INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia Eletrotécnica Energia e Automação

Integração e supervisão de múltiplas redes de automação

Paulo Alexandre Ricardo Costa

(Licenciado em Engenharia Eletrotécnica)

Dissertação para a obtenção do grau de Mestre em

Engenharia Eletrotécnica – Ramo de Automação Industrial

Orientadores:

Prof. Doutor João Carlos Pires Palma

Prof. Doutor Armando José Leitão Cordeiro

Júri:

Presidente: Prof. Doutor Luís Manuel dos Santos Redondo

Vogais: Prof.ª Doutora Mafalda Maria Morais Seixas

Prof. Doutor Armando José Leitão Cordeiro

Setembro de 2016

Dissertação realizada sob orientação de

Doutor João Carlos Pires Palma
Professor Coordenador da Área Departamental de
Engenharia Eletrotécnica Energia e Automação
Instituto Superior de Engenharia de Lisboa
e
Doutor Armando José Leitão Cordeiro
Professor Adjunto da Área Departamental de
Engenharia Eletrotécnica Energia e Automação
Instituto Superior de Engenharia de Lisboa

Resumo

O tema e procedimento desta dissertação é baseado na premissa de integração de tecnologias distintas presentes em redes de automação.

Serão estudadas as redes existentes isoladas de diversos fabricantes, estas utilizando tecnologias distintas que não são diretamente compatíveis entre si.

A solução analisada nesta dissertação é a integração das redes existentes através da compatibilização das redes tanto a nível de *hardware*, nomeadamente em termos de interfaces, conversores e meios físicos, como a nível de *software* através da compatibilização dos protocolos de comunicação. A compatibilização em termos de *software* é conseguida através da utilização de um porto de comunicação de um autómato S7-200 livre de protocolo, modo *Freeport*. Com a utilização deste modo é possível configurar e programar o protocolo pretendido. Desta forma foi possível receber, interpretar e encaminhar mensagens segundo o protocolo MODBUS RTU, capacidade que o autómato originalmente não dispõe visto ser um protocolo utilizado principalmente por outro fabricante.

A integração em uma só rede modernizada permite então o acesso, através de uma rede *Ethernet*, aos equipamentos existentes desatualizados que originalmente não dispunham dessa capacidade.

A par do acesso aos equipamentos, através de uma rede integrada, é também constituído um sistema de supervisão SCADA utilizando as mais recentes aplicações informáticas. Este sistema supervisiona a rede através de comandos e dados recolhidos, permitindo ainda a gestão dos mesmos.

Palavras chave: Automação Industrial, Rede integrada, *Freeport*, MODBUS RTU *Ethernet*, SCADA

Abstract

The theme and procedure for this thesis is based on the premisse of integration of distinct technologies present in automation networks.

The existing isolated automation networks will be analized, these use distinct technologies and as such are not directly compatible.

The solution studied in this work is the integration of the existing networks through compatibilization of hardware, interfaces, converters, and physical media, and software through compatibilization of the communication protocols. The compatibilization of software is accomplished through the use of a free S7-200 communication port, Freeport mode, the usage of this mode allows the configuration and programming of the intended communication protocol. This way it was possible to receive, decode and to forward messages in MODBUS RTU protocol.

The integration in one single modernized network allows access to modern ethernet networks from existing and outdated equipment that originally didnt have that capacity.

With this access to the equipments through the integrated network its also created a SCADA supervision system using the latest technologies, this system supervises the network through commands and gathered data, also allowing data management.

Keywords:*Industrial Automation, Integrated networks, Freeport*, MODBUS RTU *Ethernet, SCADA*

Agradecimentos

Pretendo dedicar esta página final de agradecimentos às pessoas que comigo percorreram o longo caminho da realização de uma dissertação sobre um conceito tão amplo como a integração de redes de automação.

Desejo em especial agradecer à minha família e a todos os sacrifícios que fizeram pela minha ausência ou indisponibilidade e por tudo o que tiveram de prescindir para que este trabalho fosse possível.

Assim como desejo agradecer em particular ao meu orientador, Professor Doutor Armando José Leitão Cordeiro, todo o esforço, trabalho e dedicação com que me apoiou e me guiou ao longo deste último ano.

Agradeço também a todo o Instituto Superior de Engenharia de Lisboa, aos seus quadros, docentes, não docentes, alunos e amigos todos os conhecimentos, convivência e experiência que me proporcionaram ao longo dos últimos anos na conclusão da minha licenciatura e percurso académico do mestrado culminando na elaboração desta dissertação para obtenção do grau de mestre, sem vós este percurso não seria a experiência enriquecedora que considero.

Por fim, mas não menos importante, desejo agradecer às Forças Armadas Portuguesas, a todos os que delas fazem parte, todo o apoio, incentivo e mentalidade que sempre me incutiram e me leva mais além.

Lista de siglas

API	*Application Programming Interface*
APM	*Alternate Pulse Modulation*
AR	*Address Register*
AS-i	*Actuator Sensor Interface*
ASCII	*American Standard Code for Information Interchange*
BA	*Bitwise Arbitration*
CAN	*Controller Area Network*
CBA	*Component Based Automation*
CD	*Collision Detection*
CRC	*cyclic redundancy check*
CSMA	*Carrier Sense Multiple Access*
CSIA	*Control System Integrators Association*
DCS	*Decentralized Control Systems*
DMZ	*Demilitarized Zone*
DP	*Decentralized Peripherals*
EIA	*Electronic Industries Association*
EMI	*Electromagnetic Interference*
EN	*European Standard*
FCS	*Frame Checking Sequence*
FIP	*Factory Instrumentation Protocol*
FISCO	*Fieldbus Intrinsically Safe Concept*
FMS	*Fieldbus Message Specification*
FTP	*File Transfer Protocol*
GMT	*Greenwich Mean Time*
GSM	*Global System for Mobile Communications*

HMI	*Human Machine Interface*
HSA	*Highest Station Address*
HTTP	*Hypertext Transfer Protocol*
IEC	*International Electotechnical Commission*
Imáx	Corrente máxima
IP	*Internet Protocol*
IS	*Intrinsically Safe*
ISA	*Instrumentation, Systems and Automation Society*
ISO	*International Standards Organization*
ISM	*Industrial Scientific and Medical*
IoT	*Internet of Things*
I/O	*Input/Output*
I/O Port	*Input/Output Port*
LED	*Light Emitting Diode*
MAC	*Medium Access Control*
MIL-STD	*Military Standard*
MMF	*Multi Mode Fiber*
MPI	*Multi-Point Interface*
ms	milissegundo
MTU	*Master Terminal Unit*
NRZ	*No Return to Zero*
ODBC	*Open DataBase Conectivity*
OPC	*Open Platform Comunications*
OPC DA	*OPC Data Access*
OSI	*Open Systems Interconnected*
PA	*Process Automation*
PC	*Personal Computer*
PCS	*Process Control System*
PDU	*Protocol Data Unit*
PLC	*Programable Logic Controler*
RFI	*Radio Frequency Interference*
RS	*Recomended Standard*
RTDB	*Real Time Database*

RTU	*Remote Terminal Unit*
SCADA	*Supervisory Control and Data Acquisition*
SDU	*Sevice Data Unit*
SGML	*Standard Generalized Markup Language*
SMF	*Single Mode Fiber*
ST	*Sructured Text Language*
STL	*Statement List*
S/FTP	*Shielded / Foiled Twisted Pair*
SMS	*Short Message Service*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
UV	Ultravioleta
U/UTP	*Unshielded Twisted Pair*
VBA	*Visual Basic for Application*
Vmax	Tensão máxima
VOIP	*Voice over IP*
XML	*Extensible Markup Language*
µs	microssegundo

Índice

Índice de figuras

Capítulo 1 Introdução

Resumo: O presente capítulo apresenta uma introdução ao tema da dissertação assim como o enquadramento e motivação científica para a elaboração da mesma, prosseguindo com os objetivos a concretizar e uma breve descrição da organização do trabalho.

1.1 Enquadramento e motivação

A importância da tecnologia na vida humana é um fato inquestionável, a tecnologia é um dos aspetos mais impulsionadores da economia e responsável por inúmeros contributos para a melhoria da qualidade de vida do homem. Vivemos atualmente na era da Microeletrónica e das Tecnologias de Informação onde termos como a "integração" e a "interligação" são consideradas palavras-chave em muitas áreas de atividade e são muitas vezes sinónimos de sucesso nos negócios.

Os sistemas automáticos, que ocupam lugar de destaque nos objetivos de produtividade e competitividade de muitas indústrias, são provavelmente os que mais têm progredido devido aos avanços registados na tecnologia tendo-se atingido atualmente um grau de sofisticação e complexidade nunca antes visto. Estes sistemas são muitas das vezes compostos por subsistemas provenientes de diferentes fabricantes, com diferentes soluções, aos quais se exige que desempenhem, para além das funções normais de controlo de processos, funções de segurança com elevada fiabilidade. Esta evolução nos sistemas automáticos tem ocorrido a todos os níveis dentro dos processos industriais, desde os controladores com elevada capacidade de diagnóstico de falhas até aos avançados sistemas de supervisão passando pelas redes de comunicação digital e por sensores e atuadores com funções programáveis.

Esta evolução dos sistemas automáticos é naturalmente um processo contínuo motivado por pequenos avanços num sistema globalizado em constante mudança e como em quase todos os sistemas em evolução é necessário dotá-los com capacidade de integração de novos sistemas e tecnologias. Este aspeto é fundamental dada a impossibilidade de implementação de novos sistemas automáticos a cada evolução tecnológica principalmente devido aos custos envolvidos. É nesta área que se pretende conduzir esta dissertação de mestrado, mostrando por um lado o estado da arte em termos de equipamentos industriais com novas tecnologias bem como as suas tendências e por outro a possibilidade de interligação e de integração desses mesmos equipamentos em sistemas automáticos. Para tal, desenvolveu-se uma solução de interligação de vários equipamentos existentes nos laboratórios de Automação e de Pneumática através de redes de comunicação digital, que apesar de estes não utilizarem as tecnologias mais recentes, permitem mostrar as potencialidades da sua integração. Esta solução visa tornar compatíveis equipamentos de fabricantes distintos quer seja pela sua natureza construtiva

ou porque não existiu disponibilidade financeira para aquisição de outros dispositivos específicos. Além disso este trabalho vai ao encontro de um dos objetivos principais da disciplina de Redes de Automação e Supervisão do Mestrado em Engenharia Eletrotécnica que é dotar os alunos de conhecimentos teóricos e práticos em redes de comunicação e sistemas de supervisão utilizados atualmente em Automação Industrial. Deste modo, justifica-se plenamente a realização desta dissertação de Mestrado onde houve a necessidade de efetuar um estudo aprofundado do funcionamento e potencialidades dos equipamentos e criar medidas alternativas para colocar estes equipamentos a comunicar. Considera-se assim que esta dissertação apresenta valor técnico-científico relevante para a obtenção do grau de Mestre em Engenharia Eletrotécnica.

A opção por este tema para a realização da dissertação deve-se também em grande parte a uma preferência pessoal por esta área da engenharia eletrotécnica, aliada ao gosto pelo desenvolvimento de *hardware* e *software* para aplicações de automação que envolvam redes de comunicação digital.

1.2 Objetivos

O principal objetivo desta dissertação consistiu em desenvolver uma solução que possibilitasse a interligação de diversos equipamentos industriais provenientes de diferentes fabricantes.

A solução mais fácil para resolver este problema passaria pela aquisição de novos equipamentos ou pela adição de módulos de comunicação digital compatíveis com os equipamentos existentes. No entanto, esta solução não iria permitir explorar outros aspetos dos equipamentos nem traria grande valor acrescentado pois consistiria basicamente numa simples ligação sem qualquer complexidade adicional. Assim, para cumprir o objetivo principal desta dissertação foi necessário explorar e desenvolver alguns objetivos parciais, tais como:

- Conhecer os equipamentos e as suas linguagens de programação de forma aprofundada;
- Estudar e compreender aprofundadamente diversas redes de campo e protocolos usados em Automação Industrial, nomeadamente a rede Profibus-DP, rede AS-i, rede *Ethernet*, protocolo MODBUS RTU e protocolo MODBUS TCP;

- Implementar um dos protocolos de comunicação mais conhecidos, nomeadamente o MODBUS RTU, usando as linguagens de programação previstas para os autómatos programáveis;
- Conceber uma forma possível de estabelecer comunicações digitais entre estes equipamentos de forma segura e de acordo com os requisitos de cada rede e protocolo; sem esquecer que é necessário compatibilizar o suporte físico utilizado assim como as interfaces de acesso ao meio físico de modo a que utilizem o mesmo tipo de sinais, níveis de tensão e codificações;
- Resolver problemas de sincronização entre diferentes redes de comunicação e protocolos de modo a evitar ao máximo perda de informação;
- Desenvolver um sistema de supervisão capaz de comunicar com todos estes equipamentos simulando a sua aplicação a um sistema real.

1.3 Organização e convenções

Esta dissertação divide-se em cinco capítulos distintos. No capítulo 1 é feita uma introdução ao tema da dissertação assim como o seu enquadramento e motivação científica para a elaboração da mesma. No capítulo 2 será então apresentado o estado da arte, proporcionando ao leitor uma visão global da tecnologia atual e suas tendências futuras. No capítulo 3 apresenta-se então a solução adotada para a compatibilização das redes em *hardware* e *software* com a explicação da situação existente e as necessidades de compatibilização, serão então apresentados os equipamentos instalados, as suas configurações e programação necessárias. O trabalho desenvolvido nesta dissertação de Mestrado apresenta também algumas limitações que serão descritas nesse capítulo.

A solução de integração das redes culmina com a apresentação e descrição do sistema de supervisão desenvolvido no capítulo 4. Neste capítulo serão então apresentados conceitos base relativos aos sistemas de supervisão assim como a solução desenvolvida para o envio e receção de mensagens para cada equipamento presente na rede integrada. Adicionalmente é apresentado um sistema de supervisão que pretende simular um sistema de supervisão real em uma instalação industrial. Por fim no capítulo 5 serão apresentadas as conclusões finais gerais e particulares.

Com o objetivo de facilitar a leitura e compreensão deste documento foram utilizadas convenções que se descrevem de seguida:

- Sempre que exista correspondência entre as grandezas presentes nas equações utilizam-se unidades do Sistema Internacional (S.I.) de unidades de medida. Nos múltiplos e submúltiplos dessas unidades as respetivas abreviaturas;
- Recorreu-se, sempre que possível, a conceitos presentes na Língua Portuguesa. Em determinados casos os conceitos surgem acompanhados pela designação na Língua Inglesa, colocada entre parêntesis e em itálico, para que a compreensão do seu significado seja mais fácil e menos ambígua;
- As siglas de termos, conceitos e equipamentos serão utilizadas na Língua Inglesa por forma a garantir familiaridade com as siglas utilizadas;
- A numeração de equações, figuras e tabelas foi efetuada de forma sequencial ao longo de cada capítulo, referenciadas por dois números separados por um ponto. O primeiro número refere-se ao capítulo e o segundo à sequência ordenada dentro do capítulo;
- O conjunto de todas as referências bibliográficas foi ordenado pela ordem em que surgem ao longo da dissertação. A citação das referências bibliográficas ao longo dos diversos capítulos foi realizada da forma que normalmente aparece em publicações científicas, ex.: [1]. Neste tipo de citação, o valor contido dentro dos parêntesis representa o número da referência bibliográfica.

Capítulo 2

Estado da Arte

Resumo: Este capítulo apresenta uma síntese do estado da arte relativa a controladores industriais, bem como algumas das redes de Automação e protocolos mais comuns, evidenciando-se as suas principais características. Mostra ainda as mais recentes evoluções e tendências nesta área de engenharia.

2.1 Evolução da automação industrial

A automatização de sistemas e o controlo de processos foi sempre um objetivo perseguido pela humanidade como forma de libertar o homem de tarefas repetitivas, árduas e perigosas. Embora formalmente se considerar que a automação, enquanto parte integrante de processos industriais, só teve início em meados de 1800, são diversos os documentos históricos que descrevem sistemas de controlo automáticos desenvolvidos pelo homem em muitas civilizações antigas [1].

O desenvolvimento das ciências e o consequente surgimento de novas tecnologias permitiram ao longo do último século criar meios para aumentar gradualmente a quantidade de trabalho desenvolvido pelos humanos, contribuindo para aumentar simultaneamente a eficiência na produção industrial (através da redução dos consumos de energia e do desperdício de matérias-primas) e o aumento da qualidade dos produtos finais. Para este desenvolvimento muito contribuiu o surgimento da 2ª Guerra Mundial, assim como a corrida ao espaço registada na segunda metade do século XX. Durante estes períodos foram alcançados importantes desenvolvimentos tecnológicos em eletrónica e em redes de comunicação, nomeadamente com o desenvolvimento e aperfeiçoamento dos computadores digitais e microcontroladores que viriam mais tarde a dar lugar ao surgimento dos primeiros Controladores Industriais, Controladores Lógicos Programáveis ou PLC (*Programable Logic Controler*).

De acordo com [1], o primeiro PLC introduzido no mercado, especificamente concebido para fins industriais, foi o MODICON (*MOdular DIgital CONtroller*) modelo 084 em 1968, aplicado na empresa *Bryant Chuck and Grinder* em Springfield, Vermont, Estados Unidos da América. Este modelo levou os conceitos de robustez e fiabilidade a um nível sem precedentes para um microcomputador, não tinha botão para ligar/desligar, não produzia ruído, não tinha partes propensas a desgaste nem ventoinhas para arrefecimento. Este equipamento permitia facilitar a reconfiguração e a substituição de parte dos circuitos de comando a relés existentes através de uma linguagem de programação gráfica que apresentava muitas semelhanças com a simbologia já conhecida pelos engenheiros e técnicos fabris, a linguagem de contactos ou linguagem *LADDER*. O PLC tornou-se então o equipamento de excelência utilizado para controlar processos industriais.

Inicialmente o controlo dos processos industriais com recurso ao PLC baseava-se em soluções centralizadas para onde eram encaminhados todos os sinais binários e analógicos

do processo, este desenvolvimento inicial designou-se por controlo de processo (PCS - *Process Control System*).

Com a evolução natural dos processos industriais tornou-se evidente que seria necessário criar mecanismos que permitissem responder à elevada complexidade dos automatismos e ao mesmo tempo resolver problemas de interferências eletromagnéticas nos sinais binários e (principalmente) analógicos que percorrem dezenas ou centenas de metros dentro das instalações industriais (e respetivos custos inerentes) como resultado das emissões dos equipamentos instalados. Uma parte significativa destes problemas foi mitigada com o desenvolvimento de estruturas descentralizadas (*DCS – Decentralized Control Systems*), dividindo os sistemas complexos e centralizados com longas cablagens em subsistemas mais simples e mais perto dos processos com interligação entre si através de redes de comunicação digital orientadas para o ambiente industrial, conhecidas como redes de campo (*fieldbus*) (ver fig 2.1) [2].

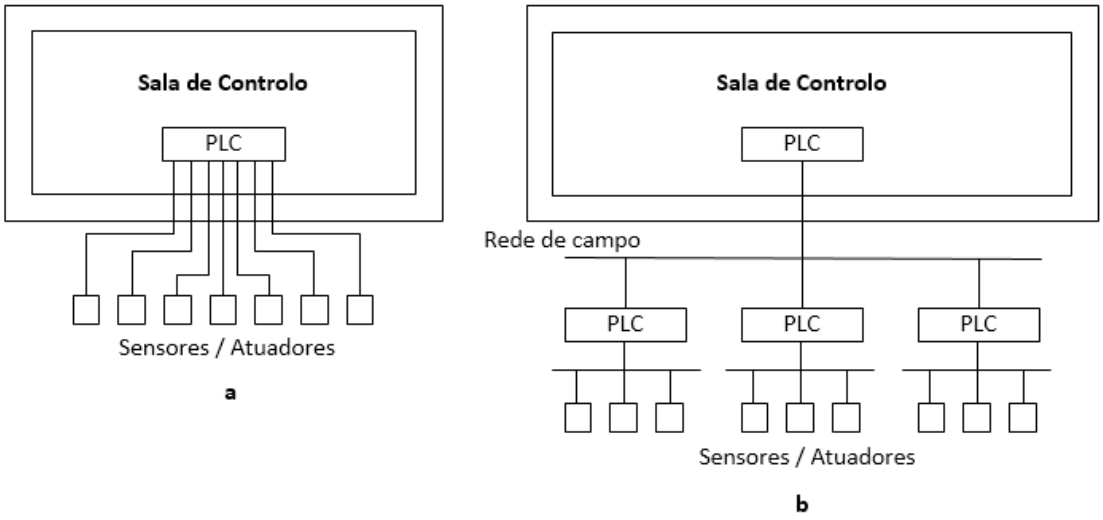

Figura 2.1 a: Estrutura tipo PCS; b: Estrutura do tipo DCS

A necessidade de transmitir informação digital entre equipamentos beneficiou naturalmente da evolução dos computadores digitais e das redes informáticas, as quais postas em prática desde há muito, quer através de comunicação paralela quer de comunicação série.

Na indústria as redes de comunicação digital série começaram a desenvolver-se a par dos autómatos programáveis, sensivelmente a partir de 1975. O primeiro caso relevante terá sido lançado nessa fase inicial pela firma americana Cutler-Hammer com o nome

Directrol. As redes Genius I/O da General Electric, Interbus-S da Phoenix e Sensoplex da Turck surgiram entre 1985 e 1987 e ainda hoje são usadas, o mesmo acontecendo com a rede Arcnet que viria a migrar das aplicações de escritório para o ambiente industrial [3].

2.2 As redes de campo em automação

Do mesmo modo que os controladores industriais derivaram dos microcontroladores também as redes de automação ou redes industriais foram inspiradas nas já existentes redes informáticas, tendo sido desenvolvidas para atender às especificidades próprias de um ambiente industrial. As redes informáticas tendo sido desenhadas para a comunicação entre computadores e para ambientes de escritórios e domésticos não dispunham das características necessárias para serem diretamente aplicadas aos controladores industriais, tais diferenças residiam principalmente nos tempos de comunicação, no determinismo, na segurança, no controlo de erros e na robustez das cablagens a ambientes industriais agressivos em termos físicos e em termos de radiação eletromagnética [4].

Tabela 2.1 Principais diferenças entre uma rede industrial e as redes comerciais [4]

	Rede Industrial	**Rede Comercial**
Função primária	Controlo de equipamentos	Transferência de informação
Aplicação	Industrial e processos	Empresarial e doméstico
Hierarquia	Vários níveis	Poucos níveis
Gravidade das falhas	Elevada	Baixa
Fiabilidade	Elevada	Moderada
Tempos de comunicação	250µs – 10ms	>50ms
Determinismo	Elevado	Não necessário
Tipo de dados	Pequenas e médias mensagens	Grandes mensagens
Ambiente	Agressivo	Limpo

Estas diferenças implicaram então a criação de estruturas físicas novas, protocolos novos, interfaces de comunicação para os controladores assim como um conjunto de

especificações e definições próprias para as comunicações digitais em automação, as denominadas redes de campo.

As redes de campo encontram-se essencialmente divididas em três níveis:

- Redes de nível alto (supervisão) para a transmissão de grandes quantidades de informação com tempos de resposta na ordem das centenas de ms;
- Redes de nível intermédio (controlo) para a interligação de controladores e dispositivos com tempos de resposta na ordem das dezenas de ms;
- Redes de nível baixo (processo) para a interligação de sensores e transdutores com tempos de resposta na ordem das dezenas de μs;

Uma rede de campo pode integrar um outro nível de rede mais baixo para funções de segurança, esse nível implica tempos de resposta muito curtos e prioridade no processamento e na utilização da rede.

No nível baixo devem ser redes que permitam diversos tipos de mensagens, tanto por eventos como por atualizações periódicas, ainda com mensagens curtas e tempos de transmissão baixos no caso de ser necessário um controlo em tempo real.

Ao nível intermédio são redes com velocidades médias que permitem a interligação de controladores sendo utilizadas para transferência de dados e parâmetros.

No nível alto já se permite a agregação de dados e mensagens mais longas com tempos de resposta nas centenas de milissegundos, este é o nível onde estão situados normalmente os sistemas de supervisão para visualização e controlo de alto nível, o controlo direto deve ser efetuado no nível intermédio.

Uma rede de nível alto pode ainda ser conectada a uma rede de empresa, por exemplo uma rede *Ethernet* comercial, para controlos estatísticos e empresariais, funcionando na ordem das centenas de milissegundos, em ambientes limpos e sem necessidades tão exigentes. Por questões de segurança da rede de campo, esta conexão deve ser separada por uma zona de proteção constituída por *firewalls* através da qual apenas passará informação pré estabelecida, esta zona é usualmente designada por DMZ (*Demilitarized Zone*) [5].

Tal composição implica então a criação de vários tipos de redes, protocolos e *hardware* para constituírem uma solução completa de ligação em rede.

Tabela 2.2 Solução normalmente adotada para uma rede dentro de uma fábrica e empresa operando em conjunto.

<table>
<tr><td rowspan="2">Empresa</td><td rowspan="2"></td><td>Empresa</td><td rowspan="2">Rede Comercial</td></tr>
<tr><td>Planeamento e Logística</td></tr>
<tr><td colspan="2">DMZ (Demilitarized Zone)</td><td colspan="2">Proteção</td></tr>
<tr><td rowspan="3">Zona de Manufatura</td><td rowspan="4">Rede de Campo</td><td>Operações</td><td rowspan="2">Nível Alto</td></tr>
<tr><td>Supervisão</td></tr>
<tr><td>Controlo</td><td>Nível Intermédio</td></tr>
<tr><td>Zona de Segurança</td><td>Processo</td><td>Nível Baixo</td></tr>
<tr><td></td><td>Segurança</td><td></td></tr>
</table>

2.3 Fabricantes e protocolos proprietários

A distinção por diversos níveis de rede e a diversidade de características relevantes para cada um deles levou ao surgimento de diversos protocolos proprietários cada um desenhado com a visão do fabricante para a implementação de uma rede mais eficaz para os seus próprios produtos em cada um dos níveis.

Surgiram ainda diversas empresas novas com produtos e protocolos adaptados para solucionarem problemas específicos, tal levou ao surgimento de inúmeros protocolos para os mais diversos tipos de aplicações e níveis de rede mas na sua generalidade incompatíveis entre si, tal acabou por criar indiretamente zonas de monopólio pois uma indústria ao aplicar equipamentos de um determinado fabricante teria de utilizar as redes e protocolos proprietários do mesmo fabricante, o mesmo se aplicando quando existisse a necessidade de expansão da instalação.

Estas zonas de monopólio forçado pela incompatibilidade levaram, nas décadas de 1970/80, à procura por uma uniformização dos protocolos utilizados liderada pela ISA (*Instrumentation, Systems and Automation Society*) e pela IEC (*International Electotechnical Commission*).

Essa procura por uniformização embora não conseguindo um consenso geral resultou, em 1978, na publicação de recomendações de normalização (*recomended standards*) pela EIA (*Electronic Industries Association*) para a especificação de interfaces

de comunicação série tais como as RS232, RS422 e RS485 e pela ISO (*International Standards Organization*) do modelo OSI (*Open Systems Interconnected*) através da norma ISO 7498 que especifica um modelo de camadas (*layers*) sobre as quais os protocolos de comunicações devem ser desenvolvidos [3].

Apesar de alguns aspetos caminharem para a uniformização ainda existe um longo caminho a percorrer em relação a protocolos de comunicação aplicáveis à generalidade dos equipamentos, desde os controladores e sensores, aos ambientes e às exigências industriais. No presente estão a ser desenvolvidas soluções com base no conceito de *Ethernet* Industrial (*Industrial Ethernet*) que se apresentam como uma possível solução futura. Como por exemplo o consórcio PROFINET liderado por empresas como a Siemens, ABB, Bosch e Phoenix Contact.

2.4 Modelo OSI

O modelo é composto por sete camadas de referência em que em cada camada é processado um aspeto específico da comunicação (a transmissão e receção de uma mensagem completa). Cada camada processa os dados recebidos e encaminha a mensagem para uma camada superior ou inferior, as três camadas mais baixas especificam o modelo de rede e as camadas superiores o modelo de aplicação.

Tal modelo permite elaborar protocolos de comunicação de uma forma estruturada e organizada, com funções específicas a cada nível, sendo ainda possível que protocolos globalmente distintos possam ser constituídos por algumas camadas idênticas e serem compatíveis a esse nível, por exemplo a primeira camada, a camada física que especifica o meio físico para a transmissão, pode definir um tipo de cabo ou terminais passiveis de serem utilizados por diversos protocolos diferentes, um exemplo disso é a rede DeviceNet que utiliza uma camada de aplicação própria mas tem como base a rede CAN (*Controller Area Network*) nas camadas mais baixas do modelo [3].

Tabela 2.3 Camadas do modelo OSI

Modelo OSI			
Camada		**PDU**	**Função**
7	Aplicação	Dados	API de alto nível, partilha de recursos, acesso remoto a ficheiros e terminais virtuais.
6	Apresentação		Transferência de dados entre a rede e a aplicação, inclui encriptação e compressão de dados.
5	Sessão		Gestão da comunicação entre dois nós, sincronização e deteção de falhas.
4	Transporte	TCP/UDP	Controlo de transmissão, reconhecimento, multiplexagem e controlo de congestionamentos.
3	Rede	Pacotes	Encaminhamento entre nós da mesma rede, controlo de trafego, *routing* e endereçamento.
2	Ligação	Tramas	Métodos de acesso, composição de tramas.
1	Meio Físico	*Bits*	Transmissão e receção de bits através de meio físico.

O funcionamento de um protocolo de comunicações baseado no modelo OSI tem por base a transmissão de pacotes de dados entre equipamentos que utilizem o mesmo processamento em todas as camadas utilizadas do modelo. Nem todos os protocolos necessitam utilizar todas as camadas do modelo, por questões de simplicidade ou funcionalidade existem protocolos que apenas implementam duas ou três camadas do modelo, sendo protocolos com menos funcionalidades e complexidade mas mais rápidos de processar.

Como indicado no ponto 2.5.6 o tempo de processamento de uma mensagem pode chegar a 94% do tempo total dessa mesma mensagem, assim sendo, para protocolos que necessitem de troca de dados num curto espaço de tempo a capacidade de processamento dos controladores e a simplicidade do protocolo são de elevada importância [6]. O protocolo CAN é um exemplo disso, é um protocolo simples e eficiente concebido inicialmente pela Bosch para a indústria automóvel, que de modo a cumprir as exigências em termos de velocidade de processamento e fiabilidade para sistemas que incluem

segurança, como os sistemas de travagem, apenas utiliza as camadas um e dois do modelo OSI.

A transmissão dos dados segundo o modelo OSI é iniciada pelas camadas superiores em que é agregada a informação a enviar em um pacote de dados chamado de *protocol data unit* (PDU), este pacote contém toda a informação necessária, contém os dados (*data*), cabeçalhos (*headers*) e fins de mensagem (*trailers*) próprios da camada que o está a enviar, este pacote de dados PDU é então enviado para uma camada inferior onde é considerado um pacote de dados chamado de *service data unit* (SDU), a este pacote de dados são então adicionados os novos cabeçalhos e fins de mensagem da formando o PDU da camada que o está a processar.

Exemplo: o PDU6 da camada 6 é o SDU6 a utilizar na camada 5, são-lhe adicionados os cabeçalhos e fins de mensagem próprios da camada 5 e constitui-se o PDU5 da camada 5, o qual será novamente considerado o SDU5 utilizado na camada 4 e assim sucessivamente até à camada 1.

Tabela 2.4 Progressão de dados ao longo das camadas do modelo OSI

Modelo OSI - SDU e PDU				
	Cabeçalho	**Dados**	**Term.**	**Camada**
	H7	Dados		7
H6	SDU 7			6
	PDU 6			
H5	SDU 6			5
	PDU 5			
H4	SDU 5			4
	PDU 4			
H3	SDU 4			3
	PDU 3			
H2	SDU 3		T2	2
	PDU 2			
H1	0101010101010101010101010101010101010101010101			1

Este modelo de constituição dos protocolos permitiu uma uniformização na forma como são constituídos e no seu princípio de funcionamento mas não na forma de processar

os dados, nas camadas a utilizar e no tipo de dados enviar, estas definições ficam ainda a cargo de quem elabora os protocolos sendo também que diferentes industrias e ambientes podem requerer diferentes particularidades nas suas redes existindo sempre protocolos diferentes para as diversas áreas. O fato de não existir um protocolo que possa ser aplicado na globalidade das aplicações implica sempre a necessidade de existir uma integração de redes em automação.

2.5 Desempenho e características de uma rede de campo

A diversidade de protocolos existentes para redes de campo resulta por um lado da necessidade que os fabricantes têm em garantir o melhor desempenho possível dos seus equipamentos ligados em rede, de forma a oferecer ao cliente soluções robustas e fiáveis, e por outro lado da necessidade de fidelizar os seus clientes de modo a que eles utilizem ao máximo os seus produtos. Nesta secção são abordados alguns aspetos relacionados com o desempenho e características típicas a ter em consideração na escolha das redes de campo.

2.5.1 Topologias

Existem tipicamente três topologias, barramento, anel e estrela, a primeira é usualmente utilizada nas redes de campo e as outras nas redes de *Ethernet* Industrial.

A escolha da topologia prende-se essencialmente com três aspetos fundamentais, o custo, a performance e a redundância ou fiabilidade [3] [5].

2.5.1.1 Barramento (bus)

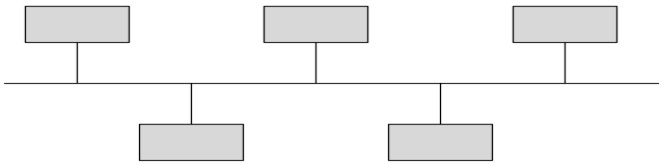

Figura 2.2 Topologia em barramento

Em termos de topologia é usual nas redes de campo a topologia em barramento (*bus*), tal configuração permite reduções de custos com cablagem. A sua simplicidade reduz custos de engenharia, comissionamento e manutenção e é facilmente extensível, apresenta também alguns contras como a dependência de um único meio físico de comunicação, o

que pode causar uma falha de modo comum em toda a rede, existem limites quanto ao comprimento e número de dispositivos presentes na rede e o aumento destes reduz a eficiência da mesma.

2.5.1.2 Anel (ring)

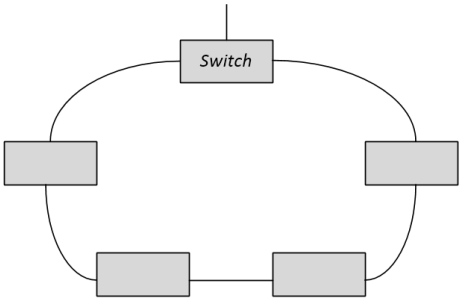

Figura 2.3 Topologia em anel

A topologia em anel é uma configuração que pretende providenciar mais fiabilidade e segurança que uma rede em barramento sem um grande aumento no custo, em caso de rutura ou desconexão de um dos troços da rede a comunicação pode continuar a ser feita por um outro troço ficando a rede em estrela.

A rede em anel é normalmente utilizada ao nível de sensor ou dispositivo e interligada a uma rede de nível superior por meio de um *switch*.

2.5.1.3 Estrela (star)

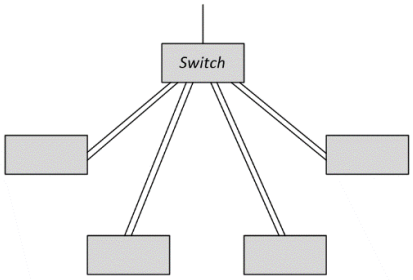

Figura 2.4 Topologia em estrela

A topologia em estrela é composta por um equipamento central, neste caso um *switch* ao qual estão ligados os restantes equipamentos, é a topologia que maior fiabilidade, performance e redundância oferece mas a um custo mais elevado pois toda a cablagem é duplicada, este tipo de rede pode facilmente recuperar de falhas múltiplas ao contrário das

anteriores e possibilita um caminho mais direto de comunicação, no entanto está dependente de um único equipamento central o qual pode originar uma falha em modo comum para toda a rede.

2.5.2 Velocidade de transmissão

A velocidade de transmissão de uma rede é medida em *bits/s* e traduz a quantidade de *bits* por unidade de tempo que podem ser transmitidos numa dada rede digital de dados.

A velocidade de transmissão especificada nos protocolos não é diretamente aplicável ao cálculo do tempo de transmissão em uma rede em si pois existem outros fatores a ter em conta como o tamanho da rede, o número de dispositivos, a velocidade do próprio meio físico e interface utilizada e o tamanho da mensagem necessária pelo protocolo. Tipicamente protocolos mais simples com eficiência maior necessitam de velocidades menores para conseguirem o mesmo tempo de transmissão. Em contrapartida, os protocolos mais simples são também os menos fiáveis e com maior taxa de *bits* ou tramas erradas.

