VIETNAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

FACULTY OF ELECTRICAL & ELECTRICONIC ENGINEERING

NGUYEN DUC TRUNG DAN

AUTOMATIC CONTROL, MONITORING AND WAREHOUSE MANAGEMENT SYSTEM FOR DAIRY PRODUCTS

UNDERGRADUATE THESIS OF AUTOMATION AND CONTROL ENGINEERING

HO CHI MINH CITY, YEAR 2024

VIETNAM GENERAL CONFEDERATION OF LABOUR

TON DUC THANG UNIVERSITY

FACULTY OF ELECTRICAL & ELECTRICONIC ENGINEERING

NGUYEN DUC TRUNG DAN – 419H0126

AUTOMATIC CONTROL, MONITORING AND WAREHOUSE MANAGEMENT SYSTEM FOR DAIRY PRODUCTS

UNDERGRADUATE THESIS OF AUTOMATION AND CONTROL ENGINEERING

Advised by
Dr. Tran Duc Anh Minh

HO CHI MINH CITY, YEAR 2024

ACKNOWLEDGMENT

To complete this graduation project, first. I would like to send my most sincere thanks to the teachers at Ton Duc Thang University. I would like to send my deepest thanks to Dr. Tran Duc Anh Minh, who has accompanied and wholeheartedly helped me to complete this graduation project. Due to my limited knowledge and lack of experience, there are still many mistakes in the process of implementing the graduation project. I sincerely hope to receive comments from teachers.

Ho Chi Minh City, July 20th, 2024

Author

Nguyen Duc Trung Dan

This thesis was carried out at Ton Duc Thang University.

Advisor: Dr. Tran Duc Anh Minh.....

.....

(Title, full name and signature)

This thesis is defended at the **Undergraduate Thesis Examination Committee** held at Ton Duc Thang University on August 10th, 2024.

Confirmation of the Chairman of the Undergraduate Thesis Examination Committee and the Dean of the faculty after receiving the modified thesis (if any).

CHAIRMAN **DEAN OF FACULTY**

……. ……

DECLARATION OF AUTHORSHIP

I hereby declare that this thesis was carried out by myself under the guidance and supervision of **Dr. Tran Duc Anh Minh**; and that the work and the results contained in it are original and have not been submitted anywhere for any previous purposes. The data and figures presented in this thesis are for analysis, comments, and evaluations from various resources by my own work and have been duly acknowledged in the reference part.

In addition, other comments, reviews and data used by other authors, and organizations have been acknowledged, and explicitly cited.

I will take full responsibility for any fraud detected in my thesis. Ton Duc Thang University is unrelated to any copyright infringement caused on my work (if any).

Ho Chi Minh City, July 20th, 2024

Author

Nguyen Duc Trung Dan

TON DUC THANG UNIVERSITY
FACULTY OF ELECTRICAL & ELECTRONICS ENGINEERING

SOCIALIST REPUBLIC OF VIETNAM
Independence – Freedom – Happiness

GRADUATION PROJECT SCHEDULE

Student's name: Nguyen Duc Trung Dan .....
Class: 19H40301 ..... Student ID: 419H0126.....
Project's title: AUTOMATIC CONTROL, MONITORING AND WAREHOUSE MANAGEMENT SYSTEM FOR DAIRY PRODUCTS.

Week/Date	Workload		Instructor's sign
	Done	Next task	
1	- Receive the topic and find out the content.	- Learn control about step motor.	
2	- Buy electrical cabinets, PLCs. Push buttons, stepper motors, ... - Test PLC and stepper motor.	- Buy the necessary equipment.	
3	- Buy wood for storage and flooring for the system. - Buy Relay, trough, camera.	- Complete xyz robot.	
4	- Complete xyz robot - Connect the PLC to the device.	- Research the control process of xyz robot.	
5	- Design SCADA interface.	- Start programming the controller.	

6	- Manual control system code.	- Come up with an idea to build a warehouse.	
7	- Research on image processing of QR code scans using C# language.	- Buy infrared sensors - Continue writing PLC programs.	
Midterm test	Evaluate completed volume 60% to continue/not continue graduation project.		
8	- Complete manual import and export mode.	- Continue writing programs - Research how to save values on the database.	
9	- Learn how to use data base with SQL.	- Continue writing programs.	
10	- Complete auto mode program.	- Research how to automatically save values into excel files in TIA software.	
11	- Read and write values on PLC using C#. - Complete hardware.	- Complete the software and check the system overview.	
12	- Check for system and program errors.	- Test Project.	
13	- Test Project.	- Test Project.	
14	- Report write a	- Submit for	

	project report.	plagiarism check.	
15	- Complete the report writing.		
Submit	Completed 100% graduation project. Graduation project's final defense.		

HỆ THỐNG ĐIỀU KHIỂN, GIÁM SÁT VÀ QUẢN LÝ KHO TỰ ĐỘNG CÁC SẢN PHẨM SỮA

TÓM TẮT

Dự án tập trung vào việc thiết kế và triển khai hệ thống kho tự động sử dụng các thành phần chính như PLC S7-1200, trình điều khiển TB6600 và động cơ bước. PLC S7-1200 được sử dụng để điều khiển và giám sát các quy trình tự động trong hệ thống kho. Trình điều khiển TB6600 chịu trách nhiệm điều khiển động cơ bước, cung cấp năng lượng và chuyển động chính xác cho các thiết bị chuyển động.

Hệ thống kho sẽ tự động thực hiện các nhiệm vụ như vận chuyển hàng hóa từ điểm A đến điểm B, phân loại và lưu trữ chúng dựa trên các lệnh nhập từ giao diện người dùng hoặc hệ thống quản lý hàng tồn kho. Tự động hóa này giúp tối ưu hóa quy trình kho, giảm thiểu sự can thiệp của con người và nâng cao hiệu quả công việc chung.

Hệ thống được điều khiển và giám sát thông qua giao diện người dùng trên PLC S7-1200, nơi người vận hành có thể quan sát trạng thái hoạt động, điều chỉnh và đưa ra các lệnh điều khiển khi cần. Việc tích hợp PLC, trình điều khiển TB6600 và động cơ bước tạo ra một hệ thống tự động linh hoạt và hiệu quả để quản lý kho.

AUTOMATIC CONTROLLER, MONITORING AND WAREHOUSE MANAGEMENT SYSTEM FOR DAIRY PRODUCTS

ABSTRACT

The project focuses on designing and implementing an automatic warehouse system using key components such as PLC S7-1200, TB6600 driver, and stepper motor. The PLC S7-1200 is utilized for controlling and monitoring automated processes within the warehouse system. The TB6600 driver is responsible for controlling the stepper motor, providing power and precise motion for the moving devices.

The warehouse system will autonomously perform tasks such as transporting goods from point A to point B, sorting, and storing them based on commands input from the user interface or inventory management system. This automation helps optimize the warehouse process, minimizing human intervention and enhancing overall work efficiency.

The system is controlled and monitored through the user interface on the PLC S7- 1200, where the operator can observe the operational status, adjust, and issue control commands as needed. The integration of PLC, TB6600 driver, and stepper motor creates a flexible and efficient automated system for warehouse management.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATION

PLC	Programmable Logic Controller
DC	Direct Current
AC	Alternating Current
CB	Circuit Breaker
SCADA	Supervisory Control and Data Acquisition
QR	Quick Response

CHAPTER 1. OVERVIEW

1.1 Purpose of the topic

Nowadays, the world is aiming to optimize everything, and storage is one of those aspects. Optimizing the storage process by automating tasks such as organizing, storing, transporting, and retrieving goods. This automation reduces dependency on human labor and increases task performance efficiency.

This project is controlled by the PLC S7-1200, providing a flexible control and monitoring platform. The PLC system not only helps in controlling but also offers quick monitoring, supervision, and interaction from the user side.

At the same time, it sets a goal to comprehensively evaluate the performance of the automated system. By measuring accuracy during transportation and response times, it provides insights into the system's operational efficiency in a real-world environment.

Additionally, the use of stepper motors and TB6600 drivers is also an important part of this project. Using stepper motors to achieve precise and flexible movement during the transportation of goods is more cost-effective than using servo motors.

Finally, developing a user-friendly control interface is an essential part. This interface will help operators easily monitor, control, and manage the system efficiently.

In summary, the project aims not only to create an automated warehousing system but also to set higher goals that can be applied in a factory or distribution warehouse environment.

1.2 Requirements of the topic

- Research and present the technical requirements for the model
- Select and present the operating principles of components: PLC S7-1200, stepper motor and driver, infrared sensors, camera, etc.

- Design and construct the model.
- Program for control and monitoring.
- Write a report and prepare a poster according to the template.

1.3 Ideas and Methodology of Implementation

The system model is designed to apply automation to the processing procedure, using wood material for the storage frame and integrating key components for control such as stepper motors, TB6600 drivers, electrical cabinets, stepper motor-driven conveyors:

Storage Frame:

- Made from wood, it consists of nine compartments designed to hold pallets with products.

Control System:

- Utilizes three stepper motors corresponding to three axes of movement (X, Y, Z) along with three TB6600 drivers to control movement and position the storage location of the products.

Electrical Cabinet:

- The cabinet measures 300x400x210 mm, protecting and organizing the wiring and control units.
- The front panel includes:
  - 4 status indicator lights (Power, Run, Stop, Fault)
  - 2 push buttons (Run and Stop)
  - 1 three-position switch for Manual/Auto mode control

Power Supply:

- Uses a 220VAC power source for the system.
- Uses a 24VDC-5A power supply unit for the PLC.

Conveyor:

- Used to transport products to the desired location.
- Speed can be controlled.

Intermediate Relay:

- Utilizes intermediate relays to control devices and functions within the system.

Image Processing with QR Codes and PLC Communication:

- Uses C# to create a WinForms interface, employing the Zxing library for reading QR codes, the S7.Net library for PLC communication, and the Aforge library for displaying images.

User Interface:

- Monitors and controls through a SCADA interface.
- Integrates indicator lights and control buttons on the electrical cabinet to provide a user control interface.

Functions:

- Manual Mode: Allows the user to manually intervene and control the system.
- Auto Mode: The system operates automatically based on input from cameras and sensors.

Conclusion:

- The system combines automation using sensor inputs, image processing, and product control to create a flexible and efficient process.

CHAPTER 2. THEORETICAL FOUNDATION

2.1 Hardware Overview

2.1.1 PLC S7-1214 DC/DC/DC

A PLC (Programmable Logic Controller) is a programmable control device that allows users to flexibly implement control logic algorithms through a programming language. Users can program the PLC to execute a sequence of events. These events are triggered by stimuli (also known as inputs) acting on the PLC or by delayed actions such as periodic timing. In practice, PLCs are also used to replace relay circuits. Figure 2.1 PLC S7-1214 DC/DC/DC.

Figure 2.1: PLC S7-1214 DC/DC/DC [1]

Structure and Configuration:

- The SIMATIC S7-1214C module is a small-sized CPU within the S7-1200 series.
- Power Supply: Uses a 24VDC input power supply, capable of operating in industrial environments that require DC power.

Specifications:

- Includes 14 digital inputs (DI) and digital outputs (DO).
- Supports three main programming languages: Ladder Logic (LAD), Function Block Diagram (FBD), and Structured Text (ST).

Communication:

- Network Communication: Supports device communication through an integrated Ethernet port.
- Expandable Connectivity: Can be expanded with additional modules to increase the number of inputs/outputs and other features.

Special Features:

- Motion Control Function: Capable of supporting applications requiring motion control such as stepper motors, servo motors, etc.
- Security: Provides features such as secure connections, program protection, and data security.

Programming Software:

- TIA Portal Versions: The use of TIA Portal includes multiple versions, offering convenience in programming and control tasks.

Integrated Control Function:

- Expandable Analog Inputs and Outputs: Capable of expanding analog input and output channels.
- Capable of supporting PID (Proportional-Integral-Derivative) calculation and control.

Connectivity and Accessories:

- USB Port: Supports USB connectivity for data transfer and configuration.
- Ethernet Port**:** Includes an Ethernet port for network communication and integration.

Operating Environment:

- Designed to operate in a wide range of environments with a broad temperature

range.

Application

- Commonly used in small to medium-sized industries for various systems.

