



UNIVERSITÀ
DEGLI STUDI
DI PADOVA



DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

MASTER THESIS IN CONTROL SYSTEM ENGINEERING

Enhancing Traceability in Welding Lines through the Integration of AI-Powered Code Readers with Automation Systems

MASTER CANDIDATE

Alireza Seyedhassanidolatabad

Student ID 2044775

SUPERVISOR

Prof. Alberto Morato

University of Padova

CO-SUPERVISOR

Mr. Francesco Meneghini

LEAS SPA

ACADEMIC YEAR
2023/2024

*To my Parents
and Friends*

Abstract

This thesis focuses on automation systems, particularly Programmable Logic Controllers (PLCs) and their integration with AI-Powered Code Readers in Welding Machine Automation. It explores the vital components of modern automation systems and their significance in various industries like automotive, aerospace, and manufacturing.

The thesis highlights the extensive use of PLCs in control systems and manufacturing automation through case studies. It showcases successful examples of code reader integration in welding machines, ensuring accurate identification, efficient tracking, and enhanced traceability of components for improved production efficiency and quality.

The study also identifies emerging trends in code reader technology, such as advanced image processing, IoT integration, mobile scanning, and AI-driven code recognition. Additionally, it examines potential advancements in welding machine automation, including collaborative robots, adaptive control algorithms, data analytics, and predictive maintenance for increased productivity and equipment reliability. Emphasizing Human-Machine Collaboration, the thesis stresses the importance of intuitive interfaces, AR support, and effective communication between operators and automated systems, promoting seamless collaboration and knowledge transfer.

In conclusion, this thesis underscores the transformative impact of automation systems and code reader technology in welding machine automation. By embracing the latest trends, industries can achieve heightened efficiency, enhanced quality control, and improved traceability, revolutionizing modern manufacturing.

Sommario

Questa tesi si concentra sui sistemi di automazione, in particolare sui Controllori Logici Programmabili (PLC) e la loro integrazione con Lettori di Codici a Barre nell'Automazione delle Macchine da Saldatura. Esplora i componenti fondamentali dei moderni sistemi di automazione e la loro importanza in diverse industrie come l'automotive, l'aerospaziale e la manifatturiera.

La tesi mette in evidenza l'ampio utilizzo dei PLC nei sistemi di controllo e nell'automazione manifatturiera attraverso studi di caso. Mostra esempi di successo dell'integrazione dei lettori di codici a barre nelle macchine da saldatura, garantendo l'identificazione precisa, il monitoraggio efficiente e la tracciabilità migliorata dei componenti per un miglioramento dell'efficienza e della qualità della produzione.

Lo studio identifica anche le tendenze emergenti nella tecnologia dei lettori di codici a barre, come l'elaborazione avanzata delle immagini, l'integrazione con Internet of Things (IoT), la scansione mobile e il riconoscimento intelligente dei codici a barre tramite intelligenza artificiale (AI). Inoltre, esamina possibili sviluppi nell'automazione delle macchine da saldatura, inclusi robot collaborativi, algoritmi di controllo adattativo, analisi dei dati e manutenzione predittiva per aumentare la produttività e la affidabilità delle apparecchiature. Mettendo in risalto la Collaborazione Uomo-Macchina, la tesi sottolinea l'importanza di interfacce intuitive, il supporto della Realtà Aumentata (AR) e una comunicazione efficace tra gli operatori e i sistemi automatizzati, promuovendo una collaborazione e un trasferimento di conoscenze senza soluzione di continuità. In conclusione, questa tesi evidenzia l'impatto trasformativo dei sistemi di automazione e della tecnologia dei lettori di codici a barre nell'automazione delle macchine da saldatura. Abbracciando le ultime tendenze, le industrie possono raggiungere una maggiore efficienza, un miglior controllo qualità e una tracciabilità migliorata, rivoluzionando la moderna manifattura.

Contents

List of Figures	xi
List of Tables	xiii
List of Acronyms	xix
1 Introduction	1
1.1 Contribution	1
1.2 History of Automation	2
1.3 What is Automation?	3
1.3.1 Simplest examples for Automation	3
1.3.2 Levels of Automation	4
1.3.3 Main types of Automation	5
2 Related Works	9
2.1 Literature Review	9
2.1.1 Modern Control System	9
2.1.2 PLC Applications	10
2.1.3 PLC Networks	11
2.1.4 Fourth Industrial Revolution	11
2.2 Overview of Welding Machine Automation	13
2.3 Principles and Working Mechanisms of Code Readers	15
2.4 Case Studies and Industry Applications	17
3 Overview of Some of the Key Automation Components	19
3.1 Introduction to Programmable Logic Controllers (PLCs)	19
3.1.1 Evolution of Control Systems: From DCS to PLCs	20
3.1.2 The Major PLCs Manufacturers	21
3.1.3 Programming Languages for PLCs and their Comparison	23

CONTENTS

3.1.4	Introduction to TIA Portal (Siemens Totally Integrated Automation)	28
3.1.5	PLC Hardware Components	31
3.1.6	Basic of PLCs Communications	32
3.2	Remote Terminal Unit (RTU)	37
3.3	Safety PLC	38
3.4	Human-Machine Interface (HMI)	39
3.5	Supervisory Control and Data Acquisition (SCADA)	40
3.5.1	Differences Between SCADA and HMI	41
3.5.2	SCADA Systems and DCS in Industrial Automation	41
3.5.3	Electrical Control Panel	42
4	Development of the Prototype	43
4.1	Problem	43
4.2	Solution	44
4.3	Components Used in the Prototype	46
4.3.1	Keyence SR-X300 AI-Powered Code Reader	46
4.3.2	PLC S7-1517-3 PN/DP	47
4.3.3	SIMATIC HMI TP900 Comfort	48
4.3.4	SIMATIC ET200SP	49
4.3.5	PATLITE WE-402UB-LAN Signal Tower	50
4.3.6	PATLITE LR7-02KTN Signal Tower	51
4.3.7	PATLITE LB6-20ILWCBW Signal Tower	52
4.3.8	Phoenix Contact FL Switch SFN 8TX	53
4.3.9	Cabur Switching Power Supply CS1024/120-230	54
4.4	Installation	55
4.4.1	Hardware Installation	55
4.4.2	Software Configuration	57
4.4.3	Programming	64
4.4.4	Signal Towers Integration	75
4.4.5	HMI Development	82
4.4.6	Production Line	88
5	Conclusion and Future works	93
5.1	Results and Conclusion	93
5.2	Future Works	95

CONTENTS

5.2.1	Emerging Trends in AI-Powered Code Reader Technology	95
5.2.2	Potential Advancements in Welding Machine Automation	96
	References	97
	Acknowledgments	101

List of Figures

1.1	Automation Pyramid	4
1.2	Advantages and Disadvantages of the Automation types	7
3.1	Ladder Diagram (LD)	23
3.2	Function Block Diagram (FBD)	24
3.3	Structured Control Language (SCL)	25
3.4	Structured Text (ST)	26
3.5	Sequential Function Chart (SFC)	27
3.6	Instruction List (IL)	28
3.7	Program Blocks	30
4.1	SR-X300 Reader	46
4.2	S7-1517	47
4.3	HMI Panel	48
4.4	ET200SP	49
4.5	WE-402UB-LAN	50
4.6	LR7-02KTN	51
4.7	LB6-20ILWCBW	52
4.8	FL Switch SFN 8TX	53
4.9	Power Supply CS1024/120-230	54
4.10	Test Desk	55
4.11	Block Scheme	56
4.12	AutoIDNetworkNavigator	57
4.13	Patlite WE-402UB-LAN	58
4.14	Topology View	60
4.15	Network View	61
4.16	Connection Mechanism	62
4.17	SR-X300 I/O	62

LIST OF FIGURES

4.18	ET200SP	63
4.19	UDTs	64
4.20	Tags	65
4.21	Tags and UDTs Connection	66
4.22	Cycle Manager FC	67
4.23	IDLE Step	68
4.24	Waiting for Request Step	69
4.25	Reading Step	70
4.26	Reading Complete Step	71
4.27	Result Data Conversion	71
4.28	Result Extraction	72
4.29	Reading Error	73
4.30	Cycle Steps on HMI	74
4.31	Modbus Block	75
4.32	Modbus Configuration	76
4.33	Modbus DataBlock	76
4.34	I/O link Master	77
4.35	I/O link Setup	78
4.36	Signal Tower Setup	78
4.37	Data transmission in Simple Mode	79
4.38	Data transmission in Level Mode	79
4.39	Process data transmission in Animation Mode	79
4.40	Data Transmission DB	80
4.41	Color Variety	80
4.42	Code Samples for testing	81
4.43	HMI Tags	82
4.44	HMI- Waiting for Request	83
4.45	HMI- Reading	83
4.46	HMI- Read Complete	84
4.47	HMI- Read Error	84
4.48	AutoTune- Waiting for Request	85
4.49	AutoTune- Tuning	86
4.50	AutoTune- Tuning Complete	86
4.51	AutoTune- Tuning Error	87
4.52	Wrong Bank-Number	87
4.53	Production Line	88

LIST OF FIGURES

4.54 Transferring on Conveyor	88
4.55 Transferring with Manipulators and a Single Motor	89
4.56 Transportation with Pallets	89
4.57 Pre-loading	90
4.58 Welding Test	90
4.59 Two types of Expansion Tanks	91

List of Tables

4.1	AutoIDNetworkNavigator Parameters	57
5.1	Some Extracted Data from Two Different Types of Tank Codes . .	93

List of Acronyms

PLC: Programmable Logic Controller

IoT: Internet of Things

RPI: Robotic Process Automation

AI: Artificial Intelligence

HMI: Human-Machine Interface

CNC: Computer Numerical Control

SCADA: Supervisory Control and Data Acquisition

MES: Manufacturing Execution System

SISO: Single-Input Single-Output

MIMO: Multiple-Input Multiple-Output

HVAC: Heating, Ventilation, and Air Conditioning

PID: Proportional-Integral-Derivative Controller

RTU: Remote Terminal Unit

I/O: Input/Output

LAN: Local Area Network

ASCII: American Standard Code for Information Interchange

CPS: Cyber-Physical Systems

MIG: Metal Inert Gas

LIST OF TABLES

TIG: Tungsten Inert Gas

LED: Light-Emitting Diode

2D: Two-Dimensional

CPU: Central Processing Unit

DCS: Distributed Control System

TIA: Totally Integrated Automation

IEC: International Electrotechnical Commission

IT: Information Technology

LD: Ladder Diagram

FBD: Function Block Diagram

SCL: Structured Control Language

SFC: Sequential Function Chart

IL: Instruction List

OB: Organization Block

FB: Function Block

DB: Data Block

IDE: Integrated Development Environment

DC: Direct Current

CRC: Cyclic Redundancy Check

IP: Internet Protocol

TCP: Transmission Control Protocol

CIP: Control Information Protocol

RPI: Requested Packet Interval

PI: PROFINET International

GSD: General Station Description

SIL: Safety Integrity Level

GUI: Graphical User Interfaces

KPI: Key Performance Indicators

UL: Underwriters Laboratories

NEMA: National Electrical Manufacturers Association

UPC: Universal Product Code

USB: Universal Serial Bus

MPI: Microsoft Message Passing Interface

URL: Uniform Resource Locator

UDT: User-Defined Types

IODD: IO Device Description



Introduction

1.1 CONTRIBUTION

This thesis presents a comprehensive exploration of automation systems, with a particular focus on Programmable Logic Controllers (PLCs) and their integration with AI-Powered code readers in Welding Machine Automation. The study aims to investigate the impact of automation technologies on various industries, including manufacturing.

The primary objective of this thesis is to investigate the functionalities and applications of PLCs in control systems design and the automation of manufacturing processes. By emphasizing the importance of Human-Machine Collaboration, intuitive interfaces, and augmented reality support, the study addresses the evolving nature of automation technologies and their integration with human operators. It explores the effective integration of AI-Powered code readers with welding machines, emphasizing their role in ensuring precise identification, streamlined tracking, and elevated traceability of components. This integration significantly enhances production efficiency and quality across assembly lines. Throughout the examination, the study also explores emerging trends in AI-Powered code readers technology alongside potential advancements in welding machine automation. The value of this thesis lies in its comprehensive analysis of the transformative impact of automation systems and AI-Powered code readers technology within welding machine automation processes.

1.2 HISTORY OF AUTOMATION

The history of automation can be traced back to significant milestones that shaped its development . In 1785, Oliver Evans introduced the world's first fully automated industrial processan automatic flour mill that operated continuously without human intervention. However, the term "automation" itself did not come into use until 1946 [28].

Throughout the centuries, automation progressed alongside technological advancements. The Industrial Revolution brought mechanization and steam power, enabling the replacement of manual labor with machines. As the 20th century dawned, digital technology emerged, leading to the creation of programmable logic controllers (PLCs) and numerical control systems. These innovations paved the way for precise and efficient task execution through automa-tion.

The late 20th century saw the rise of robotics and artificial intelligence (AI), further revolutionizing automation. Robots became capable of performing complex tasks with precision and adaptability, impacting industries such as manu-facturing, logistics, and healthcare.

In the present era, automation continues to evolve with advancements in software, data analytics, and connectivity. Automation technologies, such as process automation, robotic process automation (RPA), machine learning, and the Internet of Things (IoT), have become integral parts of various industries.

1.3 WHAT IS AUTOMATION?

Automation is the use of machinery with minimal human intervention in manufacturing processes. Industrial automation enables facilities to optimize their operations by implementing these automated processes. Manufacturers employ various machines that are controlled by different systems, such as Programmable Logic Controllers (PLCs), Human Machine Interfaces (HMIs), and Robotics. Through the application of logic and programming, automation systems provide instructions to machines on how to perform specific functions. These machines offer a high level of control, resulting in improved manufacturing performance.

The benefits of industrial automation in a manufacturing environment are numerous. Firstly, it enhances reliability by reducing the reliance on human intervention, minimizing errors, and ensuring consistent operation. Secondly, automation increases productivity by enabling faster and more efficient production processes, reducing cycle times, and increasing output. Thirdly, it enhances product quality by maintaining precise control over manufacturing parameters, reducing variability, and minimizing defects. Lastly, automation contributes to reduced labor expenses by replacing or reducing the need for manual labor, optimizing workforce utilization, and mitigating labor-related costs.

By incorporating industrial automation, manufacturing facilities can experience these advantages, leading to enhanced competitiveness, improved operational efficiency, and increased profitability[11].

1.3.1 SIMPLEST EXAMPLES FOR AUTOMATION

Imagine walking into a dark room. Normally, we would have to manually turn on the lights by flipping a switch on the wall. But there's a simpler way. We can install a special sensor that works like a switch. When it detects someone entering the room, it automatically turns on the lights. And when the person leaves, the sensor detects their absence and turns off the lights. This system saves us the effort of having to remember to switch the lights on and off.

Now let's apply the same idea to a factory. Picture a production line where raw materials come in at the start and finished products come out at the end. In between, some processes turn those raw materials into the final product. These processes can be done manually, with workers doing the job, or they can be

1.3. WHAT IS AUTOMATION?

automated. With automation, we can set up a system that takes care of the entire manufacturing process or parts of it automatically.

This concept applies not only to factories but also to other industries like automotive, oil, welding, and gas. By automating tasks, we can make things more efficient, increase productivity, improve the quality of products, and reduce the need for manual labor. Automated systems can be customized to fit specific needs, making the production process smoother and more precise[22].

1.3.2 LEVELS OF AUTOMATION

As can be seen in the following figure (fig. 1.1) [5], there exist five distinct levels of automation that range from broad to specific in their scope and function.

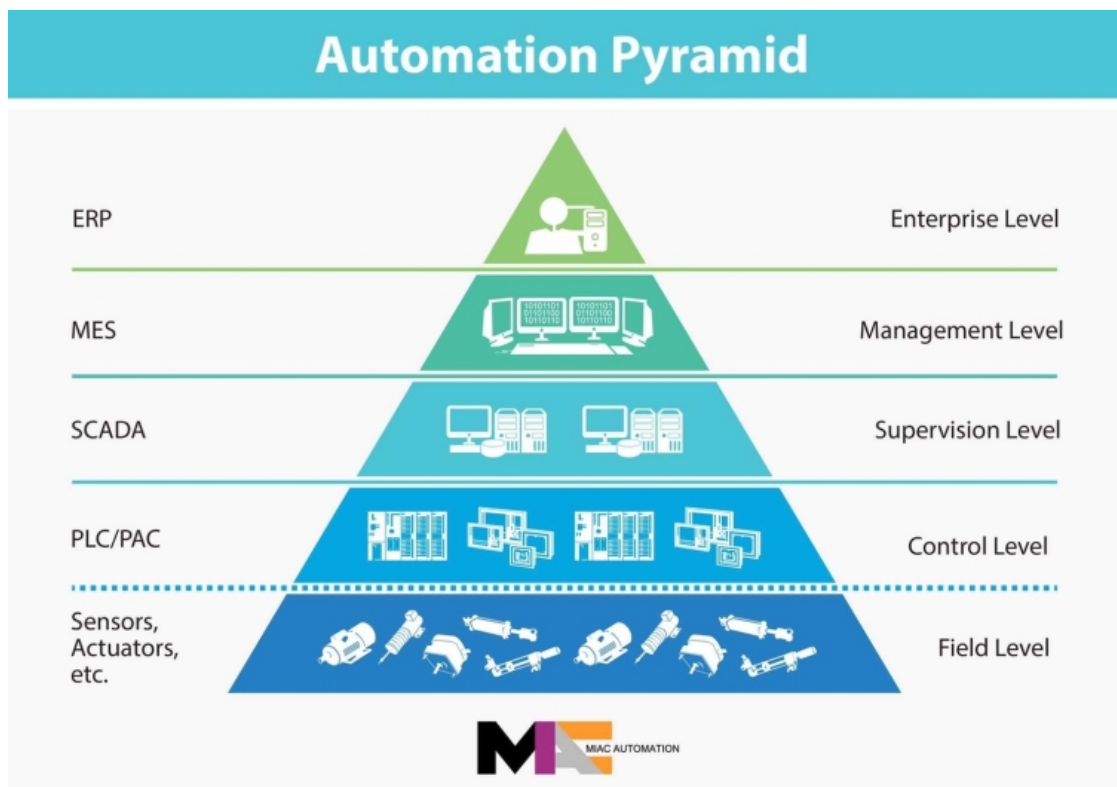


Figure 1.1: Automation Pyramid

Field Level: The lowest level of automation is the field level, which involves the deployment of devices such as sensors, code readers, valves, relays, and actuators. These devices serve the purpose of collecting and transferring data from processes and machinery to higher levels for monitoring, analysis, and control.

Control Level: The control level encompasses a range of automation devices, including robots, CNC machines, and programmable logic controllers (PLCs). These devices gather process parameters from sensors and other field-level equipment. Although human interaction is minimal at this stage, it supports the control function and strategy of automated systems.

Supervising Level: Operating at a more intermediate level, this stage involves the utilization of monitoring systems and automated devices to facilitate control and intervention within automated systems. Human-machine interfaces (HMIs) play a key role in supervising various parameters, setting production targets, managing machine startup and shutdown processes, and maintaining historical records. This level entails a higher degree of computer programming and human supervision, often employing technologies such as Supervisory Control and Data Acquisition (SCADA) or Distribution Control Systems (DCS).

Management Level: At the management level of automation, the Manufacturing Execution System (MES) serves as a central software system that connects planning and control with actual production activities. It collects data from various sources, like sensors and machines, providing real-time insights to decision-makers. MES helps monitor, control, and optimize production processes, enabling better management and boosting efficiency, quality, and productivity in manufacturing operations.

Information or Enterprise Level: This topmost level encompasses the management of the entire industrial automation system, including tasks such as production planning, order management, customer and market analysis, and sales. It primarily focuses on the commercial aspects of the company or warehouse, rather than technical operations.

1.3.3 MAIN TYPES OF AUTOMATION

Fixed Automation: Fixed automation, also known as hard automation, is a specific type of automation used in manufacturing processes where the setup and configuration remain unchanged (fig. 1.2)[5]. In other words, it is designed to perform the same task repeatedly without significant variations. Fixed au-

1.3. WHAT IS AUTOMATION?

tomation systems are controlled by programmed commands and directly connected to computers. The sequence of operations in fixed automation is relatively straightforward and not overly complex.

This type of automation is well-suited for applications that require high-speed, high-volume production with little need for flexibility or adaptability.

Fixed automation offers several advantages such as increased production capacity, improved operational efficiency, cost savings through reduced labor, enhanced quality assurance, heightened safety measures, and lower production expenses.

However, it also has its limitations, including a high initial investment, reduced flexibility, limited scalability, and difficulty in adapting to changes. Fixed automation finds applications in various industries, including machining transfer lines in the automotive industry, chemical processes, and assembly processes.

Programmable Automation: Programmable automation systems involve the use of computer-controlled machinery that can be easily reprogrammed or modified, providing the flexibility to implement new processes as needed. Two common types of programmable automation are Programmable Logic Controllers (PLCs) and Computer Numerical Control (CNC) machines. PLCs and CNC machines offer precise control and automation of industrial processes, enabling adaptation and modification of operations to meet changing requirements.

Employing programmable automation systems brings several benefits to industries. The flexibility of PLCs and CNC machines allows for efficient updates and customization of processes, optimizing operations and enhancing productivity and performance.

While programmable automation offers advantages, there are also some drawbacks to consider. The equipment can be expensive, and the complexity of implementation may require skilled technicians. Additionally, a higher initial investment is required, and maintenance and downtime can be significant factors.

Programmable automation finds applications in various fields, including logistic programming, intelligent robotic machines, and numerical control (NC) machine tools.

Flexible Automation: Flexible or Soft automation is an advanced form of programmable automation that offers enhanced flexibility and production ef-

efficiency. It combines the benefits of fixed automation and programmable automation, utilizing computer-controlled systems and robotics to perform a wide range of tasks.

What distinguishes flexible automation is its ability to handle different product types without complex reprogramming. The system can efficiently adapt and produce various products without significant reconfiguration efforts, reducing downtime and enhancing productivity.

By leveraging computer-controlled systems and robotics, flexible automation empowers manufacturers with a versatile solution that optimizes production processes. The ability to produce different product types without complex reprogramming boosts efficiency and adaptability in manufacturing operations.

Pros include the flexibility of products, increased customization, faster production, and seamless adaptation for batch production without the need for extensive reconfiguration. However, there are cons to consider, such as a higher cost per unit due to the specialized machinery required.

Flexible automation finds applications in various fields, including industrial robots and CNC machines.

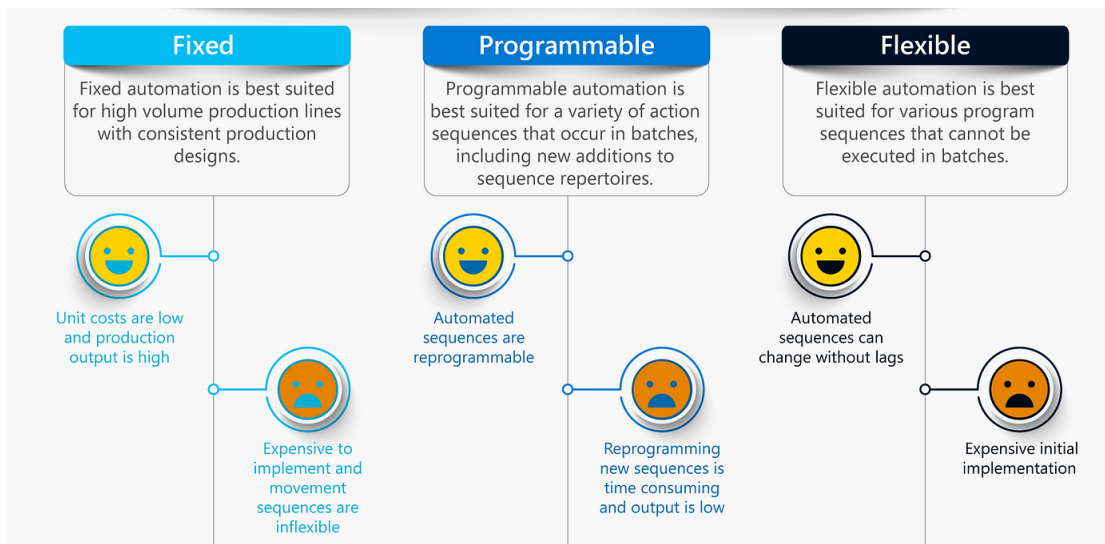


Figure 1.2: Advantages and Disadvantages of the Automation types

2

Related Works

2.1 LITERATURE REVIEW

In this chapter, important studies, theories, and real-world uses of modern control systems and automation are explored. By examining existing research, the aim is to understand how the work fits into the broader context of control theory. The focus is on key advancements, historical changes, and emerging trends. This review aids in understanding the theory and real-world impact of integrating PLCs and code readers in automation systems.

