#define HumiditySensorPin A0

int getHumidityPercentage() {

int humidityPercentage;

int sensorValue = analogRead(HumiditySensorPin);

humidityPercentage = map(sensorValue, 0, 1023, 0, 100);

return humidityPercentage;

}

void setup() {

pinMode(HumiditySensorPin, INPUT);

Serial.begin(9600);

}

void loop() {

int humidityPercentage = getHumidityPercentage();

Serial.print("Humedad = ");

Serial.print(humidityPercentage);

Serial.println("%");

delay(1000);

}

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include <SimpleTimer.h>

SimpleTimer timer;

float calibration\_value = 21.34 - 0.7;

int phval = 0;

unsigned long int avgval;

int buffer\_arr[10],temp;

float ph\_act;

// for the OLED display

#define SCREEN\_WIDTH 128 // OLED display width, in pixels

#define SCREEN\_HEIGHT 64 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)

#define OLED\_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

void initialize()

{

Wire.begin();

Serial.begin(9600);

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

display.clearDisplay();

display.setTextColor(WHITE);

timer.setInterval(500L, display\_pHValue);

}

void measure\_pHValue()

{

timer.run(); // Initiates SimpleTimer

for(int i=0;i<10;i++)

{

buffer\_arr[i]=analogRead(A0);

delay(30);

}

for(int i=0;i<9;i++)

{

for(int j=i+1;j<10;j++)

{

if(buffer\_arr[i]>buffer\_arr[j])

{

temp=buffer\_arr[i];

buffer\_arr[i]=buffer\_arr[j];

buffer\_arr[j]=temp;

}

}

}

avgval=0;

for(int i=2;i<8;i++)

avgval+=buffer\_arr[i];

float volt=(float)avgval\*5.0/1024/6;

ph\_act = -5.70 \* volt + calibration\_value;

Serial.println("pH Val: ");

Serial.print(ph\_act);

delay(1000);

}

void display\_pHValue()

{

// display on Oled display

display.clearDisplay();

display.setTextSize(2);

display.setCursor(0,0); // column row

display.print("pH:");

display.setTextSize(2);

display.setCursor(55, 0);

display.print(ph\_act);

/\*

display.setTextSize(2);

display.setCursor(0,30);

display.print("EC:");

display.setTextSize(2);

display.setCursor(60, 30);

display.print(345);

display.setCursor(95, 50);

\*/

display.display();

}

void setup()

{

initialize();

}

void loop()

{

measure\_pHValue();

}

#define TdsSensorPin A0

#define VREF 5.0 // analog reference voltage(Volt) of the ADC

#define SCOUNT 30 // sum of sample point

int analogBuffer[SCOUNT]; // store the analog value in the array, read from ADC

int analogBufferTemp[SCOUNT];

int analogBufferIndex = 0;

int copyIndex = 0;

float averageVoltage = 0;

float tdsValue = 0;

float temperature = 25; // current temperature for compensation

// median filtering algorithm

int getMedianNum(int bArray[], int iFilterLen){

int bTab[iFilterLen];

for (byte i = 0; i<iFilterLen; i++)

bTab[i] = bArray[i];

int i, j, bTemp;

for (j = 0; j < iFilterLen - 1; j++) {

for (i = 0; i < iFilterLen - j - 1; i++) {

if (bTab[i] > bTab[i + 1]) {

bTemp = bTab[i];

bTab[i] = bTab[i + 1];

bTab[i + 1] = bTemp;

}

}

}

if ((iFilterLen & 1) > 0){

bTemp = bTab[(iFilterLen - 1) / 2];

}

else {

bTemp = (bTab[iFilterLen / 2] + bTab[iFilterLen / 2 - 1]) / 2;

}

return bTemp;

}

void setup(){

Serial.begin(115200);

pinMode(TdsSensorPin,INPUT);

}

void readTdsValue(){

analogBuffer[analogBufferIndex] = analogRead(TdsSensorPin); //read the analog value and store into the buffer

analogBufferIndex++;

if(analogBufferIndex == SCOUNT){

analogBufferIndex = 0;

}

for(copyIndex=0; copyIndex<SCOUNT; copyIndex++){

analogBufferTemp[copyIndex] = analogBuffer[copyIndex];

// read the analog value more stable by the median filtering algorithm, and convert to voltage value

averageVoltage = getMedianNum(analogBufferTemp,SCOUNT) \* (float)VREF / 4096.0;

//temperature compensation formula: fFinalResult(25^C) = fFinalResult(current)/(1.0+0.02\*(fTP-25.0));

float compensationCoefficient = 1.0+0.02\*(temperature-25.0);

//temperature compensation

float compensationVoltage=averageVoltage/compensationCoefficient;

//convert voltage value to tds value

tdsValue=(133.42\*compensationVoltage\*compensationVoltage\*compensationVoltage - 255.86\*compensationVoltage\*compensationVoltage + 857.39\*compensationVoltage)\*0.5;

Serial.print("voltage:");

Serial.print(averageVoltage,2);

Serial.print("V ");

Serial.print("TDS Value:");

Serial.print(tdsValue,0);

Serial.println("ppm");

}

}

void loop(){

static unsigned long analogSampleTimepoint = millis();

if(millis()-analogSampleTimepoint > 40U){ //every 40 milliseconds,read the analog value from the ADC

analogSampleTimepoint = millis();

readTdsValue();

}