APARATO PARA EXPERIMENTAÇÃO REMOTA DO LANÇAMENTO DE PROJÉTEIS

Carlos Manuel de Oliveira Loureiro Paiva

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Automação e Controlo
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2012

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Carlos Manuel de Oliveira Loureiro Paiva, Nº 1950460, 1950460@isep.ipp.pt
Orientação científica: Doutor Gustavo Ribeiro da Costa Alves, gca@isep.ipp.pt
Coorientação científica: Doutora Maria Arcelina Marques, mmr@isep.ipp.pt

Mestrado em Engenharia Eletrotécnica e de Computadores
Área de Especialização de Automação e Controlo
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2012

Em memória dos meus pais e irmã

Não sigas pelo caminho traçado, pois ele só conduz até onde os outros foram

(Alexandre Graham Bell, cientista e inventor 1847 – 1922)

Agradecimentos

O trabalho realizado no âmbito desta tese só foi possível graças à colaboração de várias pessoas e instituições que me apoiaram.

Começo por agradecer ao ISEP, pela bolsa concedida no âmbito do projeto "*Physics LabFARM*"; à equipa de docentes e técnicos do Departamento de Física pela sua constante disponibilidade e ajuda; à Tecnogial, nas pessoas do Sr. Rui Almeida, Sr. Albino e Sr. António Azevedo, pelo apoio dado na resolução de alguns problemas de índole mecânica.

Em relação às pessoas, começo por agradecer aos meus orientadores, Doutor Gustavo Ribeiro Alves e Doutora Maria Arcelina Marques pelo apoio permanente durante o projeto e redação da tese, que me ajudaram a levar o projeto a bom porto; ao Doutor Carlos Felgueiras, pela sua amizade e conselhos dados; ao Engenheiro Paulo Ferreira, pela sua amizade e apoio constante, e empréstimo dos *kits* de desenvolvimento;

Quero ainda agradecer à equipa (Gustavo Alves, Arcelina Marques, Clara Viegas e Cristina Costa Lobo) presente na exposição internacional REV2012, pelo apoio dado na publicitação do trabalho efetuado.

Quero também agradecer aos meus colegas de mestrado, em especial ao Pedro Peixoto e ao Fernando Pinto pela amizade, companheirismo e apoio permanente ao longo destes dois anos.

Finalmente, gostaria de agradecer à minha família e amigos, em especial à minha esposa Conceição, e ao Rafael pelos muitos dias perdidos longe deles, no desenvolvimento deste trabalho.

Resumo

Num sistema de ensino cada vez mais exigente, a experimentação assume um papel fundamental na aquisição e validação do conhecimento. No ensino da Física, a necessidade de compreender a influência do meio num dado conceito teórico leva a que a experimentação tenha um carácter obrigatório.

Neste contexto, surgem três cenários capazes de suportar a aprendizagem dos conceitos teóricos adquiridos. A simulação que faz uso da velocidade e capacidades de cálculo do computador para obter o resultado de uma experiência, a experimentação tradicional em laboratório, na qual o aluno executa, presencialmente, a sua experiência e por último a experimentação remota, que permite a execução de uma experiência real sem a presença física do aluno.

Esta dissertação apresenta o projeto de um aparato para experimentação remota do "*Lançamento de projéteis*". De forma a providenciar um meio de ensino de Física mais flexível, o aparato desenvolvido permite, aos alunos, a determinação da aceleração da gravidade e o estudo da dependência do movimento de um projétil num conjunto de parâmetros. Este aparato, operado remotamente, é acedido via web, onde primeiramente é reservado um intervalo de tempo. O conjunto de parâmetros ("*Bola*", "*Altura de lançamento*" e "*Ângulo de lançamento*") da máquina permite suportar vários cenários de ensino da Física, com diferentes complexidades.

Palavras-Chave: *Experimentação remota, Lançamento de projéteis, Laboratório remoto, Física*

Abstract

In an increasingly demanding educational system, experimentation plays a key role in the knowledge acquisition and validation. In physics education, the need to understand the environment influence on a given theoretical concept, leads to experimentation having a binding character.

In this context, there are three scenarios able to support the learning theoretical concepts acquisition. Simulation, which uses the speed computer and capabilities for calculating experiment result; traditional laboratory experiments, in which student perform their experiments in situ; and, finally remote experimentation, which allows real experiment execution of a without the physical presence of the student.

This thesis presents the design of an apparatus for remote experimentation on "Projectile Launch". In order to provide a more flexible mean for teaching physics, the developed apparatus allows students to determine the acceleration of gravity and projectile motion dependence on a set of parameters. This remotely operated apparatus is accessed via web by time scheduling. This machine contains a set of parameters ("*Ball*", "*Launch height* " and "*Launch angle*"), which support multiple teaching scenarios in physics with different complexities.

Keywords: *Remote experimentation, Projectile launch, Remote Laboratory, Physics*

Nota ao leitor

Na descrição dos vários projetos (mecânico, elétrico e eletrónico, e lógico), as descrições em tabelas, esquemas e em código, foram as utilizadas no decurso dos mesmos, pelo que é possível encontrar simultaneamente descrições em português e em inglês, bem como abreviaturas. Tal facto prende-se com a necessidade de descrever, com poucas palavras, alguns conceitos, para os quais, nem sempre é possível usar a nossa língua materna de forma sucinta.

Por outro lado, partes da literatura associada às áreas onde se insere este trabalho, também têm maioritariamente terminologia em inglês. Por esta razão, optou-se por usar, sempre que beneficie a compreensão, a língua inglesa, com a formatação em *itálico*.

No capítulo 6, as referências a código e nomes de objetos de programação (classes, interfaces, funções, etc.) estão entre aspas e com a fonte "`consolas`".

Índice

Índice de Figuras

Índice de Tabelas

Lista de Acrónimos

3D	–	*Three Dimensional* (A três dimensões)
AC	–	*Alternate Current* (Corrente alternada)
ASCII	–	*American Standard Code for Information Interchange* (Código americano normativo para troca de informação)
CAD	–	*Computer-Aided Design* (Projeto assistido por computador)
CI	–	Circuito Integrado
DC	–	*Direct Current* (Corrente contínua)
DFI	–	Departamento de Física do ISEP
DLL	–	*Dynamic Link Library* (Biblioteca de ligação dinâmica)
EEPROM	–	*Electrically Erasable Programmable Read Only Memory* (Memória só de leitura programável e apagável eletricamente)
EUA	–	Estados Unidos da América
FET	–	*Field Effect Transistor* (Transístor de efeito de campo)
ICSP	–	*In-Circuit Serial Programmer* (Programador série no circuito)
IDC	–	*Insulation Displacement Connector* (Conector de cabo paralelo flexível)
IDE	–	*Integrated Development Environment* (Ambiente integrado de desenvolvimento)
IPP	–	Instituto Politécnico do Porto
IR	–	*Infrared* (Infravermelho)
ISEP	–	Instituto Superior de Engenharia da Porto
LED	–	*Light Emitting Diode* (Díodo emissor de luz)
MEC	–	Ministério da Educação e Ciência
MOSFET	–	*Metal Oxide Semiconductor Field Effect Transistor* (Transístor de efeito de campo em semicondutor de oxido de metal)
MSDN	–	*Microsoft Development Network* (Rede de desenvolvimento da Microsoft™)
OOP	–	*Object Oriented Programming* (Programação orientada a objetos)
PC	–	*Personal Computer* (Computador pessoal)
PCB	–	*Printed Circuit Board* (Placa de circuito impresso)
PHP	–	*Hypertext Preprocessor* (Pré-processador de hipertexto)
RAM	–	*Random Access Memory* (Memória de acesso aleatório)
RISC	–	*Reduced Instruction Set Computer* (Computador com grupo reduzido de instruções)
SCK	–	*Serial Clock* (Relógio de sincronização série)
SDI	–	*Serial Data In* (Entrada série de dados)
SDO	–	*Serial Data Out* (Saída série de dados)
SMT	–	*Surface Mount Technology* (Tecnologia de montagem em superfície)
SOAP	–	*Simple Object Access Protocol* (Protocolo de acesso a objetos simples)
SPI	–	*Serial Peripheral Interface* (Interface de periféricos série)
SPICE	–	*Simulation Program with Integrated Circuit Emphasis* (Programa de simulação

		com ênfase nos circuitos integrados)
SSR	–	*Solid State Relay* (Relé de estado sólido)
UART	–	*Universal Asynchronous Receiver Transmitter* (Unidade universal e assíncrona de transmissão/recepção)
UL	–	*Unit Load* (Unidade de carga)
USB	–	*Universal Serial Bus* (Barramento universal série)
UTA	–	Unidade de Trabalho Ano
WCF	–	*Windows Communication Foundation*
WPF	–	*Windows Presentation Foundation*
XML	–	*Extensible Markup Language* (Linguagem extensível de marcação)

Lista de Acrónimos (só utilizados nos anexos)

AGC – *Automatic Gain Control* (Controlo automático de ganho)
CIL – *Common Intermediate Language* (Linguagem intermédia comum)
CLI – *Common Language Infrastructure* (Infraestrutura comum às linguagens)
CLR – *Common Language Runtime* (Código de execução comum)
COM – *Component Object Model* (Modelo de objetos para componentes)
CRC – *Cyclic Redundancy Check (*Verificação de redundância cíclica*)*
CS – *Chip Select* (Permissão do circuito integrado)
ESR – *Equivalent Series Resistor* (Valor equivalente da resistência em série)
EXE – Executável
FIFO – *First-In First-Out* (Primeiro a entrar, primeiro a sair)
FTP – *File Transfer Protocol* (Protocolo de transferência de ficheiros)
GC – *Garbage Collector* (Recolha de lixo)
HTTP – *Hyper Text Transfer Protocol* (Protocolo de transferência de hipertexto)
LSB – *Least Significant Digit* (Dígito menos significativo)
MSB – *Most Significant Digit* (Dígito mais significativo)
MSMQ – *Message Queuing* (Filas de mensagens)
NRZI – *Non Return to Zero Inverted* (Sem retorno a zero, invertido)
PCM – *Pulse Code Modulation* (Modulação por código de pulsos)
PE – *Portable Executable* (Executável portável)
PID – *Product Identifier* (Identificador do produto)
PLL – *Phase Locked Loop* (Malha de bloqueio de fase)
REST – *Representational State Transfer* (Transferência de representações de estados)
ROM – *Read Only Memory* (Memória só de leitura)
RPC – *Remote Procedure Calls* (Chamada de procedimentos remotos)
RS – *Recommended Standard* (Norma recomendada)
SE0 – *Single Ended 0* (Terminação única 0)
SIE – *Serial Interface Engine* (Motor de interface série)
SMD – *Surface Mount Device* (Unidade de montagem em superfície)
SMTP – *Simple Mail Transport Protocol* (Protocolo de transporte de correio)
SOA – *Service Oriented Architecture* (Arquitetura orientada a serviços)
VID – *Vendor Identifier* (Identificador do vendedor)
VCP – *Virtual COM Port* (Porto série virtual)
W3C – *World Wide Web Consortium*
WSD – *Web Service Description* (Descrição do serviço web)
WSDL – *Web Services Description Language* (Linguagem descritiva de serviços web)

Capítulo 1

INTRODUÇÃO

Num sistema de ensino cada vez mais exigente, a experimentação assume um papel fundamental na aquisição e validação do conhecimento. No ensino da física, a necessidade de compreender a influência do meio num dado conceito teórico leva a que a experimentação tenha um carácter obrigatório. Neste contexto, surgem três cenários capazes de suportar a aprendizagem dos conceitos teóricos adquiridos: (*i*) a simulação que faz uso da velocidade e capacidades de cálculo do computador para obter o resultado da experiência; (*ii*) a tradicional experimentação em laboratório, na qual o aluno executa presencialmente a sua experiência; e (*iii*), por último, a experimentação remota, que permite a execução de uma experiência real sem a presença física do aluno.

O Instituto Superior de Engenharia do Porto (ISEP) tem disponibilizado há vários anos a experimentação remota, mas a sua aplicação na Física tem sido exclusiva no domínio dos circuitos elétricos e eletrónicos. A utilização do VISIR [1] e do *Remote ElectLab* [2] permitiu a execução de experiências remotas neste domínio. A necessidade de uma abordagem semelhante que suportasse outras experiências de Física levou à necessidade de desenvolver uma máquina capaz de efetuar remotamente a experiência "*Lançamento de projéteis*". Este aparato suportando níveis diferentes de complexidade na caraterização do movimento do projétil, deverá poder ser usado em diferentes cenários de aprendizagem. Este trabalho foi apresentado na conferência internacional de Bilbau, REV2012 [3], e deu origem à publicação do artigo "*A Flexible Online Apparatus for Projectile Launch Experiments*" no IEEE [4].

1.1. CONTEXTUALIZAÇÃO

A realização da Tese de Mestrado em Engenharia Eletrotécnica e de Computadores – Ramo de Automação e Sistemas efetuou-se no âmbito do estudo e implementação de um aparato para experimentação remota do "*Lançamento de projéteis*", para o Departamento de Física (DFI) do ISEP, responsável pelo ensino da física a todos os cursos do ISEP, no qual se inclui a preparação dos conteúdos laboratoriais para apoio às matérias teóricas lecionadas.

Para complementar os laboratórios de física, foi lançado um novo projeto ao qual foi dado o nome "*Physics LabFARM*" que consiste na elaboração e/ou compra de equipamento capaz de providenciar remotamente experiências de física, suportado integralmente por fundos do ISEP. Os primeiros sistemas que integraram o "*Physics LabFARM*" foram o VISIR [1] e o "*Remote ElectLab*" [2], e destinaram-se à experimentação remota no campo da eletricidade e da eletrónica. Posteriormente, foram lançadas duas bolsas de investigação com vista à execução de experiências remotas no domínio da física. Estas bolsas destinaram-se a um investigador na área web/programação que seria responsável pelo interface cliente/servidor que iria integrar as várias experiências e providenciar o interface para cada uma delas. A outra bolsa de investigação destinou-se ao projeto dos aparatos e respetiva integração no servidor de experiências. No início deste trabalho foi elaborada uma lista de experiências passíveis de serem efetuadas remotamente:

- Lançamento de projéteis
- Estudo do Atrito

Depois da análise das várias opções, foi decidido iniciar o trabalho pelo "*Lançamento de projéteis*" já que a sua execução era mais abrangente e poderia servir de base às outras experiências.

1.2. OBJETIVOS

O objetivo desta tese é descrever a especificação, o projeto e a verificação e validação de um aparato capaz de efetuar a experiência remota "*Lançamento de um projétil*" (bola metálica). Como variáveis, deverá ser capaz de selecionar uma de três bolas, programar um lançamento com uma determinada altura, ao qual é imposto um ângulo de lançamento. O lançamento é efetuado com uma velocidade inicial nula. O aparato executará a experiência com os dados fornecidos pelo utilizador, e após a sua conclusão, enviará para o servidor os resultados obtidos ('*tempo de voo*' e '*alcance horizontal do projétil*'), bem como uma fotografia do momento de impacto. Simultaneamente, é fornecido ao utilizador um canal de áudio e vídeo para que a experiência possa ser acompanhada em tempo real. A solução deverá ser móvel e a sua utilização necessitará unicamente de uma tomada de 230 V_{AC} e de um ponto de rede.

1.3. CALENDARIZAÇÃO

Este trabalho teve a duração de dois anos e foi escalonado em várias etapas. A Tabela 1 apresenta o diagrama de Gantt com as várias etapas do projeto.

Tabela 1 Diagrama temporal da implementação

ID	Tarefa	Início	Fim	Duração
1	Estudo do problema proposto	04-10-2010	20-10-2010	15d
2	Identificação dos requisitos estruturais	21-10-2010	06-11-2010	15d
3	Esboço da estrutura	08-11-2010	24-11-2010	15d
4	Análise nas necessidades de controlo	25-11-2010	11-12-2010	15d
5	Análise dos requisitos de acesso	13-12-2010	18-12-2010	6d
6	Análise dos requisitos de interface	20-12-2010	31-12-2010	11d

ID	Tarefa	Início	Fim	Duração
1	Desenho do elevador	01-04-2011	06-06-2011	47d
2	Execução elevador (Tecnogial)	07-06-2011	01-08-2011	40d
3	Montagem do elevador	02-08-2011	26-08-2011	19d
4	Desenho da rampa e recolha	07-06-2011	25-07-2011	35d
5	Execução da rampa e recolha	26-07-2011	30-08-2011	26d
6	Montagem da rampa e recolha	31-08-2011	30-09-2011	23d
7	Compra de material Nanotec	06-06-2011	11-07-2011	26d
8	Desenho da estrutura	26-07-2011	01-08-2011	5d
9	Projeto do sensor de bola	29-08-2011	16-09-2011	15d
10	PCB Sensor de bola - Pcb-Pool	19-09-2011	27-09-2011	7d
11	Compra de material Digikey	21-09-2011	30-09-2011	8d

ID	Tarefa	Início	Fim	Duração
1	Montagem dos sensores de bola	03-10-2011	10-10-2011	6d
2	Cablagem geral	11-10-2011	25-11-2011	34d
3	Caixa de distribuição e expansor SMCI12	31-10-2011	28-11-2011	21d
4	PCB Cx. Dist. e Exp. SMCI12-Pcb-Pool	29-11-2011	07-12-2011	7d
5	Montagem da Cx.Dist. E Exp. SMCI12	08-12-2011	28-12-2011	15d
6	Serviços WEB e interface utilizador Firmware PIC	11-10-2011	30-03-2012	124d
7	Projeto do sistema de medida	27-02-2012	13-04-2012	35d

ID	Task Name	Start	Finish	Duration
1	Projeto do sistema de medida	02-04-2012	17-04-2012	12d
2	PCB Sistema de medida	18-04-2012	26-04-2012	7d
3	Montagem do sistema de medida	27-04-2012	17-05-2012	15d
4	Estrutura, PC e instalação elétrica inferior	17-05-2012	18-06-2012	23d
5	Escrever Tese	20-06-2012	30-10-2012	95d

O diagrama temporal da implementação tem a seguinte legenda:

– Estudos de análise e viabilidade

– Projeto mecânico, eletrónico e controlo (Windows™ e PIC)

– Trabalhos executados externamente (mecânica e PCB's)

– Trabalhos de montagem (mecânica e eletrónica)

– Fornecimento de materiais (Nanotec™ e Digikey™)

– Escrita deste documento

1.4. ORGANIZAÇÃO DA TESE

Esta tese está organizada da seguinte forma: Neste capítulo introduziu-se o contexto da tese, os objetivos a atingir e a respetiva calendarização. No capítulo dois, expõe-se o problema a tratar, detalhando as questões físicas associadas ao problema, e, num estudo sobre a experimentação e suas vertentes, realça-se o estado da arte na experimentação remota. No capítulo três, é apresentada a arquitetura geral do aparato, com uma nota introdutória sobre os aspetos logísticos de aquisição de material, e enumeram-se os requisitos funcionais e de implementação para cada um dos constituintes do projeto. O capítulo quatro apresenta a implementação mecânica do aparato, onde são descritos os vários componentes intervenientes e a sua interdependência. O capítulo cinco apresenta a implementação elétrica e eletrónica, que inicia com a descrição da rede interna RS-485, questão central a todo o projeto eletrónico, detalha todo o *hardware* desenvolvido e finaliza com a distribuição de energia e interligação geral do aparato. O capítulo seis introduz a lógica de controlo, detalhando as duas partes constituintes: (*i*) o controlo de alto nível, com a implementação do serviço web, o interface de utilizador e controlador de experiência; e (*ii*) o controlo de baixo nível, de implementação nos microcontroladores individuais de cada módulo projetado para o aparato. No capítulo sete são analisados os resultados obtidos, caracterizando o tempo de utilização e de execução da experiência e os custos de desenvolvimento. Finalmente, no capítulo oito, são apresentadas as conclusões desta tese, enumerando os problemas encontrados no decorrer do projeto e apresentando algumas sugestões de melhoria.

Capítulo 2

ESTUDO DO MOVIMENTO DE UM PROJÉTIL

No capítulo anterior foram introduzidos os conceitos que suportam a utilização da experimentação como base fundamental na aquisição e validação do conhecimento. Assim, foram introduzidos os cenários experimentais possíveis, referindo a simulação que utiliza a velocidade e capacidade de cálculo de um computador para a obtenção dos resultados, a experimentação presencial em ambiente laboratorial, em que o aluno implementa a experiência desejada (pode estar previamente preparada) e obtém os seus resultados condicionados pelas condições reais (e.g. atrito), e finalmente a experimentação remota, que executa uma experiência física, sujeita às condições reais, mas efetuada sem a presença local do aluno.

A primeira secção deste capítulo apresenta os conceitos fundamentais de Física necessários à compreensão do estudo do "*Movimento de um projétil*". Finalizado este estudo, é apresentado o cenário da simulação, com referências a exemplos de simuladores existentes. A apresentação continua com o cenário de experimentação presencial, incluindo algumas das suas limitações. Finalmente, é apresentada a experimentação remota e algumas das suas vantagens/desvantagens, face aos cenários anteriormente descritos. O capítulo finaliza com a apresentação de alguns laboratórios de experimentação remota em diversas áreas da Física.

2.1. CONCEITOS FÍSICOS FUNDAMENTAIS

Nesta secção introduzem-se os conceitos fundamentais de Física, necessários à compreensão do estudo do "*Movimento de um projétil*".

2.1.1. MOVIMENTO

O estudo geral das relações entre movimento, força e energia é denominado de mecânica. A mecânica pode ser subdividida em subdisciplinas pela combinação e recombinação dos seus aspetos. O movimento é a ação de alterar uma posição. O estudo do movimento sem ter em conta as forças ou energias que possam estar envolvidas denomina-se *cinemática*. O estudo do movimento e das forças que lhe dão origem intitula-se *dinâmica*, e finalmente, o estudo das forças na ausência de alterações no movimento ou na energia é denominado *estática*.

O termo energia refere-se a uma quantidade física abstrata que não é facilmente percebida pelos seres humanos, e pode existir simultaneamente sob várias formas. A energia do movimento é denominada de *energia cinética*. Sempre que um sistema é afetado por um agente exterior (genericamente uma força), a sua energia total é alterada. Quando uma força provoca uma alteração na energia de um sistema, diz-se que a força produziu trabalho, e relacionada pela expressão matemática do teorema do trabalho-energia ou teorema da energia cinética. Quando o total de todas as diferentes formas de energia é quantificado, verifica-se que em sistemas isolados este é constante (lei da conservação de energia) [5].

2.1.2. DESLOCAMENTO E VELOCIDADE MÉDIA

Tipler [6] apresenta a definição de velocidade média de uma partícula como a razão entre a distância total percorrida pela partícula e o tempo total consumido no percurso, sendo a sua unidade no sistema internacional (SI) de metros por segundo ($m.s^{-1}$), embora a utilização da unidade quilómetros por hora ($km.h^{-1}$) seja também comum. A velocidade média não fornece qualquer informação sobre os detalhes do percurso. O conceito físico de velocidade inclui a direção do movimento. Para ser possível entender este conceito, é necessário perceber o conceito de deslocamento, que representa a variação da posição da partícula ao longo do tempo. Tipicamente, a variação de uma grandeza é representada pela letra grega delta maiúscula (Δ). Assim, a fórmula (1) descreve a definição de deslocamento:

$$\Delta x = x_2 - x_1 \tag{1}$$

A velocidade define-se como a taxa de variação da posição, sendo a velocidade média de uma partícula a razão entre o deslocamento e o intervalo de tempo (2):

$$v_m = \frac{\Delta_x}{\Delta_t} = \frac{x_2 - x_1}{t_2 - t_1} \quad (2)$$

Dado que x_2 pode ser maior ou menor que x_1, a velocidade média pode ser positiva (num referencial cartesiano indica um movimento para a direita) ou negativa (movimento para a esquerda). A menos que a velocidade seja constante, a velocidade média depende do intervalo de tempo em que está baseada.

2.1.3. VELOCIDADE INSTANTÂNEA

A velocidade instantânea (v) num certo ponto é o coeficiente angular da reta tangente à curva de x em função de t neste ponto. Assim a velocidade instantânea é o limite da razão Δ_x/Δ_t, quando Δ_t tende para zero (3).

$$v = \lim_{\Delta t \to 0} \frac{\Delta_x}{\Delta_t} = \frac{dx}{dt} \quad (3)$$

É importante distinguir cuidadosamente entre velocidade média e velocidade instantânea. Na linguagem comum, quando nos referimos a "*velocidade*" de uma forma isolada, estamos possivelmente a referir a velocidade instantânea.

2.1.4. ACELERAÇÃO

Quando a velocidade de uma partícula se altera ao longo do tempo, diz-se que a partícula está animada de uma aceleração. A aceleração média (a_m) num determinado intervalo de tempo $\Delta_t = t_2 - t_1$, define-se como a razão entre a variação da velocidade instantânea por intervalo de tempo (4):

$$a_m = \frac{\Delta_v}{\Delta_t} = \frac{v_2 - v_1}{t_2 - t_1} \quad (4)$$

No SI, a aceleração (a) expressa-se em metros por segundo ao quadrado ($m.s^{-2}$). A aceleração instantânea é o limite da razão entre a variação da velocidade instantânea, quando Δ_t tende para zero (5):

$$a = \lim_{\Delta t \to 0} \frac{\Delta_v}{\Delta_t} = \frac{dv}{dt} = \frac{d\left(dx/dt\right)}{dt} = \frac{d^2x}{dt^2} \quad (5)$$

Se a velocidade for constante, podemos verificar que a aceleração é nula, pois $\Delta_v = 0$, em qualquer intervalo de tempo.

2.1.5. MOVIMENTO UNIFORMEMENTE ACELERADO

O movimento de uma partícula com aceleração constante (movimento uniformemente acelerado) é muito comum na natureza, e.g. um corpo em queda livre cai com uma aceleração constante, provocada pela gravidade, desprezando a resistência do ar. A aceleração da gravidade é: $g = 9{,}81\ \mathrm{m.s^{-2}}$. A existência de uma aceleração constante significa que a velocidade varia linearmente com o tempo. Se v_0 for a velocidade inicial da partícula, podemos calcular o seu valor v, num instante t posterior, pela equação (6):

$$v = v_0 + at \tag{6}$$

Como consequência da velocidade variar linearmente com o tempo, a velocidade média, neste movimento, é dada pela equação (7):

$$v_m = \frac{1}{2}(v_0 + v) \tag{7}$$

O seu deslocamento resulta da equação (8):

$$\Delta x = v_m t = \frac{1}{2}(v_0 + v)t = v_0 t + \frac{1}{2}at^2 \tag{8}$$

A sua posição, com as condições iniciais ($x_0\ ;\ v_0$), é dada pela equação (9):

$$x = x_0 + v_0 t + \frac{1}{2}at^2 \tag{9}$$

2.1.6. MOVIMENTO DOS PROJÉTEIS

Um projétil é um corpo que é lançado no ar, move-se livremente sob ação da gravidade e a sua trajetória descreve uma parábola. O movimento pode ser considerado como resultante da composição (soma vetorial) de dois movimentos unidimensionais independentes:

- O projétil tem uma aceleração constante, dirigida verticalmente para baixo, com o módulo $g = 9{,}81\mathrm{m.s^{-2}}$
- Em relação ao solo, desloca-se com um movimento de velocidade horizontal constante ($v_x = v_{0x}$)

Para que este tipo de lançamento seja passível de ser efetuado, é necessária a aplicação de uma força ao projétil para que o seu lançamento se inicie com uma velocidade inicial v_0 não nula. O ângulo inicial de lançamento θ e a altura inicial h_0 em conjunto com a velocidade inicial definem as condições iniciais de lançamento. O projétil está sujeito à aceleração constante da força da

gravidade. No contexto experimental, é prática comum desprezar o atrito do ar de forma a simplificar os cálculos. A Figura 1 apresenta um diagrama da experiência desejada.

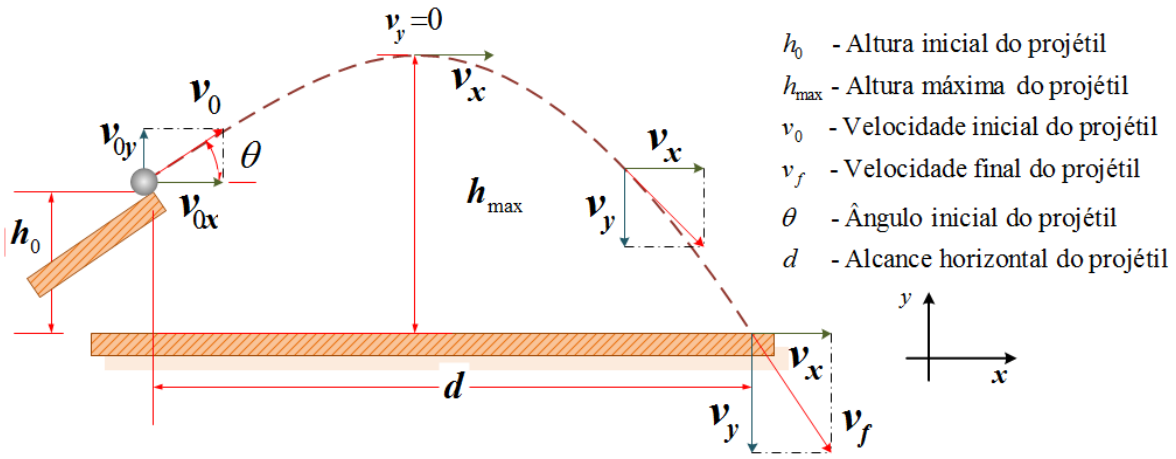


Figura 1 Diagrama do lançamento de um projétil

Consideremos as condições iniciais h_0, v_0 e θ, de uma partícula que é lançada com uma velocidade inicial, não nula. A aceleração, Equação (10), só apresenta componente no eixo vertical:

$$\begin{cases} a_x = 0 \\ a_y = -g \end{cases} \tag{10}$$

Se o projétil é lançado na origem com uma velocidade escalar inicial v_0, fazendo um ângulo θ com o eixo horizontal, então as componentes da velocidade inicial são dadas pela Equação (11).

$$\begin{cases} v_{0x} = v_0 \cdot \cos\theta \\ v_{0y} = v_0 \cdot \operatorname{sen}\theta \end{cases} \tag{11}$$

As componentes da velocidade do projétil são dadas na Equação (12), pois como não existe aceleração horizontal, a componente v_x é constante:

$$\begin{cases} v_x = v_{0x} \\ v_y = v_{0y} - gt \end{cases} \tag{12}$$

Assim as componentes do deslocamento do projétil são dadas pela equação (13):

$$\begin{cases} \Delta x = v_{0x} \cdot t \\ \Delta y = v_{0y} \cdot t - \frac{1}{2} g t^2 \end{cases} \Rightarrow \begin{cases} x = x_0 + v_{0x} \cdot t \\ y = y_0 + v_{0y} \cdot t - \frac{1}{2} g t^2 \end{cases} \tag{13}$$

O tempo de voo (14) é obtido da Equação (13), na componente do deslocamento y, fazendo $y = 0$:

$$0 = y_0 + v_{0y} \cdot t - \frac{1}{2} g \cdot t^2 \Longrightarrow t = \frac{v_{0y} + \sqrt{{v_{0y}}^2 + 2 \cdot g \cdot y_0}}{g} \tag{14}$$

O alcance horizontal do projétil é obtido diretamente da Equação (13) da componente x do deslocamento, sendo que v_{0x} é a componente x da velocidade inicial, e substituindo t pelo tempo de voo obtido na Equação (14).

O tempo para atingir a altura máxima (15) é obtido da Equação (12) das velocidades, fazendo $v_y = 0$:

$$v_y = v_{0y} - gt, v_y = 0 \Longrightarrow t = \frac{v_{0y}}{g} \quad (15)$$

A altura máxima (h_{max}) atingida pelo projétil (16) é obtida diretamente da Equação (13) da componente y do deslocamento, substituindo t pelo tempo para atingir a altura máxima (15).

$$h_{máx} = h_{inicial} + v_{0y} \cdot t - \frac{1}{2} g t^2 = h_{inicial} + \frac{{v_{0y}}^2}{2g} \quad (16)$$

2.2. EXPERIMENTAÇÃO

Na história da educação, o processo de ensino foi sendo alterado de acordo com a conceção de cada época. A investigação mais recente refere os laboratórios remotos como meio complementar às aulas laboratoriais presenciais [7]. Esta secção apresenta as várias vertentes da experimentação, iniciando pela simulação, que tende a imitar a operação de um processo ao longo do tempo, apresentando alguns exemplos de simuladores. A subsecção seguinte expõe a experimentação presencial, e mostra algumas das suas restrições. Finalmente é apresentada a experimentação remota, como meio prático e físico complementar às aulas presenciais. Aqui são apresentados alguns exemplos de laboratórios remotos disponíveis.

2.2.1. POR SIMULAÇÃO

A simulação é a imitação da operação de um processo ou sistema real ao longo do tempo. O ato de simular algo requer o desenvolvimento de um modelo que representa as principais caraterísticas ou comportamentos do processo/sistema físico ou abstrato. O modelo representa o próprio sistema e a simulação o seu funcionamento ao longo do tempo.

Segundo Sokolowski e Banks [8], *"A modelação e simulação, normalmente conhecida por M&S, têm-se tornado na primeira escolha dos estudantes, em todas as disciplinas. A M&S é uma disciplina com o seu próprio conhecimento, teoria e metodologia de investigação. No centro da disciplina, está a noção fundamental de que os modelos são aproximações do mundo real. Para se envolverem na M&S, os estudantes devem primeiramente criar um modelo aproximado de um evento. Esse modelo será seguido via simulação, que permite repetir a observação do modelo.*

Após um ou várias simulações do modelo, é necessário um terceiro passo: a análise. A análise facilita tirar conclusões, verificar e validar a investigação, e fazer recomendações baseadas nas várias interações ou simulações do modelo. Estes conceitos, juntamente com a visualização, que é a habilidade para representar informação como veículo de interface com o modelo, tornam a M&S uma disciplina baseada em problemas que permite o teste repetitivo de hipóteses."

A simulação é utilizada em diversos contextos, segurança, teste, treino, educação e jogos. Simuladores de treino, como por exemplo os simuladores de voo, destinam-se ao treino de pilotos, em situações semelhantes às reais. A simulação é utilizada na modelização científica de sistemas naturais ou humanos, para se obterem respostas a questões sobre o seu funcionamento. A simulação é essencial no estudo de sistemas inacessíveis, ou cuja manipulação é potencialmente perigosa, ou que, no limite, ainda não existem. A simulação por computador é a tentativa de modelar uma situação real ou hipotética num computador para ser estudado e verificado o seu funcionamento. A manipulação das variáveis na simulação permite determinar o comportamento do sistema em estudo. São muitos os exemplos de simuladores disponíveis na internet para as mais diversas áreas. No ensino, podemos encontrar simuladores em praticamente todas as suas vertentes (e.g. Eletrónica, Física, Estruturas mecânicas, etc.). Como exemplo de um simulador utilizado na engenharia eletrotécnica, temos *"o primeiro simulador de circuitos, o "Simulation Program with Integrated Circuit Emphasis" (SPICE), que foi desenvolvido na década de 70 para apoiar o fabrico de semicondutores, prevendo o seu funcionamento antes da execução das máscaras litográficas para o fabrico do CI."* [9].

Tendo como base o trabalho desenvolvido, apresentam-se aqui dois simuladores da experiência "*Lançamento de projéteis*":

1) A Universidade do Colorado, nos Estados Unidos da América (EUA), disponibiliza um simulador [10], em que o utilizador especifica o ângulo de lançamento do projétil, a sua massa, diâmetro e velocidade inicial (Figura 2). Tem também a possibilidade de ter em consideração a resistência do ar, fornecendo em resposta, uma imagem gráfica do lançamento, bem como os resultados obtidos ('*alcance*', '*altura*' e '*tempo de voo*'). No caso de se considerar a resistência do ar, o utilizador terá de fornecer os dados do coeficiente de atrito e a altura considerada. A figura apresenta os resultados da simulação do lançamento de um projétil, com o mesmo ângulo, massa e velocidade inicial, mas considerando a resistência do ar (lançamento a vermelho).

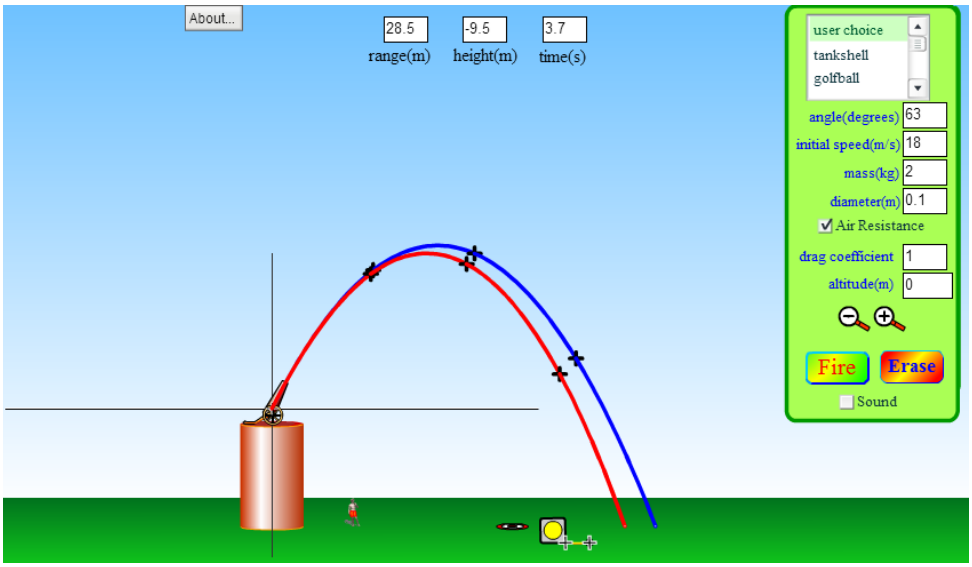


Figura 2 Simulador de "*Lançamento de um projétil*", Universidade do Colorado (EUA) [10]

2) O departamento de Física da Universidade da Virgínia, EUA, tem um vasto conjunto de programas [11], desenvolvidos em JAVA, para vários tipos de simulação. A Figura 3 apresenta o seu simulador "*Projectile Motion*" [12].

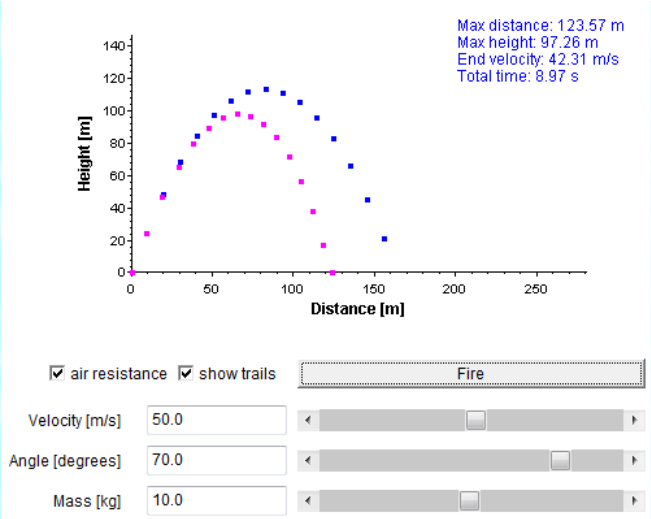


Figura 3 Simulador "*Projectile motion*" da Universidade da Virgínia (EUA) [12]

2.2.2. EM LABORATÓRIO PRESENCIAL

Um dos aspetos fundamentais no ensino dentro das áreas tecnológicas e das ciências naturais é a prática adquirida pelos alunos, na manipulação de instrumentos mecânicos e/ou eletrónicos. A experimentação permite aplicar e desenvolver os conhecimentos teóricos adquiridos.

Até alguns anos atrás, as aulas laboratoriais estavam limitadas a laboratórios clássicos, onde os custos de manutenção e aquisição de novos instrumentos são muito elevados e até restritivos para

muitas instituições. Além disso, ao utilizar um laboratório presencial, o número de alunos com acesso simultâneo é muito reduzido e com horários restritos [13].

Adicionalmente, o acesso à experiência está restrito ao tempo de aula, sob a supervisão do professor. Se os alunos tiverem dúvidas na experiência executada, ou se por um qualquer motivo os resultados não forem coerentes com a respetiva fundamentação teórica, não lhes é possível efetuar a mesma experiência para verificação dos resultados obtidos, já que os recursos laboratoriais de uma dada experiência tipicamente só estão disponíveis numa aula. Na aula seguinte, a mesma experiência estará disponibilizada a outro grupo de alunos, ou o grupo de alunos em causa estará ocupado com a execução do trabalho seguinte.

A experimentação presencial requer a replicação das várias experiências, em número suficiente para ser efetuada por todos os alunos de uma turma. O rendimento do tempo de ocupação de uma experiência é muito baixo, mesmo sendo otimizado na sua distribuição por vários grupos ao longo de uma aula laboratorial, já que no seu decorrer, a preparação da experiência, a execução desta, recolha de dados e a sua análise crítica é executada de uma forma contínua, e o recurso está inativo até que uma nova experiência seja efetuada. O facto de ser necessária a replicação das experiências implica a necessidade de laboratórios com dimensões apropriadas para suportar todas as experiências em curso numa aula.

Adicionalmente, a utilização de laboratórios presenciais requer tempos de preparação da experiência muito elevados, normalmente efetuadas por pessoas adstritas e/ou especializadas. A existência de muitas réplicas de uma experiência implica a existência de equipamento e material de substituição já que a sua manipulação constante por parte dos alunos leva à sua detioração. Em muitos casos, não é raro os alunos verem uma determinada experiência cancelada por avaria do equipamento utilizado.

2.2.3. VIA LABORATÓRIO REMOTO

O avanço da tecnologia de informação, nomeadamente a internet, possibilitou a criação e disponibilização de vários serviços em diversas áreas, desde a comercialização de bens e serviços, sistemas de comunicação (texto, voz e vídeo), correio eletrónico, bibliotecas virtuais, até às atividades lúdicas (e.g. redes sociais), etc.. Na área do ensino, o armazenamento e distribuição de informação e a gestão de conteúdos (e.g. sistema de gestão de ensino Moodle (LMS) [14]), os ambientes de trabalho colaborativo, a videoconferência e as aulas virtuais, são alguns exemplos normalmente utilizados. A experimentação remota apresenta-se como uma das áreas promissoras na utilização das tecnologias de informação na área de ensino [7]. A experimentação é, como já foi afirmado, um dos fatores fundamentais na perceção e assimilação dos conceitos teóricos ensinados.

Segundo Juarez [13],

"*Os laboratórios remotos para práticas buscam resolver de uma forma efetiva e prática os problemas de acesso aos laboratórios clássicos, com o objetivo de:*

• *Incrementar as atividades práticas em um curso (de forma que os alunos possam acessar a eles em qualquer horário, não somente quando esteja aberto o centro para temas docentes),*

• *Reduzir os custos de gestão e manutenção dos laboratórios (ao aumentar o uso em qualquer horário aos mesmos com um pessoal menor),*

• *Permitir o uso dos mesmos desde qualquer ponto geográfico de forma que se reduzam ou minimizem os custos de deslocamento, assim como a qualquer hora, permitindo desta forma resolver o problema dos fusos horários com outras zonas geográficas, e,*

• *Integrar em um mesmo ambiente as aplicações docentes das práticas, experimentação e trabalho no laboratório, com as atividades propriamente docentes mediante a integração de materiais, simulações e acesso a equipamentos e dispositivos.*"

A experimentação remota pode e deve ser usada em complemento às aulas laboratoriais. Os alunos podem no seu ambiente de estudo, em casa ou com os colegas, executar as mesmas experiências, no horário que lhes convém, e dessa forma sentirem-se mais atraídos pelas matérias a aprender.

Por outro lado, a experimentação remota abre caminho à partilha de experiências entre as várias instituições de ensino, o que leva a uma otimização da sua utilização. O seu custo, ainda que em algumas situações possa ser algo oneroso (e.g. experimentação remota de sistemas mecânicos ou químicos), poderá ser diluído pelos vários intervenientes, e.g. sob a forma de pagamento de uma taxa de utilização. Este conceito de laboratório remoto permite a integração de experiências, que de outro modo poderiam não estar disponíveis.

MÉTODOS DE ACESSO À EXPERIENCIA

Uma experiência remota, em função do seu tempo de configuração e do seu tempo de execução, é tipicamente acedida de duas formas distintas:

- Fila de espera – Os pedidos de execução são encaminhados para o servidor de experiência, são atendidos por ordem de chegada, e, após a sua finalização, enviam-se os resultados para o utilizador.
- Agendamento – O utilizador reserva num processo de agendamento, um espaço temporal para acesso exclusivo à experiência.

As experiências remotas com acesso por fila de espera são tipicamente experiências com tempos de execução muito curtos, e cuja configuração é efetuada com o sistema desligado, e.g. área da experimentação eletrónica ("*VISIR*" [1] e "*Remote Electlab*" [2]). Neste grupo de experiências remotas, a configuração dos componentes básicos é feita previamente e disponibilizada ao utilizador, para em função da experiência desejada, poder configurar um determinado circuito elétrico ou eletrónico. Aqui, o tempo de execução é muito curto, o que permite uma utilização do recurso quase em tempo real (os alunos não têm a perceção de estar numa fila de espera).

Nas experiências remotas com acesso por agendamento, são tipicamente experiências cujo tempo de execução é longo e nas quais, os recursos são de utilização exclusiva (e.g. experiências implementadas por sistemas mecânicos, com ciclos de execução demorados). Esta é a situação do aparato desenvolvido no âmbito desta tese.

2.2.4. EXEMPLOS DE IMPLEMENTAÇÕES DE EXPERIÊNCIAS REMOTAS

O primeiro exemplo apresenta um laboratório remoto existente no departamento de Física dedicado à experimentação remota de circuitos elétricos e eletrónicos. O projeto denominado "*Virtual Instrument Systems In Reality*" (VISIR) foi iniciado na Suécia, com a finalidade de disseminar métodos para abrir os laboratórios à experimentação remota [15]. Este laboratório remoto permite a execução destas experiências de uma forma simultânea, e fornece aos alunos um conjunto de instrumentos, cujo interface virtual é semelhante ao encontrado nos instrumentos de bancada existentes no laboratório. Para a execução da experiência, é disponibilizada uma placa de montagem virtual ("*Breadboard*"), na qual os alunos procedem à montagem do circuito. Para o acesso à experiência, é requerido um acesso à internet e um navegador com o módulo de FLASH ("*FLASH player*") instalado. A Figura 4 apresenta o interface de montagem de um circuito com um amplificador operacional cuja interligação foi efetuada pelo utilizador.

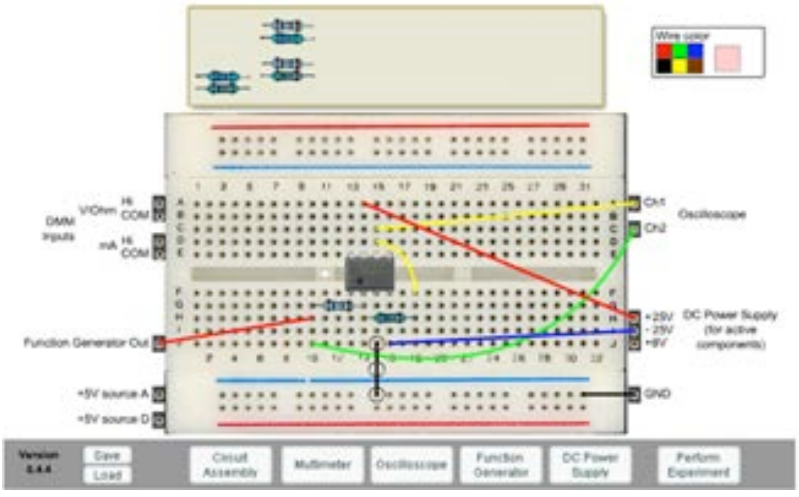


Figura 4 VISIR – Interface de montagem do circuito [15]

O laboratório remoto disponibiliza um osciloscópio de dois canais, um gerador de funções, um multímetro digital, uma fonte de alimentação tripla e uma matriz de comutação, onde é configurado fisicamente o circuito. A Figura 5 apresenta o interface virtual do osciloscópio, no qual é visualizada a forma de onda.

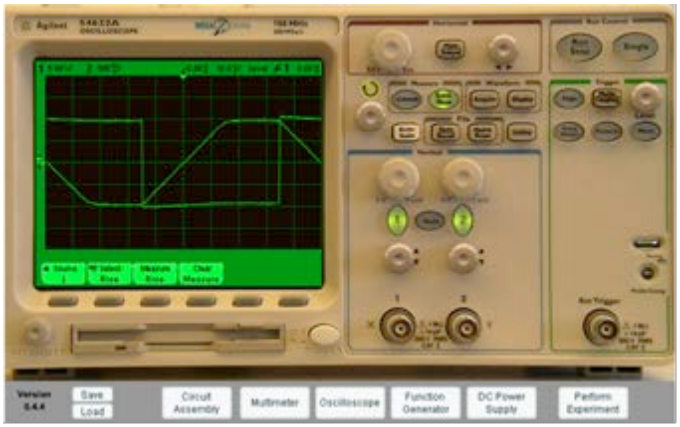


Figura 5 VISIR – Interface virtual do osciloscópio [15]

O projeto *rexlab* é um laboratório de experimentação remota desenvolvido no polo universitário de Araranguá, Santa Catarina, Brasil, que disponibiliza um conjunto significativo de experiências [16]. A página de entrada apresenta uma lista de seis experiências disponíveis. Uma das experiências disponibilizadas é o "Módulo de Young – Resistência de materiais"[1], cujo interface permite selecionar uma força (100 g, 200 g ou 300 g) a ser aplicada a uma barra. A experiência fornece um canal de vídeo, onde é possível ver o mostrador de um comparador, que apresenta a ação da força selecionada, cuja montagem é apresentada na foto da direita na Figura 6.

Figura 6 Rexlab – Experiência de resistência de materiais [16]

[1] A experiência pode ser acedida em http://rexlab.ararangua.ufsc.br/experimentos/young/

Num último exemplo, a Universidade de Trnava, na Eslováquia apresenta a experiência "Queda de um grave" [17]. A experiência eleva um magneto que é largado em queda livre numa bobina, de forma a medir a variação da força eletromotriz ao longo do tempo, no ar e no líquido (Figura 7).

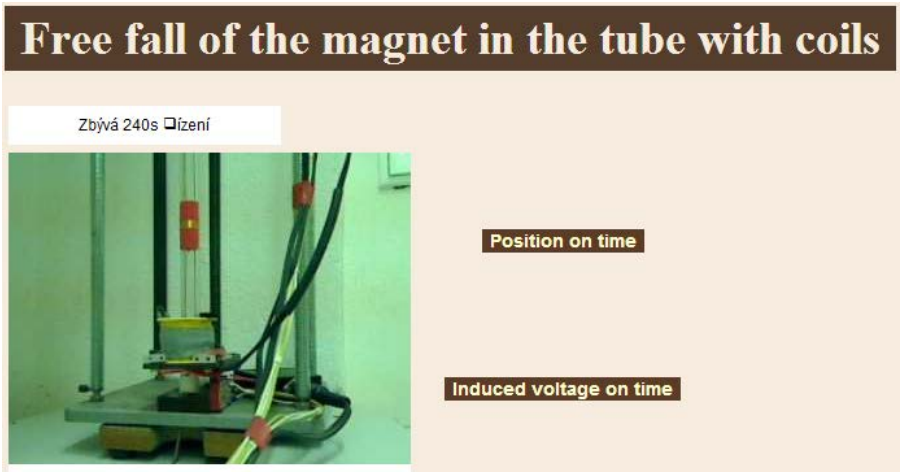


Figura 7 Queda de um magneto num tubo com bobinas [17]

O interface[2], com uma câmara web que apresenta o movimento do elevador do magneto utilizado, permite o estudo da posição ao longo do tempo, ou da tensão induzida ao longo do tempo. O interface disponibiliza o acesso à experiência de uma forma direta, pelo que não é possível perceber como são resolvidas as situações de conflito concorrencial (não se vislumbra nem um agendamento ou um sistema de fila de espera). No fim da experiência é apresentado um gráfico com as leituras, que pode ser exportado para uma folha de cálculo (Microsoft™ Excel™).

2.3. CONCLUSÃO

Este capítulo iniciou com a apresentação dos conceitos fundamentais de Física, necessários à compreensão do estudo do "*Movimento de um projétil*", definindo o conceito de movimento e as leis que o regem. A secção de conceitos fundamentais terminou com as fórmulas para determinar as variáveis em causa neste estudo ('*tempo de voo*', '*alcance horizontal do projétil*', e '*altura atingida pelo projétil*'). A segunda secção explica cenários experimentais possíveis, por simulação, experimentação presencial e experimentação remota, com apresentação de alguns exemplos existentes de simuladores e de laboratórios de experimentação remota. O próximo capítulo introduz a arquitetura geral de um aparato que visa a implementação física da experiência remota de "*Lançamento de projéteis*".

[2] A experiência pode ser acedida em http://remotelab4.truni.sk/

Capítulo 3

ARQUITETURA GERAL DO APARATO EXPERIMENTAL

Um aparato, de uma forma geral, é todo o dispositivo que executa ou ajuda no desempenho das tarefas e que depende de uma fonte de energia para o seu funcionamento, e neste caso, destina-se à execução de uma experiência física. Para o seu projeto é necessária a resolução de vários problemas, que envolvem áreas distintas da engenharia, nomeadamente, a mecânica, a eletrónica, e a informática. O presente capítulo introduz a arquitetura necessária para a resolução do problema proposto e a problemática relacionada com a aquisição de material por parte de uma instituição pública e os condicionalismos de projeto daí resultantes, seguida dos requisitos funcionais e de implementação de cada um dos seus subconjuntos.

3.1. INTRODUÇÃO À ARQUITETURA DO APARATO

A finalidade desta experiência é permitir que um aluno possa verificar remotamente e sob cenários diferentes, as leis que regem o movimento de um projétil.

A execução da experiência proposta requer um aparato com capacidade de encaminhar o projétil até à rampa de lançamento, e, após configuração dos parâmetros especificados pelo utilizador, efetuar o lançamento do projétil recolhendo em seguida todas as medidas associadas à experiência.

Posteriormente, o projétil deverá ser recolhido para possibilitar a repetição do procedimento. Como premissa do problema apresentado na Figura 1do Capítulo 1 é requerida a aplicação de uma força necessária à criação de uma velocidade inicial não nula, com o ângulo de lançamento igual ao ângulo da direção desta força com o plano horizontal.

Esta proposta de solução utiliza a força da gravidade em conjunto com um plano inclinado (rampa de lançamento) para impor a velocidade inicial ao projétil. Para permitir configurar a altura inicial 'y_o', a rampa de lançamento é dotada de movimento vertical. Para ser possível ajustar o ângulo de lançamento 'θ', a rampa tem capacidade de girar em torno do seu eixo. Desta forma é possível configurar todos os parâmetros desejados.

Pretende-se ainda que seja possível mostrar a independência das leis que regem o lançamento de um projétil relativamente à sua massa. Dessa forma, a solução disponibiliza um seletor de projéteis de massas diferentes (esferas do mesmo material com diâmetros diferentes). A Figura 8 apresenta o diagrama de blocos da proposta de solução para o problema colocado.

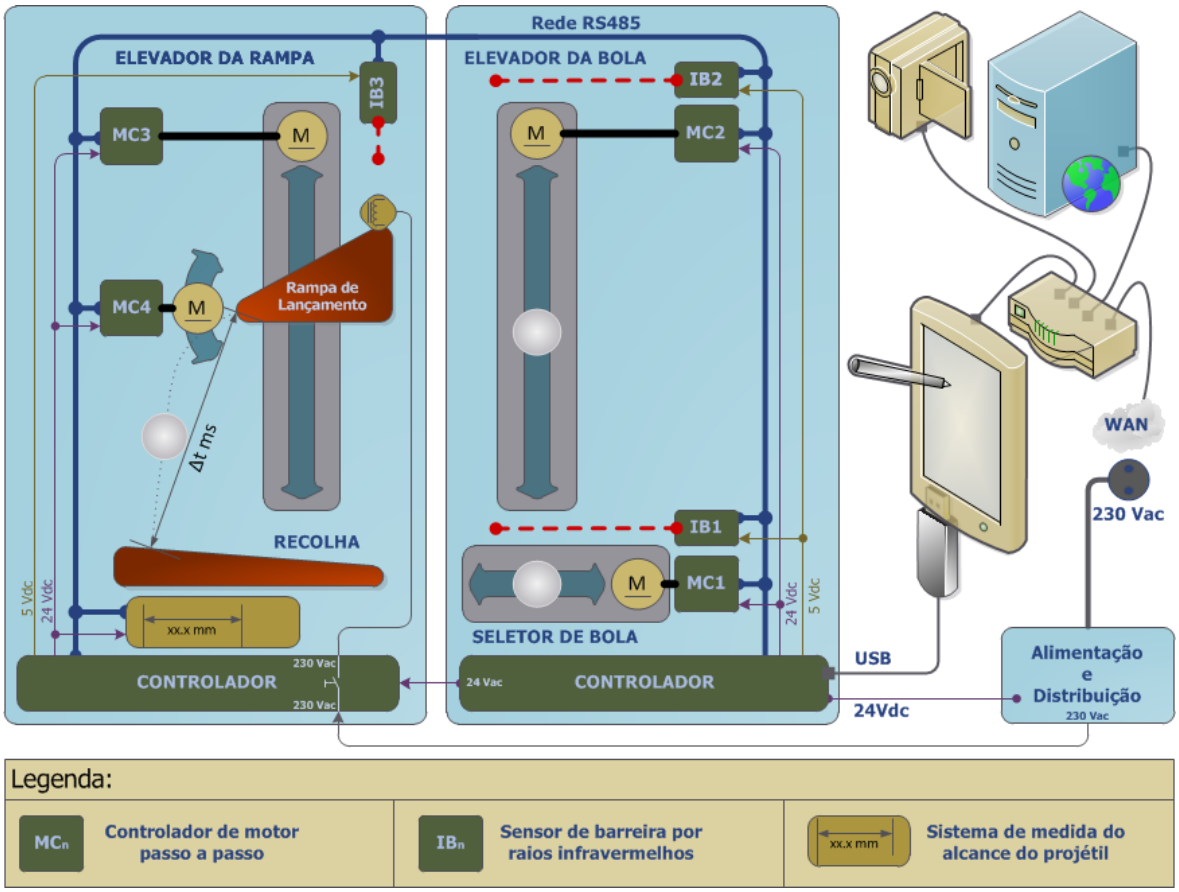


Figura 8 Diagrama de blocos do aparato

Da análise das várias operações mecânicas necessárias para viabilizar a execução da experiência foi identificado o ciclo de funcionamento normal, (após inicialização), representado na Figura 9. A

inicialização determina a localização do "zero absoluto do movimento" de cada um dos graus de liberdade. Esta ação é efetuada a uma velocidade menor que a de funcionamento normal.

O controlador da experiência fica à espera de um pedido de execução. Neste pedido são fornecidas as variáveis da experiência ('b*ola'*, '*altura da rampa'* e '*ângulo de lançamento'*). Finalizado o ciclo funcional da máquina, o controlador da experiência envia os resultados obtidos para o servidor e o aparato fica pronto para a execução da experiência seguinte.

