MECHATRONICS SYSTEM INTEGRATION

EXPERIMENT 3: MICROCONTROLLER AND COMPUTER-BASED SYSTEM

GROUP NUMBER: A

PROGRAMME: MECHATRONICS

GROUP MEMBERS	MATRIC NO
AISYAH SOFEA BINTI OTHMAN	2115386
SITI ALIAA BINTI IBRAHIM	2112618
MUHAMMAD ADLI FAHMI BIN TAJUL ARIS	2113095
ETHAR ABDALLA ABDELKARIM OSMAN	2111282
NURAIN AINAA AQILAH BINTI ROSLI	2114560

DATE OF EXPERIMENT:

Wednesday, 1st November 2023

DATE OF SUBMISSION:

Wednesday, 8th November 2023

Abstract

This experiment involved the use of an MPU6050 sensor interfaced with an Arduino board to acquire accelerometer and gyroscope data, which was then transmitted to a PC via the serial port. With the Arduino code handling data acquisition and transmission, a Python script on the PC received and displayed the real-time sensor data. The setup allowed for data collection, visualisation, and analysis, facilitating interaction with the sensor and enabling experimentation in various applications, from motion tracking to gesture recognition.

In the RFID-based access control experiment, an Arduino, and Python are used to validate RFID cards. The Arduino code distinguishes between authorised and unauthorised cards, using LEDs to signal access approval or denial. When an authorised card is detected, the green LED lights up, and a servo motor activates, while an unauthorised card causes the red LED to illuminate. This experiment offers a practical illustration of how RFID technology can be applied to access control systems, demonstrating the integration of hardware components for secure access management.

Table of Contents

Objectives

1. To implement serial communication between an Arduino and Python script (4A)
2. To acquire data from the MPU6050 sensor and visualise it in real-time using Python for hand gesture recognition (4A)
3. To demonstrate the integration of an RFID card reader with an Arduino, develop Python code for RFID card authentication, and use LEDs for visual feedback (4B)

Introduction

This experiment focuses on the integration of the MPU6050 sensor, used for its motion and orientation data capabilities. The primary objective is to establish a connection between a computer and the MPU6050 sensor using an Arduino board. This integration will require proper connection and code development to create a hand gesture recognition system that relies on the sensor's accelerometer and gyroscope measurements. This will result in a system that can detect predefined hand gestures based on the sensor's data.

The second experiment revolves around the integration of an RFID and a servo motor. The aim here is to create an access control system that authenticates users using RFID card by integrating RFID card authentication and servo motor control, with the help of Python and Arduino. Python is used to communicate with the reader, receive data from RFID cards, and make access control decisions. Additionally, LEDs are incorporated into the setup for visual feedback.

Material and Equipment

PART 4A: MPU6050 sensor

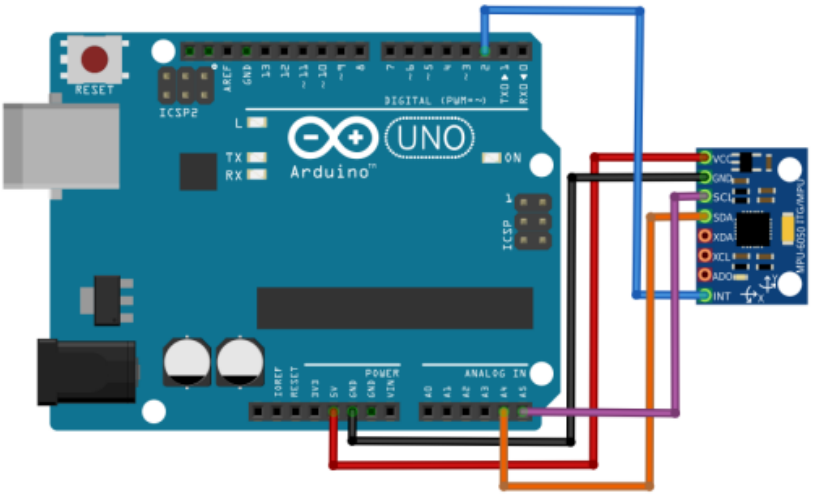
- Arduino
- Potentiometer
- MPU6050 sensor
- USB cable for Arduino
- Computer with Arduino and Python installed
- Power supply

