REPORT 3A : SERIAL COMMUNICATION

GROUP 4

MCTA 3203

SEMESTER 2 2023/2024

MECHATRONICS SYSTEM INTEGRATION

DATE OF SUBMISSION:

	NAME	MATRIC NUMBER
1.	NUR SHAHEERA BINTI KAMIS	2214040
2.	NORAMIRA HUSNA BT NORKHAIRANI	2214496
3.	MUHAMMAD EIQBAL BIN HASBOLLAH	2216911

TABLE OF CONTENT

INTRODUCTION

This study explores the integration of Arduino boards with Python scripting, showcasing a practical application: adjusting LED brightness using a potentiometer. By sidestepping traditional servo motor setups, this experiment emphasizes the simplicity of potentiometer-based control. Clear instructions are provided for hardware setup, Arduino programming, and Python interfacing, demonstrating the seamless integration of physical computing with scripting. This experiment underscores the versatility of Arduino and Python in creating interactive electromechanical systems for robotics, home automation, and education.

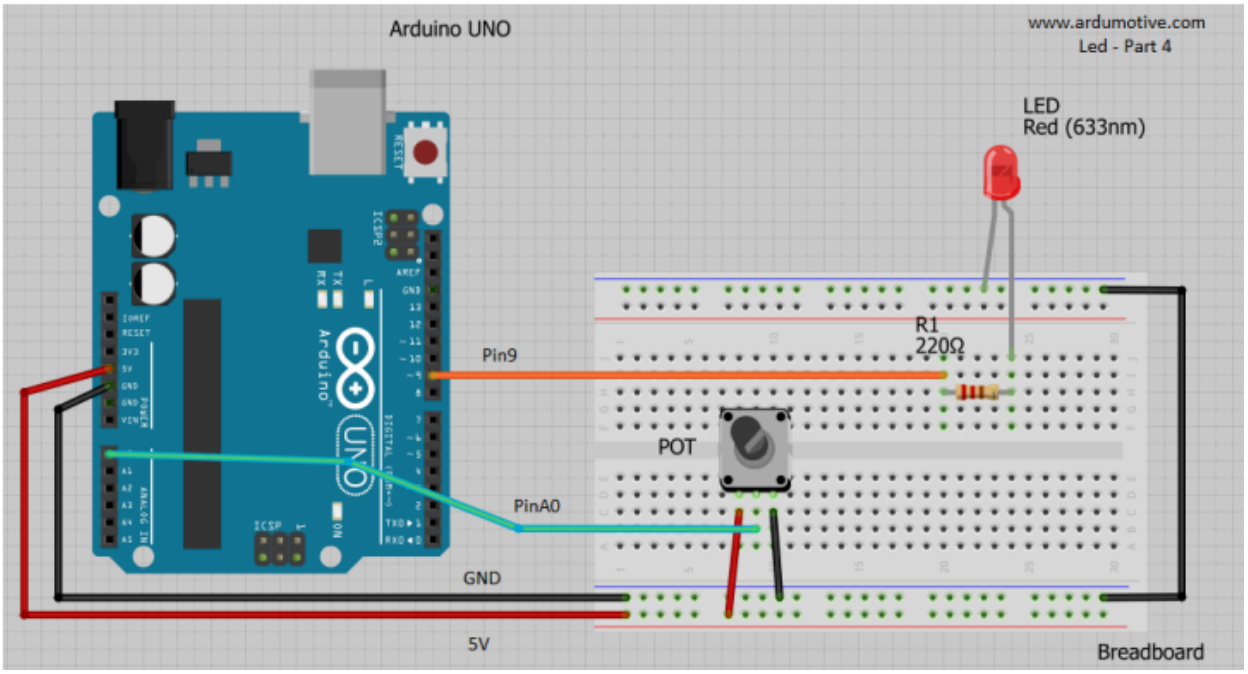
ABSTRACT

This report showcases the implementation of connecting an LED to an Arduino board and Python scripting. It demonstrates the use of a potentiometer to adjust the brightness of the LED in real-time. This experiment demonstrates the ease and efficacy of using a potentiometer as an input device by avoiding conventional servo motor configurations. The process of hardware setup, Arduino programming, and Python interface is explained clearly and concisely with precise instructions. This experiment highlights the smooth combination of physical computing with software scripting, providing a flexible platform for interactive projects in several fields, ranging from robotics to educational initiatives.

