NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

JOÃO BAPTISTA DE MORAIS ALARCÃO POTIER

BSc in Electrical and Computer Engineering

MODULAR OPEN HARDWARE EDUCATIONAL CONTROLLER

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon

September, 2023

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING

MODULAR OPEN HARDWARE EDUCATIONAL CONTROLLER

JOÃO BAPTISTA DE MORAIS ALARCÃO POTIER

BSc in Electrical and Computer Engineering

Adviser: André Dionísio Bettencourt da Silva Parreira Rocha
Assistant Professor, NOVA School of Science and Technology - NOVA University of Lisbon

Co-adviser: Leandro Henrique Monteiro Filipe
Msc, UNINOVA/CTS

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
September, 2023

Modular open hardware educational controller

Copyright © João Baptista de Morais Alarcão Potier, NOVA School of Science and Technology, NOVA University Lisbon.
The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.10.18) [1].

Para os meus Avós

Acknowledgements

Primeiro, agradecer ao meu orientador, Professor André Rocha, pela oportunidade de desenvolver esta tese, pelo seu apoio e orientação ao longos destes meses de trabalho.

Em segundo lugar, gostava de agradecer ao meu co-orientador, Leandro Filipe, por todo o seu apoio no desenvolvimento da implementação da solução.

I would also like to extend my gratitude to all PhD students who helped me with this thesis: Miguel Arvana, Fábio Oliveira, Nelson Freitas, Duarte Alemão and Francesco Lupi. Thank you all for always keeping a good environment in the lab.

Gostava tambem de agradecer ao Miguel Arvana, Fábio Oliveira, Nelson Freitas, Duarte Alemão e Francesco Lupi, Obrigado por estarem sempre disponiveis para responderem ás minhas duvidas e para me ajudarem no que fosse preciso.

Aos meus amigos da faculdade: Toni, Barrigas, André, Bia, Faria, Ruca, Bohdan, Xico, Pedro e Sardinha. Um grande obrigado por todas as memorias, por todas as noites de trabalho "árduo" e todas as entradas para cartão vermalho realizadas nestes anos.

A todos os meus amigos, GFAPM, Heitor, Pico, Miranda, Ana Rita e Bitinha. Um muito obrigado por estarem sempre lá para mim.

Ao Nuno Setubal, um grande agradecimento por toda a paciência para me aturar durante o ensino secundário.

E por ultimo, e não menos importante, um agradecimento á minha familia, especialmente á minha Mãe, ao meu Pai e á minha Irmã. Obrigado por me apoiarem sempre nas minhas decisões e estarem lá sempre para mim.

”

“The greatest thing about tomorrow is I will be better than I am today”

— Tiger Woods
(Professional Athlete)

Abstract

In the last decade, the 4Th Industrial Revolution has brought new challenges and paradigms to the manufacturing industry, in response to the consumer's requirements demanding numerous different products be highly customizable. New standards and technologies emerge faster than ever, as companies try to gain an edge over their competitors. How can students and universities keep up to date and use the different proprietary technologies?

This work proposes an architecture where a controller is developed, with modularity and flexibility as the core ideas, and that allows for the integration of open-source standards that are similar to the proprietary technologies used in today's factories.

The implementation of this architecture led to the development of a prototype modular industrial controller, that is a quarter of the cost of the new generation of industrial controllers. The developed controller was later tested and validated using some of the educational industrial kits from the UNINOVA/RICS.

Keywords: Cyber-Physical Production Systems, Modularity, Industrial Controller, ESP-32, Internet of Things

Resumo

Na última década, a 4ª Revolução Industrial trouxe novos desafios no setor industrial, baseados na crescente exigência dos consumidores em ter uma oferta de produtos altamente personalizáveis. Novas tecnologias, standards e paradigmas surgem rapidamente, impulsionados pelas empresas que tentam ganhar uma vantagem sobre as suas concorrentes. Como podem então as Universidades e os seus estudantes utilizarem as novas tecnologias desenvolvidas?

Este trabalho propõe uma arquitetura para um controlador industrial, com modularidade e flexibilidade como ideias centrais, e que permite a integração de tecnologias *open-source*, semelhantes às tecnologias proprietárias utilizadas nos sistemas industriais atuais.

A implementação resultou no desenvolvimento de um controlador industrial modular, a uma fração do custo de um controlador industrial de nova geração. O controlador desenvolvido foi posteriormente testado, utilizando alguns dos sistemas industriais educacionais do UNINOVA/RICS.

Palavras-chave: Sistemas de Produção Ciber-Físicos, Modularidade, Controladores Industriais, ESP-32, Internet das Coisas

Contents

List of Figures

List of Tables

Acronyms

I^2C	Inter-Integrated Circuit
BLE	Bluetooth Low Energy
CNC	Computer Numerical Control
CPPS	Cyber-Physical Production Systems
CPS	Cyber-Physical Systems
DAQ	Data Acquisition
FPGA	Field Programmable Gate Array
GPIO	General Purpose Input and Output
I/O	Input/Output
IIoT	Industrial Internet of Things
MASs	Multi-Agent Systems
MIMO	Multi Input Multi Output
MISO	Master-In Slave-Out
MOSI	Master-Out Slave-In
OPC-UA	Open Platform Communications Unified Architecture
PAC	Programmable Automation Controller
PCB	Printed Circuit Board
PLC	Programmable Logic Controller

RAMI4.0 Reference Architecture Model Industrie 4.0

SBC Single Board Computer
SBM Single Board Microprocessor
SCL I^2C Serial Clock
SCLK SPI Serial Clock
SDA Serial Data
SOA Service-Oriented Architecture
SPI Serial Peripheral Interface
SSn Serial Selector

TCPS Transport Cyber Physical Systems
TSN Time Sensitive Networks

WLAN Wireless Local Area Network
WPAN Wireless Personal Area Network

1

Introduction

Since the start of the First Industrial Revolution, there has been a change in the manufacturing paradigm. Before the First Industrial Revolution, we had mainly handcraft production, where specialised artisans would create one-of-a-kind products, according to the customer's request, spending weeks crafting a single product. After that, with the First and Second Industrial Revolutions, in the 18th and 19th centuries, companies developed the capability of producing large quantities of the same product, the so-called mass-production era, breaking complex craft production into more elementary and less time-consuming tasks that anyone would quickly learn, increasing the efficiency of the production process. This led to the manufacturing of products that can't be easily customised and adapted to one needs due to the rigidity of the production line [2], [3].

With the start of the Third Industrial Revolution, production lines suffered significant changes. The presentation of the first Programmable Logic Controller (PLC) in 1969, flexible automation technologies in the 1970s and Computer Numerical Control (CNC) in the 1980s originated a change in the production systems design, where flexibility and reconfigurability were now a critical factor in their development. The same production line, with a few adaptations, could now produce different products based on a generic one, allowing more customised and diverse options for the consumer. This led to a new manufacturing paradigm: Mass customisation [2], [3].

As customer demands become increasingly unique, as everyone wants a product tailored to their needs and tastes, manufacturing companies must once again adapt their production methods. This led to a new manufacturing paradigm: Mass Individualization. This new manufacturing paradigm has the possibility to combine the best parts of the previous paradigms [2]: (1)Individualised products, as in handcraft production. (2)Companies still produce a high volume of manufactured goods at a low cost, as in Mass Production. (3)Adaptability to current

market trends, as in Mass Customization.

To help with this transition, new and more flexible manufacturing systems have to be developed. At the same time, the Fourth Industrial Revolution has started, allowing disruptive technologies to *"optimise the value stream in industrial production"*[4]. As Industry 4.0 promotes modular, flexible, easily integrable and connected systems, there are components in today's industrial manufacturing environment that will need to be adapted to achieve better modularity, flexibility and the level of integration desired.

A clear example of a key industrial component that needs to evolve is the Industrial Controller. Industrial Controllers are still predominantly PLCs, a controller not developed with system interoperability and connectivity as a basis, most of which use proprietary communications. Eventually, industrial controllers will evolve into new kinds of controllers that assure an easier integration and more adequate features for the new manufacturing systems requirements [5].

1.1 Questions and Hypothesis

As companies develop the new generation of Industrial Controllers, they will still implement their proprietary technologies to sell the customers all the necessary hardware, software and maintenance services. This will create a problem, as universities and students cannot work with every single brand-new proprietary technology. This creates the question:

- How can universities and students work with new technologies that are being developed for the new generation of controllers, without having to buy every commercial controller available in the market?

To fill this gap, this thesis proposes the development of a low-cost controller, with modularity and open-source technologies/standards as its basis, to allow universities, students and others the opportunity to use similar technologies to those being developed for the new generation of Industrial Controllers.

1.2 Thesis Outline

The thesis is divided into the following chapters: Chapter 2 presents the main theoretical aspects used in developing the work. Chapter 3 presents the architecture of the system. Chapter 4 presents the implementation of all the components

of the system. Chapter 5 presents the tests and results obtained. Chapter 6 shows the final conclusions of the thesis and presents possible improvements and future work.

2

State Of The Art

The first step in the development of the controller is to look into the current research trends, to establish the theoretical basis and fundamental concepts for the controller. Firstly, the Fourth Industrial Revolution and its impact are presented. Secondly, it presents the concept of Cyber-Physical Systems (CPS), outlining its importance in this new era of industrialisation. Thirdly, the Reference Architectures that were developed to facilitate the integration of these new concepts are outlined. Fourthly, the different types of controllers that are used in today's industrial factories, and also new types of controllers are reviewed. Finally, some of the most common industrial protocols are also presented.

2.1 Industry 4.0

The Fourth Industrial Revolution, or Industry 4.0, was first presented by the German government at the Hanover Fair, a significant platform to present industry development, in 2011. The document set the directives and the main areas of research that the German government would finance to boost the development of the German manufacturing industry until 2020. This program was developed by the Industrie 4.0 Working Group, composed of members representing the prominent companies of the German manufacturing ecosystem and from the German research groups [3], [6].

This initiative's main objective is to create "a new level of socio-technical interaction between all the actors and resources involved in manufacturing."[6], where new manufacturing systems and paradigms are developed with adaptability, interoperability, and modularity as core features, shifting towards decentralized production systems, highly customizable products and resource optimization [7]. Fig 2.1 is an example of the overall objective of this initiative.

According to [8], the implementation of Industry 4.0, from the Manufacturing

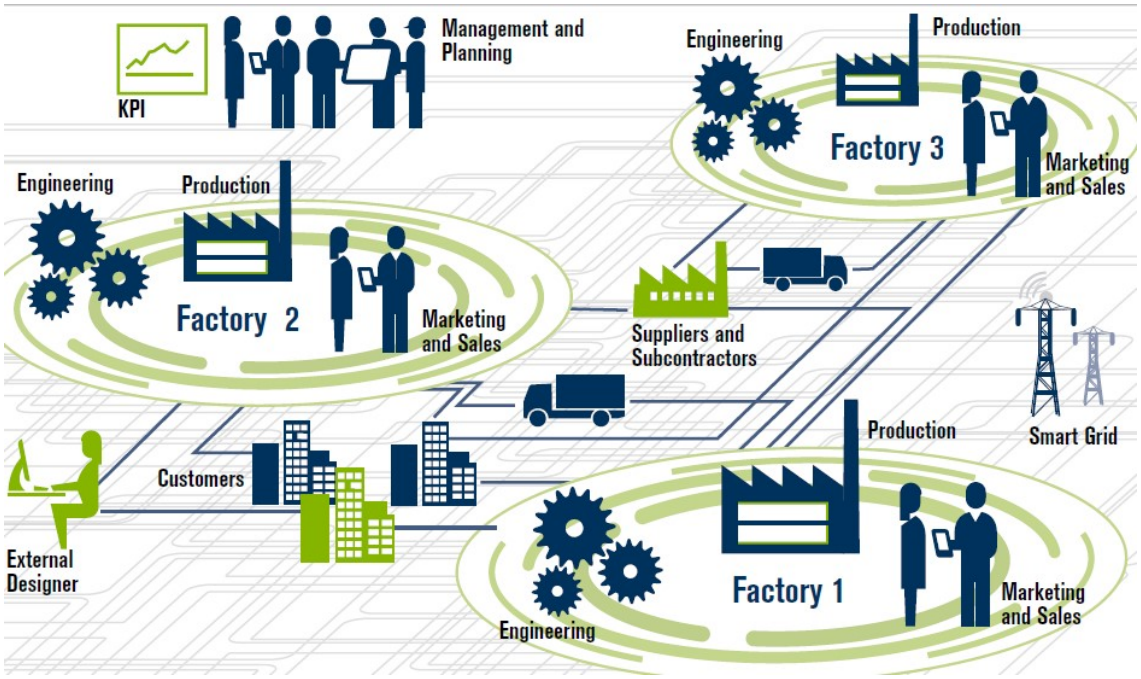

Figure 2.1: Decentralized Production Network. [6]

systems perspective is divided into three critical areas of development: Vertical Integration, Horizontal Integration and Acceleration of Manufacturing. As the manufacturing sector faces the challenge of responding to the growth in society's demand for sustainability and social responsibility, these dimensions propose a new approach to the design and development of new manufacturing systems.

Vertical Integration refers to the integration of all processes in a company, by using smart manufacturing systems and adaptable manufacturing paradigms as alternatives to traditional fixed production methods [9]. Horizontal Integration refers to the establishment of a networking environment by the multiple parties that are present in a product life-cycle, in pursuance of better collaboration between all actors in the product's value chain and better response to current market trends [9]. Acceleration of Manufacturing regards the optimisation of manufacturing systems through the integration of new technologies to make processes faster and more flexible, involving collaboration between new players representing these technologies and the traditional players in the manufacturing environment[9].

Besides these technological developments, the Boston Consulting Group identifies other 8 key technologies as the power behind the industry 4.0 development [10]:

- Autonomous Robots - New robots are developed to perform critical operations autonomously, with efficiency. Flexibility is also important, as

machine-machine and human-machine collaboration in the same spaces is essential in Industry 4.0.

- Big Data and Analytics - Utilizing large sets of data to identify patterns and predict possible outcomes will have a significant impact on decision-making for adjustments in the manufacturing ecosystem in real-time.
- Additive Manufacturing - Employ new methods of production, like 3-D printing, to test prototypes and produce highly customized goods.
- Cloud Computing - Using cloud technologies to save and share data between different parts of the value chain of a product, also allowing for data analytics to help decision-making based on information received from all stages of the value chain of a product.
- Cyber-Security - As the other technologies produce and request a bigger amount of data, and the number of machines that are interconnected are in record numbers, keeping the data and machines secure is a critical aspect of Industry 4.0.
- Industrial Internet of Things (IIoT) - Using embedded computing for all agents of the manufacturing systems, from the product to sensors and machines, will allow for a more complex network, allowing the decentralization of basic decision-making and connection of all elements of the manufacturing environment;
- Augmented Reality - Augmented Reality can be defined as the presentation of digital information in the real world, enhancing the user's perception of the physical environment through a smart devices. This new technology can have a great impact on the training of new employees, or showing instructions for resolving complex procedures as they are being approached.
- Simulation - Simulations are already used both in product development and in production line development. The next step is using real-time data to create a real-time model of the machine and perform calculations to optimize the functioning of the production line.

To conclude, Industry 4.0 represents a significant paradigm shift in which all 9 pillars converge to redefine industrial operations, with the aim of creating an interconnected industrial world.

2.2 Cyber-Physical Systems

The CPS concept was first mentioned in a workshop promoted by the National Science Foundation in 2006. According to [11], *"CPS are integrations of computation with physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa."*.

Cyber-Physical systems do not emerge as independent systems but are instead created through the integration of existing infrastructures with embedded information technology, utilizing the internet, mobile communication services, and cloud solutions. This enables one of the essential characteristics of CPS: its ability to provide data and services in real time[12].

As it is still maturing as a new concept, Cyber-Physical Systems still have some challenges in their development. CPSs are vulnerable to cyber attacks, which can seriously affect the physical systems they control. Ensuring the security and safety of CPS requires addressing cyber and physical security threats [13]. Each cyber-physical system can be composed of different technologies and use diverse data types or components from different manufacturers, which can cause constraints. Therefore, assuring interoperability between the various systems is necessary to communicate and work together effectively [14].

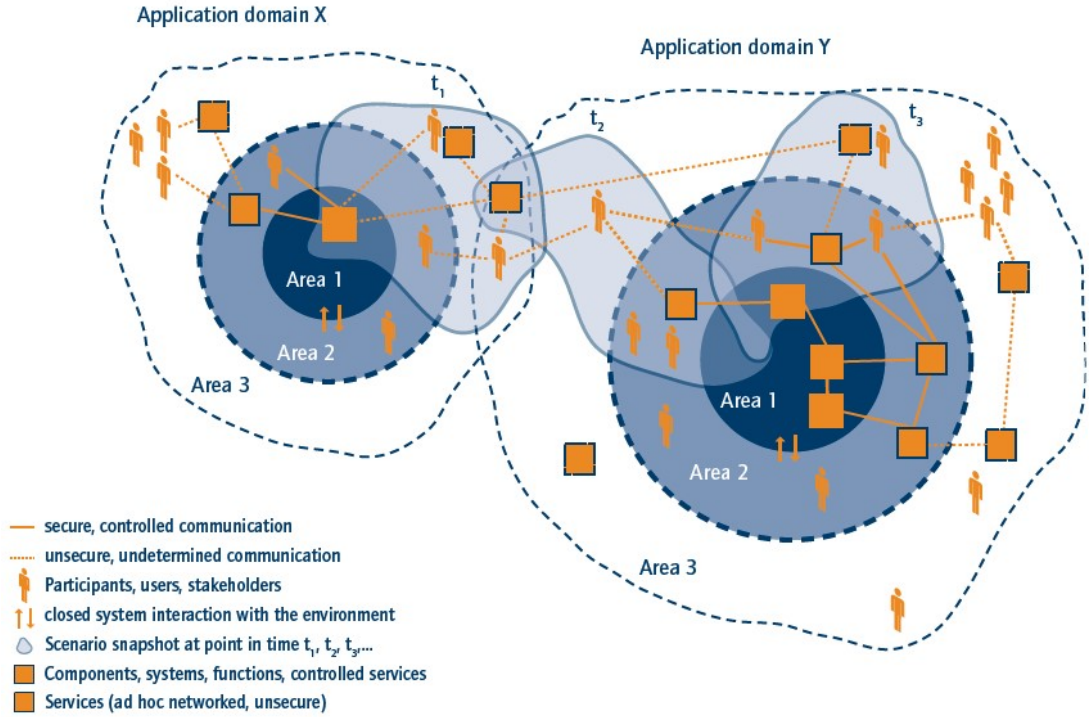

Figure 2.2: Schematic illustration of the cross-domain integration of cyber-physical systems[15]

In the context of Industrie 4.0, CPS are responsible for managing not only machines and manufacturing processes but also customer interactions, service providers, and product inventories while ensuring seamless autonomous execution across all areas [16]. The resulting systems are particularly evident in their complexity and performance when two or more domains are connected, such as when CPS from various application fields are networked together, as shown in Fig 2.2.

Cyber-Physical Systems can be applied to different areas:

- Energy Management: In recent years, the power grid has shifted to a new paradigm, the Smart Grid. The Smart Grid pretends to use some of the Industrie 4.0 main technological advances to optimize energy production and consumption and therefore reduce the usage of fossil fuels. Also providing information to the consumer in real-time about their consumption habits, through the use of Information Technologies, and developing a self-aware network that identifies possible anomalies and resolves them [17].

- Healthcare: Healthcare 4.0 is the new paradigm in Health concerning the application of disruptive technologies to shift from traditional hospital-based care to a more virtual, distributed care [18]. The application of Cyber-Physical Systems in Healthcare has already seen significant developments. In [19], a comprehensive review of these applications is presented, from medical diagnosis to patient data management/security and resource management.

- Transportation: Transport Cyber Physical Systems (TCPS) is the designation for the new systems developments in the transport area. Most often, this term is associated with the development of smart cities, as the establishment of these networks, with real-time monitoring of vehicles circulating in a city, can improve safety and efficiency, optimization of traffic flow and collection and analysis of data for future improvements in the transport infrastructures [20]. The same concepts can be applied to other forms of transportation, like ship and air navigation [21].

2.2.1 Cyber-Physical Production Systems

Cyber-Physical Production Systems (CPPS) concept emerges from the use of Cyber-Physical Systems in the Industry domain. CPPS development is influenced by the advances in Computer Science, Information and Communication Technologies, and manufacturing science and technology. CPPS incorporates autonomous

and cooperative components and sub-systems that are linked between various production levels, from processes to machines and production to logistics networks [22].

With the interaction between multiple networks and different parts, some constraints may appear, leading to a disruption in the processes CPPS are controlling. This means that there are some design concerns that must be addressed in the development of a CPPS. According to [23], a cyber-physical production system must be able to withstand external influences. It must be capable of self-regulation and self-recovery, returning to its normal state after a disruptive event. In order to minimize downtime during disruptions, the system must have a short response time and the ability to quickly implement suitable responses, using redundancy to help handle those disruptions. Every component in the system must possess a component data model containing information about its manufacturing and assembly operations, and have the ability to exchange information with other components and autonomously make decisions.

Despite being a new concept, and still the focus of theoretical development, some implementations and case studies are already being discussed:

- In [24], a CPPS Framework for real-time assessment of the environmental impact of a 3D-printed product is proposed. The CPPS is divided into 3 components: the Cyber world, the Physical world and the Data Acquisition (DAQ) system, the latter one the interface between both worlds. Data(Material and Power consumption) from the impression of the item is recorded and stored, and its environmental impact is calculated and presented as soon as the work is finished.

- In [25], a milling machine is adapted and integrated into a cyber-physical production system. In this case, the differentiation between the Cyber world and the Physical world is done through the types of signals that are in each(Digital Signals in the Cyber World, and Analog Signals in the Physical World). The DAQ system is used as a conversion interface between both worlds. In this case, a machine learning algorithm was also used to help optimize decision-making on adjusting the milling process.

Another example of innovation through the implementation of CPPS is the decentralization of manufacturing. In traditional manufacturing systems, all production decision-making is performed in one single central system, being this central system the only connection between all the other systems that compose

the manufacturing process. This leads to a very rigid system, that cannot be easily adapted to the new manufacturing trends of the 21st century [26].

With the utilization of CPPS, the different components of the manufacturing process are divided into smaller autonomous elements, that can communicate directly to each other. This leads to an optimization in resource allocation, as the manufacturing becomes highly adaptable to disruptions, and machines that are not being used can be allocated to other manufacturing processes [8], [26].

One implementation of the decentralization of manufacturing processes is through the use of Multi-Agent Systems (MASs). A MASs has in an agent its most basic unit, with each agent responsible for performing a specific task. Agents only know about their state and the necessary information to perform their tasks. As systems require different resources, agents can either be added or removed according to the necessities [27].

In the manufacturing control context, agents can be used on the factory floor, as the time constraint is very high, with decisions needing to be taken in fractions of seconds:

- In [28], a framework is presented to implement a scalable, flexible and plug-in data analysis and real-time supervision system. The CPPS in this framework serves as the binding element, collecting data from different elements, pre-processing collected data and generating output data by applying rule-based logic on the shop floor's processed data. In this framework, the implemented MAS is used to abstract the different elements of the shop floor and to realize the pre-processing of the collected data. The abstraction allows for a modular system, that can have elements be added or removed without compromising its correct functioning.

- In [29], an Agent-based CPPS is developed to directly control a production line. Four different types of agents interact between them, managing the processes control at the shop-floor level, to deliver a finished product, according to the available resources, that can be introduced or removed according to the needs, creating a highly reconfigurable system for different types of products on the same production line.

- In [30] is proposed an architecture for a distributed data analysis CPPS for an automotive factory, using a cloud manufacturing paradigm and an agent-based system. The goal was to give some degree of autonomy to the different parts of the system, allowing them to request/subscribe data from other subsystems, and after analysing the data, perform some control functions.

