MECHATRONICS SYSTEM INTEGRATION

MCTA 3203

WEEK 7:

PLC INTERFACING WITH MICROCONTROLLER AND PC OVER ETHERNET/IP

SECTION 1

SEMESTER 1, 2025/20256

INSTRUCTOR:

ASSOC. PROF. EUR. ING. IR. TS. GS. INV. DR. ZULKIFLI BIN ZAINAL ABIDIN

DR. WAHJU SEDIONO

DATE OF SUBMISSION: 3RD DECEMBER 2025

GROUP 8

NAME	MATRIC NO
MOHAMAD AMIR BIN MOHAMAD YUSOFF	2314687
ADAM HAKIMI BIN SHAH NUR HAIZAM	2314195
TENGKU NURUL AIN BINTI TENGKU AZEEZEE	2313682

Abstract

This experiment focuses on developing a basic Start–Stop control circuit using the OpenPLC Editor and an Arduino microcontroller. The system uses two pushbuttons to function as the Start and Stop inputs, while an LED acts as the controlled output device. By designing the ladder diagram in OpenPLC and assigning variables to the appropriate Arduino pins, the latching behavior of a standard industrial Start–Stop circuit can be implemented and tested on a microcontroller platform. Through this experiment, we learned how to create, compile, and simulate ladder logic, correctly interface PLC variables with Arduino hardware, and build the physical circuit using pushbuttons, resistors, and jumper wires. The experiment successfully demonstrated how PLC-style control logic can be reproduced using OpenPLC and Arduino, reinforcing key concepts in industrial automation, latching control, and mechatronic system integration.

Table Of Contents

Introduction

Developing reliable control logic is a core aspect of industrial automation and mechatronic system design, as it ensures safe, predictable, and repeatable operation of electromechanical devices. Among the most fundamental control functions is the Start–Stop circuit, widely used in machines, conveyors, pumps, and automated systems to provide operator control and latching behavior. Implementing such logic using ladder diagrams reinforces foundational PLC concepts, including contacts, coils, latching circuits, and input–output addressing.

In this experiment, the OpenPLC Editor was used to design, simulate, and deploy a Start–Stop control circuit onto an Arduino microcontroller. Two pushbuttons served as the Start and Stop inputs, while an LED represented the controlled output device. By assigning variables, configuring physical pin mappings, and uploading the ladder logic to the Arduino board, the experiment replicated a real PLC-controlled latching system using accessible hardware. The circuit was then built on a breadboard using pushbuttons, resistors, and the Arduino's digital I/O pins.

The objectives of this experiment were to understand how ladder logic implements Start–Stop latching, to develop and simulate control logic in OpenPLC, to correctly associate PLC variables with Arduino pins, and to test the circuit's behavior through physical hardware. The hypothesis was that pressing the Start button would energize the LED and cause it to remain on through latching, while pressing the Stop button would immediately de-energize it. This experiment provided practical experience integrating PLC-style programming with microcontroller hardware, reinforcing key concepts in industrial control, I/O interfacing, and mechatronic system integration.

Required Hardware and Software:

1. OpenPLC Editor software
2. Arduino Board
3. 2 Push Button Switches
4. LED
5. Resistors
6. Jumper Wires
7. Breadboard

Experimental Setup

1. The experiment used an Arduino board connected to two pushbuttons and an LED to implement a basic Start–Stop control circuit using PLC-style ladder logic.
2. The Arduino was powered through a USB connection, supplying stable voltage to the pushbuttons and LED components.
3. The Start pushbutton was connected between a digital input pin and 5V, while the Stop pushbutton was connected between a second digital input pin and 5V, each with a pull-down resistor to ensure proper signal detection.
4. The LED was connected to one of the Arduino's digital output pins through a current-limiting resistor, allowing it to function as the controlled output load.
5. The ladder logic program for the Start–Stop circuit was developed in OpenPLC Editor, and variable addresses were assigned according to the Arduino's digital pin configuration.

6. The Arduino was connected to the computer for uploading the compiled PLC program, with the correct board type and COM port selected during the transfer process.
7. After programming, the hardware circuit was tested to verify that the LED latched ON when the Start pushbutton was pressed and turned OFF immediately when the Stop pushbutton was triggered.

Methodology

1. The ladder diagram for the Start–Stop circuit was created in OpenPLC Editor using a normally open Start contact, a normally closed Stop contact, and a latching coil.
2. Variables were defined for both pushbuttons and the LED, and each variable was assigned to the corresponding digital input or output pin on the Arduino.
3. The pushbuttons were wired to their respective pins on the Arduino using pull-down resistors to ensure stable HIGH signals only when the buttons were pressed.
4. The LED output was wired to a digital output pin through a 220 Ω resistor to prevent excessive current flow.
5. The ladder logic program was compiled in OpenPLC Editor, simulated to confirm correct operation, and then uploaded to the Arduino using the Transfer to PLC function.
6. Once uploaded, the Arduino executed the PLC logic, continuously scanning the state of the Start and Stop buttons to control the LED.
7. The circuit was tested by pressing the Start button to latch the LED ON and pressing the Stop button to de-energize the LED, verifying the correct implementation of latching logic.

