mikrocontroller.net

Forum: Mikrocontroller und Digitale Elektronik

BME280 Erfahrungen

BME280 Erfahrungen

Autor: Alfred (Gast)
Datum: 04.01.2017 18:55

Schönen Tag allerseits!

Ich möchte hier kurz meine Erfahrungen mit dem BME280 von Bosch mitteilen. Um Freude zu haben, sollte man einige Punkte beachten:

1. Der gemessene Luftdruck (zuerst die Temperatur auslesen, sonst fehlt der notwendige Kalibrierwert!) stimmt erstens und zeigt zweitens den Luftdruck an dem Ort an, wo der Sensor sich tatsächlich befindet. Also bitte nicht im Zimmer messen, wenn man eine Wetterstation für draußen bauen will.

Der ausgegebene Wert für den Luftdruck entspricht aber jenem für die aktuelle Höhe und die aktuelle Temperatur. Das ev vorhandene Barometer zeigt aber wahrscheinlich den auf Meeresniveau "reduzierten" Wert (ist also höher als der vom BME280 angezeigte Wert).

Will man überprüfen, ob beide Geräte übereinstimmen, muß man den vom BME280 ausgegebenen Wert auf "Meeresniveau" umrechnen.

Das geht so:

pNN = ph * e (g*h/(R*(T+Ch*E+a*h/2)))

E = 6,112hPa * e (17,62*t(°C)/(243,12+t(°C))) ACHTUNG: Magnusformel mit Koeffizienten für t = -45 bis +60°C

a = (Tm-Tb)/(hm-hb) m ... Meßort b ... Bezugsort

- pNN ist der auf Meeresniveau umgerechnete Luftdruckwert in hPa
- ph ist der vom BME280 ausgegebene tatsächliche Luftdruckwert am Meßort
- in hPa
- e ist die e-Funktion
- g ist die Erdbeschleunigung, am 50. Breitengrad beträgt g=9,80665 m/s²
- h ist die Höhe (m) über Meeresniveau am Meßort
- R ist die Gaskonstante der Luft 287,058 J/(kg*K)
- t(°C) ist die Temperatur in °C am Meßort
- T ist die Temperatur in Kelvin T(K)=273,15+t(°C)
- Ch ist der Beiwert zu E zur Berücksichtigung der mittleren Dampfdruckänderung mit der Höhe (K/hPa)
- E (hPa) ist der Wasserdampfpartialdruck
- a (K/m) ist der vertikale Temperaturgradient

a ist oft unter Durchschnitts-Meeresniveau-Stardardbedingungen (t=15°C, pb=1013.25hPa, h=0m) mit 0.0065 angegeben - stimmt jedoch nicht wirklich, da durchaus in der Realität beträchtliche Abweichungen auftreten können >> also: besser selber berechnen!

Zur Überprüfung kann man den Meßwert des BME280 mit dem aus der Barometrischen Höhenformel ausgerechneten Wert vergleichen:

pz=pb*((1-a*(hz-hb)/(273,15+tb))^(0,03416/a))

a wird wie weiter oben berechnet, z bedeutet Werte am Zielort, b sind

BEKANNTE Werte am Bezugsort (von einer offiziellen, naheliegenden meteorologischen Station abzufragen)

2. die gemessene Temperatur gibt lediglich einen Anhalt über die Umgebungstemperatur des BME280 (siehe Datenblatt). Wer genaue Werte für die Lufttemperatur haben will, muß schon einen externen Sensor (zB DS18B20) bemühen.

3. man findet in Softwarebibliotheken (va Arduino-Bibliotheken - Adafruit weist jedoch darauf hin!) häufig Funktionen, welche die Seehöhe=Höhe über Meeresniveau, berechnet aus den BME280-Meßwerten, ausgeben. Das ist leider völlig falsch!! Ohne einen genau bekannten Referenzpunkt (p,T auf Seehöhe oder besser einen naheliegenden Ort plus p,T am Zielort) kann die Höhe am Zielort aus den Sensorwerten NICHT ermittelt werden.

