LABORATORY PROJECT REPORT

SERIAL COMMUNICATION BETWEEN ARDUINO AND PYHTON

EXPERIMENT 3A

Section 1

Group 2

Semester 1 2025/2026

NO	NAME	MATRIC NO
1.	ADAM NURUDDIN BIN HELMI	2215783
2.	NUR FATIHAH HANNANI BINTI HASMAYADI	2227062
3.	NUR SYAHZMAN AZIQ BIN MOHD SHAHMADI	2310037

Date of submission: 29th October 2025

Abstract

This experiment investigates serial communication between an Arduino Uno microcontroller and a computer running Python to achieve real-time data transmission and visualization. The setup involves interfacing a potentiometer with the Arduino to collect analog sensor data, which is then sent via a USB serial connection to a Python program using the *pyserial* library. The Python script processes and displays the incoming data graphically, allowing users to observe sensor behavior dynamically. Through this experiment, key concepts such as analog-to-digital conversion, baud rate synchronization, and data handling between hardware and software are explored. The results demonstrate successful real-time communication and visualization, emphasizing the importance of serial data exchange and sensor interfacing in embedded and mechatronic systems.

Table of Contents

1.0 **Introduction**

2.0 **Materials and Equipment**

    2.1 Electronic Components

    2.2 Equipment and Tools

3.0 **Experimental Setup**

4.0 **Methodology**

5.0 **Data Collection**

6.0 **Data Analysis**

7.0 **Results**

8.0 **Discussion**

9.0 **Conclusion**

10.0 **Recommendations**

11.0 **References**

12.0 **Acknowledgements**

13.0 **Student Clarification**

1.0 Introduction

In modern mechatronic systems, efficient communication between microcontrollers and computers is essential for data acquisition, control, and system monitoring. Serial communication provides a simple yet powerful means of transmitting data between these devices, allowing for real-time interaction and analysis. This experiment focuses on establishing serial communication between an Arduino board and a computer using Python to read and visualize data from a potentiometer sensor.

The Arduino acts as the data acquisition unit, converting the analog voltage from the potentiometer into digital values that are transmitted to the computer through a USB serial connection. A Python script, utilizing the *pyserial* library, receives and displays these readings in real-time, enabling users to observe sensor behavior dynamically.

Through this setup, students gain practical experience in fundamental concepts such as analog-to-digital conversion, serial data transmission, and synchronization of communication parameters such as baud rate. The experiment also reinforces key programming skills in both Arduino and Python while demonstrating the foundational principles behind sensor interfacing, real-time monitoring, and data visualization all of which are crucial for automation and control applications in mechatronics.

2.0 Materials and Equipment

2.1 Electronic Components

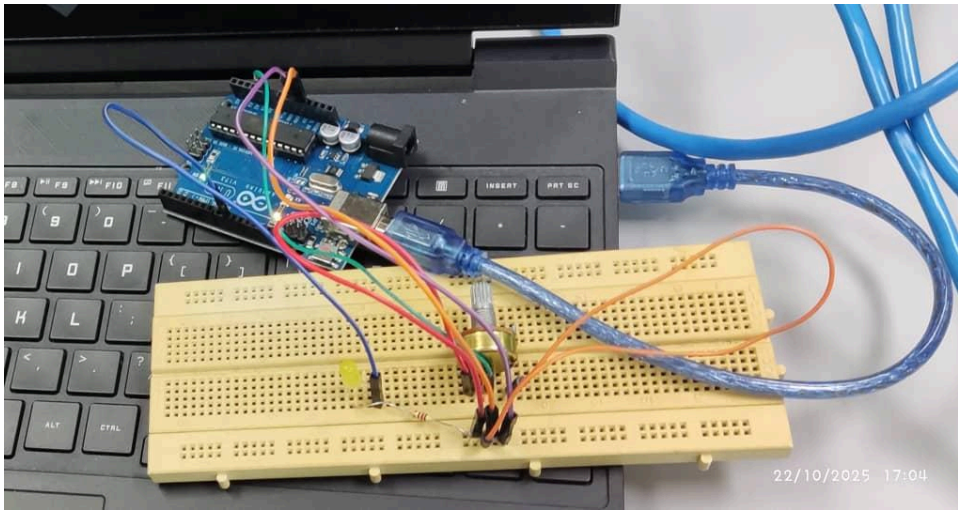
- Arduino Uno – Microcontroller board for controlling the display
- Potentiometer – A variable resistor used to generate varying analog voltage levels.
- 220-Ohm Resistors – Current-limiting resistors for each segment of the display
- LED – Optional visual indicator for circuit verification or output indication.
- Jumper Wires – For making connections between components
- Breadboard – For assembling the circuit without soldering

2.2 Equipment and Tools

- Arduino IDE – Software for writing and uploading code to the Arduino Uno
- USB Cable – For connecting the Arduino Uno to a computer
- Power Supply (5V via USB or external source) – To power the circuit
- Python with PySerial Library – Used to establish and manage serial communication with the Arduino and display real-time data.

3.0 Experimental Setup

1. The potentiometer was connected to the Arduino Uno with one leg to 5V pin on arduino, the other leg to gnd, and middle leg to pin A0.
2. Arduino was connected to the computer via USB, as shown in Figure 1, and run the Python script to begin reading potentiometer values in the terminal.
3. The potentiometer knob was adjusted to observe the values displayed in real-time
4. Serial plotter in Arduino IDE was opened to view the real time data fron the potentiometer adjustment.
5. An LED connected to the pin 9 and 220-ohm resistor added as an optional output indicator connected to a digital pin to observe visual feedback after the potentiometer value exceeded half.
6. All components were connected using jumper wires on a breadboard for a secure and solderless setup.



