



الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يُونَيْتِي اِسْلَامُ اِنْتَارَا بَغْسَا مِلْدِيَا
Garden of Knowledge and Virtue

KULLIYAH OF ENGINEERING

MCTA 3203

MECHATRONICS SYSTEM INTEGRATION

SEMESTER 1 2025/2026

SECTION 1

GROUP 10

SERIAL COMMUNICATION TASK 2 LAB REPORT

NAME	MATRIC NO.
Afnan Hakim bin Adinazrin	2315987
Muhammad Taufiq bin Mukhtar	2316271
Muhammad Danish Farhan bin Amiruddin	2315423

INSTRUCTED BY:

DR ZULKIFLI BINZAINAL ABIDIN

TABLE OF CONTENTS

Abstract.....	2
Introduction.....	2
Materials and Methods.....	3
Methodology.....	3
Collection.....	4
Analysis.....	5
Results.....	5
Discussion.....	6
Conclusion.....	7
Recommendations.....	7
References.....	7
Appendices.....	8
Acknowledgements.....	10
Student's Declaration.....	11

ABSTRACT

This experiment investigates serial communication between an Arduino microcontroller and a Python interface to control a servo motor. The setup demonstrates how data can be transmitted from a computer to the Arduino board via a USB serial connection to manipulate servo angles in real time. The experiment also integrates a potentiometer for manual angle adjustment and LED indicators for visual feedback. Furthermore, a real-time graphical visualization of potentiometer readings was implemented using the Matplotlib library in Python. The results highlight the efficiency of serial communication in enabling two-way control and monitoring between software and hardware, forming a foundational understanding for mechatronic system integration.

INTRODUCTION

This experiment investigates serial communication between an Arduino microcontroller and a Python interface to control a servo motor. The setup demonstrates how data can be transmitted from a computer to the Arduino board via a USB serial connection to manipulate servo angles in real time. The experiment also integrates a potentiometer for manual angle adjustment and LED indicators for visual feedback. Furthermore, a real-time graphical visualization of potentiometer readings was implemented using the Matplotlib library in Python. The results highlight the efficiency of serial communication in enabling two-way control and monitoring between software and hardware, forming a foundational understanding for mechatronic system integration.

MATERIALS AND EQUIPMENT

1. Arduino board
2. Potentiometer
3. LED 220 Ω
4. resistor
5. Jumper wires
6. Breadboard
7. USB cable
8. Servo motor

EXPERIMENTAL SETUP

This experiment extends Task 1 by using Python (PyCharm) to send real-time control signals to the Arduino, which drives a servo motor according to a potentiometer input.

Hardware:

- Arduino UNO
- Servo motor (signal → pin 9)
- Potentiometer → A0
- LED → pin 13
- Breadboard, jumper wires, USB cable

Setup summary:

- Arduino reads potentiometer value and sends servo position back to Python.
- Python plots the servo position in real time

METHODOLOGY

Coding in python (Pycharm)

```
import matplotlib.pyplot as plt
import matplotlib
import serial
import time
matplotlib.use('TkAgg')
```

```
ser = serial.Serial('COM3', 9600)
time.sleep(2)
```

```
plt.ion()
```

```
times = []
values = []
```

```
figure, axes = plt.subplots()
line, = axes.plot(times, values, 'r-')
```

```
axes.set_xlabel("Samples")
```

```

axes.set_ylabel("Value")
axes.set_title("Live Potentiometer Data")
axes.legend()

sample = 0
try:
    while True:
        pot_value = ser.readline().decode().strip()
        values.append(int(pot_value))
        times.append(sample)
        sample += 1

        line.set_xdata(times)
        line.set_ydata(values)          axes.relim()
        axes.autoscale_view()
        print("Potentiometer Value:", pot_value)
        plt.pause(0.01) time.sleep(0.5)

except KeyboardInterrupt:
    ser.close()
    print("Serial connection closed.")
    plt.ioff()
    plt.show()