A título de exemplo, caso se pretenda efetuar a transmissão de uma mensagem com receção de resposta, ambas com 20 *bytes* de tamanho a 11 *bits/byte* e separadores de 3.5 *bits* (ex.:MODBUS RTU), verifica-se que:

O número de *bits* a transmitir é dado por:

$$NBits' = NBytes \times \frac{NBits}{Byte} \times NMensagens + 2 \times \frac{Bits}{Separador}$$

$$= 20 \times 11 \times 2 + 2 \times 3.5 = 447\ bits\ por\ transação$$

Considerando-se 10% de falhas na comunicação serão necessários 497 *bits* por transação.

$$NBits = \frac{NBits'}{\mu} = \frac{447}{0.90} = 497\ bits\ totais\ por\ transação$$

Considerando um ritmo de transmissão de 9600 *bits/s* verifica-se que o tempo de duração de cada *bit* é de:

$$Tbit = \frac{1}{9600} = 0.104\ ms$$

O que dará um tempo total de duração da transação de:

$$T = 0.104 \times 10^{-3} \times 497 = 51.77\ ms$$

A este tempo de transação deve ainda ser adicionado o tempo de processamento da mensagem nos próprios equipamentos, a possibilidade de ocorrência de colisões (depende

do modo de acesso) e erros de mais alto nível (por ex: erros nas tramas ou na própria mensagem, como comandos ou endereços inválidos).

2.5.3 Métodos de acesso à rede

O método de acesso à rede é especificado no protocolo em si e define as regras que os dispositivos devem seguir para o envio de mensagens para a rede de modo a minimizar a probabilidade de conflitos, a preferência por um ou outro método de acesso está dependente do tipo de aplicação.

Master-Slave (mestre-escravo) é um método em que existe um dispositivo *master* que tem permissão para enviar e receber mensagens, os dispositivos *slaves* apenas têm a permissão para responder ao *master*, este método tem também a opção de *broadcast* em que o *master* envia uma mensagem de escrita a todos os *slaves*, os mesmos não respondem a um *broadcast*, é um exemplo de rede a utilizar entre um controlador e os seus periféricos.

Token-Passing (passagem de testemunho) é um método utilizado nas redes de nível médio entre controladores com topologia em anel ou em barramento em que a permissão de enviar mensagens é ciclicamente alternada entre dispositivos

Producer-Distributer-Consumer (produtor - distribuidor – utilizador) é um método utilizado nas redes em barramento para controlo de máquinas em que o produtor efetua a transmissão em *multicast* (um emissor e vários recetores) e os dispositivos que necessitarem da informação, os consumidores, processam a mensagem, este método é o utilizado por excelência quando existe a necessidade de muitos consumidores para poucos produtores.

Existem ainda métodos de acesso espontâneo como o CSMA (*Carrier Sense Multiple Access*) nas vertentes CD (*Collision Detection*) e BA (*Bitwise Arbitration*), em que qualquer dispositivo pode aceder à rede caso não seja detetada atividade, na vertente CD o emissor ao enviar uma mensagem continua a ler os dados presentes na redes e caso existam discrepâncias é porque existiu um erro ou colisão, aguarda um intervalo de tempo aleatório e volta a enviar, esta vertente é utilizada em rede maiores, em redes mais pequenas pode ser utilizada a vertente BA em que a verificação da colisão é feita *bit* a *bit*, em caso de colisão é dada prioridade ao *bit* zero o dispositivo que estiver a transmitir um zero tem prioridade e continua e o dispositivo que estiver a transmitir o *bit* um termina a transmissão.

2.5.4 Determinismo e largura de banda

Em termos de redes de campo o determinismo é a capacidade da rede enviar e receber mensagens sistematicamente com um tempo de duração igual ou muito semelhante. Este aspeto é crítico para aplicações que requerem atualizações periódicas em intervalos de tempo pré-determinados, de modo a efetuar corretamente um controlo em cadeia fechada ou obter leituras de uma variável analógica de forma sistemática. A forma como a rede está estruturada fisicamente, o seu comprimento, a velocidade de transmissão e o método de acesso à rede são aspetos importantes para o determinismo.

Um exemplo de uma rede não determinística é a *Ethernet* comercial que utiliza um método espontâneo de acesso à rede (CSMA-CD), isto leva a que tanto a transmissão como a receção de mensagens tenham tempos aleatórios. Caso existam múltiplos equipamentos a aceder em simultâneo à rede não é garantida a troca de mensagens num tempo determinado. Para redes onde o tempo de acesso e determinismo não sejam fatores críticos, como ao nível da supervisão ou da rede local da empresa, a rede *Ethernet* comercial é perfeitamente aceitável. Para a maioria das aplicações industriais que requerem redes de campo ao nível dos controladores ou sensores a rede *Ethernet* comercial não é aconselhada.

O modelo de rede utilizado pela rede *Ethernet* utiliza atualmente *switches* dotados de capacidade de encaminhamento, ou seja uma mensagem é apenas encaminhada para a rede a que pertence e não para todas as outras, reduzindo ou mesmo eliminando o desperdício de largura de banda. Outra otimização da rede *Ethernet* é a utilização de apenas um porto de comunicação do *switch* para cada elemento de uma rede com tempos de transmissão críticos, criando uma rede com apenas um elemento, o custo da rede aumenta mas é compensado com uma largura de banda dedicada apenas para um equipamento e utilizando a técnica de encaminhamento de mensagens por parte dos *switches* são então conseguidas performances satisfatórias e o determinismo suficiente para muitas das aplicações industriais [7].

2.5.5 Deteção de erros e diagnósticos

A deteção de erros na receção de uma mensagem pode ocorrer ao longo das diversas camadas do modelo OSI dependendo do protocolo utilizado, por norma na primeira camada existe logo deteção de erros ao nível da receção e identificação dos próprios *bits* dependendo da interface utilizada. Na interface RS232 está definido que o sinal elétrico de

um *bit* não pode permanecer mais de 4% do tempo de *bit* na zona de transição (-3 a +3 V) para tempos de *bit* entre 25 ms e 125 µs ou reentrar na zona de transição após ter saído, caso estas condições aconteçam o *bit* é considerado inválido.

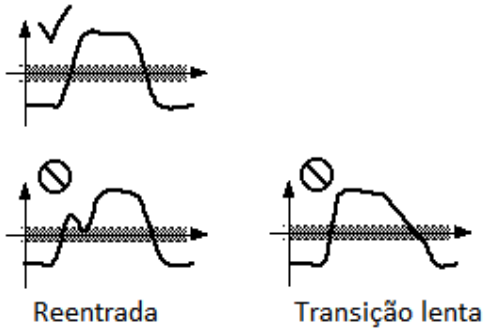

Figura 2.5 Exemplo de erros na deteção de bits segundo a norma RS232

Após a correta receção da sequência de *bits* que formam um *byte* o protocolo pode dispor de diversas ferramentas de diagnóstico e controlo de erros tais como condições de receção de início e fim de mensagem (utilizando caracteres especiais, MODBUS ASCII, ou por tempo máximo, MODBUS RTU), identificação da sequência correta de *bytes* esperados, valor dos *bytes* correto, contadores para efeitos de diagnóstico, verificação de tramas através de códigos (como os códigos CRC (*cyclic redundancy check)* ou LRC (*Longitudinal Redundancy Checking*)), entre outros.

Após a identificação de um erro podem ser tomadas diversas ações como o envio de uma resposta de erro, a recuperação dos dados da mensagem, ou a mensagem pode simplesmente ser descartada, todas estas ações estão dependentes do protocolo utilizado.

2.5.6 Eficiência e performance dos protocolos

Eficiência de um protocolo é a quantidade de *bytes* necessários para enviar um pacote de dados, normalmente medida pelo número máximo de *bytes* de dados a dividir pelo tamanho total máximo de uma mensagem.

Como verificado anteriormente cada *bit* necessita de um determinado tempo de transmissão e processamento, no exemplo do ponto 2.5.2 por cada *byte* de dados enviado são necessários 11 *bits*, um protocolo que reduza 1 *byte* de controlo por cada mensagem enviada (mensagem e resposta, ou seja 2 x 11 *bits*) reduz no mínimo 2.29 ms ao tempo necessário para a transmissão da mesma mensagem.

$$Tredução = 2 \times 11 \times 0.104 = 2.29\ ms$$

Em termos de performance devem ser tidos em conta alguns aspetos que influenciam uma rede ao nível do protocolo utilizado [6]:

- **Atrasos causados por *software*:** Segundo Mohammad Asad da Buraq Integrated Solutions em uma transmissão entre dois equipamentos ao longo de 200 metros com um *switch* central no envio de 64 *bytes* foram medidos 330 µs de processamento nos equipamentos, 10 µs de atraso no *switch* e 11 µs para a transmissão em si, cerca de 94% do tempo necessário para a comunicação foi em processamento e não em transmissão.
- **Tempos reservados para protocolos de segurança e diagnóstico:** Pelo mesmo estudo de Mohammad Asad verificou-se que adicionando diagnósticos por OPC (*Open Platform Comunications*) o tempo médio de uma mensagem passou de 0,33 ms com um desvio padrão de 0,03 ms para um tempo médio de 1,50 ms com um desvio padrão de 0,81 ms.
- **Prós e contras de uma rede partilhada entre controlo e segurança:** Uma rede *Ethernet* Industrial com as devidas características de velocidade e fiabilidade pode eventualmente ser utilizada também para as funções de segurança através da constituição de redes virtuais com o intuito de redução de custos mantendo a funcionalidade, sendo de acautelar eventuais falhas de modo comum.

2.5.7 Segurança Intrínseca

Existem protocolos e redes especificamente adaptados para utilização em locais com risco de explosão, tais como petrolíferas, tratamento de águas residuais ou indústrias químicas, as quais utilizam equipamentos físicos específicos assim como níveis de tensão e corrente apropriados, os quais regulados pelas normas IEC 61158-2 e IEC 60079-11.

Pela norma IEC 60079-27 é designado o modelo FISCO (*Fieldbus Intrinsically Safe Concept*) [8] que permite a utilização de equipamentos em zonas com risco de explosão se os mesmos cumprirem os requisitos da norma, tais como número máximo de equipamentos n = 10, tensão e correntes máximas Vmax = 17.5 V (em falha) e Imax = 380 mA, entre outros [9].

Dois exemplos de protocolos que podem implementar este modelo são o PROFIBUS PA (rede de campo) e Foundation Fieldbus H1 (*Ethernet* Industrial).

2.5.8 Segurança de dados

Existem três fatores principais a considerar em termos de segurança de dados:

- Integridade: Negação da possibilidade de alteração de dados por terceiros;
- Confidencialidade: Negação da possibilidade da recolha de dados por terceiros;
- Disponibilidade: Prevenção contra ataques à disponibilidade da rede (*Denial of Service*).

De modo a estabelecer-se uma segurança fiável em uma rede de automação é necessário conhecer todos os equipamentos com acesso à mesma e não o permitir por equipamentos não autorizados, para tal utilizam-se equipamentos e métodos de bloqueio e proteção tais como *firewalls*, *switches* e antivírus.

A utilização de *firewalls* permite criar zonas de bloqueio a endereços não registados na rede e os *switches* mais avançados, como os *managed switches*, dispõem de capacidades de adicionar *passwords*, bloquear portos, filtrar endereços MAC (*Medium Access Control*) e criar redes virtuais mais pequenas de acesso restrito [10].

2.6 Protocolos Série Profibus, Modbus e ASi

Dos protocolos existentes para redes de campo apresentam-se alguns dos atualmente existentes, os quais serão os abordados diretamente nesta dissertação. Para maior detalhe sobre cada um dos protocolos apresentados aconselha-se a consulta das respetivas normas ou especificações.

2.6.1 PROFIBUS

O protocolo de comunicação PROFIBUS (*Process Field Bus*) [8] desenvolvido originalmente como um protocolo proprietário tornou-se no projeto de investigação de um conjunto de empresas e instituições liderado pela Siemens acabando por se formar a PROFIBUS Nutzerorganisation passando então a ser um protocolo aberto, é um protocolo de comunicação para aplicação em redes série industriais definido nas normas EN 50170, IEC 61158 e IEC 61784, estando dividido em dois perfis, DP (*Decentralized Peripherals*) para redes de nível médio e PA (*Process Automation*) para redes de nível baixo, assim

como as variantes PROFIdrive para controlo de movimento e PROFIsafe para sistemas de segurança, estes perfis derivam de um terceiro perfil denominado FMS (*Fieldbus Message Specification*) entretanto já descontinuado.

O protocolo utiliza um método de acesso híbrido *Master-Slave* com passagem de testemunho (*Token Passing*) possibilitando a existência de diversos *masters* em um só barramento, permitindo até 32 equipamentos por segmento e um máximo de 126 equipamentos por rede.

A interface de acesso utilizada é a RS485 com codificação NRZ (*No Return to Zero*), podendo ainda ser utilizada a RS485-IS (*Intrinsically Safe*) para zonas com riscos de explosão.

Os meios físicos utilizados são o par entrançado de condutores com blindagem e a fibra ótica, podendo utilizar ritmos de transmissão de 9,6 *kBits/s* até 12 *MBits/s*.

Tabela 2.5 Composição do protocolo PROFIBUS

Perfil de aplicação	PROFIsafe	PROFIdrive	Encoders	Sistemas de Ident.	Outros
Tecnologia de comunicação	PROFBUS DP (DP-V0, -V1, -V2)				
Tecnologia de transmissão	Cableado RS485 RS485-IS		Fibra Ótica		*Wireless*

2.6.1.1 PROFIBUS FMS (Fieldbus Message Specification)

O perfil FMS usualmente referido na literatura como o terceiro perfil de PROFIBUS foi uma das primeiras versões criadas para utilização em redes de alto nível, entre PLC's e a supervisão ou servidores, utilizando mensagens de grandes dimensões e uma comunicação não determinística típica de uma rede de alto nível, este protocolo ainda é utilizado atualmente devido ao número significativo de equipamentos já instalados mas devido às suas características complexas e sobreposição com as redes do tipo *Ethernet* está atualmente a ser substituído pela mais recente PROFINET.

2.6.1.2 PROFIBUS DP (Decentralized Peripherals)

O perfil DP é considerado o perfil base do protocolo PROFIBUS, os restantes perfis e variantes são designados como perfis de aplicação para utilizações específicas que têm como referência o perfil DP com modificações ao mesmo, como por exemplo formatos de dados diferentes, método de acesso à rede ou interfaces de acesso ao meio físico.

O protocolo PROFIBUS-DP foi criado a partir do FMS como uma versão simplificada, mais eficiente e com comunicação determinística para utilização em redes de nível médio. O perfil DP tem três níveis de performance DP-V0, DP-V1 e DP-V2, o nível V0 é o nível base e permite a comunicação periódica e alguns diagnósticos, os outros níveis acrescem serviços ao nível zero mas tornam o protocolo mais complexo com tempos de comunicação superiores, ao nível V1 é acrescentada a possibilidade de comunicação aperiódica, parametrizações e alarmes, ao nível V2 são acrescentados serviços ainda mais específicos como a sincronização e marcações temporais, o nível de performance DP-V1 é o nível mais utilizado.

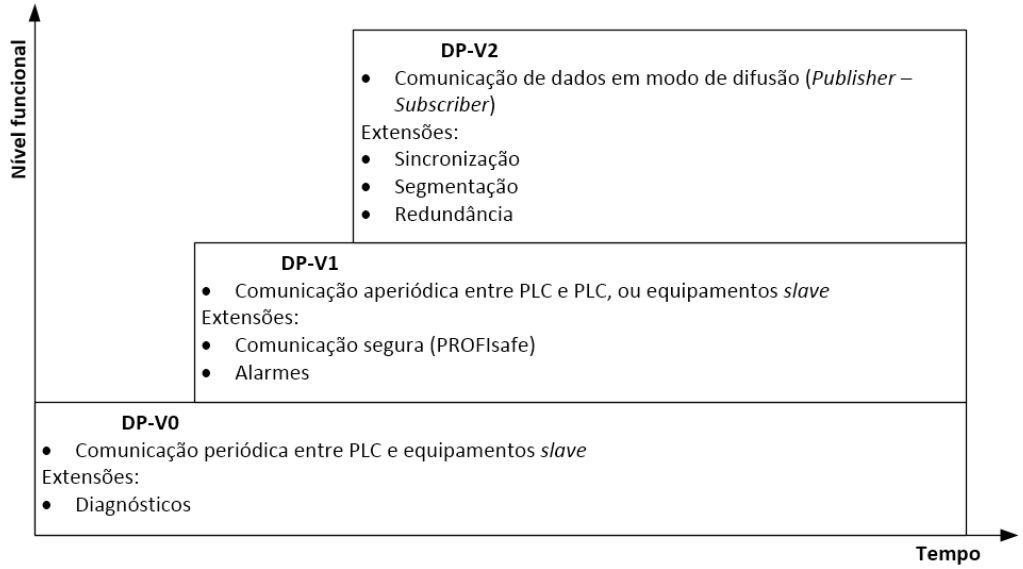

Figura 2.6 Níveis de performance em PROFIBUS DP

2.6.1.3 PROFIBUS PA (Process Automation)

O perfil PA é como o nome indica utilizado na automatização de processos, ou seja, entre o controlador e os atuadores/sensores, este tipo de rede PROFIBUS utiliza como base o perfil DP com alterações no meio físico e interface para utilização de correntes e tensões

mais baixas, sendo caracterizado por ser uma rede intrinsecamente segura e pela alimentação de energia pelo cabo de rede, o perfil PA foi desenhado especificamente para este uso contendo em cada mensagem não só o valor dos dados mas também informação de controlo normalizada para os diversos dispositivos regularmente utilizados a este nível de rede, como por exemplo o FCS (*Frame Checking Sequence*), para além desta funcionalidade na mais recente versão V3.02 a implementação do protocolo é compatível com as versões anteriores, permitindo a utilização de equipamentos novos em redes existentes.

2.6.2 Modbus

O protocolo MODBUS [11] é um protocolo *Master-Slave* com implementação numa rede série, as interfaces com os meios físicos são tipicamente RS232 ou RS485, com algumas variantes nas ligações físicas como a utilização de 2 ou 4 condutores, o protocolo permite a comunicação entre um *master* e até 247 *slaves* com dois tipos de mensagens, em *unicast*, mensagem do *master* para um *slave*, ou *broadcast*, mensagem de escrita do *master* para todos os *slaves*.

Em relação aos modos de transmissão o protocolo pode utilizar os modos RTU (*Remote Terminal Unit*) ou ASCII (*American Standard Code for Information Interchange*), no modo RTU a deteção de mensagens é feita por tempos, entre caracteres 1,5 vezes o tempo de um caracter e entre mensagens 3,5 vezes, no modo ASCII são utilizados caracteres especiais para deteção de início e fim de mensagens, o modo RTU é mais eficiente necessitando de menos *bits* de controlo.

Segundo o modo RTU por cada *byte* de dados são enviados 8 *bits* de dados, 1 *start bit*, 1 *bit* de paridade e 1 *stop bit*, ou 2 *stop bits* no caso de não ser utilizado o *bit* de paridade. O controlo de erros no modo RTU é efetuado por CRC e em modo ASCII por LRC.

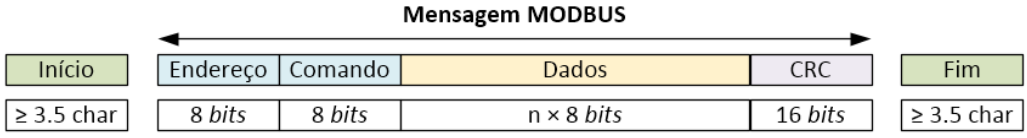

Figura 2.7 Trama MODBUS segundo o modo RTU

O protocolo de comunicação MODBUS é um protocolo aberto desenvolvido de acordo com as camadas 7, 2 e 1 do modelo OSI. A documentação do protocolo encontra-se dividida em duas partes principais:

- Especificação do protocolo de aplicação MODBUS, na camada 7;
- Especificação da implementação do protocolo numa rede série, nas camadas 1 e 2.

Nestes dois documentos são especificados quais os requisitos que o protocolo tem de cumprir e quais os que deve cumprir (não obrigatório). Para a integração das redes propostas nesta dissertação será necessário programar o protocolo MODBUS em si mesmo no autómato S7-200 de modo a cumprir os requisitos obrigatórios. Os requisitos não obrigatórios são através deste modo cumpridos apenas parcialmente. A implementação deste protocolo no autómato S7-200 encontra-se descrita no capítulo 3.

2.6.3 AS-i

O protocolo ASi (*Actuator Sensor Interface*) [3] é também um protocolo *Master-Slave* com implementação numa rede série utilizando um meio físico específico. O cabo utilizado é composto por um par de condutores não torçados e sem blindagem, pode ainda ser utilizado um segundo cabo para alimentação dos equipamentos a 24 VDC ou 230 VAC.

O sinal transmitido na rede utiliza a codificação *manchester* diferencial modulado por APM (*Alternate Pulse Modulation*), este tipo de modulação providencia uma boa imunidade a perturbações permitindo a utilização do cabo não torçado e sem blindagem.

O acesso dos equipamentos ao meio físico é feito por punção do isolamento à alma condutora, sendo efetuado pelo próprio equipamento ou pela utilização de repetidores, estes podem ser ativos para conexão de atuadores e sensores passivos, ou passivos para conexão de equipamentos ativos.

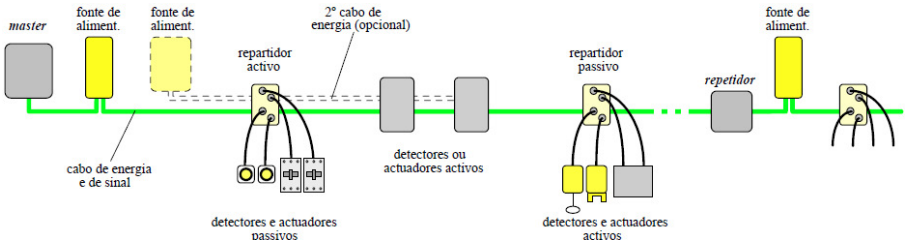

Figura 2.8 Exemplo de rede ASi

2.7 Interfaces Físicas

As interfaces físicas mais comuns em ambiente industrial são as *Recommended Standard* RS232 e RS485 publicadas pela EIA-TIA, em que são especificadas as características das ligações físicas entre os equipamentos e o meio físico de transmissão, assim como a correspondência entre os sinais elétricos e os níveis lógicos do sinal.

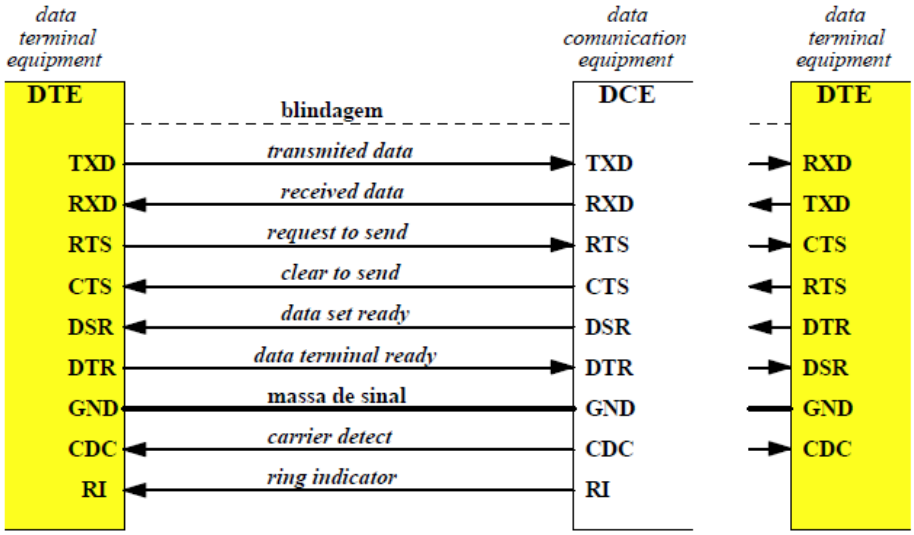

Figura 2.9 Ligações físicas entre emissor e recetor segundo a norma RS232C [3]

Em relação a níveis lógicos a norma RS232 utiliza dois níveis de tensão, entre -3 e -15 V para o nível lógico "1" e de +3 a +15 V para o nível lógico "0" e permite ritmos de transmissão até 20000 *bits/s*, sendo os mais usuais 9600 e 19200 *bits/s*, sendo utilizada para comunicação ponto a ponto entre dois equipamentos.

A norma RS485 por sua vez utiliza um sinal diferencial com os níveis de tensão entre -1.5 e -6 V para o nível lógico "0" e de +1.5 a +6 V para o nível lógico "1", permite ritmos de transmissão até 35 *Mbits/s* e pode ser utilizada para redes de 32 nós.

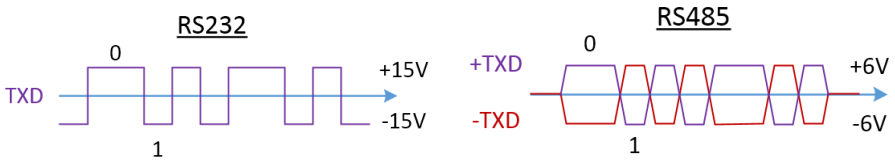

Figura 2.10 Exemplo de sequências de bits

2.8 Meios Físicos UTP, STP e Fibra Ótica

2.8.1 U/UTP

Em relação a meios físicos para a transmissão de sinais são usualmente utilizados condutores de cobre em pares entrelaçados sem blindagem, designados por cabos U/UTP (*Unshielded Twisted Pair*) (antigo UTP), os quais são constituídos por quatro pares de condutores de cobre isolados a policloreto de vinil para a propagação do sinal em tensão, sendo o entrelaçamento utilizado para mitigar a influência de perturbações por acoplamento indutivo (criando forças eletromagnéticas induzidas de sentidos contrários) provenientes dos circuitos de potência (p) figura 2.11 [12], é um cabo pouco dispendioso e regularmente utilizado para distâncias até 100 m.

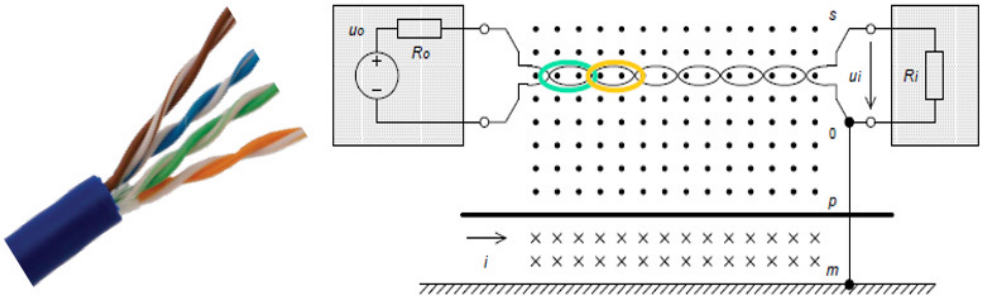

Figura 2.11 Exemplo de cabo de par torçado sem blindagem

Estes cabos são especificados pelas normas EN50288-4-1, EN50288-5-1 e EN50288-6-1 para cabos sólidos e as EN50288-x-2 para cabos flexíveis, sendo atualmente utilizados os cabos de categoria 6, 6a, 7 e 7a [13] com larguras de banda de 250 MHz a 1000 MHz utilizados para redes *Ethernet* até 10 *Gigabit/s* (*10GBASE-T Ethernet*) estando presentemente em desenvolvimento cabos de categoria 8 com larguras de banda até 2000 MHz para redes até 40 *Gigabit/s.*

2.8.2 S/FTP

No caso de existirem perturbações significativas por acoplamento capacitivo, pela presença de dispositivos que produzam campos elétricos variáveis de tensão elevada na proximidade do circuito de sinal, por exemplo circuitos de comutação, podem se utilizar

cabos blindados S/FTP (*Shielded / Foiled Twisted Pair*) (antigo STP), os quais são cabos U/UTP com uma bainha metálica (malha de alumínio) em torno de todos os pares e blindagem individual a cada par (fita de alumínio). Ao ser utilizada a bainha metálica é estabelecido um circuito fechado entre a bainha e a massa pelo qual serão encaminhadas as correntes perturbadoras, neste tipo de cabos é importante efetuar a ligação à massa apenas em uma das extremidades da bainha de modo a não criar um efeito de massas distintas [12].

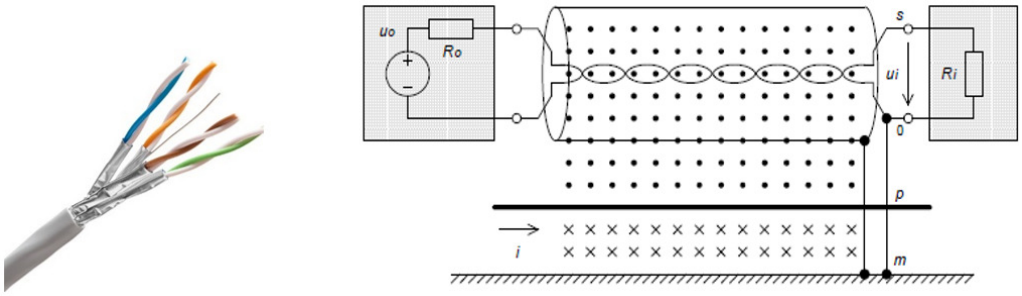

Figura 2.12 Exemplo de cabo de par torçado com blindagem

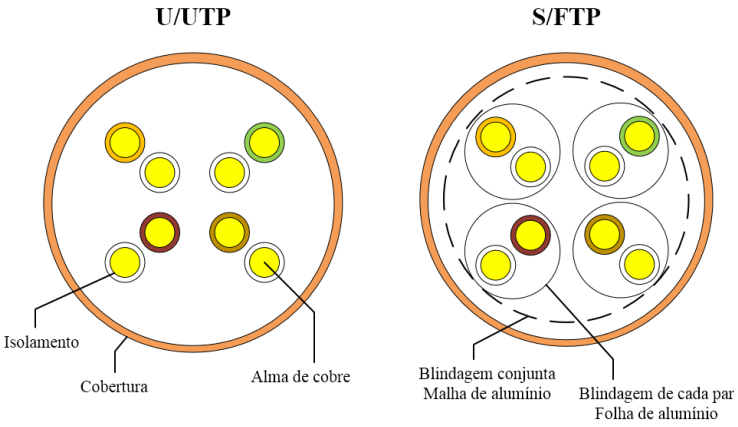

Figura 2.13 Comparação entre cabos U/UTP e S/FTP

2.8.3 Fibra Ótica

O cabo de fibra ótica é composto por um ou mais filamentos flexíveis de vidro ou plástico extrudido protegidos por um isolamento de polímero de acrilato com baixo nível de refração e uma bainha exterior para proteção mecânica, o tipo e número de bainhas pode variar consoante a aplicação.

Existem dois tipos de cabos de fibra ótica SMF (*Single Mode Fiber*) e MMF (*Multi Mode Fiber*), os cabos MMF têm um filamento mais largo, permitem mais do que um modo de propagação e são menos dispendiosos de produzir. No entanto têm uma dispersão de luz superior ao cabo SMF o que limita a sua utilização a distâncias mais curtas, normalmente utilizados até 2 kms, o cabo SMF é normalmente utilizado para distâncias mais longas com a possibilidade de atingir ritmos de transmissão de 10 *GBit/s* a mais de 100 kms [14].

A utilização de um feixe de luz em detrimento de um sinal em tensão para a transmissão de informação proporciona ao cabo de fibra ótica imunidade a perturbações do tipo elétrico como as EMI (*Electromagnetic Interference*) ou RFI (*Radio Frequency Interference*).

2.9 Integração de redes

Como referido anteriormente, a integração de redes de comunicação no domínio da automação não aparenta ter solução fácil dada a incompatibilidade direta entre equipamentos de diferentes fabricantes. Mesmo encontrando uma rede aberta que permita uniformizar a maioria das novas soluções existe já uma situação atual com equipamento instalado e investimentos feitos de tal modo que não seria possível retirar simplesmente todo o equipamento existente e investir em equipamento novo. Assim, a situação atual de equipamentos com protocolos diferentes exige a compatibilização e integração de redes de uma forma menos dispendiosa, mesmo não sendo esta a solução ideal.

Esta integração de redes requer muitas vezes a utilização de equipamentos de compatibilização entre redes, as chamadas *gateways*, módulos de comunicação específicos para cada protocolo ou mesmo com recurso à utilização dos próprios controladores para servirem de *gateways* e converterem os dados de uma rede para outra.

A necessidade de compatibilização das redes, a própria conversão de mensagens e reencaminhamento são operações que consomem tempo e recursos resultando em tempos de execução mais longos, maior probabilidade de erros e quando feito nos próprios controladores utilizará os recursos dos mesmos tais como memória e capacidade de processamento que em vez de serem utilizados para o controlo do processo como deviam serão utilizados para a compatibilização das redes. No caso em que o protocolo é suportado pelos controladores, como acontece com os protocolos proprietários, os próprios módulos de comunicação dispõem de processadores de comunicações e recursos próprios para

gerirem o tráfego da rede, libertando assim o controlador do processo dessa tarefa, conseguindo desta forma performances superiores nas redes.

Esta necessidade da integração das redes de campo levou à criação de grupos de trabalho, constituídos por técnicos e engenheiros do ramo da automação industrial, que acabariam por formar uma nova categoria profissional de integradores de sistemas. Fundada em 1994 a CSIA (*Control System Integrators Association*) é uma associação que reúne profissionais e empresas dedicadas à integração de sistemas de automação com o objetivo de melhorar e harmonizar a integração de sistemas com a publicação de recomendações, guias de boas práticas e emitindo certificações aos seus membros qualificados [15].

2.10 Protocolos abertos

Dada a existência de inúmeros equipamentos e protocolos proprietários, aliada ao investimento elevado em redes de automação e ao tempo de vida útil longo das mesmas tem levado gradualmente os clientes a procurarem sistemas de automação compatíveis com protocolos abertos que permitam algum nível de liberdade para futuras aquisições de equipamentos de diferentes fabricantes e expansões das próprias redes.

O desenvolvimento de protocolos abertos permitiu ainda o surgimento de pequenas empresas de fabrico de equipamentos como sensores e atuadores compatíveis com os mesmos assim como a partilha do conhecimento do protocolo em si reduziu os custos de desenvolvimento dos mesmos e aumentou a fiabilidade e estabilidade.

Naturalmente que com a evolução dos protocolos abertos no mercado tem-se presenciado um declínio gradual dos protocolos proprietários e os seus fabricantes incentivados a disponibilizar e a elevar os mesmos a norma com o apoio das entidades responsáveis como a IEC e a ISA.

A IEC com base nas redes PROFIBUS e FIP (*Factory Instrumentation Protocol*) e com o intuito de proporcionar meios normalizados para interligação de componentes nas redes de campo, utilizando o modelo de referência OSI publicou as normas IEC 61158 e IEC 61784 de modo a uniformizar o desenvolvimento de protocolos, a norma IEC 61158 foi então publicada em oito documentos com especificações e definições segundo as quais as redes de campo devam ser desenvolvidas:

- IEC DIS 61158-1 – *Introductory Guide*
- IEC DIS 61158-2 – *Physical Layer Specification*
- IEC DIS 61158-3 – *Data Link Service Definition*
- IEC DIS 61158-4 – *Data Link Protocol Specification*
- IEC DIS 61158-5 – *Application Service Definition*
- IEC DIS 61158-6 – *Application Protocol Definition*
- IEC DIS 61158-7 – *Fieldbus Management*
- IEC DIS 61158-8 – *Conformance Testing*

Recentemente o desenvolvimento das redes de campo tem-se centrado na investigação e desenvolvimento da rede *Ethernet* comercial para aplicações industriais, designada de *Industrial Ethernet*, com o intuito de a tornar viável como rede de campo reduzindo assim o número de redes necessárias/disponíveis, promovendo uma maior compatibilidade e facilidade de instalação, manutenção e operação das redes.

Ainda persistem alguns aspetos a melhorar no desempenho de uma rede de *Ethernet* industrial tal como o seu determinismo e a velocidade de transmissão, especialmente para redes com necessidades mais exigentes, tais como redes de segurança [4].

2.11 Industrial Ethernet

Os protocolos de *Ethernet* industrial têm como base o protocolo *Ethernet* TCP/IP utilizando no entanto uma camada de ligação (camada 2) do modelo OSI distinta, o que permite aumentar o seu determinismo e baixar a sua latência (intervalo de tempo entre o estímulo da rede e a resposta).

Em termos de *hardware* a principal diferença entre uma rede *Industrial Ethernet* e a rede *Ethernet* TCP/IP é a utilização de *hardware* específico para o ambiente industrial, desde cablagem, conectores e a utilização de fontes de alimentação redundantes de modo a providenciar um meio físico robusto e fiável.

Em termos de *software* os protocolos para *Ethernet* industrial são desenvolvidos para acomodar serviços como o controlo *multicast*, qualidade de serviço e redes virtuais que permitam à rede a transmissão de dados de forma rápida e consistente, tais requisitos provêm da diferença entre o tipo de comunicação utilizado numa rede *Ethernet* comercial e industrial. Numa rede industrial a comunicação é tipicamente *multicast* necessitando de comunicações rápidas e determinísticas para aplicações de controlo em tempo real [7]. As

redes *Industrial Ethernet* atuais têm velocidades na ordem dos *Gigabits/s*, capacidade de *full-duplex* (envio e receção de mensagens em simultâneo), priorização e redes virtuais [4].

Uma rede *Industrial Ethernet* não é diretamente compatível com a rede *Internet* mas podem ser compatibilizadas com equipamento apropriado aumentando ainda mais a funcionalidade da rede *Industrial Ethernet* com a ligação a equipamentos e redes informáticas de alto nível em pontos geográficos distantes e dispersos para aplicações de gestão e recolha de dados.

