

Faculdade de Engenharia da Universidade do Porto



**Servidor Modbus/TCP para Sistemas de
Identificação e Recolha de Dados Automáticos
(AIDC)**

José Filipe Alves Teixeira

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major Automação

Orientador: Prof. Dr. Paulo Portugal

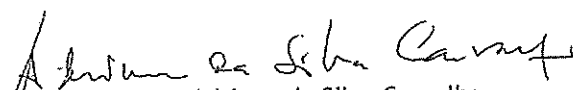
Julho de 2009

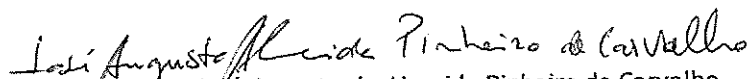
A Dissertação intitulada

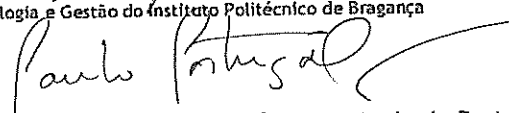
“SERVIDOR MODBUS/TCP PARA SISTEMAS DE IDENTIFICAÇÃO E DE RECOLHA DE
DADOS AUTOMÁTICOS (AIDC)”

foi aprovada em provas realizadas em 20/Julho/2009

o júri


Presidente Professor Doutor Adriano da Silva Carvalho
Professor Associado do Departamento de Engenharia Electrotécnica e de Computadores da
Faculdade de Engenharia da Universidade do Porto


Professor Doutor José Augusto de Almeida Pinheiro de Carvalho
Professor Adjunto do Departamento de Electrónica e de Computadores da Escola Superior de
Tecnologia e Gestão do Instituto Politécnico de Bragança


Professor Doutor Paulo José Lopes Machado Portugal
Professor Auxiliar do Departamento de Engenharia Electrotécnica e Computadores da
Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projecto) é da
sua exclusiva autoria e foi escrita sem qualquer apoio externo não
explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros
extractos tomados de ou inspirados em trabalhos de outros autores, e
demais referências bibliográficas usadas, são correctamente citados.


Autor - JOSÉ FILIPE ALVES TEIXEIRA

Faculdade de Engenharia da Universidade do Porto

Resumo

Nesta dissertação é feita uma apresentação de sistemas de identificação e recolha de dados automáticos e tipos de tecnologias existentes. Destes, são descritos com mais pormenor o código de barras, o RFID e os cartões de banda magnética.

Assim, propõe-se uma solução para integrar três dispositivos AIDC, um leitor RFID, um leitor de códigos de barras e um leitor de cartões de banda magnética, num sistema para ensino de tecnologias de automação. O objectivo é implementar um servidor Modbus que disponibilize funções para ler escrever dados a partir desses dispositivos. Neste sentido, a dissertação trata do estudo dos protocolos de comunicação, da criação de uma ligação TCP/IP, de uma API (*Application Programming Interface*) para as comunicações de cada leitor e da implementação do servidor Modbus.

Abstract

This dissertation is a presentation of Automatic Identification and Data Capture systems and types of existing technologies. The ones described in more detail are the barcode, RFID and magnetic stripe cards.

In this project a solution is proposed to integrate three AIDC devices, an RFID reader, a barcode reader and a magnetic stripe card reader, a system to promote the teaching of automation technologies. The aim is to implement a Modbus server that provides functions to read and write data from these devices. In this sense, the dissertation deals with the study of communication protocols, the establishment of a connection TCP/IP, an API (Application Programming Interface) for communications from each reader and the implementation of the Modbus server.

Agradecimentos

Gostaria de agradecer ao Professor Doutor Paulo Portugal todo o seu apoio, disponibilidade e conselhos dados ao longo do desenvolvimento deste trabalho.

Aos meus amigos que me ajudaram e deram apoio neste semestre, mas sobretudo ao longo de todo o percurso académico.

À minha família que sempre me motivou e ajudou. Em especial aos meus pais, que sempre acreditaram em mim e me proporcionaram esta oportunidade de terminar o curso superior.

Índice

Resumo	iii
Abstract.....	v
Agradecimentos	vii
Índice.....	ix
Lista de figuras	xi
Lista de tabelas	xv
Abreviaturas e Símbolos	xvii
Capítulo 1	1
Introdução.....	1
1.1 - Motivação	1
1.2 - Objectivo do trabalho.....	1
1.3 - Estrutura do documento	1
Capítulo 2	3
Tecnologias de Identificação e Recolha de Dados Automática	3
2.1 Identificação e recolha de dados automática	3
2.2 Código de barras	6
2.3 Identificação por Rádio Frequência - RFID	15
2.4 Cartões de Banda Magnética.....	19
2.5 Outras Tecnologias	23
Capítulo 3	27
Formulação do Problema	27
3.1 Linha de fabrico flexível de aprendizagem do DEEC	27
3.2 Objectivos	29
3.3 Leitor RFID	30
3.4 Leitor de códigos de barras	35
3.5 Leitor de cartões de banda magnética.....	37
Capítulo 4	39
Solução Proposta	39

4.1. ID ISOStart	39
4.2. XPort e ligação TCP/IP	45
4.3. API	47
4.4. Servidor Modbus	55
Capítulo 5	61
Validação da Solução	61
5.1. Leitor RFID	61
5.2. Leitores de códigos de barras e de cartões	65
Capítulo 6	69
Conclusão	69
6.1. Conclusões Gerais	69
6.2. Trabalho Futuro	70
Referências	71

Lista de figuras

Figura 2.1 - Código de barras (a) linear e (b) bidimensional.	6
Figura 2.2 - Código de barras modulado por largura.	7
Figura 2.3- Código de barras modulado por altura.	7
Figura 2.4 - <i>Narrow bar</i>	8
Figura 2.5 - <i>Quiet zone</i>	8
Figura 2.6 - <i>Start e Stop Character</i>	8
Figura 2.7 - <i>Data characters e Check character</i>	9
Figura 2.8 - <i>Intercharacter gap</i>	9
Figura 2.9 - Simbologia UPC/EAN.	10
Figura 2.10 - Simbologia Code 39.	11
Figura 2.11 - Simbologia Code 128.	11
Figura 2.12 - Simbologia Interleaved 2 of 5.	12
Figura 2.13 - Princípio de funcionamento de um leitor de códigos de barras [3].	12
Figura 2.14 - Leitor de códigos de barras portátil com feixe de luz fixo.	13
Figura 2.15 - Leitor de códigos de barra fixo.	13
Figura 2.16 - Leitor de códigos de barras sem fios.	14
Figura 2.17 - Códigos de barras 2-D do tipo <i>stacked</i>	14
Figura 2.18 - Códigos de barras 2-D do tipo <i>matrix</i>	15
Figura 2.19 - Exemplos de <i>transponders</i> de RFID.	16
Figura 2.20 - Exemplos de leitores RFID.	16
Figura 2.21 - Princípio de funcionamento do RFID [8].	17

Figura 2.22 - Exemplo de uma sequência de leitura e descodificação de uma banda magnética [10].	20
Figura 2.23 - Características da banda magnética de um cartão.	21
Figura 2.24 - Pista 1 de um cartão financeiro.	21
Figura 2.25 - Pista 2 de um cartão financeiro.	21
Figura 2.26 - Pista 3 de um cartão financeiro.	22
Figura 2.27 - Exemplos de leitores e gravadores de cartões de banda magnética.	22
Figura 2.28 - Exemplos de leitores OCR.	23
Figura 2.29 - Exemplos da tecnologia de visão industrial.	24
Figura 2.30 - Exemplos da tecnologia de cartões inteligentes.	24
Figura 2.31 - Exemplo de ecrãs tácteis.	25
Figura 2.32 - Exemplos de tecnologias biométricas.	26
Figura 3.1 - Linha de fabrico flexível de aprendizagem do DEEC	27
Figura 3.2 - Constituição da linha de fabrico flexível do DEEC	28
Figura 3.3 - Esquema do sistema da linha de fabrico flexível integrando os dispositivos AIDC.	30
Figura 3.4 - Leitor RFID ID ISC.PR101-A.	31
Figura 3.5 - Troca de dados entre o <i>host</i> e o leitor RFID utilizando <i>Configuration Commands</i> e <i>Control Commands</i>	32
Figura 3.6 - Troca de dados entre o <i>host</i> e o leitor RFID no modo <i>ISSO 15693 Host Commands: Addressed Mode</i>	33
Figura 3.7 - Troca de dados entre o <i>host</i> e o leitor RFID no modo <i>ISSO 15693 Host Commands: Non-Addressed Mode</i>	33
Figura 3.8 - Troca de dados entre o <i>host</i> e o leitor RFID no modo <i>ISSO 15693 Host Commands: Selected Mode</i>	34
Figura 3.9 - Peça da linha de fabrico flexível com a etiqueta RFID embutida.	35
Figura 3.10 - Scanner CCD 1021Plus	36
Figura 3.11 - Leitor de cartões de banda magnética Cipher 1023.	37
Figura 4.1 - Diagrama da aplicação implementada.	39
Figura 4.2 - Interface gráfica do programa ID ISOStart.	40
Figura 4.3 - Resposta ao comando <i>Inventory</i> e <i>Get System Information</i> utilizando o ID ISOStart.	41
Figura 4.4 - <i>Standard Protocol</i>	41

Figura 4.5 - <i>Advanced Protocol</i>	42
Figura 4.6 - Trama de envio e resposta do comando <i>Read Configuration</i>	43
Figura 4.7 - Trama de envio e resposta do comando <i>Write Configuration</i>	43
Figura 4.8 - Exemplo dos comandos <i>Read Configuration</i> e <i>Write Configuration</i> utilizando o ID ISOStart.	44
Figura 4.9 - Trama de envio e resposta do comando <i>Read Multiple Blocks</i>	44
Figura 4.10 - Trama de envio e resposta do comando <i>Write Multiple Blocks</i>	45
Figura 4.11 - Exemplo dos comandos <i>Read Read Multiple Blocks</i> e <i>Write Multiple Blocks</i> utilizando o ID ISOStart.	45
Figura 4.12 - XPort.	46
Figura 4.13 - Fluxograma da função <i>open_connection()</i>	48
Figura 4.14 - Fluxograma da função <i>close_connection()</i>	48
Figura 4.15 - Fluxograma da função <i>rd_config()</i>	49
Figura 4.16 - Fluxograma da função <i>wr_config()</i>	50
Figura 4.17 - Fluxograma da função <i>init_config()</i>	51
Figura 4.18 - Fluxograma da função <i>rfid_init()</i>	52
Figura 4.19 - Fluxograma da função <i>read_mb()</i>	52
Figura 4.20 - Fluxograma da função <i>write_mb()</i>	53
Figura 4.21 - Fluxograma da função <i>scan_read()</i>	54
Figura 4.22 - Fluxograma da função <i>read_bc()</i>	55
Figura 4.23 Fluxograma das funções <i>read_outwords()</i> e <i>write_outwords()</i>	58
Figura 4.24 - Fluxograma da função <i>write_outwords()</i> para o registo CMD.	59
Figura 4.25 - Exemplo do ficheiro de texto de configuração dos leitores RFID.	60
Figura 5.1 - Interface gráfica com o utilizador do programa Modbus Poll.	61
Figura 5.2 - Leitura dos registos Modbus 0 a 7.	62
Figura 5.3 - Escrita e leitura de dados de uma etiqueta a partir do Modbus Poll.	63
Figura 5.4 - Funcionamento em <i>Scan Mode</i> a partir do Modbus Poll.	63
Figura 5.5 - Alteração do registo Modbus 5.	63
Figura 5.6 - Comando <i>reset</i> a partir do Modbus Poll.	64
Figura 5.7 - Exemplos de erros Modbus.	65
Figura 5.8 - Leitura dos registos Modbus 2000 a 2006.	66

Figura 5.9 - Leitura de um código de barras a partir do Modbus Poll.	66
Figura 5.10 -	67
Figura 5.11 - Leitura dos registos Modbus 4000 a 4019.	67

Lista de tabelas

Tabela 2.1 - Comparação entre algumas tecnologias AIDC [1].	5
Tabela 2.2 - Comparação entre o código de barras e o RFID [1], [6].	15
Tabela 2.3 - Classificação dos sistemas RFID quanto à frequência de comunicação [6].	18
Tabela 3.1 - Características do leitor RFID ID ISC.PR101-A.	31
Tabela 3.2 - Funções disponíveis no modo de configuração.	32
Tabela 3.3 - Funções disponíveis no modo de controlo.	32
Tabela 3.4 - Funções disponíveis para o modo <i>ISO 15693 Host Commands</i>	34
Tabela 4.1 - Descrição da trama do protocolo de comunicação do leitor RFID.	42
Tabela 4.2- Descrição das tramas dos comandos <i>Read Configuration</i> e <i>Write Configuration</i>	43
Tabela 4.3 - Descrição das tramas dos comandos <i>Read Multiple Blocks</i> e <i>Write Multiple Blocks</i>	45
Tabela 4.4 - Dados da estrutura criada para um leitor RFID.	47
Tabela 4.5 - Dados da estrutura criada para um leitor de códigos de barras ou de cartões...	54
Tabela 4.6 - Organização dos registos Modbus para um leitor RFID.	56
Tabela 4.7 - Organização dos registos Modbus para um leitor de códigos de barras.	56
Tabela 4.8 - Organização dos registos Modbus para um leitor de cartões.	57

Abreviaturas e Símbolos

Lista de abreviaturas

ABA	<i>American Bankers Association</i>
ADC	<i>Automatic Data Capture</i>
ADN	Ácido Desoxirribonucleico
AIDC	<i>Automatic Identification and Data Capture</i>
API	<i>Application Programming Interface</i>
ASCII	<i>American Standard Code for Information Interchange</i>
Auto-ID	<i>Automatic Identification</i>
CC	Corrente Contínua
CCD	<i>Charge-Coupled Device</i>
CPU	<i>Central Processing Unit</i>
CRC	<i>Cyclic redundancy check</i>
DEEC	Departamento de Engenharia Electrotécnica e de Computadores
DST	<i>Dispersive Signal Technology</i>
EAN	<i>European Article Numbering</i>
EAS	<i>Electronic Article Surveillance</i>
FRR	<i>First Read Rate</i>
HF	<i>High Frequency</i>
IATA	<i>International Air Transport Association</i>
IC	<i>Integrated Circuit</i>
IFF	<i>Identification Friend or Foe</i>
IP	<i>Internet Protocol</i>
ISO	<i>International Organization for Standardization</i>
LED	<i>Light-Emitting Diode</i>
LF	<i>Low Frequency</i>
LSB	<i>Least Significant Bit</i>
MAC	<i>Media Access Control</i>

MICR	<i>Magnetic Ink Character Recognition</i>
MSB	<i>Most Significant Bit</i>
NRMA	<i>National Retail Merchants Association</i>
OCR	<i>Optical Character Recognition</i>
PCB	<i>Printed Circuit Board</i>
PIN	<i>Personal Identification Number</i>
PLC	<i>Programmable Logic Controller</i>
PS/2	<i>Personal System/2</i>
RF	<i>Radio Frequency</i>
RFID	<i>Radio Frequency Identification</i>
SER	<i>Substitution Error Rate</i>
SHF	<i>Super high frequency</i>
SPI	<i>Serial Peripheral Interface</i>
TCP	<i>Transmission Control Protocol</i>
UHF	<i>Ultra High Frequency</i>
UID	<i>Unique Identifier</i>
UPC	<i>Universal Product Code</i>
VLD	<i>Visible Laser Diode</i>

Capítulo 1

Introdução

1.1 - Motivação

Numa era de desenvolvimento tecnológico, a variedade de soluções de automação para um determinado sistema é vasta e surge muitas vezes a necessidade de utilizar dispositivos de diferentes marcas e tecnologias. Isto acarreta problemas ou dificuldades de os interligar tanto a nível físico, como a nível de protocolos de comunicação. Deste modo, cada vez mais se dá importância à normalização e tenta-se, a todo custo, definir interfaces e protocolos de comunicação comuns à maioria dos dispositivos.

A problemática que motivou esta dissertação parte desse contexto e da necessidade de interligar vários dispositivos com diferentes protocolos de comunicação, disponibilizando apenas um para o utilizador.

1.2 - Objectivo do trabalho

Pretende-se, com a realização deste trabalho, integrar três dispositivos de identificação e recolha de dados automática numa mesma aplicação, sem a necessidade de recorrer a protocolos de comunicação diferentes, para a sua utilização. Os dispositivos consistem num leitor RFID, um leitor de códigos de barras e um leitor de cartões de banda magnética. O protocolo de comunicação comum desejado é o Modbus. Assim, o objectivo do trabalho é criar um programa que implemente um servidor Modbus, capaz de comunicar com os três equipamentos.

1.3 - Estrutura do documento

Este documento descreve todo o trabalho que foi realizado e optou-se por dividir em 6 capítulos.

No Capítulo 1 é descrito, de forma breve, o que é pretendido com a realização deste trabalho.

No Capítulo 2 faz-se uma introdução aos sistemas de identificação e recolha de dados automáticos e apresentam-se algumas tecnologias mais comuns e o seu princípio de funcionamento.

No Capítulo 3 descreve-se a proposta de trabalho, os equipamentos utilizados na sua realização e os objectivos propostos.

No Capítulo 4 apresenta-se o trabalho desenvolvido, desde o estudo dos protocolos de comunicação dos equipamentos, a ligação TCP/IP, a API para comunicação com os leitores até à arquitectura do servidor Modbus.

No Capítulo 5 são descritos alguns testes que se realizaram para validação da solução implementada.

Por último, no Capítulo 6 apresentam-se as conclusões do trabalho implementado e propostas para um trabalho futuro.

Capítulo 2

Tecnologias de Identificação e Recolha de Dados Automática

Neste capítulo será feita uma apresentação dos sistemas de identificação e recolha de dados automáticos, vantagens e suas aplicações. Serão apresentadas ainda algumas tecnologias que existem no mercado e o seu funcionamento. As tecnologias utilizadas neste trabalho merecerão especial atenção.

2.1 Identificação e recolha de dados automática

Sistemas de identificação e recolha de dados automáticos (AIDC - *Automatic Identification and Data Capture*), também muitas vezes designados apenas por sistemas de identificação automática (Auto-ID - *Automatic Identification*), ou ainda, sistemas de recolha de dados automáticos (ADC - *Automatic Data Capture*), referem-se a tecnologias utilizadas para a entrada directa de dados num computador, PLC ou noutro tipo de microprocessador para identificar, monitorizar e recolher dados de objectos ou pessoas, sem a intervenção humana directa e com grande fiabilidade. É possível codificar uma vasta gama de informação, desde a identificação de um objecto ou pessoa até informações detalhadas dos mesmos.

Estas tecnologias estão presentes no dia-a-dia de cada pessoa, desde códigos de barras dos produtos dos supermercados, a cartões de banda magnética que permitem o acesso a quartos de hotéis até cartões inteligentes (*smart cards*) utilizados em cartões financeiros. Impressões digitais utilizadas para autenticação em computadores portáteis, reconhecimento de voz para a marcação de números no telemóvel, RFID utilizado na Via Verde são outros exemplos entre muitos. Na indústria, a sua utilização tem uma enorme importância e tem crescido cada vez mais, tanto na movimentação de materiais (*material handling*) como em aplicações de produção (*manufacturing*). A movimentação de materiais normalmente inclui acções como expedição, recepção, armazenamento e movimentação de objectos e aplicações de produção tais como, acompanhamento das ordens de fabrico, monitorização da taxa de utilização das máquinas, produtividade dos funcionários, desempenho do processo produtivo e *work-in-process*. Outras áreas de aplicação também importantes destas tecnologias são as vendas a retalho e controlo de inventário, operações de centros de distribuição e armazenamento, correios, identificação de pacientes em hospitais, sistemas de segurança, entre outros [1].

Devido à grande variedade de soluções que estas tecnologias oferecem, não existe uma solução óptima. A melhor solução de recolha de dados para uma determinada aplicação pode não ser a melhor para outra. Cada uma das tecnologias AIDC tem vantagens específicas e características que as tornam mais adequadas para algumas aplicações que outras.

Este tipo de sistemas, que começou a dar os primeiros passos nos anos setenta [2], veio substituir a recolha de dados manual. Este método implica normalmente a existência de uma pessoa a recolher dados para um papel e posteriormente a necessidade de inserir os mesmos num computador recorrendo a um teclado. Possui desvantagens significativas tais como [1]:

- Ocorrência de erros tanto na recolha de dados como na utilização do teclado para os introduzir no computador. A média da taxa de erros de utilização de um teclado, é de 1 erro a cada 300 caracteres.
- O tempo. Métodos manuais são logicamente mais lentos que os automáticos. Além disso, existe um intervalo de tempo desde que os eventos ocorrem até ao instante que os dados são inseridos.
- Custo da mão-de-obra. Nos métodos manuais de recolha de dados é necessária uma grande intervenção humana a que está associado um custo.

Com as tecnologias de identificação e recolha de dados automática estas desvantagens são parcialmente eliminadas e é possível obter maior fiabilidade e segurança na recolha de dados, redução do intervalo de tempo entre as ocorrências dos eventos e o processamento dos respectivos dados e redução nos custos de mão-de-obra e de produção. Como exemplo, considere um sistema AIDC para prevenir a perda de arquivos numa empresa. Cada arquivo é dotado de um rótulo que pode ser lido por uma máquina e sempre que muda de secretária é feita uma leitura que envia os dados para uma base de dados. Este sistema praticamente não consome tempo e é possível saber sempre por onde o arquivo passou e onde está. Isto, além de evitar a perda dos arquivos, impede que o funcionário perca tempo a procurar determinado arquivo e ainda torna possível ter uma visão de todo o processo, bem como, localizar problemas importantes como operações que atrasam o processo para eliminar *bottlenecks*.

As tecnologias AIDC podem ser classificadas em seis categorias:

- Tecnologias ópticas. Caracterizam-se por utilizar o contraste de símbolos gráficos para codificar informação e que podem ser interpretados por um scanner óptico. Incluem códigos de barras (uni e bidimensional), reconhecimento óptico de caracteres (OCR) e visão industrial.
- Tecnologias magnéticas. Caracterizam-se por codificar a informação de forma magnética, tal como numa cassette de vídeo. Incluem duas importantes técnicas que são as bandas magnéticas, usadas nos cartões de crédito bancários e o *magnetic ink character recognition* (MICR) usado nos bancos para processamentos de cheques.
- Tecnologias Electromagnéticas. O mesmo é falar em identificação por rádio frequência (RFID). Caracteriza-se por utilizar pequenas etiquetas (*tags*) que guardam informação e que podem ser lidas sem contacto através de um leitor com antena.

- Cartões inteligentes (*Smart Cards*). São cartões de plástico que possuem um circuito integrado com capacidade de armazenar e processar dados.
- Tecnologias de toque. Funcionam com a acção de um simples toque e incluem os ecrãs tácteis e os *button memory*.
- Tecnologias biométricas. Caracterizam-se por identificar pessoas através de características humanas únicas ou interpretar comandos através da voz. Incluem análise de impressões digitais, reconhecimento facial, de retina e voz.

Algumas das aplicações AIDC requerem o envolvimento de um operador no procedimento de recolha de dados, normalmente para manusear o equipamento utilizado. Este tipo de aplicações na verdade não é um método automático e são chamados de semi-automáticos. As tecnologias referidas em cima podem ter os dois tipos de aplicação. Por exemplo, os códigos de barras podem ser lidos por leitores manuseados por uma pessoa ou simplesmente de forma automática quando passam num tapete rolante.

Quase todas as tecnologias são compostas por três dispositivos principais, um codificador de dados, um leitor ou scanner e um decodificador de dados. Usualmente, o primeiro guarda os dados sob a forma de símbolos ou sinais que representam caracteres alfanuméricos, como por exemplo, os rótulos do código de barras ou as etiquetas do RFID. O leitor ou scanner lê os dados codificados e transforma-os num sinal eléctrico. Por último o decodificador de dados transforma o sinal eléctrico em dados digitais voltando à forma inicial de caracteres alfanuméricos. Estes dois últimos dispositivos normalmente estão integrados apenas num, como por exemplo um leitor de código de barras que envia os códigos ASCII dos caracteres para um computador.

Como já foi referido anteriormente, os sistemas de identificação e recolha de dados automáticos reduzem muito a taxa de erros, cerca de 10000 vezes menos que a inserção de dados manual através de um teclado para o código de barras, por exemplo. Ainda assim, os erros ocorrem e foram adoptados dois parâmetros de medida para os mesmos [1]:

- Taxa de primeira leitura (FRR). É a probabilidade de uma leitura bem sucedida na primeira tentativa do leitor.
- Taxa de erros de substituição (SER). É a probabilidade ou frequência com que o scanner lê incorrectamente um carácter, isto é, o carácter lido não corresponde ao codificado. Num conjunto de dados codificados contendo n caracteres, o número de erros esperados é igual a SER multiplicado por n .

Como se pode verificar, para qualquer aplicação AIDC pretende-se uma elevada taxa de primeira leitura e uma baixa taxa de erros de substituição. Na tabela seguinte é possível analisar a taxa de erros e comparar diferentes tecnologias AIDC.

Tabela 2.1 - Comparação entre algumas tecnologias AIDC [1].

Tecnologia	Tempo de inserção de dados	Taxa de erro	Custo do equipamento	Vantagens/(Desvantagens)
Inserção manual	Lento	Elevada	Baixo	Baixo custo inicial (Requer um operador humano) (Lento) (Elevada taxa de erro)

Código de barras 1-D	Médio	Baixa	Baixo	Rápido Grande Flexibilidade (Baixa densidade de dados)
Código de barras 2-D	Médio	Baixa	Elevado	Rápido Elevada densidade de dados
RFID	Rápido	Baixa	Elevado	A etiqueta não precisa de estar visível para o leitor Capacidade de leitura e escrita Elevada densidade de dados (Etiquetação mais cara)
Banda magnética	Médio	Baixa	Médio	Podem ser codificados muitos dados Os dados poder ser alterados (Vulnerável a campos magnéticos) (Necessita de contacto para leitura)
OCR	Médio	Média	Médio	Podem ser lidos por pessoas (Baixa densidade de dados) (Elevada taxa de erros)
Visão industrial	Rápido	Depende da aplicação	Muito elevado	Rápida (Equipamento caro) (Não adequada para as aplicações AIDC gerais)

2.2 Código de barras

Antes de surgir o código de barras, apenas existia a identificação manual. Rótulos escritos à mão ou copiados em papel químico eram anexados aos objectos que necessitavam de identificação. Em 1932, o primeiro estudo sobre a automação nas caixas de supermercados foi conduzido por Wallace Flint e em 1934 uma patente do código de barras que descrevia o uso de linhas paralelas foi registado. O primeiro código de produto universal (UPC), que é um exemplo de simbologia do código de barras, a atravessar o scanner de um supermercado foi em 1974 [2].

