Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTAMENTO DE ENGENHARIA
ELETROTÉCNICA

Melhorias em Sistema Automático de Gestão e Armazenamento de Acervos

Trabalho de Projeto para a obtenção do grau de Mestre em Engenharia Eletrotécnica

Especialização em Automação e Comunicações em Sistemas Industriais

Autor

David Antunes Correia

Orientador

Doutor Inácio de Sousa Adelino da Fonseca

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, dezembro, 2025

Resumo

Os Sistemas Automáticos de Armazenamento e Recuperação (*Automated Storage and Retrieval Systems* – AS/RS) desempenham um papel fundamental na modernização dos processos logísticos, permitindo aumentar a eficiência operacional, otimizar a utilização do espaço e melhorar o acompanhamento dos bens armazenados. No entanto, muitos sistemas atualmente em funcionamento foram desenvolvidos com tecnologias que se encontram desatualizadas, o que origina limitações ao nível da fiabilidade, da manutenção e da integração com soluções de software modernas.

O presente trabalho de projeto incide sobre a análise e a melhoria de um sistema automático de gestão e armazenamento de acervos do tipo **miniload**, pertencente à Universidade de Coimbra e em funcionamento desde o início dos anos 2000. O sistema em estudo foi analisado com base no seu funcionamento real, permitindo identificar a possibilidade de melhorias, nomeadamente ao nível da arquitetura de controlo, das comunicações industriais e da integração entre o sistema físico e o software de gestão.

Com base na análise efetuada, são propostas melhorias à arquitetura existente, com foco na modernização do sistema de controlo, na reorganização das comunicações entre o PLC e o sistema de gestão, e na integração de soluções de software mais flexíveis e escaláveis. O trabalho contempla ainda a implementação prática de parte das soluções propostas, recorrendo a tecnologias amplamente utilizadas na automação industrial, como controladores lógicos programáveis Beckhoff, o ambiente TwinCAT e a integração com aplicações desenvolvidas em Python.

Os resultados obtidos demonstram que as melhorias propostas contribuem para um sistema mais simples, robusto, flexível e alinhado com os princípios da automação industrial moderna, facilitando a manutenção, a expansão futura e a integração com sistemas de nível superior. Este trabalho evidencia a viabilidade da modernização de sistemas AS/RS existentes, constituindo uma abordagem técnica e economicamente vantajosa face à substituição integral das infraestruturas.

Palavras-chave: AS/RS; Miniload; Automação Industrial; PLC; Sistemas de Armazenamento Automático; Comunicações Industriais.

Abstract

Automated Storage and Retrieval Systems (AS/RS) play a fundamental role in the modernization of logistics processes, enabling increased operational efficiency, optimized space utilization, and improved tracking of stored goods. However, many systems currently in operation were developed using outdated technologies, resulting in limitations in terms of reliability, maintenance, and integration with modern software solutions.

This project work focuses on the analysis and improvement of an automated archive management and storage system of the miniload type, belonging to the University of Coimbra and in operation since the early 2000s. The system under study was analyzed based on its real operational behavior, allowing the identification of possible improvements, particularly regarding the control architecture, industrial communications, and the integration between the physical system and the management software.

Based on the conducted analysis, improvements to the existing architecture are proposed, with a focus on the modernization of the control system, the reorganization of communications between the PLC and the management system, and the integration of more flexible and scalable software solutions. The work also includes the practical implementation of part of the proposed solutions, using technologies widely adopted in industrial automation, such as Beckhoff programmable logic controllers, the TwinCAT environment, and integration with applications developed in Python.

The obtained results demonstrate that the proposed improvements contribute to a simpler, more robust, flexible, and modern system, facilitating maintenance, future expansion, and integration with higher-level systems. This work highlights the feasibility of modernizing existing AS/RS installations, representing a technically and economically advantageous alternative to the complete replacement of existing infrastructures.

Keywords: AS/RS; Miniload; Industrial Automation; PLC; Automated Storage Systems; Industrial Communications.

Epígrafe

O ontem é história, o amanhã é um mistério, mas o hoje é uma dádiva. É por isso que se chama presente.

Agradecimentos

Sou grato ao contributo e apoio de várias pessoas e instituições que tornaram possível a realização do presente trabalho de projeto

Em primeiro lugar, agradeço ao Professor Doutor Inácio de Sousa Adelino da Fonseca, por ser o meu orientador deste trabalho, pela a disponibilidade, o acompanhamento e a orientação científica ao longo de todo o desenvolvimento do projeto. As suas sugestões, o espírito crítico e os conhecimentos técnicos foram fundamentais para a definição do enquadramento do trabalho e para a consolidação das soluções propostas.

À Universidade de Coimbra, e em particular ao SGIP, serviço responsável pela gestão e manutenção do sistema automático de armazenamento em estudo, é devido um agradecimento especial pela oportunidade concedida. O contacto direto com o sistema *miniload*, bem como o acesso ao seu funcionamento e às suas infraestruturas, foram determinantes para a compreensão dos problemas reais associados aos sistemas industriais em operação e para a definição das propostas de melhoria apresentadas neste relatório.

Agradecer ao Instituto Superior de Engenharia de Coimbra (ISEC), porque sem este instituto não era possível todo este caminho. Agradeço à formação académica e técnica proporcionada ao longo do Mestrado em Engenharia Eletrotécnica, com especialização em Automação e Comunicações em Sistemas Industriais, que forneceu a base de conhecimentos necessária para a realização deste trabalho. Um agradecimento é igualmente dirigido a todos os docentes que, direta ou indiretamente, contribuíram para o desenvolvimento das competências aplicadas neste projeto.

Por fim, e não menos importante, agradeço aos meus pais, à minha namorada e à restante família e aos amigos pelo apoio, incentivo e compreensão demonstrados ao longo deste percurso académico, especialmente nos momentos de maior exigência e dedicação, cujo contributo foi essencial para a concretização deste objetivo.

Índice

Índice de tabelas

Índice de figuras

Lista de abreviaturas

ACO	Advanced Control Object
ADS	Automated Distribution System
AMS	Automated Manufacturing System
API	Application Programming Interface
AS/RS	Automatic Storage and Retrieval System
AWS	Amazon Web Services
DA	Data Acquisition
DB	Database
GA	Genetic Algorithm
HMI	Human–Machine Interface
HTML	HyperText Markup Language
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	Internet of Things
IP	Internet Protocol
ISEC	Instituto Superior de Engenharia de Coimbra
MQTT	Message Queuing Telemetry Transport
OPC	Open Platform Communications
PLC	Programmable Logic Controller
REST	Representational State Transfer
SCADA	Supervisory Control and Data Acquisition
SQL	Structured Query Language
SRM	Storage & Retrieval Machine
SVG	Scalable Vector Graphics
TCP	Transmission Control Protocol
VLM	Vertical Lift Module
WEB	World Wide Web
WCS	Warehouse Control System
WMS	Warehouse Management System

1 Introdução

A crescente complexidade das cadeias logísticas e a necessidade de otimização dos processos de armazenamento têm impulsionado, nas últimas décadas, a adoção de sistemas de automação industrial cada vez mais avançados. Entre estes, os Sistemas Automáticos de Armazenamento e Recuperação (*Automated Storage and Retrieval Systems* – AS/RS) assumem um papel fundamental na melhoria da eficiência operacional, na maximização da utilização do espaço disponível e no aumento do acompanhamento e segurança dos bens armazenados. Estes sistemas permitem reduzir a dependência da intervenção humana, minimizar erros operacionais e garantir tempos de resposta mais curtos, tornando-se elementos-chave no contexto da Indústria 4.0 e da digitalização dos processos industriais.

Apesar das vantagens associadas aos sistemas AS/RS modernos, muitas instalações atualmente em operação foram projetadas e implementadas há várias décadas, recorrendo a arquiteturas de controlo e tecnologias que, embora robustas à data da sua conexão, apresentam hoje limitações significativas. A evolução dos controladores lógicos programáveis (PLCs), dos protocolos de comunicação industrial, dos sistemas de supervisão e dos softwares de gestão de armazém veio introduzir novas possibilidades ao nível da integração, escalabilidade, manutenção e análise de dados. Nestes contextos, a modernização de sistemas existentes surge como uma alternativa técnica e economicamente viável à substituição integral das infraestruturas, permitindo prolongar a vida útil dos equipamentos e alinhar o seu funcionamento com os requisitos atuais.

O sistema em estudo no presente trabalho pertence à Universidade de Coimbra e corresponde a um sistema automático de armazenamento e recuperação do tipo *miniload*, atualmente em operação e dedicado à gestão de contentores num arquivo automático. Este sistema foi desenvolvido e instalado entre os anos de 2003 e 2005, refletindo as tecnologias e práticas de automação industrial disponíveis na época. O contacto com este sistema surgiu no âmbito de um estágio curricular realizado pelo autor na Universidade de Coimbra, integrado num serviço responsável pela gestão da manutenção do referido sistema. Durante esse período, foi possível acompanhar o funcionamento diário do equipamento e identificar diversas limitações técnicas e funcionais, nomeadamente ao nível da arquitetura de controlo, das comunicações entre sistemas e da integração com soluções de software modernas.

As limitações observadas, associadas à antiguidade do sistema e à ocorrência frequente de erros operacionais e dificuldades de manutenção, evidenciaram a necessidade de

uma análise aprofundada e da definição de propostas de melhoria. Neste contexto, o desenvolvimento do presente trabalho de projeto surge como uma oportunidade para aplicar conhecimentos adquiridos ao longo do Mestrado em Engenharia Eletrotécnica, com especialização em Automação e Comunicações em Sistemas Industriais, a um caso real e relevante do ponto de vista industrial e académico. O projeto centra-se na avaliação do sistema existente, na identificação das suas principais fragilidades e na proposta de soluções que permitam melhorar a sua fiabilidade, flexibilidade e integração com tecnologias mais modernas comparadas com as tecnologias atuais do sistema.

O objetivo geral deste trabalho consiste em analisar e propor melhorias para o sistema automático de gestão e armazenamento da Universidade de Coimbra, tendo como foco a modernização da arquitetura de controlo e de comunicações, bem como a integração com soluções de software mais atuais. Como objetivos específicos, destacam-se: a caracterização detalhada do sistema AS/RS existente; a identificação das limitações técnicas e funcionais da arquitetura atual; a definição de uma arquitetura melhorada, alinhada com os princípios da automação industrial moderna; a implementação de melhorias ao nível do software de controlo e da comunicação entre sistemas; e a avaliação dos benefícios resultantes das soluções propostas.

O presente relatório encontra-se organizado da seguinte forma. No Capítulo 2 é apresentado o estado da arte, abordando a evolução histórica dos sistemas AS/RS, os principais tipos existentes, bem como as tecnologias de hardware, software e algoritmos utilizados na sua implementação. O Capítulo 3 descreve o desenvolvimento do projeto, incluindo a caracterização do sistema atual, o levantamento de requisitos e a definição das propostas de melhoria. No Capítulo 4 é detalhada a implementação prática das soluções propostas, incluindo as componentes de hardware, software e integração do sistema. Por fim, o Capítulo 5 apresenta as conclusões do trabalho e identifica possíveis desenvolvimentos futuros.

2 Estado da arte

2.1 História e Evolução AS/RS

Os sistemas de armazenamento e recuperação têm sido uma peça fundamental nas cadeias de abastecimento globais há quase 60 anos. A tecnologia que agora designamos por Sistema Automático de Armazenamento e Recuperação (ASRS) foi originalmente desenvolvido pelo antecessor da Demag, a Dmag. A sua base inovadora tem sido aplicada nos modernos armazéns de grande altura e controlado eficientemente o fluxo de materiais na indústria e na logística. Nos anos 1950, a maioria das cadeias de abastecimento dentro dos armazéns ou centros de distribuição operava manualmente e situava-se perto do solo. Uma frota de empilhadores e stackers transportava a maior parte da carga e utilizava tecnologia de transportadores. As peças e cargas mais pesadas tinham de ser armazenadas no chão, enquanto as limitações de altura dos empilhadores mantinham as prateleiras a uma altura reduzida, resultando em grandes armazéns de distribuição ocupando vastas áreas de terreno. Um espaço tão grande significava também que o inventário era armazenado da forma mais ampla possível, sem um controlo orientado pela procura no fluxo de materiais e na recolha de produtos. Os engenheiros da Demag, Friedhelm Podswyna, Horst-Werner Ruttkamp e Werner Kühn, obtiveram a ideia revolucionária de virar literalmente a prateleira de armazenamento ao contrário e fixá-la ao teto. Desta forma, o mastro móvel com o dispositivo de elevação rotativo poderia deslocar-se para cima e para baixo em cada corredor de estantes, acedendo aos itens armazenados. Esta estrutura poderia ser mais alta do que qualquer empilhador conseguiria alcançar, permitindo um armazenamento mais elevado e denso. A primeira máquina de armazenamento e recuperação entrou em operação em 1962 e foi instalada no armazém do Bertelsmann Book Club, em Gütersloh, na Alemanha. Embora fosse controlada manualmente a partir da cabine no mastro, já tinha alguma automação… era operada através de cartões perfurados. Para a Alemanha, o momento desta abordagem inovadora às operações da cadeia de abastecimento não poderia ter sido mais oportuno, pois permitiu responder aos desafios dos anos 1960, após o crescimento económico dos anos 1950. O armazenamento de alta densidade ajudou as empresas a enfrentar o aumento do consumo de energia, a subida dos preços da eletricidade e a escassez de espaço nos centros de distribuição urbanos e industriais. Além disso, a capacidade de automação ajudou a mitigar o aumento dos salários na região. O ASRS veio para ficar… Até aos anos 1970, o teto dos armazéns e os carris das estantes limitavam a expansão. No entanto, os avanços na engenharia e na informática

permitiram revolucionar novamente os centros de distribuição. As prateleiras foram baixadas até ao solo, e os mastros passaram a erguer-se a partir do chão, recolhendo os inventários com mais força e rapidez. Estas mudanças reduziram a oscilação das cargas em grandes alturas, permitindo uma seleção mais rápida, frequente e precisa entre vários corredores.

Na década de 1980, os dispositivos ASRS espalharam-se pelo continente africano e alcançaram a fama na América do Norte e em muitas outras regiões do mundo. O ritmo da inovação foi tão veloz quanto o próprio sistema e alcançou novos patamares. As instalações ASRS passaram a ser designadas por armazéns elevados, devido a terem permitido aumentar a altura do elevador até 45 metros. A tecnologia do sistema também evoluiu, incluindo máquinas AS/RS capazes de se deslocar de um corredor para outro. Ao longo das décadas de 1980 e 1990, não só como as tecnologias informáticas e os softwares de gestão de armazéns foram ampliados, como os sensores, ímanes e lasers foram introduzidos para medir distâncias e posições com uma precisão sem precedentes. Os sistemas de acionamento contínuo diminuíram o consumo de energia e novos equipamentos de manuseio de carga penetraram nas estantes, permitindo que diferentes sistemas de contentores e paletes servissem em novos e diferentes tipos de mercados.

No século XXI, de facto, o ASRS continua a evoluir. Hoje, a lista de processos automatizados está a ampliar, e logo mais indústrias necessitam de Sistemas Automáticos de Armazenamento e Recuperação.

2.2 Tipos de AS/RS

As tecnologias AS/RS podem ser classificadas de acordo com as suas características físicas, do tipo de carga manuseada, da arquitetura de movimentação e do nível de automatização. Cada tecnologia apresenta vantagens e limitações que a tornam mais adequada para determinados perfis de operação logística, como armazenagem de grande densidade, de elevado *throughput*, de rapidez no acesso, ou na eficiência energética.

De forma geral, os AS/RS distinguem-se pela configuração do equipamento de movimentação, podendo operar com paletes ou caixas, através de estruturas verticais ou horizontais e utilizando veículos dedicados ou robôs autónomos. A escolha da tecnologia depende de fatores como:

- dimensões e massa das cargas,
- espacialidade do armazém (altura útil e área disponível),
- volume de pedidos por hora,
- flexibilidade operacional e escalabilidade,
- nível de investimento pretendido.

De seguida descrevem-se os principais tipos de sistemas AS/RS usados na indústria, desde soluções para cargas unitárias pesadas até sistemas altamente modulares e robóticos, representativos do estado da arte atual na armazenagem automatizada [1].

2.2.1 Unit Load AS/RS

A tecnologia Unit Load AS/RS é uma Máquina de Armazenamento e Recuperação (SRM - *Storage and Retrieval Machine*) contemplada para o manuseio de cargas de maior dimensão que normalmente pesam acima de 180 kg.

Estas máquinas estão preparadas para trabalhar com produtos num sistema de estantes projetado, que podem atingir alturas de cerca de 36 metros ou mais e a sua capacidade de processamento por corredor pode depender de vários fatores. O seu dimensionamento é feito com base no número de localizações de armazenamento necessárias e no desempenho total necessário (taxa de armazenamento e recuperação). Este cálculo de taxa (estimado acima) pode ser frequentemente utilizado para determinar rapidamente o número de corredores (ou SRMs), com base no número de transações exigidas, de modo a tornar o sistema eficiente. Também é possível variar os *designs* dos sistemas, por exemplo, requerendo extratores duplos num único SRM para aumentar a capacidade global do sistema. Além disso, em algumas aplicações onde a capacidade de armazenamento é mais crítica do que o desempenho, pode ser aplicada uma estratégia de armazenamento em profundidade dupla, permitindo que cada localização no sistema de estantes acomode duas paletes [2][3].

Figura 2.1: Sistema Unit Load AS/RS para armazenamento e recuperação automática de paletes em estantes de grande altura[4].

2.2.2 Mini Load AS/RS

Os sistemas automatizados de armazenamento e recuperação Mini Load são concebidos para transportar automaticamente caixas, tabuleiros e contentores, que contenham um peso inferior a 160 Kg, para dentro e para fora do armazenamento. Estes sistemas são, geralmente, compostos por uma estrutura de estantes, gruas empilhadoras (também chamadas de gruas, máquinas de armazenamento e recuperação ou SRMs), software que controla o sistema, como o WCS (*Warehouse Control System*) e uma estação de recolha, onde os itens são introduzidos ou retirados do ASRS [2].

Para armazenar e recuperar mercadorias, a grua desloca-se ao longo de um corredor entre as estantes de armazenamento, enquanto um dispositivo ou carro de manuseio de carga se move para cima e para baixo ao longo do mastro alto da grua, alcançando a altura correta da prateleira. Assim que chega à localização de armazenamento especificada, o dispositivo de manuseio de carga da grua utiliza um mecanismo para recolher ou depositar a carga, conforme a tarefa a ser realizada. Depois de concluída, a grua retorna ao final do corredor, onde a carga é depositada para processamento adicional ou uma nova carga é recolhida [5][6].

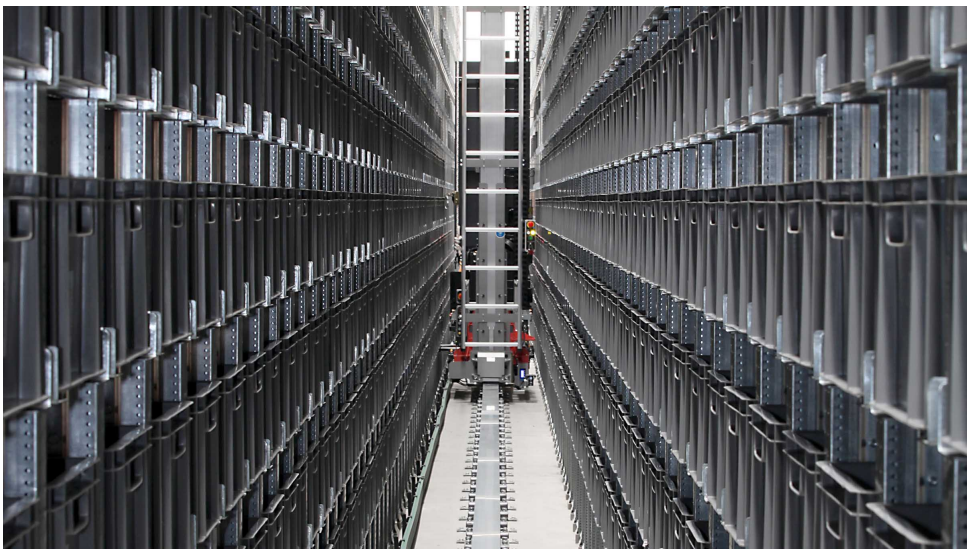


Figura 2.2: Sistema Mini Load AS/RS com grua de armazenamento e recuperação para caixas e contentores[7].

2.2.3 *Vertical Lift Module* (VLM)

Um Módulo de Elevação Vertical (VLM) é um sistema automatizado vertical de armazenamento e recuperação (AS/RS). É composto por tabuleiros dispostos na parte frontal e traseira e por um sistema de inserção/extração que se desloca pelo centro. Os tabuleiros são automaticamente recuperados e entregues numa janela de recolha para a operação de picking. Após a recolha, o sistema de inserção/extração armazena novamente o tabuleiro na posição designada ou atribui automaticamente uma nova posição,

com base na otimização do espaço ocupado ou na acessibilidade mais rápida/lenta (de acordo com as definições do utilizador). O design modular e a construção flexível do VLM permitem modificar e ajustar alturas, localizações e janelas de recolha a qualquer momento.

Os VLMs são ideais para maximizar o aproveitamento vertical do espaço até 85%, proporcionando acesso a centenas ou milhares de referências SKUs (*Stock Keeping Unit*). Os tabuleiros podem ser utilizados para armazenar caixas, contentores e itens de grande dimensão, ou podem ser subdivididos com divisórias e compartimentos para criar um grande número de posições de SKU dentro de um único tabuleiro. Tecnologias integradas, como pick-to-light, visão aumentada (realidade aumentada) e ponteiros laser, garantem uma precisão superior a 99,99% e uma elevada eficiência operacional.

Os VLMs estão equipados com controlos internos básicos e um sistema de gestão de inventário. Os tabuleiros podem ser armazenados com base na utilização, tamanho, peso ou outros parâmetros personalizados. O sistema pode ainda otimizar a densidade de armazenamento, medindo automaticamente cada tabuleiro e organizando-o para maximizar a capacidade de armazenamento.

Além disso, qualquer VLM ou conjunto de VLMs pode ser facilmente integrado em sistemas de software WMS (*Warehouse Management System*) e WES (*Warehouse Execution System*) para uma otimização total [8][2].

Figura 2.3: Vertical Lift Module (VLM) para armazenamento vertical de alta densidade com tabuleiros automáticos[9].

2.2.4 Carrossel Vertical

Os carrosséis verticais pertencem à família dos armazéns verticais devido à sua semelhante morfologia. Consistem em sistemas que possuem como base prateleiras rotativas organizadas verticalmente dentro de uma estrutura metálica fechada. Essas prateleiras são interligadas por correntes movidas por motores, permitindo a mobilidade dos produtos até a um único ponto de acesso ao operador, de modo a realizar a retirada dos produtos [10][11]. Os carrosséis verticais são projetados com o intuito de armazenar pequenos e médios produtos com um ritmo de procura moderada, reduzindo o tempo de deslocamento do operador e otimizando a separação de pedidos, diminuindo o esforço físico necessário para alcançar os produtos[12][13].

Figura 2.4: Carrossel vertical para armazenamento e picking de pequenas e médias unidades[14].

2.2.5 Carrossel Horizontal

Os carrosséis horizontais são sistemas automatizados, de armazenamento e recuperação, constituídos por uma série de prateleiras ou contentores montados sobre uma estrutura em forma de pista oval ou elíptica, que se desloca horizontalmente em torno

do operador. Ao contrário dos carrosséis verticais, que exploram principalmente a altura do armazém, os carrosséis horizontais tiram partido da área em planta.[15]

O princípio de funcionamento baseia-se no conceito goods-to-person: em vez de o operador se deslocar ao longo das estantes, é o carrossel que roda de forma a trazer o contentor pretendido para uma estação de picking fixa. Esta abordagem reduz significativamente os tempos de deslocação e o esforço físico do operador, que deste modo aumenta a produtividade e diminuindo o risco de erros na preparação de encomendas. Tipicamente, vários carrosséis horizontais podem ser agrupados e operados em conjunto (pods), permitindo que, enquanto um carrossel se posiciona, o operador trabalha noutro, o que otimiza o tempo de ciclo global.[16]

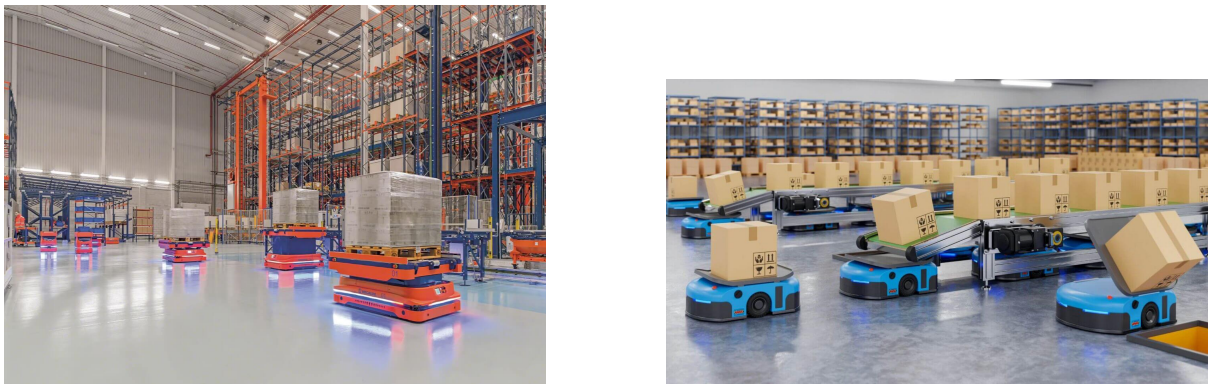
Este tipo de tecnologia é particularmente indicado para armazenar peças de pequenas e médias dimensões com elevada rotatividade, como componentes eletrónicos, peças de reposição, material de escritório ou consumíveis. Quando integrados com sistemas de gestão de armazém (WMS) e tecnologias de apoio ao picking (como pick-to-light), os carrosséis horizontais conseguem garantir um elevado nível de precisão e seguimento, sendo uma solução competitiva para operações que exigem rapidez, ergonomia e boa utilização do espaço disponível.[17]

Figura 2.5: Carrossel horizontal para operar com o princípio *goods-to-person*[18].

