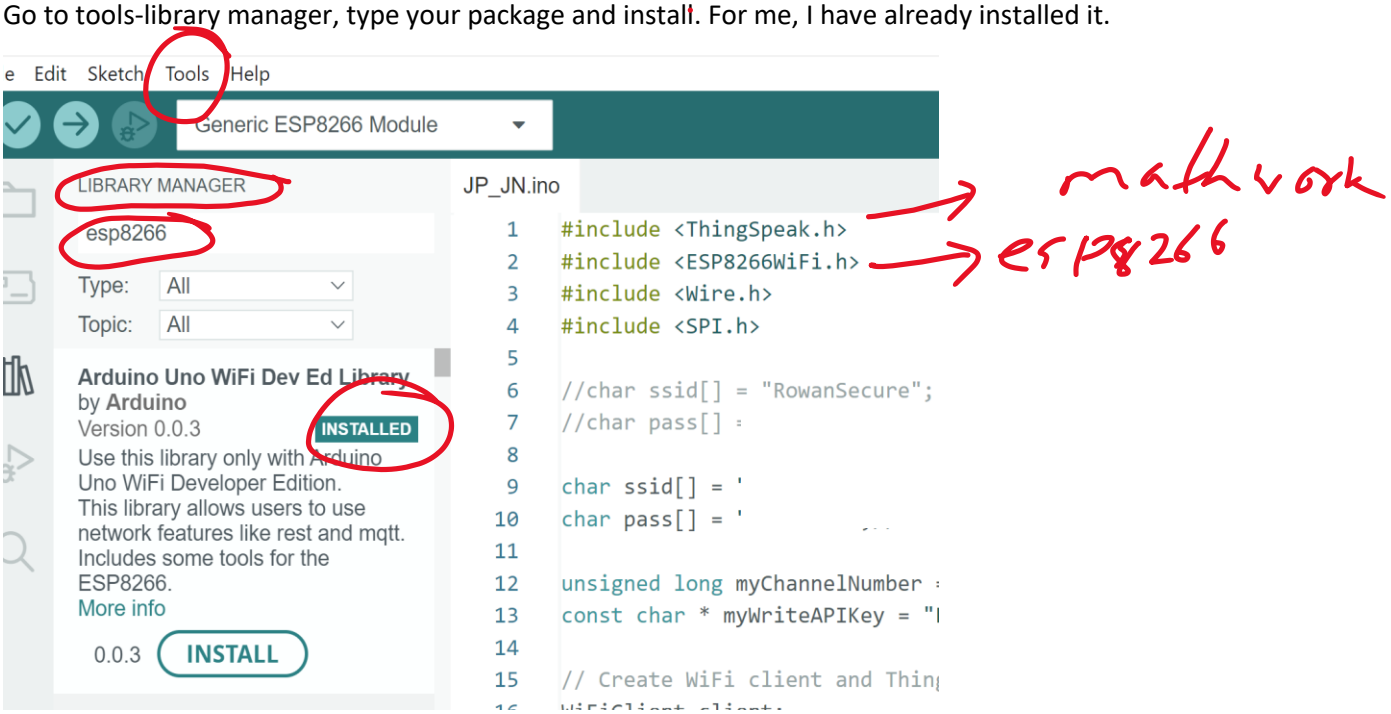
Making an IoT applications using ESP8266:

Step1:Package Installation

Go to tools-library manager, type your package and install. For me, I have already installed it.



Step2: Finding MAC address (WiFi-ESP8266). It is not required, if you use your home network