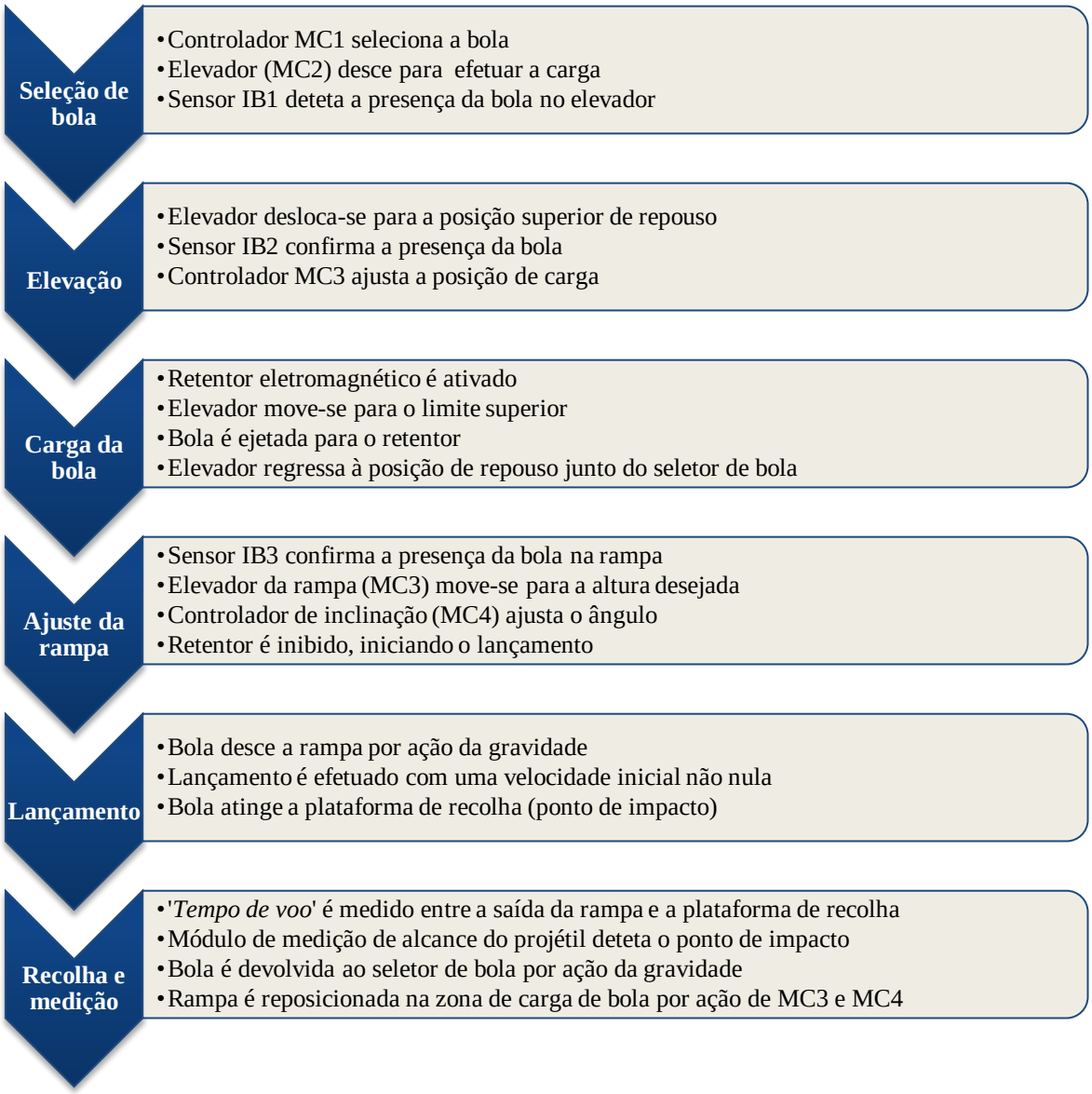


Figura 9 Ciclo de funcionamento após inicialização

3.2. AQUISIÇÃO DE MATERIAL

Este trabalho foi suportado na íntegra por fundos internos do ISEP, que é um organismo público, integrado no Instituto Politécnico do Porto (IPP), sob a alçada do Ministério da Educação e Ciência (MEC).

No decorrer do projeto, está em vigor um procedimento de aquisição que obriga à utilização de uma plataforma eletrónica, designada de VORTAL, que apresenta vários aspetos negativos quando está em causa a aquisição de pequenas quantidades de material específico. Não entrando em pormenores que escapam ao âmbito desta tese, refira-se apenas que houve um cuidado especial em procurar aquisições por via de ajuste direto, entre o ISEP e o fornecedor escolhido de forma a minimizar o tempo de espera. Ainda assim, estes processos de compra tendem a ser morosos na sua autorização, o que pode colocar em causa um projeto desta natureza. A Tabela 2 apresenta os fornecedores selecionados de forma a otimizar o processo de compra dos itens necessários.

Tabela 2 Tabela de fornecedores

igus	Igus™ – http://www.igus.com/default.asp Calhas flexíveis, chumaceiras lineares, veios de deslocamento, fusos trapezoidais e cablagem industrial.
Nanotec PLUG & DRIVE	Nanotec™ – http://en.nanotec.com/start.html Motores passo-a-passo, atuadores lineares, controladores dos motores passo-a-passo, codificadores e fontes de alimentação.
tecnogial	Tecnogial™ – http://www.tecnogial.pt Serviços de metalo-mecânica e fornecimento de elementos mecânicos (perfis de alumínio, parafusos, rodas dentadas etc.).
Digi-Key CORPORATION	Digikey™ – http://pt.digikey.com Componentes eletrónicos.
Beta LAYOUT	PCB-Pool™ – http://www.pcb-pool.com Circuitos impressos.
item	Item™ – http://www.item24.com/pt/ Perfis de alumínio e respetivos acessórios.

3.3. REQUISITOS FUNCIONAIS E DE IMPLEMENTAÇÃO

Nas subsecções seguintes são apresentados os requisitos mecânicos (3.3.1), os dispositivos de aquisição, atuação e de controlo (3.3.2), os requisitos de distribuição de energia (3.3.3), os mecanismos de comunicação de dados (3.3.4), o controlador de experiência e finalmente o servidor web local (3.3.5), permitindo assim entender em pormenor o conteúdo do diagrama de blocos (Figura 8).

3.3.1. REQUISITOS MECÂNICOS

O aparato está dividido nos seguintes subconjuntos, descritos nas seguintes subsecções por ordem de referência:

1. Seletor de bola;
2. Elevador da bola;
3. Elevador da rampa de lançamento;
4. Dispositivo de recolha;

3.3.1.1. SELETOR DE BOLA

Este subconjunto efetua a seleção de uma de três bolas existentes. As bolas são de tamanho pré-definido e configurado no programa de controlo.

O seletor de bola tem as seguintes especificações:

- Total de bolas: 3 unidades;
- Diâmetro máximo da bola: 19 mm;
- Massa máxima da bola: 30 g;
- O deslocamento máximo do carro seletor de bolas: 55 mm;

Para o ajuste inicial do seletor de bolas é necessário um sensor de limite do movimento.

3.3.1.2. ELEVADOR DA BOLA

Este subconjunto é responsável por carregar uma bola, pré-selecionada no seletor de bola, eleva-la até um ponto máximo e, quando o subconjunto "*Elevador da rampa*" estiver pronto, ejeta-la. Após a ejeção, o elevador deve retornar à posição inicial. O elevador da bola tem os seguintes requisitos de funcionamento:

- Carregar bola, disponibilizada pelo seletor de bola;
- Bloquear bola durante o percurso;
- Elevar bola a um ponto admissível pelo "*Elevador da rampa*";
- Descarregar bola, quando a rampa se encontrar na posição de carga;

As especificações são:

- Deslocação máxima estimada: 500 mm;
- Massa a deslocar: 30 g;

Para o controlo deste movimento, foram identificados os seguintes componentes ativos:

- Sensor de entrada para detetar a presença de bola;
- Sensor de limite para o ajuste inicial do elevador da bola;
- Sensor de saída para garantir a presença da bola no topo (detetar possível queda da bola durante o trajeto);
- Motor passo-a-passo para mover o elevador da bola;

3.3.1.3. *"ELEVADOR DA RAMPA DE LANÇAMENTO"*

A finalidade deste subconjunto é providenciar um sistema de lançamento, em que a altura e o ângulo de lançamento são configuráveis. Inicialmente o elevador deve mover-se para a posição de carga, e sinalizar ao elevador da bola que está pronto. No momento de carga, deve bloquear a bola de forma a poder configurar a altura e ângulo de lançamento. Posteriormente, o "*Elevador da rampa*" move-se para a altura definida e simultaneamente, a rampa é inclinada para o ângulo de lançamento. Após um breve intervalo de tempo, o sistema de bloqueio é inibido para que o lançamento seja efetuado. Finalmente, a rampa deve retornar à posição inicial.

Este subconjunto executa as seguintes tarefas:

- Carregar bola, disponibilizada pelo elevador da bola;
- Mover subconjunto da rampa para a altura desejada;
- Configurar ângulo de lançamento;
- Iniciar lançamento;
- Retornar à posição inicial;

As especificações para este módulo são:

- Deslocação máxima estimada: 300 mm;
- Variação angular da rampa: ± 15°;
- Rampa em madeira fornecida pela CIDEPE™ (utilizada em experiências manuais);

Para este controlo são necessários os seguintes componentes:

- Sensor de entrada para detetar a presença de bola na rampa;
- Sistema de bloqueio da bola;
- Motor passo-a-passo para deslocar o "*Elevador da rampa*";
- Sensor de limite para o ajuste inicial do "*Elevador da rampa*";
- Motor passo-a-passo para ajustar o ângulo de lançamento;
- Codificador (*Encoder*) para informação do ângulo e ajuste inicial;

3.3.1.4. DISPOSITIVO DE RECOLHA DA BOLA

A finalidade deste subconjunto é recolher o projétil e devolve-lo ao seletor de bolas. Este módulo funciona apenas pela ação da gravidade. Aqui, estão também incorporados os sistemas de medição do '*tempo de voo*' e do '*alcance horizontal do projétil*'.

Para a medição do '*tempo de voo*' são necessários dois sensores:

- Sensor de saída da rampa que informa o início do voo;
- Sensor do instante de impacto que informa o fim do voo;

Para a medição do '*alcance horizontal do projétil*' é necessário um sistema capaz de detetar uma bola num intervalo de tempo inferior a 1 ms e com uma resolução de 2 mm.

3.3.2. DISPOSITIVOS DE AQUISIÇÃO, ATUAÇÃO E CONTROLO

Para ser possível automatizar o processo de uma máquina são necessários dispositivos de aquisição (sensores), de atuação (atuadores) e de controlo.

3.3.2.1. SENSORES

Foram identificados os seguintes tipos de sensores necessários ao funcionamento do aparato:

- Sensor em forma de 'U', com emissor e receptor de infravermelhos, a ser utilizado na deteção do limite de movimento. Este sensor deve ter um tempo de resposta rápido, pelo que a sua leitura deve ser efetuada, se possível, de um modo direto;

- Sensor de presença, com emissor e receptor de infravermelhos, a ser utilizado na deteção da presença da bola, por meio de interrupção. Este sensor deve ter um alcance mínimo de 150 mm. A leitura destes sensores não é instantânea e devem ter a capacidade de incorporação numa rede RS-485;
- Codificador (*Encoder*) a ser instalado no sistema de configuração do ângulo de lançamento, com resolução inferior a 0,5°. Este sensor deve conter um indicador de posição inicial;
- Sistema para medição do '*alcance horizontal do projétil*' com tempo de reação inferior a 1 ms e uma resolução mínima de 2 mm, com capacidade de integração numa rede RS-485;

3.3.2.2. ATUADORES

Para a execução dos movimentos foram identificados os seguintes atuadores:

- Atuador linear com uma força maior ou igual a 2 N com deslocamento superior a 60 mm, a ser utilizado no seletor de bolas;
- Motor passo-a-passo com torque superior a 40 N.cm e com 200 passos por revolução;
- Sistema retentor eletromagnético para o bloqueio da bola durante a configuração da altura e do ângulo (obriga à utilização de esferas de material ferromagnético);
- Iluminação da máquina, controlável via rede RS-485, para possibilitar a sua utilização em qualquer horário;

3.3.2.3. CONTROLADORES

Foram identificadas as seguintes necessidades de controlo:

- Controlador de motor passo-a-passo com capacidade de programação de micro passos e passível de ser integrado numa rede RS-485. Este controlador deverá suportar a intensidade de corrente requerida pelo respetivo motor;
- Sistema com microprocessador para monitorizar os sensores não ligados via rede RS-485 e exercitar as saídas que sejam necessárias. Este sistema deve ter capacidade de comunicação via rede RS-485;
- Sistema com microprocessador para monitorizar e controlar os recursos externos de potência de forma a desligar remotamente a alimentação de potência, bem como a iluminação. Este controlador deverá estar ligado numa rede RS-485 e simultaneamente providenciar uma interface USB (*Universal Serial Bus*) de comunicação com o controlador primário;

3.3.3. DISTRIBUIÇÃO DE ENERGIA

Para otimizar a distribuição de energia no aparato, será necessário reduzir o número de fontes primárias de energia. Assim será desejável que só existam as seguintes alimentações:

- 230 V_{AC} – Distribuição de energia primária para todos os sistemas com fonte de alimentação própria bem como para o retentor eletromagnético da bola;
- Tensão 24 V_{DC} – Alimentação dos motores utilizados. Para suportar os sistemas adicionais de controlo (+5 V ou +3,3 V) utilizam-se fontes comutadas de baixa potência, integradas nos respetivos circuitos. Desta forma, sensores e controladores devem ser desenhados para tensões de entrada iguais à tensão necessária aos motores, e serem imunes ao ruído de alimentação provocado pelas cargas indutivas (e.g. eletroíman e motores passo-a-passo). A alimentação dos motores deve ser comutável por ação do programa de controlo;

3.3.4. MECANISMOS DE COMUNICAÇÃO DE DADOS

Este aparato tem necessidade de estabelecer comunicação de dados a vários níveis, desde o nível de controlo (ambiente sujeito à influência do ruído provocado pelos motores), pela ligação ao controlador de experiência e finalizando pela ligação da experiência à internet.

A conceção deste aparato é baseada na utilização de um sistema de controlo distribuído. De forma a tornar a comunicação interna mais imune ao ruído, foi escolhida a norma RS-485 [18] em modo *full-duplex* como meio físico de interligação geral. Esta rede terá um fluxo de dados a 115200 baud. A informação a transportar é ASCII com um protocolo de controlo semelhante à especificação da Nanotec™ [19] (fornecedor dos controladores dos motores passo-a-passo), para que os sistemas a serem projetados possam ser interligados juntamente com este tipo de controladores na mesma rede. Para acesso do controlador da experiência, é utilizado um conversor que interliga a rede interna RS-485 ao porto USB do controlador. De forma a ligar à internet (via ETHERNET), o controlador de experiência, o servidor web local e a câmara IP é necessário um *ROUTER*. O controlador de experiência é uma unidade com um sistema operativo (Windows™, Linux™ ou RTOS) com ecrã tátil, com um conector de acesso à rede ETHERNET e capacidade de chamar serviços web (*Web Services*) [Anexo D].

3.3.5. SERVIDOR WEB LOCAL

Para o sistema ser transportável e funcionar de uma forma autónoma na ausência de ligação ao servidor central de experiências, é necessário um computador pessoal, com capacidade de instalar um servidor PHP, com capacidade de difundir o áudio e vídeo capturados pela câmara IP.

Adicionalmente, terá instalado um servidor de dados MySQL. Esta configuração será uma cópia local do servidor central de experiências.

3.4. CONCLUSÃO

O projeto de um aparato capaz de realizar remotamente a experiência de "*Lançamento de projéteis*" é um trabalho multidisciplinar com requisitos logísticos, mecânicos, eletrónicos e de controlo de âmbito alargado. Este capítulo descreveu uma arquitetura para a solução do problema em estudo. A escolha de fornecedores condicionou o projeto a executar numa instituição pública, e dessa forma houve um especial cuidado na sua seleção. Após identificação das funcionalidades necessárias, foram especificados os requisitos de cada área a implementar (mecânica, eletrónica e de *software*). A implementação da solução será descrita nos Capítulo 4 – Projeto Mecânico, Capítulo 5 – Projeto Elétrico e Eletrónico, e Capítulo 6 – Projeto Lógico.

Capítulo 4

PROJETO MECÂNICO

Este capítulo apresenta o projeto mecânico do aparato, realizado com a ferramenta CAD (*Computer-Aided Design*) 3D (*Three Dimensional*) desenvolvida pela AUTODESK™ denominada INVENTOR™. A sua escolha foi consequência da disponibilidade de licenças para estudantes desta aplicação.

O projeto tem como base de desenvolvimento a utilização dos componentes comercializados pela Nanotec™, para atuadores, motores, controladores de motores e fontes de alimentação, as calhas de deslizamento veios e fusos da IGUS™, e os perfis técnicos de alumínio da ITEM™, dada a sua excelente relação qualidade/preço.

A implementação mecânica desta experiência, apresentada na Figura 10, foi desenhada para que os vários subconjuntos constituintes fossem independentes entre si e facilmente desmontáveis. Esta divisão também permite que um subconjunto possa ser ajustado e reparado sem ser necessário que esteja instalado na sua localização de funcionamento no aparato. No entanto, o "*Seletor de bola*" partilha a base do "*Elevador de bola*", pelo que estes dois subconjuntos formam um subconjunto maior, na prática indivisível.

Outra linha mestra deste projeto foi a necessidade de produzir uma máquina fiável pelo que se impõe a utilização de componentes de qualidade e subconjuntos de baixa ou média complexidade, com um conjunto mínimo de ajustes para que o seu desempenho possa ser determinístico.

Figura 10 Aspeto geral do aparato desenvolvido

Nas secções seguintes estão detalhadas a implementação mecânica de cada subconjunto, por ordem de referência:

- Seletor de bola;
- Elevador de bola;
- Elevador da rampa e rampa de lançamento;
- Dispositivo de recolha do projétil;
- Estrutura externa;

4.1. SELETOR DE BOLA

Pretende-se com este subconjunto efetuar a seleção de uma de três bolas, com diâmetro máximo de 19 mm.

4.1.1. TRANSMISSÃO E ATUADOR

Após o estudo para determinar o tipo de transmissão a utilizar, em função da carga (aproximadamente 30 g), tipo de movimento (linear) e distância a percorrer, foram inicialmente equacionados dois métodos possíveis:

- Transmissão por fuso (esferas ou trapezoidal);
- Transmissão por atuador linear;

Na Tabela 3 é apresentado um estudo comparativo entre a transmissão por fuso (Figura 11a) e a utilização de um atuador linear (Figura 11b).

Tabela 3 Tabela comparativa entre transmissão por fuso versus atuador linear

Característica	*Transmissão por fuso*	*Atuador linear*
Carga a deslocar	Suporta cargas de elevado valor	Normalmente pequenas cargas
Distância a percorrer	Limitada pelo tamanho do fuso	Depende do tipo de atuador
Complexidade	Média (requer um motor, um fuso, apoios para o fuso, acessório de bloqueio, etc.)	Baixa (só necessita do apoio do atuador e da fixação ao carro)

Funcionalmente, no caso de uma transmissão por fuso, existe um fuso rotativo que impõe uma deslocação do carro por ação numa fêmea. Esta pode ser do tipo trapezoidal ou de esferas.

Figura 11 Tipos de transmissão linear [19]

No caso do atuador linear, o fuso não roda e o movimento linear é imposto pelo movimento rotativo aplicado à fêmea que compõe o rotor do motor. Dadas as características máximas do deslocamento (55 mm) e de massa da bola (30 g) foi escolhido o atuador linear da Nanotec™ LP3575S.

Com base no atuador selecionado e requisitos especificados (3.3.1.1), foi desenhado o seletor de bola apresentado na Figura 12. Para maximizar a rigidez global do conjunto, a base em alumínio serve simultaneamente de apoio ao "*Seletor de bola*" e ao "*Elevador de bola*".

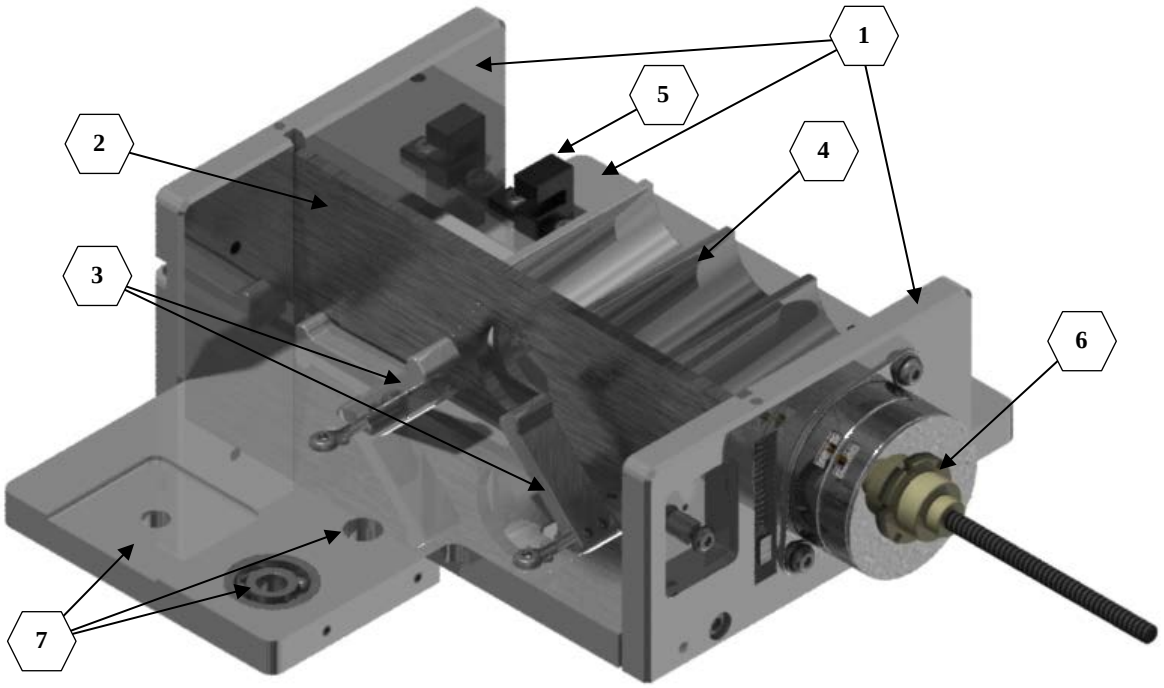


Figura 12 Vista do seletor de bola

A Tabela 4 lista os componentes mecânicos principais do seletor de bola.

Tabela 4 Lista de componentes mecânicos do seletor de bola

Item	*Referência*	*Descrição*
1	ISEP-LF001-01-01-00 (1,3,5) -01	Estrutura base de suporte
2	ISEP-LF001-01-01-002-01	Bloqueador
3	ISEP-LF001-01-01-0 (12,14) -01	Alavancas do bloqueador
4	ISEP-LF001-01-01-007-01	Carro do seletor
5	OPB840W	Sensor de limite
6	LP3575S-0504-TR3.5x1.22	Atuador linear
7	---	Pontos de fixação do elevador

4.1.2. MODO DE FUNCIONAMENTO

O seletor de bola (Figura 12) utiliza o atuador linear (6) para mover o carro do seletor (4), e dessa forma selecionar a bola pretendida. Para bloquear a queda da bola durante a seleção, o bloqueador (2) deve permanecer na posição de repouso (inferior). O controlo deste componente é efetuado por ação do "*Elevador de bola*" sobre as alavancas (3) do bloqueador. A bola selecionada é a que se

encontra no eixo central do seletor, em frente do orifício existente no bloqueador (2). No momento de carga da bola para o elevador, o bloqueador (2) sobe, e deixa passar a bola pelo orifício por ação da gravidade. Após a carga, o "*Elevador de bola*" inicia o seu trajeto de ascensão e dessa forma bloqueia a saída do seletor. O controlador do atuador linear (6) utiliza o sensor OPB840W (5) para detetar o limite do movimento, e a partir desta posição determinar a localização de cada um dos canais existentes no carro do seletor (4).

4.1.3. ESTRUTURA BASE DE SUPORTE

Na Figura 12, a estrutura base de suporte (1) é constituída pelos componentes base, suporte lateral 1 e suporte lateral 2. Os suportes são apertados na base, e utilizam cavilhas para eliminar a necessidade de alinhamentos adicionais na montagem. Este conjunto, fabricado em alumínio, forma uma estrutura rígida que é a base de suporte aos elementos móveis que compõem o seletor de bola. Os elementos laterais contêm um rasgo que serve de calha de deslizamento do bloqueador (2). O suporte lateral contém a furação necessária à instalação do atuador linear (6) e a caixa retangular de 20 x 25 mm destinada à inserção do sensor de barreira por infravermelhos. Do lado oposto está prevista a instalação do díodo emissor da barreira de raios infravermelhos. Esta base é comum ao "*Elevador de bola*", pelo que é visível o encastramento para o perfil de estrutura do elevador, a caixa do rolamento do eixo e a furação para o veio (7).

4.1.4. CARRO DO SELETOR DE BOLA

A Figura 13 apresenta uma vista do subconjunto que é deslocado por ação do veio do atuador linear. O carro (1) está solidário com o patim da série IGUS DryLin-NS-27 que se move na calha de deslizamento (2). Este patim suporta uma carga máxima de 5 N (eixo *x*) e 2,5 N (eixos *y* e *z*). As condutas do carro seletor são iguais e têm uma inclinação de 5° para permitir que a bola se desloque por ação da gravidade.

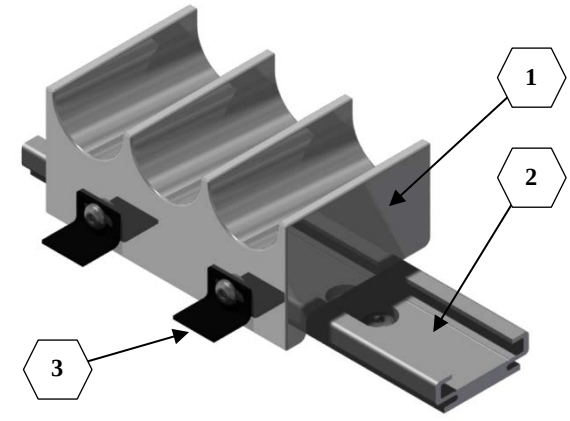


Figura 13 Carro do seletor de bola

Na zona lateral foi colocada uma lâmina (3) para atuar o sensor OPB840W [Figura 12, subconjunto (5)] utilizado para sinalizar o limite do movimento. O aperto deste conjunto à base de alumínio deve ser efetuado de forma a garantir o alinhamento com o eixo do veio. Uma forma de ajuste consiste em colocar o carro encostado a um dos lados e apertar o parafuso do extremo oposto, garantindo assim a perpendicularidade entre a calha do carro e o suporte lateral.

4.1.5. ALAVANCAS ELEVADORAS DO BLOQUEADOR

As alavancas apresentadas na Figura 14 têm uma montagem simétrica e formam uma alavanca que permite o acionamento do bloqueador por ação do "*Elevador de bola*". As alavancas são iguais e estão apertadas no eixo comum. Uma das alavancas está alinhada com o bloqueador e a outra com o extremo do bloco móvel do elevador. A existência de dois conjuntos simétricos é justificada pela necessidade de manter o bloqueador na horizontal.

Figura 14 Alavanca elevadora

Este conjunto requer o seguinte ajuste:

1. Apertar os braços do lado do bloqueador ao eixo;
2. Colocar o "*Elevador de bola*" no seu limite inferior;
3. Selecionar a segunda bola (conduta do meio);
4. Nessa posição apertar os braços do lado do elevador, para que o furo existente no bloqueador fique verticalmente alinhado com a conduta do meio e verificar que o bloqueador está horizontal;

4.2. ELEVADOR DE BOLA

Este conjunto é responsável pelas seguintes ações:

- Abrir a comporta do seletor de bola, e dessa forma carregar a bola selecionada;
- Elevar a bola até à zona de carga do "*Elevador da rampa*";
- Ejetar a bola para ser carregada pelo "*Elevador da rampa*";

A Figura 15 apresenta uma vista geral do "*Elevador de bola*" e seus componentes.

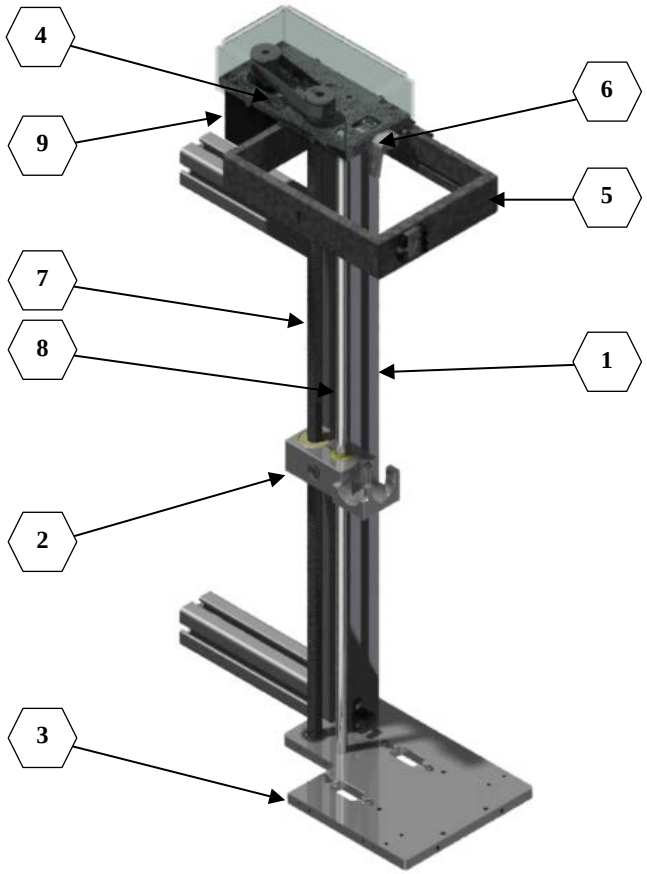


Figura 15 Vista geral do "*Elevador de bola*"

A Tabela 5 lista os componentes do "*Elevador de bola*".

Tabela 5 Lista de componentes do "*Elevador de bola*"

Item	*Referência*	*Descrição*
1	ISEP-LF001-01-00-004-01	Apoio central do elevador
2	ISEP-LF001-01-01-010-01	Base móvel do elevador
3	ISEP-LF001-01-01-001-01	Suporte inferior
4	ISEP-LF001-01-02-001-01	Suporte superior
5	ISEP-LF001-01-02-008-01	Suporte dos sensores de bola
6	ISEP-LF001-01-02-002-01	Ejetor da bola
7	ISEP-LF001-01-00-003-01	Fuso trapezoidal IGUS™ PTGSG-10x3
8	ISEP-LF001-01-00-002-01	Veio IGUS™ AWMP-08
9	PD2-O4118L1804	Motor Nanotec™ com controlador SMCI12

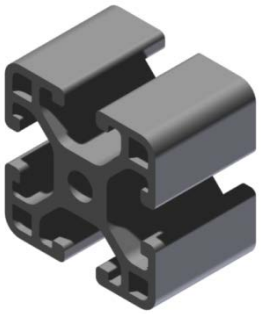
O elevador inclui os seguintes componentes:

- Base do elevador (3), comum com o seletor de bola e descrita na subsecção 4.1.3;
- Estrutura central do elevador, constituída pelo apoio central (1), o fuso (7) e o veio (8);
- Base móvel (2);
- Conjunto superior (4) que agrega a tração (9), o ejetor de bola (6) e o suporte dos sensores de bola (5);

4.2.1. ESTRUTURA CENTRAL DO "*ELEVADOR DE BOLA*"

O elevador tem como estrutura central de apoio, o perfil ITEM™ 30x30 que, de uma forma geral, foi utilizado como componente estrutural de toda a máquina. A Tabela 6 apresenta as características principais deste perfil.

Tabela 6 Caraterísticas do perfil ITEM™ 30x30

	Característica	*Valor*
	Área	3,43 cm^2
	Dimensões externas da secção	30 x 30 mm
	Massa linear	0,93 $kg.m^{-1}$
	Momento de inércia de área	2,90 cm^4
	Largura do rasgo	6 mm

O perfil, a base inferior e a base superior formam o suporte do "*Elevador de bola*". Nestas bases está a furação onde se fixam o eixo e os rolamentos do fuso (SKF626-2Z). Nas bases existe um encastramento que juntamente com as cavilhas em aço proporcionam um processo de montagem sem necessidade de alinhamento.

4.2.2. BASE MÓVEL

A Figura 16 mostra a base móvel fabricada em alumínio (1), destinada a transportar a bola desde o seletor até à rampa de lançamento. O casquilho linear (2) é o IGUS™ RJM-01-08 de 8 mm de diâmetro. A porca trapezoidal (3) é da IGUS™, modelo WSRM-2215TR10x3, para fusos trapezoidais de 10 mm de diâmetro com 3 mm de passo. A zona de recepção da bola tem uma abertura frontal para admissão da bola, uma abertura lateral para a saída da bola para a rampa, e do lado oposto a esta, existe uma passagem para o ejetor da bola. A base está ligeiramente rebaixada para impedir a saída da bola durante o trajeto. Na zona lateral está localizada a lâmina (4) para intercetar o sensor de limite do movimento.

A zona central do maciço tem uma abertura que juntamente com o parafuso (5) permite ajustar a folga entre o casquilho linear (2) e o respetivo veio.

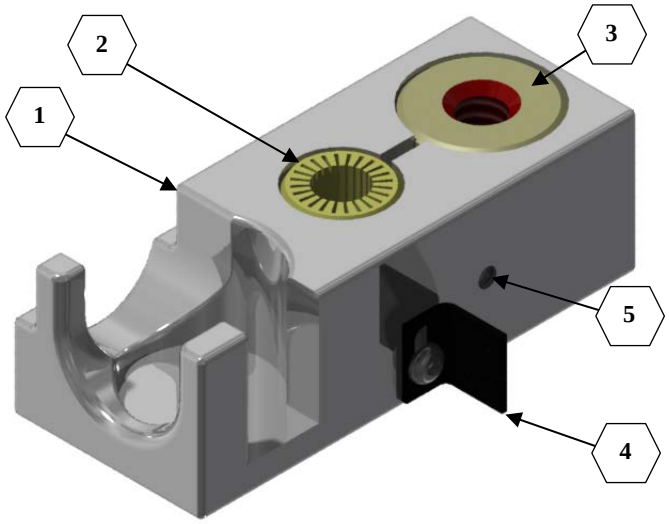


Figura 16 Base móvel do "*Elevador de bola*"

4.2.3. SUPORTE SUPERIOR DO ELEVADOR (TRAÇÃO E EJETOR DE BOLA)

O suporte superior do elevador, apresentado na Figura 17, é composto pela base (1) fabricada em alumínio, ejetor de bola (2), suporte dos sensores da bola (3) e do sistema de tração (4).

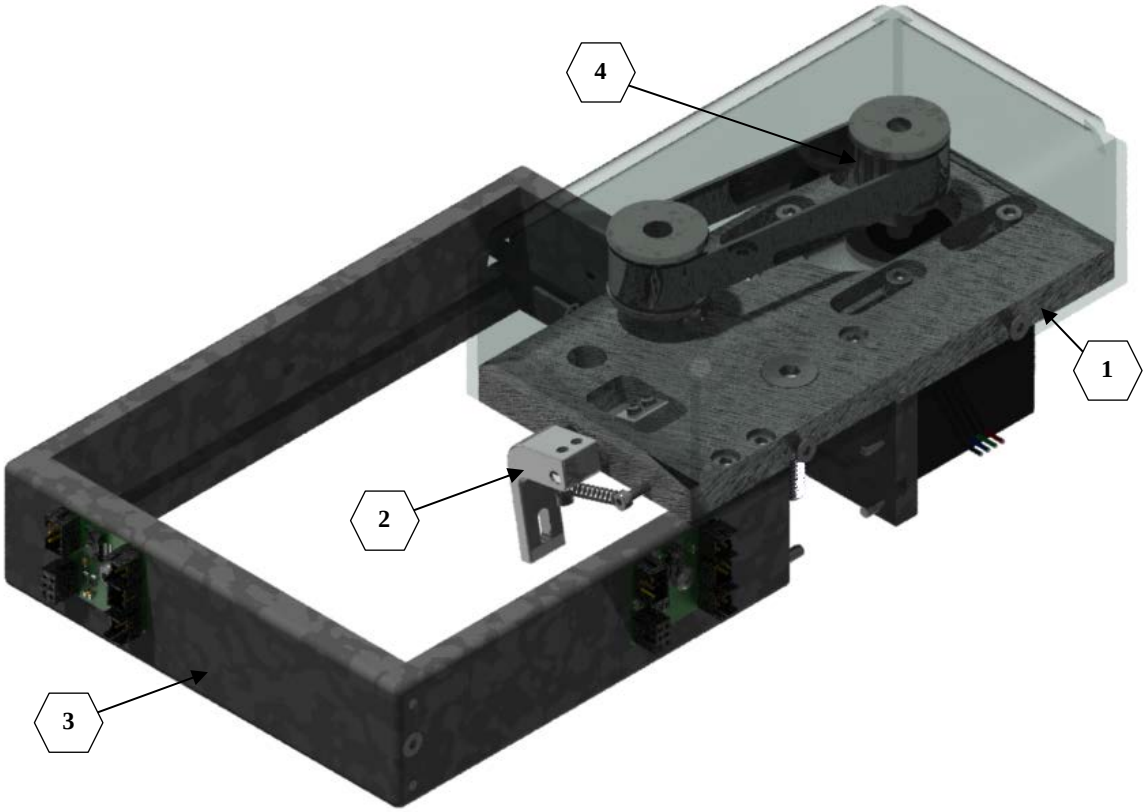


Figura 17 Parte superior do elevador (tração, sensores e ejetor da bola)

A base superior (1) aperta no perfil do apoio central do elevador e tem 4 pernos em aço para fazer o seu alinhamento no perfil. Esta base tem todas as furações necessárias para fixar os restantes componentes. O ejetor (2) apresentado na Figura 18 é uma alavanca constituída por dois braços com um eixo comum (2.3), em que o braço mais pequeno (2.1) é acionado pela superfície superior do bloco da base móvel e o maior (2.2) faz deslocar a bola para fora do elevador em direção à rampa de lançamento. O orifício oval da alavanca da bola (2.2) destina-se a deixar passar os raios infravermelhos do sensor de deteção de presença de bola no topo do elevador.

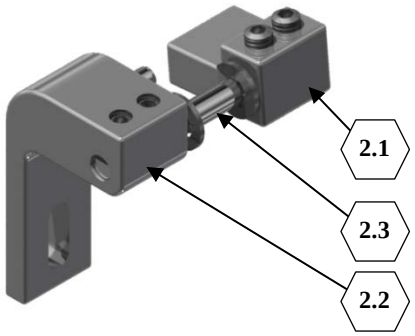


Figura 18 Ejetor da bola

O suporte dos sensores da bola (3) é composto por quatro peças apertadas entre si, nas quais os sensores de barreira por infravermelhos e respetivos emissores são instalados. Este suporte é todo cavilhado e apertado pela parte de baixo da base superior do elevador. A tração do elevador (4) está apresentada em pormenor na Figura 19.

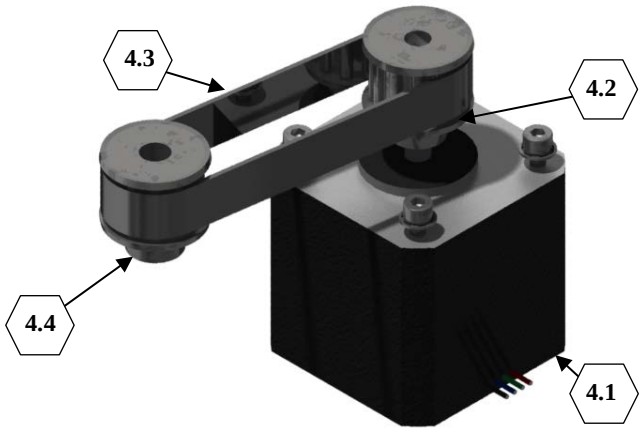


Figura 19 Tração do "*Elevador de bola*"

O elemento de tração (4.1) é da Nanotec™ da série PD2-O4118L1804, que é uma integração do motor ST4118L1804 com o controlador de motores passo-a-passo SMCI12, apresentado em maior detalhe no Capítulo 5. O motor tem acoplado no veio, uma polia de 12 dentes (4.2), que por intermédio da correia síncrona T5 de 10 mm (4.3), faz tração à polia de 12 dentes (4.4) acoplada ao fuso do elevador. Deste modo, a base móvel do "*Elevador de bola*" tem uma deslocação de 3 mm

por cada rotação do motor. A fixação do motor é efetuada por 4 parafusos M3 cujas cabeças se deslocam em rasgos existentes na base superior. Estes rasgos permitem o ajuste da tensão da correia síncrona (4.3). A zona de tração, no suporte superior, tem uma cobertura em acrílico para proteção contra possíveis danos físicos causados pela máquina quando em funcionamento, e.g. durante operações de manutenção.

4.3. ELEVADOR DA RAMPA DE LANÇAMENTO

O "*Elevador da rampa de lançamento*" foi projetado para efetuar as seguintes funções:

- Receber a bola proveniente do elevador e bloqueá-la via retentor eletromagnético;
- Mover a rampa de lançamento para o ângulo de lançamento e posição vertical definida pelo utilizador;
- Executar o lançamento do projétil inibindo o bloqueio da bola;
- A Figura 20 apresenta uma vista geral deste conjunto;

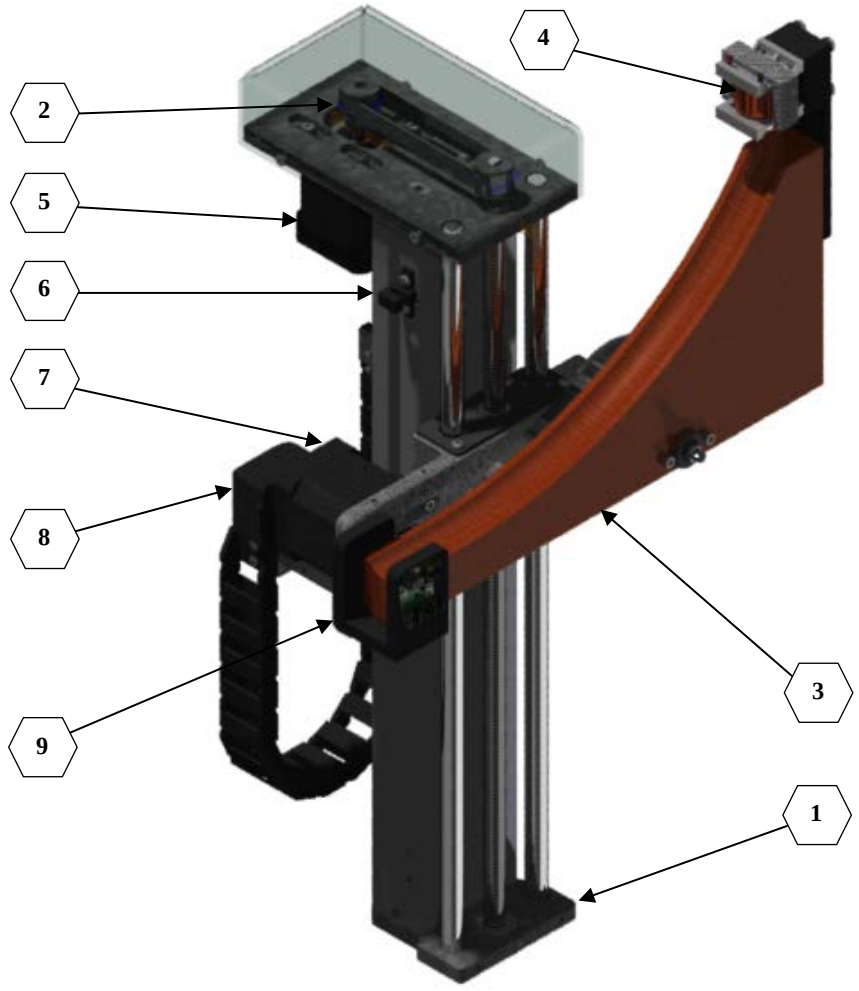


Figura 20 Vista geral do "*Elevador da rampa de lançamento*"

A Tabela 7 lista os componentes mecânicos principais do seletor de bola.

Tabela 7 Lista de componentes do "*Elevador da rampa de lançamento*"

Item	*Referência*	*Descrição*
1	ISEP-LF001-02-00-00 (2,3,5) -01	Estrutura base de suporte
2	ISEP-LF001-02-01-001-01	Subconjunto de tração vertical
3	ISEP-LF001-02-03-001-01	Subconjunto da rampa
4	ISEP-LF001-02-03-010-01	Retentor eletromagnético
5	PD2-O4118L1804	Motor de ajuste vertical da rampa
6	OPB840W	Sensor de limite
7	ST4118L1804-B	Motor de ajuste do ângulo da rampa
8	WEDS5541-B14	Codificador do ângulo da rampa
9	ISEP-LF001-98-01-001-01	Sensor de início de voo

O elevador da rampa inclui os seguintes componentes:

- Estrutura central do elevador (1), constituída pelo apoio central, base, fuso e veios;
- Conjunto de tração vertical (2);
- Conjunto móvel da rampa de lançamento (3), constituído pelo apoio central, tração, rampa e retentor;

4.3.1. ESTRUTURA CENTRAL DO "*ELEVADOR DA RAMPA DE LANÇAMENTO*"

Para o projeto da estrutura central do "*Elevador da rampa de lançamento*", foi utilizada a aplicação web "*DryLin Expert 2.0*"[3] da IGUS™ para a sua gama de produtos DryLin. Para tal foi estimada uma massa aproximada de 1,3 kg para o bloco a deslocar composto pela rampa em madeira, motor passo-a-passo, roldanas e estruturas de apoio. Primeiramente foi selecionado o sistema "DryLin R", métrico, com casquilho de polímero.

No passo seguinte, decidiu-se utilizar dois veios em alumínio endurecido com tolerância h8[4] espaçados 50 mm, com dois casquilhos por veio encostados entre si. A massa é deslocada no eixo vertical, com centro de massa em (28,2 mm; 16,5 mm; 24,3 mm) nos eixos (x; y; z) respetivamente relativo à origem do sistema em estudo. O acionamento é equidistante dos eixos e no mesmo plano.

[3] O acesso web a esta aplicação é efetuado por http://www.igus.com/drylinexpert/default.aspx

[4] As tabelas de tolerâncias ISSO estão disponíveis em http://www.engineersedge.com/iso_tolerance.htm

Os parâmetros funcionais estimados foram os seguintes:

- Carga (*Fs*): 15 N;
- Aceleração: 2 $m.s^{-2}$;
- Distância a percorrer (*vida*): 10000 km;
- Comprimento dos veios: 380 mm;

A aplicação gerou os resultados apresentados na Tabela 8.

Tabela 8 Resultados da aplicação IGUS "DryLin Expert 2.0"

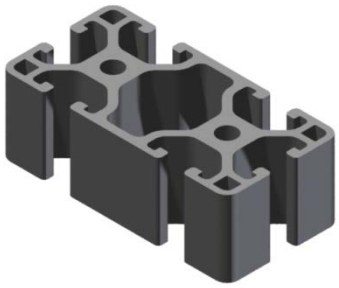
Característica	*Valor*
Características de funcionamento	Válidas
Desgaste em y e *z*	0,4 mm
Velocidade máxima de longa duração	3,6 $m.s^{-1}$
Força de acionamento necessária	25 N
Temperatura máxima admissível	90 °C
Fator de segurança (y; *z*)	(70,66; 59,78)
Folga no centro de massa	0,2 mm

Desta forma foram identificados os elementos necessários ao sistema:

- 4 Chumaceiras lineares IGUS™ RJM-01-10;
- 2 Veios em alumínio endurecido IGUS™ AWMP-10;
- 1 Fuso trapezoidal IGUS™ PTGSG-10x3-R;
- 1 Porca trapezoidal IGUS™ WSRM-2215TR-10x3;

Para suportar este conjunto foi decidido utilizar o perfil ITEM™ 60x30 como estrutura central de apoio, cujas características principais são apresentadas na Tabela 9.

Tabela 9 Caraterísticas do perfil ITEM™ 60x30



Característica	*Valor*
Área	6,13 cm^2
Dimensões externas da secção	60 x 30 mm
Massa linear	1,65 $kg.m^{-1}$
Momento de inércia de área	3,02 cm^4
Largura do rasgo	6 mm

A Figura 21 mostra a estrutura central do elevador da rampa onde esta se desloca. Este elevador satisfaz os requisitos especificados anteriormente e não tem ajustes durante a montagem. Os suportes, inferior (1) e superior (2) apertam de topo no perfil (3) e têm um encastramento que juntamente com as cavilhas em aço eliminam a necessidade de ajustes na montagem.

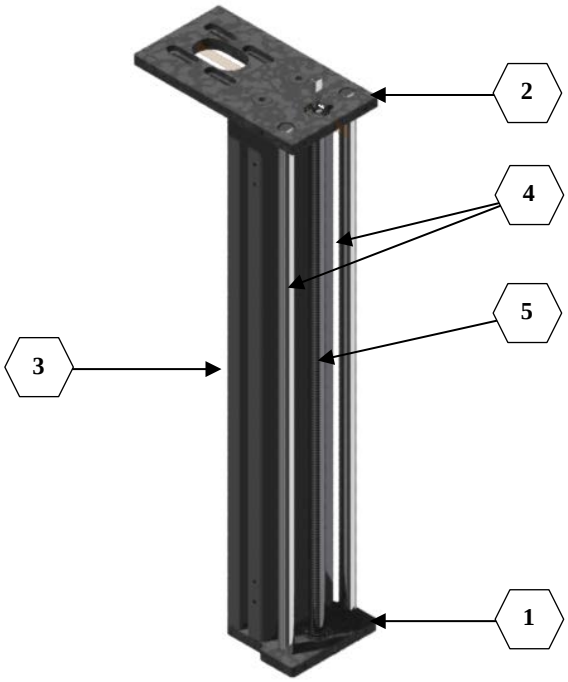


Figura 21 Estrutura central da rampa

Os eixos (4) e os rolamentos SKF626-2Z do fuso (5) são apertados nos suportes (1 e 2). O dimensionamento do fuso foi feito de forma a não ter folga vertical depois de montado. Este elevador tem uma altura entre suportes de 364 mm.

4.3.2. TRAÇÃO VERTICAL DA RAMPA

A Figura 22 apresenta uma vista da tração vertical da rampa de lançamento.

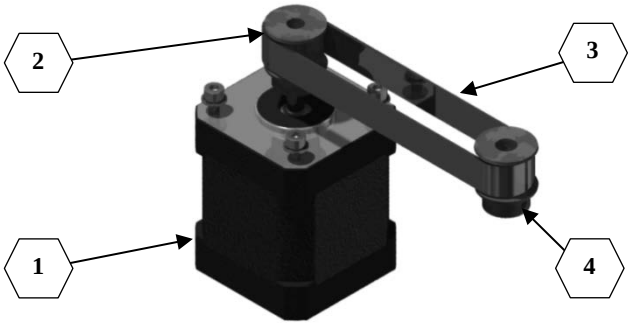


Figura 22 Tração vertical do "*Elevador da rampa de lançamento*"

A tração do elevador da rampa mostra o elemento de tração (1) da Nanotec™ da série PD2-O4118L1804, que é uma integração do motor ST4118L1804 com o controlador de motores passo-

a-passo SMCI12. O motor tem acoplado no veio a polia de 12 dentes (2) que por intermédio da correia síncrona T5 de 10 mm (3), faz tração à polia de 12 dentes (4) acoplada ao fuso do elevador. Deste modo o "*Elevador da rampa de lançamento*" tem uma deslocação de 3 mm por cada rotação do motor. A fixação do motor é efetuada por 4 parafusos M3 cujas cabeças se deslocam em rasgos existentes no suporte superior. Estes rasgos permitem o ajuste da tensão da correia síncrona (3).

A zona de tração no suporte superior tem uma cobertura em acrílico para proteção contra possíveis danos físicos causados pela máquina quando em funcionamento, e.g. durante operações de manutenção.

4.3.3. CONJUNTO MÓVEL DA RAMPA DE LANÇAMENTO

O conjunto móvel da rampa está representado na Figura 23.

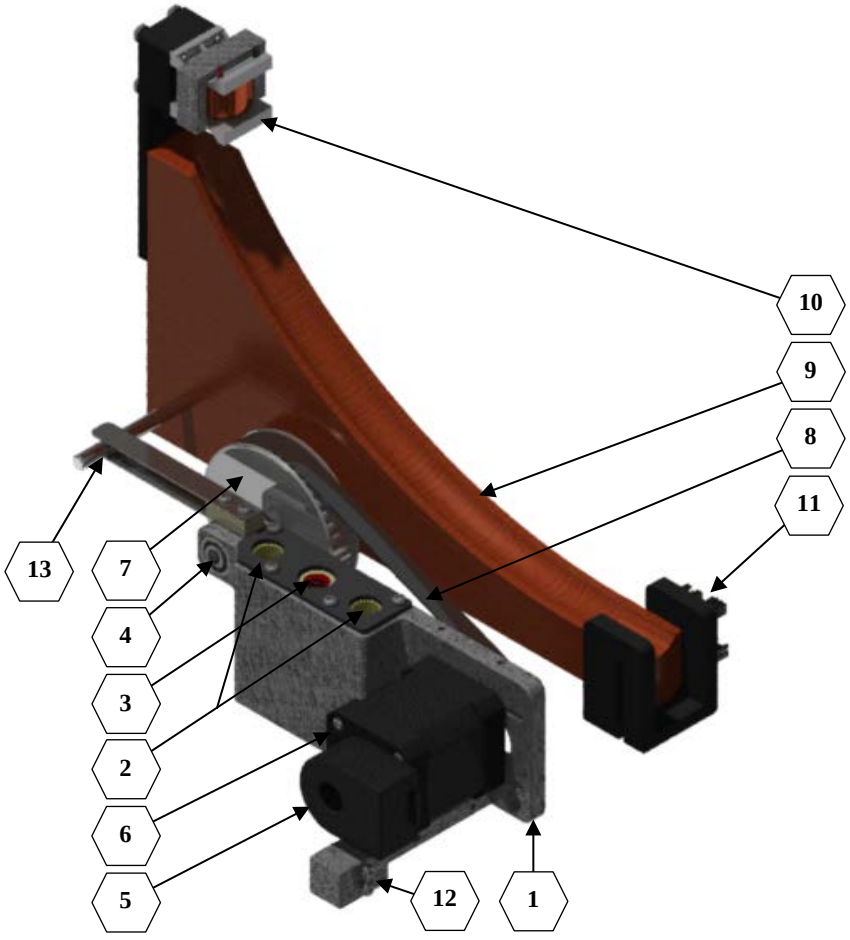


Figura 23 Conjunto móvel da rampa de lançamento

O suporte deste conjunto (1) desloca-se sobre os veios da estrutura central e utiliza quatro chumaceiras lineares (2) para reduzir o atrito. A tração é efetuada pelo fuso trapezoidal de 10 mm de diâmetro com passo de 3 mm sobre a porca trapezoidal (3). Estes 5 elementos estão em canais existentes no suporte (1) que por intermédio de um rasgo permite ajustar a folga das chumaceiras.

Este suporte serve de base de apoio à tração angular (6), ao eixo da rampa de lançamento acoplado ao rolamento SKF 625-2Z (4) e ao suporte de apoio da calha flexível (12) utilizada para a passagem dos cabos elétricos. A Tabela 10 apresenta a lista de material deste conjunto.

Tabela 10 Lista de componentes do conjunto móvel da rampa de lançamento

Item	*Referência*	*Descrição*
1	ISEP-LF001-02-03-002-01	Suporte da rampa
2	IGUS™ RJM-01-10	Chumaceiras lineares de 10 mm diâmetro
3	IGUS™ WSRM-2215TR-10x3	Porca para fuso trapezoidal 10x3 mm
4	SKF™ 625-2Z	Rolamento de esferas 5x5x16 mm
5	Nanotec™ WEDS5541-B14	Codificador do ângulo da rampa
6	Nanotec™ ST4118L1804-B	Motor de ajuste do ângulo da rampa
7	ROLISA™ 16T5-40	Polia 40 dentes T5 – 10 mm
8	ROLISA™ 10T5x365	Correia de sincronização T5 com 365 dentes
9	CIDEPE™	Rampa de lançamento em madeira
10	ISEP-LF001-02-03-010-01	Retentor eletromagnético
11	ISEP-LF001-98-01-001-01	Sensor de início de voo
12	ISEP-LF001-02-03-012-01	Suporte de apoio da calha flexível
13	ISEP-LF001-02-03-01 (3,4,5) -01	Sistema de repouso da rampa

A tração angular é efetuada por ação do motor Nanotec™ ST4118L1804-B (6) de duplo eixo, com uma polia solidária ao eixo de 10 dentes T5, na correia de sincronização T5 de 10 mm (8), que transmite o movimento à polia de 40 dentes T5 (7) solidária com a rampa CIDEPE™ (9), e estabelece uma relação entre a rotação do motor e da rampa de lançamento de 4:1 para aumentar mecanicamente a resolução de controlo do ângulo. A posição de repouso, quando o motor de tração angular não está energizado, é garantida pelo veio acoplado à rampa e molas de repouso solidárias com o suporte da rampa (13).

Para informação da posição angular foi incluído um codificador relativo Nanotec™ WEDS5541-B14 (5) de 1000 pulsos por revolução com sensor de origem acoplado na zona traseira do motor de tração angular.

A rampa CIDEPE™ (9) roda sobre o eixo apoiado nos rolamentos (4) e tem no topo o retentor eletromagnético que está preso a um apoio que permite o seu ajuste vertical. Na saída está instalado um sensor (11) para detetar a passagem da bola que determina o início de voo.

4.4. DISPOSITIVO DE RECOLHA DO PROJÉTIL

A missão do dispositivo de recolha é aparar o projétil lançado e retorna-lo por ação da gravidade ao seletor de bolas. Este dispositivo está representado na Figura 24.

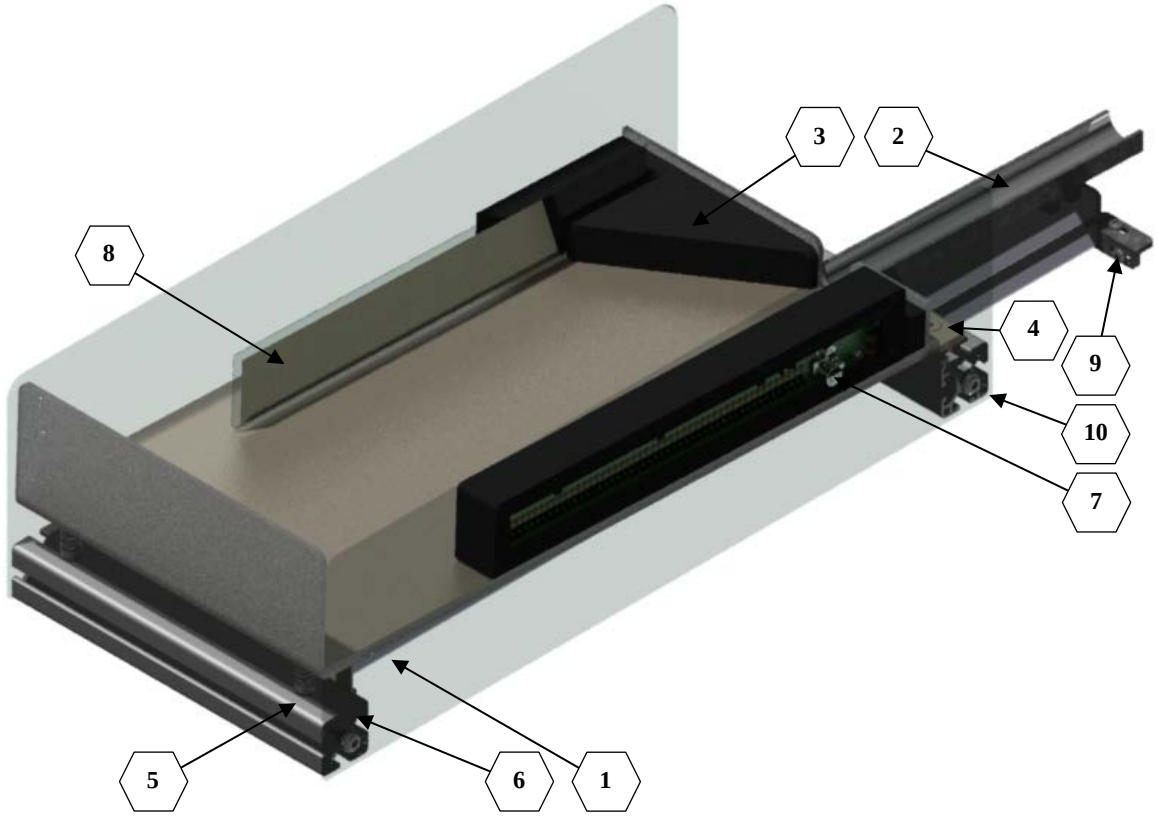


Figura 24 Dispositivo de recolha do projétil

A base de recolha (1) delimita uma área retangular que recebe o impacto do projétil. Esta base é feita em plástico e revestida por um tapete em borracha para a sua proteção. A base tem uma inclinação de 2° relativamente ao plano horizontal. Para garantir a recolha do projétil foram colocadas rampas (3) que encaminham o projétil para a entrada da conduta (2), e assim devolve-lo ao seletor de bola qualquer que seja a sua posição na base de recolha.

A base está fixa à estrutura por intermédio de duas dobradiças (4), e no extremo oposto está apoiada sobre duas molas (5). As razões para a existência destas molas são por um lado absorver a energia cinética gerada pelo impacto do projétil, e por outro lado, aproveitar a vibração causada pelo seu impacto para determinar o instante de impacto através do sensor (6) e dessa forma medir o '*tempo de voo*'.

Para a medição do '*alcance horizontal do projétil*' foi instalado o dispositivo eletrónico (7) que é uma barreira de raios infravermelhos, com emissão no dispositivo que depois de refletidos no espelho (8) são recebidos pelos sensores de infravermelhos. Este conjunto está solidário com o

seletor de bola por intermédio do acessório (9) e por acoplamento do perfil (10) ao perfil de apoio da estrutura central da rampa (Figura 21).

4.5. ESTRUTURA EXTERNA DO APARATO

Um dos requisitos do aparato (secção 1.2) é a sua fácil mobilidade. Com essa finalidade, foi desenhada uma estrutura robusta capaz de conter todos os itens necessários e passível de ser transportada. Esta estrutura é feita com o perfil ITEM™ 30x30 e tem, como dimensões externas, 1160 mm de largura, 560 mm de profundidade e 860 mm de altura. Na zona inferior foi incluído um compartimento onde está instalado o computador do servidor web local e os componentes de rede, a distribuição de energia elétrica e a fonte de alimentação primária da máquina (24 V_{DC}).