2.2.6 Robôs AMR/AGV para AS/RS

Nos últimos anos, a utilização de veículos autónomos móveis (AGVs e AMRs) tem-se tornado uma extensão natural dos sistemas AS/RS, oferecendo maior flexibilidade e adaptabilidade. De acordo com Ghelichi & Kilaru (2021), soluções baseadas em AMRs, como os modelos "Last-Mile Delivery" e "Meet-In-Aisle", podem ser integradas para complementar o desempenho dos sistemas *miniload*, reduzindo o tempo de ciclo, a dependência de operadores humanos e a necessidade de infraestruturas fixas. Os AMRs destacam-se pela sua capacidade de navegação autónoma, permitindo reconfiguração

dinâmica de processos logísticos, sendo uma alternativa interessante para armazéns de alta rotatividade e com requisitos de escalabilidade[19][20].



(a) (b)

Figura 2.6: Robôs móveis autónomos utilizados em sistemas AS/RS para transporte automático de caixas e contentores. (a) Robot AMR[21] ; (b) Robot AVG[22].

2.2.7 Cube Storage (Ex.: AutoStore)

Ao considerar um sistema Unit Load (ou qualquer tecnologia AS/RS), o sistema automatizado incluirá sempre um meio de entrega e remoção de produtos do sistema Unit Load AS/RS. Estes "sub-sistemas de entrada e saída" podem ser concebidos de forma mais manual, utilizando estações P&D (Pick & Drop), ou como um sistema totalmente automatizado de manejo de caixas/contentores (ou seja, transportadores, carros-shuttle, veículos eletrificados, AMR, AGV, etc.)[19][20].

Figura 2.7: Sistema Cube Storage para armazenamento compacto de contentores[23].

2.3 Componentes e Tecnologias

O sistema *miniload* ou AS/RS é demasiado completo e complexo, pois integra diversos componentes, tanto a nível hardware como software.

2.3.1 Componentes de Hardware: PLCs, Sensores e Atuadores

O AS/RS necessita de um corpo, como tronco, membros e cabeça. Para tal, necessita de equipamentos e componentes para sentir e atuar.

2.3.1.1 PLCs

Os Controladores Lógicos Programáveis (PLCs) são a base do controlo na automação industrial e possuí como função de implementar a lógica de comando, de processar sinais de sensores, de comandar atuadores e assegurar que o sistema funciona de forma segura, fiável e eficiente.

Os PLC's ao longo das sua gerações têm sofrido uma evolução bastante notável, não só, a nível de hardware, mas também acessível a novos protocolos de comunicações[24].

Usando um exemplo de melhorias entre o Beckhoff CX1000 e o Beckhoff CX7000/CX52xx

Geração Antiga: Beckhoff CX1000

- Baseado em processador x86 a 266 MHz, com memória limitada.
- Programação via TwinCAT 2 (IEC 61131-3).
- Suporte a protocolos como Profibus, DeviceNet e Modbus (EtherCAT disponível apenas com módulos adicionais).
- Limitações: capacidade de processamento reduzida, menor integração com tecnologias modernas (IoT, cloud), tempos de varrimento (scan time) relativamente elevados[25].

Geração Moderna: Beckhoff CX7000 / CX52xx

- Processadores ARM Cortex-A53 (CX7000) ou Intel Atom/i7 (CX52xx), com desempenho muito superior.
- Suporte nativo a TwinCAT 3, permitindo integração com IEC 61131-3, C/C++ e MATLAB/Simulink.
- EtherCAT incluído de série, assegurando comunicações determinísticas em tempo real.
- Interfaces modernas (Ethernet, USB, DVI) e suporte nativo a MQTT e OPC UA para IoT.
- Vantagens: tempos de ciclo reduzidos (<1 ms), maior escalabilidade, integração com manutenção preditiva e análise em nuvem[26].

Alternativas relevantes:

- Siemens S7-1500: elevado desempenho, integração via TIA Portal, protocolos Profinet/Profibus, diagnóstico avançado e segurança cibernética integrada[27].
- Allen-Bradley CompactLogix 5380: suporte a EtherNet/IP, indicado para sistemas críticos, integração natural com software Rockwell[28].
- Schneider Modicon M262: preparado para IoT, suporte nativo a MQTT e OPC UA, integração com plataformas cloud (AWS, Azure)[29].

2.3.1.2 Sensores

Os sensores são os "olhos"do sistema, fornecendo a informação necessária para a tomada de decisão.No caso do *miniload*, são essenciais para:

- Determinar posição do equipamento nos carris;
- Confirmar a presença e o estado dos contentores;
- Garantir a segurança durante a operação[30].

Tipos de sensores utilizado:

- Sensores indutivos: detetam a presença de metais, usados em posições de referência;
- Sensores fotoelétricos: para deteção de objetos ou posicionamento mais preciso.
- Enconders: acoplados aos motores, permitem medir deslocamentos e calcular a posição exata do equipamento.
- Lazer: utilizados para medição precisa de distância, deteção de posição e alinhamento do equipamento. Os sensores laser podem ainda ser usados para deteção de obstáculos e controlo de aproximação[30].

Evolução tecnológica:

- Sistemas antigos recorrem a sensores analógicos ou digitais simples, com ligações ponto a ponto ao PLC[31].
- A geração atual introduz sensores inteligentes com IO-Link, permitindo:
  - Diagnóstico remoto.
  - Parametrização dinâmica.
  - Monitorização de falhas e manutenção preditiva[32][33].

2.3.1.3 Atuadores e Motores

O movimento do *miniload* é assegurado por motores e sistemas de atuadores, responsáveis pela translação, elevação e pela mudança de ala através das agulhetas de carril.

Tecnologia utilizadas:

- **Motores de passo:** comuns em sistemas mais antigos, simples mas menos eficientes em cargas elevadas.
- **Servomotores:** oferecem maior precisão, velocidade e controlo em malha fechada.
- **Inversores de frequência e servodrives:** permitem ajustar a velocidade e o binário em tempo real, aumentando a eficiência energética.
- **motores com VDF:** FALTA motores VEV variadores de velocidade[34]

2.3.2 Protocolos de Comunicação

A Comunicação entre o PLC, os sensores e os atuadores é essencial para a sincronização do sistema.

Protocolos tradicionais:

- **Profibus e DeviceNet:** são muito utilizados em sistemas antigos, mas com limitações de velocidade e de diagnóstico.
- **Modbus:** é simples e universal, mas pouco adquado para controlo em tempo real.

Protocolos modernos:

- **EtherCAT:** é um protocolo de altíssima velocidade, determinístico e ideal para sistemas de tempo real
- **Profinet:** padrão Siemens, integra bem com SCADA e redes insdustriais.
- EtherNet/IP: é bastante comum em sistemas Allen-Bradley e suporta integração IoT
- **CANopen:** usado em sistemas móveis ou compactos.

2.3.3 Tecnologias de Supervisão e Monitoramento: SCADA

Um sistema de supervisão SCADA (supervisory Control and Data Acquisition) permite:

- Monitorizar variáveis do sistemas (posição, alarmes, estados de metores e atuadores)
- Registar históricos de operação.
- Criar interfaces gráficas (HMI) para os operadores[35].

Exemplos de soluções SCADA:

- **Beckhoff TwinCAT HMI:** integração nativa com PLCs Beckhoff.
- **Siemens WinCC:** é uma plataforma associada ao TIA Portal.

- **Ignition SCADA:** solução flexível, multiplataforma, compatível com OPC UA e MQTT[36].

2.3.3.1 IoT Industrial (IIoT)

A Internet das Coisas Industrial (IIoT) representa a nova tendência na automação. Ligar dispositivos industriais à nuvem ou a plataformas de análise com vista a otimizar operações[37]. **Tecnologias relevantes:**

- **MQTT:**protocolo leve para comunicação de dados em tempo real.
- **OPC UA:** garante interoperabilidade entre plataformas podem recolher dados para análise preditiva.
- **Integração em Cloud:** AWS, Azure ou outras plataformas podem recolher dados para análise preditiva[38].

Aplicações práticas:

- Monitorização remota do *miniload*
- Alarmística em tempo real para manutenção
- Dashboards de desempenho e análise de eficiência[39].

2.4 Softwares

O desempenho de um sistema automático de armazenagem (AS/RS – Automated Storage and Retrieval System) não depende apenas do hardware (*miniload*, carris, sensores e PLCs), mas também de um software capaz de gerir de forma eficiente os fluxos de materiais e a comunicação com o operador ou outros sistemas de nível superior. Estes softwares, normalmente designados por WMS (Warehouse Management System) ou WCS (Warehouse Control System), asseguram a integração entre os pedidos logísticos e a execução no nível físico do armazém[40].

Sistemas Tradicionais:

- Baseados em aplicações dedicadas e fechadas, desenvolvidas à medida para cada instalação.
- Interação limitada com o utilizador, normalmente através de terminais fixos.
- Comunicação com PLCs por meio de protocolos simples (Modbus, Profibus).
- Pouca flexibilidade para integração com sistemas externos[41].

Sistemas Modernos:

- Baseados em arquiteturas *opensource* e modulares.

- Separação clara entre WMS (nível estratégico) e WCS (nível operacional):
  - WMS (Warehouse Management System): define onde armazenar e quando retirar cada contentor, integrado com ERP.
  - WCS (Warehouse Control System): traduz ordens em comandos para o miniload, otimizando movimentos em tempo real[42].
- Interfaces web e móveis para supervisão e controlo.
- Utilização de protocolos modernos (OPC UA, MQTT, Web Services, APIs REST).
- Integração com IoT e Big Data, permitindo monitorização de KPIs e manutenção preditiva[43].

2.5 Algoritmos

Um sistema de armazenamento automático (AS/RS) depende não apenas do seu desempenho mecânico, mas também da forma como é feita a gestão das tarefas de recolha e armazenamento. Os algoritmos de otimização têm a função de minimizar o tempo de ciclo, reduzir os movimentos desnecessários do *moniload*, equilibrar a carga de esforço entre alas ou zonas e aumentar o *throughput* total do sistema. Nos últimos anos, identificaram-se várias abordagens algorítmicas que foram aplicadas à otimização de AS/RS, variando entre métodos determinísticos e heurísticos, de acordo com a complexidade e o tamanho do problema. Seguem-se os principais tipos de algoritmos utilizados e suas características[44].

2.5.1 Algoritmos Baseados em Regras (Rule-Based Algorithms)

São os métodos mais simples e tradicionalmente utilizados em sistemas industriais. Baseiam-se em regras pré-definidas, criadas com base na experiência operacional ou em políticas [45].

Algumas dessas regras poderiam ser tais como:

- *"O contentor mais próximo é servido primeiro"*, de modo a minimizar a distância percorrida.
- *"Os pedidos urgentes têm prioridade sobre os restantes."*
- *"Contentores mais utilizados são armazenados em posições de fácil acesso."*[17]

Estes métodos possuem uma maior facilidade de implementação e baixa exigência computacional, além de se tornarem uma boa resposta em sistemas de pequena dimensão ou com padrões de operação previsíveis. Todavia, apresenta falta de flexibilidade e dificuldade de adaptação a situações dinâmicas, e não garante a melhor solução quando há muitos pedidos concorrentes[46].

2.5.2 Algoritmos de Programação Linear e Inteira (Optimization via Mathematical Programming)

São métodos baseados em modelos matemáticos que procuram minimizar uma função de custo(tempo, energia, distância) sujeita a restrições (capacidade de estantes, limites de movimento, prioridades de pedidos).

Possui vantagens, tais como:

- quando o modelo é bem desenvolvido, produz soluções mais otimizadas.
- Boa base para comparar com os métodos heurísticas.

Enquanto como desvantagens, possui:

- Exige-se elevada capacidade computacional.
- Dificuldade de adaptação a ambientes dinâmicos com múltiplos pedidos em simultâneos.

Ferramentas típicas: Gurobi, CPLEX, MATLAB Optimization Toolbox[46][17].

2.5.3 Algoritmos de Otimização por Heurísticas

Heurísticas procuram soluções "boas o suficiente" num tempo reduzido, sendo mais práticas para sistemas complexos ou em tempo real.

Algoritmo do Vizinho Mais Próximo (Nearest Neighbour):

- Seleciona o próximo pedido a servir com base na menor distância ao ponto atual do *miniload*.
- Simples, rápido e eficaz em cenários de operação contínua.
- Pode ser estendido para percursos de múltiplos *miniloads* com balanceamento de carga[45][44].

Algoritmos de Agrupamento (Clustering):

- Agrupam pedidos por zonas ou níveis de estantes, reduzindo deslocações longas.
- Frequentemente baseados em k-means ou métodos hierárquicos.
- Melhoram a eficiência em armazéns com múltiplas alas[47][44].

2.5.4 Algoritmos Metaheurísticos

São métodos inspirados em processos naturais (biológicos, físicos ou sociais) e amplamente usados em sistemas logísticos complexos. Permitem otimização global em problemas com múltiplas variáveis e restrições.

2.5.4.1 Algoritmo Genético (Genetic Algorithm – GA)

- Baseia-se em princípios da seleção natural.
- Gera uma população de soluções (sequências de pedidos) e combina as melhores, introduzindo mutações para explorar novas possibilidades.
- Frequentemente usado para determinar a sequência ótima de recolha/armazenamento ou o layout ideal das estantes.
- Vantagem: elevada capacidade de adaptação;
- Desvantagem: pode exigir ajuste fino dos parâmetros[48].

2.5.4.2 Algoritmo de Colónia de Formigas (Ant Colony Optimization – ACO)

- Inspirado no comportamento das formigas na procura de caminhos curtos.
- Cada "formiga" representa uma solução possível; as melhores reforçam o caminho através de "feromonas digitais".
- Muito usado para problemas de roteamento e sequenciamento de pedidos.
- Excelente desempenho em sistemas com múltiplos *miniloads* a operar em paralelo[49].

2.5.4.3 Particle Swarm Optimization (PSO)

- Baseado no comportamento coletivo de bandos de aves ou cardumes.
- As soluções (partículas) movem-se no espaço de busca influenciadas pelas melhores soluções conhecidas.
- Aplicável a ajuste de parâmetros de controlo, planeamento de trajetórias e distribuição de tarefas[48].

2.5.5 Métodos Híbridos

Estes métodos híbridos são os mais promissores em ambientes reais, pois conciliam desempenho computacional com capacidade de adaptação a mudanças dinâmicas nos pedidos. Combinam as vantagens de diferentes algoritmos. Por exemplo:

- GA + ACO para combinar busca global (GA) com refinamento local (ACO).
- Heurística + Programação Linear para aplicar regras rápidas e depois refinar a solução[50].

O uso de algoritmos de otimização em sistemas AS/RS é essencial para elevar o desempenho e reduzir custos operacionais. Para cada tipo de sistema e do tamanho do volume de pedidos, podem ser aplicadas abordagens determinísticas, heurísticas ou metaheurísticas, sendo as soluções híbridas e baseadas em intelegência artifical as mais

propíssias para ambientes dinâmicos e de elevada complixidade, como os armazéns multialas controladas por *miniloads*[49].

2.6 Conclusão

Neste Capítulo, apresenta-se o estado da arte de forma geral sobre os diversos tipos de sistemas de armazenamento, abordando, em particular, a sua história, evolução e os diferentes tipos de sistemas existentes. Seguiu-se uma análise dos componentes e das tecnologias envolvidas, com enfoque específico no sistema em estudo. Por fim, procedeu-se à análise das arquiteturas de software e dos algoritmos de decisão relacionados com o posicionamento dos produtos (no presente caso acervos).

3 Desenvolvimento do projeto

O sistema AS/RS do Arquivo Automático, pertencente à Universidade de Coimbra, é do tipo *miniload* e foi projetado e implementado entre os anos de 2003 e 2005. Este sistema foi concebido com o objetivo de automatizar o armazenamento e a recuperação de contentores, com o objetivo de garantir maior eficiência, segurança e rastreabilidade no acesso aos documentos arquivados.

O sistema encontra-se dividido em dois espaços fisicamente distintos, como ilustra na Figura 3.1: o armazém automático e a sala do operador.

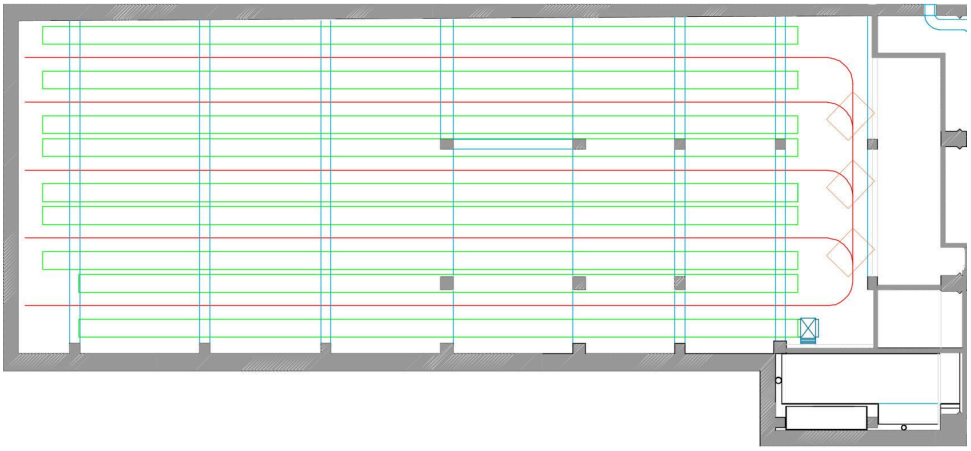


Figura 3.1: Planta do Arquivo Automático

No armazém estão localizadas as estruturas de armazenamento (racks), como demonstra na Figura 3.2a, os carris de deslocação e o equipamento *miniload* responsável pela movimentação dos contentores demonstrados na Figura 3.2b. A sala do operador concentra o quadro elétrico de alimentação, os sistemas de comando das plataformas de troca de carril e o computador responsável pela interface entre o operador e o sistema automático.

(a) (b)

Figura 3.2: Estrutura das Racks (a) Racks vazias; (b) Racks com contentores.

3.1 Descrição do Sistema Atual

Nesta secção é apresentada uma descrição detalhada do sistema atualmente em funcionamento, abordando tanto a componente física do armazém como a infraestrutura informática e de software que assegura o controlo e a gestão do sistema AS/RS.

3.1.1 Armazém físico

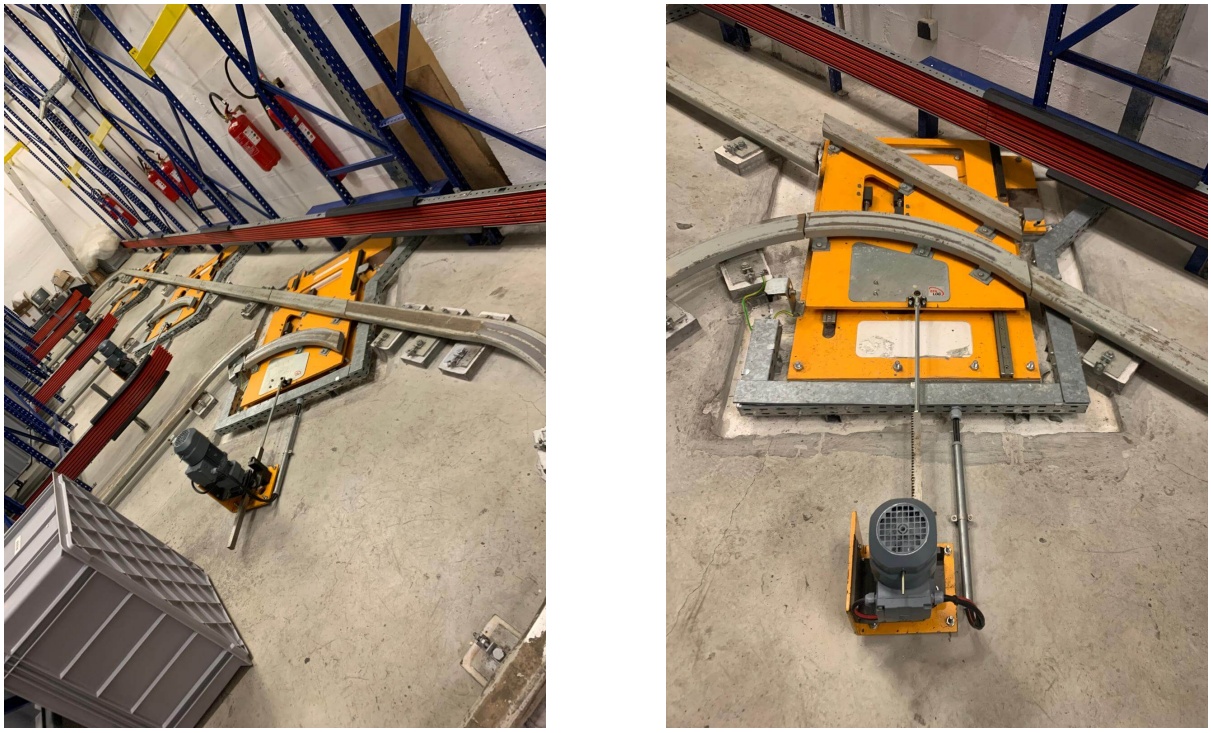
O armazém automático é composto por nove fileiras de racks, sendo que cada fileira possui oito níveis em altura e um número variável de posições na horizontal, entre 38 e 44 racks, dependendo da fileira. A disposição das racks foi projetada de forma a criar cinco corredores, nos quais o *miniload* se desloca para executar as operações de armazenamento e recuperação.

Do primeiro ao quarto corredor existem duas fileiras de racks dedicadas por corredor, enquanto o quinto corredor possui apenas uma fileira, o que reflete uma configuração assimétrica do armazém, resultante das limitações físicas do espaço disponível, como se pode observar na Figura 3.1.

A Figura 3.3a, representa a lateral das racks onde existe um sistema de carris que permite a circulação do *miniload* entre corredores. Este sistema é constituído por três plataformas de agulhetas, cada uma equipada com um fragmento de carril reto e um

fragmento curvo, como se pode observar com maior pormenor na Figura 3.3b. A posição do fragmento de carril determina o percurso do *miniload*:

- quando o fragmento curva está alinhado com um corredor, o *miniload* entra nesse corredor;
- quando é o fragmento reto permite continuidade longitudinal, o *miniload* avança para o corredor seguinte.



(a) (b)

Figura 3.3: Estrutura das Agulhetas. (a) As três Agulhetas; (b) Agulheta isolado.

Cada plataforma é acionada por um motor elétrico, responsável pela troca entre o segmento reto e o segmento curvo. Em cada extremo da plataforma existem sensores indutivos, que garantem a confirmação da posição correta antes de permitir a circulação do *miniload*. Estes motores e sensores são controlados pelo PLC localizado no quadro elétrico da sala do operador, ilustrado na seguite Figura 3.4.

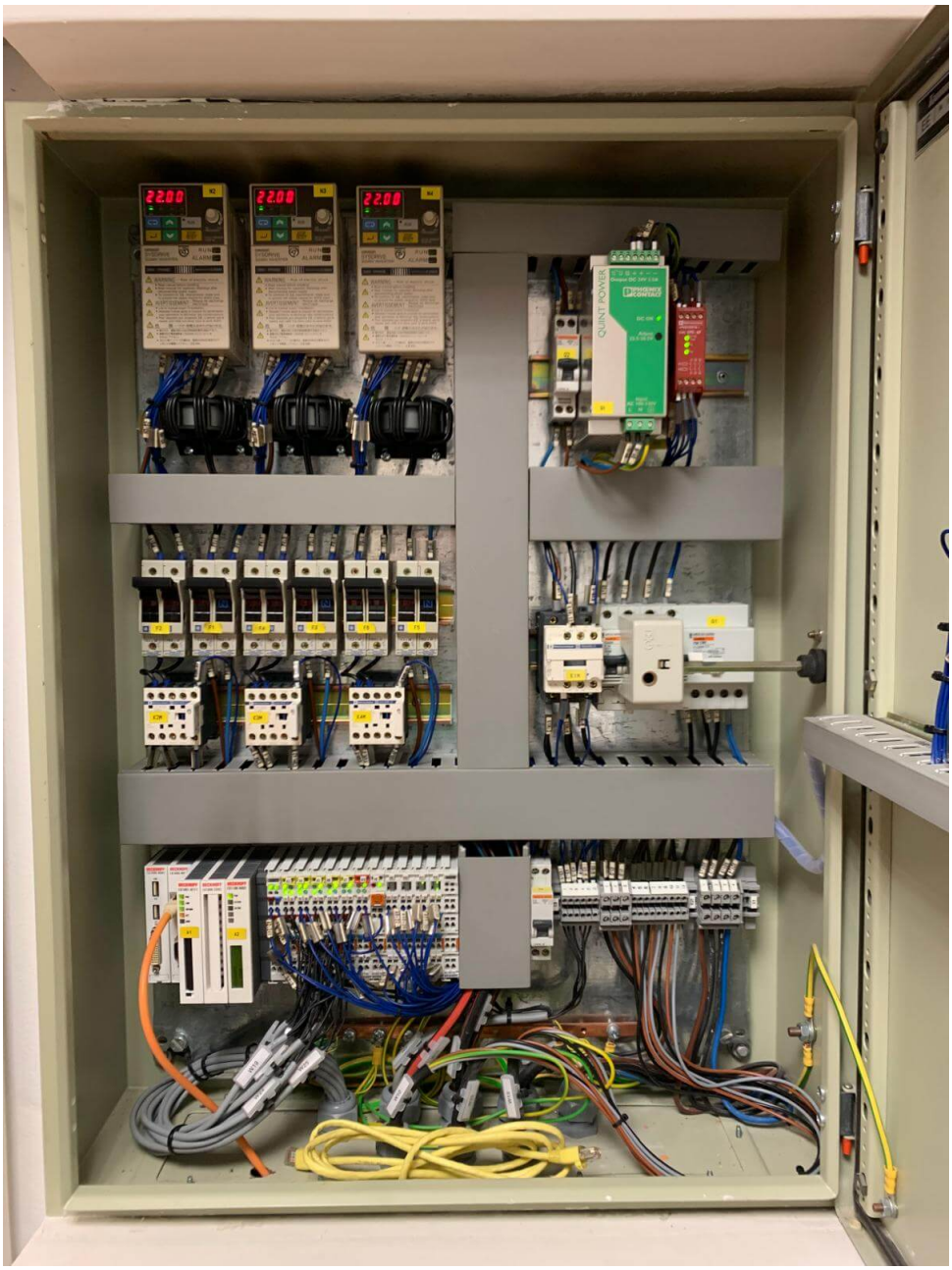
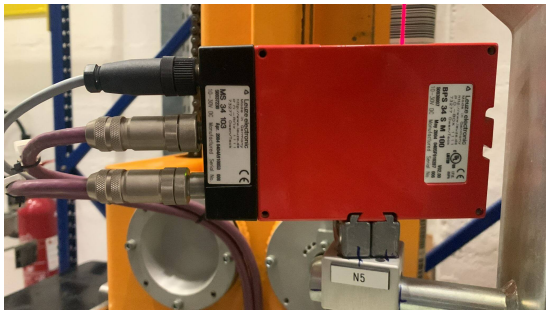


Figura 3.4: Quadro elétrico da sala de operador, onde se destaca o PLC Beckhoff.

