CSC230 – Using ThingSpeak API to Collect and Visualize Data

Joe Parisia

The purpose of this Lab is to learn how to use MATLAB’s ThingSpeak API to connect our M5 and collect data from various sensors. Our first step is to get familiar with ThingSpeak by creating a Login and going through some example code from GitHub. The GitHub link ([found here](https://github.com/nothans/thingspeak-esp-examples)) shows us how to make a “secret” file to store our SSID and Password for our Internet as well as some basic examples of code for ThingSpeak. One of the first examples was to connect the M5 to your internet and broadcast the RSSI to our newly created ThingSpeak channel (which you can see below).



Measuring RSSI using our M5 and ThingSpeak

Once we got an idea of how ThingSpeak works we were to take our previous Code from the ENV Hats assignment and combine it so that We get a temperature readout every so often (in my case I did every 30 seconds) sent to our ThingSpeak channel. In my program I decided to stretch it a bit and learn how to write to multiple fields at once. My two channels were RSSI and Temperature.



Temperature Readings were added on to the RSSI Channel

Once we program that we are to create our own program using a M5 Hat and ThingSpeak. I chose to incorporate my PIR Hat with the M5 so I can get closer to having the knowledge necessary to create my Final Project. I have my M5 report to ThingSpeak if it is at 0 (no motion) or 1 (motion detected). I will have my code used for this at the end of my report. Below you will find some of my results from the working code.

This shows 0 while I was making a sandwich and 1 when I returned and was triggering the PIR

I did run into one issue with my PIR ThingSpeak combination. I don’t think ThingSpeak is able to process the data fast enough for the PIR sensor to work as effectively as I wanted. ThingSpeak takes around 10 seconds to send the state of the M5 to my channel. I tried several different ways of coding this but nothing changed the overall time required. My motion detector is still useful but it only sends the proper result if motion is continuously triggered for 5-10 seconds instead of every instance it is detected.

The materials needed for this Lab were pretty simple.

* M5StickC
* PIR HAT
* ENV HAT
* ThingSpeak API website
* Wireless Internet

Below you will find a video of the M5 Uploading to ThingSpeak:



<https://www.youtube.com/watch?v=pbrt2WEzsug&feature=youtu.be>

The video is self-explanatory, the M5 doesn’t do anything special while it is doing this.

Below you will find my code for the ENV Version and my PIR Version of the program. I have the code commented to show how it runs properly.

**PIR ThingSpeak Code:**

#include <WiFi.h>

#include <WiFiClient.h>

#include <WiFiServer.h>

#include <M5StickC.h>

#include "ThingSpeak.h"

#include "M5\_PIR\_secrets.h"

unsigned long myChannelNumber = SECRET\_CH\_ID;

const char \* myWriteAPIKey = SECRET\_WRITE\_APIKEY;

char ssid[] = SECRET\_SSID; // your network SSID (name)

char pass[] = SECRET\_PASS; // your network password

int keyIndex = 0; // your network key index number (needed only for WEP)

WiFiClient client;

const int motion\_state = 36; //variable for my conditionals. Set to the PIR Hat Pin

const int button = 37; //variable for the button so I dont have to mess with the M5 function

void setup() {

 Serial.begin(115200);

 delay(50);

 WiFi.mode(WIFI\_STA);

 ThingSpeak.begin(client);

 M5.begin();

 M5.Lcd.setRotation(1);

 M5.Lcd.fillScreen(BLACK);

 M5.Lcd.setTextSize(2);

 pinMode(motion\_state,INPUT\_PULLUP);

 pinMode(button,INPUT\_PULLUP);

}

void loop() {

 // Connect or reconnect to WiFi

 if (WiFi.status() != WL\_CONNECTED) {

 Serial.print("Attempting to connect to SSID: ");

 Serial.println(SECRET\_SSID);

 while (WiFi.status() != WL\_CONNECTED) {

 WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line if using open or WEP network

 Serial.print(".");

 delay(5000);

 }

 Serial.println("\nConnected."); // Serial Monitor Output when the device is 'Connected'