PART 4B: RFID READER & SERVO MOTOR

- Arduino board
- RFID card reader
- RFID tags or cards
- Servo Motor
- Jumper wires
- Breadboard
- LED
- USB cable for Arduino
- Computer with Arduino and Python installed
- Power supply

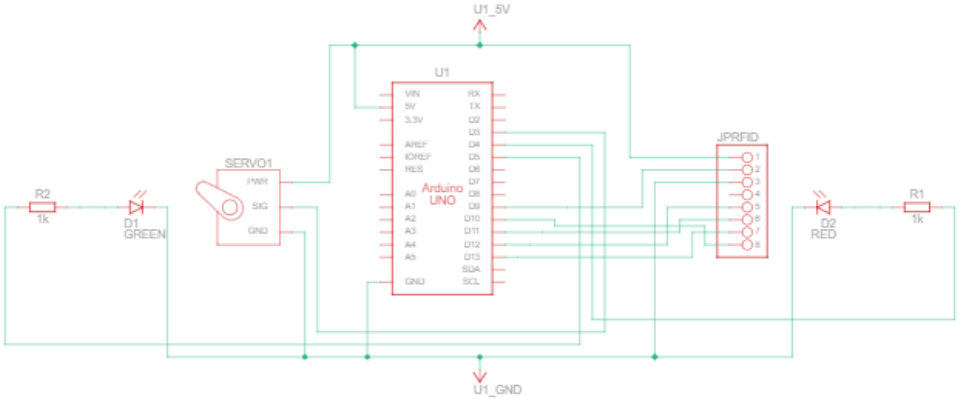
Experimental Setup

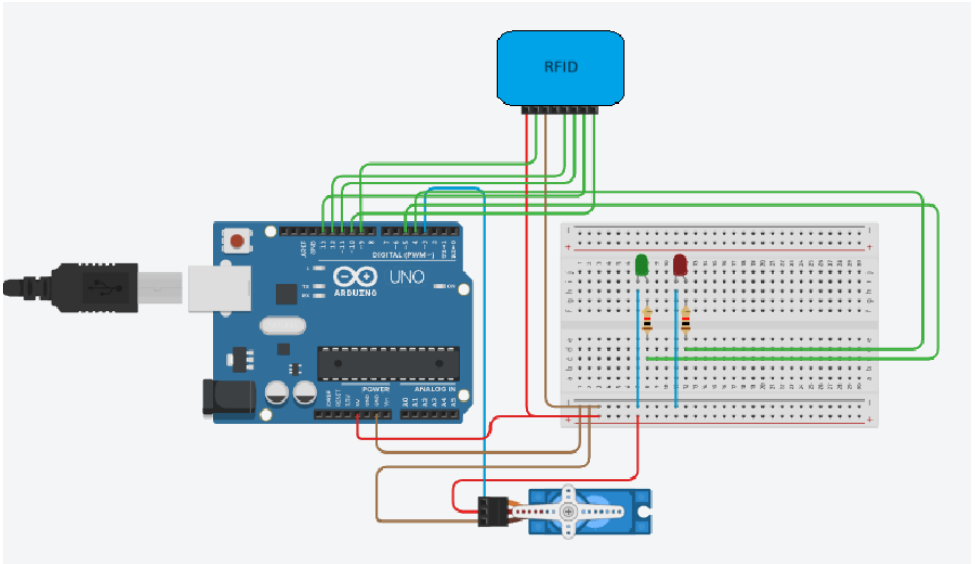
The MPU6050 module communicates over the I2C protocol, so its SDA (Serial Data) and SCL (Serial Clock) pins are connected to A4 and A5 on the Arduino board. The VCC (Voltage) pin of the MPU6050 is connected to the Arduino's 5V and the GND (ground) pin of the MPU6050 is connected to the Arduino's GND pin. Finally, the Arduino is connected to the PC for power supply.



