ser utilizadas para troca de informação das entradas e saídas e os terminais externos E/S. Estas *words* e *bits* da área de trabalho devem ser usados antes da utilização das *words* da CIO *Area*.

Esta área está compreendida entre a *word* W000 e W511 e corresponde a um total de 512 *words*. A sua representação está definida na Figura 24.

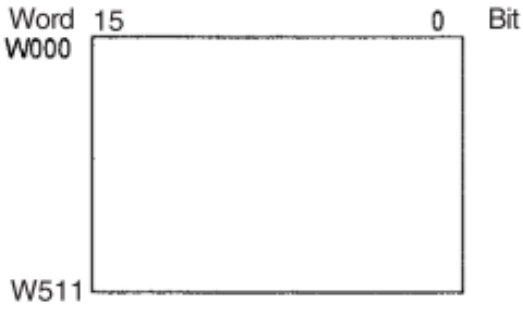


Figura 24 - Área de trabalho [11].

3.1.5 *Holding Area*

Words na *Holding Area* podem ser utilizados na programação. Estas *words* retêm a informação quando o PLC é ligado ou o modo de operação é modificado entre *PROGRAME mode* e *RUN* ou *MONITOR mode*. Esta área está compreendida entre a *word* H000 e H511 e corresponde a um total de 512 *words*.

Existe uma área suplementar de 24 *words* definida entre a *word* H512 e H535 que pode ser utilizada como bloco de funções, no entanto só podem ser utilizadas para instanciar bloco de funções *(internally allocated variable area).* Estas *words* não podem nunca ser especificadas como operadores de instrução na programação. A sua representação está definida na Figura 25.

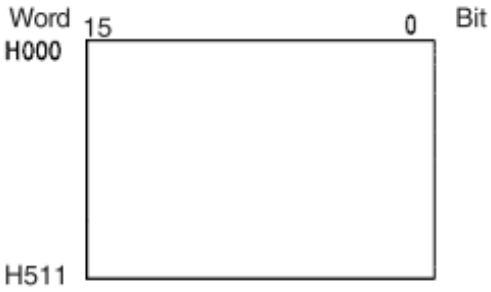


Figura 25 - Área de espera [11].

3.1.6 *Auxiliary Area (A)*

Estas *words* estão alocadas a funções especificas no sistema e são áreas de memória apenas de leitura *(Read-only area)* usadas nas configurações iniciais do autómato e podem ser consultadas no apêndice C e D do manual do PLC. Como de pode verificar na Figura 26, está subdividida em duas áreas A000 a A447 (448 *words*), apenas de leitura e A448 a A959 (512 *words*), de leitura e escrita.

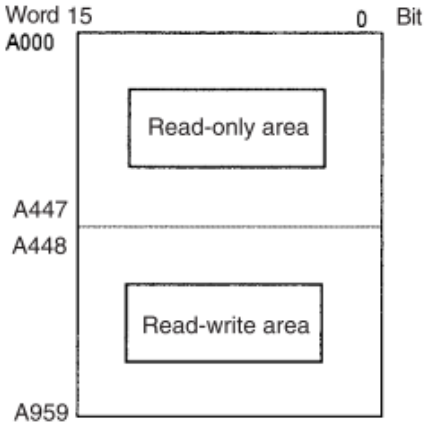


Figura 26 - Área auxiliar [11].

3.1.7 *Temporary Relay Area (TR)*

A área TR contem os *bits* que registam o estado ON/OFF dos diferentes ramos do programa. Esta area é composta por 16 *bits* endereçadas entre TR0 e TR15,

3.1.8 *Data Memory Area (D)*

A área DM representada na Figura 27 é uma área de dados multiuso que normalmente é acedida apenas em unidade de *words*. Estas *words* retêm o seu conteúdo quando o PLC é ligado ou o modo de operação é alterado entre *PROGRAM mode* e *RUN mode* ou *MONITOR*.

A área compreendida entre as *Words* D0 e D32767 representam o espaço de 32768 *words* ao qual correspondem 524288 *bits* ou 65536 *Bytes* ou 64K *Bytes*.

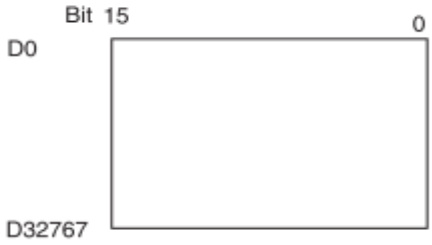


Figura 27 - Área de memória de dados [11].

3.1.9 *Timer Area (T)*

A área T, ou *Timer Area*, é composta por duas partes: A *Timer Completion Flags* e a *Timer Present Values* (PVs). Podem ser utilizados até 4096 timers entre a área definida por T0 e T4095.

As *Flags* correspondentes à área do *Timer Completion Flags* são lidas como *bits* individuais e são colocadas a ON pelo sistema quando o correspondente *timer* passa a OFF, ou seja, termina a contagem. Já a *Timer* PVs são de leitura e escrita em *words* e efetua a contagem crescente ou decrescente conforme definido para o *timer*.

3.1.10 *Counter Area(C)*

A área C, ou *Counter Area*, é composta por duas partes: A *Counter Completion Flags* e *a Counter Present Values* (PVs). Podem ser utilizados até 4096 contadores entre a area definida por C0 e C4095.
As *Flags* correspondentes à área do *Counter Completion Flags* são lidas como *bits* individuais e são colocadas a ON pelo sistema quando o correspondente contador passa a OFF, ou seja, termina a contagem. Já a *Counter* PVs é de leitura e escrita em *words* e efetua a contagem crescente ou decrescente conforme definido para o contador.

3.1.11 *Condition* Flags

Estas *Flags* incluem as *Flags* Aritméticas, como a *Flag* de Erro e a *Flag* Igualdade, que indicam os resultados da execução da instrução, bem como as *Flags* Sempre Ligado e Sempre Desligado. As *Flags* de condição são especificadas com símbolos em vez de endereços.

3.1.12 *Task Flags Area (TK)*

Uma *Flag* de Tarefa estará a ON quando a tarefa cíclica correspondente estiver no estado executável *(RUN)* e OFF quando a tarefa cíclica não tiver sido executada *(INI)* ou estiver no estado standby *(WAIT).*

3.1.13 *Index Registers (IR)*

Registos de índice (IR0 a IR15) são usados para armazenar endereços de memória do PLC (ou seja, endereços de memória absoluta na RAM) para endereçar indiretamente as palavras na memória de E/S. *Os Index Registers* podem ser usados separadamente em cada tarefa ou podem ser compartilhados por todas as tarefas.

3.1.14 *Data Registers (DR)*

Registos de dados (DR0 a DR15) são usados em conjunto com os Registos de Índice. Quando um registo de dados é inserido logo antes de um registo de índice, o conteúdo do registo de dados é adicionado ao endereço de memória do PLC no registo de índice para compensar esse endereço. Os registos de dados podem ser usados separadamente em cada tarefa ou podem ser compartilhados por todas as tarefas.

3.2 CX-Programmer

O software utilizado na programação do autómato *Omron*, é uma aplicação desenvolvida pela referida marca e que inclui uma ampla variedade de recursos permitindo ao utilizador, desenvolver o código necessário para os testes a efetuar.

Este software, disponibilizado pela marca para download na sua página *web* inclui um pacote de funcionalidades capazes de integrar projetos de grande dimensão com ferramentas avançadas e um controlo total do projeto.

Para o projeto em curso, fui utilizada a linguagem LADDER, de entre outras disponibilizadas pelo referido software, que permite para além da programação, a visualização e simulação para testes de funcionamento antes do software ser carregado para o autómato, prevenindo assim eventuais erros e danos no autómato (Figura 28).

Figura 28 - Software CX-Programmer da Omron

3.3 Linguagens de Programação

Algumas linguagens de programação utilizadas incluem:

- LADDER
- Diagrama de blocos funcional
- Texto Estruturado
- Lista de instruções

LADDER

LADDER é uma linguagem gráfica composta por diagramas de contactos e como é visível na Figura 29, assemelha-se a um sistema elétrico com interruptores e lâmpadas, onde os interruptores foram substituídos por contactos e as lâmpadas por bobinas. Para além de ser a primeira linguagem desenvolvida para PLCs é também a mais utilizada.

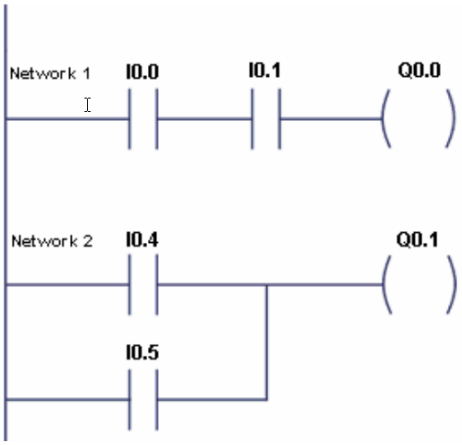


Figura 29 - Linguagem LADDER [9].

Linguagem de diagrama de blocos funcional

A linguagem de diagrama de blocos funcional, utiliza blocos de funções como o que por exemplo está representado na Figura 30. O exemplo apresentado, é um circuito *AND*, onde a saída ficará ativa quando as duas entradas estiverem no nível lógico 1. Embora o exemplo apresentado seja um circuito lógico simples, é possível conjugar diversos circuitos formando um bloco de funções mais complexo.

Figura 30 - Diagrama de blocos [9].

Linguagem de texto estruturado

Texto Estruturado *(Structured Text – ST)* é uma linguagem de alto nível muito poderosa, com raízes em Pascal e "C". Contém todos os elementos essências de uma linguagem de programação moderna, incluindo condicionais (IF-THEN-ELSE e CASE OF) e iterações (FOR, WHILE e REPEAT) [12].
Um exemplo pode ser visto na Figura 32.

```
I:=25;
WHILE J<5 DO
        Z:= F(I+J);
END_WHILE

IF B_1 THEN
        %QW100:= INT_TO_BCD(Display)
ENDIF

CASE TW OF
        1,5:    TEMP := TEMP_1;
        2:      TEMP := 40;
        4:      TEMP := FTMP(TEMP_2);
ELSE
        TEMP := 0;
        B_ERROR :=1;
END_CASE
```

Figura 31 - Texto estruturado [12].

Lista de instruções

É uma linguagem de baixo nível (linguagem máquina), que é utilizada por programadores com mais experiência, no entanto trata-se de uma linguagem muito poderosa e eficiente.

Algumas das operações, são matemáticas, como por exemplo, somar, subtrair, multiplicar ou dividir valores, enquanto outras operações podem incluir saltos para determinados pontos ou até o retorno de determinada função [13].
Um exemplo de uso desta linguagem pode ser visualizado na Figura 32.

```
LD %I1.1
R   %C8
LD  %I1.2
AND %M0
CU  %C8
LD  %C8.D
ST  %Q2.0
```

Figura 32 - Lista de instruções [13].

3.4 GRAFCET

O GRAFCET, muitas vezes conotado como uma linguagem, trata-se, no entanto, de um formalismo, que por representação gráfica, permite representar o comportamento de sistemas sequenciais (Figura 33).

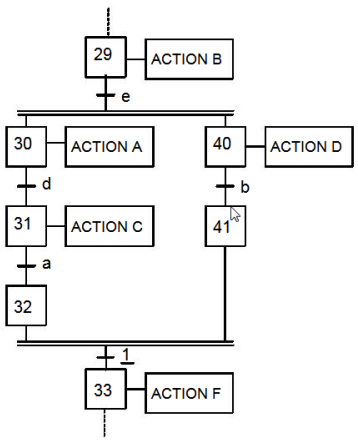


Figura 33 - GRAFCET [9].

3.5 Ciclo Scan

O ciclo scan, num PLC, é um fator muitas vezes levado em linha de conta e seletivo na escolha destes equipamentos. A sua importância reflete-se na escolha porque este é responsável pela rapidez em que este ciclo é processado pelo microprocessador, ou seja, pelo tempo de resposta à solicitação de verificação, armazenamento e processamento da informação.

Assim, o tempo de scan, é não mais que o somatório dos tempos de execução, após a inicialização, e corresponde aos tempos de leitura dos valores das entradas, registo em memória desses valores e respetiva manipulação das saídas, advindo daí a sua importância na eficiência dos tempos de resposta destes equipamentos, para sistemas onde a exigência de velocidade de processamento é um fator de seleção para uma maior produtividade.

3.6 Norma 6149

Sistemas de automação são normalmente projetados para utilização de PLCs, sendo que para sistemas de grande dimensão onde estes comunicam por rede e controlam diversos sensores e atuadores a implementação de sistemas distribuídos fica por si só dificultada em alterações futuras pelas restrições de uso de ferramentas de fabricantes diversos, tornando as empresas totalmente dependentes de um único fabricante.

Por sua vez, a norma 61499, acabou por trazer uma alternativa viável para este tipo de configurações onde a implementação de sistemas distribuídos acabou por ser bastante facilitada a todos os sistemas que utilizam esta norma, garantindo portabilidade e compatibilidade entre eles.

O uso de blocos de funções num modelo orientado a componentes facilita por si só uma reconfiguração dinâmica do sistema, em modo de execução e sem necessidade de interrupções facilitando também o uso de componentes de diversos fabricantes, diversificando a escolha, tornando as alterações menos restritivas, mais fáceis e por sua vez mais económicas.

3.7 SCADA – Controlo de Supervisão e aquisição de dados

SCADA são sistemas que que utilizam software para monitorizar sistemas de controlo.

A arquitetura básica do SCADA consiste na monitorização de sistemas que utilizam controladores lógicos programáveis (PLCs) ou unidades terminais remotas (RTUs). PLCs e RTUs são microcomputadores que se comunicam com uma série de objetos, como máquinas de fábrica, HMIs, sensores e dispositivos finais, e então encaminham as informações desses objetos para os computadores com o software SCADA. O *software* SCADA processa, distribui e exibe os dados, ajudando os operadores e outros funcionários a analisar os dados e tomar decisões importantes.

Por exemplo, o sistema SCADA notifica rapidamente um operador de que um lote de produto está apresentando uma alta incidência de erros. O operador faz uma pausa na operação e visualiza os dados do sistema SCADA por meio de uma HMI para determinar a causa do problema. O operador revê os dados e descobre que por exemplo a Máquina 4 não estava a funcionar corretamente. A capacidade do sistema SCADA de notificar o operador sobre um problema ajuda-o a resolvê-lo e a evitar mais perdas de produto[14].

Como exemplos de sistemas SCADA dos muitos utilizados por diversas empresas para supervisionar e controlar por exemplo subestações de energia (EDP), Controlo de tráfego (AP), Caminhos de ferro (CP), entre outros, utilizam normalmente interfaces do tipo HMI, como o apresentado na Figura 34, ou salas de controlo e supervisão como a que pode ser vista na Figura 35, pertencente à empresa Despachos de Lisboa.

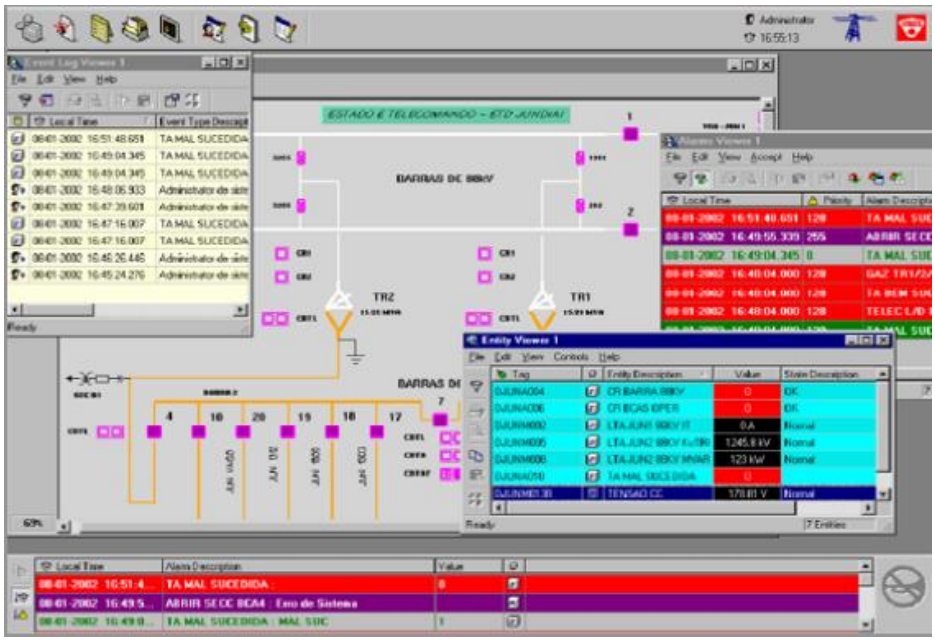


Figura 34 - Interface gráfica HMI [15].

Figura 35 - Despacho de Lisboa [15].

4 PROTOCOLO DE COMUNICAÇÃO

Resumo

Neste capítulo apresenta-se o protocolo de comunicação escolhido para o projeto.

4.1. *Ethernet*

4.2. TCP/IP

4.3. *Modbus*

4.4. *Wireless Modbus*

4.5. Modelo de Dados *Modbus TCP/IP*

Protocolo de comunicação

O protocolo de comunicação selecionado para o projeto em curso, foi o *ModBus* TCP/IP. Esta escolha deve-se essenciamente ao facto de ser o protocolo que mais se aproxima das necessidades e exigencias à realização do projeto. A imposição criada por questões essencialmente de segurança, criada pelos fabricantes de automatos, que impõe restrições e inibe o acesso a determinados pontos da memoria, limita por si só uma realização mais abrangente do que seria espetavel num projeto como este e que mais à frente serão abordados em maior detalhe. Para um melhor entendimento do que envolve a utilização deste protocolo *(Modbus* TCP/IP), serão abordados alguns elementos essenciais para um perfeito entendimento.

4.1 *Ethernet*

Ethernet é um protocolo de rede doméstica de longa data que ganhou aceitação universal em todo o mundo. Também é um padrão aberto que é suportado por muitos fabricantes e a sua infraestrutura está amplamente disponível e amplamente instalada. Consequentemente, o seu conjunto de protocolos TCP/IP é usado em todo o mundo e serve como base para o acesso à *World Wide Web*. Como muitos dispositivos já suportam *Ethernet*, é natural aumentá-lo para uso em aplicações industriais. A *Ethernet* é mais determinista através do uso de *switches Ethernet* rápidos para dispositivos de interconexão.
Esses *switches* aumentam a largura de banda de redes grandes subdividindo-as em várias redes menores ou em "domínios de colisão" separados. O *switch* também minimiza a vibração da rede, facilitando uma ligação direta entre um remetente e um recetor de tal forma que apenas o recetor receba os dados, não toda a rede [17].

4.2 TCP/IP

TCP/IP refere-se ao Protocolo de Controlo de Transmissão e Protocolo da *Internet*, que foram introduzidos pela primeira vez em 1974. O TCP/IP é a base da *World Wide Web* e forma o protocolo de camada de rede e transporte da *Internet* que comumente conecta todas as instalações de *Ethernet* em todo o mundo. Em termos simples, o TCP/IP permite que blocos de dados binários sejam trocados entre computadores. A principal função do TCP é garantir que todos os pacotes de dados sejam recebidos corretamente, enquanto o IP garante que as mensagens sejam endereçadas e roteadas corretamente.

O TCP/IP não define o que os dados significam ou como os dados devem ser interpretados, é apenas um protocolo de transporte.

4.3 *Modbus*

O *Modbus* é um protocolo de comunicação em serie, desenvolvido pela *Modicon* em 1979 como meio de comunicação com dispositivos eletrônicos industriais, como PLC, PAC ou DCS, em fios de par entrelaçado. O *Modbus* é um protocolo simples e robusto que se tornou o padrão na indústria de automação e controlo de processos. Ele permite a comunicação entre muitos dispositivos (aproximadamente 240) conectados na mesma rede, ou seja, um sistema que mede fenômenos físicos como temperatura, corrente, tensão e humidade comunicando os resultados para um mestre (computador). O *Modbus* é frequentemente usado para ligação entre um computador de supervisão e uma unidade terminal remota (RTU) em sistemas de controlo de supervisão e aquisição de dados (SCADA) [16].

Os dispositivos *Modbus* comunicam usando a técnica mestre-escravo (cliente-servidor) na qual apenas um dispositivo (mestre / cliente) pode iniciar transações (chamadas de consultas). Os outros dispositivos (escravos/servidores) respondem fornecendo os dados solicitados ao mestre ou executando a ação solicitada na consulta. Um escravo é qualquer dispositivo periférico (transdutor de E/S, válvula, unidade de rede ou outro dispositivo de medição) que processa informações e envia a sua informação da saída para o mestre usando o *Modbus*. Os Módulos de E/S *Omron*, formam dispositivos escravos/servidores, enquanto um dispositivo mestre típico é um computador *host* que executa uma aplicação apropriada. Outros dispositivos podem funcionar como clientes *(masters)* e servidores (escravos)[17].

