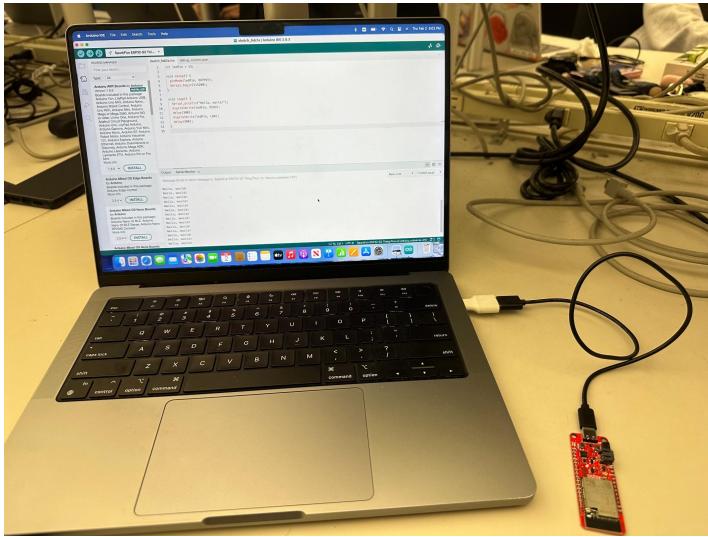


Zeid Solh  
205323715  
EC ENGR 180DA  
Lab 4

Task 1:  
MCU and IMU working pictures.



### Blink

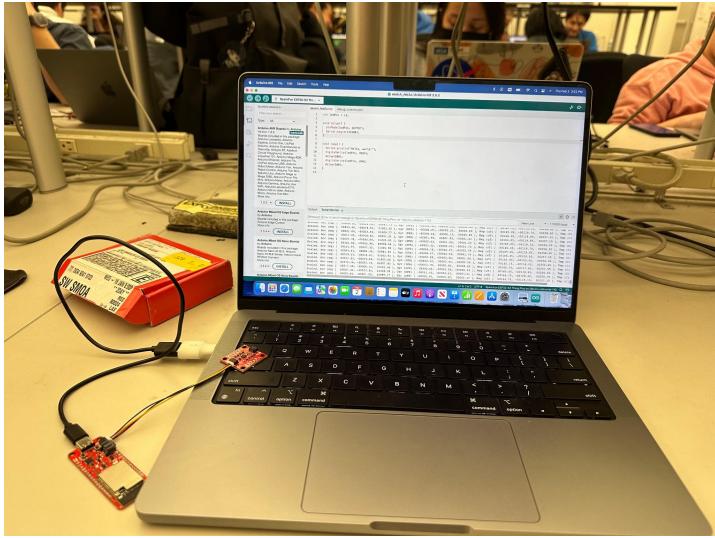
To verify the toolchain and board are properly set up on their computer, users can upload the simplest of sketches -- Blink! The LED attached to **GPIO 13** is perfect for this test. Plus, with the ESP32 attached to your computer, this is a good time to test out serial communication. Copy and paste the example sketch below into a fresh Arduino sketch:

```
int ledPin = 13;  
  
void setup()  
{  
    pinMode(ledPin, OUTPUT);  
    Serial.begin(115200);  
}  
  
void loop()  
{  
    Serial.println("Hello, world!");  
    digitalWrite(ledPin, HIGH);  
    delay(500);  
    digitalWrite(ledPin, LOW);  
    delay(500);  
}
```

[COPY CODE](#)

With everything setup correctly, upload the code! Once the code finishes transferring, open the **serial monitor** and set the baud rate to **115200**. Users should see `Hello, world!`'s begin to fly by.

