LABORATORY PROJECT REPORT

PLC INTERFACING WITH MICROCONTROLLER

EXPERIMENT 5

DATE : 14TH APRIL 2025

SECTION : 1

GROUP : 1

SEMESTER 2, 2024/2025

NO	NAME	MATRIC NO
1.	AKMAL FAUZAN BIN AZHAR	2319531
2.	AIMAN SAIFULLAH BIN AMINNURLLAH	2319185
3.	SYAFIQ HELMIE BIN SAIFULLAH ADHA	2316049

Table of Contents

1.0 Introduction

Programmable Logic Controllers (PLCs) are integral components in the automation industry. Originally developed to replace complex relay logic systems in manufacturing plants, PLCs are now central to controlling and monitoring electromechanical processes in sectors ranging from automotive assembly to food processing.

In this session, we interface a PLC system, programmed using OpenPLC Editor, with an Arduino microcontroller. OpenPLC is an open-source platform that supports several programming languages defined in the IEC 61131-3 standard, including Ladder Diagram (LD), which replicates relay logic used by electricians and automation engineers.

The hands-on component of this experiment focuses on two main tasks:

1. Blinking an LED using basic ladder logic, simulating and deploying it on an Arduino board.

2. Designing a more advanced Start-Stop control circuit using two push buttons and an LED to mimic industrial control systems.

This experiment serves to bridge theoretical knowledge with practical implementation, encouraging the student to explore both the abstract logic of control systems and their physical realization using electronic components.

2.0 Materials and Equipments

Software:

- OpenPLC Editor: For designing ladder logic, simulating it, and compiling code for Arduino microcontroller.
- Arduino IDE (optional): Used for board communication troubleshooting and verification.
- Device Manager (Windows): To confirm the active COM port used by Arduino.

Hardware:

- Arduino Mega 2560
- 2 Push Button Switches
- LED
- Breadboard
- Resistors (220Ω – 10kΩ)
- Jumper wires
- USB Cable

3.0 Experimental Setup

3.1 Diagram Setups

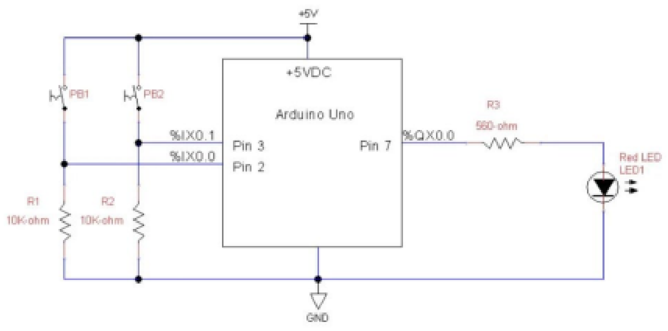
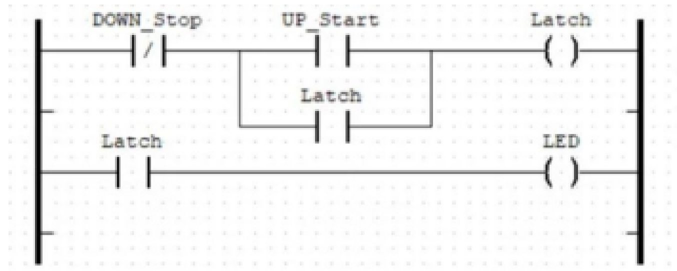
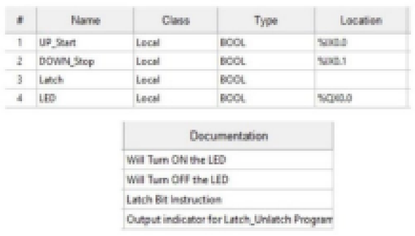


Fig 4. Start-Stop Control Circuit





#	Name	Class	Type	Location
1	UP_Start	Local	BOOL	%IX0.0
2	DOWN_Stop	Local	BOOL	%IX0.1
3	Latch	Local	BOOL	
4	LED	Local	BOOL	%QX0.0