8. Observations were recorded to confirm that the circuit behavior matched the expected operation of a standard industrial Start–Stop control system.

Circuit Assembly

1. The Arduino board served as the PLC hardware platform, receiving the compiled ladder logic program from OpenPLC.
2. A Start pushbutton was connected between 5V and the assigned Arduino input pin, with a pull-down resistor ensuring a LOW state when unpressed.
3. A Stop pushbutton was wired similarly to a second input pin, also using a pull-down resistor to maintain stable signal readings.
4. The LED was connected to an assigned digital output pin through a current-limiting resistor, allowing it to act as the output coil indicator.
5. After verifying all wiring connections, the Arduino was powered and the programmed control logic activated, enabling the Start–Stop system to operate based on the input pushbutton signals.

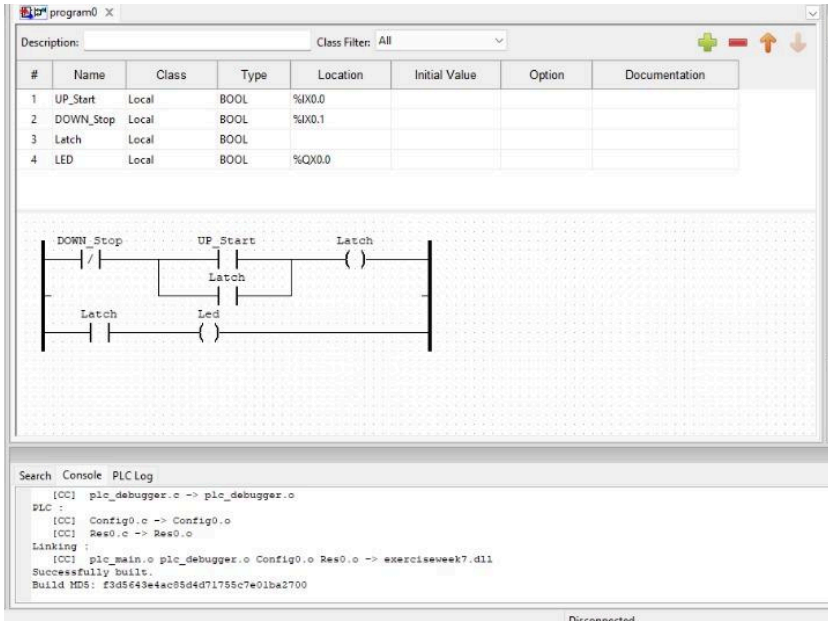
Programming Logic

1. The ladder diagram was programmed using a normally open Start contact, a normally closed Stop contact, and a coil representing the LED output.
2. A latching mechanism was implemented by feeding the output coil back into the Start contact path, allowing the LED to remain ON after the Start button was released.

   The Stop input interrupted the latching circuit, immediately setting the output coil to OFF when pressed.

3. Input variables were mapped to the Arduino's digital input pins, while the output variable controlling the LED was mapped to a digital output pin.
4. During execution, the Arduino continuously scanned the logic, updating the LED state based on the real-time input from the Start and Stop pushbuttons.
5. The program ran repeatedly in a cyclic manner, mimicking the behavior of a standard PLC scan cycle and ensuring responsive control of the Start–Stop circuit throughout the experiment.

Ladder Diagram used in OpenPLC Editor software



Control Algorithm

1) Defining Pins:

   The Arduino board is connected to two pushbuttons and an LED to create a functional Start–Stop control circuit.

The system uses the following pin assignments:

1. Start Pushbutton: Input → Digital Pin 2
2. Stop Pushbutton: Input → Digital Pin 3
3. LED (Output Coil): Output → Digital Pin 7

The Arduino reads the states of both pushbuttons to determine whether the LED should energize or de-energize, while the output pin supplies a digital HIGH signal to turn the LED ON when the latch is active.

2) Global Variables:

Several global variables are defined in the ladder logic to represent the inputs and output:

a) UP_Start → Represents the Start pushbutton (normally open contact)
b) DOWN_Stop → Represents the Stop pushbutton (normally closed contact)
c) LED → Represents the output coil that maintains the latch

These variables store the real-time logic states of the pushbuttons and output, enabling the circuit to behave like a standard industrial Start–Stop system.