2.1.1 MODERN CONTROL SYSTEM

According to an abbreviated history of automation and industrial control systems by Ernie Hayden GICSP, Michael Assante, and Tim Conway[9], control theory is divided into two main categories: classical control theory and modern control theory. Classical control theory analyzes systems in the time domain using differential equations or in the frequency domain using Laplace Transform. It mostly deals with single input single output (SISO) systems. On the other hand, modern control theory uses state space tools to analyze systems and can handle multiple input multiple output (MIMO) systems. In terms of the flow of control signals, there are two common types: feedback control and sequence control. Feedback control involves using a controller that compares the measured value with a desired value (a previous set point) to adjust input parameters. Sequence control, on the other hand, follows a fixed sequence or

2.1. LITERATURE REVIEW

logic based on different system states to perform various actions.

Additionally, in the field of automation, there is a third practice based on the concept of hysteresis. It is used in pressure switches, speed controls, and other areas where smooth operation is desired.

In the past, before the 1950s, most control systems were analog, utilizing switches, contractors, relays, timers, and counters to turn machines or processes on and off. However, the introduction of digital control systems began in 1959 with the first operational digital control system in Port Arthur, Texas. Specialized computers capable of direct digital control emerged in the late 1960s, but they were costly and not sustainable in the market. The advent of multipurpose, smaller-sized, and affordable microcomputers in the early 1970s led to their widespread adoption and dominance in the industry[6].

2.1.2 PLC APPLICATIONS

In modern industry, Programmable Logic Controllers (PLCs) play a significant role in numerous applications, enabling efficient control, monitoring, and management of industrial processes. Some common applications of PLCs include water and wastewater treatment control systems, sun-tracking systems, wind energy systems, photo-voltaic applications, HVAC control, and manufacturing processes. These applications share a common characteristic of being modeled as process control problems, where the PLCs control devices and equipment based on real-time data from the process[21].

Haoqiang Ji.[13] conducted a test on a PLC-based water tank control system using a PID controller. Their system included an HMI programmed on NI-LabVIEW and an Allen Bradley Micro830 PLC connected through Modbus RTU communication protocol. The authors focused on system modeling, including water tank, transducer, and control valve modeling, as well as the application of the PID algorithm for improved control.

Apostolos Tsagaris and Evangelos Hatzikos[26] developed a networked platform for remote monitoring and controlling PLCs, utilizing the computing abilities of PCs and various communication options. Yathov Kumar Bala Kumar [2] designed a PLC-based level control system for training engineering students on PID control processes.

2.1.3 PLC NETWORKS

According to a study by Zoltán Rajnai and István Kocsis[20] on assessing industry 4.0 readiness of enterprises, Programmable Logic Controllers (PLCs) offer various networking options for control and communication purposes. PLCs commonly support remote I/O, peer-to-peer, host computer communications, and LANs. Additionally, some PLC vendors have proprietary networks for communication within their PLCs. However, PLCs from different manufacturers can communicate with each other using an ASCII interface and platform-independent software.

Remote I/O Systems: PLCs have input and output pins located at a distance from the processor. The processor controls these I/O points, both digital and analog, through a master-slave configuration.

Peer-to-Peer Networking: In peer-to-peer networks, multiple PLCs are interconnected, allowing communication signals to pass from one point to the next sequentially. Each PLC has a duplicate memory table in its processor, and they are programmed for specific functions in different areas. Changes made in one PLC are automatically transferred to others in the network.

Computer and PLC Connection: PLCs can be directly connected to computers using various communication protocols like RS-232C, RS-422, RS-485, or Ethernet. This connectivity makes PLCs versatile and suitable for data processing, acquisition, and programming in high-level languages like ladder logic. It also facilitates easy download of programs to PLCs, documentation, and better interaction with operator interfaces or Human-Machine Interfaces (HMIs).

2.1.4 FOURTH INDUSTRIAL REVOLUTION

According to studies by Mario Hermann, Tobias Pentek, and Boris Otto [10], the history of the Industrial Revolution can be divided into four periods. The first period, during the late 18th century, saw the predominance of mechanical machines powered by water and steam. The second period, known as Industry 2.0, spanned the beginning to the mid-20th century, marked by the introduction of electricity, mass production, and assembly line practices.

2.1. LITERATURE REVIEW

The third industrial revolution, Industry 3.0, emerged after the 1960s with the introduction of computers, semiconductor devices, and the internet, which brought revolutionary changes to various industries. Presently, the focus is on the concept of Industry 4.0, characterized by cyber-physical systems (CPS), the Internet of Things (IoT), and the Internet of Service. The vision of the smart factory, where CPS communicates and cooperates in real-time via the Internet, aims to optimize internal and cross-organizational services in the value chain[8].

THE PRINCIPLES OF INDUSTRY 4.0 ENCOMPASS SIX FUNDAMENTAL DESIGN ASPECTS

Interoperability: Systems and components can seamlessly communicate and interact with each other, ensuring smooth data exchange and cooperation.

Virtualization: Physical processes are digitally represented in a virtual environment, enabling simulation and optimization.

Decentralization: Decision-making and control are distributed among various components and entities, promoting flexibility and autonomy.

Real-Time Capability: Systems and processes operate in real-time, enabling prompt responses and dynamic adjustments.

Modularity: Components and systems are organized into modular structures, facilitating flexibility and scalability.

Service Orientation: Services are designed to be reusable and accessible, promoting collaboration and efficient resource utilization.

2.2 OVERVIEW OF WELDING MACHINE AUTOMATION

Welding machine automation has become a game-changer in modern industries, where automated systems take charge of welding processes, either assisting or replacing human operators in repetitive or intricate tasks. The implementation of automation in welding brings forth a multitude of advantages, such as increased efficiency, improved weld quality, enhanced safety, and reduced labor costs. These automated systems employ cutting-edge technologies, including robotic arms, vision systems, and sophisticated control algorithms, to achieve precise and consistent welds.

The scope of automated welding machines is extensive, encompassing various welding techniques like arc welding (MIG, TIG, and stick welding), resistance welding, laser welding, and electron beam welding. These machines are engineered to handle diverse materials, thicknesses, and joint configurations, making them adaptable for use across different industries, including automotive, aerospace, construction, and manufacturing.

By integrating welding machine automation, manufacturers can significantly boost productivity, achieve faster cycle times, and ensure superior weld quality, leading to enhanced overall efficiency and a competitive edge in the market[25].

TYPES OF WELDING MACHINES USED IN AUTOMATION

Various types of welding machines are utilized in automation to streamline and enhance the welding process across different industries.

Robotic Welding Cells employ industrial robots equipped with welding torches, offering exceptional flexibility, precision, and repeatability. These systems are capable of executing complex welds in various orientations and positions, contributing to efficient and consistent welding operations.

Automated Resistance Welding Machines utilize electrical resistance to generate heat and join metal parts. Equipped with programmable controls, these machines precisely manage welding parameters such as current, voltage, and weld time. They are particularly suitable for high-speed welding applications, such as automotive manufacturing, where rapid and precise welding is essential.

2.2. OVERVIEW OF WELDING MACHINE AUTOMATION

Laser Welding Machines utilize laser beams to generate heat and create welds. They offer remarkable precision, minimal heat-affected zones, and the ability to weld intricate geometries with high accuracy. These machines are indispensable in industries where precise and high-quality welds are crucial, such as medical device manufacturing and electronics.

These diverse types of welding machines, each with its unique capabilities and applications, contribute to the efficiency, quality, and precision of welding processes in automated manufacturing environments.

2.3 PRINCIPLES AND WORKING MECHANISMS OF CODE READERS

Code readers utilize optical sensors and decoding algorithms to capture and interpret code data. The working principle involves several steps: illumination, reflection, sensing, and decoding. During illumination, the reader emits light onto the code, which is then reflected back to the sensor or camera. The sensor captures the reflected light, creating a pattern of dark and light bars, which is analyzed by the decoding algorithm to extract encoded information.

There are different types of code readers tailored for various applications. Handheld scanners are portable devices used in industries like retail and logistics for manual scanning. Fixed-mount scanners are stationary devices integrated into automated systems for continuous scanning in manufacturing and assembly lines. Wearable scanners, worn by users, enable hands-free operation, particularly useful in environments like warehouses. Mobile scanners, integrated into smartphones or tablets, facilitate on-the-go scanning for field services and inventory management.

ADVANTAGES OF USING CODE READERS IN AUTOMATION

Code readers offer several advantages in automation systems. They ensure rapid and accurate data capture by eliminating manual entry, thereby reducing errors and ensuring precise information retrieval. Additionally, they streamline workflow efficiency by automating data collection tasks, leading to improved inventory management and tracking capabilities.

Integration with automation systems enables real-time data exchange, allowing for instant transmission of captured code data to central databases or control systems. This facilitates timely decision-making and process optimization. Furthermore, code readers enhance error prevention by minimizing the risk of transcription errors and ensuring accurate identification, ultimately improving product quality and customer satisfaction.

Moreover, code readers offer scalability and flexibility in automation setups. They can handle various code types and easily adapt to changing production requirements. This scalability, coupled with cost efficiency through reduced labor costs and optimized inventory management, code readers invaluable tools for enhancing efficiency, accuracy, and productivity across different industries.

2.3. PRINCIPLES AND WORKING MECHANISMS OF CODE READERS

IMPORTANCE OF CODE READERS IN WELDING MACHINE AUTOMATION

Code readers play a crucial role in welding machine automation by efficiently and accurately identifying work-pieces, components, and materials. They enable the welding machine to retrieve welding programs and parameters automatically, based on scanned barcodes or QR-codes attached to work-pieces, ensuring correct weld setup and consistency. Additionally, code readers facilitate material traceability by verifying the authenticity of raw materials, consumables, and components, tracking their usage, and recording essential data such as lot numbers and quality certifications. This ensures proper documentation and quality control throughout the welding process.

Furthermore, code readers assist in selecting welding parameters by scanning codes associated with specific welding procedures or job setups, minimizing setup time, reducing errors, and ensuring consistent and optimized weld quality. They also play a role in data collection and analysis during the welding process, capturing information such as time stamps, weld counts, and material usage. This data facilitates production tracking, quality control analysis, and process optimization, contributing to efficient data management and informed decision-making in welding machine automation.

2.4 CASE STUDIES AND INDUSTRY APPLICATIONS

In this section, real-world applications and examples of code reader integration in automation, particularly in welding machine operations, are explored. The focus is on examining how code readers are utilized in various industrial settings, highlighting their role in enhancing efficiency and productivity within welding machine operations.

REAL-WORLD APPLICATIONS OF CODE READERS INTEGRATION IN AUTOMATION

Code reader integration finds widespread application in production line automation across various industries. In welding machine automation, these readers play a crucial role in scanning codes on work-pieces or assemblies. This triggers automated welding processes based on the scanned data, ensuring precise identification, proper setup, and optimized welding parameters. Such integration enhances efficiency and quality in production lines.

Moreover, code reader integration is pivotal in automated inventory management systems within welding machine automation[18]. By scanning codes on raw materials, consumables, or welding electrodes, real-time data on material usage, inventory levels, and reordering requirements are provided. This facilitates efficient inventory management, minimizes stock-outs, and improves supply chain visibility.

Furthermore, code reader integration is paramount for quality control and traceability in welding machine automation. By scanning codes on work-pieces, components, or finished products, essential data related to weld parameters, inspection results, and quality certifications are captured. This ensures adherence to quality standards, facilitates traceability for auditing purposes, and enables effective root cause analysis in case of quality issues.

SOME OTHER EXAMPLES OF SUCCESSFUL CODE READER INTEGRATION IN WELDING MACHINES

In automotive manufacturing, for instance, code readers seamlessly integrate into welding machines to streamline the assembly process. By scanning codes on specific components, welding machines verify compatibility and retrieve precise welding parameters for each part, ensuring error-free assembly, boosting production efficiency, and enabling reliable traceability [7].

2.4. CASE STUDIES AND INDUSTRY APPLICATIONS

Similarly, in the shipbuilding industry, code readers are integrated with welding machines to scan codes on steel plates, beams, and other parts. This provides comprehensive information about materials, welding procedures, and quality standards, ensuring meticulous documentation, stringent quality control, and a smoother welding process.

In the aerospace sector, code reader integration is essential for maintaining strict quality control and traceability standards. Code readers incorporated into welding machines scan codes on aircraft parts, such as fuselage sections and engine components. This enables accurate part identification, traceability of welding parameters, and compliance with rigorous regulatory requirements, strengthening quality control measures and ensuring adherence to industry regulations in aerospace manufacturing.

3

Overview of Some of the Key Automation Components

In this section, the fundamental components constituting the foundation of modern automation systems are explored. These components operate synergistically to facilitate smooth control, monitoring, and management of industrial processes.

3.1 INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCs)

Programmable Logic Controllers (PLCs) are digital computing devices employed to automate industrial processes. They are specifically designed to monitor input signals, execute logical instructions, and control output devices based on the programmed logic. PLCs find extensive application across various industries, including manufacturing, energy, and transportation, where they are responsible for tasks such as machinery control, process monitoring, and data collection.

PLCs consist of three essential components: input modules, a central processing unit (CPU), and output modules. The input modules receive signals from sensors and switches, while the CPU processes the logic program and makes decisions based on the input signals. Subsequently, the output modules activate actuators and devices, such as motors, valves, and indicators, by the

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

CPU's instructions.

The utilization of PLCs offers several advantages, including flexibility, scalability, reliability, and ease of programming. Their inherent programmability enables easy and efficient customization to accommodate changes in the automation process, making them highly adaptable to evolving requirements[3].

3.1.1 EVOLUTION OF CONTROL SYSTEMS: FROM DCS TO PLCs

PLCs, or Programmable Logic Controllers, have evolved as a significant advancement from earlier control systems, such as relay-based systems and Distributed Control Systems (DCS). While DCS systems offered advanced control and monitoring capabilities, they were characterized by complex and expensive centralized architectures. In contrast, PLCs were introduced in the late 1960s and early 1970s as a simpler and more cost-effective alternative. Adopting a decentralized architecture, PLCs employed distributed control modules that could be programmed to perform specific tasks. Initially replacing relay-based control systems, PLCs primarily fulfilled simple logic functions. However, over time, they underwent significant advancements, incorporating more sophisticated features and becoming widely utilized in industrial automation[23].

PLCs have been in existence since the 1970s, initially designed to replace traditional relays and timers with user-created software programs for decision-making purposes. Field devices such as switches, sensors, and motors were connected to add-on modules, which were then linked to the microprocessor module. Over time, PLCs evolved to incorporate analog control and support PID-controlled processes. On the other hand, personal computers (PCs) operate on operating systems like Microsoft Windows, Mac OS, or Chrome OS, and have a different history from PLCs. PLCs are programmed using proprietary software such as Rockwell Studio 5000 or Siemens TIA Portal, utilizing IEC 61131-3 standard languages. PC programming typically involves structured text and higher-level languages like C++. Execution of a program differs between PLCs and PCs, with PLCs following a scan-based program execution and PC software programs being event-driven.

Notable distinctions lie in their construction and environment. PLCs are designed for reliable operation in harsh industrial environments, prioritizing ease of programming and troubleshooting, and providing inherent resistance to viruses and cyber-attacks. PCs, however, are more susceptible to such risks and

are designed for friendly environments like offices or homes. PLCs offer flexibility for expansion, with the ability to add input/output modules and integrate various industrial communication protocols. Industrial PCs, a newer addition, possess an operating system similar to PCs but are designed to withstand harsh environments, extreme temperatures, and physical stress. Industrial PCs have the advantage of familiar programming using the Windows operating system, making them accessible to a wider range of users. Traditional PLC programmers are well-versed in IEC 61131-3 languages such as Ladder Diagram or Function Block. However, the Industrial PC allows a new generation of IT professionals to develop control programs using higher-level languages like C++.

3.1.2 THE MAJOR PLCs MANUFACTURERS

Multiple companies serve as prominent manufacturers of Programmable Logic Controllers (PLCs) on a global scale. Some of the major PLC manufacturers are:

Siemens: Siemens is a renowned multinational conglomerate headquartered in Germany and recognized as a leading global player in industrial automation. They offer an extensive portfolio of PLC products, with their SIMATIC series gaining prominence due to their dependable performance and scalability.

Rockwell Automation/Allen Bradley: Rockwell Automation, operating under the Allen Bradley brand, is a prominent American company specializing in industrial automation solutions. Their PLC lineup includes widely utilized models like ControlLogix and CompactLogix, known for their versatility across various industries.

Mitsubishi Electric: A multinational corporation from Japan, holds a significant presence in the PLC market. Their MELSEC series of PLCs are highly regarded for their durability and adaptability, catering to diverse automation requirements.

Schneider Electric: A multinational company headquartered in France, is renowned for its comprehensive range of automation solutions, encompassing PLCs among other offerings. Their Modicon PLCs enjoy widespread usage in sectors such as

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

manufacturing, energy, and infrastructure.

Omron: A Japanese firm specializing in automation and control products, offers a diverse array of PLCs, including the CJ and NX series. These PLCs are appreciated for their advanced features and user-friendly programming environment.

Emerson (General Electric): Emerson, formerly affiliated with General Electric (GE), is a global technology and engineering corporation. Although GE sold its automation business to Emerson, their PLCs like the PACSystems RX3i series continue to serve a range of industrial applications.

3.1.3 PROGRAMMING LANGUAGES FOR PLCs AND THEIR COMPARISON

The selection of a PLC programming language is crucial for specific applications, with each language offering unique features and suitability. Understanding the main characteristics, strengths, and weaknesses of the five IEC 61131-3 programming languages can assist in determining which languages to prioritize[4].

The languages are as follows:

Ladder Diagram (LD): Ladder Diagram (fig. 3.1), commonly referred to as Ladder Logic (LD), is a visual programming language that draws inspiration from relay ladder logic. It employs a combination of rungs and logic elements like contacts and coils to represent control logic.

LAD is extensively utilized in PLC programming, appreciated for its graphical representation and user-friendly nature, particularly benefiting electricians and technicians in comprehending the logic.

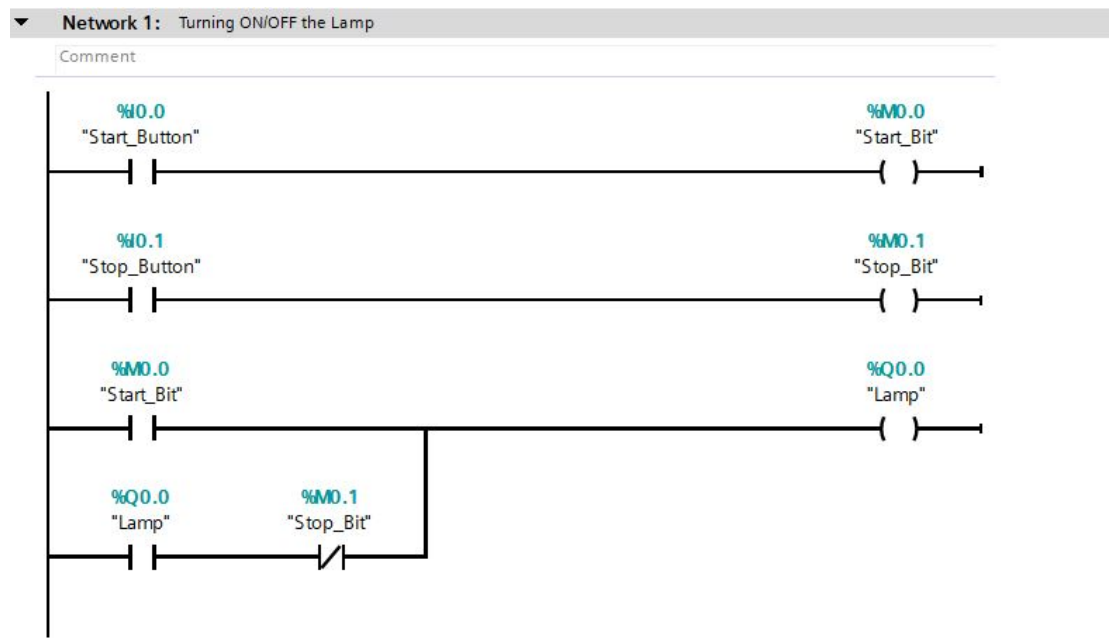


Figure 3.1: Ladder Diagram (LD)

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

Function Block Diagram (FBD): Function Block Diagram (fig. 3.2) is a graphical programming language that utilizes blocks to symbolize functions or operations. These blocks can be interconnected to define the control logic.

FBD proves to be highly suitable for handling intricate programming tasks and allows the creation of reusable function blocks, thereby enabling modular programming.

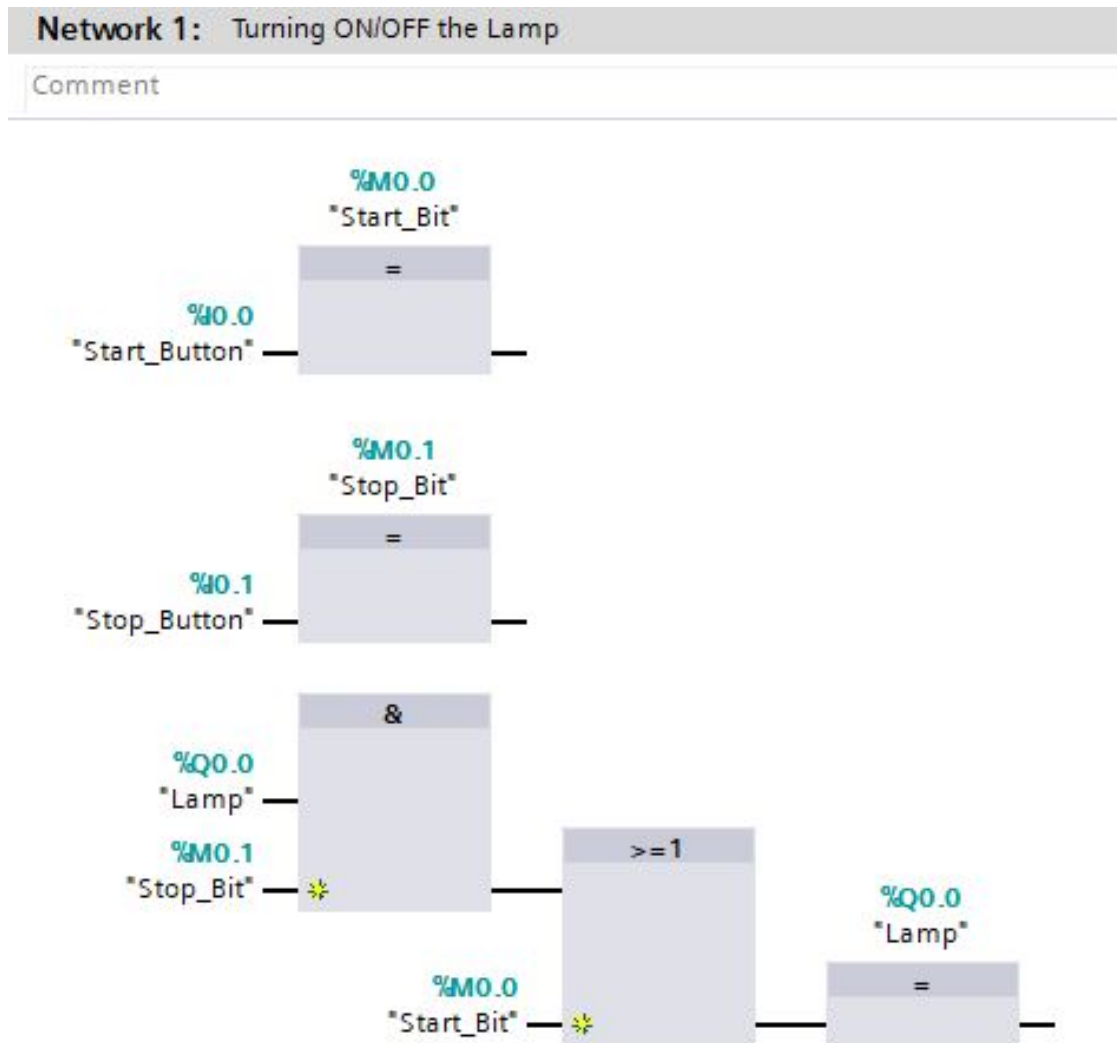


Figure 3.2: Function Block Diagram (FBD)

Structured Control Language (SCL): Structured Control Language (fig. 3.3) is a programming language that adheres to the IEC 61131-3 standard and is primarily text-based. It integrates features from high-level programming languages with specialized functionalities for programmable logic controllers (PLCs).

SCL enables structured programming techniques, including the creation of user-defined functions and data types, making it well-suited for implementing intricate control algorithms and performing sophisticated data manipulation tasks.

```
Network 2: Turning ON/OFF the Lamp
Comment
1 IF "Start_Button" THEN
2   ;
3   "Start_Bit" := 1;
4 END_IF;
5
6 IF "Stop_Button" THEN
7   ;
8   "Stop_Bit" := 1;
9 END_IF;
10
11
12 IF "Start_Bit" OR ("Lamp" AND NOT "Stop_Bit") THEN
13   ;
14   "Lamp" := 1;
15 END_IF;
```

Figure 3.3: Structured Control Language (SCL)

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

Structured Text (ST): Structured Text (fig. 3.4) is a programming language that shares similarities with conventional programming languages like Pascal. It enables programmers to write code using structured statements and offers a diverse set of programming constructs, including loops, conditions, and the ability to create custom functions.

