



الجامعة الإسلامية العالمية ماليزيا  
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA  
يُونِيسَيْتِي إِسْلَامُ أَنْتَارَا بَغْسِيَا مَلَيْسِيَا  
*Garden of Knowledge and Virtue*

CTA 3203 (MECHATRONICS SYSTEM INTEGRATION)  
WEEK 2: DIGITAL LOGIC SYSTEM

**TITLE:**

**Parallel, Serial, and USB Interfacing (Part 1): Examining interfacing methods with microcontrollers and computer-based systems for sensors and actuators.**

**SEMESTER 1, 24/25**

**SECTION 1 – GROUP 9**

**LECTURER:  
DR. WAHYU**

no	Name	Matric number
1	AZMI BIN BASHARUDIN	2211387
2	HAMZAH FAISAL	2219537
3	KHAIRULDDIN BIN ZULKIFLEI	2210527
4	MUHAMAD IZZUDIN BIN MUHAMAD	2219735
5	MUHAMMAD AIMAN BIN MOHD RODZI	2118813

DATE OF EXPERIMENT: 23 OCTOBER 2024  
DATE OF SUBMISSION: 29 OCTOBER 2024

## **ABSTRACT**

This week's experiment has two parts: the first for the LED and potentiometer, and the second for the servo motor. The primary goal of this experiment was to ensure that the LED could be turned on using a potentiometer. The Arduino's potentiometer reading will be passed. The second step was controlling the servo motor with Arduino interaction. Once the Arduino has interpreted the data, the servo motor will move at the given angle.

## TABLE OF CONTENT

NO	CONTENT	PAGE NUMBER
1	INTRODUCTION	4
2	MATERIALS AND EQUIPMENT	5
3	EXPERIMENT SETUP	6,7,8
4	METHODOLOGY	9,10
5	DATA COLLECTION	11
6	DATA ANALYSIS	12
7	RESULT	12
8	DISCUSSION	13
9	CONCLUSION	14
10	RECOMMENDATION	15
11	REFERENCES	16
12	APPENDICES	17
13	ACKNOWLEDGEMENT	18
14	STUDENTS DECLARATION	19,20

## 1. INTRODUCTION

The goal of this project is to use serial communication Arduino to transfer the readings from the Arduino script over a USB connection. There are two components to the experiments that use this implementation.

In the first part A, the Arduino script should display the potentiometer values when we turn its knob after the first section, A, sends potentiometer readings. According to the theory, the potentiometer functions as a voltage divider when its knob is rotated since it is a variable resistor. The analog voltage produced by the potentiometer will be read by Arduino. Through the use of the "serial plotter" module, Arduino can also create and display graphs. Having that, it is to have a responsive and fluid depiction of the potentiometer readings and to see the graph and potentiometer readings in Arduino as the potentiometer knob is turned.

In part B, a servo motor that was linked to an Arduino board is controlled via serial communication. The servo motor is actuated to a predetermined angle by transmitting angle data to Arduino. By improving the Arduino code, we can have a potentiometer for real-time servo motor angle adjustments. The Arduino can also pause execution by pressing the "q" key on the computer keyboard. Serial communication is perfect for real-time applications since it is based on the idea of asynchronous data transfer, in which information is transmitted one step at a time.

## **2. MATERIALS AND EQUIPMENT**

### **PART3A: LED and Potentiometer**

- Arduino
- Potentiometer
- JumperWires
- LED
- 220 $\Omega$ resistor
- Breadboard