The architecture presents three types of agent modules, that represent the different data analysis capabilities, according to the layer that they are set on. Edge Agent Modules perform simple data processing for a low-time response to events that may occur. This module is mainly implemented on the edge layer, connected to machines and production line elements. Cloud Agent Modules have the capability to process larger sets of data, focusing on event prediction and production optimisation. This module is mainly implemented on the fog and cloud layers, as it needs more processing capabilities and a wider view of the system.

All modules can request information from other modules, either on a vertical level (cloud modules request information to edge modules, or vice-versa) or on a horizontal level (between modules on the same level). Data should be available in a publish/subscribe mode, for frequent information flow, or in a request/response mode, for more specific data. This promotes collaboration throughout the entire production line.

2.3 Reference Architectures for Manufacturing System

Reference Architecture is a predefined structure for a system that provides a common approach for its design and building. In the context of Manufacturing Systems, a Reference Architecture can serve as a blueprint for designing and implementing a manufacturing system that is efficient, flexible, and modular. It defines the components, relationships, and standards that are necessary for building a manufacturing system that meets the specific requirements of the industry.

2.3.1 RAMI 4.0

Reference Architecture Model Industrie 4.0 (RAMI4.0), was presented by Platform Industrie 4.0 in 2015. It is composed of three different axes, shown in figure 2.3:

1. Hierarchy Levels
2. Life Cycle&Value Stream
3. Layers

These represent the different technologies and areas that Industrie 4.0 covers, breaking down *"complex interrelations (...) into smaller and simpler clusters"*, and helping with the migration of enterprises to Industrie 4.0 [31].

The Hierarchy Levels axis is divided into 7 levels, according to the functional characteristics of the piece, based on the IEC 62264 and IEC 61512 standards. However, these standards only cover 4 out of the 7 levels (Control Device to Enterprise), with the other three levels(Product, Field Device and Connected World) being an addition to have a proper representation of the Industrie 4.0 goal, with smart products and smart sensors being represented on the two lower levels, and the connected world level representing the interconnection between multiple enterprises [32], [33].

The Life Cycle&Value Stream axis, developed from the IEC 62890 standard, is used to define the different parts of the products and the machine's life cycle. It is split into two concepts, the type, and the instance. Type refers to the developmental stage of a product, composed of the concept of the product, research, prototyping, and tests and validation. The transition to an instance is done when the product/machine goes to the production stage, where there is a physical and

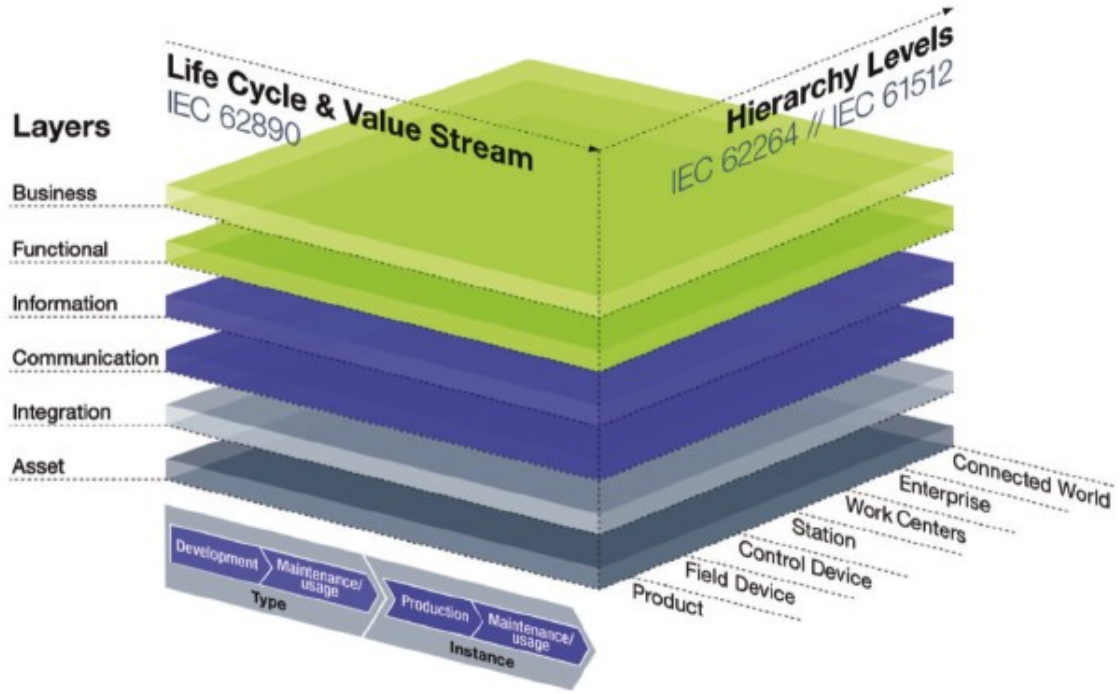

Figure 2.3: RAMI 4.0 [32]

functional product to be delivered to the customer. Both type and instance are divided into two sub-levels, development and maintenance/usage for the type, and production and maintenance/usage for the instance.[33], [34]

The third axis is composed of 6 layers, in which every layer describes a different perspective of technology [32],[33], [34]:

- The lower layer is the asset layer, where all physical reality is represented;
- Above, is the integration layer, responsible for the connection of the members of the asset layer to the virtual world, containing the computerized information about said members;
- In the third layer is the communication layer, responsible for setting all the communication protocols and formats, enabling a standard for communication between all components of a network;
- In the fourth layer is the information layer. This layer deals with all data received and events occurrence, and based on a set of rules, makes relevant information available to the upper layers;
- The functional layer is where all available services and functions are executed. This layer facilitates the horizontal integration of various functions and

creates rules and application logic. It is the only place where remote access and integration of applications occur.

- The upper layer is the business layer, in which all business-related processes and models are associated, also dealing with regulatory frameworks to ensure integrity through the value chain.

2.3.2 5C

As a new concept, CPPS require the definition of guidelines and standards for their development and implementation. So, in [35] is proposed a new architecture with guidelines for the implementation of CPS in manufacturing environments. As represented in figure 2.4, this new architecture is composed by 5 different levels:

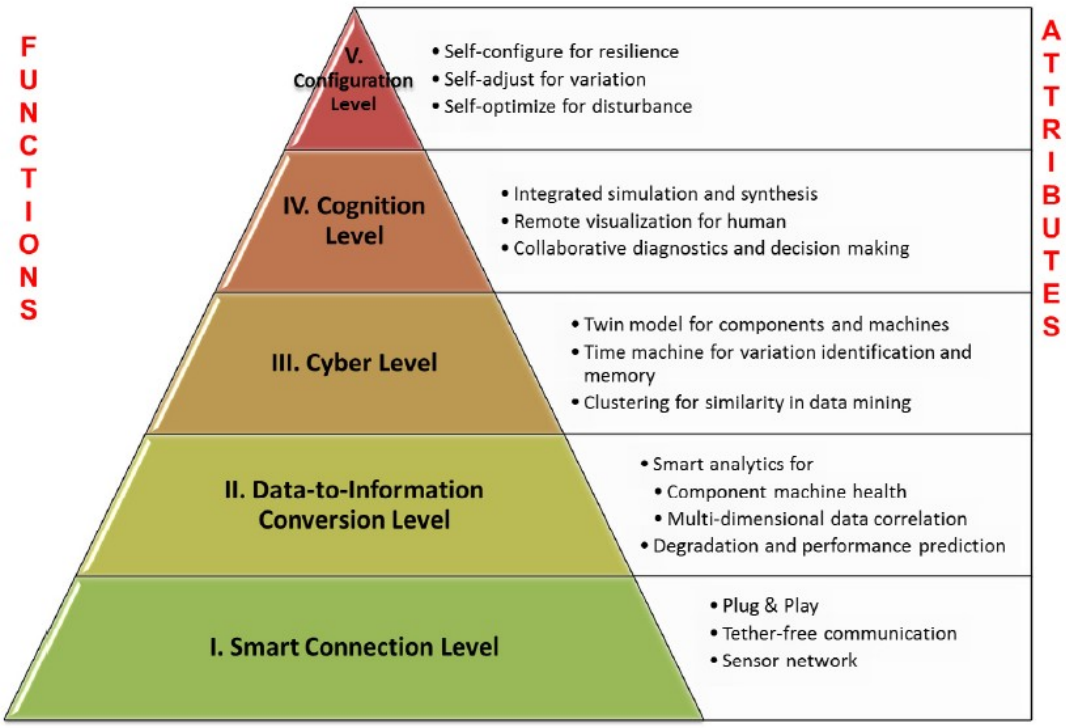

Figure 2.4: 5C architecture for implementation of Cyber-Physical System [35]

- Smart Connection Level, where all data acquisition is dealt with, either by sensors, other controllers or from the higher levels of the manufacturing hierarchy. Two main elements must be defined at this level. The first one is the definition of a reliable method for handling data acquisition and

transmission, due to the different data types that the system can interact with, and the second one is the proper selection of sensors.

- Data to Information Conversion Level, where algorithms and other tools are applied to transform data recorded by the Smart Connection Level, into relevant information. Some algorithms have been developed with the goal of giving the machine self-awareness.
- Cyber Level receives the relevant information from every machine connected to its network, and processes all information once again, in order to understand each machine's status. By making this information available, it gives the different machines the ability to compare themselves to others, and also with historic parameters, allowing the prediction of the machine's performance.
- Cognition Level, where information is presented to help decision-making and optimization of the entire system
- Configuration Level, where the information is passed back to the physical world. Also acts as a supervisory controller of the whole network, applying the decisions taken at the Cognition Level.

2.4 Industrial Controllers

Industrial controllers are essential components in industrial manufacturing systems. They are designed to control and monitor various industrial processes and ensure that these processes run correctly, smoothly and efficiently.

2.4.1 Programmable Logic Controllers

PLCs were introduced at the end of the 1960s as a replacement for hard-wired panels. The main advantages these controllers presented when compared to the hard-wired panels are their smaller size, and the facility to re-program, without needing to remove them from the shop floor. Additionally, they are easier to maintain and repair, more durable and robust, and have the ability to communicate with a central data collection system. Also, they have a lower cost [36].

PLCs are divided into 3 parts: processor, power supply, and Input/Output (I/O) modules. In the smaller PLC, they are all integrated into a single element. In the cases where there is a necessity for more I/O, modules are available to expand the standard construction. The IEC 1131 international standard defines four programming languages for PLCs: ladder logic, sequential function charts, function blocks, and a text language. The most common one is the ladder logic [36].

PLCs still are the most used industrial controller, due to their robustness and reliability. But, as each manufacturer developed their own lines of PLC, with multiple expansion modules for all types of I/O, and developed systems to integrate all their modules, their further development generated a problem with the implementation of the Industrie 4.0 initiative. When a company or organization relies on technology that a particular manufacturer develops, it can become difficult to integrate that technology with other systems and technologies, impacting the ability of the organization to make changes or upgrades. This is the case with PLCs, as each manufacturer uses their own standards and creates barriers to the integration with systems from competitors. As companies want to improve their manufacturing system's flexibility and interoperability, this presents a disadvantage in the development of new projects that want to establish a fully functional CPPS [5].

2.4.2 Field Programmable Gate Arrays

Field Programmable Gate Array (FPGA) are programmable integrated circuits that can be reconfigured after manufacture to perform specific functions. They offer several advantages compared to traditional controllers. FPGAs are highly flexible devices that can be reconfigured to adapt to changing requirements, making it possible to make quick design changes without the need for additional hardware. They are known for their high-performance capabilities, offering faster processing speeds and lower latency than traditional microprocessors. They are also well-suited for applications that require parallel processing, as they can perform multiple tasks simultaneously [37].

2.4.3 Programmable Automation Controllers

A Programmable Automation Controller (PAC) is a type of controller that combines a PLC's features with a personal computer's processing capability. With more processing power and speed compared to PLCs and being programmed with more high-level languages, PACs can handle more complex control functions. PACs can be easily integrated with other computer systems and networks, facilitating the exchange of data and information. Their versatility is shown by their ability to handle digital and analogue control variables and functions. Also, PACs are designed with modularity as a core feature, allowing for a more straightforward adaptation to changes in process requirements and ensuring their long-term use as a flexible solution [38].

2.4.4 Single Board Computers and Single Board Microprocessors

Single Board Computer (SBC) are computers composed of a processor and a memory, all in a single circuit board. Typically they have a limited number of physical I/O interfaces and are designed to be compact and low-cost. Most of them can run open source Operating Systems, like Linux or Windows, and wireless connection through WI-FI and Bluetooth. In addition, SBCs often have various types of communications protocols, to connect to sensors, actuators, and other peripheral devices. The processors used in SBCs are also optimized for low power consumption and cost rather than performance [4], [39].

These characteristics make them suitable for testing embedded applications and systems and also functioning as an edge controllers, where some pre-processing is done before data is shared with the network. Some basic control functions

are implemented, or as a smart sensor, by allowing the transmission of data of one/multiple sensors through wireless communication [39].

Single Board Microprocessor (SBM), as SBCs, are implemented on a single board with a small size. They have some interfaces and support some of the same communication protocols that support SBCs but have a smaller processing capacity, making them unable to control large machines. Usually, they run a small program on a loop embedded in their memory called firmware. SBMs are primarily focused on showing the capacities of the microprocessors they incorporate rather than utilizing them as a foundation for low-cost technology evolution[39].

In table 2.1 are presented some industrial controllers based on SBC and SBM. The presented interfaces, as well as their processing capabilities, allow them to be integrated into an industrial environment, also making them suitable for developing and testing new systems and more advanced sensors/actuators. As a big disadvantage, most of these controllers need proprietary software, limiting the full potential of these new-generation controllers.

Table 2.1: Examples of new-generation industrial controllers

Controllers	CPU and OS	Memory	Interfaces	Communication Protocols
z45 Industrial Controller [40]	•Atmel ARM9 •Linux	•128 MB RAM •256 MB Flash •MicroSD slot	•Ethernet • RS-232 •RS-485 •USB-A •MicroUSB •4 Digital/Analog Inputs •2 Digital Outputs	•Ethernet •WI-FI
ESP32 PLC 58 [41]	•ESP32 WROOM 32UE •No OS - Firmware	•4 MB Flash •520 KB RAM •MicroSD Socket	•Ethernet •RS-232 •RS-485 •21 Digital Inputs •16 Analog Inputs •15 Digital Outputs •9 Analog Outputs	•Ethernet •WI-FI •Bluetooth LE •I2C •UART •SPI
Raspberry PLC 38R [42]	•BCM2711, Cortex-A72 •Linux or Raspberry OS	•4 or 8GB RAM •MicroSD Socket	•Ethernet •USB-A •RS-485 •miniHDMI •AUX •4 Digital Inputs •8 Analog Inputs •6 Digital/Analog Outputs •16 Relay Outputs	•Ethernet •WI-FI •Bluetooth LE •I2C •UART •SPI
IceBlock [43]	•Not specified •Nxt Control Software	• N/A	•4 Digital Inputs •4 Digital Outputs •Can be customized	•WI-FI
RevPi Compact [44]	•BCM2837B0, Cortex-A53 •Linux or Raspberry OS	•1GB RAM •8GB Flash	•Ethernet •USB-A •RS-485 •HDMI •8 Digital Inputs •8 Analog Inputs •8 Digital Outputs •2 Analog Outputs	•Ethernet

2.5 Communication Protocols

Communication Protocols are used by every device to enable connection with sensors, actuators and other machines. Communication protocols can either be wireless or wired. Wired protocols are more reliable, but have a smaller range of operations. On the other hand, wireless protocols can have a wider area of action but are more susceptible to noise and data loss [45].

In Industry 4.0 development, both types are critical for the development and integration of new technologies in manufacturing systems[6].

2.5.1 I^2C

The Inter-Integrated Circuit (I^2C) protocol is a serial synchronous half-duplex protocol, based on the master-slave paradigm, developed by Philips Semiconductors in 1982, aiming to simplify connections and logic components on Printed Circuit Board (PCB). This protocol uses two lines for the communication, Serial Data (SDA) and I^2C Serial Clock (SCL), both bi-directional and pulled to the HIGH logic value. Devices connected can be either a master, responsible for starting communication, or a slave, that responds to the master request[45].

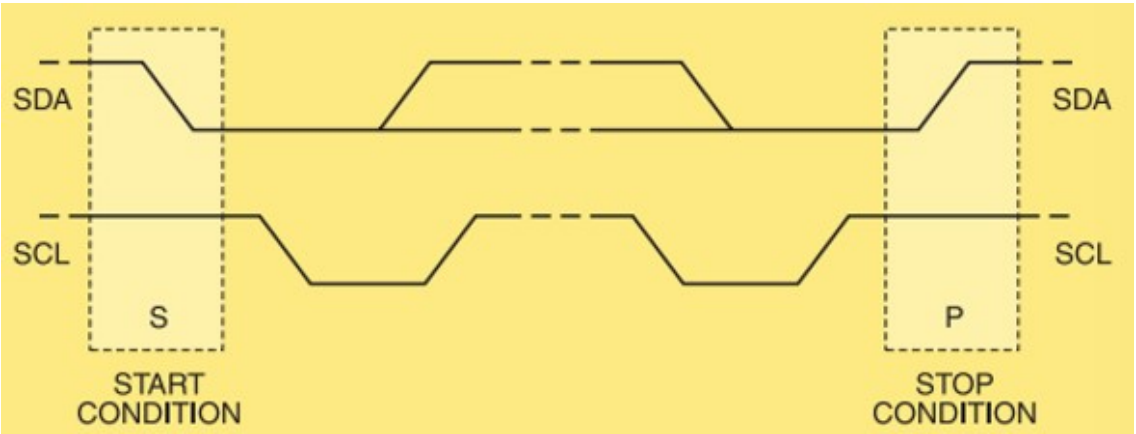

Figure 2.5: Start and Stop bits for I2C communication [46]

This protocol supports having multiple masters connected to the same lines, and due to its physical implementation, it already has a mechanism to solve collision problems. Every slave has a 7-bit address, meaning that, in theory, each master can connect to 128 different slaves. Data rates can reach up to 3.4 Mb/s[46].

The data is transmitted a byte at a time, followed by an acknowledgement from the receiver. When a master wants to start communicating, it pulls the SDA line to logic value LOW and starts writing the 7-bit address of the intended device

and 1-bit to let the slave know if it needs to transmit data or if he will receive data. The intended slave sends the acknowledge, and then communication starts. To end communications master will pull the SDA line to the HIGH logic value when the SCL line is HIGH[46]. Figure 2.5 shows the start and stop bits for the I^2C communication protocol, and figure 2.6 shows an example of a transmission sequence.

START	Slave Address	R/nW	ACK	Data	ACK	...	ACK	STOP
1 bit	7 bit	1 bit	1 bit	8 bit	1 bit	n-bits	1 bit	1 bit

Figure 2.6: Example of a transmission sequence of I2C on SDA line

2.5.2 UART

UART, or Universal Asynchronous Receiver Transmitter, is a serial asynchronous full-duplex protocol. As the name implies, it does not need a clock to synchronize the transmission of data, so the transmission data rate must be defined on the devices before starting communication. It only needs two lines, one connected between the transmitter of one device and the receiver of the other and another between the receiver of the first device and the transmitter of the second. The lines are pulled to the logic value HIGH when idle. It only allows for two devices to be connected in the same interface [45], [47].

Data is sent through data packets. Each data packet is composed of a start bit, 5 to 9 data bits, and a stop bit. There is also the possibility of adding a parity bit to help with data loss. The start bit occurs when the transmitting device pulls the idle line to the LOW logic value for a full bit transmission time. The data bits are variable, depending on the information. The parity bit, when defined, is set to 0 when the number of 1's transmitted in the data packet is even and set to 1 when the number of 1's is odd. The stop bit can vary in length, between 1/2 bit, 1 bit or 2 bit transmission time [47].

2.5.3 SPI

The Serial Peripheral Interface (SPI) protocol is a serial full-duplex protocol, based on the master-slave paradigm, developed by Motorola in 1979. To establish the SPI protocol, it's necessary a minimum of four different lines of communication. SPI Serial Clock (SCLK), to establish the clock signal from which all other signals will synchronize. A line to send data from the master to other devices, known as

Master-Out Slave-In (MOSI). A line to send data from the devices to the master, known as Master-In Slave-Out (MISO). And a line for each device that is connected to the master, so they know when they are addressed, known as Serial Selector (SSn) [46]. Figure 2.7 shows an example of a connection between a master and multiple slaves.

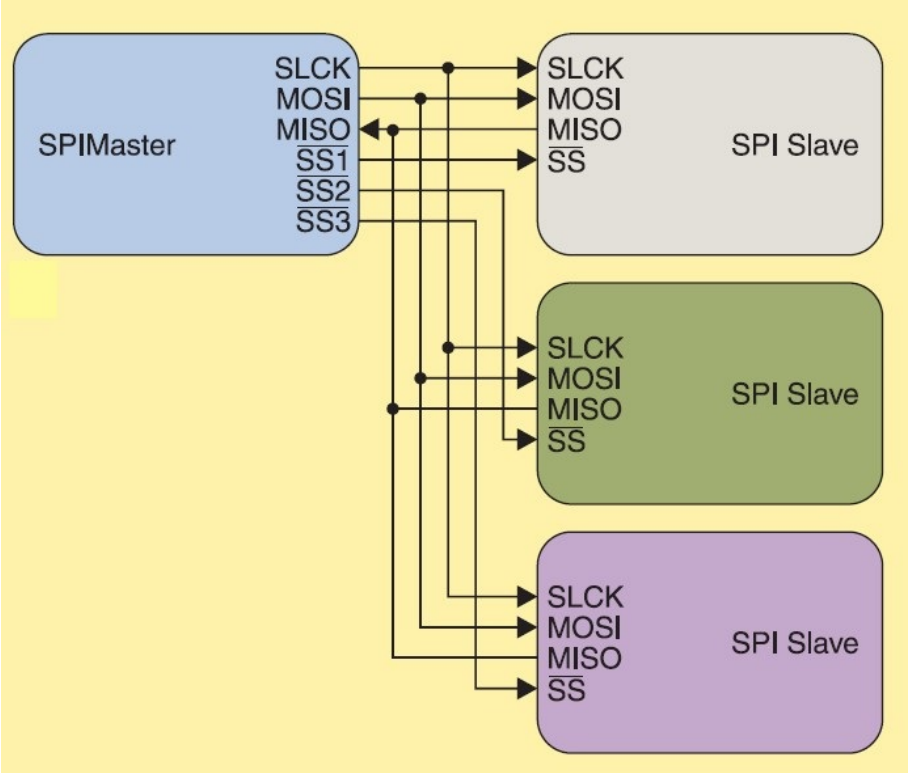

Figure 2.7: SPI topology, with multiple slaves.[46]

The SPI protocol only allows for one master on each network. When the master wants to start communicating with the slaves, he activates the SSn line of the slave he wants to communicate with and starts sending information in the MOSI line, while at the same time is sampling the MISO line. The SPI does not have a defined data rate transmission, being only limited by the SPI clock frequency, the master's capacity to interpret data, and the driving capacity of the output. In the interest of having a smooth transmission, the master and slaves must have three parameters defined [47]:

- SCKL frequency;
- CPOL, or clock polarity, which defines the idle state of the SCKL line when data is not being transferred. When is set to 0, The clock line is in the idle state LOW and when the SCKL line signal transitions from LOW to HIGH,

it indicates the start of a data transfer. When CPOL is set to 1, the inverse occurs;

- CPHA, or clock phase, defines when data is written and read on the different lines, in a clock cycle. When set to 0, data is read on the transition from idle to active level, and written on the transition from active to idle level. When CPHA is set to 1, the inverse occurs;

2.5.4 Wi-Fi

Wi-Fi, or wireless fidelity, is a wireless communication protocol based on the IEEE 802.11 standard. The IEEE 802.11 is developed by the Institute of Electrical and Electronics Engineers and covers most Wireless Local Area Network (WLAN) protocols, how they should be implemented and how to establish communication between the members of the network. Networks are considered WLANs when they cover an area from around 10 meters up to 100m. The first standard was released in 1999, and subsequent modifications of this standard have been differentiated by adding a letter in alphabetical order [48].