Os mestres podem endereçar escravos individuais ou podem iniciar uma mensagem de difusão para todos os escravos. Os escravos retornam uma resposta a todas as consultas endereçadas a eles individualmente, mas não respondem a consultas de difusão. Os escravos não iniciam mensagens sozinhos, eles apenas respondem às consultas do mestre. A consulta de um mestre consistirá num endereço de escravo (ou endereço de *broadcast*), um código de função que define a ação solicitada, qualquer dado necessário e um campo de verificação de erro. A resposta dum escravo consiste em campos que confirmam a ação tomada, qualquer dado a ser retornado e um campo de verificação de erro. A consulta e a resposta incluem um endereço de dispositivo, um código de função, mais dados

aplicáveis e um campo de verificação de erros. Se nenhum erro ocorrer, a resposta do escravo contém os dados conforme solicitado. Se ocorrer um erro na consulta recebida ou se o escravo não puder executar a ação solicitada, retornará uma mensagem de exceção como resposta [17].

4.3.1 *Modbus* ASCII

No *Modbus* ASCII, todas as mensagens são codificadas em hexadecimal usando caracteres ASCII de 4 *bits*. O *Modbus* ASCII marca o início de cada mensagem com um caractere de dois pontos ":" (hexadecimal 3A). O final de cada mensagem termina com os caracteres de retorno e de avanço de linha (hexadecimal 0D e 0A). Isso permite que o espaço entre os *bytes* seja variável, tornando-o adequado para transmissão através de alguns modems.

O *Modbus* ASCII é o mais lento dos três protocolos, mas é adequado quando são utilizadas ligações via modem ou rádio frequência (RF). Isso ocorre porque o ASCII utiliza caracteres para delimitar uma mensagem. Devido a esta delimitação da mensagem, quaisquer atrasos no meio de transmissão não farão com que a mensagem seja mal interpretada pelo dispositivo recetor. Isso pode ser importante ao lidar com modems lentos, telefones, telemóveis, ligações ruidosas ou outros meios de transmissão difíceis [16].

4.3.2 *Modbus* RTU

No *Modbus* RTU, os dados são codificados em binário e requerem apenas um *byte* de comunicação por *byte* de dados. Isso é ideal para uso em redes RS232 ou RS485 multiponto, em velocidades de 1.200 a 115.000 bit/s. As velocidades mais comuns são 9.600 e 19.200 bit/s. O *Modbus* RTU é o protocolo industrial mais amplamente usado [16].

4.3.3 *Modbus* TCP/IP

O *Modbus* / TCP (também *Modbus*-TCP/IP) é simplesmente o protocolo *Modbus* RTU com uma interface TCP que é executada na *Ethernet*. A estrutura de mensagens *Modbus* é o protocolo de aplicação que define as regras para organizar e interpretar os dados independentemente do meio de transmissão de

dados. TCP/IP refere-se ao Protocolo de Controlo de Transmissão e ao Protocolo da *Internet*, que fornece a *mídia* de transmissão para o sistema de mensagens *Modbus* TCP/IP [17].

Isto permite que os dispositivos *Modbus* / TCP comuniquem entre si de maneira fácil e imediata através das redes *Ethernet* existentes. O *Modbus* / TCP também permite muitos mais endereços do que o RS485, o uso de múltiplos *Masters* e velocidades na ordem dos *Gigabit*. Embora o *Modbus* RTU tenha uma limitação de 247 nós por rede, as redes *Modbus* / TCP podem ter tantos escravos quanto a camada física puder suportar. Muitas vezes esse número é de cerca de 1.024. A rápida adoção da *Ethernet* na indústria de automação e controlo de processos permitiu que o *Modbus* / TCP se tornasse o protocolo industrial mais amplamente utilizado, de mais rápido crescimento e suportado pela *Ethernet* [17].

Ao contrário do *Modbus* RTU e do *Modbus* ASCII, o *Modbus* / TCP permite que múltiplos mestres pesquisem o mesmo dispositivo escravo simultaneamente. Isso é permitido porque, por meio da *Ethernet* via TCP / IP, várias mensagens podem ser enviadas, armazenadas em *buffer* e entregues sem o requisito de passagem de *token* ou controlo total de barramento, o que geralmente é o caso de muitos protocolos RS485 e RS422 [16].

Portanto, e em resumo, o *Modbus* TCP/IP usa TCP/IP e *Ethernet* para transportar os dados da estrutura de mensagens do *Modbus* entre dispositivos compatíveis. Ou seja, o *Modbus* TCP/IP combina uma rede física *(Ethernet)*, com um padrão de rede (TCP/IP) e um método padrão de representação de dados (*Modbus* como protocolo de aplicação). Essencialmente, a mensagem *Modbus* TCP/IP é simplesmente uma comunicação *Modbus* encapsulada num *wrapper* (pacote) *Ethernet* TCP/IP [17].

Na prática, o *Modbus* TCP incorpora um quadro de dados *Modbus* padrão num quadro TCP, sem a soma de verificação *Modbus*, como podemos ver na Figura 36.

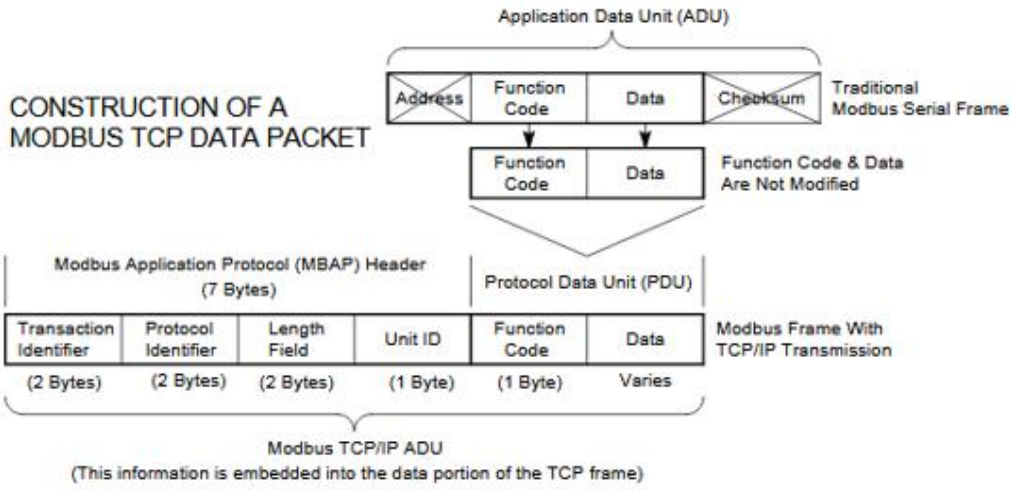


Figura 36 - Construção da trama Modbus TCP [17].

Os comandos do *Modbus* e os dados do utilizador são encapsulados no container de dados de um datagrama TCP/IP sem serem modificados de forma alguma. No entanto, o campo de verificação de erros do *Modbus* (soma de verificação) não é usado, pois os métodos padrão de soma de verificação da camada de enlace *Ethernet* TCP/IP são usados para garantir a integridade dos dados. Além disso, o campo de endereço do quadro *Modbus* é suplantado pelo identificador de unidade no *Modbus* TCP/IP e passa a fazer parte do cabeçalho do Protocolo de Aplicação *Modbus* (MBAP). A partir da Figura 36, vemos que o código de função e os campos de dados são absorvidos na sua forma original. Assim, uma Unidade de Dados de Aplicação *Modbus* TCP/IP (ADU) assume a forma de um cabeçalho de 7 *bytes* (identificador de transação + identificador de protocolo + comprimento de campo + identificador de unidade) e a unidade de dados de protocolo (código de função + dados).

O cabeçalho do MBAP tem 7 *bytes* de comprimento e inclui os seguintes campos:

- **Identificador de transação / invocação *(2 bytes):***

Este campo de identificação é usado para o pareamento de transações quando várias mensagens são enviadas pela mesma ligação TCP por um cliente sem esperar por uma resposta anterior.

- **Identificador de protocolo *(2 bytes):***

Este campo é sempre 0 para serviços *Modbus* e outros valores são reservados para extensões futuras.

- **Comprimento *(2 bytes):***

Este campo é uma contagem de *bytes* dos campos restantes e inclui o *byte* do identificador da unidade, o *byte* do código da função e os campos de dados.

- **Identificador da Unidade *(1 byte):***

Este campo é usado para identificar um servidor remoto localizado numa rede não TCP/IP (para ponte serie). Num aplicativo típico *Modbus* TCP/IP, a ID da unidade é definida como 00 ou FF, ignorada pelo servidor e simplesmente devolvida na resposta.

A Unidade de Dados de Aplicação *Modbus* TCP/IP completa é incorporada no campo de dados de uma trama TCP padrão e enviada via TCP para a conhecida porta do sistema 502, que é reservada especificamente para aplicações *Modbus*. Os clientes e servidores *Modbus* TCP/IP escutam e recebem dados *Modbus* através da porta 502 [17].

4.4 *Wireless Modbus*

A redução dos custos de instalação, em especial de cablagem é o principal benefício do uso de dispositivos sem fio em aplicações industriais. Esta tecnologia permite uma rápida instalação, bem como uma utilização facilitada em qualquer local remoto das instalações fabris.

O *Modbus* via *wireless* é transparente para o sistema de controlo ou *host* e dispositivos escravos. O sistema *host* não sabe se existe uma rede *Modbus* sem fio, pois não precisa lidar com isso. Quando um mestre *Modbus* faz um pedido a um escravo e os pacotes chegam ao *modem*, esse *modem* geralmente reordena os pacotes e encripta-os antes da transmissão. Uma vez que os pacotes RF (radiofrequência) são recebidos pelo *modem* "escravo", são encriptados e colocados de volta para representar um pacote *Modbus* válido. Assumindo que o pacote não tenha sido danificado ou corrompido, será enviado ao escravo destinado. O escravo responderá ao Mestre e o processo será iniciado novamente [16].

4.5 Modelo de dados Modbus TCP/IP

Como já antes referido, o *Modbus* TCP/IP usa TCP/IP e *Ethernet* para transportar os dados da estrutura de mensagens do *Modbus* entre os dispositivos.

O TCP/IP é, na verdade, formado a partir de um "conjunto" de protocolos nos quais toda a comunicação da *Internet* se baseia e que também é denominada de pilha de protocolos. Cada *host* ou router na *internet* deve executar uma pilha de protocolos. O uso da pilha de palavras refere-se ao Modelo de Referência em camadas TCP/IP simplificado ou "pilha" usado para projetar *software* de rede e delineado da seguinte maneira segundo o modelo OSI representado na Tabela 1.

Tabela 1 - Modelo OSI para a pilha de comunicação Modbus TCP/IP [17].

MODBUS TCP/IP COMMUNICATION STACK			
#	**MODEL**	**IMPORTANT PROTOCOLS**	**Reference**
7	Application	Modbus	
6	Presentation		
5	Session		
4	Transport	TCP	
3	Network	IP, ARP, RARP	
2	Data Link	Ethernet, CSMA/CD, MAC	IEEE 802.3 Ethernet
1	Physical	Ethernet Physical Layer	

Na Figura 37, podemos ver mais em detalhe a formação/construção do pacote de dados *Ethernet*-TCP/IP. Aqui é visível e também a partir do modelo OSI, que a camada 5 (ou camada da aplicação), responsável pelos dados do utilizador (ADU) e que é composta como já antes referido pelo código de função (FC) e dados a utilizar. Este pacote, se assim podemos chamar, representa os dados do protocolo *Modbus* e tem como capacidade máxima 1460 *bytes*. A camada seguinte (4), denominada de camada de transporte vai por sua vez adicionar uma *header* TCP, com o(s) número(s) de porta(s) pré-estabelecidas para a comunicação. Esta *header* tem um tamanho definido de 20 bytes.

A camada seguinte (camada de rede), adiciona ao pacote formado pela camada de transporte uma nova *header*, que contem o endereço IP. Esta *header* tem também por definição um tamanho de 20 *bytes*.

De seguida, a este "pacote" formado pela camada de rede, serão adicionados, dois novos dados, um de 14 *bytes* que contem a informação do endereço IP da *Ethernet* e um outro, que é responsável pela verificação da integridade dos dados formando assim a denominada *Data Link Layer* (camada de ligação).

Finalmente, a última camada, a camada física, responsável pelo transporte do "pacote" de dados formado. A esta camada pertencem os cabos, tipos de cabos, forma de sinal, entre outros.

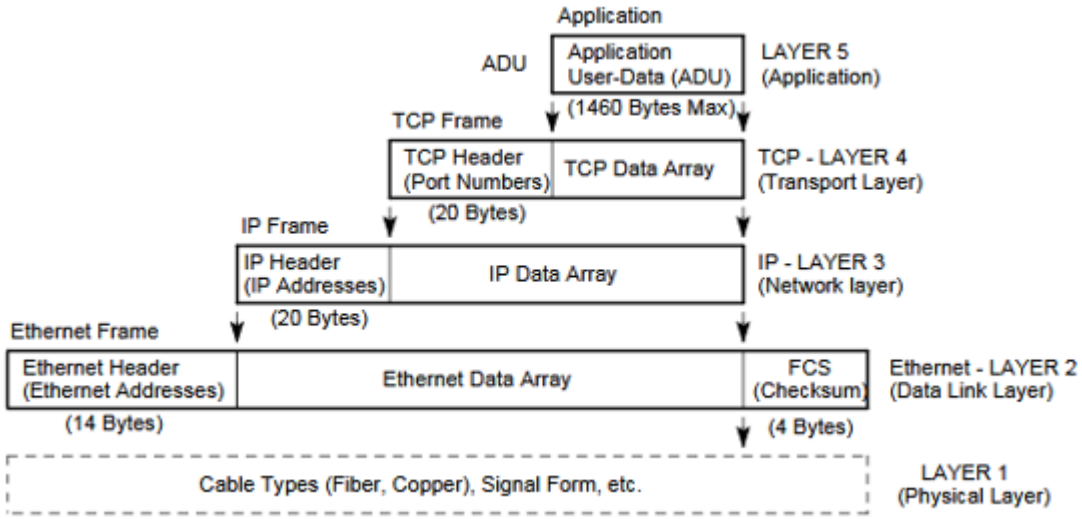


Figura 37 - Construção do pacote de dados Ethernet-TCP/IP [17].

O conjunto de protocolos TCP/IP (ou pilha de protocolos independentes) fornece todos os recursos para dois dispositivos se comunicarem entre si através de uma rede local (LAN) ou rede de longa distância global *(WAN).* Mas o TCP/IP apenas garante que as mensagens do aplicativo serão transferidas entre esses dispositivos, não garantindo que esses dispositivos realmente entenderão ou interajam entre si. Para o *Modbus* TCP/IP, esse recurso é fornecido pelo protocolo *Modbus* da camada de aplicação.

O *Modbus* opera de acordo com o modelo comum de cliente/servidor (mestre/escravo). Ou seja, o cliente (mestre) envia um pedido por mensagem (solicitação de serviço) ao servidor (escravo), e o servidor responde com uma mensagem de resposta. Se o servidor não puder processar uma solicitação, responderá com um código de função de erro (resposta de exceção) que é o código de função original mais 80H (isto é, com seu bit mais significativo definido como 1). As funções do *Modbus* operam nos registos de memória para configurar, monitorizar e controlar a E/S do dispositivo.

O modelo de dados *Modbus* possui uma estrutura simples que diferencia apenas quatro tipos básicos de dados:

- Entradas Discretas
- Bobinas (Saídas)
- Registos de entrada (dados de entrada)
- Retenção de Registos (Dados de Saída)

A solicitação de serviço (Unidade de Dados de Protocolo *Modbus*) é composta de um código de função e um número de *bytes* de dados adicionais, dependendo da função. Na maioria dos casos, os dados adicionais geralmente são uma referência variável, como um endereço de registo, já que a maioria das funções do *Modbus* operam com registo.

Os registos *Modbus* dum dispositivo são organizados em torno dos quatro tipos básicos de referência de dados indicados acima e esse tipo de dados é identificado adicionalmente pelo número inicial do endereço de referência, como se pode ver no quadro da Tabela 2:

Tabela 2 - Registos Modbus [17].

Reference	Description
0xxxx	Read/Write Discrete Outputs or Coils. A 0x reference address is used to drive output data to a digital output channel.
1xxxx	Read Discrete Inputs. The ON/OFF status of a 1x reference address is controlled by the corresponding digital input channel.
3xxxx	Read Input Registers. A 3x reference register contains a 16-bit number received from an external source—e.g. an analog signal.
4xxxx	Read/Write Output or Holding Registers. A 4x register is used to store 16-bits of numerical data (binary or decimal), or to send the data from the CPU to an output channel.

O campo de código de função da mensagem (PDU) contem um *byte* que informa ao escravo que tipo de ação deve ser executada. Códigos de função válidos são de 1-255, mas nem todos os códigos serão aplicados a um módulo e alguns códigos são reservados para uso futuro.

A tabela a seguir destaca um subconjunto de funções padrão do *Modbus* usualmente suportadas pelos módulos *Omron* (os endereços de registo de referência nos quais a função opera também são indicados). As funções abaixo são usadas para aceder os registos descritos no mapa de registo do módulo para envio e recebimento de dados. O comando *Report Slave ID* não opera no registo do mapa de registo.

Tabela 3 - Funções e registos Modbus [17].

CODE	FUNCTION	REFERENCE
01 (01H)	Read Coil (Output) Status	0xxxx
03 (03H)	Read Holding Registers	4xxxx
04 (04H)	Read Input Registers	3xxxx
05 (05H)	Force Single Coil (Output)	0xxxx
06 (06H)	Preset Single Register	4xxxx
15 (0FH)	Force Multiple Coils (Outputs)	0xxxx
16 (10H)	Preset Multiple Registers	4xxxx
17 (11H)	Report Slave ID	*Hidden*

O campo de dados solicitado pelo cliente fornece ao escravo (servidor) qualquer informação adicional requerida pelo escravo para completar a ação especificada pelo código de função na solicitação do cliente. O campo de dados geralmente inclui endereços de registos, valores de contagem e dados gravados. Para algumas mensagens, esse campo pode não existir (tem comprimento zero), pois nem todas as mensagens exigirão dados.

Por exemplo, o comando *Read Holding Registers* possui o código de função 0000 0011 (03H). Se o dispositivo escravo executar a ação solicitada sem erro, ele devolverá o mesmo código na resposta. No entanto, se ocorrer uma exceção, ele irá devolver 1000 0011 (83H) no campo de código de função e anexará um código exclusivo no campo de dados da mensagem de resposta que informará o mestre qual o tipo de erro que ocorreu ou o motivo da exceção

Quando o dispositivo escravo responde ao mestre, ele utiliza o campo de código de função para indicar uma resposta normal (sem erros) ou se algum tipo de erro ocorreu (uma resposta de exceção). Uma resposta normal simplesmente ecoa o código de função original da consulta, enquanto uma resposta de exceção devolve um código que é equivalente ao código de função original com seu bit mais significativo (msb) definido como valor logico 1.

A seguir e de forma mais detalhada, são apresentadas as funções disponíveis para comunicação e controlo do protocolo *Modbus*.

4.5.1 *Read Coil Status (01)*

Este comando irá ler o estado ON/OFF das saídas discretas ou bobinas (endereços de referência 0x) no escravo / servidor. Para módulos *Omron*, a sua resposta é equivalente a ler o *status* ativado / desativado de relés.

A consulta *Read Coil Status* especifica a bobina de partida (canal de saída) e a quantidade de bobinas a serem lidas. As bobinas correspondem aos relés de estado sólido discretos deste dispositivo e são endereçadas a partir de 0 (até 12 bobinas endereçadas como 0-11 para este modelo). Para módulos *Omron*, o *status* de saída é indicado como 1 para ON e 0 para OFF. O LSB do primeiro *byte* de dados corresponde ao *status* da bobina endereçada na consulta

Na tabela 4 é apresentado um exemplo a um pedido efetuado por uma *header Modbus* para a leitura das 4 primeiras bobinas (0 a 3).

Tabela 4 - Exemplo duma Header ADU [17].

Modbus Response ADU Example Header

MBAP Header Fields	Example Decimal (Hexadecimal)
Transaction ID High Order	0 (00) *Echoed back, no change*
Transaction ID Low Order	1 (01) *Echoed back, no change*
Protocol Identifier High Order	0 (00) *Echoed back, no change*
Protocol Identifier Low Order	0 (00) *Echoed back, no change*
Length High Order	0 (00) *Server calculates*
Length Low Order	4 (04) *Server calculates.*
Unit Identifier	255 (FF) or 0 (00) *No change*

O identificador de transação é usado para fazer corresponder a resposta com a consulta quando o cliente envia várias consultas sem esperar por uma resposta anterior. Normalmente, é um número de 1 a 16, mas o número máximo de transações do cliente e o número máximo de transações do servidor variam de acordo com o dispositivo. O identificador de protocolo é sempre 0 para o *Modbus*. O comprimento é uma contagem do número de *bytes* contidos nos dados mais o código da função *(1 byte)* e o identificador da unidade *(1 byte).*

A resposta ao pedido solicitado, pode ser visualizado na Tabela 5. Esta resposta efetua um eco da função utilizada (01), o número de *bytes* utilizados na transação e a resposta ao *status* das bobinas de saída.