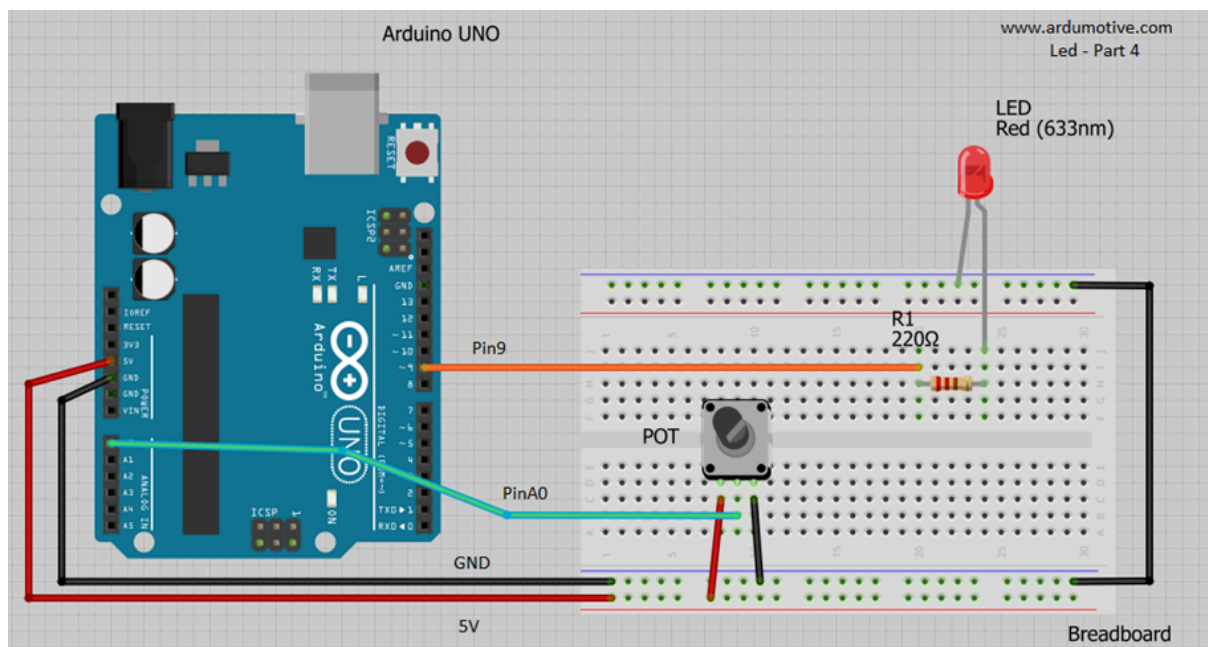
### **PART3B: Servo Motor**

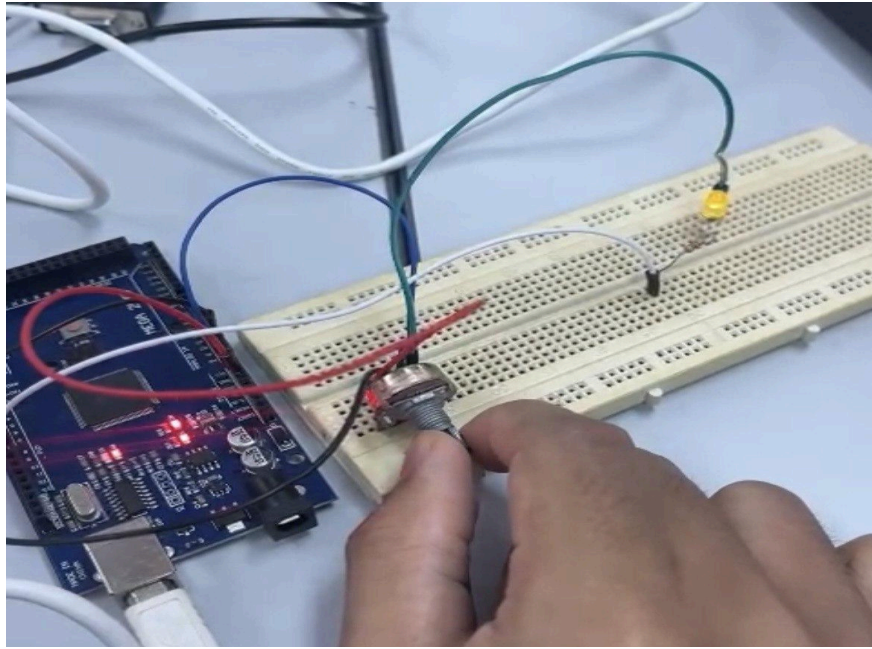
- Arduino Board
- ServoMotor
- JumperWires
- Potentiometer
- USBcablefor Arduino
- Computer with Arduino and Python installed