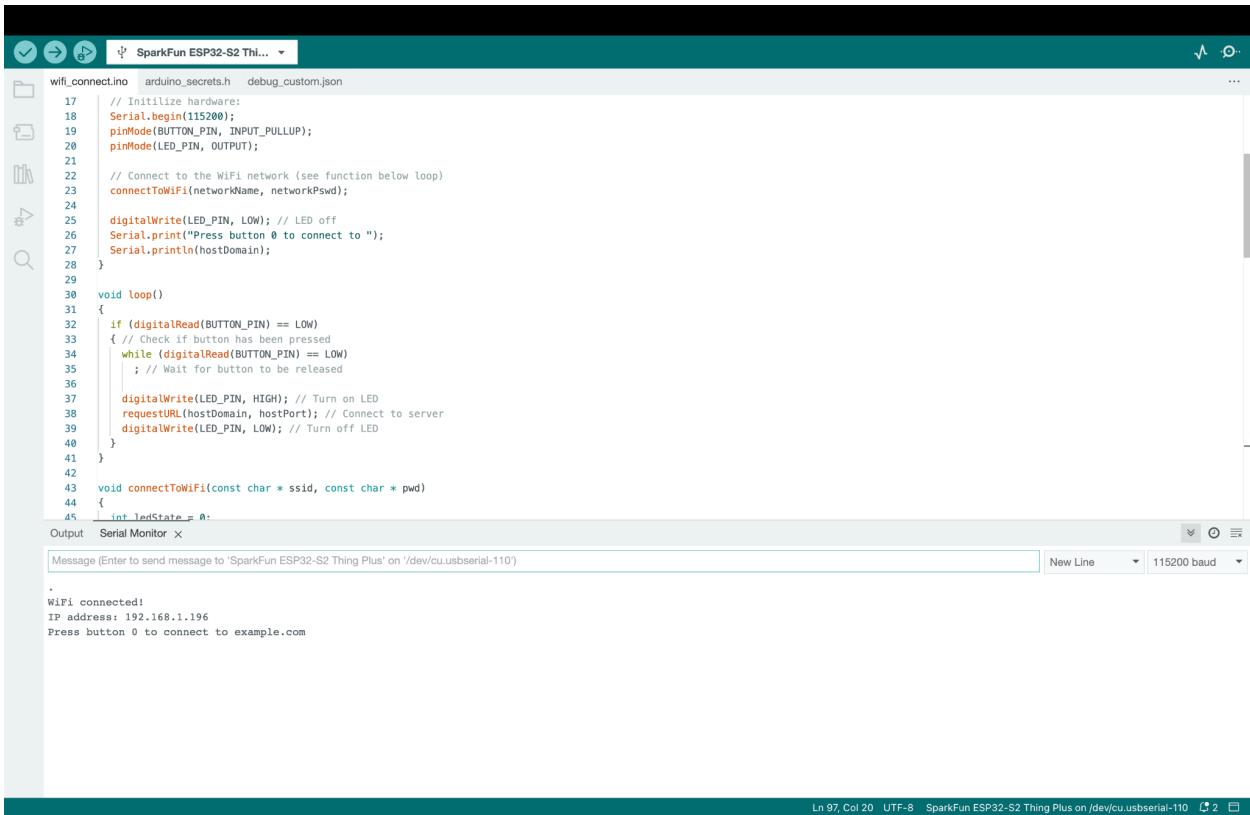
I used the above blink code for MCU.



For the IMU, I used Example/Sparkfun 9DoF.../Arduino/Example1\_Basics.

## Task 2:

Wifi Connected successfully.



```
wifi_connect.ino  arduino_secrets.h  debug_custom.json
17 // Initialize hardware:
18 Serial.begin(115200);
19 pinMode(BUTTON_PIN, INPUT_PULLUP);
20 pinMode(LED_PIN, OUTPUT);
21
22 // Connect to the WiFi network (see function below loop)
23 connectToWiFi(networkName, networkPswd);
24
25 digitalWrite(LED_PIN, LOW); // LED off
26 Serial.print("Press button 0 to connect to ");
27 Serial.println(hostDomain);
28 }
29
30 void loop()
31 {
32 if (digitalRead(BUTTON_PIN) == LOW)