Tabela 5 - Resposta à leitura do estado da bobina [17].

Modbus Response ADU Example - Read Coil Status Response

Field Name	Example Decimal (Hexadecimal)
Function Code	1 (01)
Byte Count	1 (01)
Data (Coils 3-0)	10 (0A)

Para resumir, o *status* das bobinas 0-3 é mostrado como o valor do *byte* 0A hexadecimal, ou 00001010 binário. A bobina 3 é o quinto *bit* da esquerda para a direita deste *byte* e a bobina 0 é o LSB. Os quatro *bits* restantes são zero. Lendo da esquerda para a direita, o *status* de saída das bobinas 3-0 é ON-OFF-ON-OFF (Tabela 6).

Tabela 6 – Exemplo de leitura do estado da bobina [17].

Bin	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**
Hex	**0**				**A**			
Coil	**NA**	**NA**	**NA**	**NA**	**3**	**2**	**1**	**0**

4.5.2 *Read Holding Registers (03)*

A comando *Read Holding Registers* (03) lê o conteúdo em binário dos registos de retenção (endereços de referência 4x) no dispositivo escravo. A consulta especifica o registo inicial e o número de registos a serem lidos.

No exemplo que se segue, serão lidos os registos de retenção 40006 a 40008. Na tabela 7 são apresentados, a *query* de consulta e a resposta ao pedido, respetivamente.

Tabela 7 - Leitura do registo de espera [17].

Modbus PDU Example - Read Holding Register Query

Field Name	Example Decimal (Hexadecimal)
Function Code	3 (03)
Starting Address High Order	0 (00)
Starting Address Low Order	5 (05)
Number Of Points High Order	0 (00)
Number Of Points Low Order	3 (03)

Modbus PDU Example - Read Holding Register Response

Field Name	Example Decimal (Hexadecimal)
Function Code	3 (03)
Byte Count	6 (06)
Data High (Register 40006)	(3A)
Data Low (Register 40006)	75%=15000 (98)
Data High (Register 40007)	(13)
Data Low (Register 40007)	25%=5000 (88)
Data High (Register 40008)	(00)
Data Low (Register 40008)	1%=200 (C8)

Resumindo este exemplo, o conteúdo do registo 40006 *(2 bytes)* é o MSB do canal 0 de 75% (15000 = 3A98H). O conteúdo do registo 40007 *(2 bytes)* é o LSB do canal 0 de 25% (5000 = 1388H). O conteúdo do registo 40008 é o valor da *dead band* do canal 0 *(2 bytes)* de 1% (200 = 00C8H).

4.5.3 *Read Input Registers (04)*

O comando *Read Input Registers* é responsável pela leitura do conteúdo binário dos registos de entrada (endereços de referência 3x) no dispositivo escravo.

Tabela 8 - Leitura dos registos de entrada [17].

Modbus PDU Example - Read Input Registers Query

Field Name	Example Decimal (Hexadecimal)
Function Code	4 (04)
Starting Address High Order	0 (00)
Starting Address Low Order	2 (02)
Number Of Points High Order	0 (00)
Number Of Points Low Order	2 (02)

Modbus PDU Example - Read Input Registers Response

Field Name	Example Decimal (Hexadecimal)
Function Code	4 (04)
Byte Count	4 (04)
Data High (Register 30003)	(3E)
Data Low (Register 30003)	80%=16000 (80)
Data High (Register 30004)	(00)
Data Low (Register 30004)	136 (88)

Em forma de resumo podemos dizer que neste exemplo o conteúdo do registo 30003 *(2 bytes)* é o valor de entrada do canal 1 de 80% (16000 = 3E80H). O conteúdo do registo 30004 *(2 bytes)* é a *Flag* de *status* do canal 0 de 136 (0088H).

4.5.4 *Force Single Coil (05)*

Força uma única bobina / saída (endereço de referência 0x) para ON ou OFF. A consulta *Force Single Coil* especifica o endereço de referência da bobina a ser forçada. O estado ON/OFF é indicado por meio de uma constante no campo de dados da consulta. Um valor de FF00H força a bobina a ser ligada (isto é, o correspondente relé de estado sólido é ligado ou fechado), e 0000H força a bobina a ser desligada (isto é, o relé de saída de estado sólido é desligado ou aberto). Todos os restantes valores são inválidos e não afetarão a bobina.

A resposta será sempre um eco à trama enviada, exceto ocorra algum erro. Neste caso, o código do erro será enviado na trama de resposta.

A tabela 9 apresenta um exemplo para a uma solicitação de ativação (ON) da saída 4. Neste caso como não ocorreu nenhum erro a resposta é um eco da solicitação.

.

Tabela 9 - Bobina em estado forçado [17].

Modbus PDU Example - Force Single Coil Query and Response

Field Name	Example Decimal (Hexadecimal)
Function Code	5 (05)
Coil Address High Order	0 (00)
Coil Address Low Order	3 (03)
Force Data High Order	255 (FF)
Force Data Low Order	0 (00)

4.5.5 *Preset Single Register (06)*

Este comando irá predefinir um único registo de retenção (endereço de referência 4x) para um valor específico.

A *query Preset Single Register* especifica o endereço de referência do registo a ser predefinido e o valor predefinido. A mensagem de resposta *Preset Single Registers* é um eco da consulta, devolvido após o conteúdo do registo ter sido predefinido.

No exemplo que se segue (ver tabela 10) é gravada uma taxa de transmissão de 9600*bps* para o registo 40002 (taxa de transmissão).

Tabela 10 - Registo predefinido [17].

Modbus PDU Example - Preset Holding Register Query and Response

Field Name	Example Decimal (Hexadecimal)
Function Code	6 (06)
Register Address High Order	0 (00)
Register Address Low Order	1 (01)
Preset Data High Order	0 (00)
Preset Data Low Order	2 (02)

A mensagem de resposta é simplesmente um eco (cópia) da consulta, conforme mostrado acima, devolvido após o conteúdo do registo ter sido predefinido.

4.5.6 *Force Multiple Coils (15)*

Simultaneamente, força uma série de bobinas (endereço de referência 0x) em ON ou OFF.

A consulta *Force Multiple Coils* especifica o endereço de referência da bobina inicial a ser forçado, o número de bobinas e os dados a serem gravados por ordem crescente. Os estados ON/OFF são especificados pelo conteúdo no campo de dados da consulta. Uma lógica 1 numa posição de *bit* deste campo solicita que a bobina seja ligada, enquanto uma lógica 0 solicita que a bobina correspondente seja desligada. *Bits* não utilizados num *byte* de dados devem ser definidos como zero. Observe-se que as bobinas são referenciadas começando em 0 até 4 bobinas são endereçadas como 0-3 para este exemplo e isso também corresponde ao número do canal de saída discreta.

Este exemplo representado na Tabela 11 força as saídas 1 e 3 OFF e 0 e 2 ON para as bobinas 0-3.

Tabela 11 - Múltiplas bobinas em estado forçado [17].

Modbus PDU Example - Force Multiple Coils Query

Field Name	Example Decimal (Hexadecimal)
Function Code	15 (0F)
Coil Address High Order	0 (00)
Coil Address Low Order	0 (00)
Number Of Coils High Order	0 (00)
Number Of Coils Low Order	4 (04)
Byte Count	01
Force Data High (First Byte)	5 (05)

Neste exemplo, o primeiro endereço é 00001 correspondente à bobina 0 e referenciado por 0000H. Assim, neste exemplo, o *byte* de dados transmitido endereçará as bobinas 0-3, com o *bit* menos significativo endereçando a bobina mais baixa neste conjunto como segue (observe que os quatro *bits* superiores não utilizados do *byte* de dados são ajustados para zero):

Tabela 12 - Resposta ao estado forçado de múltiplas bobinas [17].

Bin	0	0	0	0	0	1	0	1
Hex	0				5			
Coil	NA	NA	NA	NA	3	2	1	0

A mensagem de resposta *Force Multiple Coils* retorna o endereço do escravo, o código de função, o endereço inicial e o número de bobinas forçadas, depois de executar a instrução (Tabela 13).

Tabela 13 - Resposta ao estado forçado de múltiplas bobinas [17].

Modbus PDU Example - Force Multiple Coils Response

Field Name	Example Decimal (Hexadecimal)
Function Code	15 (0F)
Coil Address High Order	0 (00)
Coil Address Low Order	0 (00)
Number Of Coils High Order	0 (00)
Number Of Coils Low Order	4 (04)

4.5.7 *Preset Multiple Registers (16)*

A função 16 *(Preset Multiple Registers)* predefine um bloco de registos de retenção (endereços de referência 4x) para valores específicos.

Este comando especifica o endereço de referência do registo inicial, o número de registos e os dados a serem gravados em ordem crescente.

O exemplo que se segue representado na Tabela 14, grava um novo endereço escravo de 200, uma taxa de transmissão de 28800 *bps*, e define a paridade como par, escrevendo para registos de retenção

40001 a 40003 (as alterações no endereço do escravo, taxa de transmissão e paridade entrarão disponíveis apenas após a reinicialização do autómato).

Tabela 14 - Consulta de vários registos predefinidos [17].

Modbus PDU Example - Preset Multiple Registers Query

Field Name	Example Decimal (Hexadecimal)
Function Code	16 (10)
Starting Register High Order	0 (00)
Starting Register Low Order	0 (00)
Number Of Registers High Order	0 (00)
Number Of Registers Low Order	3 (03)
Preset Data High (First Register)	0 (00)
Preset Data Low (First Register)	200 (C8)
Preset Data High (Second Reg)	0 (00)
Preset Data Low (Second Reg)	5 (05)
Preset Data High (Third Reg)	0 (00)
Preset Data Low (Third Reg)	2 (02)

A mensagem de resposta à ação do comando *Preset Multiple Registers* retorna o endereço do escravo, o código de função, a referência do registo inicial e o número de registos definidos, após o conteúdo do registo ter sido definido.

A resposta pode ser observada na Tabela 15.

Tabela 15 - Resposta à predefinição de múltiplos registos [17].

Modbus PDU Example - Preset Multiple Registers Response

Field Name	Example Decimal (Hexadecimal)
Function Code	16 (10)
Starting Register High Order	0 (00)
Starting Register Low Order	0 (00)
Number Of Registers High Order	0 (00)
Number Of Registers Low Order	3 (03)

4.5.8 *Report Slave ID (17)*

Este comando retorna o número de modelo, número de serie e *firmware* de um dispositivo escravo / servidor *Acromag* (97xEN para este exemplo), o *status* do indicador *Run* e qualquer outra informação específica do dispositivo.

Tabela 16 - Informação à consulta para identificação do escravo[17].

Modbus PDU Example - Report Slave ID Query

Field Name	Example Decimal (Hexadecimal)
Function Code	17 (11)

A resposta para o dispositivo em questão pode ser observada na Tabela 17.

Tabela 17 - Resultado resposta à identificação do escravo [17].

Modbus PDU Example - Report Slave ID Response (Acromag 97xEN)

FIELD	DESCRIPTION
Unit ID	Echo Unit ID Sent In Query
Function Code	11
Byte Count	42
Slave ID (Model No.)	08=972EN-4004 (4 Current Outputs) 09=972EN-4006 (6 Current Outputs) 0A=973EN-4004 (4 Voltage Outputs) 0B=973EN-4006 (6 Voltage Outputs)
Run Indicator Status	FFH (ON)
Firmware Number String (Additional Data Field)	41 43 52 4F 4D 41 47 2C 39 33 30 30 2D **31 32 37** 2C 39 **37 32** 45 4E 2D **34 30 30 36** 2C *30 31 32 33 34 35 41* 2C *30 31 32 33 34 35* ("ACROMAG,9300-**127**,9**72**EN-**4006**,*serial number&rev*,*six-byteMACID*")

5 PLATAFORMA DE TESTES

Resumo

As duas plataformas apresentadas neste capítulo têm como finalidade facilitar o uso do autómato utilizado no trabalho experimental.

5.1. Plataforma A

5.2. Plataforma B

5.3. Plataforma A + Plataforma B

5.4. Testes de funcionamento

PLATAFORMAS DE TESTES

As plataformas a seguir apresentadas, têm por finalidade, a criação de uma ferramenta a ser usada nos laboratórios da Universidade, ou nas salas de aulas, de forma a dar a possibilidade aos alunos e professores, interagirem com esta fisicamente, facilitando assim o estudo e aprendizagem de automação. Ambas as plataformas, funcionam isoladamente ou em conjunto, sendo que uma delas (Plataforma A) já está equipada com um autómato, entrada para testes com controladores de alta velocidade, entre outros. Por sua vez, a Plataforma B, pese embora não esteja equipada com nenhum autómato, tem acessíveis entradas e saídas para fácil ligação através de conetores tipo banana, possibilitando uma utilização diferente, mas ao mesmo tempo complementando a Plataforma A.

5.1 Plataforma A

A plataforma A, representada na Figura 38, foi gentilmente oferecida ao autor e serviu de base a este projeto. Trata-se de uma bancada de ensaio, de fácil utilização, onde se podem efetuar simulações e testes de circuitos para projetos ou aprendizagem de automação.

Figura 38 - Plataforma A

O autómato utilizado é composto por 18 entradas digitais e 12 saídas, sendo que 16 das 18 entradas estão diretamente conectadas a outros tantos micro interruptores do tipo ON/OFF. Esta plataforma foi também previamente equipada com um *encoder* modelo E6B2-CWZ6C com três saídas (A B Z) para utilização e testes com contador de alta velocidade cujo diagrama está representado na Figura 39.
Para além dos já referidos, apresenta também duas entradas analógicas controladas por dois potenciómetros e dois voltímetros, um de entrada e outro de saída.

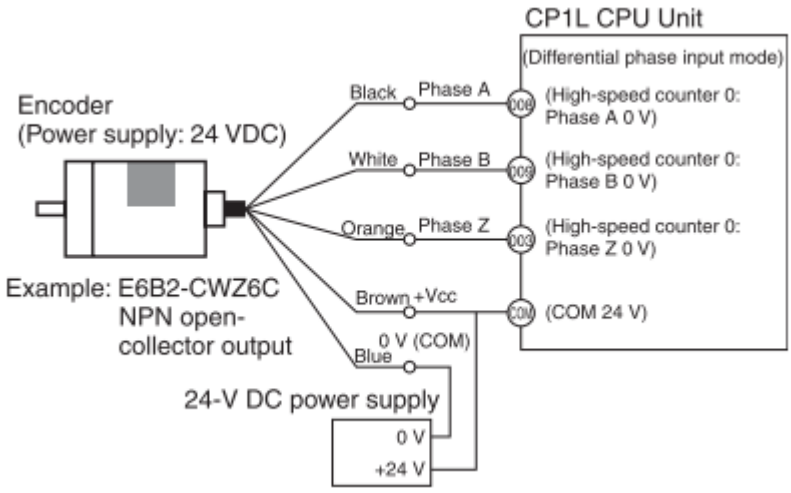


Figura 39 - Encoder [18].

Para o projeto em curso, houve necessidade de munir a Plataforma A de um adaptador de *Ethernet*, um adaptador RS-232C e de um *switch* que a seguir são abordados com maior detalhe.

Adaptador de Ethernet

Dada a necessidade de utilização do de uma rede de *Ethernet* e o facto do autómato utilizado não estar munido de uma entrada para este efeito, houve necessidade de recorrer ao uso de um adaptador deste tipo. O adaptador da Omron escolhido foi o modelo CP1W-EIP61 (Figura 40).

Figura 40 - Adaptador CP1W-EIP61.

Adaptador RS-232C

Pelas razões já atrás referidas para o adaptador de Ethernet, recorreu-se de igual modo a um segundo adaptador, este do tipo RS-232C, para uso em testes via porta série. O adaptador da marca Omron utilizado foi o modelo CP1W-CIF11 apresentado na Figura 41.

Figura 41 - Adaptador CP1W-CIF11.

Hub

Por imposição da necessidade de utilização de um *switch (Hub)* em virtude de o adaptador de *Ethernet* não suportar ligação direta uma vez que este funciona apenas como terminal, optou-se por utilizar o único disponível nos laboratórios da Universidade do Minho e que mesmo tratando-se de um equipamento de outro fabricante *(Westermo)* desempenha perfeitamente o papel que lhe está destinado. Este *switch* possui (Figura 42), 8 portas E/S, sendo que 6 delas são do tipo RJ45 e duas para ligação por cabo ótico (fibra ótica).

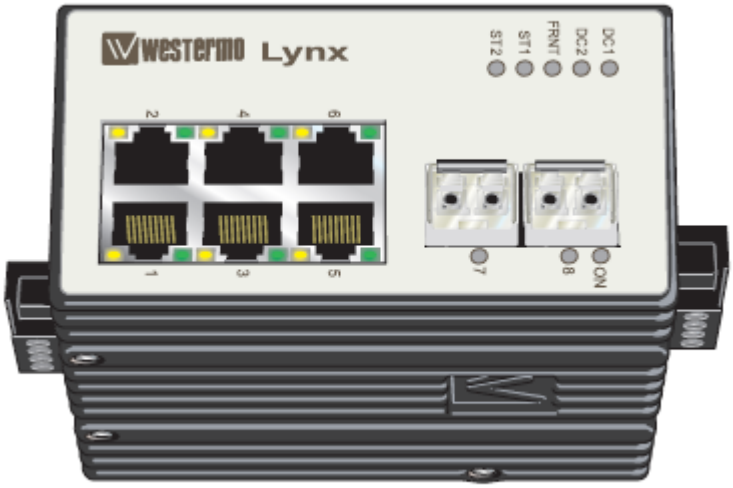


Figura 42 - Switch Industrial de Ethernet de 8 Portas.

5.2 Plataforma B

Por forma a facilitar e agilizar a utilização deste tipo de plataformas, em grupos por vezes muito extensos de utilizadores (alunos), foi pensada e construída uma nova plataforma denominada de Plataforma B, que irá conectar com a Plataforma A de modo a complementá-la.

A plataforma B, foi idealizada e construída de forma a evitar ligações diretas ao autómato, prevendo assim para além da facilidade de uso, possíveis danos no autómato provocadas por necessárias ligações conforme o projeto a utilizar.

Na figura 43, pode ver-se o início da transformação da Plataforma A para posterior conexão à Plataforma B. Esta adaptação prendeu-se essencialmente com a necessidade de ligação de todas as entradas e saídas do autómato à nova plataforma, de maneira a possibilitar a utilização desta sem restrições.

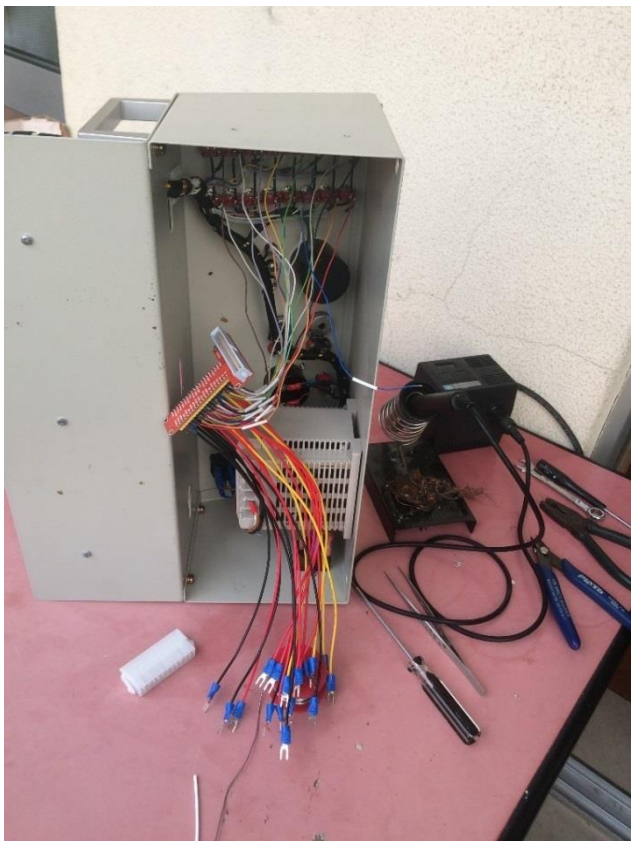


Figura 43 - Plataforma A (cablagem).

Na Figura 44, pode ver-se o estado final da alteração efetuada, bem como o cabo do tipo *flat* que irá conectar com a plataforma B.