Link: https://techtutorialsx.com/2017/04/09/esp8266-get-mac-address/

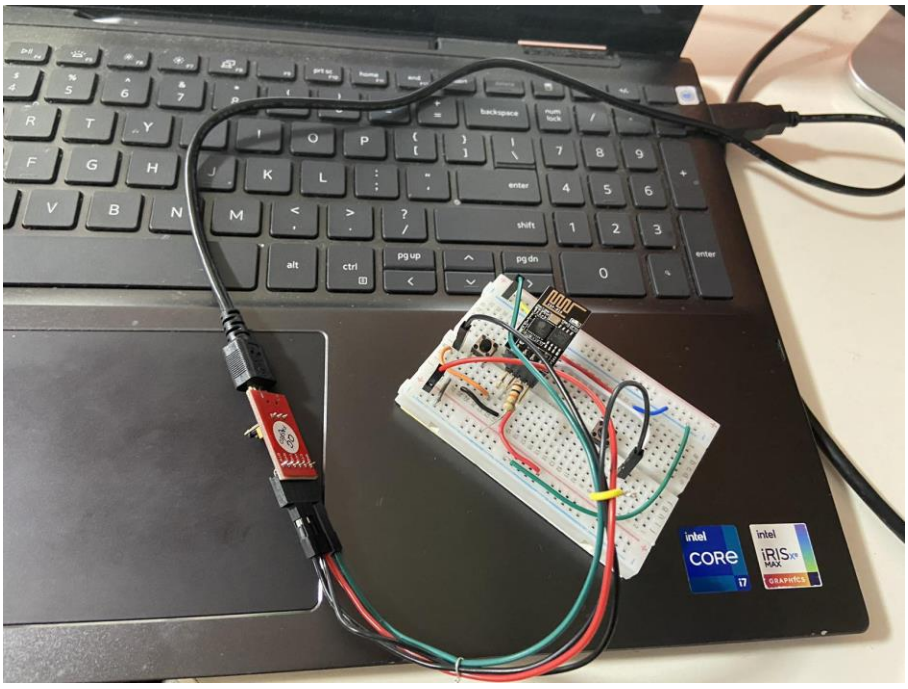
This link will show how to find MAC address for your ESP8266. You need to upload the following code to the board. Then the address will appear in the console on the Arduino IDE. You have to add this MAC in rowan.edu/devices if you use the rowan network.

```
#include <ESP8266WiFi.h>

void setup(){

   Serial.begin(115200);
   delay(500);

   Serial.println();
   Serial.print("MAC: ");
   Serial.println(WiFi.macAddress());

}

void loop(){}
```

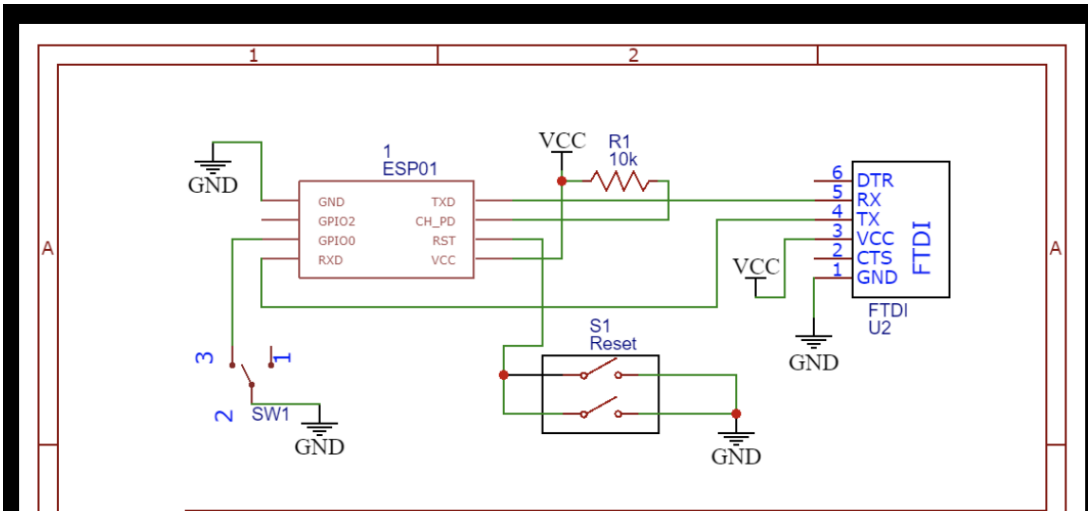
```
#include <ThingSpeak.h>
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <SPI.h>

//char ssid[] = "                // replace MySSID with your WiFi network name
//char pass[] =                  /replace MyPASS with your WiFi network password

char ssid[] = "                                                  MySSID with your WiFi network nam
char pass[] = "                                                  ; //replace MyPASS with your WiFi

unsigned long myChannelNumber = 1563508; // replace 0000000 with your channel number
const char * myWriteAPIKey = "F0252SK1A7O3D2J9"; // replace MyAPIKEY with your thingspeak wr

// Create WiFi client and ThingSpeak class.
WiFiClient client;
ThingSpeakClass ts;
```

I tried to use the rowan network. Unfortunately it didn't work for me. If you use, I will recommend to use 'RowanSecure'. I used the hotspot network from my phone.