A figura seguinte ilustra um exemplo de uma rede *Industrial Ethernet* adotada pela empresa Siemens.

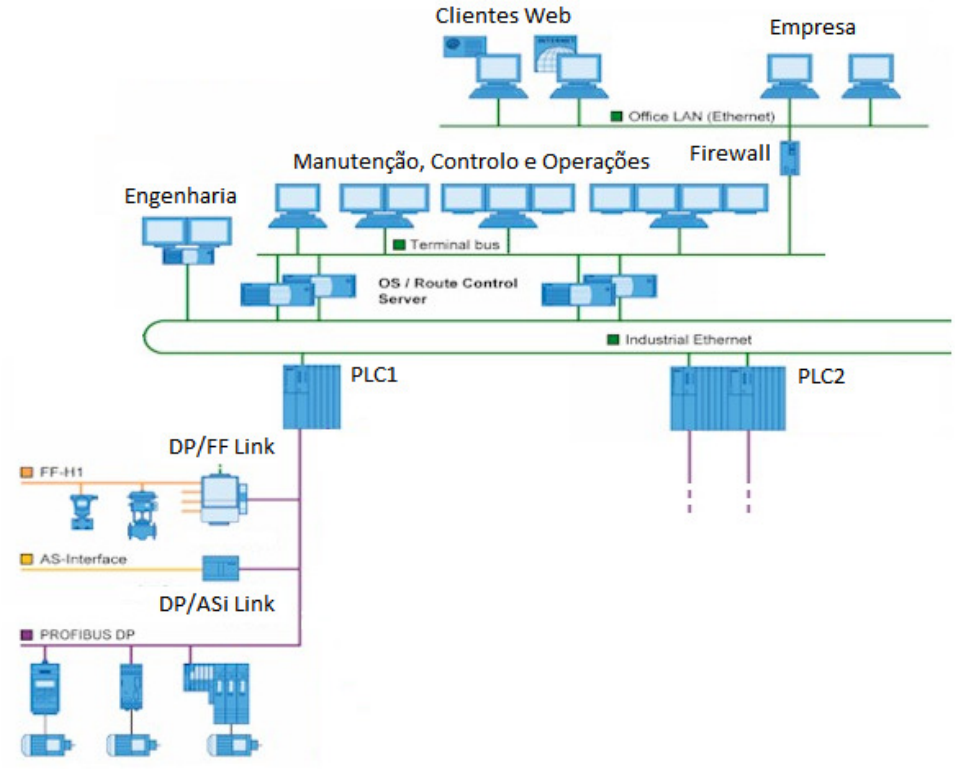

Figura 2.14 Exemplo de rede Industrial Ethernet (Siemens)

2.11.1 Serviços adicionais

Com a utilização de protocolos de *Ethernet* industrial torna-se possível fazer uso de serviços de mais alto nível provenientes da rede *Ethernet* que não estão disponíveis nas redes de campo tais como o XML (*Extensible Markup Language*), HTTP (*Hypertext Transfer Protocol*) e FTP (*File Transfer Protocol*).

XML é uma forma restrita de SGML (*Standard Generalized Markup Language*) que descreve uma classe de objetos chamados documentos XML, este tipo de ficheiros podem ser escritos ou lidos pela *Ethernet* proporcionando um ficheiro de interface entre os equipamentos e os utilizadores para operações de configuração ou manutenção.

Para uma descrição mais detalhada sobre as compatibilidades, diferenças e serviços providenciados por uma rede de *Ethernet* industrial recomenda-se por exemplo a consulta da referência bibliográfica [16].

2.11.2 Desenvolvimento futuro

A compatibilidade entre as redes de campo e as redes informáticas permite progressivamente introduzir novos conceitos de otimização da produção que certamente se instalarão na indústria. Tais como o conceito de megadados (*Big Data*) facultando a capacidade de análise de tendências e estatísticas que permitem a indústria utilizar o conceito de produção "*just in time"* e de logística *"smart logistics",* conceitos chave na redução de matérias-primas e custos de armazenamento.

A agilização e compatibilização do processo industrial e produtivo impulsiona também a aproximação do cliente à indústria permitindo a customização dos seus produtos mais rapidamente e com menores custos, a era digital da indústria é atualmente apelidada da próxima revolução industrial como *Industrie 4.0* [17].

Esta revolução é acompanhada não só pelo desenvolvimento tecnológico em redes de informação mas também pela eletrónica (micro e nano eletrónica), informática e desenvolvimento de sensores e atuadores, introduzindo cada vez mais capacidades de processamento e atuação nos próprios equipamentos como a capacidade de efetuar auto diagnósticos, correção de erros, calibração e comunicações de mais alto nível desde os próprios sensores e atuadores, estes desenvolvimentos levam ao conceito de *Internet of Everything,* no entanto com esta agregação e chegada das redes de alto nível cada vez a mais equipamentos e a níveis mais baixos surgem questões de segurança nas redes de informação, nomeadamente em relação à possível espionagem industrial ou mesmo a ataques informáticos contra as próprias fábricas como os ataques para negação de serviço (*denial of service*) [18].

2.12 Protocolos Industrial Ethernet

Apresentam-se de forma resumida alguns dos protocolos *Industrial Ethernet* e suas principais características [18]:

2.12.1 PROFINET

Um dos protocolos *Industrial Ethernet* mais utilizados na Europa é o protocolo PROFINET. O protocolo é dividido em duas vertentes, PROFINET CBA (*Component Based Automation*) e PROFINET I/O (*Input/Output*).

O PROFINET CBA é utilizado ao nível mais alto, idêntico aos níveis intermédio e alto da *Industrial Ethernet*, para redes de equipamentos via TCP/IP. O PROFINET I/O é utilizado ao nível baixo da *Industrial Ethernet*, para redes entre controladores e dispositivos utilizando o método de acesso produtor/consumidor. Uma rede PROFINET I/O pode ser integrada em uma rede PROFINET CBA.

2.12.2 Foundation Fieldbus

É um protocolo desenvolvido pela *Fieldbus Foundation* dividido em dois níveis:

- **H2:** protocolo de alto nível a velocidades elevadas (2 Mbps) para interligação de redes H1.
- **H1:** protocolo para interligação de sensores e controladores, para sinais discretos ou analógicos, a baixas velocidades (31.25 Kbps);

Nota: o nível H2 original tem sido substituído pelo HSE (*High Speed Ethernet*) com velocidades de 10 Mbps e superiores, no entanto mantendo a designação de H2.

2.12.3 EtherCAT

É um protocolo com o intuito de estabelecer comunicações em tempo real baseadas em *Ethernet* conseguindo ainda estabelecer compatibilidade de um grande número de dispositivos desde controladores a sensores. O protocolo é compatível com TCP/IP e UDP da rede *Ethernet* comercial facilitando a compatibilidade e tornando a rede mais simples.

É o protocolo utilizado pelo fabricante de componentes de automação OMRON.

2.12.4 MODBUS TCP

O protocolo desenvolvido pela Schneider é uma extensão do protocolo MODBUS com a adição de empacotamento TCP, o protocolo MODBUS TCP é uma forma simples de compatibilizar um protocolo original de uma rede de campo para uma rede *Ethernet* o que introduz alguma familiaridade e facilidade em compatibilizar equipamentos antigos, sendo que existem algumas limitações em termos técnicos pois mantém as propriedades da *Ethernet* comercial, como o determinismo ou a dificuldade em garantir a comunicação em tempo real com tempos inferiores ao milissegundo.

O protocolo MODBUS TCP ao ser um protocolo aberto pode ser desenvolvido ou utilizado por qualquer pessoa sem restrições de licenciamento. Dadas as suas características pode ser utilizado por fabricantes independentes para aplicação nos seus equipamentos permitindo a ligação dos mesmos a uma rede *Ethernet* comercial para a transmissão de mensagens com a estrutura do protocolo MODBUS original.

2.13 Tecnologia Wireless

Atualmente o desenvolvimento das redes de automação expandiu-se também para soluções com tecnologia *wireless* e para a sua possível aplicação em ambientes industriais com as exigências de uma rede de automação tal como aconteceu com as redes de *Ethernet* industrial, as redes *wireless* são uma evolução natural destas e também utilizam normas, tecnologias e equipamentos já existentes.

A tecnologia de comunicação *wireless* aplicada a redes de automação trás algumas vantagens como a diminuição de cablagem, a utilização em equipamentos móveis, ausência de cablagem em ambientes agressivos, facilidade na reconfiguração da rede, entre outros, mas também conta com algumas desvantagens como a suscetibilidade a interferências, falta de comunicação em tempo real e determinismo, sobreposição e limites das redes wireless, comunicação apenas em *half-duplex* e necessidade de mais informação de cabeçalho para uma mesma mensagem, reduzindo a eficiência do protocolo [4].

Em termos de segurança de dados as redes *wireless* também constituem uma dificuldade adicional, ao difundir a informação por um meio acessível e não contido, o ar, têm de ser aplicadas algumas medidas adicionais de segurança para garantir a confidencialidade dos dados tais como a utilização de chaves de encriptação, que aumentam o tempo de processamento e a dificuldade em conseguir um protocolo simples e rápido o

suficiente para garantir comunicações em tempo real, especialmente importantes em sinais de controlo em cadeia fechada, e a restrição da potência dos sinais de transmissão ao apenas estritamente necessário para limitar o acesso aos dados apenas a equipamentos próximos mas tal restrição pode causar o efeito de equipamento escondido, em que dois equipamentos que não estão visíveis um para o outro tentam comunicar em simultâneo com um terceiro visível aos dois, tal acontece porque não é detetada atividade na rede *wireless* com a regra de acesso CSMA.

Uma outra dificuldade na tecnologia *wireless* é a garantia da integridade dos dados pois a informação ao ser difundida pelo ar está sujeita a muitas interferências assim como as próprias antenas dos recetores estão sujeitas a perturbações por acoplamento indutivo e capacitivo, em especial num ambiente industrial em que existem muitos equipamentos com comutações a alta frequência como os variadores de frequência e campos magnéticos elevados pela utilização de correntes elevadas, como as soldaduras por exemplo.

A banda de frequências normalmente utilizadas para comunicações *wireless* de curto alcance é a 2.4 GHz ISM (*Industrial Scientific and Medical*) que por ser uma banda de frequência de utilização livre tem cada vez mais equipamentos a serem utilizados na mesma, num ambiente industrial é necessário ter em consideração o número de equipamentos emissores nesta banda de modo a que os mesmos não criem zonas de interferência ao emitirem na mesma frequência.

Uma das tecnologias *wireless* a serem desenvolvidas é a de *Bluetooth* para aplicações industriais, nomeadamente ao nível de sensores, esta tecnologia utiliza sinais a curta distância, de pequena potência, com modo de acesso *Master-Slave* e utilizando saltos em frequência (*frequency hoping*) para minimizar o efeito das colisões de mensagens, estando ainda a ser desenvolvidos protocolos específicos para minimizarem os tempos de processamento sem comprometer a segurança dos dados.

A par da tecnologia *Bluetooth* está a ser desenvolvida a *ZigBee* também para comunicações a curto alcance mas que não necessitem de transmissões frequentes, é uma tecnologia destinada ao nível de sensores e atuadores com um protocolo simples e eficiente com a mesma filosofia da rede CAN para as redes de campo.

Existem atualmente três protocolos *wireless* para utilização em redes de campo de baixo nível, ISA 100.11a, WirelessHART e WIA-PA (*Wireless Networks for Industrial Automation – Process Automation*) especificados nas normas IEC 62734, IEC 62591 e IEC 62601 [4].

2.14 Industrial IoT e Industry 4.0

Em 2006 o Governo Alemão publicou a sua estratégia de investimento em tecnologia no documento "*High Tech Strategy*" onde, com a colaboração de universidades, empresas e associações industriais, sublinhou a sua intenção de investimento em investigação e desenvolvimento no mercado tecnológico tendo sido posteriormente em 2010 atualizado como "*High Tech Strategy 2020*" onde foram incluídos temas como a energia, mobilidade, clima, comunicações e segurança, o termo *Industrie 4.0* (em inglês *Industry 4.0*) fazendo parte integrante dessa estratégia foi então apresentado em 2011 na *Hannover Messe* como sendo a quarta revolução industrial.

Industrie 4.0 é um dos projetos do "*High Tech Strategy 2020 Action Plan*" que envolve aplicar tecnologia de ponta ao tecido industrial nomeadamente a *Ethernet* Industrial, a *Internet*, *Cloud computing* e o novo conceito de *Industrial IoT* (*Industrial Internet of Things)*, de modo satisfazer as exigências do consumidor moderno com produtos customizados a baixo custo, a aumentar a eficiência, a competitividade e combater a migração da produção para os países com mão-de-obra mais barata [17].

Este conceito contempla a criação de uma indústria digital "inteligente" que através das redes de informação providencie uma ligação direta entre o consumidor e o produto final, é um termo que representa a transição industrial dos atuais sistemas embebidos para os futuros sistemas ciber-físicos, serão as próprias fábricas a interagir com o consumidor, com as equipas de manutenção e com a gestão da empresa, providenciando serviços mais rápidos, mais eficientes e decisões mais informadas.

O projeto baseia-se em quatro estágios de evolução,

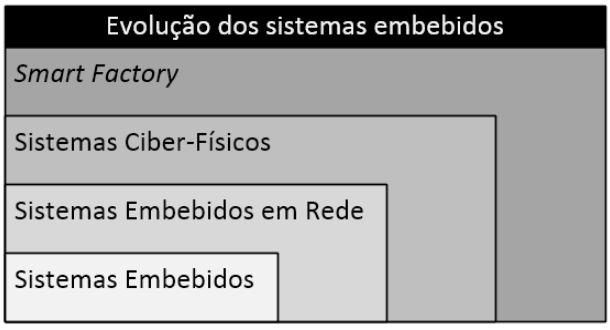

Figura 2.15 Evolução dos sistemas embebidos [19]

2.14.1 Sistemas Embebidos

Na base da criação dos sistemas ciber-físicos estão os sistemas embebidos, nos quais são utilizados processadores e microcontroladores que permitem a um equipamento processar informação digital, tais sistemas têm evoluído em capacidade de processamento e em diminuição de tamanho e consumos, o que permite adicionar a capacidade de processamento a equipamentos cada vez mais pequenos e com maiores performances, os controladores lógicos programáveis são um exemplo da utilização de sistemas embebidos em automação industrial e mais recentemente devido às evoluções descritas estão a ser desenvolvidos sensores e atuadores com capacidades de processamento próprio, dotando-os de capacidades como auto diagnósticos, auto calibração e correção de erros.

2.14.2 Sistemas Embebidos em Rede

A automação industrial atual consiste já em sistemas embebidos em rede, a qual permite não só a execução de uma tarefa, como um sistema independente, mas a execução de tarefas combinadas entre vários equipamentos com capacidade de processamento.

2.14.3 Sistemas Ciber-Físicos

Os sistemas ciber-físicos são o próximo passo evolutivo e representam a junção dos sistemas físicos, os sistemas embebidos em rede, e o mundo virtual da *Internet*, das redes de dados e serviços. Estes sistemas com capacidades de processamento de *software* cada vez mais evoluído, com acesso a redes digitais e futuramente com capacidades de inteligência artificial serão a base para *Industrial IoT*.

Os sistemas ciber-físicos serão capazes de controlar um processo físico, como uma cadeia de produção, em cadeia fechada através de uma rede de informação digital.

2.14.4 Smart Factory

O conceito de *Smart Factory* é o último estágio de evolução da *Industry 4.0* em que é estabelecida a interação entre os diversos sistemas ciber-físicos da própria fábrica, o processo produtivo, com a rede de informação de gestão, o processo de negócio, administração e comércio.

Os sistemas integrantes do processo produtivo comunicam diretamente com o *software* de gestão informando os intervenientes humanos tais como gestores, logística, manutenção e os próprios clientes para a tomada de decisões informadas em tempo real.

A evolução da automação industrial para as *smart factories* não está independente apenas fará parte de um leque alargado de ciber-sistemas com a *Internet of Things, Data and Services*, as *smart cities*, *smart grids* e *smart mobility* entre outros, criando um mundo novo onde os sistemas físicos e virtuais são parte integrante do dia-a-dia de todos.

Capítulo 3

Rede integrada

Resumo: No presente capítulo são apresentados os equipamentos e redes existentes, nomeadamente as suas características, configurações e parametrizações. Sendo posteriormente apresentada a rede integrada, a solução adotada, por ramos onde são exploradas as ligações entre autómatos e a programação necessária nos mesmos.

3.1 Introdução

Como referido no capítulo anterior um dos grandes projetos de desenvolvimento futuro das redes de automação constantes do conceito *Industry 4.0* passa pela integração das redes de campo e dos próprios equipamentos utilizados em automação nas redes de informação tais como a *Ethernet* e *Internet*.

Em termos conceptuais tal passo evolutivo torna-se viável com a modernização e atualização dos equipamentos com maiores capacidades de comunicações e de processamento, no entanto em termos reais este processo não é assim tão simples pela existência de toda uma infraestrutura já instalada a qual muitas vezes não só contém redes incompatíveis e/ou proprietárias, equipamentos desatualizados sem as capacidades necessárias e sem possibilidade de as acrescentar assim como a limitação de recursos financeiros que impede a atualização integral de uma rede existente.

Num ambiente industrial real no qual é necessário utilizar equipamentos previamente instalados, não se está muitas vezes perante uma situação em que seja possível resolver o problema de uma forma ideal mas sim perante uma situação que obriga à estruturação de uma solução de engenharia que permita ultrapassar os obstáculos presentes alcançando os objetivos pretendidos, essa solução passa pela integração de sistemas.

A integração de sistemas e redes de automação permitirá disponibilizar os recursos de certos equipamentos a outros que originalmente não os teriam ou que não seriam compatíveis.

Simulando um ambiente industrial real serão utilizados os equipamentos previamente instalados no laboratório de automação e robótica e no laboratório de pneumática com a configuração de redes já existentes sem alterar a sua disposição ou características físicas.

3.2 Equipamentos e redes existentes

Nesta secção apresentam-se os equipamentos e redes de automação existentes bem como os protocolos de comunicação utilizados nos laboratórios de automação e de pneumática.

Os equipamentos existentes nos laboratórios permitem a utilização separada de quatro protocolos de comunicação distintos. Alguns destes equipamentos são

provenientes de diversos fabricantes e quer pela sua natureza construtiva, quer por questões de concorrência, não utilizam os mesmos protocolos de comunicação.

Os equipamentos constituintes das redes existentes e os protocolos utilizados são os indicados na tabela seguinte:

Tabela 3.1 Equipamentos, redes e protocolos existentes

Equipamentos	**Protocolo**	**Rede**	**Local**
PC	MODBUS TCP	*Ethernet*	Campus
TSX57-302	MODBUS RTU	Campo	Lab. Pneumática
TSX57-103	MODBUS RTU	Campo	Lab. Automação
S7-200, S7-300, Micromaster, DP/AS-i Link	PROFIBUS-DP	Campo	Lab. Automação
Sensores e Atuadores	AS-i	Campo	Lab. Automação

Nas secções seguintes far-se-á uma apresentação geral destes equipamentos sendo posteriormente descrita a sua estrutura de ligações em *hardware* e compatibilização em *software*.

3.2.1 PC - Apresentação geral

Figura 3.1 Computador pessoal utilizado para supervisão

A supervisão da rede integrada será efetuada a partir de um computador portátil, o qual terá um programa de supervisão e estará ligado à rede *Ethernet* comercial do campus, este computador fará uma ligação com os autómatos TSX57-302 e TSX57-103, o protocolo utilizado será o MODBUS TCP. O programa de supervisão, suas configurações e ligações serão detalhadas no capítulo 4.

3.2.2 TSX57-103 - Apresentação geral

Figura 3.2 O autómato TSX57-103

O autómato é totalmente modular utilizando uma base (*standard rack*) de 6 posições, contendo os seguintes elementos:

- Fonte de alimentação PSY 2600 de 26 W a 24 VDC;
- Posição 00 – processador TSX57-103 com capacidade para controlar dois módulos de comunicação, utilizará um porto de comunicação (TER) para programação através do protocolo UNI-TELWAY LINK e um módulo de comunicação TSX SCP 111 com interface RS232;
- Posição 01 – módulo de comunicação TSX ETY 110 para comunicação através de um protocolo TCP/IP, o protocolo a utilizar será o protocolo MODBUS TCP;
- Posição 02 – módulo de comunicação SCY 11601 para comunicação através do protocolo MODBUS com interface assíncrona RS485 (não utilizada nesta dissertação).
- O autómato não dispõe de módulos de entradas/saídas, tanto digitais como analógicas.

Este autómato estará ligado à supervisão através de uma rede *Ethernet* e ao autómato S7-200 através de uma rede de campo com o protocolo MODBUS RTU, na qual funcionará como *master* [20].

3.2.2.1 Módulo Ethernet TSX ETY 110

O módulo TSX ETY 110 é um módulo de comunicações configurável que permite o envio e receção de mensagens através de um protocolo ETHWAY (protocolo proprietário) ou TCP/IP.

Caso seja pretendido um protocolo TCP/IP é necessário configurar no módulo o seu IP, a máscara de rede (*Subnetwork mask*) e o endereço do *gateway* (*Gateway address*).

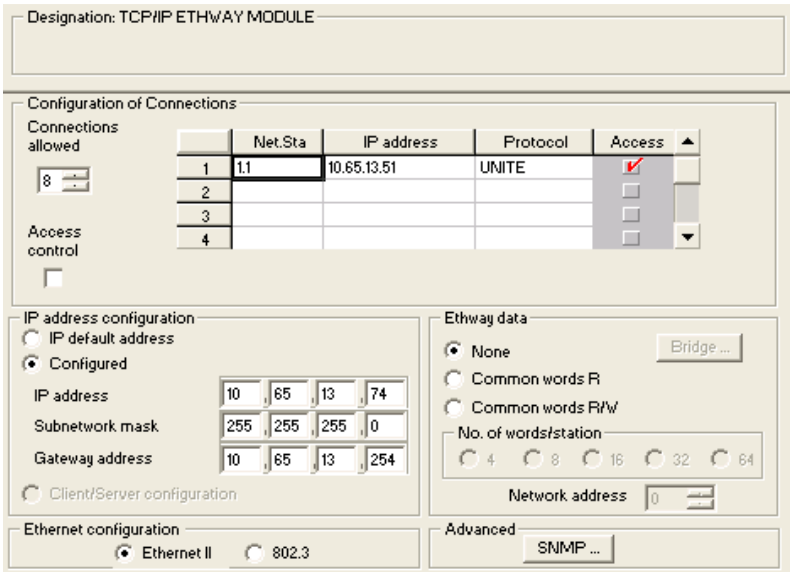

Figura 3.3 Configuração do módulo TSX ETY 110 para interface com a rede Ethernet

3.2.2.2 Módulo TSX SCP 111

O TSX SCP 111 é um módulo de comunicações configurável que permite o envio e receção de mensagens através dos protocolos MODBUS/JBUS, UNI-TELWAY, ou *Character Mode*, utilizando como interface física a norma RS232.

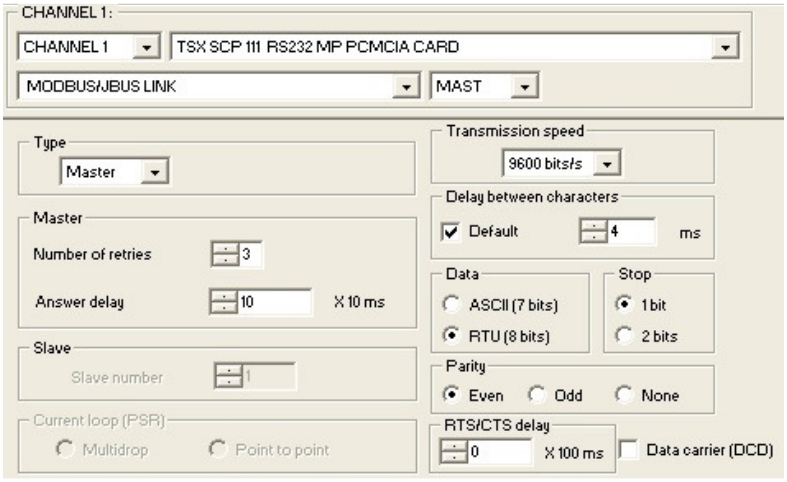

Figura 3.4 Configuração da placa TSX SCP 111

Este módulo foi configurado para o protocolo MODBUS como *master* na rede, com um *timeout* de 100 ms e 3 tentativas de comunicação, velocidade de transmissão de 9600 *bits/s*, modo RTU, com 1 *stop bit* e paridade par. O intervalo entre caracteres (*Delay between characters*) depende da velocidade de comunicação e é um dos instrumentos para deteção de mensagens de acordo com o modo RTU, em que a mensagem é considerada contínua se não for ultrapassado o intervalo de tempo definido entre caracteres e caso seja ultrapassado é considerado um final de mensagem, a confirmação se a mensagem está correta ou não será efetuada por um código de deteção de erros, neste caso o código CRC. Nesta rede a uma velocidade de 9600 *bits/s* e com 11 *bits* por caracter, o tempo previsto entre caracteres é de 4 ms.

3.2.3 TSX57-302 - Apresentação geral

Figura 3.5 O autómato TSX57-302

O autómato TSX57-302 é um autómato modular idêntico ao TSX57-103 utilizando um processador diferente, contendo os seguintes elementos:

- Fonte de alimentação PSY 2600 de 26 W a 24 VDC;
- Posição 00 – processador TSX57-302 com capacidade para controlar dois módulos de comunicação, utilizará um porto de comunicação (TER) para programação através do protocolo UNI-TELWAY LINK e um módulo de comunicação TSX SCP 111 com interface RS232;
- Posição 01 – módulo de comunicação TSX ETY 110 para comunicação através de um protocolo TCP/IP, o protocolo a utilizar será o protocolo MODBUS TCP;
- Posição 02 – módulo de comunicação SCY 21601 para comunicação através do protocolo MODBUS com interface assíncrona RS485;
- Posição 03 – módulo TSX DEY16D2 de dezasseis entradas digitais a 24VDC;
- Posição 04 – módulo TSX DSY16R5 de dezasseis saídas digitais a relé;

Este autómato estará ligado à supervisão através de uma rede *Ethernet* e simultaneamente a um analisador de energia PM500 e a um relé de proteção SEPAM através de uma rede de campo com o protocolo MODBUS RTU, na qual funcionará como *master*.

3.2.4 S7-200 - Apresentação geral

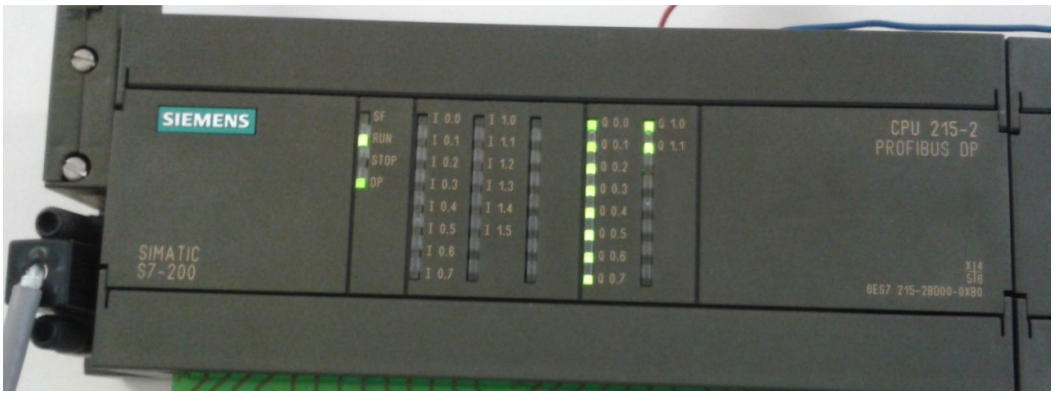

Figura 3.6 Autómato S7-200

A família de autómatos SIMATIC S7-200 contempla autómatos compactos alimentados a 24 VDC ou 230 VAC, um ou dois portos de comunicação, módulos de entradas e saídas a transístor ou relé e LEDs de indicação de funcionamento e diagnóstico, o autómato pode ainda ser equipado com módulos adicionais como módulos especiais, de comunicação, de expansão de entradas/saídas tanto analógicas como digitais, as capacidades de cada autómato estão dependentes do tipo de CPU que contém, neste caso é o CPU 215-2 DP [21].

Em termos de *hardware* o autómato está equipado com:

- 16 entradas digitais, I0.0 a I1.5;
- 10 saídas digitais, Q0.0 a Q1.1;
- Módulo EM231 de 3 entradas analógicas;
- Módulo EM232 de 3 saídas analógicas.

Em termos de *software* o autómato S7-200 será programado através do programa STEP 7 – Micro/Win em linguagem *ladder*.

Este autómato estará ligado ao autómato TSX57-103 através de uma rede de campo com o protocolo MODBUS RTU, na qual funcionará como *slave* e ao autómato S7-300 através de uma rede de campo com o protocolo PROFIBUS-DP, na qual funcionará também como *slave.*

3.2.4.1 Configurações

Os portos de comunicação do autómato S7-200 terão de ser configurados de acordo com os parâmetros definidos para cada rede, no painel *system block –> comunication ports* serão definidos os parâmetros base para as redes presentes nos portos 0 e 1, nomeadamente as redes MODBUS RTU e PROFIBUS-DP. Os parâmetros aqui configurados serão utilizados por defeito e como tal serão os utilizados para a rede PROFIBUS-DP, para a rede MODBUS RTU ao ser utilizado o modo *Freeport* estas configurações não terão efeito, a configuração do porto 0 será feita por *software* na sub-rotina 0, esta sub-rotina será executada apenas uma vez no primeiro ciclo do autómato.

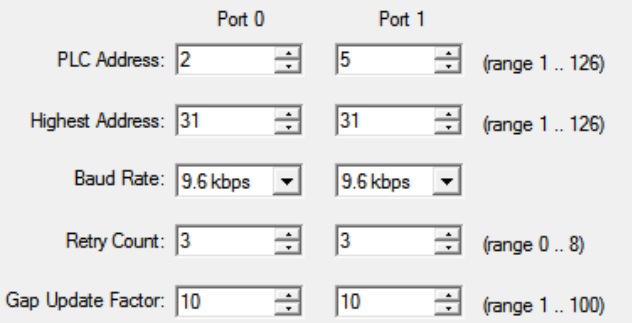

Figura 3.7 Configuração dos portos de comunicação do autómato S7-200

Para acesso à rede MODBUS RTU o autómato será ligado pelo cabo PPI *Multi-Master* ao porto 0 com interface RS485, o cabo possui pinos de configuração onde será também configurado o ritmo de transmissão de 9.6 *kbit/s*.

Este mesmo cabo será utilizado para programação do autómato, para tal será ligado a um computador (1) quando for necessário programar o autómato e será ligado ao autómato TSX57-103 (2) quando for necessário utilizar os equipamentos em rede. A conexão tanto ao computador como ao TSX57-103 será estabelecida em RS232, para tal o próprio cabo PPI *Multi-Master* dispõe de um conversor RS485/RS232.

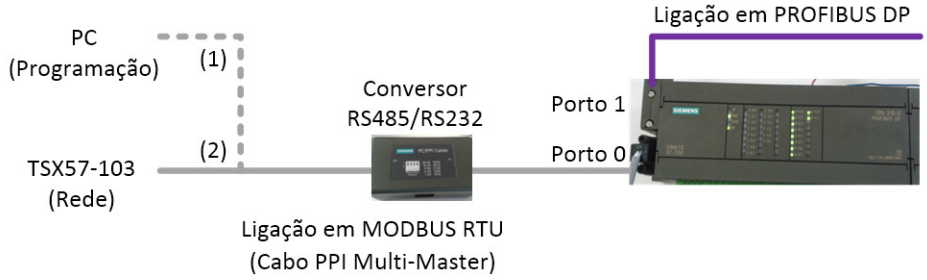

Figura 3.8 Ligações para programação e utilização em rede no S7-200

3.2.5 S7-300 – Apresentação Geral

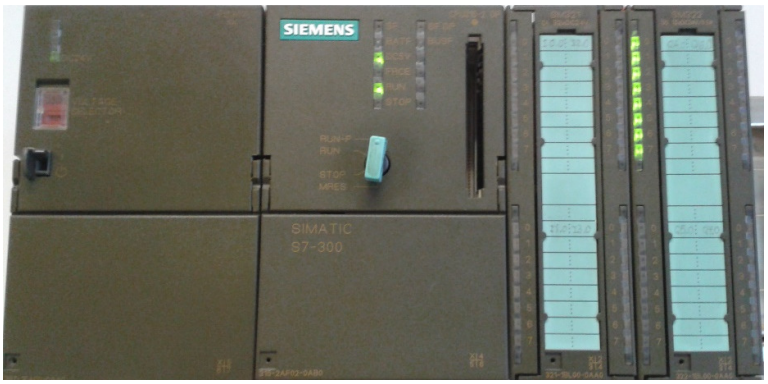

Figura 3.9 Autómato S7-300

O autómato S7-300 é composto por uma fonte de alimentação de 24 VDC, um processador CPU 315-2 DP, um módulo de sinal SM321 de 32 entradas digitais e um módulo SM322 de 32 saídas digitais, este autómato à semelhança do TSX57-103 é também totalmente modular.

O autómato contém no módulo do CPU LEDs indicadores de funcionamento, estado e erro, uma entrada para cartões de memória, um seletor de modo de funcionamento e dois portos de comunicação, o X1 com MPI (*Multi-Point Interface*) para programação e o X2 com DP para comunicação através de uma rede série com protocolo PROFIBUS-DP [22]. O porto de comunicação X2 tem de ser configurado pelo *software STEP 7 Manager.*

A programação do autómato será feita em linguagem *ladder* com algumas funções em linguagem STL (*Statement List*), a utilização da linguagem STL embora mais complexa facilita a escrita de determinadas operações tais como o endereçamento indireto a *bits* o qual será necessário para a implementação dos comandos MODBUS 01h e 0Fh.

3.2.5.1 Configuração de hardware

O autómato S7-300 sendo o *master* da rede PROFIBUS-DP terá de ser configurado para a identificação do *hardware* presente na rede, esta configuração é feita no *software STEP7 Manager* e descarregada para o autómato, nesta configuração são definidos os equipamentos presentes na rede, os seus endereços e os módulos de cada equipamento.

Na presente rede foram configurados o autómato S7-300 como *master* com o endereço 2, os restantes equipamentos são o variador de velocidade Micromaster com o endereço 3, o *gateway* DP/ASi Link 20 com o endereço 4 e o autómato S7-200 com o endereço 5.

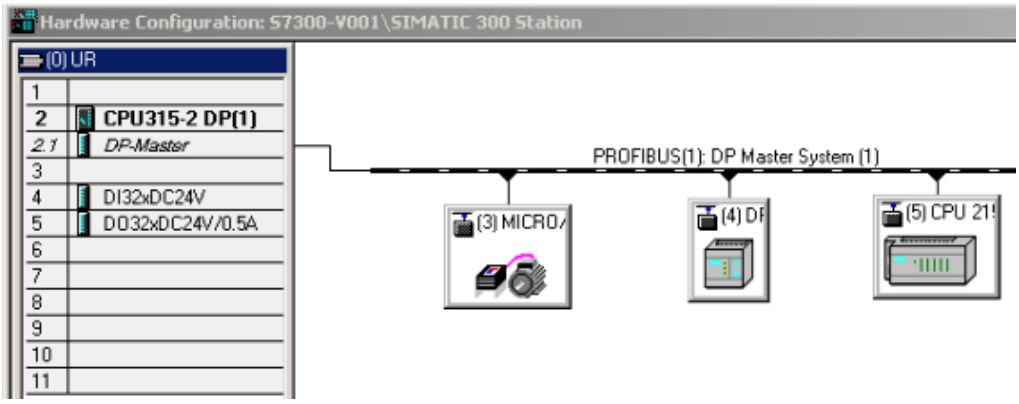

Figura 3.10 Configuração do hardware presente na rede PROFIBUS-DP

Cada equipamento presente na rede terá ainda especificado um conjunto de módulos virtuais os quais serão os endereços acessíveis através da rede PROFIBUS-DP, o *master* da rede apenas terá acesso aos módulos configurados, os endereços dos módulos serão únicos e funcionarão como endereços internos do próprio *master*, por exemplo o *gateway* DP/ASi Link 20 foi configurado com os endereços de entradas binárias de I4.0 a I19.0 estas funcionarão como sendo entradas do próprio autómato S7-300.

Tabela 3.2 Mapa de endereços na rede PROFIBUS-DP

Equipamento	**Endereços**
S7-300	I0 a I3
S7-300	Q4 a Q7
Micromaster	I256 a I263
Micromaster	Q264 a Q267
DP/ASi Link	I4 a I19
DP/ASi Link	Q8 a Q23
S7-200	I268 a I283
S7-200	Q268 a Q283

3.2.6 Outros Equipamentos

Para além dos controladores apresentados estarão ainda presentes na rede integrada outros equipamentos, nomeadamente *slaves* na rede PROFIBUS-DP e na rede ASi, todos estes equipamentos serão acedidos através do *master* da rede PROFIBUS-DP o autómato S7-300.