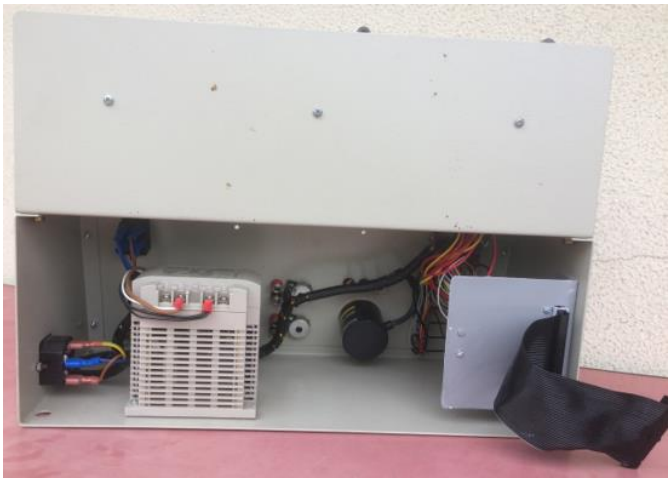


Figura 44 - Plataforma A.

Assim, como se pode ver na Figura 45, a Plataforma B, disponibiliza de forma simples através de conetores do tipo banana, acesso a todas as entradas e saídas do autómato da plataforma A, ou outro qualquer, utilizando as mesmas entradas e saídas, tornando os projetos a utilizar mais fáceis de ligação e agilizando no tempo o seu uso.

A plataforma B, para além da atrás referida disponibilização de acesso às entradas e saídas do autómato, foi dotada de 16 botões de pressão e dois interruptores tipo betoneira passiveis de serem também utilizados, alargando assim as possibilidades de uso oferecidas pela plataforma A, uma vez que esta apenas possui botões ON/OFF. Desta forma, podem ser usados comandos de programação não possíveis com esta configuração.
Para uma melhor visualização das saídas em uso, foram colocadas 12 lâmpadas do tipo LED sinalizando as referidas saídas.

Esta Plataforma, apresenta a possibilidade de conectar isoladamente um qualquer autómato, utilizando as entradas (18), disponibilizadas na parte inferior frontal ou em alternativa o conetor lateral. Este conetor pode ainda ligar-se à plataforma A para usar o autómato desta plataforma. As saídas na parte superior, podem se ligadas a atuadores, relés, entre outros, dependendo do projeto.

Para alem de tudo isto, disponibiliza também uma saída de tensão de 24 V que pode ser utilizada caso seja necessário.

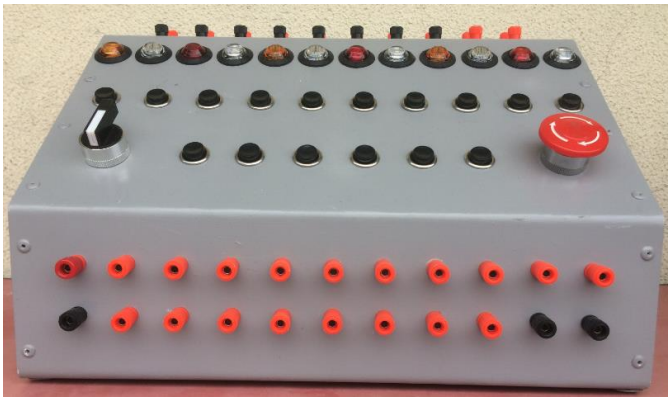


Figura 45 - Plataforma B.

Na Figura 46, pode ver-se parte do trabalho de montagem e cablagem da plataforma B.

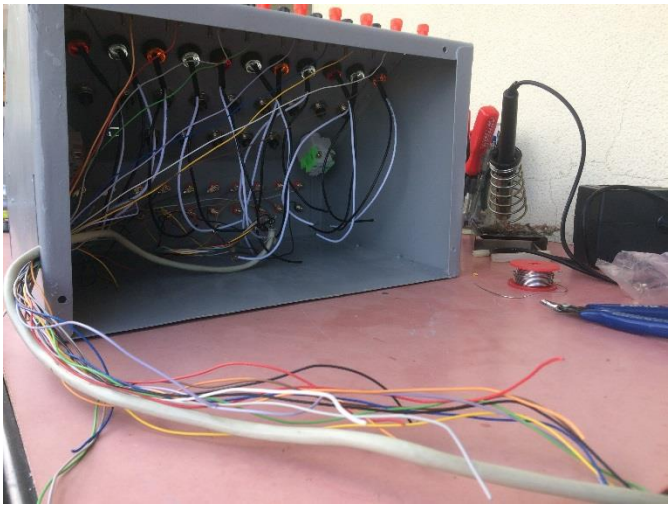


Figura 46 - Plataforma B (cablagem).

Na Figura 47, é visível o estado final da plataforma B, após montagem e cablagem, faltando apenas a colocação do tampo traseiro e ligação à plataforma A.

Figura 47 - Plataforma B (montagem final).

5.3 Plataforma A + Plataforma B

Na Figura 48, pode ver-se ambas as plataformas já conectadas entre si.

De notar que, as referidas plataformas podem ser utilizadas conjuntamente e/ou isoladamente, possibilitando assim uma maior diversidade na sua utilização.

Como particularidade especial, o facto de a plataforma B, na figura à esquerda, o facto de esta poder ser conectada isoladamente a qualquer autómato, utilizando para tal as entradas disponíveis e possibilitando ainda a ligação das saídas a diversos atuadores.

Outra possibilidade ainda disponível é a utilização do conetor de 40 pinos para ligação direta a um qualquer autómato:

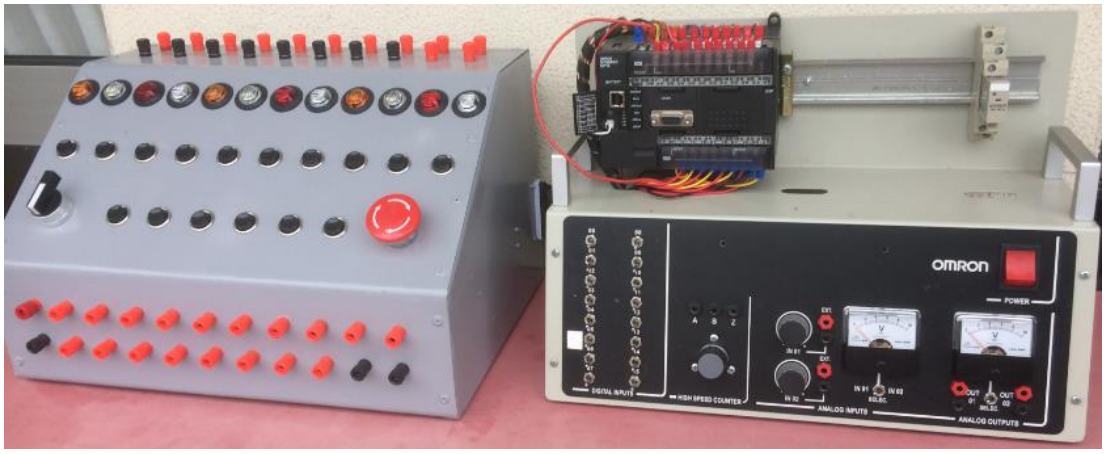


Figura 48 - Plataforma A + Plataforma B.

Na Figura 49, pode ver-se a ligação entre ambas as plataformas. O cabo utilizado é do tipo *flat cable* de 40 pinos.

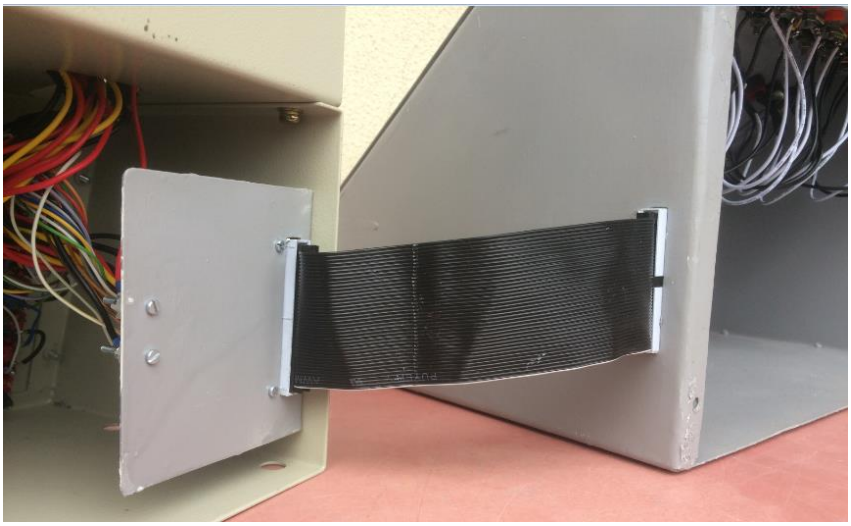


Figura 49 - Conexão entre as plataformas A e B.

5.4 Testes de funcionamento

Os primeiros testes de funcionamento das plataformas A e B, foram efetuados através da aplicação de nome *CX-Programmer* da *Omron*. Esta aplicação possibilita a programação e interação do autómato da mesma marca. Para a realização dos referidos testes iniciais foi construído um programa simples em linguagem *LADDER,* por forma a validar todas as conexões, quer entradas quer saídas. A imagem da Figura 50, apresenta o programa realizado.

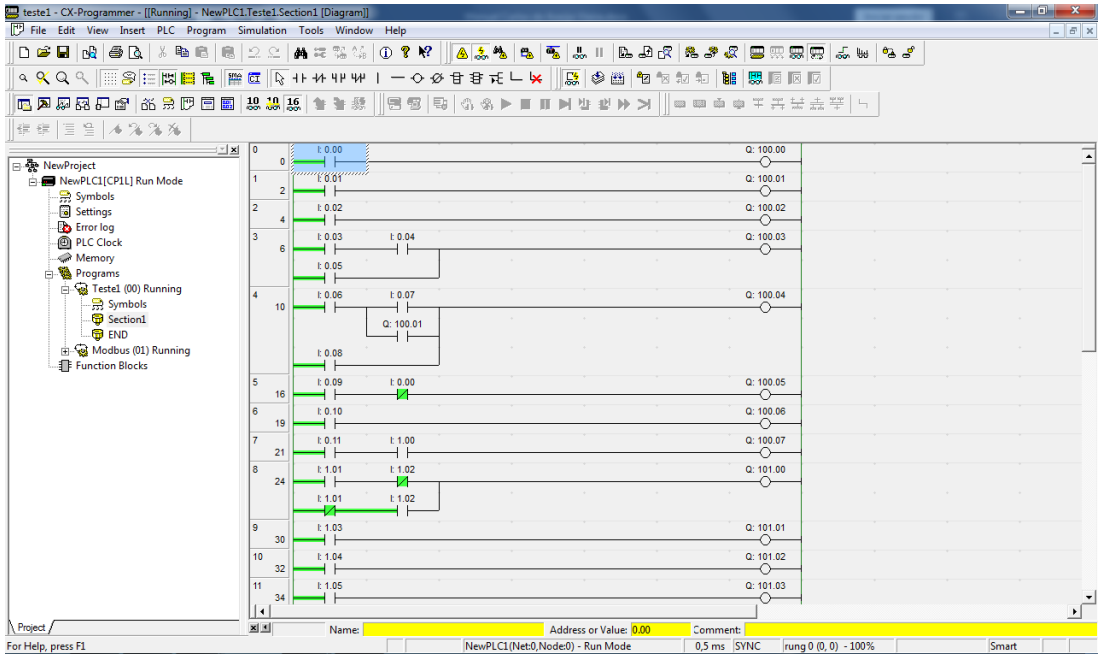


Figura 50 - CX-Programmer.

Nas figuras 51 - 54, são apresentados alguns desses testes, que atestam o funcionamento das saídas sinalizadas pelas lâmpadas de LED da plataforma B, através da manipulação das entradas efetuadas no CX–Programmer.

A Figura 51, mostra que que ao se efetuar a ação de atuação da entrada 00 *(force ON),* como seria de esperar, a saída 100.00, é ativada, como se pode facilmente comprovar pelo estado (acesa) da lâmpada que define esta saída.

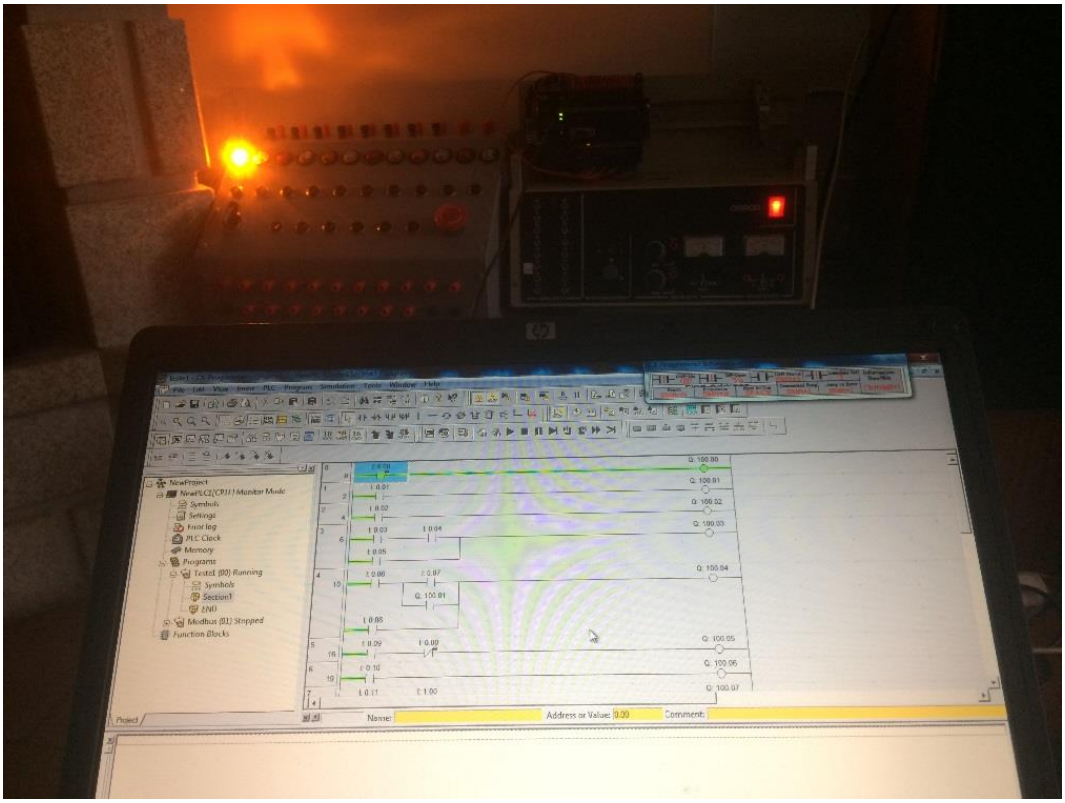


Figura 51 - Teste 1.

O mesmo sucede com a experiência seguinte, em que, pela programação efetuada, para que a saída 100.03 seja ativada, é necessário colocar as entradas 0.03 e 0.04 ou a entrada 0.05 a ON. O resultado foi também como esperado a ativação da saída 100.03 (saída 4), confirmada pela iluminação da lâmpada a verde (Figura 52).

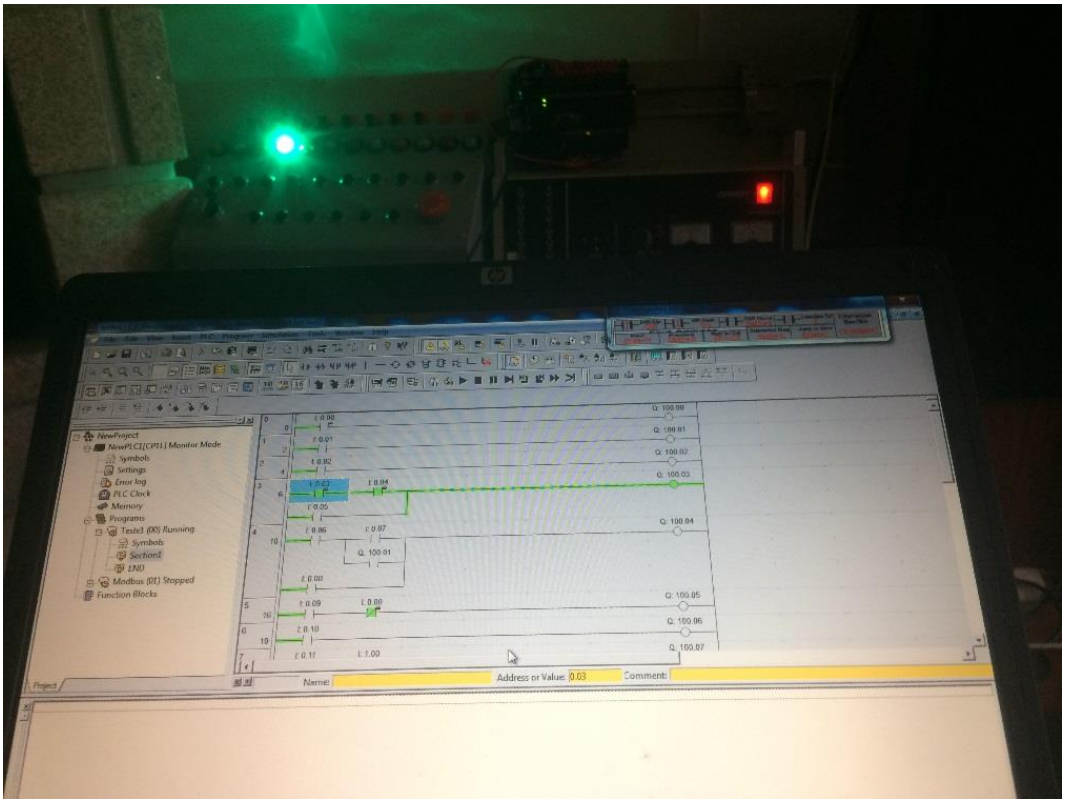


Figura 52 - Teste 2.

O teste seguinte representado na Figura 53, tinha como propósito iluminar a lâmpada 6 (saída 100.05) ao atuarmos sobre a entrada 0.09. De referir que no programa, como se pode ver na figura, foram colocados dois contactos, um NO *(normaly open)* e outro NC *(normaly closed).* Assim, logo que o primeiro é colocado a ON (contacto fechado), e porque o seguinte já se encontra fechado a saída é de imediato ativada. A saída poderá ser desativada se efetuarmos uma mudança de estado do primeiro ou do segundo, ou ambos, embora apenas um dos dois é necessário. Este teste foi também concluído com êxito.

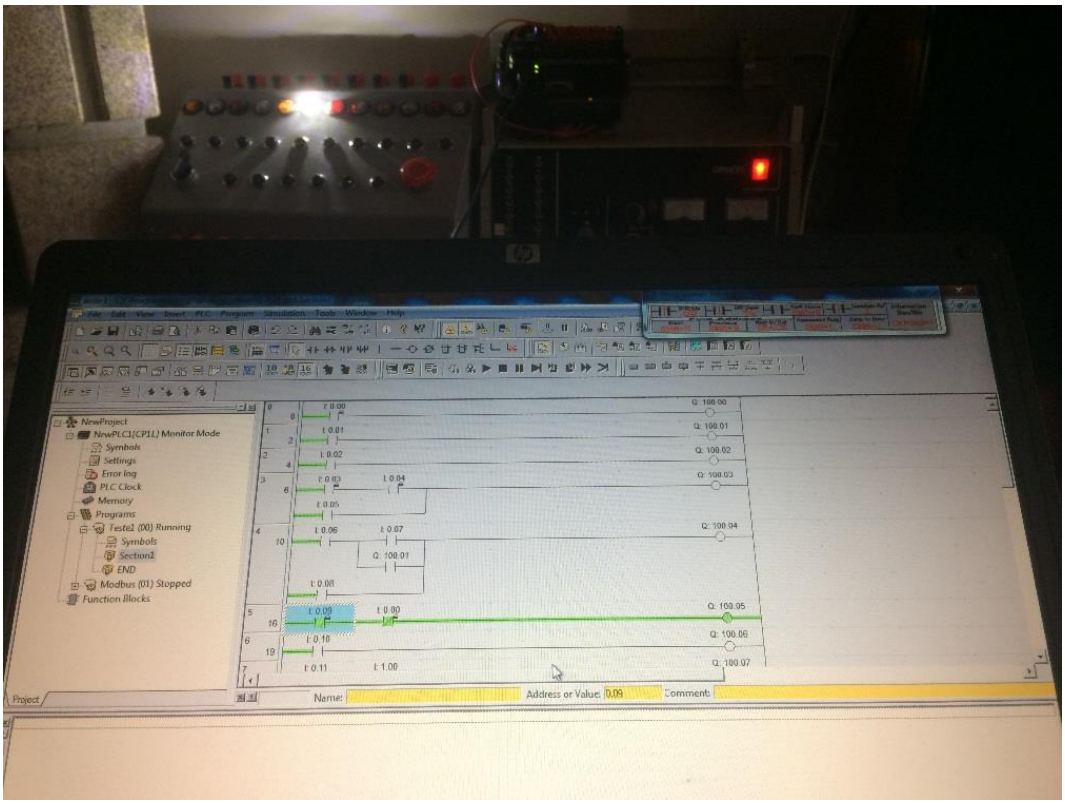


Figura 53 - Teste 3.