The most common way 802.11 networks are implemented is through the use of access points and clients. Access points are connected to the wired network, and receive and send all communication messages to any client, such as laptops or other electronic devices. The other method that is used for the establishment of 802.11 networks is the ad-hoc networks, composed only of clients that communicate directly with each other [48], [49].

The original 802.11 protocol operated on the 2.4 GHz band, with a data rate of up to 2Mbps. The 802.11b, launched in 1999, enhanced the data rate to 11 Mbps. In 2003, the 802.11g standard pumped up the data rate to 54 Mbps on the 2.4GHz band, while also implementing the 802.11a, which allowed the same data rate, but on the 5GHz band. In 2009, with the development of Multi Input Multi Output (MIMO), the 802.11n standard once again raised the maximum data rate to 500Mbps on the 2.4 and 5 GHz bands. The optimization of MIMO technologies led to the development of the 802.11ac, which allowed a maximum transmission rate of 6.93 Gbps, with the constraint of only being able to be implemented on the 5GHz band. Finally, the most recent standard, the 802.11ax, has the ability to go up to 9.6 Gbps, being able to work on both transmission bands [48], [49], [50].

2.5.5 Bluetooth

The Bluetooth protocol was released in 1999, developed by a Special Investment Group (SIG), composed of Ericsson, IBM, Intel, Nokia, and others, with the aim of connecting devices at short range, without the necessity of using cables. It is based on the IEEE 802.15.4 standard for Wireless Personal Area Network (WPAN). For all the versions of the Bluetooth protocol, the data is transmitted using the 2.4 GHz band, like Wi-Fi [51].

Bluetooth networks have two types of typologies. The Piconet topology is a unique WPAN, where a Master controls up to 7 active Bluetooth slaves. Master can communicate either in point-to-point or point-to-multi-point, and controls when a slave can send information. Slaves can only talk with their Masters. Besides active mode, slaves can go on sniff, hold or park mode, on instructions from the master, to reduce power consumption. A Master can have up to 255 devices connected on park mode. The other topology is the Scatternet. A Scatternet is the junction of multiple Piconets. The Piconets can share multiple slave devices between them, and therefore exchange data between Masters through the shared slaves [48].

The original Bluetooth protocol, or Bluetooth 1.0, has a data rate of 1 Mbps. In 2004, Bluetooth 2.0 was published, with an Enhanced data rate (EDR) that would allow communication up to 2 Mbps. In 2007, an update was done to the protocol, referred to as Bluetooth 2.1, that allowed for a data rate of 3 Mbps, and a simpler way of connecting two Bluetooth devices. Bluetooth 3.0 was launched in 2009. With this new specification, also known as Bluetooth High Speed, the data rate was increased to 24 Mbps, by including some of the 802.11 specifications on their architecture. In 2010, a new version of the protocol, Bluetooth 4.0, was published. Its main focus was Bluetooth Low Energy (BLE), which is the smaller and optimized version of the Bluetooth 1.0 protocol, allowing Bluetooth devices that use this protocol to significantly reduce their power consumption. Versions 4.1 and 4.2 of the Bluetooth protocol had improvements in security and energy consumption. In 2016, Bluetooth 5.0 was launched. This new version pumped up the data rate to 48 Mbps for the normal version and improved the 4.2 version of BLE, improving its range and reliability [51], [52],[53].

2.5.6 Ethernet

Ethernet appeared in 1976, developed by Bob Metcalfe and David Boggs, with the aim of connecting the first personal computers developed by Xerox between them. The classic Ethernet ran at 3 Mbps. In 1983, the 802.3 standard was presented, adapted from the classic Ethernet, with a speed up to 10 Mbps. At this

time, Ethernet would use a single cable to connect all computers in a building, and the necessity to have a repeater amplifying the signal every couple hundred meters[48].

In the 1990s, new standards emerged. 803.2u, or Fast Ethernet, improved the transmission to 100Mbps. 802.3ab improved once again the speed by a factor of 10 from the previous standard to 1Gbps. Ethernet has continued to evolve, with higher-speed protocols like 40 Gigabit Ethernet and 100 Gigabit Ethernet, making it one of the most dominant LAN technologies in use today [48].

In recent years, industrial communication protocols based on the Ethernet have gained a significant role in the industrial communication market, substituting the older Fieldbus protocol, as Ethernet enables higher speeds and easier connections between multiple levels of a company meaning the possibility of establishing complete vertical and horizontal integration of an automation system [54]. Examples of these Industrial Ethernet Protocols are EtherNet/IP, the PROFINET, EtherCAT and Modbus TCP/IP.

2.5.7 OPC-UA

The Open Platform Communications Unified Architecture (OPC-UA) emerged as a non-proprietary communication protocol, with a Service-Oriented Architecture (SOA), based on the TCP/IP protocol stack. The protocol developers, OPC foundation, pretended to bridge a gap set by the multiple companies in the industrial control business, that create their own standards leading up to heterogeneous systems with interoperability concerns [55].

Initially, OPC-UA relied on a client-server model, where the OPC-UA server would receive requests from OPC-UA clients, and would send the response back. In 2018, an updated version of this protocol saw the introduction of the publisher-subscribe mode. Publisher devices send data, and subscribers receive the data from the publishers to they subscribe [55]. In [56], some scenarios of integration using the OPC-UA in CPPS, with increasingly more complex scenarios with interoperability concerns that can be resolved by the use of OPC-UA.

In the beginning, the OPC-UA was mainly used on a higher level of the automation pyramid. In 2020, the OPC-UA launched the Field Level Communications initiative, to extend the communication protocol to Time Sensitive Networks (TSN), mainly the ones based on the Ethernet. Therefore, OPC-UA aims to establish an interoperability standard at all levels of the industrial pyramid, promoting the vertical and horizontal integration of enterprises and manufacturing processes [55], [57]. Figure 2.8 shows the areas OPC-UA already has a standard developed.

Notice how similar this figure is to the ones describing the Vertical and Horizontal Integration, showing how relevant is this protocol for today's industrial scenarios.

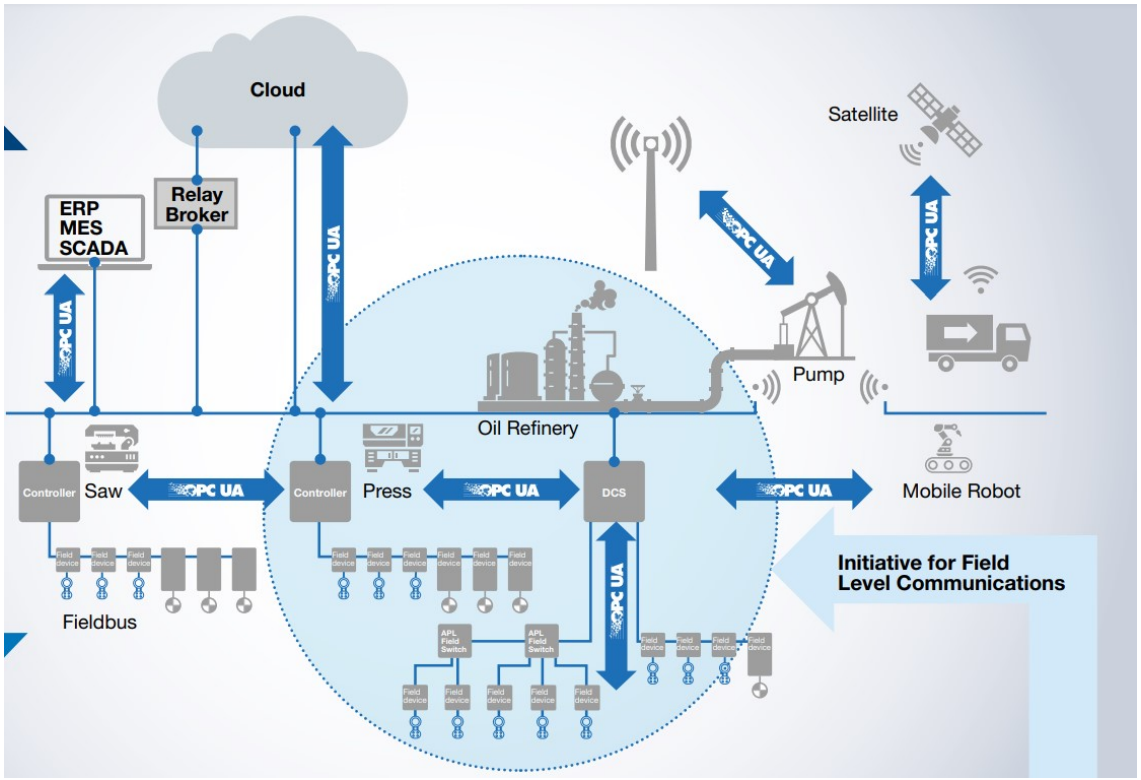

Figure 2.8: OPC-UA interoperability and areas of connection[57]

3

Architecture

This chapter presents the developed architecture for the low-cost controller, which is based on the trends presented in chapter 2. It also presents the Processing Unit Module and its main characteristics.

3.1 Proposed Architecture

The base concepts behind the idea of development for the low-cost controller are reconfigurability and modularity. A controller that is highly reconfigurable can be adapted to multiple solutions in a short time frame, making it suitable for multiple demonstrations/projects, reducing costs and unnecessary stock of parts. With modularity in mind, almost all parts of the controller have to be designed as independent modules, that can be suited for the user's needs, either by adapting, removing or adding modules.

Analysing the controllers presented in table 3.1, all present the same four key characteristics: a processing unit, multiple interfaces, different communication protocols and integrated memory. With the key characteristics of the new non-PLC industrial controllers identified, as well as the core concepts behind the idea of the development of the low-cost controller, this architecture proposes a modular controller that is composed of different types of modules. This architecture proposes four different main types:

1. Processing Unit Module, with only one present in the controller, comprised of two of the previously mentioned characteristics, the processing unit and integrated memory;

2. Power Supply Module, responsible for powering all necessary modules of the controller;

3. Interface Module, which refers to the different interfaces that a controller possesses, and comprises the multiple interfaces characteristic;

4. Communication Module, referring to the communication protocols that a controller possesses, comprising the different communication protocols characteristics;

All these modules must be highly reconfigurable, to adapt to all types of sensors, actuators and other machines. Figure 3.1 shows a Schematic of the proposed architecture.

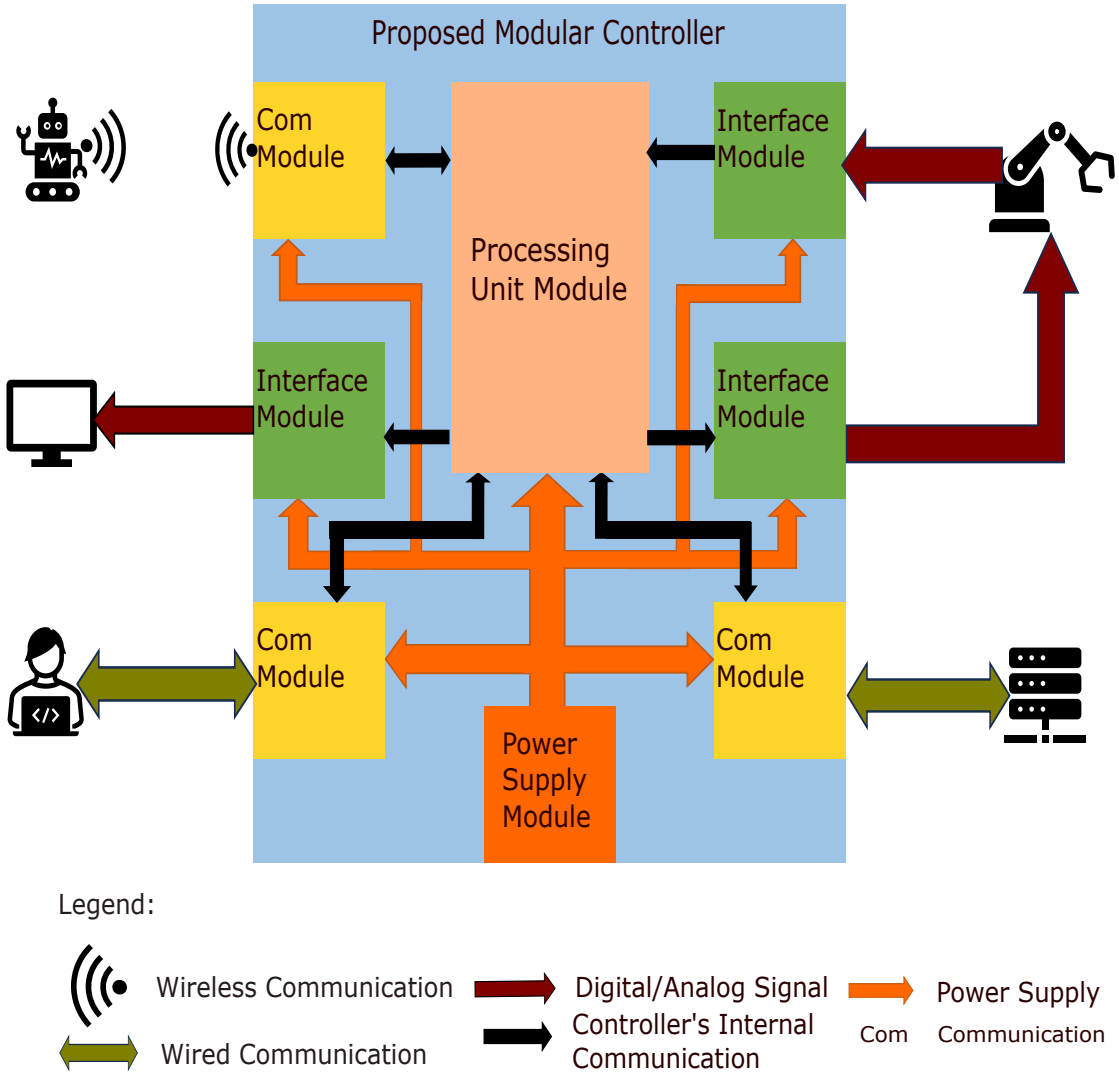

Figure 3.1: Proposed Architecture for the Modular Controller.

The core module of the controller is the Processing Unit Module. It is connected to every other module in this architecture and is where all of the control logic is realized. The Interface Module serve as a single unidirectional signal connection between the environment in which the controller is set up and the Processing Unit Module. The signal can either be generated from the processing Unit Module or from the environment. The Communication Module also serve as an interface between the Processing Unit Module and the external system, but in this case, they relate to the more complex communication protocols, like the ones presented in section 2.5. Finally, the Power Supply Module supplies all modules of the controller with power.

3.2 Processing Unit Module

This module is responsible for receiving the different signals, interpreting them and applying the necessary responses when necessary, based on coded logic. All other modules are dimensioned based on the specifications of this Module. It was defined that this Module would be based on a Single Board Computer or Microprocessor, due to the available interfaces, the abundance of information available on multiple sources, their low cost and easy manipulation and usage. To decide which one would be more suitable, research was done to identify the specifications of different options. The main criteria specified for this research were the cost, available interfaces, the dimensions of SBC/SBM, adaptability to different environments, availability, user-friendliness, and processing capabilities. The cost, dimensions and interface criteria are the three with the most weight, followed by the adaptability, user-friendliness and processing capabilities. The results of the selection process are presented in the table 3.1.

The first fact that we conclude is that all devices have similar interfaces and power options. The ESP-32 is the cheapest, costing approximately 3 times less than the next cheapest. It is also smaller than the others, making the design of the controller and incorporation of the peripherals possibly easier. The BeagleBone is the one with more GPIO, improving the controller's versatility. Also, having Wi-Fi and Bluetooth is a plus, allowing wireless communication with different devices and integration in different environments.

Regarding software development, the ESP-32 is programmed through the Arduino IDE, like the Arduino Due and the Arduino Mega 2560 Rev3, or the ESP-IDF. The first one is the most used, having multiple libraries already developed by third parties for the ESP-32. The Raspberry Pi and the Beagle Bone run Ubuntu or similar operating systems, making them powerful devices and allowing them to run more complex algorithms and computations. But, they are also less user-friendly, as it is harder for most people to develop and implement solutions with them.

For these reasons, the ESP-32 is the device chosen to be the Processing Unit Module in the low-cost controller. As many options are available from different suppliers, the one used in this thesis was the ESP-32 NodeMcu WiFi CP2102 Module from JOY-IT. In figure 3.2 can be seen the pinout of the ESP-32.

Table 3.1: Comparison of different SBC and SBM

Single-Board Computer/ Single-Board Microprocessor	**ESP-32 Node MCU [58]**	**Arduino Due [59]**	**Arduino Mega 2560 Rev3 [60]**	**RaspberryPi Model 4B [61]**	**Beagle Bone Black [62]**
Chip	ESP32-D0WD-V3	Atmel SAM3X8E ARM Cortex -M3 CPU	Atmega 2560	Broadcom BCM2711 Quad core Cortex-A72 (ARM v8)	Sitara AM3358B ZCZ100
Cost	12,71 €	42 €	42 €	57,69 €	47,81 €
Power Options	- USB Port - VIN and GND pins: Supply from 7 to 12 Volts on VIN pin -3,3V and GND pins	- USB Port - Battery - AC-DC adapter Supply from 7 to 12 Volts	- USB Port - Battery - AC-DC adapter Supply from 7 to 12 Volts	-USB Port -PoE HAT	- USB Port - 5VDC via expansion header
Memory	- 520 Kb SRAM - 448 Kb ROM -Supports outer QSPI flash and outer RAM	-256 to 512 Kb Flash -32 to 100 Kb SRAM	-256 Kb of Flash -4 Kb of EEPROM -8 Kb of SRAM	-Up to 8GB of RAM -SD slot	-512Mb RAM -4GB Storage -SD slot
Interfaces	-30 GPIO -1 SPI -1 I2C -2 UART	-54 GPIO -1 SPI -2 I2C -2 UART	-54 GPIO -1 SPI -1 i2c -3 UART	-40 GPIO -1 UART -2 SPI -1 i2c -Ethernet	-92 GPIO -1 UART -2 SPI -1 i2c -Ethernet
	-15 ADC pins -25 PWM pins -2 DAC pins -9 Touch Sensors pins	-12 ADC -12 PWM pins	-16 ADC pins -15 PWM pins	-26 PWM pins	-8 PWM pins
Communication Protocols	-WI-FI -Bluetooth	N/A	N/A	-WI-FI -Bluetooth	-WI-FI -Bluetooth
Dimensions (cm)	4,9x2,6x0,7	10,1x5,3x1,2	10,1x5,3x1,3	8,5x5,6x1,6	8,6x5,3x0,4

3.2.1 ESP-32 NodeMcu Pins and Functions

There are 3 types of pins available:

- 4 Power pins, that are used to power up the ESP-32 or to supply with a reference value to other circuits. These are VIN pin, 3.3V pin, and 2 GND pins;
- 1 EN pin, that is responsible for enabling the ESP-32 chip, that by default is set on HIGH;
- 25 GPIO pins, are used to connect sensors, actuators and other devices

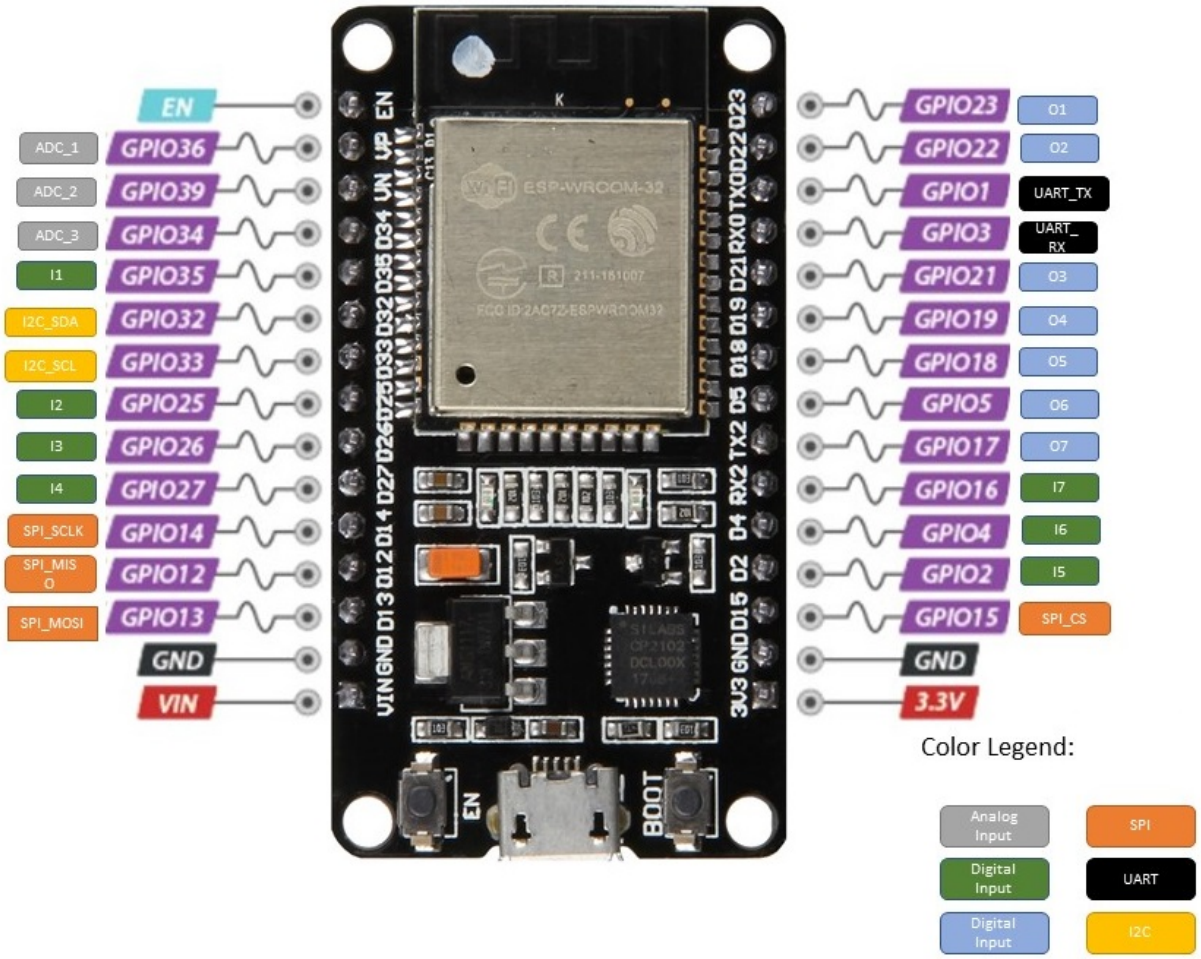

Figure 3.2: ESP-32 with the functions of GPIO pins allocated.

through communication protocols. Most of these pins have more than one function;

As some of the GPIO pins have limited functions, it is essential that the functions are well distributed, to assure as many options of communication protocols and interfaces as possible. In table 3.2 is presented the number of different modules allocated on this implementation.

