LABORATORY PROJECT REPORT
SERIAL COMMUNICATION AND SERVO CONTROL USING PYTHON AND ARDUINO
EXPERIMENT 3B

DATE : 24TH MARCH 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

DATE OF SUBMISSION :

Monday, 24TH March 2025

Abstract

This experiment explores serial communication between an Arduino board and a Python-based computer interface to control a servo motor. The objective is to establish a serial link for sending angle data from a Python script to an Arduino, which then actuates the servo motor to the specified angle. The methodology involves connecting a servo motor to an Arduino, writing Arduino and Python scripts to transmit and receive data, and observing the servo's response to user-defined inputs. The experiment highlights the importance of baud rate synchronization, serial data handling, and real-time interaction with embedded systems. The results confirm the successful integration of hardware and software, demonstrating a practical approach to serial communication for actuator control.

This experiment also explores the integration of a potentiometer to provide real-time servo control, making the setup more dynamic and interactive. The ability to modify the servo's position in real-time based on user input enhances the understanding of sensor-based control mechanisms in embedded systems.

Table of Contents

1.0 Introduction

This experiment aims to demonstrate serial communication for controlling a servo motor using Python and Arduino. The objective is to send angle values from a Python script to an Arduino via a USB serial connection, allowing the servo to move accordingly. Through this experiment, students will gain experience with embedded systems, microcontroller programming, and real-time actuator control.

Serial communication is widely used in mechatronics systems for transmitting data between sensors, actuators, and computing devices. In this experiment, a user inputs an angle in the Python script, which is then transmitted to the Arduino. The Arduino interprets the data and moves the servo to the corresponding angle using the Servo library.

The hypothesis states that if the system is properly set up, the servo will move smoothly to the specified angles based on user input. This validates the functionality of serial communication and embedded system programming for actuator control. Additionally, integrating a potentiometer allows for continuous, real-time servo adjustments without requiring constant user input through the Python script.

2.0 Materials and Equipment

2.1 Electronic Components

- **Arduino Uno** – Microcontroller board for handling serial communication
- **Servo Motor** – Actuator used for movement based on input data
- **Potentiometer** – Optional component for manual angle control
- **Jumper Wires** – Used for making electrical connections
- **USB Cable** – For connecting the Arduino to the computer

2.2 Equipment and Tools

- **Arduino IDE** – Software for writing and uploading Arduino code
- **Python with PySerial Library** – Used to send and receive serial data