No teste aqui apresentado, a saída 10 (101.01), a lâmpada correspondente à respetiva saída deveria acender-se como resposta à ativação do contacto de entrada 1.03, como de resto se veio a verificar e se pode comprovar na Figura 54.

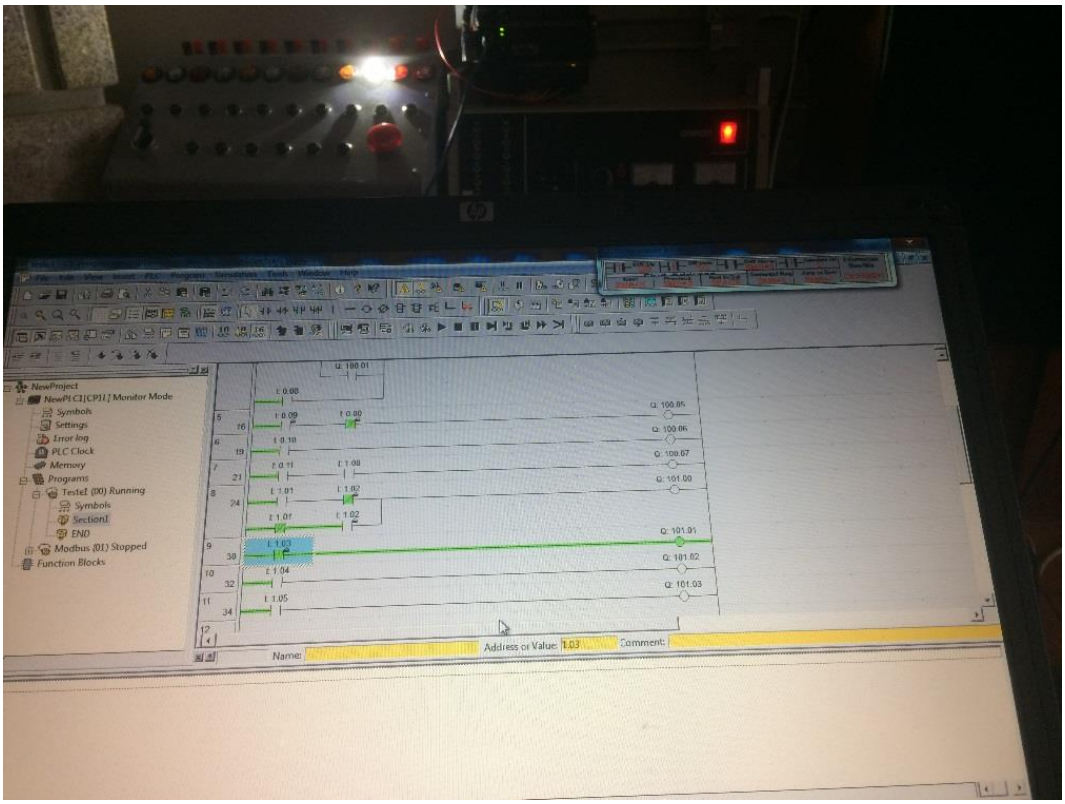


Figura 54 - Teste 4.

Os testes que se seguem foram efetuados para validar a comunicação via *Ethernet* entre o PC e o autómato, atuação sobre as entradas e observação da resposta na saída aplicada a um qualquer projeto.

Como se pode verificar na Figura 55, o PC foi conectado (8) através dum cabo de rede com ligação RJ45 a uma das portas do *switch* (6). Numa outra das portas do referido *switch* (5) foi ligado outro cabo de rede do mesmo tipo e que por sua vez foi ligado à entrada (9) do PLC.

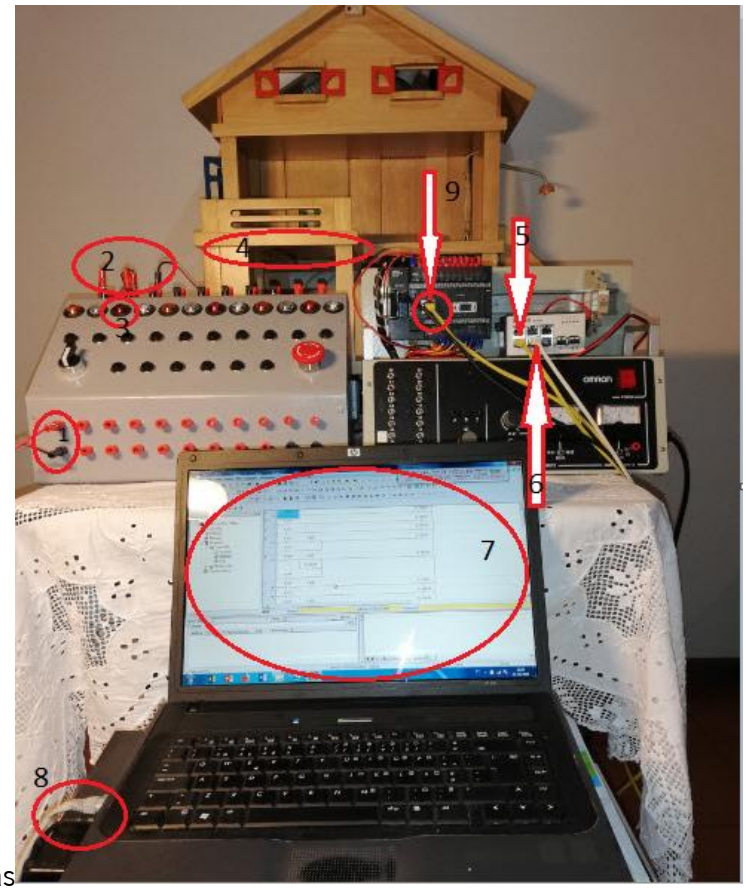
As plataformas A e B, estão ligadas entre si e a saída 3 (0.02) do *PLC*, foi ligada à casa miniatura, pertencente ao projeto atrás referido da autoria dum colega de curso. O objetivo desta utilização serve como forma de demonstração de funcionalidade a um qualquer projeto conectado às saídas das plataformas desenvolvidas.

A alimentação do *switch* foi retirada da plataforma B (1) e a saída 3 (2) liga diretamente a uma barra de leds (4), colocada no teto do *Wall* de entrada, que simula a iluminação da habitação. No PC, o programa (7) já utilizado anteriormente.

De referir, que embora apenas ter sido utilizada uma das saídas para o efeito, poder-se-ia utilizar todas as que estão disponíveis. O mesmo se poderia dizer sobre as diversas lâmpadas, sensores, detetor de intrusão, aquecimento, entre outros, instalados na casa. No entanto, os testes aqui efetuados têm por finalidade atestar a utilidade das plataformas aqui desenvolvidas e não a exploração e/ou criação de uma qualquer aplicação possível.

Seguidamente serão apresentados três testes:

- Teste 1 – Efetuado com o CX-Programmer
- Teste 2 – Utilizando a plataforma A
- Teste 3 – Utilizando a plataforma B



as

Figura 55 - Testes via Ethernet.

Teste 1

O teste 1, efetuado com recurso ao CX-Programmer, foi forçada *(Force ON)* a entrada 3 sendo visível a o led 3 (vermelho) aceso, bem como as luzes no interior da habitação (Figura 56).

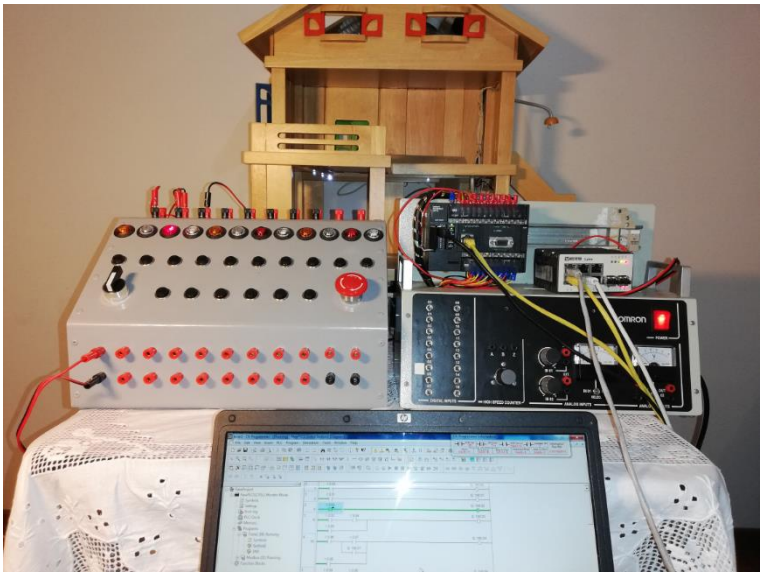


Figura 56 - Teste 1 (CX-Programmer).

Teste 2

Para o segundo teste, após ter sido desconetado o PC, foi acionado o interruptor 3, resultando de igual modo o acendimento das luzes da habitação bem como a lâmpada sinalizadora da plataforma B (Figura 57).

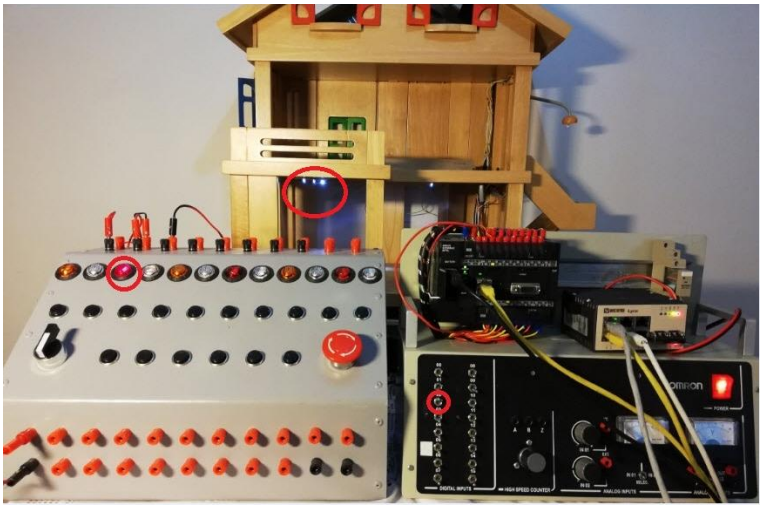


Figura 57 - Teste 2 (Plataforma A).

Teste 3

No terceiro e último teste, foi acionado diretamente o botão de pressão 3, resultando de igual modo o acendimento das luzes da habitação e led de sinalização da saída 3 (Figura 58).

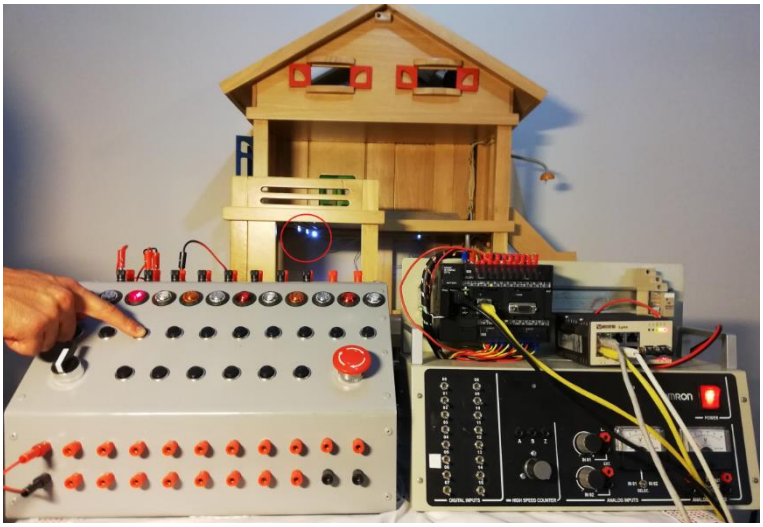


Figura 58 - Teste 3 (Plataforma B).

Mais testes foram, entretanto, realizados, como de resto seria espectável, e pese embora não estejam aqui todos representados, para não tornar esta parte demasiado extensa e entediante, todos resultaram com sucesso ficando assim, completada a fase de testes de ambas as plataformas.

De referir que estes testes foram efetuados não só a ambas as plataformas, ligadas entre si, como isoladamente

6 PLATAFORMA WEB

Resumo

Como parte integrante do projeto desenvolveu-se uma aplicação *web* de utilização fácil e intuitiva que se descreve detalhadamente neste capitulo.

6.1. Página *web* Inicial

6.2. *Página About*

6.3. Página de Registo

6.4. Página de *Login*

6.5. Página do Administrador

6.6. Página de Espera

6.7. Página de Acesso ao Laboratório

6.8. Web Server

PLATAFORMA WEB

A plataforma *web* que a seguir será apresentada, tem como função principal, a criação duma interface entre o utilizador e a plataforma de testes através da internet. Criada em HTML e alojada num *web server*, possibilita aos utilizadores o registo para uso da plataforma de testes, agendamento feito pelo Administrador da plataforma e posterior utilização da atrás referida plataforma (Plataforma de testes).

Todos os dados são armazenados numa base de dados, disponibilizada pelo provedor do serviço *web server* contratado.

6.1 Página web inicial

A pagina *web* inicial mostra de forma clara e simples o foco principal da aplicação "Laboratório Remoto de Automação".

No navegador desta página podemos escolher entre duas novas páginas, são elas, a página *About* e a página de *Login* (Figura 59).

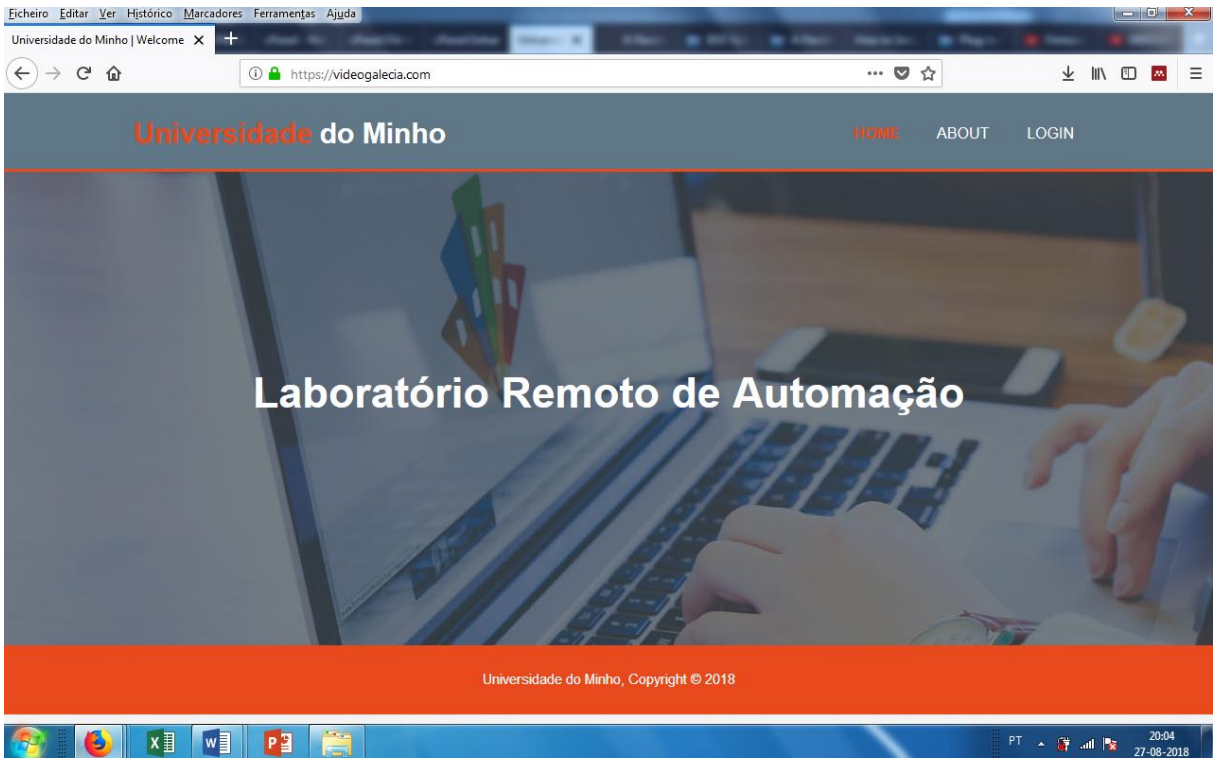


Figura 59 - Página web.

6.2 Página *About*

Caso o utilizador selecione a opção *About*, será direcionado para uma nova página *web* (Figura 60) que contêm toda a informação sobre o projeto. Aqui, poderá ser também encontrada, toda a informação necessária ao utilizador interessado para aceder à plataforma.

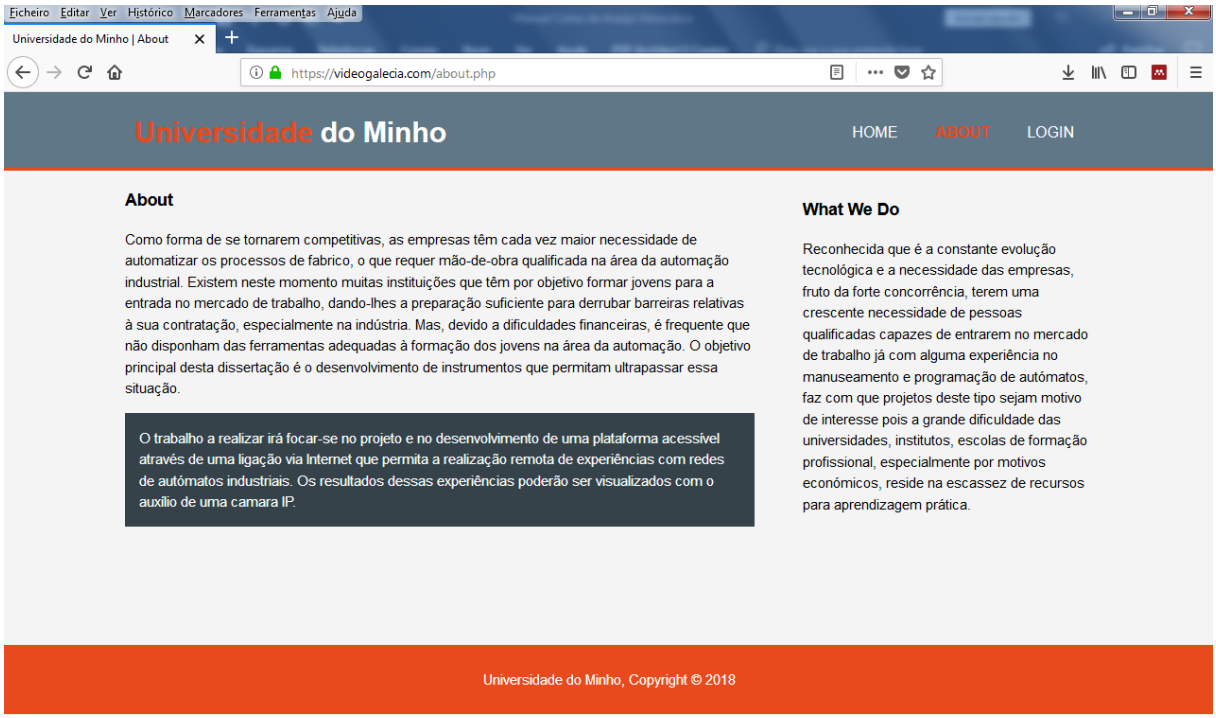


Figura 60 - Página de informação sobre o projeto.

6.3 Página de registo

Na figura 61 pode ver-se a página de registo onde o utilizador interessado, poderá, se assim entender, efetuar o registo na plataforma, no entanto a sua utilização estará sempre condicionada, à prévia validação do Administrador da referida plataforma.

Os dados introduzidos ficarão registados numa base de dados e no ato do registo, a aplicação recorrerá a esta para evitar registos duplicados, tais como nome de utilizador e email, alertando e impossibilitando o utilizador de efetuar o registo caso isso aconteça. Também não são autorizados espaços por preencher, alertando o utilizador de que é obrigatório o preenchimento de todos os itens.

Por forma a garantir maior segurança, a *password* é encriptada na base de dados com o algoritmo MD5, garantindo assim que algum acesso indevido à base de dados resulte no roubo ou usurpação dos dados de registo do utilizador.

Finalmente sempre que um hipotético utilizador interessado efetua o registo com sucesso, recebe um email, agradecendo o facto de demonstrar interesse pelo uso desta aplicação.

Por sua vez, o Administrador, receberá também um email, dando-lhe conhecimento do registo.