33 { // Check if button has been pressed
34 while (digitalRead(BUTTON_PIN) == LOW)
35 ; // Wait for button to be released
36
37 digitalWrite(LED_PIN, HIGH); // Turn on LED
38 requestURL(hostDomain, hostPort); // Connect to server
39 digitalWrite(LED_PIN, LOW); // Turn off LED
40 }
41 }
42
43 void connectToWiFi(const char * ssid, const char * pwd)
44 {
45 int ledState = 0;
}

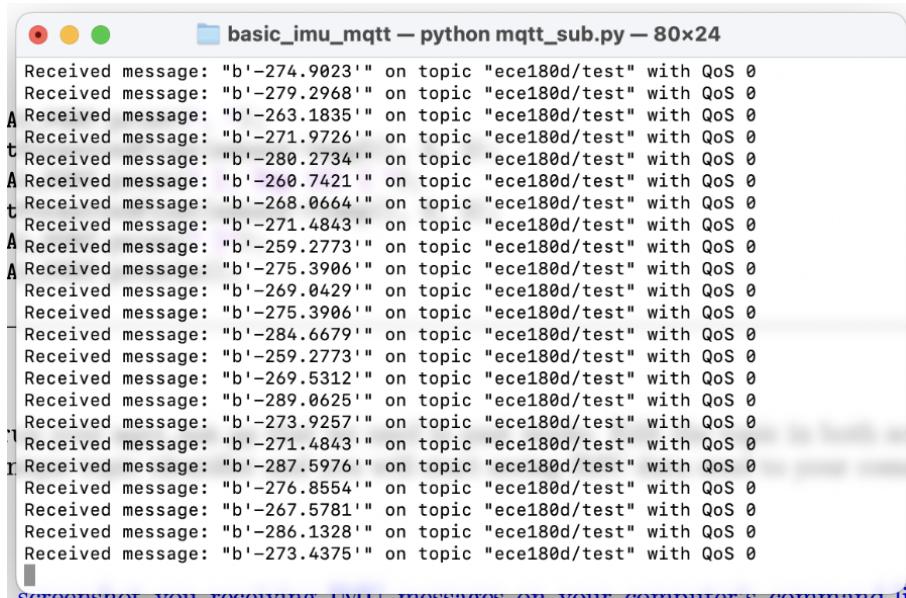
Message (Enter to send message to 'SparkFun ESP32-S2 Thing Plus' on '/dev/cu.usbserial-110')
.
WiFi connected!
IP address: 192.168.1.196
Press button 0 to connect to example.com

Ln 97, Col 20  UTF-8  SparkFun ESP32-S2 Thing Plus on /dev/cu.usbserial-110  ⌂ 2  ⌂
```

