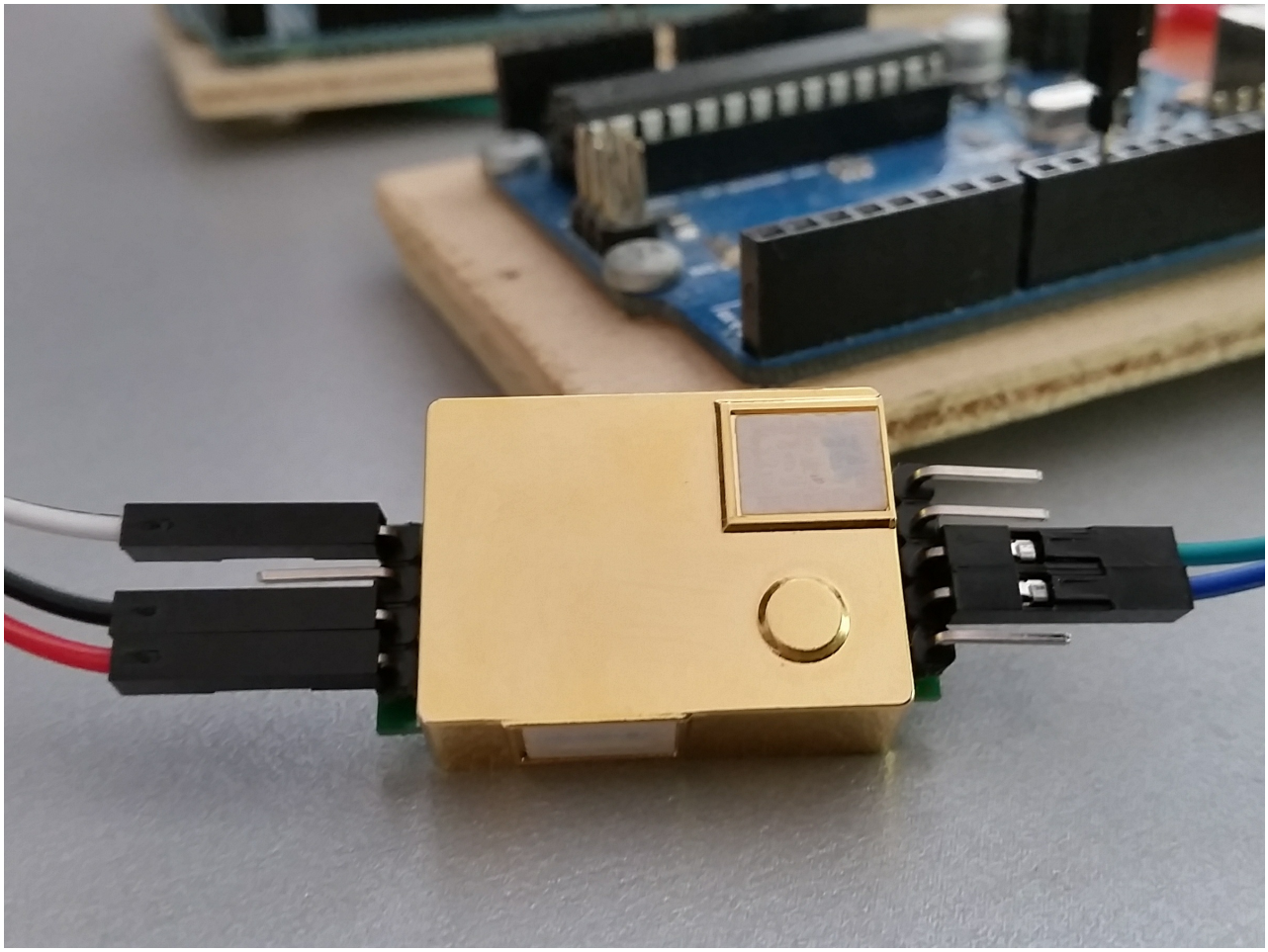
PitMH-Z19B - CO2.

