KULLIYYAH OF ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

Lab Report 4A:

Serial and USB interfacing with microcontroller and computer-based system (2): Sensors and actuators

MCTA 3203

Semester 2, 2023/2024

Lecturer:

DR. WAHJU SEDIONO

NAME	MATRIC NO
Muhammad Haziq Iskandar bin Hassan Nordin	2119327
Muhammad Aliff Iqmal bin Zainun @Zainuddin	2114805
Muhammad Nazim bin Akhmar	2114551
Muhammad Arfan Bin Mohd Zulkifli	2112945

Abstact

The primary objective of this project is to establish a serial communication interface between an Arduino microcontroller and the Python programming language. This interface aims to facilitate the transfer of data via a USB connection from a potentiometer connected to the Arduino to a computer. Such a setup simplifies the process of configuring serial communication and enables the utilization of real-time Arduino data within Python programs. The Arduino will be programmed to consistently transmit potentiometer data over a serial port to initiate serial communication. Conversely, a Python script will leverage the PySerial library to establish a serial connection and retrieve the transmitted data from the Arduino. Through thorough research, notable insights have been uncovered, elucidating the successful development of a serial communication system. The project effectively demonstrated the feasibility of establishing a reliable connection between Arduino and Python, thereby enabling the utilization of acquired data for various purposes, notably including data visualization through plotting. This achievement underscores the significance of the developed serial communication interface in facilitating seamless interaction between Arduino microcontrollers and Python programming, thereby expanding the scope of potential applications for such integrated systems.

In conclusion, this experiment illustrates the successful establishment of a serial communication interface between an Arduino microcontroller and the Python programming language. This interface facilitates the exchange of data, particularly focusing on values derived from potentiometers. The demonstrated configuration of both ends underscores the effectiveness of this communication architecture in integrating real-time data into Python programs. The insights gained from this project lay a foundational understanding of serial communication between microcontrollers and computers, thereby paving the way for diverse projects and applications in future pursuits.

Table of content

Introduction

The objective of this project is to establish a serial communication link between an Arduino microcontroller and Python, enabling the transmission of data via USB from the Arduino's potentiometer to a computer. This endeavor aims to simplify the comprehension of serial port connections and facilitate the utilization of real-time Arduino data within Python programs.

The Arduino microcontroller serves the purpose of interfacing with various devices and sensors, as well as executing hardware functions. Utilizing an Arduino facilitates the connection to peripherals and the transmission of data to the computer, as certain devices may not be directly accessible by the computer itself. Moreover, Arduino boards offer a cost-effective solution compared to conventional computers, mitigating potential losses in the event of hardware failures.

Python, renowned for its versatility, is a widely used programming language, particularly in the realm of data science. The utilization of the pyserial tool enables Python scripts to communicate with and manipulate data on serial ports. Consequently, Python scripts play a pivotal role in establishing communication between a computer and an Arduino. Additionally, Python's adaptability in data manipulation empowers users to perform intricate computations, leveraging the computational resources available.

Materials and Equipment

1) Arduino board
2) Jumper wire
3) LED
4) Resistor
5) Breadboard
6) MPU 6050 sensor

Experimental Setup:

1) The MPU6050 sensor was connected to the Arduino board.
2) I2C communication was employed, with the SDA and SCL pins of the MPU6050 connected to the Arduino pins A4 and A5, respectively.
3) The power supply and ground of the MPU6050 were linked to the Arduino's 5V and GND pins, correspondingly.
4) For data exchange, the Arduino board was connected to the PC via USB.

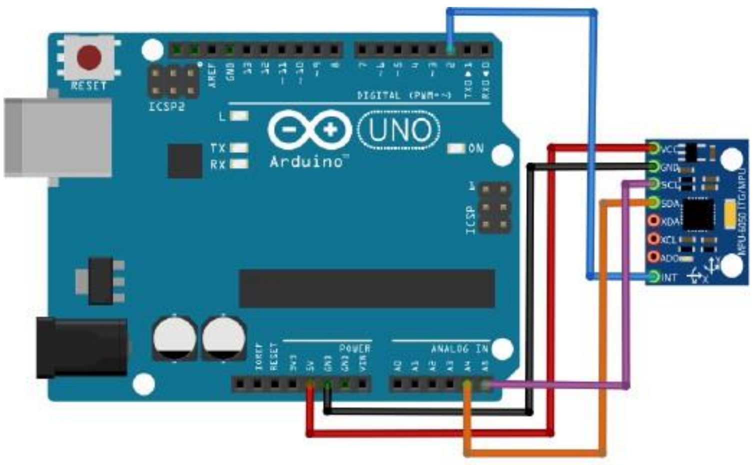


Fig. 1: Arduino-MPU6050 Connections

Software Setup:

