REPORT 4 : SOFTWARE AND HARDWARE ASPECTS OF PLC INTERFACING WITH MICROCONTROLLERS.

GROUP 4

MCTA 3203

SEMESTER 1 2024/2025

MECHATRONICS SYSTEM INTEGRATION

DATE OF SUBMISSION: 13 NOVEMBER 2024


| NO | NAME | MATRIC NUMBER |
|---|---|---|
| 1. | IRDINA NABIHAH BINTI MOHD NAZRI | 2214772 |
| 2. | KHALISAH AMANI BINTI YAHYA AZMI | 2218184 |
| 3. | LUQMAN AZFAR BIN AZMI | 2219857 |
| 4. | MOHAMAD NASRI BIN MOHAMAD NAZRI | 2219879 |
| 5. | MUHAMAD NURHAKIMIE THAQIF BIN ABDULLAH | 2213217 |

## TABLE OF CONTENT

## INTRODUCTION

A Programmable Logic Controller (PLC) is an industrial-grade computer designed to control automated processes in various applications, such as manufacturing, robotics, and machinery operation. Known for their robustness, PLCs are built to withstand harsh environments, including extreme temperatures, dust, and electrical noise, making them ideal for industrial use. They operate in real-time, processing inputs from sensors and generating outputs to control actuators with precision. The hardware of a PLC typically includes a central processing unit (CPU) for executing control logic, a power supply, input/output (I/O) modules for interfacing with external devices, communication ports for system integration, and memory for storing programs and configuration data. On the software side, PLCs are programmed using specialized tools and standardized languages such as Ladder Diagram (LD), Structured Text (ST), and Function Block Diagram (FBD), adhering to IEC 61131-3 standards. They support various communication protocols like Modbus and Ethernet/IP for seamless integration with other systems. PLCs are widely used in industries such as manufacturing, process control, building automation, and energy management, providing scalable and reliable solutions for complex automation tasks.

## ABSTRACT

This experiment explores the connection between Programmable Logic Controllers (PLC) and microcontrollers with a focus on the OpenPLC Editor for coding and interfacing. PLC are very important in industrial automation because they enable the exact control of electromechanical processes. In this experiment, a ladder diagram was created in the OpenPLC Editor simulating a blinking LED, and it was then transferred to an Arduino microcontroller for hardware testing. As a result of the modifications, a timer function was introduced to control LED blinking intervals, and enhance understanding of timing and control elements in ladder logic. Also, a Start-Stop control circuit was designed and tested using OpenPLC which shows the practical application of PLCs in managing sequential operations in automated systems. This laboratory work gives the opportunity to learn more about PLC programming, interfacing protocols, and microcontroller-based automation which are important skills for industrial automation tasks.

**EXPERIMENT 5: Controlling and automating electromechanical processes in manufacturing plants, machinery, and various other applications using PLC.**

## MATERIALS AND EQUIPMENT

- OpenPLC Editor software
- Arduino Board
- 2 Push Button Switches
- Jumper Wires
- LED
- Resistors
- Breadboard

## EXPERIMENTAL SETUP

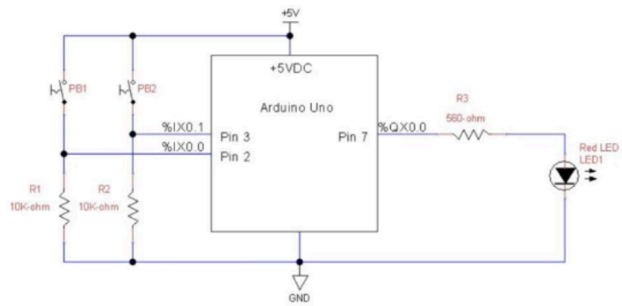
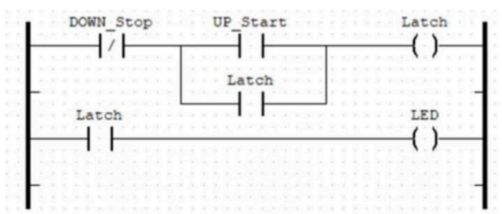



Fig. 4: Start-Stop Control Circuit




| # | Name | Class | Type | Location | Documentation |
|---|---|---|---|---|---|
| 1 | UP_Start | Local | BOOL | %IX0.0 | Will Turn ON the LED |
| 2 | DOWN_Stop | Local | BOOL | %IX0.1 | Will Turn OFF the LED |
| 3 | Latch | Local | BOOL | | Latch Bit Instruction |
| 4 | LED | Local | BOOL | %QX0.0 | Output indicator for Latch_Unlatch Program |

Fig. 5: Ladder Diagram for the Start-Stop Control Circuit

1. Create the ladder diagram shown in Fig. 5.

2. Specify all variables used in the ladder diagram.

3. Compile and simulate the ladder diagram in OpenPLC Editor.

4. Upload the ladder diagram to the Arduino board.

5. Ensure to select correct COM port number and all pin association between the

OpenPLC variables and Arduino board.