3.0 Experimental Setup

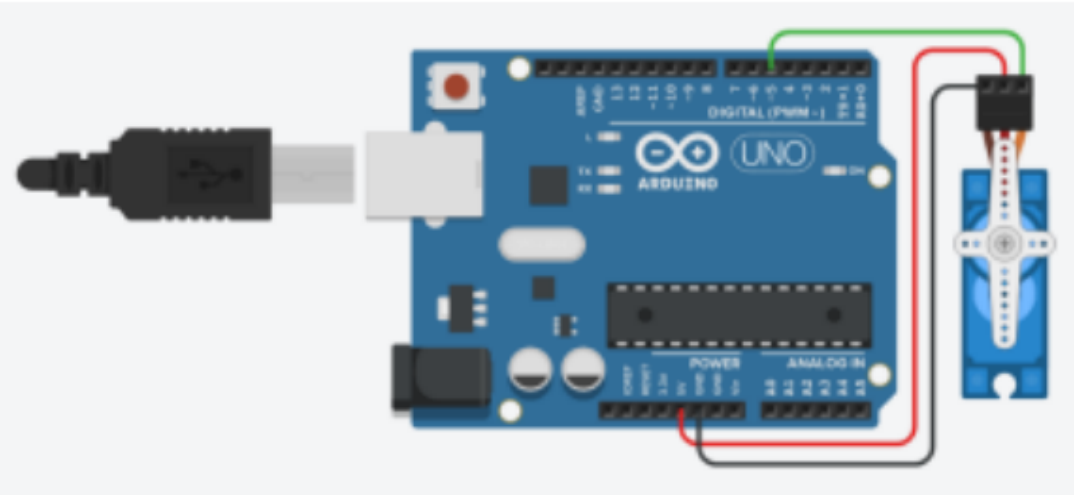

Fig. 1

Figure 3.1 Diagram of the Arduino Setup

3.1 Hardware Setup

- Connect the servo motor's signal wire to digital pin 9 on the Arduino.
- Connect the power (red) wire to the 5V pin on the Arduino.
- Connect the ground (black/brown) wire to the GND pin on the Arduino.
- Optionally, connect a potentiometer to adjust the servo angle

3.2 Coding Setup

```
#include Servo servo;
int angle = 90;
void setup()
{servo.attach(9);}
 void loop()
 {servo.write(angle);
delay(1000);
angle = 180 - angle;
 }
```

4.0 Methodology

4.1 Implementation and Testing

- Assemble the circuit by connecting the servo motor to the Arduino.
- Upload the Arduino code to initialize the servo and read incoming serial data.
- Write and execute a Python script to send angle values to the Arduino.
- Observe the servo motor's movement based on user input from Python.
- Modify the system to incorporate potentiometer-based control.

4.2 Control Algorithm

Arduino Code Logic:

1. Initialize serial communication at 9600 baud rate.
2. Attach the servo motor to digital pin 9.
3. Continuously read angle values from the serial input.
4. If the received value is between 0 and 180, move the servo accordingly.
5. If a termination signal is received, stop the servo movement.

Python Code Logic:

1. Establish a serial connection using the PySerial library.
2. Prompt the user to enter an angle value (0-180 degrees).
3. Send the inputted angle to the Arduino via the serial port.
4. Ensure safe handling of the serial connection upon termination.

5.0 Results

The experiment successfully demonstrated real-time servo control using serial communication. The servo motor accurately responded to user-defined angle inputs, moving to the specified positions without noticeable delay. When tested with a potentiometer, real-time adjustments to the servo's position were observed. The experiment confirmed stable data transmission and accurate execution of serial commands.

6.0 Discussion

The experiment validated the effectiveness of using serial communication for actuator control. The servo motor consistently responded to user inputs transmitted via Python, confirming that the system correctly interpreted and executed serial data. The integration of a potentiometer allowed real-time servo control, demonstrating the feasibility of sensor-based angle adjustments.

Possible sources of error included occasional data transmission delays, which could be minimized by optimizing the baud rate and implementing error-checking mechanisms. Additionally, using a higher-quality potentiometer may enhance precision in real-time adjustments.

Furthermore, the experiment demonstrated the importance of selecting appropriate communication protocols. The PySerial library effectively handled serial data transmission, but future implementations could explore alternative communication interfaces such as Bluetooth or WiFi for wireless control applications.

7.0 Conclusion

This experiment successfully demonstrated the control of a servo motor using Python and Arduino via serial communication. The results highlight the importance of proper baud rate synchronization, serial data handling, and real-time system integration. The methodology used in this experiment can be extended to more complex actuator control applications in mechatronics. Additionally, the integration of a potentiometer provided a deeper understanding of interactive real-time control systems.

8.0 Recommendations

To improve the system's reliability and performance, functions should be used in the Arduino and Python code to simplify repetitive tasks and improve readability. Serial Monitor debugging can be helpful in identifying communication issues. It's recommended to ensure all hardware components, especially jumper wires and potentiometers, are in good condition to avoid erratic readings. Replacing worn-out parts can significantly enhance data accuracy. For future improvements, real-time data plotting using Python libraries, integrating an OLED or I2C display for better visualization, and exploring wireless communication options like Bluetooth or WiFi could expand the system's capabilities.

9.0 References

- *Week 3b - Serial Communication (Course Material)*
- *Arduino Documentation: https://www.arduino.cc/*
- *PySerial Documentation: https://pyserial.readthedocs.io/en/latest/*

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	/