O *miniload* possui uma estrutura metálica robusta, projetada para suportar todos os componentes mecânicos, elétricos e eletrónicos necessários ao seu funcionamento, como se pode observar na Figura 3.5. O deslocamento horizontal (eixo X) é realizado através de dois motores elétricos, permitindo a movimentação ao longo dos carris dos corredores. A posição horizontal do *miniload* é determinada através de um sensor laser, ilustrado na Figura 3.6b, que fornece informação precisa sobre a sua localização.

Figura 3.5: Estrutura do *miniload*.

O movimento vertical (eixo Y) é assegurado por um motor elétrico dedicado, responsável pela elevação e descida do conjunto de garfos. Para a referência da posição vertical, o sistema utiliza um leitor de código de barras, sendo os códigos distribuídos ao longo da torre vertical do *miniload*, como se pode observar na Figura 3.6a.

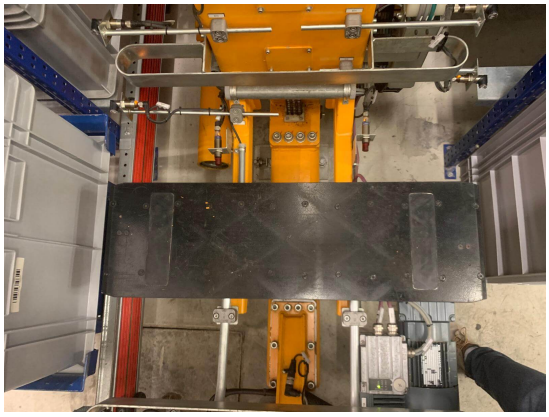


(a)

(b)

Figura 3.6: Sensores. (a) Sensor lazer leitor de código de barras; (b) Sensor lazer.

A operação de recolha e deposição dos contentores é realizada através de garfos extensíveis como ilustrado na Figura 3.7a, que incorporam uma mesa móvel acionada por um motor elétrico equipado com encoder ilustrado na Figura 3.7b, garantindo um controlo preciso do posicionamento durante as operações de carga e descarga.



(a)

(b)

Figura 3.7: Elesmentos dos garfos. (a) Base dos garfos; (b) Motor com enconder.

3.1.2 Desenvolvimento de Esquema elétrico

Foi realizado o levantamento dos dispositivos e equipamentos elétrico e eletromecânicos de todo o sistema, que se encontra disponível entre o Anexo A e Anexo B e os dispositivos listados nas tabelas do Anexo C ao Anexo E. Devido à grande dimensão e complexidade de todo o sistema, foi só possível realizar o levantamento do esquema elétrico de potência e comando, pertencente ao quadro elétrico da sala de operador. Este quadro elétrico tem a função de alimentar todo o sistema, realizar o comando das três agulhetas e fornecer segurança ao sistema, que engloba betoneiras de emergência e funções como permissão ao operador de acesso à entrada dentro do armazém. O esquema elétrico, que se encontra disponível entre o Anexo F e o Anexo Q , foi desen-

volvido no software AutoCAD Eletrical da Autodesk, devido ao Instituto Superior de Engenharia de Coimbra possuir licença. Para este levantamento ocorrer com a maior segurança, foi necessário desligar a energia a partir do quadro elétrico utilizando um multímetro para despistar e verificar as ligações entre os equipamentos. Como se pode ver, na Figura 3.4, os equipamentos e a cablagem possuíam já alguma identificação, o que facilitou a realização da tarefa.

3.1.3 Computador - Software

O computador localizado na sala do operador constitui o núcleo central da infraestrutura informática do sistema AS/RS. É neste equipamento que se encontram os softwares responsáveis pelo controlo do PLC, pela comunicação entre sistemas e pela interface com o operador, assegurando a integração entre o nível físico do armazém e o nível lógico de gestão.

3.1.3.1 CONSOVEYOR WMS - Software de gestão

O CONSOVEYOR WMS é o software responsável pelo nível de supervisão e planeamento do armazém automático. Através deste sistema, o operador pode criar pedidos de armazenamento e recolha de contentores, acompanhar o estado geral do sistema e consultar informação operacional relevante.

Para além disso, o operador consegue aceder à base de dados para obter informação sobre a posição dos contentores, os produtos armazenados e o respetivo estado operacional. O sistema permite ainda a gestão de contas de utilizador, assegurando diferentes níveis de acesso e controlo conforme o perfil do operador.

Adicionalmente, através da sua aplicação SCADA, o CONSOVEYOR WMS possibilita o envio de comandos isolados para as agulhetas, permitindo intervenções específicas no sistema sempre que necessário, seja para fins de operação, de teste ou de manutenção.

Este software comunica com o Kepware OPC Server para enviar instruções ao PLC Beckhoff, garantindo a execução das ordens no nível físico do sistema. Em paralelo, utiliza a base de dados Oracle para registar toda a informação relevante, incluindo ordens executadas, posições dos contentores, históricos de operação e alarmes. Desta forma, o WMS fornece uma interface centralizada e integrada para a gestão, supervisão e controlo do sistema AS/RS.

3.1.3.2 Oracle software

O sistema utiliza uma base de dados Oracle para o armazenamento e consulta de informação relacional. Através do Oracle Client, os softwares WMS e SCADA comunicam com a base de dados onde são registados os pedidos, as posições dos contentores, os históricos de operação, os alarmes e os logs do sistema.

O Oracle atua como intermediário entre o nível de controlo (PLC/OPC) e o nível de gestão logística. O operador submete pedidos através do WMS ou SCADA, o sistema consulta a base de dados para obter a informação necessária e, posteriormente, envia os comandos adequados ao PLC através do OPC.

3.1.3.3 OPC SERVER - KEPWARE OPC

O KEPServerEX (Kepware OPC Server) é utilizado como servidor OPC universal, funcionando como middleware de comunicação industrial. Neste sistema, o Kepware estabelece a ponte entre o PLC Beckhoff (via TwinCAT OPC Server) e os softwares de gestão e monitorização, nomeadamente o CONSOVEYOR WMS.

O Kepware permite integrar equipamentos de diferentes fabricantes, garantindo a interoperabilidade entre sistemas heterogéneos. Inclui ainda ferramentas como o OPC Quick Client, que possibilitam a leitura e escrita de variáveis em tempo real para efeitos de teste, diagnóstico e validação das comunicações.

3.1.3.4 TwiCAT OPC

Para permitir a comunicação com sistemas externos, é utilizado o TwinCAT OPC Server, que disponibiliza uma interface baseada no padrão industrial OPC. Este servidor converte as variáveis internas do PLC (tags, estados e valores de entradas e saídas) para um formato universal, permitindo que aplicações de nível superior, como sistemas WMS ou SCADA, possam aceder aos dados do PLC de forma normalizada.

Desta forma, o TwinCAT OPC Server atua como um tradutor entre o ambiente Beckhoff e os softwares de gestão, possibilitando, por exemplo, que uma variável como "Contentor_Posicao_X" seja lida ou escrita remotamente por aplicações externas.

3.1.3.5 TwinCAT 2

O sistema utiliza o TwinCAT 2 como ambiente de desenvolvimento e execução do controlo do PLC Beckoff. A sua função é:

- o controlo em tempo real de todo o sistema físico do AS/RS, através da execução da lógica programada no PLC;
- a monitorização contínua dos sinais de entrada e saída digitais e analógicos, tais como sensores de posição, estados de motores e sinalizadores;
- a atuação como elo de ligação entre o hardware físico (PLC e módulos de I/O) e os níveis superiores de software.

A comunicação entre o computador do operador e os PLCs é realizada via Ethernet (TCP/IP), recorrendo ao protocolo ADS (*Automation Device Specification*), que é o protocolo nativo dos sistemas Beckhoff.

São estes os elementos que constitui a arquitetura atual, como se pode verificar no diagrama da Figura 3.8.

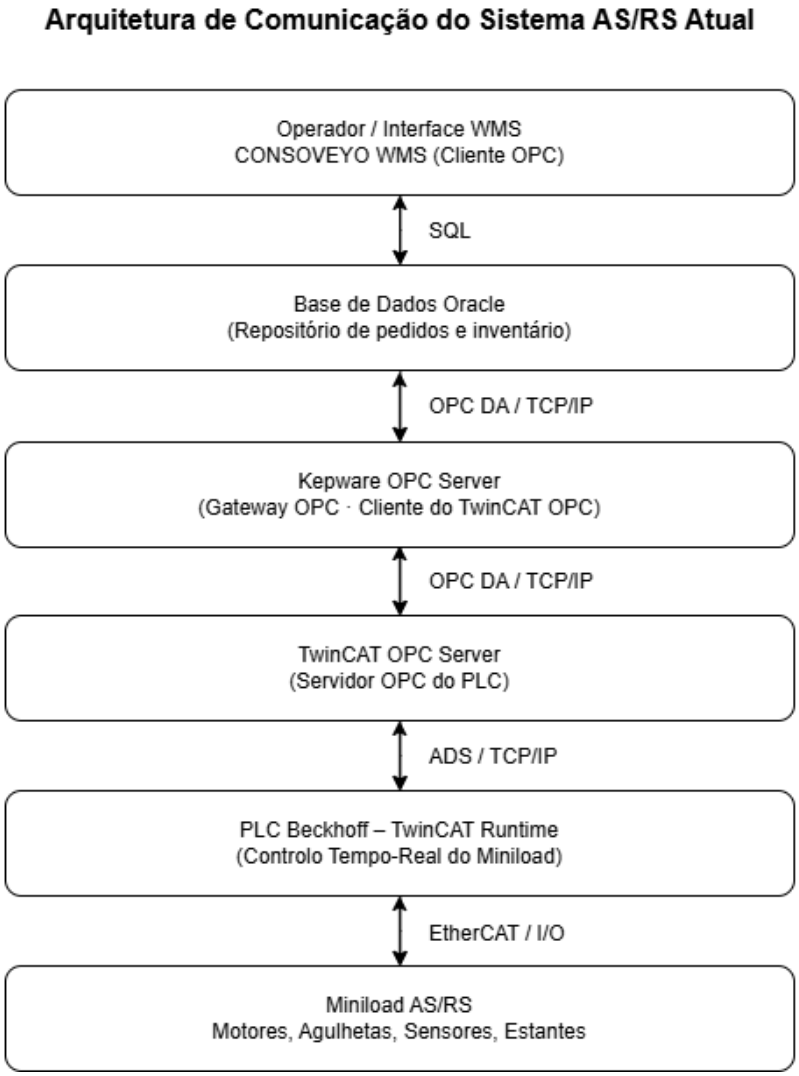


Figura 3.8: Arquitetura Atual do sistema AS/RS

3.1.4 Comunicações de TwinCAT do PC para PLC

Uma vez que os PLCs existentes se encontram em bom estado de funcionamento e apresentam fiabilidade comprovada, a proposta de melhoria do sistema não passa pela sua substituição, mas sim pela modernização dos níveis superiores de software, nomeadamente o WCS/WMS e os sistemas de supervisão. Neste contexto, tornou-se essencial analisar em detalhe o mecanismo de comunicação entre o computador do operador e os PLCs, de modo a compreender o funcionamento atual, identificar limitações e avaliar possibilidades de otimização.

Para este efeito, foi realizado um estudo aprofundado das comunicações de rede recorrendo à ferramenta Wireshark, que permite capturar e analisar pacotes de comunicação

em tempo real, como demonstra na Figura 3.9. Esta análise incidiu sobre o tráfego trocado entre o PC do operador e os PLCs Beckhoff responsáveis pelo controlo do *miniload* e do armazém automático.

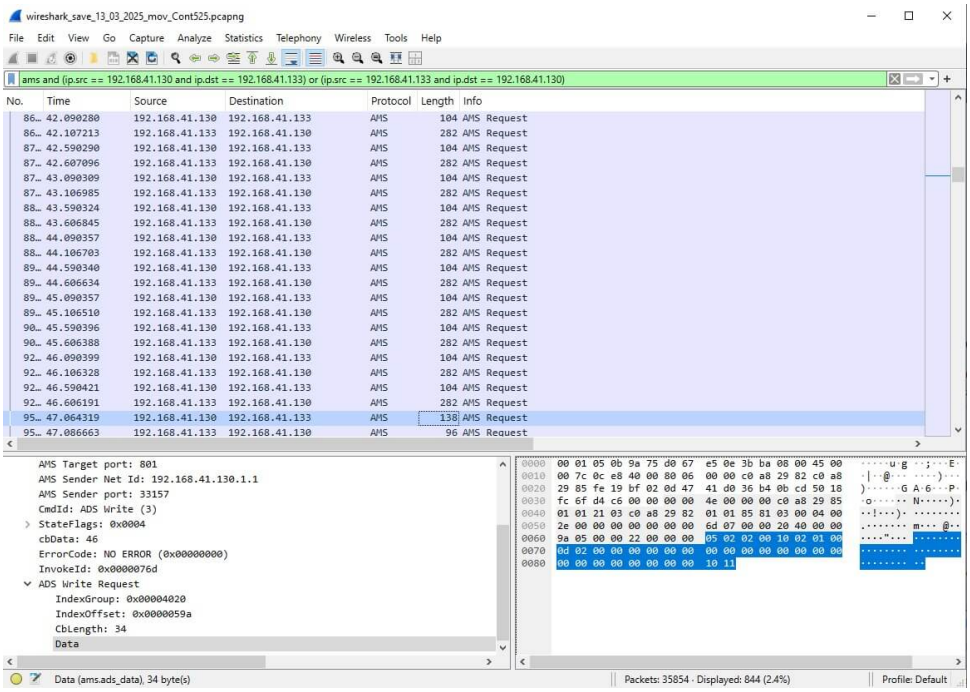


Figura 3.9: Captura e análise de pacotes AMS/ADS no Wireshark, ilustrando um pedido ADS Write Request trocado entre o PC do operador e o PLC Beckhoff durante a operação do *miniload*.

3.1.4.1 Protocolo AMS/ADS

Através da captura de pacotes foi possível verificar que a comunicação entre o PC e os PLCs é realizada através dos protocolos AMS/ADS, nativos do ecossistema Beckhoff e utilizados pelo ambiente TwinCAT 2. Estes protocolos são responsáveis pela troca de dados, leitura e escrita de variáveis, envio de comandos e monitorização do estado do sistema.

Esta tecnologia é responsável por interligar todos os componentes TwinCAT, permitindo que aplicações externas comuniquem diretamente com o runtime PLC. A grande vantagem deste modelo é que toda a comunicação entre o programa PLC e PC é abstrata, orientada a serviços, independentes do meio físico e opera igualmente sobre: TCP/IP, ligações séries, Fieldbus e ligações remotas (Internet, VPN, etc.). A função do AMS Router é o núcleo da comunicação Beckhoff, pois é ele que:

- gere todos os endereços TwinCAT (AMS Net IDs);
- encaminha mensagens entre aplicações locais e remotas;
- mantém tabelas de ligação;
- assegura entrega, temporização e prioridades.

Cada dispositivo TwinCAT, seja um PC, um PLC CX ou um módulo embutido, possui um endereço lógico único identificado por **AMS Net ID**, composto por 6 bytes (exemplo: 192.168.10.10.1.1). Em que este identificador é independente do IP e permite ao sistema distinguir unicamente cada nó no TwinCAT. O ADS funciona como um serviço de mensagens estruturado sobre o AMS. Quando um programa no PC quer ler uma variável do PLC, escrever um comando, iniciar/parar um runtime ou transferir dados estruturados, envia uma mensagem ADS para o endereço AMS correspondente ao PLC. Cada pedido de ADS contém:

- AMS Net ID destino;
- Index Group (grupo de memória ou serviço a aceder);
- Index Offset (posição dentro do grupo);
- Dados enviados e/ou número de bytes a receber.

O runtime do PLC recebe o pedido, executa-o e devolve uma resposta ADS. O PLC Beckhoff organiza internamente a memória em zonas específicas. O acesso ADS utiliza a combinação:

- Index Group (IG) → Tipo de memória;
- Index Offset (IO) → Posição exata dentro da memória.

Para a memória %M existem grupos normalizados:

Tabela 3.1: Especificações de Index Group/Offset do PLC[51]

Função	Index Group	Index Offset	Descrição
Ler/escrever bytes %M	**0x4020**	**Offset = nº do byte**	Acede diretamente à memória %MB (byte a byte).
Ler/escrever bits %MX	**0x4021**	**Offset = (byte * 8) + bit**	Endereçamento por bit, conforme %MXx.y.
Tamanho da área %M	**0x4025**	**0**	Devolve o tamanho total da memória %M em bytes.
Ler/escrever retain	**0x4030**	**Offset = nº do byte**	Lê/escreve bytes na área RETAIN (dados persistentes).
Tamanho retain	**0x4035**	**0**	Devolve o tamanho da área RETAIN.
Ler/escrever área de dados	**0x4040**	**Offset = nº do byte**	Acede à área "Data", onde o runtime guarda variáveis.
Tamanho área de dados	**0x4045**	**0**	Devolve o tamanho da área de dados.

O fluxo de comunicação ADS (PC para PLC para PC) é iniciada pela aplicação PC que prepara pedido ADS, enviando um comando. AMS Router do PC recebe o pedido,

identifica o destino, com o AMS Net ID do PLC, e encaminha pela a interface configurada (TCP/IP). O PLC recebe o pedido e o runTime TwinCAT interpreta. Pois analisa o Index Group, calcula o endereço interno o Index Offset, lê ou escreve a memória e executa validações. O PLC devolve a resposta ADS, com confirmação de escrita ou valor lido, ou estado ou erro. E por fim AMS Router devolve a resposta à aplicação PC.

3.1.4.2 Análise das mensagens ADS captadas pelo Wireshark

Na captura de pacotes de comunicação verificou-se a existência de imensuráveis comunicações e mesmo com a opção de filtragem do programa, seja no tipo de protocolo de comunicação e/ou IP's da fonte e receção, constatou-se que faz uma comunicação de leitura de dados aproximadamente entre 0.5 segundos resultando entre 140 comunicações a 844 comunicações dependendo dos comandos enviados ao *miniload*. Desse modo, foi elaborado um programa em python para verificar quando a informação nas mensagens é diferente com o objetivo de desvendar o que significa cada byte.

A biblioteca utilizada foi a **pyshark**, responsável pela leitura dos ficheiros de captura do Wireshark (.cap e .pcapng). A biblioteca **os** é utilizada para a manipulação de caminhos e nomes de ficheiros.

A variável **CAPTURE_FILE** contém o caminho completo para o ficheiro de captura do Wireshark que será analisado. A função **os.path.basename(CAPTURE_FILE)** extrai apenas o nome do ficheiro, enquanto **os.path.splitext(os.path.basename(CAPTURE_FILE))[0]** remove a extensão (.pcapng), resultando apenas no nome base, que é armazenado na variável **name_text**.

Por fim, a instrução **arquivo = open(name_text + ".txt", "w", encoding="utf-8")** cria um ficheiro de texto (.txt) com o mesmo nome do ficheiro de captura, onde será guardado o resultado completo da análise dos pacotes (Figura 3.10).

```
1  import pyshark
2  import os
3  import re
4
5  # Defina o arquivo de captura (.pcap ou .pcapng) com a fonte do ficheiro
6  CAPTURE_FILE = r'C:\Users\User\OneDrive - ISEC\Ambiente de Trabalho\ISEC_24.25\Projeto_2425\wireshark\Coleta\
7      wireshark_save_13_03_2025_mov_AgulhetaAla4_Reciclagem.pcapng'
8  # Substitua pelo seu arquivo
9  # filtro = ""
10 name_text = os.path.splitext(os.path.basename(CAPTURE_FILE))[0]
11 print(name_text)
12 arquivo = open(name_text + ".txt", "w", encoding="utf-8")
13 arquivo.write("Dados do ficheiro "+ name_text + ".pcapng\n")
```

Figura 3.10: Função principal do script Python para análise de pacotes AMS/ADS capturados no Wireshark.

A função **def analisar_pacotes(filename)** é a função principal de análise dos pacotes. Esta função recebe o caminho do ficheiro de captura **filename** e mostra uma mensagem na consola a indicar o início da análise (Figura 3.11).

Muito semelhante ao programa do Wireshark, usou-se um filtro **display_filter** descrito infra e ilustrado na Figura 3.11. Este filtro tem o propósito de filtrar apenas as comunicações do protocolo AMS e separar o tráfego bidirecional entre os IP's 192.168.41.130 (que pertence ao PC) e o 192.168.41.132 (que pertence ao PLC do plano do Armazém Automático). Ao separar as comunicações entre o PC e o PLC do *miniload* e o PC entre o PLC do Armazém Automático torna mais simples fazer o estudo e a compreensão dos dados da comunicação. Por fim, usa se dois filtros:

- "ams and (ip.src == 192.168.41.130 and ip.dst == 192.168.41.132) or (ip.src == 192.168.41.132 and ip.dst == 192.168.41.130)"
- "ams and (ip.src == 192.168.41.130 and ip.dst == 192.168.41.133) or (ip.src == 192.168.41.133 and ip.dst == 192.168.41.130)"

A variável **data_atual** guarda os dados da resposta anterior, para poder detetar alterações com os próximos dados. A variável **data_new** guarda os dados da resposta mais atual para poder comparar com os dados anteriores. Já a variável **count** tem a função de contar e numerar os comandos de escrita que vão sendo detetados.

```
19  def analisar_pacotes(filename):
20      print("🔍 Analisando pacotes AMS/ADS no Wireshark...\n")
21
22      # Abre o arquivo de captura com filtro AMS
23      #cap = pyshark.FileCapture(filename, display_filter="ams and (ip.src == 192.168.41.130 and ip.dst == 192.168.41.133)
24      # or (ip.src == 192.168.41.133 and ip.dst == 192.168.41.130)") #alterar o filtro
25      cap = pyshark.FileCapture(filename, display_filter="ams and (ip.src == 192.168.41.130 and ip.dst == 192.168.41.132) "
26      "or (ip.src == 192.168.41.132 and ip.dst == 192.168.41.130)")
27      data_atual=""   #variavel com os dados atuais para poder comparar com os dados futuros diferentes
28      data_new =""    #variavel com os dados mais recentes para poder comparar com os ultimos dados anteriores
29      count = 1
```

Figura 3.11: Definição dos filtros de captura AMS/ADS e inicialização das variáveis de controlo da análise de pacotes.

Usa um ciclo **for** para percorrer todos os pacotes que passam pelo filtro definido anteriormente. No caso do primeiro pacote (primeira iteração do ciclo for) atribuí valores às seguintes variáveis (Figura 3.12):

- **num_pacote_req:** recebe o valor do número do pacote.
- **ip_src_req:** guarda o valor do IP do dispositivo que envia o comando.
- **ip_dst_req:** recebe o valor do IP do dispositivo destinatário, ou seja, o que recebe o valor.
- **index_offset_req**: recebe o valor **0x0000059a**, o qual, após a análise de várias comunicações, foi identificado como sendo o valor utilizado quando é enviado um comando de escrita para o PLC do *miniload*.

```
31      for packet in cap:
32          try:
33              # Número do pacote
34              num_pacote = packet.number
35              if num_pacote == '1':
36                  num_pacote_req = num_pacote
37                  ip_src_req = packet.ip.src
38                  ip_dst_req = packet.ip.dst
39                  index_offset_req = "0x0000059a"
40
```

Figura 3.12: Ciclo principal de iteração sobre os pacotes filtrados e identificação do primeiro pedido ADS.

Para todos os pacotes recolhe-se a seguinte informação (Figura 3.13):

- **ip_srs:** guarda o valor do IP do dispositivo que envia o comando.
- **ip_dst:** recebe o valor do IP do dispositivo destinatário, ou seja, o que recebe o valor.
- **protocolo:** recebe toda a informação da camada mais alta identificada no pacote.

A instrução **hasattr(packet, 'ams'):** verifica se o pacote tem a camada AMS, no caso de essa camada estar presente, guarda essa camada na variável **ams** para aceder aos campos específicos (como por exemplo: cmdid, ads_indexoffset, adscblenght, etc.).

```
43              # IP de origem e destino
44              ip_src = packet.ip.src
45              ip_dst = packet.ip.dst
46
47              # Protocolo
48              protocolo = packet.highest_layer
49
50              # Verifica se é um pacote AMS com os campos necessários
51              if hasattr(packet, 'ams'):
52                  ams = packet.ams
```

Figura 3.13: Extração dos endereços IP, protocolo AMS e inicialização das variáveis após o primeiro pacote analisado.

A partir deste ponto, procede-se à verificação do conteúdo dos dados, realizando o respetivo tratamento. Inicialmente, faz-se a distinção entre comandos de leitura, identificados pelo valor (**2**), e comandos de escrita, representados pelo valor (**3**). De seguida, verifica-se se o comando é de leitura através da instrução **if(ams.cmdid == '2'):**.

Posteriormente, valida-se se se trata de um pedido de leitura (*Read Request*) com a instrução **if ip_src == "192.168.41.130"and ip_dst in ["192.168.41.133", "192.168.41.132"]:**. Uma vez que os pedidos de leitura têm sempre origem no computador e são enviados

para os PLC's, tanto o do *miniload* como o do Armazém Automático, o endereço IP de origem é comparado com o valor **"192.168.41.130"**, enquanto os endereços IP de destino podem corresponder a qualquer um dos PLC's, sendo por isso comparados com os valores **"192.168.41.133"** e **"192.168.41.132"**.

Caso esta condição seja satisfeita, a informação relevante é armazenada nas seguintes variáveis (Figura 3.14):

- **num_pacote_req:** guarda o número do pacote para auxiliar na identificação dentro do programa Wireshark.
- **index_offset_req:** recebe o valor do index_Offset, para saber de que posição na memória do PLC começa a ler os dados.
- **ip_src_req e ip_dst_req:** variável auxiliar para introduzir no ficheiro .txt.

```
53
54              if(ams.cmdid == '2'):   #verifica se o comando é leitura "read"
55                  if ip_src == "192.168.41.130" and ip_dst in ["192.168.41.133", "192.168.41.132"]:
56                      num_pacote_req = packet.number
57                      index_offset_req = ams.ads_indexoffset  #ler Indexoffset
58                      ip_src_req = ip_src
59                      ip_dst_req = ip_dst
```

Figura 3.14: Identificação do tipo de comando ADS (leitura ou escrita) com base no campo **cmdid**.

