



الجامعة الإسلامية العالمية ماليزيا  
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA  
يُونُسُ بَرَسِيَّتِي إِسْلَامُ، إِنْتَارَا بَغْسِيَا مِلْدِسِيَا  
*Garden of Knowledge and Virtue*

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

**GROUP 4**

**MCTA 3203**

**SEMESTER 2 2024/2025**

**MECHATRONICS SYSTEM INTEGRATION**

**DATE OF SUBMISSION: 9TH APRIL 2025**

NO	NAME	MATRIC NUMBER
1.	IRDINA NABIHAH BINTI MOHD NAZRI	2214772
2.	IZZAH ZAHIRA BINTI NORAZLEE	2217696
3.	NOR MAISARAH BINTI ISMAIL	2213080

## **TABLE OF CONTENTS**

<b>INTRODUCTION</b>	<b>3</b>
<b>ABSTRACT</b>	<b>3</b>
<b>MATERIALS AND EQUIPMENTS</b>	<b>4</b>
<b>EXPERIMENTAL AND SETUP</b>	<b>4</b>
<b>METHODOLOGY</b>	<b>6</b>
<b>RESULTS</b>	<b>7</b>
<b>DISCUSSION</b>	<b>8</b>
<b>CONCLUSION</b>	<b>8</b>
<b>RECOMMENDATION</b>	<b>9</b>
<b>ACKNOWLEDGEMENTS</b>	<b>10</b>
<b>STUDENT'S DECLARATION</b>	<b>11</b>

## **INTRODUCTION**

The integration of Programmable Logic Controllers (PLCs) with microcontrollers represents a fundamental skill in the field of automation and mechatronics engineering. PLC with microcontrollers is a crucial aspect of developing flexible, cost effective control solutions. This report focuses on the investigation of both the software and hardware aspect of the interfacing PLC environment. It is developed by OpenPLC Editor with an Arduino microcontroller. The objectives of this experiment include the development, simulation and real time execution of ladder logic diagrams to control simple electromechanical operations. This experiment aims to enhance understanding of PLC by designing and implementing basic ladder logic circuits. By constructing the circuits such as an LED blinking system and a Start-Stop push button control, it demonstrates the key concepts such as variable mapping, timing control and real world deployment challenges. The outcomes of this study provide foundational experience in PLC programming and hardware interfacing, crucial for applications ranging from prototyping to industrial automation systems.

## **ABSTRACT**

In this experiment, we aimed to explore the integration of Programmable Logic Controllers (PLCs) with microcontrollers using OpenPLC Editor software and an Arduino board. Hence, the objective was to understand both the software and hardware aspects of PLC interfacing through the creation and simulation of ladder diagrams. Initially, a simple LED blinking circuit was developed and simulated within the OpenPLC environment. This was followed by uploading the compiled ladder diagram to the Arduino, allowing the LED to blink according to the logic set. In the subsequent task, a more complex Start-Stop control circuit was designed using push buttons and an LED, simulating industrial control applications. The ladder diagram was modified to incorporate timing control, enabling adjustable blinking intervals. All variables were correctly mapped to the Arduino's digital I/O pins based on physical addressing documentation. Moreover, the experiment successfully demonstrates the practical skills in PLC programming, simulation, and hardware interfacing, highlighting the effectiveness of OpenPLC as a platform for educational and prototyping purposes in automation systems.

## MATERIALS AND EQUIPMENTS

### 1. Software:

- OpenPLC Editor (for ladder diagram creation and simulation)

### 2. Microcontroller:

- Arduino Board (e.g., Arduino Mega)

### 3. Electronic Components:

- 2 Push Button Switches (for Start and Stop control)
- 1 LED (for output indication)
- Resistors (e.g., R1 and R2 for current limiting)
- Jumper Wires
- Breadboard

### 4. Other Requirements:

- USB Cable (to connect Arduino to computer)
- Computer/Laptop (with OpenPLC Editor installed)
- Power Supply (typically via USB or external 5V source)
- Arduino IDE (optional, for driver and COM port configuration)

## EXPERIMENTAL AND SETUP

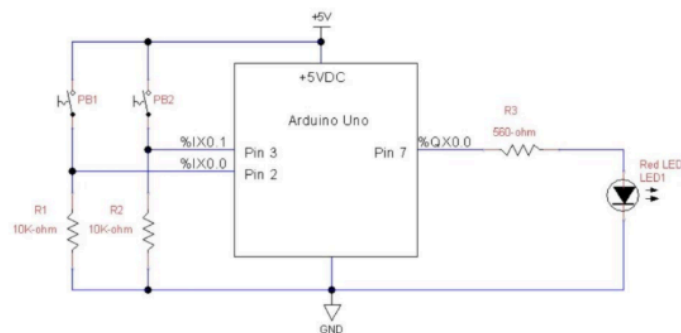
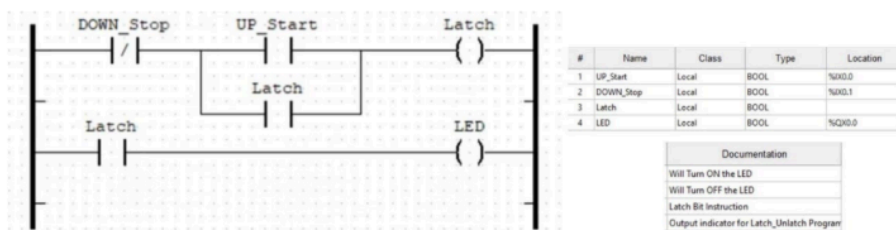