3.2.2 Code Interface

As shown in section 3.2.1, all pins have allocated functions that require different modules to correctly process the signal, either to the controller or the outputs. As the hardware connections are set, it is difficult to understand which GPIO pin

Table 3.2: GPIO Allocation

Module	*Nº of Modules Implemented*	*GPIOs Allocated*
Digital Input	6	GPIO35 GPIO27 GPIO25 GPIO2 GPIO26 GPIO4 GPIO16
Digital Output	6	GPIO23 GPIO19 GPIO22 GPIO18 GPIO21 GPIO5 GPIO17
Analog Input	3	GPIO36 GPIO34 GPIO39
Wired Communication	3	GPIOs 1&3(UART) GPIOs 32&33(I2C) GPIOs 12&13&14 &15(SPI)
Wireless Communication	2	N/A

corresponds to a certain module, and which module is connected to a certain Input or Output port. Therefore, it was necessary to develop a library that facilitates the integration of source code from the controller with the developed modules. Figure 3.3 shows how the connection between the different components through the code interface is done. Most of the communication modules have already their own libraries for integration, not justifying the development of new interfaces for these types of communication. Therefore, all communication modules are integrated on the application level.

The chosen platform for the development of source code for the controller was Arduino IDE. Arduino IDE was developed by Arduino, a company that *"designs, manufactures, and supports electronic devices and software, allowing people worldwide to easily access advanced technologies that interact with the physical world "* [63].

This platform was chosen seeing it is an open-source platform and has an active community that can help developers with any level of skill develop their projects. Also, the Arduino programming language is similar to C++, a common language used for embedded systems. Due to its open-source element, Arduino has over 4000 official libraries developed by companies or single developers to facilitate the integration of different technologies into the Arduino environment.

The application code developed by users will run on top of the Code Interface, which will then connect to the hardware modules and through them to other systems. Figure 3.3 shows a schematic representation of how the integration of the application code with the controller is done.

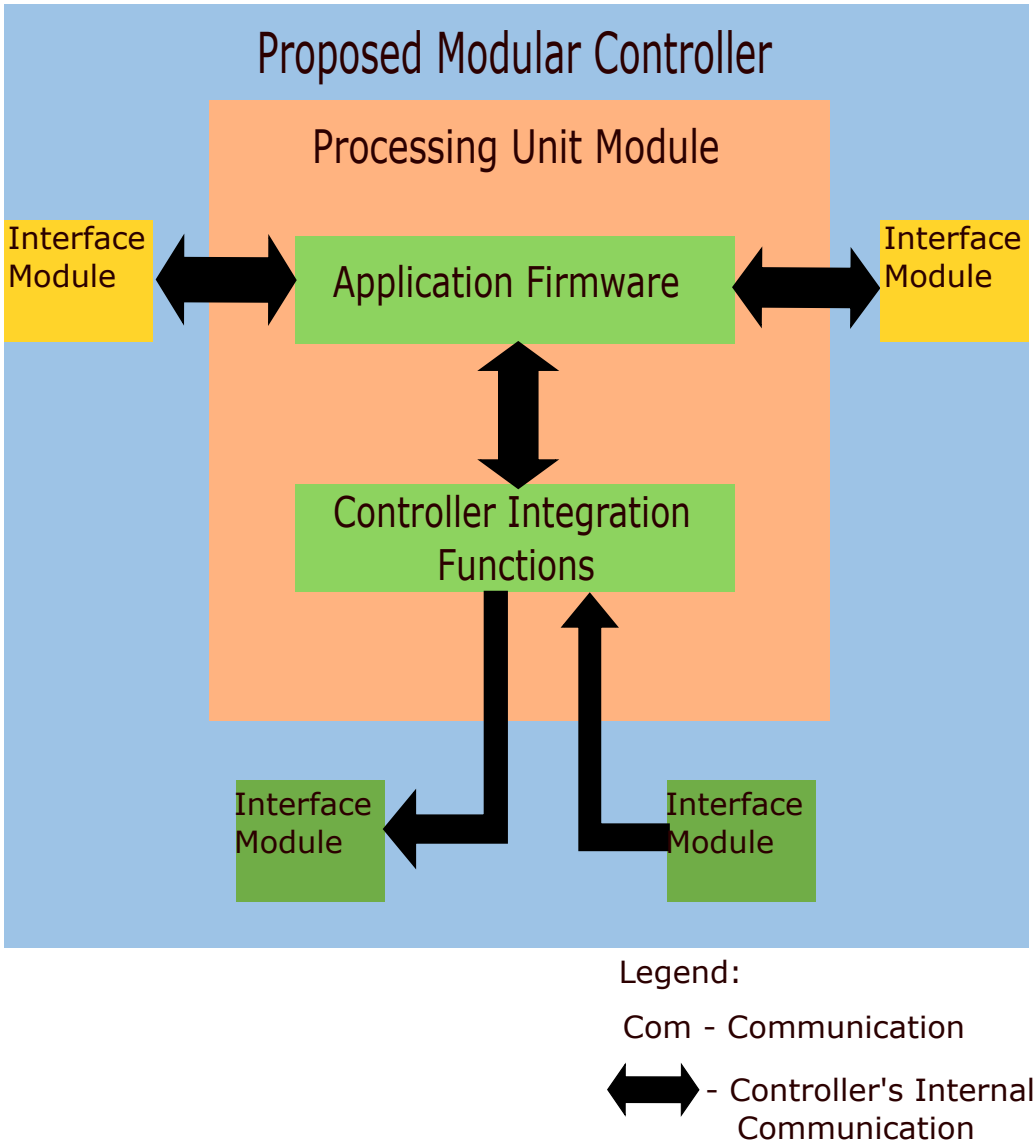

Figure 3.3: Integration of the code interface with the controller

At the base of the interface is the Arduino standard library. Basic functions allow the developer to declare a port as input or output when they are being used, according to the functions presented in figure 3.2. Pins are named according to their respective function. For example, GPIO 35 is connected to the Digital Input Module 1. When using this connection, instead of sending the number 35, as usually done while using the Arduino library, the firmware must send the string "I1"to the Controller Integration Function library, indicated on the input port, where the external wire is connected.

Table 3.3 presents all developed functions implemented in the integration

library.

Table 3.3: Developed Functions for controller integration

Function	Type	Description	Input Parameters	Output Parameters
returnDigitalInputPort (const String pinIdentifier)	private	Receives a String identifying a Digital Input Port, and returns their GPIO identifier.	String - pinIdenfier	Int - GPIO identifier
returnDigitalOutputPort (const String pinIdentifier)	private	Receives a String identifying a Digital Output Port, and returns their GPIO identifier.	String - pinIdenfier	Int - GPIO identifier
returnAnalogInputPort (const String pinIdentifier)	private	Receives a String identifying a Analog Input Port, and returns their GPIO identifier.	String - pinIdenfier	Int - GPIO identifier
initDigitalInput (const String pinIdentifier)	public	Initializes a GPIO port as a Digital Input	String - pinIdenfier	Int - Operation Success
initDigitalOutput (const String pinIdentifier)	public	Initializes a GPIO port as a Digital Output	String - pinIdenfier	Int - Operation Success
readDigitalPort (const String pinIdentifier)	public	Reads a Digital Input Port and returns its state	String - pinIdenfier	Int - Port State
writeDigitalPort (const String pinIdentifier, int value)	public	Writes a given value in a Digital Output Port	String - pinIdenfier. Int - value	Int - Operation Success
readAnalogPort (const String pinIdentifier)	public	Reads a Analog Input Port and returns the read value	String - pinIdenfier	Int - Port Value

3.3 Interface and Communication Module Description

The Interface Modules are responsible for the connection of signals between the Processing Unit Module and peripheral devices. They provide signal conditioning so the recipient of the signal can interpret it correctly. In this architecture, to differentiate between the signals that arrive at the Processing Unit Module and the ones that it produces, it is defined that the first ones are considered input signals, and the latter ones output signals. This leads to a subdivision of the Interfaces Modules: there are Input Interface Modules and Output Interfaces Modules, each providing a different signal conditioning function according to the module type.

Most of these modules will be connected to sensors, in the case of Input Interface Modules, and actuators, in the case of Output Interface Modules. Sensors are an essential part of manufacturing systems, as they allow the perception of the state of a system through the variation of a known physical characteristic, depending on the type of sensor [64]. On the other hand, actuators are used to alter the state of a manufacturing system, by receiving input signals and applying a corresponding mechanical force [65].

In the proposed architecture, when considering the development of the Input/Output Interface Modules, the only consideration done about sensors and actuators is the type of signal they produce/receive, and how the Processing Unit Module receives/sends them. Therefore, both the Input Interface Module and the Output Interface Module are subdivided into two sub-modules: Input/Output Digital Interface Module and Input/Output Analog Interface Module. Figure 3.4 summarizes the types of Interface Modules that are present in the architecture.

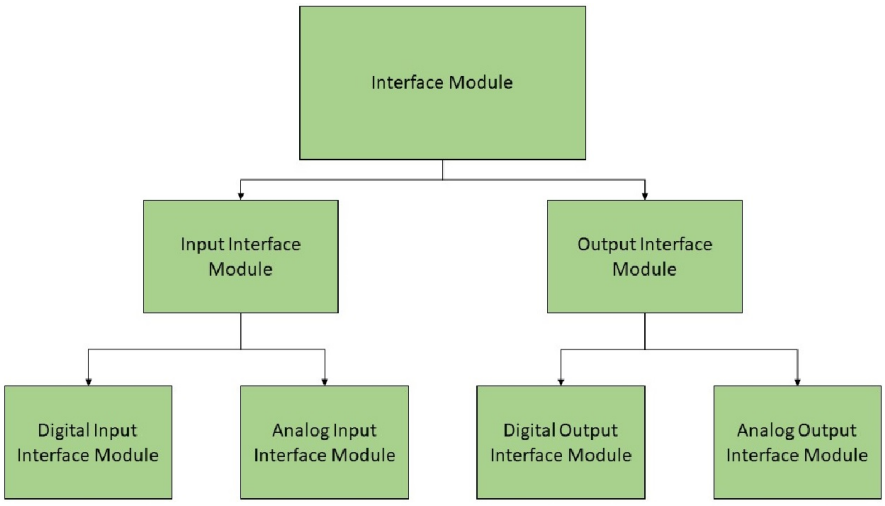

Figure 3.4: Schematic representation of the different types of Interface Modules.

The Communication Module handles the transmission of signals between the Processing Unit Module and connected devices. As presented in chapter 2.5, there are two types of communication protocols: Wired and wireless. Therefore, the Communication Module is sub-divided into two modules: the Wired Communication Module and the Wireless Communication Module. Figure 3.5 summarizes the types of Communication Modules that are present in the architecture.

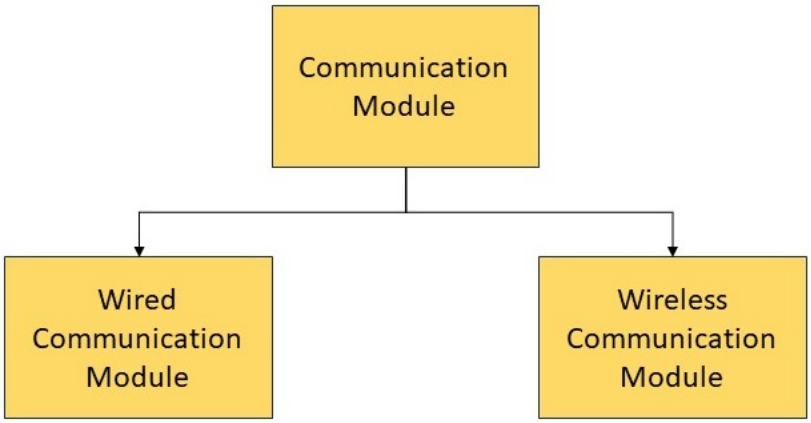

Figure 3.5: Schematic representation of the different types of Communication Modules.

4

Implementation

This chapter presents the development of the necessary modules for the low-cost controller. The specifications chosen for this controller were based on the current trends, presented in chapter 2, and the architecture defined in chapter 3.

The controller must be able to connect to various types of sensors and actuators, therefore it must provide digital inputs and outputs, as well as analog inputs and outputs. Finally, it also should integrate sensors/actuators that use the communication protocols presented in section 2.5.

For the Processing Unit Module, as many options are available from different suppliers, the one used in this thesis was the ESP-32 NodeMcu WiFi CP2102 Module from JOY-IT.

4.1 Power Supply Module Implementation

In order to power up the ESP-32, there are 3 possible options, as presented in table 3.1:

- Power through the USB port;
- Power through VIN and GND pins, making sure that in the VIN pin, the tension is between 5 V and 12 V;
- Power through 3.3V and GND pins, making sure that in the 3.3V pin, the tension is exactly 3.3 V;

Moreover, the ESP-32 cannot be powered by two of these options at the same time, as this can damage it and any other device connected to it. The safest option to power the ESP-32 is through the VIN and GND pins, as the VIN pin already has a built-in step-down conversion circuit, an electronic circuit that converts an

input tension to the desired tension, in this case, from the presented interval to 3.3 V, and protecting it from some variation in the input tension on the VIN pin.

4.1.1 Power Module

The standard tension that is used in automation equipment for control and power supply is 24 V. Most of the educational kits replicate this. So, in order to power the ESP-32, there must be a step-down circuit that converts the 24 V to the desired tension between 5 V and 12 V. As recommended by the producer in [66], the input voltage should not be higher than 7 V due to the lack of internal cooling. For these reasons, it was defined that the ESP-32 supply voltage is 7 V.

The integrated circuit LM317 has the capability to perform the required conversion. This is achieved by adding the necessary components, as depicted in figure 4.1.

To get the necessary tension on Vout, the resistances R1 and R2 have to be properly dimensioned. The datasheet of LM317 presents the equation:

$$Vout = 1.25 \times \left(1 + \frac{R2}{R1}\right) + Iadj \times R2 \tag{4.1}$$

The datasheet points out that, for most cases, $Iadj \times R2$ can be dismissed, leading to the equation 4.2.

$$Vout = 1.25 \times \left(1 + \frac{R2}{R1}\right) \tag{4.2}$$

With $Vout = 7V$, and fixating $R1 = 100\Omega$, we can calculate the necessary value for R2.

$$7 = 1.25 \times \left(1 + \frac{R2}{100}\right) \Longleftrightarrow 4.6 = \frac{R2}{100} \Longleftrightarrow R2 = 460\Omega \tag{4.3}$$

The resulting circuit is presented in figure 4.1.

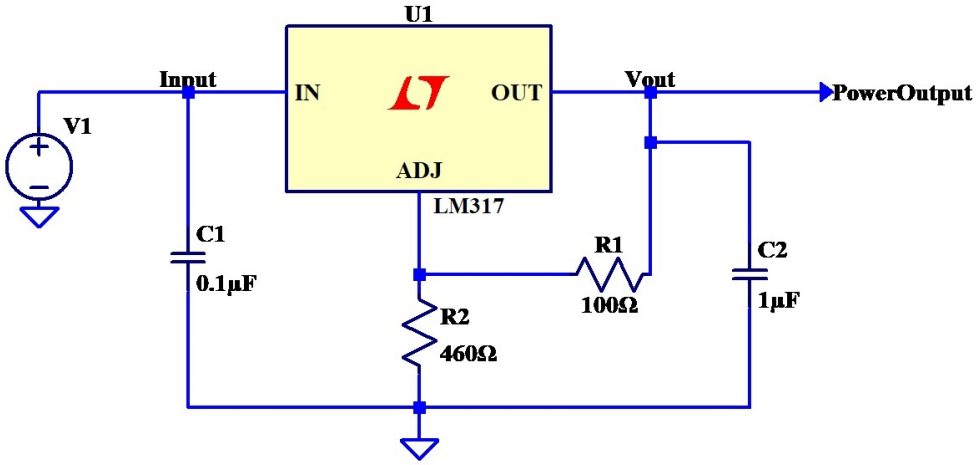

Figure 4.1: Circuit for 24 V to 7 V conversion.

4.1.2 Reference Module

For some of the circuits presented in section 4.2, there must be a reliable 3.3 V reference source. Although the ESP-32, after being powered up, can supply the 3.3V through the 3.3V pin, it is unreliable to provide many circuits. Therefore, we need a circuit that can deliver the 3.3V to all required circuits.

The chosen approach was to replicate the circuit presented in fig 4.1, and using equation 4.2, dimension the resistors to get the 3.3 V circuit. With $Vout = 3.3V$, and fixating $R1 = 100\Omega$, we can calculate the necessary value for R2.

$$3.3 = 1.25 \times \left(1 + \frac{R2}{100}\right) \iff 1.64 = \frac{R2}{100} \iff R2 = 164\Omega \tag{4.4}$$

The resulting circuit is presented in figure 4.2.

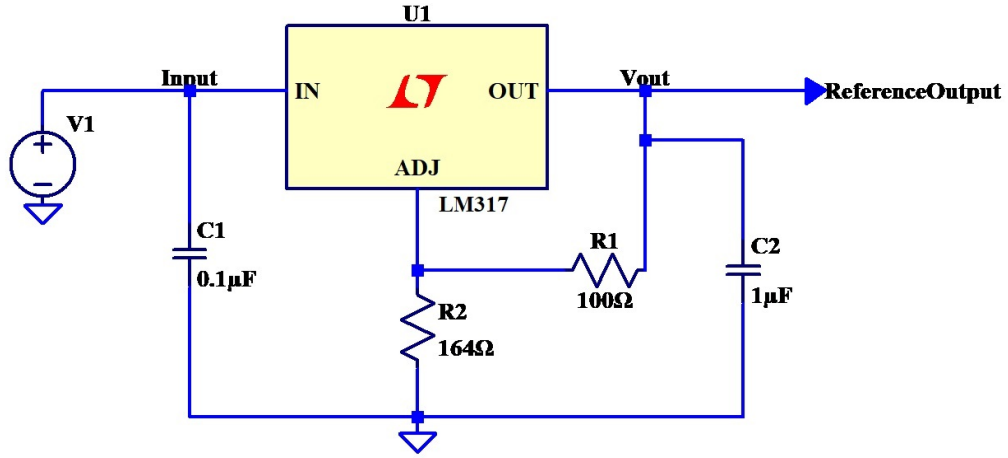

Figure 4.2: Circuit for 24 V to 3.3 V conversion.

4.2 Interface Module Implementation

As previously mentioned, the ESP-32 is only capable of handling up to 3.3V on GPIO pins. Therefore, to connect Digital or Analog Sensors and Actuators, it is necessary to process the signals, by reducing their voltage for inputs, and amplifying them for outputs.

4.2.1 Digital Input Module

In [67], the same challenge of connecting a digital input signal to a controller while working at different levels of tension was presented. The approach taken for the conversion of the digital input signals was the use of a module using optocouplers for the conversion, shown in figure 4.3.

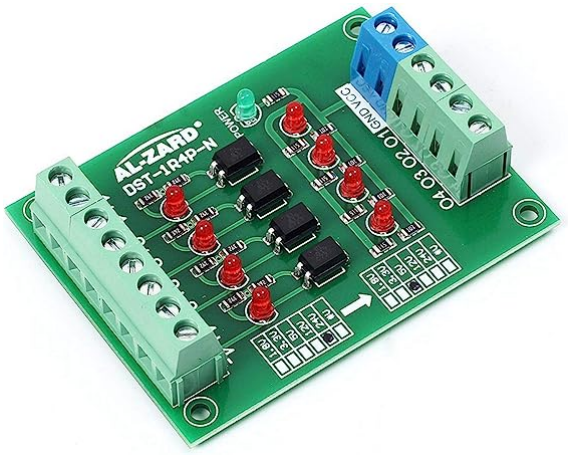

Figure 4.3: Digital Optocoupler Module [68]

An optocoupler is an optoelectronic circuit used as a separation for different electronic circuits. This separation is done through a photo transmitter and photo receiver pair. This means that no current will flow between the input and output of the optocoupler [69]. Figure 4.4 is a schematical representation of an optocoupler.

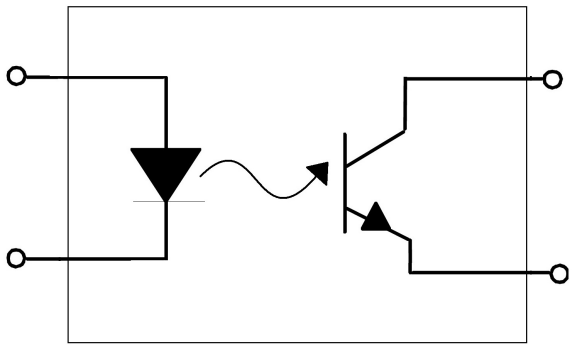

Figure 4.4: EL817 Optocoupler Schematic

Therefore, for the developed controller, the decided approach was to use the optocoupler as the base for the conversion circuit, making the necessary adjustments to guarantee its correct usage. There are many types of optocouplers, with multiple variations in the number of inputs and outputs. The optocoupler chosen for the circuit was the EL817.

Figure 4.5 shows the circuit used for the signal conversion, adapted from the circuit presented in the datasheet and the circuit presented on the optocoupler module datasheet.

Reference Input is the value used to activate the optocoupler, equal to the maximum value that the Signal Input can reach. In this case, it is 24V. Reference Output is the maximum value desired for the Signal Output, 3.3V. Signal Input represents the sensor connection.

R2 is added as a pull-down resistor, so the line stays at the LOW logic value when the line is in an idle state. R1 and R3 need to be dimensioned so the optocoupler works in the desired conditions. The optocoupler's datasheet states that for a voltage drop of 1.2 V between the diode ports, it pulls a 20 mA current value. Assuming that the signal input is 0 V, and R2 is big enough so it can be assumed as an open circuit, these conditions result in equation 4.5:

$$24 - VR1 - 1.2 = 0 \Leftrightarrow VR1 = 22.8V \tag{4.5}$$

Knowing the voltage drop on R1, and the current that passes through R1, using Ohm's Law allows us to find what is the necessary resistance value of R1, shown in equation 4.6:

$$VR1 = R1 \times IR1 \Leftrightarrow R1 = \frac{VR1}{IR1} \Leftrightarrow R1 = \frac{22.8}{0.02} \Leftrightarrow R1 = 1140\Omega \tag{4.6}$$

The R3 resistance's main objective is controlling the switch time of the optocoupler. The switch time of an optocoupler is the time that it takes to change its output based on an excitation on the input side. In this case, according to [70], a higher resistance value, over 1000Ω leads to a shorter time of response to an input excitation, but a longer time to return to the idle state after said excitation. A smaller value of the resistance leads to the opposite. The optimal interval is between the 400Ω and 500Ω, so it was chosen for R3 the value of 450Ω.

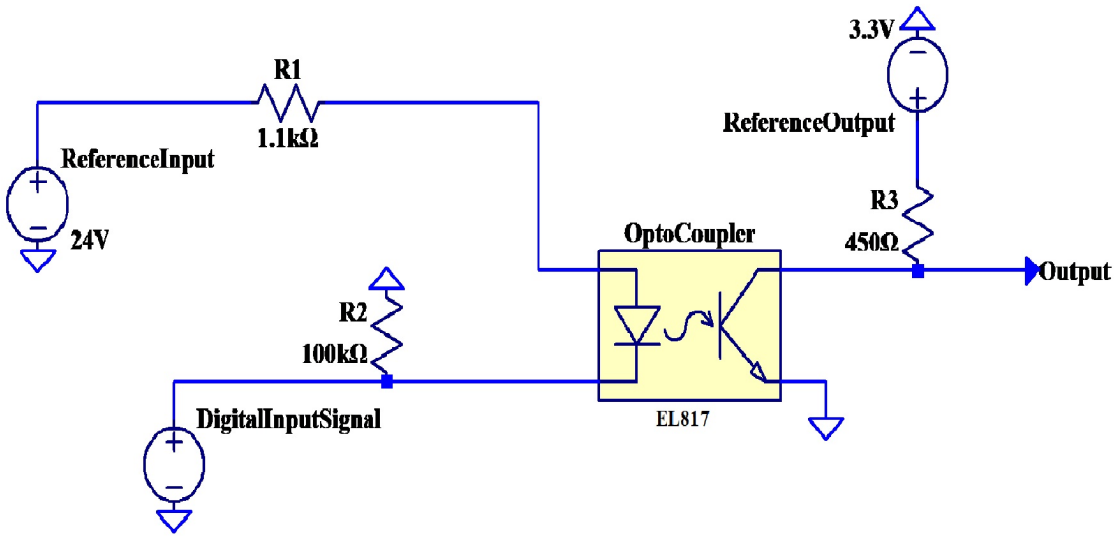

Figure 4.5: Dimensioned 24V - 3.3V Digital Input Module

4.2.2 Analog Input Module

The ESP-32 has 2 Analog to Digital Converters (ADCs), ADC1 and ADC2. The ADC1 has 8 different channels possible of which only 6 are available on the ESP-32 board. ADC2 has 10 different channels, but using these ADC pins is not recommended, as the WI-FI module of the ESP-32 uses some of the abilities of the

ADC2. In order to maximize the usage of pins and other functionalities presented by the controller, the pins used for analog input of the controller belong to ADC1.

