```
/*
 * ProjectRapidIoTandPrototypingMidterm#2
 * Description: Midterm#2v9
 * Author: Vernon Cox
 * Date: 21-MAR-2023
 */


#include <math.h>

#include <Adafruit_MQTT.h>
#include "Adafruit_MQTT/Adafruit_MQTT_SPARK.h"
#include "Adafruit_MQTT/Adafruit_MQTT.h"

#include "credentials.h"

#include "Adafruit_SSD1306.h"
#include "Adafruit_BME280.h"
#include "Grove_Air_Quality_Sensor.h"
#include "Air_Quality_Sensor.h"

#define OLED_RESET D4


/************Declare Variables*************/
unsigned int last, lastTime;
float subValue,pubValue;

int soilentGreenpin=A5; //moistSensor pin
int soilentReadgreen;//moistSensor readings


int pumpState;//is pump onOff
int pumpread;
const int pumpPIN=D11;//pump pin

int airSensorpin=A3; //airQualUnit pin
int airQuality;//air Quality reading


const int dustReadpin = D8;
int dustReadg;

unsigned long duration;
unsigned long starttime;
unsigned long sampletime_ms = 30000;//sampe 30s ;
unsigned long lowpulseoccupancy = 0.0;
float ratio = 0.0;
float concentration = 0.0;

float tempC;
float tempF;
float pressPA;
float humidRH;
int hexAddress;

int rot1=1;
int rot2=2;
int rot3=3;
int rot0=0;

/************Declare Functions*************/
void MQTT_connect();
bool MQTT_ping();
bool status;

#if (SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif




SYSTEM_MODE(SEMI_AUTOMATIC);


Adafruit_BME280 bme;
AirQualitySensor sensor(A0);
Adafruit_SSD1306 display(OLED_RESET);


/*
Copy the Adafruit.io Setup line and the next four lines to a credentials.h file

//************************ Adafruit.io Setup
*****************************************
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883        // use 1883 for SSL
#define AIO_USERNAME    "username"  // replace with your Adafruit.io username
#define AIO_KEY         "key"       // replace with your Adafruit.io key
*/
```

```
/************ Global State (you don't need to change this!)
***   ***************/
TCPClient TheClient;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and
login details.
Adafruit_MQTT_SPARK
mqtt(&TheClient,AIO_SERVER,AIO_SERVERPORT,AIO_USERNAME,AIO_KEY);

/****************************** Feeds ***************************************/
// Setup Feeds to publish or subscribe
Adafruit_MQTT_Subscribe waterPump = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME
"/feeds/pumpPIN");
Adafruit_MQTT_Publish soilMoisture = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/soilentReadgreen");
Adafruit_MQTT_Publish dust = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/concentration");
Adafruit_MQTT_Publish air = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/sensor.getValue()");




void setup() {
  pinMode(soilentGreenpin, INPUT);
  pinMode(pumpPIN, OUTPUT);
  pinMode(dustReadpin,INPUT);
  starttime = millis();//get the current time;
  Serial.begin(9600);
  waitFor(Serial.isConnected,10000);

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // initialize with the I2C addr 0x3C
(for the 128x64)

  // Connect to Internet but not Particle Cloud
  WiFi.on();
  WiFi.connect();
  while(WiFi.connecting()) {
    Serial.printf(".");
    delay(100);
  }
  Serial.printf("\n\n");

  while (!Serial);
```

```
    Serial.printf("Waiting sensor to init...");
  delay(20000);

  if (sensor.init()) {
    Serial.printf("Sensor ready.");
  }
  else {
    Serial.printf("Sensor ERROR!");
  }
  display.setCursor(0,0);
  display.display(); //this will show the Adafruit logo (splashscreen)
  delay(1500); //this will delay the splashscreen to increase the marketing
effect
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(25,25);
  display.printf("Hello\n");
  display.display();
    delay(1500);
  display.clearDisplay();
  display.setCursor(5,25);
  display.printf("World\n");
  display.display();
   delay(1500);
  display.clearDisplay();
  display.setCursor(5,25);
  display.printf("My name is");
  display.display();
    delay(1000);
  display.clearDisplay();
  display.setCursor(5,25);
  display.printf("Se%cor",164);
  display.display();
  delay(1000);
  display.clearDisplay();
  display.setCursor(3,25);
  display.printf("Vernon Cox");
  display.display();
  delay(2000);
  display.clearDisplay();
  display.setCursor(0,25);
  display.printf("BornOnDate:");
  display.display();
  delay(2000);
  display.clearDisplay();
  display.setCursor(15,25);
  display.printf("%c%c%c%c%c%c%c", 50,54,65,80,82,53,56);
  display.display();
  delay(3000);

  display.invertDisplay(true);
  delay(1000);

  status = bme.begin(0x76);
    if (status == false){
    Serial.printf("BME280 at address 0x%02X failed to start\n", hexAddress);
    }


  // Setup MQTT subscription
  //mqtt.subscribe(&subFeed);//must tell Argon to subscribe..ima leave this line
in so I know how to do it.
