```
/*
  BME280 I2C Test.ino

  This code operates a coulometric respirometer.
  Pressure is measured from a BME280 environmental sensor using I2C interface.
  If pressure drops below a threshold, current is set to an O2 generator.
  Temperature and humidity are also measured.
  Current is monitored across a 10 ohm resistor at A0.

 Data are sent to OLED display and serial port.

  GNU General Public License

  Written: Dec 30 2015.
  Last Updated: October 16, 2020.

  Connecting the BME280 Sensor:
  Sensor              ->  Board
  -----------------------------
  Vin (Voltage In)    ->  3.3V
  Gnd (Ground)        ->  Gnd
  SDA (Serial Data)   ->  A4 on Uno/Pro-Mini, 20 on Mega2560/Due, 2 Leonardo/Pro-Micro
  SCK (Serial Clock)  ->  A5 on Uno/Pro-Mini, 21 on Mega2560/Due, 3 Leonardo/Pro-Micro

*/

#include <BME280I2C.h>
#include <Wire.h>

// Include Adafruit Graphics & OLED libraries
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#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     4 // Reset pin # (or -1 if sharing Arduino reset pin)
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#define SERIAL_BAUD 9600

// GPIO pins definitions
// Relay set on pin #4
int oxygenOn = 4;


//analog in for corrent monitor at pin A0
float CurrentPin = A0;
float CurrentValue = 0;

//define time
unsigned long time;

// Threshold values
const float pressureOnThreshold = 1016; // hPa
const float pressureOffThreshold = 1017;  // hPa

BME280I2C bme;    // Default : forced mode, standby time = 1000 ms
// Oversampling = pressure ×1, temperature ×1, humidity ×1, filter off,

//////////////////////////////////////////////////////////////////
void setup()
{
  // initialize digital pins.

  pinMode(oxygenOn, OUTPUT);
  digitalWrite(oxygenOn, LOW);
  pinMode(LED_BUILTIN, OUTPUT);


  Serial.begin(SERIAL_BAUD);

  while (!Serial) {} //Wait

  Wire.begin();

  while (!bme.begin())

  {
    Serial.println("Could not find BME280 sensor!");
    delay(500);
  }


  // initialize OLED with I2C addr 0x3C
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);

}

//Get the OLED display to show values
void displayTempHumPres() {
  float CurrentValue;
   CurrentValue = analogRead(CurrentPin);

    // Read sensor
  float temp(NAN), hum(NAN), pres(NAN);

  BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);

  bme.read(pres, temp, hum, tempUnit, presUnit);

  // Clear the display
  display.clearDisplay();
  //Set the color - always use white despite actual display color
  display.setTextColor(WHITE);
  //Set the font size
  display.setTextSize(1);
  //Set the cursor coordinates and display: name of the controller box, Pressure (hPa), Humidity (%RH),
Temperature (C), current (mA)
  display.setCursor(0, 0);
  display.print("Sensor #1");
  display.setCursor(0, 12);
  display.print("Pressure:");
  display.print(pres);
  display.print(" hPa");
  display.setCursor(0, 24);
  display.print("Humidity:    ");
  display.print(hum);
  display.print(" %");
  display.setCursor(0, 36);
  display.print("Temperature: ");
  display.print(temp);
  display.print(" C");
   display.setCursor(0, 48);
  display.print("Current:   ");
  display.print(CurrentValue/2.05);
  display.print(" mA");
}
//////////////////////////////////////////////////////////////////

//Print to com4
void  printTempHumPres()
{

  // Read Sensor Output
  float temp(NAN), hum(NAN), pres(NAN);


  BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);

  bme.read(pres, temp, hum, tempUnit, presUnit);


  time = millis();  //sets time to milliseconds

  //Tag the sensor
  Serial.print ("#1");
  Serial.print(" , ");

  Serial.print(time);

  Serial.print(" , "); //for comma delimiting


  //Serial.print("Temperature = ");
  Serial.print(temp);
  //Serial.println(" *C");
  Serial.print(" , ");

  //Serial.print("Pressure = ");

  Serial.print(pres);
  //Serial.println(" hPa");

  Serial.print(" , ");
  // Serial.print("Humidity = ");
  Serial.print(hum);
  //Serial.println(" %");

  Serial.print(" , ");

  Serial.print(CurrentValue/2.05);//prints ADC input converted to V

  Serial.println(); //line break between samples
}

//read the sensor
void readTempHumPres()
{
  float temp(NAN), hum(NAN), pres(NAN);


  BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);

  bme.read(pres, temp, hum, tempUnit, presUnit);



}


// turn on the current
void turnOn()
{

  digitalWrite(oxygenOn, HIGH);
  digitalWrite(LED_BUILTIN, HIGH);
}

// turn off the current
void turnOff()


{ digitalWrite(oxygenOn, LOW);
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
}


void loop() {


  CurrentValue = analogRead(CurrentPin);




  displayTempHumPres();
  display.display();

  printTempHumPres();
  readTempHumPres();

  float temp(NAN), hum(NAN), pres(NAN);



  BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);

  bme.read(pres, temp, hum, tempUnit, presUnit);

  if (pres < pressureOnThreshold)  // turn O2 ON if if Temperature IS LOWER THAN THRESHOLD
  {
    turnOn();
  }
  if (pres > pressureOffThreshold)  // turn O2 OFF if PRESSURE IS HIGHER THAN THREHOLD
  {
    turnOff();
  }

  //Refreshes every 0.5 second
  delay(500);


}

//////////////////////////////////////////////////////////////////
```