Equipamentos adicionais:

- Micromaster, um variador de velocidade o qual permitirá o controlo de um motor assíncrono monofásico;
- DP/ASi Link, um *gateway* entre a rede ASi e a PROFIBUS-DP, permite a um *master* da rede PROFIBUS o acesso aos dispositivos da rede ASi;
- Periféricos na rede ASi, existem na rede sensores e atuadores como fins de curso, botões de pressão, contatores e LEDs.

3.2.6.1 VSD Micromaster 420

Os variadores de velocidade Micromaster 420 são equipamentos utilizados para estabelecer uma velocidade variável em motores monofásicos ou trifásicos de 120 W até 11 kW.

O variador existente contém um módulo de comunicação CB15 para acesso à rede PROFIBUS-DP. Com a utilização deste módulo é possível configurar o variador Micromaster como um *slave* na rede permitindo o acesso à parametrização e comando do mesmo através de um *master*, neste caso o *master* será o autómato S7-300 [23]. O variador será configurado com o endereço 3.

Figura 3.11 Variador de velocidade Micromaster e respetivo módulo de comunicação CB15

O variador na rede PROFIBUS-DP utiliza dois módulos, um módulo PKW de 8 *bytes* para parametrização e um módulo PZD de 4 *bytes* para comandos.

Para comunicação entre o autómato S7-300 e o variador é necessário então enviar mensagens de 4 *bytes*, as quais serão enviadas com um bloco MOVE da mesma forma como se fosse um registo interno do próprio autómato sendo utilizado um endereço específico para o variador Q264, para o envio de mensagens de parametrização com 8 *bytes* é necessário enviar os 8 *bytes* numa única mensagem para tal é necessário utilizar um bloco DP-WRITE-DATA ou DP-READ-DATA os quais são utilizados para mensagens superiores a 4 *bytes*.

Os blocos DP-WRITE-DATA e DP-READ-DATA serão utilizados para parametrização e os blocos MOVE para comandos.

Tabela 3.3 Mapa de endereços do variador de velocidade no autómato S7-300

Módulo	**Finalidade**	**Tipo**	**Endereços**
PKW	Parametrização	Saída	Q256
PZD	Comandos	Saída	Q264

A parametrização do variador é feita através da escrita de valores no módulo PKW com o endereço Q256.

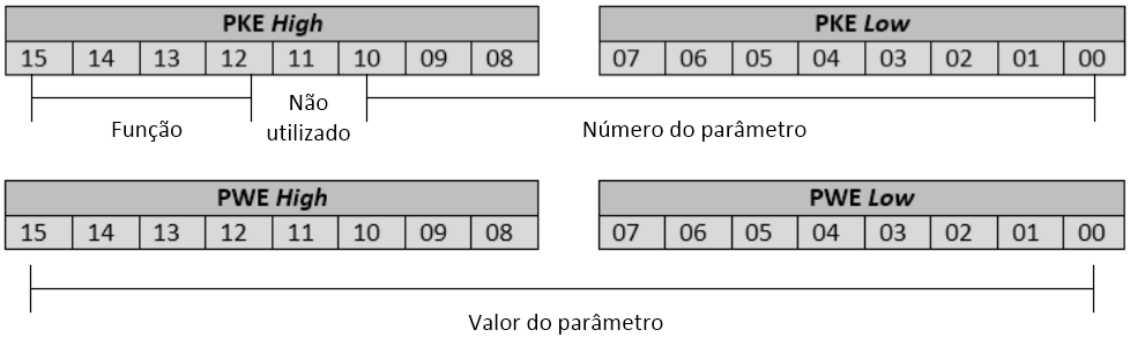

Figura 3.12 Função dos registos de parametrização PKE e PWE

As funções possíveis de serem aplicadas, os números dos parâmetros e os seus valores possíveis são específicos para cada variador e podem ser consultados no manual do equipamento em causa, por norma estes parâmetros podem ser utilizados para funções de leitura ou escrita assim como afetar a configuração do mostrador, a configuração das rampas de subida ou descida, frequências, endereços entre outros. A parametrização do variador é parte integrante da unidade curricular de Redes de Automação e Supervisão e como tal não será detalhada nesta dissertação [3].

O envio de comandos para o variador é feito através da escrita de valores no módulo PZD com o endereço Q264, é um registo do tipo *double word* de 4 *bytes* ou 32 *bits*, o qual está dividido em dois registos do tipo *word* com 2 *bytes* ou 16 *bits*, PZD1 e PZD2, destes é utilizado o registo PZD1 para envio do comando e o registo PZD2 para o envio dos dados.

O registo PZD1 tem a seguinte configuração:

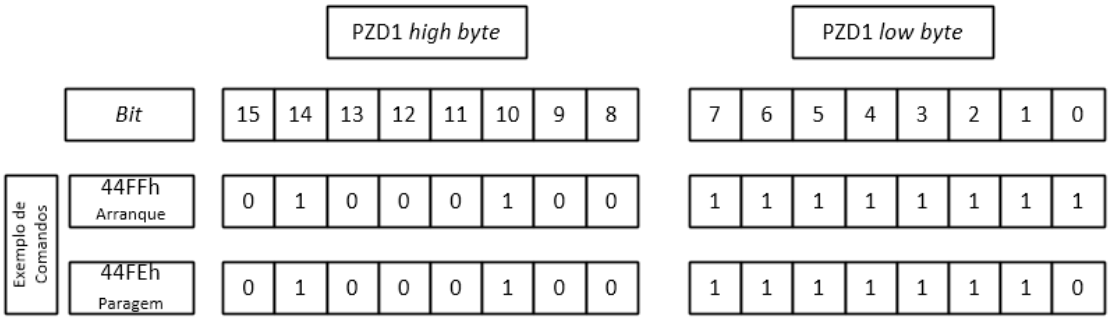

Figura 3.13 Exemplo de comandos para o variador através do registo PZD1

No registo PZD2 segue o valor pretendido para a velocidade, o valor é dado em percentagem podendo tomar os valores de 0 a 16384 em decimal ou de 0000 a 4000 em

hexadecimal a estes valores corresponderá a percentagem de 0 a 100% da frequência da rede, neste caso 50Hz.

Exemplo de comandos:

- Comando de paragem: PZD1 – 44FEh e PZD2 – 0000h;
- Comando de velocidade a 50%: PZD1 – 44FFh e PZD2 – 2000h;
- Comando de velocidade a 100%: PZD1 – 44FFh e PZD2 – 4000h;

3.2.6.2 DP/ASi Link 20

O equipamento DP/ASi Link 20 é um *gateway* entre as redes PROFIBUS-DP e a rede ASi (*Actuator Sensor Interface*), este equipamento fará a interface entre as redes permitindo assim ao *master* da rede PROFIBUS acesso aos dispositivos de mais baixo nível como atuadores e sensores localizados na rede ASi.

O DP/ASi Link será configurado como *slave* na rede PROFIBUS com o endereço de *slave* 4 configurado com acesso a 16 *bytes* de entradas e 16 *bytes* de saídas binárias.

A rede ASi é utilizada para este tipo de equipamentos de baixo nível por conter um protocolo otimizado para comunicações simples, com uma instalação de baixo custo e flexível com a possibilidade de integrar até 31 periféricos e até 124 dispositivos com tempo máximo de acesso de 10 ms [24].

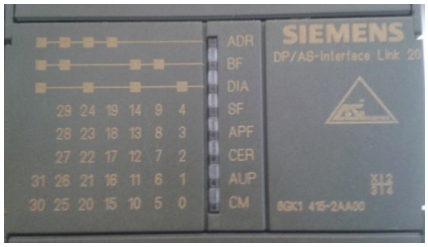

Figura 3.14 Módulo gateway DP/ASi Link 20

Os periféricos existentes na rede ASi estão divididos em dois grupos de entradas e saídas, com quatro elementos cada, o acesso aos mesmos é feito como a entradas e saídas do próprio autómato S7-300 com endereços específicos para a rede ASi, no presente caso a rede foi configurada com o endereço 9 para saídas e 5 para entradas.

Tabela 3.4 Mapa de endereços da rede ASi no autómato S7-300

Equipamento	Tipo	Endereços
Botões de pressão	Entradas	I5.0 e I5.1
Fins de Curso	Entradas	I5.4 e I5.5
LEDs	Saídas	Q9.2 e Q9.3
Contatores	Saídas	Q9.6 e Q9.7

3.3 Constituição da rede integrada

As cinco redes constituintes da rede integrada são uma rede *Ethernet* com protocolo MODBUS TCP, duas redes de campo com protocolo MODBUS RTU, uma rede de campo com protocolo PROFIBUS-DP e uma rede de campo com protocolo ASi.

Em termos de topologia a rede *Ethernet* utilizará a topologia em estrela da rede de dados, as redes com protocolo MODBUS RTU utilizarão ligação ponto a ponto, estabelecendo a ligação entre dois equipamentos e as restantes redes de campo utilizarão uma topologia em barramento típica de redes de campo de pequena dimensão e sem necessidades especiais em termos de fiabilidade ou necessidade de aumento de equipamentos na rede.

Em termos de arquitetura a estrutura da rede *Ethernet* considera-se fixa e a sua composição variável sem possibilidade de determinação do número de equipamentos instalados, considerar-se-á que a comunicação através desta rede será não determinística, com tempos de comunicação aleatórios.

A rede de campo com protocolo MODBUS RTU instalada no laboratório de pneumática é uma rede com estrutura fixa e composição fixa, é uma rede de pequena dimensão com dois equipamentos para leitura dos consumos de energia sem necessidades específicas de fiabilidade, esta rede será utilizada para recolha de informação.

A rede de campo com protocolo MODBUS RTU no laboratório de automação é o ponto da rede integrada que terá de ser compatibilizado em termos de *hardware* e com a maior complexidade em termos de *software*, será uma rede ponto a ponto entre dois autómatos de diferentes fabricantes, utilizará um ritmo de transmissão de 9600

bits/s e o meio físico será o cabo TSX SCP CD 1030 com interface RS232 no autómato TSX57-103 e cabo PPI *Multi-Master* com interface RS485 no autómato S7-200, para que a comunicação seja estabelecida com sucesso será necessário utilizar um conversor RS232/RS485 e uma caixa de adaptação para troca de pinos.

A rede de campo PROFIBUS-DP instalada no laboratório de automação é uma rede com estrutura e composição fixas, de pequena dimensão com quatro equipamentos para escrita e leitura de informação, com interfaces de acesso RS485, como meio físico utiliza o cabo PROFIBUS FC *Standard Cable* e um ritmo de transmissão de 12 *Mbits/s*.

A rede de campo ASi também instalada no laboratório de automação é uma rede com estrutura e composição fixas, contento dois repartidores ativos, o meio físico é um cabo flexível de dois condutores específico para a rede ASi e tem um ritmo de transmissão de 167 *kbits/s*.

3.4 Estrutura básica de interligação das redes

Os equipamentos constituintes das redes independentes são os já existentes no laboratório de automação e de pneumática sendo considerados nos seguintes níveis:

- **Nível Alto** – PC com programa de supervisão;
- **Nível Intermédio** – TSX57-103, TSX57-302, S7-215, S7-315, Micromaster CB15, DP/ASi Link;
- **Nível Baixo** –Fins de Curso, Botões de Pressão, LEDs.

Tabela 3.5 Composição da rede Integrada

	Tipo	Protocolo	Equipamentos	Nível	Designação
Rede Integrada	*Ethernet*	MODBUS TCP	PC	Alto	Supervisão
	Campo	MODBUS RTU	TSX57-302	Intermédio	Controlo
	Campo	MODBUS RTU	TSX57-103	Intermédio	Controlo
	Campo	PROFIBUS-DP	S7-215, S7-315, Micromaster, DP/ASi Link	Intermédio	Controlo
	Campo	ASi	S. e Atuadores	Baixo	Processo

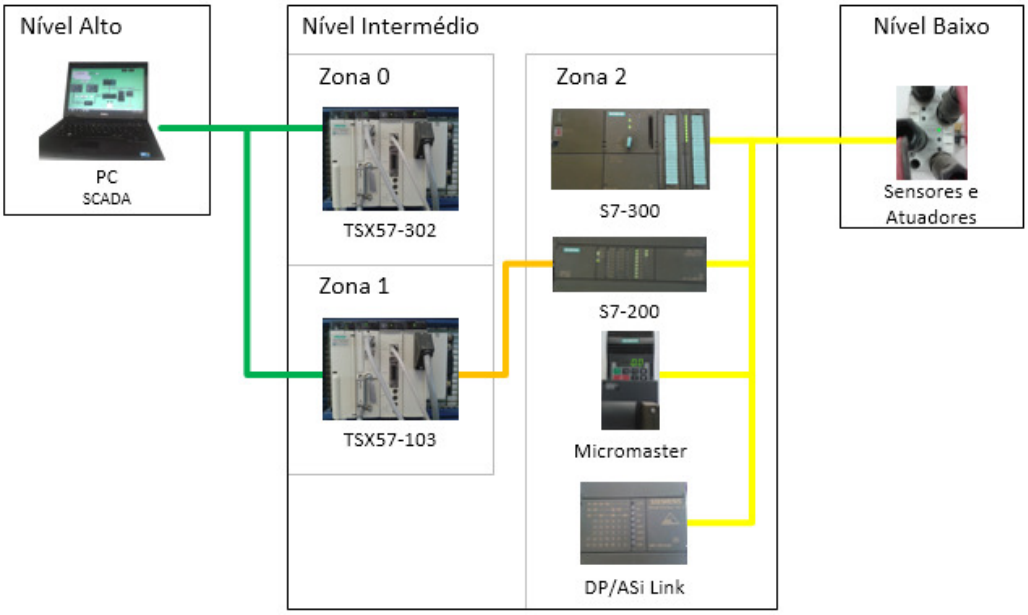

Figura 3.15 Estrutura de ligações físicas

Para interligação destes equipamentos em rede houve a necessidade de tomar diversas decisões baseadas nas suas características uma vez que os diversos fabricantes, quer seja pela sua natureza construtiva, quer seja por razões de concorrência não utilizam as mesmas redes nem os mesmos protocolos de comunicação. Foi então necessário compatibilizar as redes tanto em *hardware* como em *software*. A figura seguinte ilustra a solução adotada e a rede integrada completa.

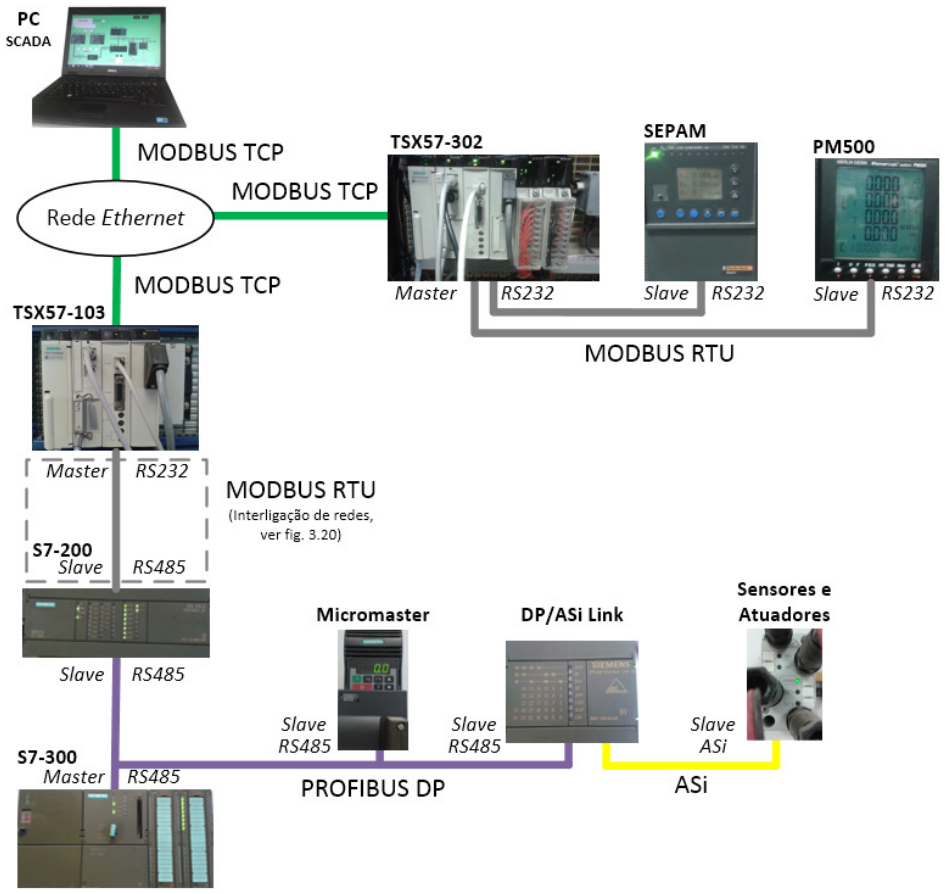

Figura 3.16 A rede integrada

Esta solução foi adotada principalmente porque ambos os autómatos TSX57 da Schneider têm capacidades de ligação a uma rede local *Ethernet* e simultaneamente módulos de comunicação para ligação a outros dispositivos do processo através do protocolo MODBUS RTU. Este fato faz deles, nesta situação, a melhor solução para o estabelecimento de uma ligação com o sistema de supervisão desenvolvido, evitando a aquisição de interfaces de rede adicionais.

Os autómatos S7-300 e S7-200 têm interfaces de comunicação RS485 para a rede PROFIBUS-DP em modo *Master-Slave*, no entanto, dada a possibilidade de programação em linguagem *Ladder* do protocolo MODBUS RTU no autómato S7-200, usando o denominado modo *Freeport*, é possível estabelecer ligação entre este controlador da Siemens e o controlador da Schneider. O modo *Freeport* permite utilizar a interface de comunicação normalmente usada para programação do autómato para efetuar comunicação série assíncrona com outros equipamentos. O autómato S7-200 é considerado nesta solução um equipamento duplamente *slave* porque é escravo na

configuração *Master-Slave* do protocolo PROFIBUS-DP com o autómato S7-300 e escravo na ligação do protocolo MODBUS RTU em modo *Master-Slave* com o autómato TSX57-103. Este autómato é um dos elementos chave na integração como descrito detalhadamente no ponto 3.4.1 deste capítulo.

O autómato S7-300 como dispositivo *master* da rede PROFIBUS-DP pode estabelecer a comunicação também com um variador de velocidade MICROMASTER ou com um módulo *gateway* DP/ASi Link 20.

Como já referido na descrição do equipamento na secção anterior o autómato TSX57-302 dispõe também de dois módulos de comunicação para o protocolo MODBUS RTU com interface RS232, um para comunicação com um relé de proteção configurável da *Merlin Gerin* (SEPAM 1000) e outro com um analisador de energia também da *Merlin Gerin* (PM500), os dados destes equipamentos ficam também acessíveis através do sistema de supervisão graças à estrutura de ligações desenvolvida.

Com a solução desenvolvida serão permitidas comunicações entre a supervisão e todos os equipamentos instalados em todas as redes independentes. Serão também permitidas comunicações entre o autómato TSX57-103 e todos os equipamentos das restantes redes. Serão ainda permitidas comunicações entre o autómato S7-300 e todos os equipamentos das restantes redes. O autómato S7-200 sendo um equipamento *slave* não tem permissões para estabelecer comunicações diretamente com outros equipamentos mas pode fazê-lo através da supervisão, do autómato TSX57-103 ou do autómato S7-300 de forma indireta. É ainda permitido a difusão de mensagens em modo *Broadcast* em todas as redes. O modo *Multicast* e o estabelecimento de mensagens prioritárias não foram desenvolvidos nesta solução.

3.4.1 Restrições da solução desenvolvida

Normalmente para se estabelecer comunicação entre duas redes com protocolos distintos é necessário utilizar uma *gateway* onde os dispositivos que querem comunicar são dispositivos usualmente *master* de cada uma das redes (quer utilizem métodos de acesso *Master-Slave*, espontâneo ou com passagem de testemunho). Tal permite que o autómato que recebe a mensagem tenha a possibilidade de reencaminhar se necessário essa informação para outros dispositivos *slave*. Por imposição da estrutura dos equipamentos existentes, a interligação entre as redes com protocolo MODBUS RTU e protocolo PROFIBUS-DP será feita através do autómato S7-200 o qual é *slave* em

ambos como já foi referido anteriormente. Dado que os equipamentos *slave* não têm permissão para iniciar uma transmissão este aspeto foi determinante para a procura de uma solução alternativa viável e condicionou em parte a solução a desenvolver. Note-se que quando se pretende enviar uma mensagem do autómato TSX57-103 para o autómato S7-300 (ou ao contrário) através da rede PROFIBUS-DP este terá que passar forçosamente pelo autómato S7-200, uma vez que este não tem permissão para iniciar a comunicação foi necessário adotar um método indireto.

A solução adotada para a resolução deste problema foi a criação de *buffers* de memória no autómato S7-200 para a receção e envio de mensagens MODBUS RTU. No caso de serem mensagens que têm como destino a rede PROFIBUS-DP estas serão encaminhadas (depois de validadas) para outros dois *buffers* de memória dedicados à rede PROFIBUS-DP. Uma vez que o autómato S7-200 não tem permissão para reenviar mensagens é então necessário utilizar no autómato S7-300 (o *Master* da rede PROFIBUS-DP) uma técnica designada por *polling* ao *buffer* da memória que foi disponibilizada pelo S7-200 para este efeito, ou seja, o autómato S7-300 irá ciclicamente efetuar uma leitura do *buffer* de envio do S7-200 e caso exista uma mensagem nova irá interpretar a mesma, enviando posteriormente a resposta para o *buffer* de receção do S7-200 de acordo com a solicitação efetuada. O autómato S7-200 por sua vez irá transferir os conteúdos do seu *buffer* de receção da rede PROFIBUS-DP para o *buffer* de envio da rede MODBUS RTU seguidamente utilizando a informação contida no mesmo para responder ao TSX57-103 que desencadeou originalmente a mensagem. Este procedimento encontra-se representado na figura seguinte.

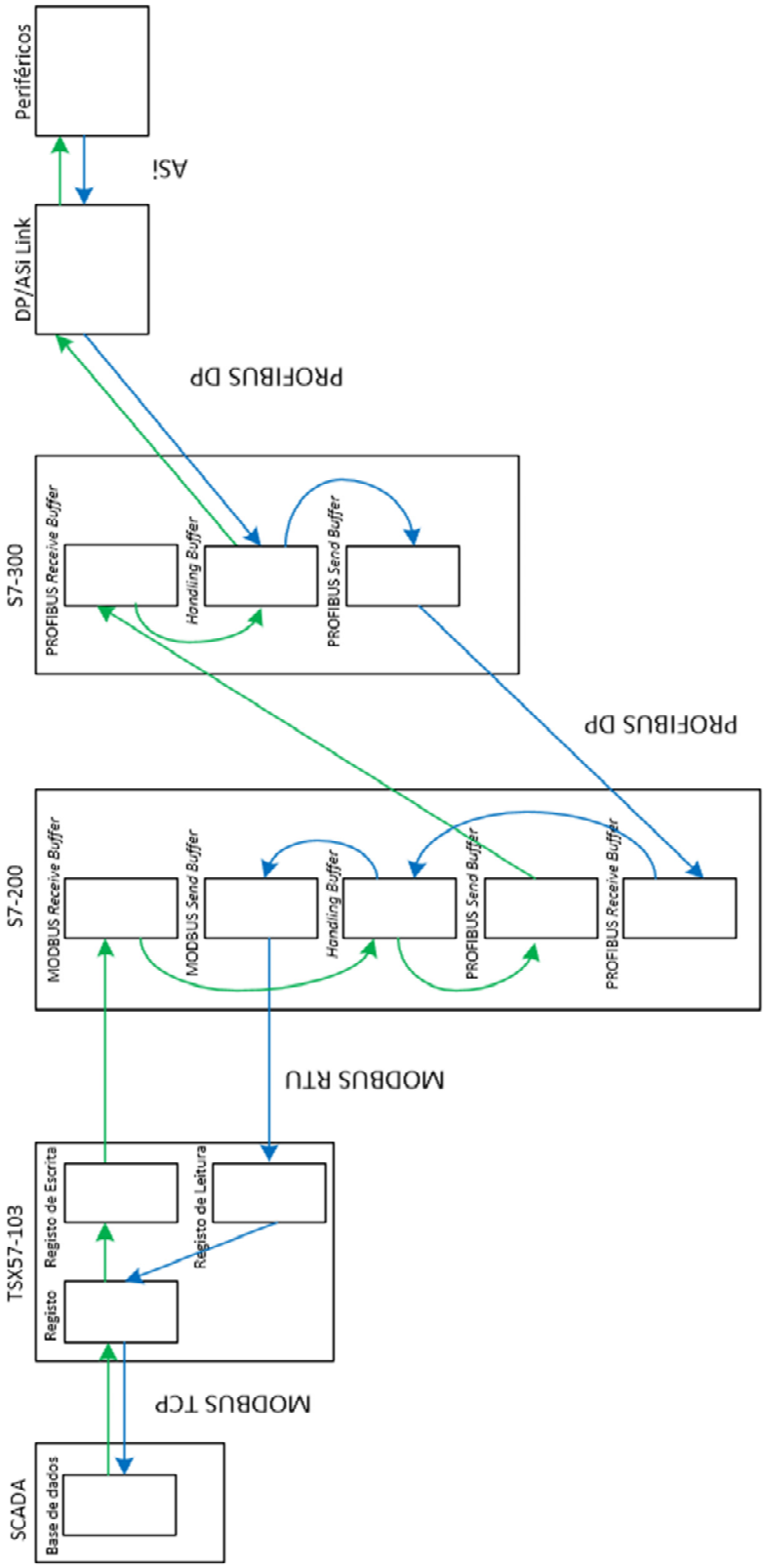

Figura 3.17 Transmissão de mensagens entre redes

É de referir que com esta técnica de *polling* ao *buffer* efetuada ciclicamente do S7-300 para o S7-200, havendo ou não mensagens novas, irá ocupar tempo de processamento e aumentar o tráfego na rede com mensagens desnecessárias por não existir qualquer pedido. No caso específico da rede PROFIBUS-DP do laboratório de automação sendo esta de pequena dimensão com um ritmo de transmissão elevado (12 *Mbits/s*) e poucos dispositivos este fato não trará qualquer inconveniente. Numa rede com um número elevado de dispositivos e com grande tráfego esta técnica pode causar atrasos significativos nos restantes equipamentos.

Para que esta operação seja possível é necessário que o tempo despendido desde que o autómato S7-200 recebe uma mensagem em MODBUS RTU até que o autómato TSX57-103 receba a resposta seja inferior ao tempo de *timeout* estabelecido para o autómato TSX57-103. A figura seguinte ilustra este aspeto.

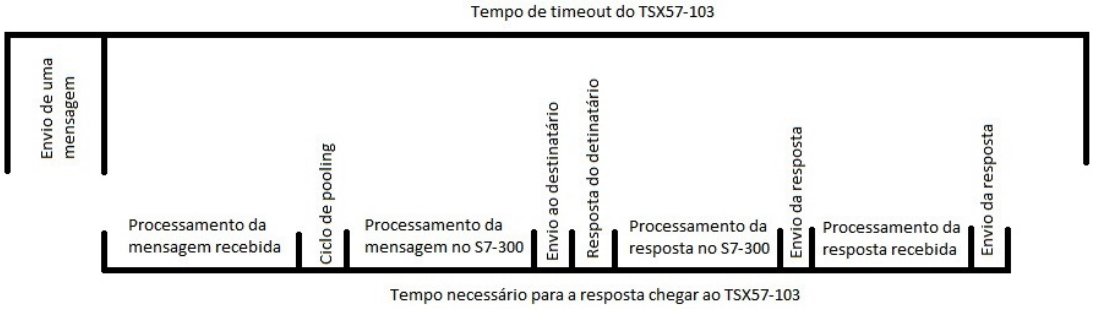

Figura 3.18 Tempos para a transmissão de uma mensagem entre o autómato TSX57-103 e o autómato S7-300

Note-se que a receção de mensagens MODBUS RTU pelo autómato S7-200 feita em modo *Freeport* não é processada diretamente pelo módulo de comunicação mas pelo próprio autómato no seu ciclo de funcionamento, assim sendo a descodificação e validação de mensagens tornam-se mais lentas em especial para mensagens grandes devido aos cálculos necessários para a deteção de erros por CRC e apenas pode ser processada uma mensagem de cada vez. Por esta razão e por questões de economia de recursos os *buffers* de memória utilizados no autómato S7-200 serão limitados a vinte *bytes*, este valor permite a concretização de todas as mensagens previstas no protocolo MODBUS RTU sem restrições significativas apenas limitando a dimensão de algumas mensagens em termos de quantidade de dados, não utilizando demasiados recursos de memória e processamento permite assim que a troca de mensagens entre as duas redes se concretize com sucesso.

3.4.2 Ligação entre o sistema de supervisão e os autómatos TSX

A comunicação por *Ethernet* será estabelecida entre a supervisão e os autómatos TSX57. Após a configuração do módulo de comunicações ETY 110 e de efetuadas as ligações físicas torna-se necessário proceder à configuração da aplicação SCADA desenvolvida. A configuração da aplicação SCADA será descrita em detalhe no capítulo 4.

Existirão dois tipos de mensagens, diretas e indiretas, mensagens diretas da aplicação SCADA para o TSX57-103 ou para o TSX57-302 e indiretas para os restantes equipamentos. As mensagens diretas apenas necessitam de ser executadas pelo autómato e na zona de dados da trama da mensagem apenas serão enviados/recebidos dados. Para as comunicações indiretas terão de ser enviados, o endereço do equipamento, o comando a executar, o endereço da variável a afetar e os dados, este tipo de mensagem ao serem recebidas pelo autómato TSX57-103 serão reencaminhadas para a rede de campo em MODBUS RTU.

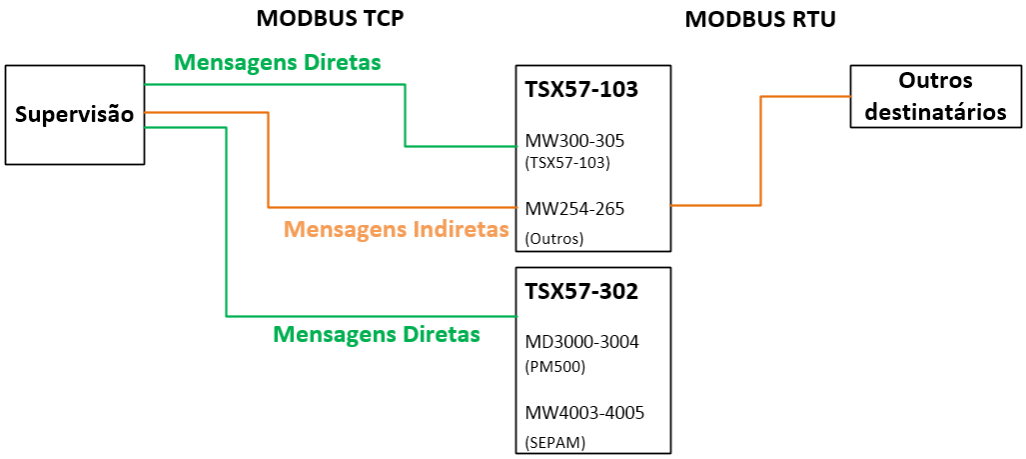

Figura 3.19 Encaminhamento para comunicações via Ethernet

Estarão disponíveis vários comandos através desta ligação, escrita de uma/várias saídas, leitura de uma/várias entradas/saídas, escrita e leitura de um/vários registos, destas serão utilizadas a escrita e leitura de um/vários registos para mensagens diretas para os autómatos TSX57, visto o autómato TSX57-103 não dispor de módulos de

entradas/saídas serão utilizadas as mensagens de escrita/leitura de saídas binárias e a escrita/leitura de registos para os restantes equipamentos.

Os registos estão localizados na zona de memória V dos autómatos. É possível ler/escrever em qualquer local da memória V dos autómatos, no entanto, o programa criado no autómato TSX57-103 para gerir o trafego de mensagens e o programa de exemplo de uma instalação industrial utilizam também essa zona de memória para as suas variáveis. De modo a não ocorrer sobreposição de valores os endereços de memória a serem escritos via *Ethernet* serão fixos. Para a comunicação direta entre a supervisão e o autómato TSX57-103 serão utilizadas as posições 300 a 305 e para comunicação com os restantes equipamentos serão as posições 254 a 265. O autómato TSX57-302 utilizará as posições de memória MD3000 a MD3004 para guardar o valor de potência ativa, reativa e aparente provenientes do analisador de energia PM500 e as posições de memória MW4003 a MW4005 para as correntes de fase do relé de proteção SEPAM.

Tabela 3.6 Mapa de memória para comunicações via Ethernet

	Mensagem	**Registos afetados**
TSX57-103	Escrita/leitura de um registo	MW300
	Escrita/leitura de múltiplos registos	MD302 (MW302 + MW303)
TSX57-302	Escrita/leitura de um registo	MW4003 a MW4005
	Escrita/leitura de múltiplos registos	MD3000 a MD3004
Outros Equipamentos	Diversas	MW254 a MW264

3.4.3 Ligação entre o autómato TSX57-103 e o S7-200

A ligação entre o autómato TSX57-103 e o autómato S7-200 é efetuada através do protocolo MODBUS RTU com método de acesso *Master-Slave*. Para isso o autómato TSX57-103 utiliza um módulo de comunicação TSX SCP 111 com interface de comunicação série RS232. O módulo de comunicação TSX SCP 111 vem com o cabo TSX SCP CD 1030 com uma ficha DB25 com adaptador para DB9. No caso do autómato S7-200 este possui uma porta de comunicação em modo *Freeport* (porto 0)

que como o nome indica é para utilização livre com interface série RS485, sendo utilizado o cabo PPI *Multi-Master* que possui um conversor para adaptação da interface RS485 para RS232. A ligação física entre as redes é feita em uma caixa de interligação. A figura 3.20 ilustra essa ligação.

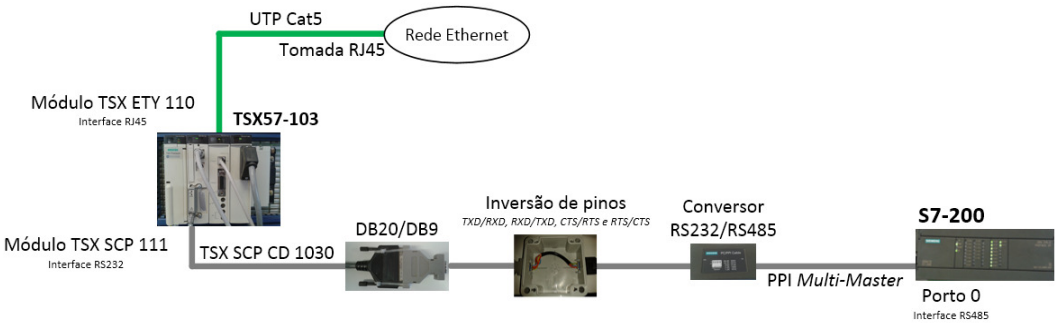

Figura 3.20 Interligação entre os autómatos TSX57-103 e S7-200

Para que seja possível estabelecer a comunicação em ambos os sentidos foi necessário inverter os pinos de envio e receção de dados assim como os de pedido de acesso ao meio físico e aceitação do pedido, os pinos TXD, RXD, RTS e CTS. Este procedimento encontra-se ilustrado na tabela 3.7.

Tabela 3.7 Troca de pinos a efetuar para compatibilização da comunicação

DTE 1		**DTE 2**	
Função	**Pino**	**Pino**	**Função**
TXD	2	3	RXD
RXD	3	2	TXD
CTS	7	8	RTS
RTS	8	7	CTS
Comum	5	5	Comum

Utilizou-se então uma caixa de interligação onde é feita a troca dos pinos assim como a troca das terminações em ficha fêmea para fichas macho onde ambos os cabos dos autómatos serão ligados (ver figura 3.21).

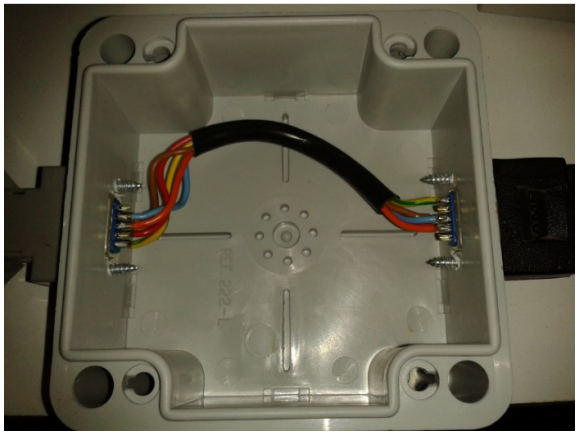

Figura 3.21 Caixa de interligação

Nesta ligação o autómato TSX57-103 é o dispositivo *master* e o autómato S7-200 o s*lave*, podendo a comunicação ser do tipo *Unicast* (mensagem com resposta) ou *Broadcast* (mensagem enviada para todos os *slaves* em simultâneo). No caso do autómato TSX57-103 a configuração dos parâmetros do protocolo é bastante simples e já foi descrita anteriormente quando se apresentou a descrição geral deste PLC e do módulo de comunicação TSX SCP 111 (ver fig. 3.4). Os parâmetros foram definidos para 9600 *bits/s*, 8 *bits* de dados, 1 *stop bit* e paridade par. O mesmo não acontece no autómato S7-200 que não dispõe de módulo dedicado de comunicação para o protocolo MODBUS RTU tendo que ser programado, as configurações e o próprio processamento de mensagens tornam-se bastante mais complexos como será visto mais adiante.