Fig. 4: Start-Stop Control Circuit



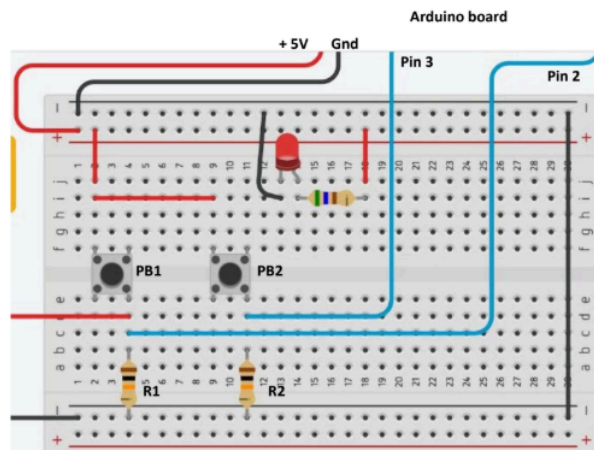


Fig. 6

1. Create the ladder diagram.
2. Specify all variables used in the ladder diagram.
3. Compile and stimulate the ladder diagram in OpenPLC Editor.
4. Upload the ladder diagram to the Arduino Board.
5. Select the correct COM port number and all pin associations between the OpenPLC variables and Arduino board.
6. Build the circuit as shown in the figure.
7. Test the functionality.

## METHODOLOGY

The experiment was divided into two primary stages: first, an OpenPLC Editor-based flashing LED circuit was developed and tested; second, an Arduino board and push buttons were used to construct a Start-Stop control circuit.

The first step involved downloading and installing the OpenPLC Editor software from the official website. "Ladder Diagram (LD)" was chosen as the programming language for the new project. A coil and a single normally closed (negated) contact were used to create a straightforward circuit that mimicked the blinking of an LED. The LED output was represented by a boolean variable. After finishing, the ladder diagram was compiled to look for mistakes and then simulated in the program to see how it blinked.

Transferring the program to an Arduino board was the next step. The computer was linked to the Arduino Mega, and the COM port was located. In OpenPLC, the analogous variable was given the address %QX0.0, and the LED was physically linked to digital pin 14 on the board. After that, the program was compiled and uploaded to the Arduino via OpenPLC Editor's integrated upload function. Following upload, the LED started to flicker in accordance with the ladder diagram's logic.

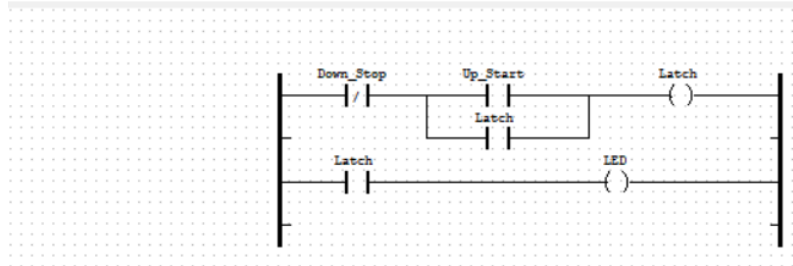
A timer block was included in the ladder diagram to improve control over the blinking LED. A time delay expression, such as T#1000ms, was defined after inserting a TONNE (Timer ON Delay) block. The LED control logic was linked to the timer's input and output, enabling the LED to blink at a predetermined period. To confirm that the blinking behaviour had changed, the updated diagram was once more assembled, simulated, and uploaded to the Arduino.

A Start-Stop control circuit was used in the experiment's second section. Two pushbuttons were included in the ladder diagram: a normally closed (NC) Stop button and a normally open (NO) Start button. The LED was represented by an output coil that was wired so that pressing the Start button activated the coil and used a holding contact to latch the output. The LED would turn off if the Stop button was pressed, breaking the circuit. The LED stayed on pin 14 (%QX0.0), while the Start and Stop buttons were linked to digital pins 2 and 3, designated as %IX0.0 and %IX0.1, respectively.

The Arduino board, resistors, and jumper wires were used to construct the actual circuit on a breadboard. As previously, the ladder logic program was compiled and submitted. In order to verify that the Start-Stop control logic was functioning properly, functionality was verified by pressing the Start button to turn on the LED and the Stop button to turn it off.