### 3. MATERIAL SETUP

#### PART 3A:LED & Potentiometer

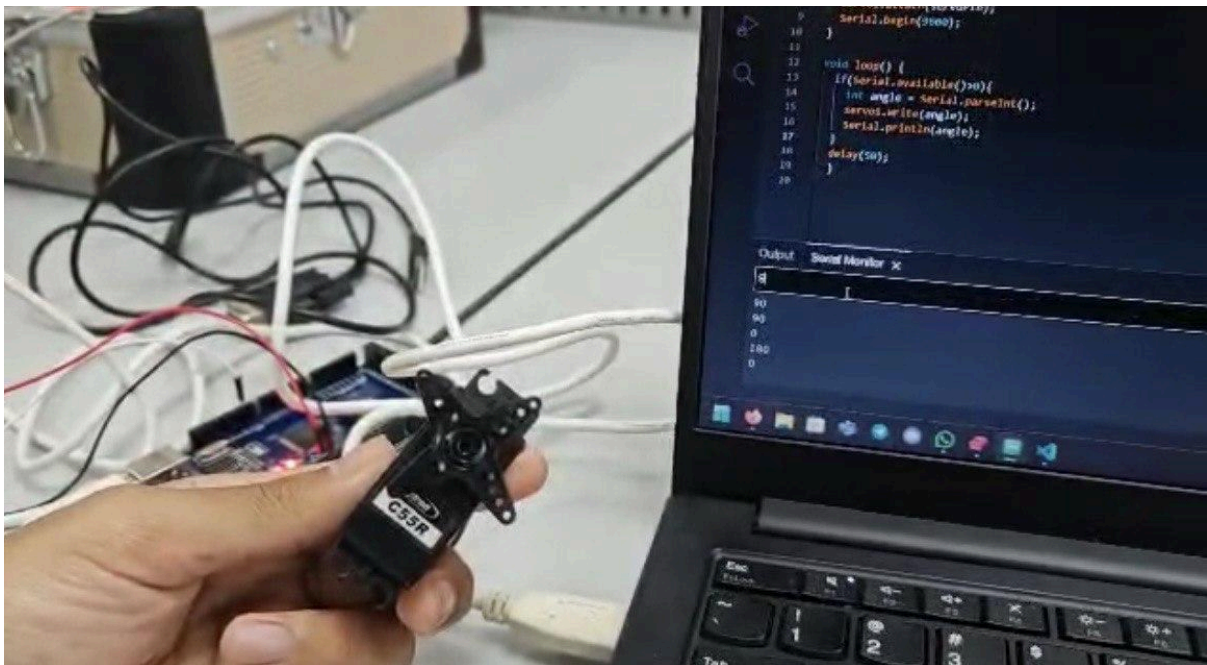
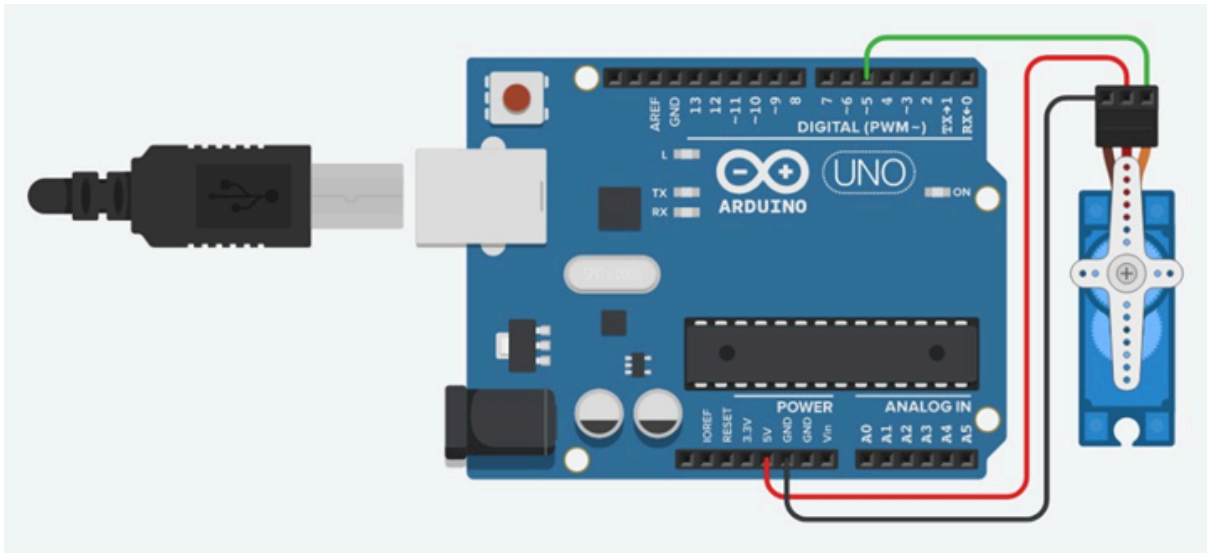
For the experiments, connect two legs of the potentiometer to 5V and ground on the Arduino. Meanwhile, the centre leg is connected to Arduino's A0 analog input pin. The negative leg, or cathode, has a shorter leg and links to the earth. All of the components are connected with jumpers. The Arduino is now connected to the computer via a USB cord. The Arduino IDE is used to code the instructions, which are then uploaded to the Arduino for execution. There is no additional setup needed to read the potentiometer readings graphically.