L'utilizzo del sensore ad infrarossi PitMH-Z19B NDIR (Non Dispersive InfraRed) per la misura della concentrazione di CO2.



Settaggi e impostazioni:

I settaggi sono documentati in modo non del tutto corretto sul datasheet ufficiale. E' quindi opportuno vedere sul sito RevSpace il loro corretto funzionamento.

In particolare si pone l'attenzione sul comando 0x99: Sensor detection range setting che diversamente da quanto indicato nel datasheet va utilizzato scrivendo i dati nelle posizioni 6 e 7 del comando.

Molto interessante relativamente al significato di alcuni byte della risposta ai comandi è quanto riportato sul sito habr che però è in russo (con google traduttore si riesce a intuire in contenuto).

Funzionamento dell'ABC:

Una cosa da capire è il corretto funzionamento del settaggio relativo all'Automatic Background Calibration (ABC). Questo è settato a on per impostazioni di fabbrica e in

relazione al fatto che si tratta di un sensore da interno lavora su un periodo di controllo di 24 ore.

Questa impostazione presuppone che nell'ambiente dove il sensore è posizionato la concentrazione di CO2 si modifichi durante le 24 ore e torni al valore di calibrazione nell'arco di questo periodo, consentendo l'autocalibrazione del tutto. In pratica il valore minimo nelle 24 ore viene fatto corrispondere a 400 ppm (per info vedere co2meter.com).

Se il sensore rimane in un ambiente con concetrazioni più elevate (caso tipico dell'ambiente domestico) nelle 24 ore il sistema si ritara facendo corrispondere a quel minimo un valore sbagliato e occorre una nuova calibrazione.

Se il sensore opera in queste condizioni è opportuno disabilitare l'ABC ponendolo a off. Ovviente prima di farlo è necessario procedere alla ricalibrazione su valori corretti.

Procedura di calibrazione:

Si riporta di seguito la procedura utilizzata che sembra aver dato risultati corretti.

Utilizzando lo sketch di seguito, attivare l'ABC sul sensore togliendo il commendo corrispondente alla linea (#define CO2_0x79_ABCon) - in questo modo essendo viene attivata (compilazione condizionale) la relativa sezione del codice viene compilata e l'ABC risulta attivo.

Posizionare il sensore in un ambiente esterno (con aria pulita) protetto dall'illuminazione diretta (meglio se al buio - ma almeno all'ombra). Avviare Arduino e lasciarlo lavorare per oltre 24 ore.

Il sensore procederà alla ricalibrazione facendo coincidere il minimo delle 24 ore (a ottobre 2018 vale circa 420) a 400 ppm.

Dopo 24 ore riportare il tutto all'interno e disabilitare l'autocalibrazione commentando la linea di prima (//#define CO2_0x79_ABCon) e togliendo il commento a quella successiva (#define CO2_0x79_ABCoff). Utilizzare il sensore regolarmente. Qualora si notassero degli errori procedere nuovamente alla ricalibrazione.

Consiglio :

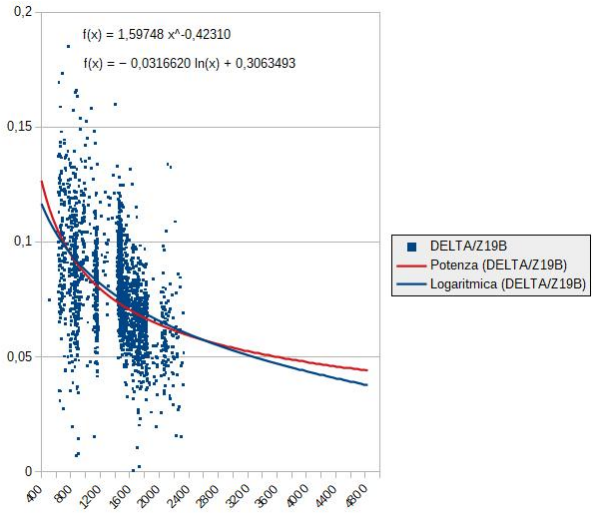
Dopo aver confrontato i risultati di due sensori, il primo calibrato come sopra descritto, il secondo con le impostazioni di fabbrica si consiglia di disabilitare l'autocalibrazione fin dall'inizio. In questo modo il sensore rimane impostato con i dati di fabbrica e non dovrebbe necessitare correzioni. Qualora il sensore si fosse starato eseguire la procedura precedentemente descritta ed eventualmente adottare una procedura di correzione.