## RESULTS

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



The experiment successfully demonstrated the interfacing of a PLC-based control system with an Arduino microcontroller using OpenPLC Editor. A Start-Stop control circuit was designed, simulated, and implemented, where two push buttons were used to control the state of an LED. The Up\_Start button, wired as a normally open (NO) contact at Pin 2 (%IX0.0), correctly energized the Latch coil when pressed, causing the LED to turn on and remain on even after the button was released. Conversely, the Down\_Stop button, configured as a normally closed (NC) contact at Pin 3 (%IX0.1), successfully reset the Latch and turned the LED off when activated. The LED was connected to Pin 7 (%QX0.0) and accurately responded to the ladder logic instructions. Throughout the testing phase, the circuit behaved as expected: the Start button activated and latched the LED on, while the Stop button deactivated the circuit and switched the LED off. The variable mapping, ladder diagram design, and hardware connections all functioned correctly, confirming the effectiveness of PLC programming in managing real-world digital inputs and outputs through microcontroller hardware.

## **DISCUSSION**

The experiment effectively demonstrated the feasibility of controlling an Arduino microcontroller using PLC programming with the OpenPLC Editor, especially through the implementation of routine automation activities such as LED flash and a Start-Stop control circuit. The ladder diagram worked flawlessly, crediting prototyping industrial control systems with low-cost microcontroller arrangements.

There were some tiny issues that were noticed, for example, exceedingly small time lags in response and timing from LEDs when the Down\_Stop button is activated. Such differences may be due to some timing variations of OpenPLC simulation and real-time execution from Arduino, with some errors being present in some breadboard connections.

Despite such minor hindrances, the experiment did show the effectiveness of employing PLC logic in performing simple automation tasks. It reasserted the importance of accurate I/O mapping and thorough simulation prior to testing on the actual setup. Though Arduino itself is a cost-effective prototyping tool, further optimization may be needed in order to meet industrial environment standards regarding reliability.

## **CONCLUSION**

All in all, this experiment successfully demonstrated the integration of Programmable Logic Controllers (PLCs) with microcontrollers, specifically through the use of OpenPLC Editor and an Arduino board. The main objective—to design, simulate, and implement ladder logic programs for controlling an LED—was fully achieved. Preliminary, a simple blinking LED circuit was created and tested, followed by a more complex Start-Stop control system utilizing push buttons and a latching mechanism.



The results showed that ladder logic, even when interfaced with open-source microcontrollers such as Arduino, can be utilised to control electromechanical operations. The application of timing and memory operations in real-world control circuits was demonstrated by the use of timer blocks and holding contacts. The experiment validated the idea that OpenPLC is a good platform for modelling and implementing automation control tasks in the real world.

The knowledge and ideas used in this experiment have wide-ranging applications outside of the lab, including instructional robots, home automation systems, and industrial automation. Students are better prepared to deal with sophisticated automation systems frequently found in the manufacturing and process control industries when they have a basic understanding of PLC programming and its hardware interface capabilities.

## **RECOMMENDATION**

To maintain the experiment running smoothly and with minimal errors, it is necessary to prepare all the required parts well in advance—Arduino board, OpenPLC Editor software, push-button switches, jumper wires, LED, resistors, and breadboard—and make sure each piece of equipment works well. Acquaintance with the OpenPLC Editor and its features is also important; go through its manual to learn how variable assignments and pin configuration should be interpreted correctly.

Careful pin mapping between Arduino pins and OpenPLC variables is critical, so always refer to the proper documentation. When constructing the circuit, adhere to the schematic as closely as possible, using color-coded wires and solid connections to minimize the potential for errors. Prior to uploading your code to the Arduino, simulate and compile the ladder diagram in the OpenPLC Editor to identify and correct any logic errors.

Double-check the hardware settings, including the correct COM port and board type, through software such as the Arduino IDE to achieve proper communication. Test the circuit in stages such as verifying the LED blink before adding more complexity. Maintain a clean workspace to prevent accidental disconnection or short circuits, and always power down the system before any changes.

Lastly, check for any change to the circuit or diagram and save multiple versions of the code for debugging with ease and even future changes. Following these measures will make a hassle-free error-minimized process of experimentation achievable.

## **ACKNOWLEDGEMENTS**

A special thanks goes out to Dr. Wahyu 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:   
Name: IRDINA NABIHAH BINTI MOHD NAZRI  
Matric Number: 2214772

Read ☒  
Understand ☒  
Agree ☒

Signature:   
Name: IZZAH ZAHIRA BINTI NORAZLEE  
Matric Number: 2217696

Read ☒  
Understand ☒  
Agree ☒

Signature:   
Name: NOR MAISARAH BINTI ISMAIL  
Matric Number: 2213080

Read ☒  
Understand ☒  
Agree ☒