Existem dois tipos de código de barras, linear ou unidimensional e bidimensional. O primeiro consiste numa sequência de barras que é lida apenas numa direcção e o segundo consiste numa matriz de barras que necessita de ser lida em duas direcções.



Figura 2.1 - Código de barras (a) linear e (b) bidimensional.

Código de barras linear

Das tecnologias de identificação e recolha de dados automática, o código de barras linear é o mais utilizado. Este tipo de código de barras pode codificar a informação de 2 modos diferentes: modulação por largura (*width-modulated*), representada na Figura 2.2, e modulação por altura (*height-modulated*), representada na Figura 2.3. Na modulação por largura, os caracteres são codificados numa sequência de barras e espaços de largura variável. Na modulação por altura, a codificação é feita através de uma sequência de barras espaçadas de altura variável.



Figura 2.2 - Código de barras modulado por largura.



Figura 2.3- Código de barras modulado por altura.

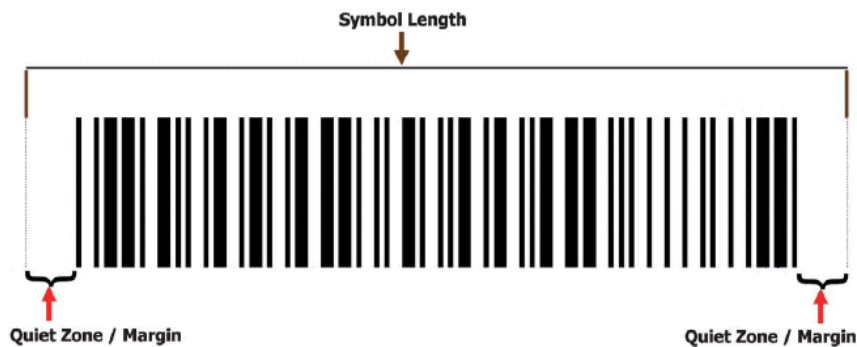
Como o código de barras modulado por largura é claramente mais utilizado que o código de barras modulado por altura, e este último não possui grandes aplicações, apenas será apresentado o primeiro.

O código de barras modulado por largura, como já foi dito, consiste num conjunto de barras estreitas e largas intercalados por espaços, também por sua vez, estreitos e largos. Normalmente, as barras apresentam cor escura ou preta e os espaços cor clara ou branca criando um maior contraste, o que facilita a leitura. Existem dezenas de simbologias e cada uma tem barras de dimensões diferentes, bem como diferentes codificações e número de caracteres representados. Algumas delas serão descritas mais à frente, mas de seguida são referidas características comuns a quase todos os tipos de código de barras.

Narrow bar ou *x-dimension* (Figura 2.4) define a dimensão (largura) nominal das barras ou espaços mais estreitos de cada simbologia de código de barras e é utilizada para determinar o tamanho da escala do código de barras. A unidade de medida normalmente utilizada é o milímetro ou nos Estados Unidos o *mil* (0.001 *inch*). Quando se escolhe o tamanho do código de barras para uma aplicação específica, deve-se optar por a *narrow bar* mais larga que a aplicação o permite. Deste modo, será mais fácil tanto imprimir como ler o código de barras e além disso diminuiu possíveis erros de leitura pois a área de leitura é maior. Associado a esta característica surge o termo densidade. A densidade de um código de barras é o número de caracteres por milímetro ou *inch* e é tanto maior quanto mais pequena for a dimensão da *narrow bar*.

Figura 2.4 - *Narrow bar*.

O tamanho do código de barras, ao contrário do que se possa pensar, não se limita à zona onde começam e acabam as barras. Existe uma zona chamada *quite zone* ou “no print” zone no início e fim do código, com pelo menos dez vezes o tamanho da largura da *narrow bar*, como se pode ver na Figura 2.5. Isto é importante uma vez que se estas zonas não forem visíveis ou contiverem algum texto ou grafismo, o código de barras pode não ser lido.

Figura 2.5 - *Quiet zone*.

Ao contrário das pessoas, que lêem sempre no mesmo sentido (geralmente da esquerda para a direita), os leitores de código de barras podem ler nos dois sentidos. Isto faz com que a maioria das simbologias utilize um carácter de início e outro de paragem (*start* e *stop character*) para indicar ao leitor a ordem dos dados codificados, Figura 2.6. No entanto, nem todas utilizam estes caracteres, como é o caso do *Pharma Code*, e por isso dependendo da direcção em que o código de barras é colocado, pode dar origem a leituras diferentes.

Figura 2.6 - *Start e Stop Character*.

A informação do código de barras é codificada nos caracteres de dados (*data characters*), como se pode ver na Figura 2.7, e o seu tamanho varia com a simbologia utilizada. São esses dados que vão ser lidos pelo scanner e apresentados ao utilizador. Também dependendo da

simbologia utilizada, o código de barras pode incluir um carácter de confirmação (*check character*), Figura 2.7. Em alguns casos esse carácter é incluído automaticamente e noutros pode ser opcional, sendo que se deve utilizar sempre que possível, ainda mais quando se pretende integridade na aplicação.



Figura 2.7 - Data characters e Check character.

A simbologia dos códigos de barras lineares pode ser classificada como contínua ou discreta. Nos códigos de barras contínuos um carácter começa imediatamente a seguir ao anterior, sem que haja intervalo entre eles. Um carácter tipicamente começa numa barra e acaba num espaço e todos os espaços fazem parte do código. Já nos códigos de barras discretos existem intervalos entre caracteres (*intercharacter gap*), ou seja, espaços que não contêm dados representados na Figura 2.8.

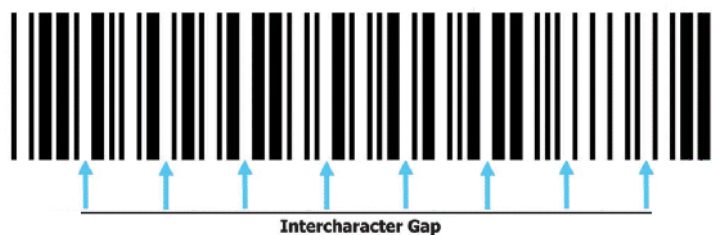


Figura 2.8 - Intercharacter gap.

Outras características de um código de barras também importantes são a altura das barras e o rácio do contraste. Relativamente à primeira, é fácil perceber que quanto maior for a altura das barras mais fácil será a leitura e ainda aumenta as possibilidades de efectuar leituras correctas com parte do rótulo danificado. O rácio do contraste é a percentagem de contraste entre as barras e os espaços e como já foi referido, facilita a descodificação do scanner e evita erros provocados pelo ruído.

Serão agora apresentadas algumas das simbologias mais comuns.

UPC/EAN

A simbologia de código de barras UPC foi a primeira a ser largamente utilizada nos Estados Unidos para identificação de produtos. O seu sucesso conduziu à criação de um código similar utilizado mundialmente, o EAN. Actualmente existem cinco versões do UPC, sendo as versões A e E as mais comuns, e duas do EAN, EAN-8 e EAN-13. Todas estas simbologias são numéricas,

ou seja, os caracteres de dados só contêm números. A versão A do UPC é composta por 12 dígitos: 1 que indica o tipo de produto, 5 que identificam o fabricante, 5 o produto e 1 de verificação (*check character*); 3 padrões de guarda constituídos por *narrow bar*, *narrow space* e *narrow bar*, no início, a meio e no fim do código. A versão E é uma forma compacta da versão A (não representa os zeros) e é composta por 6 dígitos mais 2 padrões de guarda, no início e no fim do código (*check character* é codificado nos 6 dígitos de dados). O EAN-13 tem o mesmo número de barras que o UPC versão A, mas contém 13 dígitos: 2 ou 3 que indicam o código do país, 4 ou 5 que identificam o fabricante, 5 o produto e 1 de verificação (codificado da mesma maneira que o da versão E do UPC). O EAN-8 é igual ao EAN-13 mas apenas com 8 dígitos, pois utiliza um número de identificação mais pequeno. De referir ainda que estas simbologias são contínuas e cada carácter é codificado por 2 barras e 2 espaços. Exemplos de códigos de barras são mostrados na figura seguinte.

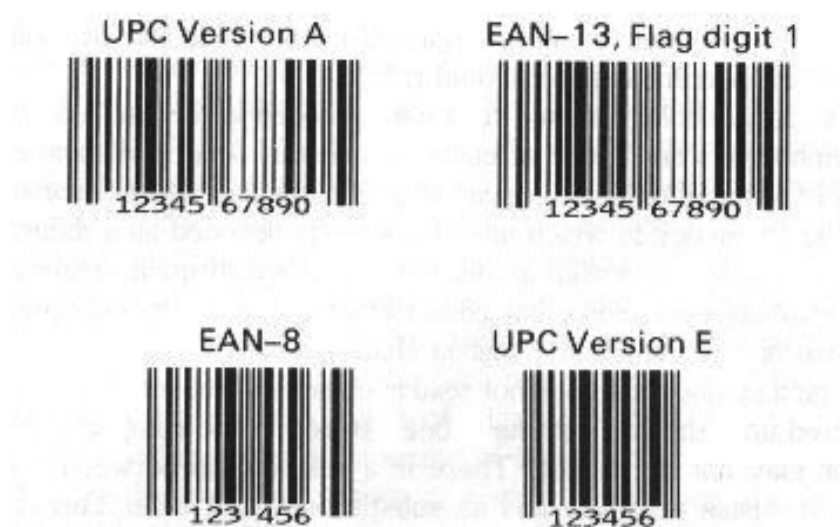


Figura 2.9 - Simbologia UPC/EAN.

Code 39

Code 39 ou Code 3 of 9 foi a primeira simbologia alfanumérica a ser desenvolvida. Não tem tamanho definido e por isso pode variar conforme o número de dados codificados. Pode representar 26 letras maiúsculas, 10 dígitos (0 - 9) e 7 caracteres especiais. Cada carácter é representado por 5 barras e 4 espaços, em que 3 destes elementos (barras ou espaços) são sempre largos (motivo pelo qual foi atribuída a designação 39) e o carácter de início e de paragem é um asterisco. Como se pode concluir, a simbologia é discreta. Quanto à sua utilização, a indústria de defesa e a indústria automóvel são as suas principais aplicações.



Figura 2.10 - Simbologia Code 39.

Code 128

Tal como na simbologia anterior, o Code 128 é alfanumérico e com tamanho variável. A sua grande vantagem é a alta densidade de codificação, que de todas as simbologias lineares é a que utiliza menos área para codificar 6 caracteres ou mais. Este tipo de código de barras permite a representação de todos os caracteres ASCII e usa 3 barras e 3 espaços (simbologia contínua) para codificar cada dígito. As barras e espaços neste tipo de simbologia podem ter 4 tamanhos diferentes. Isso resulta em 106 diferentes combinações, o que implica que, para poder codificar os 128 caracteres ASCII são necessários 3 caracteres de início diferentes (*star code A*, *start code B* e *start code C*) que indicam quais os caracteres utilizados. O *code A* contém os caracteres alfanuméricos do teclado mais os caracteres de controlo e especiais; o *code B* contém os caracteres alfanuméricos do teclado, as letras minúsculas e os caracteres especiais; o *code C* contém pares de números desde 00 até 99, o que duplica a densidade de dados numéricos. Na Figura 2.11 está representado um código do tipo Code 128.

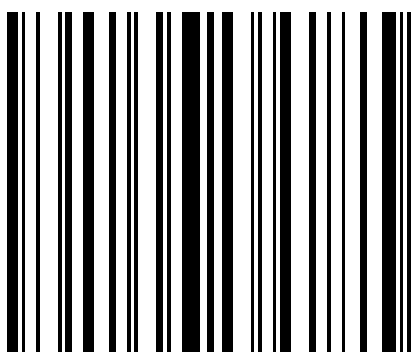


Figura 2.11 - Simbologia Code 128.

Interleaved 2 of 5

A simbologia Interleaved 2 of 5 serve apenas para codificar números. O seu tamanho é variável e possui densidade de codificação elevada. Contém um *check character* mas é opcional. Os dígitos são codificados por 5 barras (2 delas largas) ou 5 espaços (2 deles largos), isto é, um dígito está codificado apenas nas barras e o seguinte apenas nos espaços alternadamente, daí o nome *interleaved* (intercalado). Como se pode deduzir rapidamente depois disto, esta é uma simbologia contínua pois não existem espaços sem informação. Os caracteres de início e de paragem são formados pelas sequências *narrow bar*, *narrow space*,

narrow bar, narrow space e wide bar, narrow space, narrow bar, respectivamente (Figura 2.12).

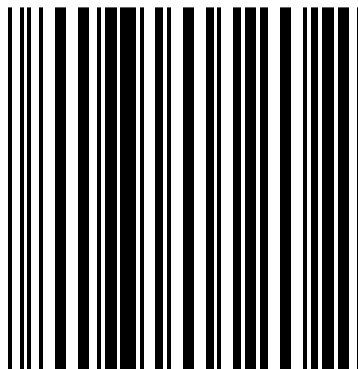


Figura 2.12 - Simbologia Interleaved 2 of 5.

Leitores de códigos de barras

A função principal de um leitor de códigos de barras é ler o código e descodificá-lo em caracteres. O seu funcionamento geral consiste num feixe de luz, criado por um LED (dispositivos CCD) ou um díodo de laser visível (VLD) e um conjunto de espelhos, que é direccionado para o código. Esse feixe de luz vai ser absorvido pelas barras pretas e reflectido pelos espaços, sendo de seguida transformado no sinal eléctrico correspondente através de um fotodíodo e convertido num sinal digital. Dependendo do leitor utilizado, existem algumas diferenças que variam com a tecnologia utilizada. Na Figura 2.13 pode-se ver o princípio de funcionamento de um leitor de código de barras com feixe móvel.

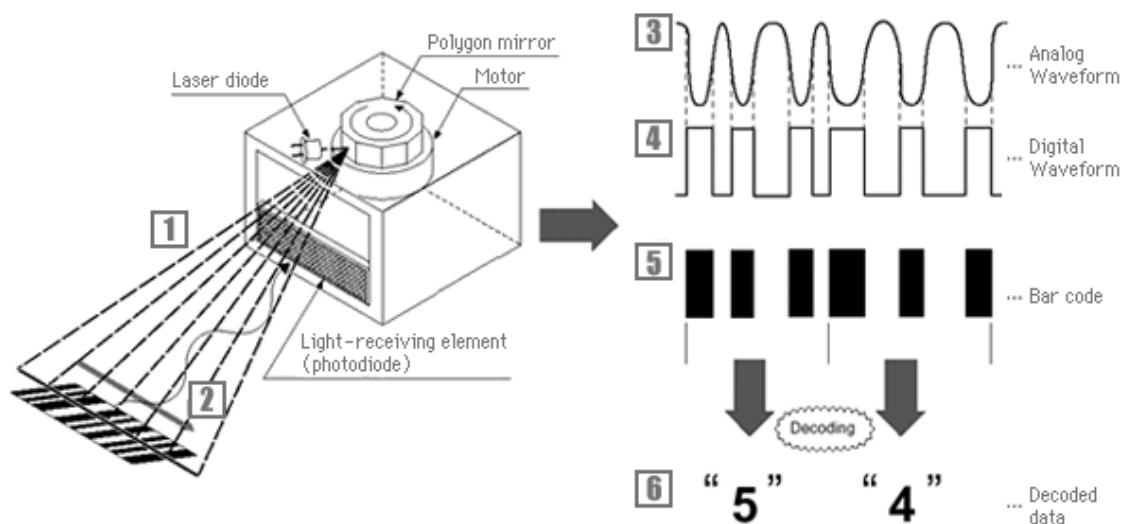


Figura 2.13 - Princípio de funcionamento de um leitor de códigos de barras [3].

Existe uma grande variedade de leitores, uns que necessitam de intervenção humana e outros totalmente automáticos, e várias formas de os classificar. Uma das classificações é quanto ao feixe de luz, que pode ser fixo ou móvel. Para uma boa leitura é necessário que o

feixe percorra todo o código de barras e em alguns equipamentos este pode ser estacionário, sendo necessário posicionar o código de barras correctamente ou a presença de um operador para manusear o leitor (Figura 2.14). Noutros equipamentos, o feixe de luz oscila e cria uma área de maior alcance que facilita leitura, como já foi referido na Figura 2.13.

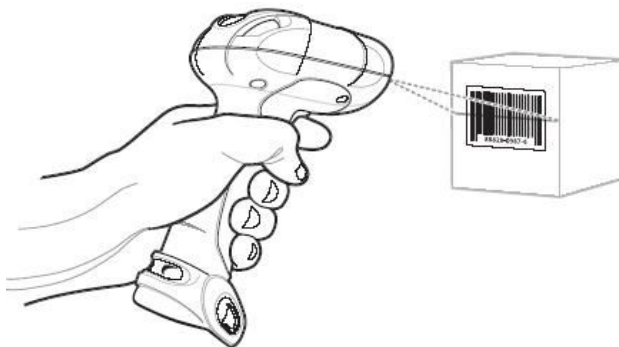


Figura 2.14 - Leitor de códigos de barras portátil com feixe de luz fixo.

Outra das classificações utilizadas distingue leitores portáteis de leitores fixos. Um exemplo de leitor portátil é o da Figura 2.14, muito utilizado em bibliotecas. Um leitor fixo (Figura 2.15) é normalmente colocado em posições não deslocáveis direccionado para a zona onde irão passar os códigos de barras. É bastante utilizado em supermercados e na indústria em tapetes rolantes.



Figura 2.15 - Leitor de códigos de barra fixo.

Ainda relativamente à classificação de leitores de códigos de barras, estes ainda podem ser com ou sem contacto. Até agora os exemplos sempre mostraram leitores sem contacto. Os leitores por contacto consistem em canetas (*wands* ou *light pens*) que funcionam movendo-as sobre o código de barras. Esta tecnologia necessita de contacto ou de grande proximidade, o que provoca danos nos rótulos após muitas utilizações.

Quanto à interface de comunicações neste tipo de equipamentos normalmente possuem saídas RS-232 (comunicação série) e de teclado (PS/2). No entanto existem alguns leitores sem fios como é o caso da Figura 2.16.



Figura 2.16 - Leitor de códigos de barras sem fios.

Código de barras bidimensional

O primeiro código de barras bidimensional surgiu em 1987. Desde então, já surgiram várias simbologias 2-D e é de esperar que surjam mais. A grande vantagem relativamente aos códigos de barras lineares é a capacidade de codificar um maior número de dados através de uma maior densidade. No entanto são necessários leitores especiais para ler os códigos 2-D e o equipamento associado a esta tecnologia é mais caro.

O código de barras bidimensional pode ser dividido em duas categorias, *stacked* e *matrix*. Códigos do tipo *stacked* consistem em multi-linhas de códigos de barras lineares agrupadas verticalmente. Exemplos de duas simbologias 2-D são o Code 49 e o Code 16K, que resultam das simbologias 1-D Code 39 e Code 128, respectivamente. O PDF417, que surgiu em 1990 [4], adicionou alguns recursos que permitiram o aumento da capacidade de dados, o aumento da densidade de dados e o aumento da fiabilidade da leitura. Na figura seguinte podem ser vistos exemplos destes códigos.

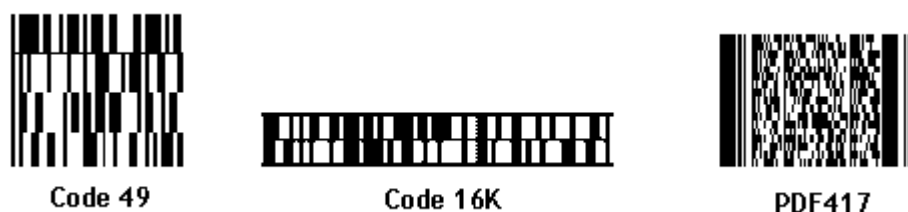


Figura 2.17 - Códigos de barras 2-D do tipo *stacked*.

Os códigos de barras bidimensionais do tipo *matrix* oferecem maior densidade de dados que os anteriores e por isso contêm mais informação. Estes consistem em padrões de células com uma forma quadrada, hexagonal ou circular de cores escuras (normalmente preto) e claras (normalmente branco). Esta simbologia tem a desvantagem de ser mais complexa e por

isso necessitar de equipamento mais dispendioso para a leitura e impressão dos códigos de barras. Exemplos destas simbologias podem ser vistos na Figura 2.18.



Figura 2.18 - Códigos de barras 2-D do tipo *matrix*.

2.3 Identificação por Rádio Frequência - RFID

Uma das primeiras aplicações da tecnologia de Identificação por Rádio Frequência surgiu nos anos quarenta. *Transponders*, que são dispositivos que recebem e enviam sinais, eram utilizados em aviões de guerra para detectar se era inimigo ou não (IFF). A tecnologia de RFID que acaba por nascer de uma junção de outras duas, o radar e as ondas de rádio (RF), continua sofrer desenvolvimentos e em 1973 surge a primeira patente de um *transponder* activo com memória de RFID [5]. Nos anos oitenta a sua comercialização iniciou-se e uma das primeiras aplicações foi na identificação de gado. O seu constante desenvolvimento levou com que hoje em dia a tecnologia se aplique em variadíssimas áreas, e cada vez mais surge como uma alternativa aos códigos de barras. Uma comparação entre as duas tecnologias pode ser vista na Tabela 2.2.

Tabela 2.2 - Comparação entre o código de barras e o RFID [1], [6].

Comparação	Código de barras linear	RFID
Tecnologia	Óptica	Rádio frequência
Capacidade de leitura e escrita	Apenas de Leitura	Leitura e escrita
Taxa de Leitura	Baixa (1 leitura de cada vez)	Elevada (Múltiplas leituras de uma só vez)
Capacidade de memória	Baixa (dezenas de caracteres)	Elevada (até milhares de caracteres)
Linha de visão	Necessária	Não necessita
Reutilização	Não reutilizável	Reutilizável
Custo rótulo/tag	Muito baixo	Aproximadamente 10 vezes mais caro que o código de barras

Durabilidade	Baixa (facilmente danificado devido à sujidade e aos riscos)	Alta (pode ser utilizado em ambientes bastante hostis, permite ser aplicado internamente aos produtos)
---------------------	--	--

Princípio de funcionamento

A tecnologia RFID consiste num leitor (Figura 2.20) que detecta a presença de um *transponder* (Figura 2.19), também designado por *tag* (etiqueta), com um determinado alcance e permite ler ou escrever dados neste último. A comunicação entre os dois é efectuada sem fios e por rádio frequência.



Figura 2.19 - Exemplos de *transponders* de RFID.



Figura 2.20 - Exemplos de leitores RFID.

A função principal do *transponder* é guardar informações sobre o objecto em que está colocado. É constituído por um circuito integrado, onde são armazenados os dados, uma antena, utilizada não só para comunicação mas também para a alimentação, e o encapsulamento, que como se vê na Figura 2.19, pode ser feito de vários materiais e ter várias formas. Os *transponders* podem ser classificados em duas categorias relativamente à sua alimentação, activos ou passivos. Etiquetas activas possuem uma bateria interna e são tipicamente de leitura e escrita. Não precisam de energia do leitor para comunicar e permitem um alcance de leitura maior que as passivas (até 30 metros [7]). Por necessitarem de bateria para o seu funcionamento têm a desvantagem do tempo de vida ser limitado (aproximadamente 10 anos), mesmo utilizando modos de poupança energia, *sleep mode* e

circuitos integrados de muito baixa potência. Dentro da classe de etiquetas activas, é comum designar uma subclasse de etiquetas semi-activas. Estas, ao contrário das anteriores, aproveitam a energia do leitor para comunicarem apresentando menor consumo. As etiquetas passivas não possuem qualquer alimentação interna, utilizando a antena e o campo electromagnético criado pelo leitor para única e exclusiva fonte de energia. O seu alcance é por isso muito mais curto e exigem leitores com maior potência. Por outro lado, o seu tempo de vida é quase ilimitado, são mais baratas e podem ser de menores dimensões.

A função principal do leitor RFID é ler e escrever dados no *transponder*. Tal como uma etiqueta, também tem uma antena que pode ser interna ou externa. A antena do leitor não funciona apenas como módulo de transmissão de dados, necessita de transferir energia às etiquetas (passivas e semi-activas) para estas funcionarem. Isto é conseguido através de um campo electromagnético criado pela própria antena que induz uma corrente na antena do *transponder* (lei de Faraday), Figura 2.21.

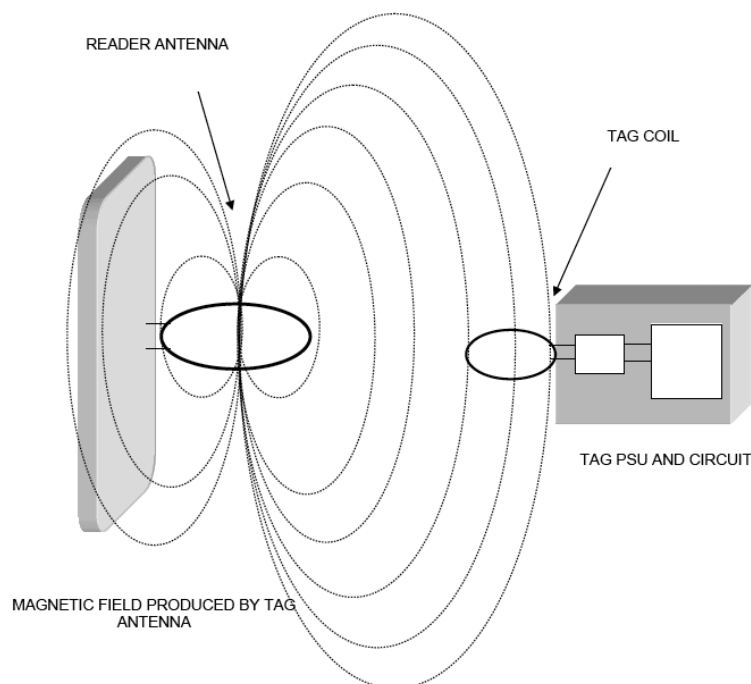


Figura 2.21 - Princípio de funcionamento do RFID [8].

Embora isto descreva o funcionamento geral, existem diferentes modos de um leitor e uma etiqueta comunicarem [9].