6. Build the circuit as shown in Fig. 6.

7. Test the functionality.

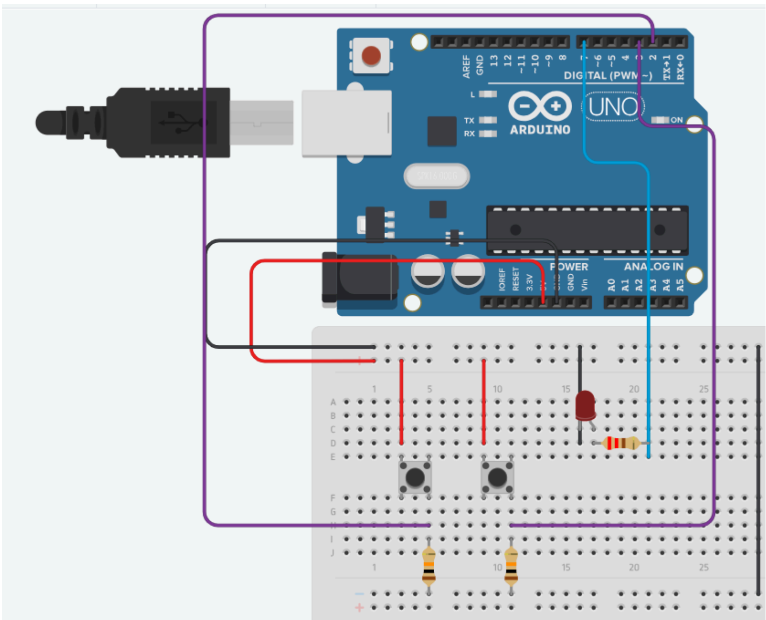
## METHODOLOGY

The appropriate methodology for conducting the experiment involves some major steps that include both integration of software and hardware.

1. **Software Setup**: Download OpenPLC Editor with which you will create and simulate the ladder diagram. You can download it from the link provided in the document.
2. **Develop the Ladder Diagram**: Once the OpenPLC Editor is opened, a new project should be created and the ladder diagram should be developed according to the instructions given herein. Power rails, contacts and coils should be added, as shown in figures below.
3. **Compilation and Simulation:** Once the ladder diagram is drawn, compile it for errors and simulate the functionality using the OpenPLC Editor. This is the step that ensures logic will act as expected prior to uploading into hardware.
4. **Upload to Arduino**: After successful simulation, it is the time for compilation of the ladder diagram and uploading into the Arduino board. One needs to select the proper COM port and ensure that all the pin associations of the OpenPLC variables and Arduino are appropriately configured.
5. **Circuit Construction**: After uploading, the circuit should actually be built according to the document, including push button switches and an LED connected with the Arduino.
6. **Functional Testing**: Finally, test whether the functionalities of the circuit work as expected; it has to be ensured that the LED will behave accordingly as controlled by the push buttons according to the logic of the ladder diagram.

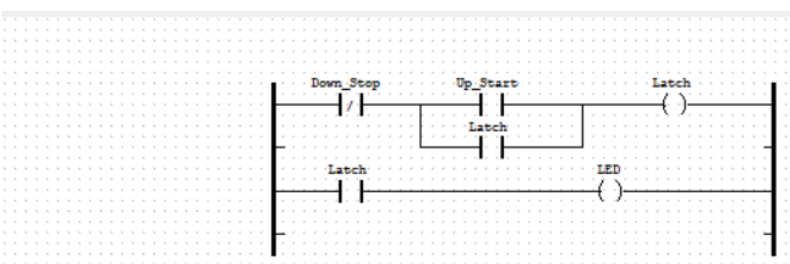
By implementing this procedure, it will be possible to interface the microcontroller with the PLC in desired control of the electromechanical processes.

**RESULTS**




**Circuit**

| | | | | |
|---|---|---|---|---|
| 1 | Up_Start | Local | BOOL | %IX0.0 |
| 2 | Down_Stop | Local | BOOL | %IX0.1 |
| 3 | Latch | Local | BOOL | |
| 4 | LED | Local | BOOL | %QX0.0 |




**Open PLC Setup**

Up_Start is wired as a normally open (NO) contact at address %I0.0 or Pin 2. Down_Stop is wired as a normally closed (NC) contact at address %I0.1 or Pin 3. Latch is used as an internal relay coil used to keep the LED on after Up_Start is pressed. LED is the output controlled by the ladder logic, located at %QX0.0 or Pin 7.

The behavior would be when Up_Start is pressed, Latch is energized and stays on even if Up_Start is released. This Latch coil keeps the LED on.

When Down_Stop is activated (connected as a normally closed switch), it breaks the circuit and resets the Latch, turning off the LED.