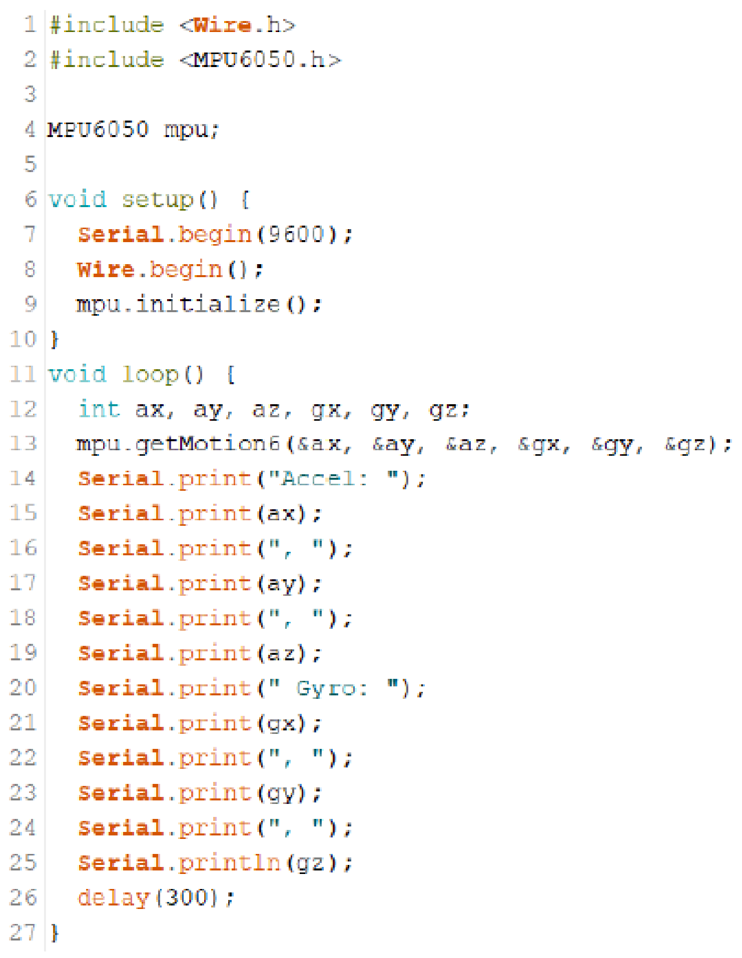
1) Code was authored and uploaded to the Arduino utilizing the Arduino IDE.
2) Arduino code was developed to facilitate communication with the MPU6050 sensor, retrieve data, and format it for transmission.
3) The PySerial package for Python was employed to establish a serial connection between the Arduino and the PC.
4) Data reception and presentation on the PC console were achieved through a Python script.
5) Within the Python script, data processing and parsing were conducted as needed to interpret the format of the data transmitted by the Arduino.

Methodology

Procedure:

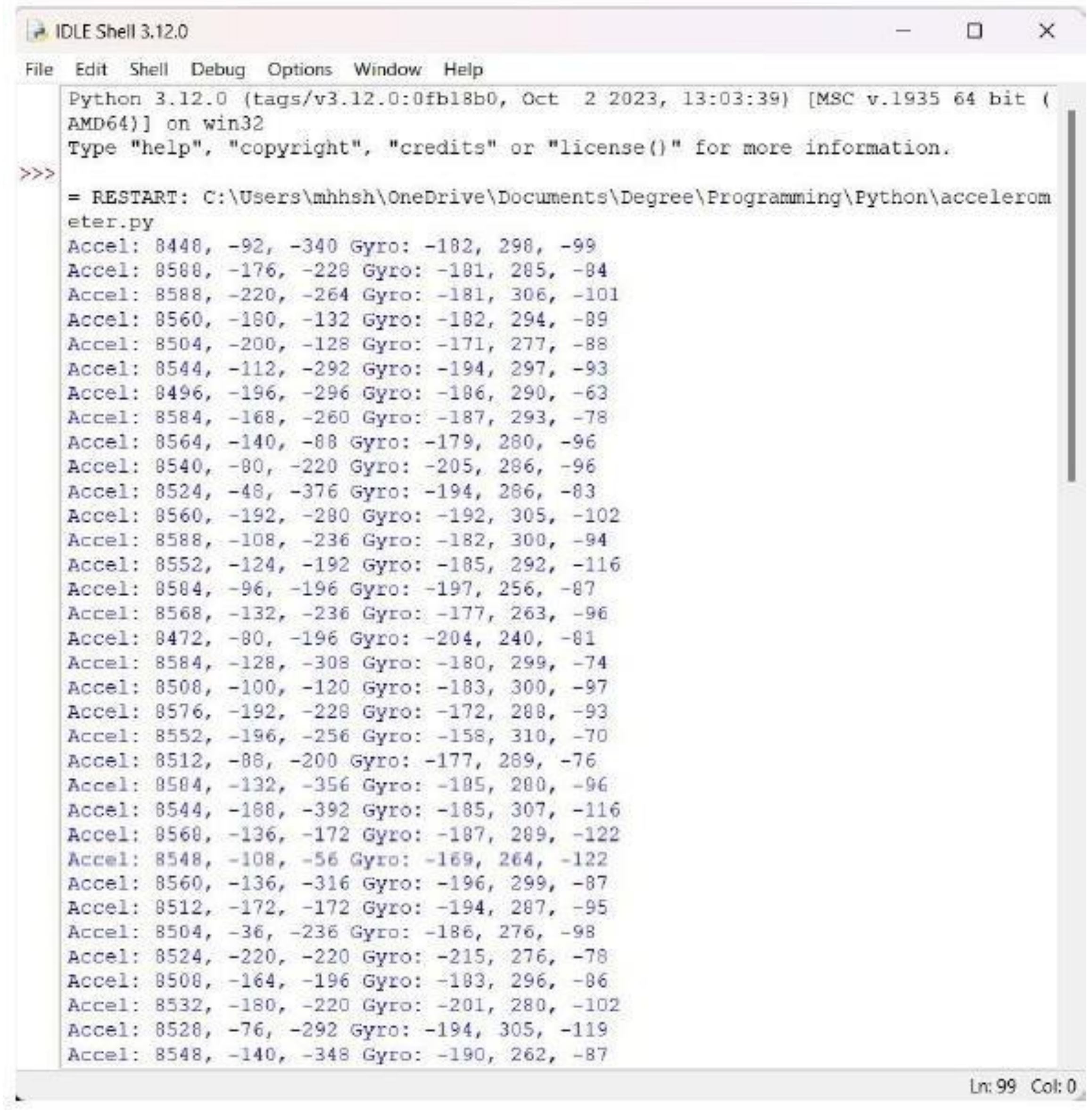
1) Following the successful completion of hardware and software setup, the Arduino code was thoroughly inspected and uploaded to the Arduino microcontroller utilizing the Arduino IDE.
2) Subsequently, to observe the outcomes of the experiment, the Python code was executed after being saved as a computer file.
3) Group members observed and recorded the values retrieved from the MPU6050, which were displayed on the Python terminal.

Coding:



```
#include <Wire.h>
#include <MPU6050.h>

MPU6050 mpu;

void setup() {
  Serial.begin(9600);
  Wire.begin();
  mpu.initialize();
}
void loop() {
  int ax, ay, az, gx, gy, gz;
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
  Serial.print("Accel: ");
  Serial.print(ax);
  Serial.print(", ");
  Serial.print(ay);
  Serial.print(", ");
  Serial.print(az);
  Serial.print(" Gyro: ");
  Serial.print(gx);
  Serial.print(", ");
  Serial.print(gy);
  Serial.print(", ");
  Serial.println(gz);
  delay(300);
}
```