ADC1 presents four levels of resolution, from 9 bits up to 12 bits. With 9 bits resolution, the analog range is divided into intervals of 6.4 mV, with a total of 516 different values. With the 12 bits resolution, the analog range is divided into intervals of 0.8 mV, a total of 4096 different values. As these different ranges represent different correlations between the desired value and the input value, and changing the bit resolution would lead to a change in the input circuit, it was defined that for the developed controller, the analog resolution would be 12 bits.

For the development of the analog conversion circuit, it was defined that input analog tension values are in the 0 - 24V range. Therefore, the signal would be converted from the 0 - 24V to 0 - 3.3V range, so the ESP-32 can receive the signal. This means that a value in the 0 - 24 V range, after the conversion, is always the same in the 0 - 3.3 V range, having a proportional relation between them. Therefore, the chosen approach for the analog conversion was to use a voltage divider circuit. A voltage divider circuit, shown in figure 4.6, is an electronic circuit where the output signal is a fraction of the input tension, given by equation 4.7.

$$Vout = Vin \times \frac{R2}{R1 + R2} \tag{4.7}$$

Defining R2's values as 1000Ω, also knowing that when Vin is 24V, Vout should be 3.3V, the value for resistance R1 can be found.

$$3.3 = 24 \times \frac{R2}{R1 + R2} \Leftrightarrow 3.3 \times R1 = 20.7 \times R2 \Leftrightarrow R1 = 6270\Omega \tag{4.8}$$

4.2.3 Digital Output Module

For the output digital circuit, the same approach as in section4.2.1 was taken for the design. In this case, there is no need for the pull-down resistor R2, shown in figure 4.5, since this line is directly connected to the ESP-32 pin, and stabilizes the signal. Figure 4.7 shows the circuit.

Taking equation 4.5, and adjusting the value of the Reference Input, from 24 to 3.3V, we get equation 4.9

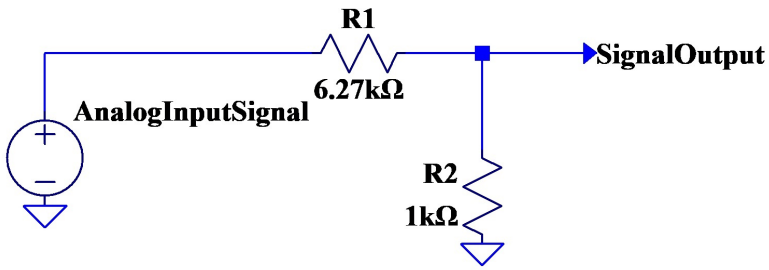

Figure 4.6: Dimensioned Analog Input Module.

$$3.3 - VR1 - 1.2 = 0 \Leftrightarrow VR1 = 2.1V \tag{4.9}$$

Knowing the voltage drop on R1, and the current that passes through R1, using Ohm's Law allows us to find what is the necessary resistance value of R1, shown in equation 4.10:

$$VR1 = R1 \times IR1 \Leftrightarrow R1 = \frac{VR1}{IR1} \Leftrightarrow R1 = \frac{1.1}{0.02} \Leftrightarrow R1 = 110\Omega \tag{4.10}$$

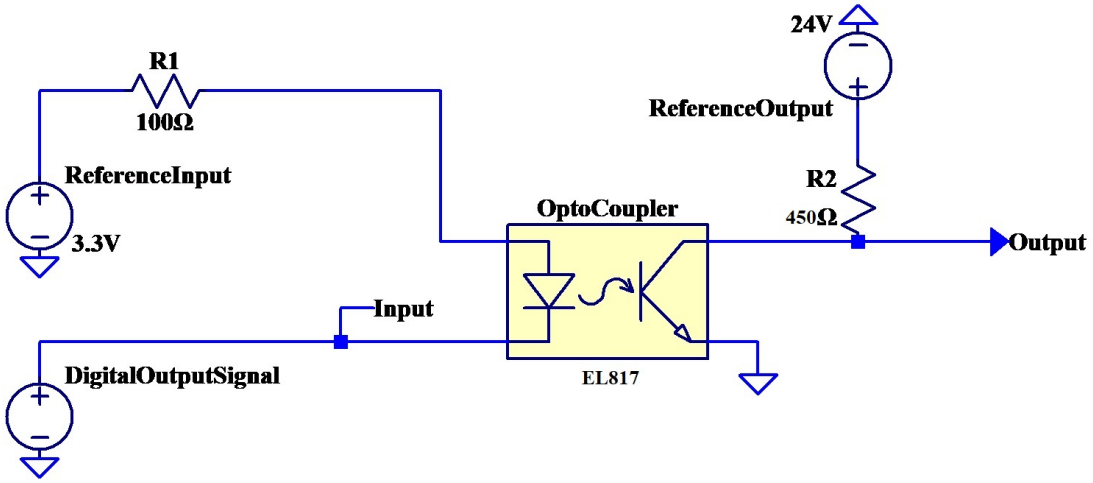

Figure 4.7: Dimensioned 3.3V - 24V Digital Output Module

As previously mentioned, resistance R2, from the circuit presented in figure4.7, the optimal value is set between the 400Ω and 500Ω, so it was chosen for R2 the value of 450Ω.

As mentioned in section 4.2.1, the optocoupler does allow current to flow from its ends. In the input circuit, this presents an advantage, as it protects the controller

from high currents. On the other side, for the output circuits, the low current makes it unable to activate actuators. Therefore, using the same approach as in [67], a voltage follower montage was done, consisting of connecting the output of the conversion circuit to the base of an NPN transistor, the collector connected to the output reference, and the signal output was taken on the emissor. Figure 4.8 shows the full-dimensioned circuit.

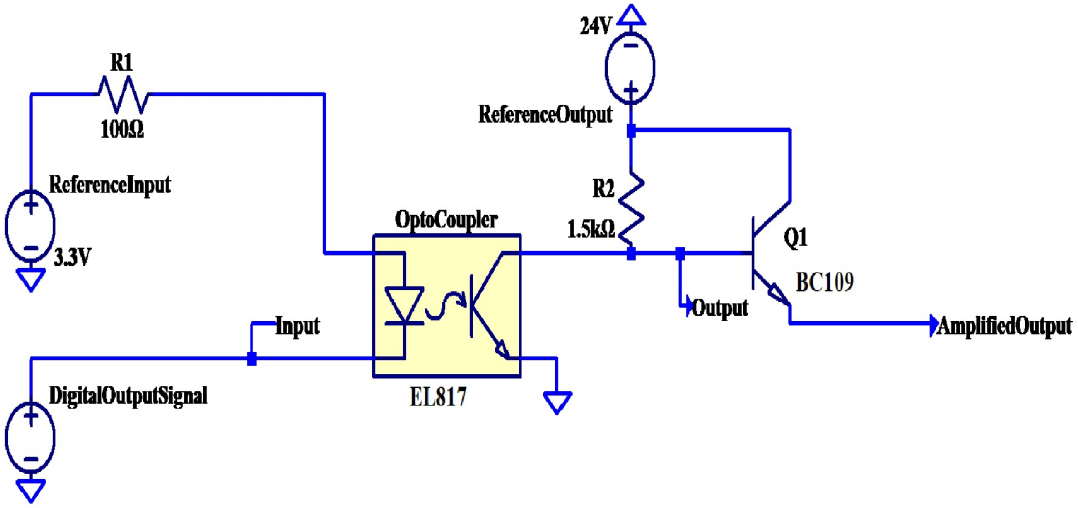

Figure 4.8: Dimensioned 3.3V - 24V Digital Output Module, with the current booster.

4.3 Communication Interface

As mentioned in section 3, Communication Modules serve as intermediaries between the Processing Unit Module and other devices. The ESP-32 Node MCU already comes with integrated hardware for the wireless communication modules, therefore it is not necessary to have any type of hardware development for these specific modules, with only integration at the software level.

For the Wired Communication Modules, there are not any limitations in terms of hardware development, with the integration of these modules(most specifically, the I2C, SPI and UART) within the software development, as the signals for the different wired communication protocols will be available at the working tension level of the Processing Unit Module(3.3V).

4.4 Controller Implementation

The controller implementation was done in four different stages:

1. Concept Validation;
2. Protoboard soldering;
3. PCB development;

Breadboard implementation was done to validate the circuits and adjust the components if necessary. The implementation on the breadboard and the results are presented in sub-chapter 5.3, as all tests with the other developed boards.

4.4.1 Protoboard Soldering

After adjusting and validating the circuits, the next approach was designing the first prototype of the controller. This prototype would also serve as second validation of the circuits, showing how they would react in a different setup.

The protoboard used in this stage requires solder to connect the components between them, also having wires soldered for some connections. Since it was only an intermediary prototype, only digital inputs and outputs were soldered, for testing. Figure 4.9 shows the protoboard prototype.

4.4.2 PCB design

In PCB prototyping, there are three key phases of the development: schematic design, PCB layout and Gerber generation. For the development of the controller's PCB, it was chosen KiCAD 7.0. KiCAD is open-source software that incorporates all three stages of PCB design and has multiple libraries with components from most electronic companies. It also has the ability to create symbols, a type of file that describes the electrical characteristics of a component, that are not on these libraries and also create footprints, a type of file that describes the physical dimensions of an electronic component.

The first step was creating the schematic for the PCB, based on the different circuits dimensioned in chapter 3, and according to the pins functions defined in figure 3.2. For the schematic, it was necessary to design the symbol of the ESP-32, since the ones presented were not from the same manufacturer, having small differences in size.

After designing the schematic, the first step is associating a reference designator with each part. Following the association, an electrical rule check was done. This

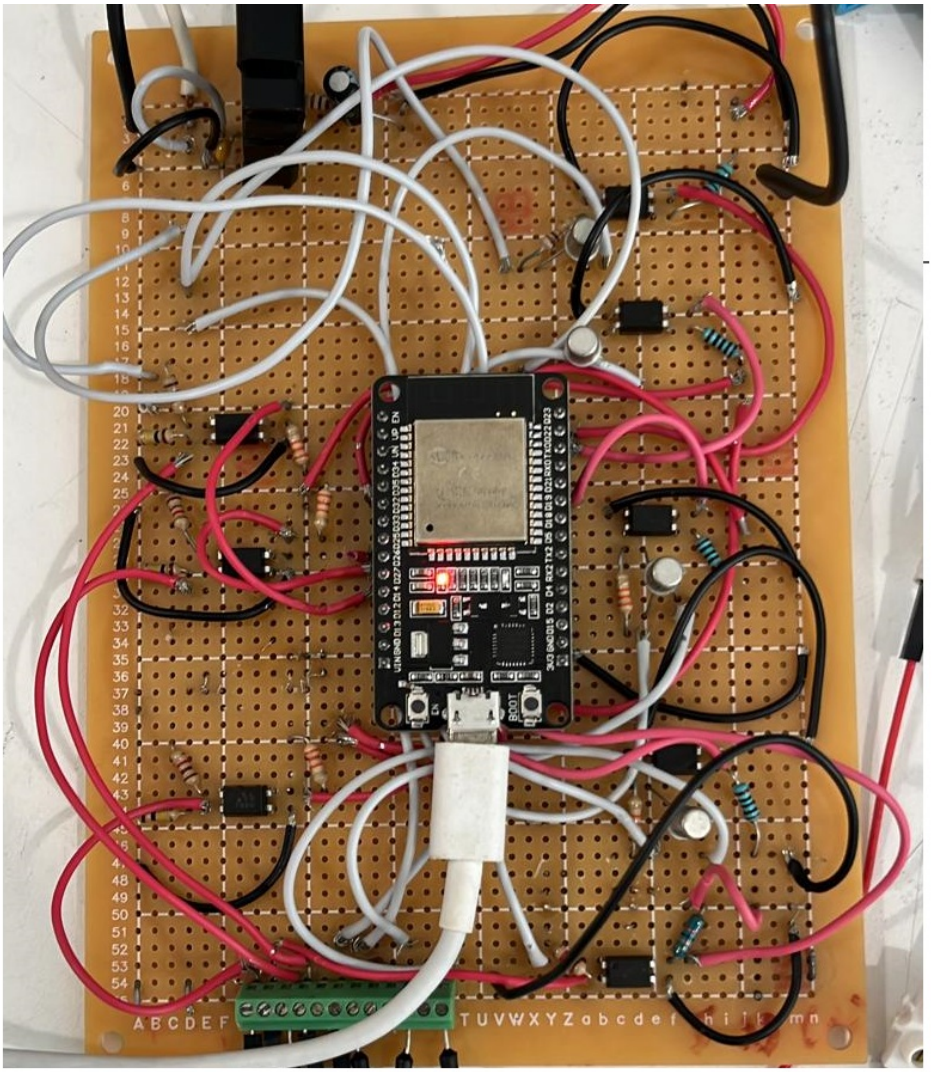

Figure 4.9: Protoboard Implementation.

ensures that no component is left unconnected and that power circuits are properly identified. The second to last step on the schematic editor is to associate a footprint with each component. Finally, the last step is generating a netlist file. A netlist file describes all the connections established between all the elements of the Schematic. In total, there are x components and y connections.

In the second phase, the netlist file is uploaded to the PCB layout editor. After grouping the components, defining the dimensions of the PCB and defining constraints to the electrical design, according to the PCB manufacturer, the electrical traces are set up through auto-routing.

Figure 4.10 shows the 3-D image of the PCB. The final step is generating the Gerber and hole drill map files, so the manufacturer can accurately produce the designed PCB.

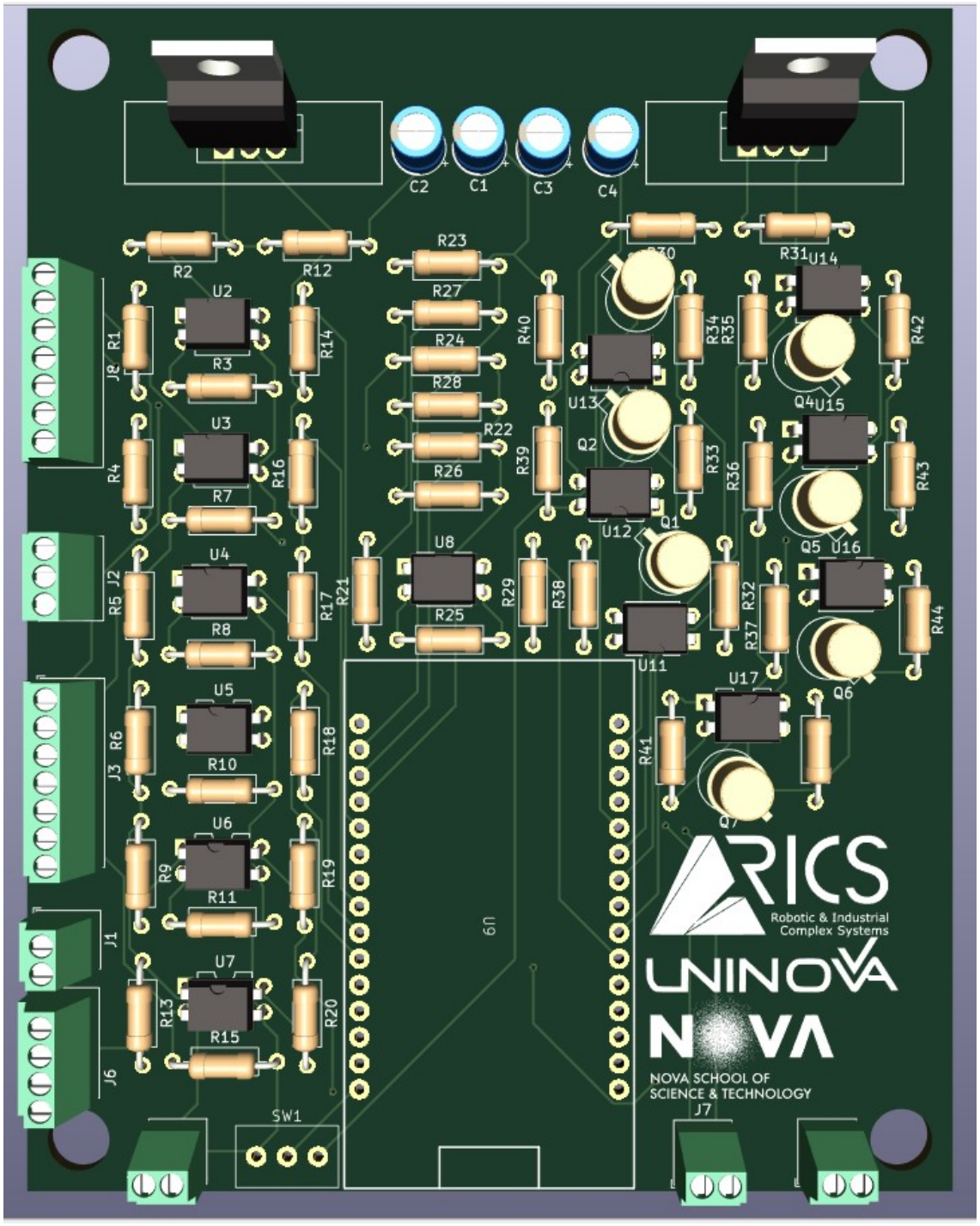

Figure 4.10: 3-D Model of the PCB

4.4.3 Controller Case

To protect both the user and the controllers, a 3-D customized case was printed for the PCB prototype. The Onshape online software was used to design the case. Using the dimensions taken from KiCad, the box was designed so the controller

could be screwed in. Figure **??** shows the final design of the box.

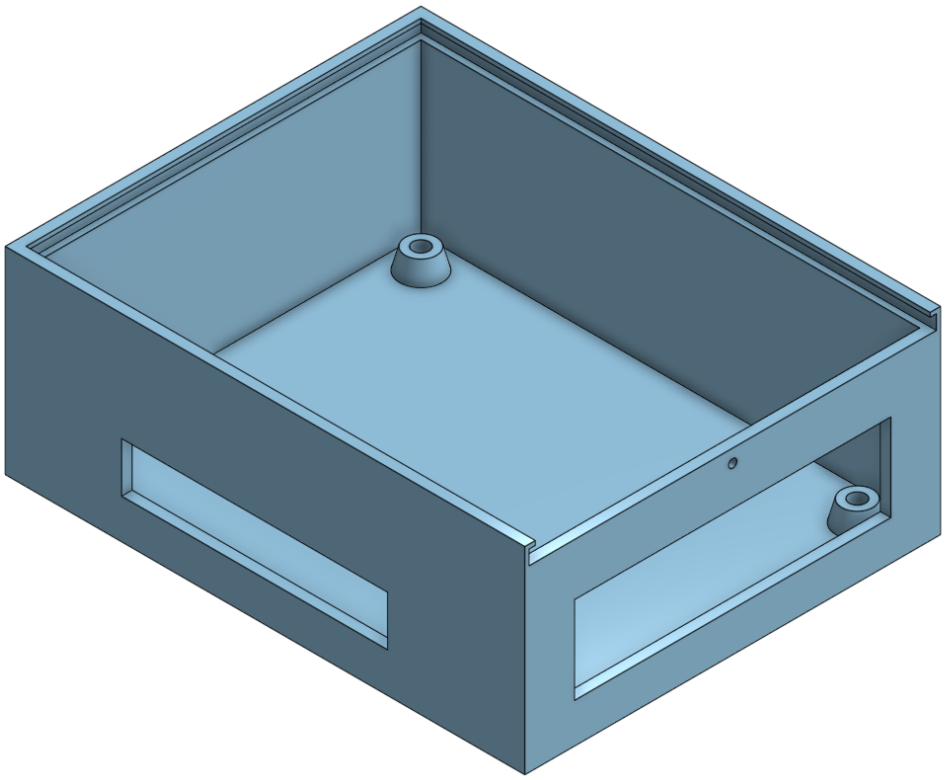

Figure 4.11: 3-D Case for the controller.

Figure 4.12 shows the final product of the development, the PCB and the case assembled together.

4.4.4 Usage precautions

Two scenarios must be taken into account while using this controller, as they can jeopardize the controller's good functioning, by burning some of the controller modules and even damaging other connected devices.

The first one occurs when uploading the code to the ESP-32 memory. ESP-32 has 5 strapping pins, that must be activated in a certain order so the firmware can be updated. In this case, the ESP-32 should be removed from the controller, to ensure the code is properly updated. The second occurs when debugging the code. To debug the code, and check the data on the Arduino IDE serial monitor, the ESP-32 must be connected to the PC using a micro USB cable. When this scenario happens, the Power Module must be switched off, as the ESP-32 cannot have 2 power sources connected at the same time.

Figure 4.12: 3-D Case and developed Controller.

5

Tests and Validation

In this chapter, all tests realized throughout the implementation process are presented.

The first step was to validate all theoretical calculations realized to dimension the different modules in sections 4.1 and 4.2. Simulating electronic circuits is the most common way to validate them before implementing them. Many free circuit simulators are available online, for people with different levels of knowledge and desired testing capabilities. The program used for the simulation of all electronic circuits in this thesis was the LTSPICE, a free SPICE-based software for circuit simulation, developed by Analog Devices. All modules were tested using a transient simulation, a type of simulation that analyses the variations of the electronic components in a specific time interval, according to the specifications established. For all modules, a 5-second interval was used.

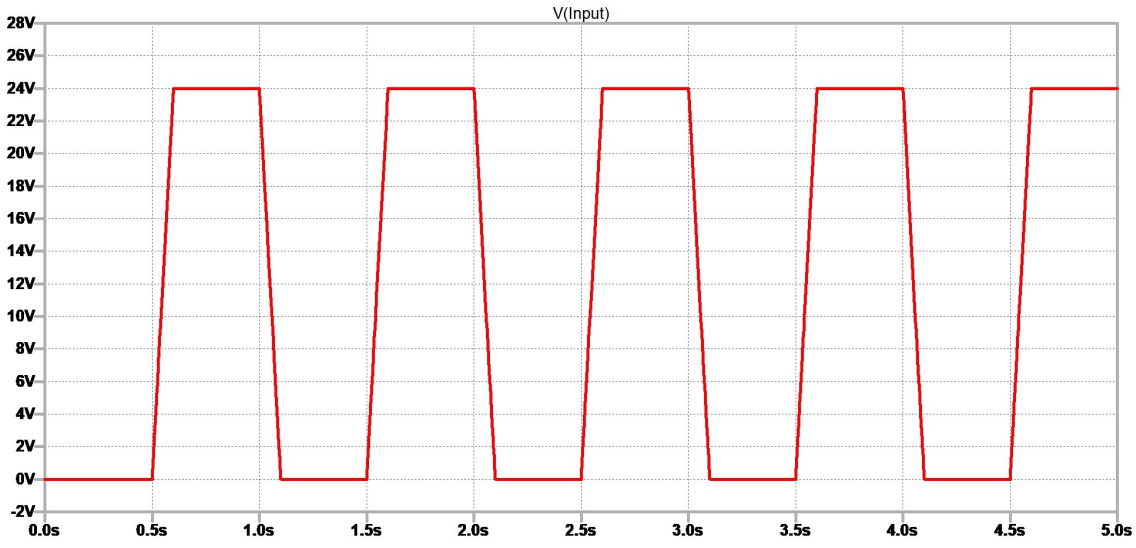

Figure 5.1: Tension variation of Input Signal for module testing.