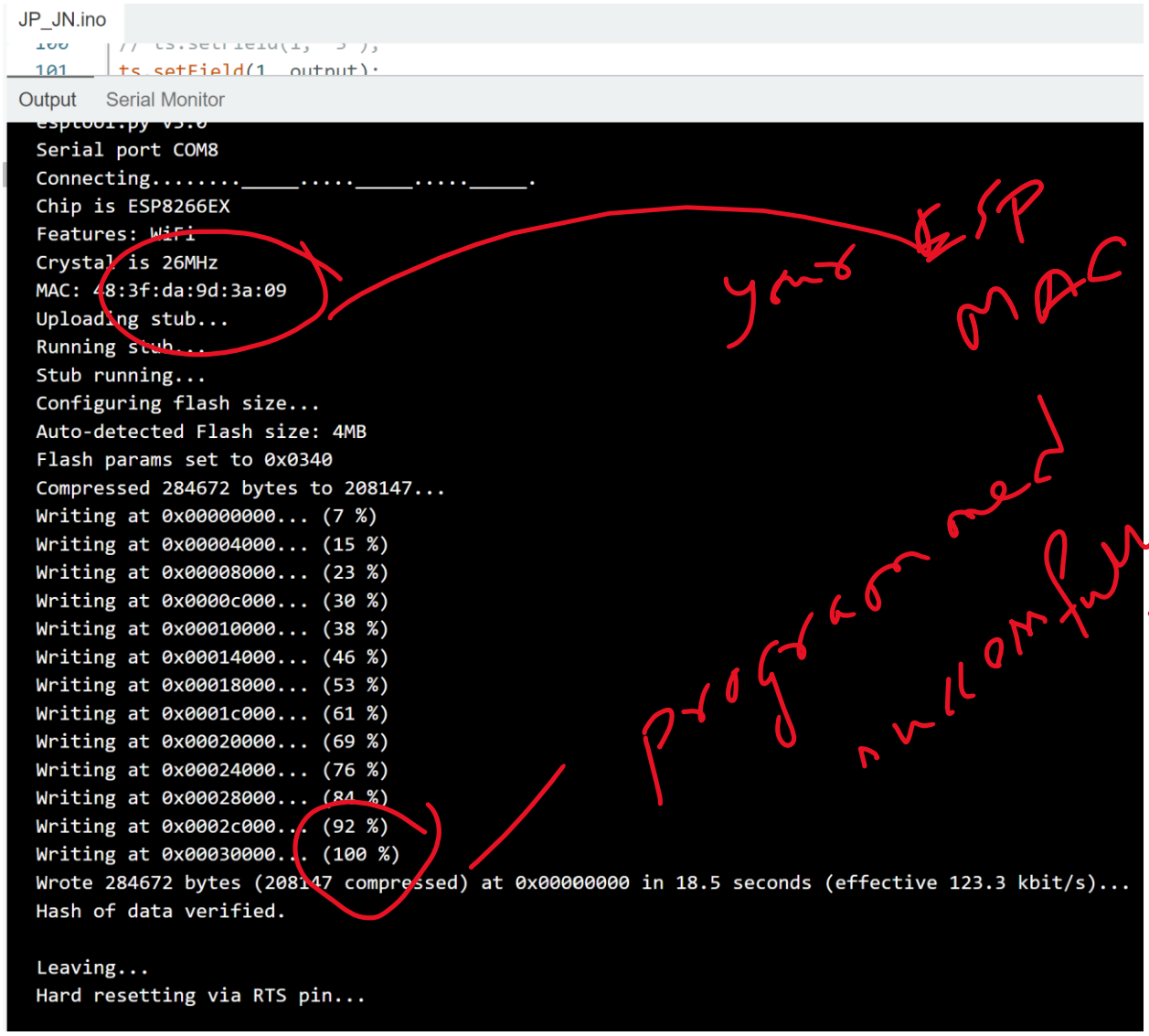
Step3: Program the Wifi through FTDI



- ✓ After adding your MAC. Now you will run a different sketch in your Arduino IDE.
- ✓ You can borrow the FTDI chip from resource center. It will look like a cable but it has a FTDI chip inside of it.
- ✓ I used the following schematic for the breadboard connection. You use your own schematic and build the circuit according to that.