Começando pelos *transponders* de 1 único bit, que são muito utilizados em vigilância electrónica de artigos (EAS), a comunicação pode ser feita pelo menos de 6 técnicas diferentes: rádio frequência, microondas, divisor de frequência, electromagnética e acústico-magnética. Estas técnicas baseiam-se em simples efeitos físicos e propriedades dos materiais e não há troca de informação. As etiquetas de um 1 bit são as normalmente utilizadas nos supermercados para prevenir roubos. O seu funcionamento consiste na activação ou desactivação do bit, ou seja, se o bit estiver activo o leitor irá detectar a etiqueta, caso contrário não será detectada.

Transponders com maior capacidade de memória funcionam de maneira diferente, pois é necessária troca de dados. Esta troca de dados pode ser feita de 2 modos, *half duplex* ou *full*

duplex. No primeiro, a informação transmitida da etiqueta para o leitor é feita após a transferência de informação do leitor para a etiqueta. No segundo, a troca de dados é simultânea e isto implica que a informação transmitida da etiqueta seja feita numa frequência diferente do leitor. Para estes dois modos de comunicação, as tecnologias usadas são o acoplamento indutivo, a retro-propagação electromagnética, acoplamento fechado e acoplamento eléctrico. As mais comuns são o acoplamento indutivo, utilizado sobretudo para etiquetas passivas e a retro-propagação electromagnética, utilizada para distâncias superiores a 1 metro e frequências altas.

Frequências de comunicação

A tecnologia RFID suporta várias frequências de comunicação entre as etiquetas e os leitores. Assim, os sistemas RFID podem ser classificados pelas frequências a que operam. Na Tabela 2.3 estão descritas algumas características para sistemas de *Low*, *High*, *Ultra High* e *Super High Frequency*.

Tabela 2.3 - Classificação dos sistemas RFID quanto à frequência de comunicação [6].

	LF (< 135 kHz)	HF (13.56 MHz)	UHF (863 to 915 MHz)	SHF (2.45 GHz)
Capacidade de dados	A partir de 64 bits para leitura até 2 kbits para leitura e escrita	Tipicamente etiquetas de leitura e escrita com 512 bits de memória (max: 8kbits particionados)	Tipicamente etiquetas de leitura e escrita com 32 bits de memória (max: 4 kbits particionados em 128 bits)	A partir de 128 bits até 32 kbits particionados
Transferência de dados	Baixa: menos de 1 kbits / s (~ 200 bits / s)	Cerca de 25 kbits / s em geral (existe em 100 kbits / s)	Cerca de 28 kbits / s	Geralmente <100 kbits / s, mas pode ir até 1 Mbits / s
Distância de leitura	Tipicamente desde o contacto até 0,5 metros para etiquetas passivas, caso contrário, 2 metros aproximadamente	Até 1 metro para etiquetas passivas	Até 1 metro para etiquetas passivas	Algumas dezenas de centímetros para etiquetas passivas e algumas dezenas de metros para as activas
Modo de leitura	Leitura única e múltiplas leituras	Leitura única e múltiplas leituras	Leitura única e múltiplas leituras	Leitura única e múltiplas leituras
Ruído	Insensíveis às perturbações electromagnéticas industriais	Ligeiramente sensíveis às perturbações electromagnéticas industriais	Sensíveis às perturbações electromagnéticas. Podem ser perturbados por outros sistemas UHF próximos	Altamente sensíveis a perturbações electromagnéticas reflectidas pelo metal e absorvidas pela água

Aplicações	Processo de produção, identificação de veículos e contentores, controlo de acessos e identificação animal	Monitorização, estacionamento automóvel, Bagagens, livrarias, Serviços de aluguer, logística	Monitorização, estacionamento automóvel, logística	Controlo de acessos, logística militar, portagens
------------	---	--	--	---

2.4 Cartões de Banda Magnética

A primeira utilização das bandas magnéticas em cartões aconteceu nos anos setenta no metro em Londres. No fim desta década foi implementado nos Estados Unidos (*Bay Area Rapid Transit*) um sistema de bilhetes em papel, com o mesmo tamanho dos cartões de crédito utilizados actualmente. Este sistema usava a informação guardada na banda magnética que era lida e escrita novamente, sempre que o bilhete era utilizado. Os cartões de crédito embora criados em 1951, só adoptaram a banda magnética após o estabelecimento de normas em 1970. Hoje em dia, os cartões financeiros constituem a maior parte de utilização dos cartões de banda magnética e seguem normas ISO para assegurar a sua interoperabilidade.

Com o desenvolvimento de novas tecnologias, muitas pessoas previram o fim dos cartões de banda magnética, no entanto isso não acontecerá, pelo menos para já. Esta tecnologia continua a ser a melhor solução em algumas aplicações, pois é muito barata e flexível. A normalização de cartões de alta coercividade para mercados financeiros e o desenvolvimento de técnicas de segurança constituem também outras vantagens para este tipo de sistemas [4].

Princípio de funcionamento

O princípio para leitura e escrita em bandas magnéticas baseia-se nas propriedades dos materiais ferromagnéticos. Estes materiais têm a capacidade de reter o magnetismo mesmo depois do campo magnetizante externo ser removido. As partes elementares de uma banda magnética são partículas ferromagnéticas com cerca de 500 nanómetros aproximadamente e cada uma actua como um pequeno íman. Estas partículas são mantidas juntas por um material especial (*resin binder*). Quando é feita a película, as partículas elementares são orientadas com os eixos Norte-Sul paralelos à banda através de um campo magnético externo, enquanto a *resin binder* as fixa.

A coercividade é uma propriedade importante nesta tecnologia. Como foi dito, cada partícula funciona como uma barra magnética com dois pólos estáveis. Mas se for sujeita a um campo magnético externo suficientemente forte com polaridade contrária, os pólos podem inverter, isto é, o pólo Norte torna-se o pólo Sul e vice-versa. A força do campo magnético necessária para isso acontecer chama-se força coerciva ou coercividade da partícula.

Uma banda magnética por codificar funciona como um único íman com um pólo em cada extremidade, o que na verdade são uma série de partículas magnetizadas todas no mesmo sentido (Norte-Sul-Norte-Sul-...). No entanto, se uma partícula for magnetizada inversamente

irá criar uma junção Sul-Sul e os fluxos irão repelir-se, criando uma concentração de linhas de fluxo nessa mesma junção (o mesmo acontece para os pólos Norte). Estas junções são chamadas de *flux reversals*. Resumindo, a codificação consiste em criar *flux reversals* e a leitura consiste em detectar os mesmos.

O campo magnético externo utilizado para trocar os pólos das partículas é produzido por um solenóide que pode inverter a sua própria polaridade trocando o sentido da corrente. O movimento da banda magnética em conjunto com a inversão da polaridade do solenóide cria os *flux reversals*. Para apagar a informação da banda magnética apenas é necessário manter o campo do solenóide constante. Para leitura o processo é semelhante, mas de modo inverso. O magnetismo das partículas induz uma corrente no solenóide, que origina picos nos *flux reversals*, pois aí o campo é mais forte.

Quanto ao modo como os dados são codificados a técnica mais comum é conhecida como *Aiken Biphase* (F2F). Esta técnica consiste em codificar o valor lógico 1 ao dobro da frequência do 0. Na Figura 2.22 pode ser vista a sequência de leitura e decodificação de uma banda magnética.

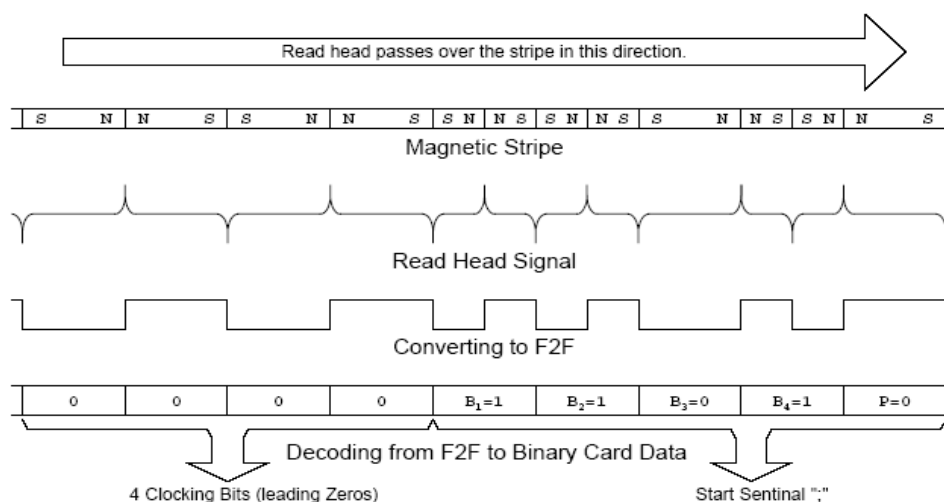


Figura 2.22 - Exemplo de uma sequência de leitura e decodificação de uma banda magnética [10].

Cartões

Os cartões de banda magnética estão sujeitos a normas que definem características físicas do cartão e características da banda magnética. No que diz respeito ao primeiro ponto, todos os cartões tem as seguintes dimensões: 85.725 milímetros de comprimento, 55.245 milímetros de largura e 0.762 milímetros de espessura. Quanto às bandas magnéticas possuem 3 pistas de dados independentes e estão definidas medidas para cada uma delas (2.794 milímetros). A distância entre a banda e a extremidade do cartão também é definida (5.664 milímetros), Figura 2.23.

	0.223"	Track		Recording Density (bits per inch)	Character Configuration (including parity bit)	Information Content (including control characters)
	0.110"	1	IATA	210	7 bits per character	79 alphanumeric characters
	0.110"	2	ABA	75	5 bits per character	40 numeric characters
	0.110"	3	THRIFT	210	5 bits per character	107 numeric characters

Figura 2.23 - Características da banda magnética de um cartão.

A pista 1 foi a primeira a ser normalizada e foi desenvolvida pela *International Air Transport Association* (IATA). Tem uma densidade de dados igual a 82 bits por centímetro, capacidade para 79 caracteres alfanuméricos e é apenas de leitura. Na Figura 2.24 está representado um exemplo do conteúdo da primeira pista de um cartão financeiro.

76 ALPHANUMERIC DATA CHARACTERS									
SS	FC	PAN	FS	NAME	FS	ADDITIONAL DATA	DISCRETIONARY DATA	ES	LRC
		Primary Account No. (19 digits Max.)		Name (26 alphanumeric characters Max.)		No. of characters Expiration date (YYMM) 4 Service Code 3	No. of characters * PVKI 1 * PVV or Offset 4 * CVV or * CVC 3 Some or all of the above fields may be found with the discretionary data		
SS	Start Sentinel	%	FC	Format Code					
FS	Field Separator	^	LRC	Longitudinal Redundancy Check character					
ES	End Sentinel	?							

Figura 2.24 - Pista 1 de um cartão financeiro.

A pista 2 foi desenvolvida pela *American Bankers Association* (ABA) para transacções financeiras *on-line*. Tem uma densidade de dados igual a 29 bits por centímetro, capacidade para 40 caracteres numéricos e é apenas de leitura. Na Figura 2.25 está representado um exemplo do conteúdo da segunda pista de um cartão financeiro.

37 NUMERIC DATA CHARACTERS									
SS	PAN	FS	ADDITIONAL DATA	DISCRETIONARY DATA	ES	LRC			
	Primary Account No. (19 digits Max.)		No. of characters Expiration date (YYMM) 4 Service Code 3	No. of characters * PVKI 1 * PVV or Offset 4 * CVV or * CVC 3 Some or all of the above fields may be found with the discretionary data					
SS	Start Sentinel	Hex B ;	LRC	Longitudinal Redundancy Check character					
FS	Field Separator	Hex D =							
ES	End Sentinel	Hex F ?							

* White boxes identify control characters

Figura 2.25 - Pista 2 de um cartão financeiro.

A pista 3 também é utilizada para transacções financeiras. A grande diferença é que pode ser de leitura e escrita. Tem uma densidade de dados igual a 82 bits por centímetro e

capacidade para 107 caracteres numéricos. Na Figura 2.26 está representado um exemplo do conteúdo da terceira pista de um cartão financeiro. Em alguns sistemas raros em que o PIN é guardado no cartão, será esta a pista onde será guardado.

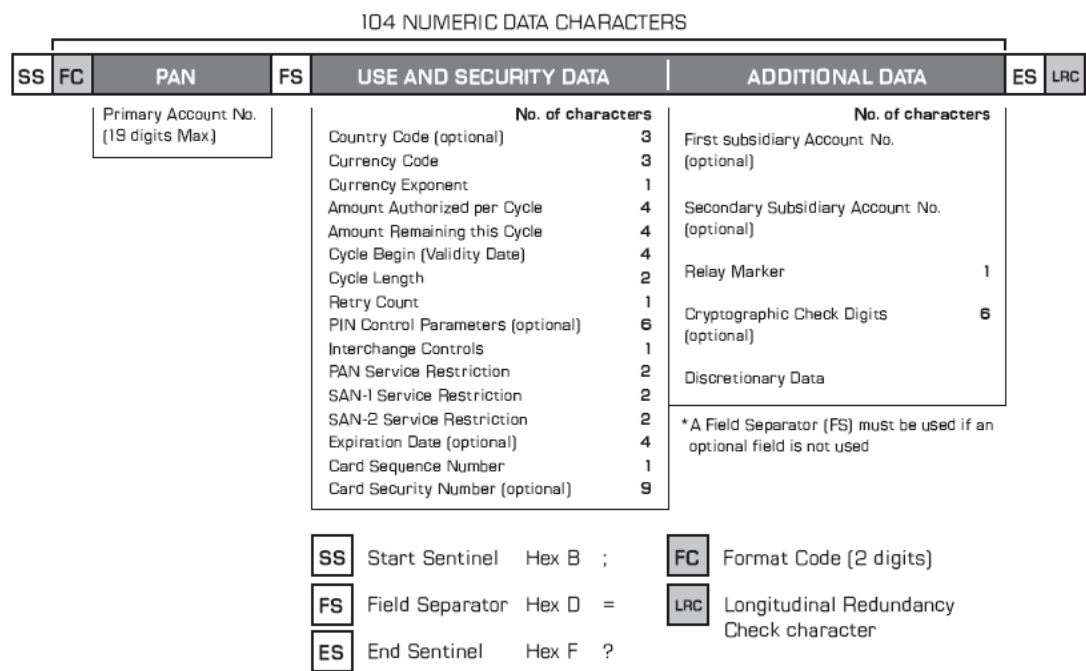


Figura 2.26 - Pista 3 de um cartão financeiro.

Leitores e gravadores de cartões

A função do leitor e do gravador de cartões consiste em ler e escrever na banda magnética (nem todos suportam as 3 pistas), utilizando o princípio de funcionamento descrito anteriormente. As bandas magnéticas podem ser de alta ou baixa coercividade, o que não influencia os leitores de cartões. No entanto, os gravadores de alta coercividade só podem ser usados em cartões de alta coercividade e vice-versa. Alguns exemplos de leitores e gravadores podem ser vistos na figura que se segue.



Figura 2.27 - Exemplos de leitores e gravadores de cartões de banda magnética.

2.5 Outras Tecnologias

Reconhecimento Óptico de Caracteres (OCR)

O reconhecimento óptico de caracteres consiste numa máquina capaz de ler e identificar um carácter através de um mecanismo óptico. Antes dos códigos de barras terem sido adoptados por uma grande parte da indústria, muitas aplicações comerciais utilizavam este tipo de tecnologia. Em 1975, a *National Retail Merchants Association* (NRMA) dos Estados Unidos chega mesmo a normalizar o OCR como a tecnologia de identificação automática. No entanto, devido a problemas associados a esta tecnologia, tais como baixa taxa de leitura, alta taxa de erro de substituição, inexistência de leitores omnidireccionais automáticos e o constante crescimento e desenvolvimento de código de barras são forçados a rever a norma e a mudar de OCR para códigos de barras.

O seu funcionamento baseia-se numa fonte de luz que cria uma imagem e posteriormente é interpretada por um *software* de reconhecimento. O *software* usa duas técnicas para identificar os caracteres: *template matching*, que consiste na comparação com uma base de dados e *feature extraction*, que consiste na análise de elementos da estrutura do carácter.

A grande vantagem desta tecnologia é a possibilidade de leitura tanto das máquinas como das pessoas. Por outro lado, apresenta grandes desvantagens como necessidade de contacto ou distâncias muito curtas, baixa taxa de leitura e elevada taxa de erros. Deste modo, as aplicações desta tecnologia são mais comuns em sistemas de alto nível, onde o *software* de reconhecimento é mais complexo e com maior capacidade de processamento (redes neuronais por exemplo).



Figura 2.28 - Exemplos de leitores OCR.

Visão Industrial

Os sistemas de visão industriais consistem na captura de uma imagem visual (câmara de vídeo) e através de um processo de análise, reconhecer marcas, formas, caracteres, estruturas de códigos ou outras propriedades da imagem predefinidas. As principais aplicações desta tecnologia, utilizadas desde há alguns anos atrás, são o controlo de qualidade, medições, montagem robótica nas indústrias automóvel, electrónica, aeroespacial e metalúrgica, entre outras. Apenas recentemente foram classificadas como tecnologias AIDC devido à sua capacidade de ler códigos de barras lineares e bidimensionais. Oferecem algumas vantagens relativamente ao leitor comum de código de barras, pois obtém uma imagem completa do código e do meio envolvente (leitura com baixo contraste por exemplo).



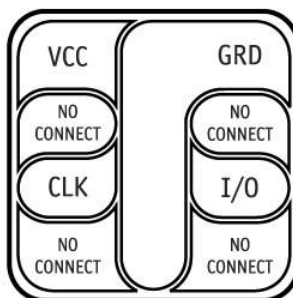
Figura 2.29 - Exemplos da tecnologia de visão industrial.

Cartões Inteligentes (*Smart Cards*)

Os cartões inteligentes consistem em cartões, de características físicas definidas pela mesma norma que a dos cartões de banda magnética, com um circuito integrado (IC) embutido. Isso confere a capacidade de processar e armazenar dados. O IC consiste num microprocessador com uma interface SPI, de alimentação externa (leitor) e com um sistema operativo (JavaCard por exemplo). A possibilidade de processamento permite guardar informação de forma segura, uma vez que a troca de dados pode ser feita recorrendo à encriptação dos mesmo. Desta forma, a sua utilização tem vindo a aumentar e sendo já os cartões telefónicos uma das maiores aplicações, tem surgido outras como por exemplo os cartões financeiros (banda magnética e IC num só cartão).



Cartões Inteligentes



Esquema dos contactos do IC



Leitores

Figura 2.30 - Exemplos da tecnologia de cartões inteligentes.

Ecrãs Tácteis

Ecrãs tácteis são monitores que permitem a inserção de dados através do toque e sem a necessidade de um teclado ou rato, de uma forma fácil e intuitiva. Podem ser classificados quanto à tecnologia utilizada na superfície sensível ao toque como resistivos capacitivos, onda acústica, infravermelhos, ópticos e de sinal dispersivo (DST). A sua robustez permite a sua utilização na indústria e estão normalmente mais relacionados com funções de controlo e supervisão.



Figura 2.31 - Exemplo de ecrãs tácteis.

Tecnologias Biométricas

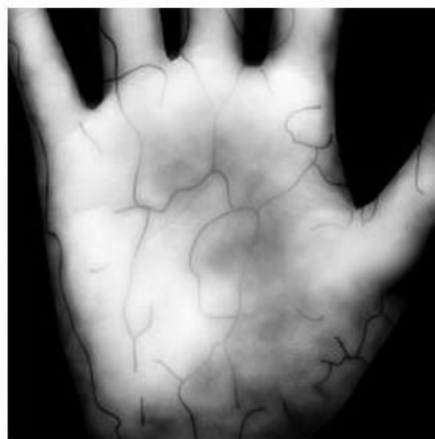
Tecnologias biométricas são métodos de reconhecimento de uma pessoa baseados nas características físicas e comportamentais. Reconhecimento de impressões digitais, face, íris, retina, geometria da mão, veias, ADN, voz e caligrafia são alguns exemplos de métodos utilizados.

Com a investigação e o desenvolvimento da tecnologia, o número de soluções de identificação e a segurança das mesmas são cada vez mais elevados. As suas aplicações não implicam apenas identificação de pessoas, mas também controlo de acessos, funções de segurança e envio de comandos. As vantagens que oferecem face a outros métodos, como palavras-chave e PIN, tem contribuído para o seu crescimento.

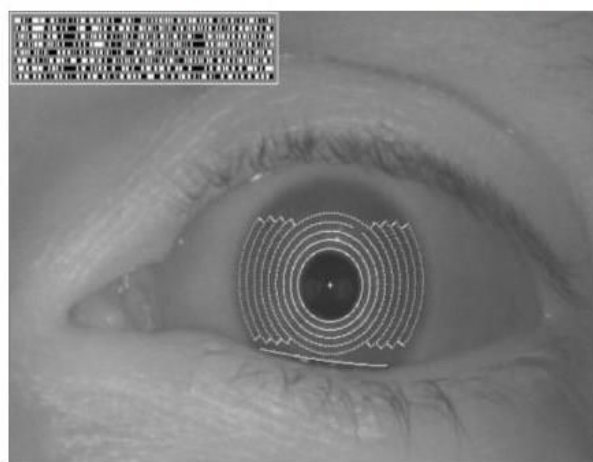
A utilização de tecnologias biométricas encontra-se em várias áreas, tais como indústria militar, governos, empresas (controlo de acessos), saúde, entre outros.



Reconhecimento geométrico da mão



Reconhecimento de veias



Reconhecimento da íris

Figura 2.32 - Exemplos de tecnologias biométricas.

Capítulo 3

Formulação do Problema

Neste capítulo será feita uma descrição da proposta de trabalho e dos equipamentos utilizados na sua realização. Inicialmente, será dada uma perspectiva geral do que se pretende e em que contexto esta surgiu, bem como os objectivos do trabalho. Seguidamente, apresentar-se-ão os equipamentos disponibilizados, as suas características e modo de funcionamento.

3.1 Linha de fabrico flexível de aprendizagem do DEEC

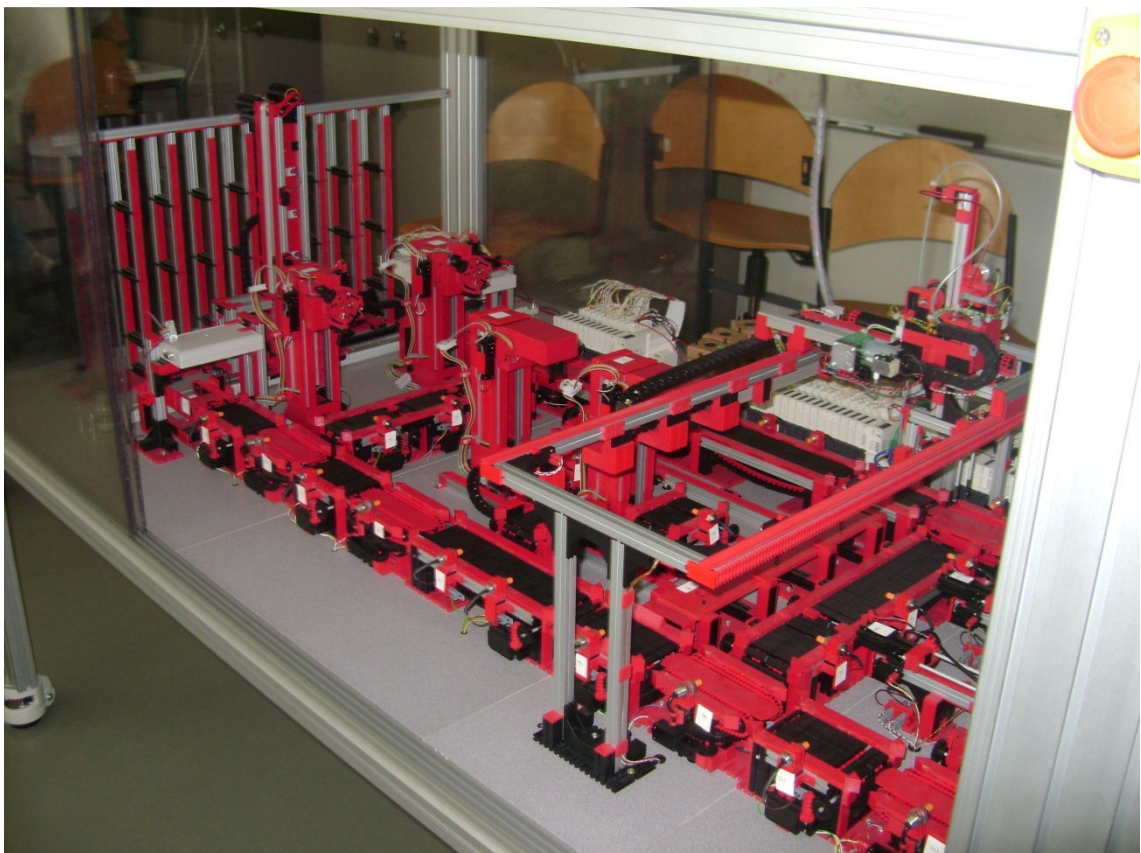
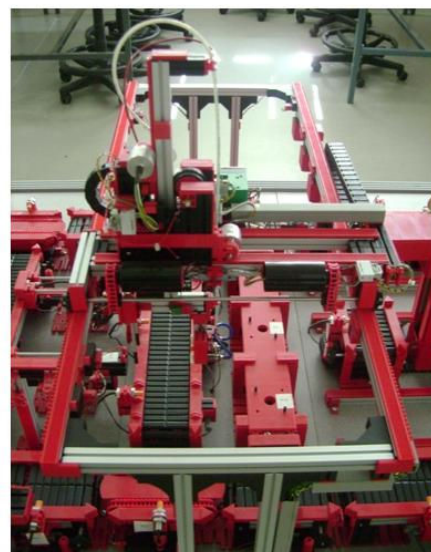


Figura 3.1 - Linha de fabrico flexível de aprendizagem do DEEC

O Departamento de Engenharia Electrotécnica e de Computadores (DEEC) da Universidade do Porto adquiriu recentemente, para um dos seus laboratórios, uma linha de fabrico flexível de aprendizagem (Figura 3.1). Com esta aquisição pretende-se disponibilizar aos alunos um meio de aprendizagem das tecnologias de automação industrial. Este sistema é constituído por um armazém, várias máquinas de ferramentas, um robot de 3 eixos e vários sistemas de transporte baseados em tapetes rolantes (Figura 3.2). As peças que circulam nesta linha possuem etiquetas RFID embebidas e existem 4 leitores para leitura e escrita das mesmas. Todos os sensores e actuadores estão ligados a ilhas locais que, por sua vez, estão ligados via Ethernet a um PLC responsável pelo controlo.