3.4.4 Considerações sobre o protocolo MODBUS RTU no TSX57-103

O processamento das mensagens no autómato TSX57-103 é feito de duas formas distintas, mensagens diretas (quando destinadas ao próprio autómato) e mensagens indiretas (quando têm como destino outros dispositivos).

As mensagens diretas são configuradas na supervisão para afetarem os registos indicados na tabela 3.6 e processadas automaticamente, no envio pelo *driver* da supervisão e na receção pelo módulo de comunicação (TSX ETY 110) do próprio autómato, como tal não requerem qualquer tratamento sendo os registos atualizados de forma automática.

Nas mensagens provenientes da supervisão com destino a outros dispositivos é necessário efetuar um tratamento prévio desses dados, são utilizadas quatro mensagens diretas para preencher um *buffer* de envio (MW254 a MW264) com as informações necessárias, nomeadamente o comando, o endereço do *slave*, o endereço da variável e o valor da mesma para serem posteriormente enviados através do módulo de comunicação TSX SCP 111 com o protocolo MODBUS RTU para o autómato S7-200. Se este não for o dispositivo de destino final será reencaminhada para outro dispositivo através deste autómato. A utilização do *buffer* permite um maior controlo sobre o número de mensagens a enviar e seus tempos para que não existam sobreposições de envio/receção. Note-se que o ciclo de processamento do programa instalado no PLC pode ser muito mais curto que o tempo total necessário para o envio e receção de mensagens e que pelas restrições apresentadas no ponto 3.4.1 apenas pode ser enviada uma mensagem de cada vez para o autómato S7-200, no caso de envio de mais mensagens as mesmas não serão processadas. Antes de cada envio de mensagem é feita uma verificação ao estado da comunicação através de um *bit* de controlo (MW90:X0), no caso de a rede estar ocupada o envio da mensagem só será feito no próximo ciclo do programa.

A forma como os autómatos TSX enviam as mensagens do protocolo MODBUS RTU não é exatamente igual ao implementado originalmente nos autómatos da MODICON uma vez que estes utilizam duas funções distintas para leitura e escrita de dados, as funções READ_VAR e WRITE_VAR (através do *software* PL7). Sendo assim os quatro primeiros códigos de função (funções de leitura) originalmente apresentadas pela MODICON, ou seja, os códigos 01h a 04h fazem agora parte da função READ_VAR e os restantes quatro fazem parte da função WRITE_VAR, os códigos 05h a 10h. O código de controlo de erros por CRC é calculado e inserido na mensagem automaticamente pelas funções de leitura e de escrita. A figura seguinte ilustra os códigos de função a utilizar e a sua ligação às funções.de leitura e escrita.

Modbus request	Function code	PLC object
Read n output bits	16#01	%M
Read n input bits	16#02	%M
Read n output words	16#03	%MW
Read n input words	16#04	%MW
Write an output bit	16#05	%M
Write an output word	16#06	%MW
Write n output bits	16#0F	%M
Write n output words	16#10	%MW

Figura 3.22 Códigos e símbolos das funções MODBUS em PL7

O mesmo sucede em relação à dimensão dos registos a ler ou escrever, quando se refere a *bits* é utilizado o símbolo %M e quando se refere a palavras de 16 *bits* é utilizado o símbolo %MW na composição da mensagem. No caso concreto do autómato TSX57-103 para comunicação com o S7-200 serão utilizados os códigos de comando 01h e 03h para leitura e os códigos 0Fh e 10h para a escrita, sendo utilizados os símbolos %M para as funções 01h e 0Fh e o símbolo %MW para as funções 03h e 10h.

Em relação ao endereço dos *slaves* visto a rede integrada ser relativamente pequena os endereços serão distribuídos por zonas sendo o primeiro algarismo referente à zona e o segundo ao *slave*, para a zona 1 os endereços serão do 11h ao 19h, para a zona 2 do 21h ao 29h. O endereço 00h será utilizado para difusão de mensagens (*broadcast*) para toda a rede, o endereço 10h para difusão na zona 1 e o endereço 20h na zona 2. A tabela 3.8 ilustra o mapa de endereços possíveis na rede.

Tabela 3.8 Mapa de endereços via MODBUS RTU

	Broadcast	*Slaves*
Rede integrada	00h	
Zona 1	10h	11h a 19h
Zona 2	20h	21h a 29h

Nota: os equipamentos na zona 3 serão acedidos através do autómato S7-300 da zona 2 como posições de memória e entradas/saídas deste devido à forma de funcionamento da rede PROFIBUS-DP e ASi. Assim os pedidos destinados a estes deverão ser feitos ao S7-300 como se se tratasse de uma entrada/saída ou um registo de memória do mesmo.

Apresenta-se na figura seguinte um exemplo de aplicação de cada uma das funções referidas implementadas pelos autómatos TSX, a função WRITE_VAR e READ_VAR.

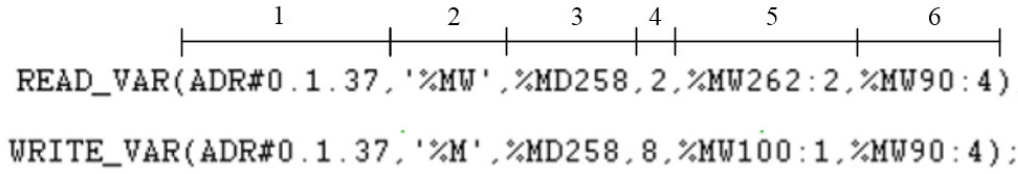

Figura 3.23 Funções de escrita e leitura em PL7

Nestes exemplos a mensagem será enviada para o módulo de comunicações instalado na posição 0 do bastidor (próprio CPU), canal 1 (TSX SCP 111), e o endereço do *slave* de destino da mensagem será o 37 ou 25h, zona 2, *slave* 5, neste caso o autómato S7-200 (1). No segundo campo (2) está indicado o tipo de registos, neste caso será uma palavra de 16 *bits* para leitura (%MW) e um conjunto de *bits* (%M) para a escrita. No terceiro campo (3) é então indicado o endereço da primeira variável, esse endereço está contido na posição de memória MD258 (endereçamento indireto). O quarto valor (4) (neste caso o valor 2 e o valor 8) indica o número de variáveis a ler ou escrever. Seguidamente (5) são indicadas as posições de memória onde os dados devem ser guardados após uma instrução de leitura (MW262 e MW263) ou de onde devem ser enviados no caso de uma instrução de escrita (MW100). Por fim é indicado um *buffer* de quatro registos de 16 *bits* (MW90 a MW93) (6) onde serão guardados elementos de

controlo do envio/receção da mensagem, tais como o número de mensagens efetuadas e o *bit* de rede ativa assim como relatórios de operação e de comunicação.

Os registos MW92 e MW93 são previamente preenchidos com os valores que se pretendem para o *timeout* e tamanho da mensagem em *bytes* respetivamente.

Word number	Most Significant Byte	Least Significant Byte
%MW*k*	Exchange number	Activity bit
%MW*k*+1	Operation report	Communication report
%MW*k*+2	Timeout	
%MW*k*+3	Length	

Figura 3.24 Registos de controlo de mensagens

O controlo de erros no processamento de mensagens será feito através do registo MW91 que contém os relatórios de operação e de comunicação, caso algum dos relatórios de operação ou comunicação apresente um valor maior que zero implica que existiu um erro na comunicação.

```
IF %MW91>0 THEN (* NO RESPONSE FROM THE COOLING TOWERS *)
     SET %MW2:X15;(* SET WARNING *)
END_IF;
```

Figura 3.25 – Exemplo de controlo de erros por análise ao valor de MW91 (Operation report e Communication report)

A figura 3.26 demonstra um exemplo completo da estrutura utilizada para o envio de uma mensagem proveniente da supervisão para o autómato S7-200 (endereço 37 ou 25h) para execução do comando 0Fh (escrever múltiplas saídas binárias).

```
IF %MW254=37 THEN
     IF %MW256=16#000F THEN
     IF NOT %MW90:X0
          THEN
          %MW90:4:=0;              (* MSG CONTROL RESET *)
          %MW93:=3;                (* TIME OUT IN 3*100ms *)
          %MW100:1:=%MW260;        (* MSG TO BE SENT *)
          WRITE_VAR(ADR#0.1.37,'%M',%MD258,8,%MW100:1,%MW90:4);
     END_IF;
     END_IF;
```

Figura 3.26 Exemplo completo de programação para o envio de mensagens segundo o protocolo MODBUS RTU

3.4.5 Programa residente no TSX57-103

A programação do autómato TSX57-103 é feita através do *software* PL7 em linguagem ST (*Sructured Text Language*), estando contemplada uma função principal (MAST) com um tempo de ciclo medido de 5 ms e sub-rotinas (SRx) para processamento de pedidos e instruções. O tempo máximo permitido para o programa é de 500 ms, limitado pelo temporizador *watchdog*. As entradas e saídas serão atualizadas por *polling* a cada ciclo do programa total, MAST e SRx utilizadas. A figura 3.27 ilustra as operações, e a sua ordem, a efetuar em um ciclo, atualização de entradas, processamento do programa e atualização de saídas.

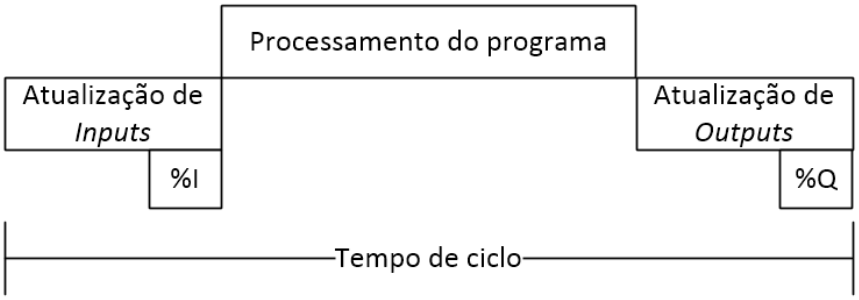

Figura 3.27 Ciclo do autómato

O tempo máximo de ciclo trouxe algumas dificuldades de programação no caso de o programa ter de processar várias mensagens por ciclo. O tempo de envio e de receção da resposta não é constante e tem um tempo máximo de *timeout* de 300 ms no caso de não haver resposta. No caso de o programa conter apenas uma mensagem por ciclo não existe problema mesmo atingindo o *timeout*. No entanto, caso existam duas ou mais mensagens por ciclo terão de ser programados contadores e temporizadores para controlo do programa que permitam a renovação de ciclos entre mensagens, caso contrário poderá ser acionado o temporizador de proteção *watchdog* (tempo máximo de ciclo de 500 ms).

Note-se que as instruções WRITE_VAR e READ_VAR ao serem executadas enviam os dados para o módulo de comunicação TSX SCP 111 sendo necessário programar uma estrutura de controlo para evitar o envio simultâneo de mensagens ou a tentativa de utilização da rede enquanto está ocupada. Para evitar o acesso simultâneo à rede deve ser utilizada uma estrutura de exclusividade, neste trabalho serão utilizadas duas formas, a utilização de *bits* de controlo e por temporização. Os *bits* de controlo

são utilizados quando as mensagens são provenientes do sistema de supervisão, o *bit* 0 e o *bit* 1 do registo MW264 serão ativados pela supervisão e desencadearão o envio de uma mensagem de escrita ou leitura consoante for o *bit* MW264:X0 ou MW264:X1.

No caso de a mensagem ser proveniente do próprio programa de aplicação existente no controlador é utilizada uma temporização de modo a permitir enviar uma mensagem e aguardar pela resposta antes de enviar uma nova mensagem, este modo implica que o programa tenha de aguardar sempre um tempo fixo pela resposta independentemente do tamanho da mensagem que enviou, esta temporização é necessária porque o autómato TSX57-103 envia uma mensagem ao autómato S7-200 e aguarda a reposta deste mas esta pode não ser proveniente diretamente do S7-200, podendo ter origem em outro equipamento, como por exemplo o S7-300. Note-se que o módulo de comunicação TSX SCP 111 tem capacidade para processar até oito mensagens MODBUS RTU de forma sequencial, no entanto o autómato S7-200 tal como referido nas limitações da rede apenas pode processar uma mensagem de cada vez e o envio de uma segunda mensagem não será processada enquanto o autómato S7-200 não processar a primeira. Os exemplos seguintes ilustram a estrutura de controlo por *bit* e por temporização.

```
! %L5:
  IF(%MW264:X0 OR %MW264:X1)THEN (* COMMS TEST FROM MAIN MENU *)
       SR6;
  END_IF;
  (* MW264:X0-SEND WRITE MSG / MW264:X1-SEND READ MSG *)
!
```

Figura 3.28 Exemplo de estrutura de controlo de mensagens por bit

```
IF %MW10>0 THEN      (* DELAY *)
     DEC %MW10;
     SR3;
     JUMP %L99;
END_IF;
```

Figura 3.29 Exemplo de chamada de uma rotina de espera (delay) para controlo de mensagens por temporização

A programação das funções de escrita e leitura e encaminhamento de mensagens necessárias proposto nesta dissertação encontram-se na rotina principal MAST e sub-rotina SR6, o programa de aplicação encontra-se na rotina principal MAST e sub-rotinas SR0, SR3 e SR5.

3.4.6 Considerações sobre o protocolo MODBUS RTU no S7-200

O autómato S7-200 da Siemens é o equipamento que fará a ligação entre o protocolo MODBUS RTU e a rede PROFIBUS-DP, ou seja, funcionará como uma *gateway* por *software* entre estes, dado que o autómato S7-200 não tem módulo de comunicações que permita o processamento automático de mensagens deste tipo. No entanto, implementando o protocolo MODBUS RTU em um programa de aplicação, configurando o porto zero (o porto zero é muitas vezes designado por "porta" de programação) em modo *Freeport* passa a ser possível efetuar esta tarefa apesar de ser bastante mais complexo.

O programa desenvolvido tem duas componentes fundamentais, um programa genérico de automação e o programa para processamento de mensagens, o programa para processamento de mensagens será o foco deste trabalho e é independente do programa genérico, é acionado por uma interrupção e desenvolve-se ao longo de diversas sub-rotinas. O programa genérico pode ser um qualquer programa de automação instalado no autómato, neste caso será utilizado o programa necessário para a disciplina de RAS desenvolvido na rotina principal (*Main*) o qual transfere o valor das entradas para posições de memória.

A implementação do protocolo MODBUS realizada neste trabalho segue as especificações descritas nos documentos MODBUS *Aplication Protocol Specification* para a implementação da camada de aplicação do modelo OSI (camada 7) *e Modbus Serial Line Specification and Implementation Guide* para a implementação das camadas física e de ligação (camadas 1 e 2) de modo a permitir que a programação do protocolo seja de acordo com as especificações originais e aplicável a qualquer equipamento que utilize o protocolo MODBUS em modo RTU, sendo que o protocolo não será programado na sua totalidade, serão implementadas as funções de escrita e leitura mais comuns e suficientes para a utilização da rede, não serão implementados os códigos de função dos contadores de diagnóstico.

3.4.7 Programa residente no S7-200

Como referido anteriormente a programação do autómato S7-200 será feita através do *software* STEP7 MicroWin SP9 em linguagem *Ladder*. A programação do protocolo será feita em sub-rotinas do programa principal de acordo com as tabelas 3.9 e 3.10.

Tabela 3.9 Composição do ciclo principal

Ciclo Principal (*Main*)	**Função**
Sub-rotina 0 – Apenas no primeiro ciclo	Configurações
Programa Genérico	
Sub-rotina 1 – Apenas após nova mensagem	Processamento de mensagem

Após a receção de uma mensagem em MODBUS RTU o programa será encaminhado para a sub-rotina 1, a qual iniciará o processamento da nova mensagem e chamará as restantes sub-rotinas consoante necessário, sendo feita uma análise ao código CRC da trama para verificação da integridade da mesma e após validação a mensagem será processada. No caso do endereço do *slave* ser o do próprio autómato S7-200 a mensagem será encaminhada para as sub-rotinas de processamento do comando entretanto programadas, após o processamento do comando será enviada uma mensagem de resposta. No caso de o endereço não ser o do autómato S7-200 a mensagem será encaminhada para o *buffer* de envio da rede PROFIBUS-DP, este *buffer* será lido ciclicamente pelo autómato S7-300. Na tabela 3.10 encontram-se especificadas as sub-rotinas e respetivas funções que implementam assim como a camada do modelo OSI a que pertencem e o documento normativo.

Tabela 3.10 Sub-rotinas implementadas e suas funções

Camada	Modelo OSI	Protocolo	Sub-rotina	Função
7	Aplicação	*Aplication Protocol Specification*	12	Resposta Normal
			11	CRC Verificação
			9	Resposta de Erro
			8	CRC Cálculo
			7	Função 10h
			6	Função 0Fh
			5	Função 06h
			4	Função 05h
			3	Função 03h e 04h
			2	Função 01h e 02h
			1	Início de Receção
2	Ligação	*Serial Line Specification*	0	Configurações
1	Meio Físico	RS485	-	Ligações Físicas

Das sub-rotinas implementadas serão aqui detalhadas as sub-rotinas 0, 1, 3, 8, 11, 9 e 12. Existirá ainda a interrupção 0 para processamento da receção de uma mensagem e a sub-rotina 13 para transferência de mensagens para a rede PRODIBUS DP a qual será detalhada na ligação entre os autómatos S7-200 e S7-300.

3.4.7.1 Sub-rotina 0 – Configurações

A sub-rotina zero compreende três configurações, a configuração do porto 0 em modo *Freeport*, do protocolo MODBUS RTU e dos *buffers* de memória. Esta sub-rotina é executada no primeiro ciclo após o reinício do autómato sendo chamada pelo *bit* especial (*Special Memory bit*) SM0.1.

Configuração 1 - Porto 0 em modo Freeport

A configuração é efetuada através dos seguintes *bytes* de memória do sistema:

SMB30 – Freeport Control Register

Neste *byte* é possível configurar o protocolo a utilizar através de SMB30.0 e SMB30.1. O valor 01h nestes dois *bits* seleciona o protocolo Freeport, nos restantes *bits* são configuradas a velocidade de transmissão, o número de *bits* de dados e a paridade.

Configuração utilizada: Paridade Par, 8 *bits*, 9600 *bits/s*, *Freeport* – 0100 1001(bin) – 49(hex)

SMB87 – Receive Message Control Byte

O *byte* SMB87 permite configurar os critérios de identificação de uma mensagem, neste *byte* pode ser configurada a utilização dos temporizadores entre mensagens e entre caracteres SMW90 e SMW92, assim como definidas as condições de início e fim de mensagem, para o modo RTU serão utilizados tempos entre mensagens. Para o modo ASCII seriam utilizados caracteres especiais como delimitadores de mensagem.

Configuração utilizada: Receção ativa, utilizar SMW90, utilizar SMW92 – 1001 0100(bin) – 94(hex)

SMB94 – Máximum number of Characters to be received

O valor introduzido neste *byte* indicará o número máximo de caracteres recebidos em uma só mensagem, este *byte* pode tomar valores entre 1 e 255 tento neste caso o valor de 19 caracteres ou *bytes*, a estes será acrescentado um *byte* com o tamanho da mensagem recebida perfazendo os 20 *bytes* de tamanho utilizado nos *buffers* de memória.

Configuração utilizada: 19(dec)

SMW90 – Idle line timer

O registo SMW90 terá o valor do tempo em milissegundos após o qual a linha é considerada livre, este tempo é também utilizado para indicar o fim de uma mensagem, o valor a utilizar será o especificado no protocolo MODBUS em modo RTU de 3,5 vezes o tempo de um caracter, a 9600 *bits/s* e com 11 *bits* por caracter o tempo de *idle line* seria 4 ms, será utilizado 6 ms de modo a garantir a receção do último caracter e a estabilização da rede.

Configuração utilizada: 6 ms

SMW92 – Inter-character timer

O registo SMW92 terá o valor do tempo máximo entre caracteres de uma mesma mensagem, caso este tempo seja ultrapassado a mensagem é considerada como tendo erro e não é processada, pela especificação do protocolo este tempo deverá ser 1.5 vezes o tempo de um caracter, a 9600 *bits/s* o tempo entre caracteres seria 1.7 ms, será utilizado o valor inteiro de 2 ms.

Configuração utilizada: 2 ms

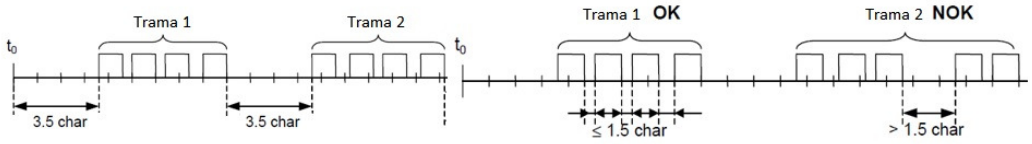

Figura 3.30 Tempos de deteção de mensagens segundo o modo RTU

Concluída a configuração são ativadas as interrupções, permitindo a execução da INT0. Durante o processamento das mensagens as interrupções poderão estar desabilitadas de modo a que não seja iniciado o processamento de uma nova mensagem enquanto ainda se está a processar uma antiga. Nesta situação a mensagem recebida não é processada e neste caso o *master* que enviou a nova mensagem se não obtiver uma resposta a tempo irá acionar um *timeout* e enviar novamente a mensagem.

Configuração 2 – Protocolo MODBUS RTU

Visto o protocolo ser programado no autómato e não através de um módulo de comunicações dedicado, os parâmetros necessários no decorrer do processamento do protocolo serão guardados em posições de memória regulares do autómato, memória V, e não em *bytes* de memória de sistema.

Serão utilizadas as seguintes posições de memória:

- VW90 – Número máximo de entradas/saídas acessíveis através de protocolo. A configuração utilizada neste registo foi 65(dec)
- VD92 – Endereço de memória para leitura de entradas analógicas. A configuração utilizada foi &VB2000 (endereçamento indireto)
- VW96 – Número máximo de posições de memória acessíveis. Configuração utilizada: 1001(dec)

- VW98 - Número máximo de entradas analógicas acessíveis. Configuração utilizada: 17(dec)

Configuração 3 – Buffers, Scratchpad e Flags

Nesta configuração é feito o *reset* a todos os valores que possam estar presentes nos *buffers* utilizados para o processamento de mensagens, assim como à zona de memória de utilização geral (*scratchpad*) e *bits* de memória utilizados como sinalizadores (*flags*).

É utilizada uma instrução FILL com o valor 0 (zero) a 100 *words* com início na VW100, ficarão a zero todos os registos de 8 *bits* de VB100 a VB300.

3.4.7.2 Interrupção 0

Após a execução da sub-rotina de configuração o programa volta ao ciclo principal executando o programa ciclicamente até ser recebida uma mensagem nova completa. Esta mensagem será recebida no *buffer* de receção de mensagens MODBUS RTU constituído pelas posições de memória de VB100 a VB119 e será ativada a interrupção 0 que dará início ao processamento de uma nova mensagem.

Na interrupção são utilizados os *bits* 2 e 3 do SMB86 (*Receive Message Control*) para verificar se a mensagem terminou por ter sido recebida uma mensagem com o tamanho ou tempo máximo SMW90 (*Idle line timer*), neste caso a informação contida no *buffer* de receção é transferida para o *buffer* de processamento (*handling buffer*) constituído pelas posições de memória de VB180 a VB199 e utilizada uma *flag* para indicação que existe uma nova mensagem no *handling buffer* a ser processada.

Caso as condições que levaram à ativação da interrupção não sejam uma receção por tamanho ou tempo máximo então a informação contida no MODBUS *Receive buffer* é apagada e o programa sai da interrupção sem ativação da *flag*.

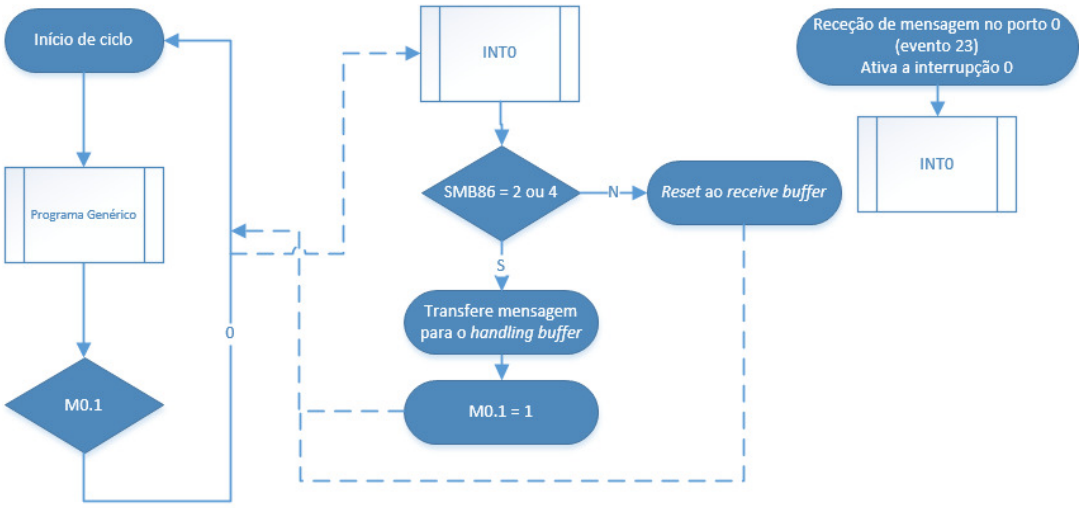

Figura 3.31 Interrupção 0 após receção de nova mensagem

3.4.7.3 SBR_1 - Processamento de uma mensagem MODBUS RTU

Como especificado na camada de aplicação deste protocolo as tramas são compostas de um modo geral por um endereço, um código de comando, o campo de dados e por fim um código de verificação de erros de acordo com estrutura apresentada na figura seguinte:

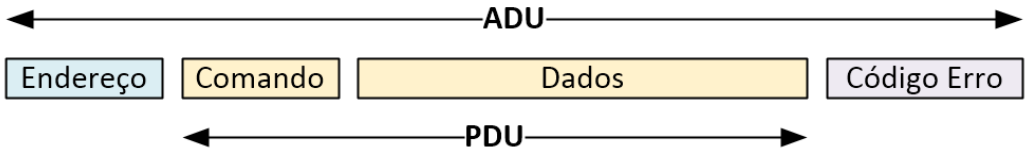

Figura 3.32 Trama do tipo MODBUS [25]

No processamento da mensagem será então necessário verificar a integridade dos dados através do código de erro, caso o código seja válido são feitas as verificações ao código de comando e endereço como detalhado em seguida, após estas verificações o programa será encaminhado para as sub-rotinas de processamento do comando que utilizarão os dados contidos na zona de dados da mensagem.

Esta sub-rotina começa por desabilitar as interrupções pois o processamento de uma mensagem pode ser longo e os dados poderiam ser sobrepostos por uma nova mensagem a meio do processamento. Seguidamente é executado o controlo da trama por código CRC, este controlo é feito em duas componentes, é executada a sub-rotina

8 que fará o cálculo do código CRC da mensagem presente no *handling buffer* e seguidamente é chamada a sub-rotina 11 que fará a comparação entre o valor do código presente na mensagem e o resultado da sub-rotina 8. Após validação do código é acionada a indicação de código CRC válido. Caso o código não seja válido é acionada a indicação de código inválido.

Novamente na sub-rotina 1 se o código for inválido é feito o *reset* aos *buffers* e *flags* e novamente ativada a receção de mensagens para estabelecer as condições de retorno ao programa principal sem processamento da mensagem e sem resposta ao *master* que enviou a mensagem.

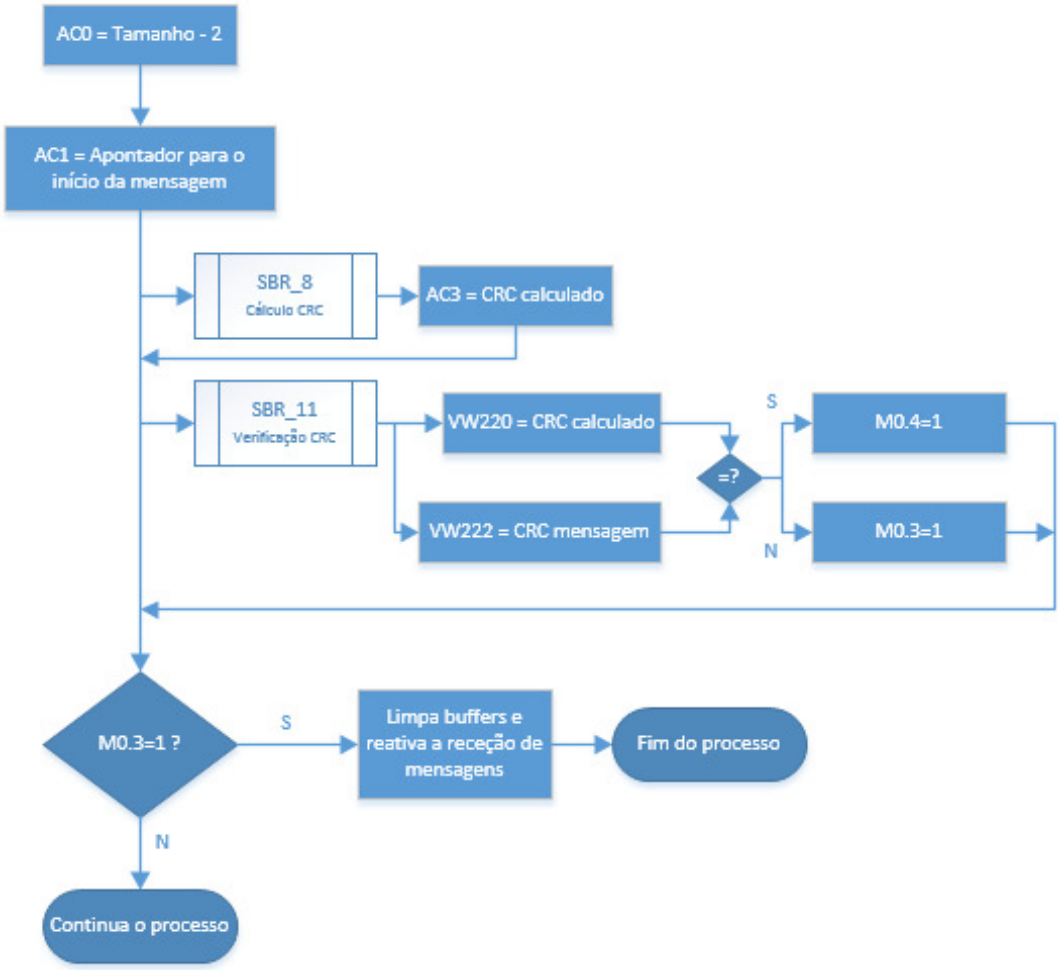

Figura 3.33 Processo de verificação do código CRC

Caso o código CRC seja válido é feita a verificação do comando recebido, caso o comando não seja válido é feito um *reset* como anteriormente e escrito o código de

exceção 01h de comando desconhecido na mensagem de resposta e chamada a sub-rotina 9 para geração e envio de uma resposta com indicação de erro.

No caso do código CRC e o comando serem válidos é lido o endereço do *slave* a que a mensagem se destina, caso o endereço seja 00h ou 20h é acionada a uma indicação de mensagem em modo *broadcast* a qual será executada mas não será enviada uma mensagem de resposta, caso o endereço seja o 25h o programa é encaminhado para as sub-rotinas 2 a 7 consoante o comando a executar, caso o endereço não seja o 25h mas seja entre 01h e 32h é ativada a indicação de uma nova mensagem destinada à rede PROFIBUS-DP e chamada a sub-rotina 13 para processamento da mesma.

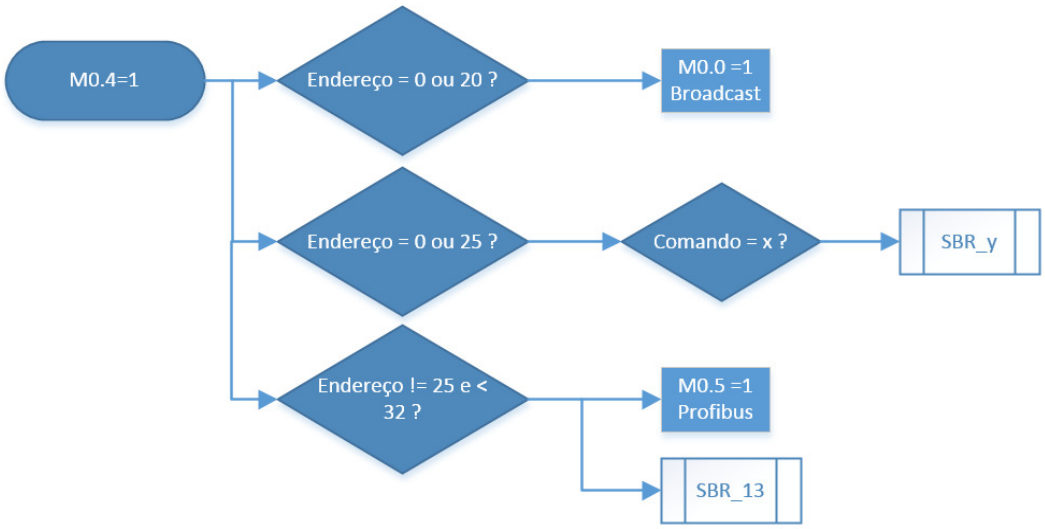

Figura 3.34 Processo de verificação do comando a executar

O processamento dos comandos disponíveis através de uma mensagem MODBUS será feito através das sub-rotinas 2 a 7. Segundo a especificação existem três categorias de comandos, públicos, definidos pelo utilizador e reservados, destes serão implementados oito comandos públicos de acordo com a tabela seguinte:

Tabela 3.11 Comandos implementados

Tipo	Acesso	Afetação	Função	Código (hex)	Sub-rotina	Reg.
Acesso a dados	1 *bit*	Entradas Binárias	Ler entradas binárias	02	SBR_2	I
		Saídas Binárias	Ler saídas binárias	01	SBR_2	Q
			Escrever uma saída binária	05	SBR_4	Q
			Escrever múltiplas saídas binárias	0F	SBR_6	Q
	16 *bits*	Entradas Analógicas	Ler entradas analógicas	04	SBR_3	AIW
		Saídas Analógicas ou Registos	Ler múltiplos registos	03	SBR_3	VW
			Escrever um registo	06	SBR_5	VW
			Escrever múltiplos registos	10	SBR_7	VW

3.4.7.4 SBR_3 – Ler múltiplos registos/entradas analógicas

A sub-rotina SBR_3 implementa os comandos MODBUS 03h de leitura de registos internos de memória e 04h de leitura de entradas analógicas, a estrutura das mensagens é a seguinte:

Tabela 3.12 Estrutura das mensagens para os comandos MODBUS 03h e 04h

Mensagem enviada pelo *master*							
Endereço	Comando	Endereço do 1º registo		Quantidade de registos		CRC	
	03h ou 04h	*H. byte*	*L. byte*	*H. byte*	*L. byte*	*Low byte*	*High byte*
1 *byte*	1 *byte*	2 *bytes*		2 *bytes*		2 *bytes*	

Resposta enviada pelo *slave*					
Endereço	Comando	Q. de *bytes*	Dados	CRC	
	03h ou 04h			*Low byte*	*High byte*
1 *byte*	1 *byte*	1 *byte*	n *bytes*	2 *bytes*	

Mensagem de erro				
Endereço	Comando	Código	CRC	
	83h ou 84h	01h a 04h	*Low byte*	*High byte*
1 *byte*	1 *byte*	1 *byte*	2 *bytes*	

Nota: é relevante verificar que os *bytes* de CRC são trocados, na mensagem são enviados pela ordem *Low byte* e *High byte* quando são efetuados os cálculos para determinação do código CRC o resultado vem na ordem inversa.

Por no autómato S7-200 não ser permitido o acesso direto aos valores das entradas analógicas estes terão de ser movidos para um registo interno de memória V o qual já pode ser acedido indiretamente, a diferença entre função 04h e a função 03h será então o endereço inicial, o endereço dos registos internos começa em VW0 e o das entradas analógicas em VW2000, a estrutura das mensagens é igual à estrutura utilizada para os comandos 01h e 02h com o endereçamento feito a *words* e não a *bits*, acesso a 16 *bits* tal como indicado na tabela 3.11.