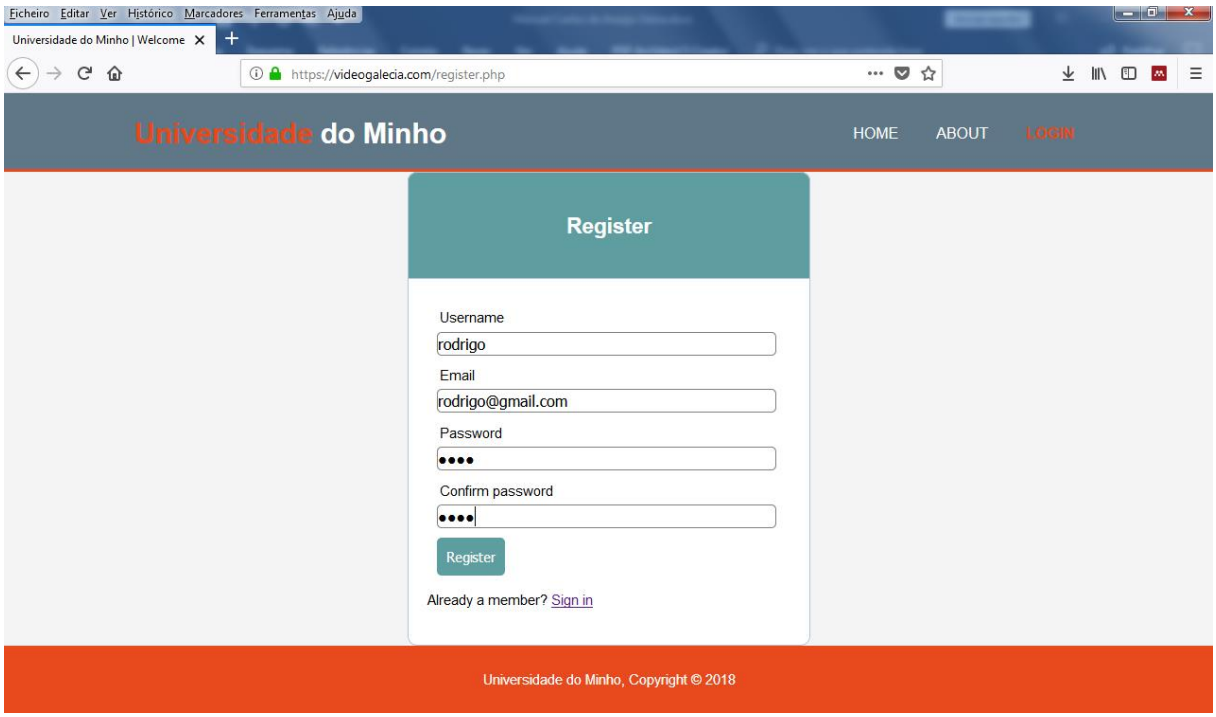


Figura 61 - Página de Registo.

6.4 Página de Login

Durante esta fase, e sem a prévia validação do Administrador da plataforma, caso o utilizador, efetue o *Login* com os dados do registo, receberá de imediato, através de mensagem na plataforma de que o seu registo carece de validação pelo Administrador, conforme se pode ver na Figura 62. Esta mensagem é direcionada ao utilizador, identificando-o pelo nome de registo.

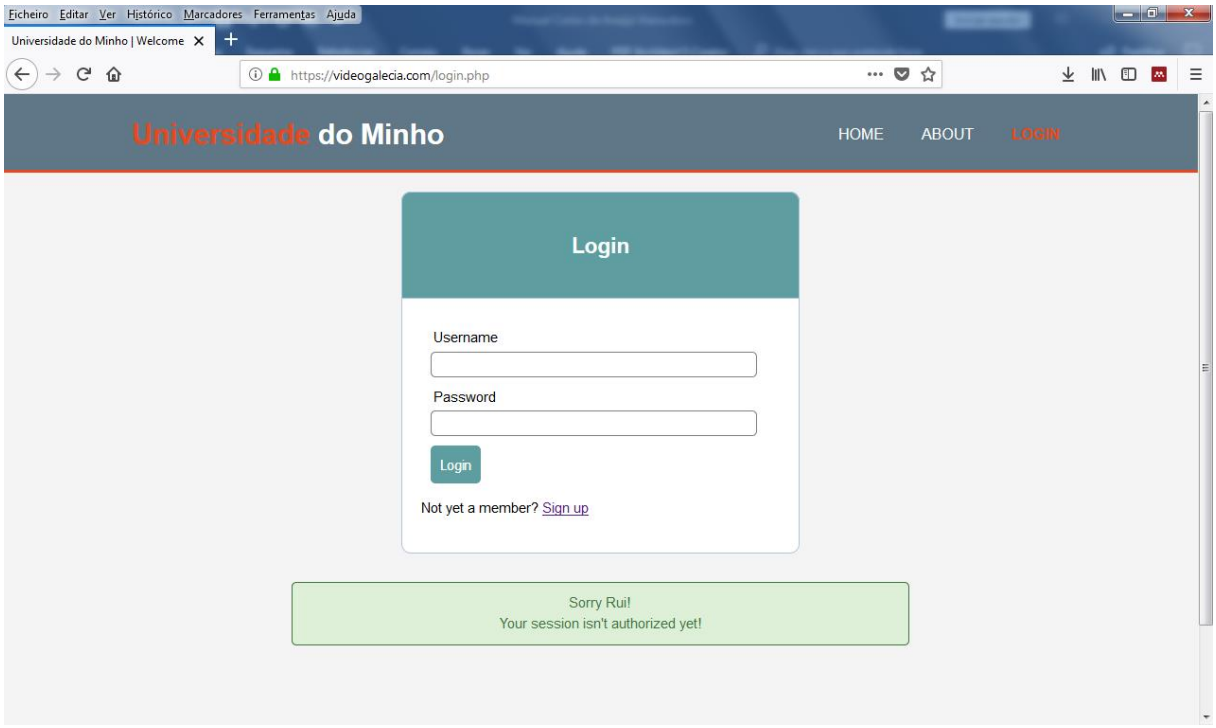


Figura 62 - Página de Login.

Também e para prever acessos não autorizados, a aplicação verificará em tempo real sob consulta à base de dados, se os dados introduzidos são ou não válidos.

Assim, caso os dados introduzidos não estejam registados na base de dados, o utilizador receberá a informação de que os dados introduzidos não são válidos ou estão incompletos (Figura 63).

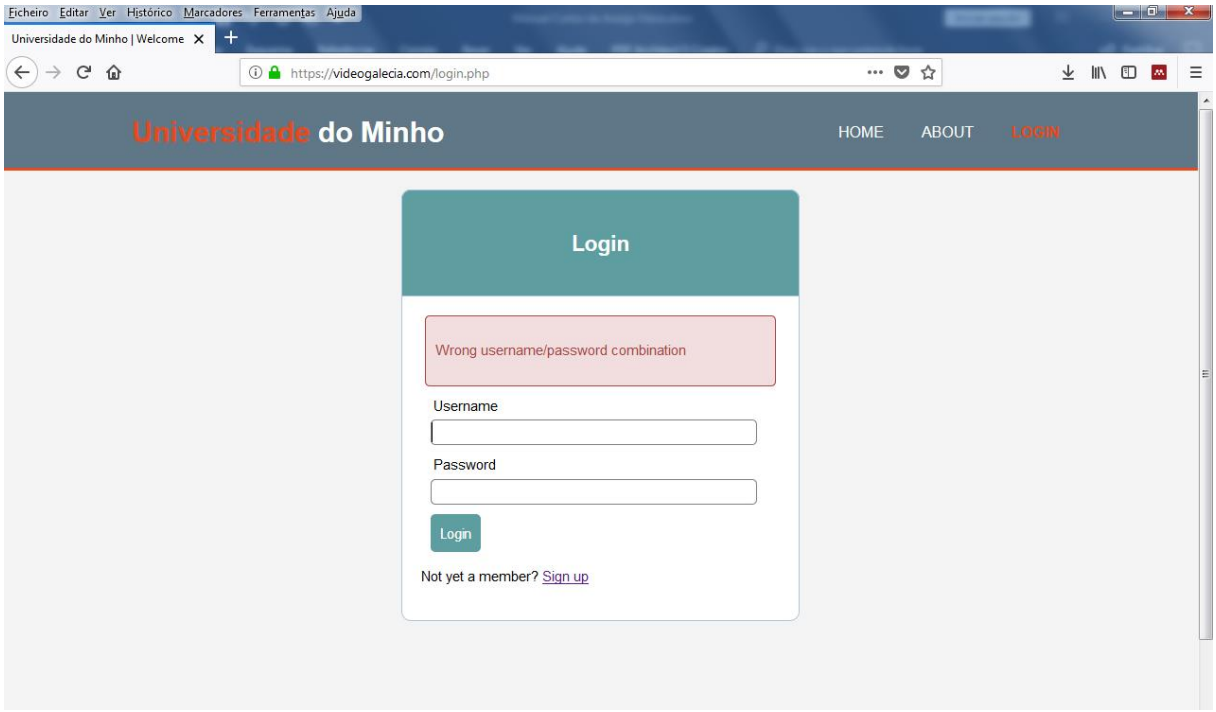


Figura 63 - Dados incorretos.

6.5 Página do Administrador

O Administrador da aplicação, pode, sempre que assim o entender, definir quando e durante o espaço de tempo que achar conveniente, determinado utilizador poderá utilizar o Laboratório Remoto de Automação. Para isso, terá de aceder à mesma plataforma, efetuar o *Login* com as credenciais de Administrador *(username e password)*, e definir no painel para o referido utilizador o dia e hora de acesso e o dia e hora de termino. Após a validação, o utilizador receberá de forma automática, um *email*, com a indicação da autorização de acesso e respetivo espaço temporal para uso desta. O Painel do Administrador pode ser visualizado na Figura 64.

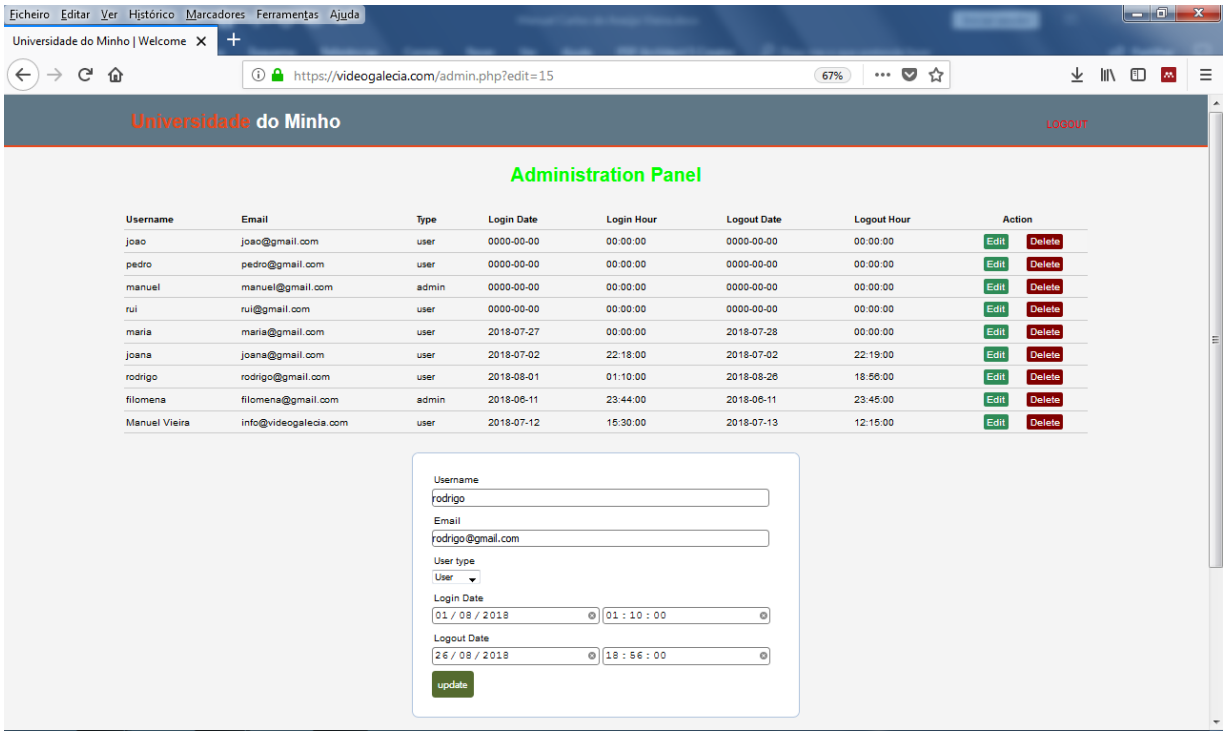


Figura 64 - Painel do Administrador.

No painel do Administrador, como se pode ver na Figura 64, toda a informação necessária é carregada automaticamente para esta página através da base de dados.

Os dados apresentados são:

- *username*
- *email*
- tipo de utilizador
- dia de início de sessão autorizada
- hora de início de sessão autorizada
- dia de termino de sessão autorizada
- hora de termino de sessão autorizada

Estão disponíveis também, dois botões, um a verde que permite ao Administrador editar os dados do(s) utilizador(es) e outro a vermelho que apaga o utilizador selecionado da base de dados, excluindo-o de forma definitiva se assim o entender (Figura 65).

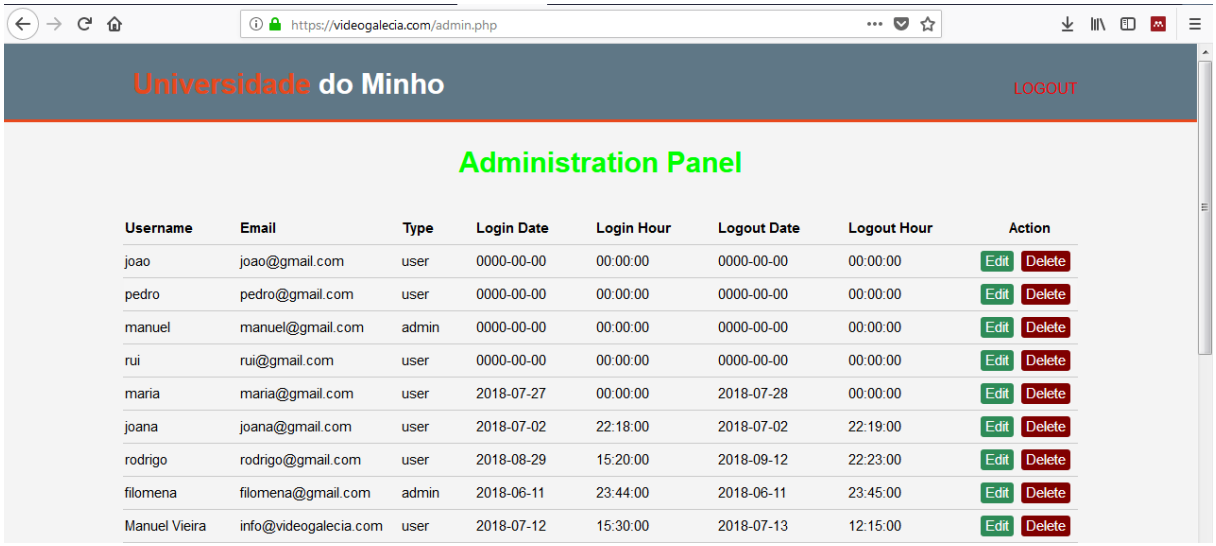


Figura 65 - Painel do Administrador.

Na parte inferior da mesma página (Painel de Administrador), representado na Figura 66, sempre que o Administrador, prime a tecla de edição de um determinado utilizador, o cursor posiciona-se sobre a área de edição pré preenchendo com os dados do referido utilizador selecionado (*username*, *email* e tipo de utilizador). No entanto, o Administrador poderá se assim achar conveniente ou a pedido do utilizador, alterar estes dados. Poderá também, caso pretenda, atribuir ao utilizador, permissões de Administrador, alterando o tipo de *user* para *Admin.*

De seguida poderá ainda definir a data e hora de *Login* e *Logout*. Estes dados, para poderem ser validados têm de corresponder a determinadas regras que são:

- A data de *Login* não pode ser inferior à corrente data e hora
- A data de *Logout* não pode ser inferior à corrente data e hora e inferior à data e hora de *Login*

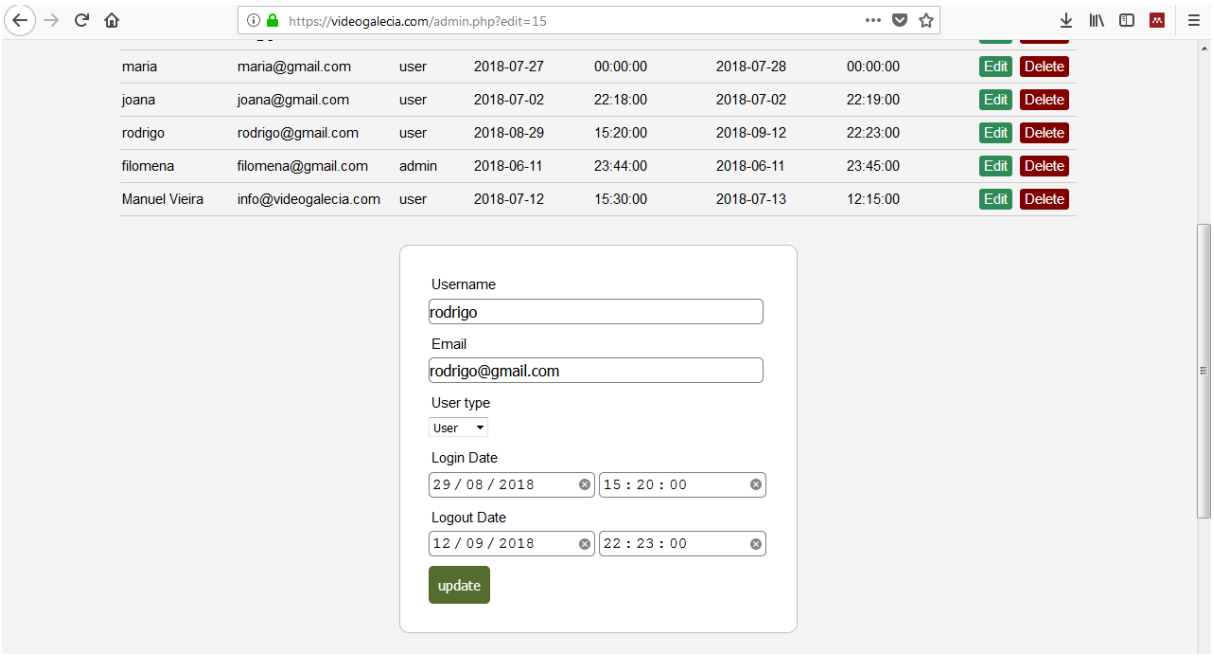


Figura 66 - Edição de dados.

Após a edição, ao premir a tecla de *update*, os dados serão enviados para a base de dados onde serão verificados e caso não existam inconformidades, serão então registados. No topo da página aparecerá então uma mensagem informando do sucesso da operação, enquanto que, e em simultâneo, os dados referentes ao utilizador serão também atualizados no Painel do Administrador (Figura 67).

Figura 67 - Update de dados.

Caso haja alguma inconformidade, que neste exemplo foi a tentativa de introdução de uma data de *Login* inferior à corrente data, surgirá uma mensagem advertindo para o erro, não permitindo a gravação dos dados introduzidos, como se pode verificar na Figura 68.

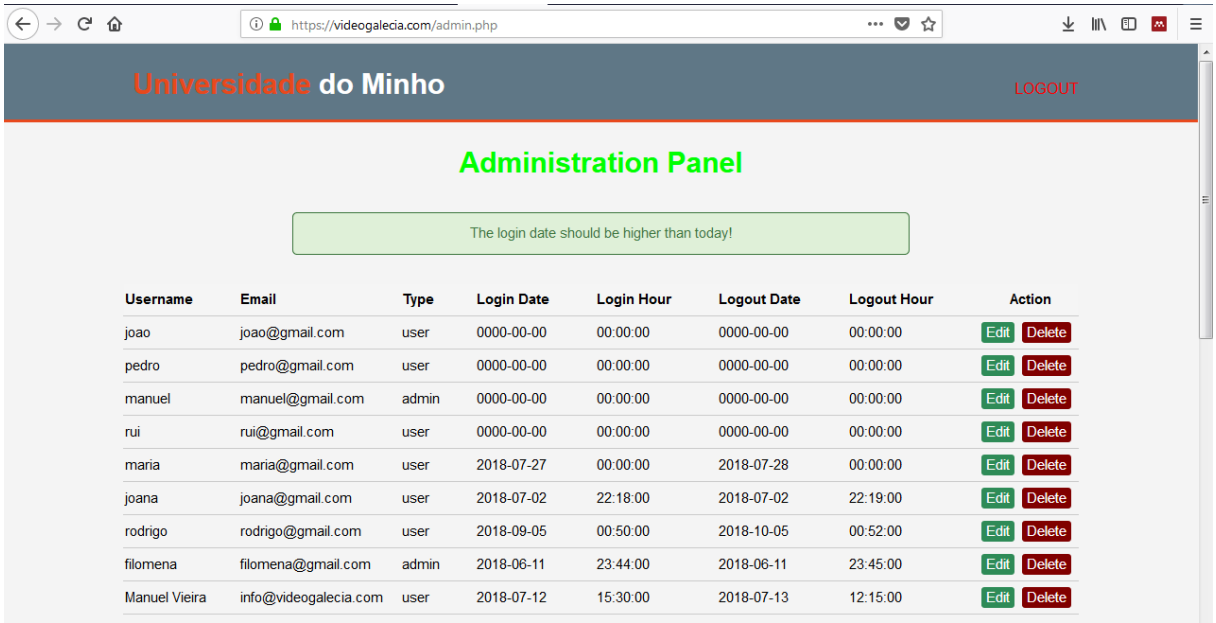


Figura 68 - Update com dados inválidos.

