ASSUMPTION UNIVERSITY

Vincent Mary School of Science and Technology

Smart band

CS 4401/ ITX 4306 Internet Of Thing

Full Report

Submitted To:	**A.Pawut Satitsuksanoh**	
Submitted By:	Mr. Jirapat Suwanjunee	621-7410
	Mr. Krittamet Chuwongworaphinit	611-1252

TABLE OF CONTENTS

Table of Figure

1 INTRODUCTION TO SMART BAND

1.1 Problem

Nowadays Technology plays an important role in society today. It has positive and negative effects on the world and it impacts daily lives. We believe that people have been influenced by electronic devices. While a smartphone, tablet, or computer can be a useful tool, it's forcing people like us to use these devices can interfere with work, school, and communication. When you spend a lot of time on social media or checking texts, emails, playing games than interacting with real people, sometimes you can't stop yourself from repeatedly. Moreover, some people can not be unable to focus on the thing that they are doing because of the phone addiction. For example, someone thinks that I will use a smartphone for 5-10 minutes but it takes longer than that.

1.2 Overview

Smart band is the idea that we come up with. By going to help people that facing electronic devices addiction. We creating a detecting band to detect people that use any kind of electronic devices that they spend more time than it use to be so it will remind users with sound effects to ensure that the user spends a lot of time on that devices.

1.3 Objective

1. Alert the user if they hold up the things for a long time.
2. Preventing addiction to electronic devices.
3. Reduce unnecessary movement.

2 EQUIPMENT

1. Arduino nano / mini esp8266

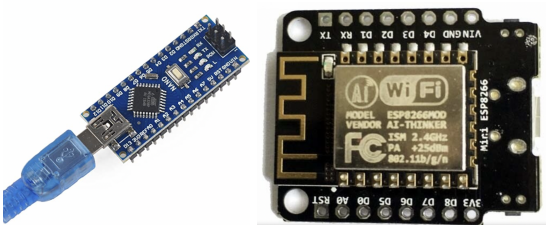

Figure 1:*Arduino nano / mini esp8266*

2. MPU6050 micro-electro-mechanical is consist of a three-axis accelerometer and three axis gyroscope it helps us to measure velocity.

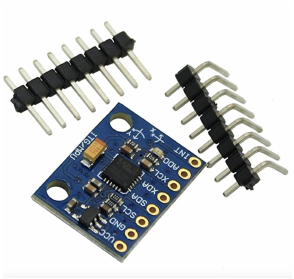

Figure 2: *MPU6050 micro-electro-mechanical*

3.Battery

Figure 3: *Battery*

4. Jumper wire

Figure 4: *Jumper Wire*

5. buzzer

Figure 5: *Buzzer*

6. ultrasonic (optional)

Figure 6: Ultrasonic

7. DSD Tech hc-05 Bluetooth (optional)

Figure 7: *DSD Tech HC-05 Bluetooth*

3. Getting Started

Based on the equipment that we mention above let we introduce The MPU-6050 Gyroscope Accelerometer Sensor The MPU-6050 is a module with a 3-axis accelerometer and a 3-axis gyroscope which be able to measure rotational velocity (rad/s), this is the change of the angular position over time along the X, Y, and Z-axis to determine the orientation of an object.

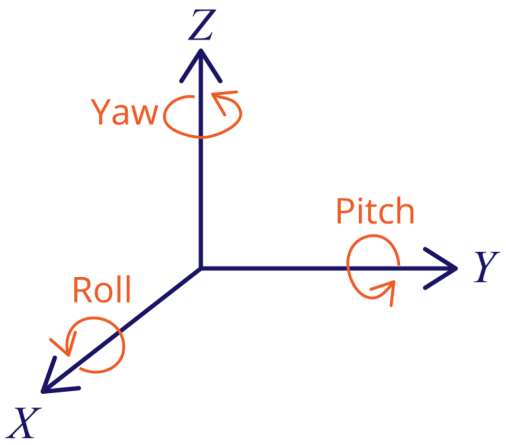

Figure 8: *X, Y, Z, Axis Dimensions*

A buzzer is a kind of loudspeaker but of low power that will emit a sound according to the frequency and amplitude of vibration. It allows to play notes and to recreate simple melodies. If you want to play sounds such as music or voices.

Figure 9: *Buzzer is creating sound*

3.1 Node MPU 6050 Connect with ESP 8266 Circuit

We have to create a connection by connecting ESP8266 NodeMCU with MPU-6050 Accelerometer, Gyroscope. The MPU-6050 IMU (Inertial Measurement Unit) is a 3-axis accelerometer and 3-axis gyroscope sensor. The accelerometer measures the gravitational acceleration and the gyroscope measures the rotational velocity. Then we will code to test node mpu 6050 to see the result of angular velocity (gyroscope) on the x, y, and z-axis, the acceleration on the x, y, and z-axis on the serial monitor.