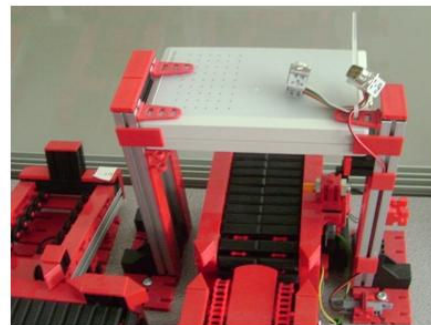
Máquinas de ferramentas



Robot de 3 eixos



Armazém automático



Leitor RFID

Figura 3.2 - Constituição da linha de fabrico flexível do DEEC

A linha de fabrico flexível destina-se unicamente para fins de aprendizagem. Os alunos tem assim, a possibilidade de se ambientarem com um sistema industrial muito próximo da realidade e aplicarem as técnicas de automação leccionadas. Como rapidamente se percebe, isto poderá originar controlos erróneos dos dispositivos da linha de fabrico flexível. Quer-se com isto dizer, que sendo alunos com pouca experiência em automação e programação de autómatos, inadvertidamente poderão enviar comandos errados para os dispositivos e provocar danos. Isto levou com que se tenha decidido implementar um sistema de *interlock*, o qual consiste em fazer a supervisão do estado da linha de fabrico e evitar que sejam enviados

comandos errados. Esta aplicação será suportada por um sistema embebido baseado em Linux (ICnova AP7000 Base) e servirá de interligação entre o autómato e as ilhas. A aplicação de *interlock* não faz parte deste trabalho, mas é importante referir para se mostrar o funcionamento geral do sistema.

A par destas especificações surge a necessidade de integrar o sistema RFID nesta solução, de forma simples para utilização dos alunos. A aplicação responsável por isso, tal como o sistema *interlock*, será suportada pelo sistema embebido. Esta aplicação servirá na realidade para disponibilizar uma interface entre o leitor RFID e o PLC. Deste modo, os alunos não necessitam de conhecer o protocolo de comunicação do leitor e apenas terão de enviar comandos para a aplicação a partir do PLC. Esses comandos incluem principalmente, ler e escrever para as etiquetas RFID.

Numa perspectiva de dar a conhecer aos alunos mais tecnologias de automação, neste caso tecnologias AIDC, este sistema irá ainda incluir um leitor de código de barras e um leitor de cartões de banda magnética. O objectivo seria dotar as peças com um código de barras e controlar o acesso ao sistema através de identificação por cartão de banda magnética (por exemplo o cartão de identificação da faculdade).

Na secção seguinte serão apresentados os objectivos deste trabalho e qual a sua finalidade.

3.2 Objectivos

Após ter sido apresentado o contexto inicial do trabalho, pode-se traçar com mais pormenor os objectivos que se pretendem obter com a realização deste projecto.

Como já foi dito, pretende-se implementar um sistema que permita aos alunos utilizar e integrar tecnologias AIDC na linha de fabrico flexível de forma simples e transparente para eles, no que diz respeito ao protocolo de comunicações. Assim, decidiu-se utilizar o protocolo Modbus TCP/IP como meio de comunicação entre o PLC e os dispositivos AIDC (do ponto de vista do aluno). Como é um protocolo simples, conhecido pelos alunos e suportado pelo PLC satisfaz os requisitos. Deste modo, o objectivo principal do trabalho é criar um servidor Modbus TCP/IP que implemente funções de leitura e escrita para o sistema RFID e funções de leitura para os leitores de códigos de barras e de cartões de banda magnética.

Como se pode constatar mais à frente, todos os dispositivos utilizados possuem uma interface de comunicação RS-232. Para ser possível integrar todos os dispositivos numa aplicação, foi disponibilizado o conversor de R-S232 para Ethernet, o XPort (Figura 4.12). Definido o modo de comunicação entre os dispositivos AIDC e a plataforma que suportará o servidor Modbus, na Figura 3.3 pode-se ver o esquema geral do que é pretendido.

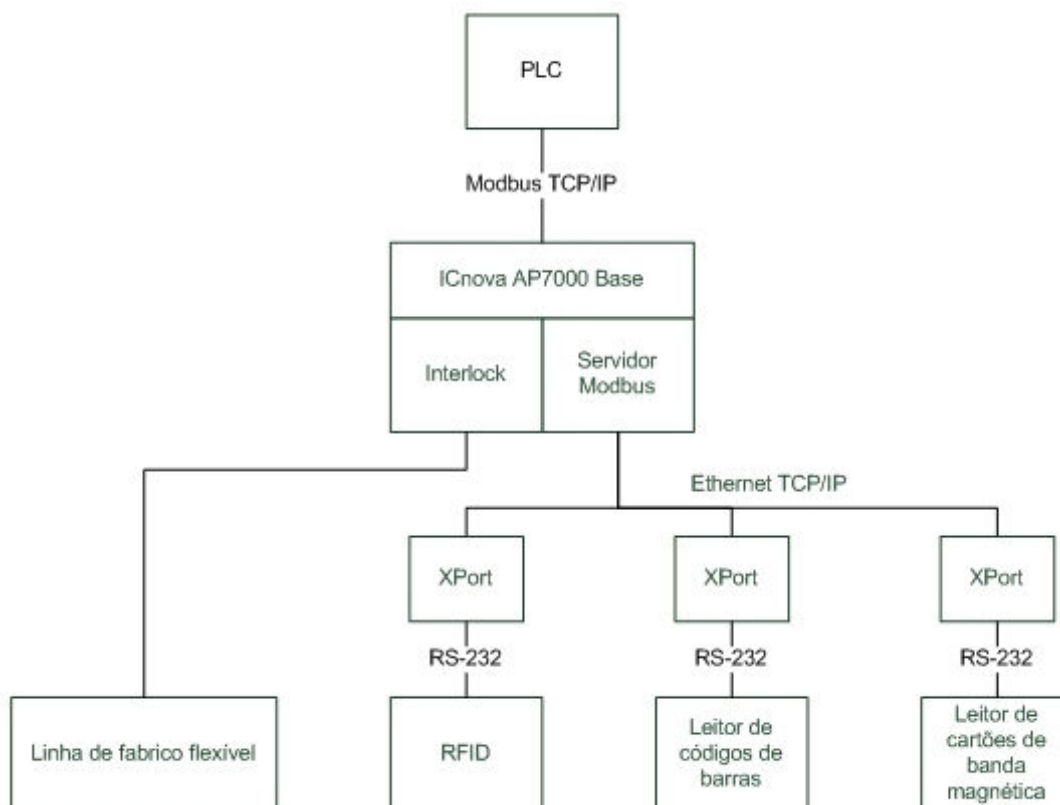


Figura 3.3 - Esquema do sistema da linha de fabrico flexível integrando os dispositivos AIDC.

Em suma, o objectivo deste projecto é criar um servidor Modbus que permita:

- Ler dados de uma etiqueta;
- Escrever dados numa etiqueta;
- Utilizar o *Scan Mode* do leitor RFID;
- Alterar algumas configurações do leitor RFID;
- Ler dados de códigos de barras;
- Ler dados de cartões de banda magnética;
- Efectuar as funções acima descritas para vários equipamentos do mesmo tipo.

Os equipamentos utilizados para este trabalho serão apresentados de seguida.

3.3 Leitor RFID

O leitor RFID disponibilizado para este projecto é da marca FEIG, modelo ID ISC.PR101-A e as suas principais características estão definidas na Tabela 3.1. A única interface visual com o utilizador é um sinal luminoso constituído por um LED vermelho e outro verde, que indica 3 estados possíveis: verde significa funcionamento normal, vermelho quando detecta uma etiqueta e laranja quando o leitor inicializa.



Figura 3.4 - Leitor RFID ID ISC.PR101-A.

Tabela 3.1 - Características do leitor RFID ID ISC.PR101-A.

Frequência	13.56 MHz
Tipo de etiqueta	Passivas
Antena	Interna
Alcance	Baixo (entre 7 a 10 cm)
Interface de comunicação	RS-232 ou RS-485
Ligação	9-pin D-sub fêmea
Alimentação	2 a 24 V CC

Quanto ao modo de funcionamento da transmissão de dados entre o leitor e o *host* (termo que será utilizado para designar o dispositivo ao qual o leitor está ligado) existem 4 modos diferentes. Os modos *ISO15693 Host Commands* e o *Scan Mode* que são utilizados para troca de dados entre a etiqueta e o *host* e os modos *Configuration Commands* e *Control Commands* que são utilizados para troca de dados com o leitor.

Configuration Commands e *Control Commands* consistem em comandos enviados pelo *host* para configuração e diagnóstico do leitor. Este, por sua vez, responde com uma confirmação ou dados pedidos à ordem que lhe foi enviada em caso de sucesso e com um erro em caso de insucesso (Figura 3.5).

Host (Terminal / PC /)		Reader	
parameter- / control command	→	parameter received and stored / control command processed	
		yes	no
	←	status / data	
	←	error status	

Figura 3.5 - Troca de dados entre o *host* e o leitor RFID utilizando *Configuration Commands* e *Control Commands*.

Relativamente à configuração (*Configuration Commands*), esta está organizada em blocos de 16 bytes cada um, sendo 2 deles para CRC. Os blocos de configuração podem ser guardados em 2 locais distintos, na memória RAM ou na memória EEPROM. No primeiro caso a configuração enviada pelo *host* é imediatamente actualizada e perde-se caso o leitor seja desligado. No segundo caso, para que a configuração fique activa é necessária uma reinicialização do leitor e não se perde após ser desligado. Os comandos disponíveis neste modo estão descritos na Tabela 3.2.

Tabela 3.2 - Funções disponíveis no modo de configuração.

Read Configuration	Lê um bloco de configuração (RAM ou EEPROM)
Write Configuration	Envia um bloco de configuração (RAM ou EEPROM)
Save Configuration	Guarda 1 ou todos os blocos de configuração da memória RAM na EEPROM
Set Default Configuration	Coloca 1 ou todos os blocos de configuração com os valores de origem (RAM ou EEPROM)

Relativamente ao modo de controlo (*Control Commands*), encontra-se na tabela seguinte a lista de funções disponíveis.

Tabela 3.3 - Funções disponíveis no modo de controlo.

Baud Rate Detection	Determina a velocidade de comunicação (RS-232/RS-485)
Start Flash Loader	Inicia a actualização do <i>firmware</i>
CPU Reset	Reinicializa o CPU do leitor
Get Software Version	Determina a versão do <i>software</i> do leitor, o seu tipo e o tipo de etiquetas que são suportadas pelo <i>software</i>
Get Reader Info	Determina a versão do <i>firmware</i> do leitor, o seu tipo e o tipo de etiquetas que são suportadas pelo <i>firmware</i>
RF Reset	A antena é desligada por 15 milissegundos
RF ON/OFF	Liga ou desliga a antena
Set Output	Controla o LED

ISO15693 Host Commands consiste num conjunto de comandos que permitem a troca de dados entre o *host* e as etiquetas através do leitor. Esta troca de dados pode ser feita de 3 formas diferentes:

- *Addressed Mode*. Antes de ler ou escrever numa etiqueta é necessário conhecer o seu UID. Para isso é enviado um comando *Inventory* (Tabela 3.4) e se uma etiqueta estiver ao alcance do leitor retorna o seu UID. Todas as funções de leitura e escrita seguintes para essa etiqueta têm de incluir o UID.

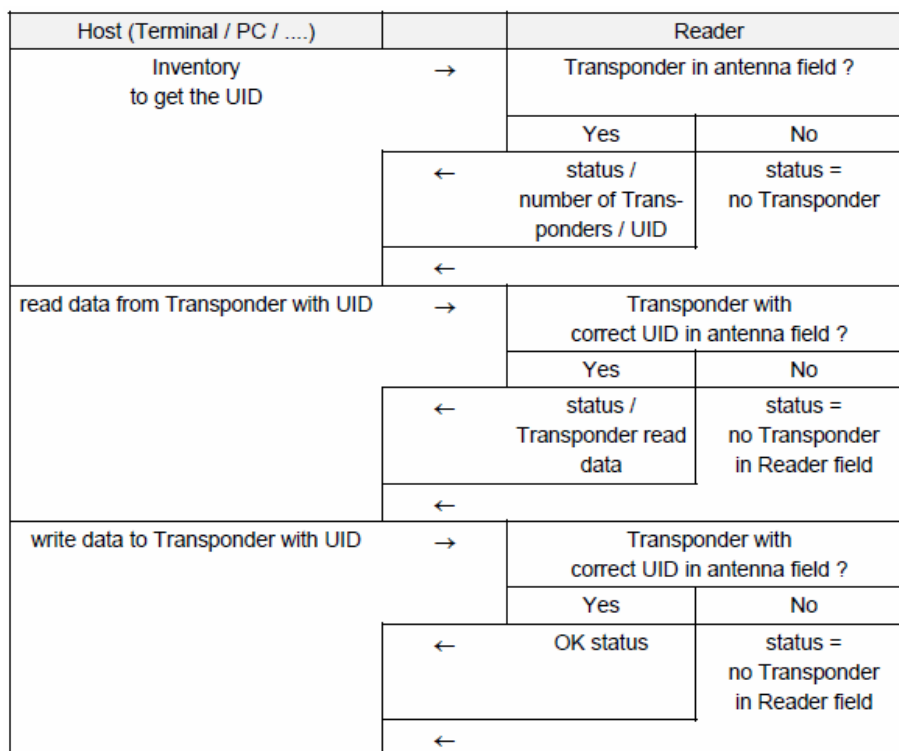


Figura 3.6 - Troca de dados entre o *host* e o leitor RFID no modo *ISSO 15693 Host Commands: Addressed Mode*.

- *Non-addressed Mode*. Neste modo não é necessário conhecer o UID da etiqueta e é útil quando há apenas uma ao alcance do leitor.

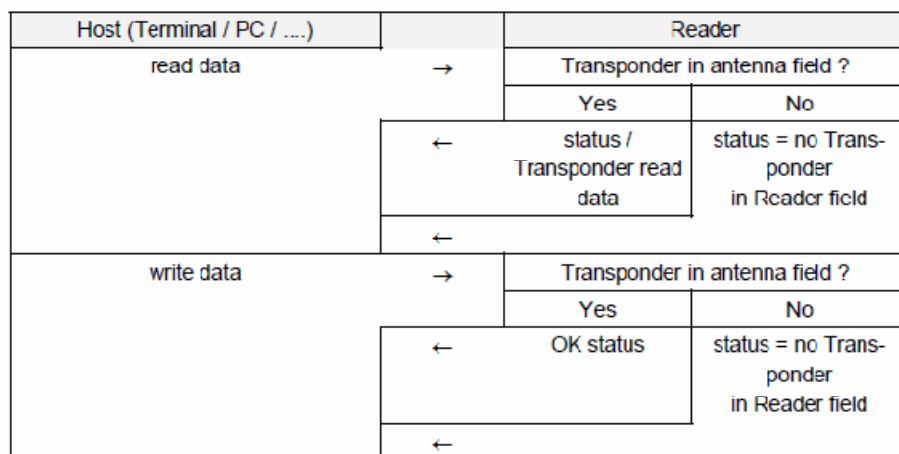


Figura 3.7 - Troca de dados entre o *host* e o leitor RFID no modo *ISSO 15693 Host Commands: Non-Addressed Mode*.

- *Selected Mode*. Tal como no *Addressed Mode* é necessário enviar o comando *Inventory* para conhecer o UID da etiqueta. Posteriormente é enviado o comando *Select* (Tabela 3.4) com esse UID. Assim, a leitura e escrita para essa etiqueta que se realize a seguir não necessita de incluir o UID.

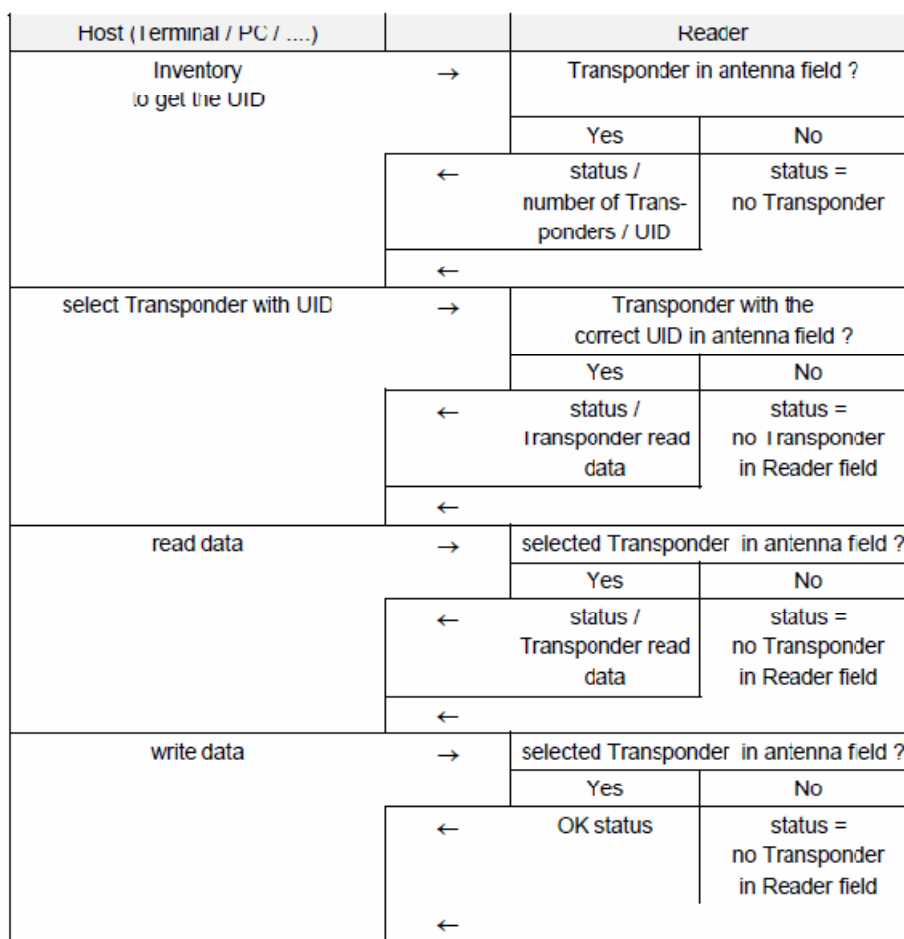


Figura 3.8 - Troca de dados entre o host e o leitor RFID no modo ISO 15693 Host Commands: Selected Mode.

O conjunto de comandos incluídos em *ISO15693 Host Commands* que o leitor RFID oferece depende do tipo de etiqueta utilizada. Na Tabela 3.4 estão listadas as funções mais importantes e que são suportadas pelo tipo de *Transponder* utilizado neste projecto.

Tabela 3.4 - Funções disponíveis para o modo *ISO 15693 Host Commands*.

Inventory	Retorna todos os UID das etiquetas ao alcance do leitor
Stay Quiet	Coloca a etiqueta em modo silencioso
Lock Multiple Blocks	Bloqueia um ou mais blocos de dados (contra escrita)
Read Multiple Blocks	Lê um ou mais blocos de dados
Write Multiple Blocks	Escreve num ou mais blocos de dados
Select	Selecciona a etiqueta (coloca em <i>Select State</i>)
Reset to Ready	Coloca a etiqueta em <i>Ready State</i>
Get System Information	Lê informação do sistema da etiqueta
Get Multiple Block Security Status	Lê o estado de segurança dos blocos de dados (se estão ou não protegidos contra escrita)

Por último, o *Scan Mode* consiste num modo automático de envio de dados da etiqueta para o *host* através do leitor RFID, ou seja, quando o *Transponder* fica ao alcance do leitor

envia de imediato informação. Essa informação pode ser configurada quanto aos blocos de dados a enviar, UID, entre outras. A informação só será enviada do leitor RFID para o *host* apenas quando todas as etiquetas detectadas tiverem enviado os dados respectivos a cada uma (para o leitor).

No que diz respeito ao *Transponder* disponibilizado para este trabalho, não foi fornecida qualquer folha de características nem indicada marca ou modelo (Figura 3.9). No entanto através de algumas funções do leitor RFID descritas anteriormente, conclui-se que a etiqueta, muito provavelmente, é o modelo Tag-it HF-I Plus da Texas Instruments. Esta etiqueta tem capacidade de dados de 256 bytes particionados em blocos de 4 bytes, ou seja, 64 blocos de dados.

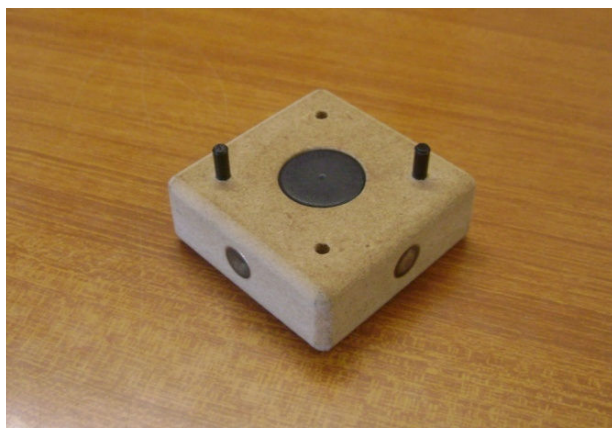


Figura 3.9- Peça da linha de fabrico flexível com a etiqueta RFID embutida.

3.4 Leitor de códigos de barras

O leitor de código de barras disponibilizado para este projecto é o scanner manual CCD 1021Plus da marca CipherLab, mostrado na Figura 3.10. É alimentado a 5 V CC e possui interface de comunicação RS-232, embora também suporte PS/2 e *Wand Emulation*. Como interface com o utilizador, contem um sinal sonoro e um luminoso, que indicam o estado do leitor, leituras bem sucedidas e erros, e um botão utilizado para efectuar leituras.



Figura 3.10 - Scanner CCD 1021Plus

Este scanner possui 8 modos de funcionamento diferentes:

- *Auto Off Mode*. Neste modo e após ser pressionado o botão, o scanner inicia a captura e continua até que um código de barras tenha sido lido ou expirado um determinado período de tempo (*timeout*).
- *Continuous Mode*. O scanner está em permanente captura.
- *Auto Power Off Mode*. Tal como o *Auto Off Mode* inicia a captura após o botão ter sido pressionado, mas apenas termina quando expirar o *timeout*. Durante esse período de tempo pode efectuar mais de que uma leitura.
- *Alternate Mode*. A captura inicia quando o botão é pressionado e termina quando for novamente pressionado.
- *Momentary Mode*. A captura mantém-se enquanto o botão estiver pressionado.
- *Repeat Mode*. Este modo de funcionamento é semelhante ao *Continuous Mode*, mas o botão tem a função de retransmissão. Isto é, após uma leitura, se o botão for pressionado num intervalo de 1 segundo, os dados serão novamente transmitidos tantas vezes quantas vezes for pressionado o botão. É particularmente útil quando se pretende ler o mesmo código de barras mais de uma vez.
- *Laser Mode*. O scanner inicia a captura depois de o botão ser pressionado e termina quando for feita uma leitura, o botão largado ou o *timeout* expirar.
- *Test Mode*. O scanner está em permanente captura, mas ao contrário do *Continuous Mode*, descodifica os códigos de barras repetidamente, ou seja, pode ler o mesmo código várias vezes.

Quanto aos códigos de barras, o scanner 1021Plus permite ler códigos de barras negativos (barras representadas com cores claras e espaços com cores escuras) e várias simbologias. Alguns parâmetros das simbologias suportadas pelo leitor podem ser configuradas, as quais incluem:

- | | |
|---------------------|------------|
| • Code 39 | • UPCE |
| • Italy Pharmacode | • EAN8 |
| • French Pharmacode | • EAN13 |
| • Industrial 25 | • Code 93 |
| • Interleave 25 | • Code 128 |
| • Matrix 25 | • EAN 128 |
| • Codabar (NW-7) | • MSI |
| • UPCA | • Plessey |

Depois de descodificados os dados após uma leitura, o leitor permite configurar a trama de dados que irá ser enviada para o *host*. Desde substituição de caracteres, inclusão de caracteres de início e fim de trama, identificação da simbologia, tamanho da trama até criação de diferentes campos de dados, podem ser configurados consoante os requisitos da aplicação.

Todas as configurações do scanner são realizadas através da leitura de códigos de barras, disponibilizados no manual.

3.5 Leitor de cartões de banda magnética

O leitor de cartões de banda magnética disponibilizado para este projecto é o modelo Cipher 1023 da marca CipherLab, capaz de ler as 3 pistas (Figura 3.11). É alimentado, neste caso, a 24 V CC e possui interface de comunicação RS-232, embora também suporte PS/2. Como interface com o utilizador, contém um sinal sonoro e um luminoso, que indicam os estado de operação do leitor e leituras bem sucedidas.

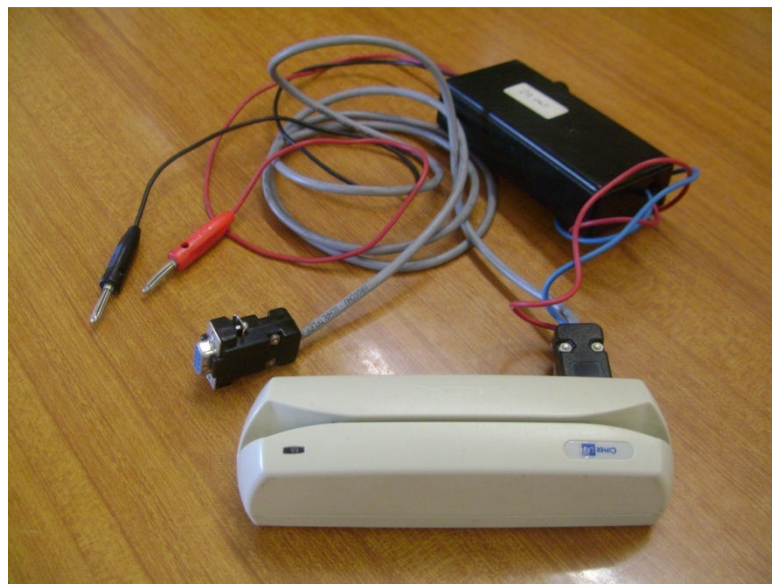


Figura 3.11 - Leitor de cartões de banda magnética Cipher 1023.

A configuração da interface de comunicação é feita através de 9 interruptores colocados na parte de trás do leitor e apenas possuem 2 posições (*on/off*). Pode-se alterar parâmetros tal como, *baud rate*, paridade, entre outros, e sempre que se alterar essa configuração é necessário reiniciar o leitor.