6.6 Página de espera

Para aceder à plataforma, e após ter recebido o email com a confirmação da validação do seu registo, onde consta também a data e hora autorizada para o seu acesso, o utilizador deverá aceder à página de *Login* e introduzir os seus dados de registo.

Logo após que a autorização é validada pelo Administrador para determinado utilizador, este ficará sempre condicionado ao espaço temporal definido por este. No entanto, caso efetue o *Login* numa data inferior à definida, será direcionado para uma página de espera onde poderá visualizar através de um relógio digital com a informação horaria atualizada do tempo de espera (Figura 69).

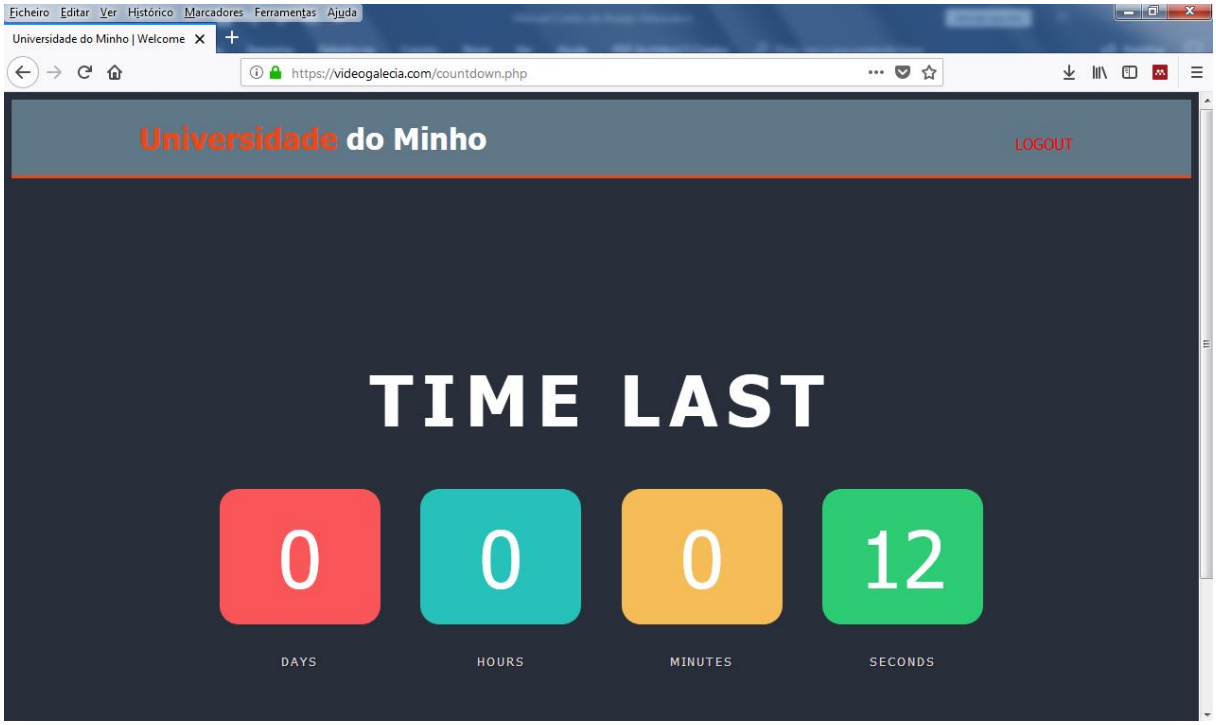


Figura 69 - Página de Espera.

6.7 Página da plataforma

Na data e hora definida pelo Administrador, o utilizador ao efetuar o *Login*, será direcionado para a página que contém a plataforma de testes e onde estão disponíveis todo os componentes de testes que irão manipular o autómato(s) remotamente.

Caso o utilizador, tenha acedido numa data e hora anterior como anteriormente foi referido, será direcionado para a página de espera, no entanto, na hora e data autorizada, será imediatamente direcionado para a página de testes.

Para uma melhor informação do tempo restante, é possível ainda visualizar um relógio digital que dá ao utilizador a indicação em tempo real sobre o tempo restante para utilização. Quando este atingir o zero, o utilizador deixará de poder utilizar a plataforma sendo de imediato redirecionada para a página inicial, possibilitando assim, a utilização da plataforma por outros utilizadores, sempre com prévia autorização do Administrador (Figura 70).

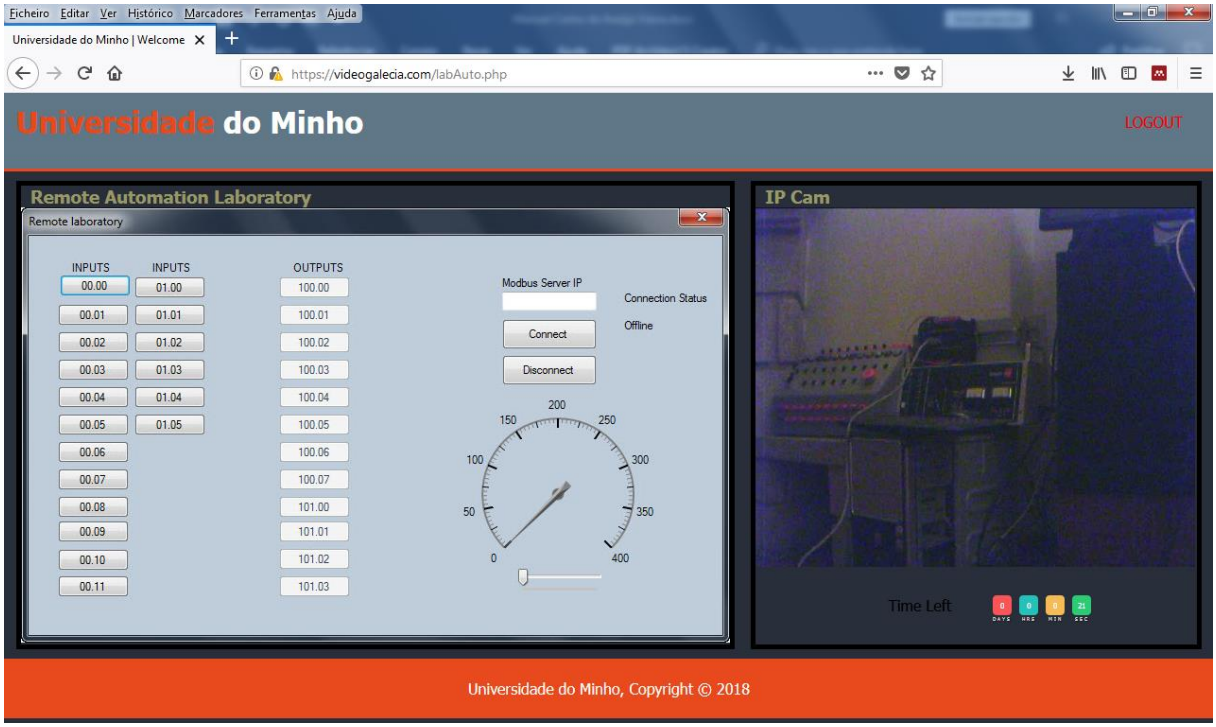


Figura 70 - Página do Laboratório.

Após ultrapassado o tempo definido pelo Administrador para uso da plataforma, e isto acontecerá quando o relógio apresentado na página de testes abaixo da imagem de vídeo da camara IP, o utilizador será redirecionado para a página principal.

Caso o utilizador tente aceder posteriormente à plataforma, receberá uma mensagem advertindo-o de que o tempo de utilização foi esgotado e de que deverá se assim o pretender, contatar o Administrador para eventual novo acesso (Figura 71). Esta mensagem está personalizada dirigindo-se ao utilizador pelo nome de registo. O utilizador poderá, no entanto, efetuar novo registo sendo que para o efeito terá de utilizar novas credenciais. O uso das mesmas credenciais só poderão ser validadas pelo Administrador.

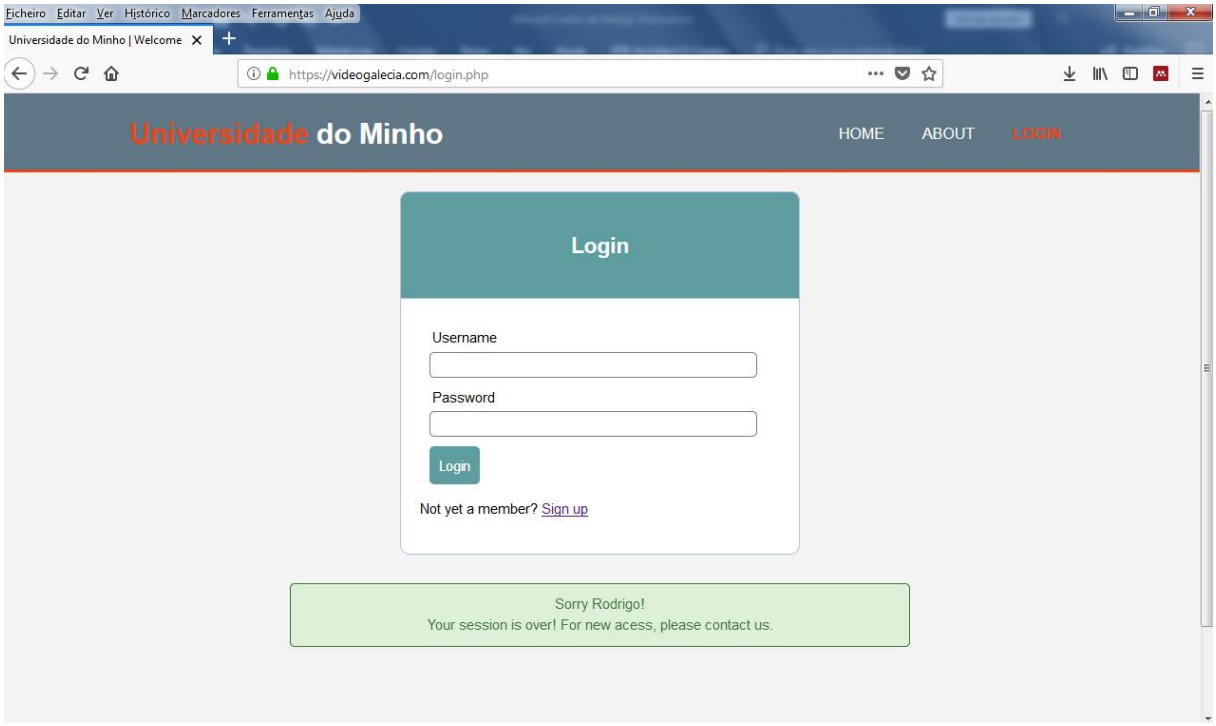


Figura 71 - Timeout.

Nesta página, existirá também, uma janela de visualização controlada por uma camara IP, facilitando assim, a visualização das ações definidas pelo programador bem como, os resultados da manipulação dos comandos disponíveis.

A camara IP utilizada da marca *Pixord*, modelo E205P tem como principais características (Figura 72):

- Resolução de imagem de 640x480 pixéis
- Compressão de imagens no formato *JPEG e Motion JPEG*
- Deteção de movimento
- Programação de scripts de eventos
- Protocolos *TCP/IP, UDP, ARP, ICMP, HTTP, FTP, SMTP, DHCP*

Figura 72 - Camara IP.

6.8 *Web Server*

Para o desenvolvimento desta aplicação, foi utilizado um domínio pessoal registado num *Web Server* sob o nome *http://videogalecia.com* , e que será futuramente por questões logísticas transferido para um a ser definido pelo Administrador do sistema.
Foi também utilizada uma base de dados *MySql*, alojada no referido servidor facilitada pela utilização de ferramentas disponibilizadas pelo serviço de alojamento contratado e com a ajuda de aplicações, tais como, *PHP MyAdmin*.

A aplicação *Web*, foi desenvolvida com a ajuda do editor "*Atom*" que através da linguagem de programação *HTML* e utilização de *Java Script, PHP, CSS*, entre outros, possibilitou a realização da referida aplicação. O código desenvolvido foi colocado por upload no *web server.*

A camara IP foi posteriormente alojada num *web server* gratuito *(NO IP)*, que fornece um IP estático evitando desta forma a perda de ligação ocasionada por eventual alteração do IP definido pelo fornecedor de *internet* uma vez que se trata de um serviço dinâmico e não estático como seria desejável. O link fornecido por este fornecedor de serviço foi http://labauto.zapto.org:81/images1full, que inclui o nome escolhido pelo autor ("labauto" de laboratório de automação) e extensão (zapto.org) atribuído pelo administrador do serviço. 81 é o número da porta que foi previamente autorizada no PC servidor da plataforma para acesso à camara IP.

6.9 C-Panel

O *C-Panel,* é um *software* de gestão de hospedagem *web*.

Este *software* disponibilizado pelo fornecedor de serviço de alojamento contratado, está repleto de aplicações que, possibilitam entre outros, a gestão de uma base de dados *MySQL*, ferramenta PHP, *phpMyAdmin*, gestor de ficheiros, contas *email*, *site publisher*, etc. Torna-se assim uma ferramenta muito útil para a, criação e alojamento de páginas *web*. A página principal pode ser vista na Figura 73.

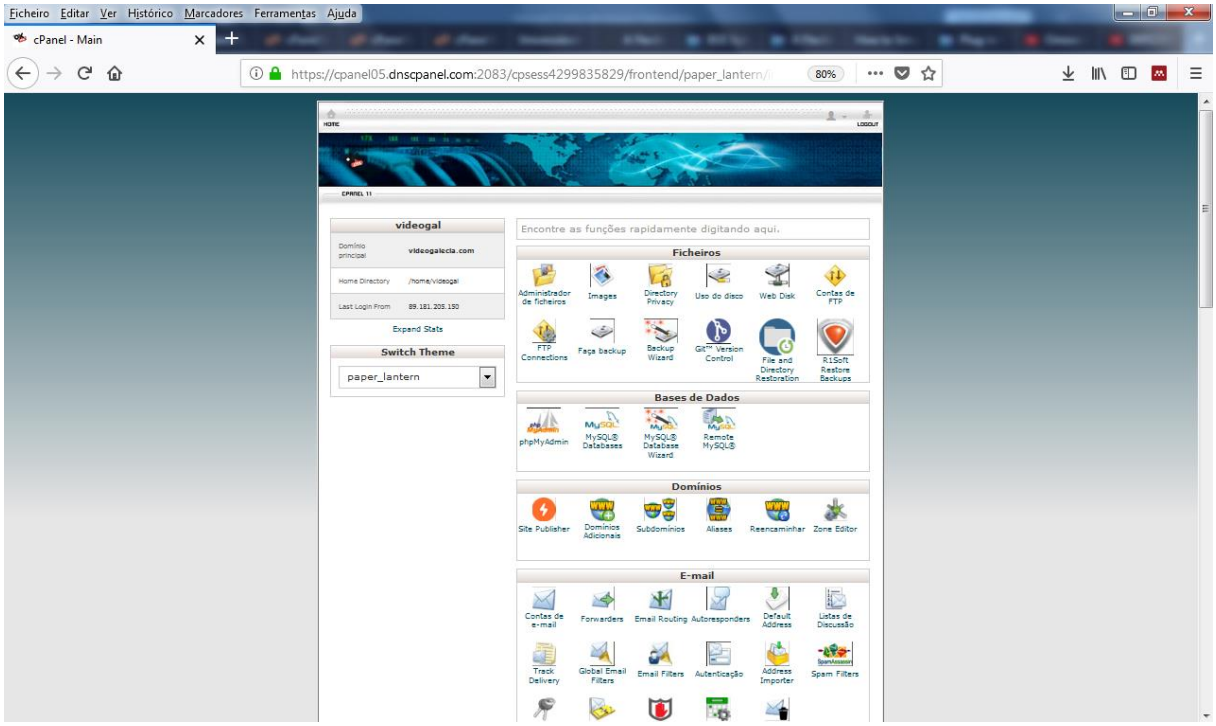


Figura 73 - C-Panel.

Na Figura 74, pode ver-se a utilização da ferramenta *phpMyAdmin*, onde é visível também os dados já adicionados na base de dados.

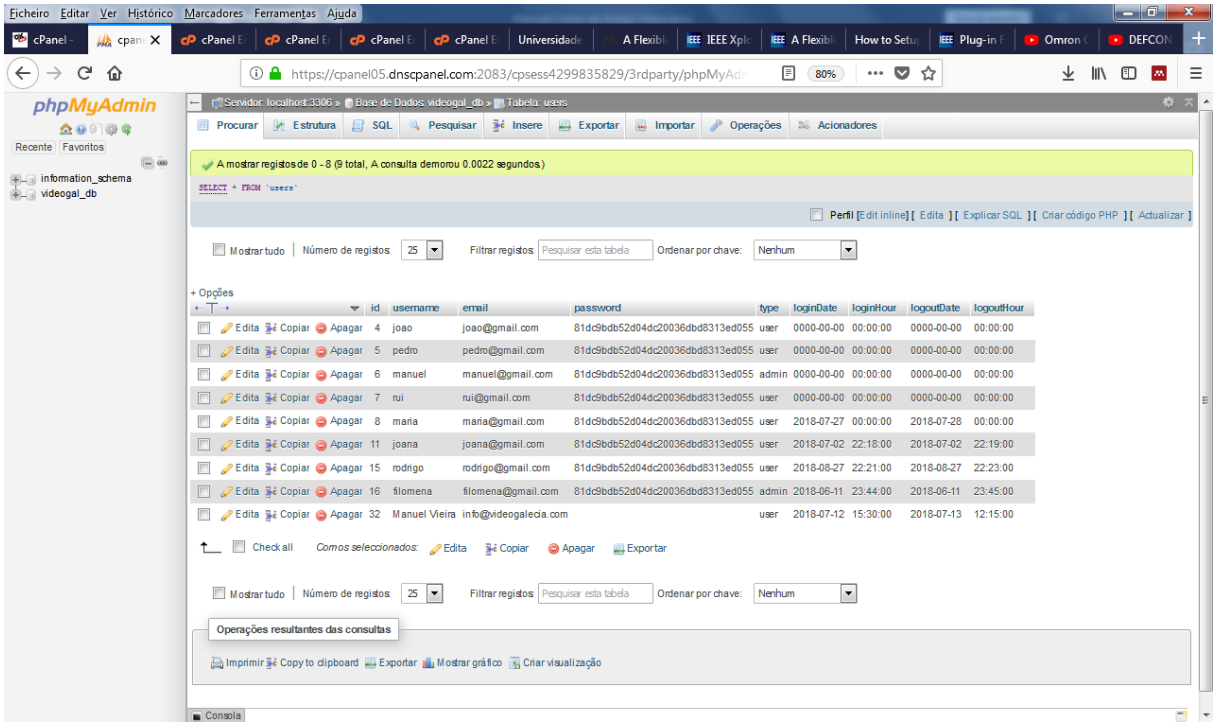


Figura 74 - Base de dados phpMyAdmin.

Na Figura 75, é apresentado o painel administrador de ficheiros para onde são carregados todos os ficheiros que contem o código desenvolvido (ficheiros *HTML*, ficheiros *PHP*, ficheiros *CSS*, ficheiros *Java Script*, entre outros).

O processo de transferência é bastante facilitado bastando para tal selecionar o menu de *upload* e escolher o ficheiro que se quer enviar. Assim, podemos estar a programar com a ajuda de um editor de texto ou outro (por ex. *Atom*) e de imediato fazer o upload do mesmo e testar o código desenvolvido para verificar se tudo funciona conforme planeado.