Ainda dentro da instrução de validação de leitura, verifica se é a resposta do pedido de leitura com a instrução **if ip_dst == "192.168.41.130"and ip_src in ["192.168.41.133", "192.168.41.132"]:**. Como se pode ver, nesta instrução faz-se o inverso. O IP destino será "192.168.41.130"e a origem um dos outros IP's e deste modo, considera-se um ADS Read Response. Neste caso é importante identificar os dados recebidos e de momento não foi possível obter os dados diretamente a partir da variável **ver qual é a variável?!?!!?**. Desse modo foi necessário ocorrer por outra via. Após uma análise das variáveis possíveis com a informação dos dados, encontrou se as seguintes variáveis:

- **packet.tcp.payload:** que contém toda a informação do protocolo ADS.
- **packet.ams.ads_cblength:** contém o número de dados da resposta.

Como a variável do número de dados está em formato de string, é necessário converter para número inteiro e guarda na variável **cb_length** . E os dados na variável obtêm os dados, como por exemplo: **'00:00:46:00:00:00:c0:a8:29:82:01:01:85:81:c0:a8:29:84:01:01:21:03:02:00:05:00:26:00:00:00:00:00:00:00:bf:31:00:00:00:00:00:00:1e:00:00:00:12:01:00:00:00:00:01:00:41:01:00:02:00:00:21:01:00:01:00:00:21:01:00:01:00:00:00:01:00:00'**. É possível identificar que cada Byte, passou para dois dígitos e a somar com os dois pontos resume a três caracteres por cada Byte. Logo os dados será sempre o triplo do número de dados, subtraí se um para retirar o primeiro dois pontos, e guarda se numa nova variável **data_new**, em que a instrução final se resume a: **data_new = packet.tcp.payload[-(cb_length*3-1):]**. De seguida verifica se as variáveis **data_new** e **data_atual** se são

diferentes, no caso de serem, vai-se escrever essa informação no ficheiro .txt, a seguinte informação:

- **arquivo.write(f"**Pedido ADS Read Request** (Pacote No. num_pacote_req)\n"):** informa que é um pedido ADS "Read Request"com o número de pacote, para facilitar ao localizar se for necessário verificar no programa de Wireshark.
- **arquivo.write(f"IP Origem: {ip_src_req} → IP Destino: {ip_dst_req}\n"):** indica o IP origem e destino do pedido.
- **arquivo.write(f"Ler a partir do endereço de memoria: {index_offset_req}\n\n"):** indica a localização do endereço de memória que inia a leitura de dados.
- **arquivo.write(f"**Resposta ADS Read Response** (Pacote No. {num_pacote})\n"):** apresenta que é a resposta ADS (Read Response) e a indicar o número de pacote.
- **arquivo.write(f"IP Origem: ip_src → IP Destino: {ip_dst}\n"):** indica o IP origem e destino da resposta.
- **arquivo.write(f"Data: {data_new}\n"):** e por fim apresenta os dados que foram lidos.

No caso de os dados lidos serem iguais aos dados mais antigos, não há necessidade de escrever novamente, pois o objetivo é escrever apenas quando são diferentes.

```
61          if ip_dst == "192.168.41.130" and ip_src in ["192.168.41.133", "192.168.41.132"]:
62              cb_length = int(packet.ams.ads_cblength)
63              data_new = packet.tcp.payload[-(cb_length*3-1):] # tamanho = (CbLenght * 3)-1
64              data_new = limpar_hex(data_new)     #limpa os dois pontos (:) dos dados
65              if(data_new != data_atual):
66                  data_atual = data_new
67                  arquivo.write(f"✅ **Pedido ADS Read Request** (Pacote No. {num_pacote_req})\n")
68                  arquivo.write(f"   📌 IP Origem: {ip_src_req} → IP Destino: {ip_dst_req}\n")
69                  arquivo.write(f"    Ler a partir do endereço de memoria: {index_offset_req}\n\n")  # Dupla q
70
71                  arquivo.write(f"✅ **Resposta ADS Read Response** (Pacote No. {num_pacote})\n")
72                  arquivo.write(f"   📌 IP Origem: {ip_src} → IP Destino: {ip_dst}\n")
73                  arquivo.write(f"   🔹 Data: {data_new}\n\n")  # Dupla quebra de linha para separar entradas
```

Figura 3.15: Processamento dos pedidos ADS Read Request enviados do PC para os PLCs.

Mas para facilitar ainda mais, quando os dados forem diferentes, vai se comparar Byte a Byte para ver onde houve modificações. Desse modo as instruções na figura 3.16 compara byte a byte, ou seja, como dito anteriormente, dois caracteres representam um Byte, os dados de leitura atual (**data_new**) com os dados da leitura anterior (**data_atual**. A variável **min_len** é o comprimento da menor string para, não sair fora dos limites, quando for realizar a comparação entre as variáveis. Se forem diferentes, adiciona à lista **difs** uma descrição da diferença (**byte antigo-> byte novo**). Se a nova leitura de dados conter mais Bytes do que na leitura anterior, regista os bytes adicionais com sendo **– > novo_byte**.

```
90                  # Comparação byte a byte
91                  difs = []
92                  min_len = min(len(data_new), len(data_atual))
93
94                  for i in range(0, min_len, 2):  # Avança 2 caracteres por vez (1 byte)
95                      byte_new = data_new[i:i+2]
96                      byte_old = data_atual[i:i+2]
97                      if byte_new != byte_old:
98                          difs.append(f"Byte {i//2}: {byte_old} → {byte_new}")
99
100                 # Se new for maior (houve extensão da data)
101                 if len(data_new) > len(data_atual):
102                     for i in range(min_len, len(data_new), 2):
103                         byte_new = data_new[i:i+2]
104                         difs.append(f"Byte {i//2}: -- → {byte_new}")
105
```

Figura 3.16: Identificação e tratamento das respostas ADS Read Response provenientes dos PLCs.

E no caso de houver diferenças, escreve-as no ficheiro .txt de forma legível e destacável para ver quais são os Bytes que modificaram. Após de escrever no ficheiro toda a informação, é necessário atualizar a variável **data_atual** com a **data_new**, para que a próxima comparação seja sempre com o último valor obtido.

```
106                 if difs:
107                     arquivo.write("    Diferenças detectadas nos dados:\n")
108                     for diff in difs:
109                         arquivo.write(f"      - {diff}\n")
110
111                 arquivo.write("\n")  # separação entre blocos
112                 data_atual = data_new  # atualiza o último valor guardado
```

Figura 3.17: Comparação byte a byte entre leituras sucessivas de dados ADS para deteção de alterações.

A parte do código para interpretar os pedidos de comando por parte do PC para os PLC, designado pelo o comando número "3"é bastante semelhante ao pedido de leitura anteriormente explicado. Mas agora os dados são enviados pelo pedido de escrita (Write Request) e como se pode observar na figura 3.18, inicio se por identificar um pacote com ADS Write Request tem origem do IP que pertence ao PC e os IP's origem são dos PLC's, se for, a variável **num_pacote_w_req** vai guardar o número do pacote, **index_offset_w_req** guarda endereço de memória onde vai ser escrita a informação e **ip_src_w_req** o IP fonte e **ip_dst_w_req** o IP destino. A variável **cb_length_w** têm o mesmo propósito que na ideia anterior, de saber o comprimento de dados que é necessário tirar à variável **packet.tcp.payload** e guarda na variável **data_write**. Depois vai escrever no ficheiro .txt o pedido ADS Write, com toda a informação disposta anteriormente numerado, para saber o número de comandos enviados.

```
115             if(ams.cmdid == '3'):
116                 if ip_src == "192.168.41.130" and ip_dst in ["192.168.41.133", "192.168.41.132"]:
117                     num_pacote_w_req = packet.number
118                     index_offset_w_req = ams.ads_indexoffset
119                     ip_src_w_req = ip_src
120                     ip_dst_w_req = ip_dst
121                     cb_length_w = int(packet.ams.ads_cblength)
122                     data_write = packet.tcp.payload[-(cb_length_w*3-1):] # tamanho = (CbLenght * 3)-1 pode ser data comando...
123                     #tipos de comando (colocar curva ou reta, etc)
124                     arquivo.write(f"✅ **Pedido ADS Write Request** (Pacote No. {num_pacote_w_req})\n")
125                     arquivo.write(f"   📌 IP Origem: {ip_src_w_req} → IP Destino: {ip_dst_w_req}\n")
126                     arquivo.write(f"    Ler a partir do endereço de memoria: {index_offset_w_req}\n\n")  # Dupla quebra de linha para separar entradas
127                     arquivo.write(f"   🔹 Data ({str(count)}): {data_write}\n\n")  # Dupla quebra de linha para separar entradas
128                     count += 1
```

Figura 3.18: Registo estruturado das diferenças detetadas entre leituras ADS no ficheiro de saída.

De seguida, o código da figura 3.19 têm a função de analisar se o pacote é interpretado como resposta ao comando como **Write Response** se o destino é o IP do PC e os IP's origem é algum que pertence aos PLC's. Também vai guardar o número do pacote na variável **num_pacote_w_resp**, os IP's de origem e destinatário e o resultado, que pode ser ou não um tipo de erro.

```
160             except AttributeError:
161                 continue
162
163         print("✅ Análise concluída!")
164         arquivo.close()
```

Figura 3.19: Processamento dos pedidos ADS Write Request, incluindo extração de dados e endereços de memória.

A próxima instrução **except AttributeError:** serve para o caso de alguns pacotes não conter todas as camadas ou campos esperados (por exemplo, pode não existir **packet.ip** ou **packet.tcp** em certos pacotes) e se faltar algmu atributo, ocorre um **AttributeError**. Nesse caso, o except apanha o erro e o ciclo passa simplesmente ao pacote seguinte, sem interromper a análise. No fim mostra na consola que a análise terminou. Fecha o ficheiro de saída com a instrução **arquivo.close()**, garantindo que todos os dados foram guardados no ficheiro e termina a função **analisar_pacotes**.

```
160             except AttributeError:
161                 continue
162
163         print("✅ Análise concluída!")
164         arquivo.close()
```

Figura 3.20: Identificação e validação das respostas ADS Write Response dos PLCs

Por fim, chama a função **analisar_pacotes** com o ficheiro de captura definido no início e é a partir daqui que o script começa efetivamente a análise quando é executado.

```
166    # Executa a análise no arquivo definido
167    analisar_pacotes(CAPTURE_FILE)
```

Figura 3.21: Execução final da função de análise e geração do ficheiro de resultados da comunicação ADS.

É possível ver um exemplo de ficheiro resultado da execução do script em AnexoX.

3.1.4.3 Análise de pacotes

Os pacotes analisados foram captados a partir de pedidos efetuados pelo software WMS, correspondentes a movimentos completos do sistema *miniload* para a recolha e posterior devolução dos contentores às respetivas posições.

Na Tabela 3.2 apresenta-se a informação relativa aos contentores e às suas localizações, utilizada nos movimentos considerados para a captura e análise dos pacotes.

Um ciclo completo de funcionamento do *miniload* é composto pelas seguintes etapas:

- 1º - Deslocação do *miniload* até à localização do contentor solicitado e remoção do contentor da respetiva rack;
- 2º - Retorno à posição da base de interface com o operador e deposição do contentor na base;
- 3º - Realização do processo de *check-in* do contentor e dos produtos associados;
- 4º - Retirada do contentor da base de interface e reposicionamento do mesmo na sua localização original na rack.

Tabela 3.2: Posições dos contentores utilizadas nos movimentos do *miniload*.

Nº do Contentor	Pos X	Pos Y	Pos Z	Nº de Ala
1	2	1	1	1
25	16	1	1	1
29	18	1	1	1
31	18	1	2	1
122	2	4	1	1
123	2	4	2	1
525	5	2	2	2
528	4	2	2	2
1291	4	2	1	3
1292	4	2	2	3
1829	13	8	1	3
1929	3	2	2	4
2596	4	2	2	5

Após a análise dos pacotes capturados, verificou-se que cada movimento completo do sistema era composto por oito comandos de escrita (Write Request), sendo que cada comando continha trinta e quatro bytes de dados.

Após a reunião, organização e conversão dos comandos para o formato decimal, foi possível identificar os seguintes padrões, descritos de seguida.

Dados do primeiro comando (*Write Request*)

A compilação de todos os primeiros comandos associados aos movimentos dos contentores deu origem à Tabela 3.3. Comparando esta tabela com a informação apresentada na Tabela 3.2, é possível observar que o primeiro byte corresponde à posição X e o segundo byte à posição Y dos contentores.O terceiro byte assume valores entre 1 e 5, estando assim associado ao número da Ala. O quarto byte apresenta sempre o valor 0, enquanto o quinto byte varia apenas entre os valores 16 e 32. O sexto byte assume valores entre 1 e 2, correspondendo à posição Z. O sétimo byte apresenta sempre o valor 1 e o oitavo byte o valor 0.

O nono byte aparenta representar diretamente o número do contentor. No entanto, a partir do contentor número 525, esta correspondência deixa de ser direta, uma vez que o valor máximo representável por um byte em hexadecimal é **FF**, correspondente a 255 em decimal. Torna-se, assim, necessária a utilização de dois bytes para representar números superiores a este valor.

Ao analisar também o décimo byte, verifica-se que a combinação dos dois bytes permite obter corretamente o número do contentor. No caso do contentor 525, o nono byte assume o valor hexadecimal 0D e o décimo byte o valor **02**. Se os bytes forem conca-

tenados na ordem 0D02 e convertidos para decimal, obtém-se o valor 3330, que não corresponde ao número do contentor. No entanto, ao inverter a ordem dos bytes, formando o valor **020D**, a conversão para decimal resulta corretamente em 525. Constata-se ainda que do décimo segundo ao vigésimo segundo byte, os valores observados mantiveram-se constantes e iguais a zero, independentemente do n.º de contentor solicitado. O trigésimo terceiro e trigésimo quarto bytes assumem sempre o valor 16 e 17, respetivamente.

Conclui-se, assim, que o nono byte corresponde ao byte menos significativo (**LSB**) e o décimo byte ao byte mais significativo (**MSB**) do número do contentor. O décimo primeiro byte apresenta sempre o valor 0.

Deste modo, conclui-se que o objetivo principal do primeiro comando é instruir o *miniload* a deslocar-se até à localização do contentor solicitado (Tabela 3.3).

Tabela 3.3: Valores dos bytes do primeiro comando (*Write Request*).

	Bytes recebidos em cada posição da trama												
	X	Y	Ala			Z			**LSB**	**MSB**			
N.º Contentor	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º ao 32º	33º	34º
1	2	1	1	0	16	2	1	0	1	0	0	16	17
25	16	1	1	0	16	2	1	0	25	0	0	16	17
29	18	1	1	0	16	2	1	0	29	0	0	16	17
31	18	1	1	0	16	1	1	0	31	0	0	16	17
122	2	4	1	0	16	2	1	0	122	0	0	16	17
123	2	4	1	0	16	1	1	0	123	0	0	16	17
525	5	2	2	0	16	2	1	0	13	2	0	16	17
528	4	2	2	0	16	2	1	0	16	2	0	16	17
1291	4	2	3	0	16	1	1	0	11	5	0	16	17
1292	4	2	3	0	16	2	1	0	12	5	0	16	17
1829	13	8	3	0	16	1	1	0	37	7	0	16	17
1929	3	2	4	0	32	2	1	0	137	7	0	16	17
2596	4	2	5	0	16	2	1	0	36	10	0	16	17

Desta forma, foi possível identificar a estrutura do corpo do comando a enviar ao *miniload*, permitindo que o sistema execute corretamente a operação de recolha de um contentor no armazém.

Dados do segundo, quarto, sexto e oitavo comando (*Write Request*)

Os valores dos dados nestes comandos são todos iguais a zero, com exceção do trigésimo terceiro byte, que assume o valor 255, e do trigésimo quarto byte, que apresenta o valor 17, conforme se pode observar na Tabela 3.4. O segundo comando é enviado imediatamente após o *miniload* iniciar o movimento de deslocação para efetuar a recolha

do contentor. O quarto comando corresponde ao movimento de retorno do *miniload* em direção à base. O sexto comando é transmitido após o *miniload* depositar o contentor na base do operador e, por fim, o oitavo comando é enviado quando é solicitada a devolução do contentor para a sua posição original no armazém. A principal conclusão desta sequência é que o byte da posição 33 com o valor a 255 ordena ao *miniload* para iniciar o movimento previamente indicado.

Tabela 3.4: 1.º ao 34.º byte do segundo comando (*Write Request*).

	Bytes recebidos em cada posição da trama					
N.º de Contentor	1º ao 9º	10º ao 19º	20º ao 29º	30º ao 32º	33º	34º
1	0	0	0	0	255	17
25	0	0	0	0	255	17
29	0	0	0	0	255	17
31	0	0	0	0	255	17
122	0	0	0	0	255	17
123	0	0	0	0	255	17
525	0	0	0	0	255	17
528	0	0	0	0	255	17
1291	0	0	0	0	255	17
1292	0	0	0	0	255	17
1829	0	0	0	0	255	17
1929	0	0	0	0	255	17
2596	0	0	0	0	255	17

Dados do terceiro comando (*Write Request*)

Após o *miniload* recolher o contentor da rack, é enviado um comando para que este se desloque até à base do operador. Os dados deste comando, conforme se pode observar na Tabela 3.5, são referentes à posição da base do operador, correspondendo às coordenadas X = 1, Y = 1 e Z = 2.

Tabela 3.5: 1.º ao 34.º byte do terceiro comando (*Write Request*).

	Bytes recebidos em cada posição da trama												
	X	Y	Ala			Z			**LSB**	**MSB**			
N.º Contentor	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º ao 32º	33º	34º
1	1	1	1	0	32	2	1	0	1	0	0	16	17
25	1	1	1	0	32	2	1	0	25	0	0	16	17
29	1	1	1	0	32	2	1	0	29	0	0	16	17
31	1	1	1	0	32	2	1	0	31	0	0	16	17
122	1	1	1	0	32	2	1	0	122	0	0	16	17
123	1	1	1	0	32	2	1	0	123	0	0	16	17
525	1	1	1	0	32	2	1	0	13	2	0	16	17
528	1	1	1	0	32	2	1	0	16	2	0	16	17
1291	1	1	1	0	32	2	1	0	11	5	0	16	17
1292	1	1	1	0	32	2	1	0	12	5	0	16	17
1829	1	1	1	0	32	2	1	0	137	7	0	16	17
1929	1	1	1	0	16	2	1	0	137	7	0	16	17
2596	1	1	1	0	32	2	1	0	36	10	0	16	17

Dados do quinto comando (*Write Request*)

Após o *miniload* chegar à base do operador, é enviado o quinto comando com o objetivo de efetuar a deposição do contentor na base do operador. Na Tabela -3.6, verifica-se que o 7º byte passa a zero, indicando a posição correspondente à deposição do contentor no operador (se valor é: 1- armazém; 0-operador).

Tabela 3.6: 1.º ao 34.º byte do quinto comando (*Write Request*)

	Bytes recebidos em cada posição da trama												
	X	Y	Ala			Z			**LSB**	**MSB**			
N.º Contentor	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º ao 32º	33º	34º
1	1	1	1	0	16	2	0	0	1	0	0	16	17
25	1	1	1	0	16	2	0	0	25	0	0	16	17
29	1	1	1	0	16	2	0	0	29	0	0	16	17
31	1	1	1	0	16	2	0	0	31	0	0	16	17
122	1	1	1	0	16	2	0	0	122	0	0	16	17
123	1	1	1	0	16	2	0	0	123	0	0	16	17
525	1	1	1	0	16	2	0	0	13	2	0	16	17
528	1	1	1	0	16	2	0	0	16	2	0	16	17
1291	1	1	1	0	16	2	0	0	11	5	0	16	17
1292	1	1	1	0	16	2	0	0	12	5	0	16	17
1829	1	1	1	0	16	2	0	0	37	7	0	16	17
1929	1	1	1	0	16	2	0	0	137	7	0	16	17
2596	1	1	1	0	16	2	0	0	36	10	0	16	17

Dados do sétimo comando (*Write Request*)

Este comando tem como objetivo recolher o contentor da base do operador e devolvê-lo à sua posição original no armazém. Conforme se pode verificar, a estrutura dos dados apresentada na Tabela 3.7 é idêntica à da Tabela 3.3.

Tabela 3.7: 1.º ao 34.º byte do sétimo comando (*Write Request*).

	Bytes recebidos em cada posição da trama												
	X	Y	Ala			Z			**LSB**	**MSB**			
Nº Contentor	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º ao 32º	33º	34º
1	2	1	1	0	32	2	1	0	1	0	0	16	17
25	16	1	1	0	32	2	1	0	25	0	0	16	17
29	18	1	1	0	32	2	1	0	29	0	0	16	17
31	18	1	1	0	32	1	1	0	31	0	0	16	17
122	2	4	1	0	32	2	1	0	122	0	0	16	17
123	2	4	1	0	32	1	1	0	123	0	0	16	17
525	5	2	2	0	32	2	1	0	13	2	0	16	17
528	4	2	2	0	32	2	1	0	16	2	0	16	17
1291	4	2	3	0	32	1	1	0	11	5	0	16	17
1292	4	2	3	0	32	2	1	0	12	5	0	16	17
1829	13	8	3	0	32	1	1	0	37	7	0	16	17
1929	3	2	4	0	32	2	1	0	137	7	0	16	17
2596	4	2	5	0	32	2	1	0	36	10	0	16	17

3.2 Levantamento de Requisitos

O levantamento de requisitos constitui uma etapa fundamental no desenvolvimento de qualquer sistema de automação industrial, pois permite identificar de forma estruturada as necessidades funcionais e não funcionais, bem como as restrições impostas pelo sistema existente. No contexto do presente projeto, esta análise tem como base o sistema AS/RS atualmente em funcionamento no Arquivo Automático, descrito na secção 3.1, e visa suportar a definição de propostas de melhoria realistas, tecnicamente viáveis e compatíveis com a infraestrutura instalada atual.

A análise efetuada permitiu identificar limitações ao nível da arquitetura de software, das comunicações e da flexibilidade do sistema, justificando a necessidade de modernização da camada de supervisão e gestão, mantendo, contudo, o controlo de baixo nível assegurado pelos PLCs existentes.

3.2.1 Requisitos Funcionais

Os requisitos funcionais definem as funcionalidades que o sistema otimizado deve assegurar, de modo a responder às necessidades operacionais e de gestão do armazém automático. Foram identificados os seguintes requisitos funcionais principais:

- O sistema deve permitir a criação, gestão e execução de pedidos de armazenamento e recolha de contentores de acordo com a intenção do operador.

- Deve ser possível monitorizar em tempo real o estado do *miniload*, das agulhetas, dos sensores e dos atuadores.
- O sistema deve assegurar a comunicação bidirecional entre o computador do operador e o PLC Beckhoff, permitindo a leitura de estados e o envio de comandos.
- Deve existir uma interface gráfica que permita ao operador acompanhar a operação do sistema, visualizar alarmes e intervir quando necessário.
- O sistema deve permitir o registo histórico de operações, incluindo pedidos executados, tempos de ciclo, falhas e alarmes.
- Deve ser possível executar comandos de manutenção e diagnóstico de forma controlada, sem comprometer a segurança do sistema.

3.2.2 Requisitos Não Funcionais

Para além das funcionalidades, o sistema deve cumprir um conjunto de requisitos não funcionais, que garantem a qualidade, robustez e sustentabilidade da solução proposta:

- **Fiabilidade:** o sistema deve operar de forma contínua e estável, minimizando falhas de comunicação e interrupções não planeadas.
- **Desempenho:** os tempos de resposta do sistema devem ser compatíveis com os requisitos de operação em tempo real do AS/RS.
- **Manutenção:** a arquitetura de software deve ser modular, permitir futuras correções simplificadas, de atualizações ou até mesmo de adicionar extensões.
- **Utilidade:** a interface com o operador deve ser intuitiva, reduzindo a probabilidade de erro humano.
- **Segurança:** as melhorias implementadas não devem comprometer os mecanismos de segurança existentes no sistema físico e no controlo do PLC.

3.2.3 Restrições do Sistema

O desenvolvimento das melhorias está condicionado por um conjunto de restrições técnicas e operacionais, resultantes da natureza do sistema em produção:

- Os PLCs Beckhoff existentes e o respetivo programa desenvolvido em TwinCAT 2 não devem ser substituídos.
- A estrutura mecânica do *miniload*, das racks e das agulhetas deve manter-se inalterada.
- O sistema encontra-se em funcionamento regular, pelo que as alterações devem ser minimamente intrusivas.

- A infraestrutura de rede existente deve ser reutilizada, sempre que possível.
- As soluções propostas devem privilegiar tecnologias compatíveis com o ambiente industrial existente.

3.3 Propostas de Melhoria

Devido à idade avançada do sistema WMS, à falta de atualizações nos últimos anos e à ocorrência de comandos incorretos enviados para o PLC, o que compromete a fiabilidade do sistema e provoca paragens indesejadas do AS/RS, afetando o trabalho dos operadores, tornou-se necessário propor melhorias. Com base no levantamento de requisitos apresentado na secção anterior, foram definidas um conjunto de ações focadas na modernização da camada de supervisão, do controlo de alto nível e das comunicações do sistema AS/RS. O objetivo principal é aumentar a flexibilidade, facilitar a manutenção, melhorar a capacidade de evolução do sistema e, ao mesmo tempo, reduzir a dependência de soluções proprietárias.

3.3.1 Limitações da Arquitetura Atual

A arquitetura atualmente em uso baseia-se na interligação entre o software CONSOVEYOR WMS, o servidor Kepware OPC, a base de dados Oracle e o PLC Beckhoff, através do TwinCAT OPC Server. Apesar de funcional, esta solução apresenta várias limitações:

- forte dependência de software proprietário e licenças comerciais.
- Arquitetura complexa, com múltiplas camadas intermédias de comunicação.
- Dificuldade de manutenção e de introdução de novas funcionalidades.
- Baixa flexibilidade para integração com tecnologias modernas, como APIs REST

Estas limitações motivam a necessidade de uma reestruturação da arquitetura de software.

3.3.2 Arquitetura Proposta

A proposta de melhoria consiste na substituição da camada de supervisão e gestão por uma arquitetura mais simples, aberta e flexível, mantendo o PLC Beckhoff como núcleo do controlo em tempo real. A nova arquitetura assenta nos seguintes princípios:

- Comunicação direta entre o computador do operador e o PLC através do protocolo ADS.
- Desenvolvimento de um novo sistema WMS/WCS em software, responsável pela gestão de pedidos, lógica de decisão e interface com o operador.

- Utilização de uma API para comunicação interna entre módulos de software.
- Adoção de uma base de dados mais leve e flexível para armazenamento de informação operacional.

Esta abordagem reduz o número de componentes intermédios, melhora a transparência do sistema e facilita a sua evolução futura.