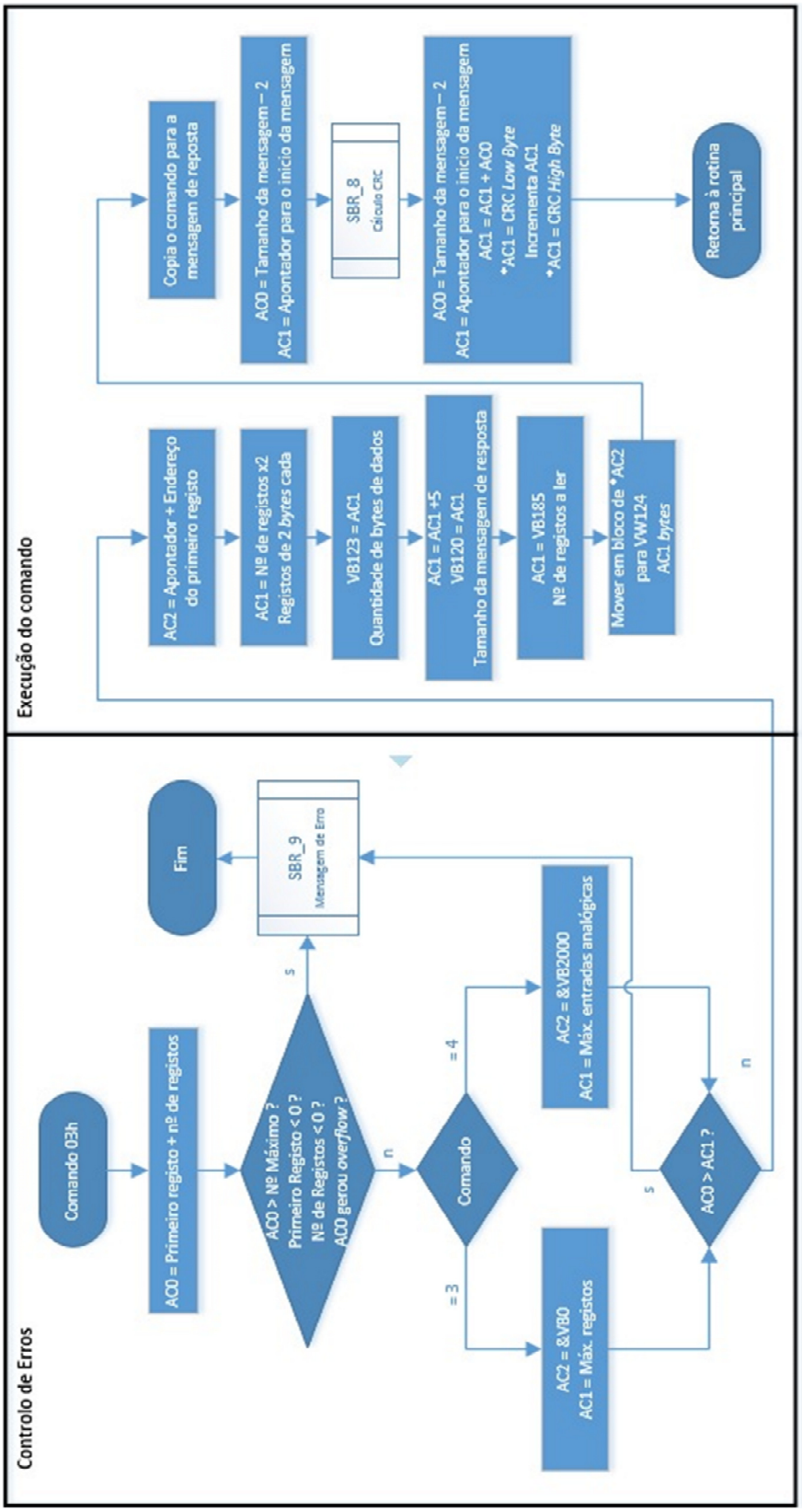

Figura 3.35 Controlo e execução dos comandos MODBUS 03h e 04h

O processamento destes comandos é conseguido em dois passos, primeiramente é feito o controlo de erros para verificação dos endereços recebidos e da quantidade de registos a aceder, caso estes sejam inferiores a zero, a sua soma seja superior ao máximo permitido pelo protocolo ou gere uma passagem por zero, ou seja de FFFFh para 0000h, no caso de alguma destas condições se verificar é escrito o código de erro 02h e acionado o envio de uma mensagem de erro como resposta.

Após o controlo de erros, o segundo passo é a execução do comando propriamente dito. Para tal é necessário determinar a quantidade de *bytes* de dados a escrever na mensagem de resposta, sendo necessário multiplicar o número de registos por dois pois são registos de 16 *bits* cada, seguidamente é calculado o tamanho total da mensagem de resposta que será igual ao número de *bytes* de dados mais cinco *bytes*, 1 *byte* de endereço de *slave*, 1 *byte* de comando, 1 *byte* de quantidade de *bytes* e 2 *bytes* de CRC. Após a obtenção destes dois valores e da sua escrita no *buffer* da mensagem de resposta é feita a transferência dos dados em si, desde o primeiro registo a ler (apontador mais endereço inicial), em linguagem *ladder* é utilizada a função BLKMOV_W para transferir a quantidade de registos de 16 *bits* apontados por AC2 para uma saída, neste caso a saída inicial será VW124, a posição de memória inicial para escrever os *bytes* de dados no *buffer* de resposta.

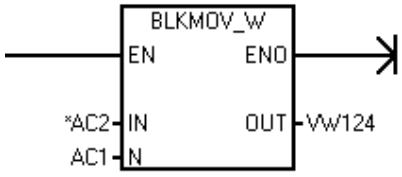

Figura 3.36 Função BLKMOV_W

Por fim é feita a chamada da sub-rotina SBR_8 onde é feito o cálculo dos *bytes* de CRC, após a colocação do código CRC na mensagem é o fim da sub-rotina SBR_3. O programa ao voltar à sub-rotina SBR_1 irá verificar se a mensagem que originou o comando foi enviada em modo *broadcast* ou não, neste caso sendo uma mensagem de leitura não poderia ser em *broadcast*, não tendo sido em *broadcast* é chamada a sub-rotina SBR_12 para transmissão da resposta, ou seja, a mensagem contida no MODBUS RTU *Send buffer* VB120 a VB139.

3.4.7.5 SBR_8 – Cálculo do código CRC

A sub-rotina SBR_8 implementa o cálculo do código CRC, este cálculo é efetuado em duas ocasiões após a receção de uma mensagem para verificação da validade da mesma e após a execução de um comando para criação do código CRC a inserir na mensagem de resposta, por esta razão o cálculo do código e a validação do mesmo têm de ser efetuados em sub-rotinas diferentes.

O cálculo do código CRC é feito com base no tamanho da mensagem recebida/a enviar menos dois *bytes* (os *bytes* do próprio código) e o conteúdo da mensagem.

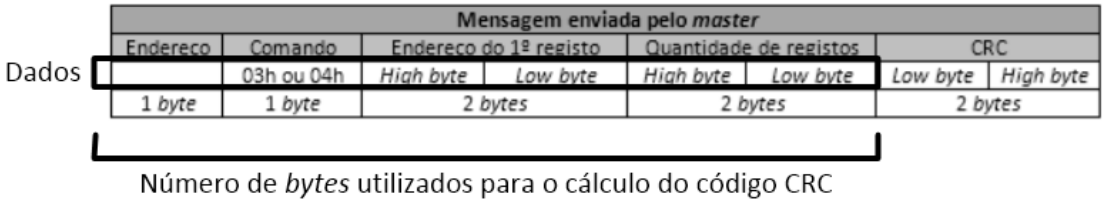

Figura 3.37 Dados utilizados para cálculo do código CRC para os comandos MODBUS 03h ou 04h

Para o cálculo do código CRC é utilizado um registo a 16 *bits*, chamado de registo CRC, preenchido com o valor FFFFh, tudo a 1's, sendo feito um ou exclusivo (XOR) entre este registo e o primeiro *byte* da mensagem, o resultado é rodado para a direita e examinado o *bit* que sai, o *bit* menos significativo, se o *bit* é zero o processo repete-se para outro *byte* da mensagem se o *bit* for um é aplicado um ou exclusivo entre o registo CRC e o valor A001h (polinómio gerador) e o processo repetido.

Processo de cálculo [11]:

- Registo CRC com o valor FFFFh;
- XOR entre o *byte* menos significativo do registo CRC e o primeiro *byte* da mensagem;
- Rodar para a direita o registo CRC e verificar o estado do LSB (*Least Significant Bit*);
- LSB = 0 repete a rotação anterior;
- LSB = 1 XOR entre o registo CRC e o valor A001h;
- Repetir os passos anteriores até serem processados 8 *bits*;

- Repetir os passos anteriores até serem utilizados todos os *bytes* da mensagem;
- O conteúdo do registo CRC é o valor do código com a ordem *high byte low byte*.

Na mensagem o código é enviado com a ordem inversa, *low byte* primeiro.

A implementação deste processo em *ladder* segue o descrito no diagrama seguinte.

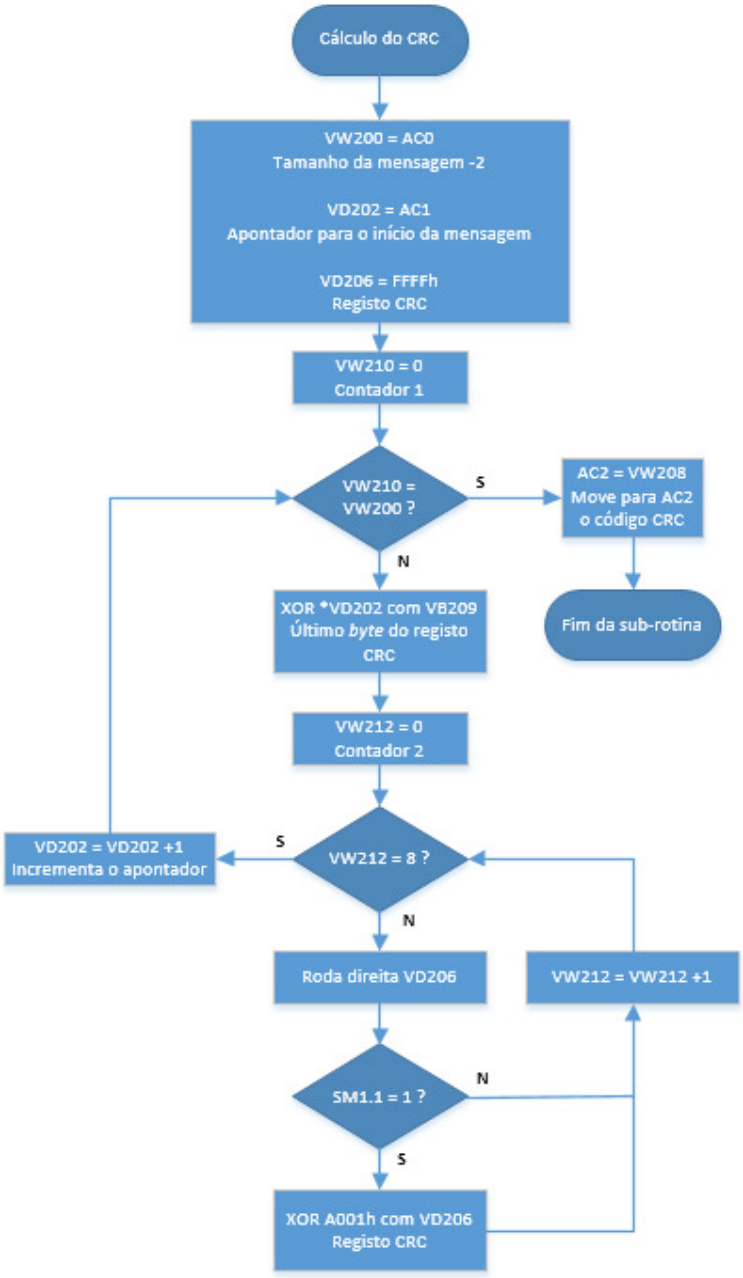

Figura 3.38 Diagrama de estados do cálculo do código CRC

3.4.7.6 SBR_11 – Validação do código CRC

A sub-rotina SBR_11 é utilizada para validação do código CRC aquando da receção de uma nova mensagem proveniente da rede MODBUS RTU. Após a receção da mensagem é executada a sub-rotina SBR_8 para criação de um código CRC e posteriormente chamada a presente sub-rotina para validação do código recebido na mensagem por comparação com o calculado. Após a validação é ativado um *bit* de memória que indicará se o código é válido ou não.

Após a validação a rotina acaba e o programa volta à rotina que a chamou, neste caso será a sub-rotina SBR_1 que continuará o processamento da mensagem ou não consoante a validade do código.

Tal como visto anteriormente o código é calculado com o *high byte* primeiro e o *low byte* depois, na mensagem recebida é o contrário daí ser necessário utilizar VW208 para inverter os *bytes* do código calculado.

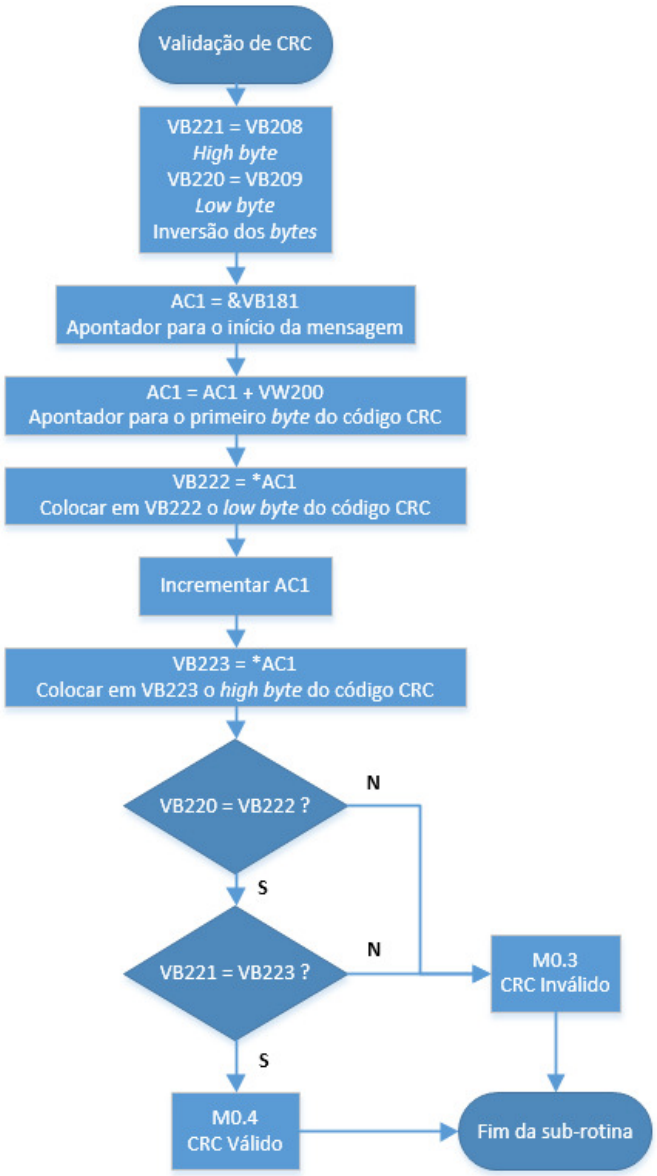

Figura 3.39 Fluxograma da validação do código CRC recebido

3.4.7.7 SBR_9 – Geração de mensagem de erro

A sub-rotina SBR_9 é utilizada para a geração da mensagem de erro, após a deteção de um erro durante o processamento de uma mensagem por uma das outras sub-rotinas é inserido no *buffer* de envio VB120 MODBUS *Send buffer* o código do erro, o código do comando e chamada a sub-rotina SBR_9.

Nesta rotina são então transferidos os restantes dados do VB180 *Handling buffer* para o MODBUS *Send buffer*, posteriormente é executada uma operação de OR entre o código de comando em VB122 e o valor 80h, após esta operação é chamada a sub-rotina SBR_8 para cálculo do código CRC e inserido o mesmo na mensagem de erro.

Após a construção da mensagem de erro a sub-rotina acaba e o programa retorna à sub-rotina que gerou o erro. A geração do erro acaba incondicionalmente retornando à sub-rotina de origem, normalmente a sub-rotina SBR_1 onde é depois ativada a transmissão da mensagem de resposta que neste caso será uma mensagem de erro.

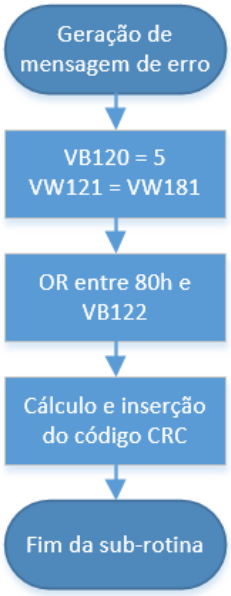

Figura 3.40 Diagrama de estados da geração de mensagem de erro

3.4.7.8 SBR_12 – Transmissão de uma mensagem

A sub-rotina SBR_12 implementa a transmissão de uma mensagem, esta sub-rotina é chamada após a escrita dos dados de uma resposta no MODBUS *Send buffer*, VB120 a VB139.

A transmissão da mensagem é feita através da configuração e execução da instrução XMT a qual será configurada para enviar X bytes com início em VB121, consoante o valor indicado no *byte* VB120, através do porto 0. No caso do valor presente em VB120 ser zero a execução da instrução XMT não envia os dados presentes mas inicia um tempo de espera com a duração de 16 *bits* ao ritmo de transmissão selecionado causando neste caso uma condição de silêncio (*break*) na linha de 1.7 ms. Esta condição irá causar a terminação de uma mensagem da qual o *master* esteja à espera de resposta. Esta pode ser também utilizada caso exista algum erro no processamento da mensagem e seja conveniente terminar a comunicação para que o *master* fique livre para outras transmissões.

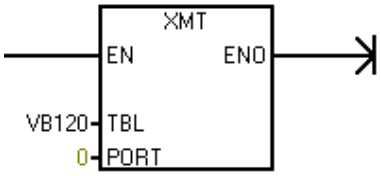

Figura 3.41 Instrução XMT

A transmissão (XMT) de uma mensagem não pode estar ativa em simultâneo com a instrução de receção (RCV), a utilização da instrução XMT com a instrução RCV ativa causa erro por sobreposição. A coordenação entre estas instruções toma elevada importância para que seja garantida a receção de todas as mensagens sem nunca existir sobreposição com a transmissão de uma resposta, em especial porque as instruções de XMT e RCV são utilizadas em sub-rotinas ou interrupções. As interrupções pela sua natureza de execução aleatória devem ser limitadas e controladas por forma a que não exista a possibilidade de sobreposição das instruções. Neste trabalho apenas a instrução RCV utiliza uma interrupção. A instrução XMT utiliza uma sub-rotina e a análise a um *bit* especial SM4.5 para deteção de fim de transmissão, este *bit* indica quando a transmissão através do porto 0 está livre.

A receção de uma mensagem é habilitada no início do programa permanecendo sempre habilitada até ser recebida uma mensagem completa, após a receção de uma mensagem completa é ativa a interrupção 0 processada a transferência dos dados e desativadas a instrução RCV e as interrupções até ao processamento completo da mensagem. No fim do processamento da mensagem é habilitada e ativada a instrução XMT para executar a transmissão da resposta. Após a indicação do *bit* SM4.5 significando que a transmissão está concluída é ativada novamente a instrução RCV.

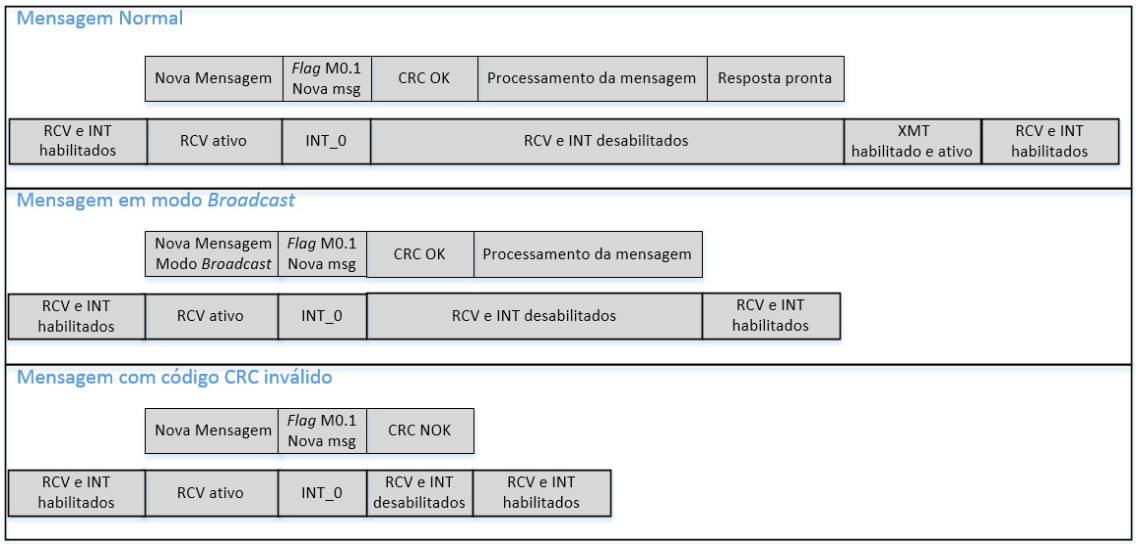

Figura 3.42 Diagrama temporal da utilização exclusiva das instruções RCV e XMT

A sub-rotina SBR_12 é chamada pela sub-rotina SBR_1 após o preenchimento dos dados no MODBUS *Send buffer*, o seu processamento é feito de acordo com o seguinte diagrama:

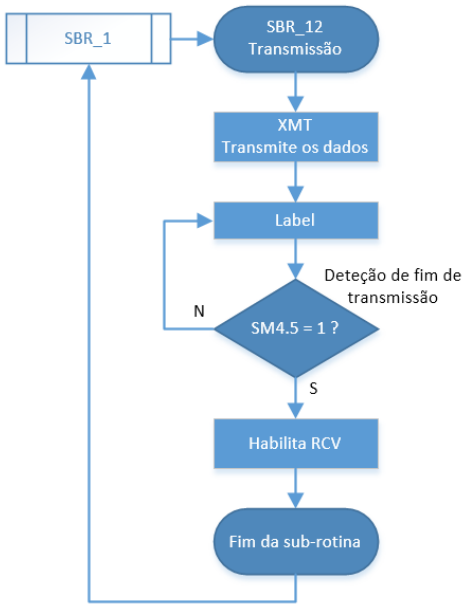

Figura 3.43 Fluxograma da transmissão de uma mensagem

É relevante relembrar que o ciclo de deteção de fim de mensagem não pode bloquear o ciclo do programa durante um tempo indeterminado, o tempo máximo de cada ciclo são 500 ms limitado pelo temporizador *watchdog*.

3.4.8 Ligação entre S7-200 e S7-300

A ligação entre o autómato S7-200 e o autómato S7-300 é efetuada através de uma rede de campo com protocolo PROFIBUS-DP e método de acesso *Master-Slave* com topologia em barramento. O autómato S7-300 (o *master* da rede) utiliza o porto de comunicação X2 com interface de comunicação série RS485, utilizando como meio físico o cabo PROFIBUS FC Standard Cable. No caso do autómato S7-200 (um dos *slaves*) possui uma porta de comunicação (porto 1) com interface série RS485 e protocolo PROFIBUS-DP.

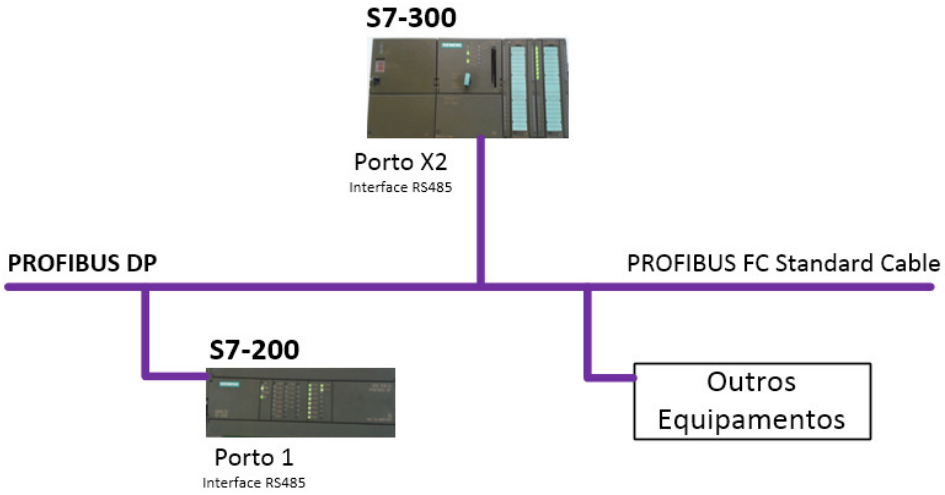

Figura 3.44 Interligação entre os autómatos S7-200 e S7-300

3.4.8.1 Considerações sobre o protocolo MODBUS RTU no autómato S7-300

A implementação do protocolo MODBUS RTU no autómato S7-300 será reduzida ao essencial para execução das funções necessárias ao funcionamento da rede, a implementação da totalidade do protocolo seria uma duplicação do já executado no S7-200 apenas com *software* diferente.

Será apresentada a função de transferência de dados programada no autómato S7-200 e as restantes funções programadas no autómato S7-300.

3.4.8.2 SBR_13 – Transferência de uma mensagem MODBUS RTU para PROFIBUS-DP

A sub-rotina SBR_13 (S7-200) executa a transferência de uma mensagem recebida via MODBUS RTU com destino a um equipamento na rede PROFIBUS-DP, é chamada a partir da sub-rotina SBR_1 quando é detetado que o endereço do *slave* é válido e diferente do endereço do autómato S7-200.

Esta sub-rotina utiliza cinco *buffers* de memória, o MODBUS *Receive buffer*, o *Handling buffer*, o PROFIBUS *Send buffer*, PROFIBUS *Receive buffer* e o MODBUS *Send buffer*, o diagrama de transferência de informação entre os mesmos é indicado na figura seguinte:

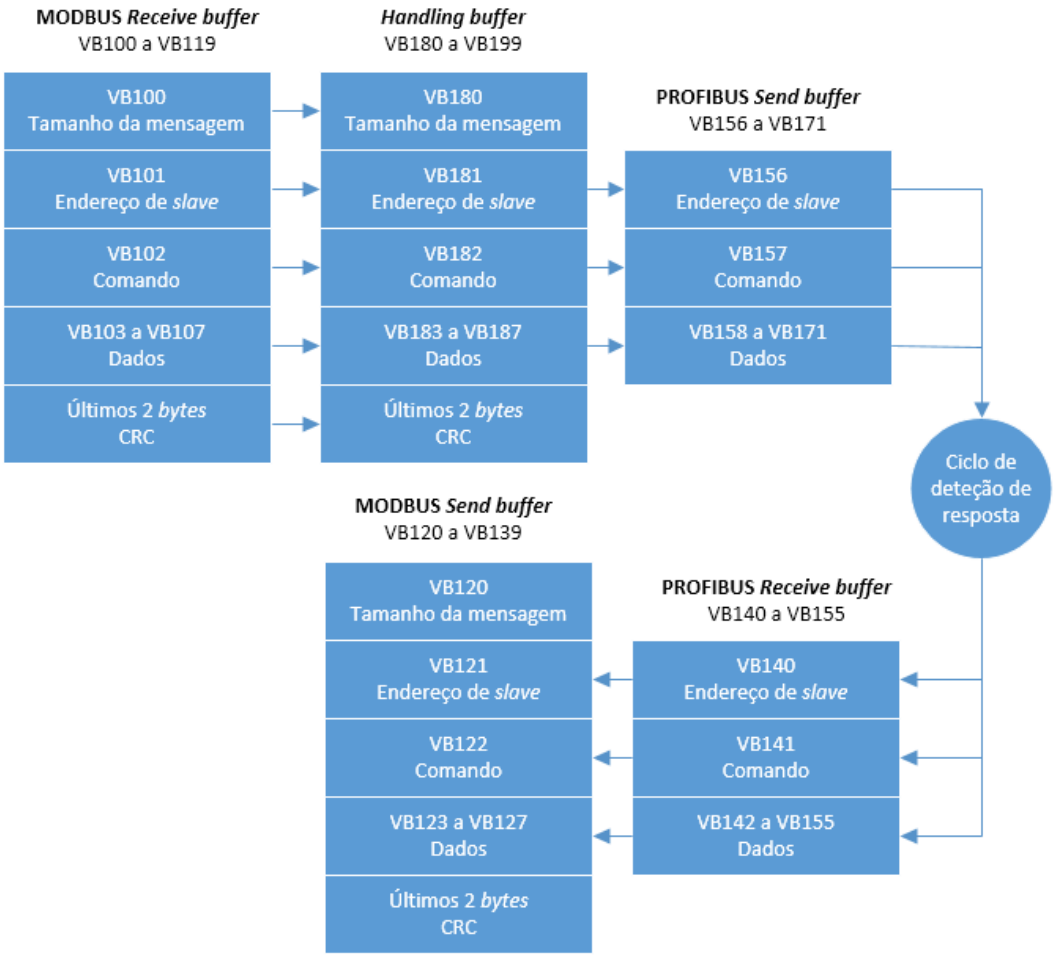

Figura 3.45 Diagrama de transferência de informação entre buffers

Tal como referido anteriormente o autómato S7-200 é um equipamento *slave* na rede PROFIBUS-DP, como tal o mesmo não fará automaticamente a transmissão do conteúdo do PROFIBUS *Send buffer*. A transferência de informação para o autómato S7-300 (o master da rede PROFIBUS-DP) será efetuada por *polling* do *master* ao PROFIBUS *Send buffer* contido no *slave*.

O autómato S7-200 ao receber uma mensagem destinada à rede PROFIBUS-DP transfere os seus conteúdos para o *buffer* de envio, o autómato S7-300 por sua vez está ciclicamente a efetuar uma leitura deste *buffer*. Ao receber dados diferentes dos anteriores (o que indicará a presença de uma nova mensagem no PROFIBUS *Send buffer*) o autómato S7-300 fará o processamento da mesma e executará uma função de escrita com a resposta para o PROFIBUS *Receive buffer* contido no S7-200.

O autómato S7-200 ao transferir uma nova mensagem para o PROFIBUS *Send buffer* tem de aguardar que o autómato S7-300 execute o comando e escreva a resposta no PROFIBUS *Receive buffer*, este tempo de espera é de elevada importância pois é o tempo responsável pela coordenação entre as redes. Se o tempo for muito longo o autómato S7-200 levará muito tempo a responder ao autómato TSX57-103 o qual acabará por ativar um *timeout* da mensagem que enviou.

O autómato S7-200 durante o ciclo de deteção de resposta estará continuamente a analisar o conteúdo do PROFIBUS *Receive buffer*. Caso existam caracteres diferentes de zero é feita a transferência do conteúdo do *buffer* para o MODBUS *Send buffer* e enviada a resposta ao TSX57-103, sendo que existirá um temporizador de 50 ms para que caso não seja recebida uma resposta por parte do S7-300 o autómato S7-200 possa terminar o ciclo de espera. Este procedimento encontra-se detalhado no diagrama seguinte.

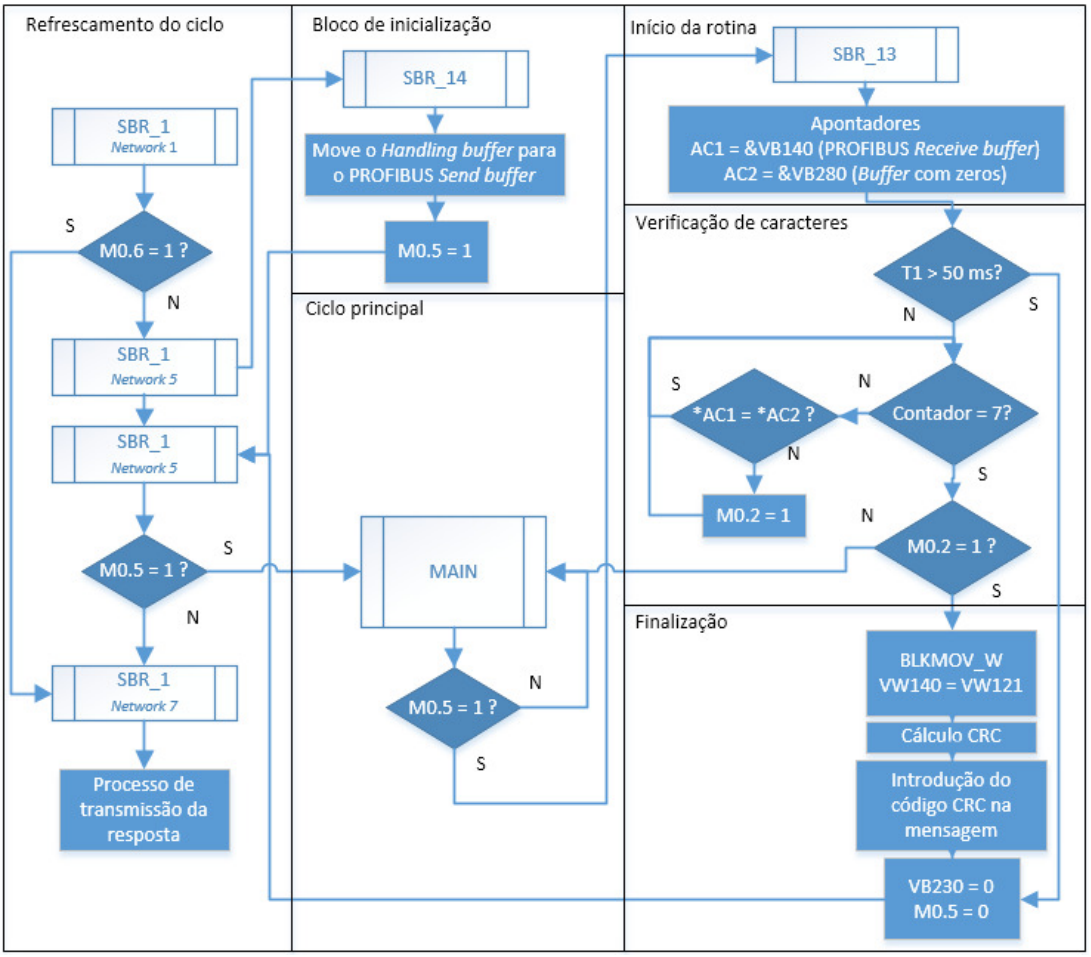

Figura 3.46 Diagrama de execução da sub-rotina SBR_13

O autómato S7-300 tem assim um tempo máximo de 50 ms para receber, executar e transmitir a resposta a uma mensagem. O autómato TSX57-103 está programado com um tempo de *timeout* de 100 ms e três *timeouts* por mensagem, num total de 300 ms até gerar um erro por falta de resposta. Veja-se a figura seguinte.

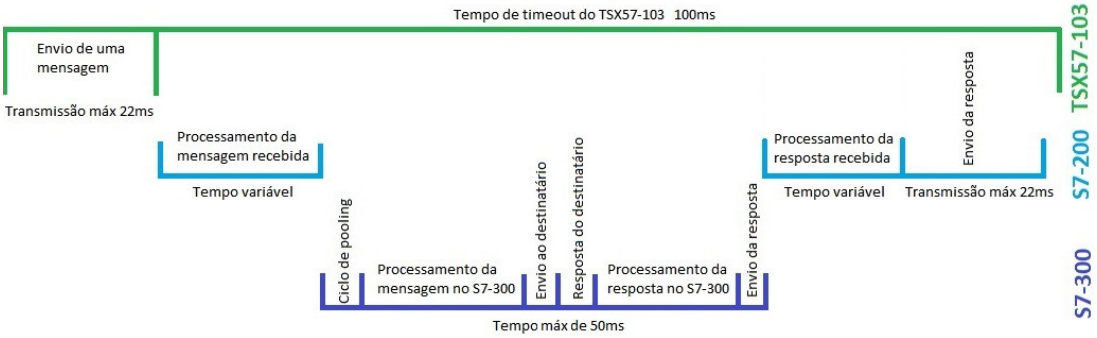

Figura 3.47 Articulação entre tempos de execução e transmissão

Dado o ritmo de transmissão na rede MODBUS RTU de 9600 *bit/s* o tempo de transmissão de uma mensagem de 19 *bytes* é de 22 ms, valor obtido segundo a seguinte expressão:

$$T = \frac{Nº\ bytes \times \frac{Nº\ bits}{byte}}{Ritmo\ de\ transmissão} \times 1000 = \frac{19 \times \frac{11}{1}}{9600} \times 1000 = 22\ ms$$

Considerando este valor como o tempo máximo de uma transmissão na rede MODBUS RTU acontece que no pior caso com uma mensagem de leitura com tamanho de 19 *bytes* (ambas, a mensagem e a resposta), o tempo máximo para o autómato S7-300 processar a mensagem e o tempo de transmissão da resposta o tempo total fica em 94 ms, restando apenas 6 ms para o autómato S7-200 processar a mensagem.

Verificou-se experimentalmente que a deteção de caracteres resulta em tempos de resposta totais inferiores a 50 ms, para proteção contra um ciclo infinito por falta de resposta foi então adicionada ao ciclo de deteção de resposta uma temporização de 50 ms como limite máximo de tempo de espera.

3.4.8.3 Processamento dos comandos MODBUS S7-300

O processamento dos comandos MODBUS recebidos será efetuado através de uma rotina principal designada OB1 que fará a deteção de novas mensagens e desencadeará o processamento das mensagens. Este será executado nas sub-rotinas FC1 que fará a identificação do endereço do *slave* e o código de comando, a execução dos comandos será implementada nas sub-rotinas FC6 a FC11. No caso do autómato S7-300 as sub-rotinas são designadas por funções (*Functions*).