ST provides flexibility and is particularly well-suited for intricate mathematical calculations and data manipulation tasks.

```
Network 6: .....  
Comment  
1      CALL CTU , "CounterL_DB"                                %DB5  
2      Int  
3      CU :=#Lbox_passed  
4      R :="Stop"                                             %I0.1  
5      PV :=0                                                0  
6      Q :=  
7      CV :="num_L"                                          %MW2  
8  
9      CALL CTU , "CounterS_DB"                                %DB6  
10     Int  
11     CU :=#Sbox_passed  
12     R :="Stop"                                             %I0.1  
13     PV :=0                                                0  
14     Q :=  
15     CV :="num_S"                                          %MW4
```

Figure 3.4: Structured Text (ST)

Sequential Function Chart (SFC): Sequential Function Chart (fig. 3.5) is a graphical programming language that enables programmers to outline the sequential behavior of a process. It presents the process as a sequence of steps and transitions, resembling a state machine.

SFC is well-suited for programming systems that require sequential control or intricate processes with various operational modes.

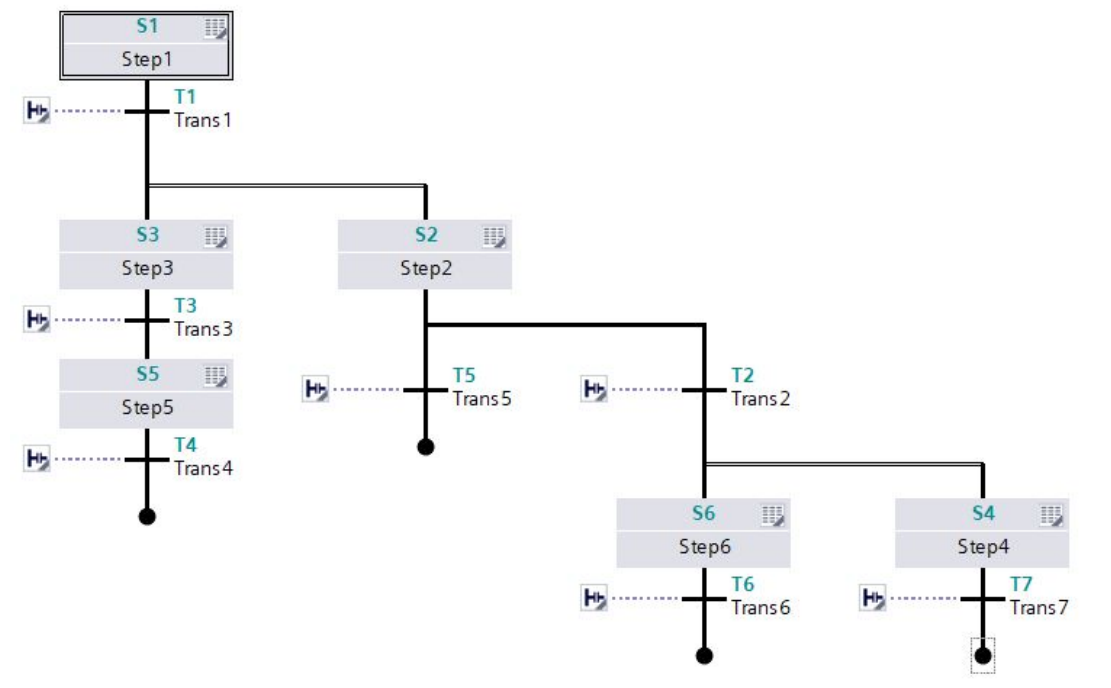


Figure 3.5: Sequential Function Chart (SFC)

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

Instruction List (IL): Instruction List (fig. 3.6) is a programming language that operates at a low level and employs mnemonic codes to express instructions. It bears a resemblance to assembly language and is appropriate for programmers with expertise in low-level programming.

IL is utilized to craft efficient and optimized code in scenarios where achieving optimal performance is of utmost importance[24].

Network 6:	
Comment	
1	CALL CTU , "CounterL_DB" §DB5
2	Int
3	CU :=#Lbox_passed
4	R := "Stop" §IO.1
5	PV :=0 0
6	Q :=
7	CV := "num_L" §MW2
8	
9	CALL CTU , "CounterS_DB" §DB6
10	Int
11	CU :=#Sbox_passed
12	R := "Stop" §IO.1
13	PV :=0 0
14	Q :=
15	CV := "num_S" §MW4

Figure 3.6: Instruction List (IL)

3.1.4 INTRODUCTION TO TIA PORTAL (SIEMENS TOTALLY INTEGRATED AUTOMATION)

The TIA Portal, which stands for Totally Integrated Automation Portal, is a software engineering framework developed by Siemens. It serves as a unified platform for programming and configuring automation systems, integrating various components such as programmable logic controllers (PLCs), human-machine interfaces (HMIs), drives, and networking devices.

With its user-friendly interface, the TIA Portal enables engineers and technicians to efficiently design, program, and maintain automation systems. It provides a centralized environment for managing projects, programming, simulating, commissioning, and diagnosing. Supporting multiple programming languages like ladder logic (LAD), function block diagrams (FBD), structured text (SCL), Instruction List (IL), and graphical programming (Graph), the TIA Portal caters to diverse programming needs.

The Siemens TIA Portal is highly regarded for its comprehensive set of tools and functionalities, simplifying the development and operation of automation systems. It facilitates seamless integration, efficient engineering workflows, and improved system performance.

Program Blocks in TIA Portal (fig. 3.7)

Organization Block (OB): OBs serve various purposes in the program, such as organizing the program flow, handling hardware and software faults, executing cyclic interrupts, and more. Among all the OBs, OB1 is the primary block responsible for program organization.

Function Block (FB): FBs are utilized for establishing logical connections between signals and variables. Each function block call necessitates an instance data block that stores function-specific private data between calls. This may include information like counters, edge bits, and other relevant data for that particular function.

Function (FC): FCs, or Function Blocks without memory, are utilized when there is no need to store any data between function calls. These blocks primarily serve for logic execution and calculations.

Data Block (DB): DBs are used to hold data in organized structures, such as send/receive buffers, data arrays, or other specific data types needed for data exchange within the program.

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

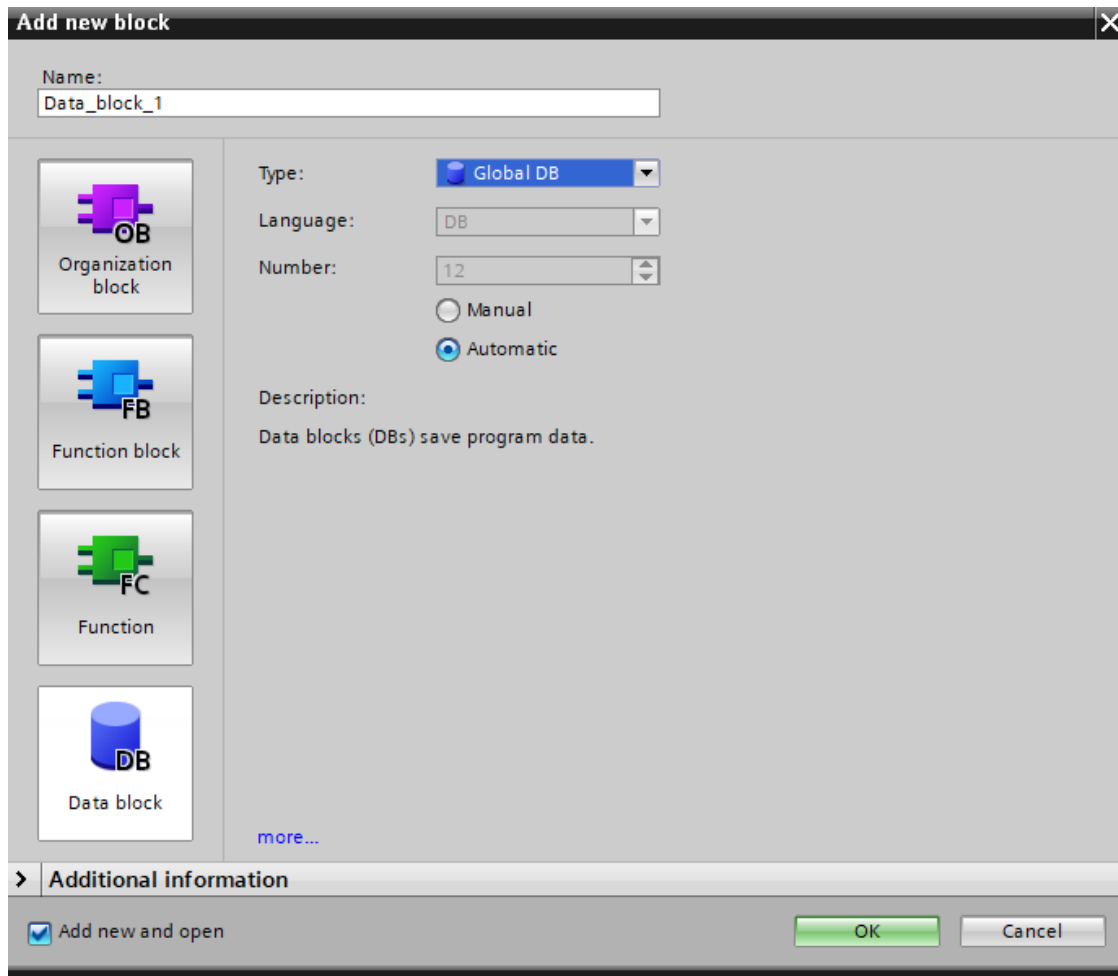


Figure 3.7: Program Blocks

KEY FEATURES AND CAPABILITIES OF TIA PORTAL FOR PLC PROGRAMMING

TIA Portal, tailored for PLC programming, offers a range of specialized features and capabilities. Central to its offerings is an Integrated Development Environment (IDE), which streamlines PLC program development by providing a unified platform encompassing programming, configuration, diagnostics, and visualization. This environment ensures consistency across multiple PLCs and modules.

TIA Portal supports an array of programming languages, including ladder logic, function block diagrams, structured text, and graphical programming, empowering programmers to choose the most fitting language for their tasks. The platform also boasts an extensive library of pre-designed function blocks, simplifying complex control algorithms and expediting development.

Simulation and testing tools allow for thorough program validation before deployment, while diagnostic and monitoring features facilitate real-time oversight of variables, system states, and communication, aiding swift issue detection and resolution. These capabilities collectively enhance the efficiency and reliability of PLC programming within the TIA Portal framework.

IMPORTANCE OF TIA PORTAL IN INDUSTRIAL AUTOMATION

The significance of the TIA Portal in industrial automation is underscored by its ability to streamline engineering workflows and centralize automation components. This unified approach boosts efficiency, accelerates development, and fosters collaborative teamwork.

TIA Portal's seamless integration of diverse automation elements, such as PLCs, HMIs, drives, and networks, fosters interconnected systems that excel in data exchange and control.

Furthermore, the platform's optimization tools and diagnostics enhance system performance while facilitating scalability for future growth. Notably, TIA Portal's standardized practices curtail engineering and maintenance costs, minimizing errors and enabling efficient troubleshooting. Additionally, its operational flexibility is amplified by remote access, data logging, and reporting capabilities, which empower informed decision-making and system optimization.

3.1.5 PLC HARDWARE COMPONENTS

A programmable logic controller (PLC) is a computerized system that comprises various hardware components. These components include a processor, power supply, input/output (I/O) modules, and a programming device.

Power Supply: The power supply, connected to the AC mains, provides the necessary DC voltage to power all the other modules associated with the PLC. However, it does not supply power to field devices.

Input/Output Modules: The I/O modules are responsible for connecting the PLC to digital or analog field devices. Input field devices, such as switches, encoders, and transmitters, provide information to the PLC. Output field devices, like relays, lamps, and proportional valves, receive commands from the PLC.

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

Central Processing Unit (CPU): The CPU is the core component of the PLC and performs the processing and control functions. It executes the logic program, communicates with input/output modules, and controls the operation of the PLC system.

Memory: PLCs have different types of memory, including program memory (for storing the logic program), data memory (for storing variables and values), and retentive memory (for storing data even when the power is turned off).

Communication Interfaces: PLCs commonly incorporate communication interfaces that facilitate connectivity with external devices or networks. These interfaces serve as vital conduits for establishing communication links and exchanging data. Widely utilized communication interfaces encompass Ethernet, serial communication (RS-232/485), and Fieldbus protocols such as Profibus, Profinet, or DeviceNet. These interfaces play a pivotal role in enabling seamless integration with diverse devices and networks, ensuring effective information exchange within the automation system.

3.1.6 BASIC OF PLCs COMMUNICATIONS

PLC communications involve the exchange of data between devices, and this is made possible through communication protocols. Think of communication protocols as sets of rules that govern how data is transmitted and received between two or more devices. They serve as the pathway or channel through which devices can connect and interact with each other.

Imagine communication protocols as a language that allows devices to understand and exchange information effectively. Without these protocols, devices can physically connect, but they will not be able to communicate or share data .

MOST COMMONLY AVAILABLE PROTOCOLS WITH PLC IN INDUSTRIES:

Modbus RTU: Modbus RTU is a widely used open serial protocol in Industrial Automation Systems (IAS), Home Automation, Building Management, Robotics, and more. It originated from Modicon (now Schneider Electric) and is known for its simplicity and reliability. Modbus RTU messages consist of a 16-bit structure with a CRC for ensuring data integrity.

PLC communication protocols primarily use RS-232 or RS-485 serial interfaces, making it easy to integrate Modbus-compatible equipment into various monitoring and control applications. Key components of communication protocols include baud rate, start/stop bits, parity bit, network length, and number of nodes.

RS-232 is an asynchronous communication method, commonly used between PLCs and devices like computers or modems. It represents data in binary format using voltage levels. RS-485, on the other hand, supports multi-drop and two-wire communication, allowing connection to multiple devices simultaneously. It can handle up to 32 devices, extendable using repeaters, with a maximum distance of 1200 meters.

RS-485 has the advantage of long-distance communication but requires careful programming due to sharing the same two wires for sending and receiving data. Only one node can transmit data at a given time, making it more complex to program compared to RS-232.

Ethernet/IP and Ethernet TCP/IP: In the world of PLC communication protocols, two significant ones are Ethernet/IP and Ethernet TCP/IP.

IP (Inter-network or internet protocol) is responsible for sending data between computers, using Ethernet links or other types of connections like Wi-Fi. Its main role is to determine the best path for each data packet to travel from the source to the destination.

TCP (Transmission control protocol) is a protocol that operates via IP and is essential for managing data. It interprets the data, using "ports" to categorize different types of information, and ensures that data packets are treated in the correct order.

Ethernet/IP is an open application protocol managed by ODVA and used by Ethernet modules in various PLCs like Allen Bradley, Schneider Electric, and Omron. It is an Ethernet adaptation of the Control Information Protocol (CIP). With Ethernet/IP, data can be exchanged deterministically in both directions between a PLC and remote devices. This communication happens at regular intervals known as the Requested Packet Interval (RPI). Ethernet/IP is a fast and user-friendly communication method, reducing the need for extensive ladder logic for communication and allowing communication even when the PLC is in Program mode.

On the other hand, TCP/IP is responsible for transmitting packets, which

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

consist of Modbus frames containing commands for reading and writing into the shared memory of a device. TCP/IP is easy to learn, as it is a standard in many programming languages, like C/C++, Matlab, and Python.

Modbus TCP/IP: Modbus TCP/IP is a straightforward Modbus protocol that operates over an Ethernet connection using a TCP interface. It allows for the smooth management and transmission of data between different layers without being influenced by the specific protocol used in the next immediate layer.

Also known as Modbus-TCP, this protocol combines Modbus, which is an application protocol, with TCP (Transmission Control Protocol) and IP (Internet Protocol). TCP and IP work together as the transport protocol for the internet. When Modbus data is transmitted using these protocols, the information is passed to TCP, where additional details are added, and then given to IP. IP then encapsulates the data into a packet (or datagram) and sends it over the network. Since TCP is a connection-based protocol, it must establish a connection before transferring data. In the case of Modbus TCP, the Master (or Client) initiates a connection with the Slave (or Server). The Server waits for incoming connections from Clients. Once a connection is established, the Server responds to queries from the Client until the client decides to close the connection.

IO Link: IO-Link is a versatile and robust communication protocol used primarily in plant and factory automation. It establishes a bidirectional connection between sensors and actuators and a central controller, usually a PLC. The connections are made using a short, unscreened three-wire cable that's no longer than 20 meters. The IO-Link master manages digital and analog signals and can integrate with various industrial communication standards like Profinet, Profibus, and Modbus.

IO-Link has four different operating modes, with the primary mode being for IO-Link communication. It operates at 24 volts, and if there are transmission errors, the message is reattempted up to two times before signaling a communication failure to the controller, alerting operators or maintenance staff.

IO-Link transmissions include Process Data (sensor or actuator states), Value Status (validity of Process Data), and Event (messages like error warnings). These transmissions occur independently of each other, ensuring that critical messages are not delayed by buffered messages.

Profibus and Profinet: PROFIBUS and PROFINET are industrial automation protocols developed by the same organization, PROFIBUS and PROFINET International (PI). While sharing a common source, these protocols differ in their communication capabilities.

PROFIBUS is a classical serial fieldbus, whereas PROFINET is an industrial Ethernet standard. They both utilize GSD files for hardware definition and share application profiles, ensuring interoperability and interchangeability among devices from various manufacturers. While PROFIBUS networks are still widely used, PROFINET offers superior performance, scalability, and future-proof capabilities due to its utilization of standard Ethernet. PROFINET's open industrial Ethernet standard allows for the adoption of new technologies, positioning it as a reliable and future-ready solution for industrial automation.

However, there are noticeable distinctions between the two protocols. In terms of the physical layer, PROFIBUS utilizes RS-485, while PROFINET opts for Ethernet, leading to swifter speeds and greater adaptability. In specifics of speed and capacity, PROFIBUS operates at 12 Mbit/s, whereas PROFINET supports 100 Mbit/s, accompanied by a larger telegram size and an unrestricted address space, outperforming the limitations of PROFIBUS. In the realm of technology, PROFIBUS operates on a master/slave communication model, while PROFINET employs a provider/consumer model. This unique approach of PROFINET enables enhanced connectivity and control, supporting machine-to-machine communication and vertical integration. Furthermore, in terms of connectivity, PROFIBUS permits wireless communication with proprietary radios, while PROFINET utilizes standard Ethernet protocols like Wi-Fi and Bluetooth, keeping up with the latest strides in wireless technology[15].

MIGRATION STRATEGIES:

To migrate from PROFIBUS to PROFINET, two approaches can be considered depending on the existing network infrastructure:

Greenfield Installation: In cases where no network exists, a new network can be designed based on the specifications and requirements of the future factory, ensuring a seamless transition to PROFINET.

3.1. INTRODUCTION TO PROGRAMMABLE LOGIC CONTROLLERS (PLCS)

Brownfield Installation: In situations where an installed network already exists, two migration approaches can be pursued:

a. **Step-wise Migration:** Legacy field buses can be gradually upgraded to industrial Ethernet in specific areas of the factory. This allows scheduling upgrades during downtime or maintenance periods, minimizing disruptions to production.

b. **Rip-and-Replace:** Outdated hardware or Fieldbus investments reaching the end of their lifecycle can be completely replaced with PROFINET. Though disruptive, this approach offers immediate benefits and can be strategically scheduled to minimize production disruptions.

For communication between the existing fieldbus and PROFINET networks, PROFINET proxies can be used. These proxies enable seamless communication and translation of PROFIBUS data within the PROFINET protocol.

The migration from PROFIBUS to PROFINET ensures a smooth and cost-effective transition, supported by various migration strategies and the availability of PROFINET proxies[27].

3.2 REMOTE TERMINAL UNIT (RTU)

Remote Terminal Unit, is an industrial device utilized in automation and control systems to supervise and regulate remote field devices and processes. RTUs are commonly employed in distant locations or distributed facilities, separate from the central control room.

The primary function of an RTU is to gather data from sensors, instruments, and other field devices, and transmit this information to a central control system. Conversely, it also receives commands or instructions from the control system to execute actions or control the remote devices.

RTUs are purpose-built to withstand challenging environmental conditions, including extreme temperatures, humidity, and electrical disturbances. They are equipped with diverse communication interfaces to establish connections with field devices, such as analog and digital input/output modules, serial ports, and network interfaces.

Aside from their data acquisition and control capabilities, RTUs often incorporate advanced features like data logging, event-based alarm notifications, and built-in diagnostics. They play a vital role in industries such as oil and gas, water and wastewater management, power generation, and telecommunications, where the ability to monitor and control remote processes is crucial.

3.3 SAFETY PLC

A Safety PLC is a crucial component of a Safety Instrumented System (SIS) or Safety Shutdown System. The primary function of the SIS is to continuously monitor equipment or processes and take action in the event of an unacceptable or unsafe condition, ensuring the safety of personnel, equipment, and the environment.

The Safety PLC serves as the Logic Solver within the Safety Instrumented System, constantly vigilant against potential plant failures. When hazardous conditions are detected, the Safety PLC responds by placing the plant in a safe state. In the past, traditional field devices and hardwired relays were used for monitoring and controlling shutdown conditions. However, with the advent of Safety PLCs, the reliance on hardwired relay logic has diminished.

Although a Safety PLC may appear similar to a regular PLC in physical appearance, it possesses distinct features and functionalities. The logic programmed into a Safety PLC is locked, incorporating safety signatures to ensure that the code remains unaltered. Moreover, a Safety PLC is equipped with additional hardware functionality and redundancy measures to meet specific Safety Integrity Level (SIL) requirements.

The Safety PLC goes beyond standard PLC capabilities by offering features such as wire fault detection, contact failure detection, and other monitoring capabilities. Additionally, it supports multi-channel redundancy for input/output monitoring and operation.

3.4 HUMAN-MACHINE INTERFACE (HMI)

A Human-Machine Interface (HMI) serves as a user interface or dashboard that facilitates the connection between a person and a machine, system, or device. Although the term "HMI" can be applied to any screen that enables user-device interaction, it is commonly associated with industrial processes.

HMIs share some similarities with Graphical User Interfaces (GUI), but they are not interchangeable terms; rather, GUIs are often incorporated within HMIs to provide visualization capabilities[12].

In industrial settings, HMIs play crucial roles, including:

Visually presenting data: HMIs display information in a visual format, making it easier for operators to comprehend and monitor critical data. Tracking production time, trends, and tags: HMIs enable tracking and analysis of production time, trends, and tags, aiding in efficient process management.

Overseeing Key Performance Indicators (KPIs): HMIs provide a centralized platform to oversee and assess KPIs, enabling operators to make informed decisions.

Monitoring machine inputs and outputs: HMIs facilitate real-time monitoring of machine inputs and outputs, ensuring smooth and reliable operations.

3.5 SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA)

SCADA systems are a combination of software and hardware components that enable the supervision and control of industrial plants, both locally and remotely. Traditionally, industrial processes were monitored and controlled manually using selector switches, push buttons, and analog dials. The introduction of relays and timers reduced the need for constant on-site personnel. However, these devices had limitations in terms of space, troubleshooting, and reconfiguration.

In the 1950s, the use of processors to control industrial plants became a reality. Industries such as gas and oil, utilities, and manufacturing embraced this technology. The term SCADA emerged in the following decade to describe systems using PLCs and microprocessors for monitoring and controlling processes on a larger scale. With advancements in computer systems, local area networking, and human-machine interface (HMI) software in the 1980s and 1990s, SCADA systems became capable of connecting to various related systems. Open system architectures and non-vendor-specific communication protocols further expanded SCADA's capabilities, leading to the development of networked SCADA systems.

Today, modern SCADA systems have evolved to leverage current technologies, offering significant advantages over their predecessors. They employ standards like SQL and web-based applications, allowing real-time access to plant information from anywhere. This accessibility enables improved plant operations, as operators can respond to SCADA system cues based on real-time field data and analysis.

In essence, SCADA systems consist of hardware and software components. Real-time data is collected from field devices, such as pumps and valves, and transmitted to processors like PLCs. The data is then distributed among a network of devices, including HMIs, end-user computers, and servers. Operators can interact with graphical representations of plant operations on HMIs and end-user computers, facilitating tasks like controlling pumps and valves[17].

3.5.1 DIFFERENCES BETWEEN SCADA AND HMI

HMI (Human Machine Interface) and SCADA (Supervisory Control and Data Acquisition) are two distinct components within industrial automation systems, each serving a specific purpose.

HMI refers to the graphical interface that enables operators to interact with and control industrial processes. It provides a visual representation of the plant's operations, allowing operators to monitor real-time data, access alarms, and alerts, and execute control actions. HMIs are typically installed on local operator panels or desktop computers, providing a user-friendly interface for operators to oversee and manage specific processes.

On the other hand, SCADA is a comprehensive system that encompasses both software and hardware components. It is designed to supervise, control, and acquire data from various industrial processes. SCADA systems collect data from field devices, such as sensors and actuators, and transmit it to a central control system. This data is then interpreted, analyzed, and displayed to operators through HMIs. SCADA systems provide advanced features like historical data logging, trend analysis, alarming, and reporting, enabling operators and engineers to make informed decisions and optimize process performance[1].

