The following code implemented in our project for the sensors. We will use Arduino cc to upload the code for the sensors. We learned and understood the code. Below there are the codes for the implementation of the following:

* MQ135 gas sensor
* MQ7 gas sensor
* MQ2 gas sensor
* PM2.5
* PM10
* Wi-Fi Module(ESp8266)

**1. Arduino Uno code For MQ135:**

#include <LiquidCrystal.h>

int sensorValue;

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup(){ lcd.begin(16, 2);

Serial.begin(9600); // sets the serial port to 9600

}

void loop(){sensorValue = analogRead(0); // read analog input pin 0

Serial.print("AirQua=");

Serial.print(sensorValue, DEC); // prints the value read

Serial.println(" PPM");

lcd.setCursor(0,0);

lcd.print("ArQ=");

lcd.print(sensorValue,DEC);

lcd.print(" PPM");

lcd.println(" ");

lcd.print(" ");

delay(100); // wait 100ms for next reading

}

**2. Arduino Uno code For MQ7:**

void setup()

{

Serial.begin(9600);

}

void loop()

{

float sensorVoltage;

float sensorValue;

sensorValue = analogRead(A0);

sensorVoltage = sensorValue/1024\*5.0;

Serial.print("sensor voltage = ");

Serial.print(sensorVoltage);

Serial.println(" V");

delay(1000);

}

**3. Arduino Uno code For MQ2:**

\*\*\*\*\*\*

All the resources for this project:

https://www.electromaniaweb.wordpress.com

https://www.arduino.cc

\*\*\*\*\*\*\*/

int LED1 = 12;

int LED2 = 11;

int buzzer = 10;

int smokeA0 = A5;

int sensorThreshold = 400;

void setup() {

pinMode(LED1, OUTPUT);

pinMode(greenLed, OUTPUT);

pinMode(buzzer, OUTPUT);

pinMode(smokeA0, INPUT);

Serial.begin(9600);

}

void loop() {

int analogSensor = analogRead(smokeA0);

Serial.print("Pin A0: ");

Serial.println(analogSensor);

// Checks if it has reached the threshold value

if (analogSensor > sensorThreshold)

{

digitalWrite(LED1, HIGH);

digitalWrite(LED2, LOW);

tone(buzzer, 1000, 200);

}

else

{

digitalWrite(LED1, LOW);

digitalWrite(LED2, HIGH);

noTone(buzzer);

}

delay(100);

}

**4. Arduino Uno code For PM2.5:**

// On Leonardo/Micro or others with hardware serial, use those!

// uncomment this line:

// #define pmsSerial Serial1

// For UNO and others without hardware serial, we must use software serial...

// pin #2 is IN from sensor (TX pin on sensor), leave pin #3 disconnected

// comment these two lines if using hardware serial

#include <SoftwareSerial.h>

SoftwareSerial pmsSerial(2, 3);

void setup() {

// our debugging output

Serial.begin(115200);

// sensor baud rate is 9600

pmsSerial.begin(9600);

}

struct pms5003data {

uint16\_t framelen;

uint16\_t pm10\_standard, pm25\_standard, pm100\_standard;

uint16\_t pm10\_env, pm25\_env, pm100\_env;

uint16\_t particles\_03um, particles\_05um, particles\_10um, particles\_25um, particles\_50um, particles\_100um;

uint16\_t unused;

uint16\_t checksum;

};

struct pms5003data data;

void loop() {

if (readPMSdata(&pmsSerial)) {

// reading data was successful!

Serial.println();

Serial.println("---------------------------------------");

Serial.println("Concentration Units (standard)");

Serial.print("PM 1.0: "); Serial.print(data.pm10\_standard);

Serial.print("\t\tPM 2.5: "); Serial.print(data.pm25\_standard);

Serial.print("\t\tPM 10: "); Serial.println(data.pm100\_standard);

Serial.println("---------------------------------------");

Serial.println("Concentration Units (environmental)");

Serial.print("PM 1.0: "); Serial.print(data.pm10\_env);

Serial.print("\t\tPM 2.5: "); Serial.print(data.pm25\_env);

Serial.print("\t\tPM 10: "); Serial.println(data.pm100\_env);

Serial.println("---------------------------------------");

Serial.print("Particles > 0.3um / 0.1L air:"); Serial.println(data.particles\_03um);

Serial.print("Particles > 0.5um / 0.1L air:"); Serial.println(data.particles\_05um);

Serial.print("Particles > 1.0um / 0.1L air:"); Serial.println(data.particles\_10um);

Serial.print("Particles > 2.5um / 0.1L air:"); Serial.println(data.particles\_25um);

Serial.print("Particles > 5.0um / 0.1L air:"); Serial.println(data.particles\_50um);

Serial.print("Particles > 10.0 um / 0.1L air:"); Serial.println(data.particles\_100um);

Serial.println("---------------------------------------");

}

}

boolean readPMSdata(Stream \*s) {

if (! s->available()) {

return false;

}

// Read a byte at a time until we get to the special '0x42' start-byte

if (s->peek() != 0x42) {

s->read();

return false;

}

// Now read all 32 bytes

if (s->available() < 32) {

return false;

}

uint8\_t buffer[32];

uint16\_t sum = 0;

s->readBytes(buffer, 32);

// get checksum ready

for (uint8\_t i=0; i<30; i++) {

sum += buffer[i];

}

/\* debugging

for (uint8\_t i=2; i<32; i++) {

Serial.print("0x"); Serial.print(buffer[i], HEX); Serial.print(", ");