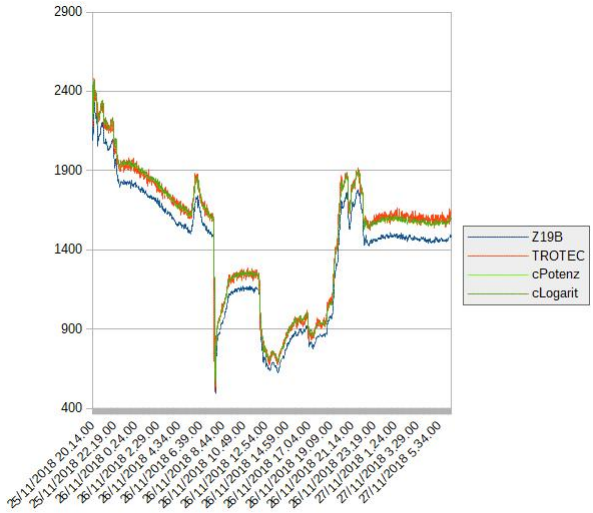
Procedura empirica di correzione :

Dopo aver calibrato il sensore su Arduino Mega è stato montato un RTC1307 e una scheda SD. E' stato caricato lo sketch PitCLAQ - nella configurazione datalogger. Il sensore è stato posizionato a fianco di un datalogger commerciale Trotec DL200L (che monta un sensore EE893). I due datalogger sono stati mantenuti (all'interno di una stanza) accesi a misurare ogni 60 secondi per oltre 24 ore e sono stati fatti i grafici e i confronti tra i dati rilevati.

Si è quindi ricercata la relazione tra risultati al fine di far coincidere i valori delle due rilevazioni. Per fare questo si assume che il datalogger TROTEC misuri il valore reale della CO2 nell'ambiente e che il dato dell'MH-Z19B debba essere corretto (nb.: entrambi i sensori dichiarano accuratezze simili dell'ordine di +/- (50 ppm+3%del valore letto)).

Si sono ricavate le funzioni che approssimano il rapporto tra la differenza dei due dati rispetto alla misura dell'MH-Z19B. Nello specifico si sono utilizzate una funzione di potenza e una funzione logaritmica. Come si nota questa seconda sembra essere la migliore.





Nell'utilizzo del sensore che ora è possibile utilizzare con una ragionevole conoscenza e approssimazione, si procederà alla correzione del dato utilizzando questa seconda curva. Nei log del sistema verranno registrati comunque anche i valori letti prima della correzione.

Cosa serve:

- MH-Z19B
- cavetti

Codice sorgente per l'uso sia seriale che PWM:

```
/*
 * Zappoco       - MH_Z19B   16/11/2018
```

```
 *
 * Nei confronti è stato utilizzato un Datalogger TROTEC DL200L questo monta un sensore EE893 che è stato utilizzato per alcuni confronti
 * ------------------------------------------------------
 * https://revspace.nl/MHZ19
 *
 *    Experimentation
 *
 *
 *    Normally, this sensor is read out using a command/response sequence over serial (9600,8N1). The following commands
 *    are known from the datasheet:
 *
 *     0x86: gas concentration reading
 *     0x87: calibrate zero point, I advise to AVOID SENDING THIS COMMAND, unless you know exactly what you are doing
 *     0x88: calibrate span point, I advise to AVOID SENDING THIS COMMAND, unless you know exactly what you are doing
 *
 *     The MH-Z19B datasheet additionally mentions to following commands:
 *
 *     0x79: ABC logic on/off (ABC = automatic baseline correction)
 *     0x99: Sensor detection range setting, this command can be used to set the measurement range (e.g. 0-2000ppm or 0-5000ppm)
 *
 *     It appears that these commands work on the MH-Z19 too, although with a little different command layout.
 *     Specifically, the range for command 0x99 is not put into bytes 3/4 but in bytes 6/7 of the command.
 *     This paragraph describes some experiments done to discover other commands than the ones mentioned in the datasheet.
 *
 * ------------------------------------------------------
 *
 *   This plug in is written by Dmitry (rel22 ___ inbox.ru)
 *   Plugin is based upon SenseAir plugin by Daniel Tedenljung info__AT__tedenljungconsulting.com
 *   Additional features based on https://geektimes.ru/post/285572/ by Gerben (infernix__AT__gmail.com)
 *
 * // 9-bytes CMD PPM read command
     byte mhzCmdReadPPM[9]       = {0xFF,0x01,0x86,0x00,0x00,0x00,0x00,0x00,0x79};
     byte mhzCmdCalibrateZero[9] = {0xFF,0x01,0x87,0x00,0x00,0x00,0x00,0x00,0x78};
     byte mhzCmdABCEnable[9]     = {0xFF,0x01,0x79,0xA0,0x00,0x00,0x00,0x00,0xE6};
     byte mhzCmdABCDisable[9]    = {0xFF,0x01,0x79,0x00,0x00,0x00,0x00,0x00,0x86};
     byte mhzCmdReset[9]         = {0xFF,0x01,0x8d,0x00,0x00,0x00,0x00,0x00,0x72};
     byte mhzCmdMeasurementRange1000[9] = {0xFF,0x01,0x99,0x00,0x00,0x00,0x03,0xE8,0x7B};
     byte mhzCmdMeasurementRange2000[9] = {0xFF,0x01,0x99,0x00,0x00,0x00,0x07,0xD0,0x8F};
     byte mhzCmdMeasurementRange3000[9] = {0xFF,0x01,0x99,0x00,0x00,0x00,0x0B,0xB8,0xA3};
     byte mhzCmdMeasurementRange5000[9] = {0xFF,0x01,0x99,0x00,0x00,0x00,0x13,0x88,0xCB};
```

```
 */
#include "SoftwareSerial.h"

#define pwmPin 10
//SoftwareSerial mySerial(A0, A1); // RX, TX
SoftwareSerial mySerial(A8, A9); // RX, TX Mega

//#define CO2_0x8D_reset
#define CO2_0x79_ABCon
//#define CO2_0x79_ABCoff

//#define CO2_0x99_setting2000
//#define CO2_0x99_setting5000

byte cmd[9] = {0xFF,0x01,0x86,0x00,0x00,0x00,0x00,0x00,0x79};
unsigned char response[9];
unsigned long th, tl,CO2_ppm, CO2_ppm2, CO2_ppm5, Temperatura, sts= 0;

void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);

  pinMode(pwmPin, INPUT);

  // calcola il checksum
  char i, checksum;

  #ifdef CO2_0x8D_reset

    cmd[2] = 0x8D; // 0x99 - Sensor detection range setting
    cmd[3] = 0x00;
    cmd[4] = 0x00;
    cmd[5] = 0x00;
    cmd[6] = 0x00;
    cmd[7] = 0x00;

    cmd[8] = 0x72;

    Serial.println("CO2_0x8D_reset");

    checksum = 0;
    for(i=1;i<8;i++)
    {
      checksum += cmd[i];
      //Serial.print("checksum p : 0x");Serial.println(checksum,HEX);
    }
    checksum = 0xff - checksum;
    checksum +=1;
    Serial.print("checksum   : 0x");Serial.println(checksum,HEX);

    mySerial.write(cmd,9);
    mySerial.readBytes(response, 9);

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