PART 4B: RFID READER & SERVO MOTOR

For the RFID reader, VCC(1) is given to the 5V pin of Arduino, RST(2) is given to the digital pin 9, GND(3) is given to the ground pin of the Arduino, IRQ(4) is not connected. MSO(5) is connected to digital pin 12, MOSI(6) is connected to digital pin 11, SCK(7) is connected to digital pin 10, and SDA(8) is connected to digital pin 9. For LEDs, both cathode legs are connected to the ground with the help of a resistor, the anode for green LED is connected to pin 4 while for red is pin 5. Lastly for the servo motor. The red wire is connected to Arduino's VCC pin, the maroon wire is connected to the ground, and the orange wire is connected to the digital pin 3.





Methodology

PART 4A: MPU6050 sensor

In this experiment, we connected an MPU6050 sensor to an Arduino board, which read accelerometer and gyroscope data from the sensor. The Arduino then transmitted this data to a PC through a serial port at a baud rate of 9600. On the PC, a Python script received and displayed the sensor data, allowing us to interact with the sensor, gather data, and analyze it for our experiment's purposes.

Arduino	Python

#include <Wire.h>
#include <MPU6050.h>
MPU6050 mpu;
const int threshold = 400; // Adjust this threshold as needed
int previousGesture = -1;
void setup() {
Serial.begin(9600);
Wire.begin();
mpu.initialize();
}
void loop() {
int gesture = detectGesture();
 	
if (gesture != previousGesture) {
Serial.print("Detected Gesture: ");
if (gesture == 1) {
Serial.println("Gesture 1");
// Perform an action for Gesture 1
} else if (gesture == 2) {
Serial.println("Gesture 2");
// Perform an action for Gesture 2
	import serial
	import matplotlib.pyplot as plt
	from time import sleep
	ser = serial.Serial('COM5', 9600)
	dt=0.05
	while True:
	data = ser.readline().decode('utf-8').strip()
	if data.startswith("Detected Gesture: "):
	gesture = data.split(": ")[1]
	if gesture == "Gesture 1":
	# Perform an action for Gesture 1
	print("Action for Gesture 1")
	elif gesture == "Gesture 2":
	# Perform an action for Gesture 2
	print("Action for Gesture 2")
	# Add more gesture actions as needed
	if ser.is_open:
	print(f"Serial port {ser.name} is open.")
	data = []
	N = 100 # number of data
	for k in range(N):
	b = ser.readline()

```
 }
 // Add more gesture cases as needed

 previousGesture = gesture;
 }
}
int detectGesture() {
 int ax, ay, az, gx, gy, gz;
 mpu.getMotion6(&ax, &ay, &az, &gx,
&gy, &gz);
 // Perform gesture recognition here based
on sensor data
 // Define conditions to recognize specific
gestures
 if (ax > threshold && ay < threshold) {
 return 1; // Gesture 1
 } else if (ax < -threshold && ay >
threshold) {
 return 2; // Gesture 2
 }
 // Add more gesture conditions as needed
 return 0; // No gesture detected
}
```

```
    strn = b.decode()
    str1 = strn.rstrip()

    data.append(strn)

    sleep(0.05)
   ser.close()
   if not ser.is_open:
      print(f"Serial port {ser.name} is
closed.")

   # Plot the reading
   plt.plot(data)
   plt.xlabel('Time(seconds)')
   plt.ylabel('gesture')
   plt.title('')
   plt.grid(True)
   plt.show()
else:
   print(f"Failed to open serial port
{serial_port}.")
```

PART 4B: RFID READER & SERVO MOTOR

In the RFID experiment, an Arduino and Python code combination is employed to check the Data Identifier (DI) of RFID cards. The Arduino code distinguishes between authorised and unauthorised cards and controls LEDs accordingly: the green LED lights up for access-granted cards, and the red LED for denied cards. Additionally, a servo motor is used to activate when access is granted, and the motor moves. This experiment provides a practical demonstration of RFID-based access control and how different components work together to grant or deny access based on card recognition.