### **PART 3B: Servo Motor**

In part B, the servo motor featured three signal wires: PWM-capable pin (orange), power wire (red), and ground wire (brown). Connect the PWM to digital pin 10, the power wire to the Arduino's 5V output, and the ground wire to ground. The 'Servo' library is used to write the 5 instructions. It reads angle data from the serial port and moves the servo as needed. Then upload to the Arduino board. The range is due to limits; the type of servo motor utilised falls within that range. To spin the servo motor to the desired angle, enter any value within the range and press the 'enter' key on the keyboard. To add a potentiometer for real-time servo motor angle adjustments, connect two legs to the Arduino's 5V and ground pins. Meanwhile, the centre leg is connected to Arduino's A0 analog input pin.





## 4. METHODOLOGY

### **PART3A : LED & Potentiometer**

Part 3A will focus on fulfilling the Arduino. The first step involves connecting the potentiometer to the Arduino. To wire the potentiometer, connect the outer pins to 5V and GND, and the central pin to analog pins like A0. The Arduino code will read the potentiometer values and communicate them over the serial port. Communication is considered established when real-time potentiometer values are displayed in the Serial Plotter while turning the knob. To turn on the LED, perform these procedures and use the interpreted data from the potentiometer.

The programming codes used as below:

```
const int potPin = A0;
const int ledPin = 9;

void setup() {

  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {

  int potValue = analogRead(potPin);
  Serial.println(potValue);
  int pwmValue = map(potValue, 0, 1023, 255, 0);
  analogWrite(ledPin, pwmValue);
  delay(10);
}
```

## **PART 3B: Servo Motor**

Part 3B shows how to integrate a servo motor with an Arduino. The first step is to connect the servo's signal line to an Arduino PWM-capable pin. We will link this portion to digital pin 9. The Arduino's GND and 5V pins power the servo. Connect 5v to the servo's power wire and ground pin to the ground wire. The uploaded Arduino code will read angle data from the serial monitor and run the servo. The Arduino code will show the requested angle data.

The programming codes used as below:

```
#include<Servo.h>

Servo servo1;
const int servoPin = 9;
void setup() {
  int angle;
  servo1.attach(servoPin);
  Serial.begin(9600);
}

void loop() {
  if(Serial.available()>0){
    int angle = Serial.parseInt();
    servo1.write(angle);
    Serial.println(angle);
  }
  delay(50);
}
```

## **5. DATA COLLECTION**

### **PART 3A: LED & Potentiometer**

LED Behavior: In this phase, there was close monitoring of how the LED responded to potentiometer knob adjustments. The objective was to understand how the LED responded to changes in the potentiometers resistance.

Potentiometer Readings: This involved collection of analog readings using Arduino .The objective was to assess the potentiometer performance and influence on the overall system.

Graphical Visualisation: In order to visualise the graphs and understand the behaviour of the potentiometer over time, Matplotlib in Python was used to create graphs of potentiometer data.

### **PART 3B: Servo Motor**

Servo Motor Control: In this phase, there was observation of the servo motor responses to Arduino sent angle commands. The initial position of the servo motor was noted as well as its movements according to changes in angles. The data collected was used for the following data analysis.

## **6. DATA ANALYSIS**

### **PART 3A: LED & Potentiometer**

When we rotate and calibrate the potentiometer, it will influence the behaviour of an LED, causing it to illuminate. The resulting data is not only visualised and displayed within the is also presented in the serial plotter, offering a graphical representation of this interaction.

### **PART 3B: Servo Motor**

The servo motor responds to angle commands sent by the arduino code through a serial connection with an Arduino Mega. It starts at 90 degrees and adjusts its position based on the serial monitor input. The angle is limited to a range of 0 to 180 degrees, ensuring that the servo stops moving if a command exceeds these boundaries.