4.0 Methodology

1. Setup the Arduino Mega 2560
2. IDE code implementation
3. Testiong the terminal in python
4. Use the serial plotter in IDE
5. Code snippet

4.1 Detailed steps

1. Assemble the circuit as stated in the experimental setup.
2. Connect the arduino to the computer via usb cable, and upload the code to arduino IDE.
3. Open the serial plotter on the Arduino to see the real time values of the potentiometer.

4.1 Programming Codes

Arduino code

```
const int potPin = A0;    // Potentiometer connected to analog pin A0

const int ledPin = 9;     // LED connected to digital pin 9 (through 220Ω resistor)



void setup() {

  Serial.begin(9600);

  pinMode(ledPin, OUTPUT);  // Set LED pin as output
```

```
}

void loop() {
  int potValue = analogRead(potPin); // Read potentiometer value (0–1023)
  Serial.println(potValue);

  // Check if potentiometer value exceeds half of maximum (1023/2 = 511)
  if (potValue > 511) {
    digitalWrite(ledPin, HIGH); // Turn LED ON
  } else {
    digitalWrite(ledPin, LOW);  // Turn LED OFF
  }
  delay(1000); // Wait for 1 second
}
```

Python code

```
import serial
ser = serial.Serial('COM5', 9600)
try:
    while True:
        pot_value = ser.readline().decode().strip()
        print("Potentiometer Value:", pot_value)
except KeyboardInterrupt:
    ser.close()
```

```
print("Serial connection closed.")
```

5.0 Data Collection

Value	LED
0	OFF
511	OFF
512	ON

6.0 Data Analysis

The Arduino circuit reads the analog signal from the potentiometer and controls the LED based on the input value. When the potentiometer is turned, it produces a voltage between 0 V and 5 V which is then converted by the Arduino to a digital value between 0 and 1023. The program compares this value with a threshold of 511. It starts with a value of 0 and the LED remains off. When the reading is higher than 511, the LED connected to pin 9 lights up.

The Python code complements this process by continuously reading and displaying the potentiometer value sent from the Arduino via the serial port. This allows real-time reading and monitoring of how the potentiometer position affects the analog reading and the behavior of the LED.

7.0 Results

The system operated well throughout the experiment. With the potentiometer readings being accurately transmitted from the Arduino interface to Python in real time. When the potentiometer was adjusted, the LED turned on at the set value, which was above 511. No delays or errors were observed during operation, and the system responded promptly to changes in the potentiometer position, confirming that both the hardware wiring and Arduino programming were working as expected. The consistency of this response confirmed the reliability of the serial communication and proper integration between the Arduino and Python environments.

8.0 Discussion

In this project, the arduino board was interfaced with a potentiometer to measure analog voltage values that fluctuate with turning of the knob. Python and the Arduino Serial Plotter were used to visualise the data that was transmitted from the potentiometer to the computer via serial

connection. The experiment effectively illustrated the real-time reading, transmission, and presentation of analogue sensor data.

The serial measurements showed proportional voltage changes when the potentiometer was cranked. This illustrated the linear connection between the potentiometer's position and the analog-to-digital converter (ADC) output values (ranging from 0 to 1023 for a 10-bit ADC). Using the Python script allows for flexibility in data processing, logging, and future integration into control systems.

Furthermore, the LED offered a visual indication of the potentiometer's position when it was included into the circuit. When the potentiometer reading went beyond half of its range, the LED switched on, confirming that the serial connection and coding logic were operating as planned.

Avoiding conflicts between the Arduino Serial Plotter and Python serial communication, as well as making sure the COM port and baud rate setup were accurate, were some of the difficulties faced. For data transfer to go well, these parameters had to be properly synchronized.

9.0 Conclusion

The experiment's goal of leveraging USB serial communication ot transfer potentiometer readings from Arduino to Python was accomplished. The ability to precisely record and analyse the analogue readings from the Arduino in Python was validated by the real-time data sharing. The LED's inclusion made it evident how external components may be controlled using communicated data. This configuration demonstrates how flexible Arduino-Python interaction is for applications including automation, monitoring, and data collection.

10. Recommendations

To prevent communication issues, it is advised to shut down the Serial Plotter when executing the Python script. Prior to beginning the experiment, always verify the baud rate and COM port. Python data smoothing can increase reading stability, and data presentation can be improvised using visualisation tools like matpotlib. Additional sensors or actuators can be added to the experiment, and error management should be incorporated to guarantee dependable communication.

11. References

Using Python and an Arduino to Read a Sensor :
https://pythonforundergradengineers.com/python-arduino-potentiometer.html

12. Acknowledgments

Special thanks to ZULKIFLI BIN ZAINAL ABIDIN & WAHJU SEDIONO for their guidance and support during this experiment

13, Student's Declaration

Certificate of Originality and Authenticity

We hereby certify that we are collectively responsible for the work presented in this report. The content reflects our original work, except where specific references or acknowledgements have been made. No part of this report has been completed or submitted by individuals or sources not identified within.

Furthermore, we confirm that this report is the result of a collaborative effort, with contributions made by all group members. The extent of each individual's contribution is documented 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: ADAM NURUDDIN BIN HELMI	Read	[/]
Matric Number: 2215783	Understand	[/]
Signature: *ADAM*	Agree	[/]

Name: NUR FATIHAH HANNANI BINTI HASMAYADI	Read	[/]
Matric Number: 2227062	Understand	[/]
Signature: *FAT*	Agree	[/]

Name: NUR SYAHZMAN AZIQ BIN MOHD SHAHMADI	Read	[/]
Matric Number: 2310037	Understand	[/]
Signature: *AZIQ*	Agree	[/]