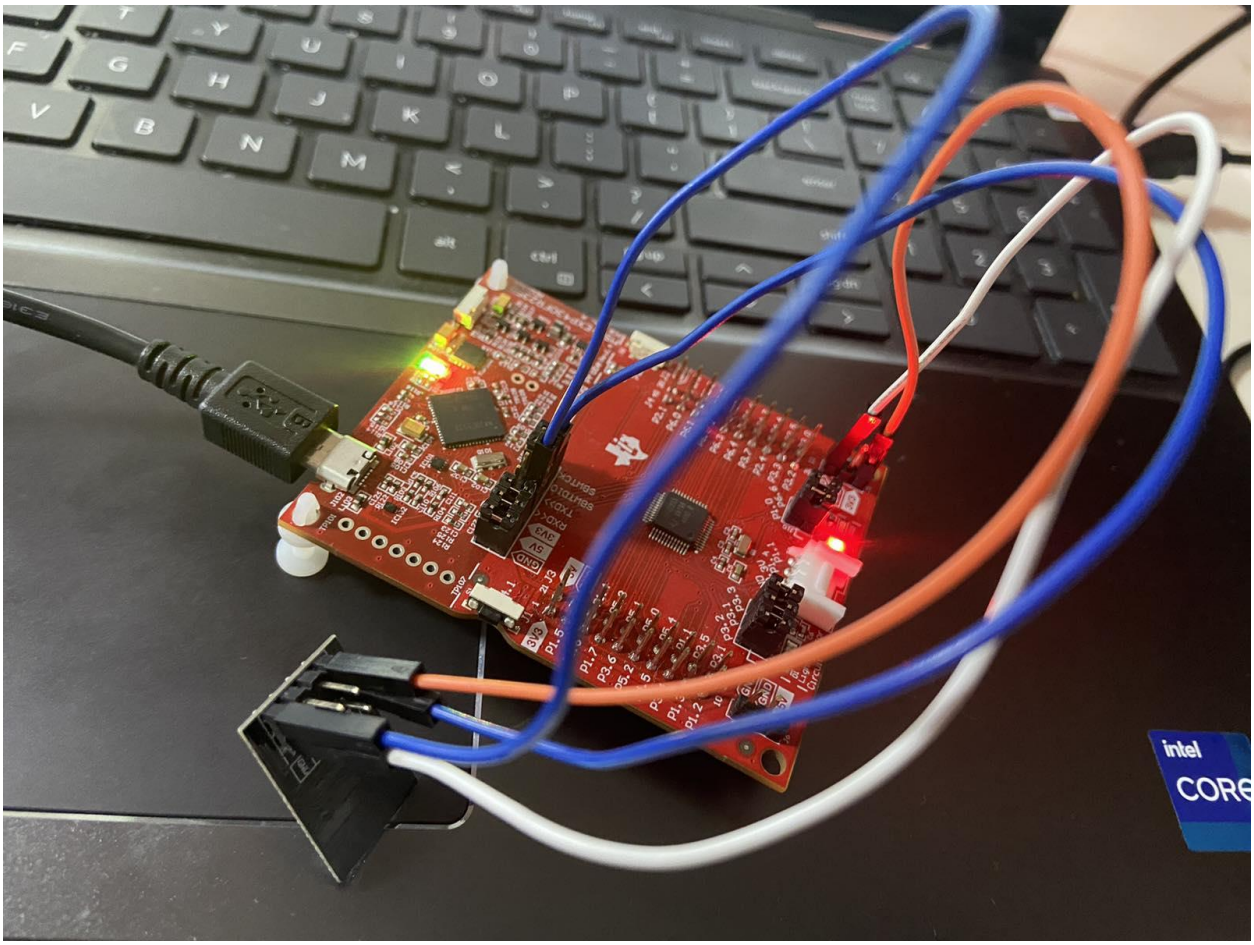


- ✓ First, be sure you tie the GPIO 0 pin on the ESP to ground to set it to programming mode. Then, you need to float that pin (no connection) to make the module work. Once you float the pin, you can reset it and your uploaded code should run.
- ✓ For the programming circuitry, be sure you have the button on the booster pack to reset the wi-fi module.
- ✓ When programming, select "Generic ESP8266 Module" as target board".
- ✓ Select your serial port. Make sure the baud rate is 115200 in your serial monitor
- ✓ This part was little tedious. You need to press your boot and reset button from the breadboard connection a couple of times and please do it, after compiling when it shows uploading.
- ✓ It may give you error but try it a couple of times. Sometimes it doesn't work for switch bouncing.
- ✓ After a successful completion of programming the ESP8266 through FTDI, you will find the following screen.