Documentation
Will Turn ON the LED
Will Turn OFF the LED
Latch Bit Instruction
Output indicator for Latch_Unlatch Program

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

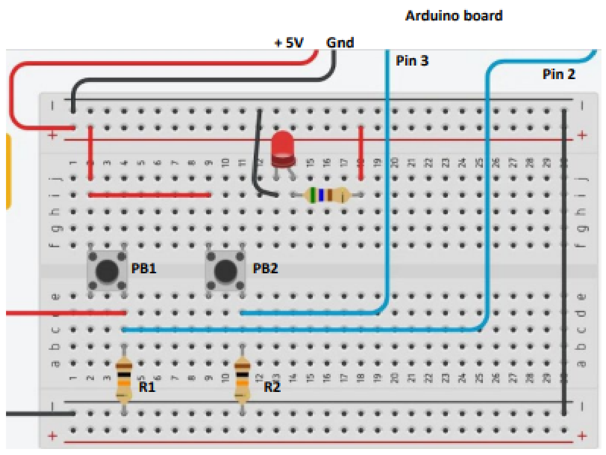


Fig 6. Circuit Diagram

3.2 Circuit Setup

1. The ladder diagram shown in Fig. 5 was created.
2. All variables used within the ladder diagram were specified.
3. The ladder diagram was compiled and simulated using the OpenPLC Editor.
4. It was uploaded to the Arduino board.
5. The correct COM port number was selected, and all pin associations between the Arduino board and OpenPLC variables were verified.
6. The circuit illustrated in Fig. 6 was constructed.
7. Its functionality was tested.

4.0 Methodology

4.1 LED Blinking Using Ladder Logic

This section served as an introduction to OpenPLC and ladder diagram logic. A simple circuit was created where an LED connected to the Arduino blinks continuously as per the logic written in the OpenPLC Editor.

Steps:

1. Download and Install OpenPLC Editor from the official website.
2. Create a New Project:
    - File → New → Select a new folder → Name the project.
    - Select LD (Ladder Diagram) as the language.
3. Define Variables:
    - Create a variable (e.g., `led1`) with the following settings:
        - Class Filter: `Local`
        - Type: `BOOL`
        - Location: `%QX00` (corresponding to digital output pin 14 on Arduino Mega)
4. Create Ladder Logic:
    - Insert power rails, negated contact, and coil as shown in the reference diagram.
    - Right-click workspace → Add → Element (contact, coil, etc.).
5. Compile and Simulate:
    - Click the build icon to compile the diagram.
    - Observe blinking LED in simulation mode.

6. Upload to Arduino:
    - Connect Arduino via USB.
    - Select appropriate board and COM port.
    - Click *Transfer to PLC*.
    - LED begins blinking as per program logic.
7. Timer Block Integration *(Exercise)*:
    - Right-click → Add Block → Standard Function Blocks → Timers.
    - Add a variable with `Expression = T#1000ms` to control the blink interval.
    - Simulate and upload updated logic to Arduino.

4.2 Start-Stop Control Circuit

In this part, we design a Start-Stop control circuit with two push buttons. This circuit is commonly found in industrial applications where motors or actuators must be safely toggled between on and off states.

Circuit Logic:

- PB1 (Start): When pressed, set the output (LED) to ON.
- PB2 (Stop): When pressed, resets the output (LED) to OFF.
- LED retains its state until an explicit signal is received.

Steps:

1. Ladder Diagram Design:
    - Configure push buttons as input variables: `%IX0.0` (PB1), `%IX0.1` (PB2).
    - Configure LED as output: `%QX0.0`.

2. Variable Configuration:

Variable	Type	Location	Description
PB1	BOOL	%IX0.0	Start Button
PB2	BOOL	%IX0.1	Stop Button
LED1	BOOL	%QX0.0	Output LED

3. Circuit Assembly:
    - Connect PB1 and PB2 to digital pins 2 and 3 respectively, with pull-down resistors.
    - Connect LED to pin 14 with a 220Ω resistor.
    - Connect components to the breadboard and Arduino.

4. Compile and Upload:
    - Validate logic via simulation.
    - Upload program to Arduino.
    - Verify correct function based on button inputs.