Figure 2.2 depicts the PLC datasheet

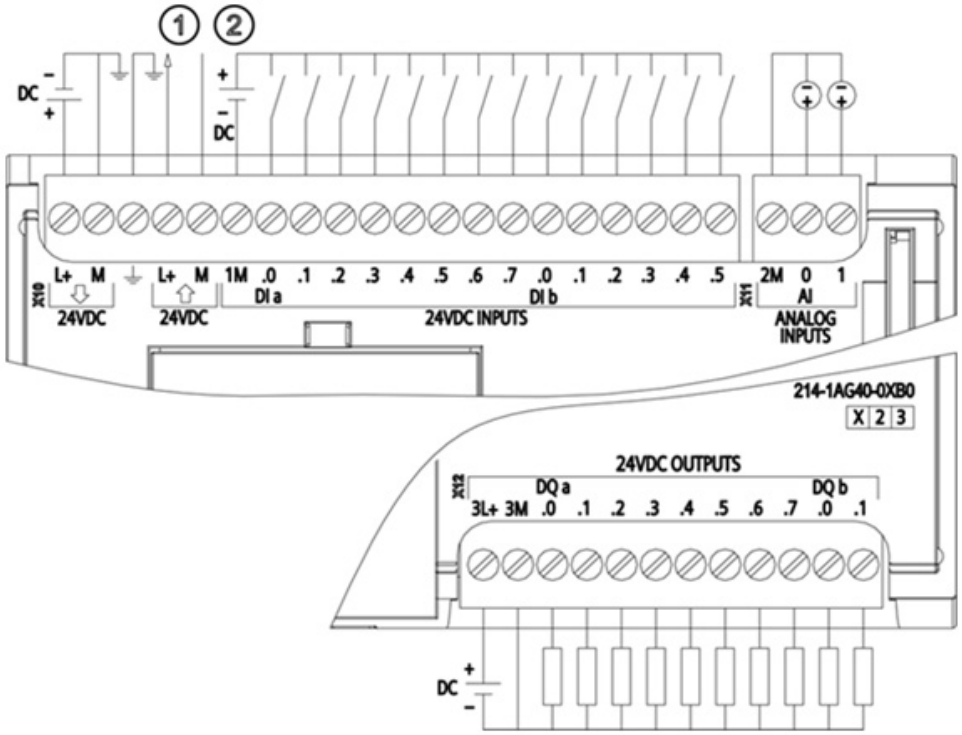

Figure 2.2: Datasheet PLC [1]

2.1.2 Step motor

A stepper motor is a specialized type of electric motor used in applications requiring precise motion in discrete steps. This motor operates by controlling currents through coils inside the motor to generate magnetic fields from magnets. The currents are precisely controlled to produce angular or linear motion steps.

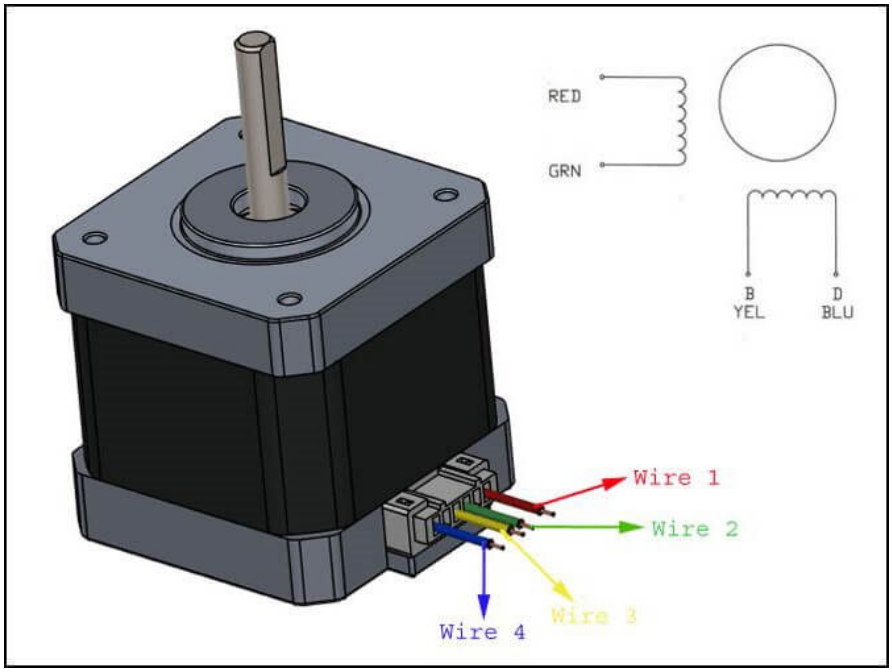

Figure 2.3: Step motor [2]

A 2-phase stepper motor operates with two coils, each connected to two lead wires. It has a step angle of 1.8 degrees, meaning each step of the motor corresponds to a movement of 1.8 degrees. Figure 2.3 describes the components of a stepper motor.

Structure:

- **Rotor:** Formed from permanent magnets.
- **Stator:** Formed from coils of wire.

Operating Principle

To move the shaft of a 2-phase stepper motor, electrical power needs to be supplied to the coils. When power is applied to a coil, the rotor moves in the direction determined by that coil. Subsequently, power is applied to the other coil, causing the rotor to move in the opposite direction. This process repeats continuously to create the desired rotational motion.

Advantage

- Affordable cost
- Available in various sizes.

- High durability.
- Supported by drivers.

Disadvantage

- Low rotational speed
- Low torque.

Real-world Applications

- CNC machines
- Robots
- 3D printer
- Engraving machines

Note:

- Selecting a driver suitable for the motor.
- Choosing a suitable input voltage for the motor.
- Controlling and using the motor correctly.

2.1.3 Driver TB6600

The TB6600 driver is a widely used and popular stepper motor driver. Figure 2.4 depicts the device

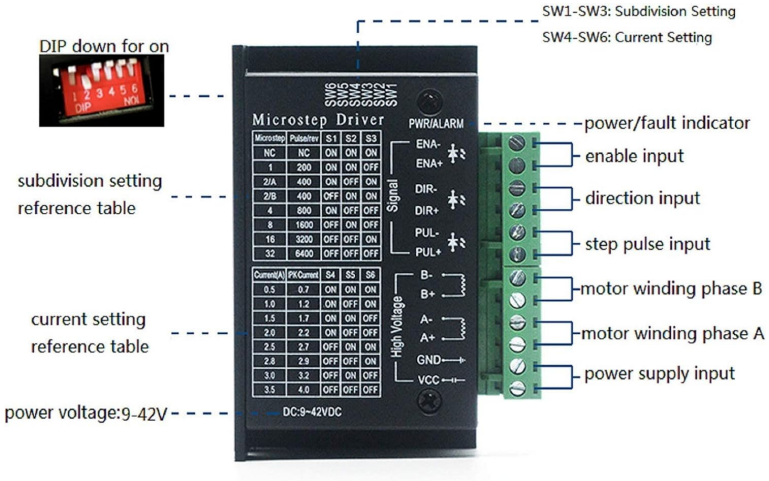

Figure 2.4: Driver TB6600 [3]

Function:

- Capable of controlling both unipolar and bipolar stepper motors

- Provides frequency division functions to adjust the speed and position of the motor

Specifications:

- Uses a DC power supply ranging from 12VDC to 48VDC.
- Provides output current for the stepper moto

Security and Safety:

- Features overcurrent and overheat protection.
- Prevents short circuits in case of a fault

Communication and Connectivity:

- Has ports for easy connection to control signals and power supply

Applications:

- CNC Machine
- Robotic

Benefit:

- TB6600, with its low cost and high efficiency, is a popular choice for projects with limited budgets.

Drawback:

- Not necessarily the best choice for every application, depending on the specific project requirements.

Conclusion:

- The TB6600 is an efficient and cost-effective stepper motor driver, suitable for various motion control applications and is an important component in CNC and robotics systems. Understanding its features and limitations will ensure proper usage in projects.

2.1.4 Conveyor

Figure 2.5:Conveyor

- The purpose of using a 2-line conveyor is to fit the project's structure. The conveyor is driven by a stepper motor and can have its speed adjusted as desired. Figure 2.5 illustrates the 2-line conveyor.

2.1.5 Infrared Sensor

An infrared obstacle sensor, also known as an IR sensor, is a device that uses infrared radiation to measure or detect objects in the environment. Figure 2.6 depicts the appearance of the sensor.

Figure 2.6: Infrared Sensor [4]

Specifications:

- Model: E3F-DS30P1
- Number of signal wires: 3 wires (2 power supply wires and 1 signal wire).
- Output signal type: PNP type - when there is no object detected, the output is at 0 level; when an object is detected, the output is at 1 level (0 level = 0V, 1 level = supply voltage).
- Power supply voltage: 6~36VDC.
- Adjustable detection range of the sensor: 5~30 cm (adjusted by a trimmer on the sensor).
- Beam angle: 3~5 degrees.
- Output current: < 300 mA.
- LED indicator: red color.
- Material: outer casing made of ABS plastic, with adhesive around to

waterproof and shockproof.

- Dimensions: 70x24 mm.

Datasheet:

- Brown wire: Positive power supply (VCC) from 6~36 VDC.
- Green wire: Ground (GND) 0 VDC.
- Black wire: Output signal wire.

2.1.6 Camera

Use a separate camera to scan QR codes to identify and manage products. Figure 2.7 Depicts a surveillance camera

Figure 2.7: Webcam [5]

Specifications:

- Resolution: 1080p.
- Material: ABS plastic.
- Features: Built-in microphone support.
- Frame rate: 30 FPS.
- System support: Windows and Mac.
- High clarity Versatile stand/base.
- Connection via USB Jack.

2.1.7 Circuit Breaker

A circuit breaker is an electrical switch designed to automatically interrupt electrical flow when it detects an overload or a short circuit within an electrical circuit. Its primary function is to protect electrical circuits and appliances from damage caused by excess current, which can lead to overheating, fire hazards, or damage to equipment.

Figure 2.8: Circuit Breaker 2 phase (220V - 16A) [6]

This project will use a 2-pole circuit breaker with an operating voltage of 22 V and a rated current of 16 A. Figure 2.7 depicts the circuit breaker

Specifications:

- Switch type: 2-pole.
- Operating voltage: 220 V.
- Rated current: 16 A.

Structure and appearance:

It has a compact design, durable and high-quality materials. With 2 poles, it

effectively protects and controls electrical circuits. The main structure includes an explosion-proof mechanism, actuation mechanism, and sensor components.

Advantages:

- Safety.
- High durability.
- High load-bearing capacity.
- No external power supply required
- Easy to install.

Disadvantages:

- Low accuracy.
- Often large.

Applications:

- Applications in power distribution systems
- Applications in backup systems
- Applications in research and development

2.1.8 Immediate relay

Schneider 5A/230VAC Series RXM2LB2P7 intermediate relay belongs to Schneider Electric's RXM2LB product line. Designed to meet the requirements in the industrial electrical sector, relays are intended to protect and control systems. Figure 2.9: Description of the shape of an intermediate relay.

Figure 2.9: 8-pin intermediate relay [7]

Specifications:

- Current rating: 5A
- Operating voltage: 230VAC
- Product code: RXM2LB2P7

Features:

- Structure: An 8-pin intermediate relay typically has 8 pins for connecting to the electrical circuit, including coil pins and contact pins.
- Contacts: May have one or more sets of contacts (usually NO - Normally Open, NC - Normally Closed).
- Coil: Controlled by a certain voltage source (AC or DC).
- Switching Capability: Can switch various signals or power sources depending on the relay's design and capacity.
- Size and Shape: Compact, easy to install in panels and systems.

Advantages:

- Separation of Control and Load: Helps to isolate the control circuit from the load circuit, protecting sensitive components from high currents.
- High Durability: Robust mechanical design that can withstand numerous switching cycles.
- Safety: Minimizes the risk of electrical short circuits and protects other devices

in the circuit.

- Easy Replacement: Often designed to standards, making it easy to replace when necessary.

Disadvantages:

- Energy Consumption: Consumes a certain amount of energy to maintain the coil in an activated state.
- Latency: There is a certain delay from the time a control signal is given until the contracts open or close.
- Limited Lifespan: Despite being durable, the contacts may wear out after a certain number of switching cycles.
- Electromagnetic Interference: Can generate electromagnetic interference during operation.