Results



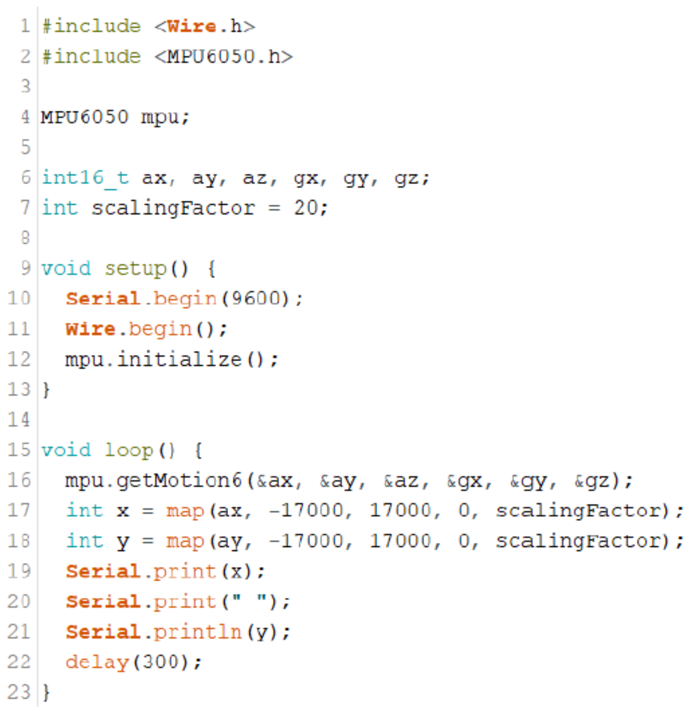
```
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\mhhsh\OneDrive\Documents\Degree\Programming\Python\accelerom
eter.py
Accel: 8448, -92, -340 Gyro: -182, 298, -99
Accel: 8588, -176, -228 Gyro: -181, 285, -84
Accel: 8588, -220, -264 Gyro: -181, 306, -101
Accel: 8560, -180, -132 Gyro: -182, 294, -89
Accel: 8504, -200, -128 Gyro: -171, 277, -88
Accel: 8544, -112, -292 Gyro: -194, 297, -93
Accel: 8496, -196, -296 Gyro: -186, 290, -63
Accel: 8584, -168, -260 Gyro: -187, 293, -78
Accel: 8564, -140, -88 Gyro: -179, 280, -96
Accel: 8540, -80, -220 Gyro: -205, 286, -96
Accel: 8524, -48, -376 Gyro: -194, 286, -83
Accel: 8560, -192, -280 Gyro: -192, 305, -102
Accel: 8588, -108, -236 Gyro: -182, 300, -94
Accel: 8552, -124, -192 Gyro: -185, 292, -116
Accel: 8584, -96, -196 Gyro: -197, 256, -87
Accel: 8568, -132, -236 Gyro: -177, 263, -96
Accel: 8472, -80, -196 Gyro: -204, 240, -81
Accel: 8584, -128, -308 Gyro: -180, 299, -74
Accel: 8508, -100, -120 Gyro: -183, 300, -97
Accel: 8576, -192, -228 Gyro: -172, 288, -93
Accel: 8552, -196, -256 Gyro: -158, 310, -70
Accel: 8512, -88, -200 Gyro: -177, 289, -76
Accel: 8584, -132, -356 Gyro: -185, 280, -96
Accel: 8544, -188, -392 Gyro: -185, 307, -116
Accel: 8568, -136, -172 Gyro: -187, 289, -122
Accel: 8548, -108, -56 Gyro: -169, 264, -122
Accel: 8560, -136, -316 Gyro: -196, 299, -87
Accel: 8512, -172, -172 Gyro: -194, 287, -95
Accel: 8504, -36, -236 Gyro: -186, 276, -98
Accel: 8524, -220, -220 Gyro: -215, 276, -78
Accel: 8508, -164, -196 Gyro: -183, 296, -86
Accel: 8532, -180, -220 Gyro: -201, 280, -102
Accel: 8528, -76, -292 Gyro: -194, 305, -119
Accel: 8548, -140, -348 Gyro: -190, 262, -87
```

As the MPU6050 undergoes movement, the Python script displays the readings of the accelerometer and gyroscope values, thereby indicating an established serial communication between the Arduino and Python. These dynamic values reflect variations in acceleration across the three axes (X, Y, and Z) when the MPU6050 is in motion. For instance, tilting the sensor to the right results in an elevation of the Y-axis accelerometer reading, while tilting it forward increases the Z-axis accelerometer reading. Furthermore, movement of the MPU6050 induces changes in the gyroscope values, signaling rotational speed alterations around the three axes (X, Y, and Z). For example, a swift rotation around the Z-axis leads to an augmentation in the gyroscope value for the Z-axis, denoting a shift in angular velocity.