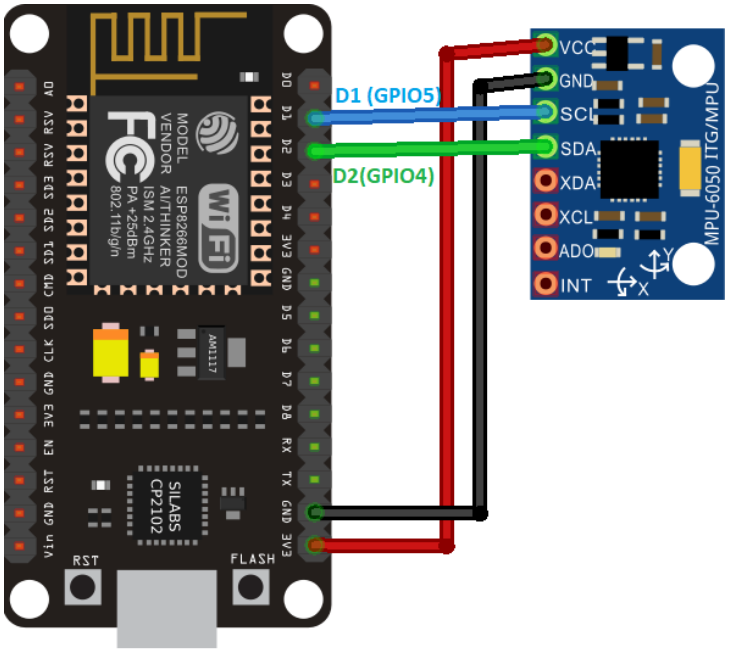

Figure 10: *Node MPU 6050 Connect with ESP 8266 Circuit*

As usual, we have started the code by including all the required libraries. The Wire. h library allows you to communicate with I2C devices while ESP8266.h library provides ESP8266 a specific Wi-Fi routine that we are calling to connect to the network. The Source code program inside the void setup(), we have started the serial monitor at the baud rate of 38400 for serial monitoring MPU 6050 on Arduino ide. Then in the void loop(), we read the MPU6050 sensor data.

```
#include "Wire.h"
#include "I2Cdev.h"
#include "MPU6050.h"
MPU6050 mpu; //ประกาศตัวแปร mpu
int16_t ax, ay, az;
int16_t gx, gy, gz;

long time1;
long previous = 0;
long current;
long axis;
const int buzzerPin = D3;
void setup()
{
  Wire.begin();
  Serial.begin(38400);
  pinMode(buzzerPin, OUTPUT);
  Serial.println("Initialize MPU");
  mpu.initialize();
  Serial.println(mpu.testConnection() ? "Connected" : "Connection failed");
}

void loop(){
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
  ax = map(ax, -18000, 18000, 0, 180);
  ay = map(ay, -18000, 18000, 0, 180);
  az = map(az, -18000, 18000, 0, 180);
  gx = map(gx, -18000, 18000, 0, 180);
  gy = map(gy, -18000, 18000, 0, 180);
```

Figure 11:*Source code of Node MPU Implementation*

This window is called the Serial Monitor and it is part of the Arduino IDE software. Its job is to allow you to both send messages from your computer to an Arduino board (over USB) and also to receive messages from the Arduino and the result that show on the figure is x, y, and z-axis, the acceleration on the x, y, and z-axis.

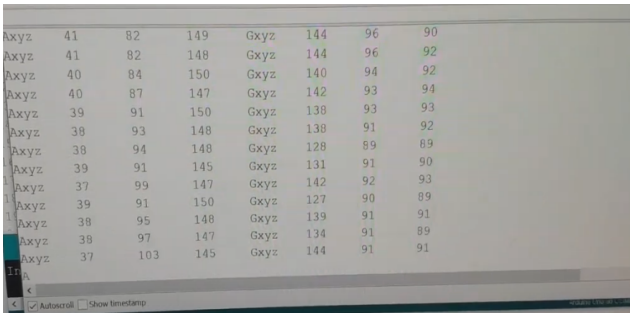

Figure 12:*Serial Monitor result*

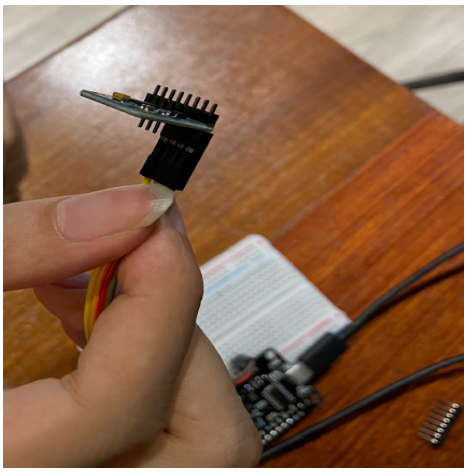

Figure 13:*Connect MPU with ESP8266*

3.2 Buzzer with MPU 6050

Smart band needs notification sound to warn the user that uses phone over time so that why we have to buzzer in this project. Continue from the previous step After we print out the node MPU 6050 serial monitor we would be able to connect the buzzer.