3.3.3 Benefícios das Melhorias Propostas

As melhorias propostas apresentam diversas vantagens face à arquitetura existente:

- um sistema WMS mais simples e moderno
- redução de custos associados a licenças de software proprietário.
- maior controlo sobre o código e sobre a lógica de funcionamento do sistema.
- facilidade de manutenção e adaptação a novos requisitos.
- melhor integração com tecnologias modernas de automação.
- aumento da robustez e da eficiência das comunicações entre sistemas.

3.4 Ferramentas e Métodos

Para a implementação das melhorias propostas foi definido um conjunto de ferramentas e métodos adequados ao contexto industrial do sistema AS/RS e às restrições identificadas.

3.4.1 Ferramentas de Software

As principais ferramentas selecionadas para o desenvolvimento do projeto são:

- **TwinCAT 2:** manutenção do ambiente de desenvolvimento e execução do PLC, assegurando a compatibilidade com o sistema existente.
- **Python:** linguagem de programação utilizada para o desenvolvimento do novo WMS/WCS, devido à sua flexibilidade e ampla disponibilidade de bibliotecas.
- **PyADS:** biblioteca utilizada para estabelecer comunicação direta entre o software em Python e o PLC Beckhoff através do protocolo ADS.
- **Framework de interface gráfica WEB:** utilizado para a criação da interface com o operador.
- **Sistema de gestão de base de dados:** responsável pelo armazenamento de pedidos, estados e históricos de operação.

3.4.2 Método de Desenvolvimento

O desenvolvimento do sistema segue uma abordagem modular, com separação clara entre as camadas de comunicação, lógica de controlo e interface com o utilizador. Foram adotados os seguintes métodos:

- Desenvolvimento incremental, com validação funcional de cada módulo.
- Testes unitários de comunicação entre o software e o PLC.
- Simulação e testes em ambiente controlado antes da integração total.
- Documentação técnica das interfaces e funcionalidades implementadas.

3.5 Conclusão

Neste Capítulo, descreveu-se o estudo preliminar do sistema existente, conduzido sem acesso a documentação de desenvolvimento, sendo apenas considerada a documentação de utilização. Inicialmente, foram identificados todos os blocos principais do sistema, bem como os softwares que os suportam. Seguiu-se a identificação detalhada dos componentes físicos, incluindo elementos elétricos, sensores e cablagens, etc. Posteriormente, procedeu-se à análise e ao levantamento das ligações elétricas, permitindo compreender a interligação entre os diferentes módulos (sensores, atuadores, etc). Numa fase subsequente, analisaram-se as comunicações entre o PC de comando e os dois sistemas de controlo baseados em autómatos Beckhoff, permitindo identificar a natureza das solicitações operacionais, as tramas de comunicação, o comportamento global do sistema e das partes. Como resultado final do trabalho descrito neste Capítulo, foi possível delinear uma estratégia fundamentada para a substituição de partes do sistema, com especial atenção nos elementos relacionados com o software, assegurando, assim, uma futura transição sem quebra operacional do serviço.

4 Implementação Prática

A implementação prática deste projeto teve como objetivo validar, em ambiente laboratorial, uma arquitetura de comunicação alternativa para os PLCs Beckhoff utilizados no sistema AS/RS do Arquivo Automático. Tendo em conta que os PLCs instalados no sistema real se encontram totalmente operacionais e não podem sofrer alterações, optou-se por desenvolver uma réplica funcional do processo de comunicação e controlo, recorrendo ao PLC disponível no laboratório do ISEC. Este ambiente permitiu testar, passo a passo, a integração entre PLC, software de controlo, aplicações externas e interfaces gráficas, simulando o comportamento do miniload e das agulhetas.

4.1 Hardware

Para o desenvolvimento do ambiente de simulação, foi necessário os seguintes equipamentos:

- um PLC Beckhoff CX1010 disponível no laboratório do ISEC, compatível com TwinCAT 2, semelhante ao do sistema do Arquivo Automático, como se pode observar na Figura 4.1.
- um PC de desenvolvimento, no qual foram instalados os seguintes softwares:
  - TwinCAT 2 para programação IEC 61131-3, simulação e conexão ao PLC real.
  - TwinCAT 3 utilizado exclusivamente como Router AMS, devido à dificuldade em estabelecer comunicações estáveis via ADS com o router nativo do TwinCAT 2 e a biblioteca **PyAds**.
  - Python.

A escolha deste hardware permitiu replicar, em escala reduzida, o mesmo tipo de comunicação existente no sistema real do armazém Automático, o que possibilita verificar a troca de comandos, leitura de estados e a operação dos atuadores simulados.

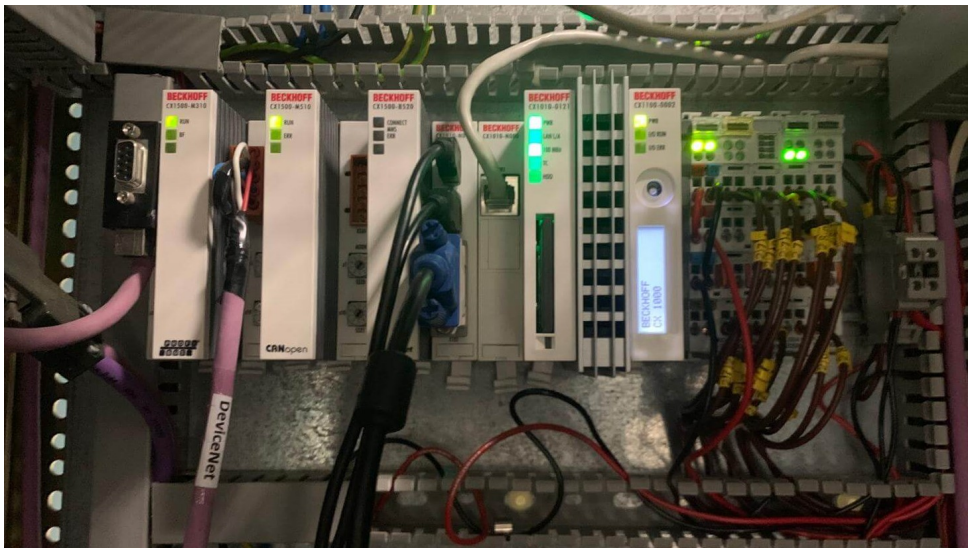


Figura 4.1: PLC Beckhoff CX1010 do Lab. de Redes Locais e Industriais do DEE/ISEC

4.2 Software

A implementação do software consistiu no desenvolvimento das quatro componentes principais:

- Programação PLC no TwinCAT 2
- Router TwinCAT 3 para comunicações AMS/ADS
- Comunicação entre PLC e Python através da biblioteca PyADS
- Desenvolvimento da interface gráfica, gestão de pedidos e API

4.2.1 Programação PLC no TwinCAT 2

No âmbito deste projeto foi desenvolvido um programa em PLC, utilizando o ambiente TwinCAT 2, com o objetivo de simular o funcionamento das agulhetas de comutação de carril, tanto na configuração reta como curva. Este programa foi concebido de forma a reproduzir, em ambiente laboratorial, a lógica de controlo existente no sistema real do Arquivo Automático.

Para permitir a comunicação com aplicações externas através do protocolo ADS, foram utilizadas variáveis mapeadas diretamente na área de memória. Este mapeamento usa os mesmos endereços do sistema real, que foram identificados com a análise das mensagens do wireshark (secção 3.1.4.2).

Foi definida a estrutura de dados **DT_Agulheta**, responsável por receber os comandos enviados pelo PC. A lógica de decisão do programa interpreta os valores recebidos e ativa ou desativa os motores das agulhetas de acordo com comandos codificados em valores hexadecimais, nomeadamente: 01, 02, 04, 08, 10 e 20, correspondendo às

diferentes configurações de cada ala.

Após o desenvolvimento, o programa foi carregado no PLC e colocado em modo Run através do TwinCAT PLC Control, permitindo a monitorização das variáveis em tempo real e a validação do correto funcionamento da lógica implementada (Figura 4.2).

O funcionamento do programa baseia-se num mecanismo semelhante ao existente no sistema real. A variável **PC_CMD.b0** recebe o código do comando, enquanto a variável **PC_CMD.b1** recebe sempre o valor fixo **12**, utilizado como verificação da validade do comando recebido. De acordo com o valor presente em **PC_CMD.b0**, o sistema executa as seguintes ações:

- **01:** é o comando de configuração reta da agulheta da Ala 2.
- **02:** é o comando de configuração curva da agulheta da Ala 2.
- **04:** é o comando de configuração reta da agulheta da Ala 3.
- **08:** é o comando de configuração curva da agulheta da Ala 3.
- **10:** é o comando de configuração reta da agulheta da Ala 4.
- **20:** é o comando de configuração curva da agulheta da Ala 4.
- **FF:** serve para limpar o valor da variável.

Cada agulheta está equipada com sensores indutivos de fim de curso, responsáveis por indicar quando a posição desejada é atingida, permitindo a paragem segura do motor. As variáveis de entrada associadas a estes sensores são:

- **B1_agulheta1**: a agulheta da Ala 2 chegou à posição reta.
- **B2_agulheta1**: a agulheta da Ala 2 chegou à posição curva.
- **B3_agulheta2**: a agulheta da Ala 3 chegou à posição reta.
- **B4_agulheta2**: a agulheta da Ala 3 chegou à posição curva.
- **B5_agulheta3**: a agulheta da Ala 4 chegou à posição reta.
- **B6_agulheta3**: a agulheta da Ala 4 chegou à posição curva.

As saídas responsáveis pelo acionamento dos motores são:

- **K1M**: motor a movimentar agulha da Ala 2 para reta.
- **K2M**: motor a movimentar agulha da Ala 2 para curva.
- **K3M**: motor a movimentar agulha da Ala 3 para reta.
- **K4M**: motor a movimentar agulha da Ala 3 para curva.
- **K5M**: motor a movimentar agulha da Ala 4 para reta.
- **K6M**: motor a movimentar agulha da Ala 4 para curva.

Como exemplo, quando o PLC recebe o comando **01:12**, a variável **PC_CMD.b0** assume

o valor 01, ativando a saída **K1M**, desde que o sensor **B1_agulheta1** se encontre a falso. O motor permanece em funcionamento até que o sensor confirme que a posição reta foi atingida, momento em que a saída é desativada e o sistema fica novamente em espera por um novo comando.

Este método garante um controlo simples, fiável e facilmente integrável com sistemas externos, permitindo validar a arquitetura de comunicação proposta sem interferir com o funcionamento do sistema real.

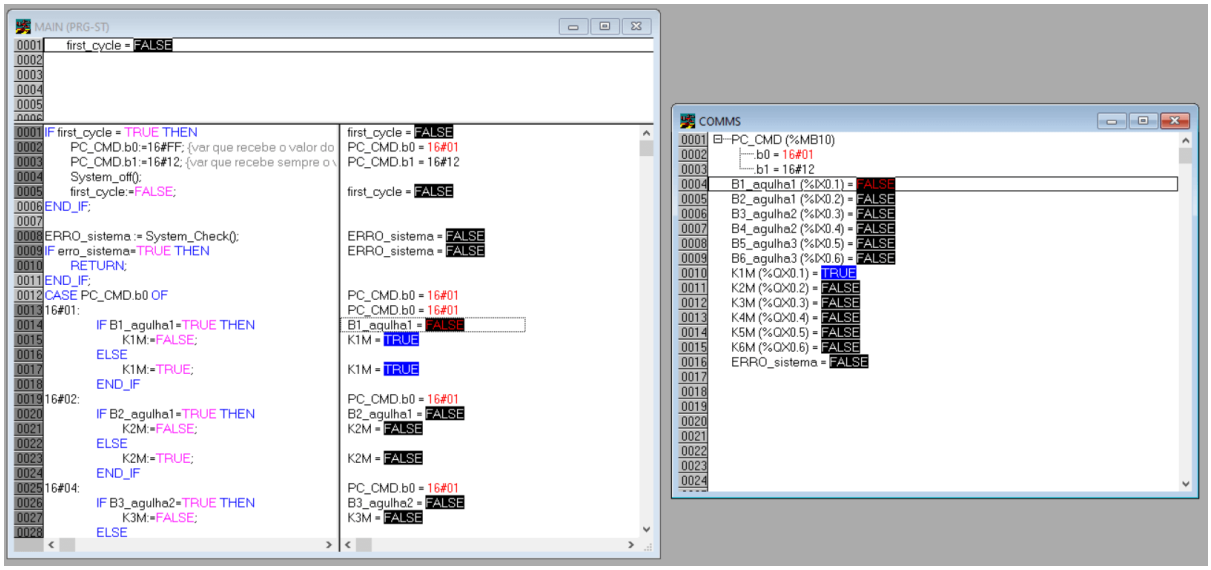


Figura 4.2: Segmento de código de comandos das agulhetas.

4.2.2 Router TwinCAT 3 para comunicações AMS/ADS

Apesar de o PLC utilizar TwinCAT 2, verificou-se que o router AMS nativo apresentava restrições na abertura de portas ADS para aplicações externas, com o *script* de python que envia comandos para o PLC. Como solução adotou-se por:

- Instalar o TwinCAT 3 XAR apenas como AMS Router no PC e configurar o AMS Router. Para configurar foi necessário editar o **AMS Net Id** e como o IP do computador Local é de 192.168.20.10, logo o **AMS Net Id** é necessário ser 192.168.20.10.1.1.
- Registar o PLC, como se pode observar na Figura 4.3, em **TwinCAT Static Routes**, pesquisar a presença do IP do PLC e adicioná-lo.

Com esta configuração, o PC passou a ser capaz de enviar comandos ADS para o PLC sem limitações, tal como observado no sistema real estudado anteriormente.

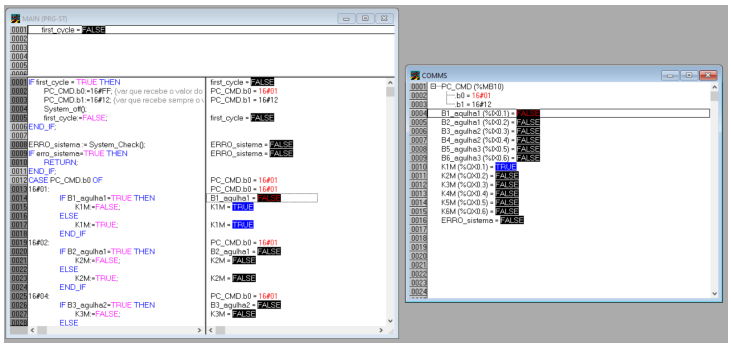


Figura 4.3: Registo do PLC no AMS Router Local.

4.2.3 Comunicação entre PLC e Python através da biblioteca PyADS

Para validar a capacidade de comandar o PLC através de aplicações externas, foi desenvolvido um script Python que:

- Utiliza a biblioteca PyADS para criar uma ligação ADS ao PLC Beckhoff.
- Escreve na memória do autómato o valor correspondente ao comando (ex.: 0x01, 0x02...).
- Confirma a alteração lendo novamente a memória para garantir a integridade.

Inicialmente, ocorreu falhas na comunicação do programa **Python** através da biblioteca PyADS, aparentemente devido a incompatibilidades com o router ADS do TwinCAT2. Foi ainda necessário reinstalar o Python localmente, com permissões administrativas.

Este mecanismo reproduz, em pequena escala, o funcionamento real entre o WMS e o OPC, assim como entre o OPC e o PLC, mas de forma mais direta e moderna, utilizando ADS sobre TCP/IP.

A comunicação entre a aplicação WMS e o software de controlo do PLC passou a ser direta, moderna e simplificada, em contraste com o sistema atualmente usado no Armazém Automático, que exige comunicação entre o WMS e o OPC, e depois entre o OPC e o software do PLC.

4.2.4 Desenvolvimento da interface gráfica, gestão de pedidos e API

Para simular uma aplicação WMS mais simples e demonstrar visualmente o movimento do miniload e das agulhetas, foi desenvolvida uma aplicação web modular composta por:

Frontend — HTML + SVG + JavaScript

- Um desenho SVG interativo representando o armazém, corredores e posição do miniload.
- Atualização dinâmica da interface sempre que ocorre alteração na base de dados.

- Comunicação com o backend através de chamadas REST.

Como se pode observar no centro da Figura 4.4, é apresentado o layout semelhante do sistema do arquivo automático, em que toda a parte das racks, das linhas de guia do miniload e a base do operador são estáticas. Mas o símbolo do miniload e o símbolo do estado das agulhetas são dinâmicos. Sempre que o estado das agulhetas e a posição do miniload variar vai ser atualizado nesta aplicação. Isso acontece devido ao script backend, estar sempre a realizar pedidos dos dados ao PLC, pelo o script de comando com a biblioteca PyAds e guarda estes valores na base de dados. Assim que são atualizados os novos dados na base de dados, o script backend vai notificar à aplicação WMS (Frontend — HTML + SVG + JavaScript) e então atualiza os símbolos dinâmicos correspondentes à mudança dos dados. Na coluna à esquerda da Figura 4.4, contêm uma simples componente de gestão de pedidos. Em que, pela ordem de superior para inferior da figura, é constituída por pedir ao miniload a busca seja pelo o número do contentor ou pelo número de produtos. Antes de fazer o pedido oficial ao PLC, é necessário validar o estado dos contentores ou produtos e para isso, é consultada a base de dados.

É necessário verificar o estado dos contentores ou produtos, pois podem não estar presentes no armazém ou até mesmo já não existir. Por isso, antes de enviar o comando oficial ao PLC, realiza-se primeiro uma consulta à base de dados para confirmar a existência do contentor ou produto e, caso exista, verificar se se encontra dentro do armazém ou com o operador.

Assim que for feito o comando e o miniload chegar à base com o contentor, passamos para a parte do check-in, que é onde se faz a confirmação que o número do contentor e os produtos dentro desse mesmo contentor, estão ambos associados, como se pode verificar na base de dados. Após o operador colocar novamente os produtos no contentor é enviado o comando para o miniload guardar o contentor dentro do armazém.

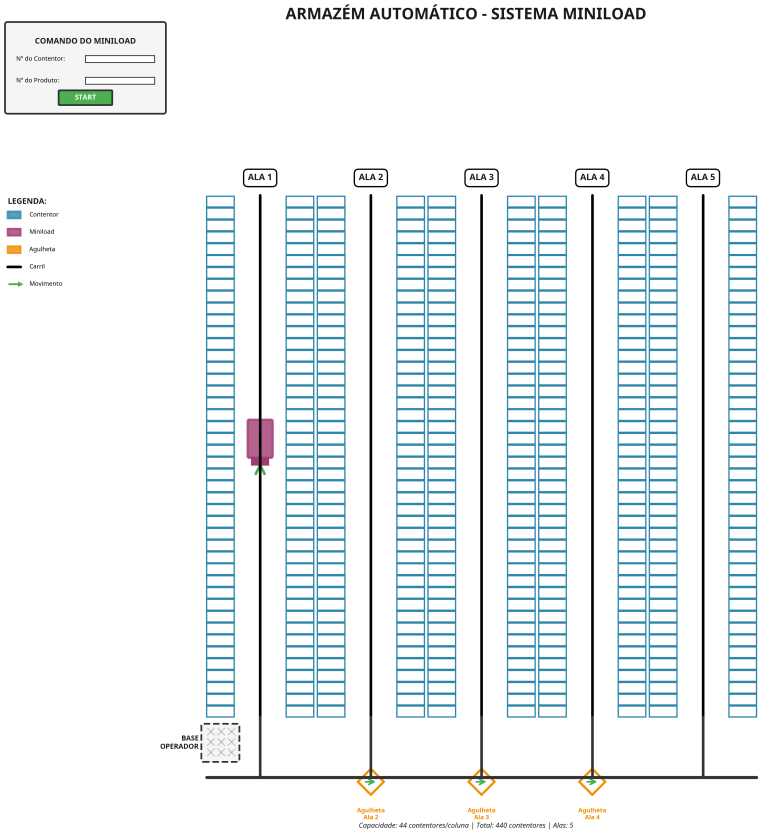


Figura 4.4: Estrutura da aplicação da WMS

As ferramentas utilizadas no desenvolvimento são descritas de seguida.

Backend — Python + Flask

- API responsável por servir dados da base de dados e receber notificações do script ADS.
- Endpoints para consultar o estado atual do miniload, comandos enviados e histórico.
- Integração direta com SQLite.

Base de dados — SQLite

- armazena eventos gerados pelo script ADS, incluindo:
  - comandos enviados,
  - timestamp de execução,
  - valor escrito no PLC,
  - estados lidos após o comando.
- permite que o frontend atualize automaticamente a animação do armazém.

4.3 Integração

Os vários componentes foram interligados de forma a permitir uma operação coerente e contínua: comandos enviados pelo script Python são propagados ao PLC, registados na base de dados e refletidos graficamente no frontend. Esta integração confirmou a viabilidade tecnológica de um sistema modular de supervisão e controlo baseado em ADS e tecnologias web.

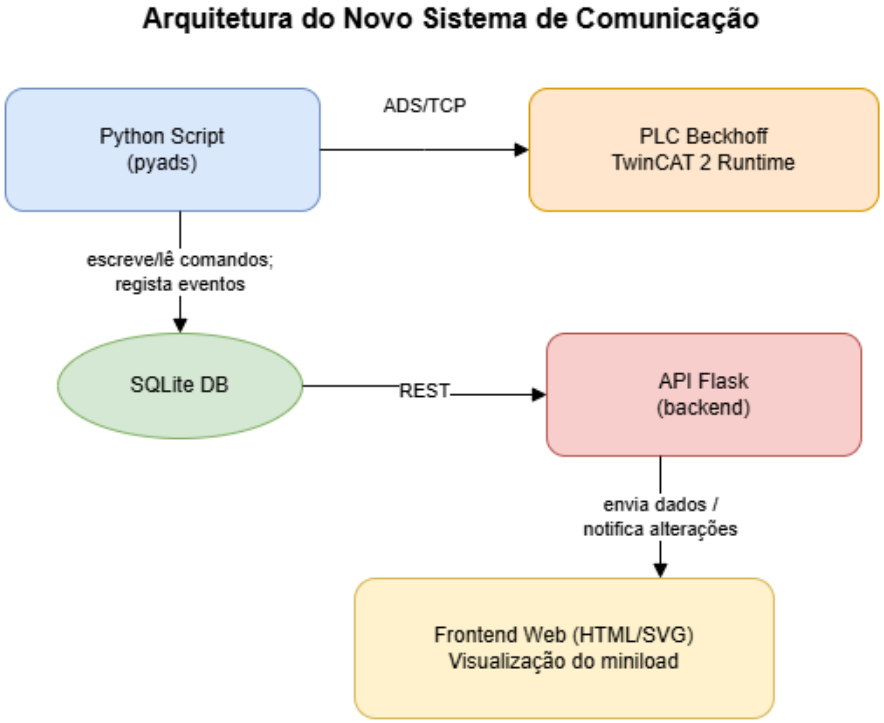


Figura 4.5: Nova arquitetura de WMS

4.4 Conclusão

Neste Capítulo, apresentou-se o trabalho prático desenvolvido em ambiente de testes com condições semelhantes às do ambiente real, o que permitiu validar a solução antes da sua implementação no sistema efetivo, permitindo garantir a continuidade de serviço do sistema existente, quando se aplicarem as alterações. As melhorias implementadas incluem a utilização de ferramentas de software open-source, tanto no nível do sistema de decisão quanto no do sistema de comunicações. Outra vantagem significativa é a possibilidade de, no futuro, realizarem-se alterações na aplicação de forma mais flexível, incluindo na programação do PLC.

5 Conclusões

5.1 Enquadramento Final

O presente trabalho de projeto teve como principal objetivo a análise e a proposta de melhorias a um sistema automático de gestão e armazenamento de acervos do tipo miniload, pertencente à Universidade de Coimbra e em funcionamento há cerca de duas décadas. A motivação para este trabalho resultou da identificação de diversas limitações técnicas e funcionais observadas num sistema real em operação, nomeadamente ao nível da arquitetura de controlo, das comunicações industriais e da integração com soluções de software modernas.

5.2 Síntese do Trabalho Desenvolvido

Numa fase inicial, foi realizada uma análise detalhada do sistema existente, abrangendo a componente física do armazém automático, os equipamentos eletromecânicos, o sistema de controlo baseado em PLC e a infraestrutura de software associada. Esta análise permitiu compreender o funcionamento global do sistema e identificar fragilidades decorrentes da antiguidade das tecnologias utilizadas, da complexidade da manutenção e da reduzida flexibilidade para integração com sistemas de nível superior.

Com base no estudo efetuado, foram definidos os requisitos funcionais e não funcionais do sistema e apresentadas propostas de melhoria orientadas para a modernização da arquitetura existente. As soluções propostas incidiram essencialmente na reorganização do sistema de controlo, na melhoria das comunicações entre o PLC e o software de gestão e na adoção de tecnologias mais atuais, amplamente utilizadas na automação industrial. O trabalho contemplou ainda a implementação prática de parte destas melhorias, recorrendo a controladores lógicos programáveis Beckhoff, ao ambiente de desenvolvimento TwinCAT e à integração com aplicações desenvolvidas em Python, permitindo validar a viabilidade técnica das soluções apresentadas.

5.3 Avaliação dos Objetivos

De um modo geral, os objetivos definidos para o presente trabalho foram alcançados. A análise do sistema permitiu identificar de forma clara as principais limitações técnicas

e funcionais da arquitetura existente, enquanto as melhorias propostas responderam adequadamente aos problemas identificados. As soluções implementadas contribuíram para um sistema mais simples, robusto e flexível, alinhado com os princípios da automação industrial moderna, criando condições favoráveis para a sua manutenção, expansão futura e integração com sistemas de nível superior.

Referências bibliográficas

[1] Ltd, "Princípios e características do as/rs," Jan. 2024. [Online]. Available: https://pt.kmstoragear.com/news/principles-and-characteristics-of-as-rs-74342515.html

[2] [Online]. Available: https://www.conveyco.com/technology/asrs/unit-load-asrs/?_page=2

[3] H. M. Hameed, *The Automatic Storage and Retrieval System: An overview*, 2017.

[4] [Online]. Available: https://www.gieicom.com/en/products/automated-storage/unitload-asrs/

[5] [Online]. Available: https://www.bastiansolutions.com/solutions/technology/asrs/mini-load/?srsltid=AfmBOorqVCnRgNBJcX02LTK_i7nYVYy0r669Y9O8aI4ZLLU-5doo5ElQ

[6] R. Gaku and S. Takakuwa, "Simulation analysis of large-scale shuttle vehicle-type mini-load as/rs systems," in *2018 Winter Simulation Conference (WSC)*. IEEE, 2018.