MATERIALS AND EQUIPMENT

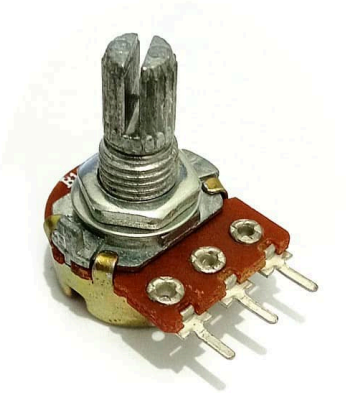
- Arduino Mega 2560 board
- Led
- Jumper wires
- Potentiometer
- USB cable for Arduino
- Resistors

EXPERIMENTAL SETUP



METHODOLOGY

Connect the potentiometer and led to the Arduino board.

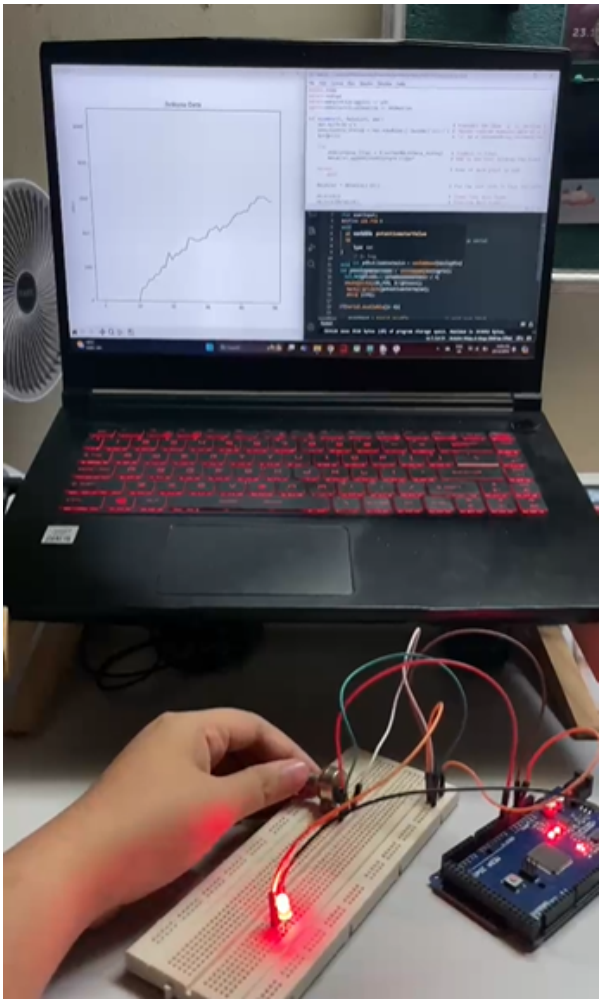


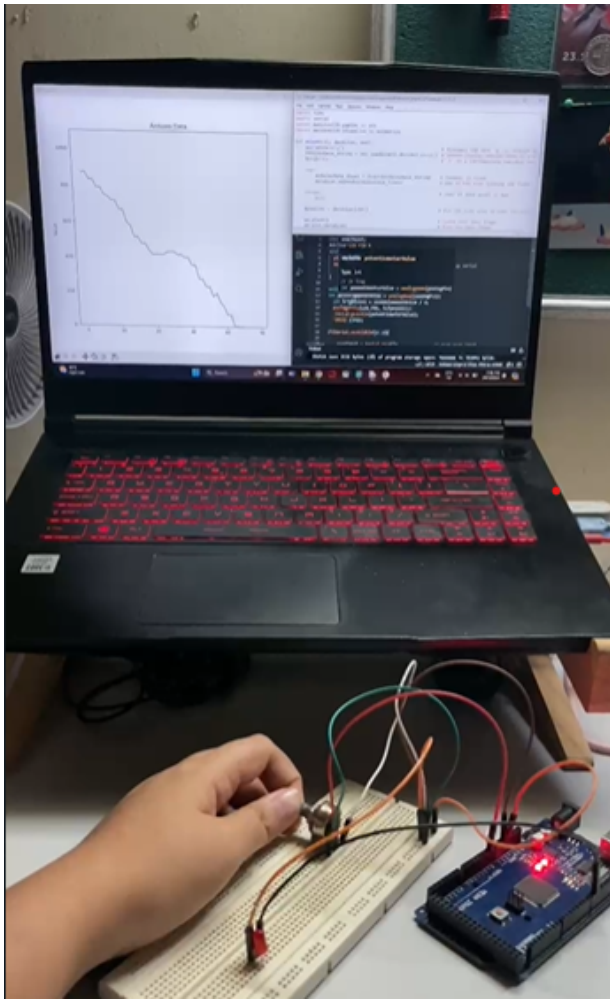
- Connect the VCC (LEFT) to 5V in Arduino
- Connect the Ground (RIGHT) to GND in Arduino
- Connect the output (MIDDLE) to PWM in the Arduino. In this experiment we use pin A0.

PROCEDURE

1. Built the circuit per the setup instructions provided.
2. Uploaded the provided Arduino code to your Arduino Uno and Python.
3. Control the LED brightness by controlling potentiometer by inserting the value on python that we already run the code.

RESULTS





DISCUSSIONS

- **Hardware**

  In order to ensure smooth data interchange in the process of establishing serial communication between Python and an Arduino, hardware components are essential.

Firstly, it is necessary that the Arduino board and the computer's USB port work together. This makes possible the real physical connection needed for communication.Connecting the analogue output of the potentiometer to the analogue input pins on the Arduino board is the first step in integrating it with the board. The process of gathering analogue sensor data starts with this phase. Effective operation also depends on making sure the potentiometer and Arduino have a steady power source. This involves thinking about how to integrate more sensors or devices while maintaining dependable serial connections. Making plans for growth ensures that the system will continue to be adaptable and responsive to changing needs.

- **Electrical**

  The setup's electrical components involve switching the potentiometer's analogue output into digital information that can be sent via serial connection. The accuracy of the data is directly affected by the analog-to-digital conversion methods used by the Arduino in this process. Potentiometer readings can be improved by using signal conditioning techniques, such as noise filtering or calibration processes. To reduce noise interference and maintain stable and clear signals during data transfer, proper grounding is essential.To avoid damage, make sure the Arduino and any connected sensors are voltage-compatiblecompatible. In order to maintain the functionality of the components, this can involve setting protection mechanisms or voltage level shifting into place. Adding circuit protection mechanisms also provides security against electrical failures or brief incidents.

- **Software**

Within the software domain, the data exchange mechanism between the Arduino and Python script is determined by the specified serial communication protocol. To achieve free of errors, effective data transmission, the right baud rate must be used.The goal of code ways to optimise is to make the Python and Arduino code more readable and efficient.In the Python script, data processing and interpretation involves extracting appropriate information out of the serial stream and preparing it for additional study or visualization. Error management methods are necessary to deal with unexpected events, including problems with communication.Plotting potentiometer readings dynamically as they are received is the final way that real-time data visualization improves the user experience. Making use of libraries or frameworks for visualization makes it easier to analyze and understand data, allowing users to make reasonable choices.

