M5StickC IMU Sensors Lab Report

# 1. Introduction ­­

In this lab, we’ll learn how to use the M5StickC sensors to measure acceleration and use this data to simulate rolling a dice.

Using the equation for the magnitude of the acceleration vector:

We’ll display the current accelerations of the X, Y, and Z axes on the screen. Below that, we’ll also show the maximum recorded values for X, Y, Z, and the overall acceleration magnitude.  
Next, we’ll observe the magnitude of acceleration when the device is rolled with a specific flick motion (like rolling a die). Our code will be set up to generate a random number between 1 and 6, simulating a dice roll. Each time a new number is generated, it will be sent to ThingSpeak. This setup allows us to visualize each roll as a real-time data point on ThingSpeak, creating a clear record of the results. By logging these random numbers, we can track the frequency of each outcome and observe patterns or trends in the simulated dice rolls on the ThingSpeak dashboard.

# 2. Bill of Materials ­­

1x computer with Arduino IDE 2.3.2

1x M5stickC

1x USBc to USBc data cord

API Access to Thingspeak

# 3. Diagram of M5StickC Axis’s­­

A drawing of a box

Description automatically generated

# 4. Demonstration

A hand holding a small device

Description automatically generated

# 5. Source Code

Variables:

A black background with a few small lights

Description automatically generated with medium confidence

Setup:

A computer screen shot of a computer

Description automatically generated

Loop:

A screenshot of a computer program

Description automatically generated

Secrets.h  
A screenshot of a computer

Description automatically generated

# 6.Code Explanation

Variable Section:  
In this section, we define the variables needed for WiFi and acceleration data.

* ssid and pass store the network's SSID and password, which allow the device to connect to WiFi.
* client is created as a WiFiClient object to manage the ThingSpeak connection.
* myChannelNumber and myWriteAPIKey are used to identify and authenticate the ThingSpeak channel where data will be sent.
* number is an integer that will store the randomly generated number (between 1 and 6), simulating a dice roll.

We also define some additional variables to control the timing and LED state:

* LED\_BUILTIN is set to 10, specifying the pin for the built-in LED on the M5StickC.
* accX, accY, and accZ will hold the instantaneous acceleration values along the X, Y, and Z axes.
* currentTime and previousTime store timestamps to manage the timing of data updates.

**Setup Function:**

In the setup function, we initialize various components and configure settings.

* M5.begin() initializes the M5StickC, setting up the display and sensors.
* M5.IMU.Init() initializes the IMU (Inertial Measurement Unit), which will provide acceleration data.
* Serial.begin(115200) opens the serial port for debugging purposes.
* A loop waits for the serial port to connect, needed on some boards for USB serial communication.
* pinMode(LED\_BUILTIN, OUTPUT); sets the LED pin as an output.
* WiFi.mode(WIFI\_STA); puts the WiFi in station mode, preparing it to connect to a network.
* ThingSpeak.begin(client); initializes the ThingSpeak library with the WiFi client, enabling data transmission.

**Loop Function:**

The loop function handles the main logic for generating and sending data to ThingSpeak.

1. WiFi Connection Check:
   1. The code checks if the device is connected to WiFi. If not, it attempts to reconnect by calling WiFi.begin(ssid, pass);, retrying every 5 seconds until successful.
2. Timing and LED Control:
   1. currentTime is updated with millis() to manage time since the program started.
   2. The built-in LED blinks by toggling its state with digitalWrite(LED\_BUILTIN, LOW); and digitalWrite(LED\_BUILTIN, HIGH);, each followed by a 100 ms delay.
3. Acceleration Data and Random Number Generation:
   1. The program reads acceleration values using M5.IMU.getAccelData(&accX, &accY, &accZ);.
   2. If acceleration on the X or Y axis exceeds 1.2, a random number between 1 and 6 is generated with number = random(0, 6);, simulating a dice roll. The code then exits the loop.
4. Sending Data to ThingSpeak:
   1. The random number (stored in number) is sent to ThingSpeak on Field 1 using ThingSpeak.writeField(myChannelNumber, 1, number, myWriteAPIKey);.
   2. If the update is successful (returns HTTP status code 200), a success message is printed to the Serial Monitor; otherwise, an error message with the HTTP error code is displayed.
5. Resetting the Dice Roll:
   1. Finally, the number is reset to 0, ready for the next dice roll simulation.

This setup allows us to visualize each dice roll on the ThingSpeak platform in real-time, with each roll's result recorded in Field 1 of the designated ThingSpeak channel.

# 7. Calculations