A Figura 25 apresenta uma vista global da estrutura. O volume da zona de trabalho é vedado por paineis de plástico acrílico transparente, para permitir a visualização da experiência a partir do exterior. Na zona frontal foram colocadas duas portas (1), para permitir o acesso ao aparato para operações de limpeza, manutenção e reparação. Foram colocadas fechaduras (2) em todas as portas para permitir que o aparato funcione em zonas públicas.

Figura 25 Estrutura externa do aparato

A zona inferior é vedada por paineis acrílicos pretos e opacos com uma porta longitudinal (3) reforçada. No painel lateral esquerdo desta zona, foi instalado uma tomada de entrada de potência (4) com interruptor e filtro. Foi instalada também uma ventoínha de grande fluxo de ar (5) e respectivo filtro, para permitir a extração do ar, com admissão a partir de uma abertura existente no painel inferior oposto.

A placa mãe do computador está acessível (6) para ser possível ligar o aparato à rede ETHERNET e, numa situação de manutenção, ligar um monitor, um rato e um teclado.

No topo da caixa foi incluida uma fonte luminosa controlada pelo controlador de experiência para permitir a utilização do aparato em qualquer condição de luminosidade.

4.6. CONCLUSÃO

Este capítulo apresentou o projeto mecânico do aparato, que seguiu uma conceção dividida em subconjuntos independentes entre si, e que permitem realizar o trabalho proposto. Assim, o aparato foi estruturalmente dividido nos conjuntos "*Seletor de bola*", "*Elevador de bola*", "*Elevador da rampa de lançamento*", e "*Dispositivo de recolha*". O capítulo terminou com o detalhe da estrutura externa do aparato, necessário à sua portabilidade. O próximo capítulo apresenta o projeto elétrico e eletrónico que visa interligar os sensores e atuadores aqui enumerados, para poderem ser controlados pelo projeto lógico apresentado no Capítulo 6.

Capítulo 5

PROJETO ELÉTRICO E ELETRÓNICO

No Capítulo 4, o aparato foi estruturalmente dividido nos conjuntos "*Seletor de bola*", "*Elevador de bola*", "*Elevador da rampa de lançamento*", e "*Dispositivo de recolha*". Em cada um deles, o projeto enumerou os atuadores e sensores necessários ao seu funcionamento. A primeira secção deste capítulo apresenta o diagrama geral da implementação, com ênfase na sua divisão em circuitos de controlo modulares, interligados por uma rede RS-485. Nas secções seguintes, são detalhadas as implementações eletrónicas de cada um dos módulos, apresentando o seu diagrama de blocos, uma descrição do seu funcionamento e terminando com a implementação física. O capítulo termina com o projeto elétrico que interliga todas as partes constituintes.

5.1. DIAGRAMA GERAL DA IMPLEMENTAÇÃO

A implementação elétrica e eletrónica foi dividida em dois conjuntos distintos ("*Seletor e elevador de bola*" e "*Elevador da rampa e sistema de recolha*"), cujo diagrama geral é apresentado na Figura 26. Tendo em conta os custos associados ao projeto e à produção de circuitos impressos (PCB), utilizou-se uma metodologia de implementação modular, com um sistema de processamento distribuído e interligado por uma rede RS-485. Os PCB's (*Printed Circuit Board*) utilizados, retângulos verdes, são:

- SMCI12 – Controlador de motor passo-a-passo da Nanotec™;

- Sensor de bola – Unidade destinada a detetar a presença de uma bola;
- Extensor SMCI12 – Sistema de expansão para suprir algumas limitações do controlador SMCI12 e facilitar a cablagem da máquina;
- Caixa de distribuição – Placa para a comutação e distribuição de energia bem como interligação dos vários componentes;
- Fonte de alimentação de +24 V;

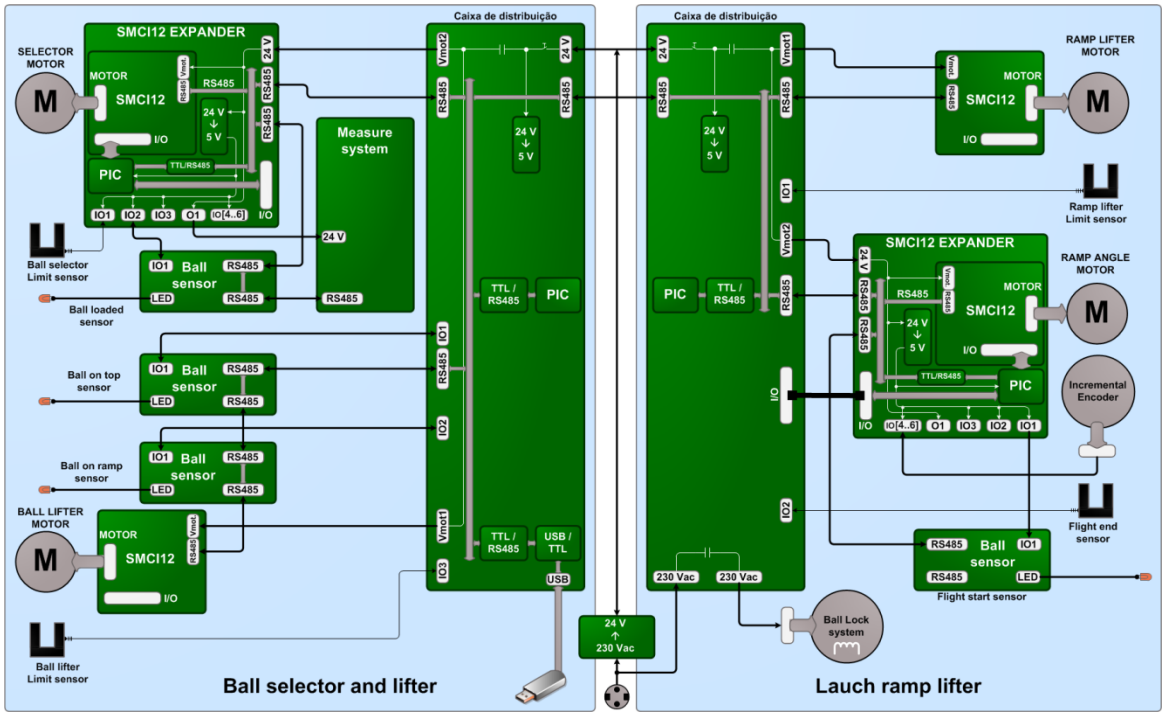


Figura 26 Diagrama geral da implementação elétrica e eletrónica

A fonte de alimentação de +24V alimenta a placa de controlo de cada uma das caixas de distribuição. A tensão de +24 V alimenta uma fonte comutada local de +5 V que fornece energia à placa de controlo e aos sensores externos ligados aos conectores IO[1…4]. A tensão de +24V passa num relé de estado sólido (SSR, "*Solid State Relay*") e é disponibilizada nos conectores VMOT[1…2] para alimentar os controladores dos motores.

As caixas de distribuição têm PCB's iguais, mas com configurações diferentes. Na "*Caixa de distribuição*" do "*Elevador de bola*", está instalado um conversor USB → RS-232 que faz interface direto à rede por intermédio de um transcetor RS-485. A "*Caixa de distribuição*" do elevador da rampa disponibiliza um relé de estado sólido (SSR) de 230 V_{AC} para controlo do retentor eletromagnético. Estas placas têm um microcontrolador PIC que interage com os seus periféricos. O PIC também está ligado via porta série à rede RS-485. A placa de distribuição disponibiliza vários nós de acesso à rede RS-485.

Os "*Sensores de bola*" são alimentados a +5V a partir das caixas de distribuição e utilizam um sistema de emissão/recepção por infravermelhos para a deteção por interrupção do feixe. A sinalização é passível de ser inquirida a partir da rede RS-485 ou diretamente da leitura do sinal lógico de saída.

O "*Sistema de expansão SMCI12*" destina-se a facilitar a cablagem necessária à interligação dos vários componentes junto do motor e suprir a necessidade de leitura do codificador ótico incremental que não é suportada pelo controlador SMCI12. Este PCB foi desenhado para acoplar diretamente no controlador SMCI12 e assim eliminar a necessidade de ligações entre os dois sistemas. Os esquemas completos estão em anexo [Anexo E].

5.2. REDE INTERNA

Todo o projeto de controlo foi pensado de forma a utilizar um sistema de processamento distribuído com base numa de uma rede interna RS-485 [18], que interliga todos os sistemas de controlo utilizados.

A Figura 27 apresenta o diagrama geral da rede RS-485 implementada no aparato, que interliga os vários circuitos impressos desenvolvidos ("*Caixa de distribuição*", "*Sensor de bola*", "*Sistema de expansão SMCI*" e "*Sistema de medição*"), destacando-se as ligações e respetivos conectores.

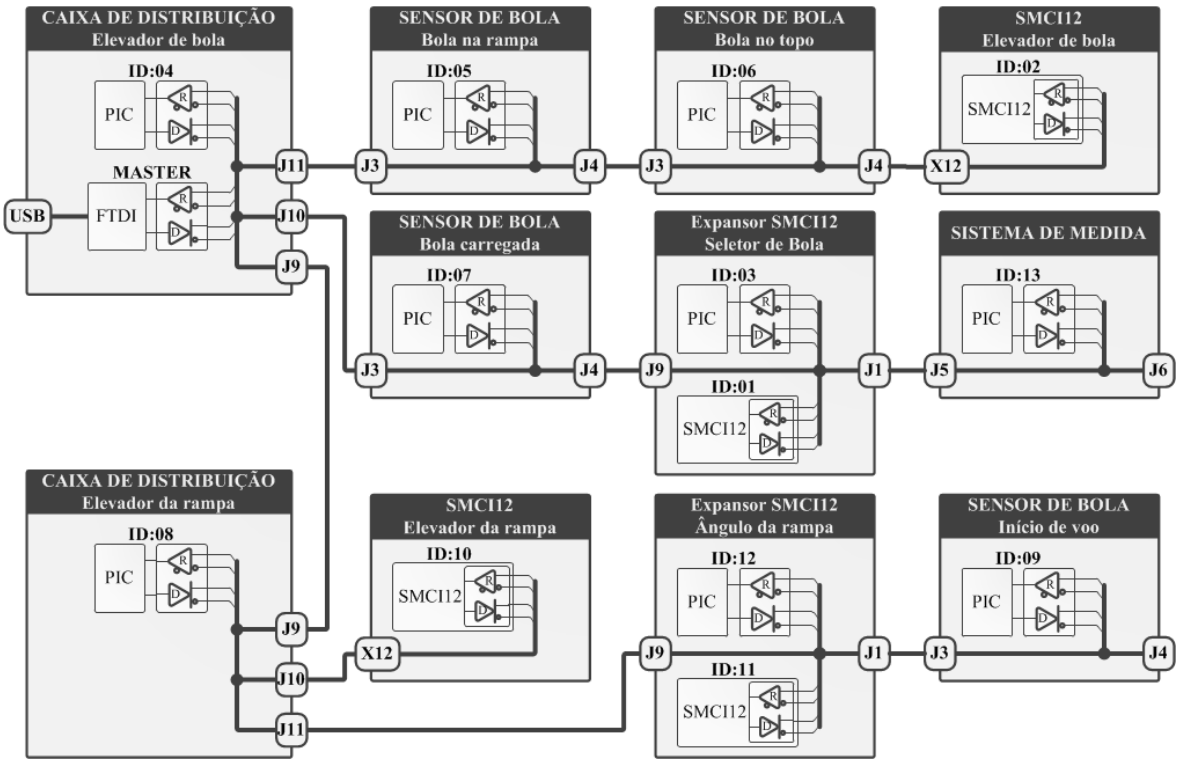


Figura 27 Identificação dos nós da rede interna do aparato

A rede tem uma topologia a quatro fios (MULTIDROP) com um nó mestre (*MASTER*) localizado na "*Caixa de distribuição*" do "*Elevador de bola*" que contém o interface USB/RS-485, e todos os restantes nós escravos (ID[01…13]). Um escravo só pode comunicar com o mestre. Se pretender comunicar com outro escravo, deve pedir ao mestre para reenviar o seu pedido. Este funcionamento é implementado ao nível do protocolo da rede.

5.3. SENSOR DE BOLA

Este sensor destina-se a detetar a presença de uma bola na zona de interseção do feixe infravermelho emitido pelo díodo e recebido pelo módulo receptor de infravermelhos. Nas subsecções seguintes são apresentados o diagrama de blocos, uma descrição do modo de funcionamento e a sua implementação, respetivamente.

5.3.1. DIAGRAMA DE BLOCOS

A Figura 28 apresenta o diagrama de blocos do "*Sensor de bola*". Este sensor utiliza o módulo receptor de infravermelhos TSOP36238 para desmodular um sinal (IRTX) enviado pelo LED (*Light Emitting Diode*) D1.

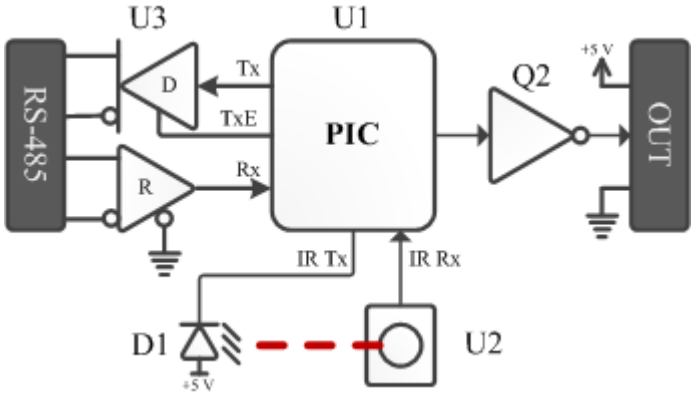


Figura 28 Diagrama de blocos do "*Sensor de bola*"

O microcontrolador PIC (U1) faz a modulação do LED D1 de acordo com os requisitos enumerados na folha de características do receptor e monitoriza a sua saída (IRRX), cujo estado é refletido na saída (Q2). Dessa forma é garantido que o sinal de saída é válido enquanto o feixe de raios infravermelhos não for interrompido. Quando a recepção de infravermelhos é interrompida, o receptor U2 comuta a saída e o PIC reflete o seu estado na saída digital Q2. Para a leitura do sensor remotamente foi incluído o transcetor RS-485 (U3) e disponibilizados dois conectores de acesso ao barramento RS-485.

5.3.2. IMPLEMENTAÇÃO

O interface entre as linhas série RS485TX e RS485RX do microcontrolador é implementado pelo transcetor SN65HVD3080E (U3) [Anexo B – II]. O sinal RS485TXE de permissão do *driver* de saída é colocado a '1' sempre que o microcontrolador pretenda aceder à rede. O circuito prevê a instalação das resistências terminadoras de 120 Ω (R2 e R6) a serem incluídas se necessário. Para facilitar a introdução na rede, foram disponibilizados os conectores J3 e J4 (Figura 29).

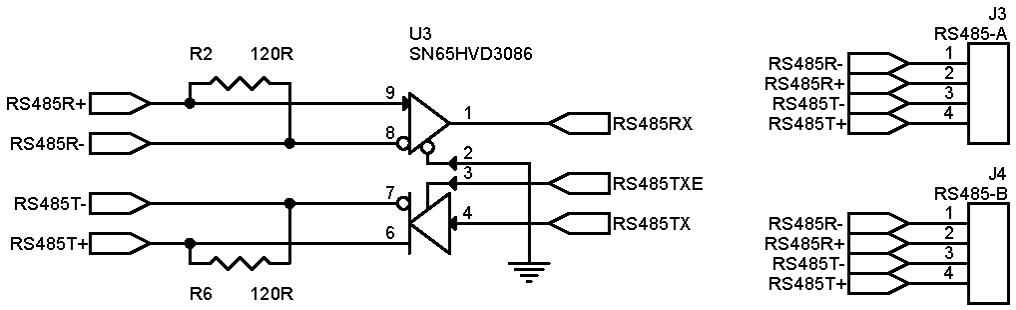


Figura 29 *"Sensor de bola"* – Esquema do interface RS232/RS485

O interface da barreira de raios infravermelhos é apresentado na Figura 30. O microcontrolador PIC faz a modulação do LED infravermelho TSML1020 através do MOSFET (*Metal Oxide Semiconductor Field Effect Transistor*) 2N7002CT com uma corrente I_F = 30 mA limitada pela resistência R1. A montagem prevê que o LED possa ser montado na placa (D1) para funcionar em modo de reflexão, ou com o LED ligado externamente via conector J1, para funcionamento por interrupção do feixe de raios infravermelhos. O circuito integrado U2 [Anexo B – III] recebe o sinal ótico, faz a sua desmodulação e disponibiliza o resultado no pino 4 pelo sinal IR-RX.

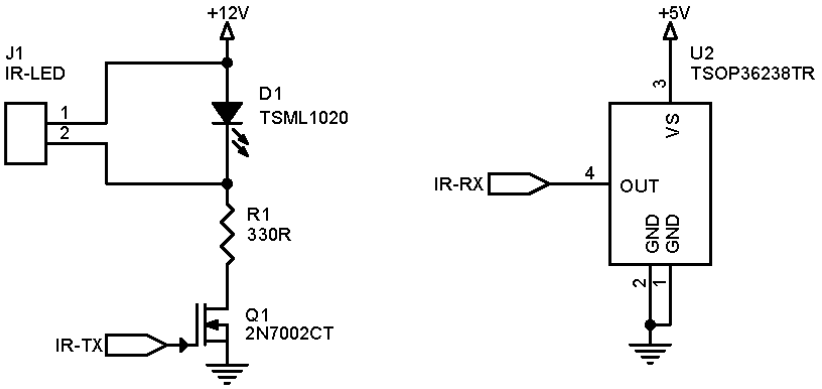


Figura 30 *"Sensor de bola"* – Esquema do interface da barreira de raios infravermelhos

Para gerir o "*Sensor de bola*" foi utilizado o microcontrolador da Microchip™ PIC16F1825-I/ST (Figura 31), que se caracteriza por ter 14 pinos, trabalhar a uma frequência de 32 MHz a partir do oscilador interno, disponibilizar 16 KB de memória FLASH para o programa monitor e 1 KB para memória de dados e disponibilizar um porto série.

Este circuito inclui o conector J5 para permitir a programação local do PIC. O LED bicolor (D2) LNJ107W5PRW providencia dois sinais de informação visual. Para proteger o PIC do exterior foi implementado o inversor formado pelo MOSFET de canal N (Q2) e a resistência R5.

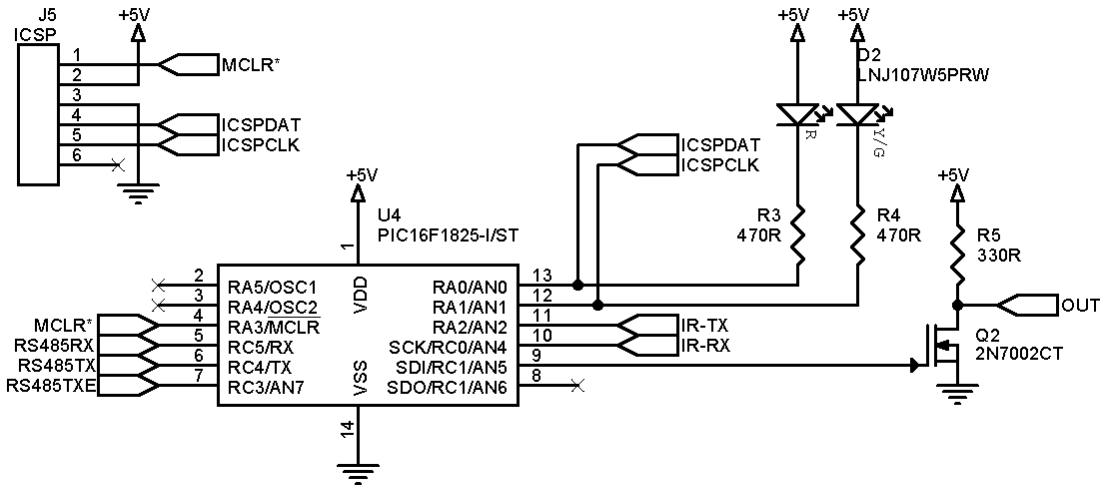


Figura 31 "*Sensor de bola*" – Esquema do microcontrolador

A Figura 32 mostra a fonte de alimentação do "*Sensor de bola*". O sensor está previsto para funcionar a partir de uma fonte externa entre 5,5 V e 45 V. Com essa finalidade foi incorporado o regulador de tensão LDO (*Low Drop*) TLE4264. Para ligar o sensor a +5 V é necessário que o CI U1 não seja instalado e que seja instalada uma ponte entre os pinos 1 e 3 do regulador LDO. Este modo foi implementado em todos os sensores instalados na máquina. Foram colocados os condensadores de filtragem de entrada (C1 e C2) e os condensadores de desacoplamento (C3 a C5). O conector J2 é a entrada para a fonte de alimentação e disponibiliza o sinal digital de saída do sensor.

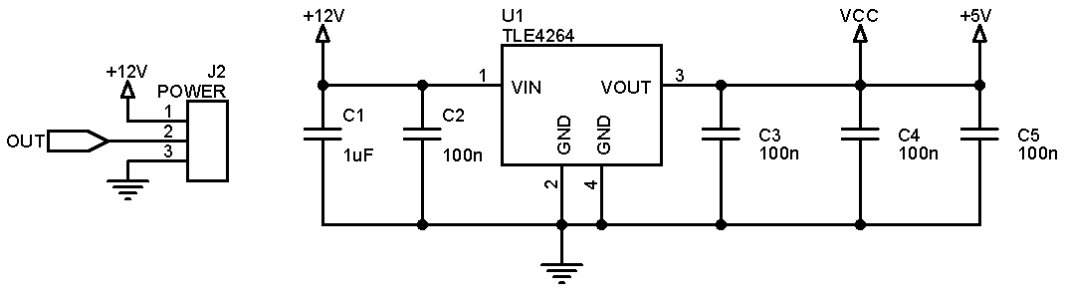


Figura 32 "*Sensor de bola*" – Esquema da fonte de alimentação

A implementação física do sensor teve como premissa ser um módulo de dimensões reduzidas para ser facilmente incorporado onde necessário. No projeto do PCB foi possível, pela utilização de tecnologia de montagem de superfície (SMT) e com recurso a conectores de dimensões reduzidas (Série HIROSE™ DF3 e DF11 com passo 2 mm), obter um sensor com 20x25 mm.

A Figura 33 apresenta os desenhos das camadas de cobre (à esquerda), respetivas implantações de componentes (ao centro) e com a montagem final à direita. A camada superior é apresentada no conjunto superior de imagens, e a camada do lado oposto no conjunto inferior. O regulador LDO (U1) não foi instalado nesta implementação dado o sensor estar alimentado a +5V.

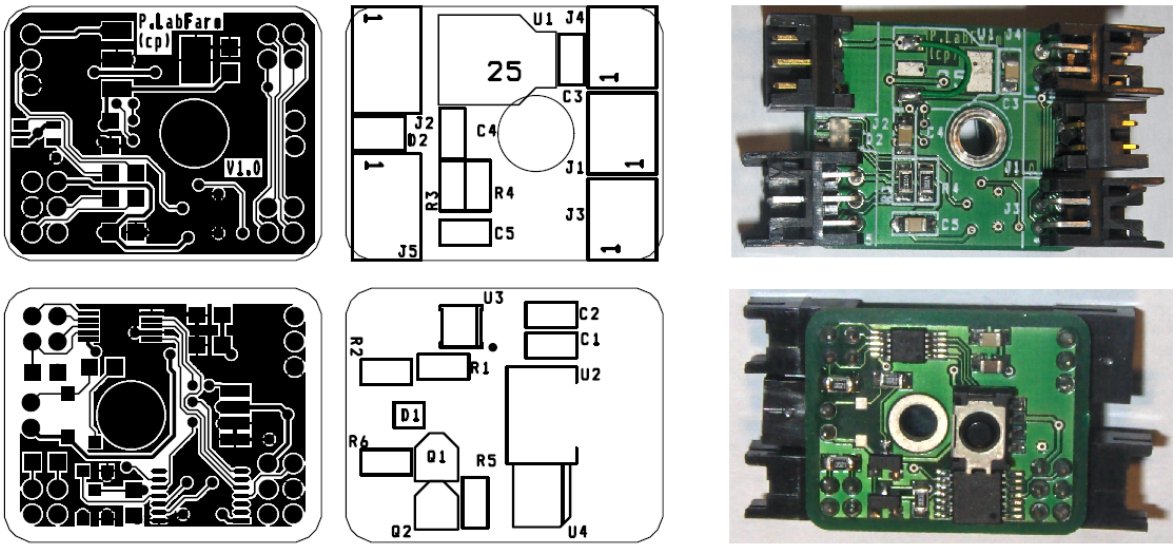


Figura 33 *"Sensor de bola"* – Circuito impresso e montagem final

5.4. CAIXA DE DISTRIBUIÇÃO

Para interligar os vários sensores, atuadores, redes (USB e RS-485) e distribuição de energia foi projetado um circuito que agrega o somatório das funcionalidades necessárias a cada grupo mecânico ("*Seletor e Elevador de bola*" / "*Elevador da Rampa e Recolha*"). Desta forma, as caixas de distribuição têm um PCB igual, mas com implementações diferenciadas, em função do grupo mecânico onde estão instaladas. Nas subsecções seguintes são apresentados o diagrama de blocos, uma descrição do modo de funcionamento e a sua implementação, respetivamente.

5.4.1. DIAGRAMA DE BLOCOS

A Figura 34 apresenta o diagrama de blocos da "*Caixa de* distribuição", que segue uma filosofia de utilização genérica. É uma placa de entradas (IN[1…4) e saídas (AC1 e ED[1…3]) ligados por *jumpers* ao controlador PIC (U5), que está ligado a uma rede RS-485 via transcetor U6. Adicionalmente foi incorporado um conversor USB→RS-485 (U1 e U2). A placa utiliza uma fonte comutada para gerar a tensão de +5 V a partir dos +24 V. Tem incorporado um relé de estado sólido para controlo da tensão dos motores. Disponibiliza dois conectores de expansão lógicos.

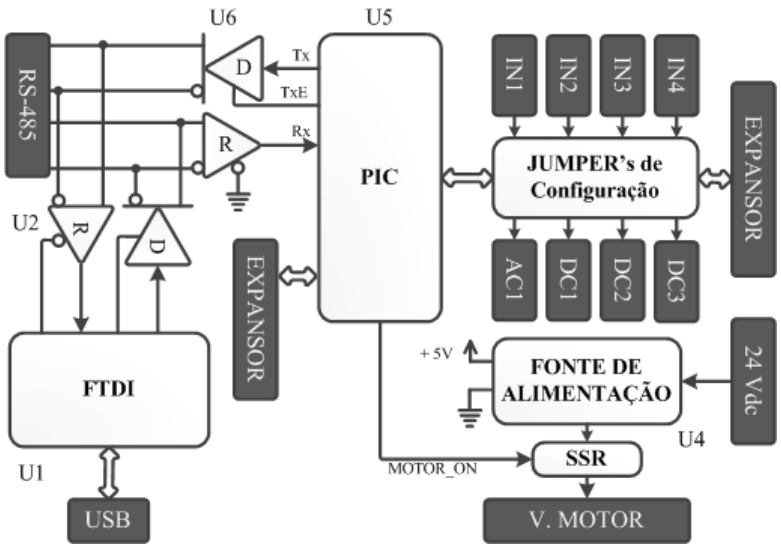


Figura 34 Diagrama de blocos da "*Caixa de distribuição*"

5.4.2. MODO DE FUNCIONAMENTO

A "*Caixa de distribuição*" disponibiliza um *hardware* de utilização genérica com acesso via rede RS-485 e providencia um meio de interligação de sensores, atuadores e alimentações à máquina de forma a minimizar a cablagem necessária e torná-la ponto-a-ponto.

5.4.2.1. CONVERSOR USB→RS-485

Incorporou-se um conversor USB→RS-485 na placa, para ser possível a ligação direta da rede RS-485 (com nós de ligação na placa) ao controlador do aparato. Este conversor tem por base o CI FT232RL da FTDI™ cujo diagrama de blocos é apresentado na secção IV do anexo B.

O circuito implementado utiliza o controlador FT232RL com o transcetor SN65HVD3086 [Anexo B – II], para disponibilizar um interface USB→RS-485 que funciona sem interferência do microcontrolador.

5.4.2.2. ENTRADAS E SAÍDAS GENÉRICAS

As entradas e saídas genéricas passam por um conjunto de pontes (*jumpers*) que permitem configurar a rota dos pinos do porto RA[0…7] do microcontrolador PIC para os conectores de entrada e saída (via SSR) ou para o conector de expansão digital. As saídas disponibilizam o controlo de uma tensão até 240 V_{AC} (RL2 do tipo CPC1976) e o controlo de três saídas até 60 V_{DC} de 700 mA (RL[3…5] do tipo CPC1002).

Para o controlo de cargas AC foi escolhido o relé de estado sólido da CLARE™ CPC1976. Este SSR é um relé AC que utiliza dois tirístores de potência na saída e inclui um detetor de passagem

por zero no seu controlo, fornecendo uma capacidade de bloqueio a tensões de 600 V_P. A utilização do detetor de passagem por zero assegura um baixo ruído na comutação de cargas AC ao minimizar os transientes gerados. A entrada está isolada opticamente e fornece um isolamento de 3750 V_{RMS} entre o controlo e a carga. A Figura 35 mostra o diagrama deste relé de estado sólido.

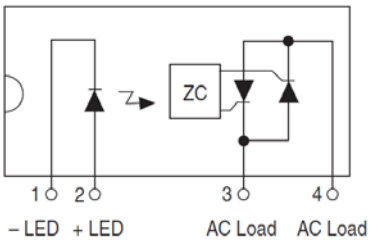


Figura 35 Diagrama do relé de estado sólido AC CPC1976 (cortesia CLARE™)

Este relé tem capacidade de controlar cargas até 2 A_{rms}, e necessita de uma corrente máxima no LED de 5 mA (I_F). A queda de tensão máxima em condução é 1,15 V_{rms}.

Para o controlo de cargas DC foi selecionado o relé de estado sólido CPC1002 da CLARE™. Este SSR é um relé DC de configuração "normalmente aberto" que utiliza um MOSFET acoplado opticamente e fornece um isolamento entre o controlo e a saída de 1500 V_{rms}. A Figura 36 mostra o diagrama de ligação e as curvas de resposta de comutação. O tempo de comutação t_{ON} é 5 ms e o tempo de comutação t_{OFF} é 2 ms.

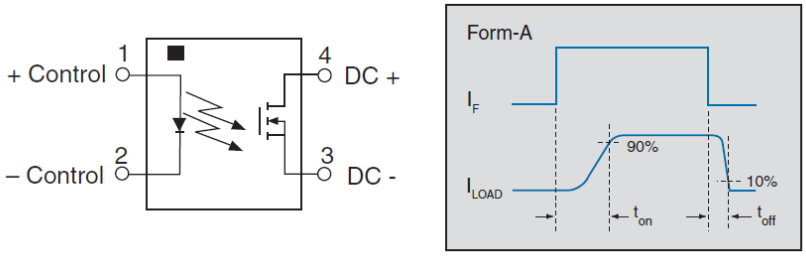


Figura 36 Diagrama do relé CPC1002, e resposta de comutação (cortesia CLARE™)

O LED de entrada tem uma queda de tensão máxima (V_F) de 1,4 V com uma corrente direta (I_F) de 5 mA. A corrente máxima da carga é 700 mA. A resistência máxima do MOSFET no estado ligado (R_{ON}) é 0,55 Ω.

5.4.2.3. FONTE DE ALIMENTAÇÃO

Para fornecer a tensão de funcionamento ao circuito (+5 V), foi utilizada uma fonte de alimentação comutada que utiliza o CI LM2574M-5.0 da *National Semiconductor*™. Este CI é um circuito integrado monolítico que disponibiliza todas as funções necessárias a um regulador comutado do tipo "*step down*" (BUCK). A Figura 37 mostra o diagrama de blocos do regulador comutado.

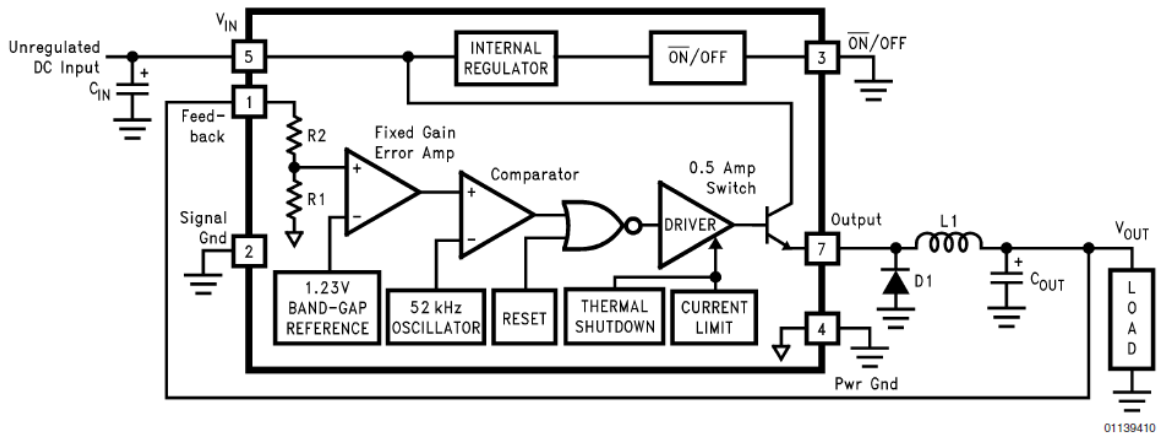


Figura 37 Diagrama de blocos do CI LM2574 (cortesia *National Semiconductors*™)

Este regulador tem capacidade de alimentar cargas até 500 mA a partir de uma tensão de entrada máxima de 40 V. Devido ao facto de utilizar poucos componentes externos, este regulador é simples de utilizar e é um eficiente substituto dos reguladores de tensão lineares. Graças à sua eficiência, as pistas do PCB são normalmente a única dissipação de calor necessária.

A fonte de alimentação implementada utiliza a versão de tensão fixa de saída para +5 V. O circuito implementado é o recomendado pelo fabricante na folha de características do componente.

5.4.2.4. CIRCUITO DE CONTROLO DA TENSÃO DOS MOTORES

Para ser possível ligar/desligar os motores remotamente, foi incluído um relé de estado sólido com capacidade de controlar a corrente dos motores. Foi escolhido o SSR CPC1708 da CLARE™. Este SSR é um relé DC de potência de configuração "normalmente aberto" que utiliza um MOSFET acoplado opticamente e fornece um isolamento entre o controlo e a saída de 2500 V_{rms}. A Figura 38 mostra o diagrama de ligação do relé de estado sólido CPC1708.

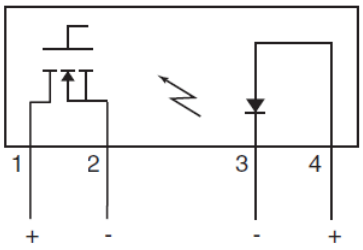


Figura 38 Diagrama do relé de estado sólido CPC1708 (cortesia CLARE™)

Este SSR tem capacidade de bloquear uma tensão de 60 V_P. O fabricante garante o funcionamento até 4 A sem dissipador de calor. Com dissipador o SSR suporta cargas até 11,85 A. A resistência em condução (R_{ON}) é 0,08 Ω. O tempo de comutação t_{ON} é 20 ms e o tempo de comutação t_{OFF} é 5 ms (ver gráfico da Figura 36). A corrente de controlo máxima no LED é de 10 mA (I_F).

5.4.2.5. MICROCONTROLADOR PIC16F1939

Para a unidade de controlo foi escolhido o microcontrolador PIC16F1939 da MICROCHIP™ [Anexo B – V]. Este microcontrolador contém um microprocessador RISC (*Reduced Instruction Set Computer*), está configurado para funcionar com um relógio interno de 32 MHz, disponibilizando 16K de memória FLASH para programa e 1024 *bytes* de RAM ("*Random Access Memory*") para dados.

5.4.3. IMPLEMENTAÇÃO

Esta subsecção apresenta a implementação da "*Caixa de distribuição*", detalhando os blocos constituintes, e finaliza com o circuito impresso resultante.

5.4.3.1. CONVERSOR USB→RS-485

O conversor USB→RS-485 implementado, conforme se mostra no esquema da Figura 39, é projetado com base no circuito integrado FT232RL (U1) da FTDI™ [Anexo B – IV]. Este conversor funciona de forma autónoma (sem intervenção do controlador PIC), e fornece um acesso pelo barramento USB do controlador de experiência à rede RS-485 interna do aparato.

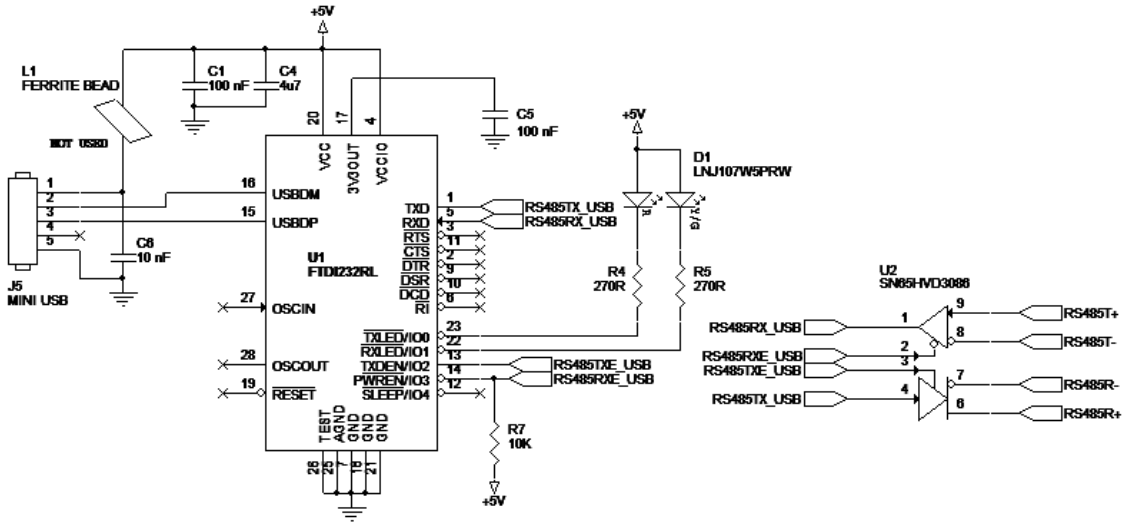


Figura 39 *"Caixa de distribuição"* – Conversor USB→RS-485

A entrada do sinal USB é feita pelo conector USB Mini-A (J5) de cinco pinos. A unidade de carga (UL, "*Unit Load*") na norma USB 2.0 está definida como 100 mA. Esta norma define que um equipamento pode puxar 5 UL de um porto USB [20] após a sua inicialização, limitando a corrente máxima fornecida a 500 mA. O circuito prevê a inclusão da bobina L1 para que seja alimentado a partir do barramento USB. No entanto esta bobina não é inserida e o circuito é alimentado a partir da fonte de alimentação.

O condensador C6 de filtragem só é necessário na configuração alimentada pelo barramento. Os condensadores C1 e C4 são usados para desacoplamento e estão instalados fisicamente próximos dos pinos de alimentação de U1. O condensador de filtragem da saída do regulador interno de +3,3 V (C5) é recomendado pelo fabricante.

O LED bicolor LNJ107W5 (D1) indica a atividade de envio e recepção de dados do UART (*Universal Asynchronous Receiver Transmitter*). Os LED's funcionam com uma corrente aproximada de 11 mA, limitada pelas resistências R4 e R5. O transcetor série/RS-485 SN65HVD3086 (U2) converte os dados a transmitir (RS485TX_USB) em sinal diferencial, com controlo pelo sinal proveniente de U1 RS485TXE_USB, e os dados recebidos no par diferencial no sinal digital RS485_RX_USB, controlado pela saída de U1 RS485RXE_USB.

5.4.3.2. ENTRADAS E SAÍDAS GENÉRICAS

O circuito de entrada e saída, representado na Figura 40, visa facilitar a cablagem ponto-a-ponto com sensores e atuadores do aparato, bem como fornecer saídas capazes de acionar sistemas cuja carga é elevada. O componente CPC1976 (RL2) fornece ao circuito um meio de controlo de uma carga AC até 2 A_{rms}, e a sua saída está disponível no conector J16. As saídas de corrente contínua estão disponíveis no conector J17 e são controladas pelo SSR CPC1002 com capacidade de atuar cargas de 700 mA com 60 V_P. Todos os LED de controlo dos SSR estão limitados a 10 mA pelas respetivas resistências limitadoras (R9, R11, R13 e R14). Os condensadores de 100 nF destinam-se a absorver possíveis ruídos nas linhas digitais de controlo dos SSR.

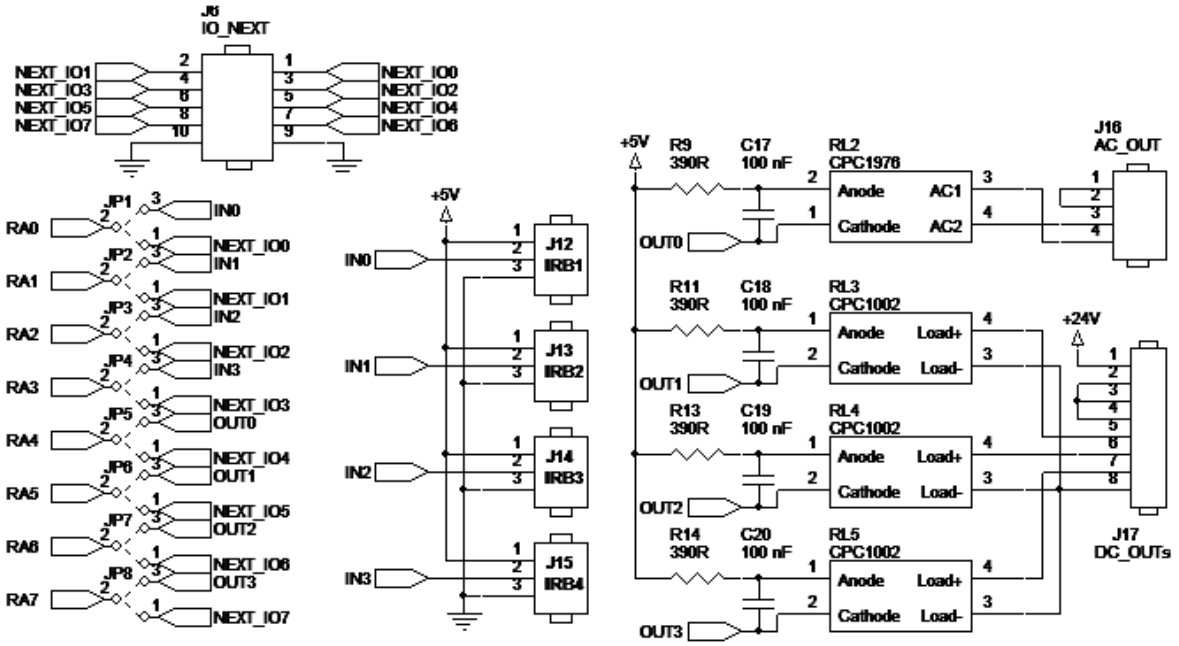


Figura 40 *"Caixa de distribuição"* – Entradas e saídas genéricas

As entradas são ligadas diretamente ao microcontrolador e os sinais a elas ligados podem ser em coletor aberto ou lógicos a +5 V. Estão implementados quatro entradas (J12… J15), que simultaneamente disponibilizam a alimentação ao sensor (+5 V).

O circuito de entradas e saídas está ligado ao porto RA[0…7] via *jumpers* JP[1…7]. Estes *jumpers* permitem reencaminhar as linhas do porto não utilizadas para o conector de expansão J6 (IO_NEXT). A finalidade deste conector é permitir o encaminhamento de sinais digitais entre sub-sistemas.

5.4.3.3. FONTE DE ALIMENTAÇÃO DE +5 V

A tensão +24 V proveniente da fonte externa é ligada ao conector J1. Após passar pelo interruptor externo SW1 e pelo fusível F1 fica disponível para alimentar circuitos externos de +24V através do conector J2, que alimenta a entrada da fonte de tensão comutada de +5 V e o circuito de alimentação dos motores.

Para alimentar os circuitos locais e externos (sensores), foi projetada a fonte de alimentação (Figura 41) para +5 V, que utiliza o regulador comutado LM2574M-5.0 (U4). A fonte está definida para funcionar com uma tensão de entrada de +24 V e fornecer uma corrente de 500 mA. O método de cálculo dos componentes da fonte de alimentação comutada está descrito no Anexo A.

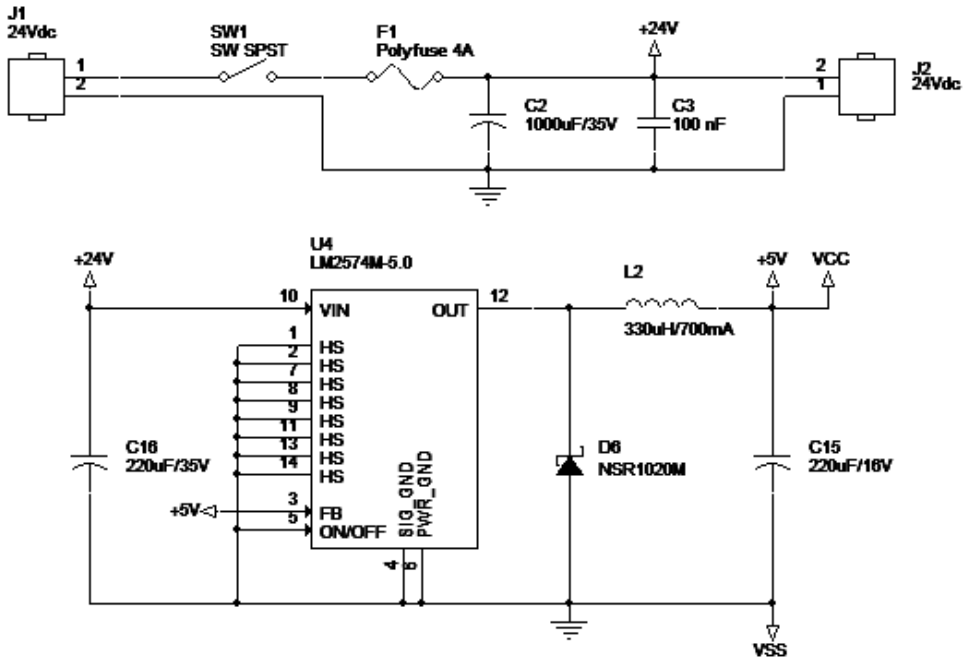


Figura 41 *"Caixa de distribuição"* – Fonte de alimentação de +5 V

5.4.3.4. CIRCUITO DE CONTROLO DA TENSÃO DOS MOTORES

O circuito de controlo da tensão dos motores (Figura 42) permite ligar/desligar a alimentação dos motores via programa de controlo da experiência. Desta forma é possível reduzir o consumo, durante os intervalos de tempo de inatividade.

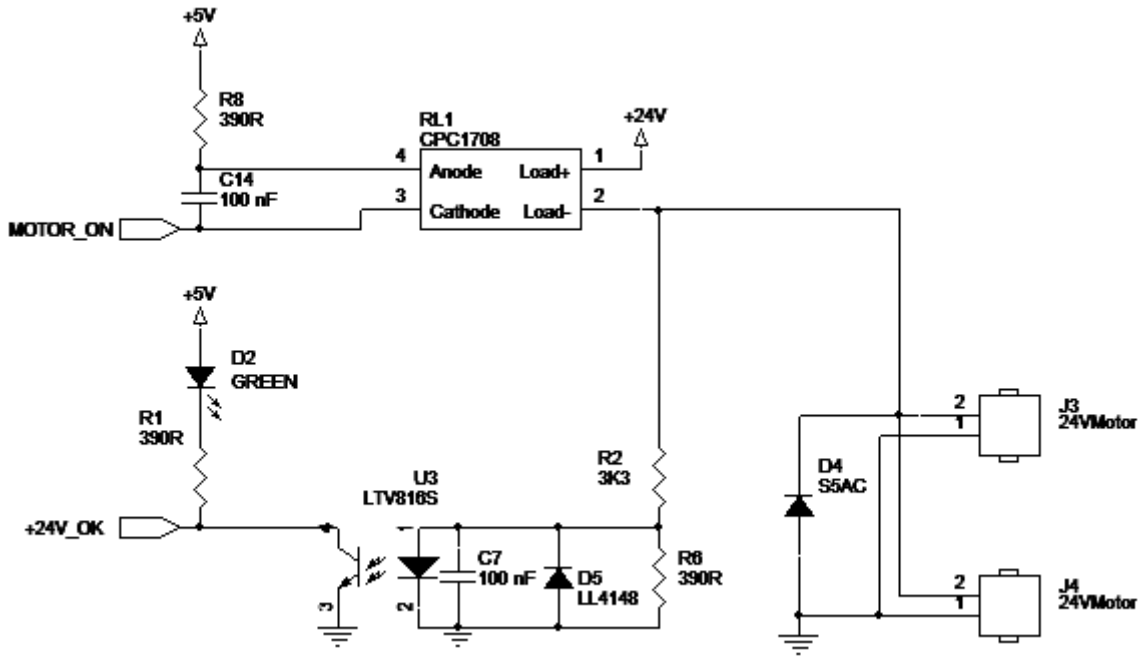


Figura 42 *"Caixa de distribuição"* – Circuito de controlo da tensão dos motores

Com essa finalidade, a tensão de alimentação dos motores (24VMotor) é controlada pelo relé de estado sólido de potência RL1. A corrente do LED de entrada está limitada a 10 mA pela resistência R8. A carga indutiva dos motores armazena energia que, no instante em que os motores são desligados, é dissipada através do díodo de roda livre (*free-wheeling*) D4.

O circuito de entrada para a alimentação dos motores é proveniente do interruptor geral da "*Caixa de distribuição*" e passa pelo fusível F1. Para que o programa de controlo possa saber se os motores estão ou não alimentados foi implementado o circuito com o foto-acoplador U3 que monitoriza a tensão dos motores pelo divisor resistivo R2/R6 e sinaliza o seu estado pela linha +24V_OK. Simultaneamente o LED D2 fica aceso indicando a presença de tensão nos motores.

5.4.3.5. MICROCONTROLADOR

A Figura 43 apresenta o circuito de controlo da "*Caixa de distribuição*" que tem como elemento central o microcontrolador da MICROCHIP™ PIC16F1939-I/PT [Anexo B – V]. Este microcontrolador tem 44 pinos, disponibiliza um total de 36 linhas de entrada e saída (E/S) e é da mesma família de controladores do utilizado no "*Sensor de bola*" (Figura 31).

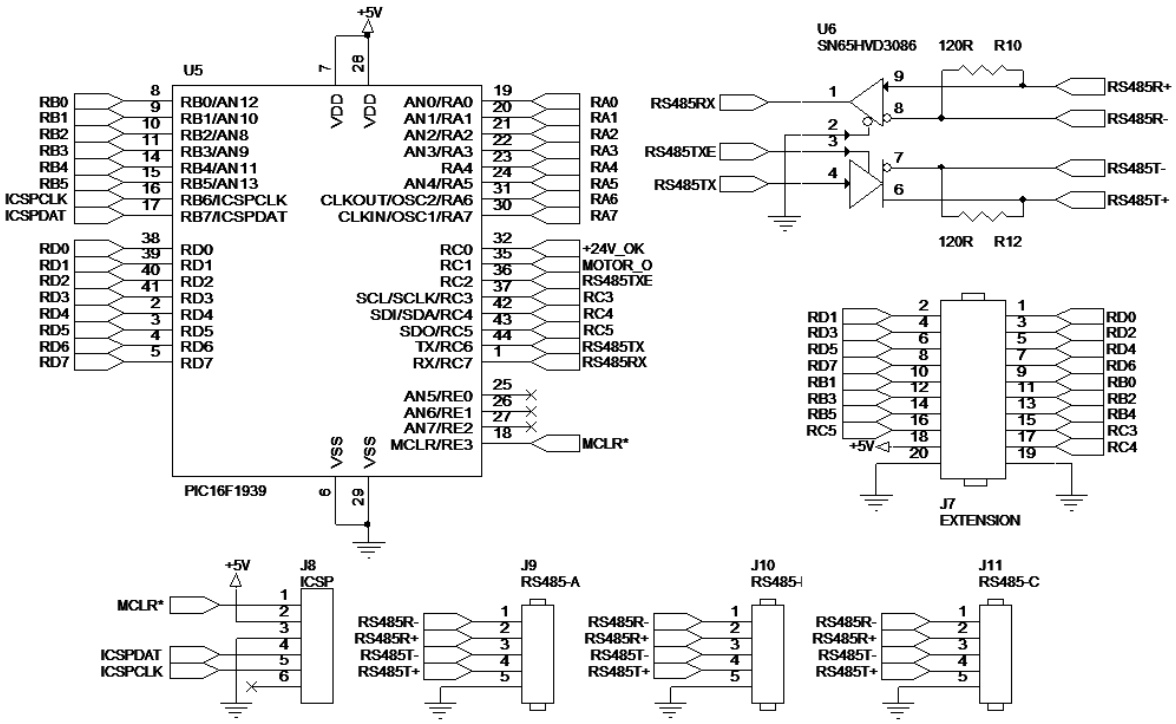


Figura 43 *"Caixa de distribuição"* – Circuito do microcontrolador PIC

O porto RA[0…7] é utilizado nas entradas e saídas genéricas (5.4.3.2). Os *bits* RC0 e RC1 são respetivamente as linhas de entrada +24V_OK e de saída MOTOR_ON. Para a comunicação série são utilizados os *bits* RC2 (RS485TXE), RC6 (RS485TX) e RC7 (RS485RX). Estes sinais são ligados ao transcetor série/RS485 (U6) que interliga a USART do PIC à rede RS-485. O circuito disponibiliza a localização para as resistências terminadoras (R10 e R12) do barramento RS-485. Dado o facto que esta placa é uma caixa de distribuição, foram incluídos vários nós de acesso da rede RS-485 (J9, J10 e J11). Os portos RB[0…5], RC[3…5] e RD[0…7] fazem parte do conector de expansão IDC (*Insulation Displacement Connector*) de 20 pinos. Este conector também disponibiliza a alimentação no pino 18 (+5 V). Para ser possível a programação do PIC diretamente na placa, foi incluído o conector J8 para ligação do programador ICSP (*In-Circuit Serial Programmer*).

5.4.3.6. CIRCUITO IMPRESSO

A implementação física da "*Caixa de distribuição*" teve como base uma caixa plástica de dimensões reduzidas que se mostrou apropriada ao seu alojamento. A finalidade principal da caixa é a interligação da cablagem e por isso, foram escolhidos os conectores modulares da série "*Combicon PTSA*" e "*Combicon PTSM*" da PHOENIX™ para a ligação de fios a PCB's sem necessidade de soldar.

A Figura 44 apresenta no topo os desenhos da camada de cobre e ao centro, os desenhos da implantação de componentes, respetivamente no lado superior e inferior do circuito impresso. Na parte inferior são apresentadas respetivamente, as vistas superior e inferior da montagem final da "*Caixa de distribuição*" do subconjunto do "*Elevador da rampa de lançamento*".

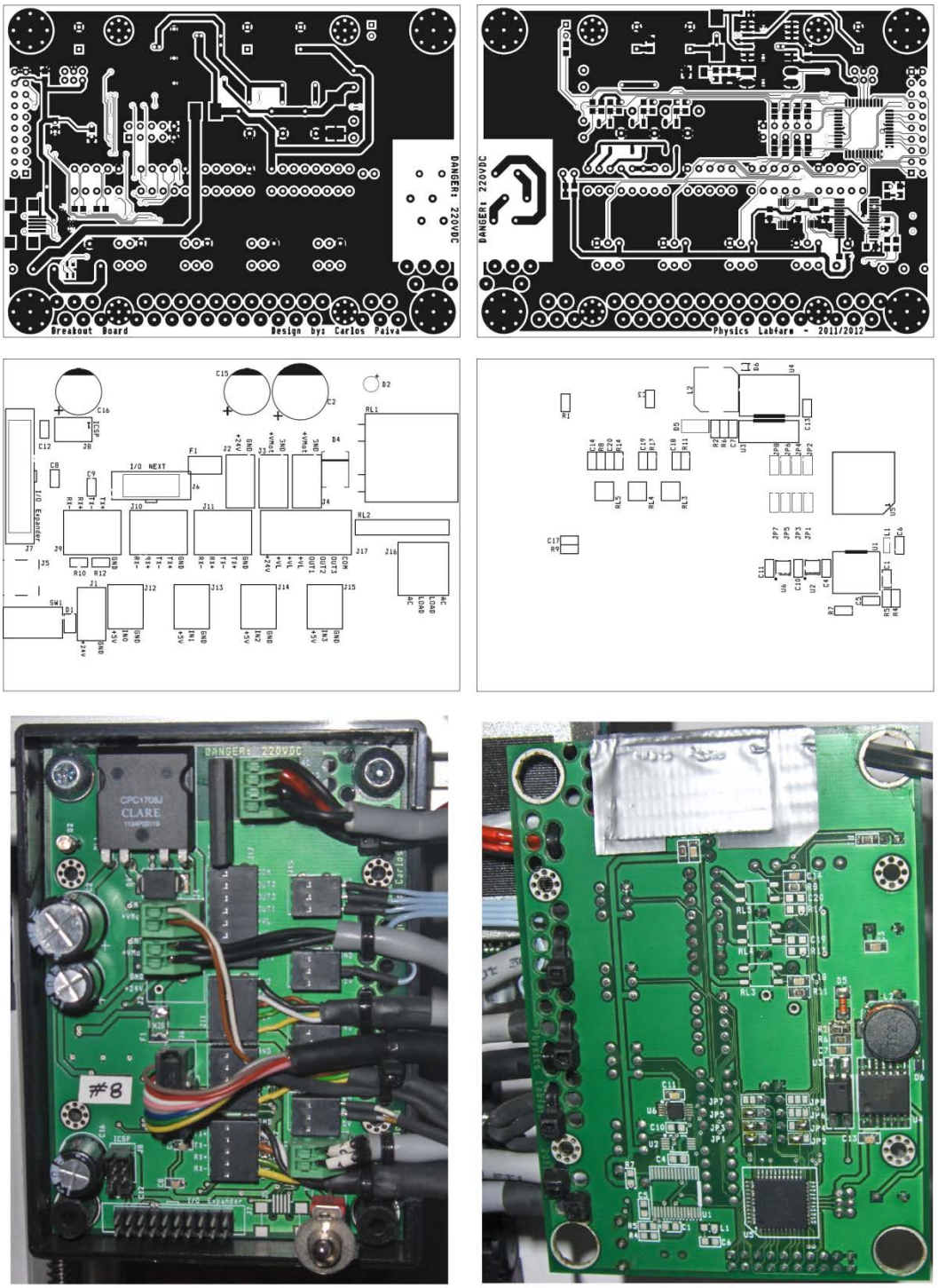


Figura 44 *"Caixa de distribuição"* – Circuito impresso e montagem final

Na vista da montagem do lado superior (foto inferior esquerda), é possível ver a fixação da cablagem à placa com abraçadeiras, utilizando os furos existentes no PCB para esse fim. Este

processo de fixação elimina as situações de fios partidos e maus contactos provenientes da sua torsão durante as operações de montagem e reparação. Na zona inferior do PCB, vê-se o interruptor para o corte manual da alimentação (+24 V_{DC}).

Na vista da montagem do lado inferior (foto inferior direita), a fita que se vê no topo, segura um isolamento plástico adicional colocado na zona sujeita a tensões potencialmente perigosas (230 V_{AC}), como forma complementar de proteção a um possível choque. Os componentes em falta na zona inferior são relativos ao circuito de interface USB→RS-485, que só está implementado na "*Caixa de distribuição*" do "*Elevador de bola*".

5.5. CONTROLO DOS MOTORES PASSO-A-PASSO

Esta secção resume o funcionamento dos motores passo-a-passo, e as suas configurações mais comuns e apresenta com maior detalhe, o controlador selecionado para os motores passo-a-passo utilizados no aparato.

5.5.1. MOTORES PASSO-A-PASSO

Um motor passo-a-passo é um motor DC sem escovas que divide uma rotação completa, num conjunto de passos iguais entre si. Assim, é possível mover o motor e parar num desses passos, sem que haja necessidade de realimentação, desde que o motor seja dimensionado para a aplicação em causa. O rotor de um motor passo-a-passo é um potente íman em ferro macio, com uns finos dentes, desfasados ligeiramente entre si, que definem o tamanho físico do passo. Estes dentes, juntamente com as bobinas do estator, formam um eletroíman que, quando energizado, tende a alinhar o dente mais próximo com a bobina. Quando a bobina seguinte é energizada, o rotor move-se nesse sentido de forma a ficar alinhado com o dente mais próximo. Este pequeno movimento do rotor entre os seus dentes é denominado de passo. Uma rotação completa é constituída por um número inteiro de passos, que num motor comercial é tipicamente 100, 200 ou 400 passos. O número de passos físicos do motor define a deslocação angular por passo (360° / total de passos).