}

Serial.println();

\*/

// The data comes in endian'd, this solves it so it works on all platforms

uint16\_t buffer\_u16[15];

for (uint8\_t i=0; i<15; i++) {

buffer\_u16[i] = buffer[2 + i\*2 + 1];

buffer\_u16[i] += (buffer[2 + i\*2] << 8);

}

// put it into a nice struct :)

memcpy((void \*)&data, (void \*)buffer\_u16, 30);

if (sum != data.checksum) {

Serial.println("Checksum failure");

return false;

}

// success!

return true;

}

**5. Arduino Uno code For PM10:**

// UNO version of PM10 Analyser

#include "SoftwareSerial.h"

SoftwareSerial mySerial(2, 3); // RX, TX for SDS011 sensor ( to keep Serial monitor available )

#include "U8glib.h"

U8GLIB\_SSD1306\_128X64 u8g(U8G\_I2C\_OPT\_NONE); // for 1306 type OLED, I2C / TWI

// Global Variables

static unsigned char buf[7], buffSDS[25];

unsigned int PM2\_5,PM10=0;

// Sub Routines

// Update OLED Display

void draw(void) {

/\* for the line with PM2.5 value \*/

if ( PM2\_5>999 ) PM2\_5=999 ;// overflow is 999

val\_to\_string(PM2\_5);

u8g.setFont(u8g\_font\_fub30);// Large font

u8g.drawStr( 0, 31, buf); //

u8g.setFont(u8g\_font\_unifont);

u8g.drawStr( 75, 10, "PM2.5");

buf[0]='µ';

buf[1] = '\0';

u8g.drawStr( 75, 10+2+10, buf);

u8g.drawStr( 82, 10+2+10, "g/m3");

// for the line with PM10 value

if ( PM10>999 ) PM10=999 ;// overflow

val\_to\_string(PM10);

u8g.setFont(u8g\_font\_fub30);// Large font

u8g.drawStr( 0, 65, buf); //

u8g.setFont(u8g\_font\_unifont);

u8g.drawStr( 75, 34+10, "PM10");

buf[0]='µ';

buf[1] = '\0';

u8g.drawStr( 75, 34+10+2+10, buf);

u8g.drawStr( 82, 34+10+2+10, "g/m3");

}

/\* convert int into buf[] to BCD string to be OLED printed \*/

void val\_to\_string(int val){

int deca[5];

deca[4]=10000;

deca[3]=1000;

deca[2]=100;

deca[1]=10;

deca[0]=1;

char digit[10];

digit[0]='0';

digit[1]='1';

digit[2]='2';

digit[3]='3';

digit[4]='4';

digit[5]='5';

digit[6]='6';

digit[7]='7';

digit[8]='8';

digit[9]='9';

buf[0]='0';

buf[1]='0';

buf[2]='0';

buf[3]='\0'; // string terminator, only 3 digits needed

buf[4]='0';

buf[5] = '\0'; // not used

for ( int8\_t i=2; i>=0 ; i=i-1 )

{

byte d=0;

while (( val-deca[i]) >= 0)

{ val=val-deca[i];

buf[2-i]=digit[++d];

}

}

}

void setup() {

// put your setup code here, to run once:

// init 1306 I2C OLED

u8g.setColorIndex(1); // monochrome

// Read SDS011 on Serial

mySerial.begin(9600); //

mySerial.setTimeout(200);

mySerial.readBytesUntil(0xAB,buffSDS,20); // read serial until 0xAB Char received

// Serial Monitor

Serial.begin(115200);

}

void loop() {

// put your main code here, to run repeatedly:

// LCD Update

u8g.firstPage();

do {

draw();

} while( u8g.nextPage() );

// Read SDS011

mySerial.readBytesUntil(0xAB,buffSDS,20);

// Serial monitor, print the HEX bytes received in buffSDS

//Serial.write(buffSDS,10);

for ( int8\_t i=0; i<10 ; i=i+1 )

{

Serial.print( buffSDS[i],HEX);

Serial.print(" ");

}

Serial.println("");

PM2\_5 = ((buffSDS[3]\*256)+buffSDS[2])/10; // extract PM2.5 value

Serial.print("PM2.5: ");

Serial.println(PM2\_5);

PM10 = ((buffSDS[5]\*256)+buffSDS[4])/10; // extract PM10 value

Serial.print("PM10: ");

Serial.println(PM10);

delay(500);

}

**6. Arduino Uno code For Wi-Fi Module(ESp8266):**

if(wifiSerial.available()>0){

String message = readWifiSerialMessage();

if(find(message,"esp8266:")){

String result = sendToWifi(message.substring(8,message.length()),responseTime,DEBUG);

if(find(result,"OK"))

sendData("\n"+result);

else

sendData("\nErrRead"); //At command ERROR CODE for Failed Executing statement

}else

if(find(message,"HELLO")){ //receives HELLO from wifi

sendData("\\nHI!"); //arduino says HI

}else if(find(message,"LEDON")){

digitalWrite(13,HIGH);

}else if(find(message,"LEDOFF")){

digitalWrite(13,LOW);

}

else{

sendData("\nErrRead"); //Command ERROR CODE for UNABLE TO READ

}

}

If you want to communicate with Arduino UNO or ask todo something, just add your condition. for example :

if(find(message,"MY CODE")){

// I found 'MY CODE' from received message

// lets do something here

}

if(find(message,"A")){

// I found 'A' from received message

// lets do something here

}