Used wifi\_connect.ino to do this task.

### Task 3:

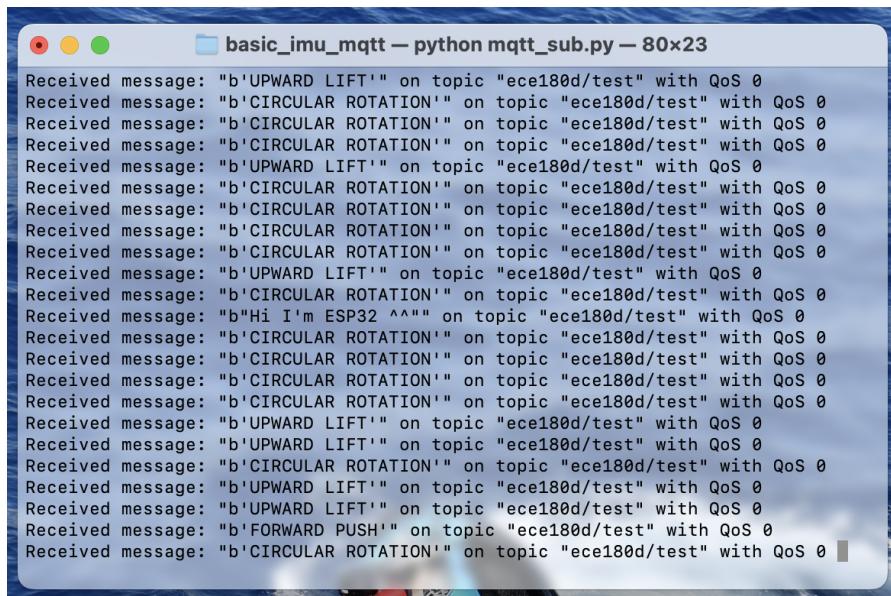
Receiving IMU messages on my computer's command line. Using basic\_imu\_mqtt.ino and mqtt\_sub.py. The messages represent the X acceleration.



A screenshot of a terminal window titled "basic\_imu\_mqtt — python mqtt\_sub.py — 80x24". The window displays a continuous stream of received MQTT messages. Each message is a byte string starting with "b'" followed by a sequence of numbers and characters, such as "b'-274.9023'", "b'-279.2968'", etc. The messages are received at regular intervals, indicating a fast data rate.

```
Received message: "b'-274.9023'" on topic "ece180d/test" with QoS 0
Received message: "b'-279.2968'" on topic "ece180d/test" with QoS 0
Received message: "b'-263.1835'" on topic "ece180d/test" with QoS 0
Received message: "b'-271.9726'" on topic "ece180d/test" with QoS 0
Received message: "b'-271.2734'" on topic "ece180d/test" with QoS 0
Received message: "b'-260.7421'" on topic "ece180d/test" with QoS 0
Received message: "b'-268.0664'" on topic "ece180d/test" with QoS 0
Received message: "b'-271.4843'" on topic "ece180d/test" with QoS 0
Received message: "b'-259.2773'" on topic "ece180d/test" with QoS 0
Received message: "b'-275.3906'" on topic "ece180d/test" with QoS 0
Received message: "b'-269.0429'" on topic "ece180d/test" with QoS 0
Received message: "b'-275.3906'" on topic "ece180d/test" with QoS 0
Received message: "b'-284.6679'" on topic "ece180d/test" with QoS 0
Received message: "b'-259.2773'" on topic "ece180d/test" with QoS 0
Received message: "b'-269.5312'" on topic "ece180d/test" with QoS 0
Received message: "b'-289.0625'" on topic "ece180d/test" with QoS 0
Received message: "b'-273.9257'" on topic "ece180d/test" with QoS 0
Received message: "b'-271.4843'" on topic "ece180d/test" with QoS 0
Received message: "b'-287.5976'" on topic "ece180d/test" with QoS 0
Received message: "b'-276.8554'" on topic "ece180d/test" with QoS 0
Received message: "b'-267.5781'" on topic "ece180d/test" with QoS 0
Received message: "b'-286.1328'" on topic "ece180d/test" with QoS 0
Received message: "b'-273.4375'" on topic "ece180d/test" with QoS 0
```