## **7. RESULT**

Serial communication between the Arduino was successfully established. Data transmission occurred seamlessly, allowing for real-time interaction. The Arduino accurately reads the potentiometer's analog values. This enabled the arduino to access and utilise the potentiometer readings for further processing. The experiment effectively demonstrated the remote control of a servo motor using arduino. As potentiometer values changed, the Arduino, instructing the servo motor to adjust its angle.

In conclusion, the experiment successfully met its goals, offering practical insight into setting up serial communication, sending potentiometer data, and showcasing remote servo motor control.

## 8. DISCUSSION

**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 visualisation 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).

**Answer:** To develop and display the graph, we first imported the necessary libraries, including serial plotter. When we turned the potentiometer knob on/off, the Arduino code displayed the reading. The serial plotter graph was displayed based on the potentiometer knob value in the Arduino code. The code was included in the methodology section.

**Question:** Enhance your Arduino and Python code to incorporate a potentiometer for real-time adjustments of the servo motor's angle. Ensure that, in the updated Arduino code, you can halt its execution by pressing a designated key on your computer's keyboard. Following the modification, restart the Python script to receive and display servo position data from the Arduino over the serial connection. While experimenting with the potentiometer, observe the corresponding changes in the servo motor's position.

**Answer:** Using the revised Arduino code from the Methodology section, we saw that turning on/off the potentiometer causes the value to increase and decrease on the serial monitor. When we push the 's' character, the Arduino code terminates the serial connection Arduino.

## 9. CONCLUSION

### **PART 3A: LED & Potentiometer**

The experiment successfully demonstrated the interaction between a potentiometer, LED, and Arduino communication. The LED's response to potentiometer resistance adjustments indicated the system's proper operation. The Arduino software successfully displayed real-time data. Using a serial plotter to visualise data increased our knowledge of it. The original issue with the 18 LED showed the need for careful wiring connections and attention to detail in mechatronic systems.

### **PART 3B: Servo Motor**

In this experiment, the group used Arduino to operate a servo motor. The servo motor's consistent reaction to angular commands sent via serial connection proved the communication system's reliability. The Arduino served as a mediator for command of the servo motor. Despite hurdles, the code was carefully rewritten to successfully address issues. Both studies provided valuable insights into component interactions in mechatronic systems, paving the way for further research in this area.

To sum up, these two experiments, which emphasise the effective collaboration of hardware and software components, are consistent with the complete description of the experiment. The Arduino board serves as the principal controller, precisely controlling the movements of the servo motor. This interaction demonstrates the integration of programming, data communication, and electrical connections to achieve dynamic control over hardware. The hardware arrangement serves as a good foundation for understanding how each component functions within the larger system. This part adds to the project's instructional value by stressing the simplicity and efficiency of using a USB connection for power and data transmission. It enhances comprehension of concepts like PWM and serial transmission by allowing for hands-on exploration. In conclusion, this arrangement not only exhibits useful applications in robotics and automation, but it also provides a thorough understanding of embedded system principles, emphasising the collaboration of hardware and software to produce controlled movement.

## **10. RECOMMENDATION**

The experiment's findings indicate the importance of pursuing future initiatives that strengthen the underlying abilities acquired in this framework. Additional sensors, such as temperature or ultrasonic distance sensors, would be useful in enabling the construction of more sophisticated interactions. For example, we could set up the Arduino to adjust the servo's movement in reaction to sensor data indicating environmental changes.

Furthermore, recording our trials and results may strengthen the insights acquired and provide useful recommendations for future research. This hands-on training not only reinforces academic concepts but also develops critical problem-solving abilities relevant to the disciplines of robotics and electronics. Moreover, to begin experimenting with serial communication Arduino, it's important to first grasp how it works. Learn the fundamentals of serial communication, such as baud rates, data bits, stop bits, and flow control, to maintain a stable connection. Also we should be warned that the port assigned to your Arduino may differ depending on the computer. Check and configure the correct COM port in the Arduino IDE's "Tools" menu. Make a note of the port used for each configuration to avoid confusion. Lastly, Long delays can impact real-time accuracy and responsiveness, despite their utility for monitoring changes. Optimise your code's delays to balance real-time monitoring with correctness. Consider employing tiny delays or alternate strategies, like reading data based on events.