QUESTION

To present potentiometer readings graphically in your Python script, you may enhance your code by introducing the capability to generate and showcase a graph. This graphical visualization can deliver a more intuitive and informative perspective for data interpretation. Be sure to showcase the steps involved in your work (Hint: use matplotlib in your Python script)

Ans:

Here's a step-by-step explanation of our enhanced code

- Arduino Code:
    1. Declare Variables:

```
int analogPin = 0;
```

```
int data = 0;

char userInput;
```

2. Define LED Pin:

```
#define LED_PIN 9
```

3. Setup Function:

```
void setup(){

   pinMode(LED_PIN, OUTPUT);

   Serial.begin(9600);

}
```

4. Loop Function:

```
void loop(){

   int potentiometerValue = analogRead(analogPin);

   int brightness = potentiometerValue / 4;        LED brightness

   analogWrite(LED_PIN, brightness);

   Serial.println(potentiometerValue);

   delay (100);


   if(Serial.available() > 0){
```

```
        userInput = Serial.read();
        if(userInput == 'g'){
            data = analogRead(analogPin);
            Serial.println(data);
        }
    }
}
```

- Python Code:

    1. Import Necessary Libraries:

```
import time
import serial
import matplotlib.pyplot as plt
import matplotlib.animation as animation
```

    2. Define Animation Function:

```
def animate(i, dataList, ser):
    ser.write(b'g')
    arduinoData_string = ser.readline().decode('ascii')
    try:
        arduinoData_float = float(arduinoData_string)
        dataList.append(arduinoData_float)
```

```
except:
    pass
dataList = dataList[-50:]
ax.clear()
ax.plot(dataList)
ax.set_ylim([0, 1100])
ax.set_title("Arduino Data")
ax.set_ylabel("Value")
```

3. Initialize Data List and Figure:

```
dataList = []
fig = plt.figure()
ax = fig.add_subplot(111)
```

4. Establish Serial Communication:

```
ser = serial.Serial("COM3", 9600)
time.sleep(0.1)
```

5. Define Animation:

```
ani = animation.FuncAnimation(fig, animate, frames=100, fargs=(dataList, ser),
interval=10)
```

6. Display Plot:

```
plt.show()
```

7. Close Serial Connection:

```
ser.close()
```

CONCLUSION

In conclusion, the experiment demonstrated a solid structure for real-time data gathering and visualisation that smoothly integrated Arduino and Python technologies. The implementation of serial communication and the use of matplotlib for graphing provided a realistic avenue for monitoring dynamic events and conducting meaningful analysis. The installation of a graphical user interface (GUI) improved user involvement by providing easy control over experiment parameters and data visualisation settings. Furthermore, the addition of data recording and exporting features assured data retention and permitted offline analysis, emphasising the experiment's importance in scientific research, technical applications, and instructional settings.

RECOMMENDATION

- Instead of Matplotlib, we could create a basic graphical interface with Python modules such as Tkinter or PyQt. This allows us to simply change experiment settings, start/stop data collecting, and tweak plot parameters.
- Implement a functionality that allows you to log data to a file (for example, CSV format) straight from your Python script. Also, we may export plotted data or snapshots for

offline analysis. This assures data protection and allows users to undertake detailed analyses using other tools.

- Use checksums or CRC to ensure data integrity, and build handshake procedures for reliable transfer. This guarantees precise data transport, particularly in loud surroundings.

ACKNOWLEDGEMENT

A special thanks goes out to Dr. Wahju Sediono, Dr. Ali Sophian, 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 that no further improvement on the reports is needed from any of the individual 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: 	**Read**	/
Name: Noramira Husna Bt Norkhairani	**Understand**	/
Matric No: 2214496	**Agree**	/

Signature: 	**Read**	/
Name: Nur Shaheera Bt Kamis	**Understand**	/
Matric No:2214040	**Agree**	/

Signature: 	**Read**	/
Name: Muhammad Eiqbal bin Hasbollah	**Understand**	/
Matric No: 2216911	**Agree**	/