Semelhante ao leitor de código de barras, este também permite alterar o modo como os dados são enviados. No entanto, para configurar o leitor de cartões de banda magnética é necessário um cartão especial de configuração, ao qual não foi possível ter acesso. Assim, para o desenvolvimento do projecto, foi utilizada a configuração já existente.

Capítulo 4

Solução Proposta

Neste capítulo pretende-se descrever todo o trabalho desenvolvido para atingir os objectivos propostos. Para isso foi criada uma aplicação com a estrutura representada na Figura 4.1.

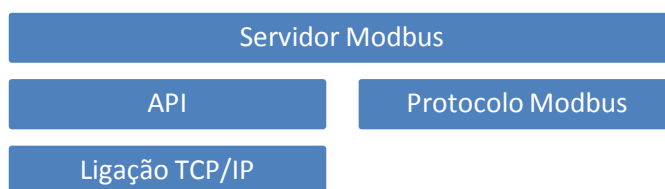


Figura 4.1 - Diagrama da aplicação implementada.

Em primeiro lugar, será abordado o estudo do protocolo de comunicação do leitor RFID e a ligação TCP/IP entre um leitor e o XPort. De seguida, será apresentada uma API (*Application Programming Interface*) desenvolvida para a comunicação com os leitores e, por fim, a arquitectura do servidor Modbus. O protocolo de comunicação Modbus, embora faça parte da aplicação, é implementado por funções que foram disponibilizadas e não será apresentado nesta dissertação.

4.1. ID ISOStart

O ID ISOStart é um programa de demonstração desenvolvido pela FEIG Electronic para familiarizar o utilizador com as funcionalidades dos seus leitores RFID. O programa permite:

- Testar a comunicação com os *Transponders*;
- Ler e alterar a configuração dos leitores RFID;
- Realizar um *upgrade* ao *firmware*.

Cada vez que é executada uma acção/comando é possível ver no ecrã o protocolo de comunicação entre o computador e o leitor RFID. Esta transparência possibilita o estudo da formação de tramas de envio e reposta por parte do *host*. Com este objectivo em mente, a primeira parte do trabalho consistiu em utilizar este programa para realizar alguns testes de funcionamento e estudar o protocolo de comunicações.

Este programa foi concebido para o sistema operativo Windows e a comunicação entre o computador e o leitor RFID é feita através da porta série (RS-232). Após completar as ligações físicas e iniciar o programa, a interface gráfica apresentada ao utilizador é a que se encontra representada na Figura 4.2. Do lado esquerdo, encontram-se os vários modos de

funcionamento do leitor descritos no capítulo 3, no lado direito estão disponíveis as funções de cada modo e no fundo pode-se ver o protocolo de comunicação (tramas enviadas e recebidas). De notar que o modo *Test and Measurement* é implementado pelo programa apenas para detectar etiquetas que estejam ao alcance do leitor RFID e consiste no envio periódico do comando *Inventory* (Tabela 3.4).

Como foi dito no capítulo anterior, à partida deste trabalho não era conhecido qual o tipo de etiqueta utilizado nas peças da linha de fabrico flexível. Assim, os primeiros testes realizados com este *software*, foram de identificação do tipo de *Transponder*. Para isso utilizaram-se as funções *Inventory* e *Get System Information* e verificou-se que a etiqueta era da marca Texas Instruments, com 64 blocos de dados de 4 bytes cada um. Comparando com os modelos existentes, chegou-se à conclusão que é o *Transponder* Tag-it HF-I Plus.

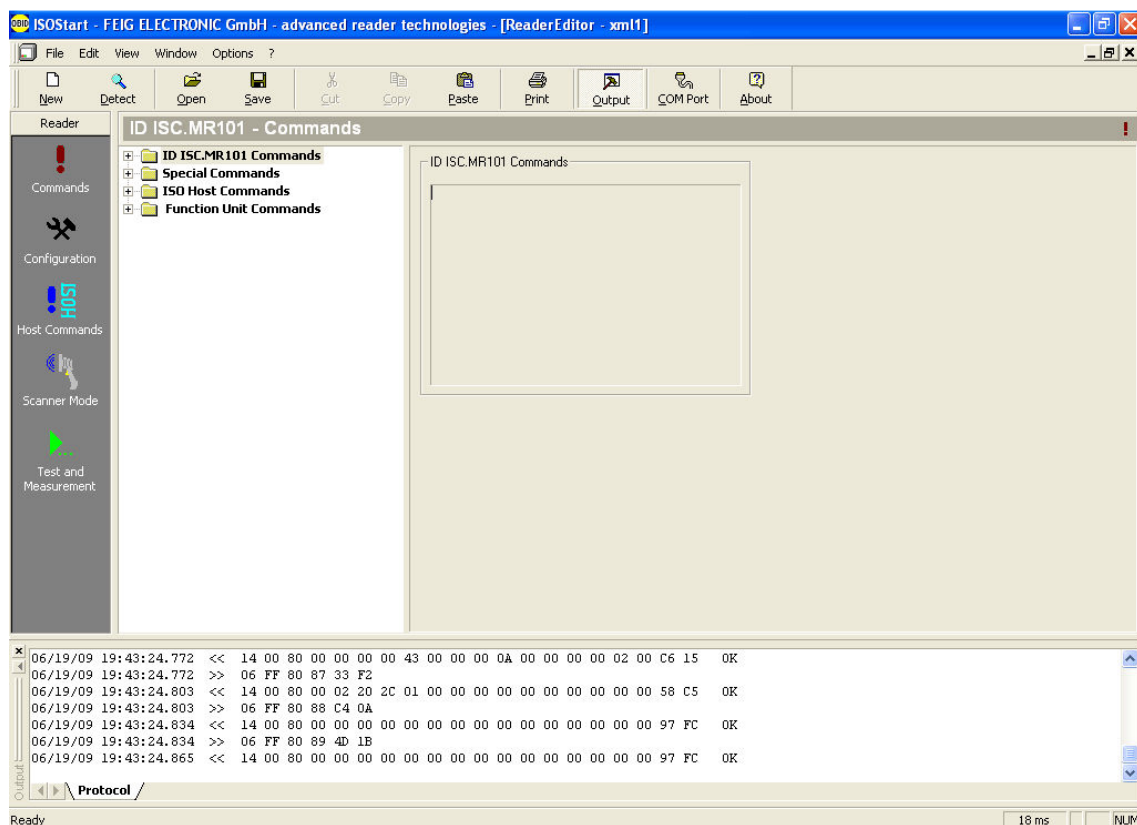


Figura 4.2 - Interface gráfica do programa ID ISOStart.

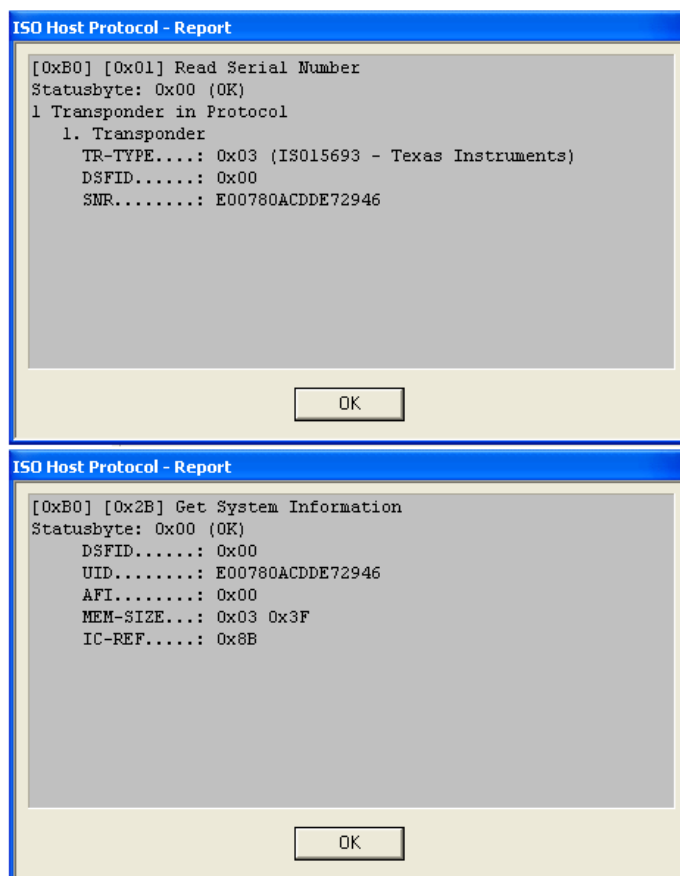


Figura 4.3 - Resposta ao comando *Inventory* e *Get System Information* utilizando o ID ISOStart.

Depois de estudado o funcionamento do leitor RFID utilizando o ID ISOStart e realizado alguns testes como ler e escreve dados de uma etiqueta, alterar parâmetros de configuração, utilizar o *Scan Mode*, entre outros, iniciou-se uma etapa de análise do protocolo de comunicações.

O leitor RFID possui 2 protocolos de troca de dados com o *host*:

Standard Protocol, que suporta tramas de dados até 255 bytes. Se o *host* escolher este protocolo para comunicar com o leitor, a trama de resposta dependerá do número de dados a transmitir. Se a trama não ultrapassar 255 bytes utilizará o *Standard Protocol*, caso contrário, o leitor responderá utilizando o *Advanced Protocol*. A estrutura das tramas de envio e resposta está definida na Figura 4.4 e na Tabela 4.1.

Advanced Protocol, para tramas superiores a 255 bytes. Neste caso, se o *host* utilizar este protocolo para comunicar com o leitor, a resposta será sempre em *Advanced Protocol*. A estrutura das tramas de envio e resposta está definida na Figura 4.5 e na Tabela 4.1.

Host → Leitor RFID

1	2	3	4...n-2	n-1	n
LENGTH = n	COM-ADR	CONTROL-BYTE	(DATA)	LSB-CRC16	MSB CRC16

Host ← Leitor RFID

1	2	3	4	(5...n-2)	n-1	n
LENGTH = n	COM-ADR	CONTROL-BYTE	STATUS	(DATA)	LSB-CRC16	MSB CRC16

Figura 4.4 - *Standard Protocol*.

<i>Host</i> → Leitor RFID								
1	2	3	4	5	(6...n-2)	n-1	n	
STX (0x02)	MSB ALENGTH	LSB ALENGTH	COM-ADR	CONTROL- BYTE	(DATA)	MSB CRC16	LSB CRC16	

<i>Host</i> ← Leitor RFID								
1	2	3	4	5	6	(7...n-2)	n-1	n
STX (0x02)	MSB ALENGTH	LSB ALENGTH	COM-ADR	CONTROL- BYTE	STATUS	(DATA)	MSB CRC16	LSB CRC16

Figura 4.5 - *Advanced Protocol*.

Tabela 4.1 - Descrição da trama do protocolo de comunicação do leitor RFID.

Byte(s)	Descrição
STX	Indica uma trama com mais de 255 bytes (<i>Advanced Protocol</i>)
ALENGTH	Número de bytes da trama incluindo STX, ALENGTH e CRC16
LENGTH	Número de bytes da trama incluindo LENGTH e CRC16
COM-ADR	Endereço do leitor caso esteja ligado a um barramento (RS-485)
CONTROL-BYTE	Define o comando a executar
STATUS	Inclui uma mensagem de estado do leitor
DATA	Campo opcional com tamanho variável (depende do comando)
CRC16	Polinómio $x^{16} + x^{12} + x^5 + 1$ Valor inicial 0xFFFF

Dos dois protocolos apresentados, optou-se por utilizar o *Standard Protocol*, uma vez que as trocas de dados que se esperam para esta aplicação não ultrapassarão os 255 bytes e principalmente porque o Modbus não permite enviar ou receber tantos dados de uma só vez (250 bytes no máximo). Portanto, deste ponto em diante sempre que se referir a estrutura das tramas, subentende-se que o protocolo utilizado é o *Standard Protocol*.

No capítulo anterior, foi dito que o leitor RFID podia comunicar com o *host* de 4 modos diferentes, *ISO15693 Host Comands*, *Scan Mode*, *Configuration Commands* e *Control Commands*. Quanto ao protocolo, 3 destes modos apenas diferem no CONTROL-BYTE (comando) e no campo de dados (data). No entanto, o *Scan Mode* não utiliza nenhum dos protocolos descritos anteriormente. Como a comunicação é feita apenas num sentido (do leitor para o *host*), os dados são enviados numa trama simples, que consiste em caracteres de início, separação e fim e com a possibilidade de se configurar.

Como não há razão para apresentar a estrutura das tramas de todos os comandos que foram estudados, apenas serão mostrados os mais importantes para esta aplicação, nomeadamente o *Read Configuration* e *Write Configuration* dos *Configuration Commands* e *Read Multiple Blocks* e *Write Multiple Blocks* dos *ISO15693 Host Comands*.

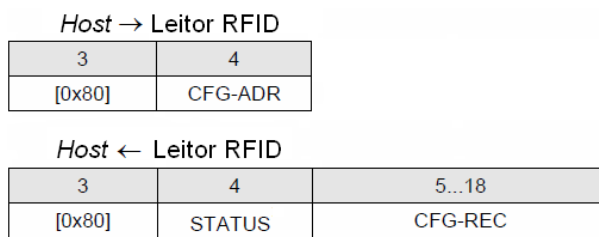


Figura 4.6 - Trama de envio e resposta do comando *Read Configuration*.

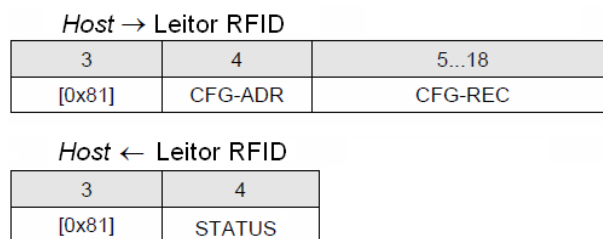


Figura 4.7 - Trama de envio e resposta do comando *Write Configuration*.

Tabela 4.2- Descrição das tramas dos comandos *Read Configuration* e *Write Configuration*.

Byte(s)	Descrição
CFG-ADR	Indica o endereço de memória do bloco de configuração e a localização (RAM ou EEPROM)
CFG-REC	14 Bytes de dados do bloco de configuração (lidos ou que se pretendem escrever)

Os comandos *Read Configuration* (0x80) e *Write Configuration* (0x81) permitem ler e escrever, respectivamente, blocos de configuração de 14 bytes. Estes blocos permitem alterar vários parâmetros do leitor, tais como, a interface de comunicação (RS-232/RS-485), activar ou desactivar o *Scan Mode*, definir a trama a enviar nesse mesmo modo, entre outros. As tramas de envio e resposta destes comandos têm a estrutura definida nas Figura 4.6 e Figura 4.7 e na Tabela 4.2. Na figura seguinte é possível ver um exemplo destes comandos utilizando o programa ID ISOStart.

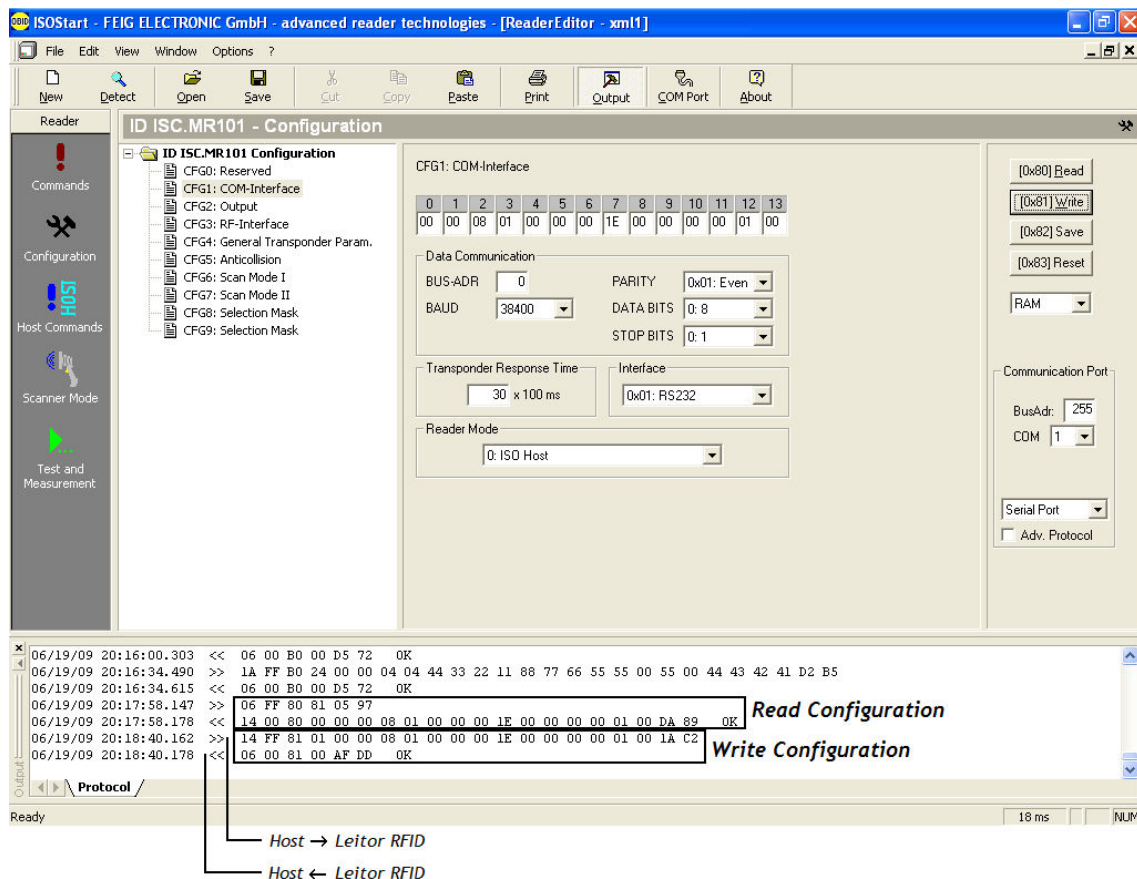


Figura 4.8 - Exemplo dos comandos *Read Configuration* e *Write Configuration* utilizando o ID ISOStart.

Os comandos *Read Multiple Blocks* (0x23) e *Write Multiple Blocks* (0x24) permitem ler e escrever, respectivamente, blocos de dados das etiquetas. O tamanho do bloco varia com o tipo de etiqueta, sendo neste caso 4 bytes. A estrutura destas tramas está definida nas Figura 4.9 e Figura 4.10 e na Tabela 4.3. Como estes comandos pertencem às funções do *ISO15693 Host Commands*, o CONTROL-BYTE tem o valor 0xB0 e só no byte seguinte está definido o comando. Na Figura 4.11 é possível ver um exemplo destes comandos utilizando o programa ID ISOStart.

Host → Leitor RFID					
3	4	5	(6...13)	6 / (14)	7 / (15)
[0xB0]	0x23	MODE	UID	DB-ADR	DB-N
Host ← Leitor RFID					
3	4	5	6	7	8...n
[0xB0]	STATUS	DB-N	DB-SIZE	SEC-STATUS	DB
Repeated DB-N times					

Figura 4.9 - Trama de envio e resposta do comando *Read Multiple Blocks*.

<i>Host → Leitor RFID</i>							
3	4	5	(6...13)	6 / (14)	7 / (15)	8 / (16)	9...n / (17...n)
[0xB0]	0x24	MODE	UID	DB-ADR	DB-N	DB-SIZE	DB
<i>Host ← Leitor RFID</i>							Repeated DB-N times
3	4						
[0xB0]	STATUS						

Figura 4.10 - Trama de envio e resposta do comando *Write Multiple Blocks*.

Tabela 4.3 - Descrição das tramas dos comandos *Read Multiple Blocks* e *Write Multiple Blocks*.

Byte(s)	Descrição
MODE	Indica o modo de comunicação com a etiqueta (<i>Non-addressed, Addressed, Selected</i>)
UID	Identificação da etiqueta (opcional, depende do MODE)
DB-ADR	Endereço do primeiro bloco de dados a ser lido ou escrito
DB-N	Número de blocos de dados a ser lidos ou escritos
DB-SIZE	Número de bytes de cada bloco de dados
SEC-STATUS	Indica se o bloco está protegido ou não
DB	Dados a ser lidos ou escritos de 1 bloco

06/19/09 19:54:18.256 << 06 00 B0 01 5C 63 Reader: No Transponder in Reader Field	
06/19/09 19:54:21.037 >> 09 FF B0 23 00 00 04 A2 7D	
06/19/09 19:54:21.069 << 1C 00 B0 00 04 04 00 12 34 56 78 00 9A BC DE FF 00 FF FF 00 00 00 CC CC DD DD 4F A7 OK	
06/19/09 19:54:24.506 >> 09 FF B0 23 00 00 04 A2 7D	
06/19/09 19:54:24.569 << 06 00 B0 01 5C 63 Reader: No Transponder in Reader Field	
06/19/09 19:54:27.772 >> 09 FF B0 23 00 00 04 A2 7D	
06/19/09 19:54:27.803 << 1C 00 B0 00 04 04 00 12 34 56 78 00 9A BC DE FF 00 FF FF 00 00 00 CC CC DD DD 4F A7 OK	<i>Read Multiple Blocks</i>
Protocol /	
Ready 27 ms NUM	
06/19/09 20:13:52.334 << 06 00 B0 01 5C 63 Reader: No Transponder in Reader Field	
06/19/09 20:14:11.350 >> 07 FF B0 2B 00 5F 88	
06/19/09 20:14:11.381 << 13 00 B0 00 00 E0 07 80 AC DD E7 29 46 00 03 3F 8B D0 6C OK	
06/19/09 20:16:00.162 >> 1A FF B0 24 00 00 04 04 12 34 56 78 9A BC DE FF FF 00 00 CC CC DD DD AB D0	
06/19/09 20:16:00.303 << 06 00 B0 00 D5 72 OK	
06/19/09 20:16:34.490 >> 1A FF B0 24 00 00 04 04 44 33 22 11 88 77 66 55 55 00 55 00 44 43 42 41 D2 B5	
06/19/09 20:16:34.615 << 06 00 B0 00 D5 72 OK	<i>Write Multiple Blocks</i>
Protocol /	
Ready 121 ms NUM	

Figura 4.11 - Exemplo dos comandos *Read Multiple Blocks* e *Write Multiple Blocks* utilizando o ID ISOStart.

4.2. XPort e ligação TCP/IP

Terminada a etapa de estudo do protocolo de troca de dados entre o *Host* e o leitor RFID, seguiu-se uma fase de implementação da conexão entre os dispositivos. Como foi explicado no capítulo anterior, a plataforma a utilizar ICnova AP700 Base, que contém o sistema operativo Linux, vai estar conectada através do XPort ao leitor RFID. Como tal, esta fase consistiu em configurar o XPort e criar uma conexão TCP/IP entre o mesmo e a plataforma. Contudo, é necessário referir que em toda a fase de implementação do trabalho, a ICnova AP700 Base foi substituída por um computador, uma vez que esta não foi disponibilizada e para a concretização do trabalho isso não interfere.

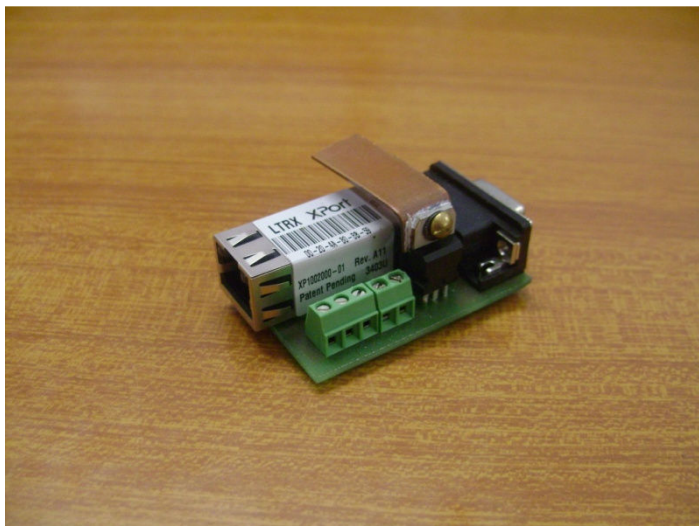


Figura 4.12 - XPort.

O XPort disponibilizado (Figura 4.12), já com o condicionamento e montagem em PCB realizados, é alimentado a 5 V CC e converte os dados transmitidos por RS-232 para Ethernet 10/100Mbit. Possui um sistema operativo e permite a comunicação por TCP/IP, entre outras características. A sua configuração pode ser feita de 3 formas diferentes:

- DeviceInstaller
- Telnet
- Porta Série

O DeviceInstaller consiste num programa para Windows que detecta os dispositivos XPort ligados à rede a partir dos seus endereços MAC (*Media Access Control*) e permite configurá-los (os dispositivos). No caso da configuração por Telnet a porta utilizada é a 9999, mas é necessário conhecer o endereço IP do XPort. Assim, neste trabalho utilizou-se inicialmente o DeviceInstaller para configurar o XPort e o seu IP e posteriormente a ligação Telnet. A configuração por porta série não foi utilizada neste caso. Os parâmetros a configurar incidiram principalmente na comunicação RS-232 (*baud rate*, paridade, entre outros) e na ligação TCP/IP (IP, porta, modo como aceita conexões, como as fecha, entre outros).

Após configurar o XPort de acordo com os requisitos da aplicação, o passo seguinte consistiu na criação de uma conexão entre este e o computador. As funções para criar conexões TCP/IP baseiam-se na utilização de *sockets*. Muito resumidamente, um *socket* é uma estrutura de dados que armazena toda a informação necessária de uma conexão, tal como, o endereço IP, porta, protocolo, tipo de socket, entre outros. Isto significa que quando se cria uma conexão, está-se a criar um *socket* e para o utilizar apenas é necessário o seu descritor (apontador para a estrutura de dados do *socket*). Do ponto de vista da conexão, a aplicação pode ser classificada como cliente ou servidor. No primeiro caso, a aplicação tenta criar a conexão, enquanto no segundo a aplicação espera por uma. Nesta aplicação, como se pode concluir, pretende-se um cliente, visto que é esta que vai iniciar a comunicação. Neste sentido implementou-se uma função para criar uma conexão do tipo TCP, que recebe como parâmetros de entrada o IP e a porta e retorna o descritor do *socket*. Embora se pudesse falar mais detalhadamente desta função e da programação baseada em *sockets*, esse não é o objectivo principal deste projecto e por isso não se entrará em detalhes.