Applications:

- Automated Control Systems: Used in PLC (Programmable Logic Controller) systems to control other devices.
- Protection Systems: Used in circuits for overload protection, leakage current protection, and other safety systems.
- Signal Control: Switching signals in telecommunications, computers, and electronic devices.
- HVAC Systems: Used in controlling air conditioning units and heating systems.
- Industry: Widely applied in manufacturing, automation, and energy sectors.

Conclusion:

An intermediate relay is an optimal and flexible solution for applications in the industrial electrical field. This device not only meets high technical requirements but also provides safety and efficiency for the system.

2.1.9 Fuse

Figure 2.10: Fuse [8]

Using CHINT fuses, the purpose of the fuses is to protect and control the electrical current. Figure 2.10: Description of the shape of an Fuse.

Technical Specifications:

- Brand: Chint.
- Type: Fuse base.
- Features: Handles high current and voltage.
- Installation: Easy to install.

Here are the main functions:

- Circuit Protection: Protects circuits from short circuits and overloads.
- Control and Synchronization: Controls and synchronizes devices within the system.

2.1.10 Power Supply 24VDC-5A

Figure 2.11: Power Supply [9]

Power Supply is used to provide 24VDC power to devices in industrial electrical and automation systems. Figure 2.11: Description of the shape of an Power Supply.

Function:

- Stable power supply
- Circuit protection
- Meeting system requirements
- Easy installation and operation
- Integrated protection

Specifications

- Output voltage: 24VDC
- Input voltage: 220VAC
- Current: 5A
- Protection against overload and short circuit

2.2 Overview of Connectivity and Data Transmission Methods

2.2.1 S7.NET

The S7.NET library is a .NET library designed to facilitate communication between .NET applications and Siemens PLCs (Programmable Logic Controllers). It provides an easy-to-use interface for interacting with Siemens S7 series PLCs, including models such as the S7-200, S7-300, S7-400, S7-1200, and S7-1500. This library is particularly useful for developers who need to integrate PLC data into their .NET applications without dealing with the complexities of low-level PLC communication protocols. [10]

Key features of the S7.NET library include:

- Connection and Communication: Allows connection to Siemens PLCs via S7 protocol to transmit and receive data.
- Read and write data: Provides methods to read and write data from variables and data blocks (Data Block) on the PLC.
- Synchronization and event processor: Supports synchronization functions to execute read and write commands efficiently, as well as an event processor to process messages and status from the PLC.
- Supports multiple PLC series: The library supports various Siemens PLC series, such as the S7-1200 and S7-1500, making it easy to integrate with a variety of systems.
- Integration into C# applications: S7.NET is written in the C# programming language, suitable for integration into Windows Forms, WPF applications or web applications via ASP.NET.

Advantages:

- Ease of Use.
- Compatibility.
- Open Source.
- Documentation and Community Support.

- Integration with .NET.

Disadvantages:

- Limited to Siemens PLCs.
- Performance.
- Feature Set.
- Dependency on .NET.

Overall, S7.NET is a powerful and user-friendly library that simplifies the process of integrating Siemens PLCs with .NET applications, offering significant advantages for developers while having a few limitations that should be considered based on the specific use case.

2.2.2 ZXING.NET

1D/2D barcode image processing library implemented in Java. ZXING.NET allows for barcode generation and reading in .NET applications. [11]

Features:

Barcode Reading:

- 1D Barcodes: Supports a wide range of 1D barcodes such as EAN-13, EAN-8, UPC-A, UPC-E, Code 39, Code 128, ITF, RSS-14, and others.
- 2D Barcodes: Supports 2D barcodes such as QR Code, Data Matrix, PDF 417, Aztec, and more.

Barcode Generation:

- Can generate various types of barcodes including both 1D and 2D formats.

Integration:

- Easily integrates with .NET applications including desktop, web, and mobile applications.
- Provides support for various image formats including PNG, JPEG, BMP, GIF, etc.

Performance:

- Optimized for high performance in reading and writing barcodes.

Customization:

- Offers a range of options for customizing the generated barcodes such as size, color, and margin.

Support for Multiple Platforms:

- Compatible with .NET Framework, .NET Core, Xamarin, and other platforms supported by .NET Standard.

Applications

- Inventory Management: Reading and generating barcodes for inventory items.
- Retail: Scanning product barcodes for point-of-sale systems.
- Logistics: Tracking shipments using barcodes.
- Healthcare: Managing patient records and medication using barcodes.
- Library Systems: Checking books in and out using barcodes.

Conclusion

ZXING.NET is a robust and flexible library for barcode processing in .NET applications. Its extensive support for different barcode formats and ease of integration makes it a valuable tool for developers in various domains.

2.2.3 AFORGE VIDEO

The AForge.Video library is a part of the AForge.NET framework, which is designed for developers and researchers working on computer vision, artificial intelligence, and robotics. The AForge.Video library provides essential tools and functionalities for video processing, allowing users to capture, manipulate, and analyze video streams from various sources. [12]

Key Features:

Video Sources:

- Supports multiple video sources, including webcams, IP cameras, video files, and more.
- Ability to connect to a variety of video devices and streams for capturing live video.

- Frame Grabbing:
- Enables capturing individual frames from the video stream for further processing.
- Supports real-time frame acquisition with efficient handling and processing.

Video Playback:

- Provides functionalities to play video files and streams within applications.
- Allows control over playback speed, frame navigation, and other playback settings.
- Video Encoding and Decoding:
- Offers encoding and decoding capabilities for different video formats.
- Supports conversion between various video codecs and formats.

Motion Detection:

- Integrates motion detection algorithms to identify and track moving objects within the video.
- Allows customization and tuning of motion detection parameters for specific applications.

Filters and Effects:

- Includes a set of filters and effects to enhance video quality and appearance.
- Provides tools for image processing, such as grayscale conversion, color adjustments, and more.

Event Handling:

- Supports event-driven programming with events for new frames, video source errors, and more.
- Allows developers to implement custom logic and responses to video events.

Applications:

- Surveillance Systems: Used for creating security and monitoring systems with live video feeds and motion detection.
- Robotics: Enables video-based control and navigation for robots.

- Computer Vision: Facilitates the development of applications that require image and video analysis, such as object recognition and tracking.
- Multimedia Applications: Supports the development of applications for video playback, editing, and streaming.

CHAPTER 3. DESIGN AND CONSTRUCTION

3.1 Design and Construction of Electrical Circuits

Use EPLAN Electric software to design electrical circuits for the system.

Figure 3.1: Eplan Electric Software [13]

EPLAN Electric is a comprehensive software solution used for designing electrical circuits and systems. It provides a robust platform for electrical engineering, offering various tools and features that streamline the design process, enhance productivity, and ensure high-quality documentation.

Advantages:

1. **Efficient Design Process:**
   - EPLAN Electric offers a highly efficient design environment with advanced tools for creating and modifying electrical schematics.
   - It supports automatic generation of documentation, reducing manual work and minimizing errors.
2. **Standardization and Reusability:**
   - Facilitates standardization by allowing the creation of reusable templates, macros, and components.
   - Ensures consistency and adherence to industry standards and regulations.
3. **Comprehensive Documentation:**
   - Automatically generates detailed reports, including bills of materials, terminal diagrams, and wiring lists.

- Provides clear and precise documentation essential for manufacturing, installation, and maintenance.

4. **Integration with Other Systems:**
    - Integrates seamlessly with other engineering software and PLM systems, enabling data exchange and collaboration across different disciplines.
    - Supports interfaces to ERP and production systems, facilitating a smooth workflow from design to production.
5. **Error Reduction:**
    - Advanced validation tools and error-checking features help identify and correct design issues early in the development process.
    - Reduces the risk of costly errors and rework during manufacturing and installation.
6. **Time Savings:**
    - Automation features, such as automatic wire numbering and cross-referencing, save significant time compared to manual processes.
    - Speeds up the design process, allowing engineers to focus on more complex tasks.

Disadvantages:

1. Cost:
    - EPLAN Electric can be expensive, particularly for small businesses or individual users.
    - Requires investment in software licenses, training, and potential customization.
2. Learning Curve:
    - The software has a steep learning curve due to its complexity and extensive feature set.
    - Requires training and practice to become proficient in using all the available tools effectively.

3. System Requirements:
    - Requires a powerful computer system to run efficiently, especially for large and complex projects.
    - May need regular updates and maintenance to ensure optimal performance.
4. Dependency on Software:
    - Heavy reliance on the software for design processes can be a disadvantage if there are technical issues or software downtime.
    - Backup and data management practices are crucial to prevent data loss.

Conclusion:

EPLAN Electric is a powerful tool for designing electrical circuits, offering numerous advantages such as efficiency, standardization, comprehensive documentation, and error reduction. While it has some drawbacks, including cost and a steep learning curve, the benefits it provides make it a valuable asset for electrical engineers and designers working on complex systems.

 # Motive Circuit System:

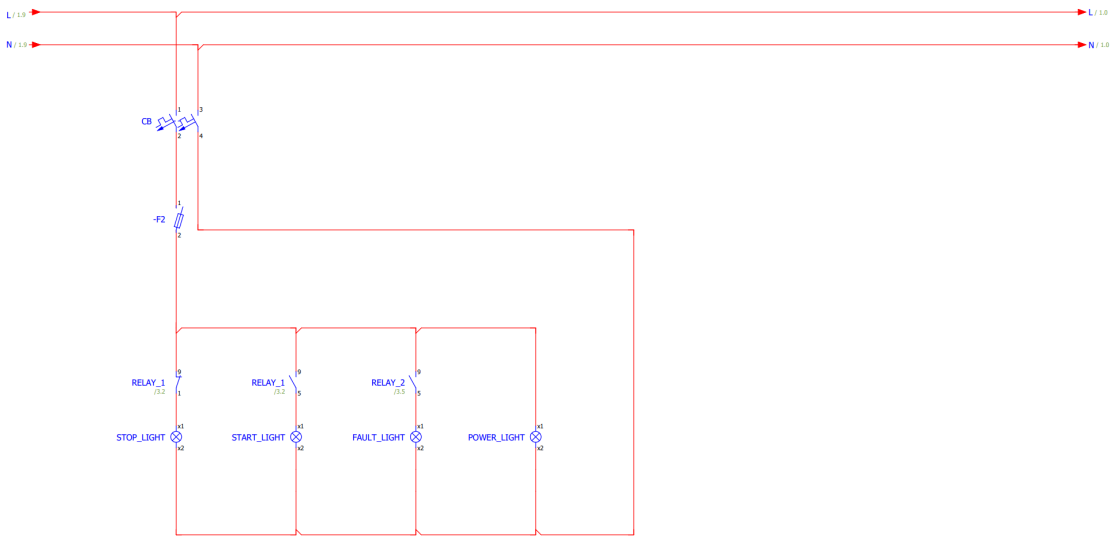

Figure 3.2: Motive Circuit System:

The dynamic circuit uses CB to turn on and off the 220VAC power source.

The phase wire is passed through the fuse to protect the device. The display indicator light is controlled by intermediate contacts. Figure 3.2 depicts a dynamic Motive Circuit System Drawing:

- Power light will be powered directly from the fuse.
- Start indicator operates through the normally open contact 9-5 of Relay 1.
- Stop indicator operates through normally closed contact 9-1 of Relay 1.
- Fault indicator will operate through the normally open contact 9-5 on Relay 2.