[7] 2022. [Online]. Available: https://www.ar-racking.com/en/storage-systems/automated-warehouses/cartons/mini-load-storage-systems/

[8] L. Nicolas, F. Yannick, and H. Ramzi, "Order batching in an automated warehouse with several vertical lift modules: Optimization and experiments with real data," *European journal of operational research*, vol. 267, no. 3, p. 958–976, 2018. [Online]. Available: http://dx.doi.org/10.1016/j.ejor.2017.12.037

[9] [Online]. Available: https://www.wolterinc.com/industrial-storage-handling/automated-industrial-storage/vertical-lift-modules/

[10] J. Custódio and R. Machado, *Impacts of implementing a high-density automated storage and retrieval system in a warehouse*, 2023.

[11] M. Kaur and V. Kumar, "Development of automated storage and retrieval system (as/rs) using carousel systems," *International Journal of Engineering Research and Applications*, vol. 12, no. 7, p. 67–72, 2022.

[12] Mecalux, "Armazéns verticais e carrosséis verticais ou horizontais." [Online]. Available: https://www.mecalux.pt/manual-de-armazenagem/sistemas-de-armazenagem/armazem-vertical-carrossel-horizontal

[13] B. Rouwenhorst, J. P. Van Den, G. J. J. A. N. Van, and W. H. M. Zijm, *Performance evaluation of a carousel system*. Braun-Brumfield inc, 1996, p. 495–511.

[14] [Online]. Available: https://www.archiexpo.com/pt/prod/kardex/product-9012-1865192.html

[15] V. Shinde and R. Patil, "Study and comparison of automated material handling systems: Carousel system and as/rs," *International Journal of Current Engineering and Technology*, vol. 11, no. 2, p. 201–204, 2021.

[16] A. Döllinger, *Selection of Automated Order Picking Systems* (*Master's thesis*), 2015.

[17] K. J. Roodbergen and I. F. A. Vis, "A survey of literature on automated storage and retrieval systems," *European journal of operational research*, vol. 194, no. 2, p. 343–362, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.ejor.2008.01.038

[18] Mecalux, "Armazéns verticais e carrosséis verticais ou horizontais." [Online]. Available: https://www.mecalux.pt/manual-de-armazenagem/sistemas-de-armazenagem/armazem-vertical-carrossel-horizontal

[19] Y. Li and Z. Li, "Shuttle-based storage and retrieval system: A literature review," *Sustainability*, vol. 14, no. 21, p. 14347, 2022. [Online]. Available: http://dx.doi.org/10.3390/su142114347

[20] M. Eder, "Analytical examination of shuttle-based storage and retrieval systems with multiple-capacity lifts," *The international journal of advanced manufacturing technology*, vol. 133, no. 9–10, p. 5053–5064, 2024. [Online]. Available: http://dx.doi.org/10.1007/s00170-024-14058-w

[21] Mecalux, "Robôs móveis autónomos (amr)." [Online]. Available: https://www.mecalux.pt/armazens-automaticos/amr-robos-moveis-autonomos

[22] Jan. 2025. [Online]. Available: https://igps.net/asrs-vs-agv/

[23] Jan. 2025. [Online]. Available: https://www.autostoresystem.com/insights/the-complete-guide-to-warehouse-automation

[24] G. Mareeswaran.B, K. Kumar, M. Karthik, and P. Ram, "Plc based automated storage and retrieval system with a robotic end effector," *International journal of engineering trends and technology*, vol. 67, no. 7, p. 11–14, 2019. [Online]. Available: http://dx.doi.org/10.14445/22315381/ijett-v67i7p203

[25] C.-x. Cx, "Cx1000 embedded pc hardware documentation." [Online]. Available: https://download.beckhoff.com/download/document/ipc/embedded-pc/embedded-pc-cx/cx1000_hwen.pdf

[26] B. A. GmbH, K. G. Co, Hülshorstweg, . Verl, and Germany, "Embedded pc with arm® cortex®-m7 and integrated i/os." [Online].

Available: https://www.beckhoff.com/en-en/products/ipc/embedded-pcs/cx7000-arm-r-cortex-r/cx7000.html

[27] [Online]. Available: https://cache.industry.siemens.com/dl/files/967/109962967/att_1333900/v1/SEP00021_SIMATIC_S7-1500_CPU_Siemens_EcoTech_Profile_V2.0.pdf

[28] U. Manual and O. Instructions, "Important: This manual links to logix 5000 controller and i/o fault codes, 17566-rd001; download the spreadsheet now for offline access." [Online]. Available: https://literature.rockwellautomation.com/idc/groups/literature/documents/um/5069-um001_-en-p.pdf

[29] [Online]. Available: https://www.se.com/pt/pt/product-range/65771-controlador-iiot-modicon-m262/#products

[30] S. Vongbunyong, P. Roengritronnachai, S. Kongsanit, C. Chanok-owat, and P. Polchankajorn, "Multiple products management system with sensors array in automated storage and retrieval systems," *IOP conference series. Materials science and engineering*, vol. 297, p. 012052, 2018. [Online]. Available: http://dx.doi.org/10.1088/1757-899x/297/1/012052

[31] T. R. Doebbert, C. Cammin, and G. Scholl, "Safety architecture proposal for low-latency sensor/actuator networks using io-link wireless," *IEEE access: practical innovations, open solutions*, vol. 10, p. 3030–3044, 2022. [Online]. Available: http://dx.doi.org/10.1109/access.2021.3128758

[32] R. Heynicke, D. Krush, C. Cammin, G. Scholl, B. Kaercher, J. Ritter, P. Gaggero, and M. Rentschler, "Io-link wireless enhanced factory automation communication for industry 4.0 applications," *Journal of sensors and sensor systems*, vol. 7, no. 1, p. 131–142, 2018. [Online]. Available: http://dx.doi.org/10.5194/jsss-7-131-2018

[33] D. Wolberg, M. Rentschler, and P. Gaggero, "Simulative performance analysis of io-link wireless," in *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, 2018.

[34] Y. Theopilus, S. S. Tjandra, and B. Sagara, "Development of low-cost multi-input automated storage and retrieval system (as/rs) for educational purposes," *IOP conference series. Materials science and engineering*, vol. 847, no. 1, p. 012072, 2020. [Online]. Available: http://dx.doi.org/10.1088/1757-899x/847/1/012072

[35] H. A. Abbas, "Efficient web-based scada system," 2015. [Online]. Available: http://arxiv.org/abs/1501.05725

[36] H. Abbas, S. Shaheen, and M. Amin, "Simple, flexible, and interoperable scada system based on agent technology," *Intelligent control and automation*, vol. 06,

no. 03, p. 184–199, 2015. [Online]. Available: http://dx.doi.org/10.4236/ica.2015.63018

[37] I. R. S. Agostino, C. Ristow, and C. M. T. Rodriguez, "Internet das coisas em sistemas logísticos: revisão da literatura recente e perspectivas de pesquisa," *Exacta*, vol. 19, no. 2, p. 251–275, 2021. [Online]. Available: http://dx.doi.org/10.5585/exactaep.2021.15999

[38] A. Jarašūnienė, K. Čižiūnienė, and A. Čereška, "Research on impact of iot on warehouse management," *Sensors (Basel, Switzerland)*, vol. 23, no. 4, p. 2213, 2023. [Online]. Available: http://dx.doi.org/10.3390/s23042213

[39] B. Tomar, N. Kumar, and M. Sreejeth, "Real time automation and ratio control using plc scada in industry 4.0," *Computer systems science and engineering*, vol. 45, no. 2, p. 1495–1516, 2023. [Online]. Available: http://dx.doi.org/10.32604/csse.2023.030635

[40] N. Boysen, R. de Koster, and F. Weidinger, "Warehousing in the e-commerce era: A survey," *European journal of operational research*, vol. 277, no. 2, p. 396–411, 2019. [Online]. Available: http://dx.doi.org/10.1016/j.ejor.2018.08.023

[41] [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/863272/Heikkila_Daniel.pdf?sequence=3

[42] W. Uriawan, A. R. Rifa'i, A. Husen, A. O. Hamza, A. M. Firdaus, A. F. Dinah, and A. A. H. Youztima, "Swif: Warehouse automation system using agile method," 2024. [Online]. Available: http://dx.doi.org/10.20944/preprints202407.0065.v1

[43] W.-T. Sung and C.-Y. Lu, "Smart warehouse management based on iot architecture," in *2018 International Symposium on Computer, Consumer and Control (IS3C)*. IEEE, 2018.

[44] D. Li, L. Wang, S. Geng, and B. Jiang, "Path planning of as/rs based on cost matrix and improved greedy algorithm," *Symmetry*, vol. 13, no. 8, p. 1483, 2021. [Online]. Available: http://dx.doi.org/10.3390/sym13081483

[45] R. Manzini, M. Gamberi, and A. Regattieri, "Design and control of an as/rs," *The international journal of advanced manufacturing technology*, vol. 28, no. 7–8, p. 766–774, 2006. [Online]. Available: http://dx.doi.org/10.1007/s00170-004-2427-6

[46] D. O. Sousa, "Desenvolvimento de algoritmos para recolha de produtos num armazém automático de alta densidade," 2020. [Online]. Available: https://repositorium.uminho.pt/entities/publication/14ab8446-7d7e-40a5-a7e2-aa61045def83

[47] [Online]. Available: http://researchgate.net/publication/368699309_Warehouse_layout_design_with_class-based_storage_approach_to_minimize_material_transfer_distance?enrichId=

rgreq-23264b8cace0b97ff7364ae718ea7933-XXX&enrichSource=
Y292ZXJQYWdlOzM2ODY5OTMwOTtBUzoxMTQzMTI4MTEyNzc3NzQzMUAxNjc5MTUzN
el=1_x_3&_esc=publicationCoverPdf

[48] M. Kordos, J. Boryczko, M. Blachnik, and S. Golak, "Optimization of warehouse operations with genetic algorithms," *Applied sciences (Basel, Switzerland)*, vol. 10, no. 14, p. 4817, 2020. [Online]. Available: http://dx.doi.org/10.3390/app10144817

[49] X. Yan, Z. Zhang, Q. Liu, C. Lv, L. Zhang, and S. Li, "An nsabc algorithm for multi-aisle as/rs scheduling optimization," *Computers industrial engineering*, vol. 156, no. 107254, p. 107254, 2021. [Online]. Available: http://dx.doi.org/10.1016/j.cie.2021.107254

[50] J. Lu, L. Xu, J. Jin, and Y. Shao, "A mixed algorithm for integrated scheduling optimization in as/rs and hybrid flowshop," *Energies*, vol. 15, no. 20, p. 7558, 2022. [Online]. Available: http://dx.doi.org/10.3390/en15207558

[51] [Online]. Available: https://infosys.beckhoff.com/english.php?content=../content/1033/tcadscommon/12440283915.html&id=

Anexos

Anexo A- Lista de dispositivos do Quadro Elétrico da sala do operador - sistema de agulhagem

Dispositivo	Quantidade	Descrição	Referência	Fabricante	Código do artigo	Família
A1	1	CPU para PLC	CX1001-0111	Beckhoff	000002994	H012
A2	1	Módulo de alimentação para CPU CX1000	CX1100-0002	Beckhoff	000002996	H709
A3	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff	000001204	H012
A4	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff	000001204	H012
A5	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff	000001204	H012
A6	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff	000001204	H012
A7	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff	000001204	H012
A8	1	Módulo alimentação terminal c/ fus 24VDC	KL9210	Beckhoff	000003105	H012
A9	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff	000002993	H012
A10	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff	000002993	H012
A11	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff	000002993	H012
A12	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff	000002993	H012
A13	1	Módulo de terminação	KL9010	Beckhoff	000001206	H012
	1	Módulo de 4 saidas digitais de 24VDC	KL2134	Beckhoff	000001205	H012
B1	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000001422	H303
B2	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003052	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000003061	H303
B3	1	Detetor indutivo M18 (2mm)	XS5-12B1PAM12	Telemecanique	000003061	H303
	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000001422	H502
B4	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000001422	H303
B5	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000001422	H303
B6	1	Detetor indutivo M12 (2mm)	XS5-12B1PAM12	Telemecanique	000003052	H303
	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
B7	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000001422	H303
B8	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
	1	Detetor indutivo M18 (5mm)	NBB5-18GM50-50-E2-V1	Pepperl	000001422	H303
B9	1	Detetor indutivo M18 (2mm)	XS5-12B1PAM12	Telemecanique	000003052	H303
	1	Conector M12, cotovelo	XZ-CC12FCM40B	Telemecanique	000003061	H502
F1	1	Fusível cilindrico 10,3x38 6A gG	133 06	Legrand	000002770	H8310
	1	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 porta-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	1	Porta-fusíveis modular 10x38 1N	DF6-N10	Telemecanique	000001227	H8311
F2	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 portas-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
F3	1	Fusível cilindrico 10,3x38 6A gC	133 06	Legrand	000002770	H8310
	1	Porta-fusíveis modular 10x38 1P	DF6-N10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 portas-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	1	Porta-fusíveis modular 10x38 1P	DF6-N10	Telemecanique	000001227	H8311
F4	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 portas-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
F5	1	Fusível cilindrico 10,3x38 6A gG	133 06	Legrand	000002770	H8310
	1	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 portas-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	1	Porta-fusíveis modular 10x38 1N	DF6-N10	Telemecanique	000001227	H8311
F6	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
	1	Barra p/ associação de 2 portas-fus. DF€	GK1-AP2	Telemecanique	000000259	H8311
	2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
FR1	1	Anel de ferrite	HD002	SEW	000001469	H13
FR2	1	Anel de ferrite	HD002	SEW	000001469	H13
FR3	1	Anel de ferrite	HD002	SEW	000001469	H13
G1	1	Fonte de alimentação 230VAC/24DC - 2,5A	QUINT-PS-100-240AC/24DC/2,5	Phoenix	000002978	H709
H1	1	Sinalizador completo LED verde 24Vdc	XB5-AVB3	Telemecanique	000002678	H810
H2	1	Sinalizador completo LED verde 24Vdc	XB5-AVB3	Telemecanique	000002678	H810
H3	1	Sinalizador completo LED amarelo 24Vdc	XB5-AVB5	Telemecanique	000002680	H810
H4	1	Sinalizador completo LED vermelho 24Vdc	XB5-AVB4	Telemecanique	000002679	H810
H5	1	Sinalizador completo LED verde 24Vdc	XB5-AVB3	Telemecanique	000002678	H810
K1A	1	Relé de segurança 24VCA/CC 3NA Cat. 4	XPS-AF5130	Telemecanique	000002986	H830
K1M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LC1-D18-DB (24VDC)	Telemecanique	000001040	H834
K2M	1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique	000000270	H834
	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LP1-K0910BD (24VDC)	Telemecanique	000002815	H834
K3M	1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique	000000270	H834
	1	Contactor 3P+NA (7,5W/400V/AC3/9A)	LP1-K0910BD (24VDC)	Telemecanique	000002815	H834
K4M	1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique	000000270	H834
	1	Contactor 3P+NA (7,5W/400V/AC3/9A)	LP1-K0910BD (24VDC)	Telemecanique	000002815	H834
M1	1	Motor-redutor c/ motor assinc. c/ freio	WF10 DT56L4-BR/HR	SEW	G603-SEW-WF10-DT56L4-0001	G603
M2	1	Motor-redutor c/ motor assinc. c/ freio	WF10 DT56L4-BR/HR	SEW	G603-SEW-WF10-DT56L4-0002	G603
M3	1	Motor-redutor c/ motor assinc. c/ freio	WF10 DT56L4-BR/HR	SEW	G603-SEW-WF10-DT56L4-0003	G603
N1	1	Ponto acesso "Wireless Ethernet" c/ ant	SB2510 / TQJ-2400A	Moxa	M81-25-002-A	M81
N2	1	Variador velocidade 1F/200V/0,2kW	3G3MV-AB002-GBR	Omron	000003537	H829
N3	1	Variador velocidade 1F/200V/0,2kW	3G3MV-AB002-GBR	Omron	000003537	H829
N4	1	Variador velocidade 1F/200V/0,2kW	3G3MV-AB002-GBR	Omron	000003507	H829
Q1	1	Bloco Diferencial 4P<25A/300mA	26533	Merlin Gerin	000001027	H8320
	1	Disjuntor C60H-Curva C-4P-4D-25A	25015	Merlin Gerin	000001579	H8321
	1	Mecanismo para manípulo rotativo	27046	Merlin Gerin	000000447	H8320
	1	Manípulo rotativo lateral	27048	Merlin Gerin	000000448	H8320
Q2	1	Disjuntor Magnetotérmico F+N 6kA C (6A)	21555	Merlin Gerin	000003413	H8320
Q3	1	Disjuntor Magnetotérmico F+N 6kA C (3A)	21554	Merlin Gerin	000002775	H8320
S1	1	Bloco de contactos 2NF	ZB5-AZ104	Telemecanique	000000365	H803
	1	Botoneira PVC c/ 2 furos	XAL-D02	Telemecanique	000001524	H514
	1	Botão verm. Cab. cog. Rodar para desenc.	ZB5-AS44	Telemecanique	000001233	H803
S2	1	Bloco de contactos 1NA	ZB5-AZ101	Telemecanique	000000209	H803
	1	Botão com chave de 2 posições fixas	ZB5-AG4	Telemecanique	000000303	H803

Página 1 de 2

Figura A.1: Listagem excel dos dispositivos existentes no sistema de agulhagem (1/2)

S3	1	Bloco de contactos 2NF	ZB5-AZ104	Telemecanique	000000365	H803
	1	Botão verm. Cab. cog. Rodar para desenc.	ZB5-AS54	Telemecanique	000000368	H803
	1	Etiqueta Circular "EMERGENCY-STOP"	ZBY-8330	Telemecanique	000001155	H803
S4	1	Actuador do Interruptor p/ porta	AZ 15/16-B2	Schmersal	000000076	H304
	1	Interruptor segurança p/ porta (2NF+1NA)	AZ 16-12 zvrk-M16	Schmersal	000001942	H304
S5	1	Bloco de contactos 1NA	ZB5-AZ101	Telemecanique	000000209	H803
	1	Botão de impulso branco	ZB5-AA1	Telemecanique	000000305	H803
S6	1	Botão de impulso branco	ZB5-AA1	Telemecanique	000000305	H803
	1	Bloco de contactos 1NA	ZB5-AZ101	Telemecanique	000000209	H803
S7	1	Botão c/ chave 2 pos. Fixas (ch 421E)	ZB5-AG212	Telemecanique	000001988	H803
	1	Bloco de contactos 1NA	ZB5-AZ101	Telemecanique	000000209	H803
S8	1	Micro-Interruptor (roldana a 90º)	ZC-Q2155	Omron	000000905	H304
W1	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W2	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W3	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W4	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W5	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W6	0	Cabo H1RV-K 4G4	H1RV-K 4G4		000000249	F2125
W7	0	Cabo JZ-500 3G1,5	JZ-500 3G1,5	Helukabel	000002084	F2125
W8	0	Cabo JZ-500 3G0,5	JZ-500 7G0,5	Helukabel	000001586	F2125
W9	0	Cabo JZ-500 3G0,5	JZ-500 5G0,5	Helukabel	000000247	F2125
W10	0	Cabo H1RV-K 4G1,5	H1RV-K 4G1,5		000002695	F2125
W11	0	Cabo H1RV-K 3G1,5	H1RV-K 3G1,5		000002748	F2125
W12	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W13	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W14	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W15	0	Cabo H1RV-K 4G1,5	H1RV-K 4G1,5		000002695	F2125
W16	0	Cabo H1RV-K 3G1,5	H1RV-K 3G1,5		000002748	F2125
W17	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W18	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W19	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W20	0	Cabo H1RV-K 4G1,5	H1RV-K 4G1,5		000002695	F2125
W21	0	Cabo H1RV-K 3G1,5	H1RV-K 3G1,5		000002748	F2125
W22	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W23	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W24	0	Cabo OZ-500 3x0,5	OZ-500 3x0,5	Helukabel	000000244	F2125
W25	1	Cabo JZ-500 3G0,5	JZ-500 3G0,5	Helukabel	000000243	F2125
XI	1	Tomada 2P+T tipo "Schunko"	916 41	Legrand	000000069	H500

Página 2 de 2

Figura A.2: Listagem excel dos dispositivos existentes no sistema de agulhagem (2/2)

Anexo B- Lista de dispositivos do Quadro Elétrico do miniload - robô móvel

Local	Dispositivo	Quantidade	Descrição	Referência	Fabricante	Código do artigo	Família
Q.E.	A1	1	CPU para PLC	CX1500-M310	Beckhoff		
Q.E.		1		CX1000-N001	Beckhoff		
Q.E.				CX1000-N000	Beckhoff		
Q.E.	A2	1	CPU para PLC	CX1001-0111	Beckhoff		
Q.E.				CX1000-COOL	Beckhoff		
Q.E.	A3	1	Módulo de alimentação para CPU CX1000	CX1100-0002	Beckhoff		
Q.E.				s/ referencias	Beckhoff		
Q.E.	A4	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A5	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A6	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A7	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A8	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A9	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A10	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A11	1	Módulo alimentação terminal c/ fus 24VDC	KL9210	Beckhoff		
Q.E.	A12	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.	A13	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.	A14	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.			Módulo de terminação	KL9010	Beckhoff		
miniload	B1	1	inductive sensor XS5 M12	XS518B1PAM12			
miniload	B2	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B3	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B4	1	sensor magnetico (switchs)	BN 85-S-2031	SCHMERSAL		
miniload	B5	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B6	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B7	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B8	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B9	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B10	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B11	1	Inductive proximity sensors	XS512B1PAM12	Telemecanique		
miniload	B12	1	Inductive proximity sensors	XS512B1PAM12	Telemecanique		
miniload	B13	1	Fotocélula	VL180-2P42433	SICK		
miniload	B14	1	Fotocélula	VL180-2P42433	SICK		
miniload	B15	1	Fotocélula	VL18-4P3340	SICK		
miniload	B16	1	Fotocélula	XUB5 BPAWM12	Telemecanique		
miniload	B17	1	Fotocélula	XUB5 BPAWM12	Telemecanique		
miniload	B18	1	Fotocélula	VL18-4P3340	SICK		
miniload	B19	1	Fotocélula	VL18-4P3340	SICK		
Q.E.	E1	1	Ventilador				
Q.E.	F1	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F2	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F3	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F4	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F5	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F6	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F7	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilíndrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	FR1	1	Anel de ferrite	HD002	SEW		
Q.E.	FR2	1	Anel de ferrite	HD002	SEW		
Q.E.	FR3	1	Anel de ferrite	HD002	SEW		
Q.E.	G1	1	Fonte de alimentação 230VAC/24DC - 2,5A	QUINT-PS-100-240AC/24DC/2,5	Phoenix Contact		
Q.E.	H1L	1	Lâmpadas T8, 60mm	L 18W/25	OSRAM		
Q.E.	H1L	3	"Pirilâmpos"	XVD C33...35	Telemecanique		
Q.E.	H2	1	Sinalizador completo LED verde 24Vdc	XB5-AVB3	Telemecanique		
Q.E.	H3	1	Sinalizador completo LED vermelho 24Vdc	XB5-AVB4	Telemecanique		
Q.E.	H4	1	Sinalizador completo LED amarelo 24Vdc	XB5-AVB5	Telemecanique		
Q.E.	K1A	1	Relé de segurança 24VCA/CC 3NA Cat. 4	XPS-AF5130	Telemecanique		
Q.E.	K2A	1	Relé de 24VDC/	RSB 2A080BD	Telemecanique		
Q.E.	K1M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LC1-D18-DB (24VDC)	Telemecanique		
Q.E.	K2M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LP1-K0610BD (24VDC)	Telemecanique		
Q.E.		1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique		
Q.E.	K3M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LA1-KN20 (24VDC)	Telemecanique		
Q.E.		1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique		
miniload	M1	1	Motor-redutor c/ motor assinc. c/ freio (motor para roda da frente)	KH37 DT80N4/BMG/WR/TF/ES1S	SEW		
miniload	M2	1	Motor-redutor c/ motor assinc. c/ freio (motor para roda de trás)		SEW		
miniload	M3	1	Motor-redutor c/ motor assinc. c/ freio (motor para mover suporte em Y)		SEW		
miniload	M4	1	Motor-redutor c/ motor assinc. c/ freio (motor dos garfos)	RF17 DT71D4/BMG/HR/MM03	SEW		
Q.E.	N1	1	VEV(no quadro)		SEW		
Q.E.	N2	1	VEV(no quadro)		SEW		

Página 1 de 2

Figura B.1: Listagem excel dos dispositivos existentes no miniload (1/2)