5.5.2. PRINCIPAIS CARACTERÍSTICAS DE UM MOTOR PASSO-A-PASSO

Os motores passo-a-passo são máquinas elétricas de potência constante. À medida que a velocidade aumenta, o torque do motor diminui. Normalmente, estes motores apresentam o torque máximo no estado estacionário. Este torque é denominado "torque de bloqueio" e define a capacidade do motor manter a sua posição face à sua carga externa.

Estes motores são mais ruidosos que os outros tipos de motores DC devido à existência de vibração mecânica no bloqueio do movimento do rotor. Em cada passo o rotor oscila em torno da zona de

bloqueio do campo magnético, como se fosse uma mola, com uma determinada frequência de ressonância. Se a frequência do motor for igual à frequência de ressonância, a oscilação aumenta levando o motor a perder o sincronismo. Um método de minimizar o ruído é acelerando rapidamente o motor nas zonas próximas da frequência de ressonância. É normal usar o controlo por micro passo de forma a suavizar o movimento.

Os motores passo-a-passo são normalmente controlados num sistema em malha aberta, em que o controlador não tem informação da posição real do motor. Por este motivo, de forma a reduzir a possibilidade de erro no movimento por perda de passos, os motores controlados em malha aberta são sobredimensionados, principalmente no caso de cargas inerciais elevadas ou muito variáveis. Esta problemática está na base de decisão de projeto, mais barato com um sistema sobredimensionado (malha aberta) ou um sistema mais dispendioso com um servomecanismo (malha fechada).

5.5.3. TIPOS DE CONFIGURAÇÃO

As bobinas de um motor passo-a-passo podem ser ligadas na configuração bipolar ou unipolar. As bobinas na configuração unipolar disponibilizam um ponto de acesso central em cada uma das fases. Dado que nesta montagem um polo magnético pode ser invertido sem comutar a corrente, o controlo requer unicamente um comutador (e.g. transístor), por cada enrolamento. Esta configuração está disponível com 6 ou 8 fios de ligação com o exterior. A configuração de 6 fios é exclusiva para a configuração unipolar. A configuração a 8 fios permite que o motor seja configurado como unipolar ou bipolar. A configuração bipolar disponibiliza unicamente um enrolamento por fase. Assim, a inversão do polo magnético é obtida pela inversão da corrente do enrolamento. O circuito de controlo é mais complicado, e é tipicamente implementado com pontes "H". O motor disponibiliza quatro fios independentes, dois por cada fase.

5.5.4. FORMAS DE ONDA DA CORRENTE DE FASE

Um motor passo-a-passo pode ser visto como um motor síncrono AC em que o número de polos, quer no rotor quer no estator, foram aumentados. Foram desenvolvidos vários métodos de controlo:

- Passo inteiro – Este método tem sempre duas fases ligadas. O motor move-se um passo de cada vez, e fornece o torque máximo disponível;
- Meio passo – O controlo alterna entre duas fases ligadas e só uma ligada. Este método aumenta a resolução angular, mas o motor tem um torque inferior (aproximadamente 70%) nas posições de ½ passo devido a só ter uma bobina energizada;

- Micro passo – Este controlo varia a corrente do enrolamento de uma forma aproximada, à forma de onda sinusoidal. À medida que os micro passos se tornam mais pequenos, o funcionamento do motor fica mais suave, reduzindo de forma significativa a ressonância;

5.5.5. CONTROLADOR SMCI12 DA NANOTEC™

O controlador de motores passo-a-passo bipolares SMCI12 da Nanotec™ tem um estágio de saída de corrente constante com controlo integrado de corrente em malha fechada. Graças à sua grande capacidade e funções de controlo disponíveis, oferece ao projetista uma forma rápida de resolução de vários cenários do controlo deste tipo de atuadores, reduzindo os requisitos de programação.

O controlador, apresentado na Figura 45, está disponível com dois interfaces de rede distintos: RS-584 e CAN. Esta unidade tem as seguintes características:

- Controlo em micro passo (1/1 → 1/64) com uma resolução por passo até 0,014° (a 1/64) em motores de 200 passos por revolução;
- Microcontrolador DSP ("*Digital Signal Processor*") com resolução de 12 *bit* de controlo da corrente;
- Interface de rede (RS-485 ou CAN) para parametrização e controlo;
- Capacidade de interligar até 254 motores (RS-485) ou 127 motores (CAN);
- Corrente de saída por fase ajustável até 2,7 A (1,8 A em modo contínuo);

Figura 45 Controlador SMCI12 da Nanotec™

Este módulo pode ser programado internamente na linguagem NanoJ™, baseada em Java, o que permite a realização local do controlo, sem a intervenção de controladores adicionais. Esta funcionalidade só está disponível nos modelos com RS-485.

O comportamento do motor pode ser ajustado e otimizado de acordo com os requisitos, definindo os parâmetros do motor a controlar. Os parâmetros podem ser ajustados com recurso à aplicação NanoPro™, reduzindo de forma significativa o tempo de configuração.

A Figura 46 apresenta a localização dos componentes no PCB, com destaque para os conectores existentes.

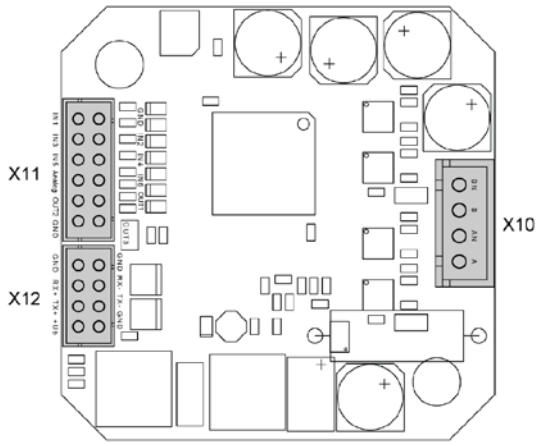


Figura 46 Localização dos conectores do controlador SMCI12

O controlador contém os seguintes conectores:

- X10 – Ligação do motor bipolar (A, AN, B e BN);
- X11 – Entradas e saídas digitais (IN[1…6], OUT[1…3]) e entrada analógica AN1;
- X12 – Alimentação e interface de comunicação (RS-485 ou CAN);

O circuito das entradas digitais aceita no máximo uma tensão de +5 V. Para o circuito ser capaz de identificar o estado lógico '0', a tensão de entrada deve ser inferior a +2 V. O estado lógico '1' é válido quando a tensão de entrada for superior a +4,5 V. O circuito das saídas digitais está configurado com MOSFET em dreno aberto e tem capacidade de comutar 24 V_{DC} com uma carga de 500 mA. A entrada analógica aceita uma tensão de entrada entre -10 V e +10 V. A tensão de funcionamento deve estar entre +12 V e +24 V e é ligada ao pino 8 do conector X12, que tem como comum o pino 7. De forma a prevenir que a tensão ultrapasse os limites de funcionamento (e.g. no instante de travagem do motor), é necessária a inclusão de um condensador de carga na tensão de alimentação (entre 4700 µF e 10000 µF). O diagrama de ligação da alimentação do controlador é apresentado na Figura 47.

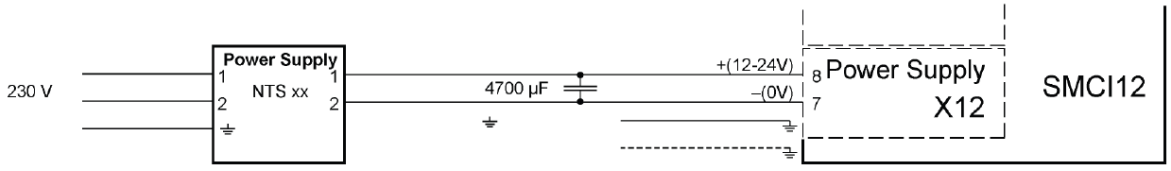


Figura 47 Diagrama de ligação da alimentação do controlador SMCI12

A ligação do módulo a uma rede (RS-485 ou CAN) é efetuada no conector X12, comum à ligação do circuito de alimentação. Na variante RS-485, os sinais da rede RX-, RX+, TX- e TX+ são ligados respetivamente aos pinos 3, 4, 5 e 6. Esta configuração a quatro fios permite o funcionamento em modo *full-duplex* e suporta comunicações assíncronas até 115200 baud.

No modelo CAN, os sinais CAN- e CAN+ são ligados respetivamente aos pinos 5 e 6. Os pinos 1 e 2 são ligados ao comum da alimentação (pino 8) e são destinados a ligar a malha da blindagem do cabo de ligação ao comum.

O programa residente de controlo pode efetuar o controlo completo do motor, sem intervenção de um controlador de nível superior, via 16 perfis de movimento totalmente configuráveis. Estes perfis permitem definir a velocidade inicial e máxima do movimento, com especificação da rampa de aceleração, tipo de movimento (absoluto ou relativo), distância a percorrer (especificada em passos, milímetros, etc.) e possível verificação de sensores no seu trajeto. No fim do seu movimento, um perfil pode especificar um outro para continuar o processo desejado.

5.6. SISTEMA DE EXPANSÃO DO CONTROLADOR SMCI12

Para ampliar as capacidades de ligação externa do controlador SMCI12 da Nanotec™, foi desenhado um circuito de expansão que encaixa diretamente no PCB do SMCI12, ligando-se aos conectores X11 e X12. Este circuito, além de facilitar a interligação dos componentes necessários ao funcionamento (sensores, alimentação e rede), prevê um circuito adicional de entrada com o intuito de interligar um codificador incremental, que sendo uma necessidade comum no controlo de motores passo-a-passo em malha fechada, não está previsto no controlador SMCI12. Nas subsecções seguintes são apresentados o diagrama de blocos, uma descrição do modo de funcionamento e a sua implementação, respetivamente.

5.6.1. DIAGRAMA DE BLOCOS

A Figura 48 apresenta o diagrama de blocos do "*Sistema de expansão SMCI12*". Este sistema de expansão tem por base o microcontrolador PIC (U1) que se liga à rede RS-485 via transcetor U2. O módulo encaixa diretamente nos conectores X11 e X12 existentes no controlador SMCI12 e além de lhe fornecer a alimentação (+24 V), liga os sinais de rede aos conectores RS-485 adicionais e todos os sinais de I/O são ligados ao microcontrolador (Portas RA e RC). Da mesma forma, todos os sinais de interação externa (IN[1…3], OUT1 e ENCODER) são ligados simultaneamente ao conector de expansão I/O IDC e ao microcontrolador (portas RB e RD).

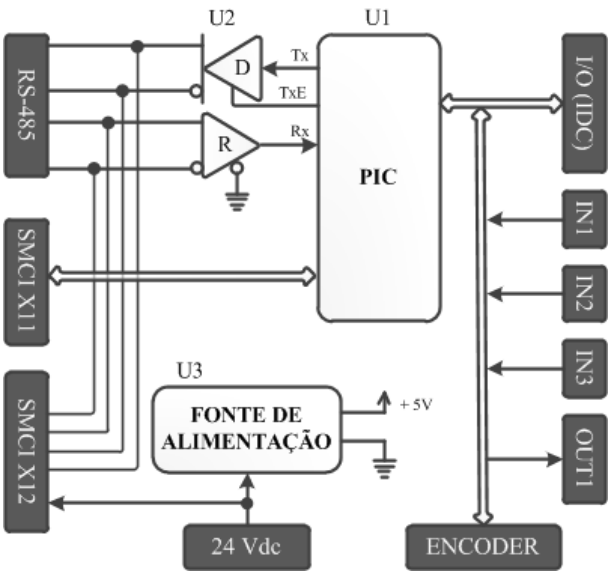


Figura 48 Diagrama de blocos do "*Sistema de expansão SMCI12*"

5.6.2. MODO DE FUNCIONAMENTO

O "*Sistema de expansão SMCI12*" visa a interligação dos componentes eletrónicos necessários a um sistema de controlo.

Tipicamente é necessário obter informação dos sensores de limite de movimento e/ou de um ou mais estados externos (e.g. passagem de um corpo por uma determinada posição), controlar um atuador (e.g. relé externo) e opcionalmente ler um codificador incremental (para obter a posição física do motor).

5.6.2.1. SENSORES DE LIMITE

Para determinar o limite de movimento linear dos vários elementos mecânicos móveis, foi escolhido o sensor de rasgo OPB840 apresentado na Figura 49. Este sensor tem um rasgo de 3,2 mm com uma altura de 8,0 mm. É composto por um LED emissor infravermelho e um foto-transístor encapsulados numa caixa plástica opaca para reduzir a sensibilidade à radiação infravermelha ambiente.

Figura 49 Sensor OPB840 (cortesia OPTEK™)

Para o funcionamento deste sensor, foi necessário desenhar o circuito da Figura 50 para alojar as resistências e os conectores necessários junto ao sensor.

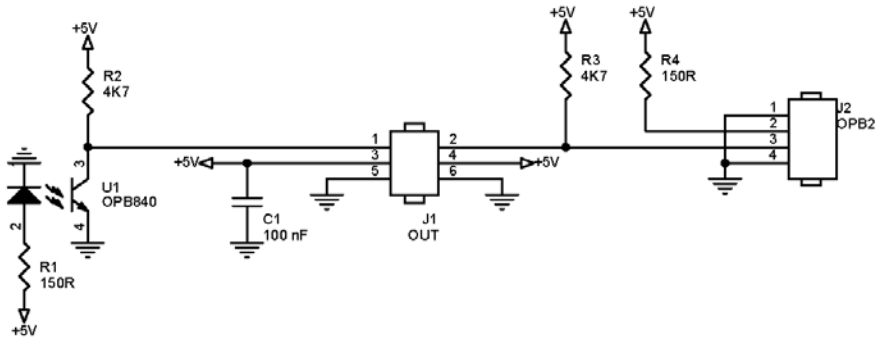


Figura 50 Esquema de ligação do sensor OPB840

O circuito é fixado ao sensor OPB840 (U1) e disponibiliza o conector de saída J1 onde estão disponíveis os sinais de saída do coletor de U1 (pino 1) e o sinal de saída de um segundo sensor OPB840 ligado a J2 (pino2). As resistências R1 e R4 limitam a corrente dos emissores a 22 mA, e as resistências R2 e R3 são as resistências de coletor.

5.6.2.2. CODIFICADORES DE POSIÇÃO

Para obter a informação do ângulo do rotor do motor foi selecionado o codificador incremental WEDS5541-B14 da Nanotec™ (Figura 51) que fornece 1000 impulsos por revolução e é mecanicamente instalável em motores de duplo veio com 5 mm de diâmetro.

Figura 51 Codificador WEDS5541-B14 [25]

Em complemento dos sinais ChA e ChB em quadratura, o codificador tem um sinal de índice (ChI) que dá a origem da contagem dos impulsos. A Figura 52 mostra o diagrama de ligação [a)] do conector e o diagrama temporal dos sinais gerados [b)].

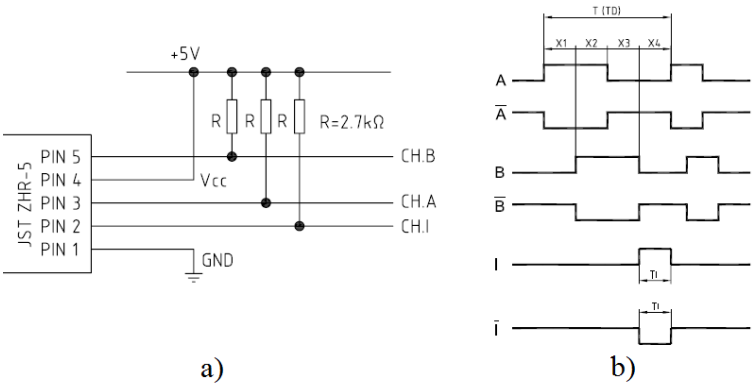


Figura 52 WEDS5541 – Diagrama de ligação e sinais gerados [25]

Os sinais $\bar{A}$, $\bar{B}$ e $\bar{I}$ referem-se ao codificador com saída em modo diferencial WEDL.

5.6.2.3. ENTRADAS E SAÍDAS

Os sinais de I/O do controlador SMCI12 são ligados diretamente ao microcontrolador (porto RA e RC). Os sinais de I/O do interface externo são fisicamente as linhas do porto RB e RD do microcontrolador. Com esta montagem, o programa residente tem capacidade de mapear os sinais do controlador SMCI12 nas linhas de I/O do interface externo que são disponibilizadas no conector digital IDC.

Este módulo de expansão é um nó da rede RS-485. Desta forma é possível atuar e ler estes sinais remotamente via rede RS-485.

5.6.3. IMPLEMENTAÇÃO

Esta subsecção apresenta a implementação do "*Sistema de expansão SMCI12*", detalhando os blocos constituintes, e finaliza com o circuito impresso resultante.

5.6.3.1. MICROCONTROLADOR

A Figura 53 apresenta o circuito de controlo do "*Sistema de expansão SMCI12*" que tem como elemento central o microcontrolador da MICROCHIP™ PIC16F1939-I/PT [Anexo B – V]. Este microcontrolador com 44 pinos disponibiliza um total de 36 linhas de entrada e saída e é o mesmo utilizado na "*Caixa de distribuição*".

O porto RA[0…7] é ligado aos sinais de entrada (SMCI_IN[1…6] e SMCI_AN1) do conector X11 do controlador SMCI12 via conector J2. O porto RB[0…5] contém os sinais de entrada deste módulo e estão ligados aos vários conectores de interface externo. O porto RC[0…2] é ligado aos sinais de saída (SMCI_OUT[1…3]) do conector X11 do controlador SMCI12 via conector J2.

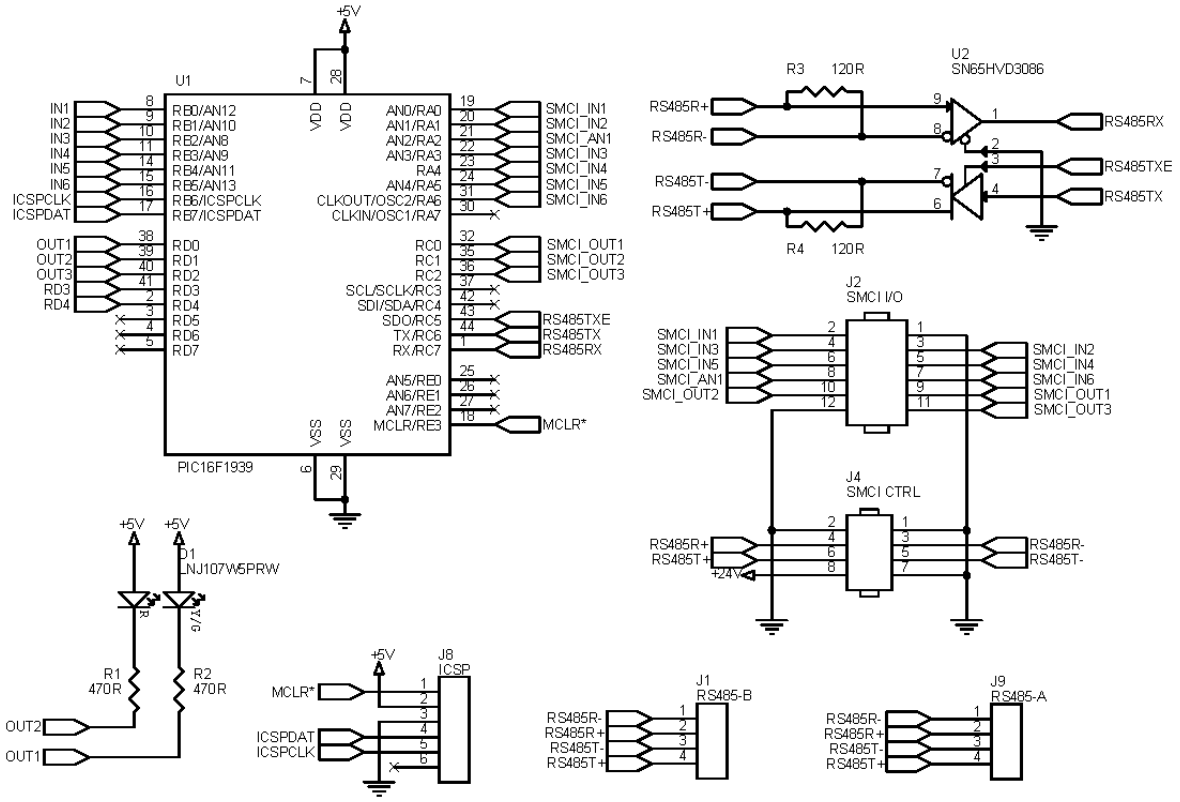


Figura 53 *"Sistema de expansão SMCI12"* – Circuito do microcontrolador PIC

O interface RS-485 utiliza as linhas de acesso ao periférico USART do microcontrolador (RS485TX e RS485 RX) presentes nos pinos RC6 e RC7 respetivamente, usa o *bit* de saída RS485TXE (RC5) para controlar o andar de saída do transmissor do transcetor RS-485 (U2). Os conectores J1 e J9 são a interface externa da rede RS-485 que é interligada ao conector X12 do controlador SMCI12 via J4. Neste conector também é disponibilizada a tensão de alimentação do motor (+24 V). O porto RD[0…4] tem os *bits* de saída OUT[0…2]. Os sinais OUT1 e OUT2 disponibilizam uma sinalização visual via LED D1, cuja corrente é limitada pelas resistências R1 e R2 a 10 mA. Para a programação do PIC diretamente na placa, foi incluído o conector J8 para ligação do programador ICSP que utiliza os sinais ICSPCLK (RB6), ICSPDAT (RB7) e MCLR* (RE3).

5.6.3.2. FONTE DE ALIMENTAÇÃO DE +5 V

A tensão +24 V proveniente da fonte externa é ligada ao conector J11, que fornece energia à fonte de tensão comutada formada pelo regulador comutado U3, pelos condensadores C2 de entrada e C3 de saída, pela bobina L1 e pelo díodo de captura D3. A fonte apresentada na Figura 54 está calculada para fornecer +5 V. Esta fonte é igual à utilizada na "*Caixa de distribuição*" (5.4.3.3). O seletor JP1 permite alimentar o módulo a +24 V ou a +5 V a partir de uma fonte externa ligada ao pino 12 do conector J10 (+5Vin). Os condensadores de desacoplamento C4, C5 e C6 são colocados junto dos pinos de alimentação dos circuitos integrados utilizados no módulo (U1 e U2).

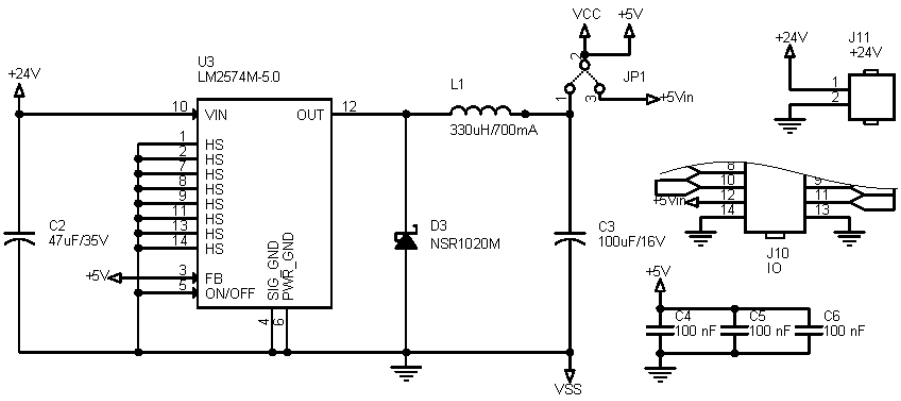


Figura 54 *"Sistema de expansão SMCI12"* – Fonte de alimentação de +5 V

5.6.3.3. ENTRADAS E SAÍDAS DIGITAIS

Os conectores de entrada apresentados na Figura 55 disponibilizam a tensão +5 V para poder ser utilizada na alimentação dos sensores a eles ligados.

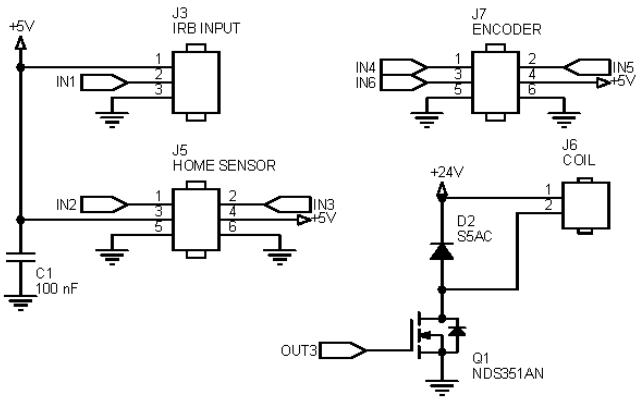


Figura 55 *"Sistema de expansão SMCI12"* – Entradas e saídas digitais

O conector J3 destina-se a ligar ao conector J2 do "*Sensor de bola*" (secção 5.3). O conector J5 é para a ligação de dois sensores de limite (IN2 e IN3) e é compatível com o conector do circuito do sensor OPB840 (Figura 50). O conector J7 destina-se a ligar um codificador incremental WEDS que necessita de três sinais de entrada ChA, ChB e ChI (Figura 52), ligados respetivamente às entradas IN4, IN5 e IN6.

O sinal de saída OUT3 disponibilizado no conector J6 utiliza o MOSFET Q1 como amplificador de corrente. Este MOSFET tem uma tensão dreno - fonte V_{DS} de 30 V e suporta uma corrente de dreno de 1,4 A (I_D) com uma resistência de condução de 160 mΩ (Rds_{ON}). Tendo em conta a necessidade de ligar cargas indutivas (e.g. relés) foi incluído o díodo de roda livre D2.

5.6.3.4. CIRCUITO IMPRESSO

Este módulo foi desenhado para ser uma expansão do controlador SMCI12 da Nanotec™, o que condicionou as suas dimensões. A Figura 56 apresenta respetivamente, da esquerda para a direita, os desenhos da camada de cobre do circuito impresso, a implantação de componentes e a sua montagem final.

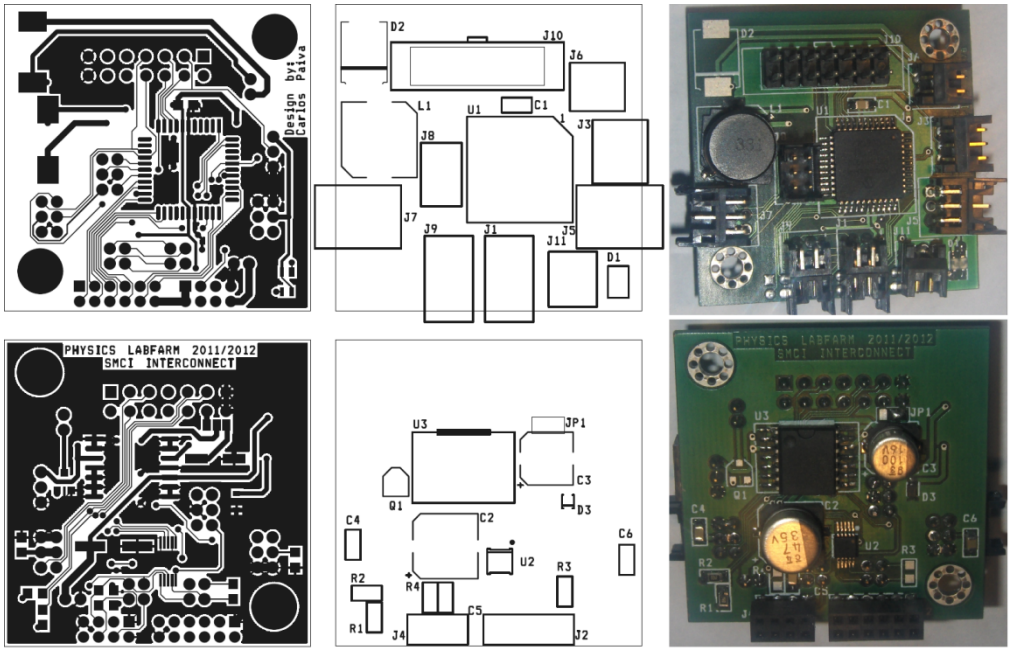


Figura 56 *"Sistema de expansão SMCI12"* – Circuito impresso

Na foto inferior direita, é possível ver os conectores de interligação entre os dois circuitos impressos. O módulo tem dois furos de 3,2 mm coincidentes com a furação do controlador SMCI12 para a junção dos dois módulos utilizando espaçadores de 8 mm de altura. Desta forma este módulo fica solidário com o controlador para que as vibrações mecânicas a que está sujeito não afetem os contactos do conector.

5.7. SISTEMA DE MEDIÇÃO DO '*ALCANCE HORIZONTAL DO PROJÉTIL*'

Um dos requisitos de projeto do aparato em desenvolvimento é a necessidade de medir o '*alcance horizontal do projétil*'. O projétil é uma bola em material ferromagnético com a superfície polida. Foram equacionadas várias opções de medição:

- Sistema de medição por ultrassons;
- Sistema de deteção por vídeo;
- Barreira de raios infravermelhos;

O sistema de medição por ultrassons provou não ser eficiente devido ao facto do projétil ser esférico, e por consequência defletir (e não refletir) o sinal enviado. O sistema de vídeo era passível de funcionar, mas tendo em conta a velocidade do projétil, requeria, para a sua deteção uma câmara de vídeo de alta velocidade, cujos custos são muito elevados.

Por fim, foi equacionada a barreira de raios infravermelhos e analisado o método possível de funcionamento. Como premissa deste sistema de medição, foi estipulado que uma resolução de 2 mm seria suficiente para o estudo em causa.

O sistema resultante implementa uma barreira de LED's emissores de infravermelhos e respetivos fotorreceptores cujos raios são refletidos por um espelho colocado do lado oposto da zona de recolha do projétil. As subsecções seguintes apresentam o diagrama de blocos, o modo de funcionamento e a sua implementação, respetivamente.

5.7.1. DIAGRAMA DE BLOCOS

A Figura 57 apresenta o diagrama de blocos do sistema de medição do '*alcance horizontal do projétil*'. Este módulo é composto por uma barreira de 48 LED's infravermelhos ligados a um controlador que forma um registo de deslocamento (série/paralelo) com controlo de corrente. A sua radiação é refletida num espelho e recebida pelos 96 foto-transístores cujas saídas estão ligadas às entradas de um registo de deslocamento (paralelo/série) de 96 *bits*.

A utilização dos registos de deslocamento permite reduzir o número de pinos do microcontrolador PIC (U7) necessários ao controlo. O microcontrolador implementa um periférico SPI (*Serial Peripheral Interface*) para o seu acesso.

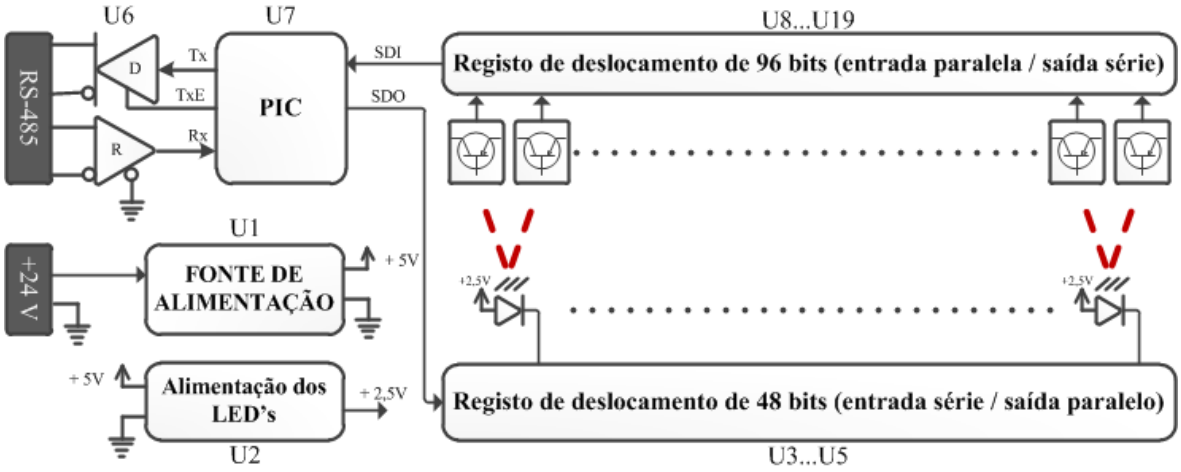


Figura 57 Diagrama de blocos do sistema de medição do alcance

O módulo dispõe de uma fonte de alimentação que converte a tensão de entrada +24 V em +5 V para alimentar os seus circuitos lógicos. Para reduzir a dissipação nos controladores dos LED's foi

incluído um regulador de 2,5 V. Este módulo acede à rede RS-485 via transcetor U6 para ser possível interagir com o sistema de medição remotamente.

5.7.2. MODO DE FUNCIONAMENTO

O sistema de medição do alcance do projétil implementa uma barreira de emissores e receptores que são acedidos por um barramento SPI [Anexo B – VI] pelo microcontrolador PIC. No seu controlo, faz-se um varrimento dos emissores (ligando-os dois a dois), e é adquirido o estado das entradas correspondentes, que em função da sua posição física, vai determinar a posição do projétil, relativamente à origem do circuito impresso.

A posição final calculada no controlador de experiência é em função da leitura aqui obtida, acrescida de uma compensação, que relaciona a posição física do sistema de medição com a saída da rampa de lançamento.

5.7.3. IMPLEMENTAÇÃO

Esta subsecção apresenta a implementação do sistema de medição do '*alcance horizontal do projétil*', detalhando os blocos constituintes, e finaliza com o circuito impresso resultante.

5.7.3.1. CONTROLADOR PIC

O controlo do módulo cujo esquema é apresentado na Figura 58, implementado pelo microcontrolador PIC16F1825-I/ST (U7), é igual ao descrito no "*Sensor de bola*" (Figura 31).

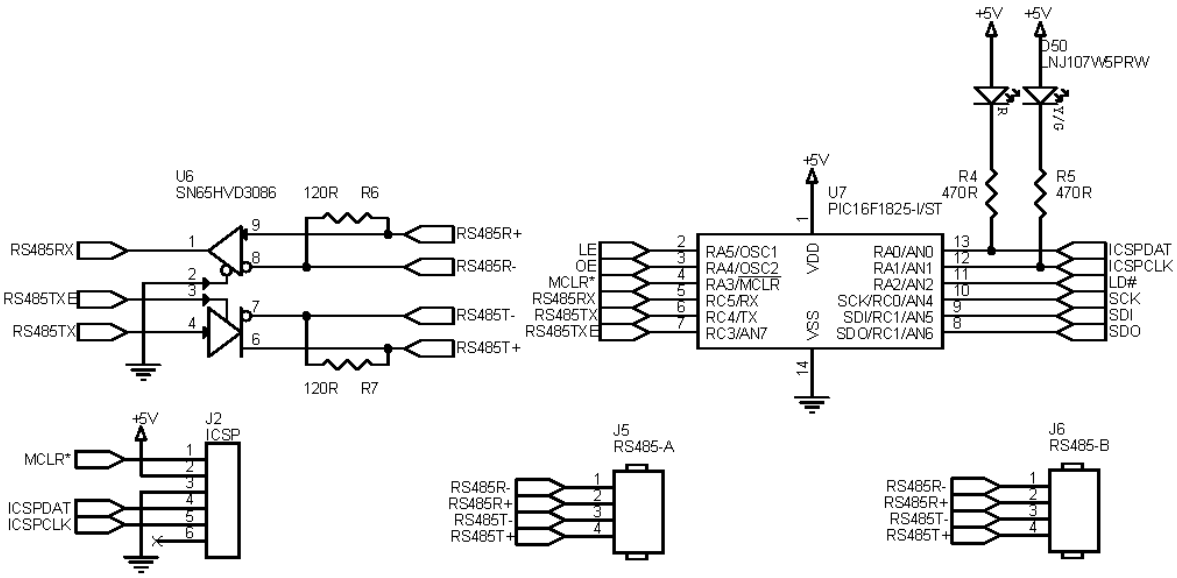


Figura 58 Sistema de medição – Circuito do microcontrolador PIC

Este microcontrolador acede à rede RS-485 via transcetor U6, com acesso físico à rede via conectores J5 e J6. O LED duplo D50 dá informação visual das saídas RA0 e RA1. O conector J2 permite a programação no PCB do microcontrolador. Os sinais SCK ("*Serial Clock*"), SDI ("*Serial Data In*"), SDO ("*Serial Data Out*"), LD*, LE e OE formam o interface SPI de controlo da barreira.

5.7.3.2. FONTE DE ALIMENTAÇÃO

A fonte de alimentação de +5V utiliza o regulador comutado de tensão LM2576S-5.0 da National Semiconductor™. A fonte está definida para funcionar com uma tensão de entrada de +24 V e fornecer uma corrente de 2 A.

O projeto de implementação segue as indicações do fabricante, sendo semelhante ao apresentado para a fonte de alimentação da "*Caixa de distribuição*" (subsecção 5.4.3.3) e [Anexo A]. O seu esquema, apresentado na Figura 59, mostra a fonte comutada formada pelo regulador de tensão U1, o condensador de entrada C3, o circuito de saída D49, L1 e C4. Para reduzir a dissipação do circuito de controlo de corrente constante dos LED's, foi implementado o regulador LDO LD1085D2M25R para 2,5 V (VLED) que só requer um condensador de saída de 22 µF (C2).

Os condensadores de desacoplamento (C6…C21) são instalados à razão de um por cada CI digital utilizado, e colocados o mais próximo possível do respetivo pino de alimentação.

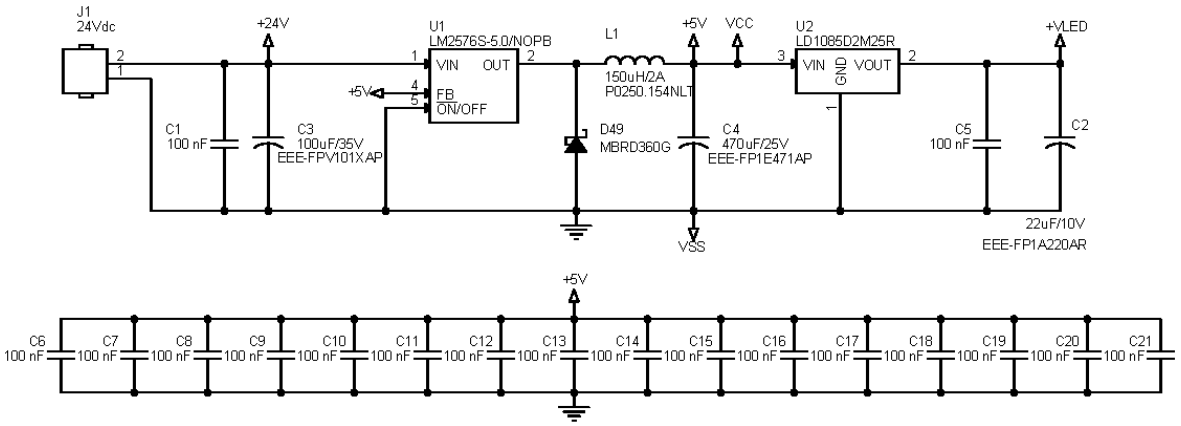


Figura 59 Sistema de medição – Fonte de alimentação

5.7.3.3. CIRCUITO DE CONTROLO DOS LED'S

O circuito de controlo dos LED's (Figura 60) tem por base o CI STP16CP05 da ST Microelectronics™ [Anexo B – VII]. Os CI's U4, U5 e U6 formam um registo de deslocamento de 48 *bits* com entrada no pino 2 SDI do IC4, proveniente da saída SDO de controlo SPI.

Os restantes sinais (SCK, LE e OE) são comuns aos três CI's. Devido à utilização simultânea do mesmo interface SPI para a leitura dos registos de deslocamento dos fotorreceptores, a saída SDO de U3 não é utilizada. As resistências R1…R3 programam uma corrente aproximada de 80 mA por cada LED.

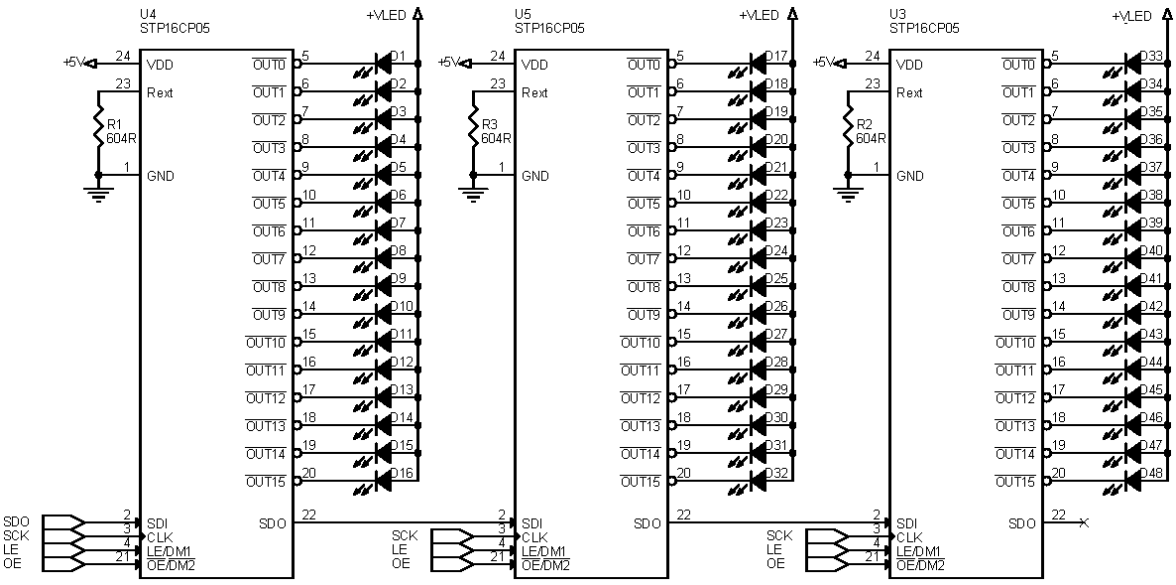


Figura 60 Sistema de medição – Circuito de controlo dos LED's

Devido ao controlador dos LED's ter uma fonte de corrente constante por cada LED, não são utilizadas resistências limitadoras individuais. A tensão de alimentação VLED é de 2,5 V e é proveniente do regulador de tensão U2.

5.7.3.4. CIRCUITO DE ENTRADA DOS FOTORRECEPTORES

O circuito de entrada dos fotorreceptores é formado pelos registos de deslocamento U8…U19 [Anexo B – VIII]. Para simplificação do esquema apresentado na Figura 61 estão apenas representados os dois primeiros e os dois últimos integrados que formam o registo de deslocamento de 96 *bits* do circuito de entrada dos fotorreceptores.

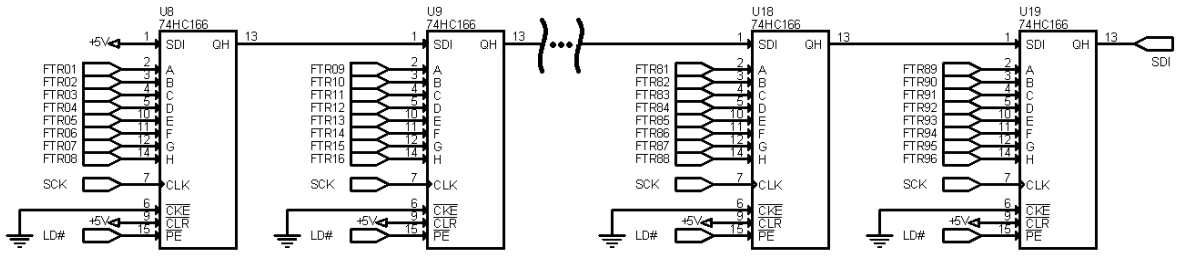


Figura 61 Sistema de medição – Circuito de entrada dos fotorreceptores

Os sinais FTR01…FTR96 são ligados ao circuito de saída do fotorreceptor. Os sinais SCK (relógio do interface SPI), LD# (sinal de permissão para a carga paralela dos registos de deslocamento) e

SDI (entrada do interface SPI no controlador PIC) são os únicos necessários para efetuar a leitura dos 96 andares de entrada. O circuito de saída do fotorreceptor (Figura 62) foi reduzido à representação dos primeiros e últimos quatro receptores do total de 96. A resistência de coletor de 68 kΩ define o ganho do foto-transístor e foi determinada em função da distância necessária a ser percorrida pela luz do LED emissor (aproximadamente 300 mm).

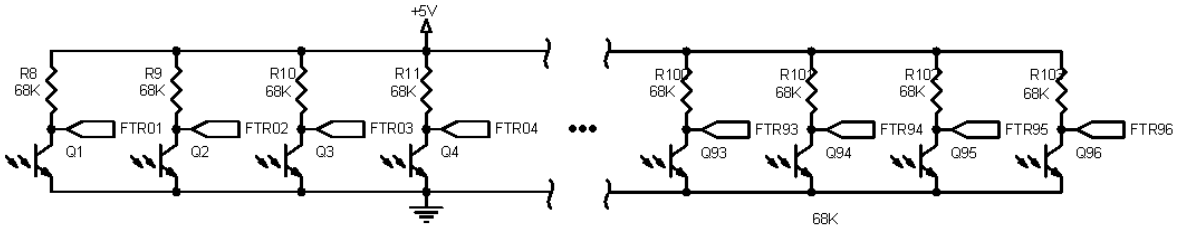


Figura 62 Sistema de medição – Fotorreceptores

5.7.3.5. CIRCUITO IMPRESSO

A implementação física do circuito de medição foi condicionada pela especificação de resolução da leitura (2 mm). Para o espaçamento entre centros dos receptores ser de 2 mm, foi necessário desfasar os receptores ímpares, dos pares, de forma a poder acomodar o fotorreceptor, mantendo a distância entre eixos desejada. As duas imagens superiores da Figura 63 apresentam o desenho do circuito impresso, lado superior e inferior do cobre, respetivamente, e nas duas seguintes, é apresentada a respetiva implementação de componentes. Finalmente, a foto inferior apresenta a montagem final do sistema de medição.

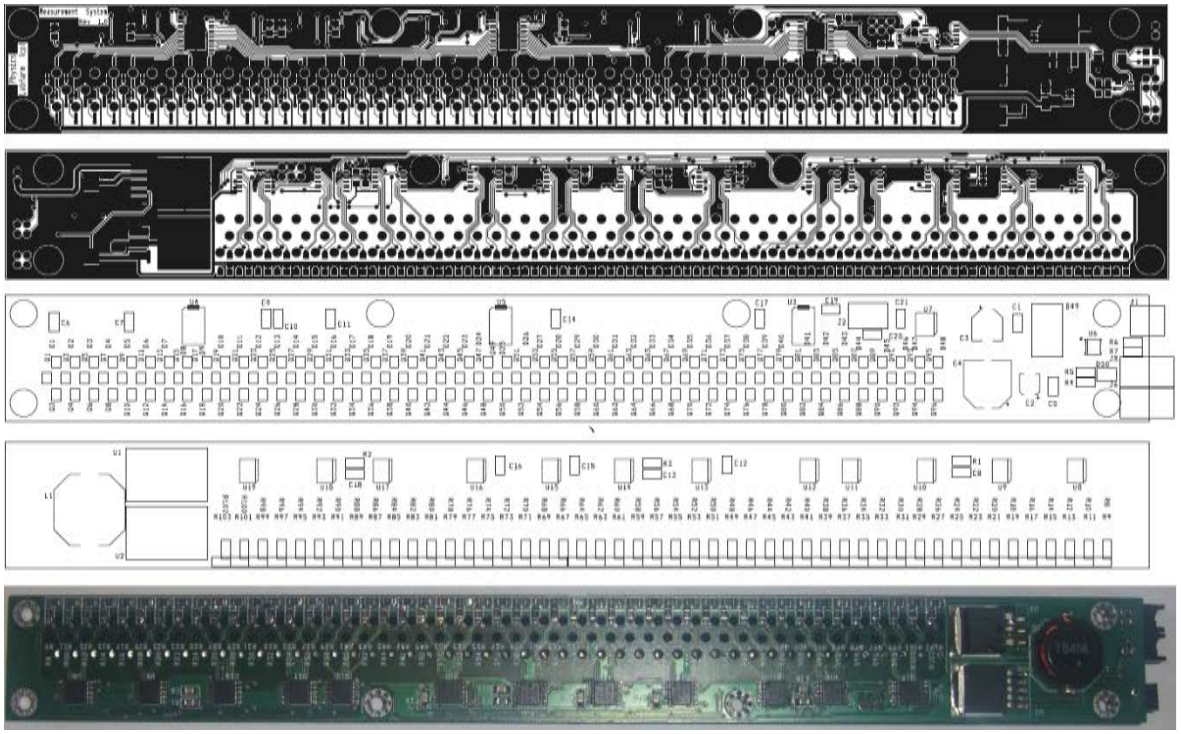


Figura 63 Sistema de medição – Circuito impresso

A face oposta da montagem final não é apresentada, em virtude de estar coberta com um verniz preto, para impedir a propagação dos raios infravermelhos entre emissores e receptores, dada a sua proximidade, pelo que não são visíveis os seus componentes.

5.8. ALIMENTAÇÃO E INTERLIGAÇÃO GERAL

Esta secção apresenta o projeto elétrico do aparato. As subsecções seguintes mostram a distribuição de 230 V_{AC} com o esquema da zona inferior do aparato, os esquemas elétricos dos subconjuntos "*Elevador de bola*" e "*Elevador da rampa de lançamento*" na zona superior.

5.8.1. ESQUEMA ELÉTRICO DA ZONA INFERIOR DO APARATO

O sistema é alimentado a partir da rede de 230 V_{AC}. Para a sua ligação, disponibiliza um módulo com entrada para cabo de potência, interruptor, fusíveis e filtro (modelo Schaffner™ FN282). Internamente é distribuído, a partir de bornes de calha DIN, para os módulos a alimentar. A Figura 64 apresenta o esquema da zona inferior do aparato que contém a distribuição de 230 V_{AC}.

Foi instalada uma ventoinha de alto fluxo de ar de 230 V_{AC}, para ventilação do compartimento inferior da estrutura, onde se encontra o servidor. Junto da saída da fonte comutada de 24 V_{DC}, foi instalado o condensador eletrolítico de 4700 µF / 63 V, recomendado pelo fabricante dos controladores dos motores passo-a-passo.

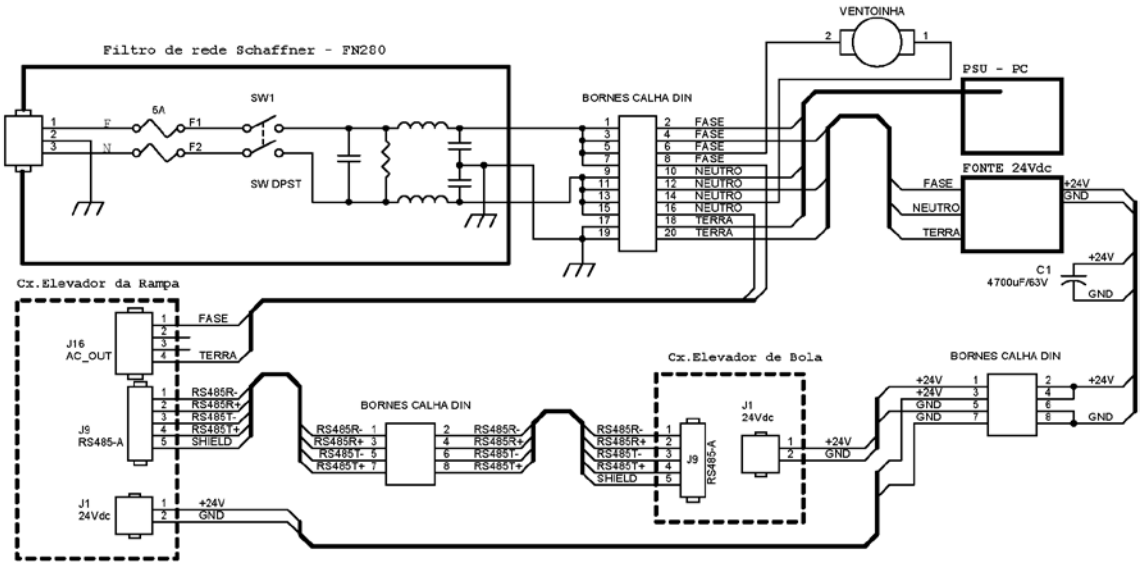


Figura 64 Esquema da zona inferior do aparato

A interligação dos vários sistemas é feita via ligadores de calha DIN, onde se encontra a distribuição de 230 V_{AC}, 24 V_{DC} e interligação da rede RS-485. A alimentação do sistema de controlo do aparato provém de uma fonte comutada que fornece a tensão de 24 V_{DC} / 5 A. Foi escolhida a fonte MEAN WELL™ NTS-24V-5A. A cablagem das interligações de rede RS-485 foi

executada com o cabo IGUS™ Chainflex™ CF11.05.02.02 (2 pares torcidos de 0,5 mm^2 de secção com malha de blindagem, para calha flexível). Para as ligações de potência, foi selecionada a gama Chainflex™ CF130 de 2 e 3 condutores com 0,75 mm^2 de secção (respetivamente CF130.07.02 e CF130.07.03).

5.8.2. ESQUEMA ELÉTRICO DA ZONA SUPERIOR DO APARATO

A Figura 65 apresenta o diagrama elétrico do subconjunto "*Elevador de bola*". A ligação de alimentação, proveniente da fonte de alimentação externa de +24 V, é ligada ao conector J1 da "*Caixa de distribuição*", de onde sai a ligação de rede RS-485 sai do conector J9 com destino à "*Caixa de distribuição*" do subconjunto "Elevador da rampa".

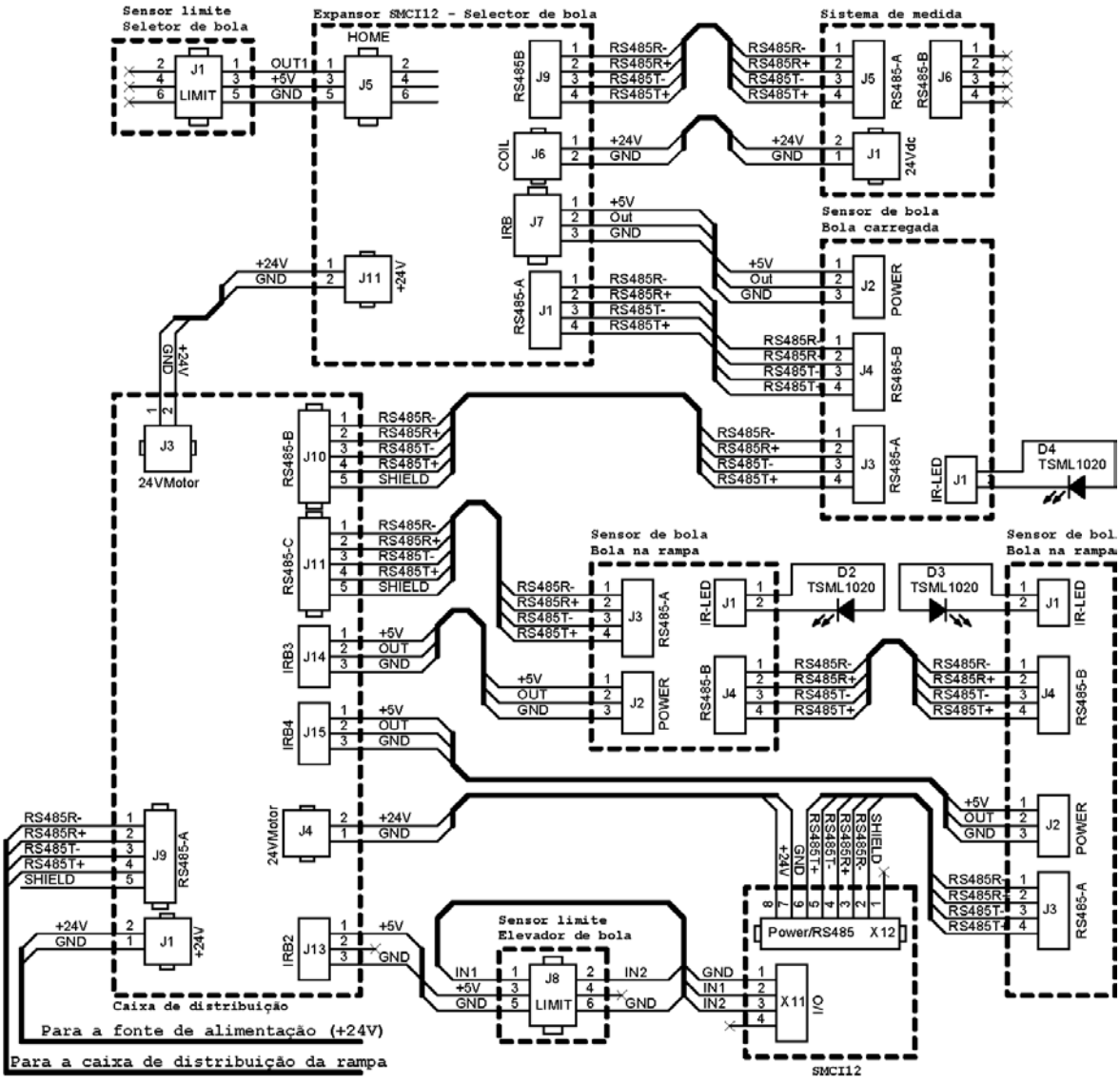


Figura 65 Esquema elétrico do subconjunto "*Elevador de bola*"

A Figura 66 apresenta o diagrama elétrico do subconjunto "*Elevador de bola*". A ligação de alimentação, proveniente da fonte de alimentação externa de +24 V, é ligada ao conector J1 da

"*Caixa de distribuição*", de onde sai a ligação de rede RS-485 com origem no conector, com destino à "*Caixa de distribuição*" do subconjunto "*Elevador de bola*". O solenoide de bloqueio da bola na rampa de lançamento é alimentado a 230 V_{AC} proveniente da distribuição AC.

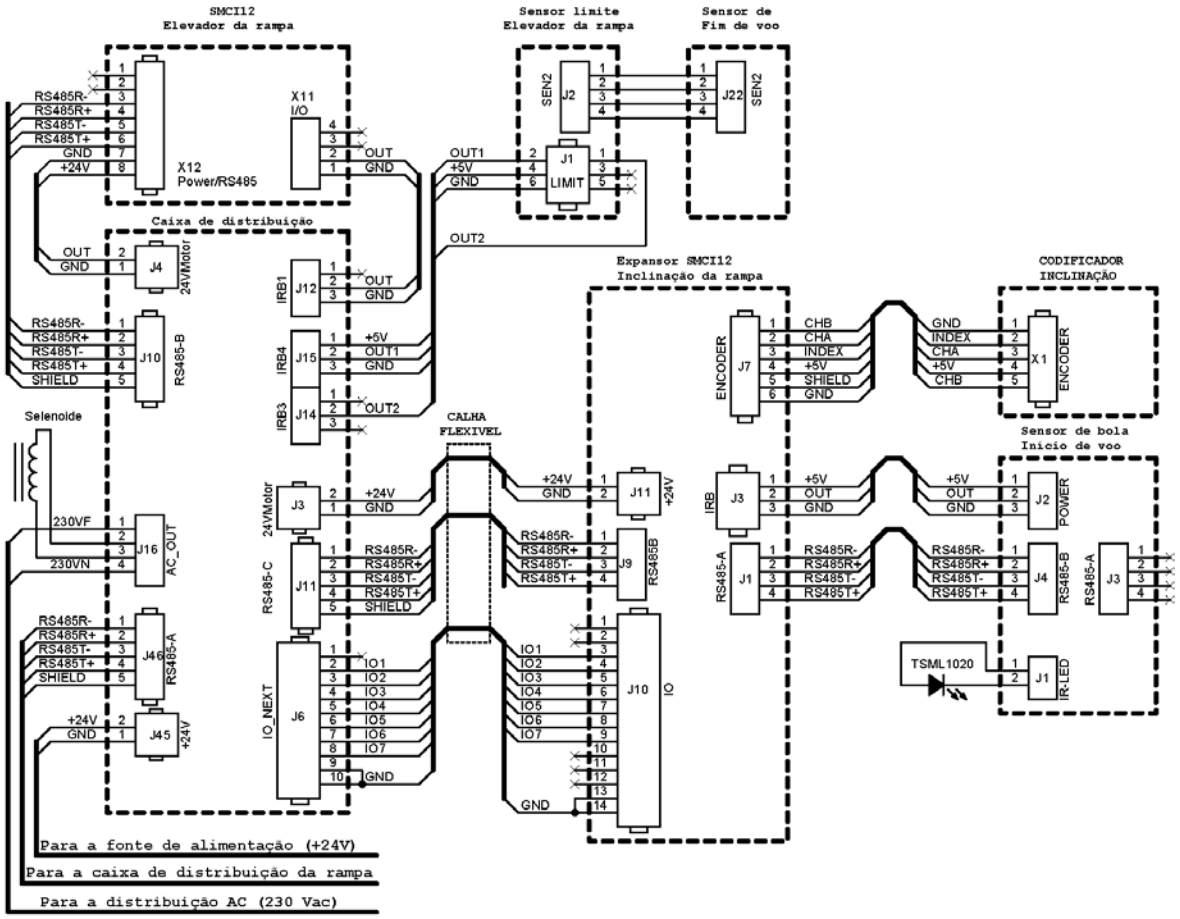


Figura 66 Esquema elétrico do subconjunto "*Elevador da rampa de lançamento*"

5.9. CONCLUSÃO

Este capítulo apresentou o diagrama geral da implementação, que dividiu o projeto nos módulos eletrónicos "*Sensor de bola*", "*Caixa de distribuição*", "*Sistema de expansão SMCI12*" e "*Sistema de medição do alcance horizontal do projétil*". Para cada um deles, foi apresentado o diagrama de blocos, uma descrição do seu funcionamento e finalizando com a implementação física. O capítulo finalizou com a apresentação do projeto elétrico de interligação de todas as partes constituintes. O capítulo seguinte apresenta os vários níveis de controlo necessários ao funcionamento do aparato.