 # Control Circuit System

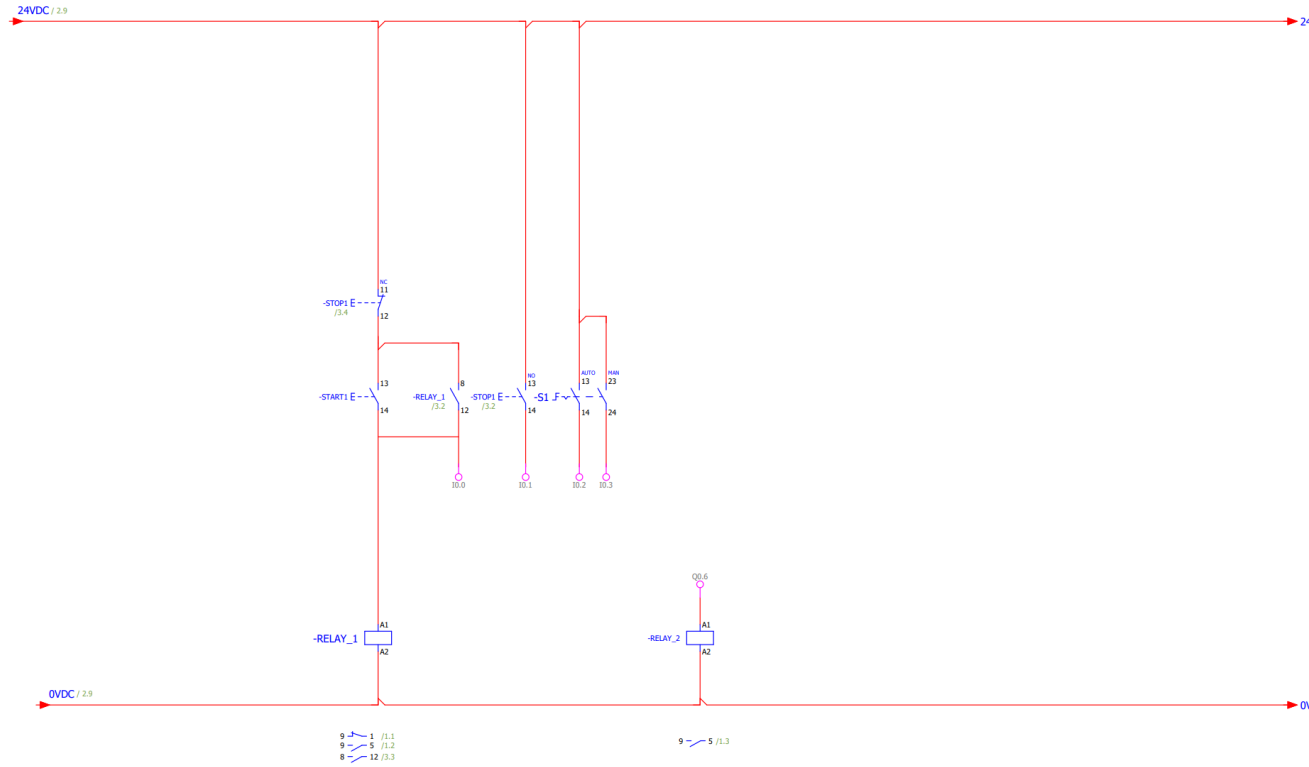

Figure 3.3: Control Circuit System

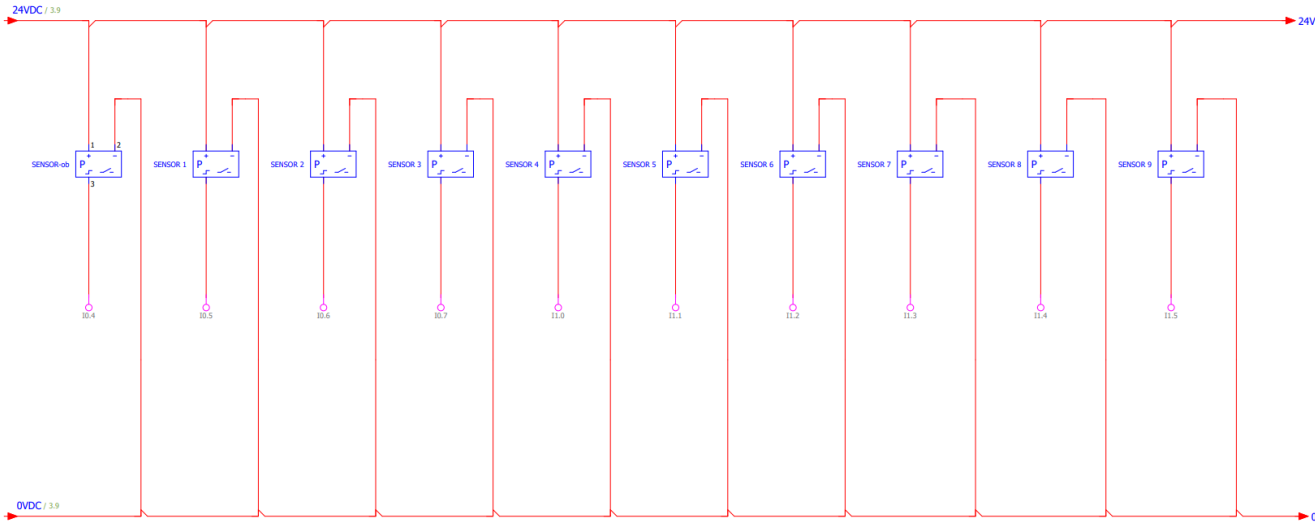

Figure 3.4: Control Circuit System (Sensor)

The control circuit includes Start, Stop buttons and a 3-position Switch for 2 Auto/Manual modes. Figure 3.3 shows a drawing of the system control circuit.

Control circuit for optical sensors. Figure 3.4 shows the system control circuit drawing (optical sensor).

 # The following is the PLC signal convention table:

Input	
I0.0	Start button
I0.1	Stop button
I0.2	Auto switch
I0.3	Manual switch
I0.4	Obstacle sensor
I0.5	Warehouse sensors (1)
I0.6	Warehouse sensors (2)
I0.7	Warehouse sensors (3)
I1.0	Warehouse sensors (4)
I1.1	Warehouse sensors (5)
I1.2	Warehouse sensors (6)
I1.3	Warehouse sensors (7)
I1.4	Warehouse sensors (8)
I1.5	Warehouse sensors (9)

Table 3.1: PLC input signal convention

Output	
Q0.0, Q0.1	X-axis driver
Q0.1, Q0.2	Z-axis driver
Q0.3, Q0.4	Y-axis driver
Q0.5, Q0.6	Conveyor-axis driver
Q0.7	Fault Alarm Light

Table 3.2: PLC output signal convention

PLC Wiring Diagram

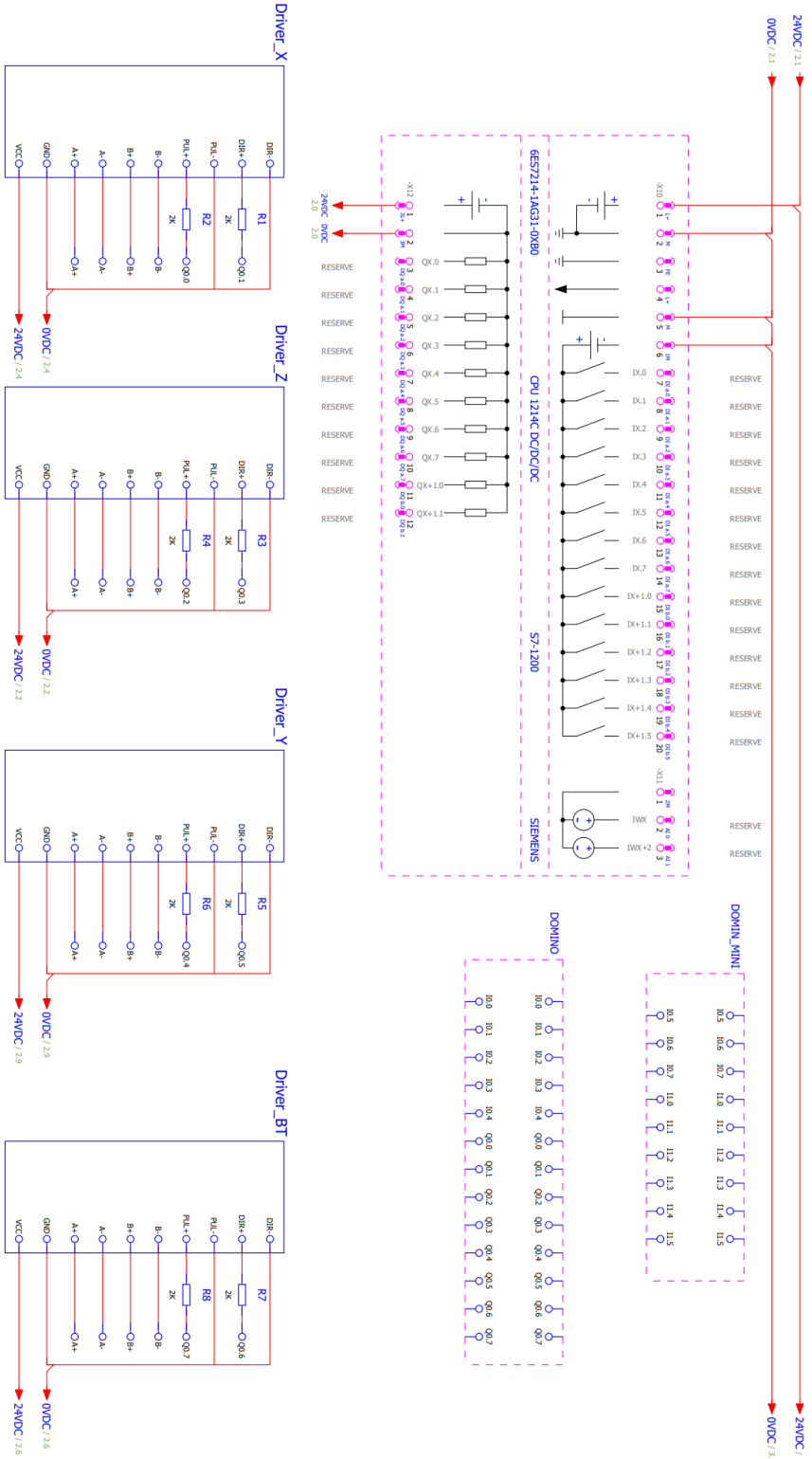

Figure 3.5: PLC Wiring Diagram

3.2 Design and Construction of Electrical Cabinets

Inside Cabinet:

Use an electrical cabinet with dimensions of 300x400x210 mm. Figure 3.6 depicts the layout of the cabinet.

Figure 3.6: Inside Cabinet

Equipment is in the cabinet:

- Honeycomb power supply 24VDC - 5A.
- PLC S7-1214 DC/DC/Dc
- CB 220V - 16A.
- Driver TB6600.
- Intermediate relay.

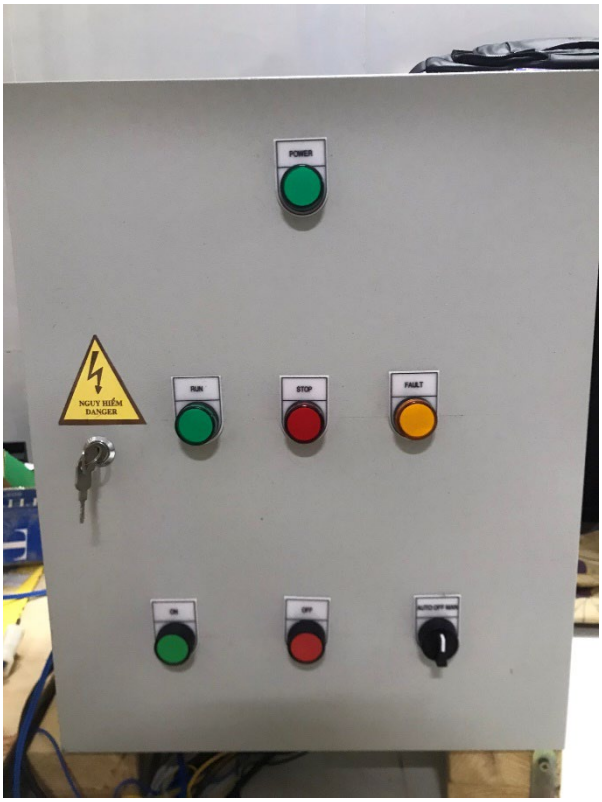

Figure 3.7: Cabinet appearance

 # Outside Cabinet

Figure 3.7 shows the cabinet appearance.

Equipment is outside the cabinet:

- Power indicator light (green).
- ON status indicator light (green).
- Light shows OFF status (red).
- Fault status display light (yellow).
- Start push button (green - 1 NO contact).
- Stop push button (red - 1 NO contact and 1 NC contact).
- Status switch (Auto/Man).

 # Complete cabinets

Figure 3.8 shows the completed cabinet.

Figure 3.8: The completed cabinet

3.3 Design and Construction of the Model

The product warehouse is made from wood and uses L-shaped and straight

shelves to connect. The cells are made the same size.