3.5.2 SCADA SYSTEMS AND DCS IN INDUSTRIAL AUTOMATION

SCADA (Supervisory Control and Data Acquisition) systems and DCS (Distributed Control Systems) are collections of software and hardware components used for the supervision and control of industrial plants, both locally and remotely.

A SCADA system is designed to gather and present data to operators, make decisions based on operator input, and control plant functions accordingly. Similarly, a DCS fulfills the same role in plant automation. SCADA systems may incorporate PLCs and RTUs for executing essential commands in plant operations. Communication protocols within SCADA systems have evolved to adapt to changing technologies, while DCS systems still employ some proprietary communications.

While there are similarities between the two systems, DCSs have integrated operator interface software with tag databases, whereas SCADA systems require additional software and tag configuration. In terms of processing time, SCADA

3.5. SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA)

systems may have a slight advantage for time-sensitive processes. However, DCSs offer advantages in terms of safety, and SCADA systems excel in open communication architecture.

3.5.3 ELECTRICAL CONTROL PANEL

Consider a control panel as the equivalent of a human body, where vital organs monitor and control our surroundings. Similarly, a control panel is a metal enclosure housing essential electrical devices responsible for the electrical control and monitoring of a mechanical process.

Let's begin by discussing the enclosure itself, which is a metal box typically made of aluminum or stainless steel. The size of the enclosure varies based on the requirements of the process it serves. Enclosures may have multiple sections, each equipped with an access door. The size of the enclosure is often described by the number of doors it possesses.

To ensure electrical safety, each enclosure is assigned an electrical safety rating by UL (Underwriters Laboratories), a regulatory body overseeing electrical safety. Enclosures also receive an IP rating or NEMA classification, indicating their suitability for indoor or outdoor use, waterproofing or water resistance, resistance to hazardous conditions, dustproofing, or explosion-proofing.

Now, let's explore the internal components of the control panel. The back panel, affixed inside the enclosure, consists of a metal sheet enabling the drilling of mounting holes for various devices. This leads us to DIN rails, standardized metal rails used to mount electrical devices inside the panel.

Another important component is the wiring duct, which facilitates organized and efficient wire routing while minimizing electrical noise between devices.

Moving to the electrical components, one encounters the main circuit breaker, serving as the entry point for power into the control panel to distribute power to all devices within. Typically, this circuit breaker includes a disconnect switch on the panel's exterior to facilitate power shutdown. However, it's crucial to note that the upper side of the circuit breaker may still hold power. The incoming power can vary from 480 volts down to 120 volts [22].

4

Development of the Prototype

4.1 PROBLEM

The customer requested a "Welding and Compression Assembly Line" aimed at producing "Expansion Vessels" or "Expansion Tanks" which comprises various stations such as: transportation by conveyors, welding station, air compressing Station, assembly station , transportation by robot and Their previous line lacked the capability to track the details of each different type of vessel, including product specifications and manufacturing history. Consequently, this deficiency led to several process-related issues:

Product Identification: It was not easy to determine the type of product on the conveyor.

Component Verification: Ensuring all components are present and correctly sorted.

Quality Control: Verifying whether the correct components were being successfully welded together was problematic.

Pressure Requirements: Determining the required pressure for each type of component and product.

4.2 SOLUTION

To address these challenges, two potential solutions were proposed. The first solution involved implementing a comprehensive tracking system along the entire assembly line, coupled with a sophisticated SCADA system for data analysis. However, due to its high cost estimated at approximately 100,000 euros for the SCADA system and an additional 30,000-40,000 euros for implementation the customer deemed it financially unfeasible.

As an alternative, a more cost-effective solution was proposed, utilizing code reader Technology. This solution involved installing seven strategically positioned code readers along the assembly line. These code readers were programmed to scan the codes affixed to the top and bottom of each vessel. Each code contained a unique serial number that corresponded to detailed information about the vessel, including its dimensions, air and water pressure parameters, production specifications, and welding parameters. This alternative, costing approximately 40,000 euros, proved to be more economically viable for the customer.

In operation, after scanning the code, our program extracted the serial number and wrote it to a specific data block (DB). Subsequently, this information was transmitted to the customer's database via S7 communication, allowing for comprehensive data collection and analysis. This implementation significantly enhanced traceability throughout the manufacturing process, enabling the customer to maintain a detailed record of each vessel's journey from production to final testing.

The first objective is to seamlessly integrate the code Reader into the TIA Portal and thoroughly test its various functions. To achieve this, I carefully selected and utilized several essential devices and software components to create a comprehensive and efficient automation system.

These components included the Keyence SR-X300 AI-Powered Code Reader, Dell Precision 3530 Laptop, Siemens PLC S7-1517-3 PN/DP, Simatic ET200SP and I/O Modules, Siemens HMI Panel T900 Comfort, PATLITE LB6-20ILWCBW Signal Tower, PATLITE LR7-02KTN Signal Tower, PATLITE WE-402UB-LAN Signal Tower, Phoenix Contact FL Switch SFN 8TX, Cabur Switch Power Supply CS1024/120-230, Steel Stand, Siemens TIA Portal V17, and AutoIDNetworkNavigator.

In the following sections, more specific details about each of these components will be delved into, elaborating on their functions and contributions to the implementation process.

4.3 COMPONENTS USED IN THE PROTOTYPE

4.3.1 KEYENCE SR-X300 AI-POWERED CODE READER

The Keyence SR-X300 AI-Powered Code Reader (fig. 4.1)[14] is a dependable solution for industrial automation's code reading needs. With its swift scanning, cutting-edge decoding algorithms, and sturdy design, it offers comprehensive features that ensure efficient and trustworthy code scanning. It excels in high-speed scenarios, capturing codes swiftly even in bustling production settings, while supporting a wide array of code symbologies, from popular linear ones like UPC and Code 128 to 2D codes like QR and Data Matrix.

Advanced decoding algorithms empower it to accurately read damaged or low-contrast codes, reducing errors and enhancing overall process efficiency. Its precision in reading comes from precise optics and high-resolution image capture, vital for tasks needing accurate identification. Seamless integration is achieved through support for various communication interfaces, such as Ethernet and USB. The reader's intuitive interface streamlines setup and its rugged design withstands harsh industrial conditions. Additional advanced functionalities, like image capture and diagnostics, enrich its capabilities, promoting system optimization.



Figure 4.1: SR-X300 Reader

4.3.2 PLC S7-1517-3 PN/DP

The CPU 1517-3 PN/DP (fig. 4.2) is a central processing unit (CPU) designed as a standard component for the SIMATIC S7-1500 automation system. It serves as the core of the control and communication processes within the system, providing essential functionalities to ensure efficient and reliable automation.

The CPU 1517-3 PN/DP offers a range of essential features for efficient industrial automation. It comes with 1 MB of work memory for temporary data storage during program execution, alongside a 5 MB load memory to store user programs with specific logic and control instructions. Equipped with 2 PROFINET interfaces, each featuring a 2-port switch, it ensures rapid real-time communication between various automation devices, facilitating seamless data exchange.

Additionally, the CPU includes a PROFIBUS interface for communication with other compatible devices, enabling integration with a diverse array of automation components. The presence of an SD card slot allows for memory expansion, accommodating extra data, backups, or firmware updates. Its USB interface enables convenient connectivity with external devices like programming tools, data storage, and Human-Machine Interfaces (HMIs). Furthermore, the CPU features an MPI/PROFIBUS DP interface, serving as a connection point for programming and HMI devices, promoting effective communication and data exchange between the CPU and engineering workstations or operator panels[24].



Figure 4.2: S7-1517

4.3. COMPONENTS USED IN THE PROTOTYPE

4.3.3 SIMATIC HMI TP900 COMFORT

The SIMATIC HMI TP900 Comfort (fig. 4.3) is a highly capable Comfort Panel designed for Human-Machine Interface (HMI) applications. It features a user-friendly touch operation interface and is equipped with a 9-inch widescreen TFT display capable of displaying 16 million colors. The panel comes with a PROFINET interface and an MPI/PROFIBUS DP interface, facilitating seamless communication and integration with various automation devices and systems. With a generous 12 MB configuration memory, the TP900 Comfort can store and manage complex HMI projects and configurations. It runs on Windows CE 6.0 operating system, providing a stable and reliable platform for HMI applications.

The TP900 Comfort is fully configurable using the WinCC Comfort V11 software, which offers a comprehensive set of tools for designing and customizing HMI applications. This allows engineers and technicians to create intuitive and visually appealing HMI screens, enabling efficient interaction and monitoring of industrial processes.

The combination of advanced features, ample configuration memory, and intuitive software integration makes the SIMATIC HMI TP900 Comfort an ideal solution for a wide range of automation applications, including monitoring, control, and data visualization in industrial environments. Its user-friendly design and powerful capabilities contribute to enhanced productivity and streamlined operations in various industries.



Figure 4.3: HMI Panel

4.3.4 SIMATIC ET200SP

The Simatic ET200SP (fig. 4.4) is a compact and versatile remote I/O system developed by Siemens and widely used in industrial automation applications. It serves as an essential component for connecting various sensors and actuators to the central programmable logic controller (PLC) in a distributed control system. The Simatic ET200SP offers key features that enhance its utility in industrial automation. Its compact design is well-suited for space-limited installations, while its modular configuration permits customization to match specific application needs. The system's high flexibility is evident in its support for diverse communication interfaces, including various fieldbus protocols and Profinet for seamless integration.

As a remote I/O system, it connects to distributed devices, simplifying wiring and promoting efficient data exchange. The scalability of the ET200SP accommodates projects of different sizes, and its robust performance ensures reliability in demanding industrial environments.



Figure 4.4: ET200SP

4.3. COMPONENTS USED IN THE PROTOTYPE

4.3.5 PATLITE WE-402UB-LAN SIGNAL TOWER

The PATLITE WE-402UB-LAN Signal Tower (fig. 4.5)[19] is a versatile and advanced signaling device used in industrial automation applications. It is designed to provide visual and audible indications to monitor the status of machines, processes, and production lines. The signal tower is equipped with various colors and sound options, making it highly effective in conveying critical information to operators and personnel in the manufacturing environment.

The PATLITE WE-402UB-LAN Signal Tower finds diverse applications across industries. It effectively communicates status updates on manufacturing lines, aiding in process tracking. Its role extends to machine monitoring, promptly signaling errors, maintenance needs, or warnings. In process control, the signal tower assists operators in monitoring vital parameters and taking timely actions. Additionally, it enhances material handling efficiency in conveyor systems by indicating material presence or absence.



Figure 4.5: WE-402UB-LAN

4.3.6 PATLITE LR7-02KTN SIGNAL TOWER

The PATLITE LR7-02KTN Signal Tower (fig. 4.6)[19] is a compact and highly efficient signaling device commonly used in industrial automation applications. Its main purpose is to provide clear and visible indications to operators and personnel about the status of machines, processes, and production lines in the manufacturing environment. The signal tower is equipped with multiple color modules and sound options, making it an effective tool for conveying important information and alerts.

The PATLITE LR7-02KTN Signal Tower serves a broad spectrum of industrial applications, including real-time production line monitoring, where it offers live status indications for distinct production stages, streamlining operational oversight. It plays a pivotal role in machine management by displaying machine status, errors, or maintenance needs, promoting swift operator intervention. In automated processes, aids in visualizing critical parameter statuses and highlighting potential deviations, contributing to efficient process control.



Figure 4.6: LR7-02KTN

4.3. COMPONENTS USED IN THE PROTOTYPE

4.3.7 PATLITE LB6-20ILWCBW SIGNAL TOWER

The PATLITE LB6 signal Tower (fig. 4.7)[19] offers three versatile operating modes to accommodate a range of applications. In "Simple Mode," simplifies equipment status visualization by allowing you to segment the lighting area into 1 to 5 tiers and customize the colors and flashing patterns for each tier. The "Level Meter Mode" provides a visual representation of tank level and temperature data, triggering preset lighting patterns based on the tank level. This mode facilitates material replenishment timing and machine status tracking. In "Animation Mode," you can visualize the Takt Time in the manufacturing process by changing preset colors from bottom to top or top to bottom to reflect elapsed time.

Additionally, the LB6 streamlines installation and reduces man-hours by enabling easy wiring with a single IO-Link cable, and it conveniently revert to factory default settings when a replacement cable is connected. This unique and user-friendly setup tool opens up various possibilities for customizable indications.



Figure 4.7: LB6-20ILWCBW

4.3.8 PHOENIX CONTACT FL SWITCH SFN 8TX

The Phoenix Contact FL Switch SFN 8TX (fig. 4.8) is a reliable and efficient industrial Ethernet switch designed to facilitate seamless communication and networking in industrial automation applications. This switch offers various features and benefits that enhance its performance and suitability for demanding industrial environments.

The Phoenix Contact FL Switch SFN 8TX is a robust and high-performance Ethernet switch that delivers reliable networking capabilities for industrial automation systems. It features LED indicators for local diagnostics, facilitating quick status monitoring and efficient troubleshooting. Equipped with RJ45 ports, the switch supports both 10 Mbps and 100 Mbps speeds, ensuring effective data transfer. Its IP20 protection rating enhances safety and safeguards against accidental electrical contact. Designed for DIN rail mounting, the switch offers secure and space-saving installation. Complying with IEEE 802.3 standards, it seamlessly communicates with other Ethernet devices in the network. With compact dimensions, measuring 50 mm in width, 120 mm in height, and 70 mm in depth, this switch is ideal for space-constrained applications. These features collectively underscore the FL Switch SFN 8TX's value as a pivotal component in constructing efficient and resilient industrial networks.



Figure 4.8: FL Switch SFN 8TX

4.3. COMPONENTS USED IN THE PROTOTYPE

4.3.9 CABUR SWITCHING POWER SUPPLY CS1024/120-230

The Cabur Switching Power Supply CS1024/120-230 (fig. 4.9) is a high-quality power supply manufactured by Cabur, a renowned company known for its reliable electrical products.

The CS1024/120-230 model is specifically designed to provide stable and efficient power to various industrial and automation applications. As a power supply, its primary function is to convert input voltage from either 120V or 230V sources into a regulated and suitable output voltage required for powering electrical devices and components.



Figure 4.9: Power Supply CS1024/120-230

4.4 INSTALLATION

4.4.1 HARDWARE INSTALLATION

The experiment began with the setup phase on the test desk (fig. 4.10). The wiring process was carried out meticulously, ensuring that the PLC, ET200SP, Ethernet switch, signal towers, and code readers were properly connected to the power supply. The two SRX-300 readers and signal towers were then mounted on the sturdy stand to provide a stable platform for testing and experimentation.



Figure 4.10: Test Desk

4.4. INSTALLATION

In (fig. 4.11) can be seen a simple scheme block diagram that shows the interaction of the components.

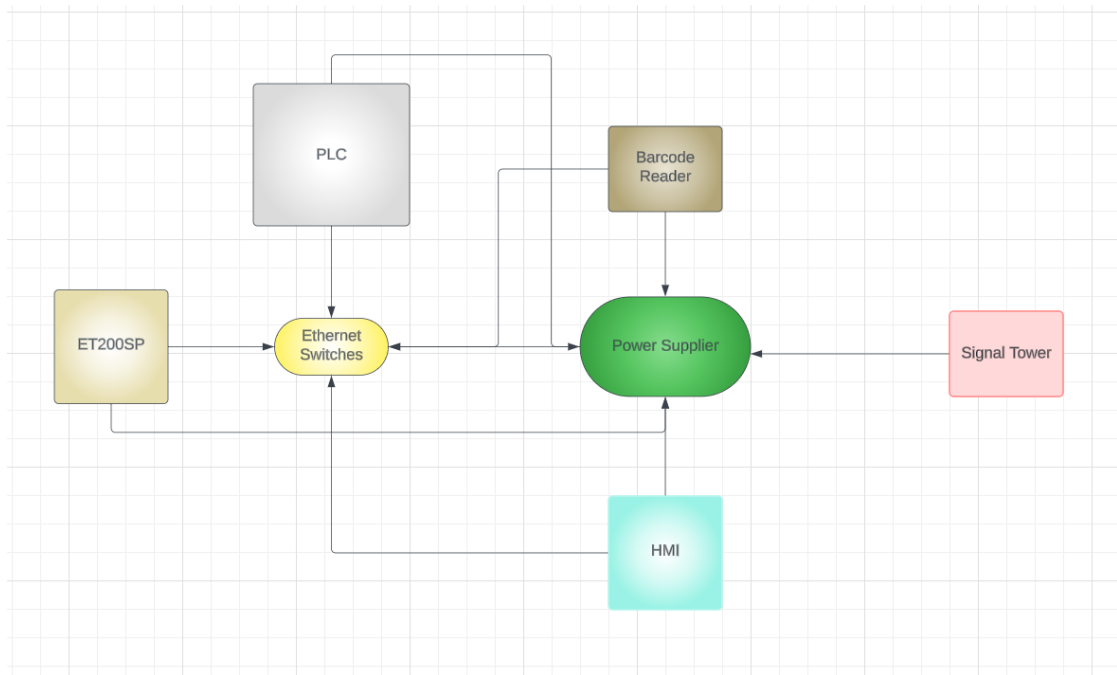


Figure 4.11: Block Scheme

4.4.2 SOFTWARE CONFIGURATION

AutoIDNetworkNavigator Software: Upon wiring the SR-X300 AI-Powered Code Reader, the configuration process commenced using the AutoIDNetworkNavigator Software (fig. 4.12) .

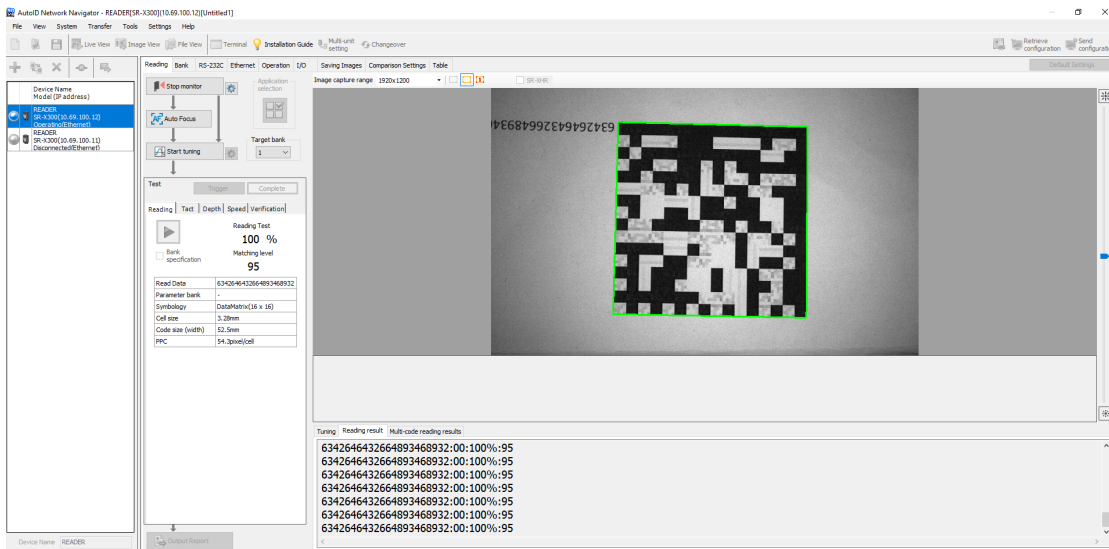


Figure 4.12: AutoIDNetworkNavigator

This software facilitated precise tuning and optimization of various parameters to ensure optimal performance. Below (table 4.1) is a detailed description of the parameters optimized during the configuration process:

	Optimization Criteria
IP Address	Manually set to match the subnet mask of the PC and PLC.
Application Selection	Tailored setups for stationary and moving code readers.
Bar Width and Line Speed	Narrow bar width set to 0.254mm and line speed to 40m/min.
Quality Monitoring	Utilized start monitoring and auto-tuning functions.
Code Definitions	Saved different types of codes in separate banks.
Communication Type	Command input by computer and TCP communication protocol.
Port Configuration	Defined port 9004 for the communication
Configuration Transfer	Transferred configuration from PC to PLC and code reader.
Timing Parameters	Adjusted trigger on delay and trigger off delay for optimal results.

Table 4.1: AutoIDNetworkNavigator Parameters

4.4. INSTALLATION

The IP address was manually configured to facilitate detection of the code reader within the application environment. Furthermore, specific applications were selected based on the intended mounting location of the reader along the production line. For instance, parameters such as bar width and line speed were adjusted to match the requirements of the customer's production line.

During the quality monitoring phase, the start monitoring and auto-tuning functions were utilized to ensure accurate reading of codes. Different types of codes were defined and stored in separate banks within the application to enhance detection accuracy.

For communication between the PC and code reader, the command input by the computer and TCP communication protocol were selected. Additionally, port 9004 was defined to establish a connection between the PC and code reader, facilitating seamless data transfer.

Finally, certain timing parameters, including trigger on delay and trigger off delay, were fine-tuned to optimize the reading behavior of the code reader, ultimately ensuring the best results in terms of accuracy and efficiency.

PATLITE WE-402UB-LAN: For the PATLITE WE-402UB-LAN Signal tower (fig. 4.13), a dedicated website (just need to enter its IP address in the URL tab of the browser) was utilized to configure its settings. Various options, such as tires, buzzers, and other visual indicators, were carefully adjusted to suit the specific requirements of the project.

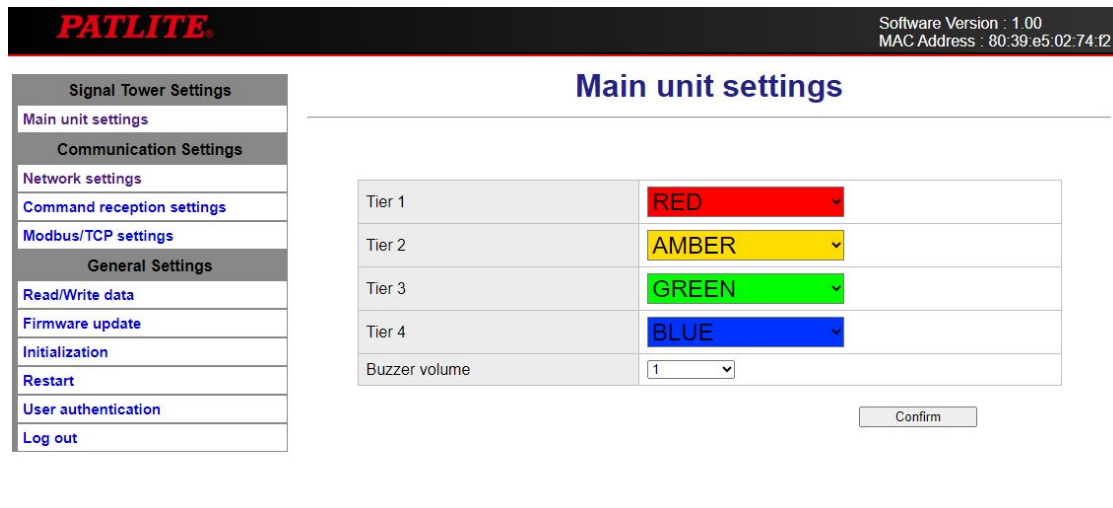


Figure 4.13: Patlite WE-402UB-LAN

TIA Portal: In the TIA Portal software, the key components of the system were defined, encompassing the PLC S7-1517, ET200SP, SR-X300 AI-Powered Code Readers, HMI, and Ethernet switch. These components were logically interconnected to facilitate seamless communication and data exchange between them.

COMMUNICATION PROTOCOLS

For the interconnection of these devices, Profinet and MODBUS TCP/IP and I/O Link Connection protocols were employed.

Profinet was utilized to establish communication between the PLC and other devices in the system. This protocol ensures high-speed data exchange and real-time communication, enabling efficient coordination between the components.

The MODBUS TCP/IP Connection protocol was specifically utilized for transferring data between the Signal Tower and the PLC. This protocol facilitates reliable data transmission over Ethernet networks, allowing for the integration of the Signal Tower into the automation system.

IO/link Protocol was utilized for transferring data between PLC and PATLITE LB6-20ILWCBW.

CONFIGURATION PARAMETERS

The configuration parameters for each communication protocol were carefully set to optimize performance and ensure reliable operation:

For the Profinet Configuration, parameters such as device names, IP addresses, subnet masks, and cycle times were configured to establish a robust Profinet network. These settings facilitate seamless communication between the PLC and other Profinet-compatible devices.

For the MODBUS TCP/IP Configuration, parameters including device addresses, function codes, and data formats were defined to enable efficient data exchange between the Signal Tower and the PLC. These settings ensure accurate transmission and interpretation of data packets between the devices.

TOPOLOGY VIEW

The topology view (fig. 4.14) of the devices provides a visual representation of their interconnection using Profinet and MODBUS TCP/IP protocols. This

4.4. INSTALLATION

view offers insights into the network architecture and the relationships between the various components, facilitating system monitoring and troubleshooting.