Arduino	Python

#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>
 	
#define SS_PIN 10 // MOSI of RFID
#define RST_PIN 9 // RST of RFID
#define LED_G 4 // Define green LED pin
#define LED_R 5 // Define red LED
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance.
Servo myServo; // Define servo name
 	
void setup()
{
Serial.begin(9600); // Initiate a serial communication
SPI.begin(); // Initiate SPI bus
mfrc522.PCD_Init(); // Initiate MFRC522
myServo.attach(3); // Servo pin
myServo.write(0); // Servo start position
	import serial
	import json
	
	with open("people.json", "r") as f:
	VIP = json.load(f)
	
	with serial.Serial('COM4', 9600) as ser:
	print("RFID Data ready!")
	
	try:
	while True:
	data = ser.readline().decode()
	if data.startswith("Card UID:"):
	card = data.split(":")[1].strip()
	if card in VIP:
	print(VIP[card]["Name"])
	ser.write("A".encode())
	else:
	print("Hello?")
	ser.write("B".encode())

```
  pinMode(LED_G, OUTPUT);
  pinMode(LED_R, OUTPUT);
  Serial.println("Put your card to the
reader...");
  Serial.println();
}

void loop()
{
  // Look for new cards
  if (!mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  // Select one of the cards
  if (!mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  // Show UID on the serial monitor
  Serial.print("UID tag :");
  String content = "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
    Serial.print(mfrc522.uid.uidByte[i] <
0x10 ? " 0" : " ");
    Serial.print(mfrc522.uid.uidByte[i],
HEX);

content.concat(String(mfrc522.uid.uidByte[i
] < 0x10 ? " 0" : " "));
```

```
    except KeyboardInterrupt:
        ser.close()
        print("Serial Close")
```

JSON :

```
{
    "03 4A 84 F5": {
        "Name": "Dli"
    },
    "33 0F B3 EC": {
        "Name": "Sof"
    }
}
```

```
content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  Serial.println();
  Serial.print("Message : ");
  content.toUpperCase();
  if (content.substring(1) == "03 4A 84 F5")
  {
    Serial.println("Authorized access");
    Serial.println();
    delay(500);
    digitalWrite(LED_G, HIGH);
    myServo.write(180);
    delay(3000);
    myServo.write(0);
    digitalWrite(LED_G, LOW);
  }
  else
  {
    Serial.println("Access denied");
    digitalWrite(LED_R, HIGH);
    delay(1000);
    digitalWrite(LED_R, LOW);
  }
}
```

Data Collection

PART 4A: MPU6050 sensor

For this experiment, acceleration and rotational velocity data are continually recorded in three dimensions (x, y, and z) via the MPU6050 sensor. This data is collected to help understand the motion and direction of the gestures detected. Additionally, this data is essential for

real-time monitoring and knowing the behaviour of the sensor in response to physical movements.

PART 4B: RFID READER & SERVO MOTOR

As for this experiment, the RFID reader scans RFID cards to get their unique identification or UID. User authentication requires this UID. Next, the system determines the authenticity of the RFID card that has been presented, indicating the status by the illumination of LEDs (green indicating permitted access and red indicating unauthorised access). Despite the constraints in the experiment, ideally, the servo motor should also move as part of the data-collecting process.

Data Analysis

PART 4A: MPU6050 sensor

When a new gesture is detected, the Arduino sends a message, and Python displays the detected gesture (for example, "Action for Gesture: 1"). Essentially, when the MPU module is held upward (the component is upward, the label is downward, and the wire connection is exactly in the front), it will read as gesture 1, indicating that it detects a different motion in the x-axis. Meanwhile, if the MPU module is pushed or tilted to the right, it will be read as gesture 2 (different in the y-axis).