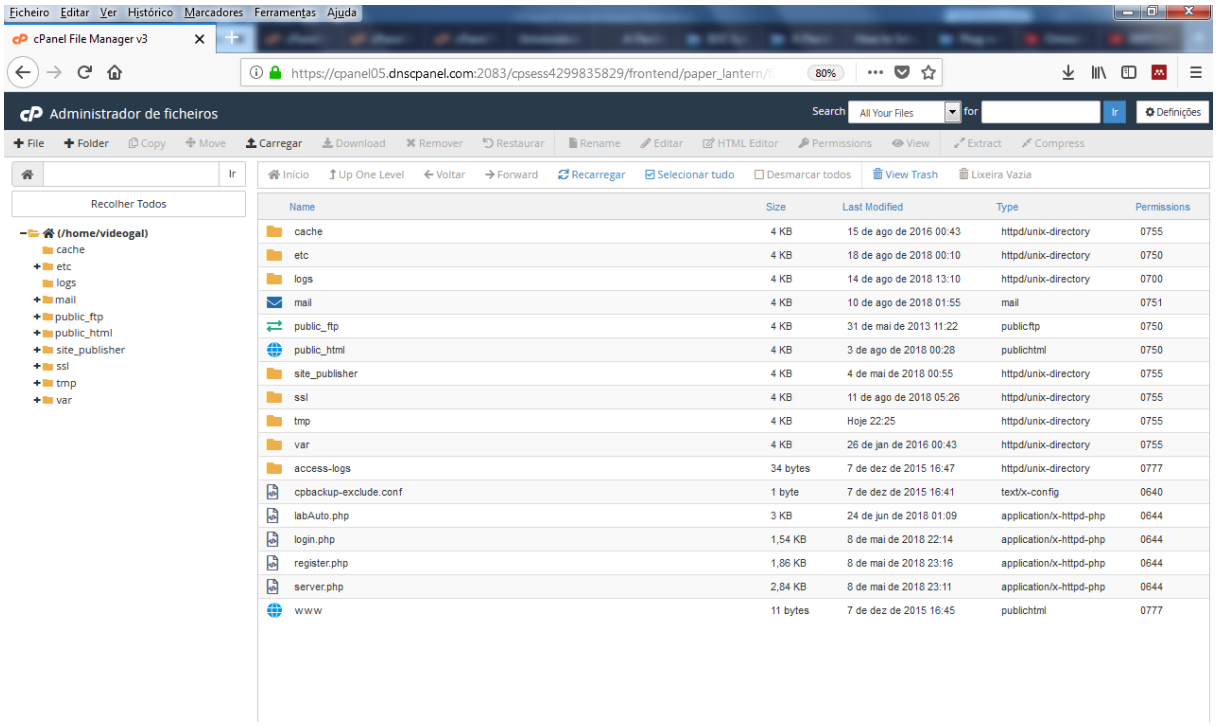


Figura 75 - Painel Administrador de Ficheiros.

6.10 Aplicação de Comando

A aplicação de comando de forma mais detalhada pode ser vista na Figura 76. Esta aplicação integrada na plataforma *web*, tem por função promover a interface de comando via *web* entre o utilizador e a(s) plataforma(s) disponíveis no laboratório.

Como forma de comunicação, foi selecionada a comunicação *Modbus* TCP/IP. A sua escolha, de entre várias disponíveis prendeu-se essencialmente pelo facto de ser *open source*, fiável e a que mais se adaptava à realização do projeto em curso.

A referida aplicação foi pensada para que o utilizador possa manipular as entradas (00.00 a 01.15) obtendo a resposta nas saídas (100.00 a 101.03). O controlador *Gauge*, na parte inferior direita da aplicação, permite manipular o contador de alta velocidade da plataforma A, ou registar o movimento do *encoder* (contador de alta velocidade) se este for movimentado no laboratório.

Na janela do canto superior direito, regista o IP do servidor e é colocado pelo utilizador quando está em modo de simulação.

Outras janelas com outras configurações serão realizadas de acordo com o propósito e interesse de cada utilização.

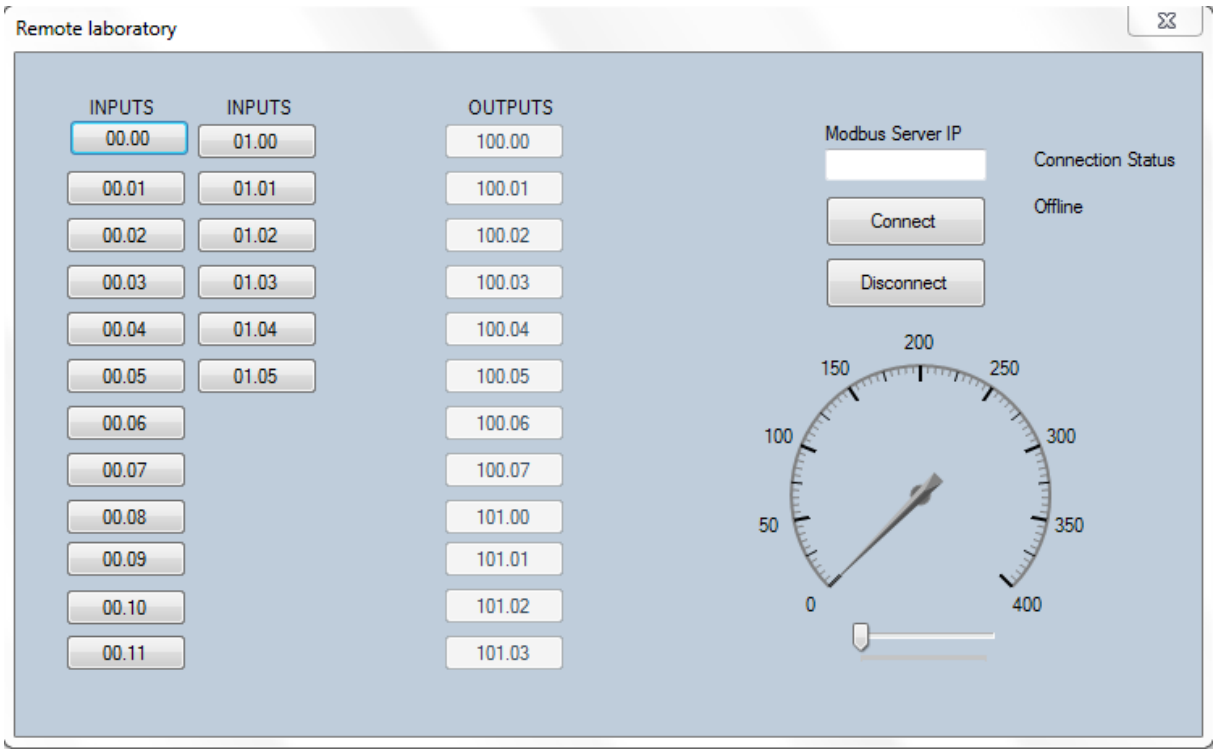


Figura 76 - Aplicação de comando.

6.10.1 Aplicação em modo de simulação

Para os primeiros testes de funcionamento da aplicação de comando, recorreu-se a um simulador *Modbus*. A sua função tem por finalidade simular um autómato real, que posteriormente será comandado via *web* por esta aplicação de comando.

Dado o simulador ter ficado instalado no PC, foi necessário descobrir o IP do computador que estava a ser utilizado. Para tal recorreu-se ao comando *IPCONFIG* na janela do *script* CMD do *Windows*.
O endereço *IPv4* da máquina, como podemos verificar na Figura 77, foi 192.168.1.11.

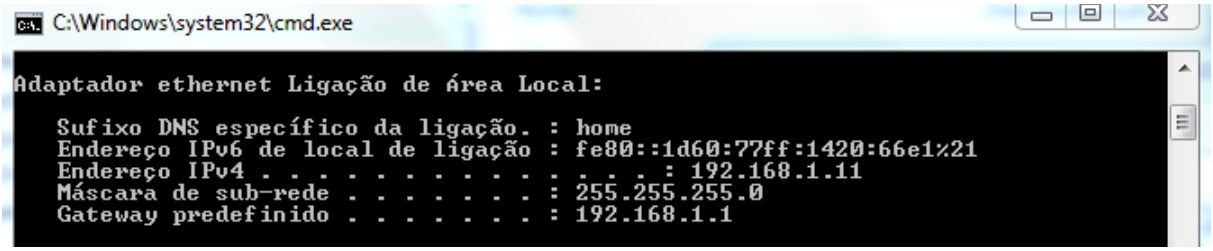


Figura 77 - CMD.

A Figura 78, mostra a aplicação de simulação *Modbus*. Foi selecionado o modo *Modbus* TCP/IP, bobinas de saída *(Coil Outputs),* porta 502 (porta utilizada em *Modbus*) e forçadas a ON, as saídas 0, 5, 6 e 10 (cinza escuro).

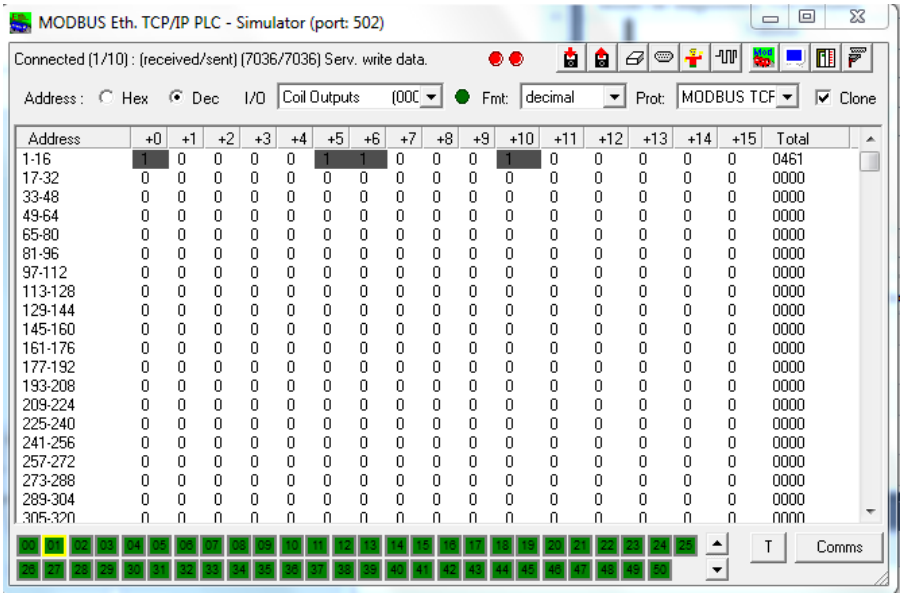


Figura 78 - Simulador Modbus.

Para ligação ao Mestre, neste caso o simulador, no aplicativo do programa, foi colocado o IP e acionado o botão de ligação *(connect).* Como resultado podemos verificar na Figura 79, a azul as entradas 0, 5, 6 e 10, bem como as saídas 100.00, 100.05, 100,06 e 101.02, que correspondem às entradas e saídas ativas.

Figura 79 - Leitura de dados.

De seguida foi efetuado também um teste ao *Holding Register*, colocando manualmente o valor de 350 (Figura 80).

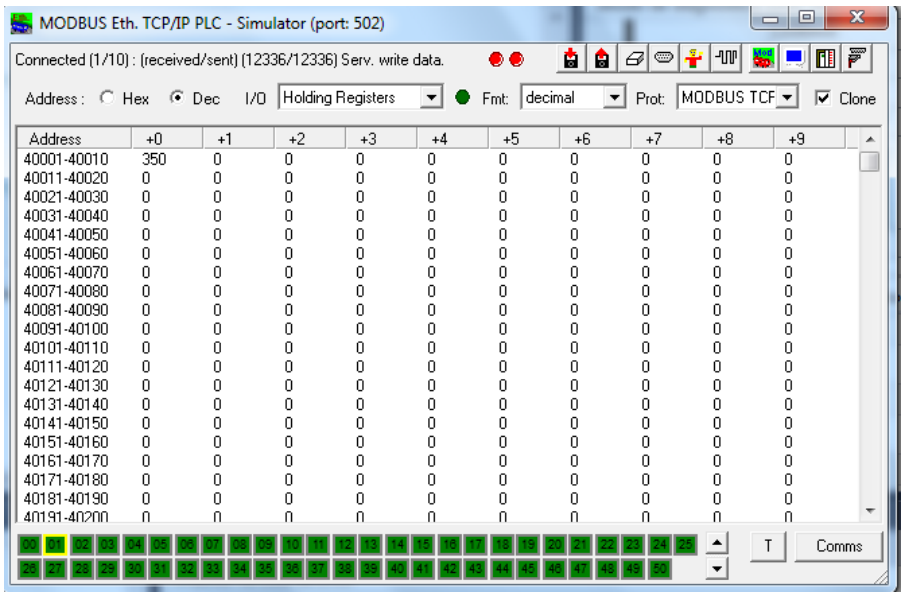


Figura 80 - Simulação com Holding Registers.

Como se pode ver na Figura 81, o controlador de *Gauge*, passou a indicar o mesmo valor.

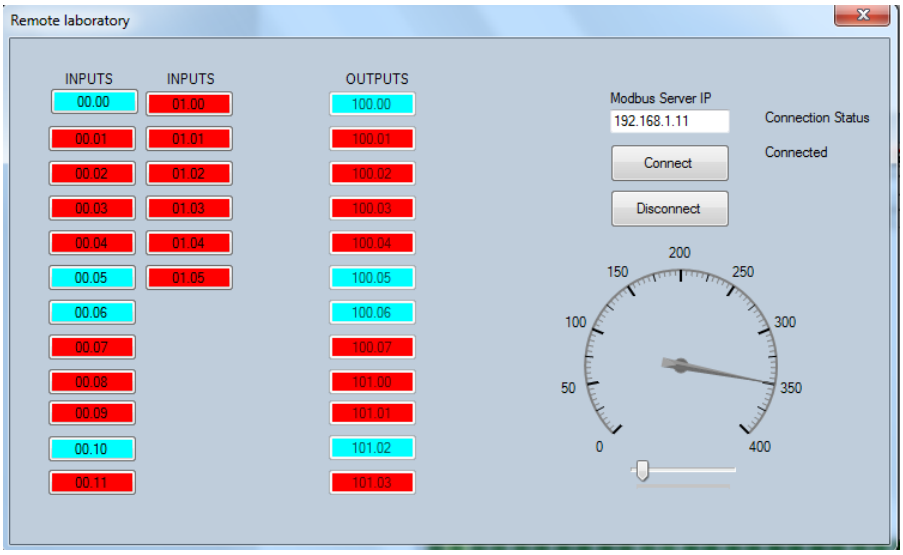


Figura 81 - Leitura de dados (Gauge).

Concluídos os testes de simulação, pode ainda acrescentar-se que a configuração da aplicação de comando, pode ser efetuada em várias outras formas, ou incluir mais janelas com inclusão de gráficos, leitores digitais ou analógicos, por exemplo para visualização de medições de temperatura ou outras. Consoante o projeto ou projetos, facilmente se poderá efetuar as modificações necessárias.

Embora não tenha sido aqui demonstrado, poderia ter-se atuado do lado da aplicação (Cliente) colocando a ON ou a OFF as saídas do Mestre (neste caso o simulador), dado tratar-se de mais uma entre outras, das particularidades do protocolo *Modbus*.

7 CONCLUSÕES E TRABALHO FUTURO

Resumo

Este capítulo apresenta as principais conclusões do projeto desenvolvido no âmbito desta dissertação, bem como algumas sugestões de trabalho futuro.

CONCLUSÕES E TRABALHO FUTURO

O trabalho realizado no âmbito da Dissertação, teve como principal objetivo a criação de meios e o desenvolvimento de ferramentas na área da Automação Industrial. Assim, foi planeada e desenvolvida duma forma estruturada um conjunto de ações e projetos capazes de no final, terem como resultado um Laboratório Remoto para o Ensino de Automação capaz de incentivar o ensino/aprendizagem nesta área.

Tendo em conta, que, na maior parte das aulas dos cursos lecionados, o tempo útil e o grande número de alunos dificulta ou em muitos casos, impossibilita o contacto físico com estas ferramentas e tendo em conta a valência adquirida por todos aqueles que têm a "sorte" de as poderem usar no seu percurso académico, podemos inferir a grande importância de projetos como este.
A construção de plataformas, como as que aqui foram construídas, facilitam o uso pelos alunos em aulas de Automação e agiliza em tempo a sua utilização, permitindo desta forma uma maior dinâmica de utilização.

A primeira plataforma a ser construída, atrás identificada como Plataforma B, teve como finalidade replicar as entradas e saídas dum qualquer autómato, disponibilizando para o efeito 18 entradas digitais, 12 saídas com indicador luminoso LED para identificação do *status* da saída (ON/OFF) e ainda 16 botões de pressão e 2 betoneiras para utilização em testes de simulação.
Esta plataforma, pode ser usada individualmente em aulas de laboratório utilizando para o efeito um autómato conectado nas entradas e saídas com acesso facilitado por ligações através de conetor do tipo banana.
Foi também efetuada uma alteração numa outra plataforma (Plataforma A) existente para possibilitar a utilização em conjunto de ambas alargando assim as possibilidades de uso e a utilização do autómato já instalado. Esta alteração consistiu na criação duma interface, entre ambas com ligação por flate cable de 40 pinos de fácil conexão.

Por forma a complementar a utilização destes recursos, foi também construída uma plataforma web. Esta plataforma, tem por objetivo dar a possibilidade a que muitas outras pessoas possam, na Universidade, em casa ou em qualquer outro lugar usufruir da vantagem de utilizar as atrás referidas plataformas, como se estivessem a usá-las diretamente. Ora isto, por si só, resulta numa abrangência

de recursos disponibilizados a um grande número de utilizadores interessados em aprender ou melhorar o seu conhecimento.

Por forma a dotar a aplicação web duma utilização controlada e dada a necessidade de manter o sistema, eficaz, seguro e disponível para todos, esta plataforma foi munida de ferramentas de segurança, com obrigatoriedade de registo para a sua utilização, uma base de dados para arquivar todos os registos e utilizadores e um painel de Administrador que tem por função possibilitar a este o agendamento do dia e hora da utilização da referida plataforma.

Possui ainda um painel construído em linguagem C#, onde podem ser visualizados os resultados das ações definidas pelo utilizador, botões e áreas onde podem ser colocados valores a serem enviados remotamente para os autómatos, ligados às plataformas físicas.

Como forma de comunicação entre a aplicação *web* e os autómatos foi desenvolvida uma interface que funciona sob o protocolo *Modbus* TCP/IP. Esta interface tem por finalidade estabelecer a comunicação entre os autómatos ligados às plataformas (A e B) por forma a possibilitar a troca de dados via web entre a plataforma(s) física(s) situada(s) nos laboratórios da Universidade e a aplicação na *web page*.

Para trabalho futuro, fica a vontade de evoluir o projeto com construção de painéis físicos pré construídos, onde possam ser ligadas as plataformas físicas aqui construídas e simulados circuitos formados por redes de autómatos industriais ou outros.

REFERÊNCIAS

[1] M. Kalúz, P. Orduña, J. García-Zubia, M. Fikar, and Ľ. Cirkaš, "Sharing control laboratories by remote laboratory management system weblab-deusto," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2013, vol. 10, no. PART 1, pp. 345–350.

[2] "Lab - ud-demo-pld." [Online]. Available: https://weblab.deusto.es/weblab/labs/PLD experiments/ud-demo-pld/#. [Accessed: 14-Dec-2017].

[3] A. Borracha, "Laboratório Remoto de Automação Industrial (Lab-RAI)," 2012.

[4] "Laboratório Remoto de Automação Industrial (Lab-RAI) - PDF." [Online]. Available: http://docplayer.com.br/1390563-Laboratorio-remoto-de-automacao-industrial-lab-rai.html. [Accessed: 14-Dec-2017].

[5] "Laboratório Remoto." [Online]. Available: http://web.ist.utl.pt/ist11390/aind.htm. [Accessed: 14-Dec-2017].

[6] L. M. S. Martins, "Projecto dum laboratório remoto para automação de processos industriais," 2013.

[7] E. De Engenharia, "Nuno Daniel Carneiro de Carvalho WALC-AI : Laboratório para Aprendizagem do Controlo e Automação Industrial," 2010.

[8] M. Kalúz, J. García-Zubía, M. Fikar, and Ľ. Čirka, "A Flexible and Configurable Architecture for Automatic Control Remote Laboratories," *IEEE Trans. Learn. Technol.*, vol. 8, no. 3, pp. 299–310, 2015.

[9] A. Bradley, "CONTROLADORES LÓGICOS PROGRAMÁVEIS - CLP CONTROLADORES LÓGICOS PROGRAMÁVEIS - CLP Em 1968 , cientes das dificuldades encontradas na época para se implementar controlos lógicos industriais . David Emmett e William Stone da General Motors Corporation solici."

[10] F. D. Petruzella, *w.EngineeringBooksPdf.c*, Fo u r t h. New York: McGraw-Hill.

[11] S. Cp and S. Cp, *CP1L-EL/EM CPU Unit*. .

[12] USP, "A Norma IEC 61131," *Edisciplinas Usp*, vol. 2004, 2004.

[13] I. List, "Syllabus :," 2013.

[14] "What is SCADA? Supervisory Control and Data Acquisition." [Online]. Available: https://inductiveautomation.com/what-is-scada. [Accessed: 29-Aug-2018].

[15] "SCADA - Supervisão e Controlo SCADA : Componentes," pp. 1–8, 2007.

[16] Amplicon, “Process Control and Automation using Modbus Protocol,” vol. 44, no. 0, 2012.

[17] M. I. U. S. A, “Introduction To Modbus Tcp / Ip,” *Instrumentation*, vol. 44, no. 248, 2005.

[18] S. Cp and S. Cp, “Operation Manual.”