No caso do autómato S7-300 as mensagens MODBUS virão através do campo de dados das mensagens PROFIBUS-DP e como tal não será necessário proceder à validação da integridade das mensagens via código CRC pois o protocolo e os módulos de comunicações PROFIBUS-DP fazem automaticamente a validação da mensagem.

A implementação dos comandos MODBUS no autómato S7-300 é idêntica à já efetuada no autómato S7-200 mas necessária para o funcionamento da rede, assim serão apenas implementados alguns dos comandos mais necessários como a escrita e leitura de saídas digitais e registos internos. Não será também implementado o controlo de erros e mensagens de erro, tal implementação seria apenas uma repetição do já feito no

autómato S7-200 não contribuindo para o propósito de aprendizagem e exploração de conteúdos da presente dissertação.

Os endereços de *slave* permitidos e comandos implementados no S7-300 são descritos na tabela seguinte:

Endereço	Equipamento	Comando	Designação	Função
21h	S7-300 e Rede ASi	0Fh	Escrever múltiplas saídas binárias	FC6
		10h	Escrever múltiplos registos	FC7
		01h	Ler múltiplas saídas binárias	FC10
		03h	Ler múltiplos registos	FC11
23h	Micromaster	10h	Escrever múltiplos registos	FC8
		03h	Ler múltiplos registos	FC9

A estrutura do programa no autómato S7-300 é a seguinte:

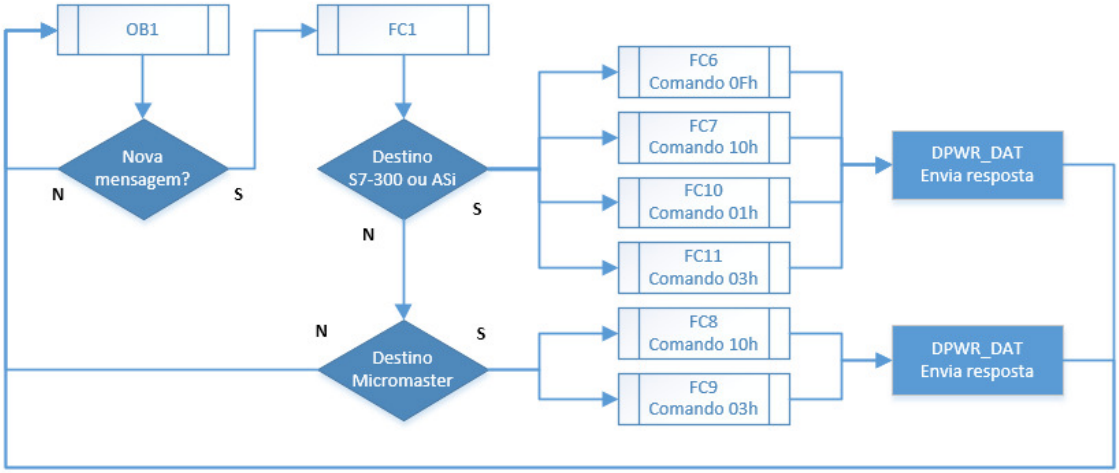

Figura 3.48 Diagrama de execução do programa no autómato S7-300

3.4.8.4 OB1

A rotina principal OB1 implementa um ciclo contínuo de deteção de nova mensagem. Tal é conseguido através da comparação entre os dados contidos no *handling buffer* e no *receive buffer*.

Os dados contidos no *receive buffer* são copiados para o *handling buffer* e executados ciclicamente com uma instrução de leitura (DPRD_DAT) que irá enviar

uma mensagem de leitura ao autómato S7-200 para obtenção do PROFIBUS *Send buffer*. Estes dados são guardados no *receive buffer* do autómato S7-300 e caso os dados presentes nos *buffers* sejam diferentes isto indica de que foi recebida uma nova mensagem proveniente da rede MODBUS RTU sendo então chamada a função FC1 para processamento da mesma. O diagrama seguinte ilustra este procedimento.

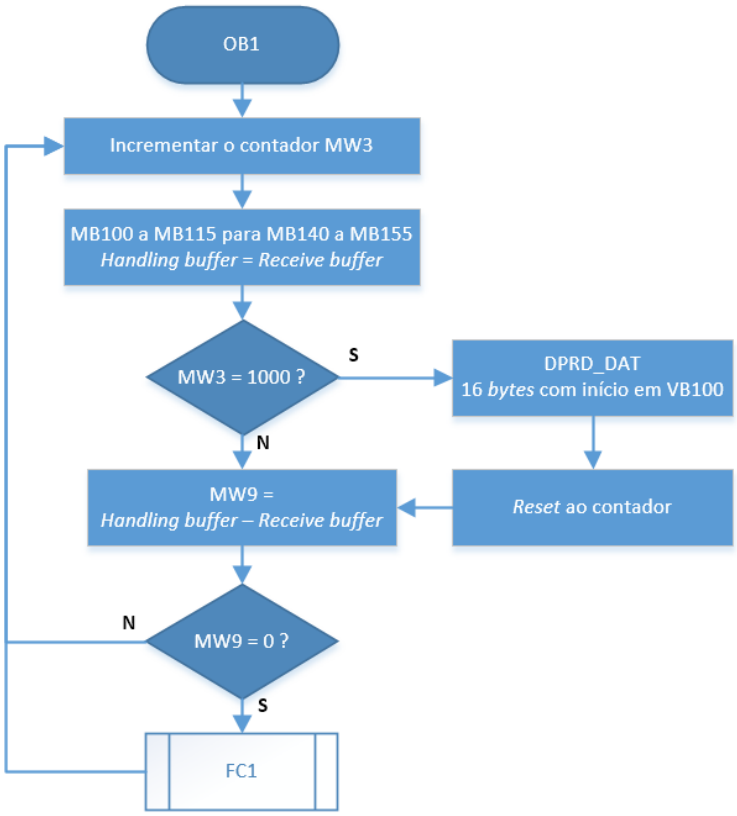

Figura 3.49 Diagrama de execução da rotina principal OB1

3.4.8.5 FC1

A função FC1 fará a deteção do endereço do *slave* e do código de comando. Caso sejam válidos o programa é encaminhado para a função que executa o comando e retorna novamente à função FC1 e são enviados os dados presentes no s*end buffer* contidos nos registosVB120 a VB135, retornando à rotina principal OB1.

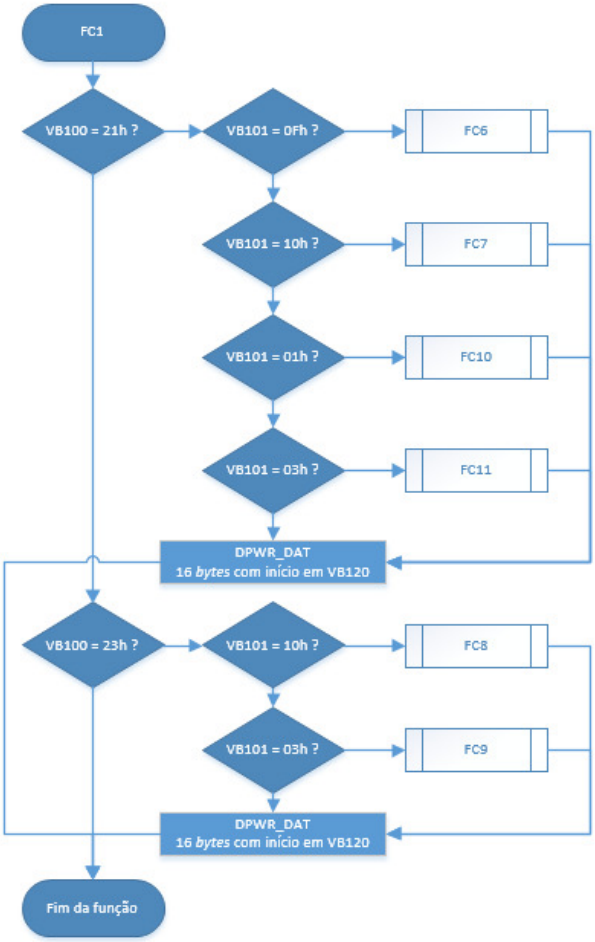

Figura 3.50 Diagrama de execução da função FC1

3.4.8.6 FC6

A função FC6 implementa o comando MODBUS 0Fh para escrever múltiplas saídas binárias. Este comando destina-se ao autómato S7-300 e aos periféricos na rede ASi visto que ambos são endereçados como saídas do próprio autómato. Este comando foi implementado no autómato S7-200 em linguagem *Ladder* e no autómato S7-300 em linguagem STL. A utilização da linguagem STL embora mais complexa apresenta maior flexibilidade, permitindo o endereçamento indireto ao nível do *bit* e não apenas ao nível do *byte* como acontece em *Ladder*. O que permite implementar o comando tal como descrito no MODBUS *Application Protocol* [25].

O autómato S7-300 tem dois registos especiais utilizados para guardar endereços, os registos AR1 e AR2 *(Address Register)*. Estes registos podem guardar uma variável

por exemplo 107.0, ou podem guardar um endereço específico como por exemplo Q4.0. Para a execução deste comando MODBUS são utilizadas ambas as situações.

A implementação do comando começa pelo carregamento no AR1 do endereço 107.0 que é o endereço no *receive buffer* de onde vão ser lidos os dados e no AR2 o endereço do primeiro *bit* da primeira saída do autómato, neste caso a saída Q4.0.

Seguidamente é carregado no acumulador o valor do endereço da primeira saída contido em MB103 e a este adicionado o conteúdo de AR2, ou seja o endereço de Q4.0, após a soma o registo AR1 terá o endereço Q4.0+X, por exemplo Q4.1.

Após a definição do primeiro endereço é carregado no acumulador MB105 a quantidade de variáveis a escrever, esta quantidade será transferida para MW58 que servirá de contador ao ciclo FOR utilizado para escrever as saídas.

A cada ciclo FOR é transferido o valor do *bit* apontado pelo registo M107.0+X para o registo Q4.0+X os endereços contidos em AR1 e AR2 são incrementados e o valor do contador MW58 transferido para o acumulador, ao próximo ciclo FOR o valor do contador é decrementado pela instrução FOR.

```
      LAR1  P#107.0
      LAR2  P#Q 4.0
      L     MB   103
      +AR2
      L     MB   105
NEXT: T     MW    58
      A     M [AR1,P#0.0]
      =      [AR2,P#0.0]
      L     1
      +AR1
      L     1
      +AR2
      L     MW    58
      LOOP  NEXT
```

Figura 3.51 Instruções para escrita de múltiplas saídas binárias

Após a escrita de todas as saídas o programa efetua a transferência dos valores contidos no *receive buffer* para o *send buffer* e termina a função FC6.

3.4.8.7 Mensagens MODBUS RTU para comando do variador de velocidade

Os comandos MODBUS RTU implementados para acesso ao variador são o comando MODBUS 10h para escrita de múltiplos registos e o comando MODBUS 03h para leitura de múltiplos registos. O comando 10h será utilizado para escrita de PZD1 e PZD2 no endereço PQD264 e o comando 03h será utilizado para leitura de PZD2 no endereço PIW266.

Tabela 3.13 Estrutura de uma mensagem e resposta para o comando MODBUS 10h

Mensagem enviada pelo *master*						
Endereço	Comando	Endereço do 1º registo		Quantidade de registos		Quantidade de *bytes*	Dados
	10h	*High byte*	*Low byte*	*High byte*	*Low byte*		
1 *byte*	1 *byte*	2 *bytes*		2 *bytes*		1 *byte*	N *bytes*
MB100	MB101	MW102		MW104		MB106	MB107 a MB115

Resposta enviada pelo *slave*					
Endereço	Comando	Endereço do 1º registo		Quantidade de registos	
	10h	*High byte*	*Low byte*	*High byte*	*Low byte*
1 *byte*	1 *byte*	2 *bytes*		2 *bytes*	
MB120	MB121	MW122		MW124	

A execução do comando 10h encontra-se implementada na função FC8 e a do comando 03h na FC9.

3.4.8.7.1 FC8:

A quantidade de dados a receber é fixa com um tamanho de 4 *bytes* para escrita dos valores de PZD1 (2 *bytes*) e PZD2 (2 *bytes*). A execução da escrita será feita através

da instrução MOVE. Os valores dos dados começam em VB107 e a terminam em VB110 sendo movidos para o endereço de PZDQ264.

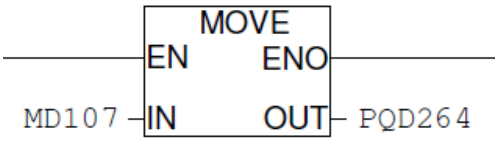

Figura 3.52 Instrução MOVE

A escrita da resposta no *Send buffer* é executada pela transferência dos dados contidos no *Receive buffer*.

```
L     MW   100
T     MW   120
L     MW   102
T     MW   122
L     MW   104
T     MW   124
```

Figura 3.53 Transferência de dados para o Send buffer

Capítulo 4

Sistema de Supervisão (SCADA)

Resumo: No presente capítulo apresentam-se alguns conceitos base sobre sistemas SCADA e de seus componentes, sendo apresentados posteriormente dois projetos SCADA implementados, o projeto para comunicação entre a supervisão e os controladores e um projeto exemplo de uma instalação industrial.

4.1 A supervisão nos processos automatizados

A importância dos sistemas de supervisão é largamente conhecida sendo o seu objetivo primordial facilitar o desempenho de qualquer processo automatizado. Convencionalmente os sistemas de supervisão são utilizados para a monitorização de variáveis do processo, gestão de alarmes e arquivo de dados. A interface do sistema automatizado com os operadores é uma das principais funções da supervisão. Os recursos desta natureza receberam a designação anglo-saxónica consagrada na palavra SCADA (*Supervisory Control And Data Aquisition*) [26] [27].

Um sistema SCADA é um sistema de controlo distribuído constituído por um pacote de *software* instalado em uma estação principal MTU (*Master Terminal Unit*) destinado ao controlo local de processos distribuídos através da aquisição de dados provenientes dos controladores locais RTU (*Remote Terminal Unit*), executando o processamento dos mesmos, compilando históricos, gerando alarmes, criando bases de dados e estatísticas, entre outros, permitindo assim efetuar um controlo de alto nível de uma rede de automação por comandos automáticos (*scripts*) ou manuais através de um painel sinótico de comandos e informações o designado HMI [28].

Figura 4.1 Exemplo de sala de controlo com painéis HMI e estações de controlo MTU (Siemens)

Um sistema de supervisão divide-se essencialmente em três aspetos principais:

- Aquisição de dados: Recolha do estado e valores dos equipamentos de campo e a compilação da informação em um equipamento central.
- Processamento de dados: Criação de históricos, alarmes, bases de dados e processamento automático por estruturas pré programadas (*scripts*) e disponibilização de dados para níveis superiores, por exemplo uma rede empresarial ou estratégica.
- Interface humano-máquina: Disponibilização de informação relevante do sistema para um operador através de painéis sinóticos permitindo a execução de comandos manuais e definição de pontos de funcionamento (*setpoints*) para operação dos processos na rede.

Em termos de arquitetura um sistema SCADA é um sistema no qual é utilizado um equipamento central que permite a supervisão e gestão global do sistema e um ou mais controladores locais (RTU's), o controlo do processo é feito localmente e a supervisão centralmente [29].

O controlo do processo sendo local permite um controlo mais simples, rápido e fácil de diagnosticar. A supervisão central permite um controlo mais abrangente e maiores capacidades de processamento com uma visão global de toda a rede, menos cablagem (e consequentemente menores custos com a mesma) e maior resiliência a falhas locais, uma falha em uma RTU pode apenas ter consequências locais e caso necessário podem existir RTU's e MTU's redundantes.

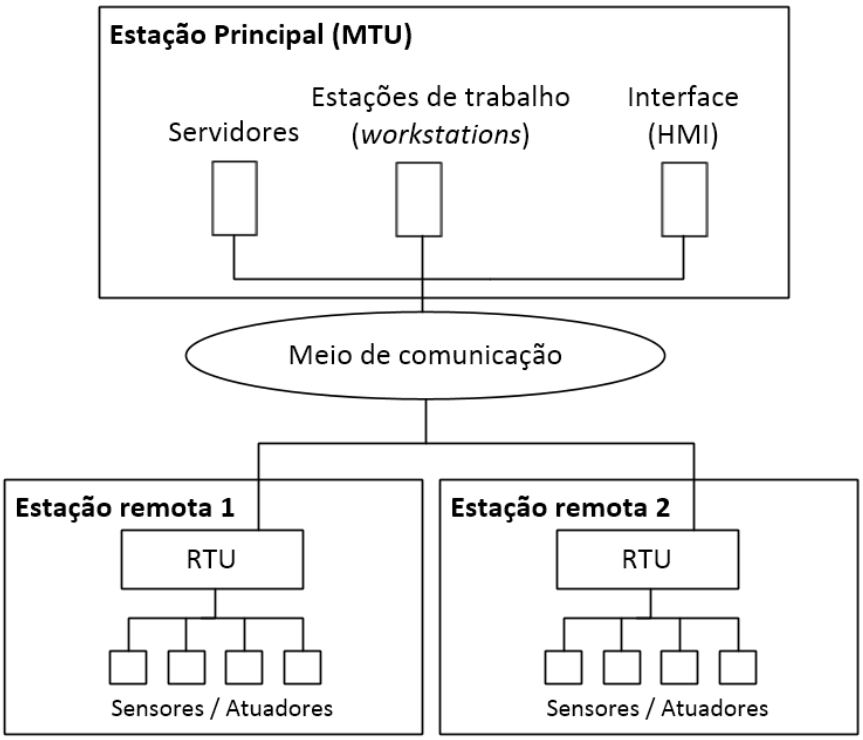

Figura 4.2 Arquitetura de um sistema SCADA [30]

4.1.1 MTU (Master Terminal Unit)

A estação principal ou MTU é o equipamento ou conjunto de equipamentos principais de um sistema SCADA. É o equipamento que inicia a comunicação com as unidades remotas, guarda a informação, processa-a e envia instruções para as unidades remotas através de uma rede de comunicações e para o operador através de um HMI, o envio de comandos para as RTU e a atualização dos valores na base de dados são dois exemplos do tipo de mensagens trocadas entre a MTU e as RTU.

A MTU pode ser composta por um equipamento processador central, servidores de dados, impressoras, ecrãs de visualização e outros sistemas de informação [31].

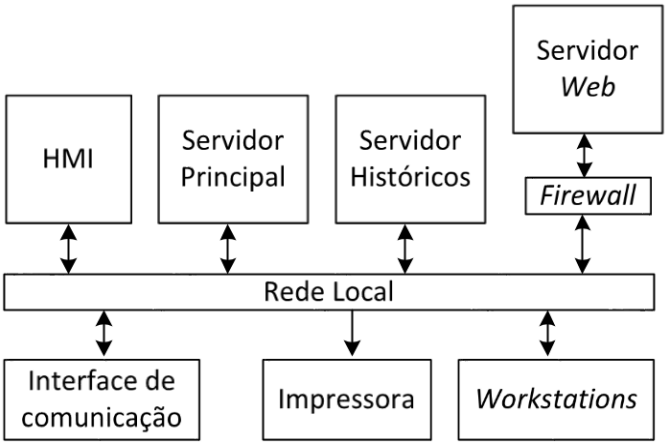

Figura 4.3 Diagrama de uma MTU

Dada a vulnerabilidade física e virtual e a importância da MTU para uma rede de automação pode existir uma estação principal redundante com o propósito de providenciar acesso à informação e ao comando da instalação em caso de uma falha ou ataque à MTU. A estação principal redundante pode consistir desde apenas um servidor SCADA adicional até uma completa MTU independente com estrutura física própria e canais de comunicação próprios [32].

4.1.2 RTU (Remote Terminal Unit)

As unidades remotas ou RTU's têm como função recolher a informação dos equipamentos de campo (sensores e atuadores) e através de uma rede de comunicação enviar e receber informação para e de a MTU, algumas RTU's como os PLC's podem ainda ter a capacidade para controlar processos [33].

A função principal das RTU's é a capacidade de enviar a informação dos equipamentos de campo como os sensores e atuadores para uma estação principal como a MTU não sendo necessário que a própria RTU tenha capacidade de controlo.

Uma RTU pode ainda ser utilizada como estação transmissora para outros controladores que não disponham de capacidade de comunicação, no presente trabalho é utilizado um PLC, o TSX57-103, como RTU permitindo o acesso da supervisão a outros controladores, PLC's S7, que não dispõem de acesso direto a uma rede *Ethernet*.

A evolução tecnológica dos PLC's que atualmente integram processamento, controlo e comunicação a custos acessíveis torna redundantes outros tipos de RTU que não disponham de capacidades de controlo.

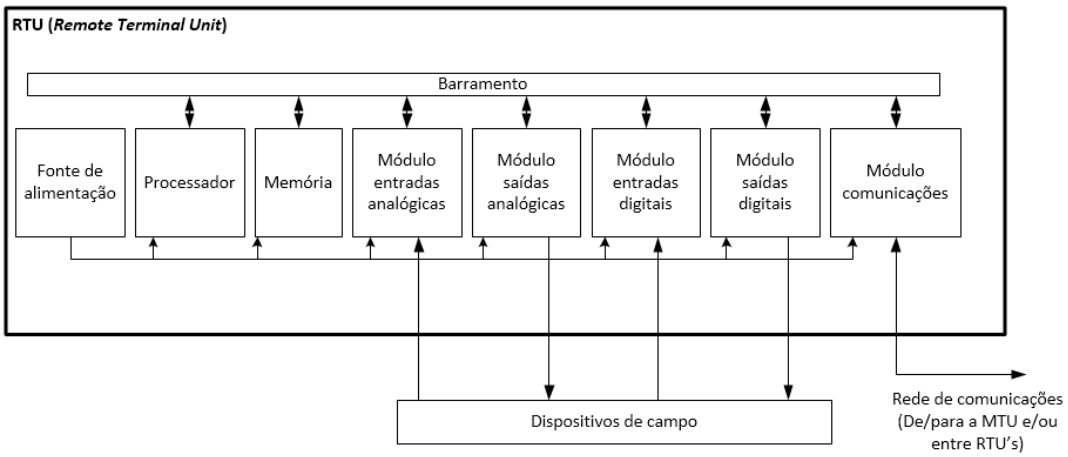

Figura 4.4 Diagrama exemplar de uma RTU

4.1.3 Comunicação

A comunicação entre a MTU e as RTU's é conseguida através de uma rede de comunicações, quer seja ela uma rede de campo com um protocolo industrial para sistemas mais pequenos utilizando um cabo como meio físico de transmissão, uma rede local *Ethernet* para instalações de média dimensão. Para instalações de dimensões maiores ou de difícil acesso podem ainda ser utilizados outros meios como ondas rádio *UHF/VHF*, micro-ondas, *wireless*, GSM (*Global System for Mobile Communications*), entre outros, estes meios são designados como canal de comunicação [31] [34].

Os sistemas SCADA ao efetuarem um controlo de alto nível podem utilizar redes e protocolos compatíveis com as características das redes comerciais ao invés das redes industriais pois não existe necessidade de comunicações determinísticas e com tempos de transmissão na ordem de poucos milissegundos, tal é economicamente vantajoso mas pode implicar questões de segurança. Na solução de automação apresentada neste trabalho é utilizada a rede *Ethernet* comercial como meio de comunicação para o sistema SCADA.

4.1.4 Workstations

Tal como referido anteriormente na MTU podem existir equipamentos adicionais como *workstations* para operadores, estas por norma não dispõem de capacidade de armazenamento apenas de visualização e comando, as *workstations* funcionam através

de uma rede local com a estação principal a qual funciona como servidor de dados das *workstations* [33].

Estas *workstations* podem ser utlizadas para dividir a operação de supervisão por setores, áreas ou funções e permitir que mais operadores interajam com o sistema SCADA.

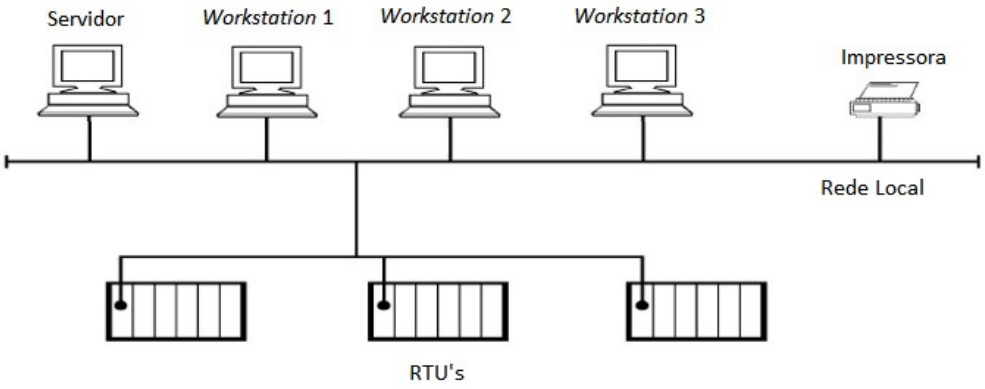

Figura 4.5 Exemplo de sistema SCADA com servidor de dados e workstations

Por exemplo, por questões de segurança de dados e de operação segundo [35] um sistema de supervisão pode ser dividido em cinco níveis de segurança para os quais podem existir cinco *workstations* diferentes ou com acessos diferentes.

Tabela 4.1 Exemplo de níveis de segurança num sistema SCADA

Nível de segurança	**Colaboradores**	**Funções disponíveis**
A	Todos	Visualização
B	Operadores Estagiários	Funções nível A Alterar *setpoints* Reconhecimento de alarmes Ligar e desligar equipamentos
C	Operadores Qualificados	Funções nível B Alterar alarmes Desativar controladores/alarmes
D	Técnicos Instrumentação	Funções nível A

		Calibrar controladores Relatórios de alarmes Configurações simples
E	Engenheiro de sistemas	Configurações complexas Códigos de segurança Gestão de contas

4.1.5 Software SCADA

Com o estabelecimento do *hardware*, MTU, RTU's e meio de comunicação entre ambos existe um caminho físico para a transferência de informação entre os dispositivos de campo e a supervisão, no entanto para que a informação seja obtida e processada de forma útil é necessário existir um *software* SCADA, o qual pode ser proprietário ou aberto.

Um *software* SCADA é um pacote de programas destinado à receção da informação em formato digital, salvaguarda da mesma em bases de dados, processamento automático da informação e criação e conexão dos écrans sinóticos às bases de dados e ao processamento da informação, este *software* pode ainda disponibilizar outros serviços de alto nível como servidores OPC, controladores (*Drivers*), entre outros. Estes serviços são disponibilizados como módulos independentes e podem ou não pertencer todos a um só "pacote" de SCADA, podem ainda ser executados em equipamentos de *hardware* diferentes. Por exemplo, alguns módulos podem correr em servidores *web*, de históricos ou de bases de dados providenciando acesso aos dados pelas *workstations* [36].

Tipicamente um *software* SCADA consiste em quatro módulos funcionais:

- Aquisição de dados (*Inputs / Outputs*)
- Base de dados (Armazenamento)
- Controlo e processamento (Alarmes, relatórios e estatísticas)
- Interface (*Displays*)

4.1.6 Aquisição de dados

A aquisição de dados de um sistema SCADA inicia-se nos dispositivos de campo (sensores e atuadores) que enviam e recebem sinais do seu controlador de processo (PLC-RTU) ou para um transmissor (RTU), a RTU envia os dados para a MTU através de uma rede de comunicação, rede industrial ou comercial.

Os dados provenientes das RTU podem chegar à MTU em um qualquer protocolo definido para a rede utilizada, no caso deste trabalho será utilizada a *Ethernet* comercial e o protocolo MODBUS TCP, para que a informação proveniente da RTU seja interpretada pela MTU é necessário utilizar um controlador de comunicações (*driver*) para o protocolo em causa, o módulo de *drivers* pode ou não ser parte integrante do *software* SCADA.

Após a interpretação dos dados pelo *driver* é então construída uma base de dados com os valores obtidos, os quais serão disponibilizados pela base de dados e posteriormente utilizados pelas restantes aplicações como a base de dados de histórico, alarmes e eventos e as interfaces com o utilizador [37].

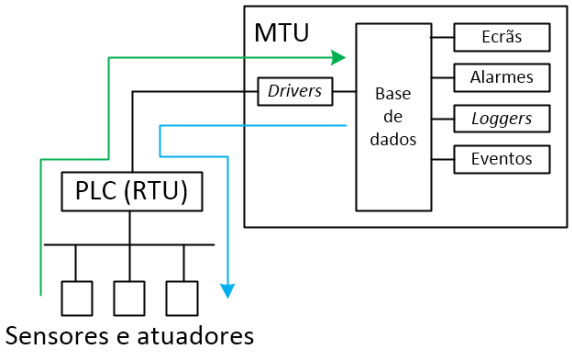

Figura 4.6 Fluxo de dados em um sistema SCADA

4.1.6.1 Controladores (Drivers)

Os controladores ou *drivers* são programas de *software* que permitem fazer a descodificação do sinal elétrico recebido nos portos de comunicação para um valor digital utilizável por outros programas de *software*.

Os *drivers* escritos num ficheiro *.dll* contêm uma configuração base para o tipo de ligação que fazem, sendo no entanto necessário configurar algumas características pretendidas, tais como o tempo de espera entre mensagens, o tempo máximo de espera

para a resposta (*timeout*), a prioridade do protocolo no caso de serem utilizados mais do que um e a possibilidade de designar um ficheiro de registos (*log file*) para a escrita de mensagens em caso de erro, tal ficheiro permitirá fazer posteriormente o *debug* e correção de erros.

4.1.6.2 OPC (Open Platform Comunications)

OPC é uma norma que consiste em especificações que permitem a uniformização na partilha de dados entre um cliente e um servidor, nomeadamente equipamentos de fabricantes distintos [38], esta norma permite a criação de uma interface entre redes de campo com protocolos proprietários e programas de *software* genéricos, utilizados para criar programas de supervisão em ambiente *windows* por exemplo

De acordo com a especificação OPC são criados pacotes de *software* chamados de servidores OPC DA (*OPC Data Access*) ou OPC *servers*. Estes servidores contêm conjuntos de *drivers* para estabelecer comunicações com redes de automação que utilizem protocolos de redes de campo e criam ligações entre os equipamentos da rede, por exemplo controladores, e uma base de dados em *software* que por sua vez contém os dados em um formato que pode ser acedido por outros programas informáticos, permitindo assim estabelecer uma interface entre um programa informático genérico e um controlador de uma rede de campo, a única exigência é que o formato dos dados obedeça à norma OPC.

Um programa de supervisão pode ser compatível com esta norma e como tal após utilizar os seus *drivers* para criar uma base de dados em tempo real a mesma é compatível com OPC, podendo ser utilizada por outros programas que recebam dados de um servidor OPC, os programas que não contêm *drivers* próprios necessitam de um servidor OPC externo para criar a base de dados, sendo considerado um cliente OPC.

Os programas que recorrem à base de dados OPC podem ser dos seguintes tipos:

- Servidor OPC: através de *drivers* de ligação aos dispositivos de campo cria uma base dados no formato OPC disponibilizando a informação aos restantes programas;
- Cliente OPC: recorre a uma base de dados no formato OPC criada por um outro programa;
- Servidor/Cliente OPC: executa ambas as funções.

Nota um programa cliente OPC apenas pode ser cliente pois não tem a capacidade de criar a base de dados mas um programa servidor OPC pode ser um servidor dedicado, criando a base de dados e disponibilizando a informação, ou ser um servidor/cliente com a capacidade de criar a base de dados e de utilizar bases de dados OPC criadas por outros programas.

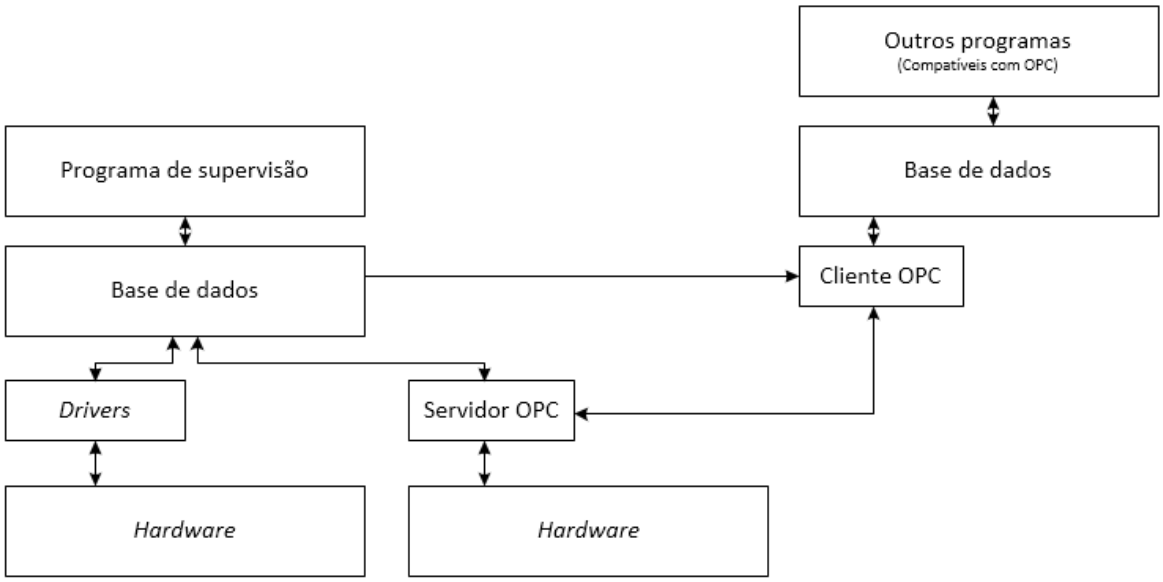

Figura 4.7 Fluxo de dados entre clientes e servidores OPC [37]

A motivação para a criação da OPC derivou da enorme diversidade de protocolos e *drivers* que podem existir em automação, tanto abertos como proprietários, a OPC passou então a ser a forma de criar uma ligação entre os programas informáticos de gestão e supervisão com uma rede de automação, permitindo o acesso dos mesmos aos dados da rede sem terem de ser os próprios programas a serem compatíveis com todos os dipositivos e redes de campo ou terem todos os *drivers* do mercado, criou-se então uma ferramenta que faz a ponte entre a informática e a automação [38].

4.1.7 Base de dados

O acesso à base de dados, também designada por base de dados em tempo real RTDB (*Real Time Database*), pode ser feito através de uma API (*Application Programming Interface*) designada por ODBC (*Open DataBase Conectivity*) a qual é uma aplicação normalizada para bases de dados, permitindo assim utilizar aplicações independentes para a criação e gestão da base de dados como a MSAccess, SQLServer

ou Oracle, os dados são acedidos na base de dados através de identificadores designados por *tags*.

4.1.7.1 Variáveis (Tags)

As *tags* são um tipo de meta dados que possibilita acrescentar outras informações a uma variável que não apenas o seu valor, como nomes, tipos, e outras propriedades.

O programa SCADA utiliza as *tags* como variáveis, estas assumem os valores digitais ou analógicos pretendidos permitindo a execução e controlo do programa, as *tags* contêm ainda diversas propriedades configuráveis.

Algumas das propriedades mais importantes das *tags* são:

- O nome configurável, que permite um fácil reconhecimento da mesma;
- O tipo de variável, *word*, *double word*, entre outros;
- Retenção, que permite manter o valor da variável após o desligar de uma ligação;
- A propriedade dinâmica, que permite estabelecer a ligação entre a *tag* e uma tarefa (*task*) no *driver* de comunicação.

4.1.7.2 Tarefas (Tasks)

As tarefas são instruções de leitura/escrita através do *driver*.

Cada *task* executa apenas uma instrução de leitura/escrita.

Cada *tag* pode conter apenas uma *task* na sua propriedade dinâmica mas uma *task* pode ser solicitada por diversas *tags*.

4.1.8 Controlo e processamento

O controlo da rede de automação (automático ou manual) e o processamento dos dados adquiridos são conseguidos através dos recursos ou API's do programa de supervisão, estes recursos podem ir desde ficheiros pré programados (*scripts* ou receitas) a históricos e alarmes, entre outros.

4.1.8.1 Scripts

Scripts são ficheiros escritos em linguagem VBA (*Visual Basic for Application*) que podem ser utilizados para executarem o código pretendido após uma condição de execução, por exemplo executar uma janela de aviso ou alterações a variáveis após a sinalização de um alarme ou evento.

4.1.8.2 Receitas

As receitas são ficheiros de texto que contêm valores pré definidos para as variáveis do programa sendo utilizados para ativar pontos de funcionamento gerais (conjunto de *setpoints*), por exemplo a receita 1 pode conter os *setpoints* para o fabrico do produto A e a receita 2 os *setpoints* para o produto B, ou para o funcionamento da instalação no modo X ou no modo Y.

4.1.8.3 Data Logger

A *Data Logger* é uma API que permite a criação de uma base de dados para gravação de variáveis ao longo do tempo, as variáveis selecionadas para serem gravadas podem posteriormente ser utilizadas para análise do sistema através de históricos, receitas, gráficos e construção de relatórios.