Discussions

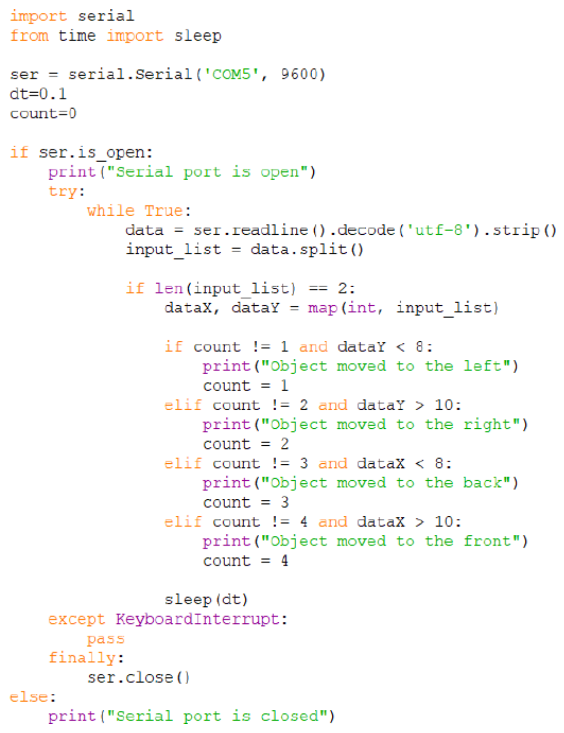
As instructed in the manual, we were tasked with developing a simple hand gesture recognition system by capturing accelerometer and gyroscope data while executing predefined hand movements. We utilized an algorithm to discern and classify these gestures based on the acquired sensor data. Furthermore, we visualized the trajectories of hand movements within an x-y coordinate system. Below are the Arduino and Python codes provided for reference:

Coding for Arduino:



```
#include <Wire.h>
#include <MPU6050.h>

MPU6050 mpu;

int16_t ax, ay, az, gx, gy, gz;
int scalingFactor = 20;

void setup() {
  Serial.begin(9600);
  Wire.begin();
  mpu.initialize();
}

void loop() {
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
  int x = map(ax, -17000, 17000, 0, scalingFactor);
  int y = map(ay, -17000, 17000, 0, scalingFactor);
  Serial.print(x);
  Serial.print(" ");
  Serial.println(y);
  delay(300);
}
```

Coding for python:



```
import serial
from time import sleep

ser = serial.Serial('COM5', 9600)
dt=0.1
count=0

if ser.is_open:
    print("Serial port is open")
    try:
        while True:
            data = ser.readline().decode('utf-8').strip()
            input_list = data.split()

            if len(input_list) == 2:
                dataX, dataY = map(int, input_list)

                if count != 1 and dataY < 8:
                    print("Object moved to the left")
                    count = 1
                elif count != 2 and dataY > 10:
                    print("Object moved to the right")
                    count = 2
                elif count != 3 and dataX < 8:
                    print("Object moved to the back")
                    count = 3
                elif count != 4 and dataX > 10:
                    print("Object moved to the front")
                    count = 4

                sleep(dt)
    except KeyboardInterrupt:
        pass
    finally:
        ser.close()
else:
    print("Serial port is closed")
```

Link for recorded video
https://drive.google.com/file/d/1onyiGxjEg8nuz55KMnZSxZJzbcZD2hEw/view?usp=drive_link
https://drive.google.com/file/d/1uMSMlT9PyCCD5Griw70bJWVxvFvJ1W_r/view?usp=drive_link

Conclusion

A productive and instructive laboratory experiment has been conducted through the utilization of the MPU6050 accelerometer and gyroscope in tandem with an algorithm designed for gesture identification, culminating in the development of a hand gesture recognition system. The integration of sensor data acquisition and signal processing capabilities to discern and categorize predetermined hand gestures exemplifies the diverse range of potential applications inherent in this technology. Despite persisting challenges associated with noise mitigation and the detection of intricate gestures, the experiment lays the groundwork for future research initiatives, elucidating the considerable potential of this technology within domains such as wearables, assistive technologies, and human-computer interaction.

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 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 have read and understand the content of the total report and no further improvement on the reports is needed from any of the individual contributor 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: *haziq*	Read
Name: Muhammad Haziq Iskandar bin Hassan Nordin	Understand
Matric Number: 2119327	Agree
Signature: *aliff*	Read
Name: Muhammad Aliff Iqmal bin Zainun @Zainuddin	Understand
Matric Number:2114805	Agree
Signature: *nazim*	Read
Name: Muhammad Nazim bin Akhmar	Understand
Matric Number: 2114551	Agree
Signature: *arfan*	Read
Name: Muhammad Arfan Bin Mohd Zulkifli	Understand
Matric Number: 2112945	Agree