4.3. API

Uma vez criada a base da comunicação entre o computador e o leitor RFID (a mesma para o leitor de códigos de barras e leitor de cartões), criou-se uma API para implementar algumas funções importantes que seriam utilizadas na criação do servidor Modbus. Para isso definiu-se uma estrutura (*reader_rfid_t*) que guardaria os dados de um leitor RFID e que seriam necessários para as funções, definidos na Tabela 4.4.

Tabela 4.4 - Dados da estrutura criada para um leitor RFID.

host	<i>String</i> que guarda o IP do leitor RFID (XPort)
service	<i>String</i> que guarda a porta do leitor RFID (XPort)
s	Descritor da conexão
cfg	Matriz que guarda os dados dos blocos de configuração do leitor RFID
con_status	Variável que indica se existe conexão ou não
conf_status	Variável que indica se o leitor RFID está configurado ou não
rfid_status	Guarda o byte <i>Status</i> enviado pelo leitor RFID

As funções implementadas têm o objectivo de tratar de todo o protocolo de comunicação com o leitor RFID, visto anteriormente, e oferecer uma interface simples de troca de dados com o leitor. Assim, criaram as seguintes funções:

- `open_connection()`
- `close_connection()`
- `rd_config()`
- `wr_config()`
- `init_config()`
- `rfid_init()`
- `read_mb()`
- `write_mb()`

A função *open_connection()*, que cria uma conexão, recebe como parâmetro de entrada um apontador para uma estrutura do tipo *reader_rfid_t* e retorna, em caso de sucesso, o descritor do *socket* criado. A função referida no subcapítulo anterior é aqui utilizada para iniciar a conexão. Na Figura 4.13 pode-se ver o fluxograma desta função. Note-se apenas, que quando se diz “Atualiza estado” significa que actualiza o valor das variáveis *con_status* e/ou *conf_status*. Embora não se tenha detalhado no fluxograma, os erros estão bem definidos e para cada um estão definidos valores diferentes (negativos), tanto nesta função como nas seguintes.

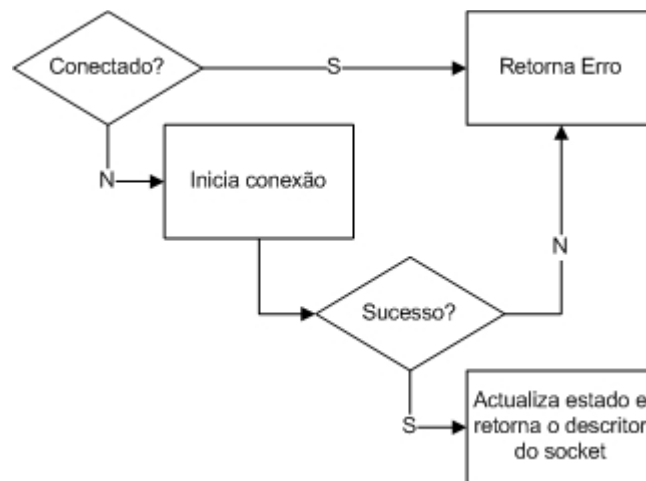


Figura 4.13 - Fluxograma da função `open_connection()`.

A função `close_connection()` é semelhante à anterior, mas em vez de iniciar, fecha a conexão. O seu fluxograma está representado na Figura 4.14.

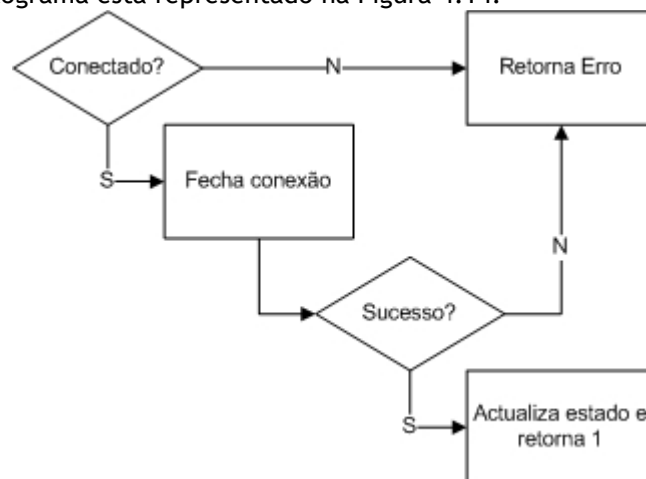


Figura 4.14 - Fluxograma da função `close_connection()`.

A função `rd_config()` lê um bloco de configuração do leitor RFID e guarda na matriz `cfg`. Recebe como parâmetros de entrada um apontador para uma estrutura do tipo `reader_rfid_t` e o endereço do bloco de configuração. Na Figura 4.15 está representado o seu fluxograma. A função começa por verificar se o leitor RFID está conectado e de seguida limpa o `buffer` do `socket`. Este passo é necessário, ao contrário de outras funções a seguir apresentadas, pois se o leitor estiver em *Scan Mode*, o `buffer` pode conter dados que serão interpretados como trama de resposta ao envio do comando *Read Configuration*. Após enviar a trama para o leitor RFID a função espera por a resposta apenas durante um certo período de tempo (*timeout*). Se não ocorrer nenhum erro o bloco de configuração é guardado e a função retorna o valor 1.

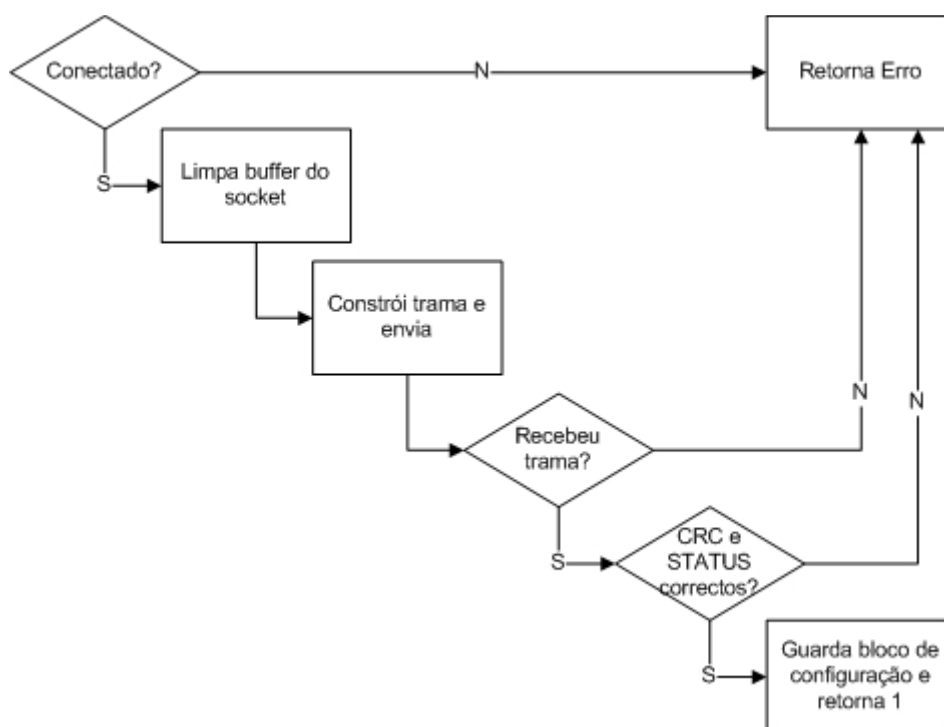


Figura 4.15 - Fluxograma da função `rd_config()`.

A função `wr_config()` escreve um bloco de configuração da matriz `cfg` no leitor RFID. Recebe como parâmetros de entrada um apontador para uma estrutura do tipo `reader_rfid_t` e o endereço do bloco de configuração. Na Figura 4.16 está representado o seu fluxograma. Embora possa parecer estranho a verificação se o leitor está configurado (`conf_status`), esta justifica-se pois é necessário que a matriz `cfg` esteja preenchida, ou seja, o leitor configurado.

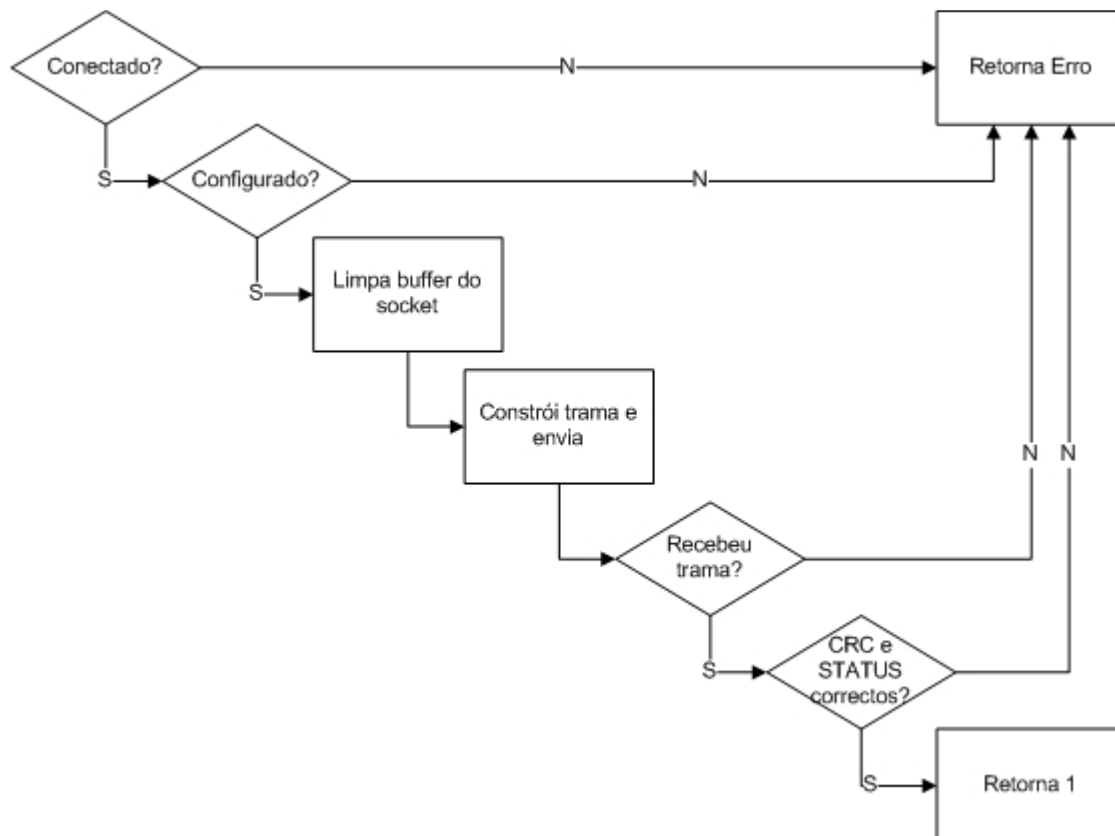


Figura 4.16 - Fluxograma da função `wr_config()`.

A função `init_config()` utiliza as duas funções acima descritas para iniciar a configuração do leitor. Neste sentido, definiu-se 2 modos diferentes de o fazer. Pode-se iniciar o leitor com uma configuração predefinida (*default*) ou carregar a configuração actual que o leitor contém. A função recebe como parâmetros de entrada um apontador para uma estrutura do tipo `reader_rfid_t` e uma variável que indica o tipo de inicialização (*mode*). Na figura seguinte está representado o seu fluxograma. No caso da configuração *default*, antes de ser chamada a função `wr_config()`, a matriz `cfg` é preenchida com valores predefinidos. Em ambos os casos existe um ciclo que se repete 9 vezes, pois corresponde ao número de blocos de configuração presente na matriz. Outro aspecto a realçar é a presença de um tempo de paragem de 5 milissegundos. Isto deve-se ao facto de o leitor RFID necessitar de um intervalo mínimo de repouso entre comunicações. Se a função for executada com sucesso, a variável `conf_status` é actualizada e o valor 1 é retornado.

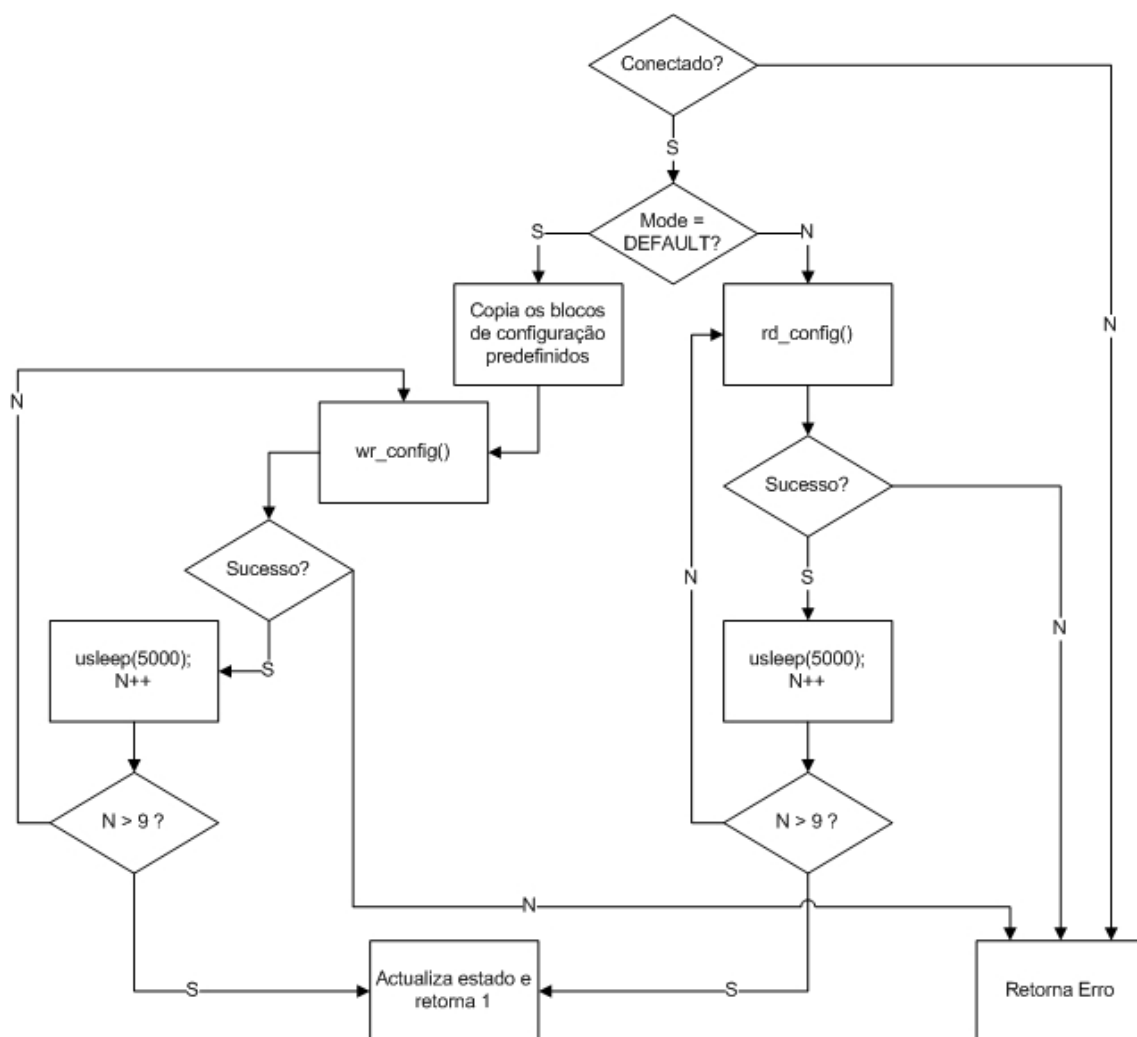


Figura 4.17 - Fluxograma da função *init_config()*.

A função *rfid_init()* integra as funções já apresentadas e inicia a conexão e a configuração do leitor RFID. Esta acaba por ser a função que será utilizada pela aplicação para inicialização. Os parâmetros de entrada são os mesmos que a função *init_config()* e retorna 1 em caso de sucesso. O fluxograma relativo a esta função está representado na Figura 4.18 e o seu funcionamento é bastante simples. Convém apenas referir que a acção “Actualiza estado” é na verdade uma inicialização das variáveis *con_status* e *conf_status*.

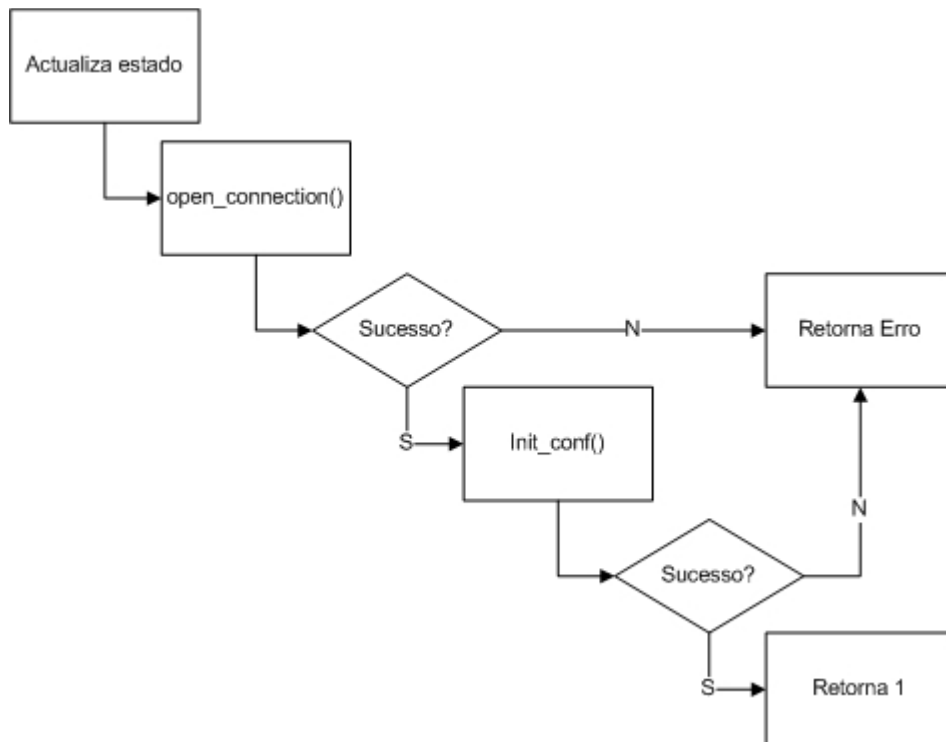


Figura 4.18 - Fluxograma da função `rfid_init()`.

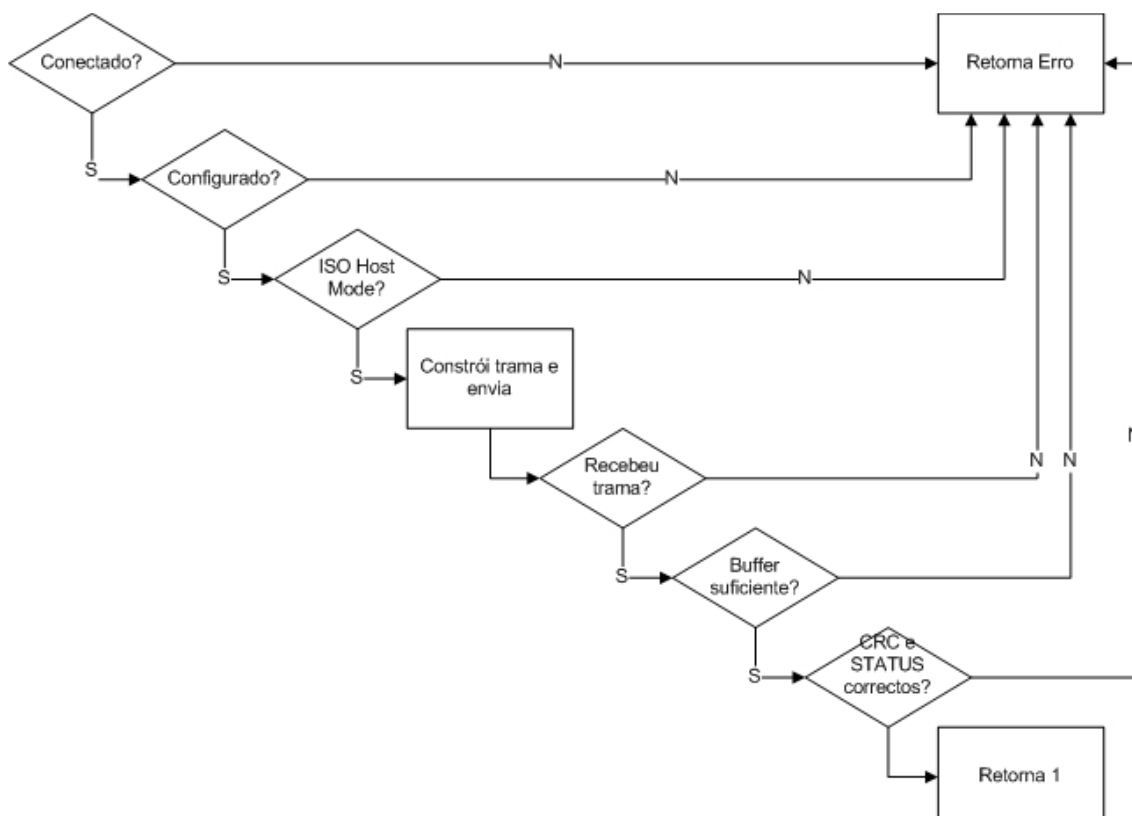


Figura 4.19 - Fluxograma da função `read_mb()`.

A função `read_mb()` consiste na execução do comando *Read Multiple Blocks* (em *Non-addressed Mode*). Recebe como parâmetros de entrada um apontador para uma estrutura do

tipo *reader_rfid_t*, o endereço do bloco de dados, o número de blocos, um apontador para um *buffer* (onde serão guardados os dados) e o seu tamanho. O seu funcionamento está descrito no fluxograma da Figura 4.19. Após verificar se o leitor RFID está conectado e configurado, é necessário verificar se o leitor não se encontra em *Scan Mode*. Caso isso aconteça, não é possível executar o comando *Read Multiple Blocks* e retornará um erro. Se pelo contrário, não ocorrer nenhum erro, os dados serão guardados no *buffer* e a função retornará 1.

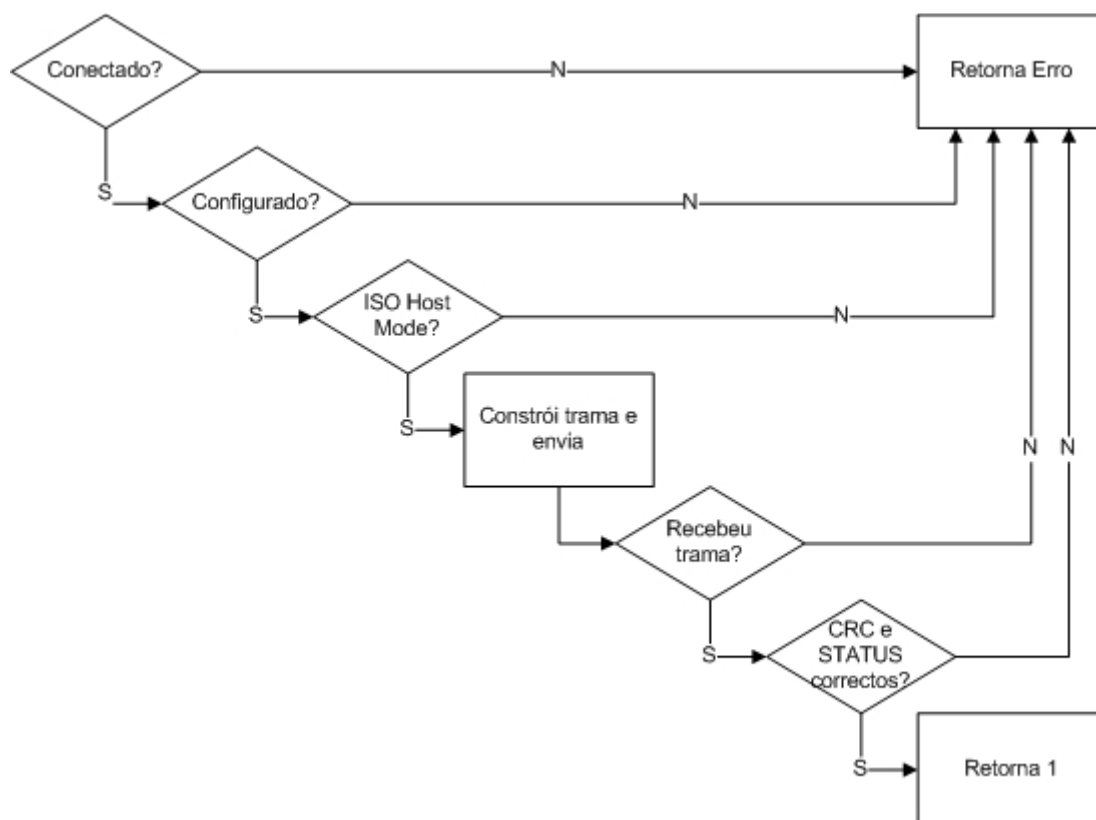


Figura 4.20 - Fluxograma da função *write_mb()*.

A função *write_mb()* consiste na execução do comando *Write Multiple Blocks* (em *Non-addressed Mode*). Recebe como parâmetros de entrada um apontador para uma estrutura do tipo *reader_rfid_t*, o endereço do bloco de dados, o número de blocos e um apontador para um *buffer* (onde estão os dados a serem escritos). Como se pode ver pelo fluxograma da Figura 4.20, o seu funcionamento não é muito diferente da função *read_mb()*.

A função *scan_read()* foi implementada para ler os dados transmitidos pelo leitor RFID quando este se encontra em *Scan Mode*. Os seus parâmetros de entrada são um apontador para uma estrutura do tipo *reader_rfid_t*, um apontador para um *buffer* (onde serão guardados os dados), o seu tamanho e um valor de *timeout*. O seu funcionamento encontra-se descrito no fluxograma da Figura 4.21. O *timeout* é utilizado quando a função verifica se recebeu dados, ficando a espera até atingir o tempo definido. Como se pode ver, se não ocorrerem erros a função retorna o número de bytes recebidos.