For the circuit there is no need for an external latch as the PLC Ladder Diagram has done its job. So the connection would be standard as an arduino connected to 5v and ground with pin 2 and 3 used to connect to the button for reading button state with a pull down resistor to ensure the pin read low (0) when the button is pressed. Then the LED is connected to Pin 7 as in address %QX0 in PLC address.

## DISCUSSION

This experiment successfully showcased how PLC programming with OpenPLC Editor can be applied to control an Arduino microcontroller, demonstrating basic automation tasks like blinking an LED and using a Start-Stop control circuit. The ladder diagram design worked as expected, showing the potential for using affordable microcontroller setups to prototype industrial control processes.

Some minor discrepancies were observed, such as slight delays in LED timing and response when pressing the Down_Stop button. These issues likely stemmed from hardware timing differences between OpenPLC simulation and actual Arduino performance, as well as occasional inconsistencies in breadboard connections.

Despite these small challenges, the experiment highlighted the practicality of PLC logic for simple automation, underlining the importance of precise I/O mapping and simulation before hands-on testing. While Arduino is a cost-effective choice for prototyping, additional adjustments may be necessary to ensure the reliability required in industrial applications.

## CONCLUSION

This laboratory experiment successfully demonstrated the practical integration of PLC programming concepts with Arduino microcontroller hardware through the implementation of ladder diagrams using OpenPLC Editor. The successful development and testing of both a basic LED blinking circuit and a more complex Start-Stop Control Circuit validated the effectiveness of using OpenPLC Editor for ladder logic implementation and simulation before physical deployment.The direct experience that was gained from ladder logic programming, hardware interfacing, and circuit implementation can be applied to industrial automation systems and process control. In addition, the successful transfer of PLC programming to Arduino hardware that is affordable shows the potential of cost-effective prototyping of industrial control systems in the environment of education. This way, software simulation and hardware testing ultimately lead to more complex industrial automation systems. Finally,the students were able to identify the significance of proper digital I/O handling and control system implementation by solving mechatronic problems.

## RECOMMENDATION

To ensure the experiment runs smoothly and minimizes errors, it is crucial to prepare all necessary components, such as the Arduino board, OpenPLC Editor software, push-button switches, jumper wires, LED, resistors, and breadboard, verifying their functionality before starting. Familiarizing yourself with the OpenPLC Editor software and its features is essential; consult the documentation for clarity on variable associations and pin configurations. Pay careful attention to pin assignments, ensuring accurate mapping between OpenPLC variables and Arduino pins by referencing the correct documentation. When assembling the circuit, follow the schematic meticulously, using secure connections and color-coded wires to avoid mistakes. Before uploading the program to the Arduino, compile and simulate the ladder diagram in the OpenPLC Editor to detect and resolve any logic errors. Verify hardware connections, such as the correct COM port and board type, using tools like the Arduino IDE to confirm communication. Test the circuit incrementally, validating each stage, like the blinking LED, before introducing modifications or additional complexity. Keep a clean workspace to avoid accidental disconnections or short circuits, and always disconnect power while making adjustments. Additionally, document any modifications to the diagram or circuit and save multiple versions of the program to enable easy troubleshooting and revisions. By adhering to these steps, the experiment will proceed with minimal errors and greater efficiency.

**ACKNOWLEDGEMENTS**

A special thanks goes out to Dr. Wahju Sediono and Dr. Zulkifli Bin Zainal Abidin, my teaching assistant, and my peers for their invaluable help and support in finishing this report. Their advice, feedback, and experience have greatly influenced the level of quality and understanding of this work. Their time, patience, and commitment to supporting my academic success are greatly appreciated.

**STUDENT'S DECLARATION**

**Certificate of Originality and Authenticity**

This is to certify that we are **responsible** for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or persons.

We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report**. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of the individual's contributors to the report.

We therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us**.

| | |
|---|---|
| Signature: | Read ☑ |
| Name: IRDINA NABIHAH BINTI MOHD NAZRI | Understand ☑ |
| Matric Number: 2214772 | Agree ☑ |

| | |
|---|---|
| Signature: | Read ☑ |
| Name: KHALISAH AMANI BINTI YAHYA AZMI | Understand ☑ |
| Matric Number: 2218184 | Agree ☑ |

| | |
|---|---|
| Signature: | Read ☑ |
| Name: LUQMAN AZFAR BIN AZMI | Understand ☑ |
| Matric Number: 2219857 | Agree ☑ |

Signature: Nasri
Name: MOHAMAD NASRI BIN MOHAMAD NAZRI
Matric Number: 2219879

| | |
|---|---|
| Read | ✓ |
| Understand | ✓ |
| Agree | ✓ |

Signature:
Name: MUHAMAS NURHAKIMIE THAQIF BIN ABDULLAH
Matric Number: 2213217

| | |
|---|---|
| Read | ✓ |
| Understand | ✓ |
| Agree | ✓ |