5.1 Power Unit Module Simulation

The Power Unit module has two different sub-modules that require validation: The Power module, which powers the Processing Unit Module, and the Reference module, which supplies the 3.3V reference for some of the Interface Modules. For both modules, the same test simulation parameters were applied: a rectangular 24V power signal was supplied on the Input node of figures 4.1 and 4.1, switching between 0V and 24V, with a period of 1s, a rising edge and falling edge time of 0.1s, and 0.4s in the tension levels previously mentioned. Figure 5.1 shows the variation of the tension level in the Input node.

The first simulated module was the Power Reference Module, shown in figure 4.2. By checking the tension variation on the Vout Node, shown in figure 5.2, the tension reaches 3.3V when the input supply is 24 V and goes back to 0V when the input supply is 0V, validating the module's implementation in the theoretical level.

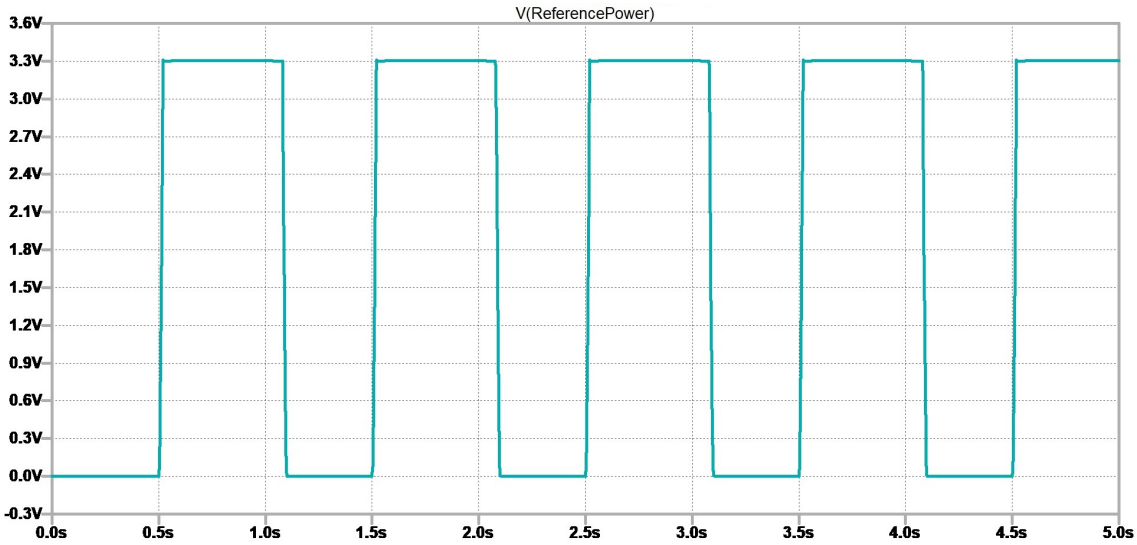

Figure 5.2: Tension variation on the Vout node of figure 4.2.

With the Power Supply Module, shown in figure 4.1, similar results and conclusions can be taken. By checking the tension variation on the Vout Node, shown in figure 5.3, the tension reaches 7V when the input supply is 24 V and goes back to 0V when the input supply is 0V, confirming the theoretical calculations.

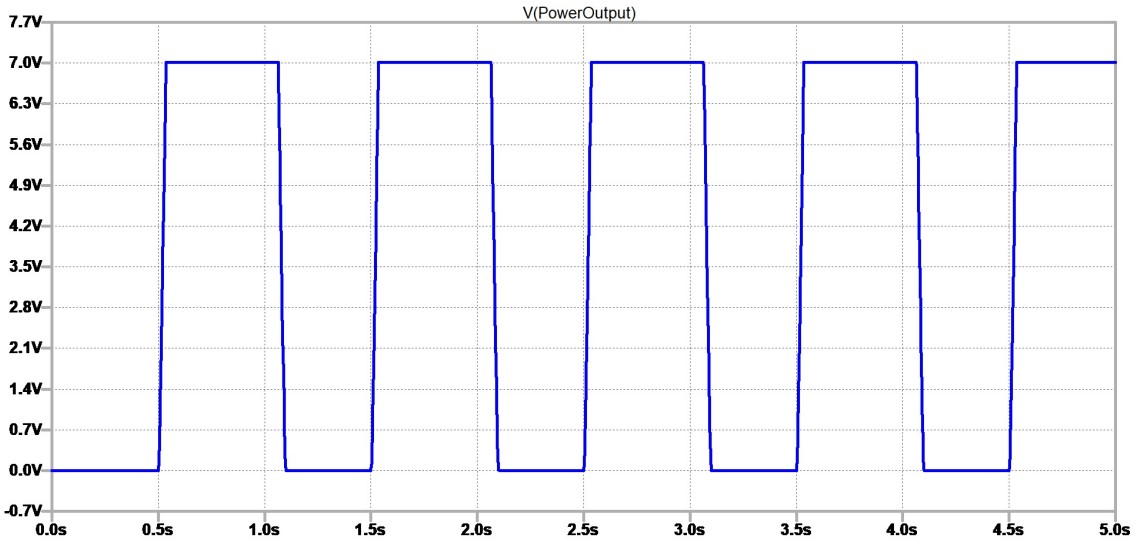

Figure 5.3: Tension variation on the Vout node of figure 4.1.

5.2 Interface Module Simulation

The Interface Module has three different sub-modules that require validation: the Analog Input Module, the Digital Input Module and the Digital Output Module.

5.2.1 Analog Input Module

The Analog Input Module does not require any tension reference value to operate. To test this module, a sinusoidal signal was supplied on the Analog Input Signal of figure 4.6, with an amplitude of 24V and a period of 1 second. Figure 5.4 shows the variation of the Analog Input Signal. This figure only shows the positive part of the signal, since that is the desired input value variation for analysis of the module's response.

Figure 5.5 shows the Signal Output tension variation. Compared with the Input Signal shown in figure 5.4, the Output Signal only changes the tension value by the calculated factor, maintaining the desired signal characteristics, and therefore validating the Analog Input Module theoretical calculations.

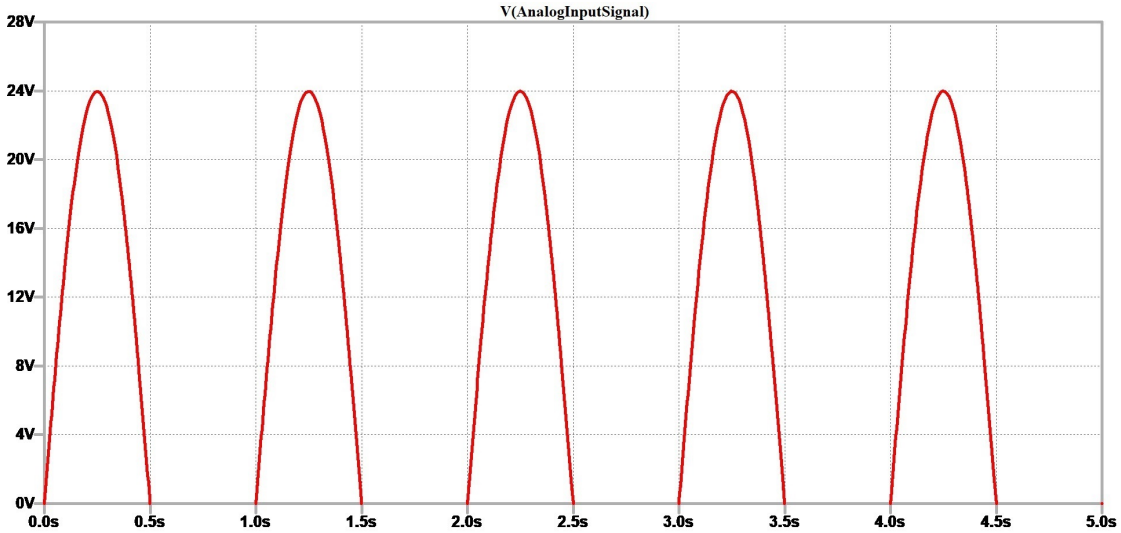

Figure 5.4: Analog Input Signal variation on figure 4.6.

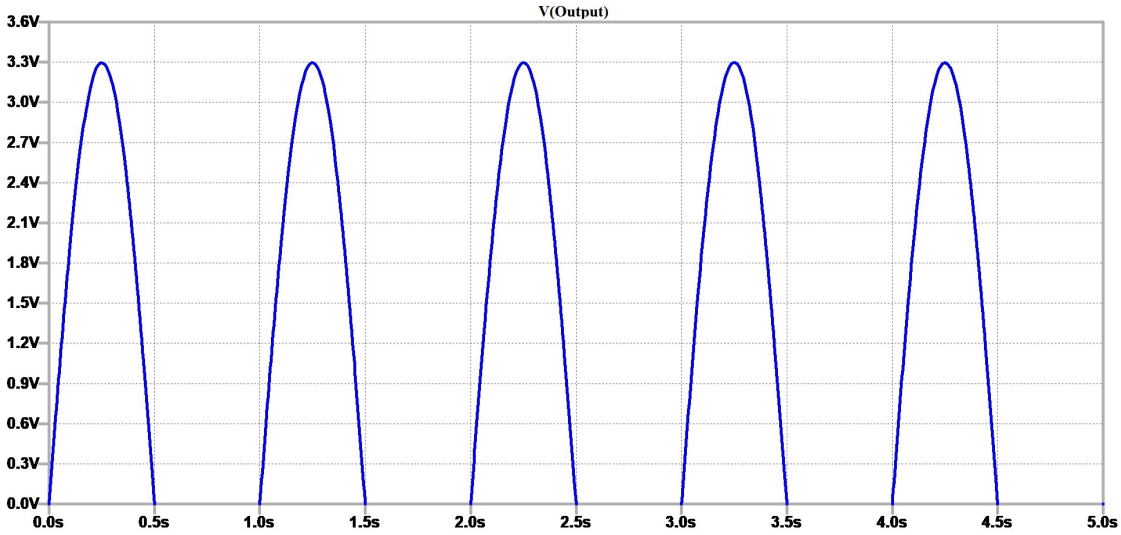

Figure 5.5: Signal Output tension variation on figure 4.6.

5.2.2 Digital Input Module

For the Digital Input Module, it is necessary 24V and 3.3V reference values to ensure the module can properly function. In the Digital Input Module's simulation, the Digital Input Signal was excited with the same signal as the Power Unit Modules, shown in figure 5.1. The resulting signal is presented in figure 5.6 When the Input signal is at 0V, the output signal does not reach 0V, showing that this module might be under-dimensioned. According to the ESP-32 datasheet, the processor will assume a LOW logic value when the tension passes below 25% of the reference value for the ESP, in this case, 3.3V. Therefore, based on this information, the

Digital Input Module is validated.

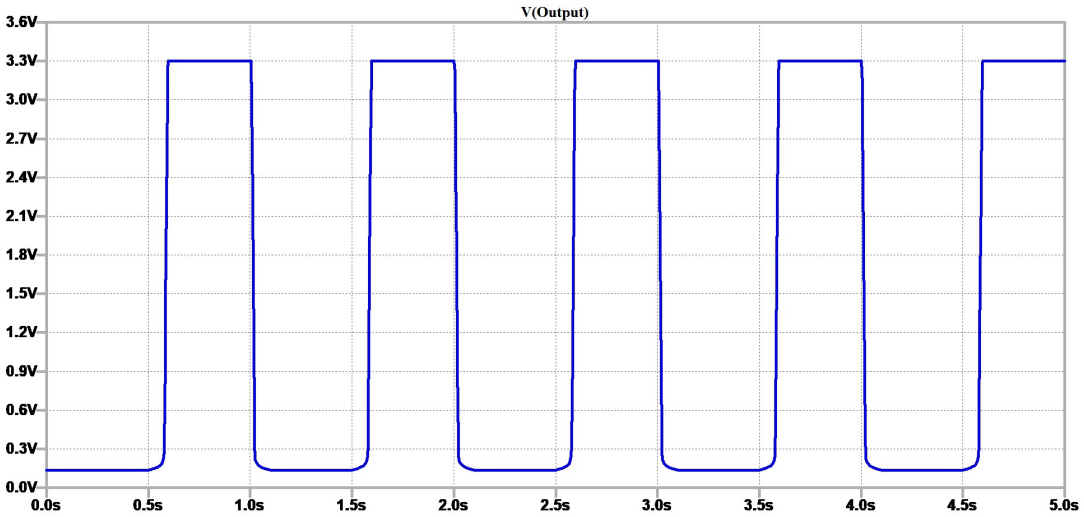

Figure 5.6: Signal Output tension variation on figure 4.5.

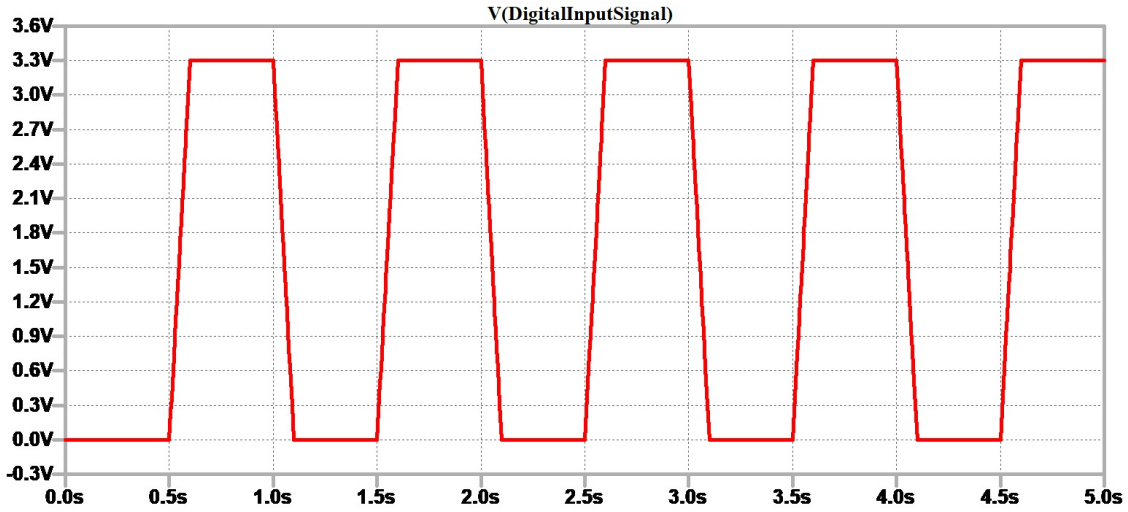

Figure 5.7: Digital Output Signal Variation on figure 4.8.

5.2.3 Digital Output Module

For the Digital Output Module, it is also necessary 24V and 3.3V reference values to ensure the module can properly function. In the Digital Output Module's simulation, the Digital Output Signal node was excited with a signal with the same characteristics as the one used for the Digital Input Module, except for the

maximum value, which was 3.3V. This signal is shown in figure 5.7. Figure 5.8 shows the resulting signal on the AmplifiedOutput node of figure 4.8. The tension on this node reaches 24V when the input supply is 3.3 V and goes back to 0V when the input supply is 0V, validating the module's implementation at the theoretical level.

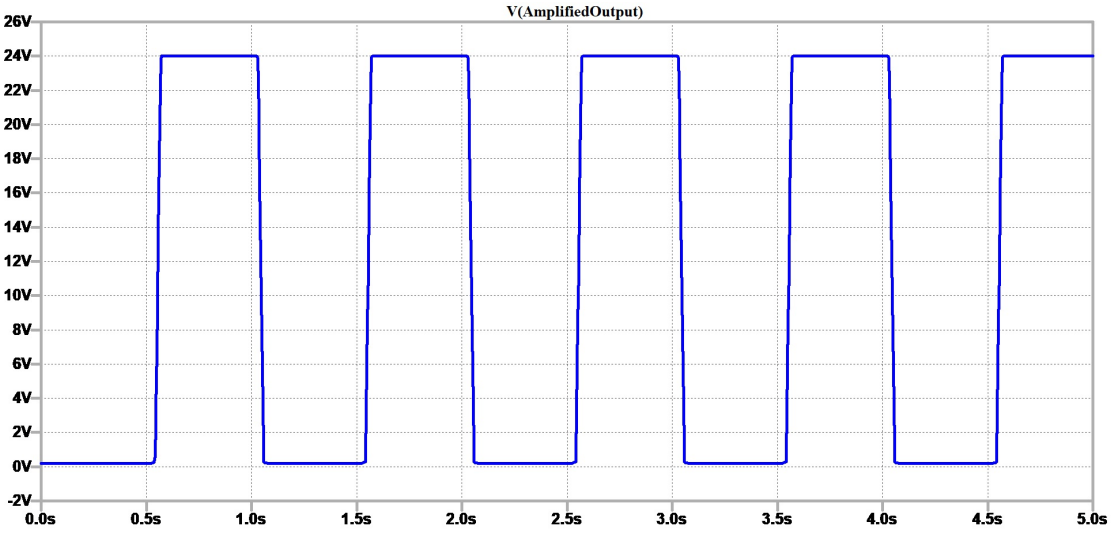

Figure 5.8: Tension variation on the AmpOutput node of figure 4.8.

5.3 BreadBoard Tests

The Breadboard tests serve to validate, in the first place, the implementation of the different modules developed, and in the second place test the implementation of the controller, using multiple modules at the same time. All tests included the Processing Unit Module, allowing at the same time the test of the Code Interface previously developed.

For most of the tests run, an Industrial Educational kit from UNINOVA was provided. This educational kit is composed of a bi-directional conveyor belt and a punching machine. The kit has 4 sensors: one photo sensor at each end of the conveyor belt, and one push sensor at each end of the punching machine track. It has 4 actuators, to move the punching machine up or down and to move the conveyor belt front or back.

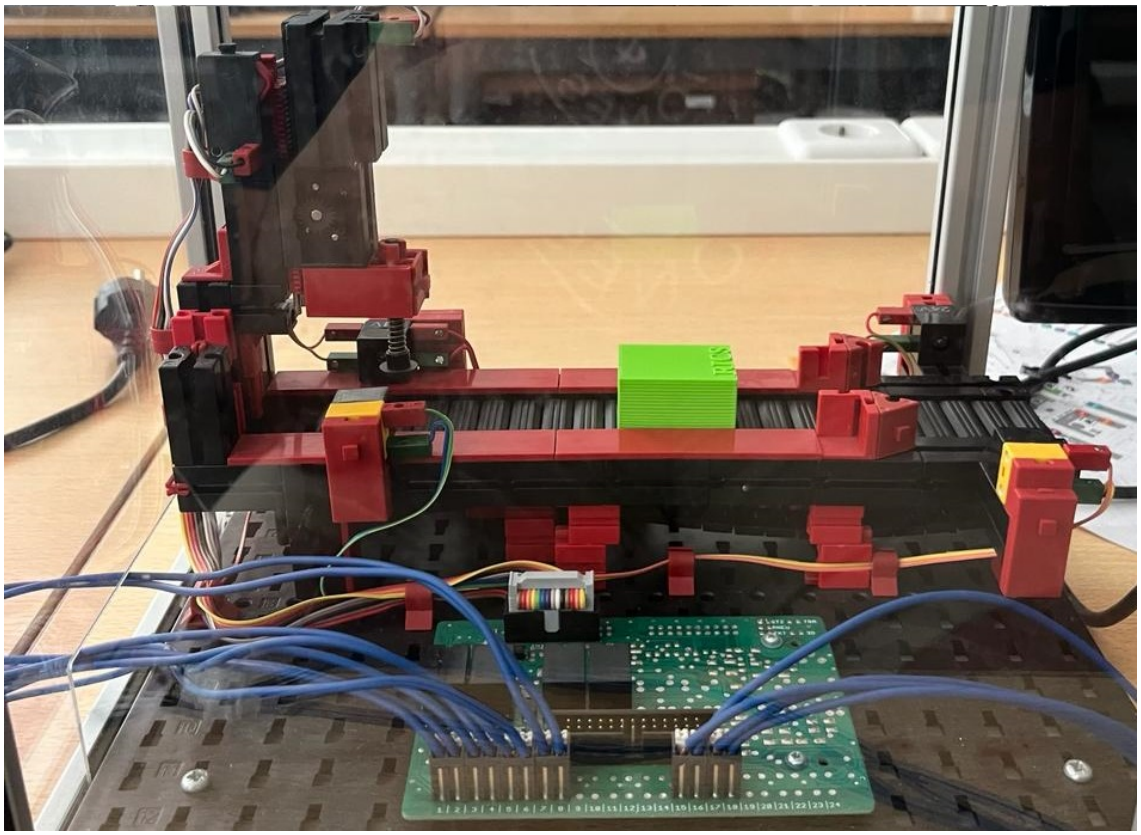

Figure 5.9: Industrial Kit provided for tests.

As the Industrial Kit needs a 24V power supply, like some of the Modules that would be tested, an external Power Supply was also used for providing the 24V power supply and Ground Reference Value.

The kit movement is presented in figure 5.10. The kit receives a piece on the entry sensor and moves it using the conveyor belt. When it reaches the machine

sensor, the conveyor belt stops. Following the arrival at the machine, it comes down, pressing the piece. When the machine arrives at the low-position sensor, it comes back up until it arrives at the high-position sensor, stopping there. This movement frees up the piece, which is then moved once again using the conveyor belt to the entry sensor, where it can be picked up.

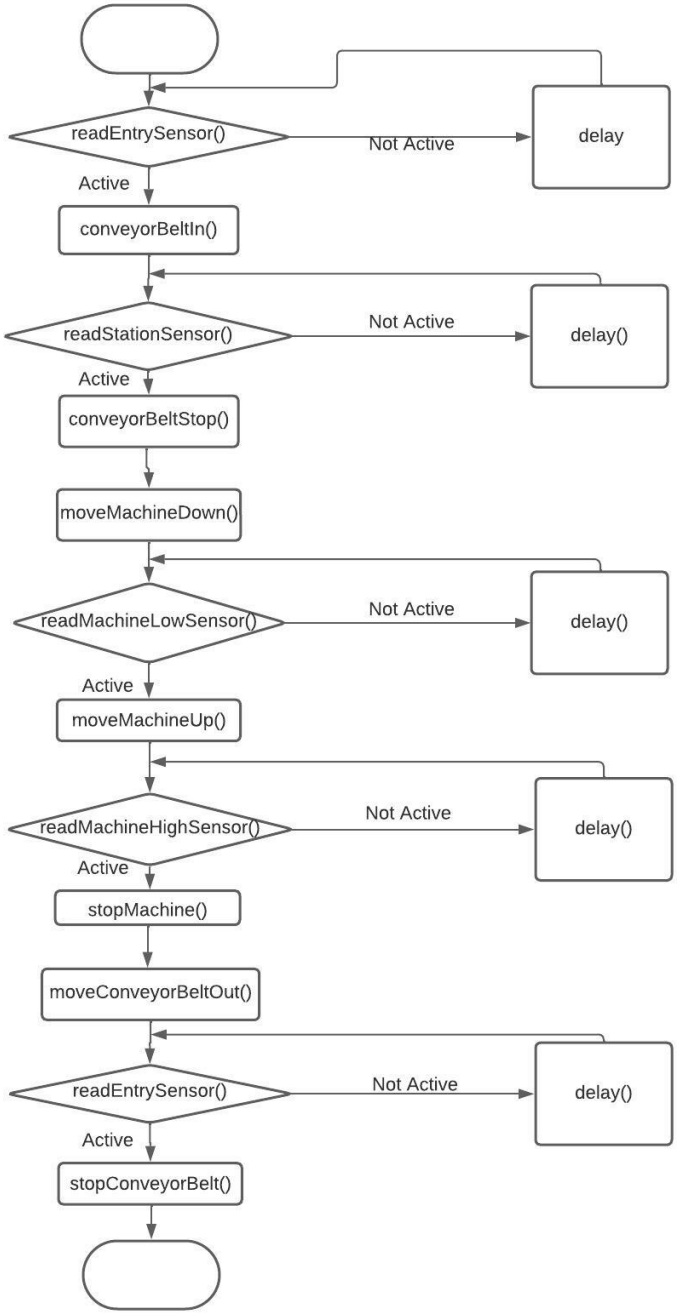

Figure 5.10: Flowchart of Educational Kit Movement

5.3.1 Proof of Concept

The first test using the Industrial Kit was to validate the Code Interface, making sure that the developed functions were working correctly and that the Processing Unit Module could operate with multiple inputs and outputs. Instead of using the dimensioned Digital Interface Modules, as they were still not validated on a breadboard and using them could cause more problems while troubleshooting for errors, the Digital Optocoupler Modules, shown in figure 4.3, were used as interfaces for the Digital Input and Output Interface. The Processing Unit Module was set on a breadboard, along with the necessary transistors for the voltage follower montage. Figure 5.11 shows the setup for the test.