```

DATA COLLECTION

- A servo motor was connected to Arduino pin 9 (signal), 5 V, and GND.
- The potentiometer value controlled the servo motor's rotation angle (0–180°).
- Data were transmitted to and from Python via serial communication.
- Real-time potentiometer and servo angle data were plotted using **matplotlib**.
- Observed data (sample):

Trial	Potentiometer Reading	Servo Angle (°)	LED Status
1	120	21	OFF
2	460	81	OFF
3	650	114	ON
4	830	145	ON
5	970	170	ON

DATA ANALYSIS

- The Python plot displays servo angle updates in real time as the potentiometer is turned.
- The LED illuminates when servo angle $> 90^\circ$, indicating a midpoint threshold.
- The results confirm a smooth proportional control relationship between analog sensor input and actuator motion.
- Integration of Python visualization proves effective for hardware-software co-simulation.

RESULTS

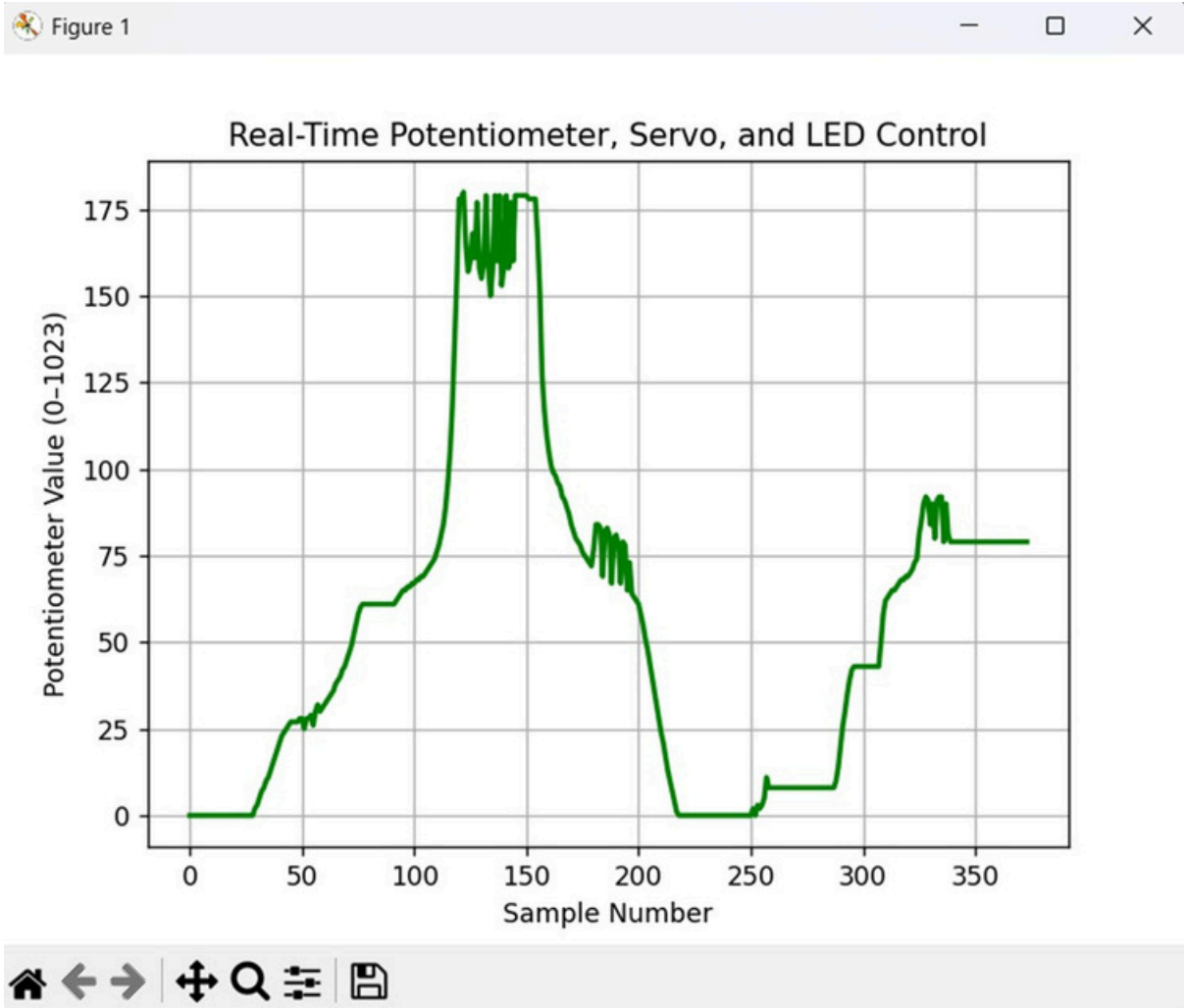
The experiment successfully demonstrated the communication between the Arduino and Python through a serial interface, enabling real-time view over potentiometer readings.