Capítulo 6

PROJETO LÓGICO

A divisão do aparato nos conjuntos mecânicos "*Seletor de bola*", "*Elevador de bola*", "*Elevador da rampa de lançamento*", e "*Dispositivo de recolha*", implicou que o projeto eletrónico, apresentado no capítulo anterior, implementasse os módulos eletrónicos "*Sensor de bola*", "*Caixa de distribuição*", "*Sistema de expansão SMCI12*" e "*Sistema de medição do alcance horizontal do projétil*". Neste capítulo, a primeira secção apresenta o diagrama de controlo lógico, e nas secções seguintes são apresentadas as suas componentes de alto e baixo nível.

6.1. DIAGRAMA DO CONTROLO LÓGICO

O sistema de controlo de um aparato, manipulável remotamente, requer a implementação de vários sistemas de acesso. É necessária uma porta de entrada dos pedidos funcionais remotos e um meio local de controlo primário, com interface de utilizador, que permita a manipulação do aparato em situações de reparação e/ou manutenção. Definiu-se a utilização de serviços web [Anexo D] que devem ser chamados pelo servidor web de experiência, desenvolvido numa qualquer tecnologia web, com possibilidade de chamar serviços web, i.e. PHP (*Hypertext Preprocessor*), cujo desenvolvimento e implementação não estão no âmbito desta tese. Estes serviços web foram projetados de uma forma não bloqueante. O servidor web faz um pedido, e em função do estado da máquina, recebe imediatamente uma resposta que reflete o estado do aparato:

- Em progresso – pedido de execução aceite e iniciado;
- Ocupada – pedido rejeitado. Está uma experiência em curso;
- Erro – pedido rejeitado. O aparato está indisponível;

Posteriormente, o servidor pergunta, num outro método do serviço web, se a experiência está finalizada, e nesse caso, recebe os resultados obtidos. O controlador de experiência implementa a máquina de estados do aparato e disponibiliza os métodos necessários de controlo de alto nível. Aqui estão definidos métodos que interagem com os dispositivos eletrónicos de controlo via rede RS-485. A camada de interface à rede RS-485 trata da implementação do protocolo série, que tem como base o protocolo de comunicação dos controladores SMCI12, e da sua transmissão/recepção via porta série.

Ao nível da camada física foi desenvolvida uma estrutura, comum a todos os dispositivos da rede RS-485, que implementa o interface série e respetivo protocolo. Adicionalmente, cada um dos microcontroladores PIC contém uma série de funções específicas de controlo da funcionalidade requerida pelo dispositivo em causa. A Figura 67 apresenta o diagrama geral do controlo lógico do aparato. O "*Controlo de experiência*" contém os métodos para iniciar e retornar os resultados de uma experiencia, bem como a sua máquina de estados. Este controlo define dois modos de funcionamento:

- Modo local – O "*Interface do utilizador*" controla cada um dos subsistemas, de forma manual, para a calibração e reparação do aparato, via "*Controlo funcional*";
- Modo remoto – O "*Controlo de experiência*" espera um pedido proveniente da rede exterior via serviço web, executa a experiência e retorna os resultados obtidos;

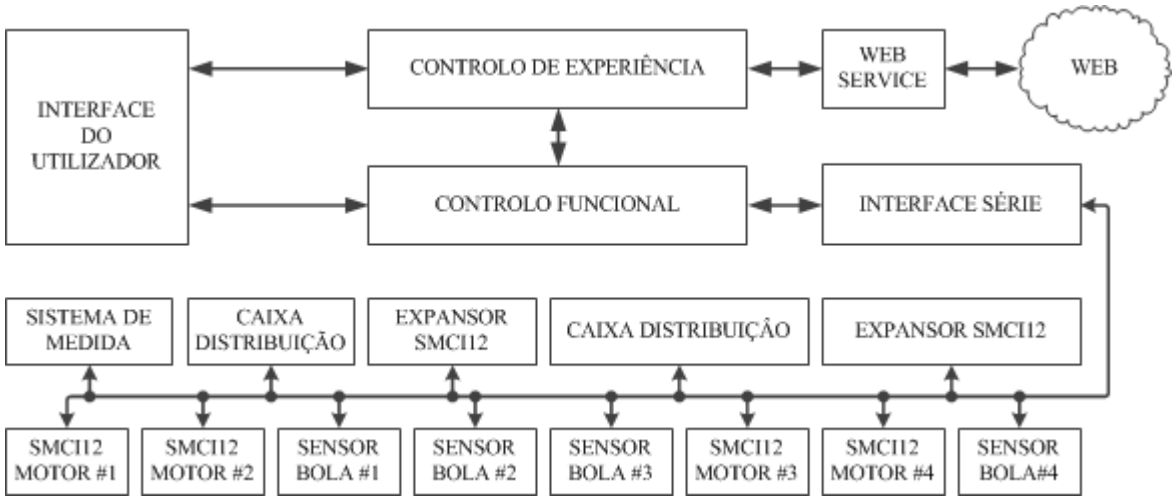


Figura 67 Diagrama geral do controlo lógico

A comunicação entre a camada de controlo superior, residente no sistema operativo Windows™, e o controlo da camada de baixo nível, residente nos vários microcontroladores enumerados no

diagrama de controlo, é feita via "*Interface série*" que utiliza o meio físico RS-485 para a comunicação. O controlo de alto nível utiliza as estruturas de *software* "*Windows Presentation Foundation*" (WPF) [Anexo C-V] e "*Windows Communication Foundation*" (WCF) [Anexo C-IV] com implementação em C# [Anexo C-III]. Para o seu desenvolvimento utilizou-se o IDE ("*Integrated Development Environment*") da Microsoft™ "*Visual Studio 2010*" [Anexo C-I]. Com a exceção dos controladores SMCI12, que dispõem de um controlo pré-programado pelo fabricante, todos os periféricos utilizam microcontroladores da família PIC16F1xxx cujo programa foi implementado em C. Para o seu desenvolvimento utilizou-se o IDE da Microchip™ "*MPLabX*" e o compilador de C da HI-TECH™.

6.2. CONTROLO DE ALTO NÍVEL

Esta secção detalha a implementação do código executado na plataforma Windows™ [Anexo F]. A aplicação usa uma metodologia OOP ("*Object Oriented Programming*") e está dividida em várias classes e interfaces, que agregam, de forma concisa, as funcionalidades implementadas. O diagrama de classes apresentado na Figura 68 mostra as classes primárias da aplicação ("*UI*", "`RemoteLab`" e "`LabfarmWS`"), o interface de implementação "`ILabfarmWS`", e as classes de dados utilizadas no pedido e resposta do serviço web ("`UserVariableObj`" e "`SystemVariableObj`").

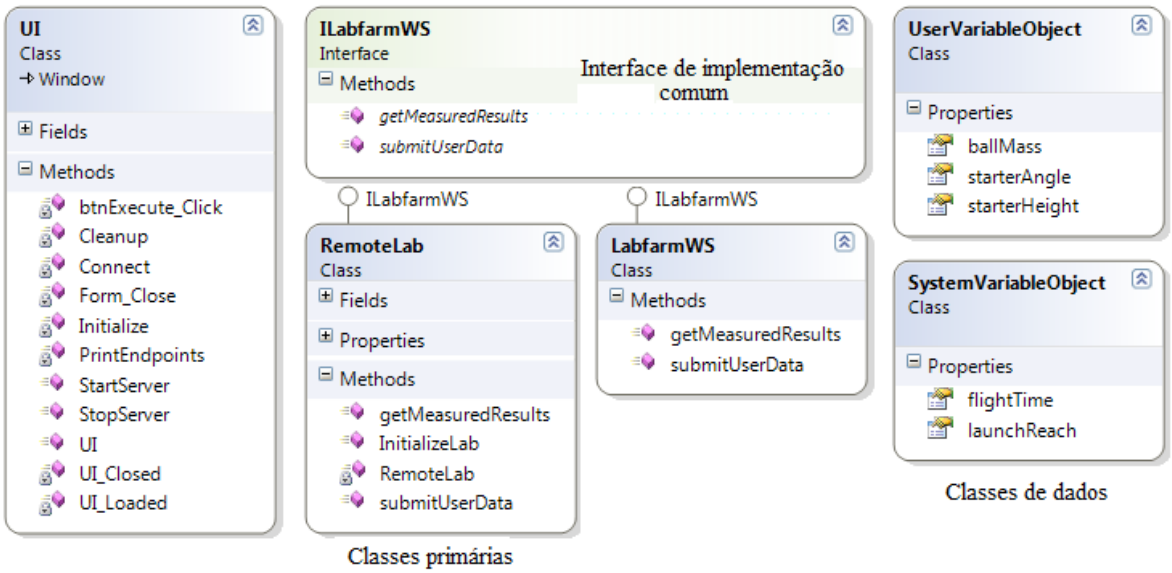


Figura 68 Diagrama de classes – Elementos principais

A classe de entrada "`App`" instancia e apresenta o interface de utilizador "`UI`". Após a carga e inicialização deste interface, é instanciado e iniciado o serviço web [Anexo D – I] "`LabfarmWS`". Este serviço recebe como parâmetro a classe de contrato de dados "`UserVariableObject`" (com

atributo "*DataContract*"), e devolve o resultado da experiência via classe de contrato de dados "*SystemVariableObject*". Estas classes de contrato de dados são os conteúdos das mensagens SOAP ("*Simple Object Access Protocol*") [Anexo D – II] trocadas com o servidor web de experiência, via serviço web. A classe "*RemoteLab*" cria uma instância estática única da experiência, à qual fornece o acesso via interface "*ILabfarmWS*", comum ao serviço web. A Figura 69 apresenta o diagrama de classes do interface ao aparato, que controlam o fluxo da experiência. A classe "*Experiment01*" implementa os comandos de alto nível necessários ao controlo do aparato, que herda a classe "*NanotecProtocol*" via "*NanotecDataContext*". A classe "*NanotecDataContext*" disponibiliza o registo das comunicações (enviadas e recebidas) a serem utilizados no interface de utilizador.

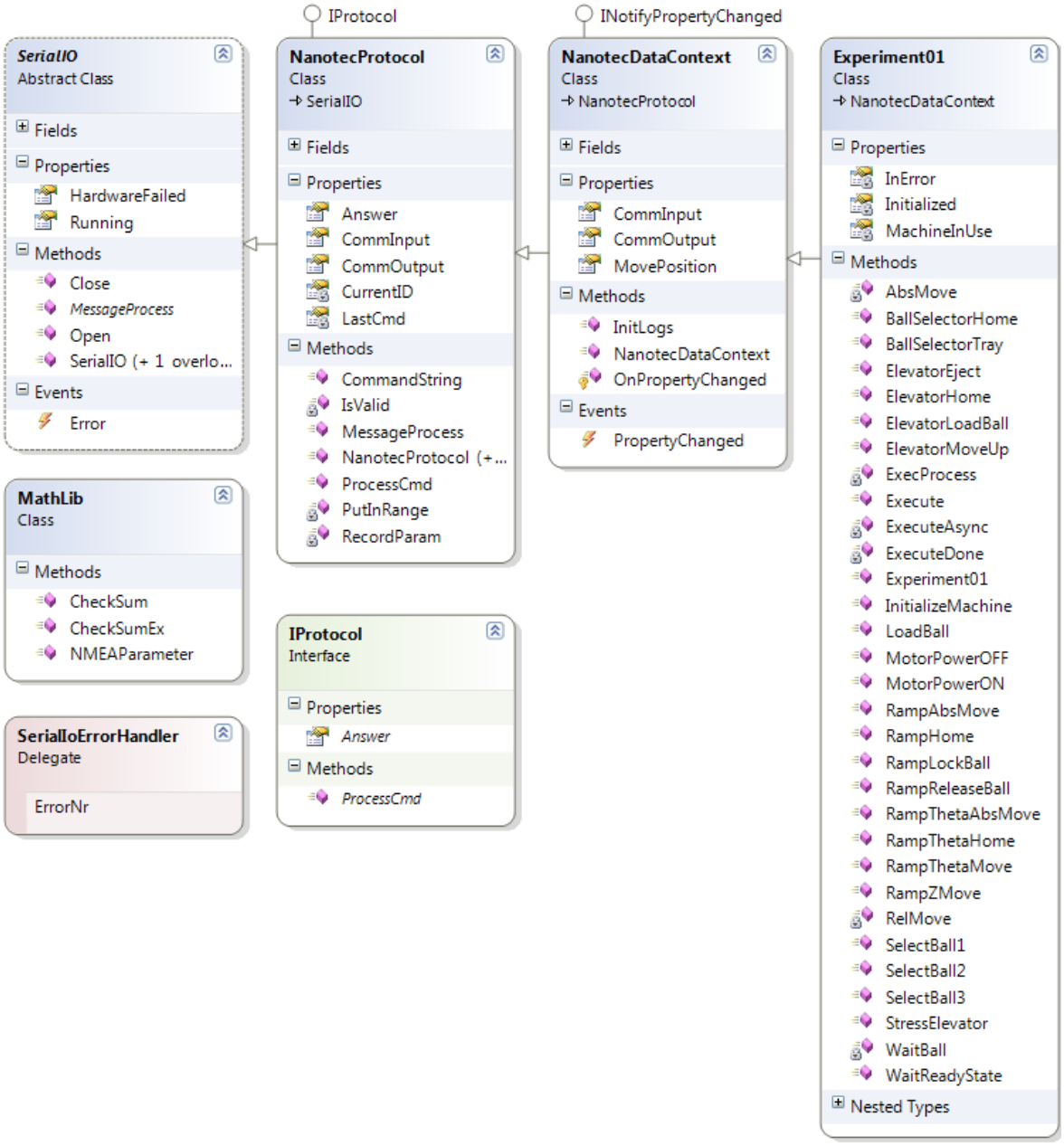


Figura 69 Diagrama de classes – Interface ao aparato

O protocolo de comunicação é implementado na classe "`NanotecProtocol`" cujos conteúdos são transmitidos à porta de comunicação série via classe base "`SerialIO`". A classe "`MathLib`" contém métodos auxiliares utilizados no tratamento das mensagens.

6.2.1. APLICAÇÃO PRINCIPAL E INTERFACE DE UTILIZADOR

A aplicação principal ("`App`") é o ponto de entrada de execução, e a sua finalidade é a instanciação do interface de utilizador ("`UI`"), com o respetivo tratamento de exceções de nível global. O seu código de instanciação está inserido num bloco "`try/catch`" que visa capturar as exceções cujo tratamento não está implementado, por não ser importante o seu tratamento e poderem ser descartadas. O ficheiro "*App.xaml*" define um dicionário com recursos utilizados no desenho do interface de utilizador, que estão numa DLL separada ("*AppSkin.dll*"), cujo código não está detalhado no âmbito desta tese. Os manipuladores de eventos (*handlers*) implementados disponibilizam um tratamento centralizado às exceções não tratadas de WPF.

A classe de interface ao utilizador está implementada em WPF, cuja vista geral é apresentada na Figura 70. Este interface foi desenhado com a finalidade de ajudar ao desenvolvimento e manutenção do aparato. Estão disponibilizadas duas listas ("Dados recebidos" e "Comandos enviados") onde é feito o registo de toda a informação série comunicada entre o controlador de experiência e a rede RS-485. Antes de ser possível qualquer interação com o aparato, é necessário a aplicação ligar-se a uma porta COM existente (instalada automaticamente quando é ligado o cabo USB à "*Caixa de distribuição*" do "*Elevador de bola*"), via botão "*Connect*". Por omissão, está especificada a COM1, que pode ser alterada na caixa de texto.

Figura 70 Interface de utilizador

Na parte superior do interface encontram-se os botões de inicialização ("*Init Seletor de bolas*", "*Init RampZ*" e "*Init Elevador*") que tem por finalidade referenciar a origem dos movimentos das estruturas mecânicas associadas ("*Seletor de bola*", "*Elevador da rampa de lançamento*" e "*Elevador de bola*", respetivamente). O botão "*Inicialização*" executa as três inicializações e é o primeiro método executado no arranque do aparato.

Após a inicialização, é possível controlar, de forma independente, cada um dos movimentos. Os botões "*Bola 1*", "*Bola 2*" e "*Bola 3*" movimentam o seletor de bola, de modo a selecionar cada uma das bolas disponíveis. O botão "*Ciclo Elevador*" efetua o ciclo completo de "*Elevador de bola*", movendo-o para a posição de carga, elevando a bola até à posição superior e regressando à posição de repouso. Ao longo deste ciclo são verificados os vários sensores de deteção de presença de bola. Os botões "*RampMove*" e "*ThetaMove*" utilizam o valor introduzido na caixa de texto associada como posição absoluta de destino (respetivamente '*altura da rampa*' e '*ângulo da rampa*'). Estes valores estão limitados ao intervalo de valores fisicamente aceitáveis pelo aparato (altura da rampa: [53, 348] mm, ângulo: [-15,+15] °).

Os botões "*Power ON*" e "*Power OFF*" efetuam o controlo manual da alimentação dos motores, via SSR das caixas de distribuição. Os botões "*Lock Ball*" e "*Release Ball*" controlam o funcionamento manual do retentor de bola. O botão "*Stress*" põe o "*Elevador de bola*" a executar 10 vezes o seu ciclo completo, com a finalidade de verificar falhas provocadas por aquecimento.

6.2.2. SERVIÇO WEB

O serviço web [Anexo D – I] tem por base a infraestrutura WCF [Anexo C – IV], e é instanciado pela classe "`UI`", após a sua carga em memória (evento "`UI_Loaded`"), via método "`UI.StartServer`". O servidor de serviço web ("`ServiceHost`") é criado com base na classe "`LabfarmWS`". Esta classe implementa a interface "`ILabfarmWS`" que contém a declaração dos métodos necessários à troca de informação com o servidor web. O método "`submitUserData`" recebe como parâmetro, a classe "`UserVariableObject`", que implementa os membros "`ballMass`", "`starterheight`" e "`starterAngle`". Estes membros, do tipo propriedade, refletem os parâmetros de entrada, respetivamente "*número da bola*", "*altura da rampa de lançamento*" e "*ângulo de lançamento*", definidos pelo utilizador para a experiência a iniciar. Estes parâmetros são passados ao método "`Experiment.Execute`" para iniciar uma nova execução da experiência.

O servidor web utiliza o método "`getMeasureResults`" para obter os resultados da experiência, via classe "`SystemVariableObject`". Esta classe implementa os membros "`flightTime`" e

"*launchReach*" onde são guardados respetivamente os resultados '*tempo de voo*' e '*alcance do projétil*'.

A classe de implementação dos serviços web ("*LabfarmWS*") serve unicamente de interface externo ao controlador de experiência. Assim, os métodos aqui implementados ("*submitUserData*" e "*getMeasuredResults*") são mapeados nos métodos de execução ("*Execute*" e "*ExperimentResult*") do controlador de experiência.

A configuração de acesso ao serviço web está definida no ficheiro "app.config". Este ficheiro, em XML (*eXtensible Markup Language*) define os seguintes parâmetros do serviço web:

- Protocolo de ligação (ServiceModel.Bindings) do tipo "basicHttp";
- Endpoint – "basicHttpBinding" com o contrato "ILabfarmWS";
- Endereço de acesso – http://172.18.57.110:50000/Labfarm;

6.2.3. INTERFACE SÉRIE

O interface série está implementado numa classe base "*SerialIO*" que deve ser herdada para poder ser utilizada (a classe está definida como "`abstract`"). Esta classe instancia o recurso série "*SerialPort*", definido na biblioteca de sistema "`IO.Ports`", e permite a abertura opcional do porto durante a instanciação, efetuada no método "`Open`". Este método, após a instanciação da classe "*SerialPort*" (membro público "*PortRS*"), tenta abrir o porto com o nome passado em parâmetro. Caso a abertura seja bem sucedida, é iniciada uma *thread* separada, de implementação externa (declarada como "`public abstract void MessageProcess();`"), onde é processado o ciclo de verificação de mensagens recebidas.

No caso de erro na abertura do porto (e.g. porto inválido), a execução salta para a cláusula de tratamento de erro ("*catch*"), que sinaliza internamente com a *flag* "*HardwareFailed*" e informa o utilizador com a caixa de diálogo, onde é colocada a mensagem do erro capturado. A classe disponibiliza o método "*Close*", que deve ser chamado no encerramento da aplicação, para que a *thread* de recepção de mensagens possa ser finalizada. O estado da *thread* pode ser verificado pela propriedade pública "*Running*".

6.2.4. IMPLEMENTAÇÃO DO PROTOCOLO DE COMUNICAÇÃO

Para o controlo do aparato, utilizou-se um protocolo baseado em texto que está implementado nos controladores SMCI12 da Nanotec™. O manual de programação da Nanotec™ apresenta a estrutura de comandos do protocolo, descrita sucintamente em seguida:

- Um comando inicia com o caracter '#';
- Segue-se o endereço do nó da rede ('1' a '254'). Se for utilizado o asterisco '*', define uma mensagem de *broadcast* a ser recebida por todos os nós ligados na rede;
- Segue-se o comando composto por um caracter ASCII e um número ASCII opcional (escrito em notação decimal e com os prefixos '+' e '-'). No caso do envio de parâmetros, o prefixo '+' é opcional;
- Termina com um "*carriage return*" '\r';
- Todos os caracteres entre o início e fim de comando são do tipo ASCII;

A resposta do controlador deve ser o eco do comando recebido, sem o caracter inicial '#'. No caso de o comando ser desconhecido, o eco enviado tem como sufixo uma interrogação '?'. Exemplo:

- Comando válido: "#1G10000000\r" → "1G10000000\r";
- Comando inválido: "#1°\r" → "1°?\r";

Este protocolo define um segundo formato para comandos longos (comandos formados por mais de um caracter). Este formato é semelhante ao anterior, com a exceção de que após o endereço do nó de destino, o comando é precedido de ':' e seguido do texto do comando (pode conter os caracteres 'A' a 'Z' ou 'a' a 'z' e o caracter '_'). A sintaxe é sensível a maiúsculas e minúsculas e não pode conter dígitos. Exemplos:

- Ler parâmetro: "#1:CL_motor_pp\r" → "1:CL_motor_pp+50\r";
- Modificar parâmetro: "#1:CL_motor_pp=100\r" → "1:CL_motor_pp=100\r";

A implantação deste protocolo está residente na classe "*NanotecProtocol*", que herda a classe de interface série "*SerialIO*". O construtor[5] pode opcionalmente abrir a porta série com os parâmetros passados. Esta classe define o método estático:

```
static public string CommandString(CmdID cmd){…}
```

onde a enumeração de comandos "*CmdID*" é convertida na *string* associada. O método "*ProcessCmd*" recebe o endereço do nó de destino (enumeração "*Devices*"), o identificador do comando (enumeração "*CmdID*"), um sinalizador que indica se é uma leitura, e uma lista opcional de parâmetros eventualmente necessária para alguns comandos. Com estes parâmetros é criada a mensagem a enviar para o porto série, que simultaneamente é adicionada ao registo de comunicações enviadas ("*CommOutput*"). Após uma sinalização à *thread* de recepção, fica à espera da mensagem de resposta, que retorna à função que originou o pedido.

A enumeração "*CmdID*" contém todos os comandos necessários ao controlador SMCI12 e os comandos adicionais, implementados nos microcontroladores PIC. Em alguns dos comandos

[5] Um construtor, em programação C# é um método de inicialização cujo nome é igual ao da classe e é chamado automaticamente no momento de criação do objeto.

implementados (e.g. '*read*') foram utilizados os mesmos caracteres do controlador SMCI12, cuja funcionalidade é semelhante.

Os métodos "*CommInput*" e "*CommOutput*" são implementados na classe "*NanotecDataContext*", destinada exclusivamente à ligação, via "*data binding*", com o interface de utilizador. Esta classe herda a classe "*NanotecProtocol*", à qual é adicionado o interface de notificação "*INotifyPropertyChanged*" que declara o evento de notificação à camada superior WPF ("*PropertyChanged*").

6.2.5. CONTROLADOR DE EXPERIÊNCIA

A classe "*RemoteLab*" tem a finalidade de instanciar uma única vez o controlador de experiência ("Experiment01"). Este modelo de instanciação é conhecido como "*Singleton Pattern*" e está incluído num conjunto de modelos e boas práticas (i.e. aumentar o nível de reutilização). O modelo "*Singleton*" destina-se a restringir a instanciação de uma dada classe, cujo acesso é global. Tipicamente, quando é necessário um acesso exclusivo a um qualquer recurso, o seu controlador não pode ser instanciado mais de uma vez. Uma referência global de um objeto resolve a necessidade de acesso centralizado, mas não impede que o mesmo seja instanciado múltiplas vezes. Uma melhor solução é a própria classe se certificar da existência de uma única instância, ao intercetar um pedido de criação de uma nova instância [21].

Para a implementação do modelo "*Singleton*" em C# foi utilizada a metodologia apresentada pela Microsoft™ [22], que embora siga a solução apresentada em [21], foi modificada de forma a tirar partido das características específicas da linguagem C#. O extrato de código seguinte mostra a implementação "*Singleton*" da classe "*RemoteLab*".

```
public sealed class RemoteLab : ILabfarmWS
{
  public Experiment01 Experiment;
  private static volatile RemoteLab instance;
  private static object syncRoot = new object();
  private RemoteLab() { Experiment = new Experiment01()}
  public static RemoteLab Instance
  {
    get
    {
      if (instance == null)
      {
        lock (syncRoot);
        {
          if (instance == null) instance = new RemoteLab();
        }
      }
    }
  }
}
```

Uma vez instanciado, o "*RemoteLab*" fornece um acesso único e global à classe "*Experiment01*", que implementa os métodos e propriedades necessárias ao controlo lógico da experiência de "*Lançamento de projéteis*". A existência de uma classe denominada "*RemoteLab*" é justificada pelo requisito inicial do projeto de enquadrar esta experiência num conjunto superior, denominado "*Physics LabFARM*". Desta forma, a classe implementaria uma "*fábrica de classes*" destinadas a fornecer, cada uma delas, um determinado tipo de experiência. Esse contexto ficou reduzido à experiência atual onde só está instanciada a primeira experiência "*Experiment01*", acedida pelo campo público "*Experiment*". A classe permite, ao ser instanciada, abrir opcionalmente o porto série, via construtor "`public Experiment01(string portName)`", ao qual é fornecido o texto de ligação (o nome do porto a ser utilizado, e.g. "COM1"). O porto série é configurado no método "*Connect*" para uma comunicação a 115200 baud, sem paridade, oito *bits* de dados, um *bit* de início e um *bit* de paragem. A classe define os seguintes métodos:

1. Métodos de controlo do "*Seletor de bola*":
   - SelectBall1…3 – Move carro do seletor para a primeira, segunda e terceira posição, respetivamente;
   - BallSelectorHome – Move carro do seletor para limite do movimento, retornando-o à primeira posição;
   - BallSelectorTray(*trayNr*) – Move carro para a posição indicada por '*trayNr*';

2. Métodos de controlo do "*Elevador de bola*":
   - ElevatorHome – Move elevador para o limite do movimento retornando-o à posição de repouso;
   - LoadBall – Carrega bola no elevador;
   - ElevatorMoveUp – Move elevador para posição superior;
   - ElevatorEject – Envia bola para rampa (ejeta bola) e retorna à posição de repouso;
   - ElevatorLoadBall – Efetua o ciclo do elevador (carga, elevação, ejeção e retorno à posição de repouso);
   - StressElevator – Efetua 10 vezes o ciclo completo do elevador;

3. Métodos de controlo do "*Elevador da rampa de lançamento*":
   - RampHome – Move rampa para o limite do movimento, retornando-a à posição de repouso;
   - RampZMove(*height*) – Move rampa verticalmente para a posição especificada por '*height*';
   - RampThetaHome – Inclina rampa para a posição de limite e retorna-a a 0°;
   - RampThetaMove(*angle*) – Inclina rampa para ângulo especificado por '*angle*';
   - RampLockBall – Liga retentor de bola;
   - RampReleaseBall – Desliga retentor de bola;

4. Métodos genéricos:
   - InicializeMachine – Inicializa todos os movimentos;
   - WaitReadyState(*Id*) – Espera conclusão do movimento do motor indicado por "*Id*";
   - MotorPowerON – Liga alimentação dos motores;
   - MotorPowerOFF – Desliga alimentação dos motores;
   - AbsMove(*device, steps, speed*) – Move motor indicado por '*device*', para a posição absoluta '*steps*' à velocidade '*speed*';
   - RelMove(*device, steps, speed*) – Move motor indicado por '*device*', para a posição relativa '*steps*' à velocidade '*speed*';
   - WaitBall(*sensorId, state*) – Espera que sensor '*sensorID*' esteja no estado '*state*';

5. Métodos de execução da experiência:
   - Execute(*height, angle, ball*) – Executa um ciclo de experiência, com a altura de rampa '*height*', com o ângulo da rampa '*angle*' e com a bola '*ball*';
   - ExecuteAsync(*height, angle, ball*) – Inicia execução assíncrona da experiência;
   - ExecProcess – Implementa ciclo de experiência;
   - ExecuteDone – Chamada pela thread de execução quando a experiência finaliza. Retorna os resultados da experiência;
   - GetFlightTime – Obtém '*tempo de voo*';
   - GetProjectileRange – Obtém '*alcance horizontal do projétil*';

De forma a não bloquear o pedido do serviço web, a execução da experiência é efetuada de forma assíncrona, numa *thread* separada (Figura 71).

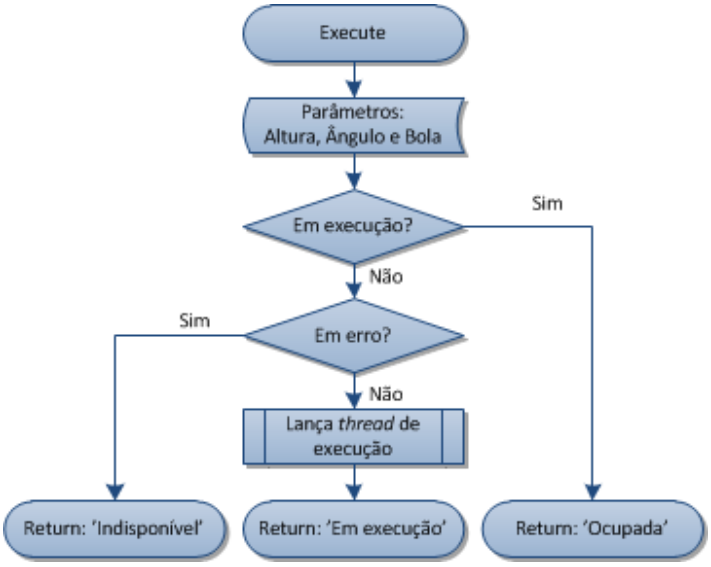


Figura 71 Fluxograma do método de execução assíncrona da experiência

A execução de uma experiência inicia com a verificação de necessidade de inicializar a máquina (é inicializada na primeira execução ou quando for detetado um erro de posicionamento num dos controladores dos motores). Seguidamente a bola selecionada é carregada e após a sua deteção no elevador, é elevada para o topo. Findo o trajeto do elevador, é confirmada a presença da rampa na posição de recolha e nesse caso o retentor é ligado e a bola é ejetada para a rampa. A rampa é movida para a altura e ângulo definidos. Após um intervalo de 1 segundo, é iniciado o lançamento do projétil, com retorno às posições de repouso do "*Elevador de bola*" e da rampa, colocando esta última na posição horizontal.

A Figura 72 mostra o fluxograma da *thread* de execução (implementada no método '`ExecProcess`') e da *thread* de resposta (implementada no método '`ExecuteDone`').

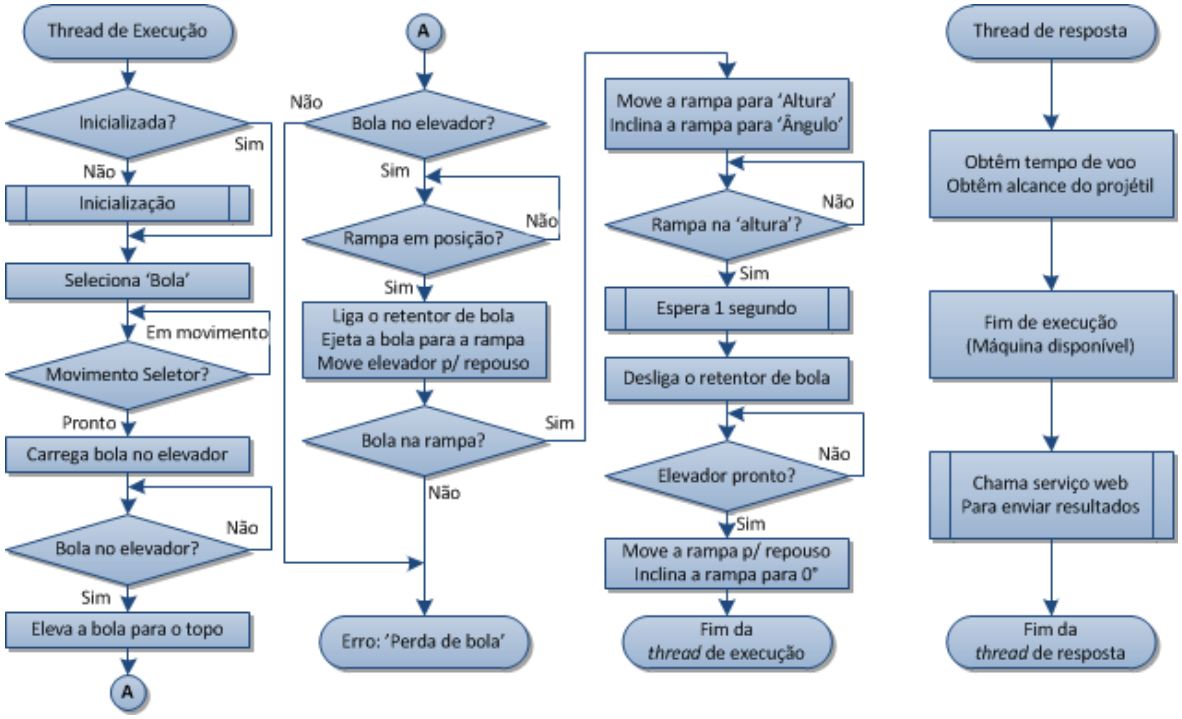


Figura 72 Fluxograma das *threads* de execução e de resposta

A *thread* de resposta, lançada automaticamente quando termina a *thread* de execução, obtém os dados do '*tempo de voo*' e do '*alcance horizontal do projétil*' e espera a chamada do servidor para enviar os resultados da experiência. Nessa altura o aparato passa ao estado de '*Disponível*'.

6.3. CONTROLO DE BAIXO NÍVEL

As aplicações de controlo foram desenvolvidas no IDE MPLAB-X da Microchip™, com recurso ao compilador HI-TECH para a família PIC16. Neste aparato, como foi apresentado no Capítulo 5, foram desenvolvidos vários circuitos eletrónicos com requisitos de controlo ("*Sensor de bola*", "*Sistema de expansão SMCI12*", "*Caixa de distribuição*" e "*Sistema de medição*"). Adicionalmen-

te, alguns destes módulos contêm implementações diferentes de código (e.g. a implementação de controlo da "*Caixa de distribuição*" do "*Elevador de bola*" é diferente da "*Caixa de distribuição*" do elevador da rampa), no entanto, a estrutura implementada em cada um dos microcontroladores é semelhante. A estrutura de implementação do código dos microcontroladores está em anexo [Anexo G].

6.3.1. ESTRUTURAS DE CONTROLO COMUM

Esta subsecção apresenta várias estruturas de código que são comuns a todas as implementações. Assim, o código de controlo da porta série (periférico USART), dos temporizadores, ciclos de inicialização e a rotina de entrada, com pequenas alterações, têm uma implementação semelhante, que está descrita em pormenor nos parágrafos seguintes. Os fluxogramas dos temporizadores e do ciclo principal apresentam, sob a forma de sub-rotina, um bloco de código denominado "*Processo específico de cada implementação*" com cor diferente, cujo conteúdo é detalhado nas subseções respetivas.

6.3.1.1. CONTROLO DA PORTA SÉRIE

Um dos requisitos, comum a todos estes módulos, é a necessidade de comunicação série, que de uma forma eficiente garanta a fiabilidade da comunicação enviada e recebida pela rede RS-485.

A implementação de controlo da porta série, (módulo "cpSerialAndTimer.c" e respetivo cabeçalho ".h"), utiliza um sistema de envio e recepção de dados por interrupções utilizando *buffers* circulares. Este tipo de implementação visa libertar, o mais rápido possível, as funções que necessitam de comunicar pela porta série. Um *buffer* circular ou em anel ("*Ring buffer*") é uma estrutura para guardar dados que utiliza um *array* de tamanho fixo, cujos extremos estão logicamente interligados.

Para o seu funcionamento, são necessários dois apontadores, um que indica a posição de escrita e outro que indica a posição de leitura. Antes da operação de leitura, se ambos os apontadores indicam a mesma posição do *array*, significa que não existem dados válidos para ler (a estrutura está vazia). Se antes da operação de escrita o apontador de escrita for igual ao apontador de leitura - 1, significa que a estrutura está cheia, e não existe espaço para escrita. A Figura 73 mostra o diagrama de um *buffer* circular com 16 posições em que quatro estão ocupadas. Os 12 *bytes* entre o apontador de escrita e o de leitura não são válidos.

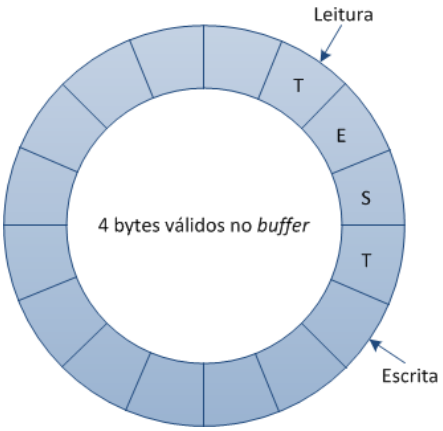


Figura 73 Diagrama de um exemplo de *buffer* circular

Para o ciclo de controlo primário poder utilizar a porta série, é necessário proceder à sua configuração. Este módulo define a função "`InitUSART`" onde o periférico USART é configurado para 115200 baud e 8 *bits* sem paridade, e ativado para envio e recepção. Nesta função são também inicializados os apontadores de recepção ("`RXBufIn`" e "`RXBufOut`") e de envio ("`TXBufIn`" e "`TXBufOut`"), o que equivale a limpar os *buffers* circulares.

Na família PIC16F só existe um vetor para as interrupções geradas pelo microcontrolador, no qual estas são distinguidas com recurso a sinalizadores que definem a origem da interrupção. A Figura 74 mostra o fluxograma do bloco relativo às interrupções da porta série.

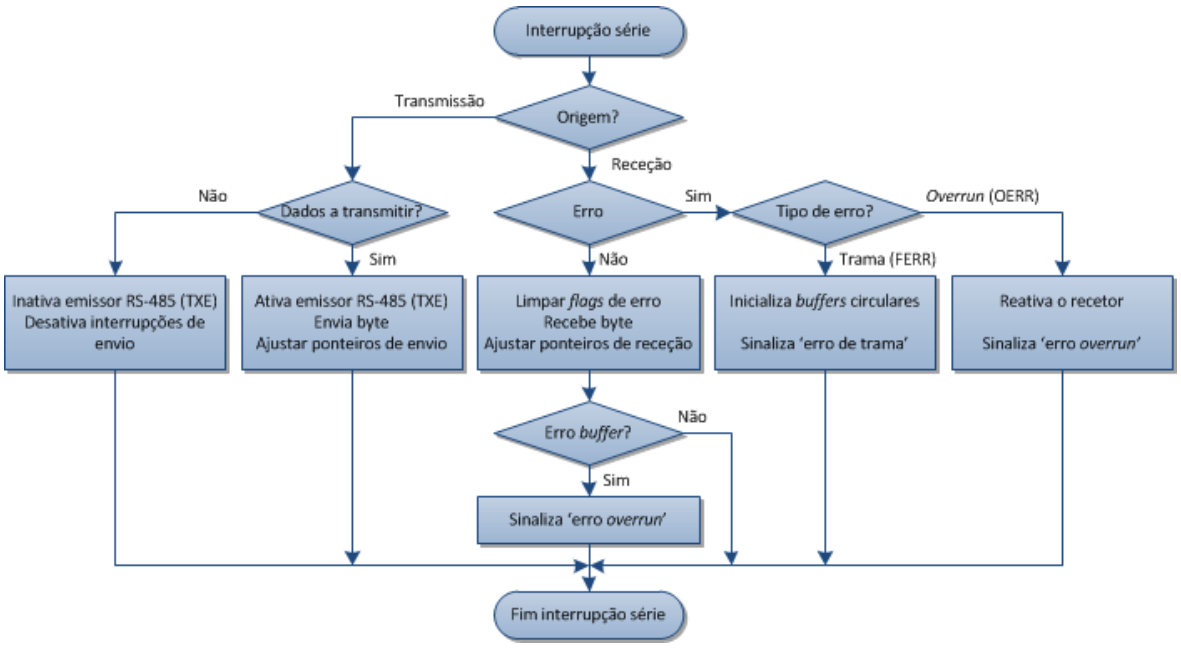


Figura 74 Fluxograma das interrupções da porta série

Para enviar um *byte*, a rotina de escrita "`void putch(unsigned char c)`" verifica se o *buffer* de envio está cheio ("`TXBufIn +1 = TXBufOut`") e nesse caso espera por espaço livre. Nessa altura, o *byte* a enviar é escrito no *buffer* por "`TXBuf[TXBufIn++] = c;`", e o ponteiro de envio

é incrementado e ajustado no caso de ultrapassar a última posição do *array*. Finalmente, as interrupções de envio são ativadas para que a USART proceda ao envio.

Para receber um *byte*, a rotina de leitura "`unsigned char mygetc()`", após verificar a existência de dados no *buffer* de recepção, retorna a posição indexada pelo apontador de leitura: "`ch = RXBuf[RXBufOut++];`". No caso de não existir dados disponíveis, a função retorna 0. Esta informação é utilizada pela função "`unsigned char waitc()`" para ficar à espera de um *byte* (a função bloqueia até que um *byte* disponível na porta série). A biblioteca disponibiliza duas funções adicionais "`void putst(register const char *str)`" e "`void Repeat-Chr(char c, unsigned char qty)`" que permitem respetivamente o envio de uma *string* (terminada por 0) e a repetição de um caracter.

O periférico USART disponibiliza duas informações de erro de recepção de dados (OERR – Erro de *overrun* e FERR – Erro de trama) que são tratados na rotina de interrupção. O erro de trama, limpa o *buffer* de recepção, indicando que houve perda de informação. O erro de *overrun* é limpo e sinaliza a perda do último caracter.

6.3.1.2. TEMPORIZADORES E FUNÇÕES DE ESPERA

Para utilizar os temporizadores, é necessário proceder à sua configuração. A inicialização está implementada no módulo "cpSerialAndTimer.c" que define a função "`InitTimer0`" e "`InitTimer1`" onde os registos associados respetivamente ao "Timer 0" e "Timer 1" são inicializados. O Timer 1 é utilizado para gerar atrasos programáveis. O "Timer 0" é utilizado em algumas implementações (e.g. no "Sensor de bola") como gerador da frequência portadora.

A Figura 75 mostra o fluxograma do bloco relativo às interrupções do temporizador 1 (Timer1). Este temporizador está configurado para gerar uma interrupção a cada 100 µs. O contador ("`tmrcnt`") fornece o número de frações de 100 µs que passaram desde que foi inicializado. Quando uma função necessita de um atraso (de resolução igual a 1 ms), utiliza a função "`void DelayMS(unsigned int ms)`" que carrega o contador de atraso ("`delayCnt`") com o tempo de atraso desejado, e espera que o mesmo seja 0. O bloco de código indicado a cor diferente ("*Processo específico de cada implementação*"), varia de acordo com a implementação específica para cada módulo de *hardware*.

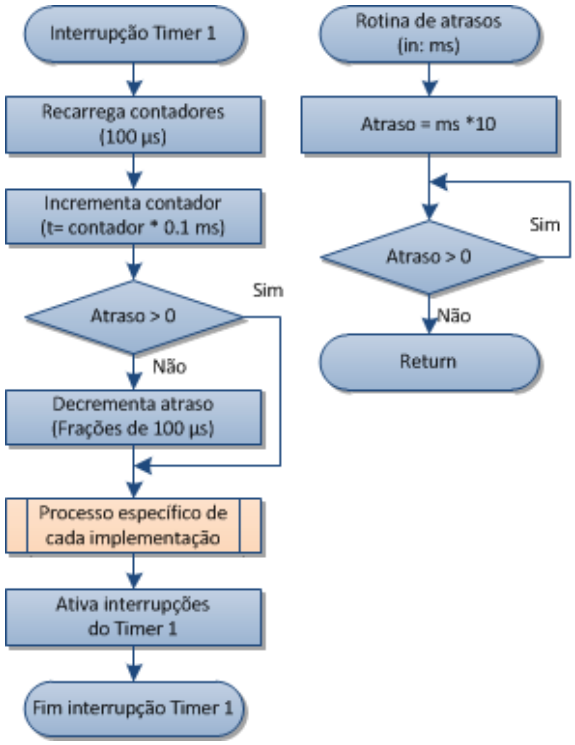


Figura 75 Fluxograma da interrupção do 'Timer 1' e rotina de atrasos

6.3.1.3. CICLO PRINCIPAL E INICIALIZAÇÃO

O ponto de entrada de execução é no ciclo principal "*`main()`*" cujo fluxograma é apresentado na Figura 76. O programa começa por efetuar a inicialização dos periféricos necessários ao seu funcionamento. A rotina de inicialização começa por configurar o oscilador interno para funcionar a 32 MHz. São inicializados os pinos de I/O, de acordo com a implementação necessária.

Segue-se a inicialização do porto série e dos temporizadores. Findo o processo de inicialização, as interrupções são ativadas para dar início ao ciclo de controlo. O último passo do arranque é a leitura da EEPROM para obter o endereço a que o programa deve responder ("*`NodeID`*").

Após este passo, a função principal fica num ciclo fechado, sinalizado pelo piscar do LED verde à frequência de 1 Hz. Neste ciclo, o processador está constantemente a verificar se foi recebido algum caracter na porta série. Se receber o caracter de início de comando '#', o microcontrolador começa a guardar os caracteres subsequentes até receber o caracter de fim de comando '\r' ou até ser ultrapassado o fim do *buffer* de recepção de comando (o que torna o comando inválido).

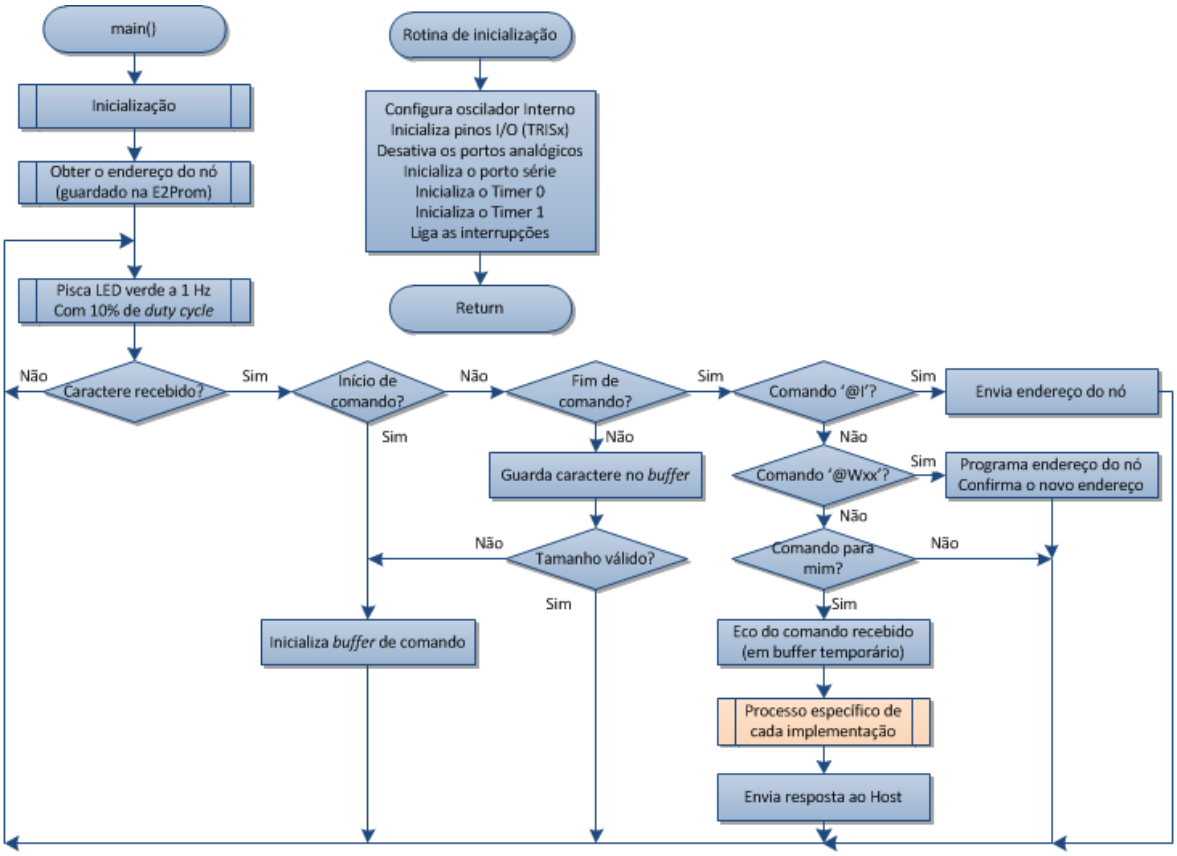


Figura 76 Fluxograma do ciclo principal e rotina de inicialização

Após conclusão da recepção do comando, primeiramente verifica-se se é um comando de configuração do nó (iniciado por '@'). Este comando destina-se a ler e programar o endereço de rede a ser utilizado, e só pode ser executado fora da rede (com uma ligação direta). Se não for um comando de configuração, o algoritmo verifica se o endereço de destino da mensagem é o próprio, e nesse caso passa para a verificação da validade do comando. Neste estágio, é enviado o eco do comando recebido, que inclui a sua resposta se tal for o caso.

A descrição dos comandos implementados em cada um dos módulos de *hardware* está descrita nas subsecções seguintes. A Tabela 11 mostra a lista completa dos comandos implementados em cada módulo. Nas duas colunas da direita, é apresentada a resposta, sempre com o eco do comando enviado, afirmativa '1' e negativa '0', respetivamente, no caso de comandos de estado. Os comandos de ação retornam o resultado da ação. No caso das leituras de parâmetros ('*tempo de voo*', '*velocidade inicial do projétil*' e '*alcance do projétil*'), a resposta é enviada depois do eco do comando, e com ';' a separar os parâmetros.

Tabela 11 Lista dos comandos implementados em cada módulo

Módulo	Descrição	Comando	Resposta (V / F)	
Sensor de bola	Sensor <id> intercetado?	#<id>Z	<id>Z1	<id>Z0
Sistema de expansão do seletor de bola	Seletor de bola no limite?	#3L	3L1	3L0
	Bola carregada?	#3B	3B1	3B0
C. Distribuição Elevador de bola	Bola na rampa?	#4R	4R1	4R0
	Elevador de bola no topo?	#4T	4T1	4T0
	Alimentação dos motores ligada?	#4M?	4M?1	4M?0
	Liga alimentação dos motores	#4M1	4M1	
	Desliga alimentação dos motores	#4M0	4M0	
C. Distribuição da rampa	Liga retentor de bola	#8S1	8S1	
	Desliga retentor de bola	#8S0	8S0	
	Elevador de rampa no topo?	#8T	8T1	8T0
	Alimentação dos motores ligada?	#8M?	8M?1	8M?0
	Liga alimentação dos motores	#8M1	8M1	
	Desliga alimentação dos motores	#8M0	8M0	
	Tempo de voo?	#8V	8V<velocidade.>;<t.voo>	
Sistema de medição	Alcance do projétil?	#13Z	13Z<alcance>	

6.3.2. SENSOR DE BOLA

O CI utilizado no "*Sensor de bola*" requer um sinal modulado com uma frequência portadora de 38 kHz para validar o feixe IR (*Infrared*) recebido. O diagrama da Figura 77 apresenta a resposta temporal do sensor ao estímulo do sinal de teste [Anexo B – III].

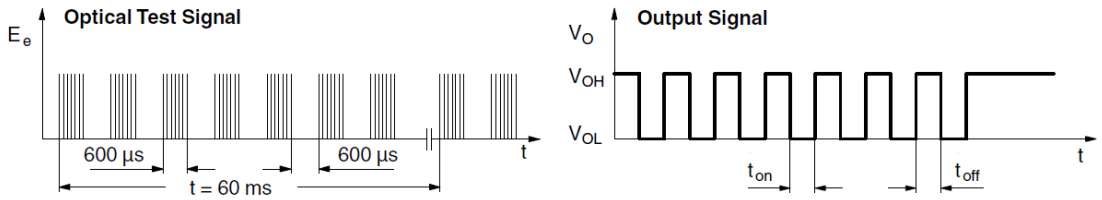


Figura 77 Diagrama temporal do sensor TSOP36238TR (cortesia Vishay™)

A Figura 78 apresenta o fluxograma da implementação do "Timer 0" utilizado para gerar a frequência portadora com um tempo de intervalo de 26,3 µs (38 kHz), e da implementação do "Timer 1" (indicado no fluxograma da Figura 75 como "*Processo específico*").

A saída que liga o LED é modulada pelo sinal "BURST" que é invertido a cada 600 µs, e neste tempo é adquirido o estado de saída do sensor, guardado na variável indicadora de bola detetada (feixe IR interrompido). O estado do sensor é copiado para a linha de saída do sensor (RC1) e indicada via LED vermelho (RA0).

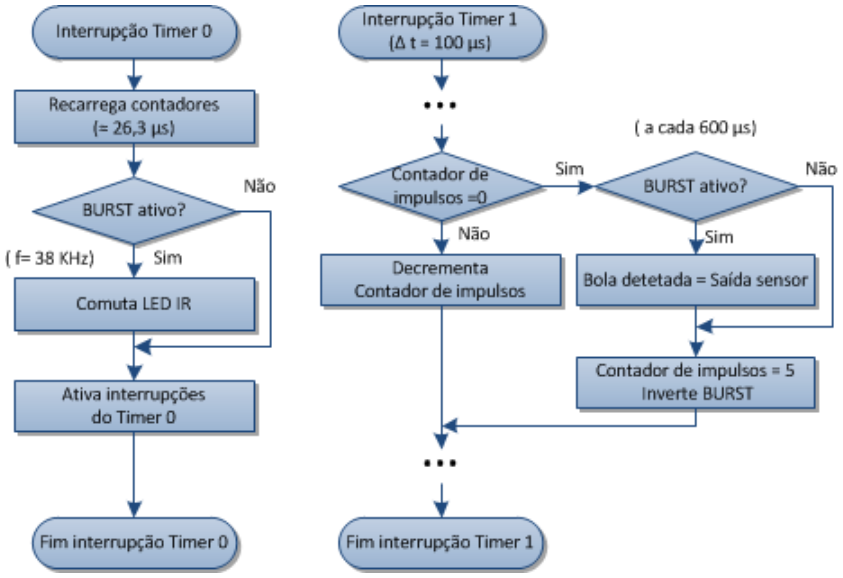


Figura 78 Fluxograma da temporização do sensor de bola

6.3.3. SISTEMA DE EXPANSÃO SMCI12

O "*Sistema de expansão SMCI12*" tem por finalidade interligar os sinais recolhidos dos sensores externos ao controlador SMCI12 associado. O módulo do seletor de bola liga o sensor de limite de movimento do carro (RB1) à linha de entrada IN1 do controlador SMCI12 (via RA0). O estado deste sensor é indicado pelo LED vermelho. Adicionalmente, o LED verde indica o estado do sensor, de bola carregada quando ligado.

O "*Sistema de expansão SMCI12*" do elevador da rampa, dado o controlador SMCI12 não conseguir detetar um impulso de duração inferior a 15 ms, alarga o sinal proveniente do sensor de limite do codificador angular, para um valor admissível ao controlador SMCI12. O Timer 0 fornece um impulso de 25 ms quando é detetada a comutação do sensor de limite do codificador angular (≈1 ms). O estado do sensor é indicado pelo LED vermelho.

6.3.4. CAIXA DE DISTRIBUIÇÃO

A "*Caixa de distribuição*" controla a alimentação dos motores e faz o mapeamento dos sinais lógicos entre os sensores a ela ligados. Existem duas implementações distintas de controlo, uma para a do "*Elevador de bola*" e outra para a do "*Elevador da rampa de lançamento*". A implementação da "*Caixa de distribuição*" do "*Elevador de bola*" só implementa o controlo da alimentação dos motores. Com essa finalidade, é utilizado o 'Timer 0' para obter informação do estado da alimentação ('*Power_Good*') disponível no pino RC0 do microcontrolador.

A "*Caixa de distribuição*" do "*Elevador da rampa de* lançamento" implementa o controlo da alimentação dos motores e o cálculo do '*tempo de voo*'. Com essa finalidade, é utilizado o "Timer

0" para obter informação do estado da alimentação ('*Power_Good*') disponível no pino RC0 do microcontrolador, e o sinal do sensor de limite da rampa ('RA3') é mapeado na entrada IN0 do controlador do elevador da rampa ('RA0'). A principal função de controlo da "*Caixa de distribuição*" da rampa é a medição do '*tempo de voo*'. O 'Timer 1' está continuamente a verificar se o sensor existente à saída da rampa foi intercetado pela bola ('*início de voo*'). Quando tal acontece, é iniciada a contagem do tempo de voo e do tempo de passagem da bola (tempo em que o sensor de '*Início de voo*' está intercetado). O tempo de voo continua a ser contado até que o sensor do plano de recolha é acionado (indicando o '*fim de voo*'). Nesta altura, o sistema de contagem conclui o processo, e os contadores '*tempo de voo*' e '*tempo de passagem*' contém os tempos medidos. O '*tempo de voo*' (t_{voo}) é calculado pela fórmula (17):

$$t_{voo} =' Tempo\ de\ voo' + \frac{'Tempo\ de\ passagem'}{2} \qquad (17)$$

A Figura 79 apresenta o fluxograma do algoritmo implementado no 'Timer 0' para a determinação do '*tempo de voo*'.

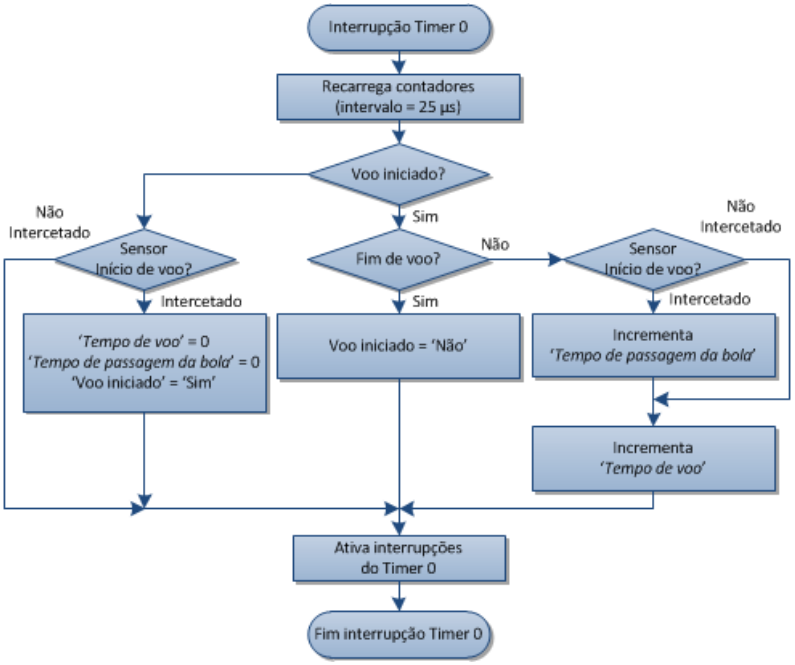


Figura 79 Fluxograma da medição do '*tempo de voo*'

O comando '*tempo de voo*' devolve o valor do tempo calculado em frações de 25 µs, e.g. 1250 = 31,25 ms. Tendo em conta a utilização de uma variável do tipo '`unsigned int`', o tempo máximo possível (em segundos) é dado pela fórmula (18):

$$t_{max} = (2^{16} - 1) * 25^{-6} = 65535 * 0{,}000025 = 1{,}638375\ \text{s} \qquad (18)$$

Esta restrição não é importante tendo em conta que o '*tempo de voo*' é na ordem das poucas centenas de milissegundos, conforme apresentado na secção 7.4.