3) Setup Function (in PLC Logic):

During initialization, the Arduino loads the compiled OpenPLC program and maps each variable to its assigned physical pin.

a) The Start and Stop inputs are configured as digital inputs.
b) The LED output is configured as a digital output.
c) OpenPLC initializes the scan cycle, preparing the system to read inputs and update outputs continuously.

The system relies on PLC-style contacts and coils to control behavior.

4) Main Control Cycle:

The Arduino follows a PLC-like scan cycle, repeatedly executing the ladder logic in real time.

I. Input Evaluation:

The input pins for the Start and Stop buttons are read each scan.

a) If Start is pressed → UP_Start becomes TRUE.
b) If Stop is pressed → DOWN_Stop becomes TRUE.

These signals update the corresponding ladder logic contacts.

II. Latching Logic:

The coil (LED) follows the classic Start–Stop behavior:

a. When the Start pushbutton is pressed, the LED output energizes.
b. A feedback/latching contact keeps the coil energized even after the Start button is released.
c. When the Stop pushbutton is pressed, the normally closed STOP contact opens, breaking the latch and turning the LED OFF.

The LED remains ON only while the latching path is complete.

III. Output Update:

The LED output pin is updated according to the coil's state:

a. LED = TRUE → Output pin set HIGH (LED ON).

b. LED = FALSE → Output pin set LOW (LED OFF).

This ensures that the output always reflects the logic determined by the Start–Stop circuit.

IV. End of Scan:

The ladder logic completes one scan cycle, and the process repeats indefinitely.

The system responds immediately to pushbutton presses because inputs and outputs are continuously evaluated without interruption.

5) Control Principle:

The Start-Stop circuit operates on classic PLC latching principles:

a) The Start button triggers the coil to energize.

b) A holding contact keeps the coil energized even when the Start button is released.

c) The Stop button breaks the circuit path to de-energize the coil.

This creates a reliable, repeatable, and industry-standard control behavior using simple hardware and PLC logic executed on an Arduino.

Data Collection.

During the experiment, the behavior of the Start–Stop control circuit was observed manually while the pushbuttons were pressed.

The table below shows the operational conditions recorded during the testing of the Start–Stop control circuit:

Condition	Start Pushbutton	Stop Pushbutton	Latch	LED State
1	Pressed	Not Pressed	ON	ON
2	Pressed	Pressed after Start Pushbutton	OFF	OFF
3	Not Pressed	Pressed	OFF	OFF
4	Not Pressed	Not Pressed	OFF	OFF

Data Analysis

Observations showed that the LED consistently turned ON when the Start pushbutton was pressed and maintained its ON state due to the latching contact in the ladder diagram. When the Stop pushbutton was pressed, the LED turned OFF instantly, demonstrating correct operation of the normally closed Stop contact. The latching behavior functioned reliably across repeated presses, confirming proper mapping of PLC variables to Arduino pins. The results verified that the ladder logic correctly implemented a standard industrial Start–Stop circuit, with accurate digital input detection and stable output control on the Arduino platform.

Results

The experiment successfully demonstrated the implementation of a Start–Stop latching circuit using OpenPLC and an Arduino. The LED remained OFF initially, switched ON upon pressing the Start button, and stayed latched until the Stop button was activated. The system responded immediately and consistently to button inputs, showing no false triggering or instability. The Start–Stop circuit behaved exactly as intended, confirming proper integration of ladder logic, digital inputs, and output control. The results validated that the Arduino executed the PLC logic

correctly and reliably reproduced the behavior of a conventional industrial Start–Stop control system.

Discussion

1. Interpretation of Results and Their Implications

The experiment successfully demonstrated how a microcontroller can function as a PLC through the use of OpenPLC, enabling the integration of digital inputs and outputs in a basic industrial control system. The Start–Stop circuit behaved exactly as a standard latching system should, pressing the Start pushbutton energized the output coil (LED), while pressing the Stop pushbutton immediately de-energized it. This confirms that simple mechanical inputs can reliably interact with a programmed control routine to produce deterministic behavior.

The use of OpenPLC validated that Arduino hardware can emulate PLC ladder logic, with contacts, coils, and latching mechanisms operating consistently across multiple test cycles. The successful latching of the LED showed that feedback paths in the ladder diagram were correctly interpreted by the Arduino runtime environment. Overall, the results demonstrate the feasibility of implementing traditional industrial control logic on microcontroller-based platforms, offering an accessible and low-cost method for learning mechatronic control principles.