Local	Dispositivo	Quantidade	Descrição	Referência	Fabricante	Código do artigo	Família
Q.E.	A1	1	CPU para PLC	CX1500-M310	Beckhoff		
Q.E.		1		CX1000-N001	Beckhoff		
Q.E.				CX1000-N000	Beckhoff		
Q.E.	A2	1	CPU para PLC	CX1001-0111	Beckhoff		
Q.E.				CX1000-COOL	Beckhoff		
Q.E.	A3	1	Módulo de alimentação para CPU CX1000	CX1100-0002	Beckhoff		
Q.E.				s/ referencias	Beckhoff		
Q.E.	A4	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A5	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A6	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A7	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A8	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A9	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A10	1	Módulo de 4 entradas digitais de 24 VDC	KL1114	Beckhoff		
Q.E.	A11	1	Módulo alimentação terminal c/ fus 24VDC	KL9210	Beckhoff		
Q.E.	A12	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.	A13	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.	A14	1	Módulo de 4 entradas digitais de 24 VDC	KL2404	Beckhoff		
Q.E.			Módulo de terminação	KL9010	Beckhoff		
miniload	B1	1	inductive sensor XS5 M12	XS518B1PAM12			
miniload	B2	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B3	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B4	1	sensor magnetico (switchs)	BN 85-S-2031	SCHMERSAL		
miniload	B5	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B6	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B7	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B8	1	inductive sensor XS5 M18	XS530B1PAM12	Telemecanique		
miniload	B9	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B10	1	magnet driver	BN 65-RZ/V	SCHMERSAL		
miniload	B11	1	Inductive proximity sensors	XS512B1PAM12	Telemecanique		
miniload	B12	1	Inductive proximity sensors	XS512B1PAM12	Telemecanique		
miniload	B13	1	Fotocélula	VL180-2P42433	SICK		
miniload	B14	1	Fotocélula	VL180-2P42433	SICK		
miniload	B15	1	Fotocélula	VL18-4P3340	SICK		
miniload	B16	1	Fotocélula	XUB5 BPAWM12	Telemecanique		
miniload	B17	1	Fotocélula	XUB5 BPAWM12	Telemecanique		
miniload	B18	1	Fotocélula	VL18-4P3340	SICK		
miniload	B19	1	Fotocélula	VL18-4P3340	SICK		
Q.E.	E1	1	Ventilador				
Q.E.	F1	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F2	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F3	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F4	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F5	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F6	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	F7	2	Porta-fusíveis modular 10x38 1P	DF6-AB10	Telemecanique	000000258	H8311
Q.E.		1	Barra p/ associação de 2 portas-fus. DF6	GK1-AP2	Telemecanique	000000259	H8311
Q.E.		2	Fusível cilindrico 10,3x38 2A aM	130 02	Legrand	000000264	H8310
Q.E.	FR1	1	Anel de ferrite	HD002	SEW		
Q.E.	FR2	1	Anel de ferrite	HD002	SEW		
Q.E.	FR3	1	Anel de ferrite	HD002	SEW		
Q.E.	G1	1	Fonte de alimentação 230VAC/24DC - 2,5A	QUINT-PS-100-240AC/24DC/2,5	Phoenix Contact		
Q.E.	H1L	1	Lâmpadas T8, 60mm	L 18W/25	OSRAM		
Q.E.	H1L	3	"Pirilâmpos"	XVD C33...35	Telemecanique		
Q.E.	H2	1	Sinalizador completo LED verde 24Vdc	XB5-AVB3	Telemecanique		
Q.E.	H3	1	Sinalizador completo LED vermelho 24Vdc	XB5-AVB4	Telemecanique		
Q.E.	H4	1	Sinalizador completo LED amarelo 24Vdc	XB5-AVB5	Telemecanique		
Q.E.	K1A	1	Relé de segurança 24VCA/CC 3NA Cat. 4	XPS-AF5130	Telemecanique		
Q.E.	K2A	1	Relé de 24VDC/	RSB 2A080BD	Telemecanique		
Q.E.	K1M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LC1-D18-DB (24VDC)	Telemecanique		
Q.E.	K2M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LP1-K0610BD (24VDC)	Telemecanique		
Q.E.		1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique		
Q.E.	K3M	1	Contactor 3P+NA (7,5W/400V/AC3/18A)	LA1-KN20 (24VDC)	Telemecanique		
Q.E.		1	Circuito Diodo+Diodo Zener 12...24DC	LA4-KC1B	Telemecanique		
miniload	M1	1	Motor-redutor c/ motor assinc. c/ freio (motor para roda da frente)	KH37 DT80N4/BMG/WR/TF/ES1S	SEW		
miniload	M2	1	Motor-redutor c/ motor assinc. c/ freio (motor para roda de trás)		SEW		
miniload	M3	1	Motor-redutor c/ motor assinc. c/ freio (motor para mover suporte em Y)		SEW		
miniload	M4	1	Motor-redutor c/ motor assinc. c/ freio (motor dos garfos)	RF17 DT71D4/BMG/HR/MM03	SEW		
Q.E.	N1	1	VEV(no quadro)		SEW		
Q.E.	N2	1	VEV(no quadro)		SEW		

Página 1 de 2

Figura B.2: Listagem excel dos dispositivos existentes no miniload (2/2)

Anexo C- Esquemas do Quadro Elétrico da sala do operador - sistema de agulhagem

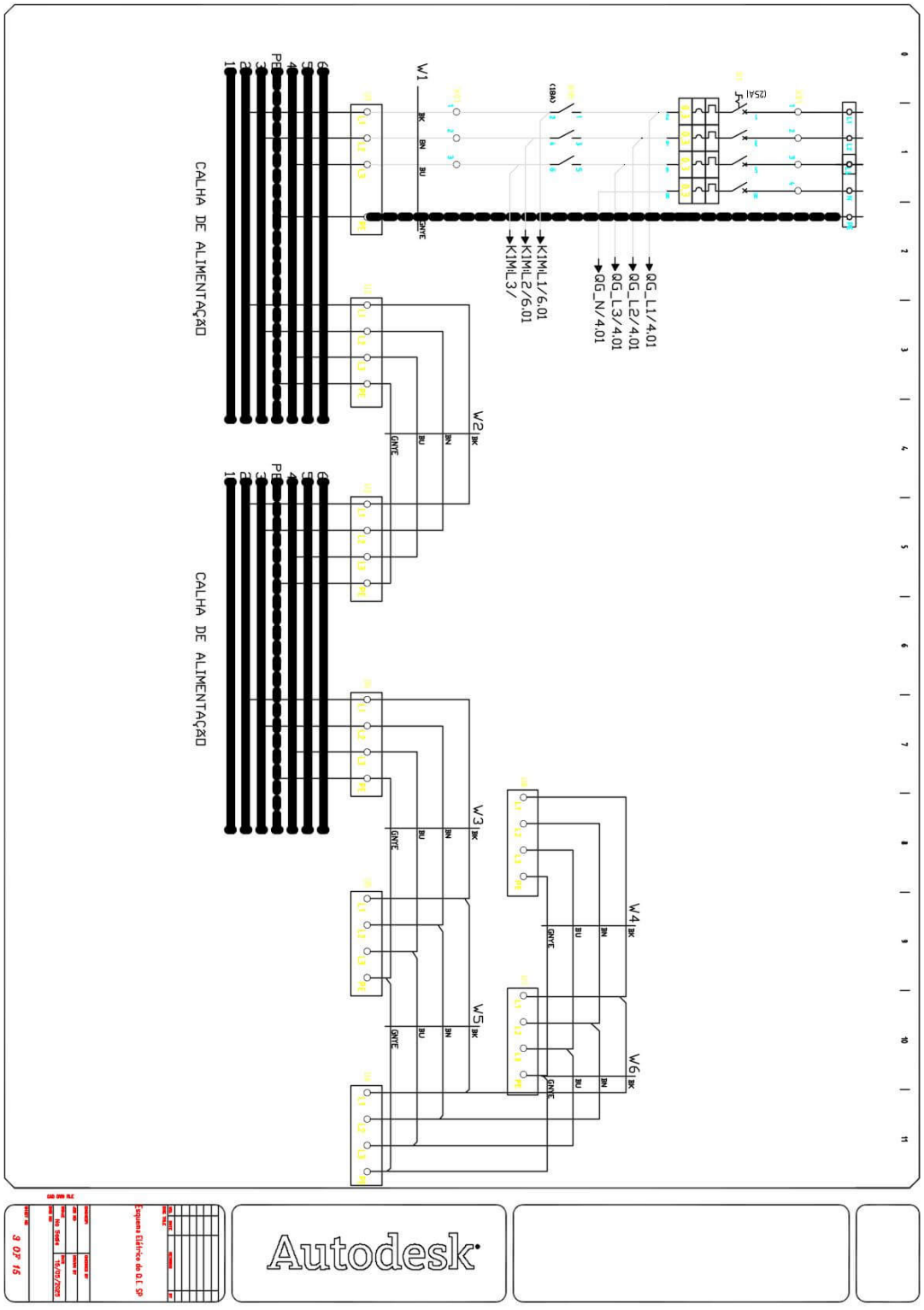


Figura C.1: Esquema Elétrico do sistema de agulhagem (1/13)

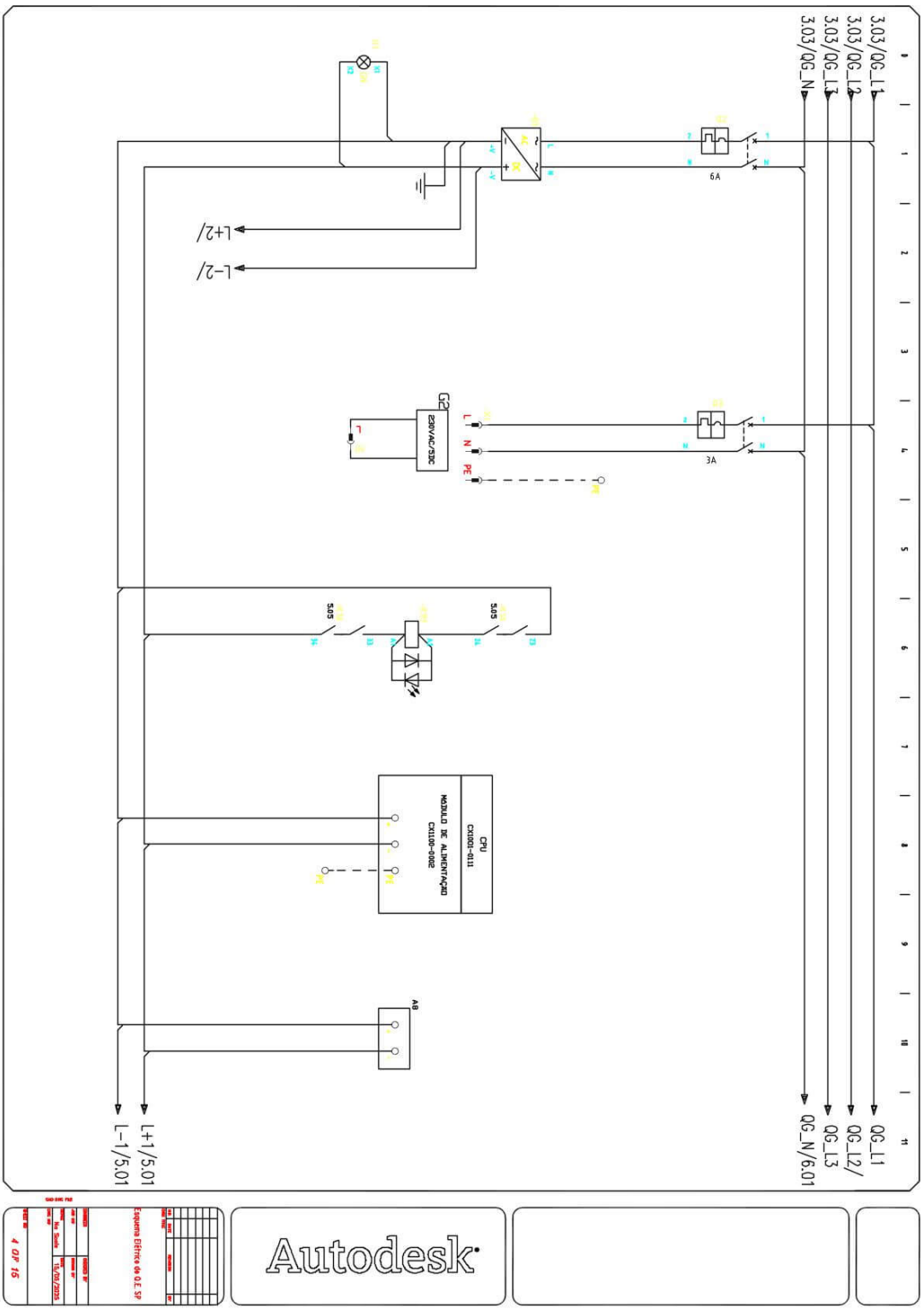


Figura C.2: Esquema Elétrico do sistema de agulhagem (2/13)

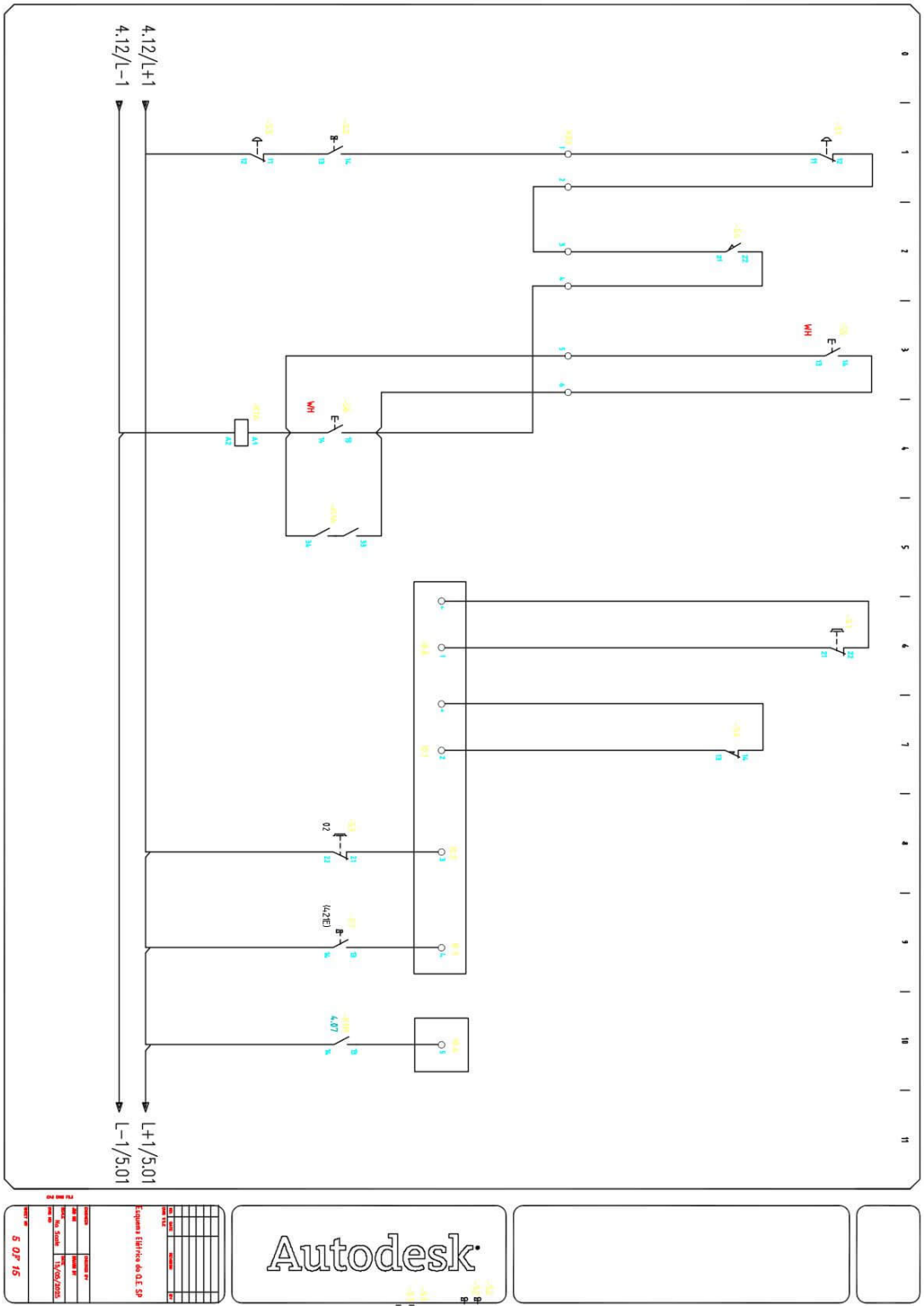


Figura C.3: Esquema Elétrico do sistema de agulhagem (3/13)

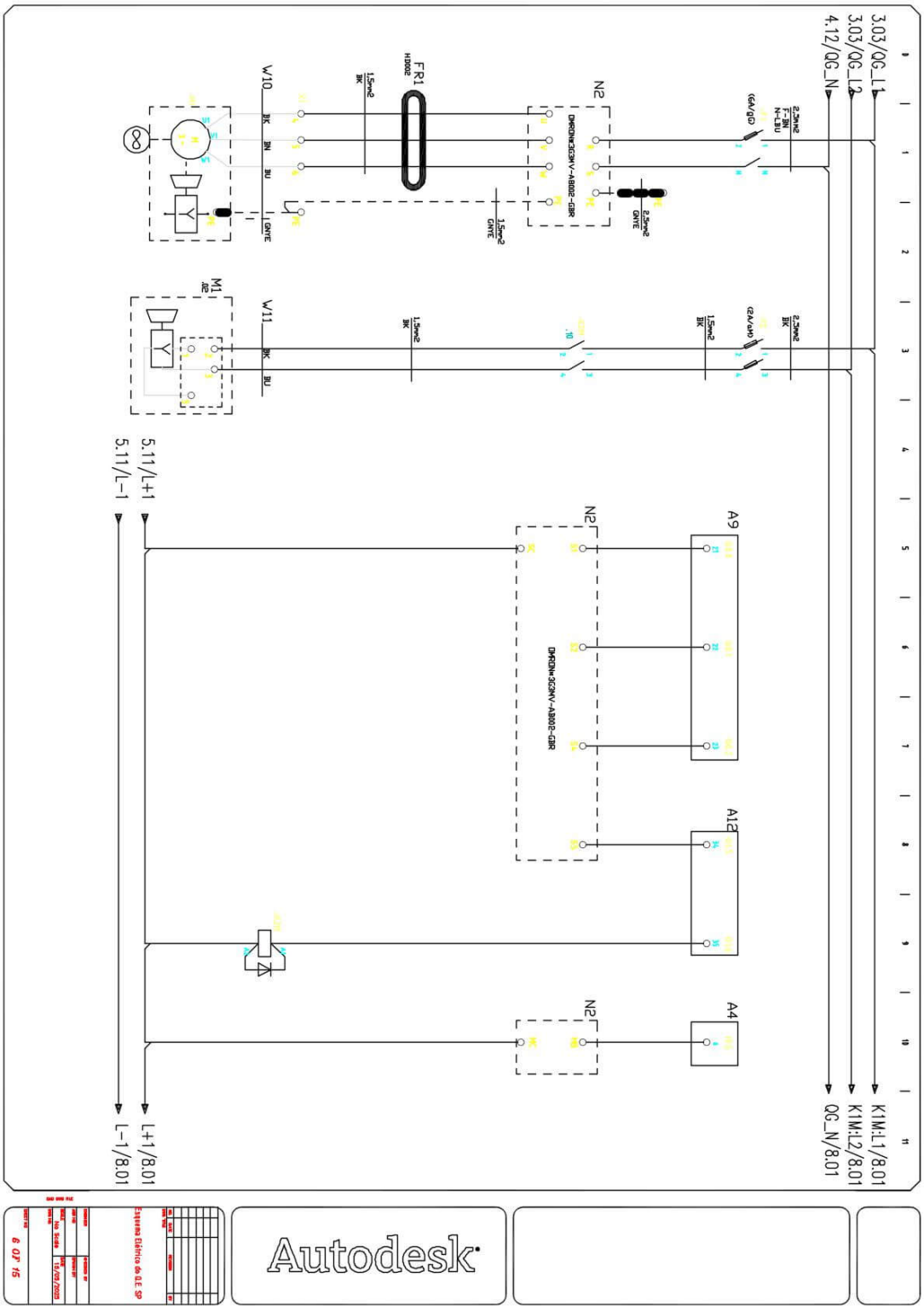


Figura C.4: Esquema Elétrico do sistema de agulhagem (4/13)

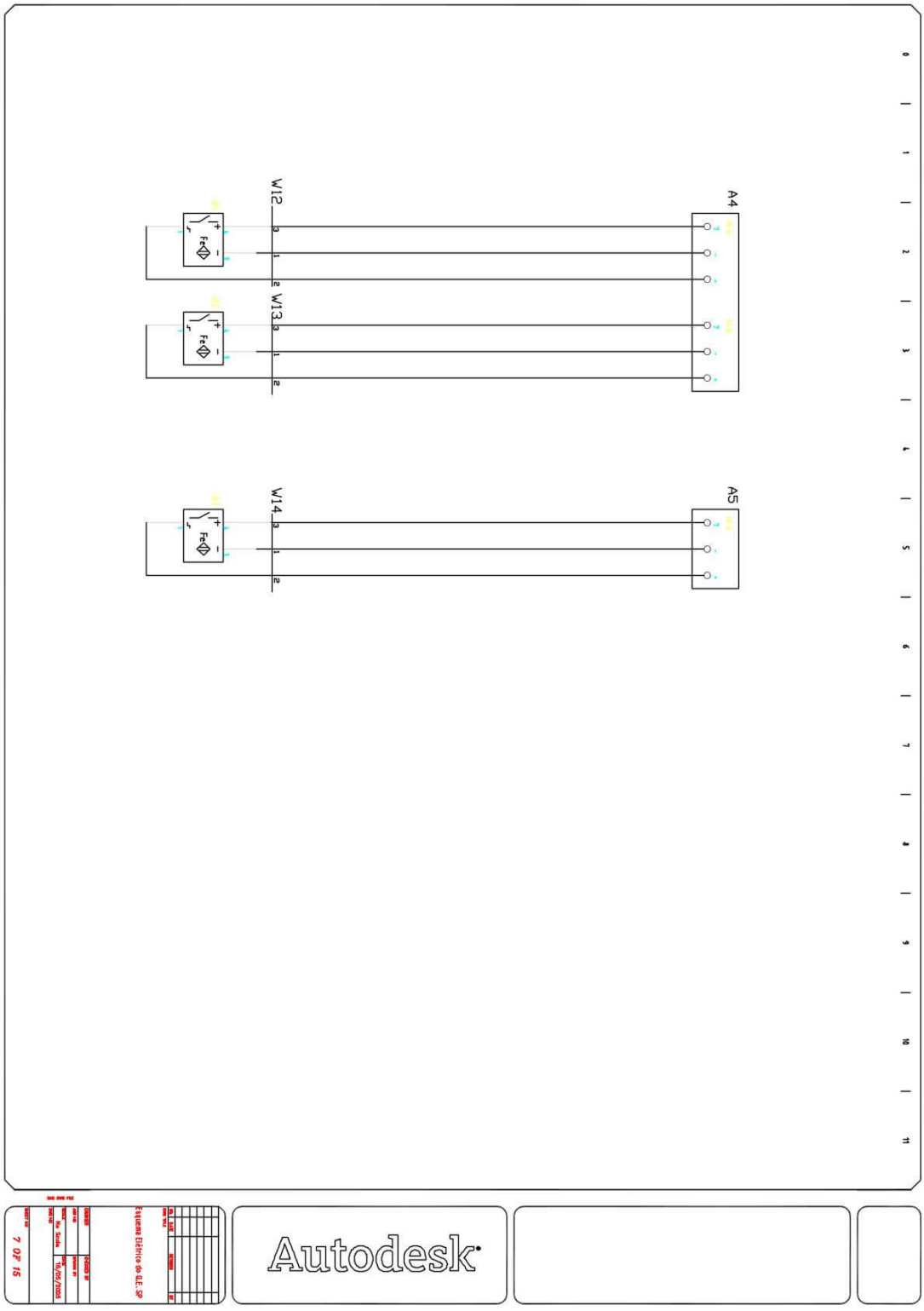


Figura C.5: Esquema Elétrico do sistema de agulhagem (5/13)

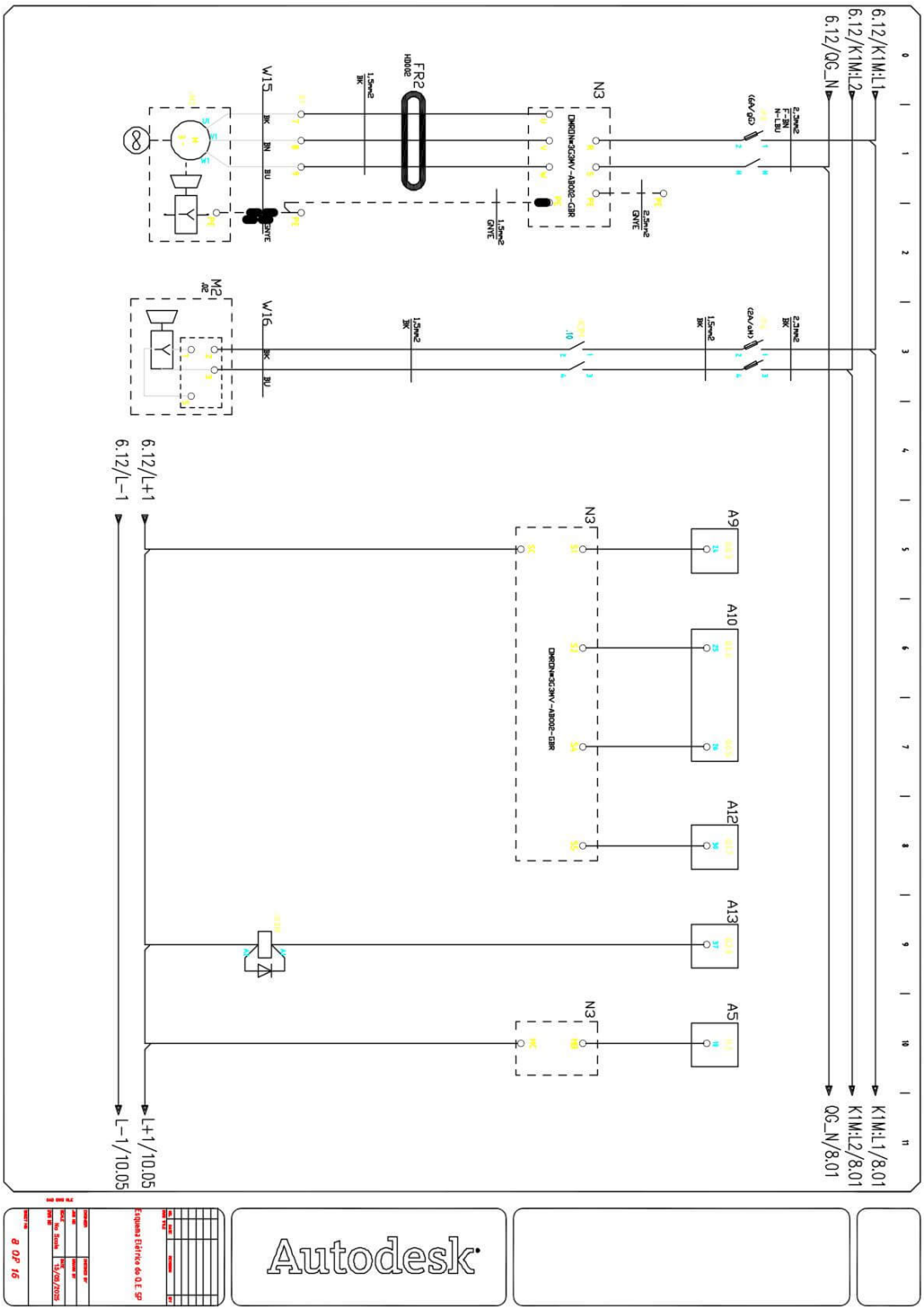


Figura C.6: Esquema Elétrico do sistema de agulhagem (6/13)