Warehouse size: 82x62 cm. Figure 3.9 depicts the warehouse sketch. Figure 3.10 Completely constructed warehouse.

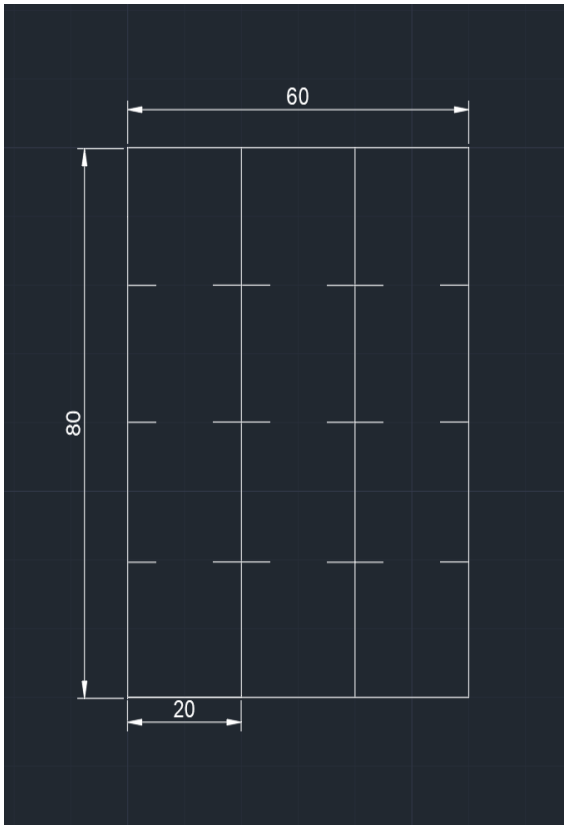

Figure 3.9: Depicts the warehouse sketch

Figure 3.10: Completely constructed warehouse

Objects are made in 3 sizes (low, mid, high) Figure 3.11 depicts the object

Figure 3.11: Depicts the object

Coordinate Forklift:

Axis travel: X: 100 cm; Y: 60 cm; Z: 60 cm

Using stepper motor for 3 axes:

- The X axis has a heavier load than the other axes, so use a motor with 2A current and step angle of 1.8°.

- Y axis uses 1.2A stepper motor, step angle 1.8°.
- Z axis uses 1.8A stepper motor, step angle 1.8°.
- Use 2 100cm sliders as support for the X axis, the belt mounted stepper motor is attached with a belt to move the X axis.
- Lead screw rod 60 cm long used for Z axis.
- The Y axis is combined between motor and belt to move.

Figure 3.12: Coordinate Forklift

3.4 Designing Interface and Programming for Management and Monitoring System

Main Screen:

Figure 3.13 Shows the main monitoring interface of the system

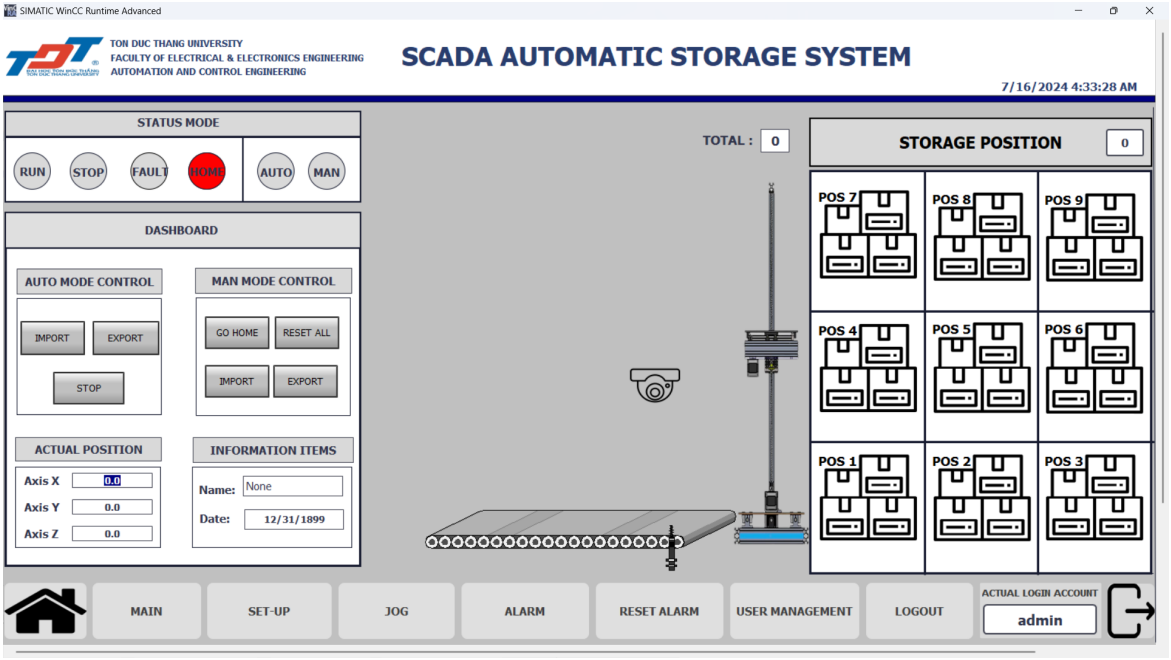

Figure 3.13: Main Screen

- The light displays the status on the status bar.
- Displays the status of conveyors, self-propelled vehicles and the location of packaging boxes for easy monitoring.
- The control panel bar represents the mode control buttons and displays the product name, manufacturing date, and product coordinates.

Setting Screen

This settings screen provides an overview of 5 key information that users can easily monitor and adjust:

1. Start state of the engine and set the coordinate origin:
   - The leftmost position displays the Step Motor's starting parameters.
   - Provides the ability to set the coordinate origin for the lifting axis.
2. Axis setting parameters:

- The next column displays parameters that allow the user to set the position and speed of the axes.

3. Move Job column:
   - This column allows the user to move the axes as desired.
   - Helps perform moving tasks conveniently and accurately.
4. Current position of the axes:
   - Users can track the current location of the system.
5. Control stepper motor speed

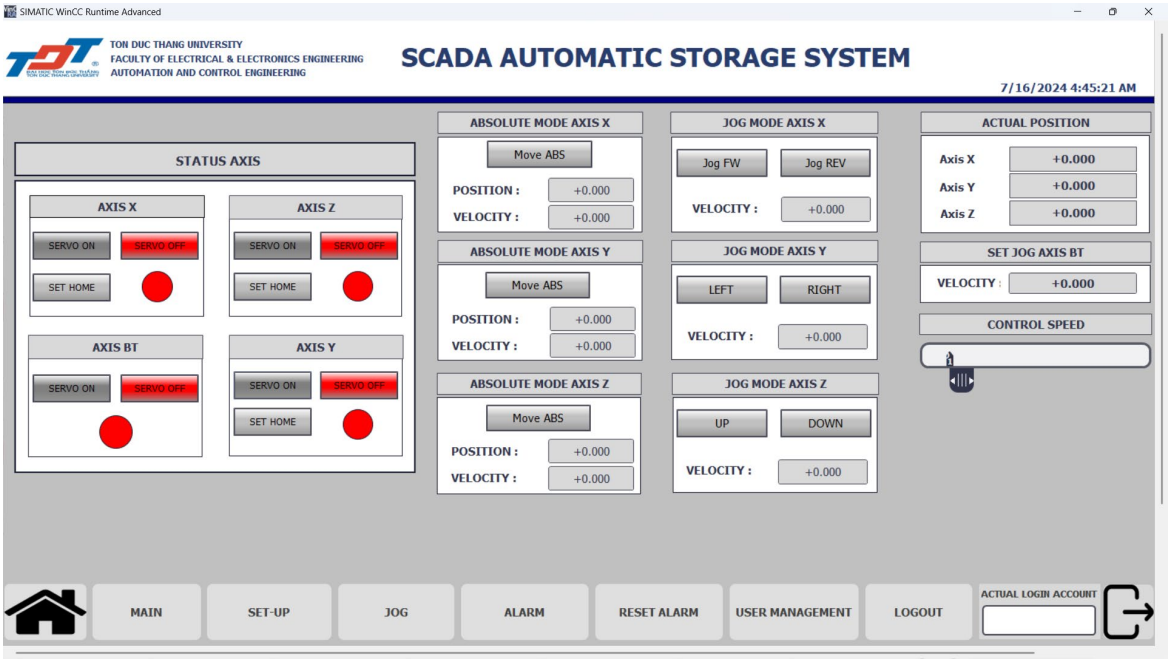

Figure 3.14: Setting Screen

 # Coordinate setting screen

This screen interface allows users to set the coordinates of row cells. Figure 3.15 shows the parameters for setting the import status of the cells. Figure 3.16 shows the export status settings of the cells.