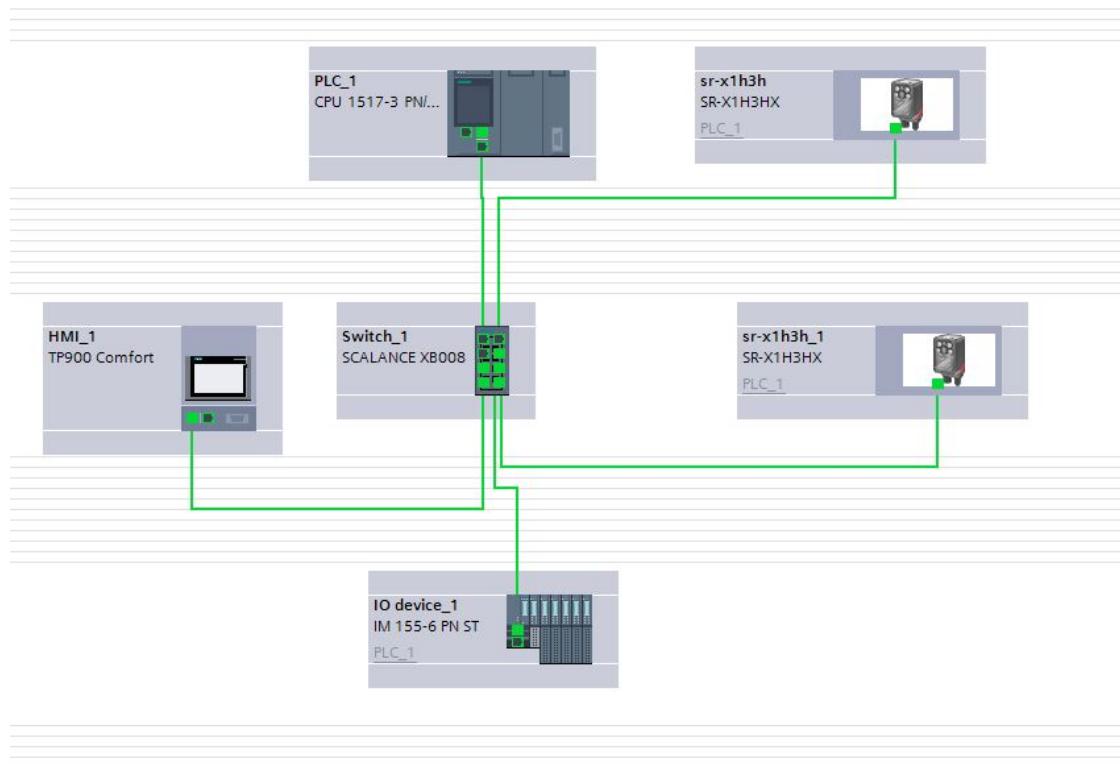


Figure 4.14: Topology View

After that in the Networking view (fig. 4.15) , a subnet is created for the CPU (PLC), and other devices are connected to them. Also, each device was assigned a unique IP address to establish a seamless network.

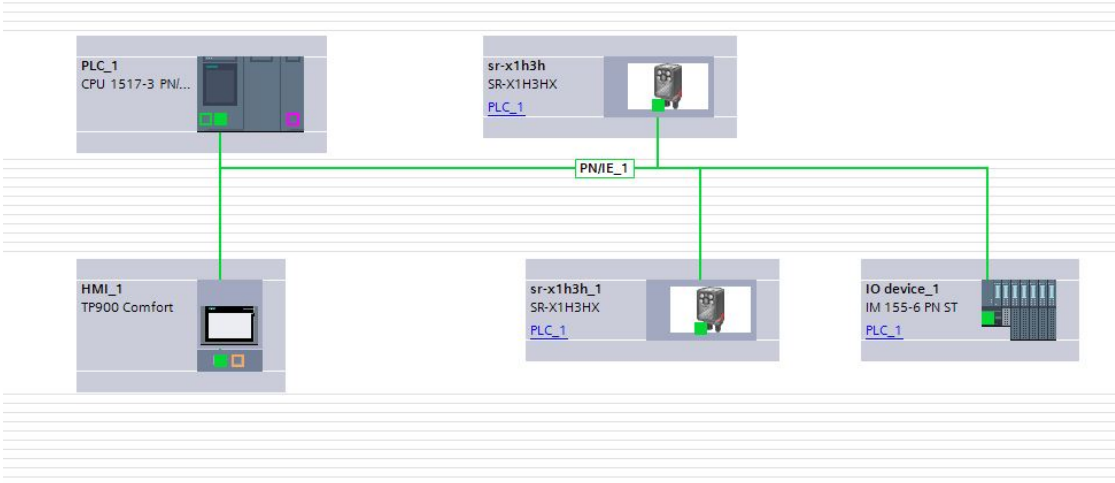


Figure 4.15: Network View

4.4. INSTALLATION

To enable data exchange between devices, the "put/get" function (fig. 4.16) was activated in the PLC. This function allowed the system to write and read data from other devices, contributing to the overall efficiency of the automation system.

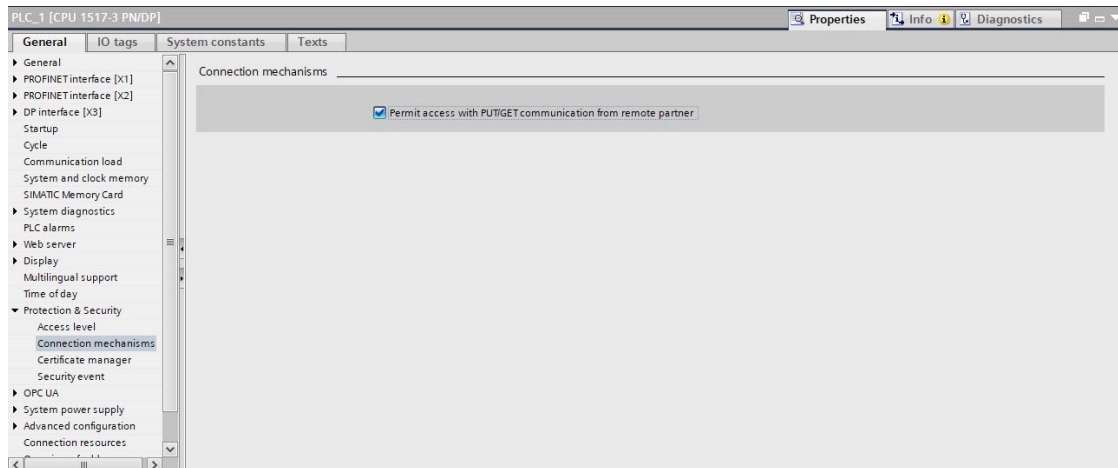


Figure 4.16: Connection Mechanism

The setup of the code readers in the TIA Portal involved defining tags and I/O addresses (fig. 4.17). A comprehensive understanding of the user manual was crucial during this process to ensure accurate configuration.

Device overview									
Module	Rack	Slot	I address	Q address	Type	Article no.	Firmware	Comment	
sr-x1h3h	0	0			SR-X1H3HX	SR-X1H3HX	V1.0.0		
Reader1	0	0 X1			sr-x1h3h				
Handshake and General Error...	0	1	501		Handshake and Ge...				
BUSY Status Bits_1	0	2	502		BUSY Status Bits				
Completion Status Bits_1	0	3	503		Completion Status ...				
Error Status Bits_1	0	4	504		Error Status Bits				
Terminal Status Bits_1	0	5	505		Terminal Status Bits				
Unstable Read Status Bits_1	0	6	506		Unstable Read Stat...				
Matching Level and Total Ev...	0	7	507...514		Matching Level and...				
Operation Result Status_1	0	8	515...534		Operation Result St...				
Read Data 246Byte_1	0	9	535...788		Read Data 246Byte				
	0	10							
	0	11							
	0	12							
Latch and Error Clear Contro...	0	13	501		Latch and Error Cle...				
Operation instruction Contr...	0	14	502		Operation instructi...				
Completion Clear Control Bit...	0	15	503		Completion Clear C...				
Parameter Bank Number_1	0	16	504...505		Parameter Bank Nu...				
User Data 252Byte_1	0	17	506...759		User Data 252Byte				

Figure 4.17: SR-X300 I/O

CHAPTER 4. DEVELOPMENT OF THE PROTOTYPE

Also for the IO device ET200SP (fig. 4.18), I have added all the physical modules in the software and then specified the tags.

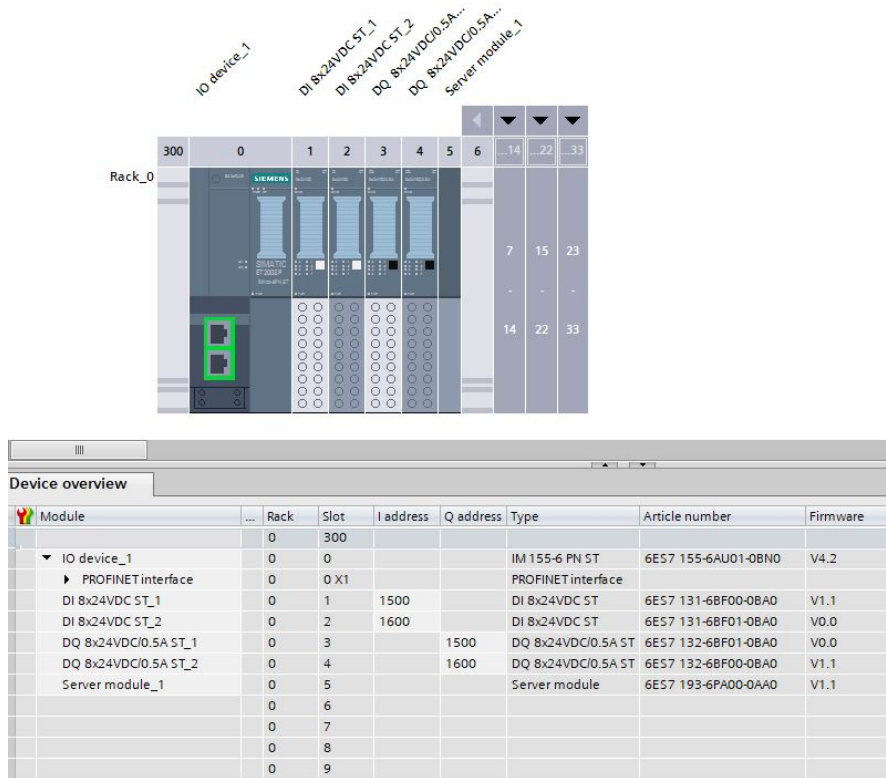


Figure 4.18: ET200SP

4.4. INSTALLATION

4.4.3 PROGRAMMING

User Data Type and Tags: After completing the configuration of the applications, a tag table (fig. 4.20) was defined and all tags were added to it to ensure consistency and standardization in the program. Data types for the reader tags were then defined to maintain uniformity.

Additionally, two User-Defined Types (UDTs) (fig. 4.19) were created: "Interface" and "Interface HMI." The "Interface" UDT was designed to be generic and applicable to various code readers that LEAS might use in the future, enhancing the modularity and reusability of the program. On the other hand, the "Interface HMI" UDT was tailored to suit the tags used on the HMI screen, providing a clear and organized interface for the operator.

UDT_SR-X300_Interface									
	Name	Data type	Default value	Accessible f...	Write...	Visible in ...	Setpoint	Supervision	Comment
1	Cmd	*UDT_SR-X300_Inte...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
2	StartReadingButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Go to Reading Step
3	StopReadingButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
4	ReadClearButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
5	StartTuningButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
6	StopTuningButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
7	TuneClearButton	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
8	HandShake_Enable	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
9	RetrieveResultData	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
10	ResultDataLatch	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Writing to Result Data Device Permitted
11	ErrorClear	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Error Clear
12	ReadRequest	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Start Reading , 1 -> 0 : Stop Reading
13	PresetRequest	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Preset Read Start , 1 -> 0 : Preset Read Stc
14	RegisterPresetDataRequest	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Preset Data Registration
15	TuneRequest	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Start Tuning , 1 -> 0 : Stop Tuning
16	BackupLoadRequest	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Start Backup Load
17	ReadCompleteClear	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Reading Complete Clear
18	PresetCompleteClear	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Preset Reading Complete Clear
19	RegisterPresetDataComple	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Preset Data Registration Complete Clear
20	TuneCompleteClear	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Tuning Complete Clear
21	BackupLoadCompleteClear	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : Backup Load Complete Clear
22	ExternalRequestComplete...	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		0 -> 1 : External Instruction Operation Complete
23	BankNumber_BackupLoa...	USInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Bank Number (1 to 16) , Backup Load File Numbe
24	UserDataSize	USInt	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Preset Data Size
25	UserData	Array[0..251] of Char		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		Preset Data
26	UserData_String	String[252]	"	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
27	Status	*UDT_SR-X300_Inte...		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
28	Ready	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
29	Errors	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
30	Busy	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
31	Reading	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
32	Tune	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
33	Preset	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
34	BackupLoad	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		
35	UnstableReading	Struct		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		

Figure 4.19: UDTs

CHAPTER 4. DEVELOPMENT OF THE PROTOTYPE

SR-X300 TIA PORTAL ▶ PLC_1 [CPU 1517-3 PN/DP] ▶ PLC tags ▶ SR-X300 [105]

SR-X300

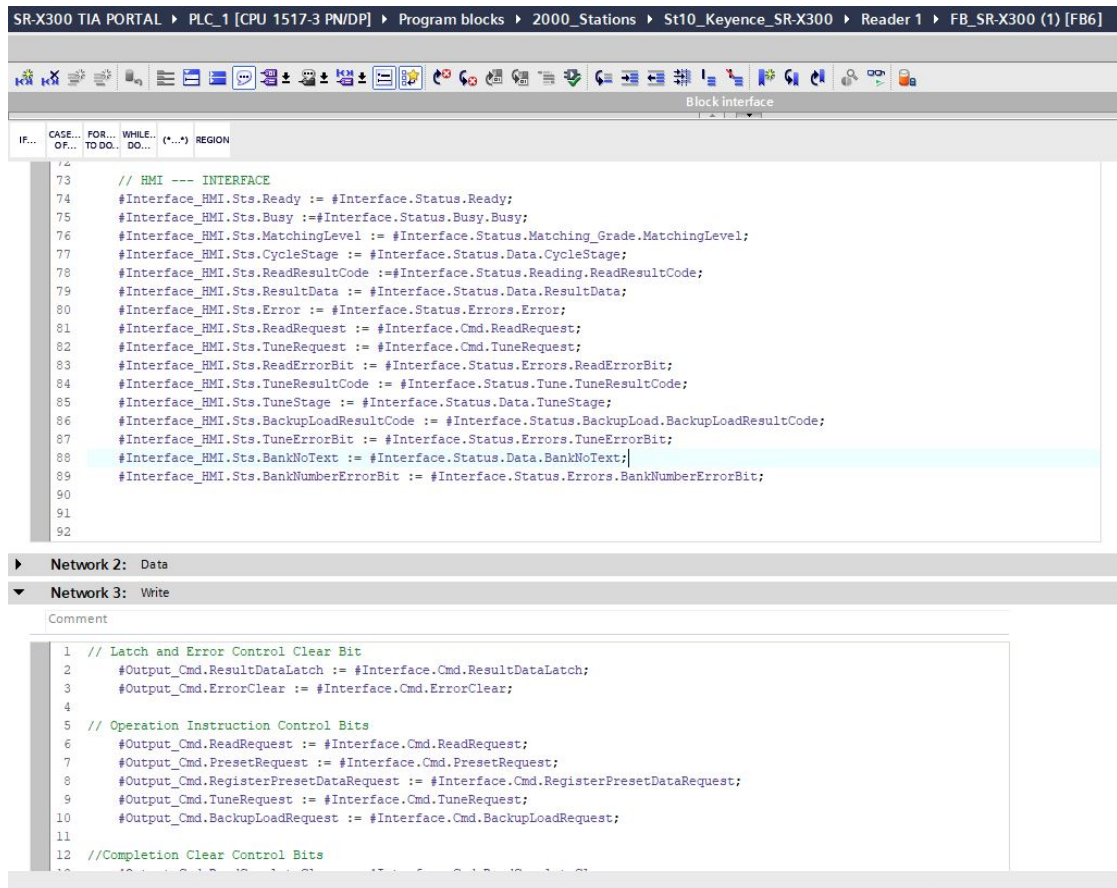
	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Supervision
1	▼ A19_CodeReader_Input	*UDT_SR-X3...	%I501.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2	▼ Status	UDT_SR-X300_...	%I501.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
3	Error	Bool	%I501.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4	ResultDataAvailable	Bool	%I501.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
5	ResultDataStrobe	Bool	%I501.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
6	freebit_1	Bool	%I501.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
7	freebit	Bool	%I501.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
8	freebit_2	Bool	%I501.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
9	BufferOverflowError	Bool	%I501.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
10	GeneralError	Bool	%I501.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
11	Busy	Bool	%I502.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
12	TriggerBusy	Bool	%I502.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
13	LockBusy	Bool	%I502.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
14	ModeBusy	Bool	%I502.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
15	ErrorBusy	Bool	%I502.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
16	freebit_3	Bool	%I502.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
17	Ready	Bool	%I502.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
18	freebit_4	Bool	%I502.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
19	ReadComplete	Bool	%I503.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
20	PresetComplete	Bool	%I503.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
21	RegisterPresetDataComp.	Bool	%I503.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
22	TuneComplete	Bool	%I503.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
23	BackupLoadComplete	Bool	%I503.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
24	freebit_5	Bool	%I503.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
25	freebit_6	Bool	%I503.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
26	ExternalRequestComplete	Bool	%I503.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
27	ReadFailure	Bool	%I504.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
28	PresetFailure	Bool	%I504.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
29	RegisterPresetFailure	Bool	%I504.2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
30	TuneFailure	Bool	%I504.3		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
31	BackupLoadFailure	Bool	%I504.4		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
32	freebit_7	Bool	%I504.5		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
33	freebit_8	Bool	%I504.6		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
34	ExternalRequestFailure	Bool	%I504.7		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
35	In1	Bool	%I505.0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
36	In2	Bool	%I505.1		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Figure 4.20: Tags

4.4. INSTALLATION

Tags and UDTs Connection : To establish a clear connection between tags and User-Defined Types (UDTs), a Function Block (FB) (fig. 4.21) was developed. This FB serves as a bridge, facilitating the seamless integration of tags with both the Interface UDTs and Interface HMI UDTs.

Within this Function Block, the values of tags are written into the Interface UDT, ensuring efficient data management and organization. Additionally, the variables of the UDT HMI, representing user interactions such as button presses on the HMI screen, are also written into the Interface UDT. This approach simplifies the programming process, as the Interface UDT can be consistently used throughout the program, enhancing its structure and understandability for future developers.



```
SR-X300 TIA PORTAL > PLC_1 [CPU 1517-3 PN/DP] > Program blocks > 2000_Stations > St10_Keyence_SR-X300 > Reader 1 > FB_SR-X300 (1) [FB6]
Block interface
IF... CASE... FOR... WHILE... (*...*) REGION
OF... TO DO... DO...

72
73 // HMI --- INTERFACE
74 #Interface_HMI.Sts.Ready := #Interface.Status.Ready;
75 #Interface_HMI.Sts.Busy := #Interface.Status.Busy.Busy;
76 #Interface_HMI.Sts.MatchingLevel := #Interface.Status.Matching_Grade.MatchingLevel;
77 #Interface_HMI.Sts.CycleStage := #Interface.Status.Data.CycleStage;
78 #Interface_HMI.Sts.ReadResultCode := #Interface.Status.Reading.ReadResultCode;
79 #Interface_HMI.Sts.ResultData := #Interface.Status.Data.ResultData;
80 #Interface_HMI.Sts.Error := #Interface.Status.Errors.Error;
81 #Interface_HMI.Sts.ReadRequest := #Interface.Cmd.ReadRequest;
82 #Interface_HMI.Sts.TuneRequest := #Interface.Cmd.TuneRequest;
83 #Interface_HMI.Sts.ReadErrorBit := #Interface.Status.Errors.ReadErrorBit;
84 #Interface_HMI.Sts.TuneResultCode := #Interface.Status.Tune.TuneResultCode;
85 #Interface_HMI.Sts.TuneStage := #Interface.Status.Data.TuneStage;
86 #Interface_HMI.Sts.BackupLoadResultCode := #Interface.Status.BackupLoad.BackupLoadResultCode;
87 #Interface_HMI.Sts.TuneErrorBit := #Interface.Status.Errors.TuneErrorBit;
88 #Interface_HMI.Sts.BankNoText := #Interface.Status.Data.BankNoText;
89 #Interface_HMI.Sts.BankNumberErrorBit := #Interface.Status.Errors.BankNumberErrorBit;
90
91
92

Network 2: Data
Network 3: Write
Comment
1 // Latch and Error Control Clear Bit
2 #Output_Cmd.ResultDataLatch := #Interface.Cmd.ResultDataLatch;
3 #Output_Cmd.ErrorClear := #Interface.Cmd.ErrorClear;
4
5 // Operation Instruction Control Bits
6 #Output_Cmd.ReadRequest := #Interface.Cmd.ReadRequest;
7 #Output_Cmd.PresetRequest := #Interface.Cmd.PresetRequest;
8 #Output_Cmd.RegisterPresetDataRequest := #Interface.Cmd.RegisterPresetDataRequest;
9 #Output_Cmd.TuneRequest := #Interface.Cmd.TuneRequest;
10 #Output_Cmd.BackupLoadRequest := #Interface.Cmd.BackupLoadRequest;
11
12 //Completion Clear Control Bits
13
```

Figure 4.21: Tags and UDTs Connection

Cycle Manager Function (FC) : To effectively manage the operational steps of the reader and have a proper cycle, a Function (FC) called Cycle Manager (fig. 4.22) was introduced. The sequence of steps was carefully arranged to ensure smooth transitions between each phase. The FC was then integrated into another Function Block called Cycle, further enhancing the readability and organization of the program.

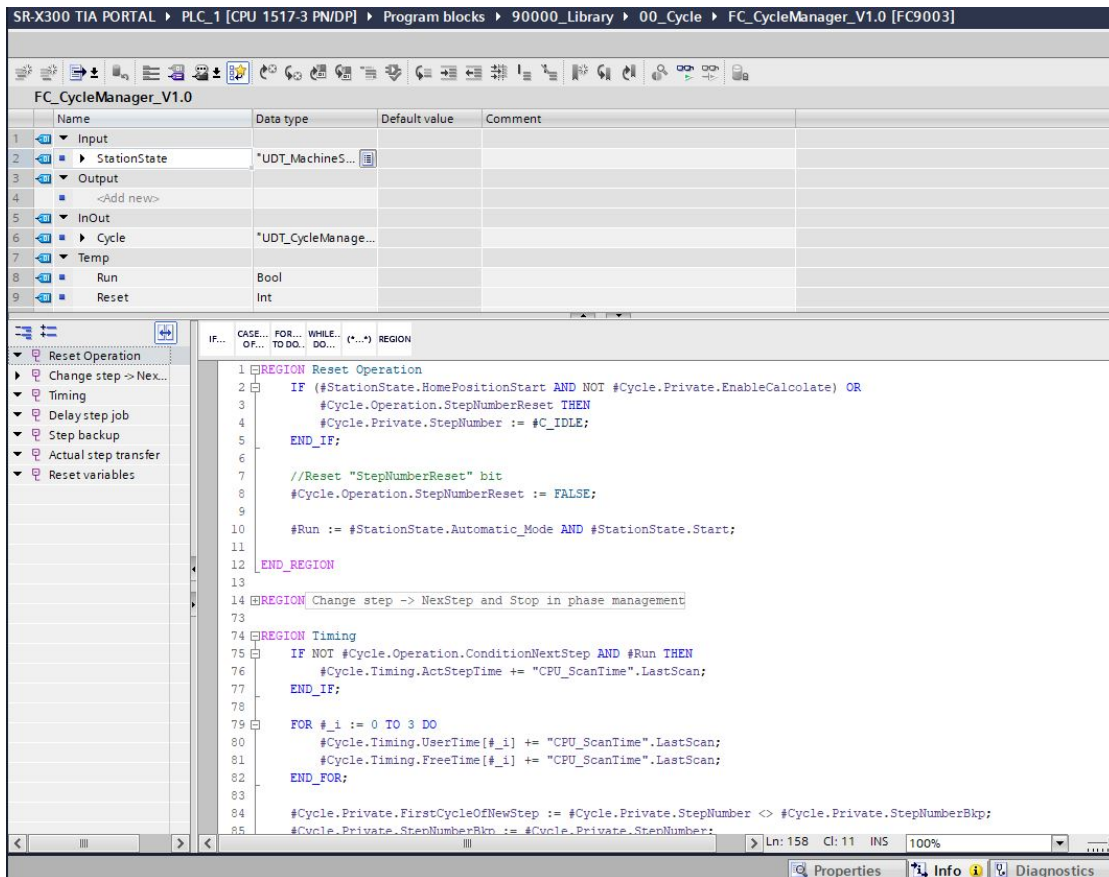


Figure 4.22: Cycle Manager FC

4.4. INSTALLATION

Auto Cycle: In order to provide the customer with a fully automatic line, an Auto Cycle Function Block was required. This FB was designed to operate sequentially, progressing from one step to the next according to the predefined cycle manager. As each step concluded within a specified time frame, the subsequent step would initiate automatically. The initial step, known as the IDLE step (fig. 4.23), indicated that the device was in standby mode and not engaged in any activities.