4. man kann weiters auch die Absolute Luftfeuchte

aF(kg/m³)=rF*E/(Rw*T)*10

mit Rw=461,51 J/(kg*K) und rF (relative Feuchte) als Quotient (nicht als%-Zahl!)

und den Taupunkt (wieder Magnusformel für t=-45 bis +60°C heranziehen)

$\tau(\phi,\vartheta) = K3 \cdot ((K2 \cdot \vartheta)/(K3 + \vartheta) + \ln \phi)/(K2 \cdot K3/(K3 + \vartheta) - \ln\phi)$

ϕ: relative Luftfeuchte (Quotient) ϑ: Temperatur in °C
K1=6.112hPa
K2=17,62
K3=243,12K

aus den Meßwerten herleiten (Details bitte aus dem Internet entnehmen).

Re: BME280 Erfahrungen

Autor: Alfred (Gast)
Datum: 04.01.2017 18:57

Nachtrag: Ch hat einen Wert von 0.12 K/hPa

Re: BME280 Erfahrungen

Autor: Juergen P. (optronik)
Datum: 04.01.2017 19:16

Danke für diese Informationen, gerade wenn man sich mit der Materie nicht so beschäftigt und nur den Druck messen will kann man schnell mal den Offset von NN übersehen.

Re: BME280 Erfahrungen

Autor: daVinciClaude (Gast)
Datum: 05.01.2017 20:08

Wow. Vielen Dank für die Infos. Habe auch ein Projekt mit dem Bosch Sensor in der Pipeline. Werde Deine Infos auf jeden Fall noch genauer studieren.

LG
Claude

Re: BME280 Erfahrungen

Autor: Gerhard G. (xmega)
Datum: 06.01.2017 13:24

Hallo,

schaut sehr professionell aus. Jetzt ist die normale C-Routine für dem BME280 schon aufwendig. Nun kommt nochmals eine mathematische Routine dazu.
Schafft das ein normaler Mikrocontroller?

Zeige mal Deine C-Routine als Beispiel.
Ist der Mehraufwand gerechtfertigt?

Gruß G.G.

Re: BME280 Erfahrungen

Autor: Gerd E. (robberknight)
Datum: 06.01.2017 13:38

In diesem Vergleichstest hat der BME280 zumindest für die Luftfeuchtigkeit sehr gut abgeschnitten:
http://www.kandrsmith.org/RJS/Misc/Hygrometers/cal...

Re: BME280 Erfahrungen

Autor: Alfred (Gast)
Datum: 06.01.2017 20:26

Hallo Gerhard G.!