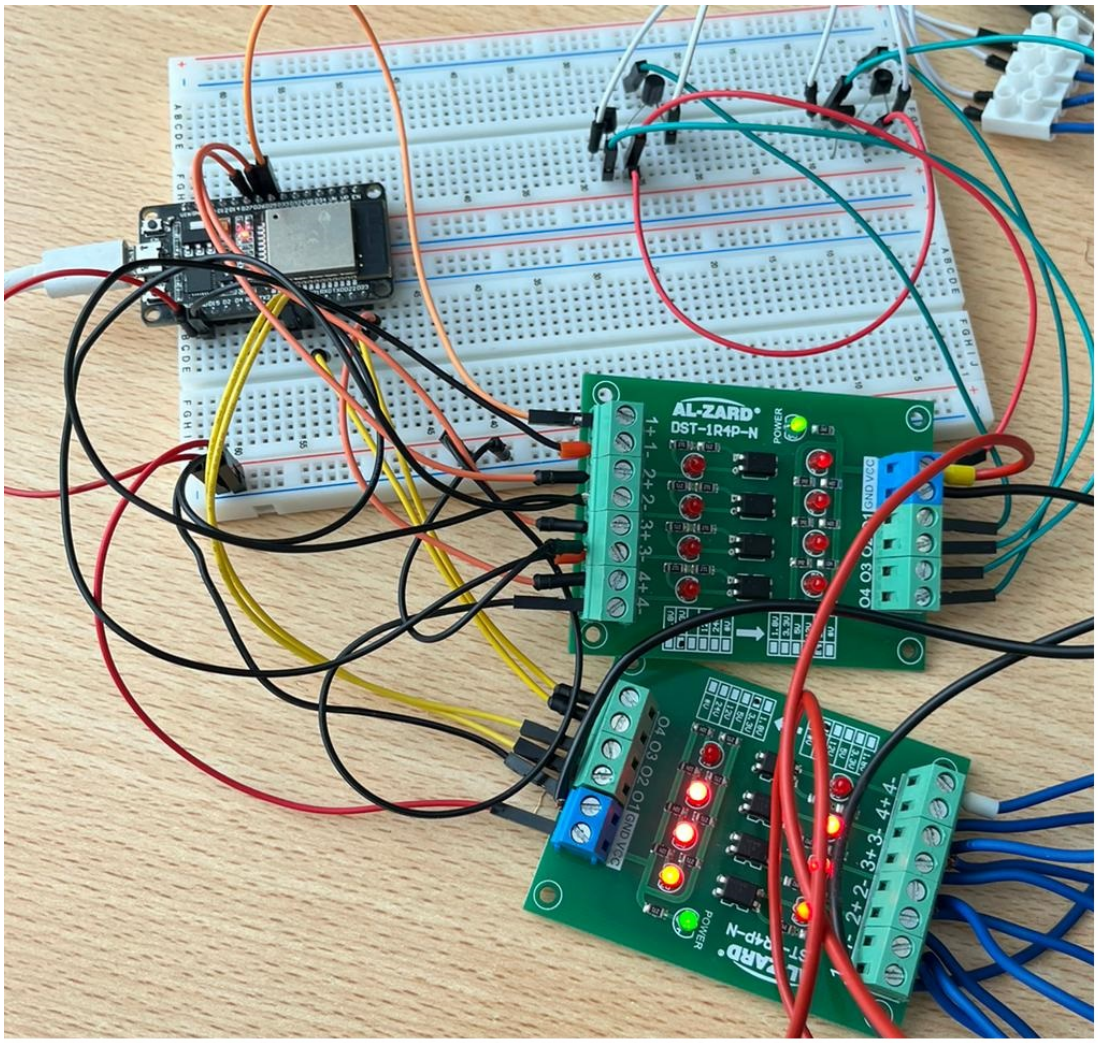

Figure 5.11: Controller's Setup for Proof of Concept

The first stage of code development was to make sure that the controller could read every sensor of the Kit. Each sensor was connected to the Optocoupler 24-3.3

Module, and activated/deactivated to make sure the Processing Unit Module and the Code Interface were reading the values correctly. In the second stage, the conveyor belt actuators, Q1 and Q2, were connected to the exit of their respective booster montage and activated alternately, to test the conveyor belt in the two possible movements. Following the success of this part, the connection of the punching machine actuators, Q3 and Q4, was realized, and as was done previously, activated alternately.

After confirming the developed system was working as intended, the third stage consisted of combining the activation of the actuators according to the sensor's information. First, a box was placed on the conveyor belt and the Processing Unit Module would activate one of the conveyor's actuators. When it reaches the end sensor of the conveyor belt, the Processing Unit Module activates the other actuator, making the conveyor belt move in the opposite direction. The same methodology was applied to the punching sensor. When the punching machine reaches a position sensor, the Processing Unit Module activates the other actuator, moving the punching machine in the other direction.

In the final stage, all actuators and sensors are combined to create a production process. The kit receives a box on the end of the conveyor belt and moves it to the punching machine. When arrives there, the conveyor stops, and the punching machine comes down and marks the box. After the punching machine returns to its original position, the conveyor delivers the box to its original position. Figure 5.12 shows the stages taken in integrating the kit with the controller.

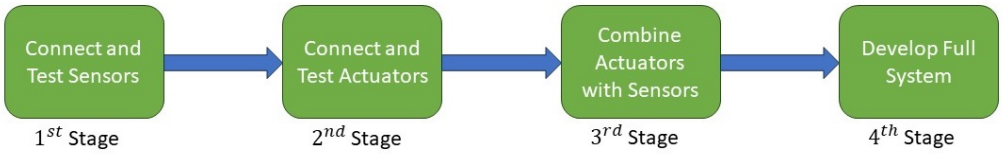

Figure 5.12: Integration Stages of the Industrial kit with the controller

As predicted, the approach taken for this test had great results, as it validated the Code Interface developed, proved the Industrial Kit could be integrated with the chosen Processing Unit Module and the test code for future tests also works.

5.3.2 Power Supply Modules

The test of the Power Modules pretends to verify if the Power Modules' theoretical calculations were correctly done. Starting with the Power Module, to verify if it could power the Processing Unit Module. After assembling it and connecting it

to the 24V Power Supply, the Processing Unit started as intended, and by verifying the tension value on the VIN pin, with the help of a multimeter, could be seen the 7V tension, therefore confirming this module's theoretical calculations

For the Reference Module, a similar test was conducted. After assembling it, the module was connected to the Digital Optocoupler Modules, to substitute the reference 3.3V voltage source used in the previous tests. The test was a success, as the Digital Optocoupler Modules worked without any problems, and using a multimeter to verify that the tension on the exit of the Reference Module was indeed 3.3V, the test was considered a success.

Even though this Module testing was a success, it became clear that after just a short time of being powered up, the integrated circuit responsible for the power conventions was emitting a lot of heat radiation. The solution was to place a heat sink over the integrated circuit to reduce this effect, with this solution having good results.

5.3.3 Digital Interface Modules

In the first phase of testing the Digital Interface Modules, only a Digital Input Module and a Digital Output Module were set, in order to confirm the circuit theoretical calculations and simulations. The Digital Output Module was connected to one of the conveyor belt actuators, and the Digital Input Module to the sensor on the end of the conveyor belt.

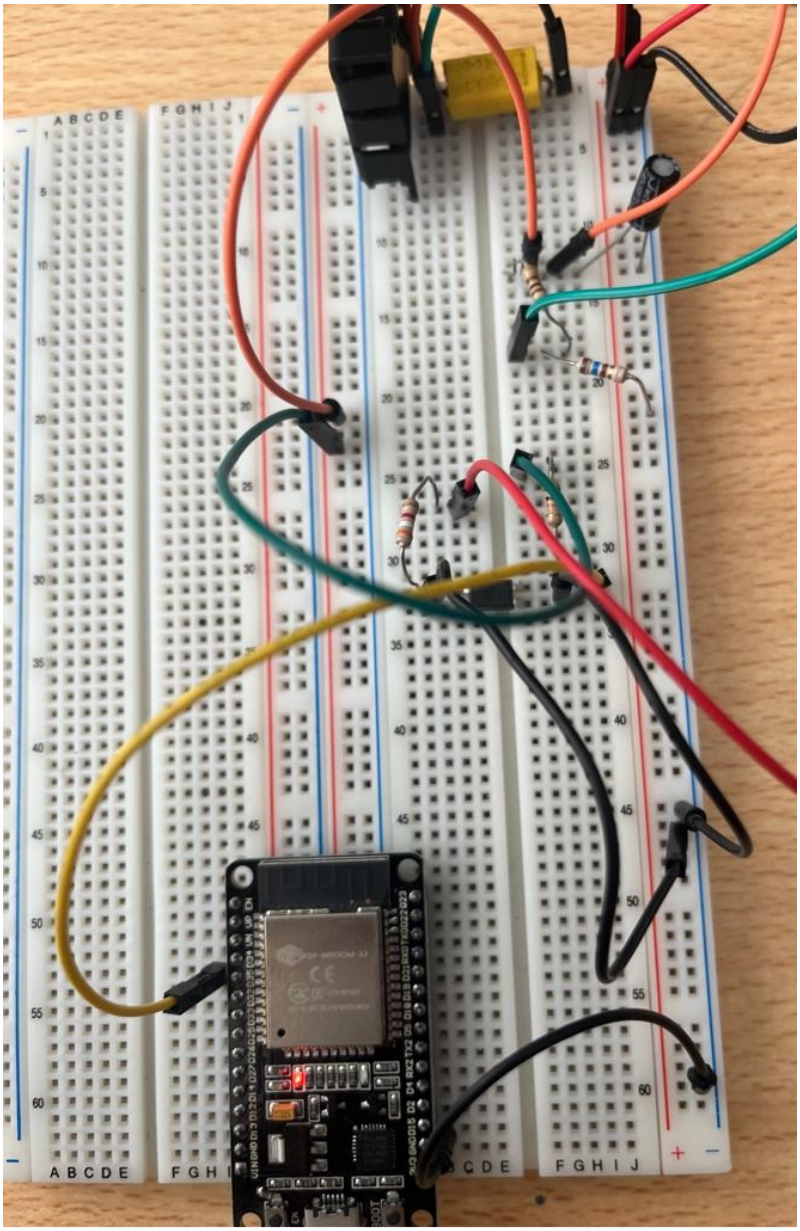

Figure 5.13: Digital Input and Output Modules.

After setting up the modules, it became clear that they were not working properly. After some troubleshooting, where all components were analysed, it became clear that the resistor R3 on the Digital Input Module and R2 of the Digital

Output Module were under-dimensioned, although the resistor's value was chosen based on the Optocoupler datasheet. Using trial and error, by substituting the previously identified resistors with a $33k\Omega$ Resistor we got the module switching values as desired. Figures 5.14 and 5.15 show the adjusted values for the Digital Output and Input Modules, respectively. Figures 5.16 and 5.17 show the simulated tension values of the output of these Modules.

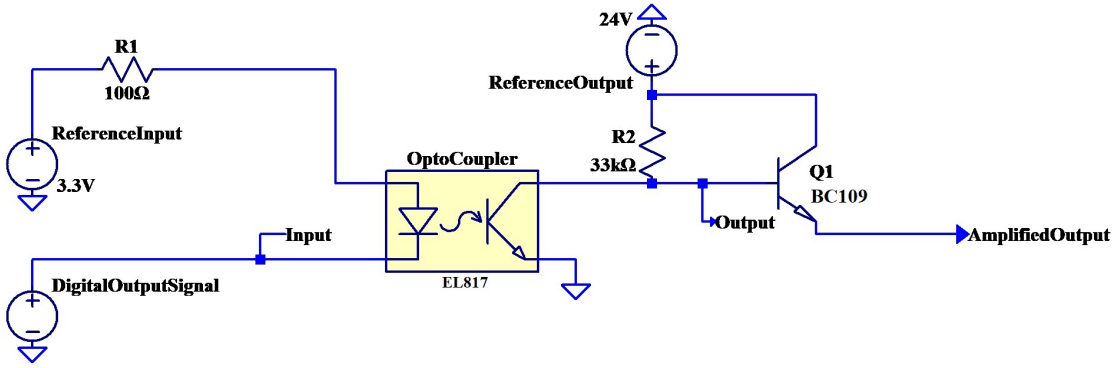

Figure 5.14: Adjusted 3.3V - 24V Digital Output Module, with the current booster.

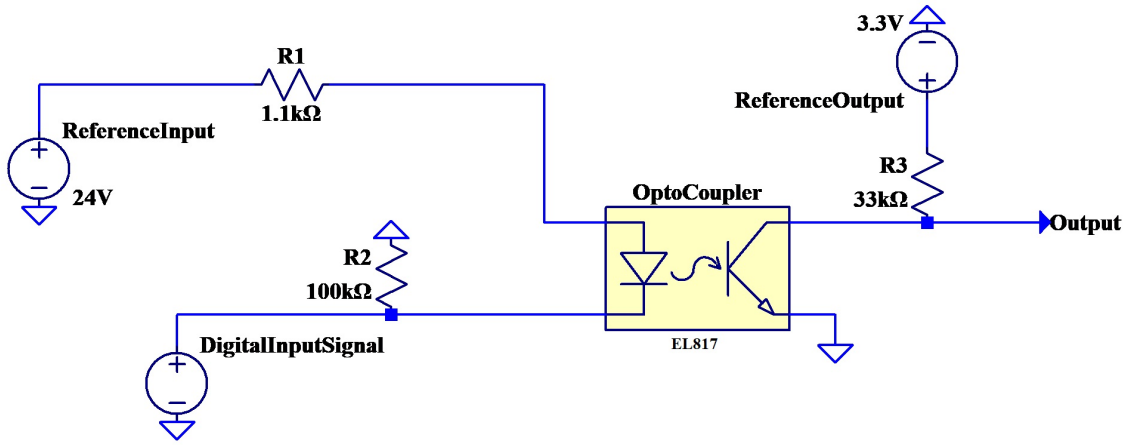

Figure 5.15: Adjusted 24V - 3.3V Digital Input Module,

After adjusting the Digital Modules, the necessary Digital Modules for testing using the Industrial Kit, combined with the Processing Unit Module and the Power Supply Module, while using the test code developed for the first test. The tests were a success, creating a full-functioning discrete controller implemented on the breadboard.

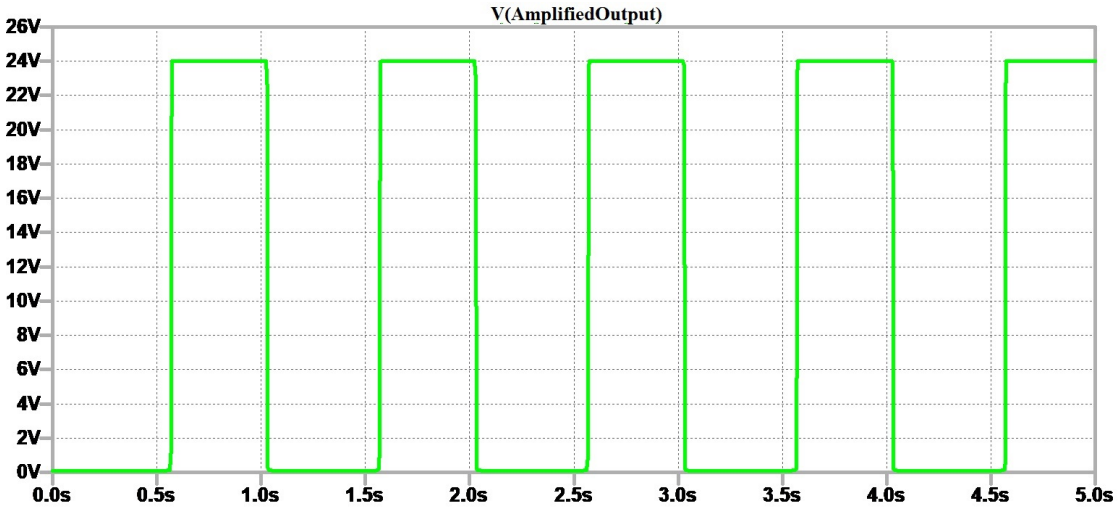

Figure 5.16: Tension variation on the AmpOutput node of figure 5.14

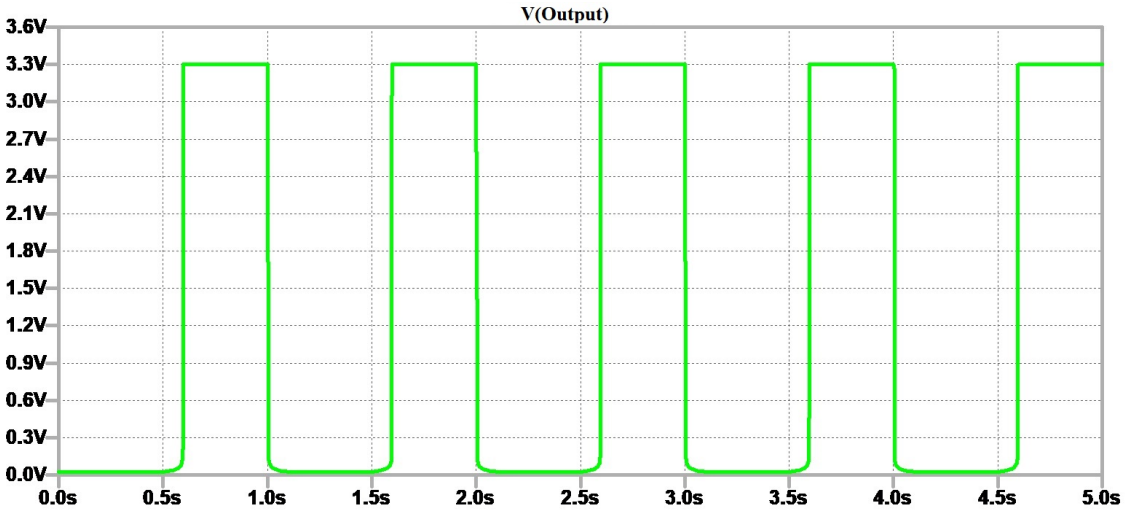

Figure 5.17: Adjusted tension variation on the Output Node of figure 5.15.

5.4 Protoboard Tests

As previously explained, the main goal of the protoboard montage was to assemble the first prototype of the low-cost controller, checking once again if all modules would work as intended. The same approach that was used for the breadboard test was used in this case. First, the reference power module was set up, giving the 3.3 reference value for the digital modules. After setting the reference module, a digital input module was soldered, checking once again for possible problems with the resistors' values. No problems emerged, leading to the first functioning module on the protoboard. After that, a digital output

module was soldered, and once again, no problems emerged from the test, giving the validation for the digital output module on the protoboard. The remaining necessary modules for controlling the industrial kit were soldered. Figure 5.18 shows the final test set-up.

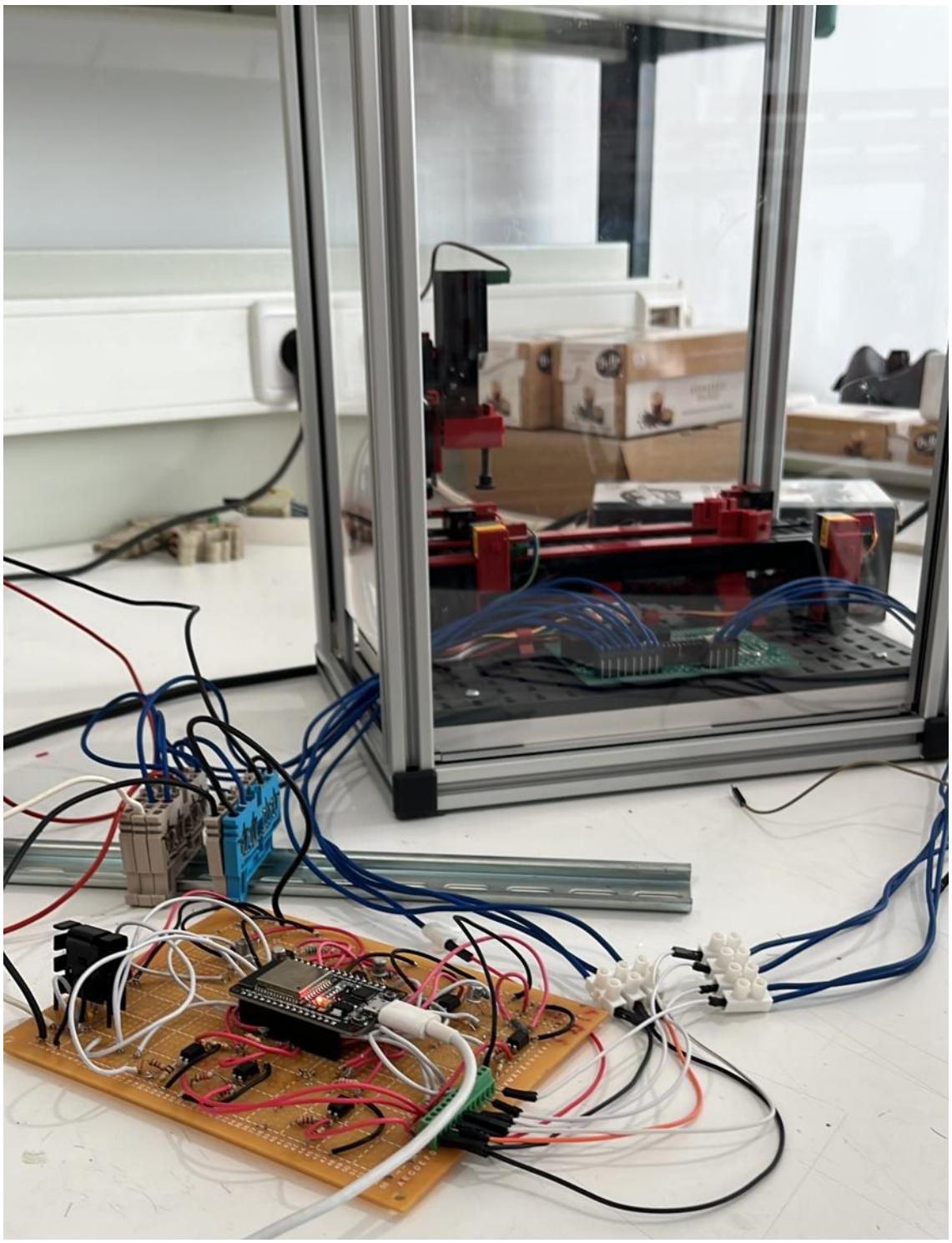

Figure 5.18: Protoboard connected to the Industrial Kit.