```
  Serial.println ("byte            : 00 01 02 03 04 05 06 07 08");
  Serial.print   ("cmd             :");
  for (int ico=0; ico<9 ;ico++ )
  {
    Serial.print (" ");
    if(cmd[ico] < 16)
      Serial.print ("0");
    Serial.print (cmd[ico],HEX);
  }
  Serial.println("");
  Serial.print   ("response        :");
  for (int ico=0; ico<9 ;ico++ )
    {
      //Serial.print ("%x", (unsigned char) response[ico]);
      Serial.print (" ");
      if(response[ico] < 16)
        Serial.print ("0");
      Serial.print (response[ico],HEX);
    }
    Serial.println("");

#endif



#ifdef CO2_0x79_ABCon

  cmd[2] = 0x79;
  cmd[3] = 0xA0; //ABC on
  cmd[4] = 0x00;
  cmd[5] = 0x00;
  cmd[6] = 0x00;
  cmd[7] = 0x00;
  cmd[8] = 0xE6;


  Serial.println("CO2_0x79_ABCon");
  checksum = 0;
  for(i=1;i<8;i++)
  {
    checksum += cmd[i];
    //Serial.print("checksum p : 0x");Serial.println(checksum,HEX);
  }
  checksum = 0xff - checksum;
  checksum +=1;
  Serial.print("checksum  : 0x");Serial.println(checksum,HEX);

  mySerial.write(cmd,9);
  mySerial.readBytes(response, 9);

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

  Serial.println ("byte            : 00 01 02 03 04 05 06 07 08");
```

```
  Serial.print   ("cmd             :");
  for (int ico=0; ico<9 ;ico++ )
  {
    //int valore = cmd[ico];
    //Serial.print ("%x", (unsigned char) response[ico]);
    Serial.print (" ");
    if(cmd[ico] < 16)
      Serial.print ("0");
    Serial.print (cmd[ico],HEX);
  }
  Serial.println("");
  Serial.print   ("response        :");
  for (int ico=0; ico<9 ;ico++ )
    {
      //Serial.print ("%x", (unsigned char) response[ico]);
      Serial.print (" ");
      if(response[ico] < 16)
        Serial.print ("0");
      Serial.print (response[ico],HEX);
    }
    Serial.println("");

#endif


#ifdef CO2_0x79_ABCoff

  cmd[2] = 0x79;
  cmd[3] = 0x00; //ABC off
  cmd[4] = 0x00;
  cmd[5] = 0x00;
  cmd[6] = 0x00;
  cmd[7] = 0x00;
  cmd[8] = 0x86; // 0x79 + 1 = 0x7A --> 0xFF-0xFA = 0x85 + 1 = 0x86


  Serial.println("CO2_0x79_ABCoff");
  checksum = 0;
  for(i=1;i<8;i++)
  {
    checksum += cmd[i];
    //Serial.print("checksum p : 0x");Serial.println(checksum,HEX);
  }
  checksum = 0xff - checksum;
  checksum +=1;
  Serial.print("checksum  : 0x");Serial.println(checksum,HEX);

  mySerial.write(cmd,9);
  mySerial.readBytes(response, 9);

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

  Serial.println ("byte             : 00 01 02 03 04 05 06 07 08");
  Serial.print   ("cmd              :");
```

```
  for (int ico=0; ico<9 ;ico++ )
  {
   //int valore = cmd[ico];
   //Serial.print ("%x", (unsigned char) response[ico]);
   Serial.print (" ");
   if(cmd[ico] < 16)
     Serial.print ("0");
   Serial.print (cmd[ico],HEX);
  }
  Serial.println("");
  Serial.print   ("response        :");
  for (int ico=0; ico<9 ;ico++ )
    {
     //Serial.print ("%x", (unsigned char) response[ico]);
     Serial.print (" ");
     if(response[ico] < 16)
       Serial.print ("0");
     Serial.print (response[ico],HEX);
    }
    Serial.println("");

#endif

#ifdef CO2_0x99_setting2000

  cmd[2] = 0x99; //0x99 - Sensor detection range setting
  cmd[3] = 0x00;
  cmd[4] = 0x00;
  cmd[5] = 0x00;
  cmd[6] = 0x00;
  cmd[7] = 0x00;

  cmd[6] = 0x07; // --> 2000/256;
  cmd[7] = 0xD0; // --> 2000%256;
  cmd[8] = 0x8F;