Anbei ein rasch zusammengezimmertes Beispiel für obige Überlegungen mit der Sparkfun-Bibliothek und dem umgebauten Sparkfun-Beispiel:

```
**************************************************************************
Board:  Arduino UNO
Sensor: BME280

ACHTUNG: bei der Berechnung der Seehöhe muß man den aktuellen Luftdruckwert (Pa)
         auf Meeresniveau als Parameter übergeben >> sonst kommt Mist heraus
**************************************************************************/

#include <stdint.h>
#include "SparkFunBME280.h"
//Library allows either I2C or SPI, so include both.
#include "Wire.h"
#include "SPI.h"
#include <math.h>

#define a 0.00602      // ACHTUNG: 0.0065 gilt nur für Standardbedingungen h=0m T=15°C p=1013,25hPa
                       // besser aktuell berechnen: a= (Tz-Tb)/(hz-hb)   z=Zielpunkt b=Bezugspunkt (kann Ort am Meer sein)
```

```
#define pSL 100800.0

//Global sensor object
BME280 mySensor;

float La, RH, TP, E, H, T, pNN, pH;

void setup()
{
  //***Driver settings********************************//
  //commInterface can be I2C_MODE or SPI_MODE
  //specify chipSelectPin using arduino pin names
  //specify I2C address.  Can be 0x77(default) or 0x76

  //For I2C, enable the following and disable the SPI section
  //mySensor.settings.commInterface = I2C_MODE;
  //mySensor.settings.I2CAddress = 0x77;

  //For SPI enable the following and dissable the I2C section
  mySensor.settings.commInterface = SPI_MODE;
  mySensor.settings.chipSelectPin = 10;


  //***Operation settings*****************************//

  //renMode can be:
  //  0, Sleep mode
  //  1 or 2, Forced mode
  //  3, Normal mode
  mySensor.settings.runMode = 3; //Normal mode

  //tStandby can be:
  //  0, 0.5ms
  //  1, 62.5ms
  //  2, 125ms
  //  3, 250ms
  //  4, 500ms
  //  5, 1000ms
  //  6, 10ms
  //  7, 20ms
  mySensor.settings.tStandby = 0;

  //filter can be off or number of FIR coefficients to use:
  //  0, filter off
  //  1, coefficients = 2
  //  2, coefficients = 4
  //  3, coefficients = 8
  //  4, coefficients = 16
  mySensor.settings.filter = 0;

  //tempOverSample can be:
  //  0, skipped
  //  1 through 5, oversampling *1, *2, *4, *8, *16 respectively
  mySensor.settings.tempOverSample = 1;

  //pressOverSample can be:
  //  0, skipped
  //  1 through 5, oversampling *1, *2, *4, *8, *16 respectively
    mySensor.settings.pressOverSample = 1;

  //humidOverSample can be:
  //  0, skipped
  //  1 through 5, oversampling *1, *2, *4, *8, *16 respectively
  mySensor.settings.humidOverSample = 1;
```

```
  Serial.begin(57600);
  Serial.print("Program Started\n");
  Serial.print("Starting BME280... result of .begin(): 0x");

  //Calling .begin() causes the settings to be loaded
  delay(10);  //Make sure sensor had enough time to turn on. BME280 requires 2ms to start
up.
  Serial.println(mySensor.begin(), HEX);

  Serial.print("Displaying ID, reset and ctrl regs\n");

  Serial.print("ID(0xD0): 0x");
  Serial.println(mySensor.readRegister(BME280_CHIP_ID_REG), HEX);
  Serial.print("Reset register(0xE0): 0x");
  Serial.println(mySensor.readRegister(BME280_RST_REG), HEX);
  Serial.print("ctrl_meas(0xF4): 0x");
  Serial.println(mySensor.readRegister(BME280_CTRL_MEAS_REG), HEX);
  Serial.print("ctrl_hum(0xF2): 0x");
  Serial.println(mySensor.readRegister(BME280_CTRL_HUMIDITY_REG), HEX);

  Serial.print("\n\n");

  Serial.print("Displaying all regs\n");
  uint8_t memCounter = 0x80;
  uint8_t tempReadData;
  for(int rowi = 8; rowi < 16; rowi++ )
  {
    Serial.print("0x");
    Serial.print(rowi, HEX);
    Serial.print("0:");
    for(int coli = 0; coli < 16; coli++ )
    {
      tempReadData = mySensor.readRegister(memCounter);
      Serial.print((tempReadData >> 4) & 0x0F, HEX);//Print first hex nibble
      Serial.print(tempReadData & 0x0F, HEX);//Print second hex nibble
      Serial.print(" ");
      memCounter++;
    }
    Serial.print("\n");
  }


  Serial.print("\n\n");

  Serial.print("Displaying concatenated calibration words\n");
  Serial.print("dig_T1, uint16: ");
  Serial.println(mySensor.calibration.dig_T1);
  Serial.print("dig_T2, int16: ");
  Serial.println(mySensor.calibration.dig_T2);
  Serial.print("dig_T3, int16: ");
  Serial.println(mySensor.calibration.dig_T3);

  Serial.print("dig_P1, uint16: ");
  Serial.println(mySensor.calibration.dig_P1);
  Serial.print("dig_P2, int16: ");
  Serial.println(mySensor.calibration.dig_P2);
  Serial.print("dig_P3, int16: ");
  Serial.println(mySensor.calibration.dig_P3);
  Serial.print("dig_P4, int16: ");
  Serial.println(mySensor.calibration.dig_P4);
  Serial.print("dig_P5, int16: ");
  Serial.println(mySensor.calibration.dig_P5);
  Serial.print("dig_P6, int16: ");
  Serial.println(mySensor.calibration.dig_P6);
```

```
  Serial.print("dig_P7, int16: ");
  Serial.println(mySensor.calibration.dig_P7);
  Serial.print("dig_P8, int16: ");
  Serial.println(mySensor.calibration.dig_P8);
  Serial.print("dig_P9, int16: ");
  Serial.println(mySensor.calibration.dig_P9);

  Serial.print("dig_H1, uint8: ");
  Serial.println(mySensor.calibration.dig_H1);
  Serial.print("dig_H2, int16: ");
  Serial.println(mySensor.calibration.dig_H2);
  Serial.print("dig_H3, uint8: ");
  Serial.println(mySensor.calibration.dig_H3);
  Serial.print("dig_H4, int16: ");
  Serial.println(mySensor.calibration.dig_H4);
  Serial.print("dig_H5, int16: ");
  Serial.println(mySensor.calibration.dig_H5);
  Serial.print("dig_H6, uint8: ");
  Serial.println(mySensor.calibration.dig_H6);

  Serial.println();
  delay(5000);
}

void loop()
{
  //Each loop, take a reading.
  //Start with temperature, as that data is needed for accurate compensation.
  //Reading the temperature updates the compensators of the other functions
  //in the background.

  T=mySensor.readTempC();
  Serial.print("Temperatur: ");
  Serial.print(T,2);
  Serial.println(" °C");

  /*Serial.print("Temperature: ");
  Serial.print(mySensor.readTempF(), 2);
  Serial.println(" degrees F");*/

  pH=mySensor.readFloatPressure()/100.0;
  Serial.print("Luftdruck vor Ort: ");
  Serial.print(pH, 2);
  Serial.println(" hPa");

  H=mySensor.readFloatAltitudeMeters(pSL);

  E=6.112*exp((17.62*T)/(243.12+T));
  pNN=pH*exp(9.80665*H/(287.058*((273.15+T)+0.12*E+a*H/2)));

  Serial.print("Luftdruck reduziert: ");
  Serial.print(pNN, 2);
  Serial.println(" hPa");

  Serial.print("Höhe über Meer: ");
  Serial.print(H, 2);
  Serial.println("m");

  /*Serial.print("Altitude: ");
  Serial.print(mySensor.readFloatAltitudeFeet(), 2);
  Serial.println("ft"); */

  RH=mySensor.readFloatHumidity();
  Serial.print("Relative Luftfeuchte: ");
  Serial.print(RH, 2);
```

```
  Serial.println(" %");

  La=RH*E/(461.51*(273.15+T))*10000.0;

  Serial.print("Absolute Luftfeuchte: ");
  Serial.print(La, 2);
  Serial.println(" g/m³");

  TP=243.12*((17.62*T)/(243.12+T)+log(RH/100.0))/((17.62*243.12)/(243.12+T)-log(RH/100.0));
  Serial.print("Taupunkt: ");
  Serial.print(TP, 2);
  Serial.println(" °C");

  Serial.println();

  delay(5000);

}
```