6.3.5. SISTEMA DE MEDIÇÃO DO '*ALCANCE HORIZONTAL DO PROJÉTIL*'

O sistema de medição do '*alcance horizontal do projétil*' utiliza para o seu controlo um canal de comunicação série síncrono do tipo SPI, para o qual é utilizado o periférico SPI do microcontrolador. A cadeia de comunicação implementada necessita de enviar 48 *bits* e de receber 96 *bits*. Com essa finalidade, foram implementadas as funções '`wrOutputs`' e '`rdInputs`' que, respetivamente, envia o *buffer* de 6 *bytes* ('`LedBuf`') e recebe o *buffer* de 12 *bytes* ('`InpBuf`'). A Figura 80 apresenta o fluxograma do algoritmo implementado. Um ciclo de varrimento é implementado pela função '`Led_IO`' (à esquerda) que chama as sub-rotinas necessárias ('`ledShift`', '`wrOutputs`', '`rdInputs`' e '`updateMeasurement`').

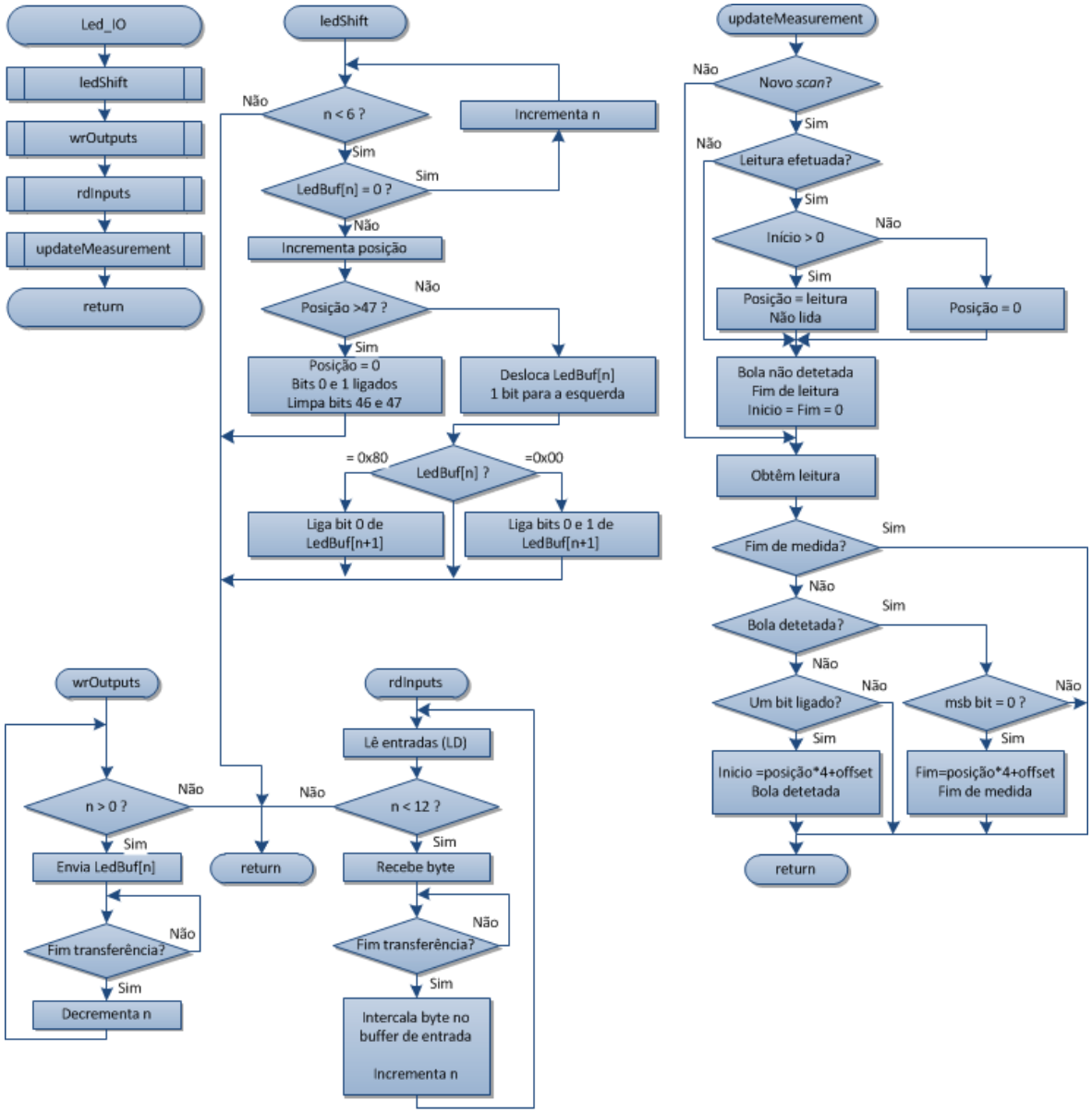


Figura 80 Fluxograma do sistema de medição

Um ciclo de leitura começa pela função '*ledShift*' que desloca dois *bits* ligados consecutivos ao longo do *buffer* de 6 *bytes*. O resultado desta operação é enviado pela função '*wrOutputs*' que liga os emissores de infravermelhos programados. De seguida é lido o registo de deslocamento de entrada (96 *bits*) cujo resultado é guardado no *buffer* '*InBuf*'. A função '*UpdateMeasurement*' calcula a distância relativa atingida pelo projétil. Este ciclo é repetido até o varrimento tenha passado por todos os emissores.

6.4. CONCLUSÃO

Este capítulo apresentou o projeto lógico de controlo do aparato, que possui duas camadas distintas de implementação:

- Controlo de nível superior, implementado em sistema operativo Windows™, com recurso a tecnologias diversas ('Serviços web' implementados na infraestrutura WCF, interface de utilizador implementados na infraestrutura WPF). O código foi desenvolvido em linguagem C#, utilizando uma metodologia orientada a objetos.
- Controlo de nível inferior, implementado em cada um dos módulos projetados, utilizando uma plataforma comum (ciclo principal, implementação do protocolo, recepção série e temporizações), escrita em linguagem C para os microcontroladores PIC16F1xxx. Esta configuração visou a implementação de um sistema simples, no qual o servidor de experiência faz um pedido e o módulo respetivo responde/executa.

Foram apresentados os fluxogramas detalhados de implementação, remetendo-se para anexo o detalhe do código de implementação. O presente capítulo termina a apresentação da implementação do aparato. O capítulo seguinte apresenta os resultados obtidos e a sua validação.

Capítulo 7

VERIFICAÇÃO E VALIDAÇÃO

Neste capítulo descrevem-se os resultados e o processo de validação do aparato. Na caracterização do tempo de utilização do aparato é apresentado o ambiente laboratorial a que se destina, quantificando o número de execuções de uma experiência por aluno, em função do espaço temporal disponibilizado pela aplicação de acesso web. Para a caracterização do tempo de execução da experiência, descreve-se a influência temporal de cada um dos parâmetros envolvidos, iniciando na influência da bola selecionada, altura e ângulo de lançamento. A análise dos custos de desenvolvimento detalha os custos dos materiais utilizados, que englobam os de manufatura no fornecedor de componentes mecânicos, de mão-de-obra, estimados em função do número de dias necessários a cada tarefa, e de alguns custos adicionais. Os resultados experimentais apresentam a influência de cada um dos parâmetros no '*tempo de voo*' e na '*velocidade inicial do projétil*', obtido para cada uma das variáveis de entrada ('*altura de lançamento*', '*ângulo de lançamento*' e '*tipo de projétil*'), fixando as duas restantes.

7.1. CARACTERIZAÇÃO DO TEMPO DE UTILIZAÇÃO DO APARATO

O desenvolvimento deste aparato teve como premissa, a realização de uma máquina fiável, de cariz profissional, para ser utilizada remotamente nas aulas laboratoriais de Física. Uma aula laboratorial de Física tem um número máximo de 18 alunos por turma, que desenvolvem o seu trabalho

laboratorial em aulas de duas horas, distribuídos por grupos com três alunos (situação das aulas laboratoriais de Física no ISEP). Para distribuir um dado recurso de forma igual pelos grupos presentes no laboratório, pode-se afirmar que cada grupo terá esse recurso disponível aproximadamente 20 minutos.

O estudo do "*Lançamento de projéteis*" requer a realização de vários lançamentos, com diferentes parâmetros ('*ângulo de lançamento*', '*altura de lançamento*' e '*tipo de projétil*'), de forma a relacionar o seu alcance e o tempo de voo com a velocidade inicial, altura, ângulo de lançamento e massa da bola. O aparato foi desenvolvido para ser utilizado de modo contínuo e executar a experiência de uma forma parametrizável, em que os resultados obtidos, para uma determinada configuração, sejam repetíveis e homogéneos. Assim, para o fim didático a que se propõe, a realização da experiência deve ser determinística e repetível.

O interface web para acesso remoto ao aparato foi desenvolvido no âmbito do projeto "*Physics LabFARM*" pelo colega Pedro Nogueira e não está incluído no âmbito desta tese. No entanto, com a finalidade de caracterizar o tempo de utilização, torna-se necessário a sua referência. O interface de utilizador desenvolvido utiliza um processo de agendamento (Figura 81) no qual os alunos reservam um intervalo de tempo (15 m) para a sua utilização. Durante este tempo, os alunos dispõem de acesso exclusivo ao aparato, com um canal de áudio e vídeo para acompanhamento das várias experiências.

Figura 81 Interface web de agendamento

Para a realização de uma experiência é apresentado o interface da Figura 82 onde são introduzidos os parâmetros '*bola*', '*altura de Lançamento*' e '*ângulo de lançamento*' e é dada a ordem de execução. Finalizada a experiência, são devolvidos os resultados da experiência realizada, e uma foto do instante de impacto do projétil. Tendo em conta o tempo máximo de execução de uma experiência, obtido na amostra 10 da Figura 83 (36,8 s), é possível verificar que se o aluno executar de uma forma continuada a mesma experiência, na pior condição poderá realizar $900/36{,}8 \approx 24$

ciclos. No entanto este número não tem significado prático, dada a necessidade de introdução e recolha manual dos dados.

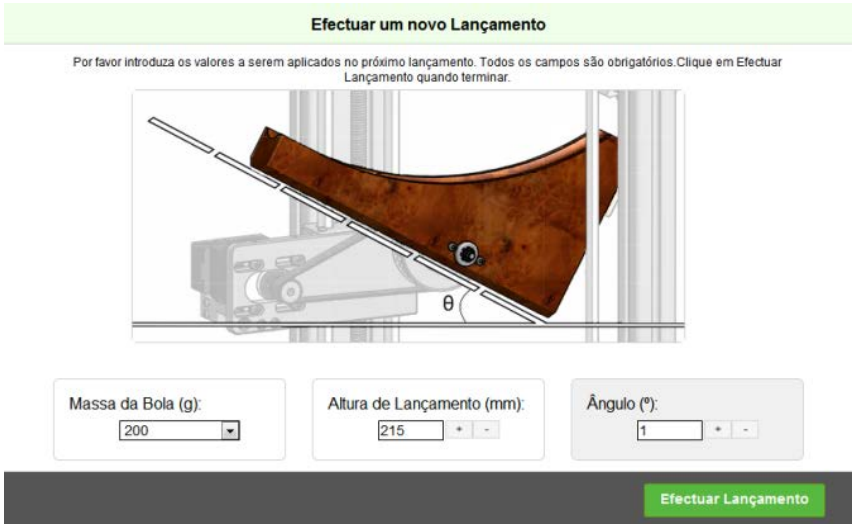


Figura 82 Interface web para introdução dos parâmetros da experiência

Neste momento é difícil avaliar na prática o número total de ciclos experimentais que um aluno poderá realizar (falta a validação da utilização do aparato em ambiente laboratorial), mas poder-se-á estimar ser possível a realização de 6 a 8 lançamentos, o que dá um tempo aproximado por experiência entre 2,5 e 1,8 minutos respetivamente. Este tempo é suficiente para a introdução dos parâmetros, desenvolvimento de pensamento crítico e análise de resultados.

7.2. CARACTERIZAÇÃO DO TEMPO DE EXECUÇÃO DA EXPERIÊNCIA

O tempo de um ciclo de execução da experiência varia em função da "*altura da rampa de lançamento*" e da "*bola selecionada*". Por esse motivo, o ciclo de execução da experiência foi subdividido nas seguintes etapas:

1. Carregar a bola no elevador;
2. Elevar a bola até ao topo do elevador;
3. Ejetar a bola para a rampa de lançamento;
4. Configurar a altura e ângulo especificados;
5. Executar o lançamento;
6. Retornar a rampa e o "*Elevador de bola*" à posição de repouso;

Para a recolha dos vários tempos, o programa de controlo da experiência foi modificado, de forma a ser efetuado um registo em tempo real, dos parâmetros de execução e respetivos tempos obtidos.

A Tabela 12 apresenta os valores obtidos na execução de 10 lançamentos consecutivos, em que a bola a utilizar, a altura selecionada e o ângulo de lançamento foram gerados aleatoriamente.

Tabela 12 Tempos por operação

Altura (mm)	Ângulo (°)	Bola	Carga da bola	Rampa pronta	Bola no topo	Bola na rampa	Pronto a lançar	Lança-mento	Fim de experiência	Tempo Total
270,9	5,2	2	4,97	4,38	8,88	2,13	4,25	1,12	8,75	34,47
289,2	-5,1	3	6,22	4,39	8,90	2,12	3,50	1,14	9,43	35,69
325,4	2,1	2	6,06	4,38	8,88	2,13	1,63	1,13	11,39	35,57
126,7	9,5	3	6,20	4,38	8,93	2,13	10,84	1,13	2,13	35,72
296,9	-1,1	3	7,34	7,38	5,89	2,13	2,88	1,13	10,05	36,78
174,9	11,8	2	6,02	4,40	8,88	2,13	8,78	1,12	4,25	35,58
338,3	-14,1	2	4,97	6,27	7,00	2,13	2,88	1,13	10,15	34,51
192,8	-3,0	1	3,54	4,40	8,88	2,13	7,50	1,13	5,50	33,08
222,4	-11,8	3	4,95	5,38	7,88	2,13	6,80	1,13	6,16	34,41
293,0	0,1	3	7,32	4,43	8,91	2,18	3,13	1,13	9,75	36,83

Para a análise da variação dos tempos de operação, gerou-se a Tabela 13 com os dados estatísticos calculados, onde se verifica que as operações, com a exceção da "*Bola na rampa*" e "*Lançamento*", com desvio padrão próximo de zero, são influenciadas pelos parâmetros da experiência.

Tabela 13 Dados estatísticos dos tempos por operação (em segundos)

	Carga da bola	Rampa pronta	Bola no topo	Bola na rampa	Pronto a lançar	Lançamento	Fim de experiência	Tempo Total
Máximo:	7,339	7,375	8,925	2,176	10,843	1,135	11,386	36,828
Mínimo:	3,544	4,375	5,885	2,125	1,625	1,125	2,125	33,075
Diferença:	3,795	3,000	3,040	0,051	9,217	0,011	9,260	3,753
Média:	5,760	4,975	8,300	2,130	5,218	1,126	7,754	35,262
Desvio Padrão:	13,310	10,001	10,149	0,002	84,200	0,000	83,561	12,020

O gráfico da Figura 83 apresenta o tempo necessário para efetuar cada uma das operações. Este gráfico apresenta as barras empilhadas para ser possível visualizar o tempo total de execução.

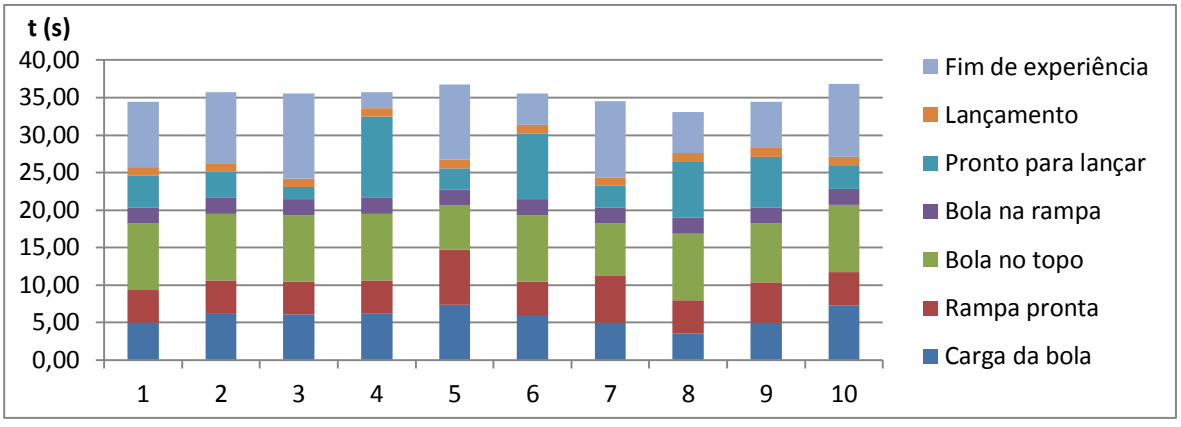


Figura 83 Gráfico do tempo por tipo operação

Nas subsecções seguintes, vão ser analisadas as influências de cada um dos parâmetros da experiência, '*bola*', '*altura de lançamento*' e '*ângulo de lançamento*' respetivamente.

7.2.1. INFLUÊNCIA DA BOLA SELECIONADA

Para a análise da influência da bola, foi realizado o gráfico da Figura 84, que relaciona o tempo "*Carga da bola*" com a bola selecionada. O seletor de bola inicia o seu percurso na posição da bola escolhida no lançamento anterior. Tendo em conta a possibilidade de perda de passos durante o trajeto, foi decidido que este movimento, quando é iniciado, faz uma calibração (verifica a posição do sensor de limite). O trajeto total do carro do seletor é a soma do trajeto da posição da última experiência até ao sensor de limite, somado ao tempo do trajeto do sensor de limite até à nova posição (a posição da 'bola 1' é a mais próxima do sensor de limite). Os tempos mais elevados (ciclos 5 e 10) são resultado da seleção da 'bola 3', em ambas as experiências (experiência anterior e em curso).

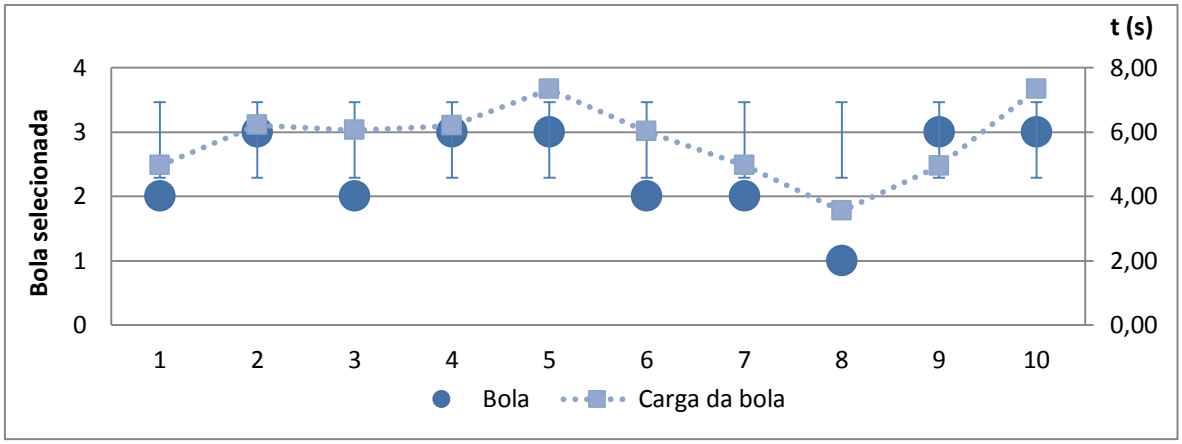


Figura 84 Gráfico do tempo de carga da bola

A variação de tempo da operação de carga da bola no elevador [3,544; 7,339] mostra uma área de melhoria[6] em que o tempo máximo da operação pode ser reduzido se a sequência de funcionamento for alterada (e.g. só calibrar quando for selecionada a '*bola 1*'), e o sistema guardar a última posição escolhida (dessa forma, pode mover-se diretamente para a nova posição).

7.2.2. INFLUÊNCIA DA ALTURA DE LANÇAMENTO

A posição de carga da bola na rampa de lançamento (posição de repouso) é próxima do valor máximo de altura da rampa. Após efetuar um lançamento, o elevador calibra (move-se até ao

[6] A opção de não incluir esta melhoria prende-se com o espaço temporal em que foi detetada, que inviabilizou a sua inclusão no âmbito desta tese

sensor de limite do movimento, no topo do "*Elevador da rampa de lançamento*") e retorna à posição de carga (no topo do "*Elevador de bola*"). O perfil de controlo do movimento do motor da rampa (Figura 85) está configurado para uma velocidade máxima de 22,5 mm.s^{-1}. O gráfico apresenta a velocidade instantânea no decorrer do tempo total do movimento (≈4,5 s), para uma deslocação de 100 mm. A rampa tem como intervalo absoluto de movimento [100;348], o que resulta numa deslocação relativa máxima de $348 - 100 = 248$ mm, cujo tempo necessário para o percorrer é dado pela fórmula (19):

$$t_{total} = 2 * t_a + (d - 2 * 0{,}62) * \frac{1}{22{,}5} = 2 * 0{,}04836 + (248 - 2 * 0{,}62) * \frac{1}{22{,}5} \approx 11{,}1\ s \quad (19)$$

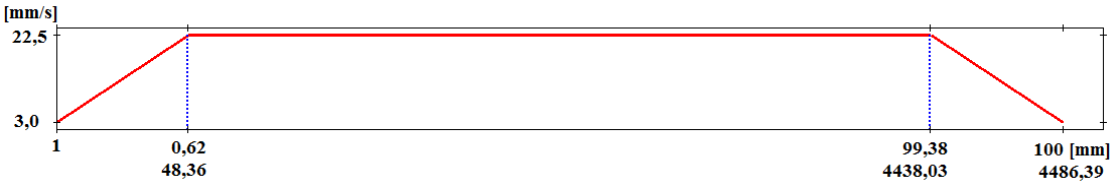


Figura 85 Perfil de controlo do movimento do motor da rampa

O "*Elevador de bola*" apresenta um perfil de movimento semelhante, no qual a velocidade máxima é de 37,5 mm.s^{-1}. O gráfico da Figura 86 apresenta o perfil do movimento do elevador entre a posição de carga, na saída do seletor de bola, até ao ponto de descarga (topo do "*Elevador de bola*"). Este movimento demora aproximadamente 12,7 s.

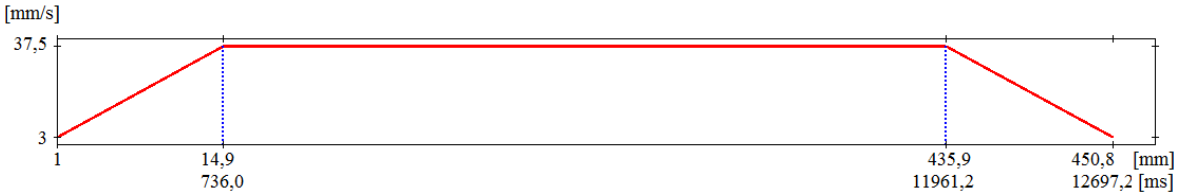


Figura 86 Perfil de controlo do movimento do motor do *"Elevador de bola"*

O gráfico da Figura 87 apresenta o tempo para a rampa se deslocar até à posição de lançamento programada ("*Pronto a lançar*"), cujo eixo vertical (lado esquerdo) foi invertido, para se poder relacionar diretamente o tempo versus altura.

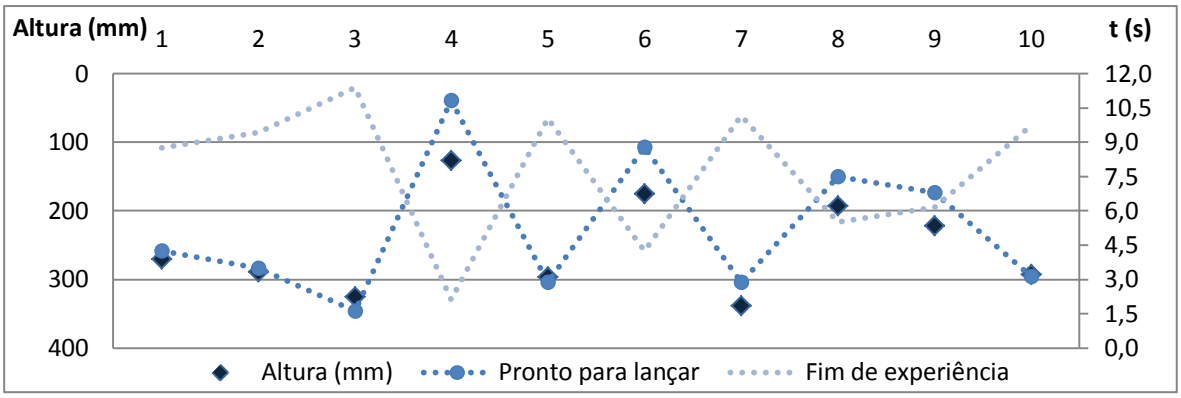


Figura 87 Gráfico temporal do movimento da rampa

O tempo "*Fim de experiência*" é o tempo resultante do retorno à posição de repouso de ambos os elevadores. Este gráfico mostra o impacto da altura de lançamento no tempo total da experiência que se torna mais significativo à medida que a altura de lançamento é menor (mais distante do ponto de carga). O tempo final de experiência é aproximado do maior tempo de deslocação, comparando os tempos de retorno do "*Elevador de bola*" (12,7 s) com o do "*Elevador da rampa de lançamento*" ($\approx Altura/22{,}5$) ao ponto de repouso, visto o programa de controlo esperar a finalização de ambos os movimentos. De notar que o "*Elevador de bola*" inicia o retorno imediatamente após a ejeção da bola para a rampa de lançamento.

7.2.3. INFLUÊNCIA DO ÂNGULO DE LANÇAMENTO

O tempo do movimento de configuração do ângulo de lançamento é desprezável em relação ao tempo necessário da configuração da altura da rampa. O tempo máximo para rodar a rampa de -15° até +15 ° é ≈ 0,5 s, e está incluído no tempo "*Pronto a lançar*" do gráfico da Figura 87.

7.3. CUSTOS DE DESENVOLVIMENTO

Para o desenvolvimento de um projeto de automação é necessária a aquisição das matérias-primas/componentes necessários e a quantificação da mão-de-obra envolvida. As seções seguintes apresentam o detalhe dos custos relativos ao aparato.

7.3.1. CUSTOS DE MATERIAIS E OUTROS

Os materiais necessários ao projeto deste aparato dividem-se nas seguintes categorias:

1. Componentes mecânicos produzidos a partir dos desenhos
2. Componentes mecânicos genéricos (parafusos, porcas, polias, correias, etc.)
3. Componentes de controlo (controladores dos motores, fontes de alimentação, etc.)
4. Componentes eletrónicos
5. Circuitos impressos
6. Cabos elétricos

Os componentes mecânicos (1 e 2) e os cabos elétricos (6) foram fornecidos pela Tecnogial™ [23]. A Bibus™ [24], distribuidora da Nanotec™ [25] em Portugal, forneceu os motores, controladores e acessórios (3). O fornecimento dos componentes eletrónicos (4) esteve a cargo da Digikey™ [26]. Os circuitos impressos (5) foram fornecidos pela PCB-Pool™ [27]. A Tabela 14 apresenta os custos globais dos materiais por fornecedor, que totalizaram, no final do projeto 8.110,00 €. Os custos adicionais, compostos por custos de deslocação a fornecedores, compras de ferramentas necessárias à montagem etc., foram aproximadamente 550,00 €, o que totalizou 8.660,00 €.

Tabela 14 Distribuição de custos por fornecedor

Fornecedor	*Descrição*	*Custo total*
Tecnogial™	Componentes mecânicos do elevador e seletor de bola	1.550,00 €
	Componentes mecânicos da rampa	1.760,00 €
	Componentes mecânicos da recolha	980,00 €
	Estrutura e peças adicionais	1.400,00 €
Nanotec™	Motores	330,00 €
	Controladores	370,00 €
	Fontes de alimentação e cabos e acessórios	500,00 €
Digikey™	Componentes elétricos e eletrónicos	670,00 €
PCB-Pool™	Circuitos impressos	550,00 €
Outros	Deslocações a fornecedores e compra de ferramentas	550,00 €
	Total:	8.660,00 €

7.3.2. CUSTOS DE MÃO-DE-OBRA

Os custos de mão-de-obra são estimados em função do número de dias necessários a cada tarefa. As tarefas dividem-se em três grandes grupos: pesquisa, desenvolvimento e execução. A pesquisa reúne as tarefas de estudo do problema e análise dos requisitos. No desenvolvimento está englobado o projeto mecânico, elétrico e eletrónico e o projeto lógico. O tempo de montagem engloba a montagem dos vários grupos de componentes mecânicos, a montagem dos circuitos impressos e cablagem do aparato. A Tabela 15 detalha o tempo de desenvolvimento do projeto, quantificado em dias úteis de trabalho de oito horas, subdividindo-o nas suas tarefas.

Tabela 15 Distribuição de tempo por tarefa

Grupo de tarefa	Tarefa	Dias	Subtotal
Pesquisa	Estudo do problema proposto	15	
	Identificação dos requisitos estruturais e esboço inicial	30	
	Análise dos requisitos de controlo, acesso e interface	32	77
Desenvolvimento	Projeto mecânico do "*Elevador de bola*"	47	
	Projeto mecânico do "*Elevador da rampa de lançamento*" e "*Sistema de recolha*" e estrutura	40	
	Projeto eletrónico, PCB e projeto elétrico	75	
	Serviço web e interface do utilizador	45	
	Programação de controlo microcontroladores PIC	45	252
Execução	Montagem do "*Elevador de bola*"	19	
	Montagem do "*Elevador da rampa de lançamento*" e "*Sistema de recolha*"	23	
	Montagem estrutura e cablagem (inferior e superior)	57	
	Montagem "S. de bola", "Cx. dist." e "Expansão SMCI12"	21	
	Montagem do "*Sistema de medição*"	15	135
	Total:	464	

O tempo de mão-de-obra total do projeto foi 1,93 UTA (Unidade de trabalho ano) [7]. Para a quantificação total do custo de mão-de-obra será necessário definir o vencimento anual.

7.4. RESULTADOS EXPERIMENTAIS

A experiência implementada neste aparato tem três variáveis de entrada: '*bola*', '*altura de lançamento*' e 'â*ngulo de lançamento*'. Esta secção apresenta o '*tempo de voo*' e '*velocidade inicial do projétil*' obtido para cada uma das variáveis de entrada, fixando as duas restantes. Deste modo é possível caracterizar experimentalmente o aparato. Os resultados foram obtidos a partir de condições pré-definidas para cada um dos casos, por leitura do '*tempo de voo*' e '*velocidade inicial*'. Estes resultados são fornecidos pela "*Caixa de distribuição*" da rampa em resposta ao comando "#8F". O resultado deste comando são dois números inteiros que significam o número de vezes que o temporizador mediu (em frações de 25 µs) o tempo de passagem da bola e o tempo de voo, respetivamente.

7.4.1. INFLUÊNCIA DO PROJÉTIL

Para a medição da influência da bola no '*tempo de voo*', foram efetuados 10 lançamentos com cada uma das bolas, a uma altura de 340 mm com um ângulo de lançamento de 0 °. Este teste, além de caracterizar a influência do projétil no '*tempo de voo*', permite obter a precisão do aparato. O gráfico da Figura 88 apresenta os resultados obtidos nos lançamentos efetuados com as bolas (1 a 3) e respetivas velocidades iniciais (Vi1 a Vi3). Como é possível verificar, os tempos de voo (ms) são praticamente constantes (o desvio padrão calculado é 0,823 ms, 0,577 ms, e 1,364 ms respetivamente). As velocidades iniciais são obtidas de modo indireto, dividindo o diâmetro das bolas (16, 18 e 19 mm respetivamente) pelo tempo medido e são praticamente constantes (o desvio padrão calculado é 0,020 $m.s^{-1}$, 0,023 $m.s^{-1}$, e 0,011$m.s^{-1}$, respetivamente).

Tendo em conta que a diferença de altura é constante (posição entre o retentor eletromagnético e a saída da rampa), a velocidade inicial do projétil devia ser a mesma, independentemente da massa. No entanto é possível verificar que a velocidade inicial da bola 3 é menor que as restantes.

Após análise da origem do problema, foi possível determinar que o mesmo se deve ao atrito. A rampa de lançamento tem um sulco aproximado de 18,5 mm de diâmetro, e esta característica impõe que as bolas de menor diâmetro (16 e 18 mm respetivamente) só tocam num ponto, em cada

[7] É o trabalho de uma pessoa por ano civil a tempo completo. 1 UTA = 240 dias de 8 horas.
(fonte: INE *site*: http://smi.ine.pt/Conceito/Detalhes?id=1783&lang=PT)

instante do seu percurso na rampa. A ‘bola 3’, dado ser maior que o sulco, fica apoiada nos limites do sulco (dois pontos de contacto), o que provoca um aumento do atrito a que a bola está sujeita.

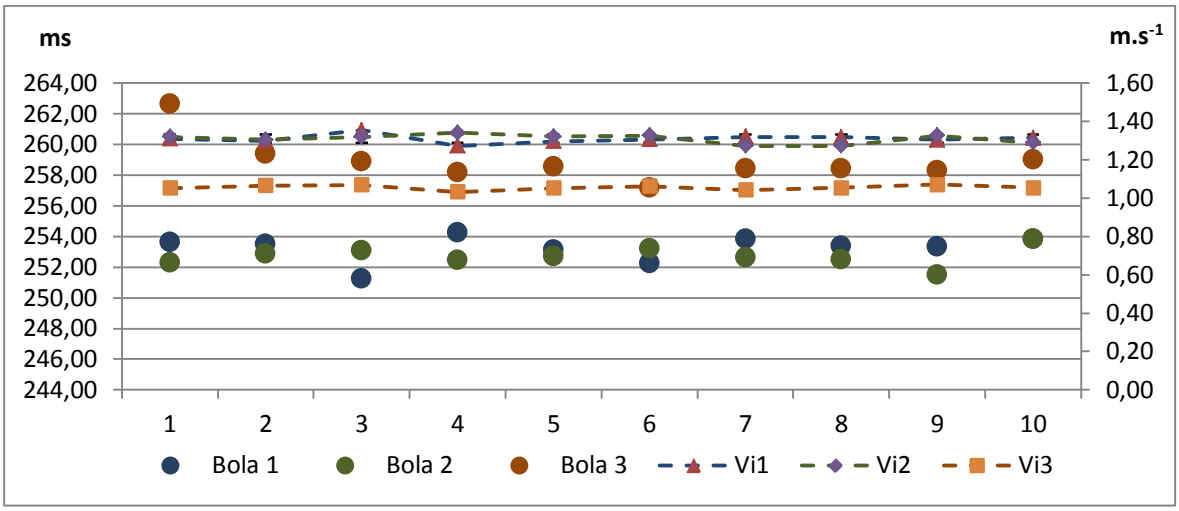


Figura 88 Gráfico do ‘*tempo de voo*’ versus projétil

A razão da variação do ‘*tempo de voo*’ da ’bola 3’ deve-se ao facto de a sua altura de lançamento estar aumentada da distância entre a bola e o fundo do sulco (a bola não toca o sulco, ficando alguns milímetros acima deste).

7.4.2. INFLUÊNCIA DA ALTURA DE LANÇAMENTO DO PROJÉTIL

Para o teste da influência da “*altura de lançamento*” no ‘*tempo de voo*’ foram efetuados 10 lançamentos com o projétil de tamanho intermédio (“*bola 2*”) com um ângulo de lançamento de 0°. A altura de lançamento foi estabelecida com intervalos de 25 mm. A Figura 89 apresenta o gráfico dos valores obtidos nos lançamentos efetuados.

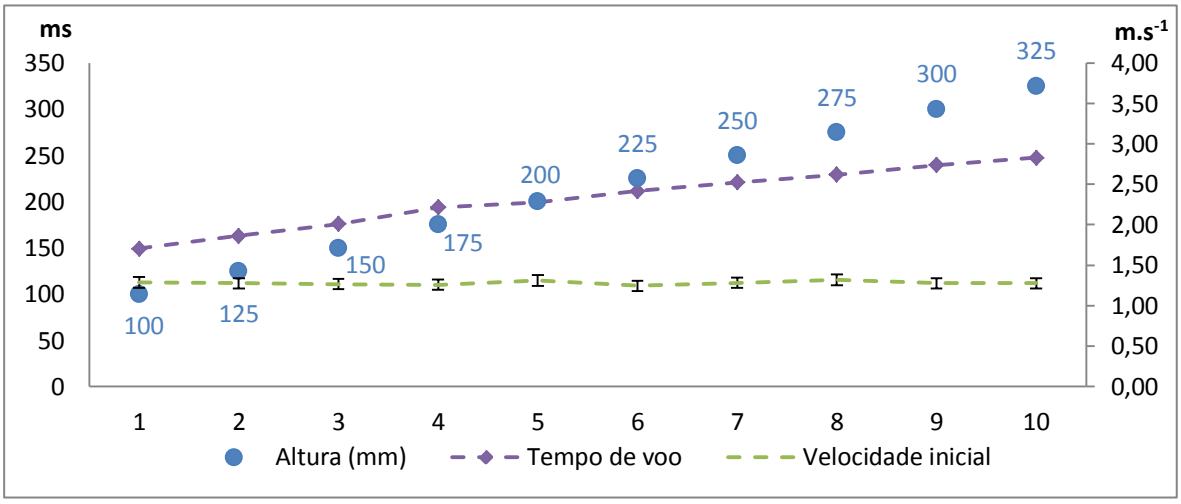


Figura 89 Gráfico do ‘*tempo de voo*’ versus ‘*altura de lançamento*’

Podemos verificar que o '*tempo de voo*' aumenta proporcionalmente com o aumento de altura de lançamento (o espaço a percorrer pelo projétil é maior). Tendo em conta que o ângulo de lançamento é constante verifica-se que a velocidade inicial também é constante (o desvio padrão calculado é 0,022 $m.s^{-1}$).

7.4.3. INFLUÊNCIA DO ÂNGULO DE LANÇAMENTO DO PROJÉTIL

Para o teste da influência do "*ângulo de lançamento*" no '*tempo de voo*' foram efetuados 11 lançamentos com o projétil de tamanho intermédio ("*bola 2*") com uma "*altura de lançamento*" de 325 mm. O "*ângulo de lançamento*" inicial (+15°) foi decrementado em intervalos de 3° até atingir o valor mínimo (-15°). A Figura 90 apresenta o gráfico dos valores obtidos nos lançamentos efetuados.

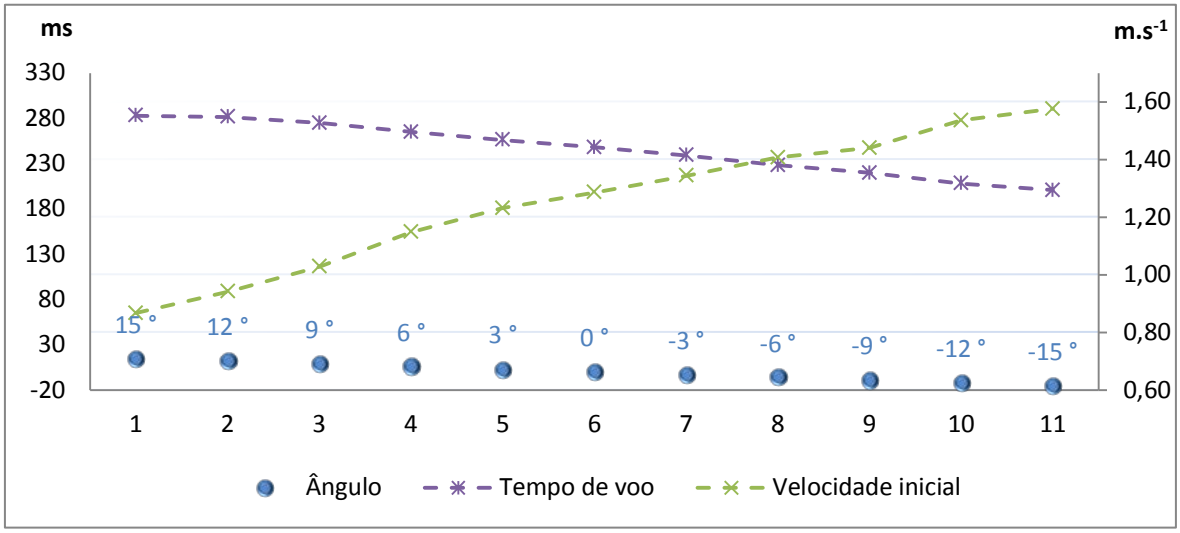


Figura 90 Gráfico do '*tempo de voo*' versus '*ângulo de lançamento*'

A altura inicial de lançamento (diferença entre a altura do retentor e a saída da rampa) vai aumentando à medida que o ângulo de lançamento diminui, e por consequência, a velocidade inicial do projétil também aumenta (pelo princípio da conservação de energia) [28].

7.5. CONCLUSÃO

Neste capítulo descreveram-se os resultados e o processo de validação do aparato, caracterizando os contextos temporais da experiência (execução de uma experiência e utilização por aluno), os custos associados ao projeto, os resultados experimentais e problemas encontrados no seu decurso.

Na caracterização do tempo de execução de uma experiência, foi descrita a influência que cada um dos parâmetros fornecidos ('*bola*', '*altura de lançamento*' e '*ângulo de lançamento*') tem no tempo total de execução da experiência. O tempo de utilização do aparato estimou o número de execuções

passíveis de serem efetuadas por aluno tendo em conta o espaço temporal disponibilizado pelo agendamento fornecido pela aplicação de acesso web.

Nos custos associados ao projeto foram detalhados os custos relativos aos materiais, englobando os de venda geral (componentes mecânicos, elétricos e eletrónicos), os serviços fornecidos externamente (fabrico de componentes mecânicos e circuitos impressos), os custos de mão-de-obra, estimados em função do número de dias necessários a cada tarefa e alguns custos adicionais não englobáveis das duas categorias anteriores.

Os resultados experimentais apresentaram a influência dos parâmetros da experiência no resultado obtido do '*tempo de voo*' e '*velocidade inicial*'. Adicionalmente, a análise dos resultados obtidos para cada um dos projéteis, permitiu obter uma informação sobre a precisão e repetibilidade de uma experiência.

Capítulo 8

CONCLUSÕES

Nesta tese propôs-se uma solução para a experimentação remota do "*Lançamento de projéteis*", baseada num aparato, com um sistema mecânico projetado de raiz, com essa finalidade. O seu controlo elétrico e eletrónico utilizou como elemento central, uma rede baseada na norma RS-485 que interligou todos os componentes de controlo, com a finalidade de utilizar módulos autónomos, cujo programa de controlo é simples, (a estrutura geral de controlo dos microcontroladores é igual em todos os módulos), eficiente (cada módulo é autónomo nas suas funções) e fiável (sistemas com menos variáveis de entrada são mais fáceis de depurar).

Assim, o sistema de controlo foi dividido em dois grandes grupos, i) de alto nível, responsável pela implementação do serviço web, por onde são recebidos os pedidos de execução, em modo automático, o controlador de experiência, e o interface série de acesso à rede RS-485, e ii) de baixo nível, num sistema de controlo distribuído pelos vários módulos eletrónicos implementados.

No decorrer de um projeto com um tempo de execução de aproximadamente dois anos, com um âmbito pluridisciplinar, foi necessário solucionar vários problemas, desde questões logísticas, a questões mais específicas de cariz técnico. As secções seguintes apresentam alguns problemas encontrados, e algumas sugestões de trabalho futuro, agrupados segundo a natureza de enquadramento seguida nesta tese (logística, mecânica, elétrica/eletrónica, e lógica).

8.1. QUESTÕES LOGÍSTICAS

O primeiro grande problema enfrentado foi a questão relativa à aquisição de bens e serviços numa instituição pública. Como referido na secção 3.2, a necessidade de minimizar o tempo de espera no fornecimento de materiais obrigou a procurar aquisições por via de ajuste direto, ultrapassando os aspetos negativos associados à utilização da plataforma eletrónica VORTAL, a que presentemente estão condicionadas as instituições públicas portuguesas.

Este ajuste direto obrigou a justificações técnicas que além de complexas, tendem a tornar a sua autorização morosa, o que em várias alturas, condicionou o fluxo normal do projeto. Como solução, foi proposto agrupar o fornecimento maioritário de materiais e serviços por âmbito de atuação do fornecedor selecionado (Tabela 2).

O processo inicial de especificação do aparato teve origem em várias reuniões, que englobaram vários professores do Departamento de Física, com a finalidade de especificar qual a experiência que seria objeto de implementação, e posteriormente, de que forma seria implementada a experiência escolhida, i.e. a de "*Lançamento de projéteis*".

Como especificação desta experiência, foi inicialmente proposta a utilização de uma experiência manual produzida pela Cidepe™ [29], a qual seria automatizada, com recurso a uma plataforma de desenvolvimento baseada em *Arduino*™ [30] e alguns atuadores. No desenrolar do estudo inicial, cheguei à conclusão que a utilização da referida experiência manual era inviável, tendo no horizonte a criação de uma solução eficiente e profissional. Após um estudo mais pormenorizado, com um esboço 3D da ideia a implementar, realizado em Autodesk Inventor™, foi possível obter a aprovação necessária à execução de um projeto mais ambicioso.

8.2. PROJETO MECÂNICO

O projeto mecânico apresentou-se como um desafio de grande escala, tendo em conta que o próprio edital de publicação da bolsa de investigação não referia a experiência anterior nesta área científica. Contudo, o meu conhecimento prático de trabalho com sistemas mecânicos de alta precisão na indústria dos semicondutores permitiu-me responder ao desafio proposto.

Nesta área, o principal problema teve a ver com o tempo de execução de cada peça, e a necessidade de que o desenho dos vários componentes constituintes estivesse isento de erros, já que tipicamente, a sua correção implicaria fabricar uma nova peça. Foi fundamental a utilização da ferramenta 3D (Autodesk Inventor™) de forma a verificar a funcionalidade de cada subconjunto mecânico.

Ao nível da implementação, o maior problema foi o facto da resposta funcional dos veios e chumaceiras escolhidos não ser a esperada. As chumaceiras em plástico da IGUS™ apresentaram folgas que se revelaram problemáticas no funcionamento dos elevadores. Esta questão obrigou a alterar o processo de aperto para que se pudesse reduzir a folga existente. No caso do "*Elevador de bola*", a vibração durante o movimento, limitava a velocidade máxima. Para atenuar esta vibração, foi colocado um feltro (entre a base do elevador e a estrutura central) que permitiu absorver parte da vibração produzida.

Um outro problema prende-se com o ajuste do sensor da base de recolha do projétil que serve para determinar o fim de voo. Este sensor permite um ajuste vertical de aproximadamente 2 mm. No entanto o fabrico das molas de apoio da base levou a que o seu comprimento ultrapassasse o valor projetado, o que implicou na incapacidade de ajuste do referido sensor. Este problema foi solucionado com recurso a umas anilhas colocadas entre a lâmina de interceção e a base de apoio de forma a compensar a altura adicional. A solução final deste problema passa por redesenhar a lâmina de interceção do sensor para as novas medidas.

8.3. PROJETO ELÉTRICO E ELETRÓNICO

Ao nível do controlo eletrónico houve a necessidade de miniaturizar todos os circuitos desenvolvidos. Esta miniaturização teve influência não só no tipo de componentes a utilizar, mas também na cablagem do aparato. A utilização de conectores de passo reduzido (2 mm) implicou um atraso significativo na sua montagem, e teve uma duração de 34 dias úteis (Tabela 1).

O maior problema encontrado no âmbito do projeto eletrónico foi a implementação do sistema de medição. Este sistema tem por base a utilização de emissores e receptores de infravermelhos de montagem superficial, espaçados 4 mm entre si. Finalizada a montagem, e aquando dos primeiros testes, verificou-se que o sistema não funcionava como idealizado (os receptores estavam sempre em condução, o que significava estarem a receber luz, mesmo com a lente completamente tapada). Os testes em bancada provaram que os emissores de luz deixavam passar os raios infravermelhos através do plástico que, embora em pequena quantidade, dada a proximidade dos receptores, era suficiente para os colocar em condução. A resolução do problema passou por recalcular o ganho dos receptores (resistência de coletor) e por colocar um verniz negro a tapar completamente os emissores e receptores, para os isolar entre si, opticamente.

Um outro problema foi a opção de se utilizar o solenoide da Cidepe™ como retentor da bola. Este solenoide utiliza uma tensão de 200 V_{DC} e tornava perigosa a sua aplicação junto dos circuitos de controlo eletrónicos. Houve a tentativa de o rebobinar para 24 V_{DC}, mas após esta alteração,

verificou-se que a corrente necessária ao seu funcionamento era demasiado elevada (o consumo de 2,6 A obrigava à instalação de uma fonte de alimentação só para esse fim).

Uma das melhorias futuras é a substituição do retentor eletromagnético por um sistema mecânico de retenção, com controlo por motor. As vantagens são várias:

- Ao nível da segurança, é possível eliminar a existência de 230 V_{AC} no circuito impresso de controlo ("*Caixa de distribuição*" do elevador da rampa)
- Funcionalmente será possível estender a experiência a projéteis não ferromagnéticos (e.g. plástico ou madeira)
- O sistema de controlo pode utilizar os recursos já existentes na "*Caixa de distribuição*", não necessitando, em princípio, de *hardware* adicional.

Adicionalmente, como forma de reduzir o impacto do tempo necessário a elevar a bola (utiliza um fuso trapezoidal com passo de 3 mm), poder-se-á implementar um sistema de tração direta, com transmissão por cabo de aço (semelhante ao existente nas cabeças das impressoras de jato de tinta). Este processo permite cobrir a distância desejada com poucas rotações do motor, otimizando o tempo de deslocação. No entanto, esta solução requer o redesenho completo do "*Elevador de bola*".

8.4. PROJETO LÓGICO

Ao nível do projeto lógico, não foram encontrados problemas durante o seu desenvolvimento. No entanto, embora em situação completamente funcional, esta é uma das áreas com mais melhorias identificadas.

Em primeiro lugar surge o controlador de experiência. Este controlador está implementado com serviços web num computador pessoal, onde está residente o servidor web. O projeto "*Physics LabFARM*" prevê a inclusão de várias experiências remotas numa plataforma comum. Assim sendo, o servidor web deixará de ser local à experiência para passar a ser um servidor comum a todas as experiências. Nesse caso, a necessidade de existência de um PC (*Personal Computer*) só para o controlo da experiência passa a ser questionável. Ainda no âmbito deste trabalho foi aventada a hipótese de utilizar o controlador "*FriendlyArm*" [31] como controlador de experiência. Contudo, a demora na autorização de compra eliminou a possibilidade da sua inclusão no âmbito desta tese.

A passagem para este controlador não é imediata, e requer alteração na interligação ao servidor web. Neste momento, o servidor web "chama" os serviços disponibilizados pelo servidor de serviços web, implementado no código do controlador de experiência. O controlador "*Friend-*

lyArm" tem o sistema operativo Windows™ CE 6.0 e disponibiliza a programação em .NET. No entanto, a versão .NET disponibilizada neste sistema operativo (".NET Compact 3.5") não implementa os componentes de servidores necessários (apenas estão implementados os componentes cliente). Por este motivo, será necessário alterar a filosofia de funcionamento, na qual deixa de existir um pedido de execução à máquina para a existência de uma verificação por parte desta da necessidade de executar uma experiência ("*polling*").

Adicionalmente, o facto da versão ".NET Compact" não disponibilizar a infraestrutura WPF obriga a refazer na totalidade o interface de utilizador. No entanto, os ganhos com esta alteração são significativos:

- Dispensa de utilizar um PC dedicado ao controlo. O controlador "*FriendlyArm*" custa cerca de 100 €, e contém um ecrã táctil de 3,5" com recursos de rede, necessários à comunicação remota;
- Permite fazer um painel de controlo embebido na máquina para utilização em operações de reparação e manutenção;
- Adicionalmente poderá ser responsável pela captura da foto do instante de impacto, já que tem acesso em tempo real aos recursos necessários (neste momento a foto do instante de impacto é responsabilidade do servidor web, simultaneamente com a transmissão do vídeo);
- Torna o sistema mais portátil, sem necessidade adicional de teclado, rato e monitor;

Funcionalmente é possível melhorar o desempenho do aparato, alterando o modo de funcionamento do controlo de algumas secções.

A alteração do controlo do seletor de bola pode ganhar alguns segundos, se a calibração só for efetuada quando o seletor é deslocado para a primeira bola (mais próxima do sensor de limite) e manter um estado da posição atual do carro. Contudo poderá ser necessário executar a calibração se num conjunto significativo de lançamentos, não tiver sido escolhida a 'bola 1'.

Ao nível do programa residente nos microcontroladores PIC é necessária a sua reconfiguração de forma a incluir no ciclo primário um ciclo de vigilância (*Watchdog*) para prevenir eventuais bloqueios (e.g. perda de comunicação). Uma outra alteração diz respeito à forma de programação dos microcontroladores. Neste momento, é utilizado o conector ICSP de cada módulo para proceder à sua programação. Dada a existência de uma rede, é aconselhável a sua utilização para a programação dos microcontroladores. Desta forma seria possível efetuar atualizações ao código remotamente. Para tal ser possível, é necessária a inclusão de um programa residente capaz de programar o microcontrolador a partir de uma imagem recebida pela porta série ("*Boot loader*").

O programa monitor dos sensores de bola deve ser alterado de forma a incluir um código binário único. Esta medida permite eliminar situações de falsas receções devidas a reflexão dos raios infravermelhos provenientes de outros sensores de bola na proximidade.

Finalmente, a inclusão de uma câmara PTZ ("*Pan, Tilt & Zoom*"), semelhante às utilizadas em vídeo vigilância, permitirá resolver o problema ótico da câmara de vídeo atual, cuja distância focal não permite que a mesma se encontre instalada no interior da estrutura do aparato. A utilização de uma câmara PTZ permitirá ao controlador de experiência controlar a câmara de forma a acompanhar os passos intermédios da sua execução, enviando-lhe as suas coordenadas.

8.5. CONSIDERAÇÕES FINAIS

Este trabalho, graças à diversidade de temas que abordou, foi um desafio constante, que à medida que os problemas surgiam, se tornava mais aliciante. Aliar o projeto eletrónico e de controlo, (na área da engenharia eletrónica e computadores, no ramo de automação e controlo), ao projeto mecânico é um trabalho muito motivador.

Este trabalho serve, de uma forma geral, para apresentar um modelo de controlo em automação - a utilização remota - que requer um projeto cuidado em cada uma das suas áreas, para reduzir as possibilidades de erro.

A presença na conferência internacional sobre engenharia remota e instrumentação virtual (REV), que teve a sua edição de 2012 [3], em Bilbao, Espanha, no mês de Julho, permitiu fazer a demonstração deste projeto inovador, resultante do trabalho desenvolvido no último ano, e publicar um artigo indexado no IEEE Xplore [4]. Uma nova publicação [32], sobre a comparação entre vertentes "Profissionais" e "Académicas" deste projeto, foi submetida à próxima edição desta conferência (REV 2013).

A concluir, refere-se ainda que este trabalho foi distinguido com o prémio "REV 2012 – *Best Demo Award*", nesta mesma conferência [33].

Referências

[1] Alves, G. R.; Marques, M. A.; Viegas, C.; Lobo, M. C.; Barral, R. G.; Couto, R. J.; Jacob, F. L.; Ramos, C. A.; Vilão, G. M.; Covita, D. S.; Alves, J.; Guimarães, P. S.; Gustavsson, I.; (2011). "*Using VISIR in a large undergraduate course: Preliminary assessment results*", *International Journal of Engineering Pedagogy*, 1 (1), pp. 12-19. ISSN: 2192-4880

[2] Nuno Sousa, Gustavo R. Alves, e Manuel G. Gericota, "*An Integrated Reusable Remote Laboratory to Complement Electronics Teaching*", *IEEE Transactions on Learning Technologies*, Julho-Setembro 2010, Vol. 3, nr. 3, pp. 265 – 271, ISSN: 1939-1382

[3] Conferência Internacional "*REV - Remote Engineering & Virtual Instrumentation*", Edição de 2012, Bilbau, Espanha, http://www.rev-conference.org/REV2012/, verificado em Outubro de 2012.

[4] Carlos Paiva; Pedro Nogueira; Gustavo Alves; Arcelina Marques; Pedro Guimarães; Ruben Couto; "*A Flexible Online Apparatus for Projectile Launch Experiments*", *IEEE Conference Publications, Remote Engineering and Virtual Instrumentation (REV)*, 2012 *9th International Conference*, Setembro 2012, ISBN: 978-1-4673-2540-0

[5] "*The Physics Hypertext Book*", http://physics.info/motion/ , acedido em Outubro de 2012

[6] Paul A. Tipler, "Física, Volume 1 – Mecânica", Livros Técnicos e Científicos Editora S.A., 3ª edição, 1995

[7] Froyd, J.E.; Wankat, Phillip C.; Smith, Karl A.; "*Five Major Shifts in 100 Years of Engineering Education*", *Proceedings of the IEEE*, Vol. 100, Maio 2012, ISSN:0018-9219, pp. 9 – 11

[8] John A. Sokolowski; Catherine M. Banks, "*Principles of Modeling and Simulation, A multidisciplinary Approach*", John Wiley Publication, 2009, pp. 3 – 24

[9] Carlos Felgueiras, "Apoio à depuração e teste de circuitos mistos compatíveis com a norma IEEE 1149.4", Tese de doutoramento, FEUP, Porto, Portugal, Cap. 2

[10] Simulador, "*Projectile Motion*", "*PhET Interactive Simulations*", Universidade do Colorado, http://phet.colorado.edu/sims/projectile-motion/projectile-motion_en.html, verificado em Outubro de 2012

[11] Simuladores, "*Fowler's Physics Applets*", Universidade da Virgínia, EUA, http://galileoandeinstein.physics.virginia.edu/more_stuff/Applets/home.html, verificado em Outubro de 2012

[12] Simulador, "*Projectile Motion*", Universidade da Virgínia, EUA, http://galileoandeinstein.physics.virginia.edu/more_stuff/Applets/ProjectileMotion/jarapplet.html, verificado em Outubro de 2012

[13] Juarez Bento da Silva, "A Utilização da Experimentação Remota como Suporte para Ambientes Colaborativos de Aprendizagem", Tese de doutoramento, Universidade Federal de Santa Catarina, Florianópolis, Brasil, 2006, Cap. 6

[14] *Learning Management System* (LMS), https://moodle.org/, verificado em Outubro de 2012

[15] *OpenLabs Electronics Laboratory*, VISIR *project*, Suécia, http://openlabs.bth.se/electronics/, verificado em outubro de 2012

[16] "rexlab – Laboratório de Experimentação Remota", Universidade Federal de Santa Catarina, Ararangua, Santa Catarina, Brasil, http://rexlab.ararangua.ufsc.br/experimentos/young/, verificado em outubro de 2012

[17] M. Ožvoldová; F. Schauer; M. Beňo; "*Remote Free Fall Experiment for Dynamic Studies*", 3rd *International Symposium for Engineering Education*, 2010, *University College Cor*k, Ireland

[18] ANSI/TIA/EIA-485-A-1998. "*Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems*", *Telecommunications Industry Association*, 1998

[19] Nanotec, "*Programming Manual for Stepper Motor Positioning Controls*", Nanotec Electronic GmbH & Co. KG, firmware version 25, Aug, 2011, pp. 10 – 12

[20] Usb.org, "*Universal Serial Bus Specification Revision 2.0*", 7.2 *Power Distribution*, pp. 171 – 178.

[21] Gamma, Erich; Helm,Richard; Johnson, Ralph; Vlissids, John; "*Design patterns, Elements of Reusable Object-Oriented Software*", Addison-Wesley, ISBN:0-201-63361-2 (Outubro de 1994), pp. 144–149

[22] MSDN, "*Implementing Singleton in C#*", Microsoft, http://msdn.microsoft.com/en-us/library/ff650316.aspx, verificado em Outubro de 2012.