}

void loop() {
  //connecting the MQTT
  MQTT_connect();
  MQTT_ping();
  soilentReadgreen=analogRead(soilentGreenpin);
  dustReadg=analogRead(dustReadpin);

         //Getting data from BME280
    tempC=bme.readTemperature(); //deg C
    tempF=((tempC*1.8)+32);
    pressPA=bme.readPressure(); //pascals
    humidRH = bme.readHumidity(); //%RH


  // this is our 'wait for incoming subscription packets' busy subloop
  Adafruit_MQTT_Subscribe *subscription;

       //Getting data from BME280
    tempC=bme.readTemperature(); //deg C
    tempF=((tempC*1.8)+32);
    pressPA=bme.readPressure(); //pascals
    humidRH = bme.readHumidity(); //%RH

  Serial.printf("The temp in is: %0.1fC\n",tempC); //shows the temp in c
  Serial.printf("The temp in is: %0.1fF\n",tempF); //shows the temp in f
  display.clearDisplay();
  display.setCursor(0,17);
  display.setTextSize(2);
  display.printf("TheTemp in is: %0.1fC\n",tempC); //shows the temp in c
  display.display();
  display.setCursor(0,17);
  delay (1500);
  display.clearDisplay();
  display.printf("TheTemp in is: %0.1fF\n",tempF); //shows the temp in f
  display.display();
  display.setCursor(0,17);
  delay (1500);

  Serial.printf("The pressure in pas is: %0.1f\n",pressPA); //shows the pressure
  display.clearDisplay();
  display.setTextSize(2);
  display.printf("ThePsr is: %0.1f\n",pressPA); //shows the temp in c
  display.display();
  display.setCursor(0,17);
  delay (1500);

  Serial.printf("The HumidT is: %0.1f%%\n",humidRH); //shows the temp in c
  display.clearDisplay();
  display.setTextSize(2);
  display.printf("The HumidT is: %0.1f%%\n",humidRH); //shows the temp in c
  display.display();
  display.setCursor(0,17);
  delay (1500);


  //testing the dust levels
    duration = pulseIn(dustReadpin, LOW);
    lowpulseoccupancy = lowpulseoccupancy+duration;

    if ((millis()-starttime) > sampletime_ms)//if the sampel time == 30s
    {
        ratio = lowpulseoccupancy/(sampletime_ms*10.0);  // Integer percentage
0=>100
        concentration = 1.1*pow(ratio,3)-3.8*pow(ratio,2)+520*ratio+0.62; //
using spec sheet curve
      //  if(lowpulseoccupancy>0){
      //   Serial.printf("Dust Concentration is %f \n",concentration);
      // }
        lowpulseoccupancy = 0;
        starttime = millis();
    }


 //lines below for publishing
  if((millis()-lastTime > 9000)) {
    if(soilentReadgreen>=200){
    if(mqtt.Update()) {
      soilMoisture.publish(soilentReadgreen);
      air.publish(sensor.getValue());
      dust.publish(concentration);
      Serial.printf("Moisture reading is %i which is decent :)
\n",soilentReadgreen);
        if(soilentReadgreen>2000) {
        Serial.printf("Plantsoil is too dry at %i \n",soilentReadgreen);
         digitalWrite(pumpPIN,HIGH);
         Serial.printf("Plant is getting H20 at %i \n",soilentReadgreen);
         delay(350);
         digitalWrite(pumpPIN,LOW);
       }

          if(soilentReadgreen<=1500) {
          Serial.printf("Plantsoil is too wet at %i \n",soilentReadgreen);
          }
               if(concentration>1){
        Serial.printf("Dust Concentration is %f \n",concentration);
        }
        //Serial.printf("Dust Concentration is %f \n",concentration);


  //testing the air quality levels

    int quality = sensor.slope();

  if (quality == AirQualitySensor::FORCE_SIGNAL) {
    Serial.printf("High pollution! Force signal active.");
  }
  else if (quality == AirQualitySensor::HIGH_POLLUTION) {
    Serial.printf("We have HIGH POLLUTION because our sensor value is: %i OH
NO!!! \n", sensor.getValue());
  }
  else if (quality == AirQualitySensor::LOW_POLLUTION) {
    Serial.printf("We have LOW POLLUTION because our sensor value is: %i YEAH for
LOW POLLUTION!!! \n", sensor.getValue());
  }
  else if (quality == AirQualitySensor::FRESH_AIR) {
    Serial.printf("We have FRESH AIR because our sensor value is: %i YEAH for
FRESH AIR!!! \n", sensor.getValue());
  }
    lastTime = millis(); //gets the current time
    }
  //OLED display information
display.setTextSize(2);
display.clearDisplay();
display.setTextSize(1);
display.setCursor(5,25);
display.printf("Quality of is: %i FRESH AIR!!! \n", sensor.getValue());
display.display();
delay(12000);

display.clearDisplay();

display.setRotation(rot2);
display.setCursor(5,15);
display.printf("Vernon Cox");
display.display();
delay(2000);

display.clearDisplay();

display.setRotation(rot3);
display.setCursor(5,15);
display.printf("Vernon Cox");
display.display();
delay(2000);

display.clearDisplay();
display.setTextSize(2);
display.setRotation(rot0);
display.setCursor(5,25);
display.printf("Vernon Cox");
display.display();
delay(2000);


  }
  }

}
```

```
// Function to connect and reconnect as necessary to the MQTT server.
void MQTT_connect() {
  int8_t ret;
  // Return if already connected.
  if (mqtt.connected()) {
    return;
  }
  Serial.print("Connecting to MQTT... ");
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
       Serial.printf("Error Code %s\n",mqtt.connectErrorString(ret));
       Serial.printf("Retrying MQTT connection in 5 seconds...\n");
       mqtt.disconnect();
       delay(5000);  // wait 5 seconds and try again
  }
  Serial.printf("MQTT Connected!\n");
}

bool MQTT_ping() {
  static unsigned int last;
  bool pingStatus;

  //keep the connection alive

  if ((millis()-last)>120000) {
      Serial.printf("Pinging MQTT \n");
      pingStatus = mqtt.ping();
      if(!pingStatus) {
        Serial.printf("Disconnecting \n");
        mqtt.disconnect();
      }
      last = millis();
  }
  return pingStatus;
}
```