There is a small amount of lag present given that we try to send data at a really fast rate (30 ms). If we reduce for instance the rate at which we send messages, for instance from 30 ms to 100 ms or 500 ms, we get much better results and less lag.



A screenshot of a terminal window titled "basic\_imu\_mqtt — python mqtt\_sub.py — 80x24". The window displays a stream of received MQTT messages. These messages are now more meaningful, representing detected actions like "UPWARD LIFT", "CIRCULAR ROTATION", and "FORWARD PUSH". The messages are still received at a high frequency, but the content is more descriptive than the raw X acceleration values.

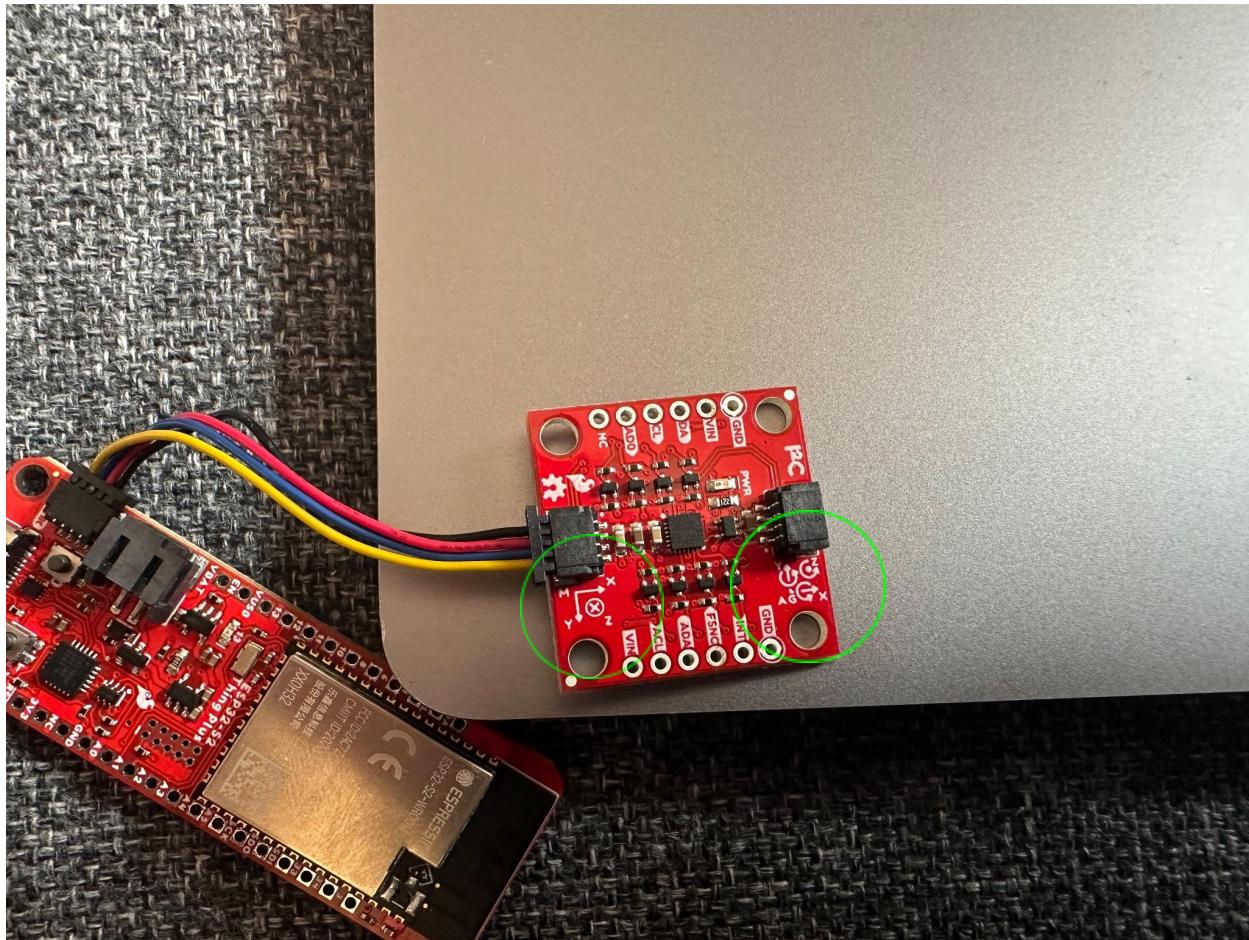
```
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'Hi I'm ESP32 ^^^" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'UPWARD LIFT'" on topic "ece180d/test" with QoS 0
Received message: "b'FORWARD PUSH'" on topic "ece180d/test" with QoS 0
Received message: "b'CIRCULAR ROTATION'" on topic "ece180d/test" with QoS 0
```