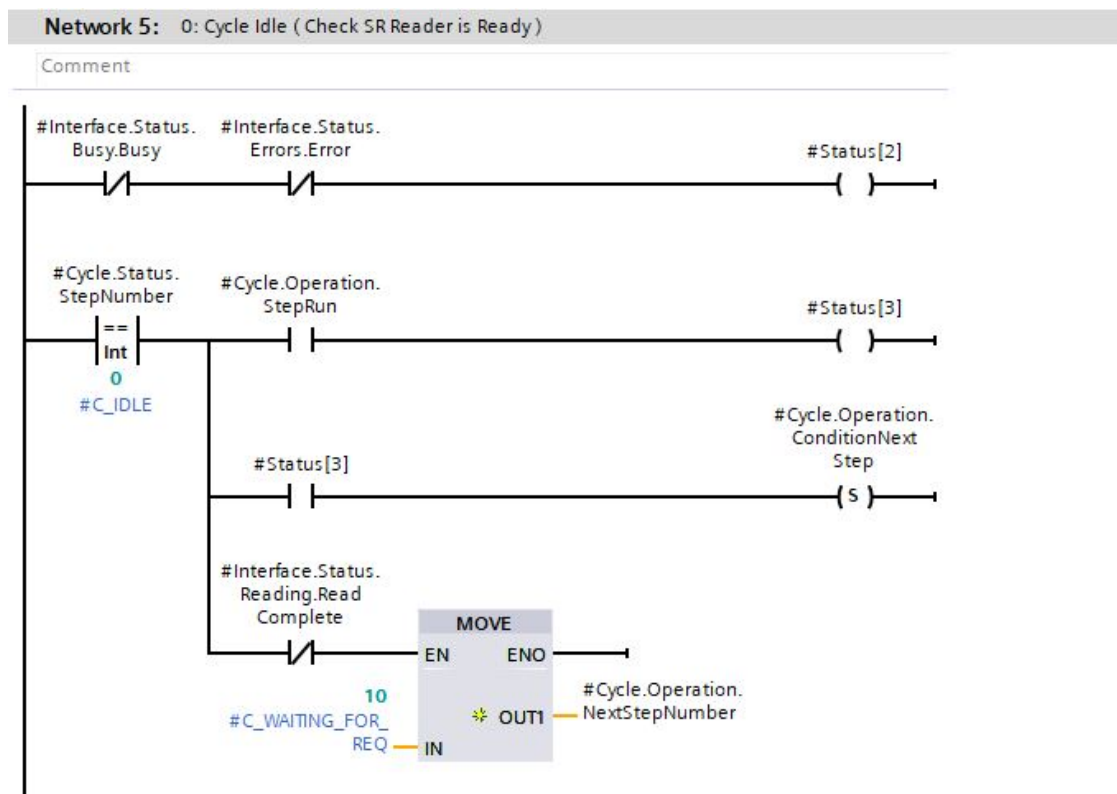


Figure 4.23: IDLE Step

Upon activation of the run button, the cycle progresses to the "WAITING FOR REQ" step (fig. 4.24), awaiting the operator's initiation to commence reading. Subsequently, upon pressing the start reading button, the cycle advances to the "READING" step, wherein the code reader actively scans for codes.

In detail, when the code Reader is in the IDLE Step, and the operator activates the "Automatic run" on the HMI screen, the "Cycle.Operation.ConditionnextStep" becomes energized, indicating the condition for transitioning to the next step of the cycle, which is "Waiting For Request." At this point, the code reader is prepared and awaits the operator's action to push the "Start Reading" button on the HMI. In the future, a memory bit can be defined to automatically energize the "Start Reading" button when a proximity switch sensor detects the approach of a product to the code reader. This automation will prompt the code Reader to proceed to the subsequent step, which is "Reading," (fig. 4.25) initiating the scanning of codes passing along the conveyor.

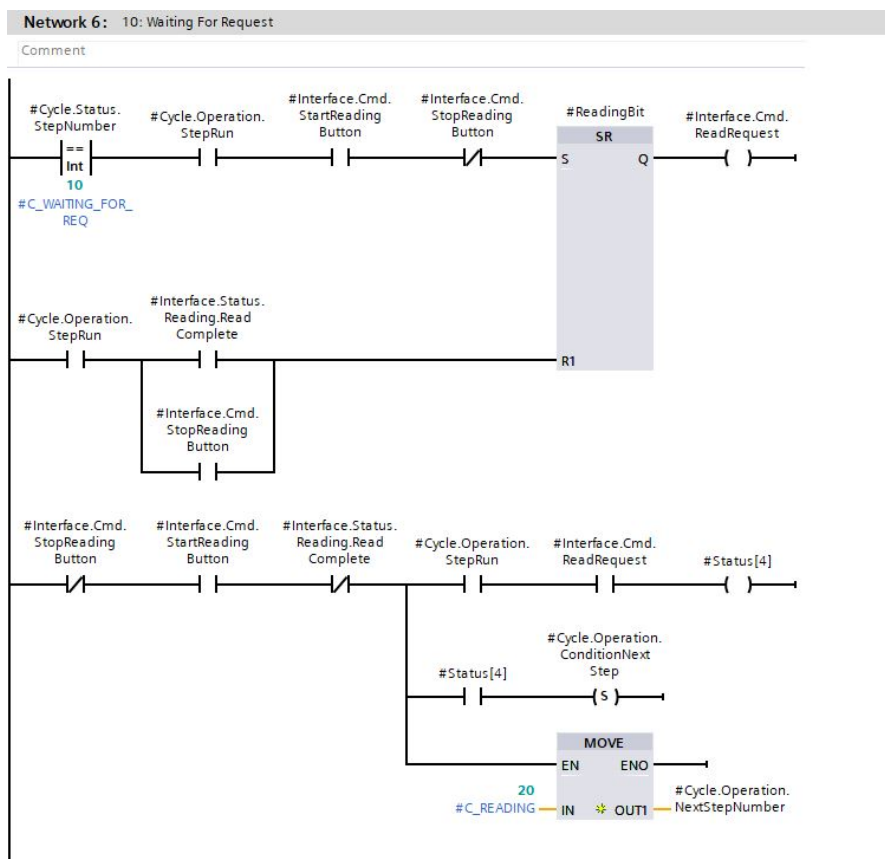


Figure 4.24: Waiting for Request Step

4.4. INSTALLATION

In this step, the code reader starts to read the code, and based on the reading result, two possible outcomes were defined, "Reading Complete" (fig. 4.26) or "Reading Error" (fig. 4.29).

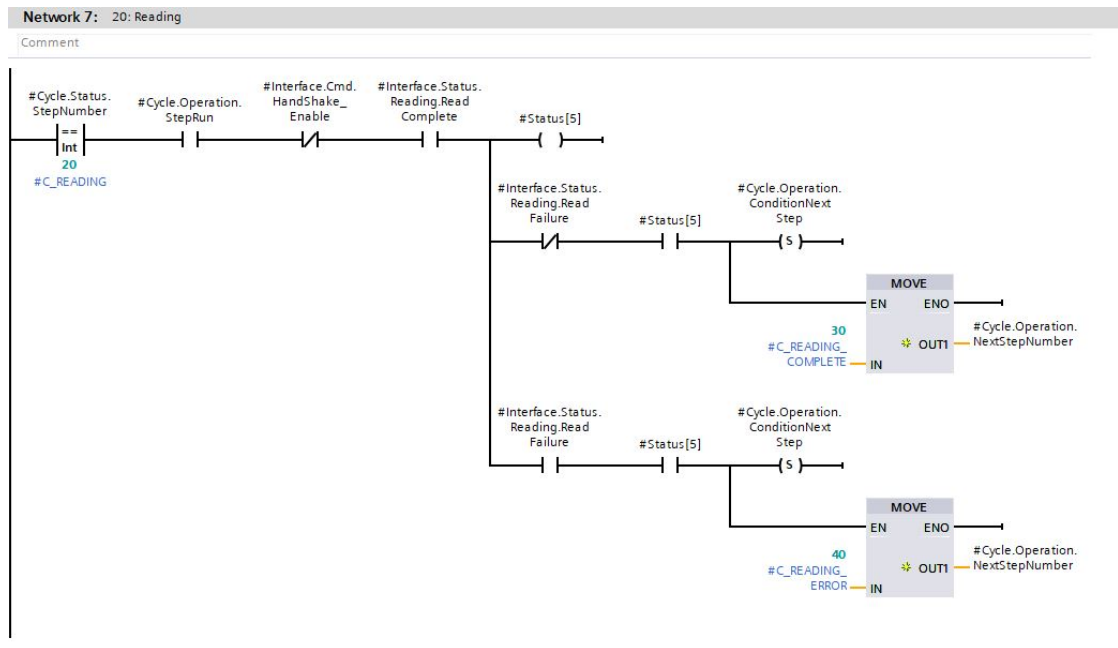


Figure 4.25: Reading Step

If the reader successfully detects a code, it advances to the READING COMPLETE step.

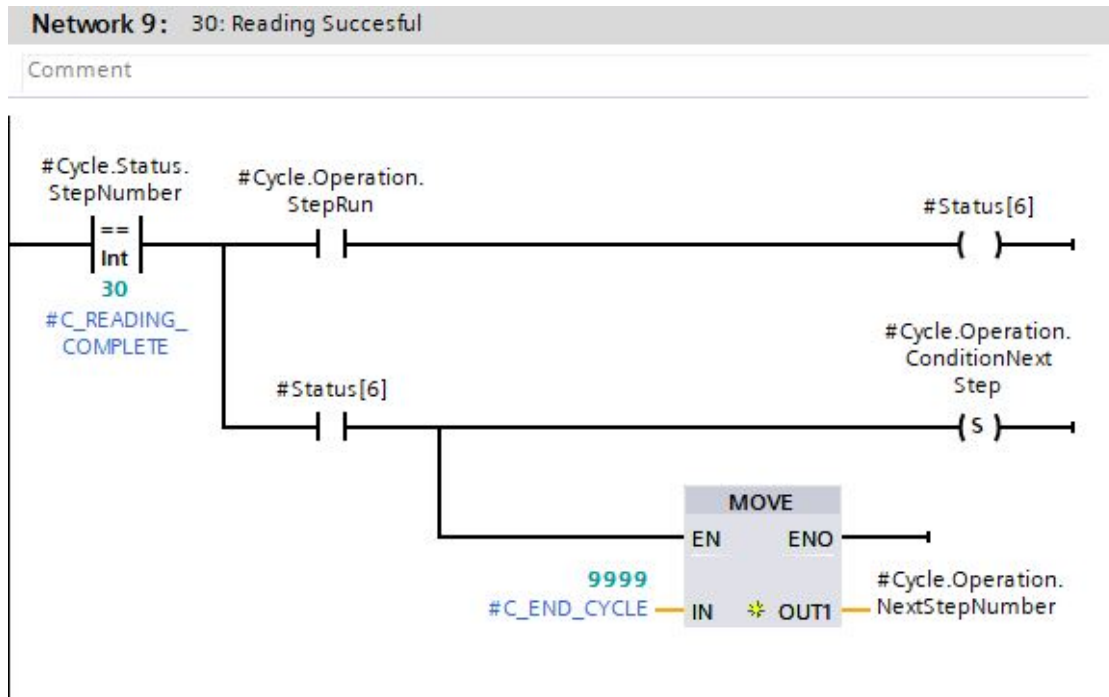


Figure 4.26: Reading Complete Step

After a successful reading, the code (result data) is saved in our ResultData memory (fig. 4.27). The task at this point was to convert this data from an array of characters to a string.

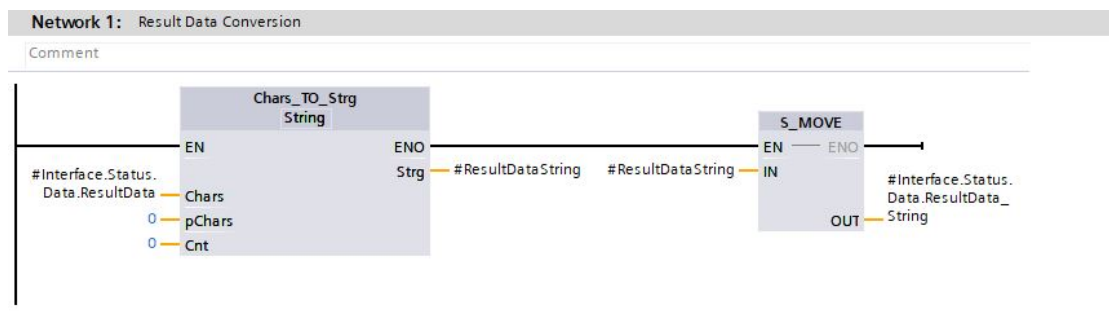


Figure 4.27: Result Data Conversion

4.4. INSTALLATION

Following the extraction of the code, an analysis was conducted to discern product specifications such as length, height, panel number, and serial number (fig. 4.28). It is envisaged that these specifications will be adjusted as per project requirements upon integration into the main program at a later stage (Air pressure , Welding number and ...).

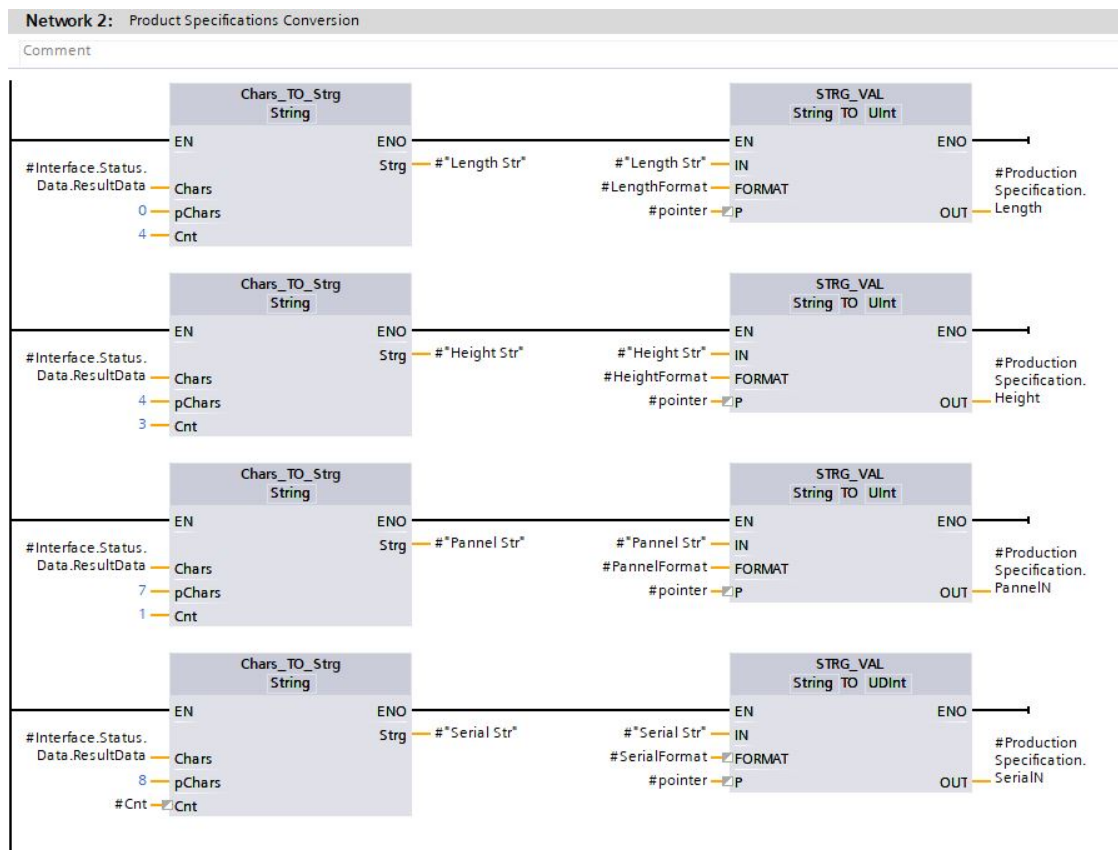


Figure 4.28: Result Extraction

In the case of a reading error, the cycle proceeded to the READING ERROR step (fig. 4.29), where an error code and message were shown on the HMI screen. To resolve the error, the operator could simply press the reset button, clearing the error code and message, and making the reader ready for the next scanning process.

The experiment also accounted for potential errors and malfunctions that could occur during the reader's operation. Error handling was integrated into the cycle, covering scenarios such as Reading Failure, Error Busy, General Error, Buffer Overflow Error, and more.

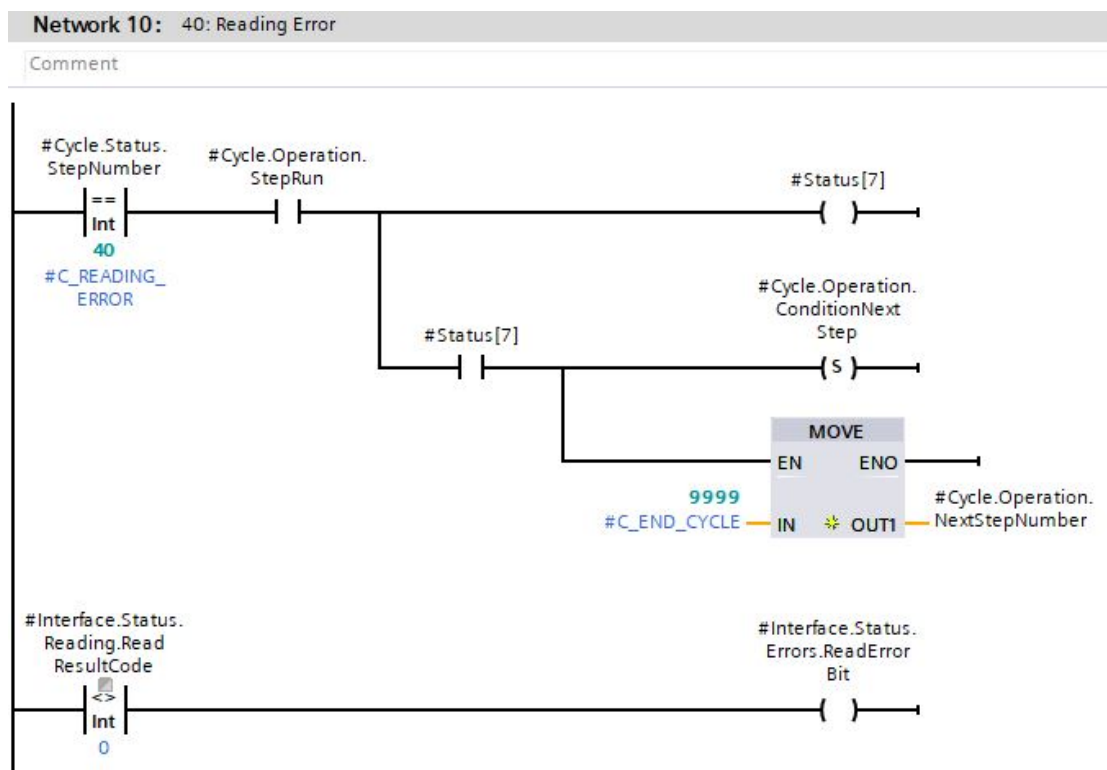


Figure 4.29: Reading Error

4.4. INSTALLATION

After the reading steps, which include successful readings or errors, I took an approach to help operators understand what's happening with SCL language (fig. 4.30).

This code was designed to show clear information on the Human-Machine Interface (HMI). Its main goal was to let the operator know if the reading was successful or if there was a problem. This way, the operator could make decisions faster and smarter.

```
Network 11: Cycle Stage and Errors
Comment
1 IF #Interface.Cmd.BankNumber_BackupLoadFileNumber>16 OR #Interface.Cmd.BankNumber_BackupLoadFileNumber <11 THEN
2 ;
3 #Interface.Status.Data.BankNoText := ' Invalid Input ! Enter a Number (11-16)';
4
5
6 END_IF;
7
8
9 IF NOT #Interface.Cmd.StartTuningButton AND NOT #Interface.Cmd.TuneRequest AND NOT #Interface.Status.Errors.TuneErrorBit THEN
10 ;
11 #Interface.Status.Data.TuneStage := 'WAITING FOR REQ';
12
13
14
15 ELSIF #Interface.Cmd.TuneRequest AND NOT #Interface.Status.Tune.TuneComplete THEN
16 ;
17 #Interface.Status.Data.TuneStage := 'TUNING';
18
19
20
21 ELSIF #Interface.Status.Errors.TuneErrorBit THEN
22 ;
23 #Interface.Status.Data.TuneStage := 'TUNING ERROR';
24
25
26 ELSIF #Interface.Status.Tune.TuneComplete THEN
27 ;
28 #Interface.Status.Data.TuneStage := 'TUNING COMPLETE';
29
30 END_IF;
31
```

Figure 4.30: Cycle Steps on HMI

4.4.4 SIGNAL TOWERS INTEGRATION

The next step of the experiment involved choosing the most suitable signal tower for the application. To make an informed decision, the user manuals of the three options, PATLITE LR7-02KTN, PATLITE WE-402UB-LAN, and PATLITE LB6-20ILWCBW, were thoroughly studied.

The configuration of the PATLITE LR7-02KTN was straightforward. It didn't require any special requirements. It was simply wired to the power supplier and the ET200SP, and its tags were set in TIAPortal. It could be easily activated by referring to its tags in the program.

The configuration of the second signal tower proved to be much more challenging. To send data from the PLC to the WE-402UB-LAN Signal Tower, a new Function Block was created, utilizing the Modbus client function (fig. 4.31)(fig. 4.32).

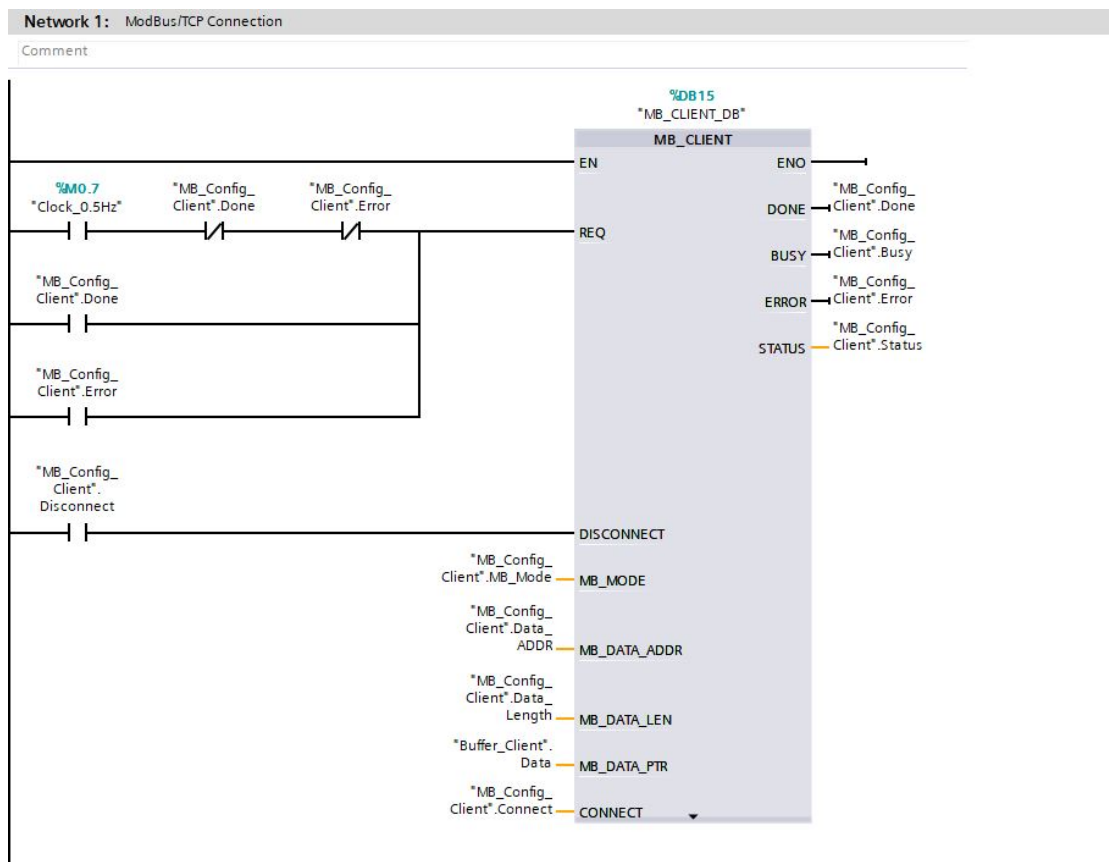


Figure 4.31: Modbus Block

4.4. INSTALLATION

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervision
1	▼ Static			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	Status	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	Error	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	Busy	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	Done	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
6	▼ Connect	TCON_IP_v4		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7	Interfaceld	HW_ANY	64	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8	ID	CONN_OUC	16#1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
9	ConnectionType	Byte	16#B	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
10	ActiveEstablished	Bool	1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
11	▼ RemoteAddress	IP_V4		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
12	▶ ADDR	Array[1..4] of Byte		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
13	RemotePort	UInt	502	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
14	LocalPort	UInt	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
15	Data_Length	UInt	6	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
16	Data_ADDR	UDInt	40001	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
17	MB_Mode	USInt	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
18	Disconnect	Bool	false	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Figure 4.32: Modbus Configuration

The data was transmitted in the form of an Array of Words(fig. 4.33), with each word corresponding to different Register Addresses. The proper configuration of these Register Addresses was vital to achieve the desired outcomes. For instance, changing the word corresponding to the topmost light of the signal tower to 16#0101 would activate the red light, while other combinations would trigger different visual indications. Similarly, altering the word corresponding to the speed of the flashing light allowed the system to control the light’s blinking rate.