Verwendet wurde ein herkömmliches chinesisches Arduino-UNO-board.

Die IDE gibt beim Kompilieren folgendes aus:

"Sketch uses 12746 bytes (39%) of program storage space. Maximum is
32256 bytes. Global variables use 1171 bytes (57%) of dynamic memory,
leaving 877 bytes for local variables. Maximum is 2048 bytes."

Bei dem völlig un-optimierten Code scheint mir der Speicherbedarf
eigentlich nicht zu groß zu sein.

Re: BME280 Erfahrungen

Autor: Gerhard G. (xmega)
Datum: 06.01.2017 22:02

Hallo Alfred,

danke für die rasche Antwort. Werde mich die nächsten Tage mit der
Effizienz des Codes beschäftigen.

Gruß G.G.

Re: BME280 Erfahrungen

Autor: Hanspeter Roth (Gast)
Datum: 16.09.2018 19:42

readFloatAltitudeMeters()
Use to get altitude in units of meters, as a float. This function
calculates based off the measured pressure. **Takes no Arguments.**

But in the Example You Need:
#define pSL 100800.0

[H=mySensor.readFloatAltitudeMeters(pSL);]

```
and the result is:

LCD_PressureTemperature_Ro:246:42: error: no matching function for call
to 'BME280::readFloatAltitudeMeters(double)'

    H=mySensor.readFloatAltitudeMeters(pSL);


Can You help me? (I'm german language)
```

Re: BME280 Erfahrungen

Autor: my2ct (Gast)
Datum: 16.09.2018 20:15

```
Alfred schrieb:
> 1. Der gemessene Luftdruck (zuerst die Temperatur auslesen, sonst fehlt
> der notwendige Kalibrierwert!) stimmt erstens und zeigt zweitens den
> Luftdruck an dem Ort an, wo der Sensor sich tatsächlich befindet. Also
> bitte nicht im Zimmer messen, wenn man eine Wetterstation für draußen
> bauen will.

Ob der Luftdruck im Zimmer oder draußen gemessen wird, ist nun wirklich
schnuppe, solange man in einem normalen Haus ohne aktive Lüftung wohnt.
```