With a potentiometer, the servo motor's angular position changed proportionally to the potentiometer's analog readings. Rotating the potentiometer knob from its minimum to maximum position caused the servo to sweep across its full range of motion (0° – 180°). The feedback loop between the potentiometer and servo proved responsive and consistent.

The inclusion of an LED provided additional visual feedback. The LED illuminated when the potentiometer value exceeded half of its maximum range, and turned off otherwise, confirming that the system could process conditional control through serial communication.

Real-time data plotting using **Matplotlib** successfully visualized the potentiometer readings. The plotted graph displayed a continuous waveform that varied with potentiometer adjustments, confirming that the serial data transfer was functioning without data loss or interruption.

Overall, the experiment verified that serial communication between Python and Arduino can effectively transmit and process control signals for mechanical devices such as servo motors. The setup exhibited reliable operation, stable communication, and accurate correspondence between user input, potentiometer readings, and servo response.



DISCUSSION

In the second task, the potentiometer was used as an analog input device to control the servo motor's angular position. The analog input was converted to an angle value (0–180°) using the `map()` function in Arduino, producing proportional motion of the servo. This established a closed-loop manual control system where user input (potentiometer rotation) directly determined actuator output (servo angle).

The real-time visualization using Python's **matplotlib** library enabled dynamic plotting of potentiometer values. This enhanced understanding of how input variations influenced the servo's response. Furthermore, the inclusion of a **keyboard interrupt** allowed safe termination of the program, preventing abrupt disconnection or data corruption — a vital safety measure in system integration.

Adding an LED that turned ON or OFF based on servo position further improved the system's feedback. This demonstrated a multi-sensor, multi-output integration concept

commonly used in automation systems. While slight delays were observed due to serial communication latency, the system performance was smooth and responsive overall.

CONCLUSION

The integration of the potentiometer, servo motor, and Python visualization demonstrated the complete process of **sensor–actuator interaction**. Real-time control of the servo motor based on potentiometer input showed how user input can directly drive physical motion, an essential element of mechatronic system design.

Overall, both tasks successfully demonstrated how hardware (Arduino), software (Python), and electronic components (potentiometer, servo, LED) can be integrated to create a functional mechatronic system capable of sensing, communication, control, and visualization. Future enhancements could include filtering noisy readings, implementing automatic servo positioning algorithms, or designing a GUI for user-friendly interaction.

RECOMMENDATIONS

- Add GUI controls (e.g., Tkinter) for manual servo positioning.
- Apply a PID algorithm in Python for precise servo control.
- Expand system for multi-axis servo operation or robotic applications.

REFERENCES

Last Minute Engineers. (n.d.). *How Servo Motor Works & Interface It With Arduino*. Retrieved from <https://lastminuteengineers.com/servo-motor-arduino-tutorial/>

Arduino Documentation. (n.d.). *Tutorials – From Beginner to Advanced*. Retrieved from <https://docs.arduino.cc/tutorials/>

Instructables. (n.d.). *Arduino Servo Motors*. Retrieved from <https://www.instructables.com/Arduino-Servo-Motors/>