Test Case	Description	Action	Observed Behaviour	Result
1	Led Blinking	Upload base ladder logic	LED blinks continuously	Pass
2	LED Blinking (Timer-controlled)	Integrate Timer block	LED blinks every 1 sec	Pass
3	Start-Stop Control - Start Pressed	Press PB1	LED turns ON	Pass
4	Start-Stop Control - Stop Pressed	Press PB2	LED turns OFF	Pass
5	No Button Press	Do not press PB1 or PB2	LED retains state	Pass

5.0 Results

The test cases validate that both basic logic and more advanced conditional logic can be successfully implemented using OpenPLC and deployed to Arduino.

6.0 Discussion

This experiment emphasizes the fundamental principle of control logic abstraction—how simple ladder logic, built using symbolic contact and coil representations, can govern real-world electrical outputs.

By using OpenPLC, we abstract complex microcontroller code into intuitive logic structures. This is particularly advantageous for industrial technicians and engineers who may not have in-depth programming backgrounds.

The Start-Stop logic implemented is a classic latching circuit, where the system maintains its output even after the input is released. This mimics real-world applications such as conveyor belt systems or motor starters. Understanding how to properly set and reset outputs using basic AND/OR logic, combined with physical button inputs, is foundational in industrial automation.

Incorporating the timer block added further depth by demonstrating how temporal control can be integrated into logic, a critical feature in automated systems where timing, delays, and scheduling are important.

7.0 Conclusion

As a conclusion, ladder logic programming of Start-Stop control circuits and LED blinking was successfully implemented on OpenPLC, showcasing how Arduino design power could be leveraged through OpenPLC hardware to create simple programmable logic control hardware. The integration of ladder diagram design, simulation, and deployment through this project helped to solidify fundamental principles in automation and control systems.

The not only showcased OpenPLC's accessibility and efficiency for both new users and seasoned professionals alike, but also demonstrated the power of OpenPLC to turn correct visual logic into working embedded control. Timers and latching mechanisms: complex behavior from simple logical constructs.

In general, the project was an effective way to gain a hands-on knowledge of the way industrial control strategies can be implemented and tested on low-cost hardware, ensuring that students and engineers are prepared for business automation projects.

8.0 Recommendations

Though, this section will discuss how the project can expand and improve in the future by integrating more advanced concepts of ladder logic programming and exploring complex control system applications such as sensor-triggered automation or multi-output coordination. Adding safety features, such as an emergency stop button, limit switches, or fault indicators, would simulate actual industrial safety practice and lead to a more reliable system. Integrating basic HMI elements—like LCD screens, pushbutton interfaces, or status LEDs—will also help improve user interaction and system behavior transparency. Another good concept to practice is using other OpenPLC supported IEC 61131-3 languages (like Structured Text, or Function Block Diagram). Make Hardware More Complex: Using relays, actuators or analog sensors in addition to the standard setup can add complexity to the problem automation will solve, making it feel more realistic and giving additional challenge to some of these tasks. Ultimately this would lead to better organized systems, easier debugging, and help make the jump from simulation to physical prototype a bit easier if the system could be documented, kept track of and adjusted in a version controlled manner.

9.0 References

1. Autonomy Logic – OpenPLC Official Site: https://autonomylogic.com

2. Control.com:
    - PLC Ladder Logic on Arduino – Introduction to OpenPLC
    - PLC Ladder Logic on Arduino – Building a Start-Stop Circuit

10.0 Acknowledgement

Special thanks to Dr. Wahju Sediono and Dr. Zulkifli Bin Zainal Abidin, as well as the teaching assistants and peers, for their guidance and support in completing this experiment.

11.0 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.

Name: Aiman Saifullah bin Aminnurllah	Read	/
Matric Number: 2319185	Understand	/
Signature:	Agree	/

Name: Akmal Fauzan Bin Azhar	Read	/
Matric Number:2319531	Understand	/
Signature:	Agree	/

Name: Syafiq Helmie bin Saifullah Adha		Read	/
Matric Number: 2316049		Understand	/
Signature:		Agree	/