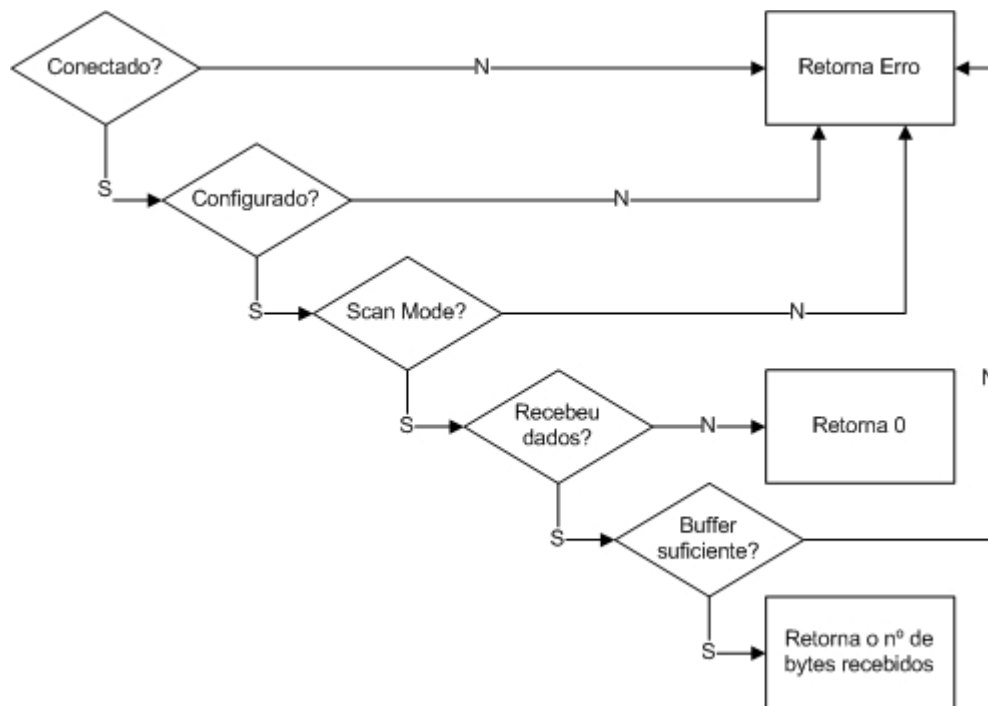


Figura 4.21 - Fluxograma da função *scan_read()*.

De forma análoga ao leitor RFID, também foi criada uma API para os leitores de códigos de barras e de cartões de banda magnética. Uma vez que, estes dois dispositivos possuem um funcionamento igual quanto à forma como transmitem os dados, apenas foi criada uma API comum aos dois. Tal como no caso anterior, criou-se uma estrutura (*reader_bc_t*) para guardar os dados necessários (Tabela 4.5) e implementaram-se as funções:

- *open_connection_bc()*
- *close_connection_bc()*
- *bc_init()*
- *read_bc()*

Tabela 4.5 - Dados da estrutura criada para um leitor de códigos de barras ou de cartões.

host	<i>String</i> que guarda o IP do leitor de códigos de barras ou de cartões (XPort)
service	<i>String</i> que guarda a porta do leitor de códigos de barras ou de cartões (XPort)
s	Descritor da conexão
con_status	Variável que indica se existe conexão ou não

As funções *open_connection_bc()* e *close_connection_bc()* são iguais às anteriormente apresentadas para o leitor RFID, enquanto a função *bc_init()* apenas difere da *rfid_init()*, pois não necessita de iniciar a configuração do leitor. Assim sendo, apenas se apresenta na Figura 4.22 o fluxograma da função *read_bc()*. Esta função recebe como parâmetros de entrada um apontador para uma estrutura do tipo *reader_bc_t*, um apontador para um *buffer* e o tamanho deste. O seu funcionamento consiste em verificar se recebeu dados, e em caso afirmativo, guardá-los no *buffer* e retornar o número de bytes recebidos.

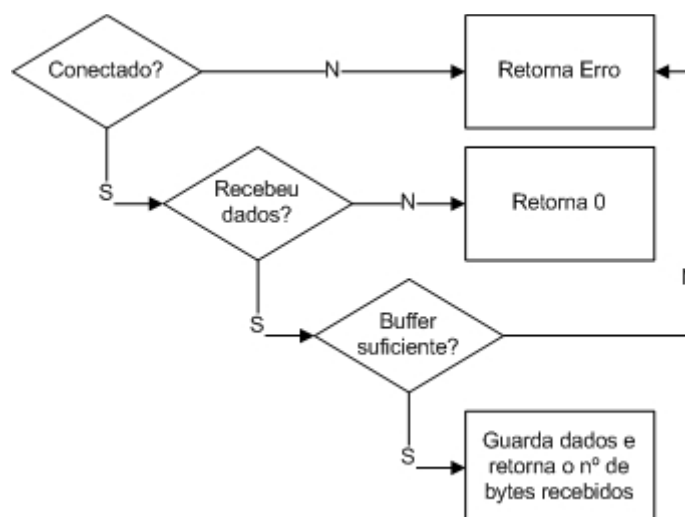


Figura 4.22 - Fluxograma da função *read_bc()*.

Após a conclusão das API para o leitor RFID e para os leitores de códigos de barras ou de cartões de banda magnética, realizaram-se alguns testes para comprovar o funcionamento das funções implementadas, tais como, ler e escrever numa etiqueta, alterar a configuração, testar o funcionamento em *Scan Mode*, ler códigos de barras, ler cartões de banda magnética, entre outros. Obtidos os resultados pretendidos, iniciou-se uma nova etapa, a criação do servidor modbus.

4.4. Servidor Modbus

O Modbus é um protocolo de comunicação muito utilizado em dispositivos de automação devido à sua simplicidade e versatilidade. Pode ser utilizado em diferentes meios físicos de comunicação, tal como, Ethernet ou RS-232. Neste caso irá ser utilizado o Modbus TCP sobre Ethernet. Os dados do Modbus podem ser de 4 tipos diferentes:

- *Discretes*. São variáveis de 1 bit e apenas podem ser lidas.
- *Coils*. São variáveis de 1 bit e podem ser lidas e escritas.
- *Registers*. São variáveis de 16 bits e apenas podem ser lidas.
- *Holding registers*. São variáveis de 16 bits de leitura e escrita.

Para aceder a estas variáveis, o protocolo Modbus define endereços de 0 a 65535. O seu funcionamento baseia-se numa arquitectura mestre-escravo, em que o mestre envia um código de uma função e os dados necessários para a execução da mesma e recebe uma resposta da parte do escravo. Neste trabalho pretende-se implementar um escravo, visto que a aplicação espera ordens vindas do PLC.

Para a criação do servidor Modbus foram disponibilizadas funções que já implementam o protocolo para um escravo, mas apenas para duas funções Modbus:

- *Read Holding Registers* (código da função 03). Esta função permite a leitura de registos de 16 bits (*Input registers*).
- *Write Multiple Registers* (código da função 16). Esta função permite a escrita de registos de 16 bits (*holding registers*).

Deste modo, definiu-se uma organização dos registos para o leitor RFID, leitor de códigos de barras e leitor de cartões que se podem ver nas Tabela 4.6, Tabela 4.7 e Tabela 4.8, respectivamente. Note-se que os registos estão classificados como de leitura (L) e de leitura e escrita (L/E).

Tabela 4.6 - Organização dos registos Modbus para um leitor RFID.

Endereço	Registo	Tipo	Descrição
0-3	ID	L/E	8 Caracteres que identificam o leitor. Os registos podem ser acedidos individualmente.
4	CMD	L/E	Bit 15: <i>reset</i> . Este bit estará sempre a zero. No entanto se for escrito o valor 1, o leitor RFID será reiniciado. Bit 14: define o modo de configuração inicial. Quando é feito um <i>reset</i> , este bit indica se o leitor inicia com a configuração <i>default</i> ou com a configuração actual. Bit 0: coloca o leitor RFID em <i>Scan Mode</i> . Se for escrito o valor 1, o leitor funcionará em <i>Scan Mode</i> , caso contrário funcionará em <i>ISO Host Mode</i> .
5	SCAN_ADDR_LEN	L/E	Bit 8 a 15: Endereço do primeiro bloco de dados a ser lido em <i>Scan Mode</i> . Bit 0 a 8: Número de bytes de dados a serem lidos em <i>Scan Mode</i> . Através dos blocos de configuração é possível definir quais os blocos de dados a serem transmitidos em <i>Scan Mode</i> . Este registo altera esses blocos de configuração.
6	READ_ADDR	L/E	Endereço do primeiro bloco de dados a ser lido em <i>ISO Host Mode</i> .
7	WRITE_ADDR	L/E	Endereço do primeiro bloco de dados a ser escrito.
8-127	BUFFER	L/E	É utilizado para leitura e escrita de dados. Se o <i>buffer</i> é lido, serão enviados os dados da etiqueta, se for escrito, os dados recebidos serão escritos na etiqueta.
130	STATUS_RFID	L	Byte Status que o leitor RFID envia.
131	STATUS_CON	L	Estado da conexão.
132	STATUS_CONF	L	Estado da configuração.
133	ERRORS	L	Valor do último erro que ocorreu, definido na API implementada.
134	ERRORS_NO	L	Sempre que uma função retornar um <i>errno</i> , este será guardado neste registo.

Tabela 4.7 - Organização dos registos Modbus para um leitor de códigos de barras.

Endereço	Registo	Tipo	Descrição
2000-2003	ID	L/E	8 Caracteres que identificam o leitor.

2004	STATUS_CON	L	Estado da conexão.
2005	ERRORS	L	Valor do último erro que ocorreu.
2006	ERRORS_NO	L	Valor da variável <i>errno</i> .
2007-2131	READ	L	<i>Buffer</i> de leitura.

Tabela 4.8 - Organização dos registos Modbus para um leitor de cartões.

Endereço	Registo	Tipo	Descrição
4000-4003	ID	L/E	8 Caracteres que identificam o leitor.
4004	STATUS_CON	L	Estado da conexão.
4005	ERRORS	L	Valor do último erro que ocorreu.
4006	ERRORS_NO	L	Valor da variável <i>errno</i> .
4007-4131	READ	L	<i>Buffer</i> de leitura.

Esta estruturação permitiria executar as funções inicialmente pretendidas, como ler e escrever para as etiquetas, utilizar o *Scan Mode*, ler códigos de barras e cartões de banda magnética. No caso de haver mais do que um leitor do mesmo tipo, os endereços terão a mesma ordem mas 200 registos a seguir. Isto é, se houver 4 leitores RFID, para aceder ao registo CMD do 3º, por exemplo, utiliza-se o endereço 404. Assim, rapidamente se percebe que a aplicação suportará, no máximo, 10 leitores de cada tipo. Um ponto importante que ainda não foi referido, é o facto de, os registos não representarem propriamente uma posição de memória. Embora o Modbus defina endereços, estes podem ser considerados apenas como referências no funcionamento do programa, como é o caso. Nesta aplicação estes registos funcionarão como comandos, ou seja, quando se pretende ler o registo 4007, por exemplo, o que se faz na verdade, é chamar a função *read_bc()* e enviar os dados.

Depois de definida a arquitectura dos registos, iniciou-se a implementação de duas funções:

- *read_outwords()* que lê os registos e envia os dados correspondentes;
- *write_outwords()* que escreve nos registos os dados recebidos.

Estas funções, que tem como parâmetros de entrada o endereço de um registo (primeiro endereço para o qual se quer ler ou escrever), o número de registos (que se querem ler ou escrever) e um apontador para um *buffer* de registos (de onde serão lidos ou escritos dados), serão passadas como um argumento à função que trata da comunicação Modbus e que foi disponibilizada. Assim, sempre que forem chamadas as funções do Modbus *Read Holding Registers* e *Write Multiple Registers*, o código que executará será o das funções *read_outwords()* e *write_outwords()*.

O funcionamento das duas funções é semelhante em termos gerais e pode ser visto na Figura 4.23. Em primeiro lugar, as funções verificam a qual dos dispositivos o endereço do registo corresponde. Se não estiver na gama de valores acima referidos é retornado o valor -2 que corresponde a endereço inválido. De seguida, e de forma igual para todos os dispositivos, as funções executarão os comandos correspondentes a cada registo, explicadas mais a frente. Se o número de registos for superior a 1, ou seja, pretende-se executar mais do que um comando, no fim da primeira execução, o número de registos é decrementado e o endereço incrementado. Nos registos de *buffer* isso não se verifica, uma vez que o número de registos indica o número de dados que se quer ler ou escrever. Se não ocorrerem erros as funções retornam o valor 0, caso contrário, retornam o valor -1 que indica que houve um erro. Se isto

acontecer, o erro estará guardado no registo ERRORS e em alguns casos poderá ser consultado o *errno*. No que diz respeito à acção “Executa função correspondente”, será a seguir explicado, resumidamente, em que consiste cada comando dos registos endereçados.

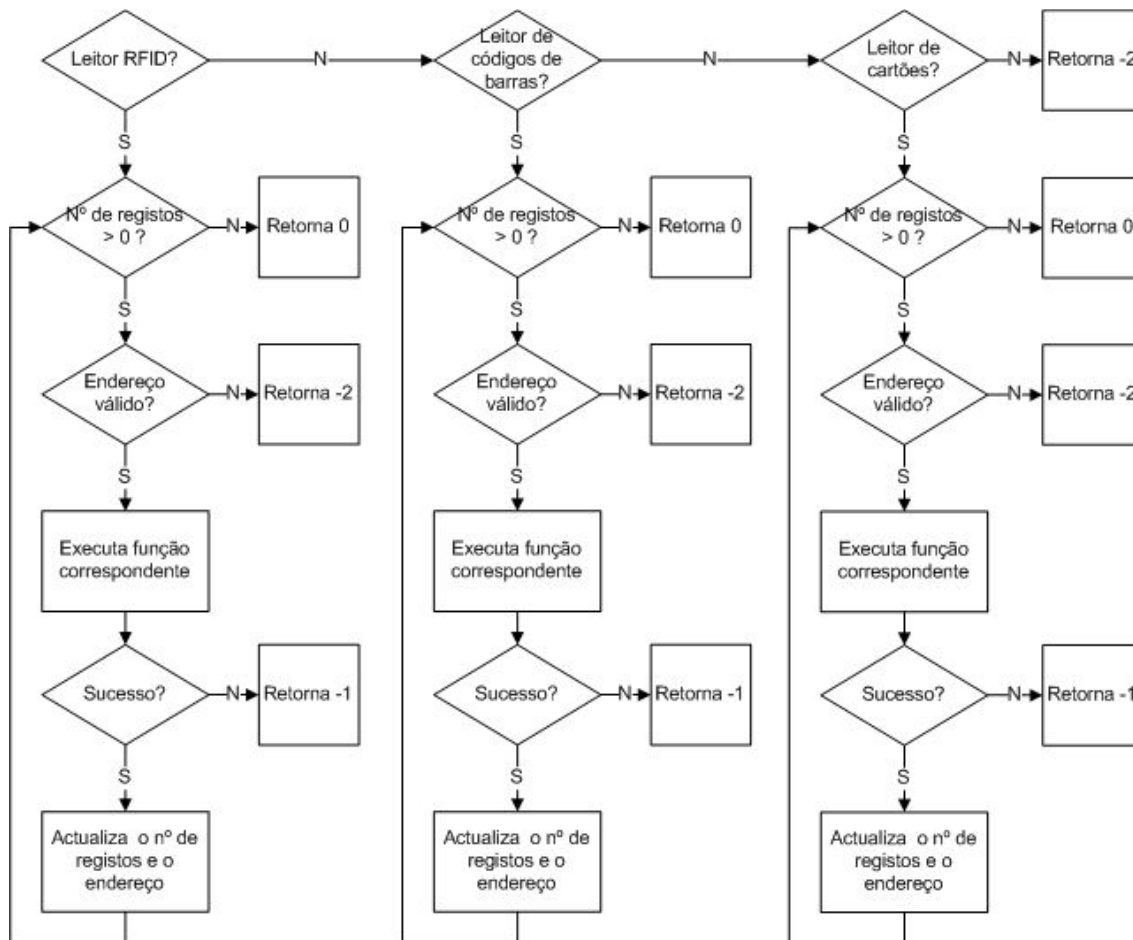


Figura 4.23 Fluxograma das funções *read_outwords()* e *write_outwords()*.

Quanto à função *write_outwords()*, o endereçamento para os registos ID, READ_ADDR e WRITE_ADDR consiste em guardar os valores, passados no parâmetro *buffer* (da função *write_outwords()*) numa estrutura definida com esse propósito. Já o CMD é um pouco mais complexo e o seu fluxograma encontra-se representado na Figura 4.24. Como se vê, o *reset* consiste em fechar a conexão e chamar a função *rfid_init()*, que utilizará o bit 14 para definir o modo de inicialização. Caso se pretenda configurar o leitor para *Scan Mode* ou voltar a colocá-lo em *ISO Host Mode*, é necessário alterar a configuração do leitor, utilizando para isso a função *wr_config()*. Neste caso foi também criada uma *thread* quando o leitor se encontra em *Scan Mode* para periodicamente chamar a função *scan_read()* e guardar os dados lidos. Quando se escreve para o registo SCAN_ADDR_LEN, a função *wr_config()* é utilizada para alterar as configurações relativas *Scan Mode*. Por último, o registo BUFFER quando utilizado pela função *write_outwords()*, irá chamar a função *write_mb()* para escrever os dados na etiqueta.

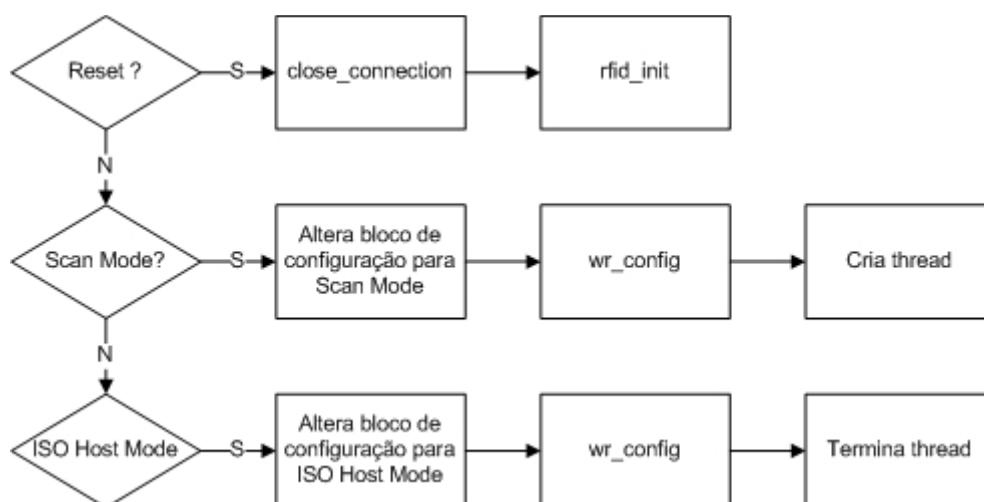


Figura 4.24 - Fluxograma da função `write_outwords()` para o registo CMD.

Quanto à função `read_outwords()`, o endereçamento para qualquer registo, exceptuado o BUFFER, consiste em copiar os dados, anteriormente guardados numa estrutura, para o parâmetro de entrada `buffer` (da função `read_outwords()`). Se o endereço a ler for o registo BUFFER, então nesse caso e para o leitor RFID, a função `read_mb()` é chamada e os dados lidos da etiqueta são passados. Isto, se o leitor se encontrar em ISO Host Mode, senão os dados copiados serão os que foram lidos na thread criada para o Scan Mode. Como já foi dito, esta thread executa periodicamente a função `scan_read()` e guarda os dados da última leitura, ou seja, se não forem lidos dentro desse período de tempo serão perdidos (requisito específico para este trabalho). Para os leitores de códigos de barras e de cartões, a função chamada quando se lê o registo BUFFER é a `read_bc()`.

Definidas as funções para ler e escrever nos registos Modbus, apenas faltaria a inicialização da aplicação. Como se pretendia ter a possibilidade de alterar alguns parâmetros como, número de leitores de cada tipo, IP de cada um, entre outros, criaram-se 3 ficheiros de texto, um para cada tipo de leitor, que guardariam essas informações e seriam lidos sempre que o programa fosse executado. Na Figura 4.25 pode-se ver um exemplo de um desses ficheiros.

Evidentemente, esta aplicação contém muitos mais detalhes que não foram explicados, pois o objectivo é apresentar o funcionamento geral do algoritmo e não entrar em pormenores de programação. Uma última nota, apenas para referir que se utilizou a linguagem C para a implementação do programa.

```

Ficheiro de configuração
*****
Número de leitores RFID:
2
*****
Leitor
-----
Identificação(string com 8 caracteres no máximo):
RFID1
-----
IP do XPort(String):
192.168.105.40
-----
Porta de XPort(string):
10001
-----
Conf_Mode (1-não altera a configuração do leitor 0-carrega a configuração predefinida):
0
*****
Leitor 2
-----
Identificação(string com 8 caracteres no máximo):
RFID2
-----
IP do XPort(String):
192.168.105.41
-----
Porta de XPort(string):
10001
-----
Mode (1-não altera a configuração do leitor 0-carrega a configuração predefinida):
0
*****

```

Figura 4.25 - Exemplo do ficheiro de texto de configuração dos leitores RFID.

Capítulo 5

Validação da Solução

Após terminada a fase de implementação da aplicação, procedeu-se a uma fase de testes e validação da solução. Para isso, utilizou-se um *software* que simula um mestre Modbus, o Modbus Poll. Neste capítulo, serão apresentados alguns desses testes para verificação do funcionamento da solução implementada.

5.1. Leitor RFID

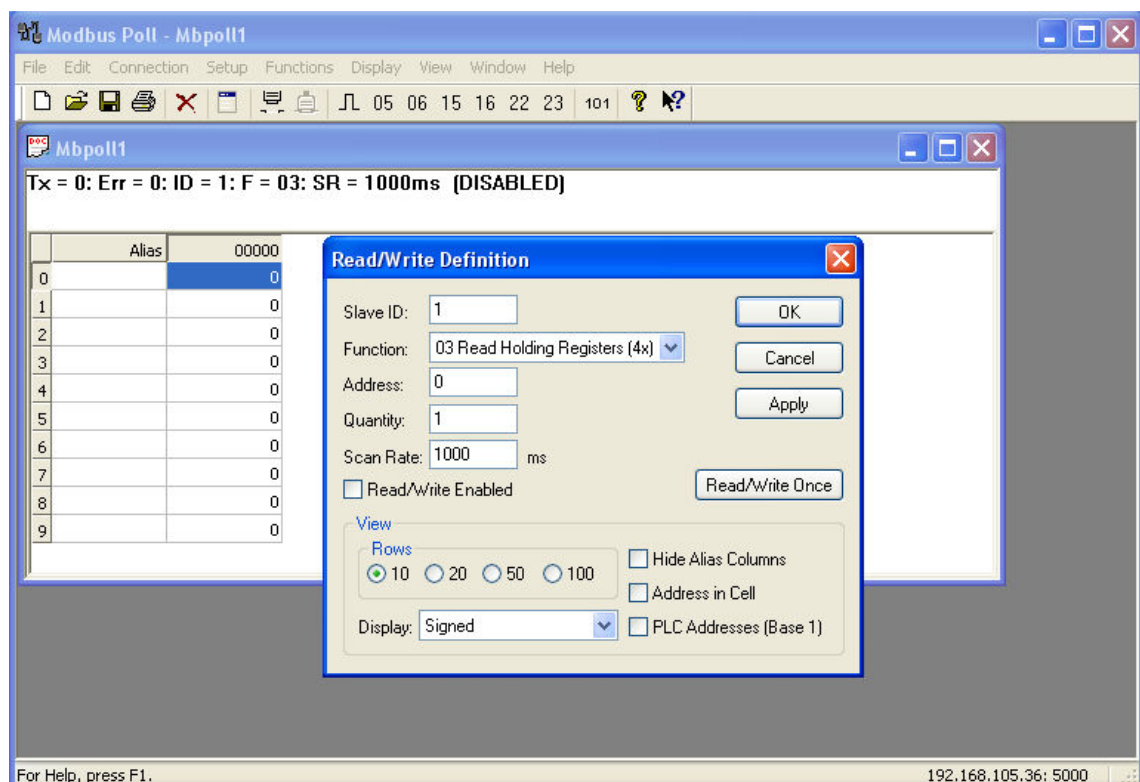


Figura 5.1 - Interface gráfica com o utilizador do programa Modbus Poll.

O Modbus Poll foi utilizado em detrimento de um PLC, por uma questão de facilidade e porque não oferece nenhuma desvantagem face ao PLC, para a finalidade que se pretende. A

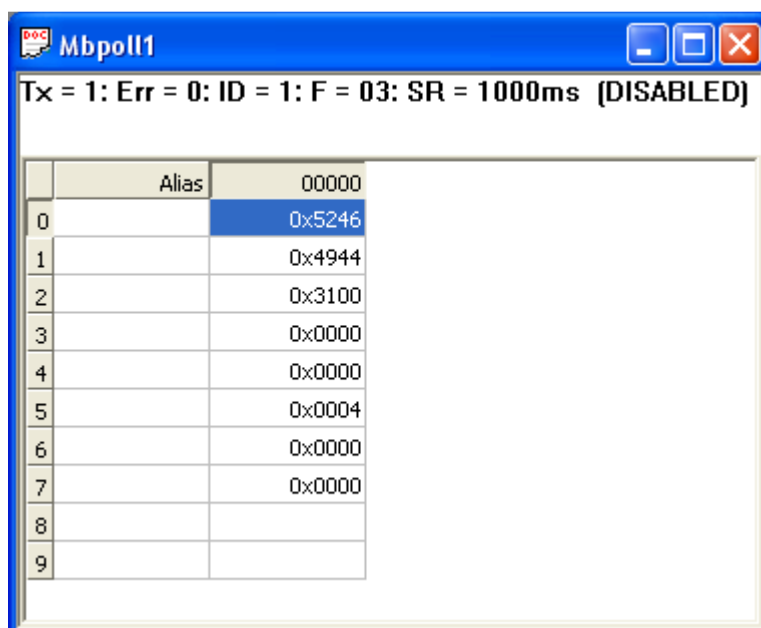
sua interface gráfica com o utilizador é bastante simples e intuitiva como se pode ver na Figura 5.1. Para enviar os comandos Modbus apenas é necessário iniciar a ligação com o escravo, escolher a função, definir o endereço do primeiro registo e quantos registos se pretendem ler ou escrever.

Para a realização dos testes apenas estava disponível um XPort. Assim, não foi possível testar o funcionamento de todo o sistema ligado em conjunto, tendo-se realizado testes separados para cada leitor. O primeiro leitor utilizado foi o de RFID e serão a seguir apresentados alguns dos resultados.