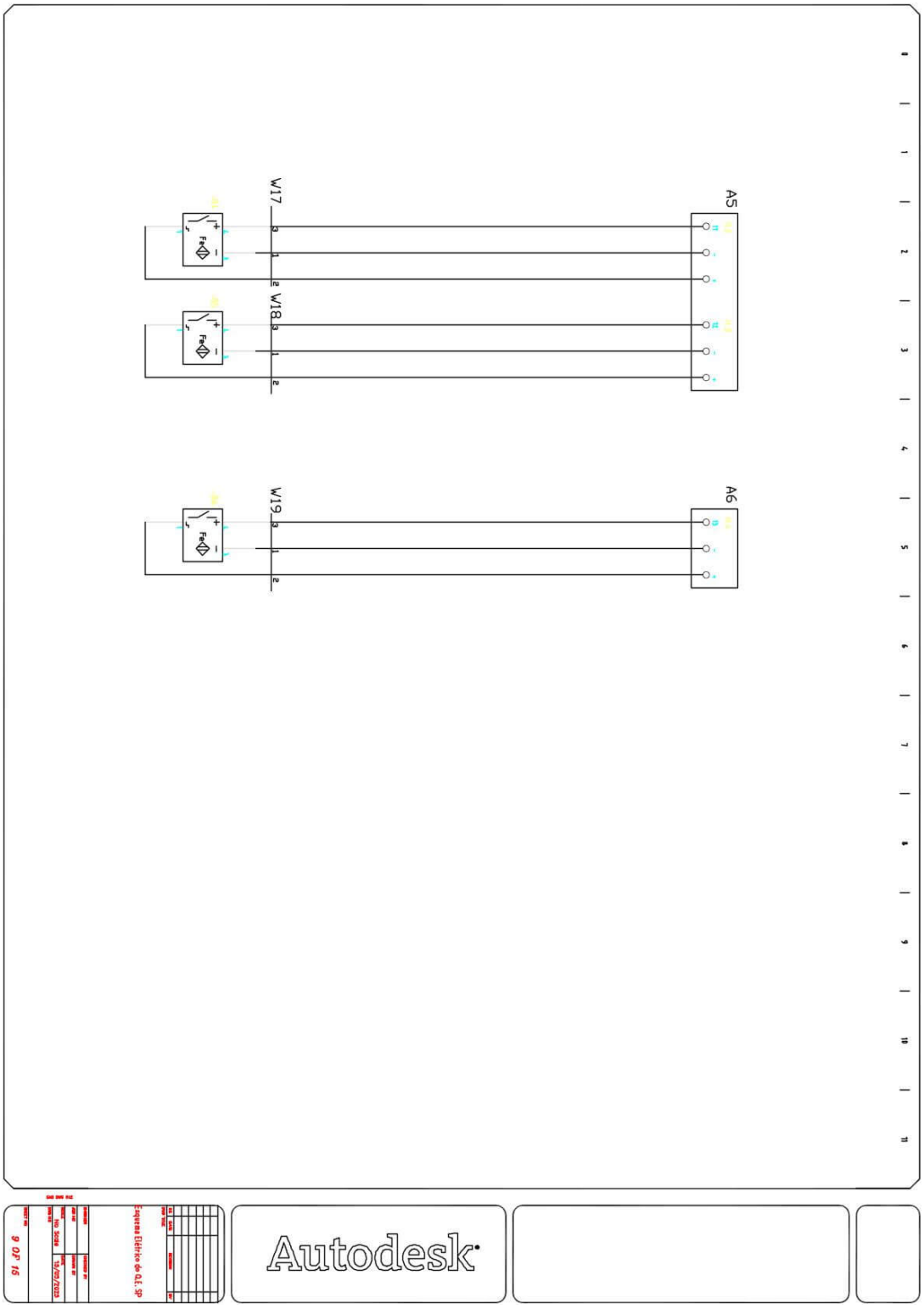


Figura C.7: Esquema Elétrico do sistema de agulhagem (7/13)

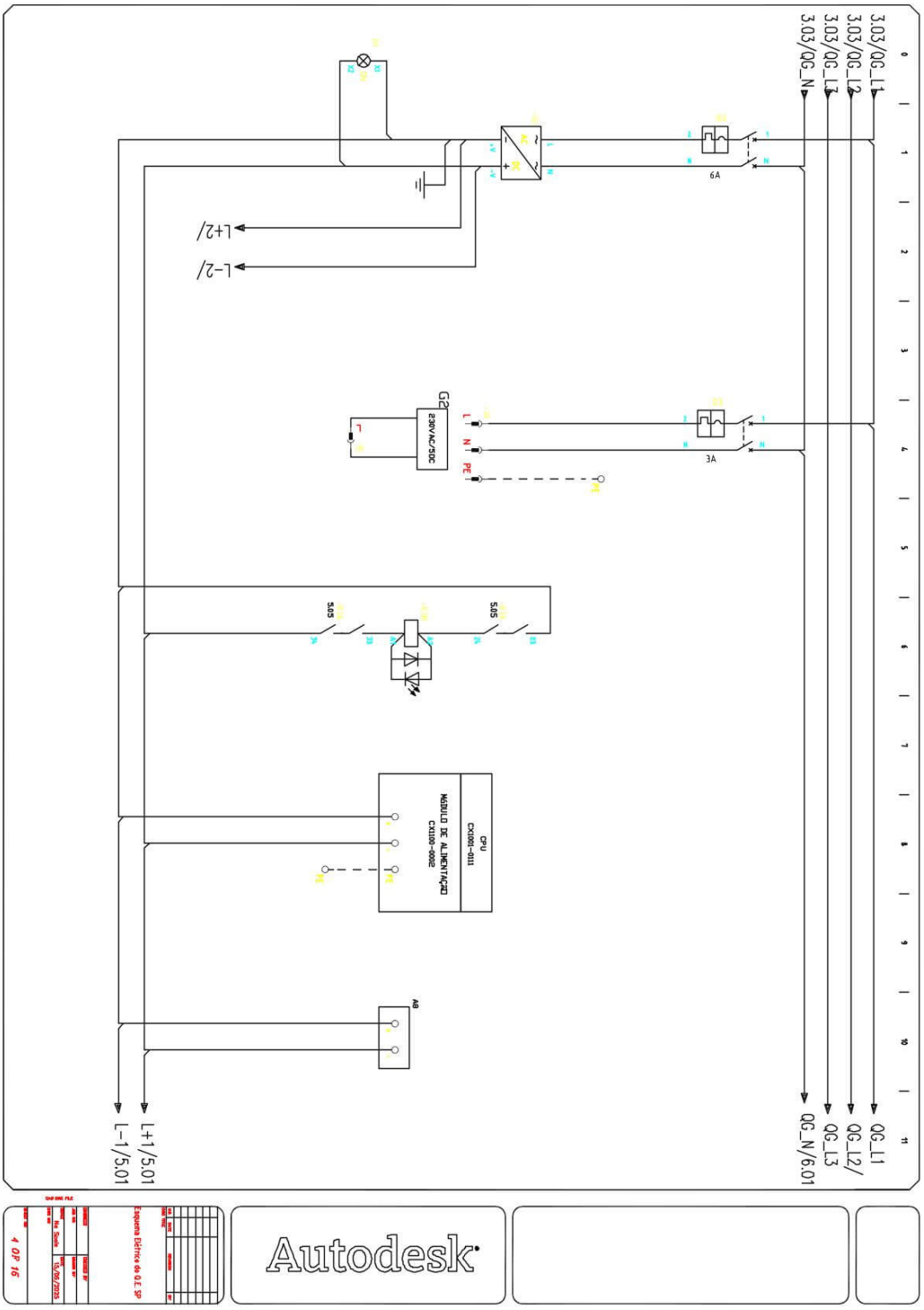


Figura C.8: Esquema Elétrico do sistema de agulhagem (8/13)

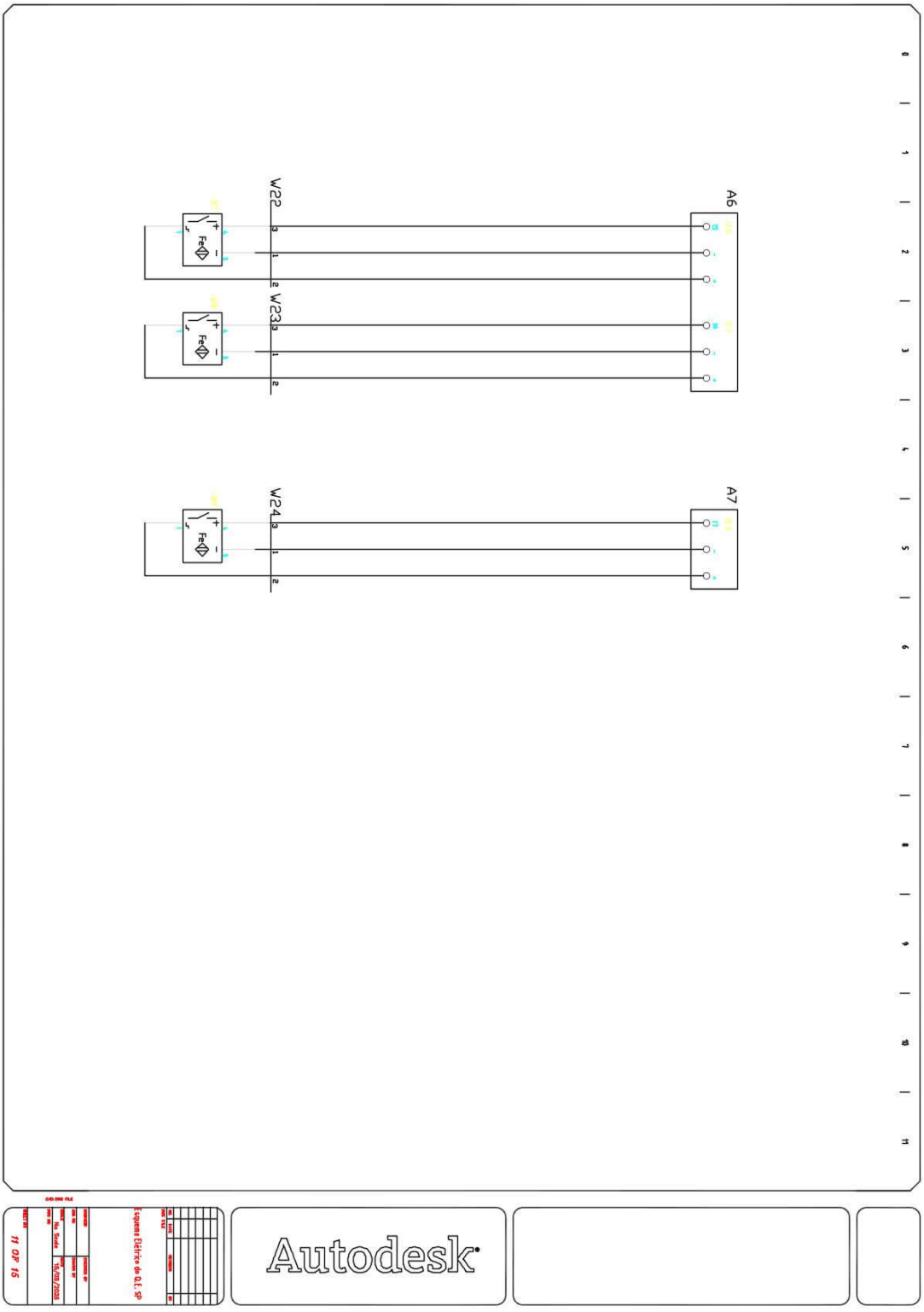


Figura C.9: Esquema Elétrico do sistema de agulhagem (9/13)

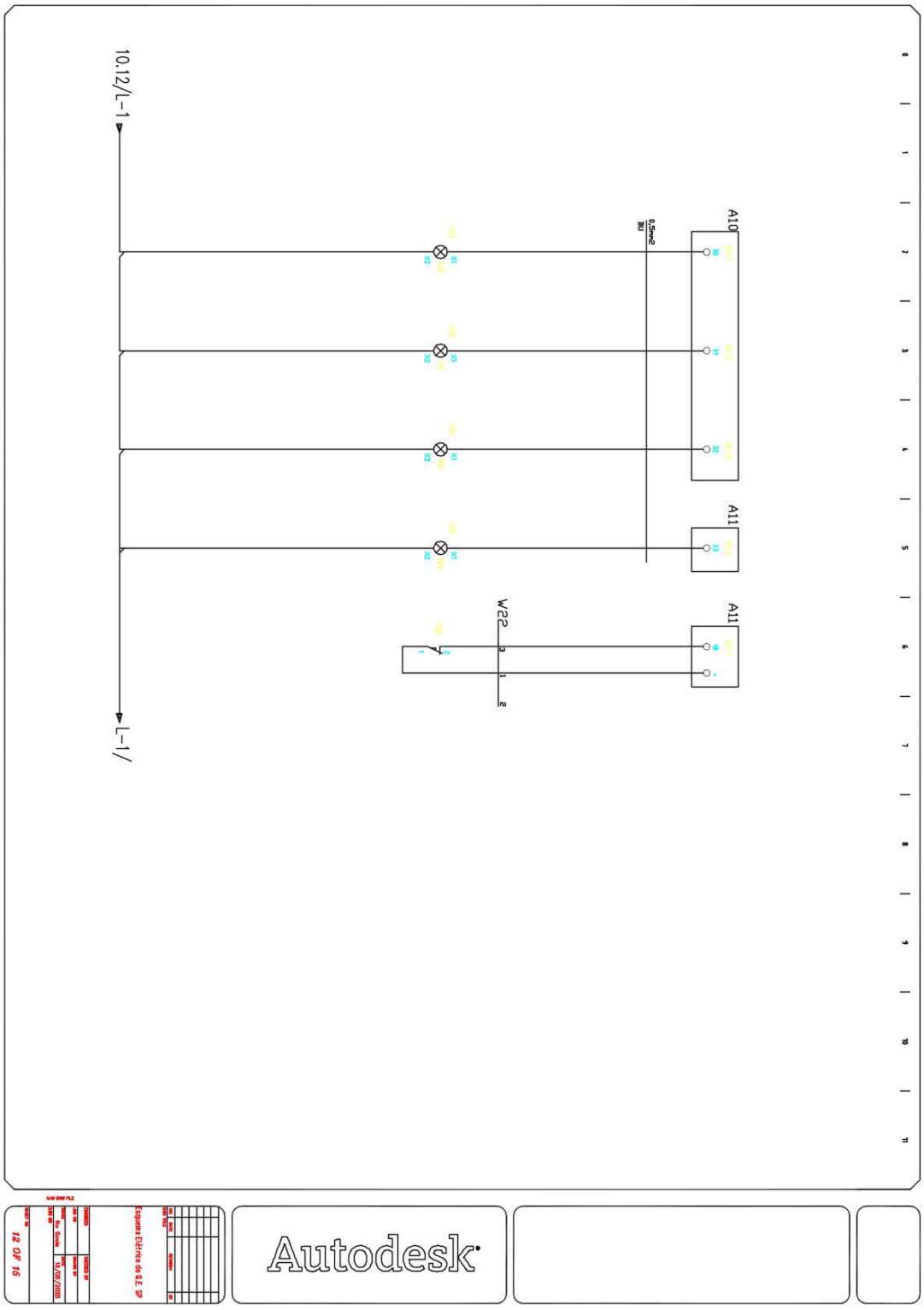


Figura C.10: Esquema Elétrico do sistema de agulhagem (10/13)

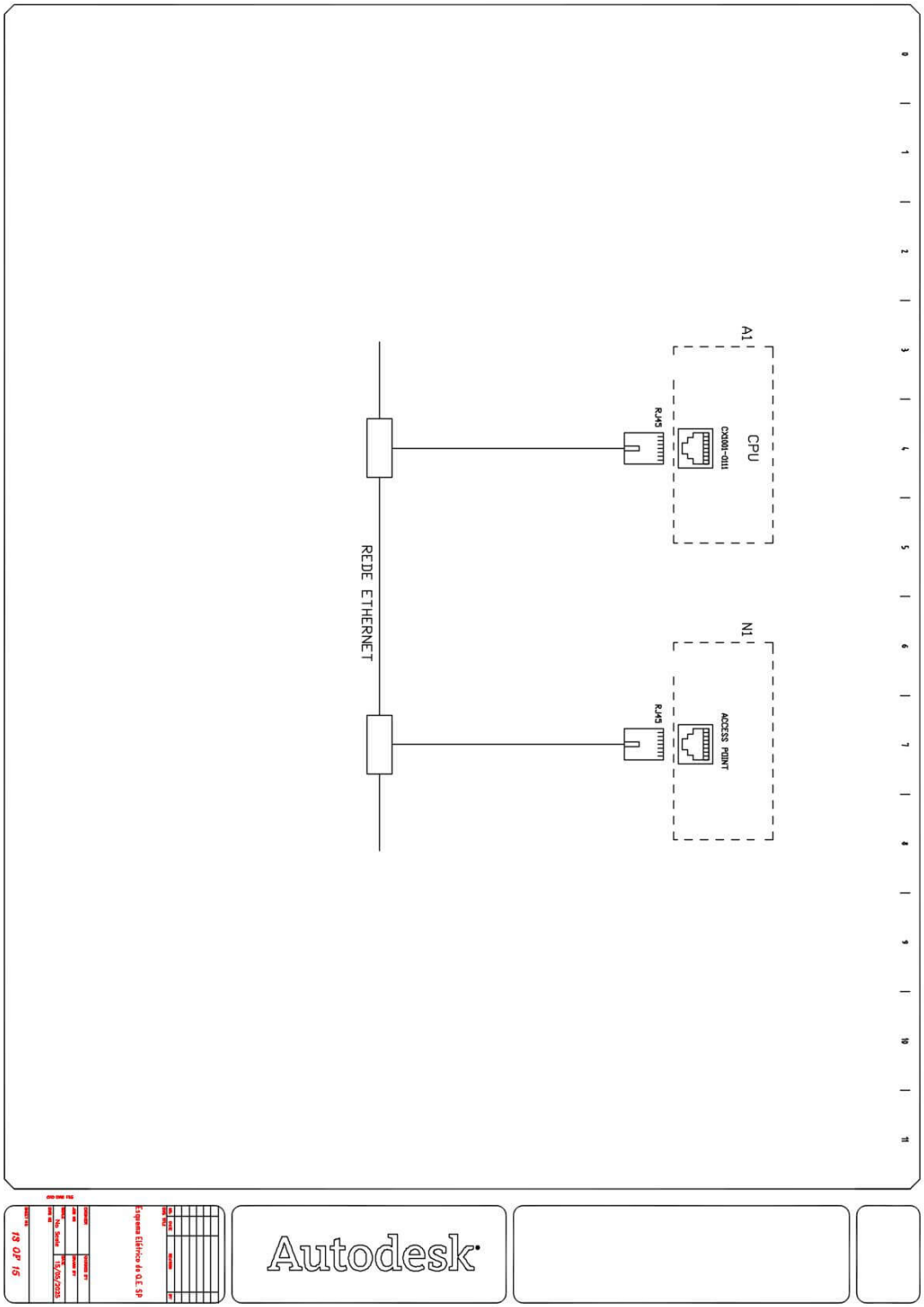


Figura C.11: Esquema Elétrico do sistema de agulhagem (11/13)

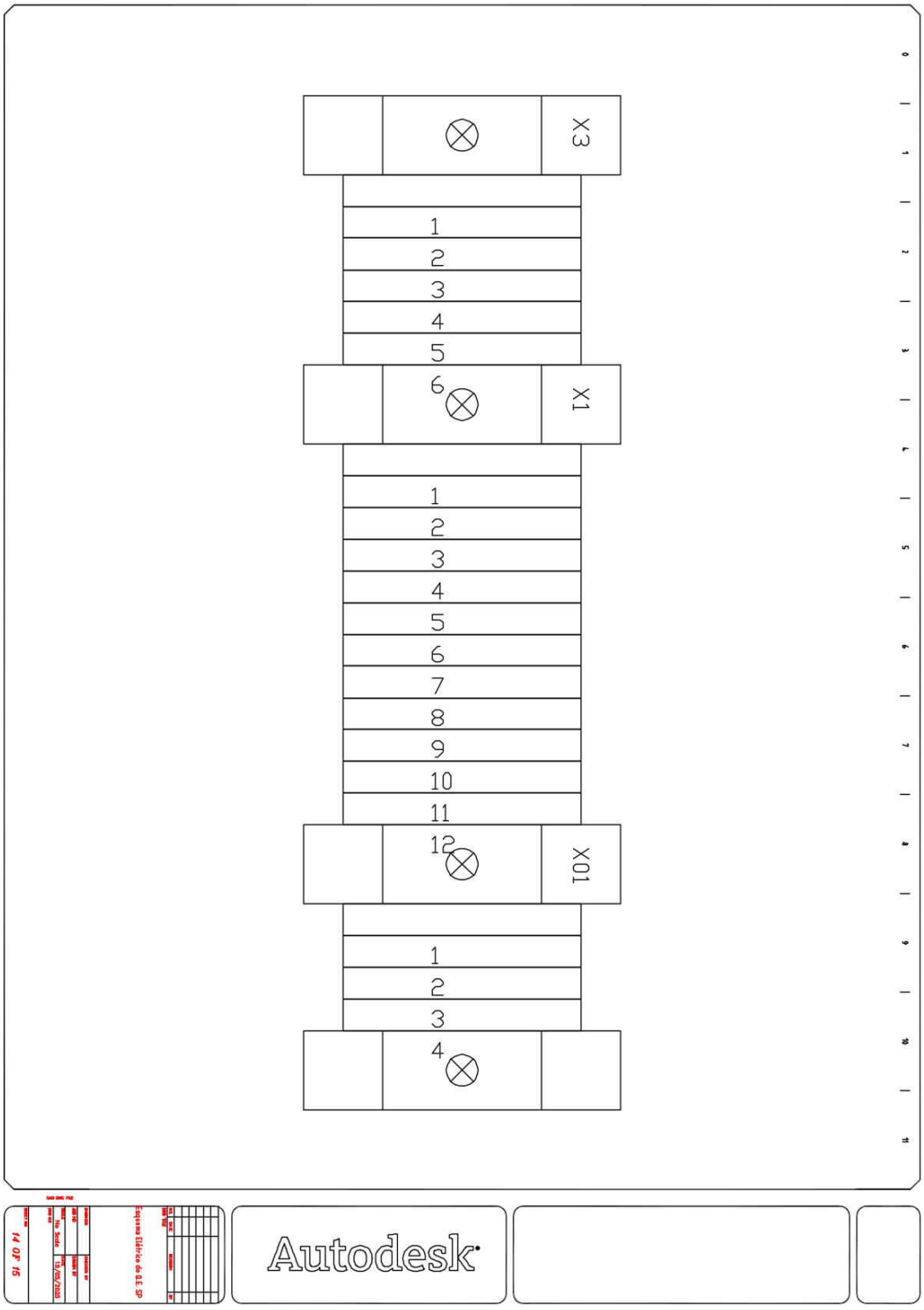


Figura C.12: Esquema Elétrico do sistema de agulhagem (12/13)

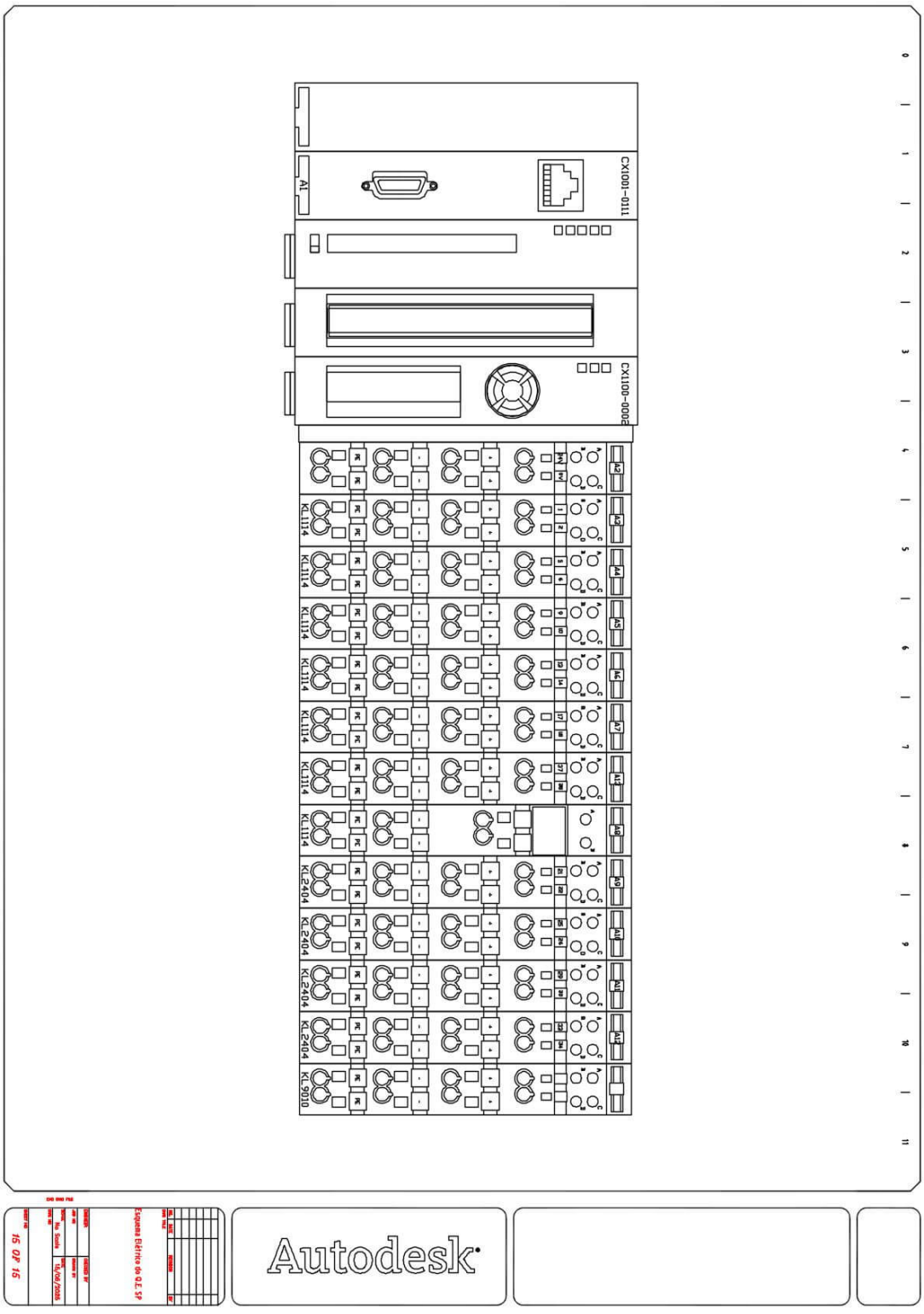


Figura C.13: Esquema Elétrico do sistema de agulhagem (13/13)

Instituto Superior de Engenharia
Politécnico de Coimbra