However, the reading displayed in Python is not as rapid as the reading presented on the Arduino's serial monitor. When using the Python code, the MPU module must be jerked or tapped harder for it to read the motion.

PART 4B: RFID READER & SERVO MOTOR

Green LED illuminates indicating the RFID reader identifies an authorised UID is scanned and access is granted otherwise red LED will turn on when a different UID card is used which led to denied access. The servo motor is a critical component for physically granting access, but it's not behaving as expected

Result

Sensor integration for real-time gesture detection, ‘Hand Gesture Recognition’ using MPU6050 was successfully established using the connection between the sensor and Arduino board. The communication link between the two allows access to the motion and orientation data from the sensor which is then successfully displayed in python, using code that utilised the accelerometer and gyroscope measurements from the sensor.

In the second part of the experiment, USB integration for RFID card readers and the computer also was successfully established. The integration that was achieved by arduino and Python integration allowed the access control system to authenticate users using RFID cards.

In conclusion, the experiment successfully met its goals, offering practical insight and knowledge into setting up it. Thus demonstrating the potential and versatility of sensor integration and USB interfacing.

Discussion

PART 4A: MPU6050 sensor

- **Arduino**

The MPU6050 sensor is initialised by the Arduino code, which also reads accelerometer and gyroscope data. Then the data was sent to the computer via the serial port. A simple gesture recognition system was included in the Arduino code, which uses preset thresholds for accelerometer data to identify particular hand gestures. When the hand gesture is recognized, it will be transmitted to the PC.

- **Python**

Python was used to receive and then process the data that was transmitted from the Arduino. The communication between Arduino and Python is facilitated via the Python ‘Pyserial’ library. As shown in the Arduino code, the Python code interprets the data it has received and acts on the gestures it has identified.

- **MPU6050 sensor**

The MPU6050 sensor which contains both gyroscope and accelerometer was utilised to capture the data on hand motion. Using Arduino, the sensor is initialised, a threshold value is

set, and real-time sensor data is read. By using the collected sensor data, the gestures were identified and categorised either with gesture 1 or gesture 2.

Task

Create a straightforward hand gesture recognition system by capturing accelerometer and gyroscope data during the execution of predefined hand movements. Employ an algorithm to identify and categorise these gestures using the collected sensor data. Additionally, visualise the paths of hand movement in an x-y coordinate system.

To generate and plot a graph using Python, we should use the matplotlib library. When the gestures are detected, the data and reading will be shown in the Python script graph. The results that we could obtain should come from the gesture detection of the MPU6050 sensor.

PART 4B: RFID READER & SERVO MOTOR

- **Arduino**

From the Arduino code at the methodology part, the Arduino acts as a hardware bridge or intermediary that allows RFID and Python to communicate with each other. Arduino is made for a basic access control system with LEDs, a servo motor, and an RFID reader. Using a particular card UID access is granted, while different card UIDs are denied access. The code is designed to add a visual indicator, like a green LED, to light up when the RFID reader detects a recognized UID and to turn on a red LED when an unrecognised card is read.

- **Servo motor error**

The Servo motor should turn or move after access for a certain UID card is granted.
The Arduino code is supposed to be designed to offer a basic example of how to control the servo. However, due to the servo's requirement for 5V, it was unable to move or turn.

Expected: If the UID card is granted access, the servo motor should turn or move and the green LED will light up. If the UID card is denied access, the servo motor should not turn or move and the red LED will light up.

Observed: After the UID card is granted access, only the green LED is turned on however the servo motor is still and does not move or turn.

Enhance the existing code to introduce a visual indicator, such as illuminating a green LED, when a recognized UID is detected by the RFID reader, and conversely, activate a red LED when an unrecognised card is read. Incorporate structured JSON data handling within your code for better organisation and flexibility. Add some options for the user to freely set the angle position of the servo.