Implementing these principles improves the reliability and efficiency of serial communication experiments, reducing potential errors. Overall, the project establishes a solid foundation for future research into embedded systems while also encouraging creativity and innovation in engineering efforts.

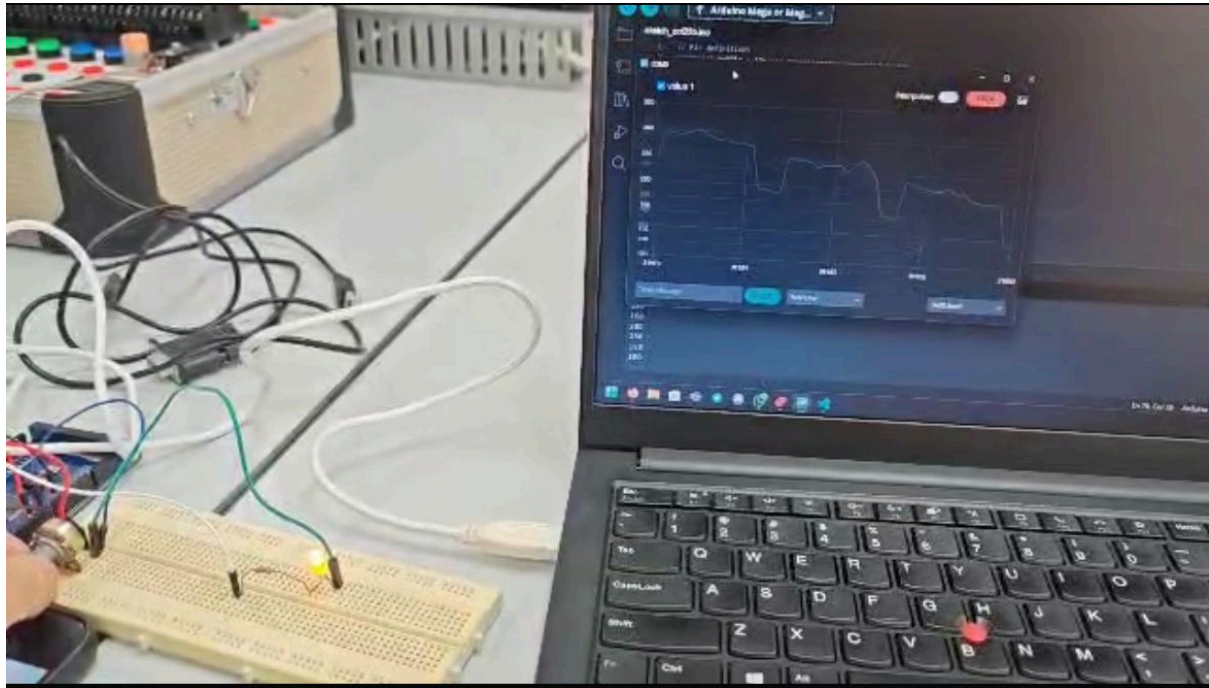
## 11. REFERENCES

- “Arduino Integrated Development Environment (IDE) v1 | Arduino Documentation | Arduino Documentation,” *Arduino.cc*, 2022.  
<https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics/>
- Zulkifli. (2014). Mechatronics Interfacing Lab Manual, (Rev. ed.). Unpublished Class Materials.