[23] Fornecedor, "Tecnogial, Projetos e Tecnologia Industrial, Lda.", Perafita, Matosinhos, Portugal, http://www.tecnogial.pt/, verificado em Outubro de 2012

[24] Fornecedor, "Bibus, Lda.", S. Mamede de Infesta, Matosinhos, Portugal, http://www.bibus.pt/pt/inicio/, verificado em Outubro de 2012

[25] Fornecedor, "Nanotec", Munique, Alemanha, http://en.nanotec.com/start.html, verificado em Outubro de 2012

[26] Fornecedor, "Digikey Corporation ", Minesota, EUA, http://www.digikey.pt/, verificado em Outubro de 2012

[27] Fornecedor, "PCB-Pool, Beta LAYOUT ", Shannon, Irlanda, http://www.pcb-pool.com/ppuk/index.html, verificado em Outubro de 2012

[28] Célia A. De Sousa; Elisa P. Pina, "A lei da conservação da energia: aplicação ao rolamento com e sem deslizamento", Gazeta da Física, http://www.gazetadefisica.spf.pt/magazine/article/534/pdf, verificado em Outubro de 2012

[29] Fornecedor, "Cidepe, Centro Industrial de Equipamentos de Ensino e Pesquisa", Canoas, Rio Grande do Sul, Brasil, http://www.cidepe.com.br/, verificado em Outubro de 2012

[30] Arduino, "*Open-source electronics prototyping platform*", http://www.arduino.cc/, verificado em Outubro de 2012

[31] FriendlyArm, "*ARM based Development Boards and Modules*", http://www.friendlyarm.net/, verificado em Outubro 2012

[32] Gustavo Alves; Arcelina Marques; Carlos Paiva; Pedro Nogueira; Pedro Guimarães; Rubem Couto; Laryssa Cherem; Vítor Borba; Golberi Ferreira; Stefan Koch; Andreas Pester, "A Remote Lab for Projectile Launch Experiments: Professional and academic perspectives", Submetido em Outubro de 2012 para a conferência internacional REV2013, a realizar em Sidney, Austrália

[33] REV2012 *Awards*, http://www.rev-conference.org/REV2012/, Awards New! "*Best demo award*" verificado em Outubro de 2012.

Anexo

O anexo desta tese está dividido em dois conjuntos:

- Impresso, nas páginas seguintes com o seguinte índice:

  Anexo A. CÁLCULO DA FONTE COMUTADA COM CI LM2574
  Anexo B. INTRODUÇÃO AO HARDWARE UTILIZADO
  Anexo C. TECNOLOGIAS MICROSOFT™ PARA WINDOWS™
  Anexo D. SERVIÇOS WEB
  Anexo E. ESQUEMAS
  Anexo F. ESTRUTURA DE IMPLEMENTAÇÃO – APLICAÇÃO WINDOWS™
  Anexo G. ESTRUTURA DE IMPLEMENTAÇÃO – MICROCONTROLADORES
  Anexo H. ARTIGO PUBLICADO NA CONFERÊNCIA INTERNACIONAL REV2012

- Em formato eletrónico, com a seguinte estrutura de ficheiros:

\Datasheets	– Folhas de dados dos componentes utilizados
\Esquemas	– Ficheiros em PDF de todos os esquemas
\Estado da arte	– Ficheiros em PDF dos artigos consultados
\Fotos	– Fotos do aparado
\Images	– Imagens utilizadas em toda a tese
\Logs do aparato	– Ficheiros obtidos para recolha de dados
\Normas	– Normas utilizadas
\Papers	– Ficheiros em PDF dos artigos publicados
\Pcb	– Ficheiros em PDF dos vários circuitos impressos
\Visio	– Ficheiros de VISIO com os diagramas
\Tese Final.pdf	– Ficheiro em PDF deste documento
\Anexos.pdf	– Ficheiro em PDF dos anexos deste documento

Anexo A. CÁLCULO DA FONTE COMUTADA COM CI LM2574

***Resumo*— Cálculo dos componentes das fontes comutadas, baseadas no CI LM2574.**

A Tabela 1 apresenta os requisitos da fonte.

Tabela 2 – Requisitos da fonte comutada

Dados da fonte	
Tensão de entrada:	+24V
Tensão de saída:	+5V
Corrente de carga máxima:	400 mA

A Figura 1 apresenta o esquema tipo de uma fonte comutada, baseada no CI LM2574M na National Instruments™ [1].

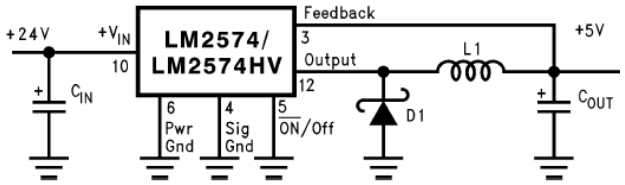


Figura 1 Esquema tipo de fonte comutada baseada no CI LM2574M

I. DETERMINAÇÃO DA BOBINA

O fabricante apresenta o gráfico da Figura 2 como guia de seleção da bobina (L1) em função da tensão máxima de entrada e da corrente máxima na carga. A linha vermelha define a tensão de entrada desejada (+24 V). É possível verificar que com uma bobina de 330 µH o regulador fornece correntes superiores a aproximadamente 425 mA.

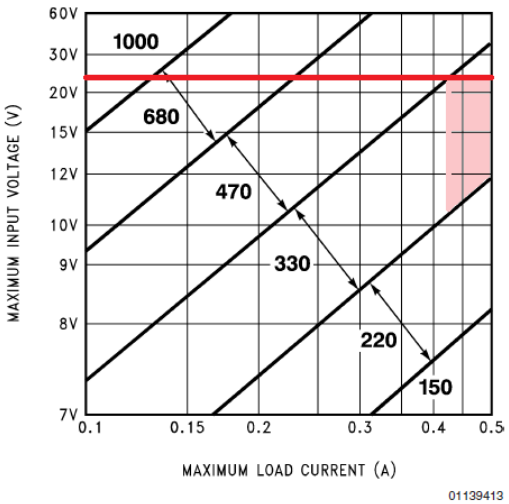


Figura 2 Seleção da bobina do CI LM2574-5.0 (µH)

Esta bobina deve estar dimensionada para operar à frequência de comutação do regulador (52 kHz) e suportar uma corrente 1,5 vezes maior que a corrente necessária (750 mA).

II. DETERMINAÇÃO DO CONDENSADOR DE SAÍDA

O valor do condensador de saída juntamente com a bobina define o par de polos dominantes da realimentação do regulador comutado. Para o funcionamento estável e uma tensão de *ripple* de saída aceitável (aproximadamente 1% da tensão de saída) é recomendado pelo fabricante a utilização de um condensador com o valor entre 100 µF e 470 µF. A tensão do condensador deve ser no mínimo 1,5 vezes maior que a tensão de saída do regulador. Os condensadores eletrolíticos de maior tensão, têm normalmente um valor equivalente da resistência em série (ESR) menor e por essa razão foi escolhido um condensador de 220 µF para 16 V.

III. SELEÇÃO DO DÍODO DE CAPTURA

A corrente do díodo de captura (D1) deve ser pelo menos 1,5 vezes a corrente da carga. Se a fonte de alimentação for projetada para suportar um curto-circuito contínuo na saída, o díodo deve suportar pelo menos a corrente de limite máxima do regulador. A condição de maior dificuldade de funcionamento para o díodo é na presença de uma sobrecarga ou de um curto-circuito. A tensão inversa do díodo deve ser igual ou superior a 1,25 vezes a tensão de entrada do regulador. Foi selecionado o díodo Schottky NSR1020MW2T1G que suporta uma corrente direta (IF) de 1 A e uma tensão inversa máxima de 30 V (VR).

IV. REFERENCIAS

[1] LM2574/LM2574HV Datasheet, National Semiconductor, Julho de 1999.

Anexo B. INTRODUÇÃO AO *HARDWARE* UTILIZADO

***Resumo*— O *hardware* utilizou alguns conceitos e componentes que requerem uma descrição mais detalhada.**

I. NORMA RS-485

A norma TIA/EIA RS-485-A com o título "*Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems*" define as caraterísticas dos circuitos de interface RS-485[1]. As normas RS-422 e RS-485 tal como são conhecidas no presente, são esquemas de transmissão balanceada de dados que oferecem solução robustas para a sua transmissão a grande distância e em ambientes ruidosos [2]. A norma RS-485 foi escrita de forma a ser eletricamente compatível com a norma RS-422. Uma rede pode ser partilhada por até 32 pares transmissores/recetores.

A. SINAIS E TRANSMISSÃO DE DADOS

A norma define dois estados da rede:

- MARK ou 1 (estado desligado) – Quando o terminal "A" do transmissor é negativo relativamente ao terminal "B".
- SPACE ou 0 (estado ligado) – Quando o terminal "A" do transmissor é positivo relativamente ao terminal "B"

B. CONTROLO DA SAÍDA DOS TRANSMISSORES

Para ser possível mais de um transmissor enviar dados pelo barramento, é necessário que a sua saída seja colocada em alta-impedância no final da transmissão. Alguns sistemas usam o estado da linha para gerar o controlo, enquanto outros utilizam o programa de controlo (protocolo de acesso ao barramento) para gerir os acessos ao barramento.

C. TOPOLOGIA DA REDE

A configuração da topologia da rede não é definida pela norma RS-422 ou RS-485. Normalmente o projetista usa a configuração que melhor se adapta aos requisitos físicos do sistema [3]. A Figura 1 apresenta o diagrama da topologia de rede a dois fios (*MULTIPOINT*).

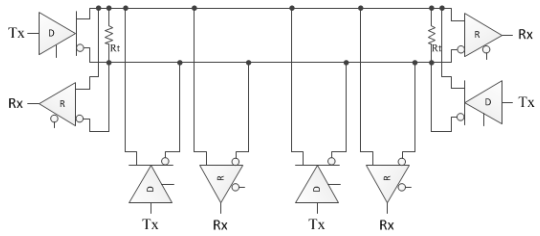


Figura 1 Topologia de rede a dois fios (*MULTIPOINT*)

A Figura 1 apresenta a topologia a quatro fios (MULTIDROP).

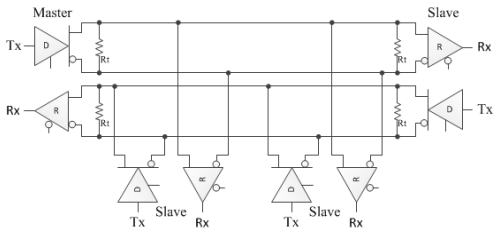


Figura 2 Topologia de rede a quatro fios (*MULTIDROP*)

A topologia de rede a dois fios é obrigatória sempre que for necessária a existência de mais de um mestre (*Master*) na rede. Como é possível verificar no diagrama da topologia a quatro fios, esta só permite um mestre (*Master*) na rede, o que impede que os escravos (*Slave*) comuniquem entre si.

D. TERMINAÇÃO DO BARRAMENTO

No projeto de um sistema que utiliza transmissores, recetores e/ou transcetores de acordo com a norma RS-485 é necessária uma cablagem e respetiva terminação apropriadas para obter aplicações fiáveis sem reflexões no barramento (ou minimizadas). De uma forma geral, as redes RS485 requerem a utilização da terminação em ambas as extremidades do cabo. A existência de uma terminação paralela (Rt) tem a vantagem de eliminar as reflexões, o que permite o aumento da distância de comunicação bem como das taxas de transmissão.

No entanto tem como desvantagem o aumento da dissipação de potência no amplificador de transmissão e a tensão diferencial de entrada dos recetores ser nula quando todos os transmissores estiverem em descanso (IDLE), e se os cabos de ligação à rede principal forem longos, são reintroduzidas reflexões. No caso da topologia da rede ser MULTIPOINT, então a terminação paralela tem de ser colocada em ambas as extremidades do barramento, o que aumenta ainda mais a carga dos transmissores.

II. TRANSCETOR RS-485

Para o interface ao barramento RS-485 foi escolhido o circuito integrado (CI) SN65HVD3080E da Texas Instruments (TI) com o diagrama apresentado na Figura 3. Este CI tem um transmissor e um recetor balanceado projetado para ser utilizado num barramento RS-422 ou RS-485 em modo *full-duplex* e é totalmente compatível com a norma TIA/EIA RS-485A [4].

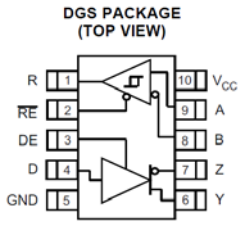


Figura 3 Diagrama do transcetor RS-485

Esta unidade suporta taxas de transmissão entre 200 kbps e 20 Mbps. A corrente de alimentação (I_{CC}) é inferior a 1 mA (excluindo a carga). No modo inativo, esta corrente desce para poucos nA o que o torna indicado para aplicações sensíveis ao consumo. A caixa disponibiliza os pinos 2 (RE*) e 3 (DE) como linhas de controlo respetivamente do recetor (ativo a '0') e do transmissor (ativo a '1'). A tensão de alimentação (V_{CC}) pode variar entre 4,5 V e 5,5 V. A tensão admissível nos pinos do barramento A, B, Z, Y (V_{IC}), deve estar compreendida entre -7 V e +12 V. O módulo da corrente máxima de saída $|I_O|$ é 60 mA.

III. CIRCUITO RECETOR DE INFRAVERMELHOS

O elemento central do projeto do sensor de bola é o CI TSOP36238 (Figura 4). Este CI é um módulo de receção de infravermelhos em tecnologia de superfície (SMT) de tamanho reduzido [5].

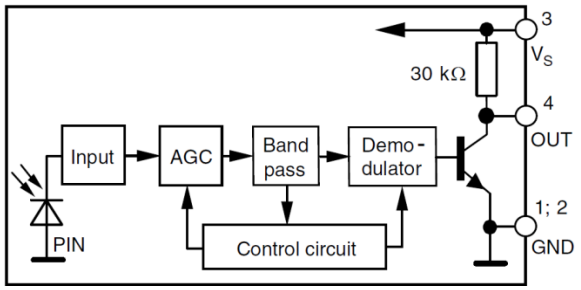


Figura 4 Recetor de infravermelhos CI TSOP36238

O sinal desmodulado de saída pode ser descodificado diretamente por um microprocessador e tem uma frequência portadora de 38 kHz. As suas principais características são:

- Foto-detetor e pré-amplificador na mesma caixa;
- Filtro interno para a frequência PCM (Pulse Code Modulation);
- É possível a transmissão contínua;
- Saída é ativa a '0';
- Baixo consumo;
- Grande imunidade à luz ambiente;
- Alimentação entre 2,7 V e 5,5 V;
- Comprimento de onda (λP): 950 nm;

Este módulo foi desenhado para eliminar impulsos devidos a ruido ou a sinais de perturbação. Para tal utiliza um filtro passa-banda, um estágio integrador e um controlo automático de ganho (AGC). Os parâmetros que distinguem entre sinal válido e perturbação são a frequência da portadora, o comprimento do trem de impulsos (*BURST*) e o *duty cycle*. Para um sinal recebido ser válido tem de preencher os seguintes requisitos:

- A frequência portadora deve estar próxima da frequência central do filtro passa-banda:
- Um BURST deve ter entre 10 e 70 impulsos, com um intervalo de pelo menos 14 impulsos entre BURST's;
- Por cada BURST maior que 1,8 ms é necessário um intervalo de pelo menos 6 vezes maiores que o tempo de BURST;
- Podem ser recebidos até 800 BURST's por segundo de forma contínua;

Como emissor de raios infravermelhos foi selecionado o díodo TSML1020 com as seguintes características principais:

- Caixa do tipo SMD de dimensões reduzidas (2,5 mm x 2,0 mm x 2,7 mm);
- Comprimento de onda de pico (λP): 940 nm;
- Elevada potência e intensidade radiantes;
- Ângulo de emissão: ± 12°;
- Tensão direta (VF): 1,2 V @ {20 mA, tp= 20 ms}; 2,2 V @ {1 A, tp= 100 µs};

IV. CONVERSOR USB→RS-485

O conversor tem por base o CI FT232RL da FTDI™ cujo diagrama de blocos é apresentado na Figura 5 [6].

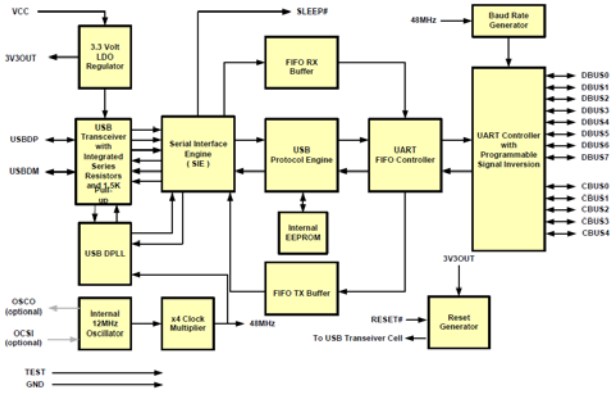


Figura 5 Conversor USB→SÉRIE CI FT232RL

Este CI é um interface de USB para UART série com características avançadas:

- Todo o protocolo USB é manipulado pela unidade;
- A geração de relógio é interna sem necessidade de utilização de cristal externo;

- As resistências de terminação do barramento são internas;
- Pinos de I/O configuráveis (CBUS);
- Sinais de indicação de receção e transmissão;
- Suporte para alimentação pelo barramento USB ou alimentação externa;
- EEPROM interna de 1024 bit para configuração interna;
- *Buffers* internos (128 bytes para receção e 256 bytes para transmissão);
- Os barramentos têm conversores de nível para interface entre +1,8 V e +5 V;
- Compatível com a norma USB 2.0;

A FTDI™ disponibiliza um *driver* denominado "*Virtual Com Port*" (VCP) que facilita o interface às aplicações (elimina a necessidade de desenvolvimento do driver USB).

A EEPROM interna é utilizada para guardar os descritores de configuração USB, "*Vendor Identifier*" (VID), "*Product Identifier*" (PID). Esta memória também é utilizada para a configuração das funções dos pinos do barramento CBUS. Esta memória é programável diretamente pelo barramento USB, utilizando o utilitário de programação da FTDI™ FT_PROG[1].

O regulador interno LDO de +3,3 V gera a tensão de referência para o transcetor USB e requer um condensador de desacoplamento externo. Este regulador pode servir para alimentar uma pequena carga externa (máximo 50 mA).

O transcetor USB providencia o interface físico ao cabo USB compatível com as normas USB 1.1 e USB 2.0. O andar de saída fornece um controlo do *slew rate* dos sinais, e o andar diferencial de entrada providencia a entrada de dados e as condições "*Single Ended 0*" (SE0) e RESET. Este transcetor também inclui as resistências de terminação do barramento.

O USB DPLL sincroniza o sinal "*Non Return to Zero Inverted*" NRZI de entrada e gera os sinais de *CLOCK* e *DATA* para o motor de interface serie (SIE).

O oscilador interno de 12 MHz gera a frequência de referência interna. Este sinal é usado no multiplicador do relógio, e nos estágios SIE, *FIFO's* e no controlador do Protocolo USB. O multiplicador/divisor do relógio gera as frequências 6 MHz, 12 MHz, 24MHz e 48 MHz a partir do oscilador interno. A frequência de 48 MHz é utilizada pelo bloco USB DPLL e pelo gerador de *baud rate*.

[1] O utilitário FT_PROG é disponibilizado em http://www.ftdichip.com/Support/Utilities/FT_Prog%20v2.6.8.zip

O motor de interface série (SIE) executa a conversão série/paralelo e paralelo/série da informação. Simultaneamente, de acordo com a norma USB 2.0, faz a geração do código para a verificação de redundância cíclica (CRC) CRC5/CRC16. Também faz a verificação do CRC na trama de dados de USB.

O gestor de protocolo USB (*Protocol Engine*) manipula a informação da trama de dados do ponto de vista da unidade USB. Este gestor manuseia os pedidos de baixo nível do protocolo USB gerados pelo controlador USB e os comandos para controlar os parâmetros funcionais da unidade universal e assíncrona de transmissão/receção (UART).

Os dados recebidos pela unidade são guardados numa memória do tipo FIFO de 128 byte "*First-In, First-Out*" (FIFO RX), e posteriormente movidos para o registo de envio do controlador do UART a fim de serem enviados pelo canal de comunicação série. Os dados recebidos pelo registo de receção do controlador do UART são guardados na memória FIFO de 256 byte (FIFO TX). O controlador USB remove a informação desta memória ao enviar um pedido de dados.

O controlador do UART, juntamente com as memórias FIFO manipula a transferência de dados entre as memórias FIFO e os registos de transmissão e receção. Aqui é efetuada a conversão de paralelo/série e série/paralelo com 7 ou 8 bits de dados para o interface RS-232, RS-422 ou RS-485.

O gerador de *baud* fornece o relógio de entrada para o controlador do UART a partir da frequência de referência de 48 MHz. Tem um *prescaler* de 14 bit e três bits de ajuste fino que determinam o *baud rate* do UART entre 183 *baud* e 3 *Mbaud*. O gerador de RESET fornece no arranque um sinal de inicialização fiável, sendo possível ser sincronizado externamente pela entrada RESET#.

V. MICROCONTROLADOR PIC16F1939

Para o controlo foi escolhido o microcontrolador PIC16F1939 da MICROCHIP™. Este microcontrolador contém um microprocessador RISC (*Reduced Instruction Set Computer*), cujo conjunto de instruções foi minimizado e otimizado para a execução rápida e eficiente do processamento de dados [7]. O microprocessador tem um conjunto de 49 instruções que são executáveis num único ciclo de instrução (com exceção das instruções de salto). O microcontrolador, que trabalha com uma velocidade entre 0 Hz e 32 MHz a partir de um oscilador interno (com um ciclo de instrução entre 0 ns e 125 ns), disponibiliza 16K x 14 palavras de memória de programa do tipo FLASH, 1024 bytes de memória RAM ("*Random Access*

Memory") para dados voláteis e 256 bytes de memória EEPROM ("*Electrically Erasable Programmable Read Only Memory*") para dados não voláteis. As interrupções gravam automaticamente o contexto e tem uma pilha (*Stack*) de 16 níveis. O processador disponibiliza os modos de endereçamento direto, indireto e relativo, com acesso de leitura à memória FLASH.

Este microcontrolador disponibiliza 35 pinos de entrada/saída (I/O) com capacidade de fornecer correntes elevadas e programar individualmente a geração de interrupções na mudança de estado e a existência de resistências "*Pull-Up*". Também contém um vasto conjunto de periféricos que partilham os vários pinos de entrada/saída:

- Conversor A/D de 10 bit de resolução com 14 canais independentes;
- Quatro Timers de 8 bit e um Timer de 16 bit com PRESCALER programável;
- Dois módulos de captura, comparação e PWM ("*Pulse Width Modulation*") de 16 bit;
- Porta série síncrona com capacidade de interface a barramentos I2C e SPI;
- Porta série assíncrona compatível com RS-232 e RS-485;
- Trinco SR para emulação do temporizador NE555;
- Duplo comparador com histerese;
- Módulo de tensão de referência com DAC ("*Digital Analog Converter*") de 5 bit;

A Figura 6 apresenta o diagrama de blocos do microcontrolador PIC16F1939.

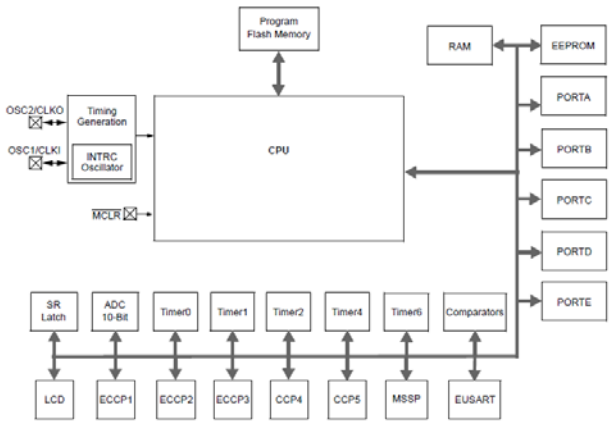


Figura 6 Diagrama de blocos do microcontrolador PIC16F1939

O diagrama da Figura 7 mostra a estrutura da unidade central de processamento (CPU).

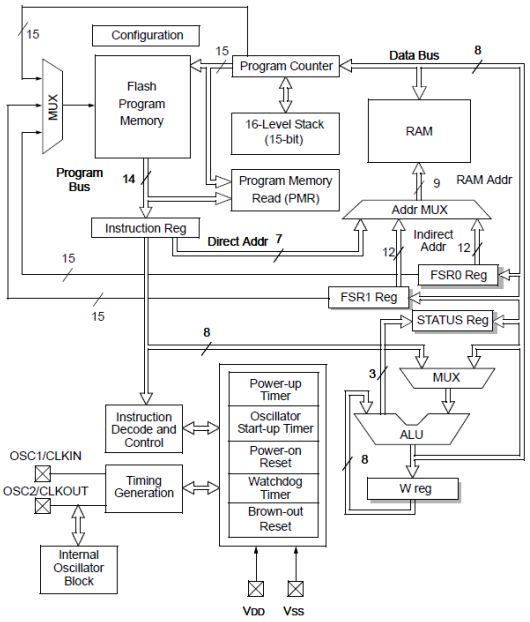


Figura 7 Diagrama da CPU

A CPU tem duas estruturas de memória distintas. A memória de programa é implementada em memória FLASH e é normalmente só de leitura. No entanto, este microcontrolador disponibiliza uma forma de escrita a partir do programa pela utilização de um grupo dedicado de registos. A memória de dados é subdividida em 32 bancos de 128 bytes. Cada banco é subdividido da seguinte forma:

- Registos do núcleo comuns a todos os bancos (12 bytes);
- Registos de funções especiais (20 bytes);
- Memória RAM de uso geral (80 bytes);
- Memória comum a todos os bancos (16 bytes);

Os registos do núcleo, associados diretamente às operações da CPU, estão disponíveis simultaneamente em todos os bancos. Entre eles contam-se o registo de estado (STATUS) e o contador de programa (PCL e PCLATCH) de 15 bit que é colocado a 0x0000 após a inicialização (RESET). A pilha é um conjunto dedicado de 16 registos de 15 bit, de implementação separada da memória de dados. Esta pilha funciona como *buffer* circular o que leva à corrupção do endereço 0x00 da pilha após a 16 escrita. Os registos de funções especiais estão diretamente associados aos periféricos implementados no microcontrolador. Estes registos são de leitura, escrita ou leitura e escrita, e alguns tem a opção de serem endereçados ao bit. A memória RAM disponível para dados (1024 bytes) está implementada nos bancos [0…12] no bloco de memória RAM de uso geral. Este microcontrolador prevê um acesso linear à memória de dados e de programa pela utilização de um apontador de 16 bit formado pelos registos do núcleo FSRH e FSRL.

VI. BARRAMENTO SPI

O barramento SPI é um sistema padrão de interligação série síncrono, definido pela Motorola™ e que funciona em modo *full-duplex*. Este barramento estabelece uma comunicação *Master/Slave* em que a comunicação é sempre iniciada pelo *master* e destina-se à comunicação entre circuitos integrados. A existência de vários *slaves* no barramento requer a utilização de linhas individuais de permissão ("*chip select*") [8]. O barramento define um conjunto de quatro sinais:

- SCLK – Relógio série, com origem no Master;
- MOSI – Saída do master e entrada no *slave* ("*master out, slave in*);
- MISO – Entrada do master e saído do *slave* ("*master in, slave out*);
- SS – Permissão do *slave* ("*slave select*"), com origem no master;

Em alternativa, é comum a utilização dos acrónimos SCK ("*Serial clock*"), SDI ("*Serial data in*"), SDO ("*Serial data out*") e CS ("*Chip select*") respetivamente. Normalmente o sinal de permissão é ativo a '0'. No caso de só existir um único *slave*, a linha de permissão pode estar sempre ativa, desde que o *slave* o aceite (há unidades em que a comunicação é iniciada no flanco descendente do sinal SS). Tipicamente o *hardware* de comunicação é implementado com registos de deslocamento montados em anel como mostra a Figura 8.

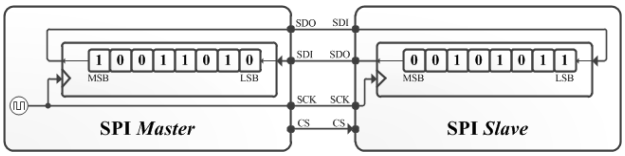


Figura 8 Diagrama de interligação SPI

O *master* inicia a comunicação ativando a linha de permissão do *slave* de destino e, após uma pausa (de acordo com o tipo de unidade), inicia o envio de ciclos de relógio na linha SCLK (SCK). Para cada ciclo de relógio existe uma transmissão em *full-duplex*. O master envia um bit pela linha MOSI (SDO) e simultaneamente, recebe um bit pela linha MISO (SDI). Os registos de deslocamento enviam primeiro o bit mais significativo. O número de bits envolvidos na comunicação não está definido, mas tipicamente o *master* tem um registo de deslocamento de 8 bits e os *slaves* são organizados em múltiplos de 8 bits. Se um *slave* no barramento está inativo, não pode receber os sinais SCLK e MOSI e deve ter o sinal MISO em alta impedância.

A especificação SPI define o modo de interação do relógio pelas opções CPOL ("*Clock polarity*") e CPHA ("*Clock phase*"). A opção CPOL=0 define que o nível em repouso do relógio é '0' e por consequência a sua primeira transição é o flanco ascendente. Se CPOL=1, o nível em repouso é '1' e a sua primeira transição é o flanco descendente. A opção CPHA=0 define que o bit de dados é capturado no primeiro flanco do relógio (ascendente se CPOL=0, descendente se CPOL=1). CPHA=1 define que o bit de dados é capturado no segundo flanco do sinal de relógio. Há duas configurações SPI de múltiplos *slaves*:

- De forma independente: Cada *slave* tem uma linha de permissão independente;
- Em cadeia ("*daisy chain*"): A saída do master é a entrada do primeiro *slave*. A saída desse *slave* é a entrada do seguinte, etc. e por fim a saída do último *slave* é a entrada do master;

Principais vantagens deste barramento:

- Comunicação simultânea de entrada e saída (*full-duplex*)
- Taxa de transmissão elevada;
- Flexibilidade da informação (não está limitada a palavras de 8 bits);
- O interface de *hardware* é muito simples e reduz-se normalmente a registos de deslocamento;
- Todos os sinais são unidirecionais o que permite o isolamento galvânico;
- Não limita a frequência de relógio;

Algumas das suas desvantagens:

- Requer mais pinos de permissão;
- Não é possível o endereçamento com base nos dados enviados;
- Não existe controlo de fluxo, correção de erros ou sinalização de receção válida;
- Só pode existir um master no barramento;
- Destinado unicamente a comunicação a curtas distâncias, normalmente no mesmo PCB;

VII. CONTROLADOR DE LED'S

Para o controlo dos LED's foi escolhido o circuito integrado STP16CP05 da ST Microelectronics™ que é um *driver* de corrente constante de 16 canais [9]. A Figura 9 apresenta o seu diagrama de blocos. O interface série é composto pelo registo de deslocamento de 16 bits do tipo "entrada série"/"saída paralela". O registo de deslocamento tem associado em paralelo, um registo de dados onde a informação enviada pelo interface série, é armazenada por ação do sinal LE ("*Latch enable*"). O andar de saída de cada um dos bits do registo de dados é composto por uma fonte de corrente constante, programável por uma resistência externa, que fornece uma corrente entre 5 mA

e 100 mA, para uma tensão máxima de 20 V. O CI suporta tensões de alimentação entre 3,3 V e 5,0 V.

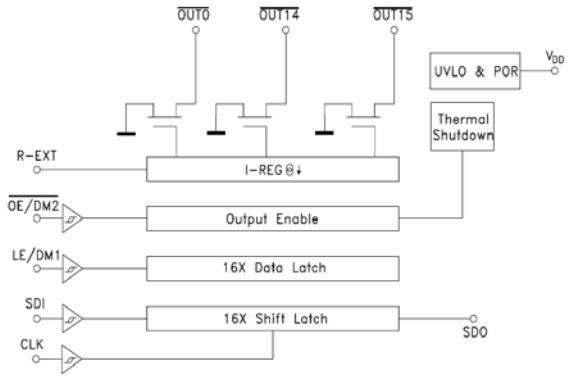


Figura 9 Diagrama de blocos do STP16CP05

Na Figura 10 é apresentado o seu diagrama temporal, onde é possível verificar que os dados existentes no registo de deslocamento são guardados no registo de dados por ação do sinal LE e são disponibilizados nas saídas enquanto habilitados pelo sinal OE.

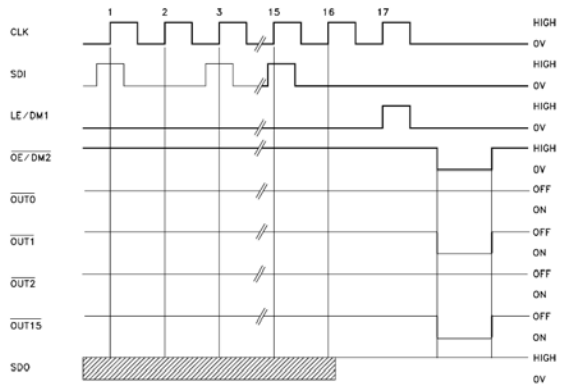


Figura 10 Diagrama temporal do STP16CP05

VIII. REGISTO DE DESLOCAMENTO DO TIPO ENTRADA PARALELO/SAÍDA SÉRIE

Para o acesso série aos fotorreceptores via interface SPI, foi escolhido o registo de deslocamento 74HC166, do tipo "entrada paralela"/"saída série" com 8 bits cujo diagrama lógico é apresentado na Figura 11 [10].

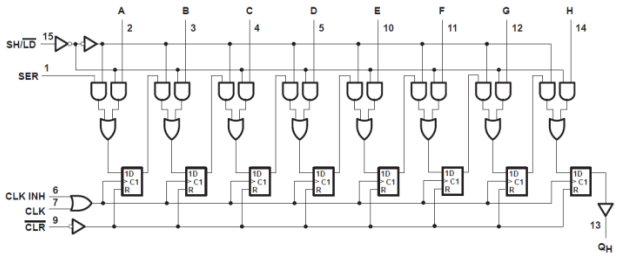


Figura 11 Diagrama lógico do CI 74HC166

O sinal SH/LD* = '1' habilita a entrada série SER e permite o deslocamento dos bits a cada transição do sinal de relógio CLK, que aparece externamente no pino Q_H. Quando SH/LD* = '0' ficam habilitadas as entradas paralelas (A…H) que serão carregadas nos respetivos flip-flops na transição seguinte do relógio. O deslocamento ocorre no flanco ascendente do relógio, quando permitido pelo sinal CLKINH = 0. O sinal CLR* limpa assincronamente todos os flip-flops. O diagrama temporal da Figura 12 mostra os vários modos de funcionamento.

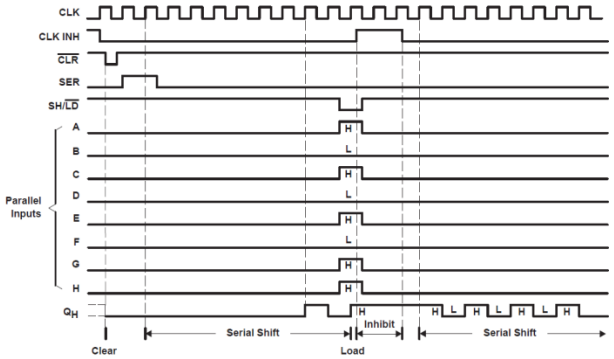


Figura 12 Diagrama temporal do CI 74HC166

Referencias

[1] ANSI/TIA/EIA-485-A-1998. "Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems", Telecommunications Industry Association, 1998

[2] Soltero, Manny; Zhang, Jing; Cockril, Chris. "RS-422 and RS-485 Standards Overview and System Configuration", Texas Instruments, Application Report, Revisão Maio, 2010

[3] B&B Electronics. "*RS-422 and RS-485 Application Note*", B&B Electronics, Application Note, Revisão Junho 2006, pp. 13 – 19.

[4] Datasheet 'SN65HVD3080E', "Low-Power RS-485 Full-Duplex Drivers/Receivers", Texas Instruments, Novembro de 2006.

[5] Datasheet 'TSOP362…'," IR Receiver Modules for Remote Control Systems", Vishay, Fevereiro 2012

[6] Datasheet 'FT232R', "FT232R USB UART IC", Future Technology Devices International Ltd, version 2.10, 2010

[7] Datasheet 'PIC16F193x/LF139x', "28/40/44-Pin Flash-Based, 8-Bit CMOS Microcontrollers with LCD Driver and nanoWatt XLP Technology", Microchip Technology, DS41364D, 2009

[8] Wikipedia, "*Serial Peripheral Interface Bus*", http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus, verificado em Outubro de 2012

[9] Datasheet 'STP16CP05', "Low voltage 16-bit constant current LED sink driver", ST Semiconductors, Revisão 10, Janeiro de 2010

[10] Datasheet 'SN74HC166', "8-bit Parallel-Load Shift Registers", Texas Instruments, Revisão Setembro de 2003.

Anexo C. TECNOLOGIAS MICROSOFT™ PARA WINDOWS™

O desenvolvimento de uma aplicação requer sempre a utilização de várias ferramentas de desenvolvimento. Quando a plataforma escolhida é o Windows, existem várias opções, desde a utilização de pacotes *freeware* (e.g. Java) até aos sistemas profissionais Microsoft™. Tendo em conta a disponibilidade de todos os pacotes de desenvolvimento da Microsoft™ para estudantes, através da "*MSDN Academic Alliance*" foi decidido utilizar a plataforma .NET, programada em Visual C#. O IDE utilizado foi o "*Microsoft Visual Studio 2010*". Para a programação do interface do utilizador foi utilizada a infraestrutura WPF. Para a programação dos serviços web, foi utilizada a infraestrutura WCF. As subsecções seguintes apresentam uma breve descrição de cada uma destas tecnologias.

I. VISUAL STUDIO 2010

O "*Visual Studio*" é um conjunto de ferramentas baseadas em componentes de desenvolvimento de *software*, que permite criar aplicações de alto desempenho [1].

Este IDE disponibiliza todas as ferramentas de *software* necessárias para criar aplicações web ASP.NET, serviços web XML, aplicações para computadores pessoais e aplicações móveis. As linguagens de programação disponíveis, Visual Basic, Visual C# e Visual C++, partilham o mesmo IDE, o que permite a partilha de ferramentas e o desenvolvimento de soluções com várias linguagens de programação. O IDE suporta o desenvolvimento em múltiplos monitores, permitindo mover as várias janelas (Editor de código, ajuda online, vista de desenho) para fora do IDE.

O editor de código utiliza cores para realçar o código e utiliza a tecnologia *IntelliSense*™, para completar as instruções de forma automática, em função do contexto de utilização. Esta tecnologia, de uma forma adicional, serve como documentação e distinção dos nomes das variáveis, funções e métodos.

O editor utiliza a tecnologia "*Code Outlining*" para alterar a visibilidade do código, e facilitar a navegação em ficheiros de grandes dimensões. A janela do editor disponibiliza no topo, duas listas pendentes (*ComboBox*), que enumeram as classes existentes no ficheiro (à esquerda) e respetivos elementos (propriedades, métodos, funções, eventos etc.) à direita.

O *debugger* permite a paragem (*breakpoint*) de execução em qualquer parte do programa, numa ou mais tarefas, com possibilidade de análise de variáveis e memória, permitindo, em alguns casos, a sua alteração e continuação de execução, sem necessidade de recompilar o código fonte.

A árvore da solução permite agregar vários projetos, com linguagens de programação distintas, apresentando as propriedades e referências específicas de cada projeto, bem como o código associado, podendo este estar organizado da forma que o programador achar mais conveniente.

O sistema de pesquisa permite localizar qualquer fragmento frásico em qualquer parte da solução, gerando uma lista de todas as localizações encontradas.

II. INFRAESTRUTURA .NET

A infraestrutura .NET foi desenvolvida pela Microsoft™, e executa primariamente nos sistemas operativos Windows. É composta por uma vasta biblioteca e permite a partilha entre linguagens de desenvolvimento (uma linguagem pode utilizar código escrito noutra).

Os programas escritos para esta infraestrutura, são executados num ambiente de *software* (em contraste com os ambientes de hardware), denominado CLR ("*Common Language Runtime*"), que é uma máquina virtual que fornece serviços tais como segurança, gestão de memória, gestão de erros etc..

A biblioteca de base fornece as estruturas necessárias para o interface de utilizador, acesso a bases de dados, criptografia, desenvolvimento web, cálculo e comunicações. O código de uma aplicação, é o conjunto do código desenvolvido pelo programador com o código das bibliotecas .NET [2].

Outro elemento fundamental da infraestrutura .NET é o CLI (*Common Language Infrastructure*). Esta infraestrutura fornece uma plataforma neutra relativamente às linguagens de programação, para o desenvolvimento e execução das aplicações, incluindo a gestão de exceções, e gestão de memória (GC – *Garbage Collector*). As linguagens são compiladas para um código comum CIL (*Common Intermediate Language*) que é guardado com formato PE (*Portable Executable*), em DLL (*Dynamic Link Library*) e EXE (código executável no sistema operativo Windows).

O CLR libertou o programador dos problemas relativos à gestão de memória, alocando memória na instanciação dos objetos, e libertando-a quando deixar de haver referências a um dado bloco. O GC é um sistema de recolha e compactação de memória não determinístico. O

GC suspende a execução da aplicação quando é executado, para proceder à recuperação de memória. Este facto é tido como um dos maiores contras da infraestrutura .NET, já que introduz atrasos na execução do programa, que não estão sob o controlo do programador. Outro problema apontado ao .NET é o facto de ser possível obter todo o código escrito, em qualquer linguagem, a partir dos DLL e EXE. Esta questão é mais importante, quando estão em causa segredos das empresas e mecanismos de controlo das licenças de *software*. Com essa finalidade, foi introduzido o *Dotfuscator*, que altera os nomes de funções e variáveis, para tornar mais difícil a compreensão do código lido.

III. VISUAL C#

A linguagem de programação C# é uma linguagem estruturada, que requer a declaração do tipo de variável ("*Strong Type*"), declarativa, funcional, genérica e orientada ao objeto e ao componente. Embora tenha sido projetada pela Microsoft™, foi mais tarde definida como norma ECMA-334 e ISO/IEC 23270:2006.

A linguagem foi estruturada de forma a ser simples, moderna, de uso genérico e orientada ao objeto. Outros requisitos do seu projeto foram a sua portabilidade, suporte para internacionalização, requisitos reduzidos de memória e poder computacional [3]. De seguida, é apresentado o código em C# do exemplo canónico "*Hello World*".

```
using System;

class Hello
{
    static void Main()
    {
        Console.WriteLine("Hello world");
    }
}
```

É possível identificar neste trecho, os seguintes componentes:

- A diretiva "using" para referenciar um espaço de nome ("namespace"), neste caso o "System", que contém a classe "Console" referenciada no método "Main()". Desta forma não é necessária a referência completa da classe (System.Console);
- O método Main() é um membro da classe Hello;
- O ponto de entrada para uma aplicação C# é sempre o método estático Main();
- A saída "*Hello world*" é produzida por uma classe da biblioteca do CLI;

Contrariamente ao C/C++ temos:

- Não são utilizados os operadores "::" ou "->". Em C# é utilizado o ponto '.' Como separador de nomes compostos;
- A ordem de declaração é irrelevante;
- O programa não utiliza #include para importar código;
- O C# não tem funções ou variáveis globais. Todos os métodos ou variáveis têm de ser declaradas no âmbito de uma classe. Contudo os membros de uma classe estática têm um comportamento semelhante a variáveis ou funções globais;
- Suporte nativo para tipos booleanos (*bool*). Os restantes tipos pré-definidos são: object, string, sbyte, short, int, long, byte, ushort, uint, ulong, float, double, char e decimal;
- O C# define tipos de valor e de referência ("*value type*" e "*reference types*"). O programador define novos tipos de valor via enumerações e estruturas, e novos tipos de referências via classes, interfaces e delegados;
- Os *arrays* são do tipo referência, podem ter uma ou mais dimensões, e podem ser inicializados na sua declaração:

```
int[] a1 = new int[] {1, 2, 3};
int[,] a2 = new int[,] {{1, 2, 3}, {4, 5,
6}};
int[,,] a3 = new int[10, 20, 30];
int[][] j2 = new int[3][];
j2[0] = new int[] {1, 2, 3};
j2[1] = new int[] {1, 2, 3, 4, 5, 6};
j2[2] = new int[] {1, 2, 3, 4, 5, 6, 7, 8,
9};
```

- É possível executar métodos sobre qualquer valor, até sobre os valores primitivos como um int:

```
using System;
class Test
{
  static void Main()
  {
    Console.WriteLine(3.ToString());
  }
}
```

Neste exemplo, é possível a execução do método `ToString()` sobre o número '3', resultando na cadeia de carateres (*string*) "3".

- As variáveis têm de ser declaradas e antes de serem utilizadas, é obrigatória a sua inicialização. Os parâmetros são normalmente passados por valor. Para passar o parâmetro por referência, deve ser precedido por '*ref*' ou '*out*' na declaração do método:

```
void Swap(ref int a, out int b, params
int[] args) {…}
```

- O parâmetro do tipo '*out*' é semelhante a '*ref*', com a exceção de que o valor de entrada não é utilizado. Só pode existir um modificador '*params*', que tem de ser o último da declaração e ser um *array* de uma dimensão,

permite passar opcionalmente, um ou vários parâmetros adicionais ao método;

- Uma classe é do tipo referência, pode herdar de outra classe e pode implementar interfaces. Os seus membros podem incluir constantes, campos, métodos, propriedades, eventos, indexadores, operadores, construtores (de instância ou estáticos) e finalizadores;
- A acessibilidade dos membros pode ser pública (acesso ilimitado), protegido (acesso limitado à classe ou tipos derivados), interna (acesso limitado ao programa), interno protegido (acesso limitado ao programa ou a tipos derivados) e privado (acesso limitado ao tipo declarado);
- A propriedade é um membro que fornece acesso a uma característica de um objeto ou classe. A primeira parte da declaração é semelhante à declaração de um campo, e o restante, define os assessores ('*get*' e/ou '*set*'):

```
public class Button
{
private string caption;
  public string Caption {
    get {
      return caption;
    }
    set {
      caption = value;
      Repaint();
    }
  }
…
}
```

- Um evento é um membro que permite à classe fazer notificações. A classe define um evento utilizando a palavra-chave *event*, e o seu tipo deve ser um delegado;

```
public delegate void EventHandler (object
sender, EventArgs e);
public class Button
{
  public event EventHandler Click;
  public void Reset() {
   Click = null;
  }
}
```

- Um indexador é um membro que permite que um objeto seja indexado da mesma forma de um *array*;
- O construtor implementa as ações necessárias à inicialização da instância da classe, opcionalmente com vários parâmetros. Se não for declarado, é fornecido automaticamente um construtor sem parâmetros. Podem ser declarados vários construtores (distinguidos pelo número e tipo dos parâmetros):

```
class Point
{
  public double x, y;
  public Point(double x, double y) {
this.x = x; this.y = y; }
  public Point() { this.x = 0; this.y = 0;
}
  …
}
```

- O finalizador implementa as ações necessárias à finalização da classe, não podem ter parâmetros, modificadores de acesso e serem chamados diretamente. A sua execução é automática pelo GC;

As classes suportam a herança simples, no qual, a classe `object` é a classe base de todas as classes. Um método, propriedade ou indexador definido como virtual, permitem que a sua implementação seja sobreposta na classe derivada. Uma classe pode não ser definida na totalidade (quando a intenção é definir uma classe base), pela utilização do modificador '`abstract`'. Uma classe abstrata pode definir membros abstratos, que obrigatoriamente devem ser implementados nas classes derivadas. Uma classe que não deve ser instanciada, deve ser declarada como estática, e os seus membros devem ser todos estáticos. A declaração de uma classe pode ser dividida em dois ou mais ficheiros de código, se a sua declaração utilizar a palavra-chave '`partial`'.

- Uma estrutura é semelhante a uma classe (pode implementar interfaces e ter o mesmo tipo de membros de uma classe). Contudo, uma estrutura é do tipo valor (as classes são do tipo referência) e não podem ser herdadas.
- Um interface define um contrato. A classe ou estrutura que o implementa é obrigada a aderir a esse contrato. Um interface pode conter as declarações de métodos, propriedades, eventos e indexadores como membros;

```
interface IExample
{
  void   F(int value);
  string P { get; set; }
}
```

As interfaces suportam heranças múltiplas, e as classes/estruturas podem implementar múltiplas interfaces.

- Os delegados permitem cenários semelhantes aos que algumas linguagens resolveram via apontadores para funções. Contudo os delegados são orientados a objetos e de tipo seguros. O código seguinte mostra a declaração do delegado 'SimpleDelegate' (sem argumentos e sem retornar um resultado), e a sua utilização;

```
delegate void SimpleDelegate();
class Test
{
    static void F() {
    Console.WriteLine("Test.F");
  }
```

```
  void MultiCall(SimpleDelegate d, int
count) {
    for (int i = 0; i < count; i++) {
    d();
  }
}
```

É visível que o método '`MultiCall`' não sabe qual é o método que está a chamar, qual a sua acessibilidade ou se o método é ou não estático. A única informação conhecida é a sua estrutura declarativa, definida pelo delegado '`SimpleDelegate`'.

- A linguagem C# permite que classes, estruturas, interfaces e métodos sejam parametrizáveis pelo tipo de dados que guardam e manipulam. Esta característica é denominada "genéricos". A utilização de genéricos elimina a utilização de casts. O código seguinte mostra o mesmo código sem utilização de genéricos:

```
public class Stack
{
  private object[] items = new ob-
ject[100];
  public void Push(object data) {…}
  public object Pop() {…}
}
// Implementação da classe com um tipo
Customer
Stack s = new Stack();
s.Push(new Customer());
Customer c = (Customer)s.Pop();
// Implementação da classe com um tipo
inteiro
Stack s = new Stack();
s.Push(3);
int i = (int)s.Pop();
```

O trecho seguinte de código apresenta a mesma implementação, com recurso à utilização de genéricos:

```
public class Stack<ItemType>
{
  private ItemType [] items = new ItemType
[100];
  public void Push(ItemType data) {…}
  public ItemType Pop() {…}
}
// Implementação da classe com um tipo
Customer
Stack< Customer> s = new Stack();
s.Push(new Customer());
Customer c = s.Pop();
// Implementação da classe com um tipo
inteiro
Stack<int> s = new Stack();
s.Push(3);
int i = s.Pop();
```

- Em C#, um método *callback* é normalmente invocado indiretamente via delegados. Tal implementação pode ser efetuada com recurso a métodos anónimos:

```
delegate bool Action(Node n);
static void Walk(Node n, Action a)
{
  while (n != null && a(n)) n = n.Next;
}
// Implementação com métodos anónimos:
  …
  Walk(list, delegate(node n)
    {
      Console.WriteLine(n.name);
      Return true;
    }
  }
  …
```

IV. *WINDOWS COMMUNICATION FOUNDATION* (WCF)

Existe um largo conjunto de tecnologias para a construção de aplicações distribuídas, entre elas, .NET *Remoting*, MSMQ (*Message Queuing*), e COM+/ Serviços Empresariais. O WCF (*Windows Communication Foundation*) reúne todas as tecnologias numa infraestrutura dedicada à construção e consumo de serviços. Esta infraestrutura foi introduzida pelo .NET 3.0 [4].

A utilização da infraestrutura WCF oferece vantagens únicas ao programador:

- Suporte para o envio de mensagens não só em HTTP, mas também em vários protocolos de rede (e.g. FTP) e capacidade de trocar de protocolo de uma forma simples
- Capacidade de fornecer serviços na aplicação, em vez de utilizar um servidor web com suporte interno para as normas web mais recentes (SOAP 1.2 e WS-*) e facilidade para suportar novas normas que possam vir a existir;
- Suporte para transações, protocolos seguros e fiabilidade
- Suporte para o envio de mensagens com formatos distintos de SOAP, e.g. REST (*Representational State Transfer*);

O *Visual Studio*™ permite criar as estruturas base para um projeto WCF, que resulta em dois ficheiros de código, contendo respetivamente a interface de implementação e a implementação do serviço desejado. O código seguinte mostra o interface e a implementação do serviço genérico `Service1`:

```
[ServiceContract]
public interface IService1 {
  [OperationContract]
  string GetData(int value);
  [OperationContract]
```

```
  CompositeType GetDataUs-
ingDC(CompositeType composite);
 … // Add service Operations
}
[DataContract]
public class CompositeType {
  bool boolValue = true;
  string stringValue = "Hello ";

  [DataMember]
  public bool BoolValue {
    get { return boolValue; } set {
boolValue = value; }
  }

  [DataMember]
  public string StringValue {
    get { return stringValue; } set {
stringValue = value; }
  }
}
```

O servidor local típico é implementado com o código seguinte:

```
using System.ServiceModel;
using System.ServiceModel.Description;

class WCFServiceHost
{
  static void Main()
  {
    ServiceHost typeHost = null;
    try
    {
      typeHost = new Service-
Host(typeof(CompositeType));
      ServiceEndpoint productEndpoint =
         produc-
tHost.AddServiceEndpoint(typeof(IService1)
,
         new NetTcpBinding(),

"net.tcp://localhost:9010/Service1 ");
         typeHost.Faulted += new
EventHandler(typeHost_Faulted);
      typeHost.Open();
      Console.WriteLine(
        "O serviço está à escuta no
endereço:");
        Console.WriteLine("{0} ({1})",
        productEnd-
point.Address.ToString(),
        productEndpoint.Binding.Name);
      Console.WriteLine("\nPrimir para
parar o serviço.");
      Console.ReadKey();
    }
    finally
    {
      if (typeHost.State == Communication-
State.Faulted)
        typeHost.Abort();
      else
        typeHost.Close();
    }
  }
  static void typeHost_Faulted(object
sender, EventArgs e)
  {
    Console.WriteLine("O serviço Service1
falhou.");
  }
}
```

V. *WINDOWS PRESENTATION FOUNDATION* (WPF)

O WPF é a nova infraestrutura da Microsoft™ para a construção de interfaces gráficas nas aplicações clientes com visuais apelativos. Com o WPF é possível, além da construção de formulários semelhantes aos criados com o "*Windows Forms*", construir conteúdos interativos, ricos em elementos multimédia, animações e documentos tradicionais [5]. O WPF não se resume só a apresentação gráfica, unifica todas as formas comuns de representação, UI (*User Interface*), média, gráficos vetorizados e documentos, numa única infraestrutura. O resultado é a capacidade para misturar controlos UI, conteúdos (documentos e média) e gráficos vetorizados de uma forma única.

Simultaneamente, o WPF suaviza o processo de desenvolvimento, ao reduzir a quantidade de código procedimental na especificação da aplicação, permitindo a separação entre o *design* gráfico do interface (escrito numa linguagem declarativa denominada XAML e o código de ação (escrito numa qualquer linguagem .NET). Tendo em conta que o WPF está montado em cima do motor gráfico Direct3D, a elevada performance obtida é função do hardware gráfico do computador, retirando do CPU o respetivo processamento. A descrição mais detalhada da infraestrutura WPF sai do âmbito deste documento, dada a sua ampla área de ação. Fica no entanto, um exemplo simples, de um formulário com um botão, como forma representativa da ligação entre o código XAML e o código de execução (C#).

```
--- Código XAML ----------------------
<Window x:Class="WpfTeste.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/
2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winf
x/2006/xaml"
  Title="MainWindow" Height="211"
Width="415">
```

```
  <Grid>
    <Button Name="button1" Content="Olá
WPF" Height="30" Width="120"
Click="button1_Click" />
  </Grid>
</Window>
--- Código C# ------------------------
using System.Windows;
namespace WpfTeste
{
  public partial class MainWindow : Window
  {
    public MainWindow()
    {
      InitializeComponent();
    }
    private void button1_Click(object
sender, RoutedEventArgs e)
    {
      MessageBox.Show("Olá mundo!", "WPF
App...");
    }
  }
}
```

Este código apresenta um formulário com um botão com o texto "Olá WPF", que quando premido, Apresenta uma caixa de diálogo com o título "WPF App…" que diz "Olá mundo!". Algumas características distintivas do WPF:

- Separação do código de apresentação (XAML) do de execução (C#, VB.NET etc.);
- Estrutura de implementação dos controlos em árvore;
- Capacidade de alterar o aspeto visual por defeito dos controlos, sem perder as características funcionais;
- Possibilidade de incorporar uns controlos nos outros, de uma forma praticamente ilimitada;
- Os controlos têm por norma, coordenadas relativas;
- Suporte para especificação declarativa, do mapeamento entre representação visual dos dados e a sua origem, via "*Data Binding*";
- Opção de conversão direta de uma aplicação de computador numa aplicação web (com execução no navegador de internet;

Referencias

[1] . Visual Studio 2010, "*Quick Tour of the Integrated Development Environment*", Microsoft, MSDN, http://msdn.microsoft.com/en-us/library/ee336123.aspx, verificado em Outubro de 2012

[2] NET Framework, "Getting Started with .NET Framework", Microsoft, MSDN, http://msdn.microsoft.com/en-us/library/vstudio/hh425099.aspx, verificado em Outubro de 2012

[3] ECMA, "*C# Language Specification*", ECMA-334, 4ª Edição, Junho de 2006

[4] Green,Robert; "*Introduction to Windows Communication Foundation*", Microsoft, MSDN - http://msdn.microsoft.com/en-us/library/dd936243.aspx, verificado em Outubro de 2012

[5] Sklar, David F.; "*An Introduction to Windows Presentation Foundation*", Microsoft, MSDN - http://msdn.microsoft.com/en-us/library/aa480192.aspx, acedido em Outubro de 2012.

Anexo D. SERVIÇOS WEB

***Resumo*— Os serviços web são o veículo de informação para a ligação do aparato ao mundo exterior. Este anexo apresenta um breve resumo do seu funcionamento, com referência às tecnologias web utilizadas.**

I. SERVIÇOS WEB

O ponto de entrada de controlo em modo automático é a internet via serviços de web (WS – "*Web Services*"). As subsecções seguintes apresentam a teoria de funcionamento dos WS e a sua implementação.

O consórcio W3C ("*World Wide Web Consortium*") define um serviço web como "Um sistema de *software* desenvolvido para o suporte interoperável de interações máquina-máquina através de uma rede". Têm uma interface descrita num formato capaz de ser processado pela máquina, especificamente a linguagem descritiva para serviços web (WSDL – *Web Services Description Language*). Os sistemas interagem com o serviço web, da forma indicada na sua descrição, utilizando mensagens SOAP (*Simple Object Access Protocol*), normalmente transmitidas via HTTP (*Hyper Text transfer Protocol*), com uma serialização em XML (*Extensible markup Language*), juntamente com outros *standards* web [1].

Um serviço web destina-se a disponibilizar uma funcionalidade em nome do seu proprietário, individual ou empresarial. A entidade fornecedora é a pessoa ou organização que fornece o agente apropriado à implementação de um serviço específico. A entidade requerente é a pessoa ou organização que deseja utilizar o serviço web disponibilizado. Para que a troca de informação seja possível, o fornecedor e o requerente têm de acordar a semântica e mecânica de troca de mensagens.

A mecânica de troca de mensagens está documentada na descrição do serviço web (WSD), que é uma especificação, capaz de ser lida por uma máquina, da interface do serviço, escrita em WSDL. Esta especificação define o formato das mensagens, os tipos de dados, os protocolos de transporte e os formatos de serialização para transporte que devem ser usados entre o agente requerente e o agente fornecedor.

Simultaneamente especifica uma ou mais localizações da rede onde o agente fornecedor disponibiliza informação relativa à troca de mensagens. Essencialmente, a descrição do serviço representa um acordo que rege a mecânica da interação com esse serviço.

Existem várias formas de o requerente contratar e utilizar um serviço web. De uma forma geral, são necessários os seguintes passos, para negociar um serviço web, como ilustrado na Figura 1:

- As entidades, requerente e fornecedor, devem-se conhecer mutuamente;
- Estas entidades devem acordar de alguma forma a descrição do serviço e a semântica que irá governar a interação entre elas;
- A descrição do serviço e semântica é realizada por ambas as entidades;
- As entidades trocam mensagens com as quais executam uma tarefa;

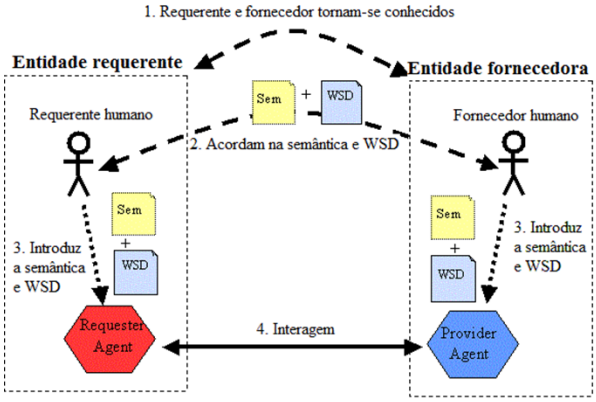


Figura 1 **Processo geral de negociação de um serviço web**

O XML soluciona um requisito tecnológico comum, ao fornecer uma norma de formato de dados flexível e extensível, que reduz significativamente o trabalho de implementação das várias tecnologias necessárias para garantir o sucesso dos serviços web. Os aspetos mais importantes do XML são a sua sintaxe, o esquema XML (XML *Schema*) e XML *Namespaces*.

II. SOAP

O SOAP oferece uma forma *standard* extensível para o encapsulamento e troca de mensagens. No contexto dos serviços web, o SOAP 1.2 fornece um mecanismo conveniente para as necessidades de referenciação (tipicamente pela utilização de cabeçalhos). As mensagens SOAP podem ser transportadas por vários protocolos de rede, nomeadamente HTTP, SMTP (*Simple Mail Transport Protocol*) e FTP (*File Transfer Protocol*). A versão SOAP 1.2 define um conjunto de regras de codificação para definir instâncias de tipos de dados definidos pela aplicação e uma convenção para a representação de procedimentos remotos RPC (*Remote Procedure Calls*).

III. WSDL

O WSDL é a linguagem para descrever os serviços web. Começa por descrever as mensagens trocadas entre requerente e fornecedor, de uma forma abstrata, e posteriormente, ligadas a um protocolo de rede e formato de mensagem específico.