By doing processing on the IMU itself so that only a single message is sent when something is recognized, we get better results as we are no longer sending messages at 30 ms intervals whether we detected something or not. Above is an example of me doing so (combining my classifier.ino and basic\_imu\_mqtt.ino).

I think the best way to get around lag time, if we have to have lag, is to do some processing on the board itself and as soon as we have interesting results we send it back. We can also reduce the rate at which we are sending messages. Alternatively, we can use a cable, if wifi connectivity is not a must!

Task 4:

4.1



In the picture above, we can see the directions of x, y, and z and can confirm the roll, pitch, yaw values. I circled them in the picture.

Yes, when idle, we can see the gravity acceleration in one or more of the components (x, y, z) which takes values around  $\sim 980\text{-}1000$  (which seems to be in  $\text{cm/s}^2$ ).

Using the test file Example/Sparkfun 9DoF.../Arduino/Example1\_Basics:

```

Example1_Basics.ino debug.custom.json
1 //*****
2 * Example1_Basics.ino
3 * ICM 20948 Arduino Library Demo
4 * Use the default configuration to stream 9-axis IMU data
5 * Owen Lyke @ SparkFun Electronics
6 * Original Creation Date: April 17 2019
7 *
8 * Please see License.md for the license information.
9 *
10 * Distributed as-is; no warranty is given.
11 *****/
12 #include "ICM_20948.h" // Click here to get the library: http://librarymanager/All#SparkFun_ICM_20948_IMU
13
14 //##define USE_SPI      // Uncomment this to use SPI
15
16 #define SERIAL_PORT Serial
17
18 #define SPI_PORT SPI // Your desired SPI port.      Used only when "USE_SPI" is defined
19 #define CS_PIN 2      // Which pin you connect CS to. Used only when "USE_SPI" is defined
20
21 #define WIRE_PORT Wire // Your desired Wire port.      Used when "USE_SPI" is not defined
22 // The value of the last bit of the I2C address.
23 // On the SparkFun 9DoF IMU breakout the default is 1, and when the ADR jumper is closed the value becomes 0
24 #define ADR_VAL 1
25
26 #ifdef USE_SPI
27 ICM_20948_SPI myIMU; // If using SPI create an ICM_20948_SPI object
28 #else
29 ICM_20948_I2C myIMU; // Otherwise create an ICM_20948_I2C object
30
31 Serial Monitor x
32
33 Message (Enter to send message to 'SparkFun ESP32-S2 Thing Plus' on '/dev/cu.usbserial-210')
34
35 Scaled. Acc (mg) [ 00082.52, 00104.49, 01028.81 ], Gyr (DPS) [ 00001.73, -00002.09, -00000.60 ], Mag (uT) [ -00048.45, 00052.35, 00082.20 ], Tmp (C) [ 00029.37 ]
36 Scaled. Acc (mg) [ 00088.38, 0013.07, 01028.81 ], Gyr (DPS) [ 00003.62, -00004.56, -00001.63 ], Mag (uT) [ -00048.15, 00053.10, 00081.00 ], Tmp (C) [ 00029.28 ]
37 Scaled. Acc (mg) [ 00077.15, 00141.60, 00996.58 ], Gyr (DPS) [ 00003.85, -00001.11, -00002.25 ], Mag (uT) [ -00046.95, 00053.10, 00081.45 ], Tmp (C) [ 00029.56 ]
38 Scaled. Acc (mg) [ 00157.23, 001025.39 ], Gyr (DPS) [ 00002.24, -00002.85, 00001.15 ], Mag (uT) [ -00048.30, 00052.50, 00082.05 ], Tmp (C) [ 00029.37 ]
39 Scaled. Acc (mg) [ 00092.29, 00148.93, 01036.13 ], Gyr (DPS) [ 00004.56, -00000.98, -00008.04 ], Mag (uT) [ -00046.95, 00052.35, 00081.75 ], Tmp (C) [ 00029.37 ]
40 Scaled. Acc (mg) [ 00090.82, 00142.09, 01022.95 ], Gyr (DPS) [ 00003.78, 00000.26, 00004.59 ], Mag (uT) [ -00047.55, 00052.20, 00081.60 ], Tmp (C) [ 00029.28 ]
41 Scaled. Acc (mg) [ 00086.43, 00113.77, 01016.60 ], Gyr (DPS) [ 00002.13, 00001.56, 00000.56 ], Mag (uT) [ -00047.25, 00052.95, 00080.85 ], Tmp (C) [ 00029.47 ]
42 Scaled. Acc (mg) [ 00086.43, 00131.39, 01025.39 ], Gyr (DPS) [ 00001.08, 00000.27, -00000.38 ], Mag (uT) [ -00048.30, 00052.20, 00080.85 ], Tmp (C) [ 00029.66 ]
43 Scaled. Acc (mg) [ 00098.63, 00154.30, 01033.69 ], Gyr (DPS) [ -00000.11, 00002.69, -00000.52 ], Mag (uT) [ -00048.15, 00051.15, 00082.20 ], Tmp (C) [ 00029.37 ]
44 Scaled. Acc (mg) [ 00095.70, 00147.95, 01031.25 ], Gyr (DPS) [ 00001.41, -00001.71, -00001.31 ], Mag (uT) [ -00048.30, 00051.75, 00081.30 ], Tmp (C) [ 00029.47 ]
45 Scaled. Acc (mg) [ 00099.61, 00141.60, 01016.60 ], Gyr (DPS) [ -00001.37, 00000.78, 00000.00 ], Mag (uT) [ -00048.75, 00051.00, 00082.35 ], Tmp (C) [ 00029.42 ]
46 Scaled. Acc (mg) [ 00091.80, 00118.16, 01025.39 ], Gyr (DPS) [ 00000.36, 00001.66, -00001.08 ], Mag (uT) [ -00048.75, 00052.20, 00082.05 ], Tmp (C) [ 00029.56 ]
47 Scaled. Acc (mg) [ 00076.17, 00142.09, 01029.79 ], Gyr (DPS) [ 00000.47, 00004.11, -00001.24 ], Mag (uT) [ -00048.45, 00051.90, 00081.75 ], Tmp (C) [ 00029.61 ]
48 Scaled. Acc (mg) [ 00100.10, 00135.25, 01047.36 ], Gyr (DPS) [ -00001.76, -00000.09, 00001.15 ], Mag (uT) [ -00048.00, 00053.25, 00081.60 ], Tmp (C) [ 00029.56 ]