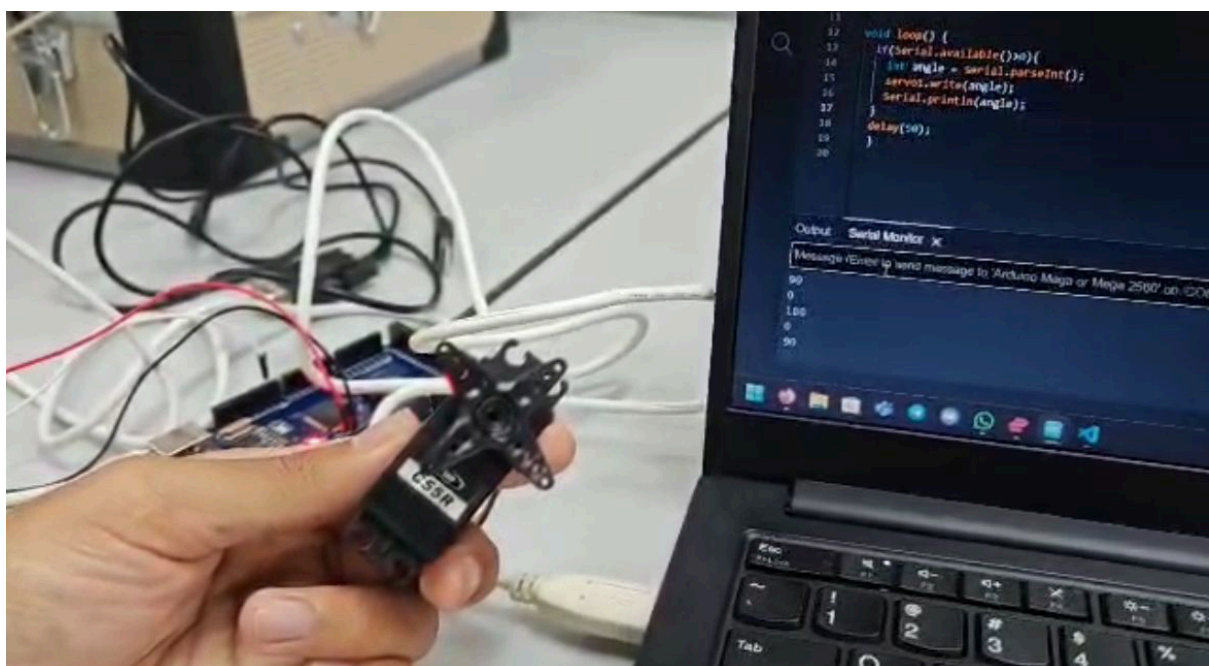


## 12. APPENDICES

### PART 3A: LED & Potentiometer



### PART 3B: Servo Motor



### **13. ACKNOWLEDGEMENT**

We want to thank everyone who helped, guided, and supported us throughout this endeavour. First and foremost, we thank Assoc Prof Dr. Zulkifli Bin Zainal Abidin, and also Dr. Wahyu Sediono for providing extensive guidance and supervision during the experiment. Their comments, suggestions, and excitement helped us understand Arduino programming.

Our fellow group members deserve special recognition for their teamwork and support. Our conversations, information sharing, and problem-solving sessions dramatically improved our understanding of the experiment's topics and the overall learning experience. Our group members' joint efforts not only enhanced our learning experience, but also greatly contributed to the effective completion of this project.

## 14. STUDENTS 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.

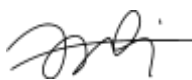
We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of the individual's contributors to the report.

We therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us**.

SIGNATURE		
NAME	Azmi bin basharudin	Read <input checked="" type="checkbox"/>
MATRIC NUMBER	2211387	Understand <input checked="" type="checkbox"/>
CONTRIBUTION	Introduction, material and equipment	Agreed <input checked="" type="checkbox"/>

SIGNATURE		
NAME	Hamzah faizal	Read <input checked="" type="checkbox"/>
MATRIC NUMBER	2219537	Understand <input checked="" type="checkbox"/>
CONTRIBUTION	Experiment setup, methodology, data collection	Agreed <input checked="" type="checkbox"/>

<b>SIGNATURE</b>		
<b>NAME</b>	Khairulddin bin zulkiflei	<b>Read</b> <input checked="" type="checkbox"/>
<b>MATRIC NUMBER</b>	2210527	<b>Understand</b> <input checked="" type="checkbox"/>
<b>CONTRIBUTION</b>	Data analysis, result, discussion	<b>Agreed</b> <input checked="" type="checkbox"/>

<b>SIGNATURE</b>		
<b>NAME</b>	Muhamad Izzudin Bin Muhamad	<b>Read</b> <input checked="" type="checkbox"/>
<b>MATRIC NUMBER</b>	2219735	<b>Understand</b> <input checked="" type="checkbox"/>
<b>CONTRIBUTION</b>	Conclusion, recommendation, references	<b>Agreed</b> <input checked="" type="checkbox"/>

<b>SIGNATURE</b>	<i>arman</i>	
<b>NAME</b>	Muhammad aiman bin mohd rodzi	<b>Read</b> <input checked="" type="checkbox"/>
<b>MATRIC NUMBER</b>	2118813	<b>Understand</b> <input checked="" type="checkbox"/>
<b>CONTRIBUTION</b>	Appendices, acknowledgement, students declaration	<b>Agreed</b> <input checked="" type="checkbox"/>