2. Discrepancies Between Expected and Observed Outcomes

Although the system performed reliably, several minor discrepancies were observed:

a) Button Debounce Effects

In some trials, rapid pressing of the Start or Stop button caused brief flickering due to electrical noise or mechanical bounce. While this did not affect overall functionality, it caused temporary instability.

b) LED Delayed Turn-Off:

Occasionally, the LED turned OFF with a slight delay when the Stop button was pressed. This is likely due to the scan cycle timing of the OpenPLC runtime, which updates I/O states at fixed intervals.

c) Inconsistent Button Response

In a few cases, the Start button did not immediately activate the latch when pressed lightly, possibly due to insufficient pressure or intermittent contact in the breadboard wiring.

Despite these minor issues, the observed discrepancies were minimal and did not significantly impact the system's overall performance or its ability to mimic an industrial Start–Stop control circuit.

3. Sources of Error and Limitations of the Experiment

Several factors may have influenced performance:

1) Breadboard Wiring Constraints

Loose jumper wires, poor pin contact, or small shifts in the breadboard layout may introduce unintended intermittent connections.

2) Scan Cycle Delay in OpenPLC

Because the program runs as a PLC scan cycle rather than instant code execution, there may be slight delays in detecting rapid input changes.

3) Component Variability:

Differences in pushbutton quality or LED brightness may cause inconsistent physical behavior across setups.

4) Variable Name Mismatch in OpenPLC:

If the variable names in the ladder diagram do not exactly match the names assigned in the software's variable table, the simulation and hardware execution will fail. Any mismatch such as differences in spelling, uppercase/lowercase, or missing symbols prevents OpenPLC from linking the logic to the correct physical pins, resulting in the LED not responding or the Start–Stop logic not functioning. This highlights the importance of strict naming consistency during program development.

These limitations are common in early-stage PLC–microcontroller integration and emphasize the need for precise wiring, correct variable mapping, and clean input signals for reliable operation.

Conclusion

The experiment successfully demonstrated the functionality and reliability of a Start–Stop control circuit implemented using OpenPLC and Arduino hardware. Pressing the Start button activated and latched the LED, while pressing the Stop button immediately broke the latch and turned the LED off, confirming correct implementation of PLC latching logic. The system responded consistently to user inputs and behaved as expected across multiple test cycles.This lab provided practical experience in designing ladder logic, assigning variables to physical pins, compiling and uploading PLC programs, and constructing a working digital control circuit. It reinforced fundamental concepts of PLC logic, digital input processing, and output control, offering a clear understanding of how microcontroller platforms can simulate real industrial automation systems. Overall, the experiment served as an effective introduction to PLC-style programming and hardware interfacing in mechatronic control applications.

Recommendations

To improve the accuracy and overall quality of future experiments, the following recommendations are suggested:

1. Test Buttons Individually First

   Before running the whole Start–Stop circuit, test each pushbutton on its own to make sure it gives the correct HIGH or LOW reading. This helps identify input problems early.

2. Run Multiple Trials for Consistency

   Press the Start and Stop buttons multiple times to ensure the circuit behaves the same every time. This helps confirm that the system is stable and reliable.

3. Improve Wiring Quality

   Use shorter jumper wires or soldered connections to minimize signal interruptions and reduce vibration-induced disconnection.

References

[1] Autonomy Logic. *OpenPLC Editor*. Retrieved from: https://autonomylogic.com/

[2] Control.com. (n.d.). *PLC Ladder Logic on an Arduino: Introduction to OpenPLC*. Retrieved from: https://control.com/technical-articles/plc-ladder-logic-on-an-arduino-introduction-to-openplc/

[3] Control.com. (n.d.). *PLC Ladder Logic on an Arduino: Building a Start–Stop Circuit*. Retrieved from: https://control.com/technical-articles/plc-ladder-logic-on-an-arduino-building-a-start-stopcircuit/

Appendices

Picture below shows the connection circuit for Task :

Acknowledgments

Alhamdulillah, all praise and gratitude be to Allah SWT for His blessings, guidance, and mercy that enabled us to successfully complete this project. Throughout the development process, we faced several challenges, but with His will and guidance, we managed to overcome them and complete the project successfully.

We would also like to express our heartfelt appreciation to our lecturer for their continuous guidance, support, and valuable feedback throughout this project. Their advice has greatly helped us in understanding both the theoretical and practical aspects of microcontroller systems and digital display design.

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 report** has been **verified by us.**

Signature: *Amir*	Read [/]
Name: Mohamad Amir Bin Mohamad Yusoff	Understand [/]
Matric Number: 2314687	Agree [/]
Signature: *Adam*	Read [/]
Name: Adam Hakimi Bin Shah Nur Haizam	Understand [/]
Matric Number: 2314195	Agree [/]
Signature: *Ain*	Read [/]
Name: Tengku Nurul Ain Binti Tengku Azeezee	Understand [/]
Matric Number: 2313682	Agree [/]