Antes disso, é conveniente referir alguns pontos para melhor compreensão das figuras apresentadas:

- As iniciais “Tx” indicam o número de envios do comando, ou seja, se for igual a 1, quer dizer que o comando já foi enviado uma vez.
- “Err” indica o número de envios que resultaram em erro.
- “F” indica o código da função Modbus. Neste caso só será utilizado o 03 (leitura) e o 16 (escrita).
- Os dados estão em base hexadecimal.

O primeiro teste, apresentado na Figura 5.2, foi ler os registos de 0 a 7. Como se vê, foi realizado com sucesso e retornou os dados que eram esperados. A identificação do leitor continha os caracteres “RFID1”, definidos no ficheiro de texto de configuração, e os restantes registos continham os valores de inicialização.

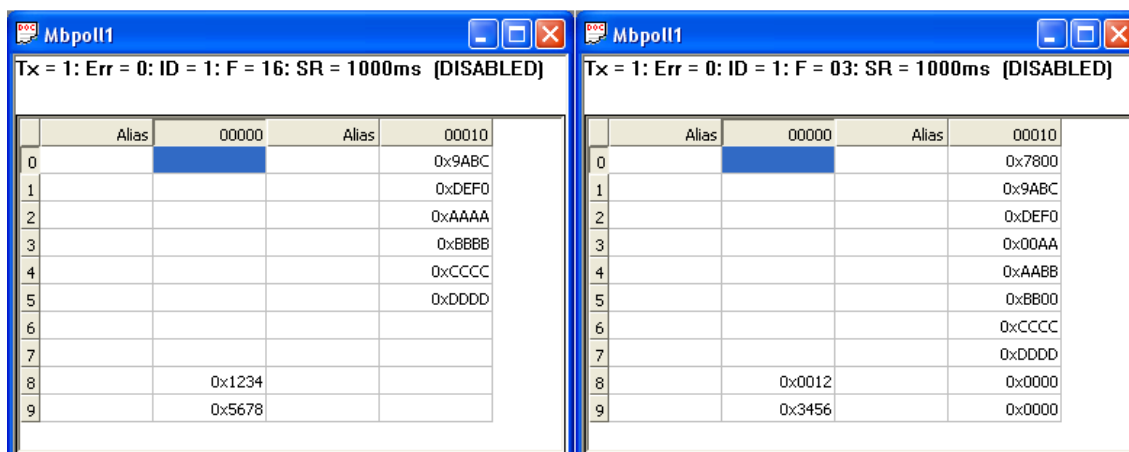


The screenshot shows a window titled 'Mbpoll1' with a status bar at the top displaying 'Tx = 1: Err = 0: ID = 1: F = 03: SR = 1000ms [DISABLED]'. Below this is a table with two columns: 'Alias' and a numerical value. The table contains data for registers 0 through 9. Register 0 has a value of 0x5246, while registers 1 through 9 have values ranging from 0x4944 to 0x0000. The table is as follows:

	Alias	
		00000
0		0x5246
1		0x4944
2		0x3100
3		0x0000
4		0x0000
5		0x0004
6		0x0000
7		0x0000
8		
9		

Figura 5.2 - Leitura dos registos Modbus 0 a 7.

O segundo teste consistiu em escrever e ler dados de uma etiqueta, Figura 5.3. Foram escritos 8 registos (16 bytes) e a seguir lidos 12. Analisando a figura, pode-se concluir que tudo correu como o esperado e as funções de leitura e escrita nas etiquetas funcionam. Note-se que os dados lidos contêm o byte SEC-STATUS, referido no capítulo 4, no início de cada bloco de dados (4 bytes).

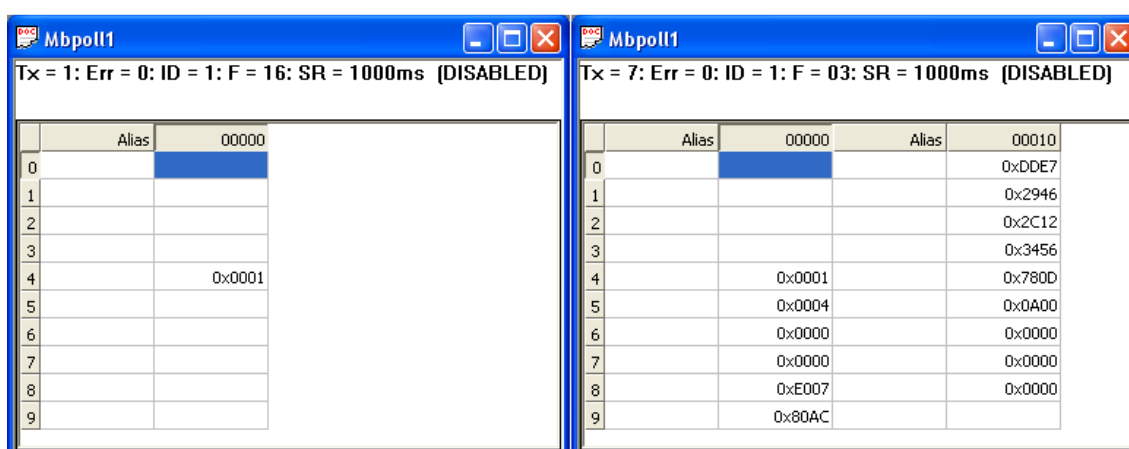


Two screenshots of the Mbpoll1 software interface. The left window shows the configuration: Tx = 1; Err = 0; ID = 1; F = 16; SR = 1000ms (DISABLED). The right window shows the configuration: Tx = 1; Err = 0; ID = 1; F = 03; SR = 1000ms (DISABLED).

	Alias	00000	Alias	00010
0				0x9ABC
1				0xDEFO
2				0xAAAA
3				0xBBBB
4				0xCCCC
5				0xDDDD
6				
7				
8		0x1234		
9		0x5678		

	Alias	00000	Alias	00010
0				0x7800
1				0x9ABC
2				0xDEFO
3				0x00AA
4				0xAABB
5				0xBB00
6				0xCCCC
7				0xDDDD
8		0x0012		0x0000
9		0x3456		0x0000

Figura 5.3 - Escrita e leitura de dados de uma etiqueta a partir do Modbus Poll.

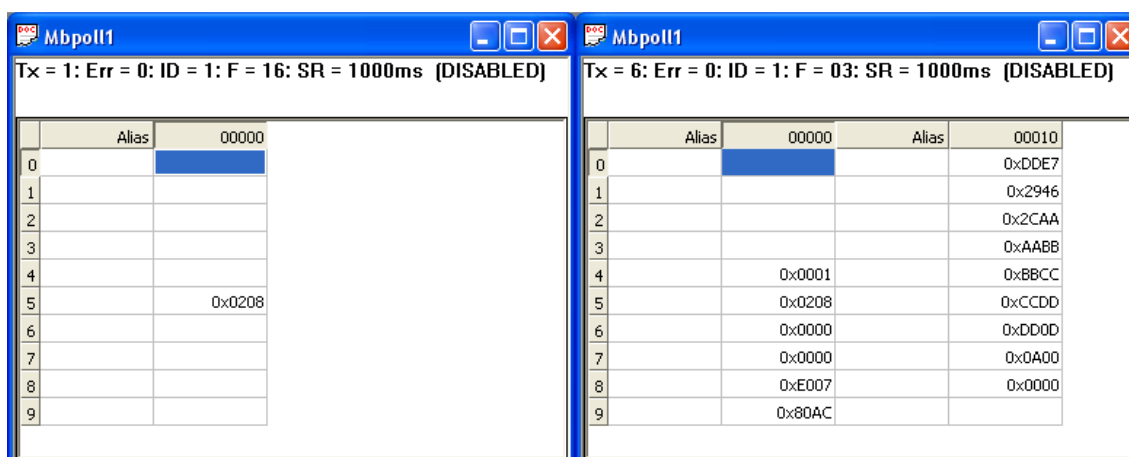


Two screenshots of the Mbpoll1 software interface. The left window shows the configuration: Tx = 1; Err = 0; ID = 1; F = 16; SR = 1000ms (DISABLED). The right window shows the configuration: Tx = 7; Err = 0; ID = 1; F = 03; SR = 1000ms (DISABLED).

	Alias	00000
0		
1		
2		
3		
4		0x0001
5		
6		
7		
8		
9		

	Alias	00000	Alias	00010
0				0xDDE7
1				0x2946
2				0x2C12
3				0x3456
4		0x0001		0x780D
5		0x0004		0x0A00
6		0x0000		0x0000
7		0x0000		0x0000
8		0xE007		0x0000
9		0x80AC		

Figura 5.4 - Funcionamento em Scan Mode a partir do Modbus Poll.



Two screenshots of the Mbpoll1 software interface. The left window shows the configuration: Tx = 1; Err = 0; ID = 1; F = 16; SR = 1000ms (DISABLED). The right window shows the configuration: Tx = 6; Err = 0; ID = 1; F = 03; SR = 1000ms (DISABLED).

	Alias	00000
0		
1		
2		
3		
4		
5		0x0208
6		
7		
8		
9		

	Alias	00000	Alias	00010
0				0xDDE7
1				0x2946
2				0x2CAA
3				0xAABB
4		0x0001		0xBBCC
5		0x0208		0xCCDD
6		0x0000		0xDD0D
7		0x0000		0x0A00
8		0xE007		0x0000
9		0x80AC		

Figura 5.5 - Alteração do registo Modbus 5.

De seguida, testou-se o funcionamento em *Scan Mode*, Figura 5.4. Activou-se o bit correspondente do registo 4 (CMD) e, posteriormente, leu-se desde o registo 4 até ao 18. Mais uma vez, comprovou-se o bom funcionamento e foram lidos os dados esperados. Isto é, os primeiros 8 bytes lidos desde o registo 8 correspondem ao UID da etiqueta, seguidos de um

carácter de separação 0x2C, de dados da etiqueta definidos no registo 5 e dois caracteres de fim de trama 0x0D e 0x0A. Para provar o funcionamento do registo 5, alterou-se o seu valor e fez-se nova leitura, Figura 5.5. Desta vez, os dados lidos da etiqueta foram 8 e a começar no terceiro bloco (0xAAAABBBBCCCCDDDD).

O teste que a seguir se realizou foi efectuar um *reset* à aplicação, Figura 5.6. Para isso, foi escrito no registo 4, o valor 1 no bit 15 e 0 no bit 14 (configuração *default*), e em seguida leu-se desde o registo 0 a 7. Como se vê pela figura, os registos 4 e 5 voltaram a ter os seus valores iniciais, o que significa que, o leitor voltou à configuração inicial.

Mbpoll1		
Tx = 1: Err = 0: ID = 1: F = 16: SR = 1000ms (DISABLED)		
	Alias	
0		00000
1		
2		
3		
4		0x8000
5		
6		
7		
8		
9		

Mbpoll1		
Tx = 1: Err = 0: ID = 1: F = 03: SR = 1000ms (DISABLED)		
	Alias	
0		0x5246
1		0x4944
2		0x3100
3		0x0000
4		0x0000
5		0x0004
6		0x0000
7		0x0000
8		
9		

Figura 5.6 - Comando *reset* a partir do Modbus Poll.

Por fim, na Figura 5.7 pode-se ver o que acontece com alguns erros mais comuns. Se houver uma tentativa de leitura ou escrita para um endereço não definido, é retornado um erro de endereço inválido. Neste caso o registo 200 é um endereço inválido, pois apenas existe um leitor RFID.

Quanto a erros que acontecem na aplicação, por exemplo, tentativa de ler dados de uma etiqueta que não esteja ao alcance do leitor RFID, é retornado um erro do dispositivo (escravo). Quando isso acontece, pode-se ler os registos que guardam os erros para verificar que tipo de erro aconteceu. No canto inferior esquerdo da Figura 5.7, o valor 0xB (11) significa que houve um erro no byte STATUS do do leitor RFID e o valor 0x1 (1) desse byte significa que não há *Transponder*.

No caso de acontecer um erro que retorne um *erro*, como perder a ligação, também foi testado e pode-se ver no canto inferior direito da figura.

Após estes testes, pensa-se estar comprovado o funcionamento da aplicação, pelo menos para o leitor RFID.

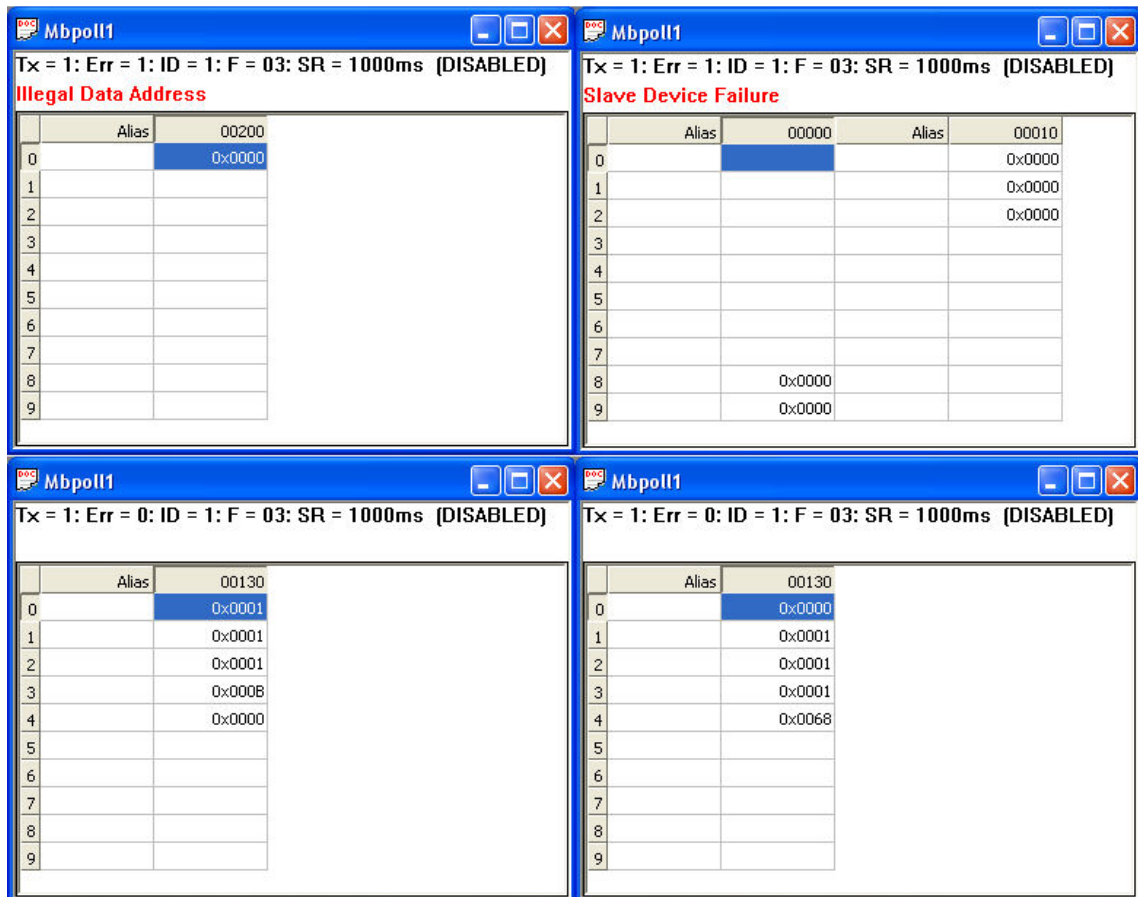
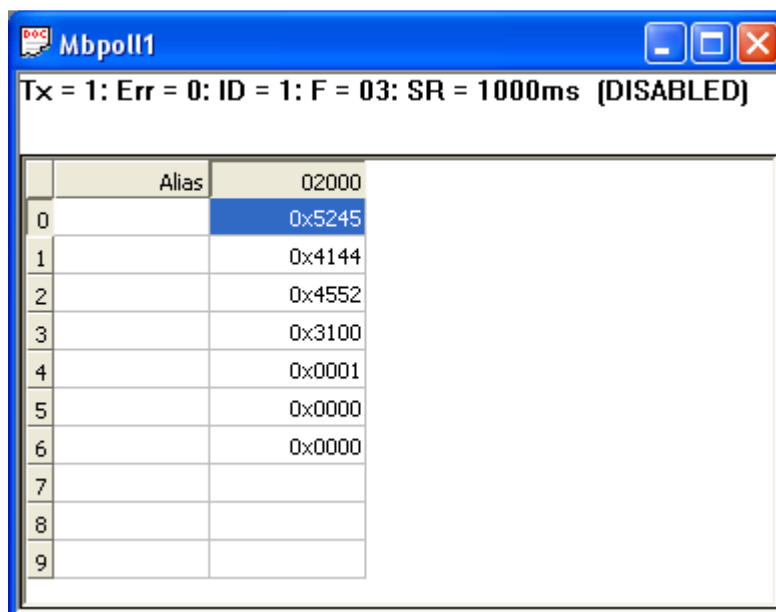


Figura 5.7 - Exemplos de erros Modbus.

5.2. Leitores de códigos de barras e de cartões

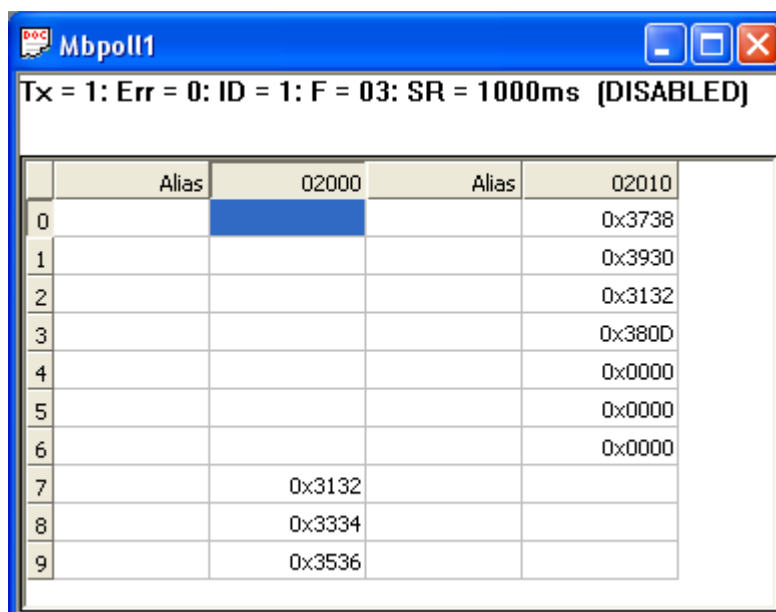
Os testes realizados para estes leitores foram os mesmos pois o seu funcionamento é igual. Para o leitor de cartões começou-se por ler desde o registo 2000 a 2006 e o resultado revelou-se correcto, Figura 5.8. A identificação do leitor corresponde a “READER1”, existe conexão e não ocorreram erros.



	Alias	02000
0		0x5245
1		0x4144
2		0x4552
3		0x3100
4		0x0001
5		0x0000
6		0x0000
7		
8		
9		

Figura 5.8 - Leitura dos registos Modbus 2000 a 2006.

A seguir leu-se um código de barras e posteriormente, leu-se 10 registos a partir da posição 2007, Figura 5.9. O resultado também foi satisfatório, uma vez que, os dados foram lidos correctamente.



	Alias	02000	Alias	02010
0				0x3738
1				0x3930
2				0x3132
3				0x380D
4				0x0000
5				0x0000
6				0x0000
7		0x3132		
8		0x3334		
9		0x3536		

Figura 5.9 - Leitura de um código de barras a partir do Modbus Poll.

O teste que se realizou a seguir foi ler dados quando nenhum código de barras foi lido ainda. Este caso deve provocar um erro e isso foi comprovado como se vê na Figura 5.10. Do lado direito da figura, pode-se verificar que a aplicação deu um erro 0x10 (16) que significa que não existem dados para transmitir.

	Alias	02000	Alias	02010
0				0x0000
1				0x0000
2				0x0000
3				0x0000
4				0x0000
5				0x0000
6				0x0000
7		0x0000		
8		0x0000		
9		0x0000		

	Alias	02000
0		0x5245
1		0x4144
2		0x4552
3		0x3100
4		0x0001
5		0x0010
6		0x0000
7		
8		
9		

Figura 5.10 -

Como o funcionamento do leitor de cartões é idêntico ao leitor de código de barras, apresenta-se na figura seguinte, apenas uma leitura a todos os registos para comprovar que o endereçamento está correcto e que lê, correctamente, dados de um cartão de banda magnética.

	Alias	04000	Alias	04010
0		0x5245		0x3030
1		0x4144		0x3630
2		0x4552		0x3730
3		0x3100		0x3038
4		0x0001		0x3030
5		0x0010		0x3F0D
6		0x0000		0x0000
7		0x3B30		0x0000
8		0x3230		0x0000
9		0x3230		0x0000

Figura 5.11 - Leitura dos registos Modbus 4000 a 4019.

Capítulo 6

Conclusão

Neste capítulo serão apresentadas as conclusões relativas a todo o trabalho desenvolvido e propostas algumas ideias para trabalho futuro.

6.1. Conclusões Gerais

As tecnologias AIDC, que surgiram nos anos setenta, estão, hoje em dia, presentes em quase toda a parte. Utilizadas largamente na indústria, constituem um factor decisivo na competitividade, pois influenciam de forma importante as operações. Estas tecnologias podem ser classificadas em 6 categorias: ópticas, magnéticas, electromagnéticas, cartões inteligentes, de toque e biométricas. O código de barras, tecnologia óptica, é dos sistemas AIDC mais utilizados devido à sua simplicidade e baixo custo. No entanto, o RFID, tecnologia electromagnética, já é também muito procurado devido a outras vantagens como maior capacidade de dados e robustez para ambientes adversos.

Três destas tecnologias, RFID, código de barras e cartões de banda magnética, foram utilizadas para implementar um sistema na linha de fabrico flexível do DEEC, que permitisse a sua utilização a partir do protocolo Modbus.

O estudo do protocolo de comunicações dos três equipamentos permitiu implementar uma API que realiza funções de leitura para cada um deles e de escrita para o leitor RFID. A sua utilização poderá passar não só por esta aplicação como para outras.

Para a implementação do servidor (escravo) Modbus, definiu-se um conjunto de registos, bem como a sua organização, que permitisse implementar as funções inicialmente propostas. Assim, foram realizadas com êxito, as funções de leitura e escrita numa etiqueta, utilização do *Scan Mode* do leitor RFID, alteração das suas configurações, ler dados de códigos de barras e de cartões de banda magnética. Embora a aplicação esteja preparada para ter mais do que um destes equipamentos, não foi possível testar o seu funcionamento, devido ao facto de apenas estar disponível um XPort. É claro que também existem algumas limitações nesta solução. O funcionamento do leitor RFID, por exemplo, apenas está preparado para o modo *Non-Addressed Mode*, o que significa que, se estiverem duas etiquetas ao alcance do leitor, este não permite a leitura ou escrita de dados. A nível de configuração também não foram implementadas muitas opções, originando uma solução pouco flexível, o que nem é uma desvantagem neste caso, pois não se pretende dar liberdade aos alunos de modificar parâmetros do leitor RFID, que possam provocar o mau funcionamento deste. Com esse

intuito, criou-se ficheiros de texto de configuração para ser possível alterar alguns parâmetros, como o endereço IP do leitor (XPort), sem ser necessário compilar o programa de cada vez que isso fosse alterado.

Quanto ao tratamento de erros por parte da aplicação, foi disponibilizado ao utilizador registos onde esses possam estar descritos e o seu funcionamento revelou-se importante, inclusive para a fase de testes.

Em suma, conseguiu-se criar um servidor Modbus que permite aos alunos, no sistema da linha de fabrico flexível do DEEC, utilizar três tipos de tecnologias AIDC com funções de leitura e escrita, de uma forma simples e transparente para eles quanto aos protocolos de comunicação.

6.2. Trabalho Futuro

Embora se tenham cumprido os objectivos propostos a priori, este trabalho pode ser melhorado. Para esta utilização específica, a linha de fabrico flexível do DEEC, pode-se por exemplo:

- Implementar mais funções para a API, disponíveis no leitor RFID.
- Implementar os modos de acesso à etiqueta *Addressed Mode* e *Selected Mode*.
- Disponibilizar mais parâmetros de configuração do leitor RFID.
- Acrescentar novas tecnologias.

No entanto, a aplicação desenvolvida também pode ser utilizada noutros sistemas que comuniquem por Modbus TCP. Como proposta para trabalhos futuros, pode-se adaptar esta aplicação a esses sistemas que necessitem de tecnologias AIDC.

Referências

- [1] M. P. Groover, "Automation, production systems, and computer-integrated manufacturing", Upper Saddle River: Pearson Prentice Hall, 2008.
- [2] K. Michael, "Trends in the Selection of Automatic Identification Technology in Electronic Commerce Applications," em *Trends in the selection of automatic identification technology in electronic commerce applications*, Chile: N. Cerpa & P. Bro 2003, pp. 135-152.
- [3] Keyence, "Bar Code Reader Technical Guide", Disponível em http://www.keyence.com/topics/barcode/bar_code/t_guide.php. Acesso em Junho de 2009.
- [4] AIM, Disponível em <http://www.aimglobal.org/>. Acesso em Junho de 2009.
- [5] R. World, Disponível em <http://www.rfidworld.com/>. Acesso em Junho de 2009.
- [6] A. R. Solutions, Disponível em <http://www.atlasrfid.com/rfid-vs-barcode.asp>. Acesso em Junho de 2009.
- [7] S. Preradovic e N. C. Karmakar, "RFID Transponders - A Review," em *Electrical and Computer Engineering, 2006. ICECE '06. International Conference on*, 2006, pp. 96-99.
- [8] G. Whyte, "Evaluating the Effect of EMI in Transceivers (Readers) Used to Identify Animals Electronically," 2008.
- [9] "Fundamental Operating Principles," em *Fundamentals and Applications in Contactless Smart Cards and Identification*, 2nd edition ed: Wiley & Sons LTD, 2003, pp. 29-59.
- [10] "Cartões Magnéticos", Disponível em <http://www.deetc.isel.ipl.pt/sistemasdigitais/MC/bibliografia/CartoesMagneticos.pdf>. Acesso em Junho de 2009.
- [11] Microscan, Disponível em <http://www.microscan.com/en-us/TrainingAndResources/AutoID.aspx>. Acesso em Junho de 2009.
- [12] A. Pereira e C. Poupá, "Como escrever uma tese monografia ou livro científico usando o Word 0008", Lisboa: Edições Sílabo, 2004.
- [13] K. Michael, "Innovative Automatic Identification and Location-Based Services", Idea Group Inc, 2009.
- [14] AIM, "Optical Character Recognition (OCR)," 2000.
- [15] S. Shepard, "RFID: Radio Frequency Identification ", McGraw-Hill Companies, Inc, 2005.