```

We can see gravity here in the z component.

```

Example1_Basics.ino debug.custom.json
1 //*****
2 * Example1_Basics.ino
3 * ICM 20948 Arduino Library Demo
4 * Use the default configuration to stream 9-axis IMU data
5 * Owen Lyke @ SparkFun Electronics
6 * Original Creation Date: April 17 2019
7 *
8 * Please see License.md for the license information.
9 *
10 * Distributed as-is; no warranty is given.
11 *****/
12 #include "ICM_20948.h" // Click here to get the library: http://librarymanager/All#SparkFun_ICM_20948_IMU
13
14 //##define USE_SPI      // Uncomment this to use SPI
15
16 #define SERIAL_PORT Serial
17
18 #define SPI_PORT SPI // Your desired SPI port.      Used only when "USE_SPI" is defined
19 #define CS_PIN 2      // Which pin you connect CS to. Used only when "USE_SPI" is defined
20
21 #define WIRE_PORT Wire // Your desired Wire port.      Used when "USE_SPI" is not defined
22 // The value of the last bit of the I2C address.
23 // On the SparkFun 9DoF IMU breakout the default is 1, and when the ADR jumper is closed the value becomes 0
24 #define ADR_VAL 1
25
26 #ifdef USE_SPI
27 ICM_20948_SPI myIMU; // If using SPI create an ICM_20948_SPI object
28 #else
29 ICM_20948_I2C myIMU; // Otherwise create an ICM_20948_I2C object
30
31 Serial Monitor x
32
33 Message (Enter to send message to 'SparkFun ESP32-S2 Thing Plus' on '/dev/cu.usbserial-210')
34
35 Scaled. Acc (mg) [ -00022.95, 00966.80, -00101.56 ], Gyr (DPS) [ 00003.72, -00002.92, 00004.69 ], Mag (uT) [ -00052.80, 00056.70, 00031.20 ], Tmp (C) [ 00029.90 ]
36 Scaled. Acc (mg) [ -00199.53, 00981.45, -00116.70 ], Gyr (DPS) [ 00004.13, 00001.29, -00002.69 ], Mag (uT) [ -00053.55, 00054.00, 00030.90 ], Tmp (C) [ 00030.19 ]
37 Scaled. Acc (mg) [ -00025.39, 00973.14, -00102.05 ], Gyr (DPS) [ -00000.92, 00002.00, -00001.29 ], Mag (uT) [ -00053.55, 00054.90, 00030.45 ], Tmp (C) [ 00030.43 ]
38 Scaled. Acc (mg) [ -00021.48, 00993.16, -00113.77 ], Gyr (DPS) [ 00000.73, 00002.92, -00001.32 ], Mag (uT) [ -00052.65, 00053.85, 00031.80 ], Tmp (C) [ 00030.19 ]
39 Scaled. Acc (mg) [ -00033.20, 00989.75, -00128.91 ], Gyr (DPS) [ 00001.31, 00000.22, -00001.20 ], Mag (uT) [ -00052.95, 00055.35, 00031.05 ], Tmp (C) [ 00030.19 ]
40 Scaled. Acc (mg) [ -00016.11, 00980.96, -00103.52 ], Gyr (DPS) [ 00000.47, 00000.40, -00001.60 ], Mag (uT) [ -00053.55, 00052.95, 00031.20 ], Tmp (C) [ 00029.99 ]
41 Scaled. Acc (mg) [ -00020.51, 00959.47, -00102.05 ], Gyr (DPS) [ 00002.53, -00000.05, 00001.15 ], Mag (uT) [ -00053.25, 00054.00, 00032.70 ], Tmp (C) [ 00030.14 ]
42 Scaled. Acc (mg) [ -00016.60, 00964.84, -00101.07 ], Gyr (DPS) [ 00001.50, 00003.62, 00002.12 ], Mag (uT) [ -00053.55, 00055.20, 00032.25 ], Tmp (C) [ 00030.19 ]
43 Scaled. Acc (mg) [ -00012.70, 00964.36, -00096.19 ], Gyr (DPS) [ 00001.00, 00000.64, -00000.31 ], Mag (uT) [ -00052.80, 00055.20, 00031.95 ], Tmp (C) [ 00030.19 ]
44 Scaled. Acc (mg) [ -00025.88, 00996.09, -00091.80 ], Gyr (DPS) [ 00003.37, 00000.27, -00002.60 ], Mag (uT) [ -00053.70, 00056.25, 00031.05 ], Tmp (C) [ 00030.28 ]
45 Scaled. Acc (mg) [ -00013.67, 01000.98, -00119.14 ], Gyr (DPS) [ 00001.38, -00001.07, 00003.27 ], Mag (uT) [ -00053.40, 00056.25, 00032.10 ], Tmp (C) [ 00030.14 ]
46 Scaled. Acc (mg) [ -00001.95, 00934.08, -00090.82 ], Gyr (DPS) [ 00002.07, 00000.24, 00002.14 ], Mag (uT) [ -00053.25, 00054.90, 00030.45 ], Tmp (C) [ 00030.43 ]
47 Scaled. Acc (mg) [ -00010.74, 00980.47, -00114.26 ], Gyr (DPS) [ 00001.10, 00000.97, -00001.14 ], Mag (uT) [ -00053.25, 00054.90, 00031.50 ], Tmp (C) [ 00029.95 ]

```

We can see gravity here in the y component.