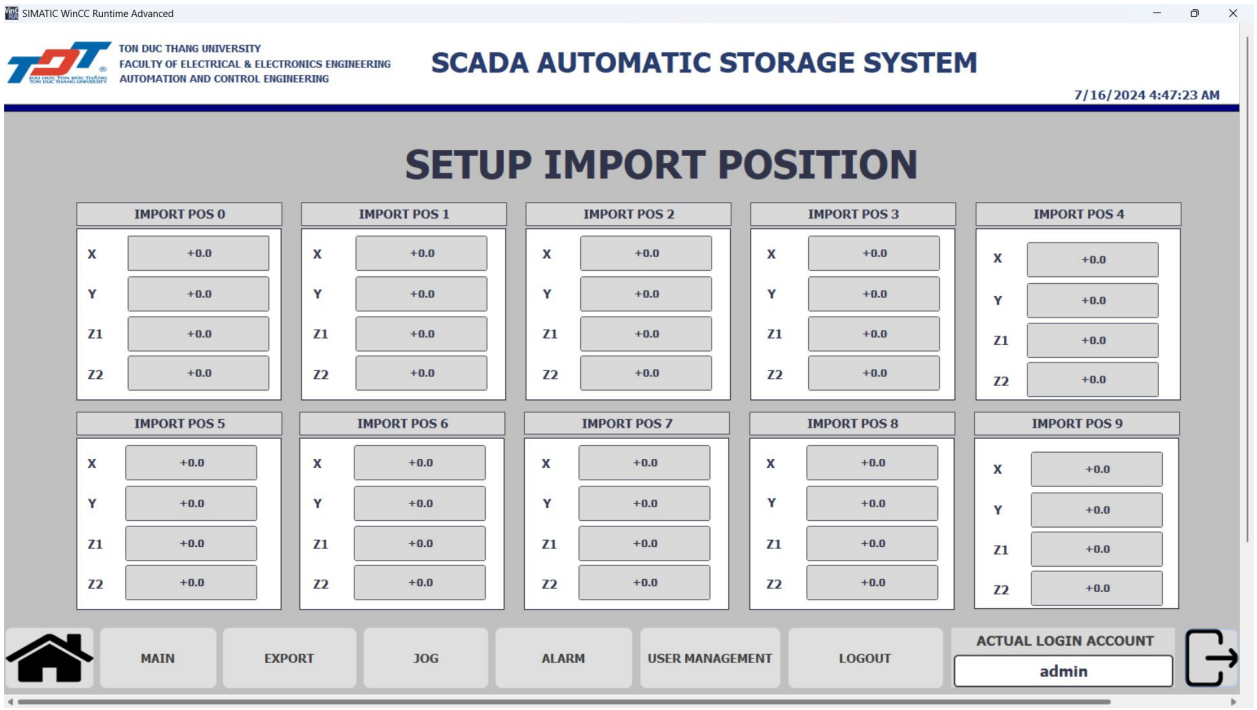

Figure 3.15: Import setting screen

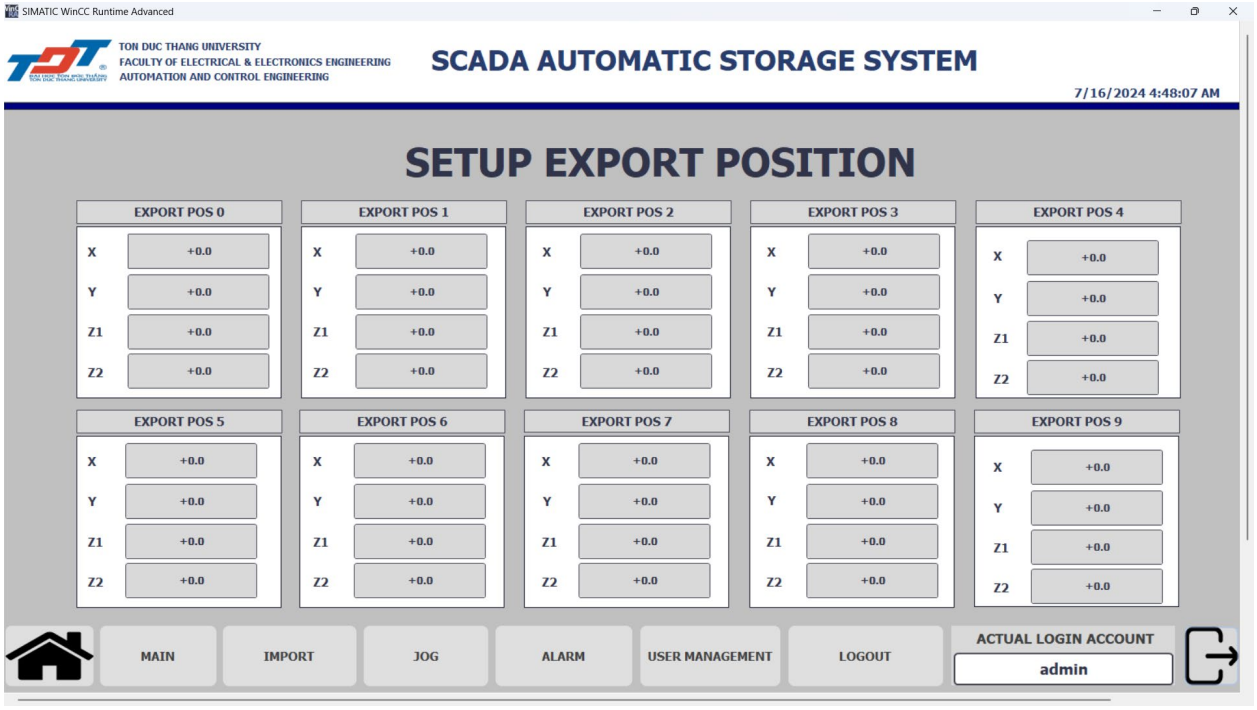

Figure 3.16: Export setting screen

Login object management screen

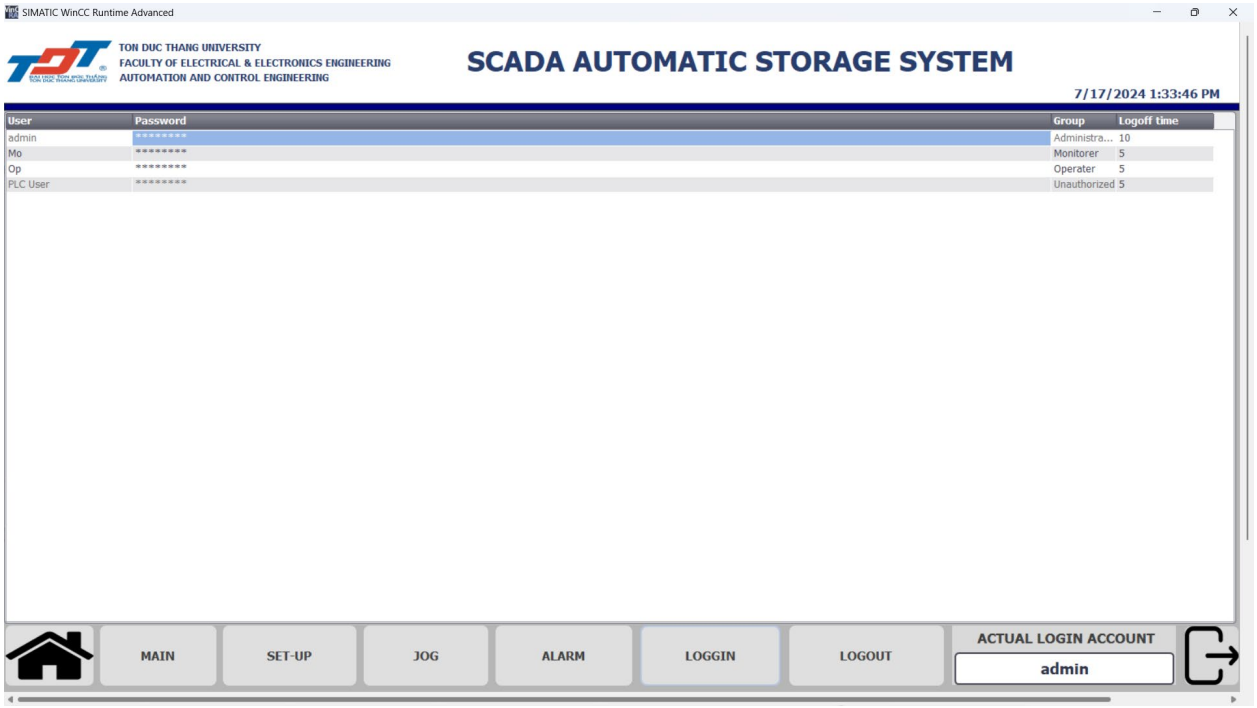

Figure 3.17: Management screen

CHAPTER 4. CONTROL AND ALGORITHMS

4.1 Algorithm Flowchart

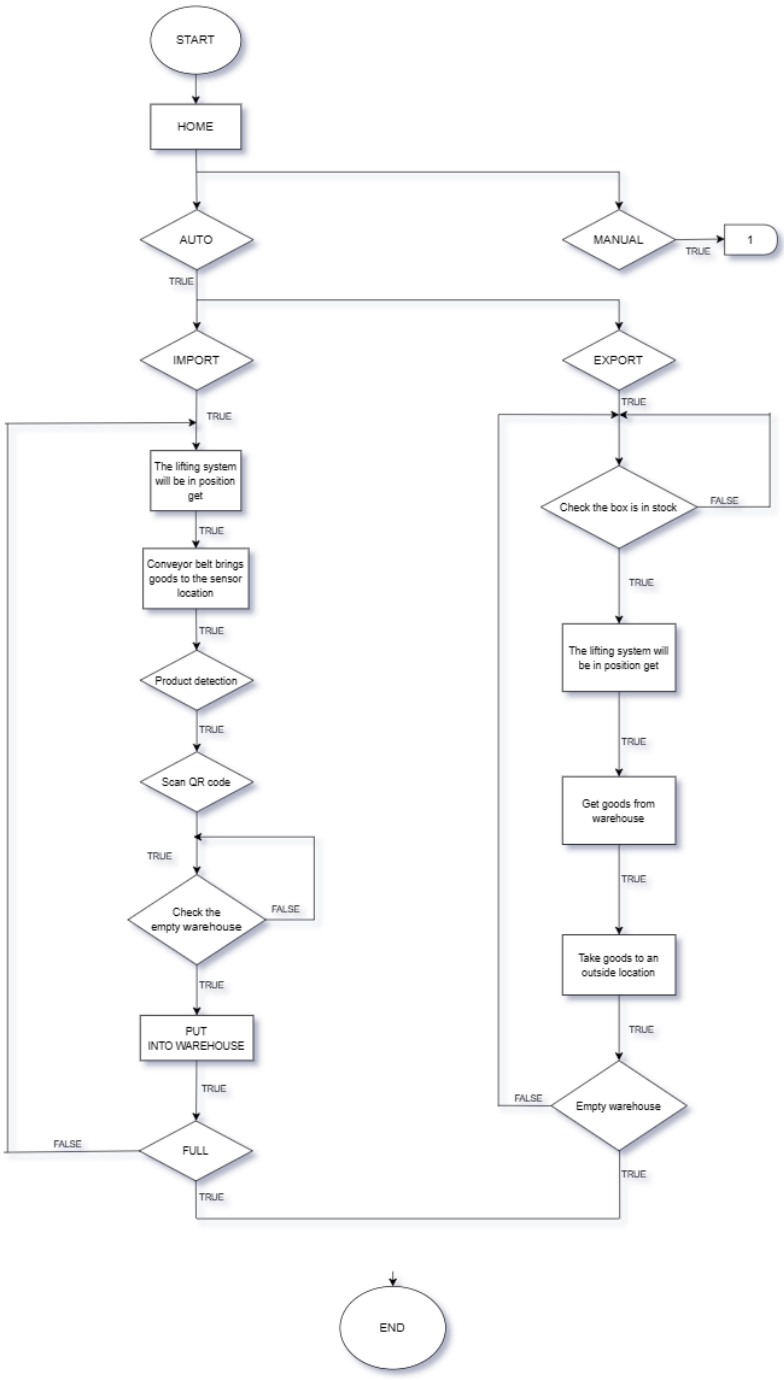

Figure 4.1: Auto Mode Algorithm Flowchart

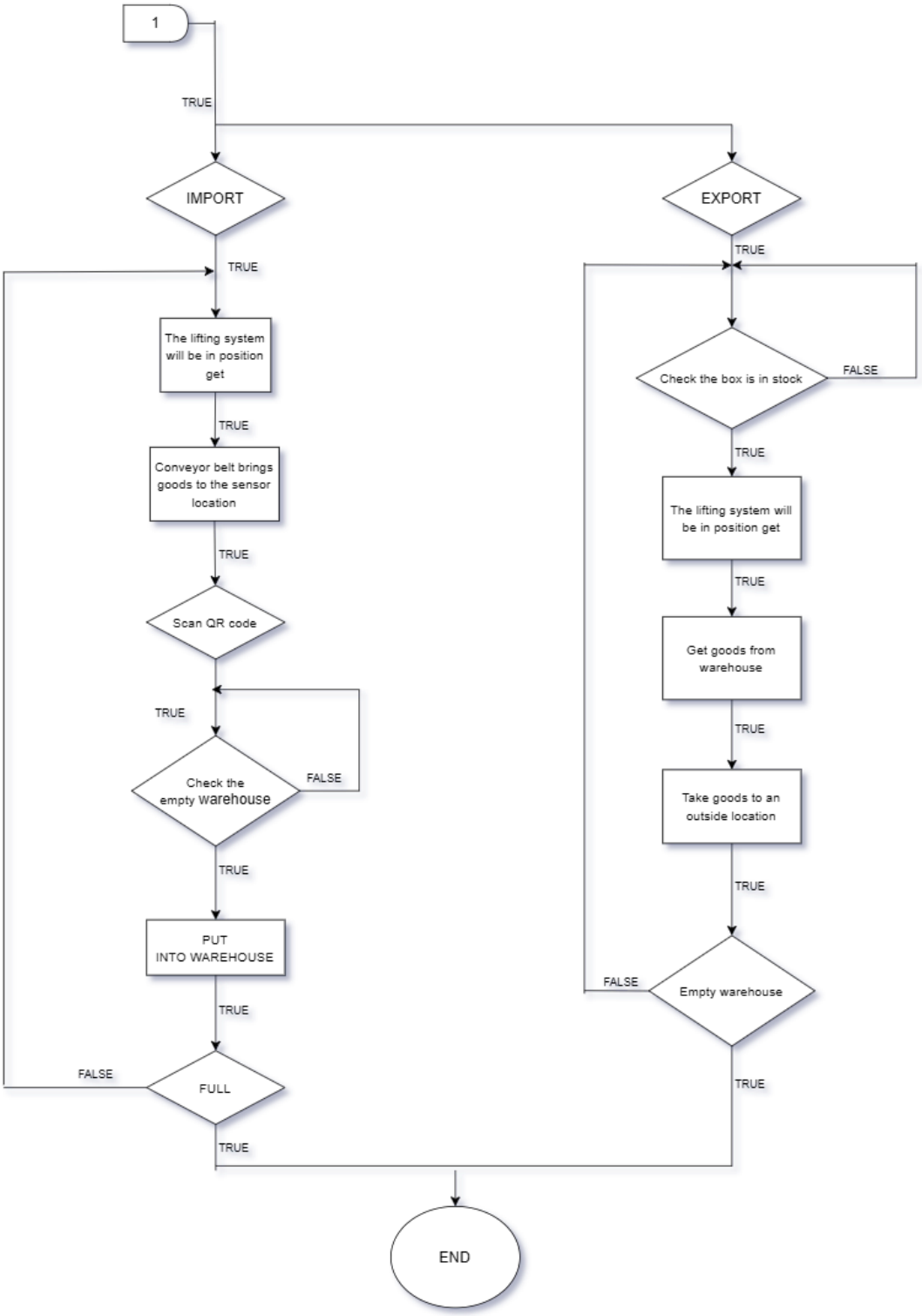

Figure 4.2: Manual Mode Algorithm Flowchart

4.2 Operation Process of PLC Controller

- Power the entire system by turning on the CB.