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervision
1	▼ Static			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	▼ Data	Array[1..6] o...		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
3	Data[1]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
4	Data[2]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5	Data[3]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
6	Data[4]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
7	Data[5]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
8	Data[6]	Word	16#0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Figure 4.33: Modbus DataBlock

The configuration of the third signal tower, the PATLITE LB6-20ILWCBW, was the most complicated. This was primarily because its user manual did not refer to the correct memory byte. To configure it, it was necessary to use the I/O link protocol. This required the use of an I/O Link Master (fig. 4.34), which was accomplished by adding an I/O link module to the ET200SP.



Figure 4.34: I/O link Master

After wiring, the configuration of the I/O link device in TIAPortal (fig. 4.35) was necessary. The first step was to download the IODD file of the signal tower and import it into S7-PCT. This made it possible to configure the IO link on the port.

4.4. INSTALLATION

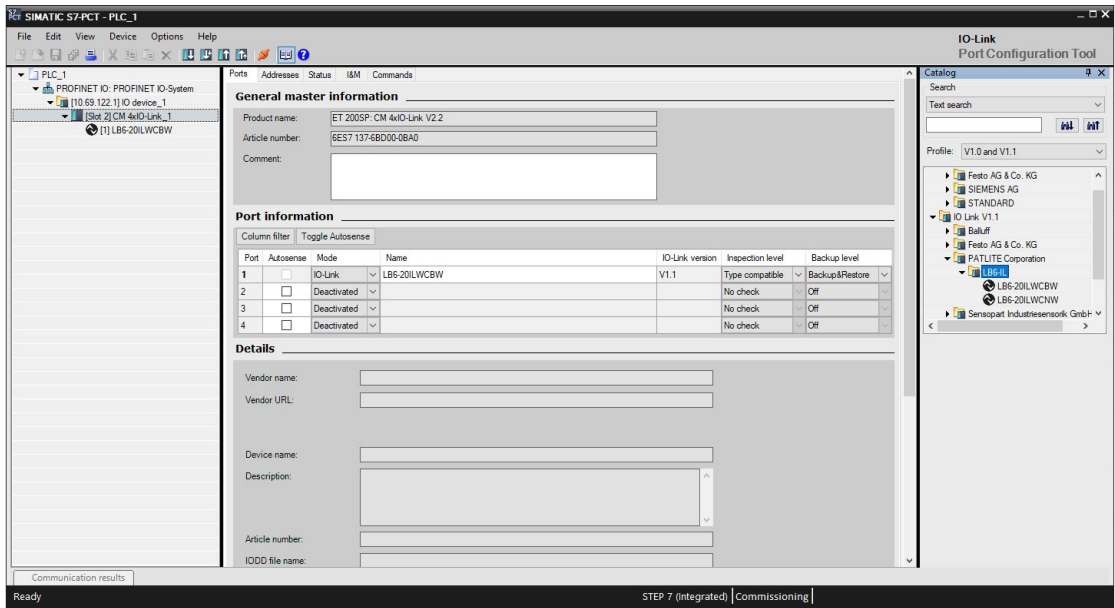


Figure 4.35: I/O link Setup

Then, it was necessary to set up the desired settings of the signal tower (fig. 4.36), such as the mode, the sound, and level of the buzzer, the colors, and so on.

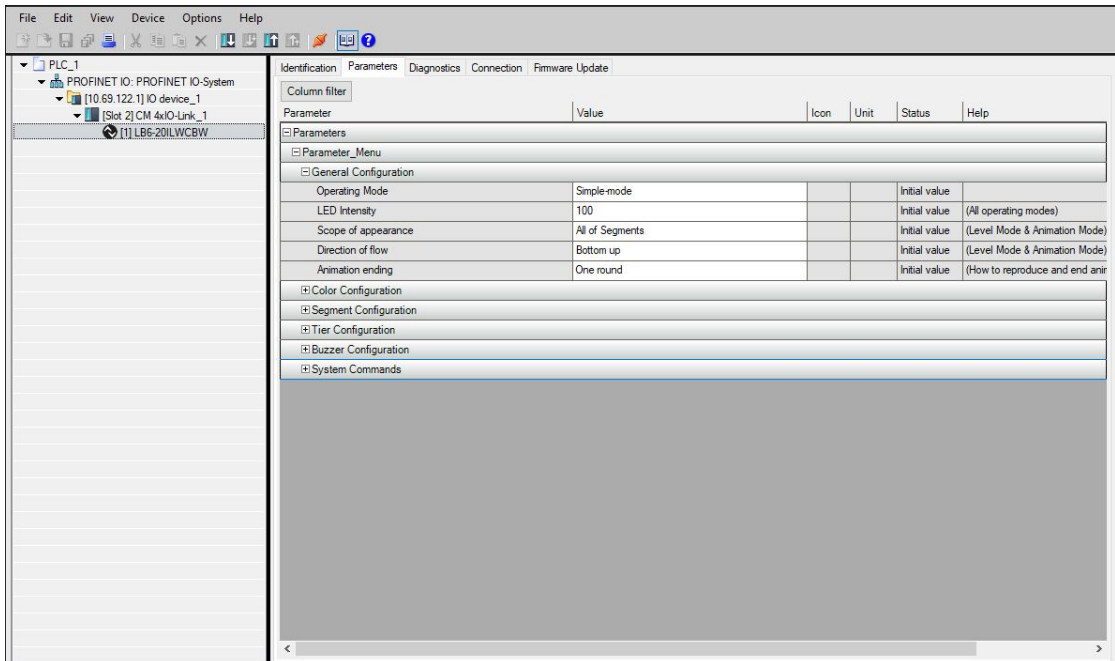


Figure 4.36: Signal Tower Setup

After that, was the time for sending data. In the following figures can be seen the the Data Transmission in three different modes of signal tower (fig. 4.37)(fig. 4.38)(fig. 4.39).

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Byte 0	User Preference Select		Buzzer ON/OFF	LED Tier 5 ON/OFF	LED Tier 4 ON/OFF	LED Tier 3 ON/OFF	LED Tier 2 ON/OFF	LED Tier 1 ON/OFF
Byte 1	Not used							

Figure 4.37: Data transmission in Simple Mode

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Byte 0	User Preference Select		Buzzer ON/OFF	Not used				
Byte 1	Not used	Level value* 0x00 - 0x64 (0 - 100)						

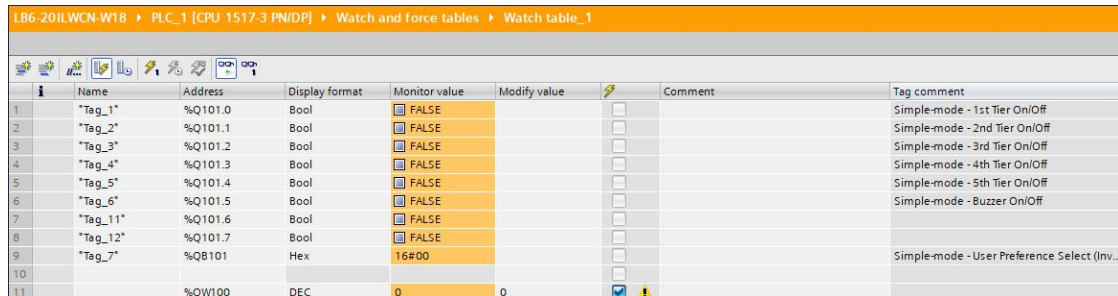
Figure 4.38: Data transmission in Level Mode

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Byte 0	User Preference Select		Buzzer ON/OFF	Not used			Animation Reset	Animation Enable
Byte 1	Not used						Animation Speed	

Figure 4.39: Process data transmission in Animation Mode

4.4. INSTALLATION

For instance, in the Simple Mode, a tag was defined as an integer. The process was completed by converting the binary code to decimal (fig. 4.40). This method was used for process data transmission in both Simple Mode and Level Mode.



The screenshot shows a PLC Watch table with the following data:

#	Name	Address	Display format	Monitor value	Modify value	Comment	Tag comment
1	"Tag_1"	%Q101.0	Bool	<input type="checkbox"/> FALSE			Simple-mode - 1st Tier On/Off
2	"Tag_2"	%Q101.1	Bool	<input type="checkbox"/> FALSE			Simple-mode - 2nd Tier On/Off
3	"Tag_3"	%Q101.2	Bool	<input type="checkbox"/> FALSE			Simple-mode - 3rd Tier On/Off
4	"Tag_4"	%Q101.3	Bool	<input type="checkbox"/> FALSE			Simple-mode - 4th Tier On/Off
5	"Tag_5"	%Q101.4	Bool	<input type="checkbox"/> FALSE			Simple-mode - 5th Tier On/Off
6	"Tag_6"	%Q101.5	Bool	<input type="checkbox"/> FALSE			Simple-mode - Buzzer On/Off
7	"Tag_11"	%Q101.6	Bool	<input type="checkbox"/> FALSE			
8	"Tag_12"	%Q101.7	Bool	<input type="checkbox"/> FALSE			
9	"Tag_7"	%QB101	Hex	16#00			Simple-mode - User Preference Select (Inv...
10							
11		%QW100	DEC	0	0	<input checked="" type="checkbox"/>	

Figure 4.40: Data Transmission DB

In the images that follow (fig. 4.41), you will see various modes and colors of the Signal Tower in the Simple Mode.

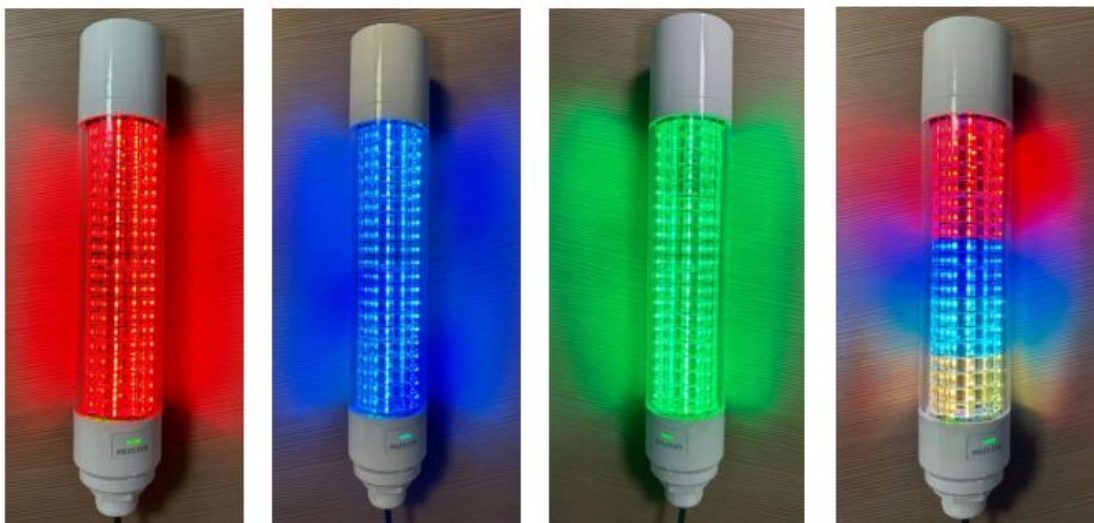


Figure 4.41: Color Variety

After successfully configuring the signal towers, they were included in the cycle for a comparative analysis. All signal towers demonstrated a remarkable response speed, each completing its tasks in less than 10 milliseconds. However, when multitasking scenarios were considered, the LR7-02KTN showed a slightly faster response. While the LR7-02KTN had the advantage of simple configuration, the WE-402UB-LAN and LB6-20ILWCBW offered support for many colors and various models, while the LR7-02KTN had only 4 colors. Taking into

account the potential future requirements and versatility of the machinery line, the LB6-20ILWCBW Signal Tower has been selected, due to its rich color options, variety of modes, buzzers, and other features.

After that, I successfully integrated the LB6-20ILWCBW signal tower and SR-X300 reader into the system to visually display the device's operational status through different colored lights.

The overall status of the reader was visually represented by the signal tower lights. A green light indicated the Ready state, while a yellow light indicated the Busy mode when the reader was performing a task. The Red light indicated an Error state.

Codes Samples

In the following figure (fig. 4.42) , various types of codes that have already been generated for testing the program and the code can be observed :

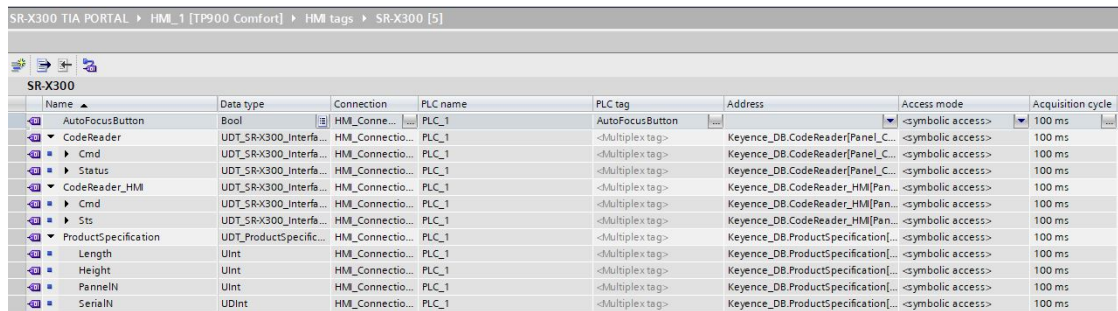


Figure 4.42: Code Samples for testing

4.4. INSTALLATION

4.4.5 HMI DEVELOPMENT

The development of the HMI Screen was the subsequent task. This involved defining and designing all the buttons, indicators, and essential details for the operator interface. The INTERFACE HMI data types, previously linked to the tags (fig. 4.43), were effectively incorporated into the HMI screen.



Name	Data type	Connection	PLC name	PLC tag	Address	Access mode	Acquisition cycle
AutoFocusButton	Bool	HMI_Conne...	PLC_1	AutoFocusButton		<symbolic access>	100 ms
CodeReader	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader[Panel_C...	<symbolic access>	100 ms
Cmd	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader[Panel_C...	<symbolic access>	100 ms
Status	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader[Panel_C...	<symbolic access>	100 ms
CodeReader_HMI	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader_HMI[Pan...	<symbolic access>	100 ms
Cmd	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader_HMI[Pan...	<symbolic access>	100 ms
Sts	UDT_SRX300_Interfa...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.CodeReader_HMI[Pan...	<symbolic access>	100 ms
ProductSpecification	UDT_ProductSpecifc...	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.ProductSpecification[...	<symbolic access>	100 ms
Length	UInt	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.ProductSpecification[...	<symbolic access>	100 ms
Height	UInt	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.ProductSpecification[...	<symbolic access>	100 ms
PannelN	UInt	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.ProductSpecification[...	<symbolic access>	100 ms
SerialN	UDInt	HMI_Conne...	PLC_1	<Multiplex tag>	Keyence_DB.ProductSpecification[...	<symbolic access>	100 ms

Figure 4.43: HMI Tags

The screen provided a user-friendly experience(fig. 4.44), allowing the operator to interact with the automation system seamlessly. For example, by pressing the start reading button on the HMI, the reader begins scanning for codes(fig. 4.45). Upon successful reading, the screen displayed the "Reading Complete" (fig. 4.46) message and the scanned code. In the event of an error, such as error code 201, the HMI indicated the "Reading Error," (fig. 4.47) enabling the operator to consult the user manual for troubleshooting and resolution.

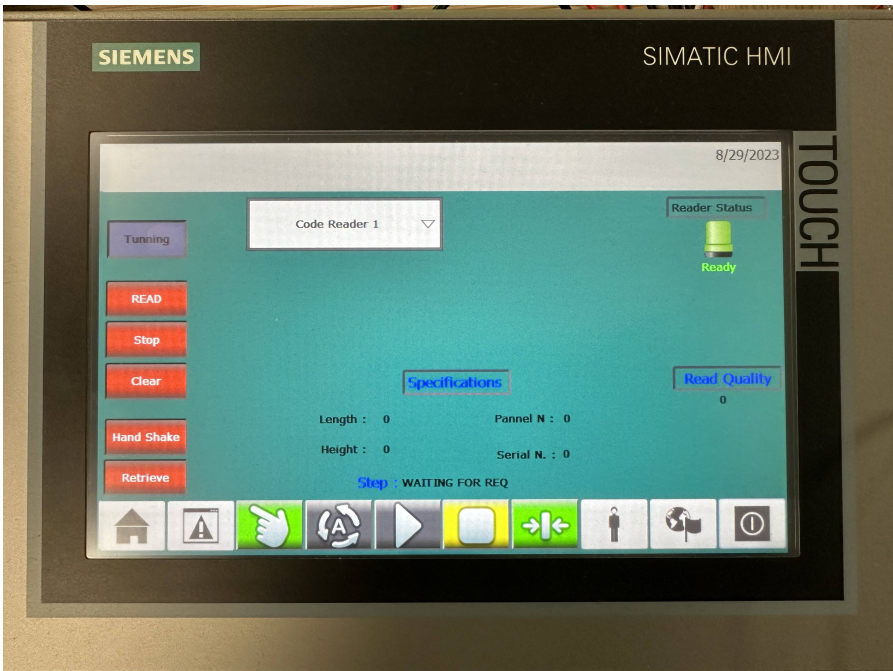


Figure 4.44: HMI- Waiting for Request

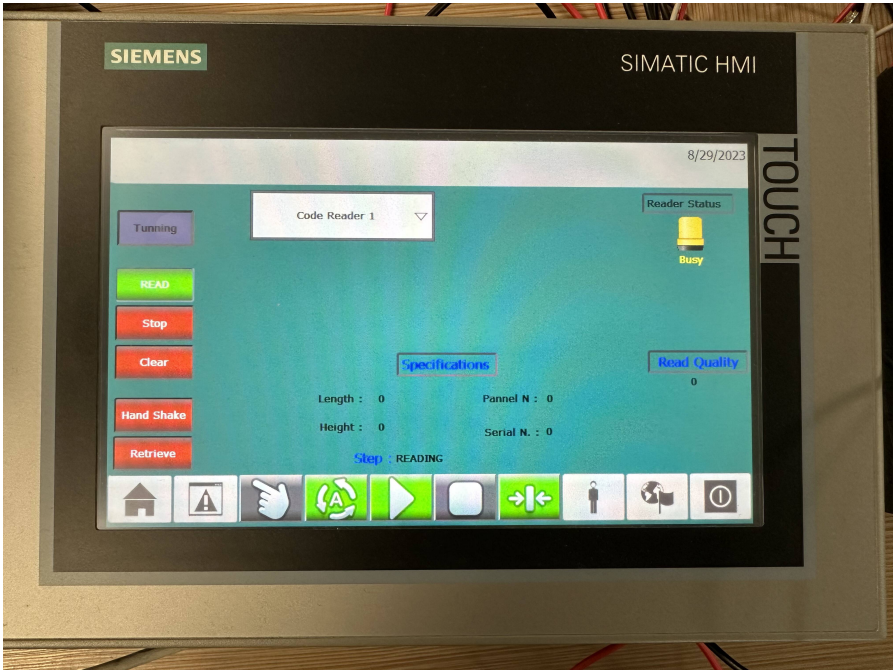


Figure 4.45: HMI- Reading

4.4. INSTALLATION



Figure 4.46: HMI- Read Complete

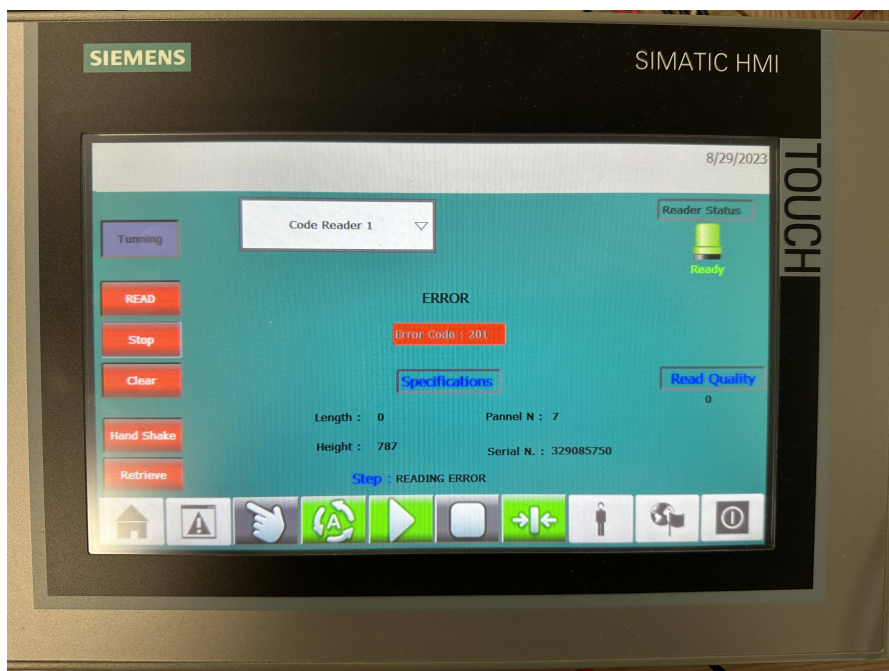


Figure 4.47: HMI- Read Error

A notable challenge encountered during the experiment was, that if a code or a product with a different size appeared on the conveyor, and its configuration was not present in the AutoIdnetwork software, the reader would not be able to read it.

To address this limitation, the AutoTune button has been implemented on the HMI (fig. 4.48). This feature enabled the operator to handle codes not defined in the available banks. When an unidentified code was detected, the operator could select a bank number from 11 to 16 (fig. 4.49) (since banks 1 to 10 were already in use). By pressing the "Start Tuning" button, the reader would tune and configure the new code to be recognized. After successfully completing the tuning process (fig. 4.50), the code type will be automatically saved into the code Reader's memory bank, allowing it to detect this type of code in the future. However, if the tuning encounters an error due to the reader's instability or unclear codes, an error message will appear on the HMI screen (fig. 4.51).