The screenshot shows a terminal window with the title "SparkFun ESP32-S2 Thing". The window displays the following content:

```
Example1_Basics.ino debug_custom.json
1 //*****
2 * Example1_Basics.ino
3 * ICM 20948 Arduino Library Demo
4 * Use the default configuration to stream 9-axis IMU data
5 * Owen Lyke @ SparkFun Electronics
6 * Original Creation Date: April 17 2019
7 *
8 * Please see License.md for the license information.
9 *
10 * Distributed as-is; no warranty is given.
11 ****
12 #include "ICM_20948.h" // Click here to get the library: http://librarymanager/All#SparkFun\_ICM\_20948\_IMU
13
14 // #define USE_SPI      // Uncomment this to use SPI
15
16 #define SERIAL_PORT Serial
17
18 #define SPI_PORT SPI // Your desired SPI port.      Used only when "USE_SPI" is defined
19 #define CS_PIN 2     // Which pin you connect CS to. Used only when "USE_SPI" is defined
20
21 #define WIRE_PORT Wire // Your desired Wire port.    Used when "USE_SPI" is not defined
22 // The value of the last bit of the I2C address.
23 // On the SparkFun 9DoF IMU breakout the default is 1, and when the ADR jumper is closed the value becomes 0
24 #define ADR_VAL 1
25
26 #ifdef USE_SPI
27 ICM_20948_SPI myIMU; // If using SPI create an ICM_20948_SPI object
28 #else
29 ICM_20948_T2C myT2C; // Otherwise create an TCM_20948_T2C object
30
Output: Serial Monitor x
Message (Enter to send message to 'SparkFun ESP32-S2 Thing Plus' on '/dev/cu.usbserial-210') New Line 115200 baud
Scaled. Acc (mg) [ 01002.44, -00159.67, -00000.49 ], Gyr (DPS) [ 00002.21, -00003.35, 00000.05 ], Mag (uT) [ -00072.90, 00041.10, 00035.85 ], Tmp (C) [ 00030.43 ]
Scaled. Acc (mg) [ 00988.28, -00129.88, -00005.86 ], Gyr (DPS) [ 00001.55, -00003.28, 00000.40 ], Mag (uT) [ -00074.55, 00040.50, 00036.30 ], Tmp (C) [ 00030.67 ]
Scaled. Acc (mg) [ 00983.54, -00123.54, -00001.05 ], Gyr (DPS) [ -00006.44, -00006.09, 00003.41 ], Mag (uT) [ -00073.80, 00040.95, 00034.95 ], Tmp (C) [ 00030.43 ]
Scaled. Acc (mg) [ 00995.61, -00117.19, -00007.81 ], Gyr (DPS) [ 00000.39, -00004.75, 00001.82 ], Mag (uT) [ -00074.70, 00039.90, 00034.65 ], Tmp (C) [ 00030.47 ]
Scaled. Acc (mg) [ 00799.49, -00007.37, -00021.00 ], Gyr (DPS) [ -00000.63, -00003.66, 00007.49 ], Mag (uT) [ -00076.50, 00039.75, 00036.30 ], Tmp (C) [ 00030.47 ]
Scaled. Acc (mg) [ 00986.82, -00141.11, -00063.96 ], Gyr (DPS) [ 00002.52, 00001.31, 00004.21 ], Mag (uT) [ -00073.05, 00038.55, 00035.25 ], Tmp (C) [ 00030.43 ]
Scaled. Acc (mg) [ 00962.68, -00043.46, -00000.62, 00001.24 ], Gyr (DPS) [ -00003.88, -00000.62, 00001.24 ], Mag (uT) [ -00074.55, 00039.30, 00035.55 ], Tmp (C) [ 00030.81 ]
Scaled. Acc (mg) [ 00899.75, -00152.83, -00011.23 ], Gyr (DPS) [ 00000.90, 00000.78, -00000.95 ], Mag (uT) [ -00073.95, 00039.60, 00034.35 ], Tmp (C) [ 00030.47 ]
Scaled. Acc (mg) [ 00797.79, -00160.64, -00022.46 ], Gyr (DPS) [ 00002.92, 00000.78, -00004.20 ], Mag (uT) [ -00073.05, 00038.55, 00035.55 ], Tmp (C) [ 00030.43 ]
Scaled. Acc (mg) [ 00996.09, -00147.95, -00019.53 ], Gyr (DPS) [ 00002.23, 00003.24, -00003.33 ], Mag (uT) [ -00074.55, 00040.20, 00034.50 ], Tmp (C) [ 00030.33 ]
Scaled. Acc (mg) [ 00997.56, -00112.30, -00024.90 ], Gyr (DPS) [ -00003.12, 00002.99, -00000.55 ], Mag (uT) [ -00073.20, 00038.85, 00035.10 ], Tmp (C) [ 00030.52 ]
Scaled. Acc (mg) [ 00982.91, -00134.77, -00026.37 ], Gyr (DPS) [ 00001.77, -00002.61, 00000.05 ], Mag (uT) [ -00073.05, 00039.75, 00036.00 ], Tmp (C) [ 00030.33 ]
Scaled. Acc (mg) [ 00993.65, -00139.16, -00021.97 ], Gyr (DPS) [ 00001.56, 00002.73, -00000.74 ], Mag (uT) [ -00074.55, 00040.20, 00035.25 ], Tmp (C) [ 00030.52 ]
Scaled. Acc (mg) [ 00967.77, -00128.42, -00029.30 ], Gyr (DPS) [ -00001.34, 00004.82, -00000.63 ], Mag (uT) [ -00072.90, 00040.35, 00035.10 ], Tmp (C) [ 00030.43 ]
```

We can see gravity in the x component.

It just depends on the orientation of our IMU.

4.2

Roughly, when in idle state, the values of acceleration and gyro tend to be around 0 (except for acc in the direction of gravity). They tend to drift a tiny bit in idle state if the IMU isn't 100% steady. A good way to classify idle vs non-idle is by checking if a certain variable changed by a lot.

4.3

In order to build my classifier to detect forward push and upward lift, I used the accelerations in the x and z directions to detect any sudden changes, by comparing the values stored 30 ms before.

4.4

In order to build a classifier for 3 actions, I had to use the gyro as well to detect circular motions. I detected the gyro in the z direction. I think it does a fairly decent job, although it is not perfect. I think just moving in one direction, along x, y, or z direction is a little bit easier than detecting circular motions. It's important to note that when moving the IMU upward or forward and due to the sensitivity of the IMU, it will always detect circular motion with it.