The green and red LEDs are added as indicators to the user. Using the Arduino and Python code that we attached in the methodology part, we could notice that the green LED will light up if the card UID that was read is given access meanwhile the red LED will turn on if the card UID that was read is not compatible with specific card UID and the access is denied.

Conclusion

PART 4A: MPU6050 sensor

In this experiment, we successfully integrated the MPU6050 sensor with an Arduino board to capture motion and orientation data, facilitating the creation of a hand gesture recognition system that relies on the sensor's accelerometer and gyroscope measurements to identify the gestures by its preset thresholds which will then transmitted to the computer via serial port to the python. The Python script effectively presented real-time data upon gestures applied to it.

PART 4B: RFID READER & SERVO MOTOR

In this experiment, we successfully integrated the RFID card reader with an Arduino board for the access control system to authenticate users using RFID cards. The Arduino-python communication was crucial for the authentication and access to the card to read the card's serial number. Thus, two LEDs (red and green) were added to the connection:

the ‘green’ light up because it recognized the card or it could read the card’s serial number, and the ‘red’ light up when it was not. Although the access control system was successful, we are not able to implement the servo motor into the connection because the servo motor needs a separate power supply.

Recommendation

The need for improvement is always needed to make sure the accuracy, reliability, and adaptability of the experiment results. Researchers can enhance their techniques and cut down on possible error sources through continuous improvement. The findings become more reliable as a result, yielding more exact and accurate outcomes. Below are a few recommendations that we can improve in the future.

1. **Enhance gesture detection:** For part 4A, we can consider refining the gesture recognition system. Instead of only detecting two simple gestures which is gesture and gesture 2, we should expand the system so that it can detect more complex gestures using additional sensors if needed by updating the Arduino and Python codes.
2. **Testing and calibration:** In the future, we should ensure that the sensor ais calibrated properly in the experiment so that the sensor data is accurate. For part 4A, testing and calibration are also important to maintain and improve the reliability of the system and its results.
3. **Servo motor troubleshooting:** In part 4B, the servo motor is not moving as we expected. It is critical to address the servo motor issue in the experiment so that it will not happen in the future. For example, we need to ensure that the power supply is prepared if the servo needs it, the connections are correct and secure, and the code to control the movement of the servo motor is written properly.
4. **User feedback:** In part 4B we can improve the user experience by adding audio feedback to the user. Instead of only using LEDs as indicators, we can use buzzers to indicate feedback to the user whether their card UID is granted or denied.
5. **Power supply for servo motor:** We should consider using a separate and steady power supply for the servo motor. Servo motors can consume a substantial amount of

electricity, therefore a dedicated power supply helps prevent interference with other components and assure reliable performance.