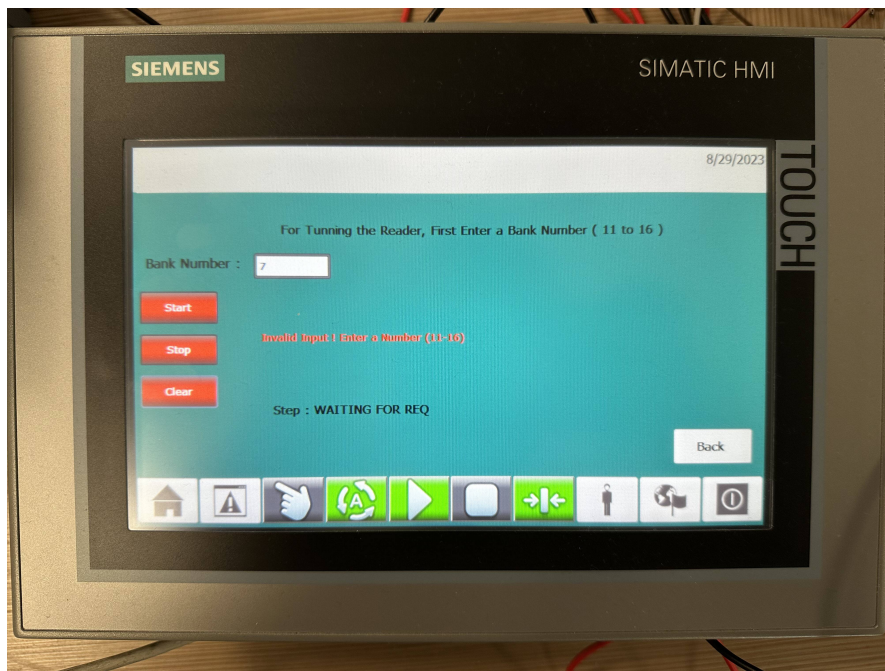


Figure 4.48: AutoTune- Waiting for Request

4.4. INSTALLATION

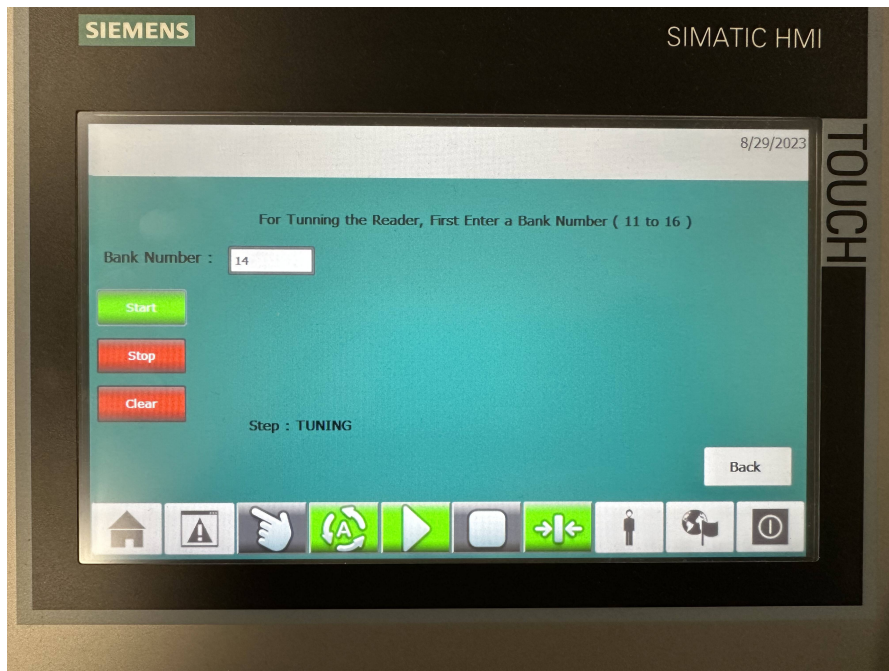


Figure 4.49: AutoTune- Tuning

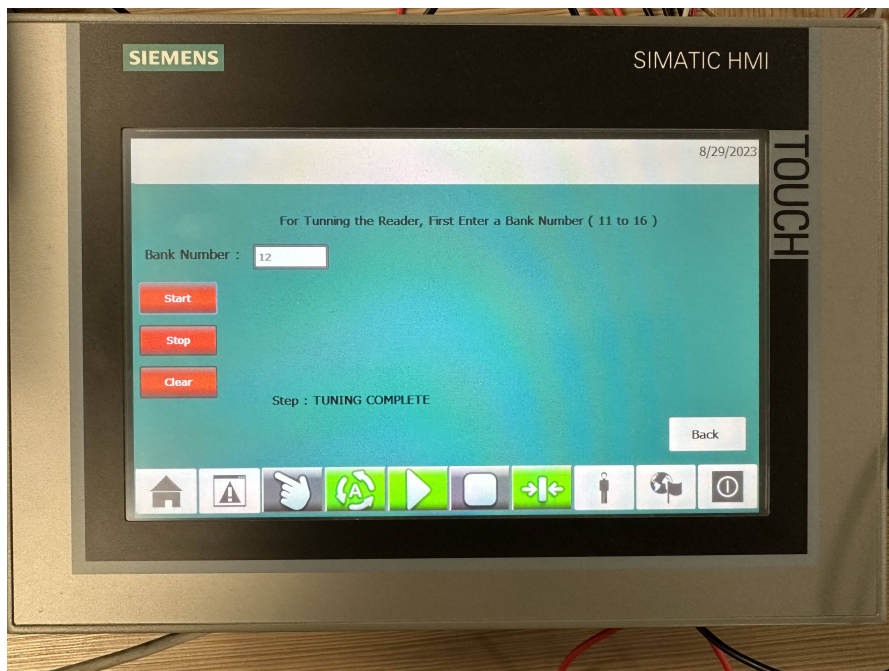


Figure 4.50: AutoTune- Tuning Complete

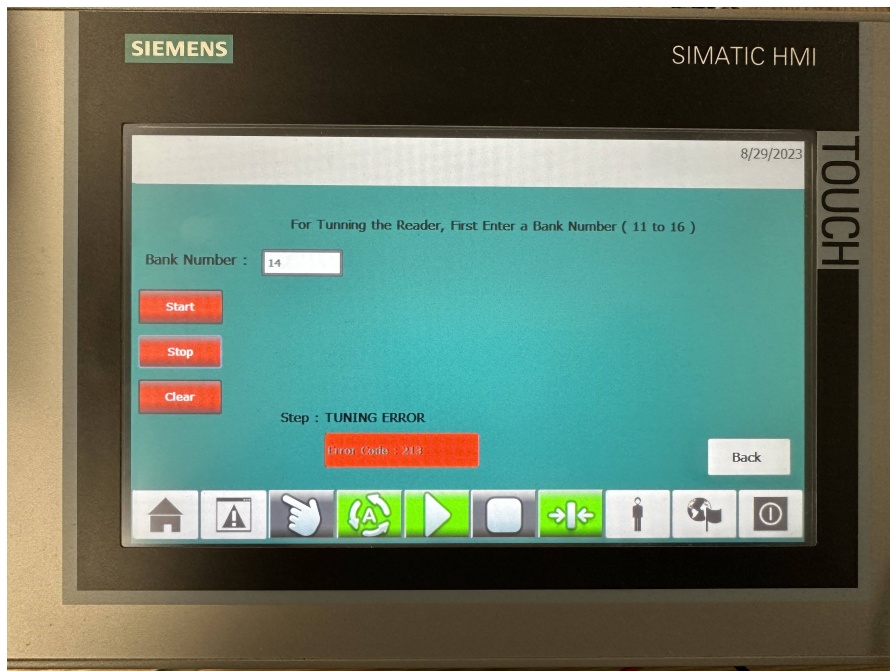


Figure 4.51: AutoTune- Tuning Error

In cases where the operator entered a bank number outside this range, an error message would appear on the HMI (fig. 4.52), prompting the operator to select a number within the valid range.

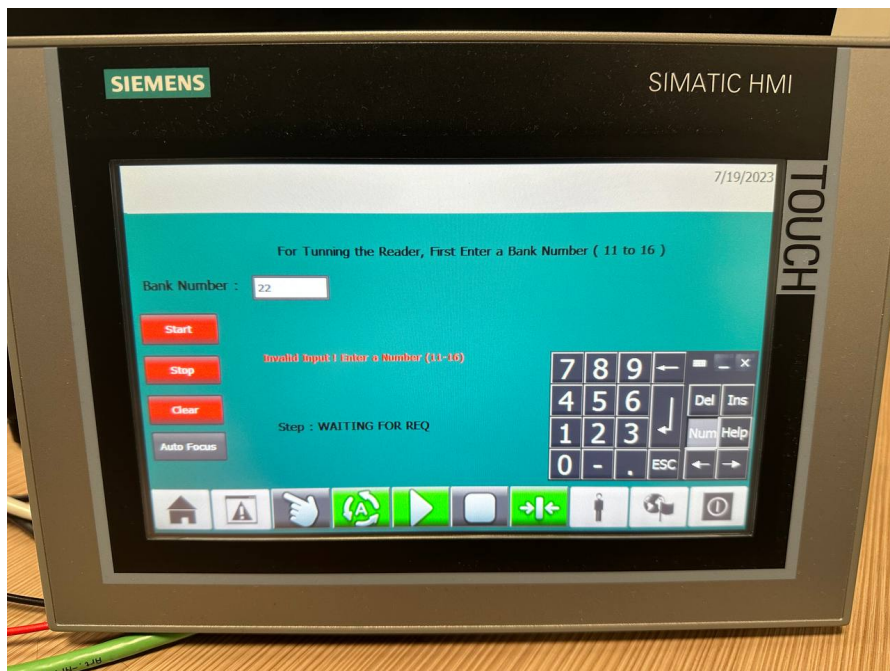


Figure 4.52: Wrong Bank-Number

4.4. INSTALLATION

4.4.6 PRODUCTION LINE

After successfully implementing the Keyence SR-X300 AI-Powered Code Readers and completing the programming, they were mounted at various points along the assembly line. In the following figure (fig. 4.53), a complete view of the entire production line can be observed.

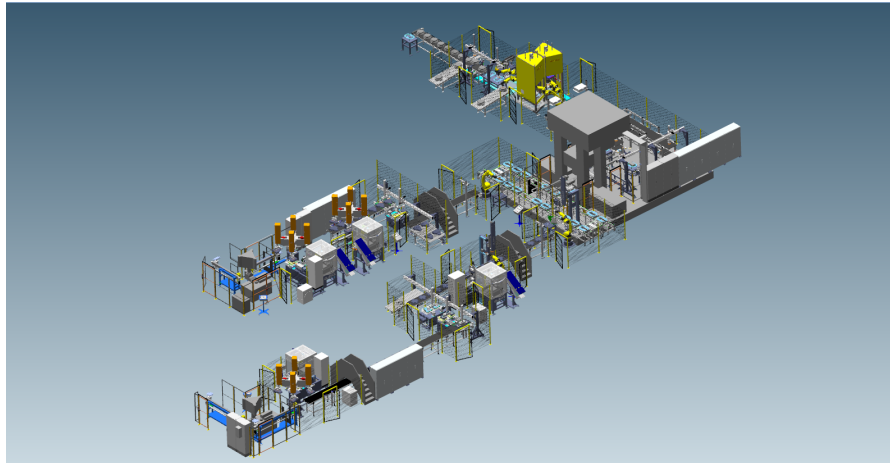


Figure 4.53: Production Line

The first station (fig. 4.54) where a code reader was mounted on, is the "Transferring on Conveyor" station. The code reader was installed on this station for vessels identification.

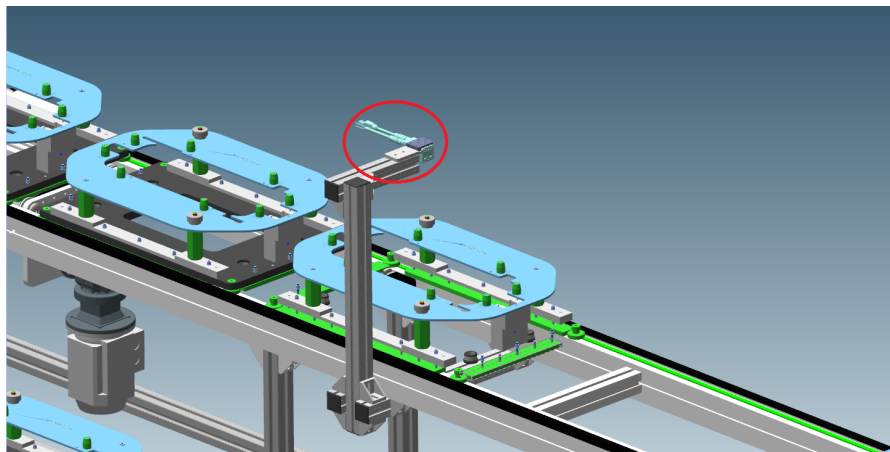


Figure 4.54: Transferring on Conveyor

Two code readers were mounted on the next station "Transferring with two united manipulators with a single horizontal drive" (fig. 4.55), again for vessels identification.

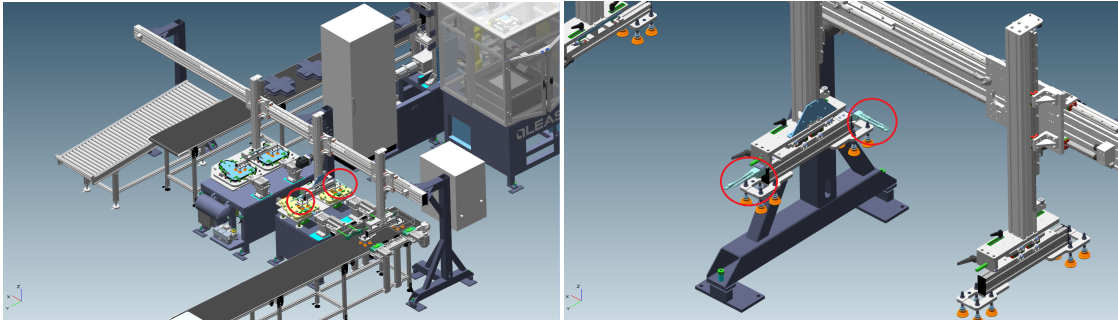


Figure 4.55: Transferring with Manipulators and a Single Motor

After the final junction before reaching the crimping press (fig. 4.56), an upper and a lower device for reading the code on the coupled pots have been mounted.

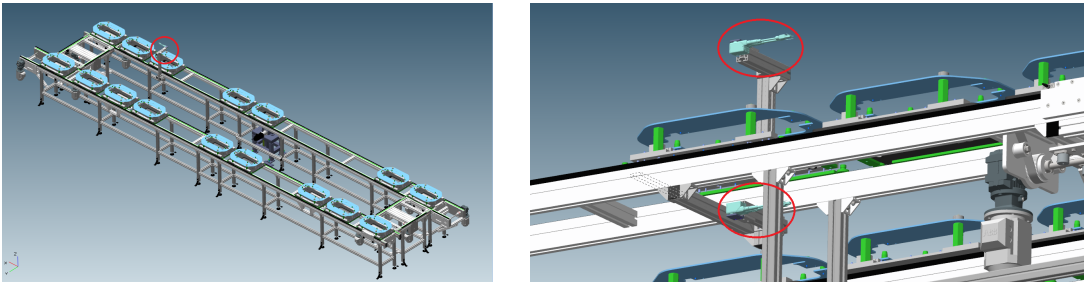


Figure 4.56: Transportation with Pallets

4.4. INSTALLATION

A single code reader has been installed at the "Pre-loading" station (fig. 4.57), while another code reader has been installed at the "Tanks Welding Test" station (fig. 4.58).

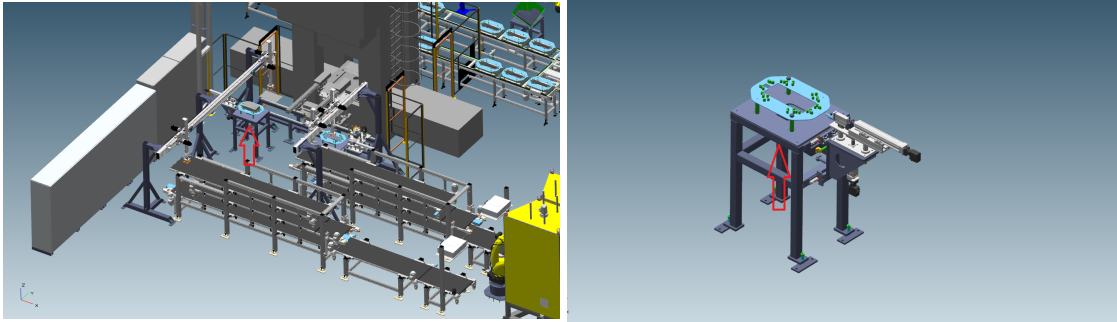


Figure 4.57: Pre-loading

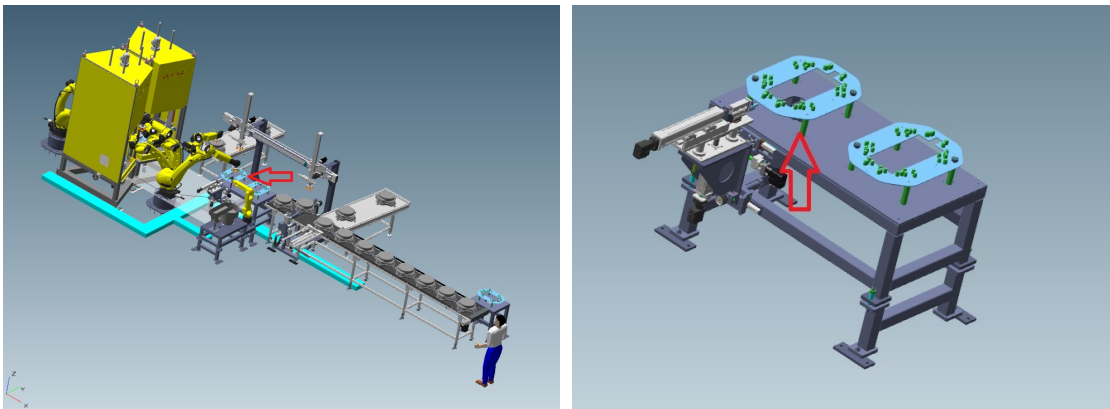


Figure 4.58: Welding Test

In the following figure (fig. 4.59), two types of vessels produced on this line are depicted.



Figure 4.59: Two types of Expansion Tanks

5

Conclusion and Future works

5.1 RESULTS AND CONCLUSION

In the provided table (table 5.1), the extracted data from two different codes on the vessels, representing Tank A and Tank B respectively, are compared. Each code contains a serial number, date, and time of scanning, along with specific information about the type of tank, its dimensions, required pressure, pressure stabilization time, and side protection thickness. These details are extracted from a database upon scanning the code, enabling automatic adjustment of parameters for subsequent assembly processes.

	Tank A	Tank B
Serial N.	0103241402wnv376	0103241407sv450
Date	01/03/2024	01/03/2024
Time	14:02	14:07
Type	White Nylon Vessel	Steel Vessel
Height	220 mm	230 mm
Length	440 mm	430 mm
Pressure	4.55 bar	4.80 bar
Pressure Stabilization	≈ 5s	≈ 4s
Fixed Side Protection	3 mm sheet metal	4 mm sheet metal

Table 5.1: Some Extracted Data from Two Different Types of Tank Codes

5.1. RESULTS AND CONCLUSION

In conclusion, this thesis has delved into various aspects of welding machine automation, focusing on the integration of AI-Powered code readers, PLCs, and other automation components. Through meticulous configuration and optimization processes, a robust system was successfully established, capable of efficiently managing welding processes while ensuring adherence to quality and safety standards.

The implementation of AI-powered code reader technology, along with the integration of automation systems and components such as PLCs, HMIs, and Signal Towers, using Profinet and Modbus protocols for communication and data transfer, significantly enhanced the traceability, functionality, and performance of the automated expansion vessels production line.

5.2 FUTURE WORKS

Looking ahead, there are several avenues for future research and development in welding machine automation. One potential area of focus is the integration of artificial intelligence and machine learning algorithms to enhance predictive maintenance capabilities, optimize welding parameters, and improve overall system performance. Additionally, further exploration of human-machine collaboration, augmented reality interfaces, and digital twin technology could lead to more intuitive and efficient automation systems.

Moreover, ongoing efforts should be directed towards standardization and interoperability among automation components, facilitating seamless integration and compatibility across different systems and manufacturers. Continued research into advanced sensing technologies, materials science, and robotics could also drive innovation in welding automation, unlocking new possibilities for increased efficiency, precision, and versatility in industrial applications.

5.2.1 EMERGING TRENDS IN AI-POWERED CODE READER TECHNOLOGY

AI-Powered code reader technology is evolving rapidly, driven by the need for better performance and functionality. One significant development is the integration of advanced image processing and machine learning algorithms into AI-Powered code readers. These enhancements enable the recognition of damaged or poorly printed codes, ensuring accurate data capture even in challenging conditions. Additionally, AI-Powered code readers are now being linked with Internet of Things (IoT) platforms and cloud-based systems, allowing for seamless data exchange and centralized management across different locations.

Another notable trend is the increasing popularity of mobile code scanning, thanks to the widespread use of smartphones and tablets. This allows for greater flexibility and convenience, particularly in tasks like inventory management and field services. AI-Powered code readers are also becoming more versatile with improved connectivity options like Bluetooth and Wi-Fi, facilitating wireless data transfer and integration with various automation systems. Moreover, the integration of artificial intelligence (AI) technologies enables smarter code recognition, enhancing scanning accuracy across different code types and environmental settings. These advancements highlight the ongoing innovation in AI-Powered code reader technology, promising more efficient and adaptable

5.2. FUTURE WORKS

solutions for diverse industries and applications.

5.2.2 POTENTIAL ADVANCEMENTS IN WELDING MACHINE AUTOMATION

Collaborative robots, known as cobots, represent a significant advancement in automation, particularly in welding machine applications. These robots are specifically designed to work alongside human operators, enhancing productivity and safety. In welding automation, cobots play a vital role in tasks like positioning workpieces, assisting with pre-weld tasks, and supporting the welding process itself. By working in collaboration with humans, cobots improve efficiency while ensuring a safe working environment.

Moreover, the integration of advanced control algorithms and real-time sensing technologies enables adaptive welding control. These systems monitor and adjust welding parameters dynamically based on real-time feedback, ensuring consistent and high-quality welds even in changing conditions. Additionally, advancements in data analytics and predictive maintenance allow for proactive monitoring of welding parameters and equipment performance. By leveraging machine learning algorithms, anomalies can be detected early, potential failures predicted, and maintenance activities scheduled proactively, reducing downtime and improving equipment reliability. These advancements highlight the ongoing evolution of welding machine automation towards seamless human-machine collaboration and enhanced productivity[16].

References

- [1] Suman Adhikari. *Operator Machine Control using Siemens PLC and HMI*. The University of Toledo, 2018.
- [2] Yathov Kumar Bala Kumar. “Progetto di banco prove didattico per la verifica dell’affidabilità di componenti e sistemi= Design of didactic test bench to verify the reliability of components and systems”. PhD thesis. Politecnico di Torino, 2019.
- [3] Gavali Amit Bhimrao and S Patil Mahadev. “PLC based industrial automation system”. In: *International Conference on Recent Trends in Engineering and Management Science*. 2014.
- [4] William Bolton. *Programmable logic controllers*. Newnes, 2015.
- [5] Conger. *6 types of Automation*. URL: <https://www.conger.com/types-of-automation/>.
- [6] ControlEngineeringMagazine. *ControlEngineeringMagazine*. URL: <https://www.controleng.com/>.
- [7] Datalogic. *Barcode Reader solutions in manufacturing*. URL: <https://www.datalogic.com/eng/industries-applications/manufacturing-sup-1.html>.
- [8] ABDELRAHMAN GABER and MAMDOUH MOHAMED KAMEL. “Industry 4.0 and lean manufacturing across different companies with analysis of its implementation”. In: (2019).
- [9] Ernie Hayden GICSP, Michael Assante, and Tim Conway. “An abbreviated history of automation & industrial controls systems and cybersecurity”. In: *SANS Institute, Tech. Rep.* (2014).

REFERENCES

- [10] Mario Hermann, Tobias Pentek, and Boris Otto. "Design principles for industrie 4.0 scenarios". In: *2016 49th Hawaii international conference on system sciences (HICSS)*. IEEE. 2016, pp. 3928–3937.
- [11] IBM. *what is Automation?* URL: <https://www.ibm.com/topics/automation>.
- [12] InductiveAutomation. *what is HMI?* URL: <https://inductiveautomation.com/resources/article/what-is-hmi>.
- [13] Haoqiang Ji. "PLC Programming For A Water Level Control System: Design and System Implementation". In: (2017).
- [14] Keyence. *SR-X Series Barcode Readers*. URL: <https://www.keyence.it/>.
- [15] microdigisoft. *PLC communication protocols and Its Types*. URL: <https://microdigisoft.com/plc-communication-protocols-and-its-types/>.
- [16] MillerWelds. *The welding technology trends*. URL: <https://www.millerwelds.com/resources/article-library/the-latest-welding-technologies-can-save-time-and-money>.
- [17] Nikhita Nadgauda and Senthil Arumugam Muthukumaraswamy. "Design and development of industrial automated system using PLC-SCADA". In: *2019 IEEE 10th GCC Conference & Exhibition (GCC)*. IEEE. 2019, pp. 1–6.
- [18] Pochevskiy Oleg. *Enhancing Accuracy and Efficiency: The Role of Barcodes in Inventory Management*. URL: <https://www.cleverence.com/articles/business-automation/enhancing-accuracy-and-efficiency-the-role-of-barcodes-in-inventory-management/>.
- [19] Patlite. *Patlite*. URL: <https://www.patlite.it/>.
- [20] Zoltán Rajnai and István Kocsis. "Assessing industry 4.0 readiness of enterprises". In: *2018 IEEE 16th world symposium on applied machine intelligence and informatics (SAMI)*. IEEE. 2018, pp. 000225–000230.
- [21] Vidya S Rao, K Praveen Shenoy, and KV Santhosh. "Design of PLC based automated food processing machine". In: *International Journal of Mechanics* 15 (2021), p. 22.
- [22] realpars. *what is Automation?* URL: <https://realpars.com/>.
- [23] Vanessa Romero Segovia and Alfred Theorin. "History of Control History of PLC and DCS". In: *University of Lund* (2012).
- [24] Siemens. *Siemens Products*. URL: <https://www.siemens.com/global/en.html>.

REFERENCES

- [25] American Welding Society. *Journal of the American Welding Society*. Vol. 8. American Welding Society, 1929.
- [26] Apostolos Tsagaris and Evangelos Hatzikos. "Implementation of environmental monitoring system with PLC and SCADA". In: *International journal of mechanical & mechatronics engineering* 14.06 (2014), pp. 33–38.
- [27] USProfinet. *Comparison and Migration Strategies*. URL:https://us.profinet.com/wp-content/uploads/2018/07/PB-vs-PN_v2.pdf.
- [28] Wikipedia. *Wikipedia*. URL:<https://en.wikipedia.org/wiki/Automation>.

Acknowledgments

Firstly, I would like to express my heartfelt gratitude to Professor Alberto Morato, whose invaluable guidance and support have been instrumental in the successful completion of this thesis.

I extend my sincere appreciation to Mr. Francesco Meneghini, my esteemed manager at LEAS SPA, for granting me this invaluable opportunity to undertake my internship and thesis within their company. I am deeply grateful for the opportunity to work under his supervision.

I also thank the entire team at LEAS SPA and everyone who has played a role in this journey, I extend my heartfelt thanks.