Por defeito a base de dados da *Data Logger* vem com os seguintes campos:

- Hora (referência GMT)
- Hora Local (outra referência)
- Duração (em ms)
- Utilizador (utilizador ativo)
- Razão (comando que despoletou a gravação)

Para o projeto de exemplo foi criado um *Data Log* de 1 segundo para gravação de um alarme designado por "*Generators not stable*" indicado pela *tag* DI00.7 (oitavo *bit* do registo zero de 16 *bits* do autómato TSX57-103), após a ativação do alarme a *Data Logger* fará uma leitura e registo do estado da *tag* a cada segundo.

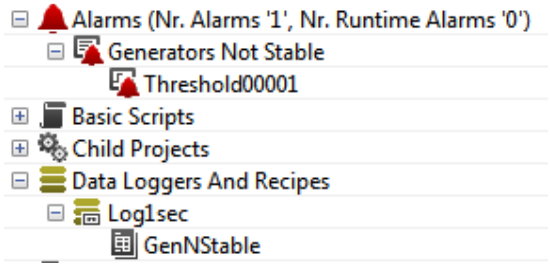

Figura 4.8 Configuração de um Data Log para um evento de alarme

4.1.8.4 Alarmes

Os alarmes são constituídos por condições definidas sobre os dados presentes na RTDB e a sua função é alertar o operador e gerar um relatório quando a *tag* afeta ao alarme atingir a condição definida, a condição para o alarme x é definida como a *tag* y atingir o valor z.

Após a condição de alarme ser atingida é registada a ativação do alarme e guardada em uma base de dados de alarmes, a qual pode ser utilizada por outras aplicações para gerar comandos, alertas em ecrãs informativos, janelas de alarmes, envio de SMS (*Short Message Service*) ou *email*.

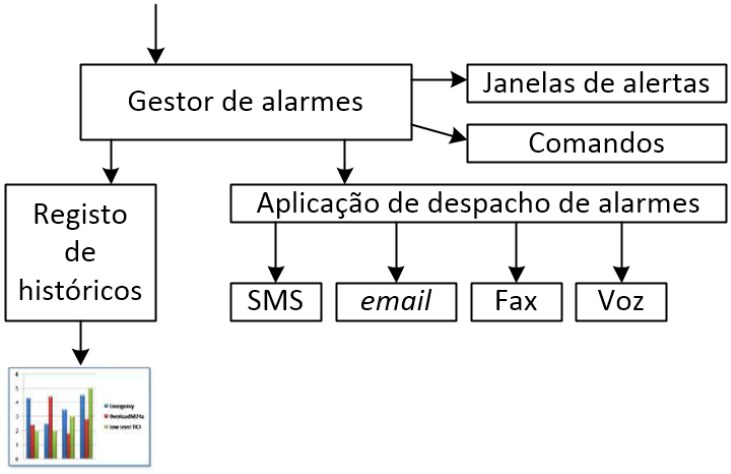

Figura 4.9 Aplicação de gestão de alarmes

4.1.8.5 Utilizadores

Um programa SCADA deve conter uma lista de utilizadores com acessos e propriedades distintas tais como administradores, desenvolvedores, utilizadores e visitantes por exemplo.

Cada grupo de utilizadores e cada utilizador podem ter diversas propriedades como o nível de acesso, descrição, *passwords*, tempo de validade das *passwords*, contactos e acesso a notificações.

4.1.9 HMI (Human Machine Interface)

O HMI é um painel ou ecrã de visualização e comando que permite a interface do sistema SCADA com os operadores, de acordo com [34] a interface para a operação deve ser construída de acordo com a norma MIL-STD-1472G que define os critérios e recomendações para os painéis de informação, capacidades de controlo, cores e interação com o utilizador.

Segundo os requerimentos gerais da norma MIL-STD-1472G os HMI devem, providenciar ambientes de trabalho que promovam a eficácia, padrões de trabalho e segurança do pessoal, devem ainda minimizar a possibilidade de erro humano, permitir a execução das tarefas de forma eficiente e precisa, ser desenhado de acordo com as capacidades e limitações dos operadores, entre outros, sendo ainda de elevada importância que os sistemas SCADA e em particular os HMI sigam regras normalizadas para os controlos, *displays*, marcações, codificações, painéis e *layouts* e utilizem funções uniformes com os restantes equipamentos e sistemas.

O painel é construído através de ecrãs sinópticos (*screens*) nos quais são desenhados botões de comando, caixas de texto e etiquetas de texto (*labels*) assim como ligações para outros painéis, cada botão de comando ou caixa de texto pode ser interligado a uma variável (*tag*), os comandos podem ainda conter conjuntos de instruções (*scripts*) a utilizar em determinadas situações.

4.2 Projeto SCADA

Nesta dissertação serão desenvolvidos dois projetos SCADA, um projeto de comunicação que permitirá enviar e receber mensagens dos equipamentos de campo de acordo com o preconizado no tema proposto e um projeto de exemplo de uma aplicação industrial. Os projetos serão criados por um *software* SCADA neste caso será utilizado o *software* Movicon, os mesmos são constituídos por um conjunto de recursos e aplicações dos quais se destacam os seguintes:

- Base de dados em tempo real
- Controladores (*Drivers*)
- Variáveis (*Tags*)
- Ecrãs
- Alarmes
- *Data Loggers*
- Utilizadores

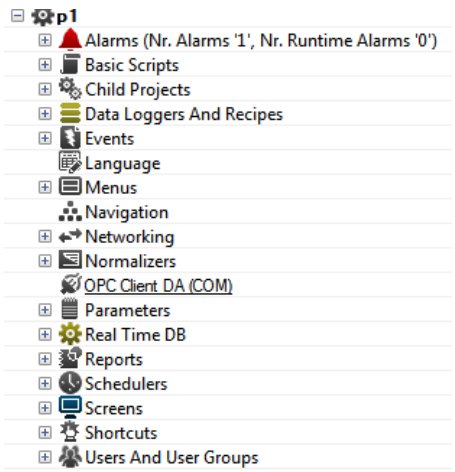

Figura 4.10 Estrutura de recursos do projeto SCADA

4.2.1 Estrutura física e meio de comunicação do projeto SCADA

A estrutura física utilizada é a estrutura em estrela através de uma rede *Ethernet*, a MTU composta por um computador pessoal com o *software* SCADA é ligada através de um cabo UTP cat5 a uma tomada do tipo RJ45 proveniente de um *switch ethernet*, o qual estará ligado a um outro *switch ethernet* mais próximo da rede de automação, que por sua vez fará a ligação a uma nova tomada RJ45 à qual estará ligada à RTU, sendo neste caso utilizadas duas RTU, os autómatos TSX57.

Será utilizado o protocolo MODBUS TCP para comunicação entre os autómatos TSX57 e a supervisão, através de uma rede *Ethernet* e o *driver* para ligação *Ethernet* MODBUS TCP.

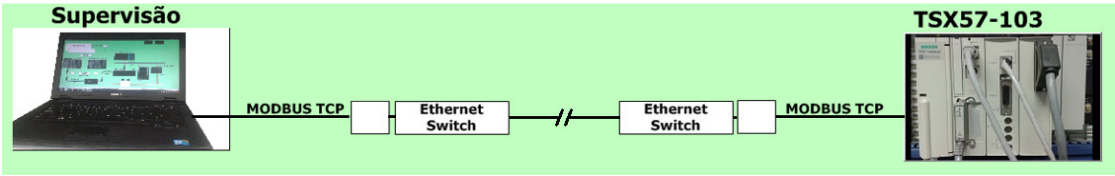

Figura 4.11 Ligação física entre a supervisão e a RTU (PLC TSX57-103)

4.2.2 Software SCADA utilizado

Para implementação do projeto SCADA será utilizado o *software* MOVICON.

Movicon é um *software* SCADA proprietário, desenvolvido pela Progea, que permite a criação de um programa de supervisão com os módulos usuais de um sistema SCADA complexo num só pacote de *software*, uma vantagem deste *software* é que a integração destes módulos simplifica e minimiza o número de programas necessários, a complexidade e a possibilidade de erros na implementação de um sistema completo, tornando o *software* ideal para criar programas de supervisão baseados em apenas um equipamento MTU, no caso desta dissertação e por se tratar de um programa pouco complexo é utilizado apenas um computador pessoal como MTU.

A base de dados a utilizar será a RTDB criada, utilizando os *drivers* do próprio *software* MOVICON conjuntamente com a aplicação MSAccess.

Serão utilizadas duas estações, uma para cada autómato TSX57 em que ambas utilizam o *driver* MODBUS TCP.

Em termos de tarefas (*tasks*) serão utilizadas apenas tarefas dinâmicas associadas uma tarefa a cada variável, as tarefas utilizadas serão de leitura e escrita de registos, tanto de 16 *bits* como de 32 *bits* em diversas posições de memória dos autómatos.

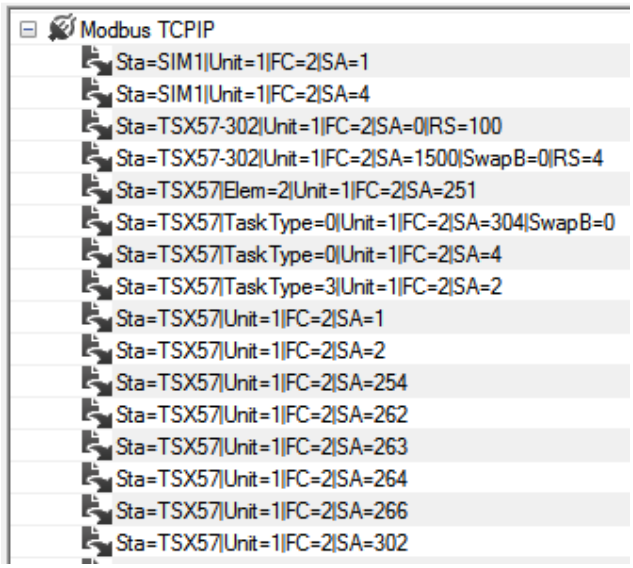

Figura 4.12 Tarefas utilizadas no driver MODBUS TCP

Não será utilizado um servidor OPC mas em caso de necessidade a base de dados criada pode ser utilizada por outros programas sendo este projeto o servidor OPC.

Neste projeto não serão utilizados *data logger*, *scripts*, receitas ou alarmes no projeto exemplo de instalação industrial é utilizado um *data logger* e alarmes.

4.2.3 Configurações

No *software* Movicon as RTU são designadas por estação (*Station*) e cada RTU com ligação à supervisão deve ser configurada no programa de acordo com o seu *driver* de comunicação. Note-se que cada *driver* de comunicação pode conter várias estações, cada uma com um endereço único.

Após a configuração dos *drivers* é necessário criar uma *station* com o *driver* MODBUS TCP à qual será estabelecida a ligação à rede *ethernet*, no caso deste projeto a *station* a criar será referente ao autómato TSX-57 103 da MODICON.

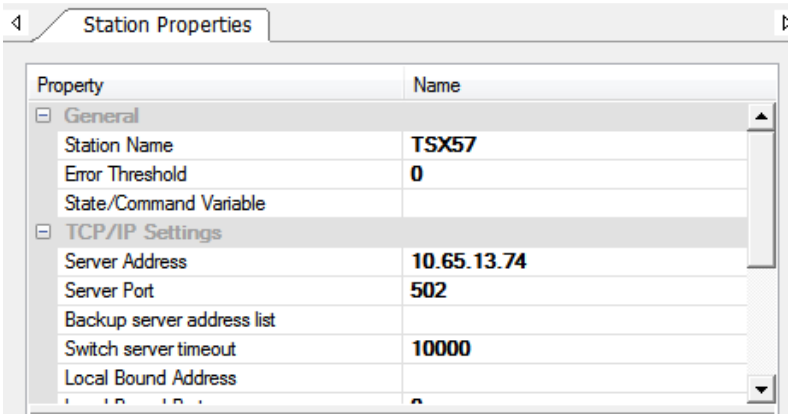

Figura 4.13 Configuração da estação TSX57 no driver para MODBUS TCP

Para que a ligação entre o *software* SCADA e o autómato seja estabelecida é necessário configurar o endereço IP do autómato tanto na configuração de *hardware* do mesmo como na estação do *software* SCADA, neste caso foi configurado com o endereço IP 10.65.13.74, o endereço IP da RTU.

Por fim é necessário também indicar qual o I/O Port (*Input/Output Port*) do computador a utilizar, neste caso foi utilizado o porto por defeito, o porto 502.

A estrutura de *software* utilizada no programa SCADA compreende o painel sinótico com as caixas de texto com ligação à base de dados e às variáveis através das *tags*, cada uma com a sua *task* que através do *driver* fazem a ligação à *station* pela porta de comunicação.

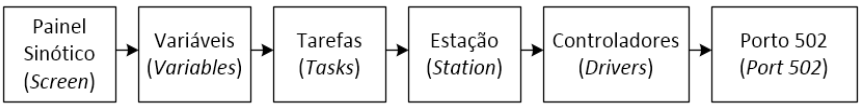

Figura 4.14 Fluxo de dados desde um comando no painel sinótico até ao porto de comunicações

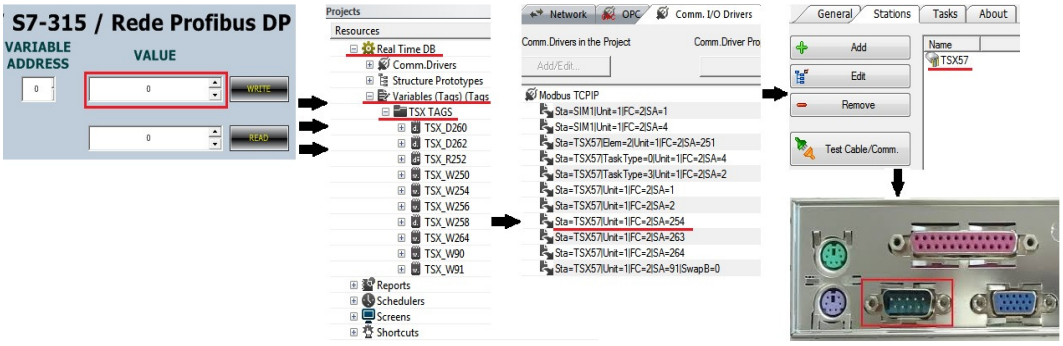

Figura 4.15 Fluxo de dados do projeto criado com o software Movicon

4.2.4 Protocolo MODBUS TCP

O protocolo de comunicação a utilizar entre o *software* SCADA e o autómato TSX57-103 será o MODBUS TCP, o qual utiliza uma mensagem base em MODBUS acrescentando um cabeçalho TCP à mensagem original para identificação da MODBUS ADU (*MODBUS Application Data Unit*).

O cabeçalho TCP acrescenta informação à mensagem original necessária em uma rede *ethernet* o que não acontece em uma rede de campo mais simples, aumentando o número de *bytes* necessários à sua transmissão e consequentemente aumentando o tempo de transmissão e processamento, sendo que a velocidade de transmissão na rede *ethernet* é muito superior às redes de campo utilizadas, tornando este aumento do tamanho da mensagem negligenciável.

O protocolo MODBUS TCP utiliza ainda um endereço IP para identificar o destinatário em vez do endereço de escravo (*slave address*) como no protocolo original MODBUS.

4.2.5 Tipos de Mensagens e Estrutura

Os tipos de mensagens possíveis através do protocolo MODBUS TCP são os mesmos que no protocolo MODBUS, escrita/leitura de saídas/entradas binárias, saídas/entradas analógicas e registos.

Neste projeto o tipo de mensagens enviadas/recebidas em MODBUS TCP será apenas na forma de escrita/leitura de registos, tanto registos únicos como duplos (*single register/multiple registers*) de variáveis de 16 e 32 *bits*, *word* e *double word*.

O autómato TSX57-103 encontra se equipado apenas com módulos de comunicação e não com módulos de entradas/saídas, sendo que para as restantes mensagens o processo é o mesmo bastando apenas alterar o tipo de mensagem pretendida.

Como descrito no capítulo anterior, o autómato TSX57-103 quando receber uma mensagem em determinados registos processa e converte a mensagem para MODBUS RTU e envia-a para os respetivos *slaves* se for caso disso.

Torna-se assim possível enviar e receber informação para outro tipo de saídas, por exemplo saídas binárias do S7-200, através de mensagens do tipo registo.

4.2.6 Ecrãs HMI

O painel principal criado para este projeto dá uma indicação clara da topologia e composição da rede integrada, onde é possível ter uma indicação visual dos equipamentos utilizados. Através de *labels* ou etiquetas são indicados os nomes dos equipamentos, a sua função na rede (*master* ou *slave*), o tipo de interface para a rede (RS232 ou RS485) e o protocolo utilizado na mesma (MODBUS RTU, MODBUS TCP, PROFIBUS DP ou AS-i). As cores das ligações são representativas das cores das cablagens utilizadas.

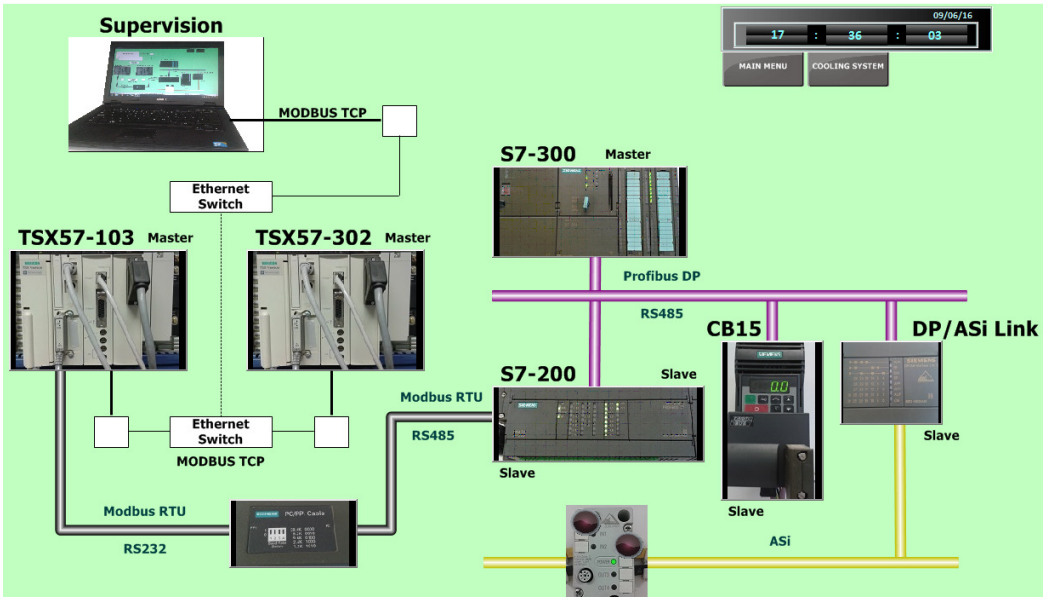

Figura 4.16 Ecrã principal do HMI criado para o projeto SCADA

Através do ecrã principal é possível selecionar cada equipamento presente na rede e abrir um painel secundário para envio de uma mensagem da supervisão para esse equipamento, as mensagens são enviadas no formato do protocolo MODBUS e são enviadas para o autómato TSX57-302 ou para o TSX57-103 que fará o encaminhamento das mesmas para outros dispositivos se necessário.

Neste ecrã existe ainda no canto superior direito dois botões de comando para alternar entre o ecrã principal e um projeto SCADA de exemplo de um sistema de arrefecimento de uma instalação industrial.

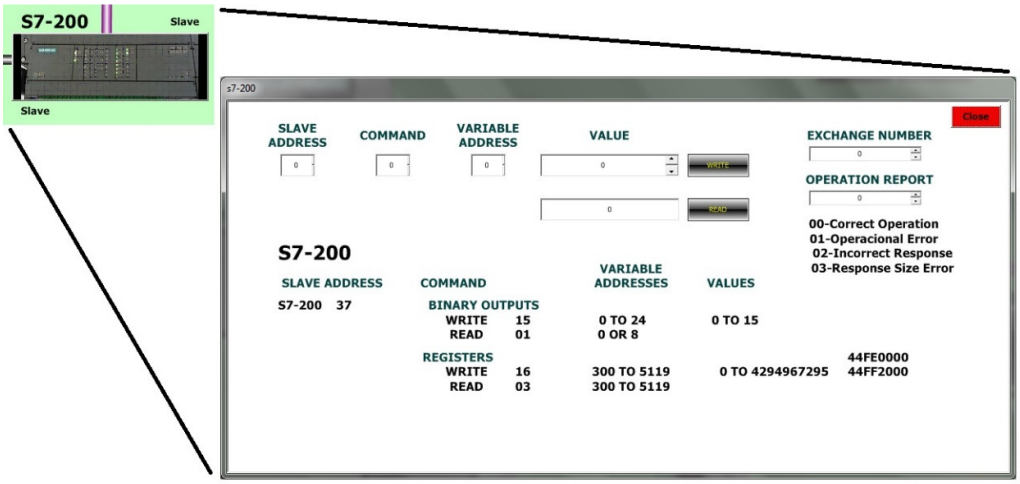

Figura 4.17 Ecrã secundário do HMI para envio de mensagens para o autómato S7-200

No ecrã para envio de mensagem é necessário indicar os valores para a mensagem conforme os campos previstos no protocolo, tais como o endereço de escravo, comando e endereço de variáveis. Na zona inferior são indicados valores (em binário) de exemplo para mensagens de escrita ou leitura de variáveis binárias de saída ou registos. No campo *VALUE* na parte superior é escrito o valor pretendido para uma mensagem de escrita e na parte inferior é apresentado o resultado de uma mensagem de leitura.

4.2.7 Projeto exemplo

O projeto de exemplo de um sistema de supervisão de uma instalação industrial permitirá observar a aplicação destes sistemas de supervisão assim como algumas das suas capacidades de supervisão do processo de forma automática e utilização dos seus recursos, como por exemplo a utilização de alarmes.

O projeto exemplo de supervisão permitirá então a visualização e supervisão do comportamento da instalação em funcionamento normal e em caso de falha no fornecimento de energia por parte da rede de distribuição. A falha é simulada através de uma variável binária associada a um botão de comando onde após o acionamento da mesma é automaticamente ativado o procedimento para fornecimento de energia por parte de um grupo gerador de socorro. Este procedimento é ativado automaticamente e realizado através de instruções pré programadas e troca de mensagens entre autómatos. No ecrã de supervisão será possível visualizar o resultado da operação e quais os equipamentos em funcionamento a cada instante. Após o restabelecimento de energia da rede (fim da simulação) é automaticamente executado o processo inverso.

Em caso de falha dos geradores é acionado um alarme que dará essa indicação ao operador através de sinalização visual e sonora, assim como fará o registo temporal e histórico dos alarmes.

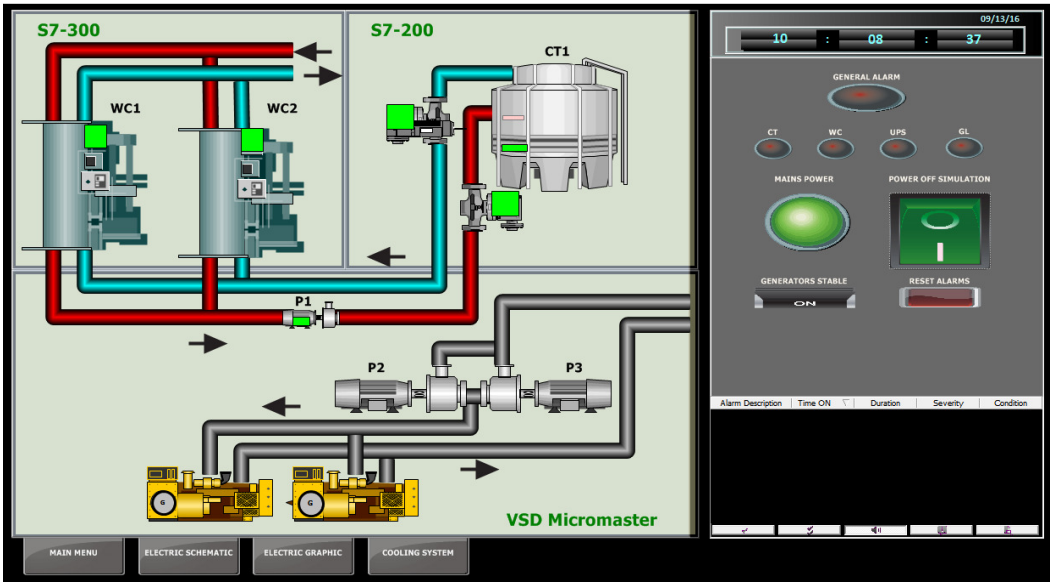

Figura 4.18 Ecrã principal do HMI criado para o projeto exemplo

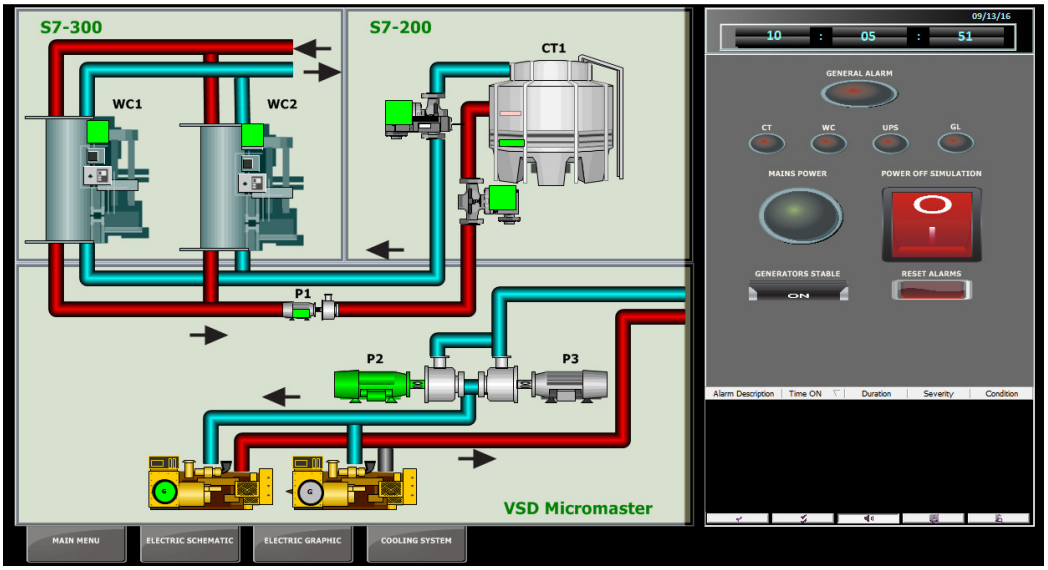

Figura 4.19 Painel HMI com os geradores em funcionamento

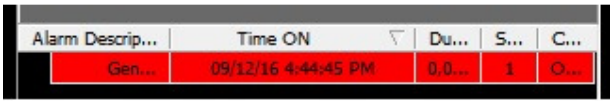

Figura 4.20 Exemplo de alarme ativo

Capítulo 5

Conclusões

Resumo: No presente capítulo expõem-se as conclusões retiradas com a elaboração desta dissertação no âmbito da área de automação industrial e apresentam-se diversas possibilidades para estudos futuros.

5.1 Conclusões

Esta dissertação foi elaborada com o intuito de explorar a integração de redes de automação provenientes de diferentes fabricantes de modo a analisar as suas potencialidades e, na qual se utilizaram os equipamentos já existentes nos laboratórios de Automação e Robótica sem alterações significativas em termos de disposição ou capacidades por forma a realizar o estudo na forma mais próxima possível de uma situação real.

A rede utilizada nesta dissertação, que não tendo as necessidades específicas de uma rede de *Ethernet* Industrial para a componente de supervisão, por questões económicas, de demonstração de capacidades e práticas utiliza a rede *Ethernet* comercial já existente no campus como rede de automação.

Resultante da análise efetuada aos equipamentos existentes verificou-se que ambos os autómatos TSX57 possuíam ligação à rede *Ethernet* comercial e que com algum esforço de configuração e programação seria possível ligar diretamente estes equipamentos à supervisão. Esta análise veio a confirmar-se e foi possível ligar com sucesso estes equipamentos ao sistema de supervisão desenvolvido. Verificou-se que o autómato S7-300 e os seus equipamentos *slave* ligados na rede de campo PROFIBUS-DP e rede ASi sendo suportados pelo mesmo fabricante apenas apresentariam dificuldades em termos de configurações. Estes já tinham provas dadas de bom funcionamento nas aulas de Redes de Automação e Supervisão.

A situação mais complexa, que veio de facto a verificar-se, foi então a integração das duas redes de campo, uma com equipamentos da Schneider com protocolo MODBUS e interface de acesso RS232 e outra com equipamentos da Siemens com protocolo PROFIBUS-DP e interface de acesso RS485. A solução passou pela utilização de um cabo com conversor de interfaces e com uma adequada programação do protocolo MODBUS no autómato S7-200 da Siemens. A utilização do autómato S7-200 trouxe algumas dificuldades adicionais por ser um dispositivo *slave* na rede PROFIBUS-DP com consequente incapacidade para enviar mensagens, tendo-se tornado necessário adotar um sistema de *buffers* com leitura cíclica dos mesmos. Inicialmente temeu-se que o atraso proveniente pela leitura cíclica, execução das mensagens e sua validação por CRC poderiam adicionar atrasos significativos nas mensagens levando a falhas no envio das respostas para o autómato TSX57-103 e

consequentemente para a supervisão. Tal facto não veio a verificar-se experimentalmente, constatando-se que a deteção de mensagens novas por variação de caracteres nos *buffers* e a limitação no tamanho destes *buffers* é uma solução perfeitamente viável não se obtendo ocorrências de *timeout* causadas por tempos de mensagem superiores a 100 ms.

No entanto pode concluir-se também que a programação do protocolo no ciclo principal do autómato S7-200 implica que o mesmo tenha tempos de ciclo elevados e a utilização de recursos do próprio autómato, tal como como a sua memória disponível. Além disso poderá ser necessária a compatibilização do protocolo com o programa específico a usar de modo a não afetar a receção assíncrona de mensagens, nomeadamente a nível de interrupções, temporizações e utilização de memória interna no caso de serem necessários em programas de controlo mais complexos.

É de referir que a programação do protocolo no ciclo principal do S7-200 incidiu na receção e processamento de uma única mensagem e consequente envio de resposta. A receção de múltiplas mensagens pode ser possível sendo que implicará uma programação bastante mais complexa e a utilização ou reserva de recursos adicionais de modo a conseguir um encadeamento em exclusividade perfeita entre a ativação da receção e a transmissão das respostas. A forma como foi programado o protocolo não impedirá que existam tentativas de receção de mensagens em simultâneo com a transmissão de uma resposta o que resultará em erro na mensagem a receber.

Na implementação do protocolo MODBUS RTU no autómato S7-200 foram sempre consideradas as disposições das normas aplicáveis [11] e [25] com o intuito de criar um programa capaz da receção de mensagens segundo o protocolo de um modo generalista, não sendo restrito a esta aplicação em particular. Apesar de não terem sido contempladas todas as funções possíveis o programa implementado cumpre os requisitos propostos para esta dissertação e poderá permitir a comunicação do autómato S7-200 com outros dispositivos que utilizem o protocolo MODBUS em modo RTU, assim como permitirá a sua utilização em conjunto com outros programas de controlo que se possam implementar no autómato. Este tipo de implementação (no programa principal em modo *Freeport*) permite total flexibilidade na implementação do protocolo e como tal é possível criar versões customizadas e adaptadas do protocolo, implementando apenas os comandos estritamente necessários de modo a facilitar a gestão de eventuais incompatibilidades, pode utilizar menos recursos e podem ser

implementados apenas os comandos necessários. No que diz respeito ao autómato S7-300 foi implementada uma versão mais simplificada deste mesmo protocolo.

Em termos de *hardware* verificou-se que a necessidade de criação de uma ligação física entre as redes (caixa de ligação) implica que a mesma tenha de cumprir os mesmos requisitos do restante *hardware* da rede, nomeadamente em termos de fiabilidade e adequação ao ambiente onde se insere, este fato terá implicação na qualidade das soldaduras, materiais, isolamentos e terminais.

Conclui-se neste caso que a integração das diversas redes e equipamentos disponíveis é viável e funcional para aplicações não muito complexas ou exigentes em termos de tempos de comunicação para controlo em tempo real. Para aplicações mais complexas recomenda-se a utilização de equipamentos apropriados e módulos de comunicação dedicados ao efeito disponibilizados pelos próprios fabricantes.

Em termos de supervisão conclui-se que estes sistemas podem ser de elevada complexidade, integrando tanto a automatização de processos como a informática em termos de tratamento e visualização de dados, funcionando como poderosas ferramentas de gestão e apoio à decisão.

5.2 Perspetivas de desenvolvimento futuro

Em termos de desenvolvimento futuro propõem-se os seguintes pontos:

- Implementação completa do protocolo MODBUS (modos ASCII e RTU) no programa principal de um autómato, na sua forma geral e compatível com qualquer programa de controlo e possibilidade de receção de múltiplas mensagens.
- Estudo da qualidade de serviço de uma rede integrada, através da análise dos tempos de transmissão e processamento, quantidade de erros ocorridos e estudo de formas de melhoramento.
- Estudo sobre a aplicação de redes e protocolos de *Ethernet* industrial em redes de automação envolvendo controlo em tempo real e sistemas de segurança de pessoas e bens, nomeadamente pela aplicação e influência dos *switches* (*managed* ou *unmanaged*) e *gateways* ou com a introdução na rede de autómatos com protocolos programados no seu ciclo principal.

Bibliografia

[1] E. Hayden, A. Michael e C. Tim, "An Abbreviated History of Automation & Industrial Controls Systems and Cybersecurity," 2014.

[2] H. Kirrmann, "Automation Overview," 2005.

[3] J. Palma, "Introdução às redes de campo de automação," 2004.

[4] B. Galloway e G. P. Hancke, "Introduction to Industrial Control Networks," 2012.

[5] Cisco; Rockwell Automation, "Converged Plantwide Ethernet Design and Implementation Guide," 2011.

[6] M. Asad, "Essentials of Industrial Network".

[7] Cisco, "Industrial Ethernet: A Control Engineer's Guide," 2010.

[8] PROFIBUS International, "PROFIBUS System Description," 2010.

[9] International Electrotechnical Commission, "IEC 60079-27," 2005.

[10] IDC Technologies, "IDC Engineering Pocket Guide - Industrial Automation," 2007.

[11] Modbus.org, "MODBUS over Serial Line - Specification and Implementation Guide," 2006.

[12] V. Soares, "Aquisição e Processamento de Sinais," 2014.

[13] ANACOM, "Manual ITED 3ª Edição," 2015.

[14] The Fiber Optics Association, "thefoa.org," [Online].

[15] CSIA, "controlsys.org," [Online].

[16] Schneider Electric, "Industrial Networks".

[17] Germany Trade and Invest, “Industrie 4.0, Smart Manufacturing for the Future,” 2014.

[18] N. Mahalik, Fieldbus Technology, 2003.

[19] acatech, “Cyber-Physical Systems,” 2011.

[20] Schneider Electric, “TSX 57/PCX 57 Processors - Implementation Manual Volume 1,” 2008.

[21] SIEMENS, “SIMATIC S7-200 Programmable Controller System Manual,” 2008.

[22] SIEMENS, “SIMATIC S7-300 CPU 31xC and CPU 31x: Technical specifications,” 2011.

[23] SIEMENS, “MICROMASTER 420 0.12 kW - 11 kW Operating Instructions,” 2006.

[24] SIEMENS, “AS-Interface – Introduction and basics,” 2006.

[25] MODBUS, “MODBUS Application Protocol Specification V1.1b3,” 2012.

[26] A. Daneels e W. Salter, “What is Scada?,” em *International Conference on Accelerator and Large Experimental Physics Control Systems*, 1999.

[27] L. Amy, “Automation Systems for Control and Data Acquisition,” 1992.

[28] G. Clarke, D. Reynders e E. Wright, Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems, 2004.

[29] Department of the Army, “SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) SYSTEMS FOR C4ISR FACILITIES,” 2006.

[30] United States General Accounting Office, “CRITICAL INFRASTRUCTURE PROTECTION - Challenges in Securing Control Systems,” 2003.

[31] A. M. Alihussein, “A SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) FOR WATER DISTRIBUTION SYSTEM OF GAZA CITY,” 2010.

[32] BUREAU OF INDIAN STANDARDS, “SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) SYSTEM FOR POWER SYSTEM APPLICATIONS,” 2011.

[33] NATIONAL COMMUNICATIONS SYSTEM, “Supervisory Control and Data Acquisition (SCADA) Systems,” 2004.

[34] The Institute of Electrical and Electronics Engineers, Inc., “IEEE Std C37.1-1994,” 1994.

[35] S. Boyer, SCADA: Supervisory Control and Data Acquisition 3rd Edition, 2004.

[36] D. Bailey e E. Wright, “Practical SCADA for Industry,” 2003.

[37] Progea, “MOVICON Programmer Guide,” 2012.

[38] OPC Foundation, “OPC Unified Architecture - Pioneer of the 4th industrial (r)evolution,” 2014.

Outra Bibliografia:

- IEC 61158-2
- SIEMENS; “TP041B ”, Program Example, 1999
- SIEMENS; “TP052B”, Program Example, 1999
- Microwin STEP7 V4.0 – Help
- PL7PRO - Manual
- thefoa.org
- Modbus.org
- Profibus.com
- Support.industry.siemens.com
- plctalk.net
- ad.siemens.com.cn
- simplymodbus.ca
- w3.org
- opcfoundation.org
- controleng.com/