- Press the Start button on the electrical cabinet to start the stepper motors and start the program.
- There are 2 modes for users to use: Auto and Manual

#Manual Mode

Control the switch button via manual mode on the electrical cabinet.

Import:

- Select the location of the cell to be imported on the SCADA interface.
- Press Import button to import the product into the warehouse.
- The cell location in the warehouse will light up green (notifying that there is stock).

Export:

- Select the location of the cell to be exported on the SCADA interface.
- Press the Export button to export the product.
- If you select a cell that has no stock, the screen will display an error and the Fault light on the cabinet will beep and flash yellow.

#Auto Mode

Control the switch button through the automatic mode on the electrical cabinet.

Import:

- Press the input button on the interface, the system will automatically process the preset input operations.
- If the warehouse is full, the screen will display a message and the Error light on the cabinet will sound and flash yellow.

Export:

- Press the Output button on the interface, the system will automatically output goods from the top cell from high to low.
- When the warehouse is out of stock, the system will display the warning "Out of stock" on the SCADA interface.

4.3 Principle of reading QR code and pushing data to Database

Connect to PLC

Use S7.NET library to set up and connect to PLC via IP address "192.168.0.1".

Operating principle:

Use the algorithm of the ZXing library to read the QR code, then push the value to the PLC to classify the product. When we import or export, the information will be saved and pushed to the database for storage to easily manage the product.

4.4 Operating Principles of the System

- Use the driver to control the 3 axes X, Y, Z.
- The axes are set to pulse parameters on the driver of 1600 pulses/s and basic step angle of 1.8°.
- Step angle = Basic step angle/ Micro step = 1.8°/4 = 0.45°.
- Therefore, 1 pulse rotates 0.45° and needs 1600 pulses to rotate 1 revolution.
- Before entering the algorithm, we need to determine the position of coordinate origin Home for 3 axes.
- In this project, I do not use a limit switch to Set Home. Instead, I use a memory variable.
- Then, we need to know the exact coordinates of the nine shelves. We use the Move Job block of the Siemens support library to move the axes to the desired position.

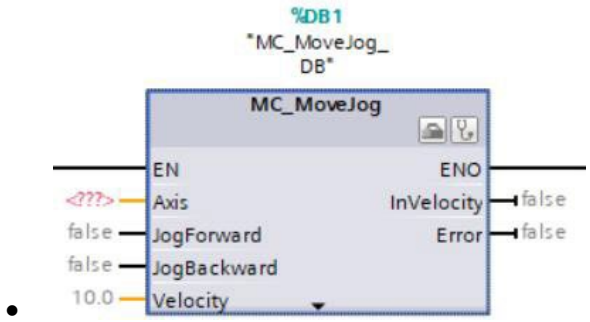

Figure 4.3: Move Jog Block

After determining the necessary coordinates, we will save the coordinates of the 3 axes into the Data Block. When the desired row cell is called, we will use the Move block to write the value into the Move Absolute block. This block allows the stepper motor to move to the predetermined coordinates.

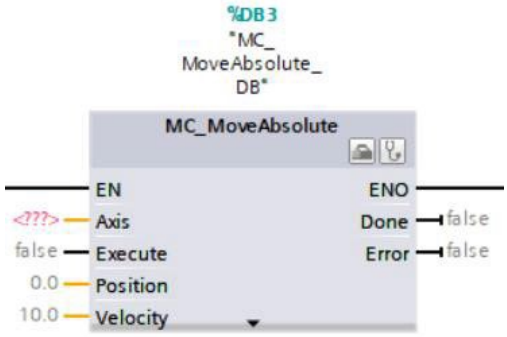

Figure 4. 4: Move Absolute Block

And all necessary coordinates use PLC's internal memory variables.

 # The process of returning home

The Y axis always returns to the origin coordinates first, then the X and Y axes return to the origin, to avoid the case of the Y axis colliding with obstacles.

 # Manul mode.

Each cell has the necessary coordinates assigned. When we click on any cell, the coordinates will be immediately recorded in the Move Absolute block. Next, when we click Import, the axes will automatically move according to the preset coordinates and always follow the principle that the Y axis always moves before the X and Z axes. The Y axis must complete the cycle before the remaining axes are allowed to operate.

When the item is placed in position, the selected cell will light up green using the optical sensor to create this effect. Similarly, for Export, we also select the cell to export, and the coordinates of the exported item will also be recorded in the Move Absolute block to move, after the item is removed from the shelf, the optical sensor will turn off.

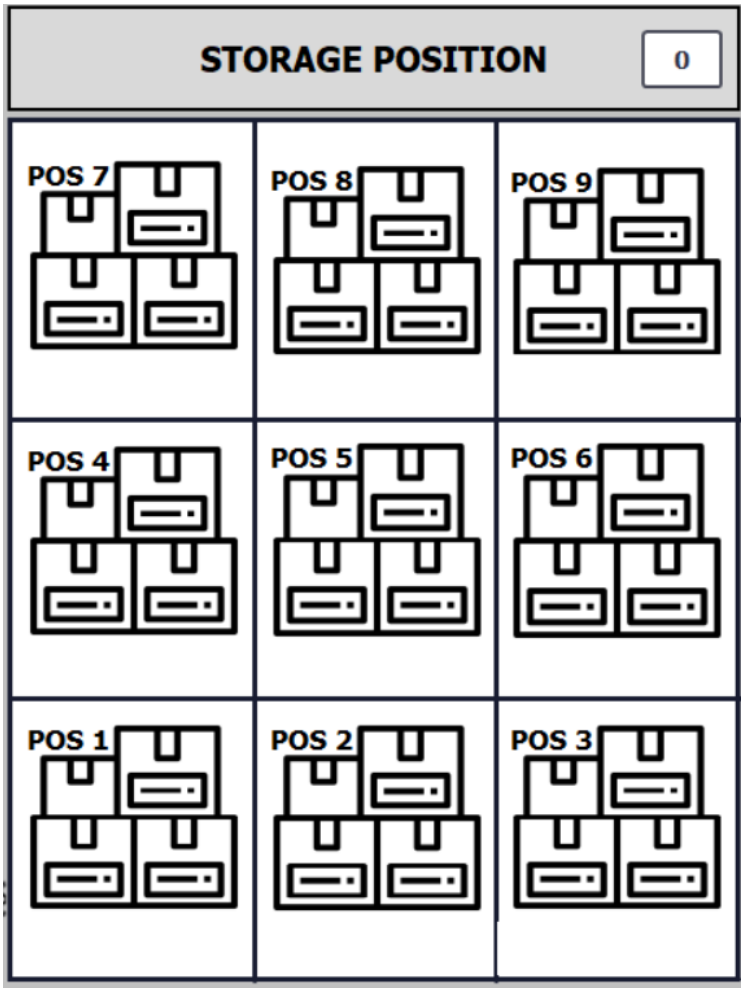

Figure 4.5: Row position

 # Mode Auto

The image processing part will send the value from the C# program to the PLC, we get that value and compare it with a waiting variable on the PLC, if the value matches the variable on the PLC, we will write the coordinates into the Move Absolute block to move. The Export part will automatically export from the highest cell to the lowest cell. Any cell with stock will be exported to the waiting shelf position. And there is an export mode according to the size of the product.

 # Database

When receiving imported goods, it will be stored in the PLC's internal memory variable and will be read by C# and the values will be pushed to the Database on Microsoft SQL software to be stored for management purposes.

System errors:

- In Manual mode, when we import a cell that already has goods and we still select that position, an error will be reported on the HMI and the Fault light on the cabinet will be displayed.
- And similarly, when we export in Manual mode when there are no goods and we still select Export, an error will also be reported.
- When Import is in auto mode, the rows and cells will be automatically arranged based on the received value.
- In automatic mode, select the output mode by row according to size, when the position containing the box of the same size is empty, an alarm will be sent to notify that this row is empty.
- When in manual mode, if the wrong position of the box with the specified size is selected, an error will be sent to the alarm for the operator to recognize.

CHAPTER 5. CONCLUSION

5.1 Conclusion on Data Read/Write Capability between C# and PLC

Through the process of working and testing, the communication between the C# program and the PLC through the S7.NET library took place smoothly and stably. However, during the data storage process, problems occur.

5.2 Conclusion about system operability

The system has worked as required. The axes use stepper motors to help move positions accurately. Stability and accuracy during operation are the system's strengths.

5.3 Conclusion interactivity and display

#Advantage

- The buttons and indicator lights on the electrical cabinet are conveniently designed, making it easy for the operator to interact with the system.
- Switching between Auto and Manual modes directly increases the operator's ability to operate optimally.
- The monitoring interface is fully designed, intuitive, and easy to understand, creating favorable conditions for users.
- Manage products when they are being released and imported.
- Can change the speed as desired

#Disvantage

- Data upload to database is not smooth.
- There is no remote monitoring and control system.

5.4 Development

- To improve the system, it is necessary to design the monitoring and management interface to become more intuitive. In addition, it is necessary to improve and optimize data management more effectively.

- Further develop Web Server for remote monitoring and control.
- Improve, optimize and develop warehouse management.

REFERENCES

[1] Product Details - Industry Mall - Siemens WW

[2] 2 Phase Hybrid Stepper Motor (yumpu.com)

[3] TB6600 Stepper Motor Driver (bulkman3d.com)

[4] Cảm biến khoảng cách E3F-DS30P1 PNP 10-30cm | Công ty cổ phần Linh Kiện Việt Nam (linhkienvietnam.vn)

[5] Webcam Xiaomi Xiaovv 6320S Full HD 1080p - USB (ankhang.vn)

[6] Catalogue MCB LS mới năm 2024 (thietbibenthanh.com)

[7] RXM2LB2P7 - miniature, Harmony Electromechanical Relays, 5A, 2CO, with LED, 230V AC | Schneider-electric Việt Nam (se.com)

[8] Fuses | Chint Global

[9] Nguồn tổ ong 24V 5A - Nshop (nshopvn.com)

[10] 636791138282392049691459 6.pdf (siemens.com.cn)

[11] GitHub - zxing/zxing: ZXing ("Zebra Crossing") barcode scanning library for Java, Android

[12] NuGet Gallery | AForge.Video 2.2.5

[13] EPLAN Electric P8 (eplan-software.com)