The final kit control test ran as predicted, functioning smoothly. The protoboard functioning also provided a controller for the test of other modules, while the PCB was developed.

5.4.1 Wi-Fi Communication Testing

This was the first test run after the protoboard validation. In this test, the Processing Unit Module is connected to a Wi-Fi network and creates a Web Server, that can be accessed by connecting to the Module's IP address. The goal was to test if the controller could receive the activation signal through the Web server, and start the Kit process movement.

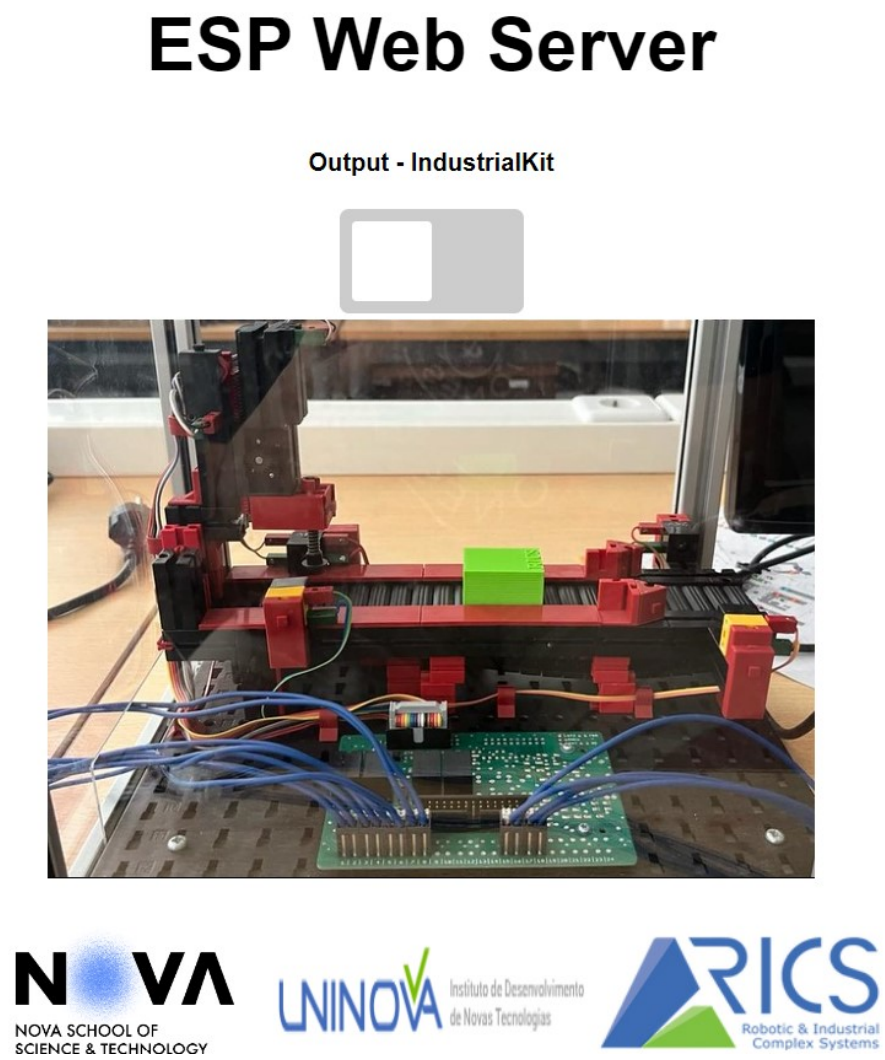

Figure 5.19: Asynchronous Web Server running on Processing Unit Module, for interaction with peripherals.

The Web Server set on the Processing Unit Module was asynchronous. It allows

for a stable connection compared to a synchronous Web Server. A switch button, which when toggled, sends an HTTP request to the Processing Unit Module, which activates the Kit movement. While the switch button is toggled, it allows the continuous activation of the Kit process. When deactivated, the Industrial Kit would stop the movement at the beginning of the next iteration. Figure 5.19 shows the Web Server interface.

In this test, the communication with the web server worked correctly and all tests ran as intended. A slight adjustment was made to the machine's movement, as the punching was not always returning to the correct position.

5.5 PCB Tests

The final tests were run on the developed PCB. The tests run were all previously shown, testing the multiple capabilities of the controller. Figure 5.20 shows the test montage of the circuit.

All tests ran smoothly, providing the expected results, and validating the PCB developed.

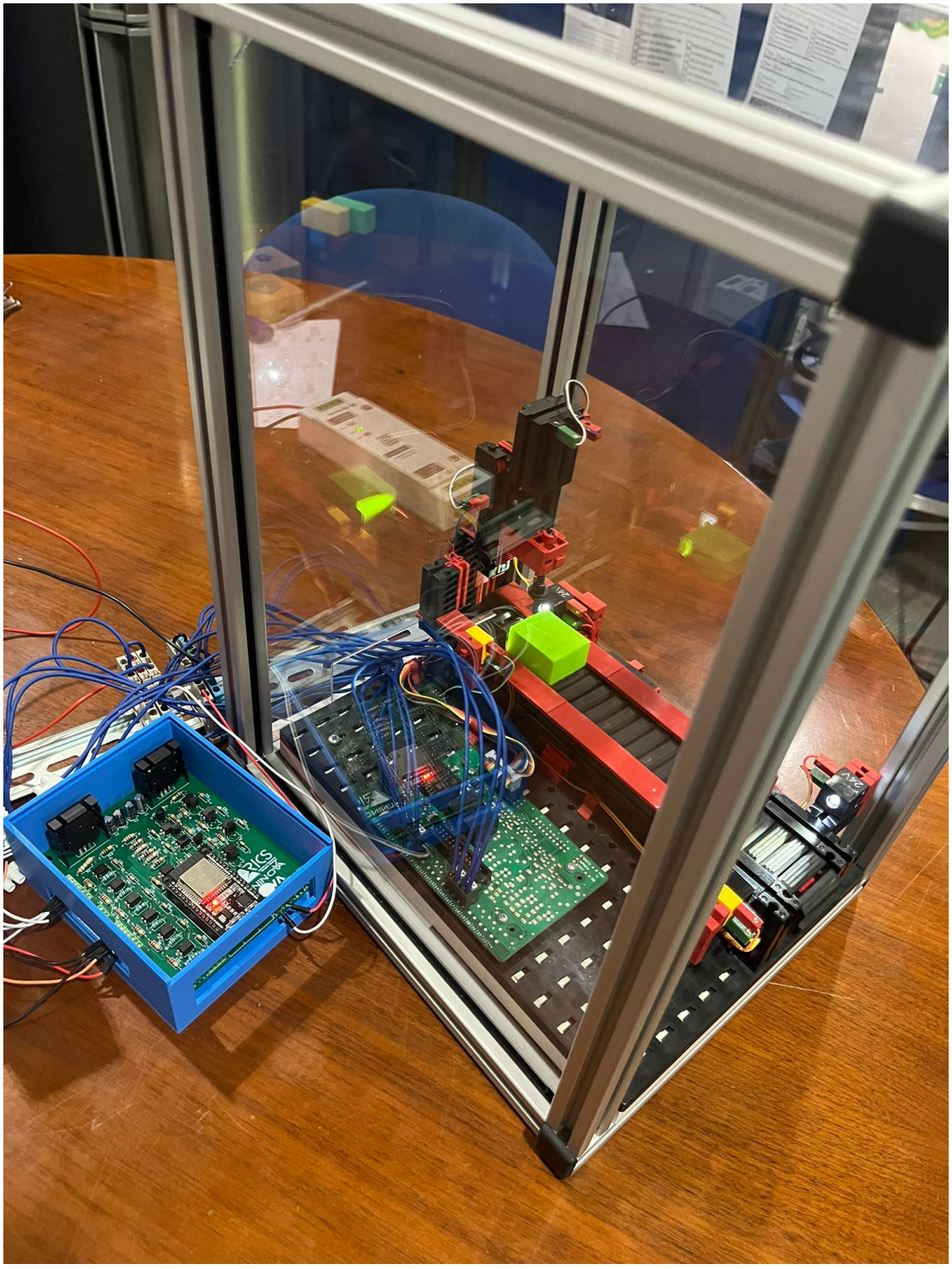

Figure 5.20: PCB Montage for tests and validation.

6

Conclusions and Future Work

6.1 Conclusions

For this thesis, the main goal was to develop a modular, flexible solution for a new low-cost educational industrial controller using open-source standards for universities and students.

With the developed controller, all objectives were obtained: 1)Using the ESP-32 as the core element was the right decision, as its multiple interfaces allow for various applications and tests. For a more complex Industrial Kit, it would be necessary to verify if the module is suited for the desired application;2)The implemented modules are highly flexible, allowing for various adaptations based on the user's needs, the testing environment and the Processing Unit Module tension values;3)The test results show that this controller can adjust to new Industry 4.0 paradigms, providing a base for future developments.

The opportunity to use an educational industrial kit from UNINOVA contributed significantly to the results, as the modules could be easily tested after the implementation phase, and adjusted once again.

After assembling the protoboard prototype and validating its correct functioning, it was used for multiple demonstrations by the Robotics and Integrated Manufacturing section, to showcase some of the work being developed by master's students to outside entities, as well as what young students can expect to work with when they frequent the master's course.

After all tests and consequent results, it was proven that a low-cost controller development was successful.

6.2 Future Work

As future work, it is proposed the development of other relevant modules, like an Analog Output Module that is able to step up the tension value to the 24V used in the Industry.

Another future work possibility would be the integration of the controller on an educational supervisory system, through the use of some of the industrial communication protocols mentioned previously.

It also would be relevant to test the controller's performance with more powerful processing unit modules, as it would possibly increase the number of modules that could be connected to the controller.

Bibliography

[1] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/main/template.pdf.

[2] X. Gu and Y. Koren. "Mass-Individualisation–the twenty first century manufacturing paradigm". In: *International Journal of Production Research* (2022). ISSN: 1366588X. DOI: 10.1080/00207543.2021.2013565.

[3] R. Drath and A. Horch. *Industrie 4.0: Hit or hype? [Industry Forum]*. 2014. DOI: 10.1109/MIE.2014.2312079.

[4] E. Uhlmann, E. Hohwieler, and C. Geisert. *INTELLIGENT PRODUCTION SYSTEMS IN THE ERA OF INDUSTRIE 4.0-CHANGING MINDSETS AND BUSINESS MODELS*. 2017.

[5] J. Eastburn. *Industrial Controllers: Past, present, future*. 2020-05. URL: https://www.controleng.com/articles/industrial-controllers-past-present-future/.

[6] H. Kagermann, W. Walhster, and J. Helbig. *Recommendations for implementing the strategic initiative Industrie 4.0: Final report of the Industrie 4.0 Working Group*. 2013.

[7] H. Lasi et al. "Industry 4.0". In: *Business and Information Systems Engineering* 6 (4 2014-08), pp. 239–242. ISSN: 18670202. DOI: 10.1007/s12599-014-0334-4.

[8] L. M. Camarinha-Matos, A. D. Rocha, and P. Graça. "Collaborative approaches in sustainable and resilient manufacturing". In: *Journal of Intelligent Manufacturing* (2022). ISSN: 15728145. DOI: 10.1007/s10845-022-02060-6.

[9] L. M. Camarinha-Matos, R. Fornasiero, and H. Afsarmanesh. "Collaborative networks as a core enabler of industry 4.0". In: vol. 506. Springer New York LLC, 2017, pp. 3–17. ISBN: 9783319651507. DOI: 10.1007/978-3-319-65151-4_1.

[10] M. Rüßmann et al. *Industry 4.0 The Future of Productivity and Growth in Manufacturing Industries*. 2015.

[11] E. A. Lee. "CPS foundations". In: 2010, pp. 737–742. ISBN: 9781450300025. DOI: 10.1145/1837274.1837462.

[12] D. G. Pivoto et al. *Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review*. 2021-01, pp. 176–192. DOI: 10.1016/j.jmsy.2020.11.017.

[13] A. Humayed et al. "Cyber-Physical Systems Security - A Survey". In: *IEEE Internet of Things Journal* 4 (6 2017-12), pp. 1802–1831. ISSN: 23274662. DOI: 10.1109/JIOT.2017.2703172.

[14] D. Serpanos. "The Cyber-Physical Systems Revolution". In: *Computer* 51 (3 2018-03), pp. 70–73. ISSN: 00189162. DOI: 10.1109/MC.2018.1731058.

[15] acatech-National Academy of Science and Engineering. *Cyber-Physical Systems Driving force for innovation in mobility, health, energy and production acatech (Ed.)* 2011. URL: www.acatech.de.

[16] M. Hermann, T. Pentek, and B. Otto. "Design principles for industrie 4.0 scenarios". In: vol. 2016-March. IEEE Computer Society, 2016-03, pp. 3928–3937. ISBN: 9780769556703. DOI: 10.1109/HICSS.2016.488.

[17] G. Dileep. "A survey on smart grid technologies and applications". In: *Renewable Energy* 146 (2020-02), pp. 2589–2625. ISSN: 18790682. DOI: 10.1016/j.renene.2019.08.092.

[18] M. Wehde. "Healthcare 4.0". In: *IEEE Engineering Management Review* 47 (3 2019-07), pp. 24–28. ISSN: 19374178. DOI: 10.1109/EMR.2019.2930702.

[19] A. Gupta and A. Singh. "A Comprehensive Survey on Cyber-Physical Systems Towards Healthcare 4.0". In: *SN Computer Science* 4 (2 2023-03). ISSN: 26618907. DOI: 10.1007/s42979-023-01669-5.

[20] A. Pundir et al. "Cyber-Physical Systems Enabled Transport Networks in Smart Cities: Challenges and Enabling Technologies of the New Mobility Era". In: *IEEE Access* 10 (2022), pp. 16350–16364. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3147323.

[21] D. P. Moller and H. Vakilzadian. "Cyber-physical systems in smart transportation". In: IEEE, 2016-05, pp. 0776–0781. ISBN: 978-1-4673-9985-2. DOI: 10.1109/EIT.2016.7535338.

[22] L. Monostori et al. "Cyber-physical systems in manufacturing". In: *CIRP Annals* 65 (2 2016), pp. 621–641. ISSN: 17260604. DOI: 10.1016/j.cirp.2016.06.005.

[23] N. Galaske and R. Anderl. "Disruption Management for Resilient Processes in Cyber-physical Production Systems". In: vol. 50. Elsevier B.V., 2016, pp. 442–447. DOI: 10.1016/j.procir.2016.04.144.

[24] R. Kumar et al. "Live Life Cycle Assessment Implementation using Cyber Physical Production System Framework for 3D Printed Products". In: *Procedia CIRP* 105 (2022), pp. 284–289. ISSN: 22128271. DOI: 10.1016/j.procir.2022.02.047.

[25] B. J. Ralph et al. "Transformation of a rolling mill aggregate to a cyber physical production system: from sensor retrofitting to machine learning". In: *Journal of Intelligent Manufacturing* 33 (2 2022-02), pp. 493–518. ISSN: 0956-5515. DOI: 10.1007/s10845-021-01856-2.

[26] M. Suvarna et al. "Cyber–Physical Production Systems for Data-Driven, Decentralized, and Secure Manufacturing—A Perspective". In: *Engineering* 7 (9 2021-09), pp. 1212–1223. ISSN: 20958099. DOI: 10.1016/j.eng.2021.04.021.

[27] K. P. Sycara. "Multiagent Systems". In: (1998).

[28] R. S. Peres et al. "IDARTS – Towards intelligent data analysis and real-time supervision for industry 4.0". In: *Computers in Industry* 101 (2018), pp. 138–146. ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2018.07.004. URL: https://www.sciencedirect.com/science/article/pii/S0166361517306759.

[29] L. Ribeiro et al. "Self-organization in automation - the IDEAS pre-demonstrator". In: IEEE, 2011-11, pp. 2752–2757. ISBN: 978-1-61284-972-0. DOI: 10.1109/IECON.2011.6119747.

[30] J. Queiroz et al. "Agent-Based Distributed Data Analysis in Industrial Cyber-Physical Systems". In: *IEEE Journal of Emerging and Selected Topics in Industrial Electronics* 3 (1 2021-07), pp. 5–12. ISSN: 2687-9735. DOI: 10.1109/jestie.2021.3100775.

[31] G. Koschnick, M. Hankel, and B. Rexroth. *RAMI 4.0-Structure The Reference Architectural Model Industrie 4.0 (RAMI 4.0) Contact: Reference Architectural Model Industrie 4.0*. 2015. URL: www.zvei.org/.

[32] P. Adolphs and U. Epple. *Status Report Reference Architecture Model Industrie 4.0 (RAMI4.0)*. 2015-07. URL: www.vdi.de.

[33] D. Alemao et al. "How to Design Scheduling Solutions for Smart Manufacturing Environments Using RAMI 4.0?" In: *IEEE Access* 10 (2022), pp. 71284–71298. ISSN: 21693536. DOI: 10.1109/ACCESS.2022.3187974.

[34] Y. Wang, T. Towara, and R. Anderl. "Topological Approach for Mapping Technologies in Reference Architectural Model Industrie 4.0 (RAMI 4.0)". In: 2017, pp. 982–990. ISBN: 9789881404848.

[35] J. Lee, B. Bagheri, and H. A. Kao. "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems". In: *Manufacturing Letters* 3 (2015-01), pp. 18–23. ISSN: 22138463. DOI: 10.1016/j.mfglet.2014.12.001.

[36] K. Erickson. "Programmable logic controllers". In: *IEEE Potentials* 15.1 (1996), pp. 14–17. DOI: 10.1109/45.481370.

[37] E. Monmasson et al. "FPGAs in industrial control applications". In: *IEEE Transactions on Industrial Informatics* 7 (2 2011), pp. 224–243. ISSN: 15513203. DOI: 10.1109/TII.2011.2123908.

[38] T. Walter and I. Bell. *Where others fear to tread THE USE OF PROGRAMMABLE AUTOMATION CONTROLLERS (PACS) IN PROCESS CONTROL AND AUTOMATION IS GROWING RAPIDLY. IN MANY APPLICATIONS, HOWEVER, PACS ARE NOT REPLACING PLCS; RATHER, ENGINEERS USE THEM TO EXPAND PROCESS MONITORING AND CONTROL INTO AREAS WHERE PLCS FEAR TO TREAD*. 2006-04. URL: www.theiet.org/control.

[39] J. L. Álvarez, J. D. Mozo, and E. Durán. "Analysis of single board architectures integrating sensors technologies". In: *MPDI-SENSORS* 21 (18 2021-09). ISSN: 14248220. DOI: 10.3390/s21186303.

[40] DIGI. "Z45 Industrial Controllers". In: *https://www.digi.com/products/networking/infrastructure-management/industrial-automation/z45-industrial-controllers* (2023).

[41] I. Shields. "ESP32 PLC 58". In: *https://www.industrialshields.com/shop/034001000600-esp32-plc-58-2909* (2023).

[42] I. Shields. "RASPBERRY PLC 38R". In: *https://www.industrialshields.com/shop/012003000300-raspberry-plc-38r-4gb-2377* (2023).

[43] Flexbridge. "IceBlock". In: *https://flexbridge.se/iceblock/* (2023).

[44] R. Pi. "RevPi Compact". In: *https://revolutionpi.com/revpi-compact* (2023).

[45] A. Glória, F. Cercas, and N. Souto. *Comparison of Communication Protocols for Low Cost Internet of Things Devices*. 2017-11.

[46] F. Leens. *An Introduction to I2C and SPI Protocols*. 2009.

[47] J. Chen and S. Huang. "Analysis and Comparison of UART, SPI and I2C". In: Institute of Electrical and Electronics Engineers Inc., 2023, pp. 272–276. ISBN: 9781665462532. DOI: 10.1109/EEBDA56825.2023.10090677.

[48] A. S. Tanenbaum and D. J. Wetherall. *Computer Networks*. 2011.

[49] S. Banerji and R. S. Chowdhury. "On IEEE 802.11: Wireless Lan Technology". In: *International Journal of Mobile Network Communications & Telematics* 3 (4 2013-08), pp. 45–64. DOI: 10.5121/ijmnct.2013.3405.

[50] A. F. Rochim et al. "Performance comparison of wireless protocol IEEE 802.11ax vs 802.11ac". In: Institute of Electrical and Electronics Engineers Inc., 2020-02. ISBN: 9781728130835. DOI: 10.1109/ICoSTA48221.2020.1570609404.

[51] M. A. A. Khan et al. *COMPARISON AMONG SHORT RANGE WIRELESS NETWORKS: COMPARISON AMONG SHORT RANGE WIRELESS NETWORKS: BLUETOOTH, ZIGBEE, & WI-FI*. 2016. URL: http://hdl.handle.net/20.500.11948/1466Downloadedfromhttp://dspace.library.daffodilvarsity.edu.bd,.

[52] S. Zeadally, F. Siddiqui, and Z. Baig. "25 years of bluetooth technology". In: *Future Internet* 11 (9 2019-09). ISSN: 19995903. DOI: 10.3390/fi11090194.

[53] M. B. Yaakop et al. *Bluetooth 5.0 Throughput Comparison for Internet of Thing Usability A Survey*. IEEE, 2017.

[54] M. Vukovic et al. "Digital Twins in Industrial IoT: A survey of the state of the art and of relevant standards". In: Institute of Electrical and Electronics Engineers Inc., 2021-06. ISBN: 9781728194417. DOI: 10.1109/ICCWorkshops50388.2021.9473889.

[55] S. K. Panda et al. "Real-time Industrial Communication by using OPC UA Field Level Communication". In: *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. Vol. 1. 2020, pp. 1143–1146. DOI: 10.1109/ETFA46521.2020.9211998.

[56] M. Schleipen et al. "OPC UA & Industrie 4.0 - Enabling Technology with High Diversity and Variability". In: vol. 57. Elsevier B.V., 2016, pp. 315–320. DOI: 10.1016/j.procir.2016.11.055.

[57] O. Foundation. *Initiative: Field Level Communications (FLC)*. Tech. rep. 2021.

[58] E. Systems. *ESP32 Series Datasheet*. 2023-01. URL: https://www.espressif.com/en/support/download/documents..

[59] ARDUINO. *ARDUINO DUE*. 2023.

[60] ARDUINO. *ARDUINO Mega 2560 Rev3*. 2023.

[61] R. PI. *Raspberry Pi 4 Model B Datasheet*. Raspberry Pi, 2023. URL: https://www.raspberrypi.org.

[62] BEAGLEBOARD. *BeagleBone Black*. 2023.

[63] ARDUINO. *ARDUINO - WHAT IS IT*. 2023.

[64] C. W. de Silva. *Sensors and Actuators: Engineering System Instrumentation*. 2º Edition. CRC PRESS, 2016.

[65] H. Janocha. *Actuators*. Ed. by H. Janocha. Springer Berlin Heidelberg, 2004. ISBN: 978-3-642-08266-5. DOI: 10.1007/978-3-662-05587-8. URL: http://link.springer.com/10.1007/978-3-662-05587-8.

[66] JOY-IT. *SBC-NodeMCU-ESP32*. 2017.

[67] A. L. Ramião, J. A. B. D. Oliveira, and A. D. B. D. S. P. Rocha. "Sistema Cíber-Físico de Produção Modular". Faculdade de Ciências e Tecnologias, 2017.

[68] AL-ZARD. *DST-1R4P-N-190551*. 2023.

[69] J. Ben et al. *Study and Modelling of Optocouplers Ageing*. 2008-09. URL: https://www.researchgate.net/publication/278810313.

[70] E. E. Co. *4 PIN DIP PHOTOTRANSISTOR PHOTOCOUPLER EL817 Series*. Everlight, 2010. URL: http://www.everlight.com.