As definições dos serviços web podem ser mapeadas para qualquer linguagem de implementação, plataforma, modelo de objeto ou sistema de mensagens. Desde que aja acordo na descrição do serviço entre o fornecedor e o requerente, a sua implementação para lá dos serviços web, pode ser qualquer coisa.

Quando uma entidade requerente deseja interagir com uma entidade fornecedora sem saber qual o agente a utilizar, pode ser necessário descobrir um candidato capaz. A descoberta é "o ato de localizar uma descrição de um serviço web, que pode ter sido desconhecida anteriormente, e que se adequa a um determinado critério funcional". O objetivo é localizar um serviço web apropriado. Para um programa de um computador interagir com outro é necessário o seguinte:

- Deve existir uma ligação física entre eles para permitir a troca de informação;
- Deve existir um acordo nas estruturas dos dados a trocar (texto, estruturas XML, etc.);
- Os programas devem estar de acordo relativamente ao significado da informação;
- Deve existir acordo relativamente ao processamento das mensagens trocadas;

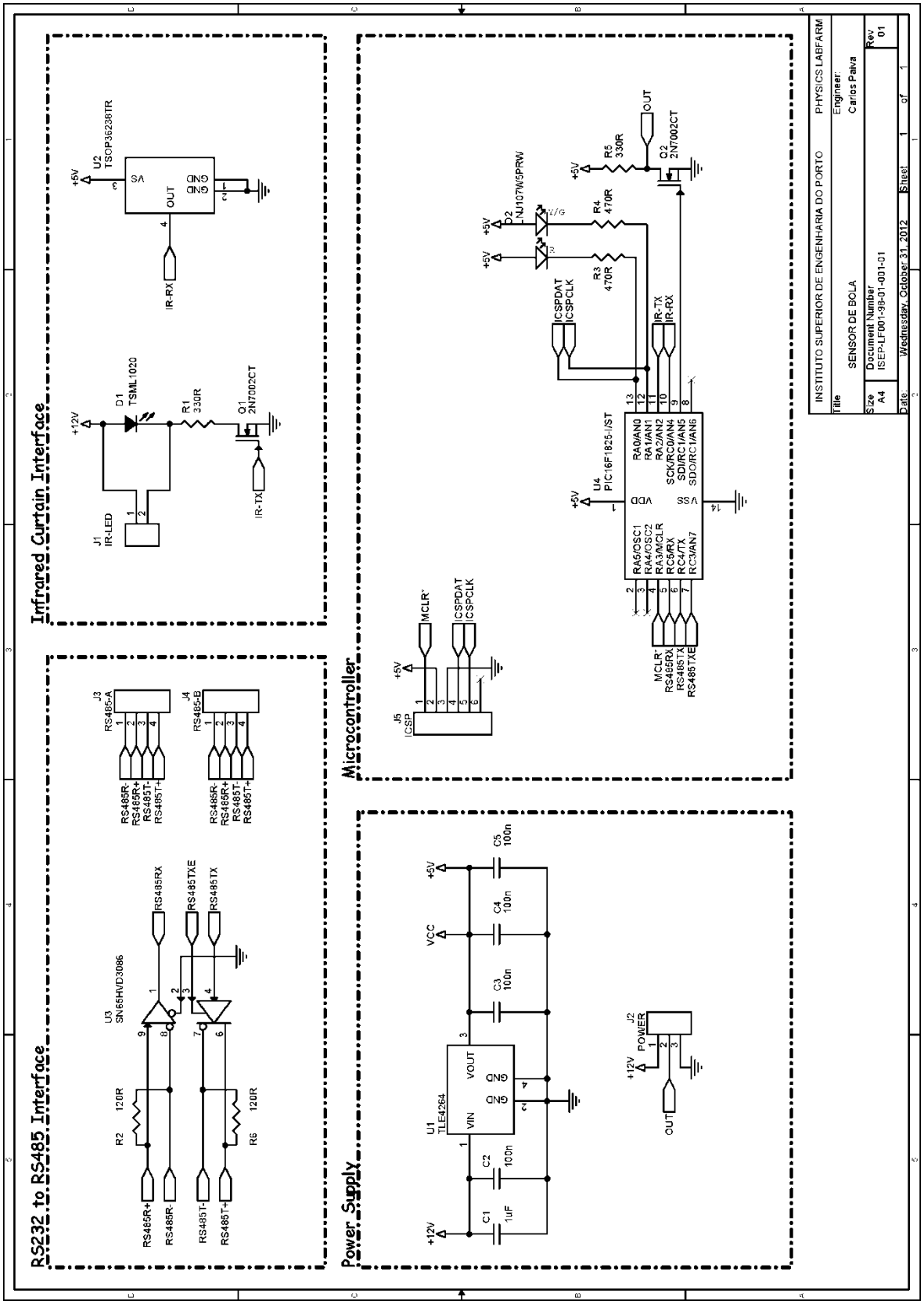
O facto de focalizar na mensagem, em vez da ação causada pela mensagem, significa que as arquiteturas SOA (*Service Oriented Architecture*) têm uma boa visibilidade (as partes interessadas podem inspecionar o fluxo de mensagens e certificar-se quanto aos serviços invocados e regras associadas), o que que significa, que os intermediários, e.g. *firewalls*, podem executar melhor a sua função, ao olhar para a estrutura da mensagem e tomar decisões previsíveis sobre segurança [2].

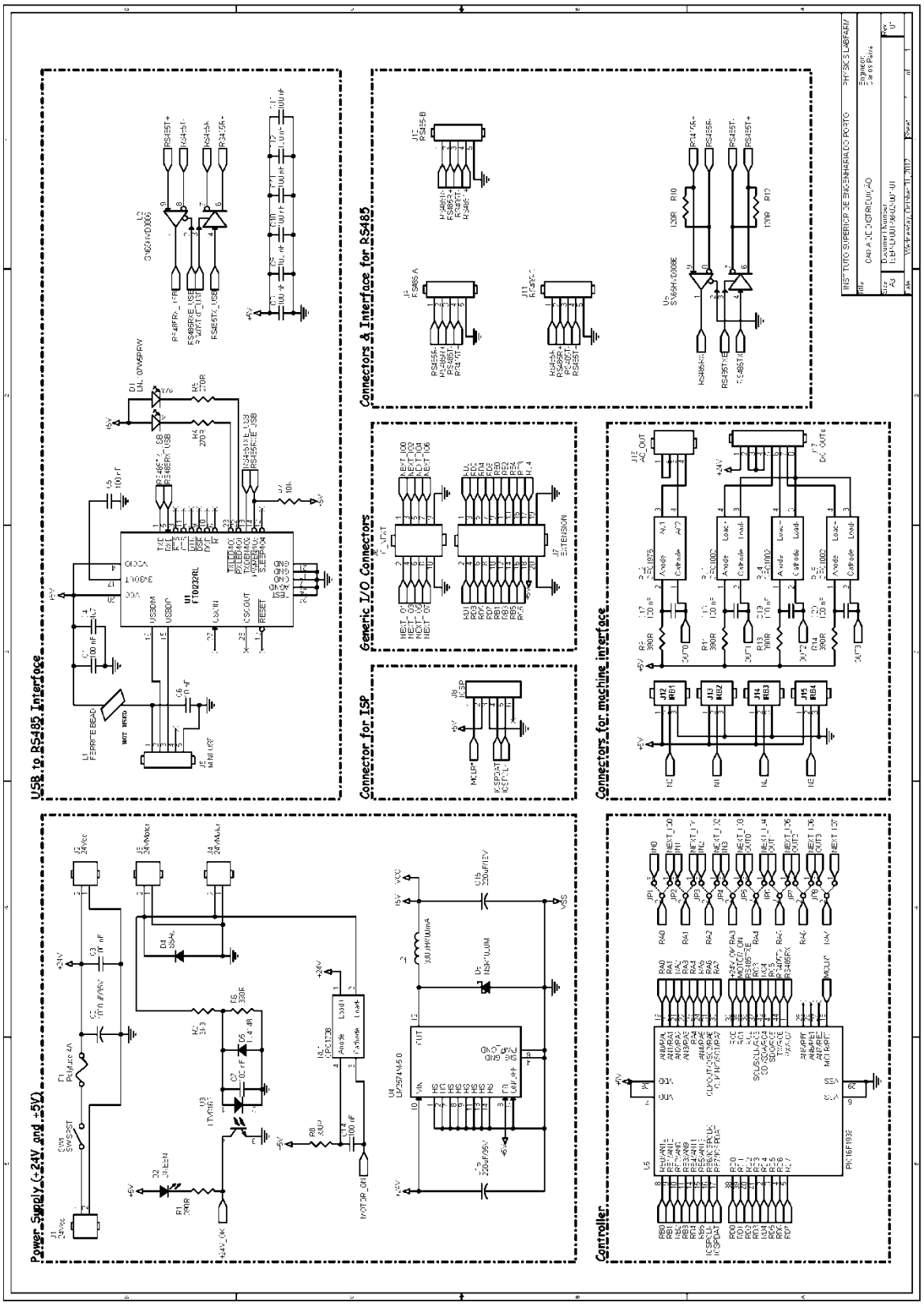
Referencias

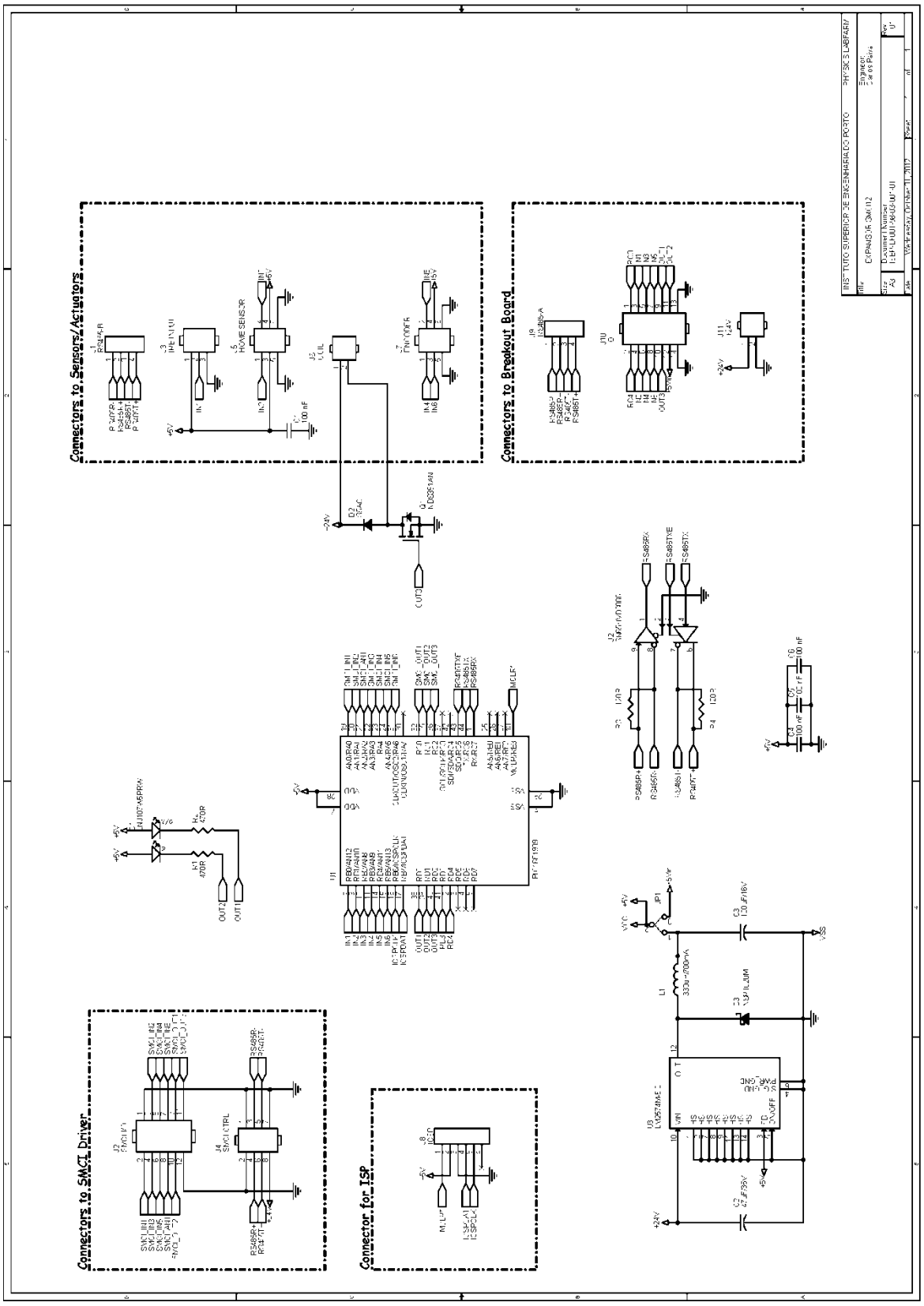
[1] Haas, Hugo; Allen Brown, "*Web Services Glossary*", W3C, W3C Working Group Note, Revisão 11 de Fevereiro de 2004, http://www.w3.org/TR/ws-gloss/, acedido em Outubro de 2012.

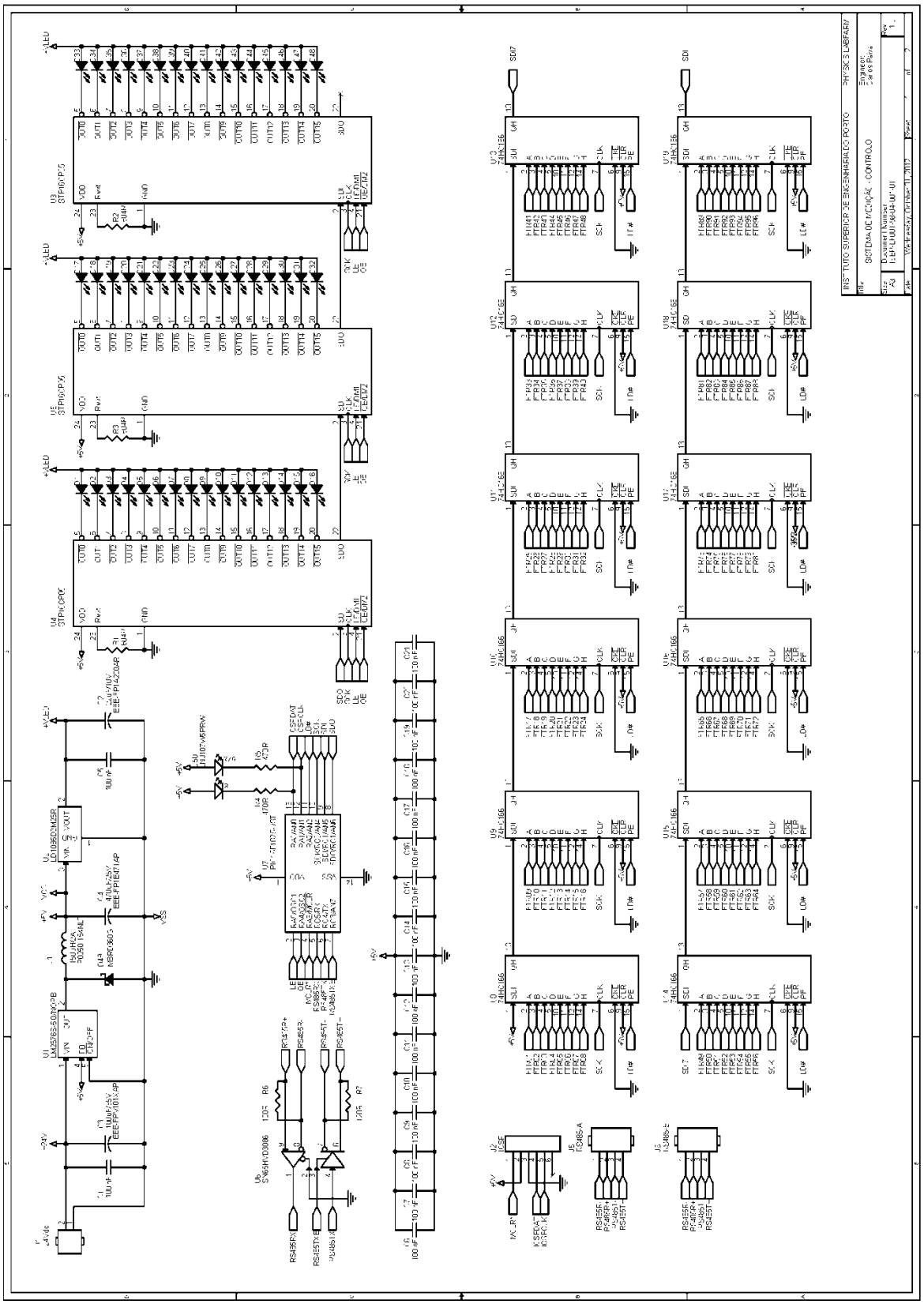
[2] Booth, David; Haas, Hugo; McCabe, Francis; "*Web Services Architecture*", W3C, W3C Working Group Note, Revisão 11 de Fevereiro de 2004, http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#wsdisc, acedido em Outubro de 2012.

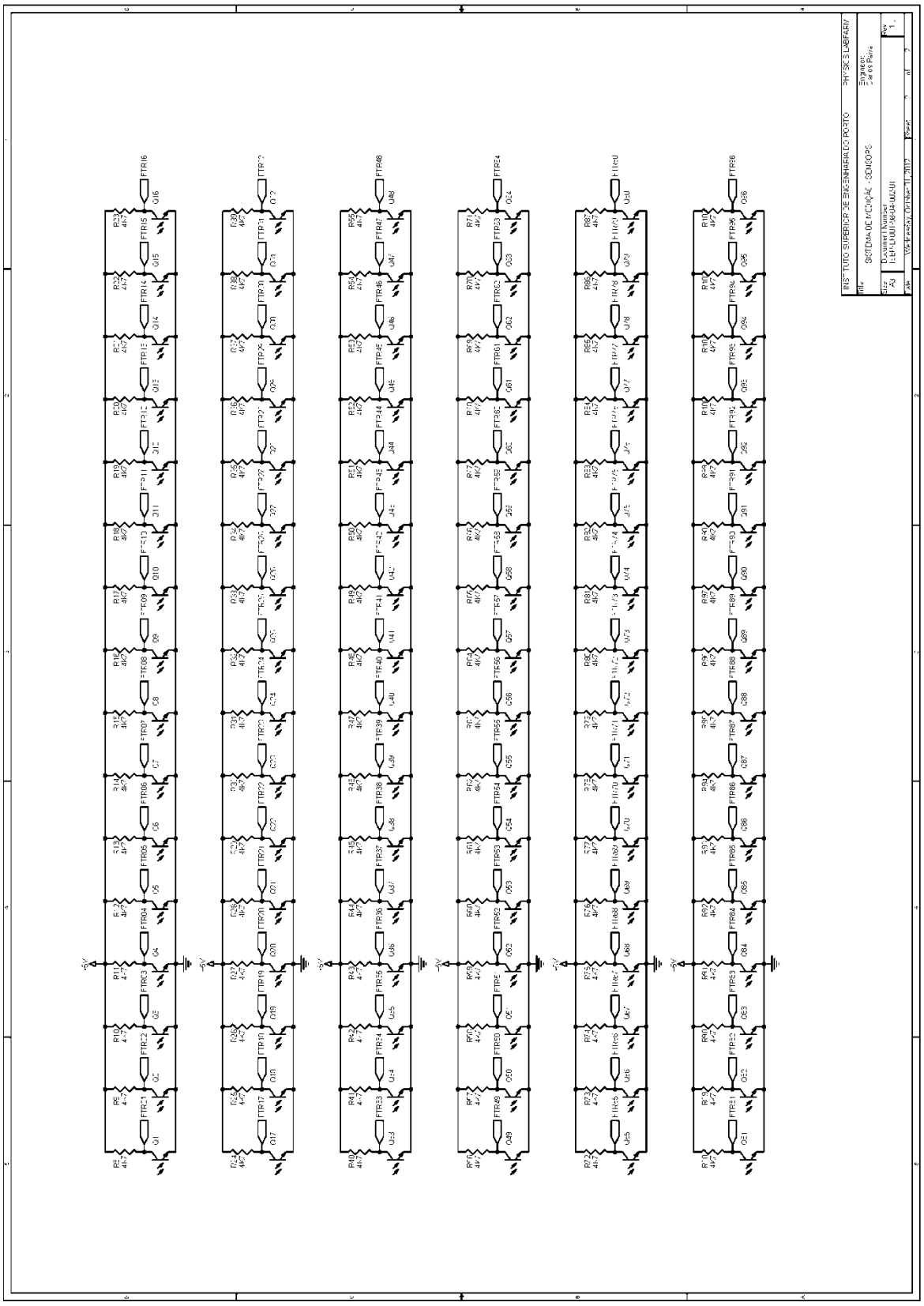
Anexo E. ESQUEMAS

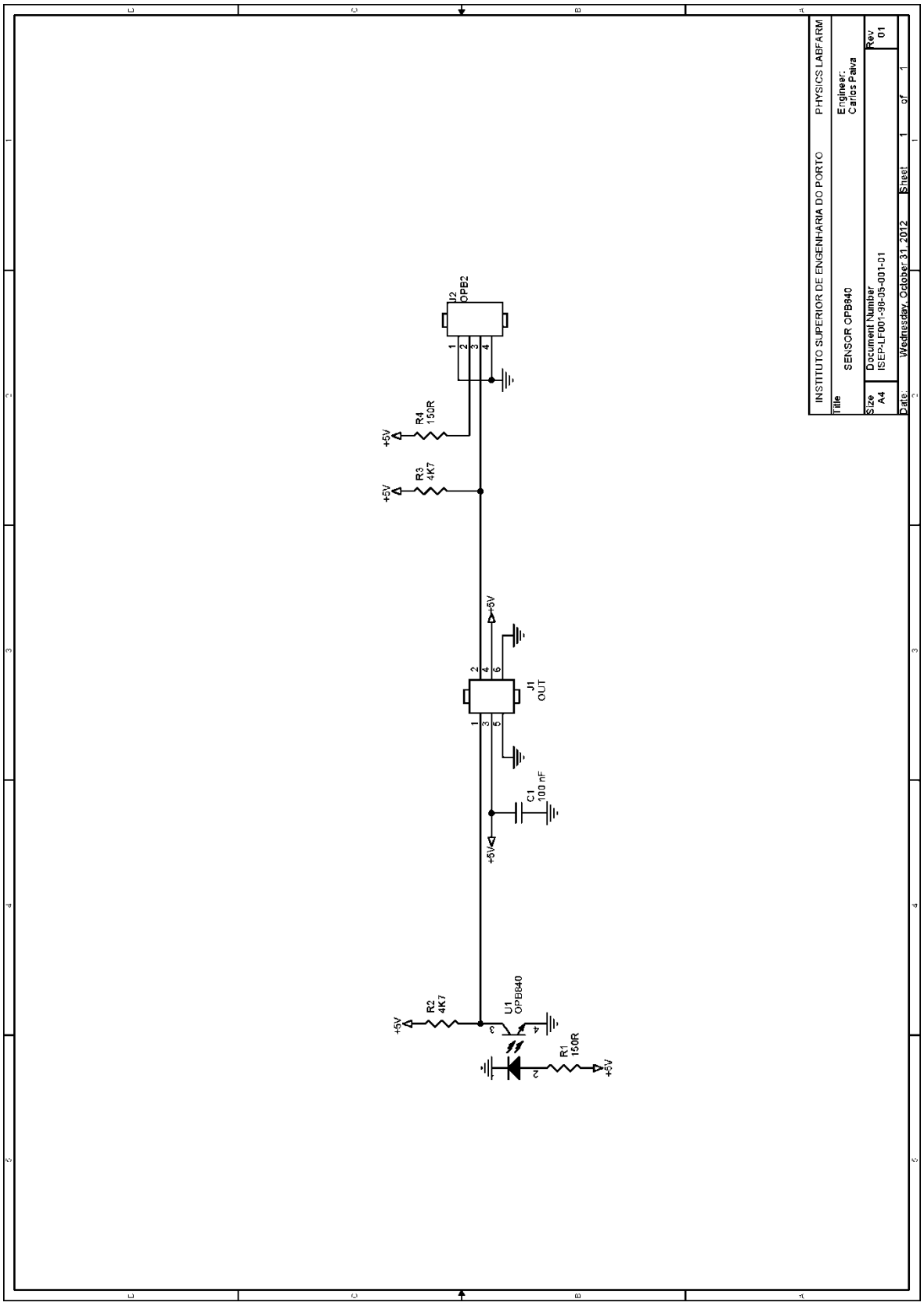


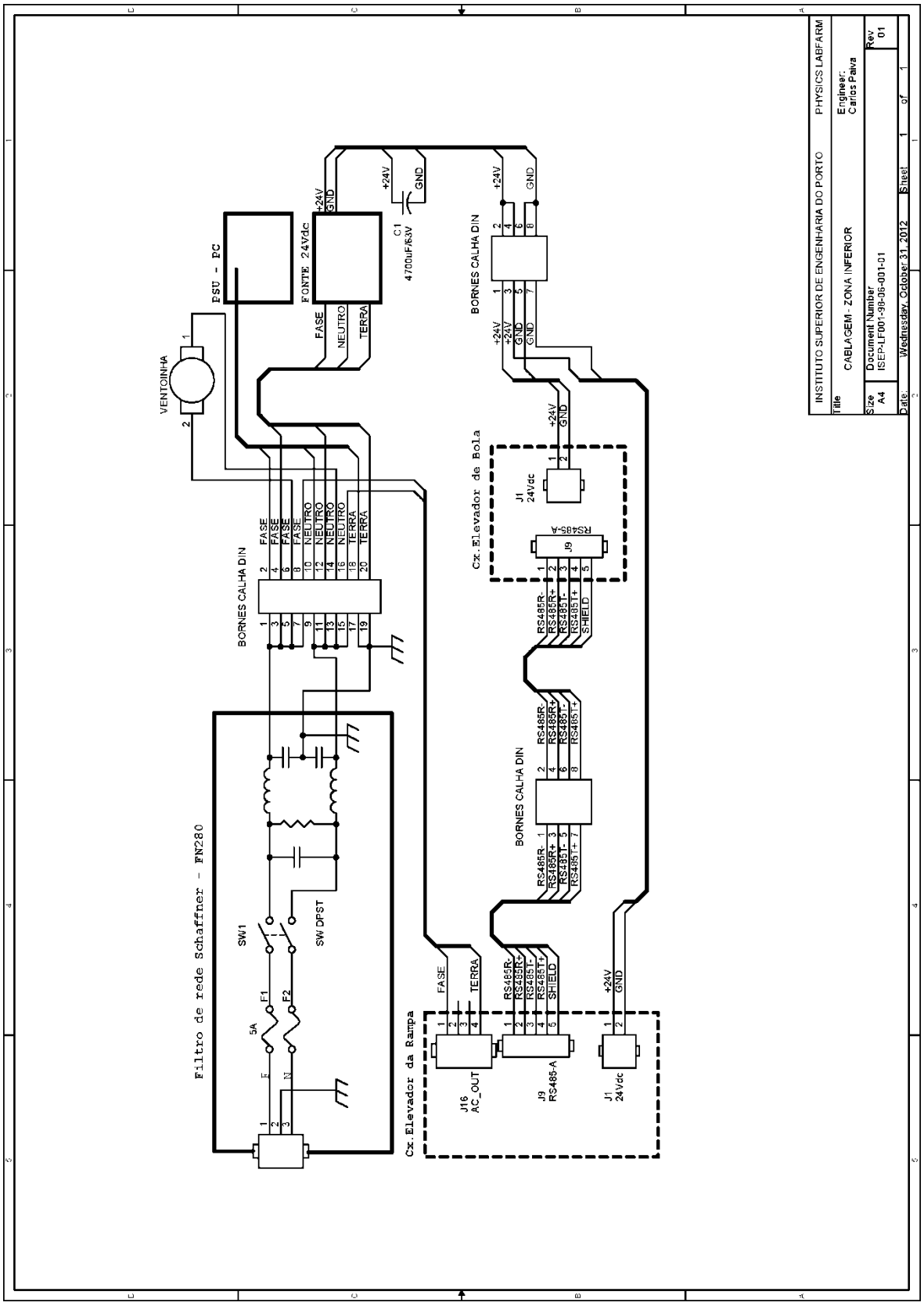

USB to RS485 Interface
Connectors A Interface for RS485
Generic I/O Connectors
Connector for ISP
Connectors for machine interface
Power Supply (-24V and +5V)
Controller

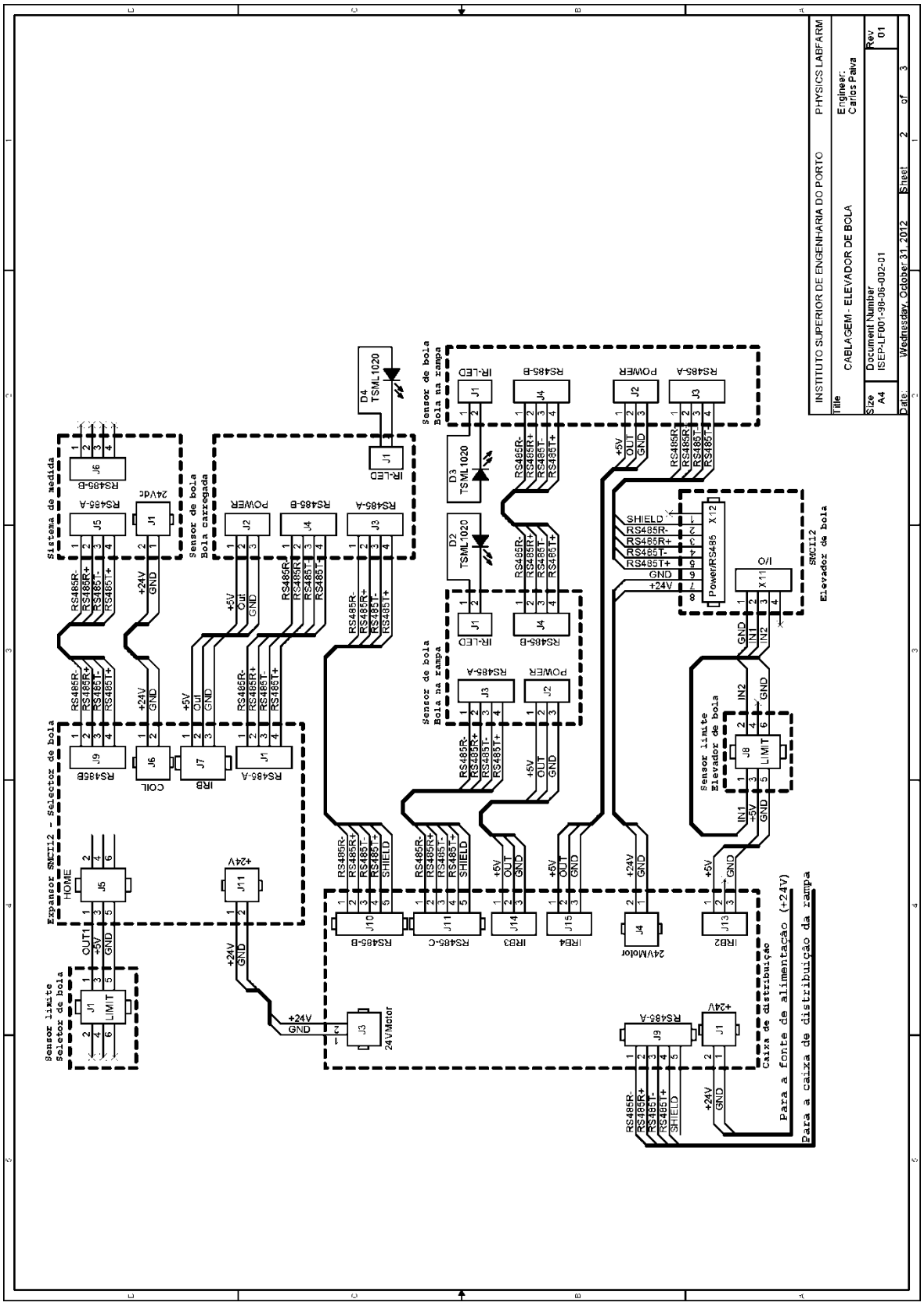

Connectors to Sensors/Actuators
Connectors to Breakout Board
Connectors to SMCI Driver
Connector for ISP

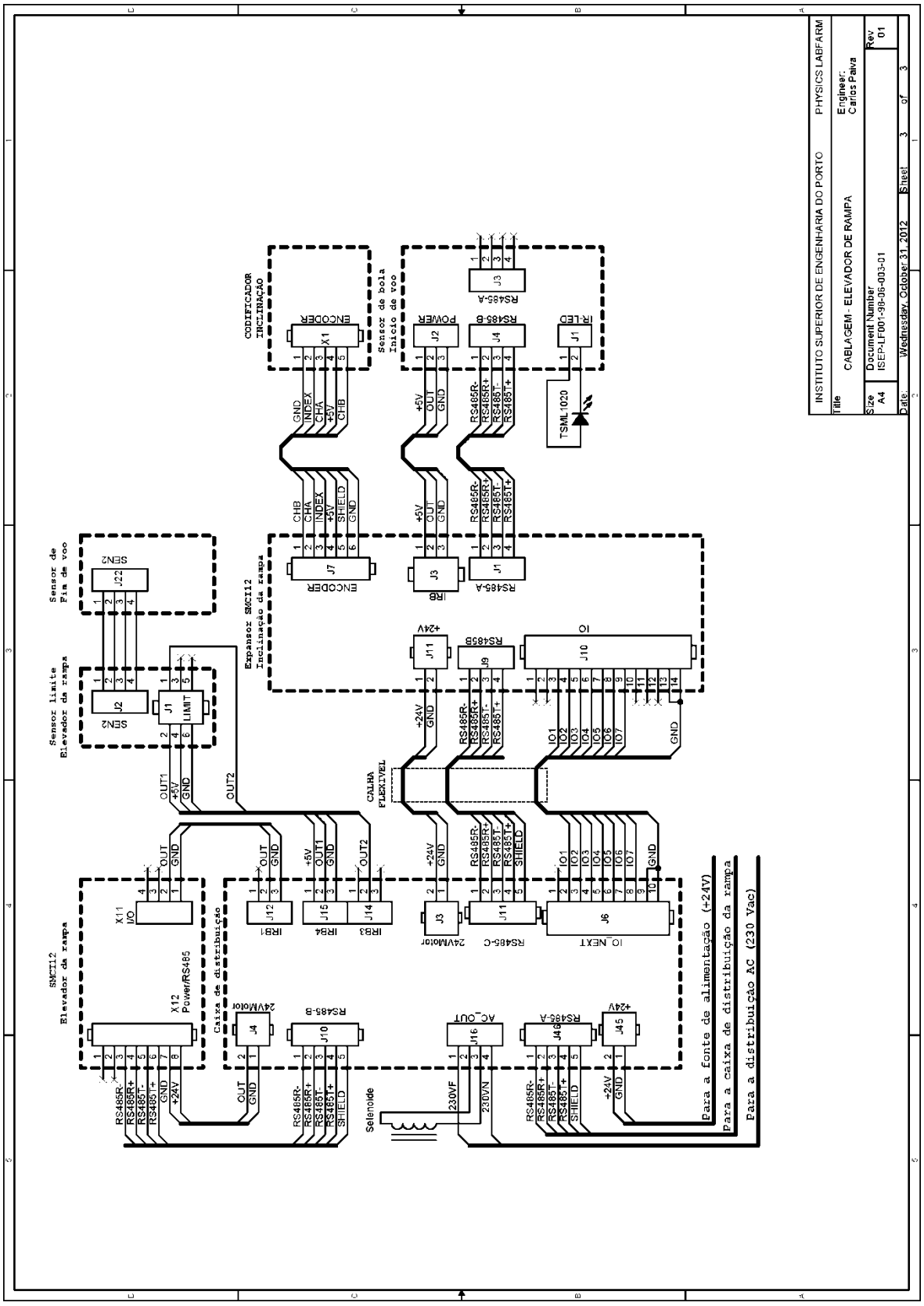





INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
PHYSICS LABFARM
Title
SENSOR OPB840
Engineer:
Carlos Paiva
Size
A4
Document Number
ISEP-LF001-98-05-001-01
Rev
01
Date:
Wednesday, October 31, 2012
Sheet
1
of
1
J2
OPB2
R4
150R
R3
4K7
+5V
J1
OUT
C1
100 nF
R2
4K7
U1
OPB840
R1
150R


INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
PHYSICS LABFARM
Title
CABLAGEM - ZONA INFERIOR
Engineer:
Carlos Paiva
Size
A4
Document Number
ISEP-LF001-98-06-001-01
Rev
01
Date:
Wednesday, October 31, 2012
Sheet
1
of
1
PSU - PC
FONTE 24Vdc
+24V
GND
C1
4700uF/63V
FASE
NEUTRO
TERRA
VENTOINHA
BORNES CALHA DIN
Cx.Elevador de Bola
J1
24Vdc
RS485-A
J9
RS485R-
RS485R+
RS485T-
RS485T+
SHIELD
Filtro de rede Schaffner - FN280
SW1
SW DPST
F1
F2
5A
F
N
Cx.Elevador da Rampa
J16
AC_OUT
J9
RS485-A
J1
24Vdc


INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
PHYSICS LABFARM
Title
CABLAGEM - ELEVADOR DE BOLA
Engineer:
Carlos Paiva
Size
A4
Document Number
ISEP-LEF001-98-06-002-01
Rev
01
Date:
Wednesday, October 31, 2012
Sheet
2
of
3
Sistema de medida
Sensor de bola
Bola carregada
Sensor de bola
Bola na rampa
Sensor de bola
Bola na rampa
Expansor SMCI12 - Selector de bola
Sensor limite
Selector de bola
Sensor limite
Elevador de bola
SMCI12
Elevador de bola
Caixa de distribuição
Para a fonte de alimentação (+24V)
Para a caixa de distribuição da rampa
TSML1020
24V/dc
24V/Motor
PowerRS485
HOME
LIMIT
COIL
IRB
IRB2
IRB3
IRB4


INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
PHYSICS LABFARM
Title
CABLAGEM - ELEVADOR DE RAMPA
Engineer:
Carlos Paiva
Size
A4
Document Number
ISEP-LF001-98-06-003-01
Rev
01
Date:
Wednesday, October 31, 2012
Sheet
3
of
3
CODIFICADOR INCLINAÇÃO
Sensor de bola Início de voo
Sensor de Fim de voo
Sensor limite Elevador da rampa
Expansor SMCI12 Inclinação da rampa
CAIXA FLEXIVEL
SMCI12 Elevador da rampa
Caixa de distribuição
Selenoide
Para a fonte de alimentação (+24V)
Para a caixa de distribuição da rampa
Para a distribuição AC (230 Vac)

Anexo F. ESTRUTURA DE IMPLEMENTAÇÃO – APLICAÇÃO WINDOWS

Classe `Labfarm.App`

```
<Application x:Class="Labfarm.App"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Startup="AppStartup">
    <Application.Resources>
     …
    </Application.Resources>
</Application>

namespace Labfarm
{
  public partial class App : Application
  {
    UI mainWindow = null;
    private void AppStartup(object sender, StartupEventArgs e) { … }
    void App_DispatcherUnhandledException(object sender,
                        DispatcherUnhandledExceptionEventArgs e) { … }
    static void MyHandler(object sender, UnhandledExceptionEventArgs args) { … }
  }
}
```

Interface Labfarm.WebService.ILabfarmWS

```
namespace Labfarm.WebService
{
  [ServiceContract]
  public interface ILabfarmWS
  {
    [OperationContract] ExecuteResult submitUserData(UserVariableObject data);
    [OperationContract] SystemVariableObject getMeasuredResults();
  }
}
```

Classe Labfarm.UI

```
<Window x:Class="Labfarm.UI"
  <Window.Resources> … </Window.Resources>
  <Border Style="{StaticResource outterBorder}" Height="420" Width="695">
    <Grid Name="root" Margin="10,0">
      <Grid.ColumnDefinitions> … </Grid.ColumnDefinitions>
      <Grid.RowDefinitions> … </Grid.RowDefinitions>
      <TextBlock Style="{StaticResource title}" … />
      <StackPanel … >
        <Button x:Name="btnConnect" Content="Connect" … />
        <Button Click="Form_Close" Content="X" … />
      </StackPanel>
      <TextBlock Text="Serial Port" … />
      <TextBox Text="COM1" … />
      <TextBlock Text="Dados recebidos … />
      <TextBox Name="CommInput" … />
      <TextBox Name="CommOutput" … />
      <TextBlock Text="Comandos enviados" … />
      <Button Name="BtnLoadBall" … >Ciclo Elevador</Button>
      <Button Name="BtnInit" … >Inicialização</Button>
      <Button Name="BtnBallInit" Content="Init Seletor de bolas" … />
      <Button Name="BtnElevatorInit" Content="Init Elevador" … />
      <Button Name="BtnBall1" Content="Bola 1" … />
      <Button Name="BtnBall2" Content="Bola 2" … />
      <Button Name="BtnBall3" Content="Bola 3" … />
      <Button Name="BtnStress" Content="Stress" … />
      <Button Name="btnMove" Content="RampMove" … />
      <Button Name="btnInitRampZ" Content="Init RampZ" … />
      <Button Name="btnClrLog" Content="Clear Log" … />
      <TextBox Name="txtPosition" … />
      <Button Name="btnTheta" Content="ThetaMove" … />
      <Button Name="btnPowerOn" Content="Power ON" … />
      <Button Name="btnPowerOff" Content="Power OFF" … />
      <TextBlock Text="Altura" … />
      <TextBlock Text="Angulo" … />
      <TextBlock Text="Bola … />
      <TextBox Name="txtHeight … />
```

```
      <TextBox Name="txtAngle" … />
      <TextBox Name="txtBall" … />
      <Button Name="btnExecute … </Button>
      <Button Content="Lock Ball" … />
      <Button Content="Release Ball" … />
      <Button Content="Flight Time" … />
      <Button Content="Range" … />
    </Grid>
  </Border>
</Window>


namespace Labfarm
{
  public partial class UI : Window
  {
    public void StartServer() { … }
    public void StopServer() { … }
    private void PrintEndpoints(ServiceHost host) { … }
    public UI() { … }
    private void Initialize() { … }
    void UI_Loaded(object sender, RoutedEventArgs e) { … }
    void btnExecute_Click(object sender, RoutedEventArgs e) { … }
    private void Cleanup(){ … }
    void UI_Closed(object sender, EventArgs e) { … }
    private void Form_Close(object sender, RoutedEventArgs e) { … }
    private void Connect(object sender, RoutedEventArgs e) { … }
  }
}
```

Classe Labfarm.WebService

```
namespace Labfarm.WebService
{
  public class LabfarmWS : ILabfarmWS
  {
    public ExecuteResult submitUserData(UserVariableObject data) { … }
    public SystemVariableObject getMeasuredResults() { … }
  }
}
```

Classe Labfarm.RemoteLab

```
namespace Labfarm
{
  public sealed class RemoteLab : ILabfarmWS
  {
    public string PortName { get; set; }
    public Experiment01 Experiment;
    public bool Initialized { get; private set; }
    private RemoteLab(){ … }
    public static RemoteLab Instance { … }
    public ExecuteResult submitUserData(UserVariableObject data) { … }
    public SystemVariableObject getMeasuredResults(){ … }
    public void InitializeLab(){ … }
  }
}
```

Classe Labfarm.Experiment01

```
namespace Labfarm
{
  public class Experiment01 : NanotecDataContext
  {
    public TextWriterTraceListener TraceWriter;
    public bool HeaderWrited = false;
    private bool MachineInUse { get; set; }

    private bool InError { … }
    private bool Initialized { get; set; }
    public Experiment01(){ … }
    public Experiment01(string portName) : this(){ … }
    public void Connect(string portName) { … }
    public void LoadBall() { … }
    public void SelectBall1(){ … }
    public void SelectBall2(){ … }
```

```
  public void SelectBall3() { … }
  public void BallSelectorHome() { … }
  public void BallSelectorTray(int trayNr) { … }
  public void ElevatorHome() { … }
  public void ElevatorLoadBall() { … }
  public void ElevatorMoveUp() { … }
  public void ElevatorEject() { … }
  public void StressElevator() { … }
  public void RampHome() { … }
  public void RampZMove(float Height) { … }
  public void RampAbsMove(){ … }
  public void RampThetaHome(){ … }
  public void RampThetaMove(float Angle) { … }
  public void RampThetaAbsMove(){ … }
  public void RampLockBall(){ … }
  public void RampReleaseBall(){ … }
  public void RampGetFlightTime(){ … }
  public void InitializeMachine(){ … }
  public void WaitReadyState(Devices Id) { … }
  public void MotorPowerON(){ … }
  public void MotorPowerOFF(){ … }
  private void AbsMove(Devices device, int steps, int speed { … }
  private void RelMove(Devices device, int steps, int speed { … }
  private void WaitBall(Devices sensorId, States state = States.Off) { … }
  public void ReadRange(){ … }
  public delegate void ExecuteAsyncCaller(float Height, float Angle, int Ball);
  public ExecuteResult Execute(float Height, float Angle, int Ball) { … }
  private void ExecuteAsync(float height, float angle, int ball) { … }
  private class ExecParameters
  {
    public float Height { get; set; }
    public float Angle { get; set; }
    public int Ball { get; set; }
  }
  private void ExecProcess(object parameters) { … }
  private void ExecuteDone(IAsyncResult ar) { … }
}

internal class LogInfo
{
  public  DateTime Date { get; private set; }
  public string Source { get; private set; }
  public string Message { get; private set; }
  public TimeSpan ElapsedTime { get; private set; }
  public bool IsValue { get; private set; }
  public float Value { get; private set; }
  public LogInfo(string source, string message, TimeSpan time) { … }
  public LogInfo(string source, string message, float value { … }
}
internal enum LogType {Info, Profiler, Warning, Error, Exception}
internal class AutoStopwatch : Stopwatch, IDisposable
{
  public DateTime StartTime { get; private set; }
  public DateTime EndTime { get; private set; }
  public string Source { get; private set; }
  public List<LogInfo> List = new List<LogInfo>();
  public AutoStopwatch(string source) { … }
  public void AddEntry(string job) { … }
  public void AddValue(string job, float value) { … }
  public string Header { … }
  public string TimeRecord { … }
  public void Dispose(){ … }
  private void OutputLog(){ … }
}

public static class Logger
{
  public static string Source { … }
  public static string SourceParent { … }
  public static void Error(string message) { … }
  public static void Error(Exception ex) { … }
  public static void Warning(string message) { … }
  public static void Info(string message) { … }
  private static void WriteEntry(date, source, message, type, time) { … }
```

```
    private static void WriteEntry(time, source, message, type) { … }
  }
}
```

Classe Labfarm.NanotecDataContext

```
namespace Labfarm
{
  public class NanotecDataContext : NanotecProtocol, INotifyPropertyChanged
  {
    public NanotecDataContext() { }
    override public string CommInput { … }
    override public string CommOutput { … }
    public float MovePosition { … }
    public event PropertyChangedEventHandler PropertyChanged;
    protected void OnPropertyChanged(string name) { … }
    public void InitLogs() { … }
  }
}
```

Classe Labfarm.NanotecProtocol

```
namespace Labfarm
{
  public class NanotecProtocol : SerialIO, IProtocol
  {
    public NanotecProtocol() { }
    public NanotecProtocol(portName, baudRate, portParity, dataBits, stopBits): base … { }
    private string RecordParam(RecParamEnum recParam) { … }
    static public string CommandString(CmdID cmd) { … }
    public override void MessageProcess(){ … }
    public string ProcessCmd(Devices Device, CmdID cmdId, bool IsRead, params int[] values) { … }
    private string LastCmd { get; set; }
    public string Answer { get { … } }
    private int PutInRange(int value, int A, int B) { … }
    private bool IsValid(string msg) { … }
    virtual public string CommInput { get; set; }
    virtual public string CommOutput { get; set; }
  }
}
```

Classe Labfarm.SerialIO

```
namespace Labfarm
{
  public abstract class SerialIO
  {
    public SerialPort PortRS = null;
    public event SerialIoErrorHandler Error;
    public bool HardwareFailed { get; private set; }
    protected SerialIO() { … }
    protected SerialIO(portName, baudRate, portParity, dataBits, stopBits): this(){ … }
    public abstract void MessageProcess();
    public void Close() { … }
    public bool Running { get { … } }
    public void Open(portName, baudRate, portParity, dataBits, stopBits) { … }
  }
}
```

Interface Labfarm.IProtocol

```
namespace Labfarm
{
  public interface IProtocol
  {
    string ProcessCmd(Devices Device, CmdID cmdId, bool IsRead, params int[] values);
    string Answer { get; }
  }
}
```

Anexo G. ESTRUTURA DE IMPLEMENTAÇÃO – MICROCONTROLADORES

Seletor de bola, Expansor SMCI12 elevador, Expansor SMCI12 Rampa

```
void main(void) { … }
char CheckSum(const char *sentence) { … }
void ClearBuffers() { … }
int ReadValue(unsigned char nr) { … }
void Initialize() { … }

struct { … } Status;
void InitUSART() { … }
void InitTimer1() { … }
void InitTimer0() { … }
void DelayMS(unsigned int ms) { … }
void interrupt ISR(void) { … }
void putch(unsigned char c) { … }
unsigned char mygetc() { … }
unsigned char waitc() { … }
void putst(register const char *str) { … }
void RepeatChr(char c, unsigned char qty) { … }
```

Caixa distribuição Rampa

```
void main(void) { … }
char CheckSum(const char *sentence) { … }
void ClearBuffers() { … }
int ReadValue(unsigned char nr) { … }
void SwitchMotor(int state) { … }
void SwitchSelenoid(int state) { … }
void Initialize() { … }

struct { … } Status;
void InitUSART() { … }
void InitTimer1() { … }
void InitTimer0() { … }
void DelayMS(unsigned int ms) { … }
void interrupt ISR(void) { … }
void putch(unsigned char c) { … }
unsigned char mygetc() { … }
unsigned char waitc() { … }
void putst(register const char *str) { … }
void RepeatChr(char c, unsigned char qty) { … }
```

Caixa distribuição Elevador

```
#include "..\include\IRBarrier_IOdefs.h"
#include "..\include\main.h"
#include "..\include\cpSerialAndTimer.h"

void main(void) { … }
char CheckSum(const char *sentence) { … }
void ClearBuffers() { … }
int ReadValue(unsigned char nr) { … }
void SwitchMotor(int state) { … }
void Initialize() { … }

struct { … } Status;
void InitUSART() { … }
void InitTimer1() { … }
void InitTimer0() { … }
void DelayMS(unsigned int ms) { … }
void interrupt ISR(void) { … }
void putch(unsigned char c) { … }
unsigned char mygetc() { … }
unsigned char waitc() { … }
void putst(register const char *str) { … }
void RepeatChr(char c, unsigned char qty) { … }
```

Sistema de medição

```
#include "..\include\IRBarrier_IOdefs.h"
#include "..\include\main.h"
#include "..\include\cpSerialAndTimer.h"

void main(void) { … }
char CheckSum(const char *sentence) { … }
void ClearBuffers() { … }
void Initialize() { … }

struct { … } Status;
void InitUSART() { … }
void InitTimer1() { … }
void InitTimer0() { … }
void DelayMS(unsigned int ms) { … }
void interrupt ISR(void) { … }
void putch(unsigned char c) { … }
unsigned char mygetc() { … }
unsigned char waitc() { … }
void putst(register const char *str) { … }
void RepeatChr(char c, unsigned char qty) { … }

void InitSPI(void) { … }
unsigned int ReadPosition(){ … }
void led_IO(void) { … }
static void ledShift(void) { … }
static void wrOutputs(void) { … }
static void rdInputs(void) { … }
static void updateMeasurement(void) { … }
static unsigned char getRisingEdgeIRValue(void) { … }
static unsigned char getFallingEdgeIRValue(void) { … }
static void getIRValue(void) { … }
```

Anexo H. ARTIGO PUBLICADO NA CONFERÊNCIA INTERNACIONAL REV2012

A Flexible Online Apparatus for Projectile Launch Experiments

Carlos Paiva1, Pedro Nogueira[1], Gustavo Alves[1], Arcelina Marques[2], Pedro Guimarães[2], Rubem Couto[2]
[1] Polytechnic of Porto – School of Engineering/Electronic Engineering Department, Porto, Portugal
[2] Polytechnic of Porto – School of Engineering/Physics Department, Porto, Portugal

***Abstract* — In order to provide a more flexible learning environment in physics, the developed projectile launch apparatus enables students to determine the acceleration of gravity and the dependence of a set of parameters in the projectile movement. This apparatus is remotely operated and accessed via web, by first scheduling an access time slot. This machine has a number of configuration parameters that support different learning scenarios with different complexities.**

***Index Terms* — Remote Experiment, projectile launch.**

INTRODUCTION

Distance education has been in use for several years at the Polytechnic of Porto – School of Engineering (ISEP), but the application of remote laboratories in physics was exclusively being used in the electric and electronic fields. For that purpose, VISIR [1, 2] and Remote ElectLab [3] supported remote experiments in electronics. The need for a similar approach supporting other experiments in Physics has led us to develop a machine able to perform projectile launch experiments remotely. This type of apparatus, supporting different complexity levels on the projectile movement characterization, can be used to address several topics on a typical physics curriculum, under different learning scenarios.

A similar approach with a free body fall apparatus to achieve gravity acceleration determination has been developed by Martin Connors and described in [4].

APPARATUS LAYOUT

This apparatus is a self-contained box that requires a power plug and an Ethernet connection. The box has an internal web server accessible through a SCORM-compliant scheduler. This server also enables a local mode for use in exhibitions and demonstrations. The apparatus is shown in Fig.1 and is divided in four sub-assemblies:

A. Ball Selector

Selects one out of three balls with diameters ranging from 14 mm to 18 mm, allowing the study of mass (non-) influence on projectile motion.

B. Main Elevator

Executes three operations in the process. Firstly, it loads the ball by moving to its lower limit loading position. Then it lifts-up the ball to a position leveled with the ramp. If the ramp is on place it will move up for a few millimeters to mechanically launch the ball to a zone where an electromagnet placed on the ramp will collect it.

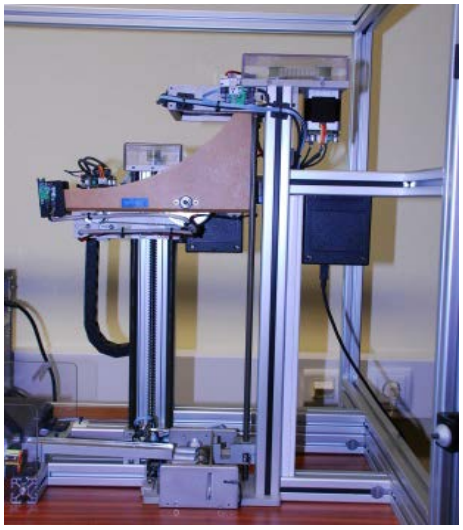


Figure 1. Apparatus side view

C. Ramp Elevator

Ramp Elevator has two main purposes: user setup angle and user setup height. The ramp angle can range from -20º to +20º. The launch height can be set up to 380 mm above touch down plane. Switching off the electromagnet triggers the ball movement down the ramp.

D. Ball Collector (landing zone)

Works by gravity and its main purpose is to collect the launched ball back to the ball selector. This assembly also contains the projectile horizontal range measuring system. This is an optical system based on a reflective infrared light barrier. The impact point measurement is obtained by interrupting a set of photo-detectors, 2 mm apart.

WEB INTERFACE

The web interface (Fig. 2) allows users to setup their experiment and receive the experimental data. It also provides the user with a live video stream of the whole process, captured by a webcam. Together with the experimental data, a photo taken at the moment the ball touches the landing zone is also sent via the user interface. The purpose of this photo is to allow users to make the actual measurement of the maximum horizontal displacement in order to compute the projectile range. All data is available for inspection via the user interface.

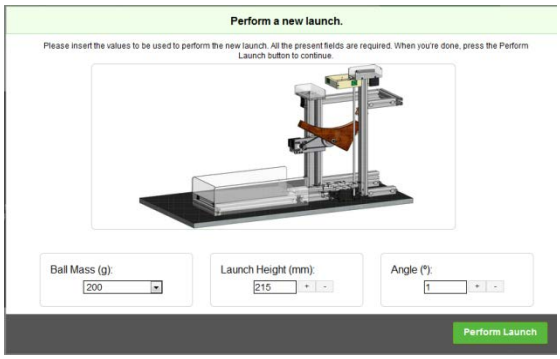


Figure 2. Web User Interface

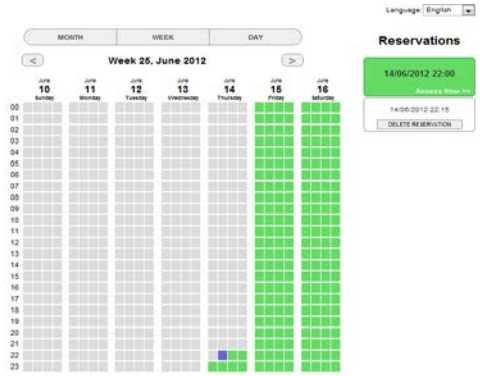


Figure 3. Scheduler Interface

EXPERIMENT FLOW

The experiment is accessed through a scheduler shown in Fig.3 via web. The student reserves a predefined time slot to execute the pretended experiments. During this period, he may submit data (using web services) to the machine as many times as possible.

For each trial, in the first interface window (Fig.2) students are requested to specify the experiment variables. These variables will be reflected in the machine setup. Upon reception of request, the machine validates all incoming data, verifies the availability of resources, checks for errors and, if everything is correct, returns a flag signaling "Experiment in progress". In case an error is found, the error code is returned.

As stated before, the machine process starts by selecting the desired ball at "Ball Selector" (Fig. 1). After selection, the main elevator goes down in order to load the selected ball into the elevator. This main elevator will raise the ball up to the ramp load position at its top most location and wait for ramp to signaling to be ready to accept the ball transfer.

When the ramp is ready, an electromagnet is turned ON to grab the ball. At this moment, the main elevator will push the ball and starts its trip down to be ready to receive next trial ball. The ramp with the ball coupled to the electromagnet turns to the desired angle and starts moving to reach the user selected height. After a small delay necessary to stabilize the mechanical vibration, the electromagnet is switched off and the projectile launch starts. The ball rolls down the ramp and will hit the landing zone in the ball collector where a sensor by detecting this event, computes the ball travel time. Also, the infrared light barrier detects the impact point and determines the maximum horizontal displacement, which in turn, will be used to compute the projectile range of motion. Finally, the ball collector will return it by gravity to the ball selector, therefore reaching the stop point necessary to the start of a new trial.

After completion of the machine cycle, experimental data returns to the web server and sent to the user.

RESULTS AND VALIDATION

This remote experimentation apparatus is being tested in order to be available to students in the upcoming school year. It will then be tested and validated under different learning scenarios. These tests will provide reports of working time, number of failures, recovery from failures, maintenance average time and will give MTBF (mean time between failures) and MTTR (mean time to repair) indices. From the server database, we will get users statistics, scheduler usage, system's idle time and user waiting time, amongst others. The learning gains will also be measured against the correlation among students' autonomous usage and students' performance on exam questions related to the same topic.

REFERENCES

Alves, G. R.; Marques, M. A.; Viegas, C.; Lobo, M. C.; Barral, R. G.; Couto, R. J.; Jacob, F. L.; Ramos, C. A.; Vilão, G. M.; Covita, D. S.; Alves, J.; Guimarães, P. S.; Gustavsson, I.; (2011). Using VISIR in a large undergraduate course: Preliminary assessment results. International Journal of Engineering Pedagogy, 1 (1), pp. 12-19. ISSN: 2192-4880).

Costa Lobo, M. C., Alves, G. R., Marques, M. A., Viegas, C., Barral, R. G., Couto, R. J., Jacob, F. L., Ramos, C. A., Vilão, G. M., Covita, D. S., Alves, J., Guimarães, P. S., Gustavsson, I., (2011).Using remote experimentation in a large undergraduate course: initial findings. In Proceedings 41st ASEE/IEEE Frontiers in Education Conference (FIE 2011), ISBN: 978-1-61284-467-1. (12-15 October 2011)

Nuno Sousa, Gustavo R. Alves, and Manuel G. Gericota, "An Integrated Reusable Remote Laboratory to Complement Electronics Teaching", IEEE Transactions on Learning Technologies, July-Sept. 2010, Vol. 3, nr. 3, pp. 265 – 271, ISSN: 1939-1382

Martin Connors, Christy Bredeson, Farook Al-Shamali,Distance Education Introductory Physics Labs: Online or In-Home?, chapter 15 in Using Remote Labs in Education, edited by Javier Garcia Zubía and Gustavo R. Alves, University of Deusto Press, ISBN: 978-84-9830-335-3, December 2011

AUTHORS

Carlos Paiva, Pedro Nogueira, Gustavo Alves, Arcelina Marques, Pedro Guimarães and **Rubem Couto** are with the Polytechnic of Porto – School of Engineering, R. Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal (e-mail: {cmolp, pmcno, gca, mmr, psg, prtc}@ isep.ipp.pt).

This work was supported by ISEP through internal funds.