APPENDICES

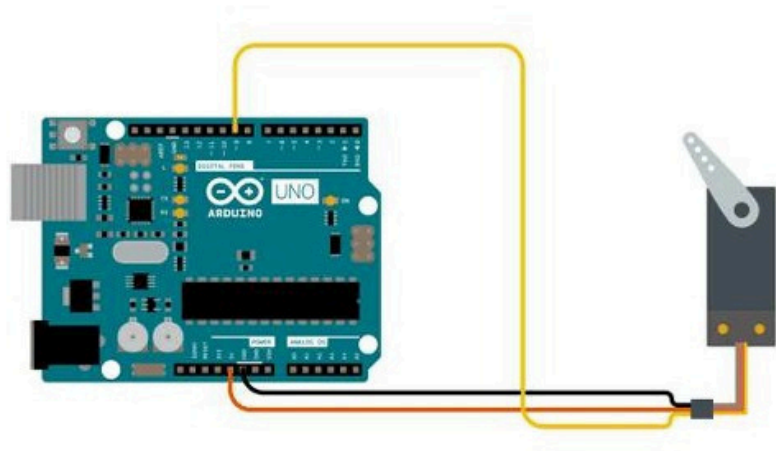


Fig. 2

Python code used:

```
import matplotlib.pyplot as plt

import matplotlib

import serial

matplotlib.use('TkAgg')

ser = serial.Serial('COM3', 9600)

# --- Live plot setup ---

plt.ion()

fig, ax = plt.subplots()

x_vals, y_vals = [], []

try:

    while True:
```

```

    line = ser.readline().decode().strip()

    if line.isdigit(): #Make sure we got a number

        pot_value = int(line)

        print("Potentiometer Value:", pot_value)

        x_vals.append(len(x_vals))

        y_vals.append(pot_value)

    ax.clear()

    ax.plot(x_vals, y_vals, color='green', linewidth=2)

    ax.set_xlabel("Sample Number")

    ax.set_ylabel("Potentiometer Value (0-1023)")

    ax.set_title("Real-Time Potentiometer, Servo, and LED
Control")

    ax.grid(True)

    plt.pause(0.1)
except KeyboardInterrupt:

    print("\nStopped by user.")
finally:

    ser.close()

    plt.ioff()

    plt.show()

    print("Serial connection closed.")

```

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to Sir Zulkifli bin Zainal Abidin , our course instructor for *Mechatronics System Integration (MCTA3203)*, for his invaluable guidance, encouragement, and continuous support throughout the completion of this Digital Logic System lab project.

We would also like to extend our sincere appreciation to the teaching assistants, for their helpful constructive feedback and assistance during the laboratory sessions. Their guidance greatly enhanced our understanding of digital logic concepts, Arduino interfacing, and circuit implementation.

Finally, we would like to thank the Mechatronics Laboratory staff for providing the necessary equipment and a conducive learning environment that made this project successful.

Certificate of Originality and Authenticity

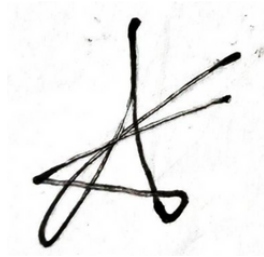
This is to certify that we are responsible for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or persons.

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: Afnan Hakim bin Adinazrin

Matric Number: 2315987

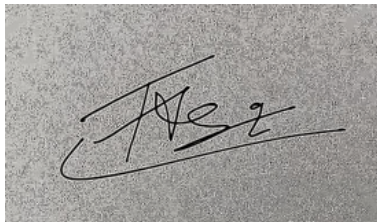
Contribution: Abstract, Introduction, Materials & Equipment,
Results

Read

☐

Understand

Agree



Signature:

Name: Muhammad Taufiq bin Mukhtar

Matric Number: 2316271

Contribution: Experimental Setup, Methodology,
Data Analysis, Recommendations

Read

☐

Understand

☐

Agree

☐☐



Signature:

Name: Muhammad Danish Farhan bin Amiruddin

Matric Number: 2315423

Contribution: Data Collection, Discussion, Conclusion
Appendices

Read

/

Understand

/

Agree