APPENDIX - PROGRAM ON VISUAL STUDIO SOFTWARE

```
using System;
using System.CodeDom;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;
using AForge.Video;
using AForge.Video.DirectShow;
using S7.Net; // Thu vien ket noi plc
using ZXing;
namespace test_SQL_part_2
{
   public partial class Form1 : Form
   {
      private FilterInfoCollection capture;   // List
      private VideoCaptureDevice camera;      // Video capture device
      private Plc PLC_S71200;                 // PLC instance
      private CancellationTokenSource cancellationToken;
      private readonly object lockObject = new object();
      private System.Windows.Forms.Timer qrCodeTimer;
      // Các biến toàn cục
      private string name = "";
      private string status = "";
```

```
private string location = "";
private string size = "";
private DateTime date = DateTime.MinValue;
public Form1()
{
    InitializeComponent();
    capture = new FilterInfoCollection(FilterCategory.VideoInputDevice); // Initialize video capture devices
    foreach (FilterInfo item in capture)
    {
        cb_box.Items.Add(item.Name); // Add each connected camera to the combo box
    }
    PLC_S71200 = new Plc(CpuType.S71200, "192.168.0.1", 0, 0); // Initialize PLC connection
    qrCodeTimer = new System.Windows.Forms.Timer();
    qrCodeTimer.Interval = 10000; // Đặt khoảng thời gian là 10 giây
    qrCodeTimer.Tick += QrCodeTimer_Tick; // Thêm trình xử lý sự kiện tick
}
private void QrCodeTimer_Tick(object sender, EventArgs e)
{
    // Xóa kết quả mã QR nếu không có mã mới được quét trong 10 giây
    txt_box.Text = "";
}
private void Form1_Load(object sender, EventArgs e)
{
    if (PLC_S71200.Open() == ErrorCode.NoError)
    {
        timer1.Enabled = true;
        status_note.Text = "Kết nối thành công";
    }
    else
```

```
        {
            status_note.Text = "Kết nối thất bại";
            MessageBox.Show("Không thể kết nối PLC", "Lỗi", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
    private void button_start_Click(object sender, EventArgs e)
    {
        cancellationToken = new CancellationTokenSource();
        var sourceToken = cancellationToken.Token;

        Read_Data_QR(sourceToken); // Bat day QR code
        Write_Data(sourceToken); // bat dau viet data toi PLC
        if (cb_box.Text == "")
        {
            MessageBox.Show("Vui lòng chọn camera", "Lỗi", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
        else
        {
            camera = new VideoCaptureDevice(capture[cb_box.SelectedIndex].MonikerString);
// Hien thi chon nguon camera
            camera.NewFrame += Camera_NewFrame; // Dang ki nguon camera
            TopMost = true;
            camera.Start(); // Bat dau khoi dong camera
        }
        // Check PLC connection
        if (PLC_S71200.IsConnected)
        {
            PLC_S71200.Write("M1.7", true);
            Thread.Sleep(100); // Delay
```

```
            PLC_S71200.Write("M1.7", false);
            MessageBox.Show("Kết nối thành công", "Thông báo", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            timer1.Enabled = true; // Enable timer
            timer1.Interval = 1000; // Set timer interval
        }
        else
        {
            status_note.Text = "Kết nối thất bại";
            PLC_S71200.Close();
            MessageBox.Show("Không thể kết nối PLC", "Lỗi", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
    private void Camera_NewFrame(object sender, NewFrameEventArgs eventArgs)
    {
        // Cap nhap hop anh cua camera da chon tu truoc
        if (pic_box.InvokeRequired)
        {
            pic_box.Invoke((MethodInvoker)delegate
            {
                pic_box.Image = (Bitmap)eventArgs.Frame.Clone();
            });
        }
        else
        {
            pic_box.Image = (Bitmap)eventArgs.Frame.Clone();
        }
    private void button_stop_Click(object sender, EventArgs e)
    {
        // Stop camera
```

```
    try
    {
        if (camera != null && camera.IsRunning)
        {
            camera.SignalToStop();
            camera.WaitForStop();
            camera = null;
            pic_box.Image = null;
        }
    }
    catch
    {
    }
    finally
    {
      Application.Exit();
    }
}
public void Read_Data_QR(CancellationToken sourceToken)
{
    // Task để liên tục đọc mã QR
    Task.Factory.StartNew(() =>
    {
        BarcodeReader reader = new BarcodeReader();
        while (!sourceToken.IsCancellationRequested)
        {
            Thread.Sleep(300);
            pic_box.BeginInvoke((MethodInvoker)delegate
            {
                if (pic_box.Image != null)
```

```
            {
                try
                {
                    Result result = reader.Decode((Bitmap)pic_box.Image);
                    if (result != null)
                    {
                        txt_box.Text = result.Text; // Hiển thị dữ liệu mã QR
                        qrCodeTimer.Stop(); // Dừng timer
                        qrCodeTimer.Start(); // Khởi động lại timer
                    }
                }
                catch (Exception ex)
                {
                }
            }
        });
    }
}, sourceToken);
}
public void Write_Data(CancellationToken sourceToken)
{
    // Ghi du lieu lien tuc vao PLC
    Task.Factory.StartNew(() =>
    {
        while (!sourceToken.IsCancellationRequested)
        {
            Thread.Sleep(300);
            Write_Data_To_PLC(); // Write data to PLC
        }
    }, sourceToken);
```

```
}
void Write_Data_To_PLC()
{
   try
   {
      string qrText = txt_box.Text;
      if (!string.IsNullOrEmpty(qrText) && qrText.Length > 2)
      {
         //  first digit, second digit, and the remaining part
         string firstDigit = qrText.Substring(0, 1);
         string secondDigit = qrText.Substring(1, 1);
         string remainingDigits = qrText.Substring(2);
         // Convert first digit, second digit, and remaining part to integers
         if (ushort.TryParse(firstDigit, out ushort firstDigitValue) &&
            ushort.TryParse(secondDigit, out ushort secondDigitValue) &&
            uint.TryParse(remainingDigits, out uint remainingDigitsValue))
         {
            // Write first digit to DB2.DBW148 PLC
            PLC_S71200.Write("DB2.DBW148", firstDigitValue);
            // Write second digit to DB2.DBW152 PLC
            PLC_S71200.Write("DB2.DBW146", secondDigitValue);
            // Write remaining part to DB2.DBW150 PLC
            PLC_S71200.Write("DB2.DBW150", remainingDigitsValue);
         }
      }
   }
   catch (Exception ex)
   {
      MessageBox.Show("Error writing to PLC: " + ex.Message);
   }
```

```
    }
    void Read_Data_From_PLC_Name()
    {
        int dbNumber = 4; // Data Block number
        int dbstartAddr = 770; // Start address within the Data Block
        int dataLength = 50; // Max data length to read
        byte[] data = PLC_S71200.ReadBytes(DataType.DataBlock, dbNumber, dbstartAddr,
dataLength);
        if (data.Length >= 2)
        {
            byte maxLength = data[0]; // Maximum length of the string
            byte actualLength = data[1]; // Actual length of the string
            if (data.Length >= actualLength + 2)
            {
                byte[] dataBytes = new byte[actualLength];
                Array.Copy(data, 2, dataBytes, 0, actualLength);
                name = Encoding.ASCII.GetString(dataBytes); // Convert byte array to string
            }
        }
    }
    void Read_Data_From_PLC_Status()
    {
        int dbNumber = 4; // Data Block number
        int dbstartAddr = 258; // Start address within the Data Block
        int dataLength = 50; // Max data length to read
        byte[] data = PLC_S71200.ReadBytes(DataType.DataBlock, dbNumber, dbstartAddr,
dataLength);
        if (data.Length >= 2)
        {
            byte maxLength = data[0]; // Maximum length of the string
            byte actualLength = data[1]; // Actual length of the string
```

```
            if (data.Length >= actualLength + 2)
            {
                byte[] dataBytes = new byte[actualLength];
                Array.Copy(data, 2, dataBytes, 0, actualLength);
                status = Encoding.ASCII.GetString(dataBytes); // Convert byte array to string
            }
        }
    }
    void Read_Data_From_PLC_Location()
    {
        int dbNumber = 4; // Data Block number
        int dbstartAddr = 514; // Start address within the Data Block
        int dataLength = 50; // Max data length to read

        byte[] data = PLC_S71200.ReadBytes(DataType.DataBlock, dbNumber, dbstartAddr,
dataLength);
        if (data.Length >= 2)
        {
            byte maxLength = data[0]; // Maximum length of the string
            byte actualLength = data[1]; // Actual length of the string
            if (data.Length >= actualLength + 2)
            {
                byte[] dataBytes = new byte[actualLength];
                Array.Copy(data, 2, dataBytes, 0, actualLength);
                location = Encoding.ASCII.GetString(dataBytes); // Convert byte array to string
            }
        }
    }
    void Read_Data_From_PLC_Date()
    {
        string test;
```

```
        int dbNumber = 4; // Data Block number
        int dbstartAddr = 1026; // Start address within the Data Block
        int dataLength = 50; // Max data length to read
        byte[] data = PLC_S71200.ReadBytes(DataType.DataBlock, dbNumber, dbstartAddr,
dataLength);
        if (data.Length >= 2)
        {
           byte maxLength = data[0]; // Maximum length of the string
           byte actualLength = data[1]; // Actual length of the string
           if (data.Length >= actualLength + 2)
           {
              byte[] dataBytes = new byte[actualLength];
              Array.Copy(data, 2, dataBytes, 0, actualLength);

              test = Encoding.ASCII.GetString(dataBytes); // Convert byte array to string
              if (double.TryParse(test, out double daysSince1990))
              {
                 date = ConvertToDate(daysSince1990);
              }
              //txt_date.Text = date.ToString("dd/MM/yyyy");
           }
        }
     }
     void Read_Data_From_PLC_Size()
     {
        int dbNumber = 4; // Data Block number
        int dbstartAddr = 2; // Start address within the Data Block
        int dataLength = 50; // Max data length to read

        byte[] data = PLC_S71200.ReadBytes(DataType.DataBlock, dbNumber, dbstartAddr,
dataLength);
```

```
        if (data.Length >= 2)
        {
            byte maxLength = data[0]; // Maximum length of the string
            byte actualLength = data[1]; // Actual length of the string
            if (data.Length >= actualLength + 2)
            {
                byte[] dataBytes = new byte[actualLength];
                Array.Copy(data, 2, dataBytes, 0, actualLength);
                size = Encoding.ASCII.GetString(dataBytes); // Convert byte array to string
            }
        }
    }
    DateTime ConvertToDate(double daysSince1990)
    {
        DateTime baseDate = new DateTime(1990, 1, 1);
        return baseDate.AddDays(daysSince1990);
    }
    //------------------------------------------------------------------------------
    // SQL TO DATA TABLE FUNCTIONS
    //------------------------------------------------------------------------------
    void SQL_cnn()
    {
        string connectstring = @"Data Source=DAN\SQLEXPRESS;Initial
Catalog=Quan_li_kho;Persist Security Info=True;User ID=nekohimi;Password=trungdan123";
        SqlConnection conn; // ket noi sql
        SqlCommand cmd; // doc thu vien
        SqlDataAdapter adt;
        DataTable data = new DataTable();
        conn = new SqlConnection(connectstring);
        try
        {
```

```
            conn.Open();
            cmd = new SqlCommand("select * from DATN", conn);
            adt = new SqlDataAdapter(cmd);
            SqlDataReader dr = cmd.ExecuteReader();
            data.Load(dr);
            dataGridView1.DataSource = data;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.ToString());
            conn.Close();
        }
    }
    void insert_SQL()
    {
        string connectstring = @"Data Source=DAN\SQLEXPRESS;Initial
Catalog=Quan_li_kho;Persist Security Info=True;User ID=nekohimi;Password=trungdan123";
        string table_name = "DATN"; // table SQL
        SqlConnection conn = new SqlConnection(connectstring);
        SqlDataReader docdulieu;
        var tag = (bool)PLC_S71200.Read("DB2.DBX158.0"); //read trigger
        bool tag_trigger = tag;
        bool tag_trigger_done = false;

        try
        {
            string query = "INSERT INTO " + table_name + " (Name, Location, NSX, Status, Size,
Date_time) VALUES (@Name, @Location, @NSX, @Status, @Size, @Datetime)";

            if (tag_trigger == true && tag_trigger != tag_trigger_done)
            {
```

```
            conn.Open();
            using (SqlCommand com = new SqlCommand(query, conn))
            {
               com.Parameters.Add("@Name", SqlDbType.NVarChar).Value = name;
               com.Parameters.Add("@Location", SqlDbType.NVarChar).Value = location;
               com.Parameters.Add("@NSX", SqlDbType.Date).Value = date;
               com.Parameters.Add("@Status", SqlDbType.NVarChar).Value = status;
               com.Parameters.Add("@Size", SqlDbType.NVarChar).Value = size;
               com.Parameters.Add("@Datetime", SqlDbType.DateTime).Value =
DateTime.Now;
               com.ExecuteNonQuery();
            }
            conn.Close();
            tag_trigger_done = tag_trigger;
         }
      }
      catch (Exception ex)
      {
         MessageBox.Show(ex.ToString());
         conn.Close();
         tag_trigger = false;
      }
   }
   private void btn_data_show_Click(object sender, EventArgs e)
   {
      SQL_cnn();
   }
   private void timer1_Tick_1(object sender, EventArgs e)
   {
      Read_Data_From_PLC_Status();
      Read_Data_From_PLC_Location();
```

```
            Read_Data_From_PLC_Date();
            Read_Data_From_PLC_Name();
            Read_Data_From_PLC_Size();
            insert_SQL();
        }
        private void btn_trigger_off_Click_1(object sender, EventArgs e)
        {
            Report_Form report_Form = new Report_Form();
            report_Form.ShowDialog();
        }
    }
}
```