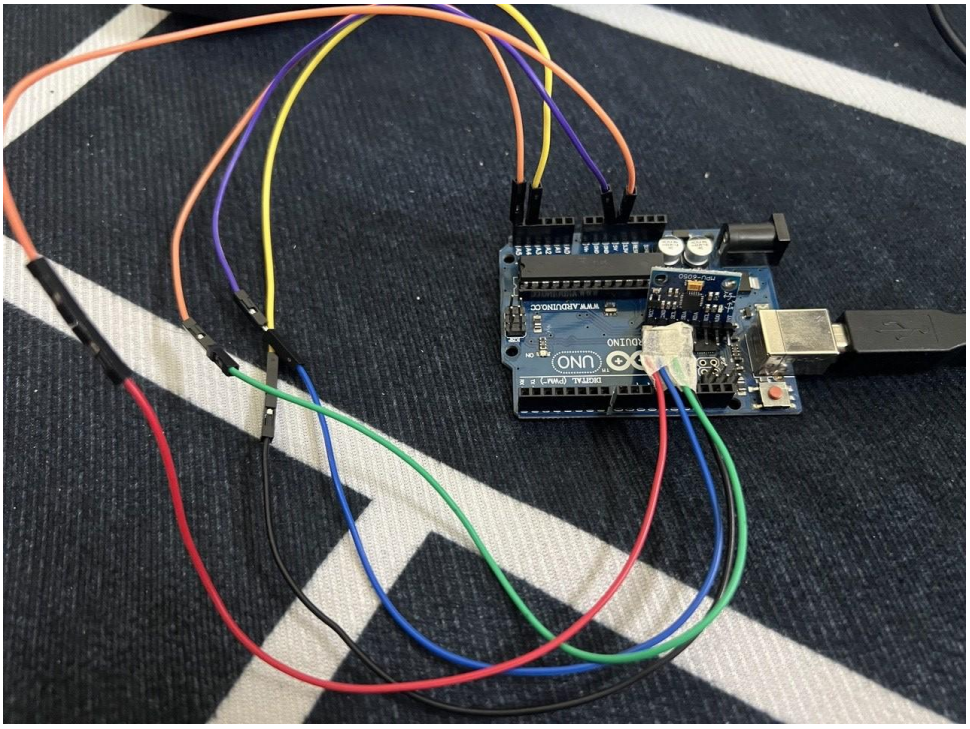
By implementing these recommendations, we can improve the system's reliability and efficiency. Hence, it will reduce the errors that might occur in the future.

References

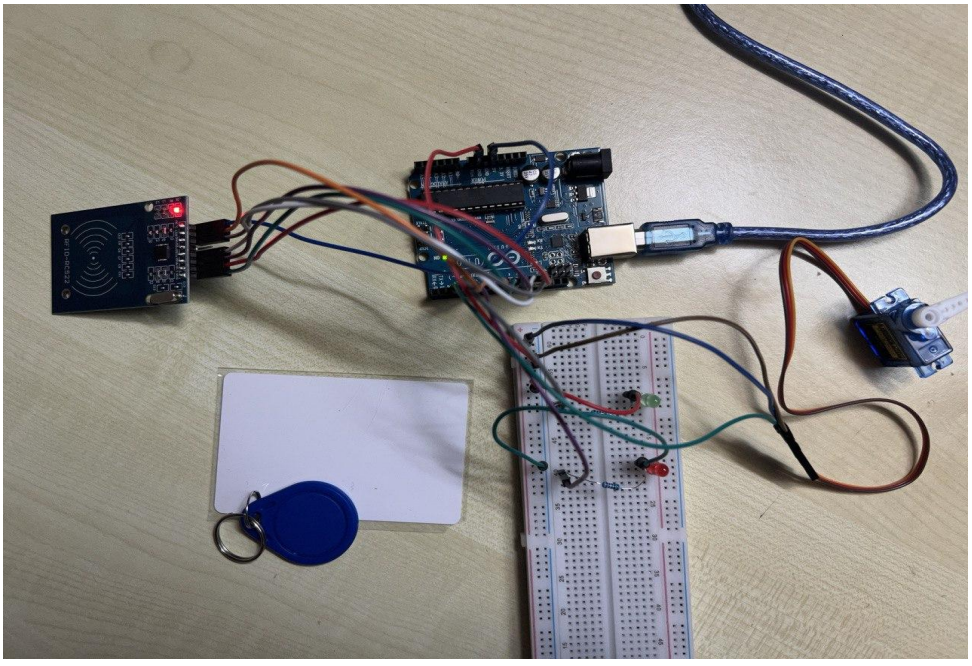
Kumar, S., Das, M. R., Kushalkar, R., Venkat, N., Gourshete, C., & Moudgalya, K. M. (2021, June). Microcontroller Programming with Arduino and Python. https://www.researchgate.net/profile/Sudhakar-Kumar-2/publication/353849431_Microcontroller_programming_with_Arduino_and_Python/links/61177e891ca20f6f861ecc9c/Microcontroller-programming-with-Arduino-and-Python.pdf