 }

 //This is the beginning of my PIR Code

 if (digitalRead(motion\_state) == 1){ //if Motion is triggered run this code

 M5.Lcd.fillScreen(RED);

 M5.Lcd.setCursor(40, 20);

 M5.Lcd.println("ALERT!");

 Serial.println("Motion Detected");

 int httpCode\_alert = ThingSpeak.writeField(myChannelNumber, 1, 1, myWriteAPIKey);

 delay(5000);

 if (httpCode\_alert == 200) {

 Serial.println("Channel write successful.");

 }

 }

 if (digitalRead(motion\_state) == 0){ //to reset the PIR to activate again

 Serial.println("Nothing Detected");

 M5.Lcd.fillScreen(BLACK);

 int httpCode\_alert = ThingSpeak.writeField(myChannelNumber, 1, 0, myWriteAPIKey);

 delay(1000);

 if (httpCode\_alert == 200) {

 Serial.println("Channel write successful.");

 }

 }

}

**ENV ThingSpeak Code:**

#include <WiFi.h>

#include <WiFiClient.h>

#include <WiFiServer.h>

#include <M5StickC.h>

#include <Wire.h>

#include "Adafruit\_Sensor.h"

#include "DHT12.h"

#include "ThingSpeak.h"

#include "M5\_ENV\_secrets.h"

unsigned long myChannelNumber = SECRET\_CH\_ID;

const char \* myWriteAPIKey = SECRET\_WRITE\_APIKEY;

DHT12 dht12;

char ssid[] = SECRET\_SSID; // your network SSID (name)

char pass[] = SECRET\_PASS; // your network password

int keyIndex = 0; // your network key index number (needed only for WEP)

WiFiClient client;

String myStatus = "";

void setup() {

 Serial.begin(115200);

 delay(100);

 WiFi.mode(WIFI\_STA);

 ThingSpeak.begin(client);

 M5.begin();

 M5.Lcd.setRotation(1);

 Wire.begin(0,26);

}

void loop() {

 // Measure Signal Strength (RSSI) of Wi-Fi connection

 long rssi = WiFi.RSSI();

 // Measures the temperature from the DHT Sensor

 float tmp = dht12.readTemperature();

 // Converts C to F

 float tmp\_f = tmp \* 1.8 + 32;

 //Setting Fields So I update multiple simultaneously

 ThingSpeak.setField(1, rssi);

 ThingSpeak.setField(2, tmp\_f);

 // Connect or reconnect to WiFi

 if (WiFi.status() != WL\_CONNECTED) {

 Serial.print("Attempting to connect to SSID: ");

 Serial.println(SECRET\_SSID);

 while (WiFi.status() != WL\_CONNECTED) {

 WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line if using open or WEP network

 Serial.print(".");

 delay(5000);

 }

 Serial.println("\nConnected."); // Serial Monitor Output when the device is 'Connected'

 }

 M5.Lcd.fillScreen(GREEN);

 M5.Lcd.setTextColor(BLACK);

 M5.Lcd.setTextSize(2);

 M5.Lcd.setCursor(5,20);

 M5.Lcd.printf("Connected");

 M5.Lcd.setCursor(5,40);

 M5.Lcd.printf("TEMP: %2.2f", tmp\_f);

 M5.update();

 ThingSpeak.setStatus(myStatus);

 // Write value to Field 1 of a ThingSpeak Channel

 int httpCode = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);

 if (httpCode == 200) {

 Serial.println("Channel write successful.");

 }

 else {

 Serial.println("Problem writing to channel. HTTP error code " + String(httpCode));

 M5.Lcd.fillScreen(RED);

 M5.Lcd.setTextColor(BLACK);

 M5.Lcd.setTextSize(2);

 M5.Lcd.setCursor(0,20);

 M5.Lcd.printf("Connection Error");

 }

 // Wait 30 seconds to update the channel again

 delay(30000);

}

Here are some bonus pictures of the M5 in action:



Picture 1 Shows my output for the ENV
Picture 2 I stuck my M5 to my air vent to get different readings (cold)

Picture 3 Shows when Motion is detected during my PIR program