Now your Serial monitor will show ‘connected’

Step4: Disconnect FTDI, Connect WiFi with MSP430



Step6: Make sure, you use the same data format from CCS and Arduino. In the following code we used such as 25; 26; 90; format

```
#include <ThingSpeak.h>
#include <ESP8266WiFi.h>
#include <Wire.h>
#include <SPI.h>

char ssid[] = "xxx";//
char pass[] = "xxx";//"

unsigned long myChannelNumber = 1563508; // replace 0000000 with your channel number
const char * myWriteAPIKey = "F0252SK1A7O3D2J9"; // replace MyAPIKEY with your thingspeak write API key

// Create WiFi client and ThingSpeak class.
WiFiClient client;
ThingSpeakClass ts;
```

```
// WiFi initialization
void init_WIFI()
{
  // Serial print commands for testing.
  // If you are using an FTDI and aren't in programming mode, you'll be able to
see these.
  Serial.print("Attempting to connect to SSID: ");
  Serial.println(ssid);

  // While not connected, attempt to connect
  while (WiFi.status() != WL_CONNECTED)
  {
    // Connect to the network
    WiFi.begin(ssid, pass);
    // Delay 5 seconds
    delay(5000);

    Serial.print(".");
  }

  Serial.println("\nConnected.");
}

// Initial board set-up
void setup()
{
  // Initialize UART to 115200 baud rate
  Serial.begin(115200);

  // Set WiFi mode
  WiFi.mode(WIFI_STA);

  // Initialize ThingSpeak client
  ts.begin(client);
}

// Loop runs constantly
void loop()
{
  //If not connected, connect to wifi
  if (WiFi.status() != WL_CONNECTED)
  {
    init_WIFI();
  }
```

```
  int newByte = 0; // incoming byte from serial input
  char c; //Variable to hold incoming character
  String output = ""; //Variable to hold the concatanated string

  // Wait until there is any data available on the serial buffer
  while (Serial.available() == 0)
  {
    delay(100);
  }

  // Boolean to track if the message is finished
  bool receivedEOM = false;

  // Until an end-of-message character is received...
  while (!receivedEOM)
  {
    // If there is data in the serial buffer...
    if (Serial.available() > 0)
    {
      // Read in the new data and convert to a character
      newByte = Serial.read();
      c = (char) newByte;

      // If a semi-colon (end-of-message character) is received, end the message
      if (c == ';')
      {
        receivedEOM = true;
      }
      // Otherwise, add it to the message
      else
      {
        output += c; //add it to the string
      }
    }
  }

  // Set a ThingSpeak field. If you have multiple sensors, you'll use multiple
fields.
 // ts.setField(1, '3');
 ts.setField(1, output);

  // Upload the data to ThingSpeak. Receive an integer code back from the site.
  int code = ts.writeFields(myChannelNumber, myWriteAPIKey);
```

```
  // Code 200 indicates a successful upload. Other codes indicate errors.
  if (code == 200)
  {
    Serial.println("Channel update successful.");
  }
  else
  {
    Serial.println("Problem setting Field 1. HTTP error code " +
    String(code));
  }

  delay(15500);
}
```

Please make sure the data is coming in the same way from your CCS.

```
if(m==0){
          //initialize_Adc();
          PMMCTL0_H = PMMPW_H;
                    PMMCTL2 |= INTREFEN | TSENSOREN | REFVSEL_0;
                        ConfigureAdc_temp1();
                        ADCCTL0 |= ADCENC + ADCSC +ADCMSC;        //
                        while((ADCCTL0 & ADCIFG) == 0);    // check t
                        _delay_cycles(200000);

                temp1 = ADCMEM0;                    // read the converte
                ADCCTL0 &= ~ADCIFG;

                IntDegC1 = (temp1-CALADC_15V_30C)*(85-30)/(CALADC_15V_85
                itoa(IntDegC1,result,10);
                int acount =0;
              while(result[acount]!='\0')
            {
                  while((UCA1IFG & UCTXIFG)==0);
                  UCA1TXBUF = result[acount++] ;                    //Tr
            }
            m=1;

  if(m == 1){
                            _delay_cycles(20000);

                        int acount =0;
                         result[acount]=';';

                         while((UCA1IFG & UCTXIFG)==0);
                               UCA1TXBUF = result[acount] ;

                            m=0;


                    }
```

Step7: Look in your thing speak channel, data will be there.