Zulkifli. (2014). *Mechatronics Interfacing Lab Manual*, (Rev. ed.). Unpublished Class Materials

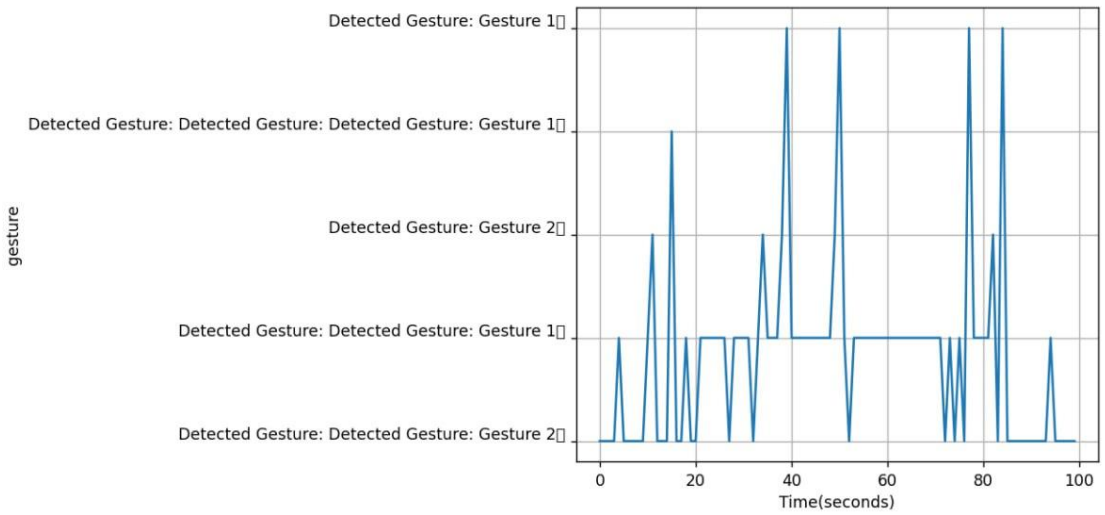
Appendices



Final version for circuit connection in experiment 4A



Final version for circuit connection in experiment 4B



Acknowledgments

We would like to express our sincere gratitude to the individuals who provided invaluable assistance, guidance, and support during this experiment. First and foremost, we extend our appreciation to Dr. Wahju Sediono for their comprehensive instruction and mentorship throughout the experiment. Their insights, feedback, and enthusiasm played an important role in our understanding of Arduino programming.

Our fellow group members also deserve special acknowledgment for their collaboration and support. Our discussions, knowledge-sharing, and problem-solving sessions greatly enriched our understanding of this experiment's concepts and enhanced the overall learning experience. The collective contributions of our group members have not only enriched our learning experience but have also significantly contributed to the successful completion of this project.

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 acknowledgment, and that the original work contained herein has 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 **read** and **understand** the content of the report and no further improvement on the report is needed from any of the individual's contributions 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	sofea	
Name	AISYAH SOFEA OTHMAN	**Read** ✅
Matric Number	2115386	**Understand** ✅
Contribution	- Abstract - Methodology	**Agree** ✅

Signature	aliaa	
Name	SITI ALIAA BINTI IBRAHIM	**Read** ✅
Matric Number	2112618	**Understand** ✅
Contribution	- Discussion - Recommendation	**Agree** ✅

Signature	adli	

Name	ADLI FAHMI	**Read** ✅
Matric Number	2113095	**Understand** ✅
Contribution	- Experimental setup - Material & Equipment - Data Analysis	**Agree** ✅

Signature	*ethar*	
Name	ETHAR OSMAN	**Read** ✅
Matric Number	2111282	**Understand** ✅
Contribution	- Introduction - Data collection	**Agree** ✅

Signature	*ainaa*	
Name	NURAIN AINAA AQILAH BINTI ROSLI	**Read** ✅
Matric Number	2114560	**Understand** ✅
Contribution	- Result - Conclusion	**Agree** ✅