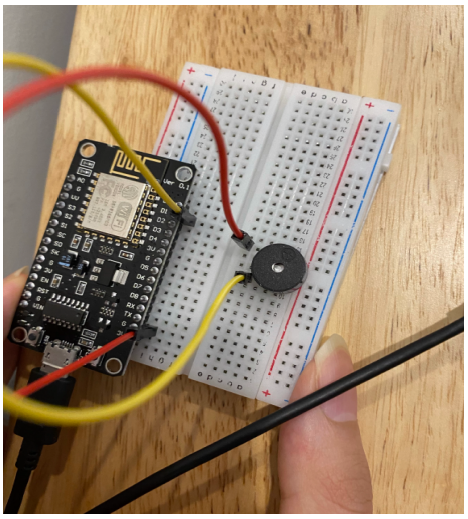

Figure 14: *Connecting Buzzer*

3.3 Result and Implementation

This is a smart band when we combine the node Mpu esp8266, node MCU 6050 together. The picture bottom shows the finalized project that contains mpu6050, esp8266, jumper wire, and buzzer.

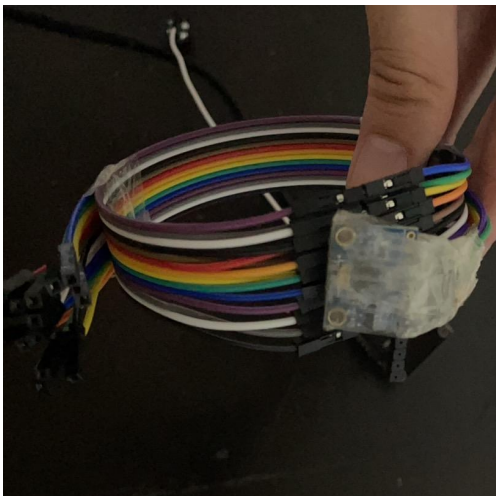

Figure 15: *Smart Band*

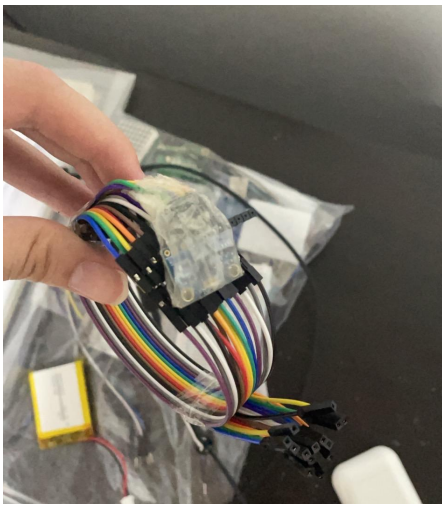

Figure 16: *Smart Band show with MPU*

4. Code and Implementation

```
24  void loop(){
25    mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
26    ax = map(ax, -18000, 18000, 0, 180);
27    ay = map(ay, -18000, 18000, 0, 180);
28    az = map(az, -18000, 18000, 0, 180);
29    gx = map(gx, -18000, 18000, 0, 180);
30    gy = map(gy, -18000, 18000, 0, 180);
31    gz = map(gz, -18000, 18000, 0, 180);
32
33    axis = (ax + ay + az)/3;
34    Serial.print("axis");
35    Serial.println(axis);
36    if (axis > 110) {
37      current = millis()/1000 - previous;
38      Serial.print("current");
39      Serial.println(current);
40      if (current > 20) {
41        while (axis > 110) {
42           mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
43            // แปลงค่าแกนให้เป็นมุม 0-180 องศา
44           ax = map(ax, -18000, 18000, 0, 180);
45           ay = map(ay, -18000, 18000, 0, 180);
46           az = map(az, -18000, 18000, 0, 180);
47           gx = map(gx, -18000, 18000, 0, 180);
48           gy = map(gy, -18000, 18000, 0, 180);
49           gz = map(gz, -18000, 18000, 0, 180);
```

Figure 17: code in void loop part 1()

```
44           ax = map(ax, -18000, 18000, 0, 180);
45           ay = map(ay, -18000, 18000, 0, 180);
46           az = map(az, -18000, 18000, 0, 180);
47           gx = map(gx, -18000, 18000, 0, 180);
48           gy = map(gy, -18000, 18000, 0, 180);
49           gz = map(gz, -18000, 18000, 0, 180);
50         previous = millis()/1000;
51         Serial.print("peep peep");
52         tone(buzzerPin, 500);
53         axis = (ax + ay + az)/3;
54         Serial.print(axis);
55         delay(1000);
56       }
57
58     }
59   } else {
60     previous = millis()/1000;
61     Serial.println("my");
62   }
63   noTone(buzzerPin);
64   delay(50);
65
66 }
```

Figure 18: code in void loop part 2()

In the coding part, we create a variable call axis that will keep the average of rotation x,y,z of mpu6050. On line 36, the number 110 is an average of x,y,z when you move your hand up or hold something up. On line 40, the number 20 is a number of timing that we set when it will alert, so in this code, it will alert after 20 seconds and it will continue alert until you change your movement.