  Serial.println("CO2_0x99_setting2000");

  checksum = 0;
  for(i=1;i<8;i++)
  {
   checksum += cmd[i];
   //Serial.print("checksum p : 0x");Serial.println(checksum,HEX);
  }
  checksum = 0xff - checksum;
  checksum +=1;
  Serial.print("checksum   : 0x");Serial.println(checksum,HEX);

  mySerial.write(cmd,9);
  mySerial.readBytes(response, 9);

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

  Serial.println ("byte             : 00 01 02 03 04 05 06 07 08");
```

```
  Serial.print   ("cmd             :");
  for (int ico=0; ico<9 ;ico++ )
  {
   Serial.print (" ");
   if(cmd[ico] < 16)
     Serial.print ("0");
   Serial.print (cmd[ico],HEX);
  }
  Serial.println("");
  Serial.print   ("response        :");
  for (int ico=0; ico<9 ;ico++ )
    {
     //Serial.print ("%x", (unsigned char) response[ico]);
     Serial.print (" ");
     if(response[ico] < 16)
       Serial.print ("0");
     Serial.print (response[ico],HEX);
    }
    Serial.println("");
#endif

#ifdef CO2_0x99_setting5000

  cmd[2] = 0x99; // 0x99 - Sensor detection range setting
  cmd[3] = 0x00;
  cmd[4] = 0x00;
  cmd[5] = 0x00;
  cmd[6] = 0x00;
  cmd[7] = 0x00;

  cmd[6] = 0x13; // --> 5000/256
  cmd[7] = 0x88; // --> 5000%256
  cmd[8] = 0xCB;

  Serial.println("CO2_0x99_setting5000");
  checksum = 0;
  for(i=1;i<8;i++)
  {
   checksum += cmd[i];
   //Serial.print("checksum p : 0x");Serial.println(checksum,HEX);
  }
  checksum = 0xff - checksum;
  checksum +=1;
  Serial.print("checksum   : 0x");Serial.println(checksum,HEX);

  mySerial.write(cmd,9);
  mySerial.readBytes(response, 9);

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

  Serial.println ("byte             : 00 01 02 03 04 05 06 07 08");
  Serial.print   ("cmd              :");
  for (int ico=0; ico<9 ;ico++ )
  {
```

```
      Serial.print (" ");
      if(cmd[ico] < 16)
        Serial.print ("0");
      Serial.print (cmd[ico],HEX);
    }
    Serial.println("");
    Serial.print   ("response         :");
    for (int ico=0; ico<9 ;ico++ )
      {
        //Serial.print ("%x", (unsigned char) response[ico]);
        Serial.print (" ");
        if(response[ico] < 16)
          Serial.print ("0");
        Serial.print (response[ico],HEX);
      }
      Serial.println("");
  #endif




  delay(5000);
  cmd[2] = 0x86;//0x86 - Read CO2 concentration

  cmd[3] = 0x00;
  cmd[4] = 0x00;
  cmd[5] = 0x00;
  cmd[6] = 0x00;
  cmd[7] = 0x00;

  cmd[8] = 0x79;

}

void loop() {

  mySerial.write(cmd,9);
  mySerial.readBytes(response, 9);

  //Serial1.write(cmd,9);
  //Serial1.readBytes(response, 9);

  unsigned int responseHigh = (unsigned int) response[2];
  unsigned int responseLow = (unsigned int) response[3];
  CO2_ppm = (256*responseHigh)+responseLow;
  Temperatura = response[4] - 40; // alcuni fanno -44 - non documentato sul datasheet
  sts = response[5]; // status
  unsigned int uHigh = (unsigned int) response[6]; //forse umidità H - non documentato sul
datasheet
  unsigned int uLow  = (unsigned int) response[7];  //forse umidità L - non documentato sul
datasheet
  unsigned int chck  = (unsigned int) response[8];   //checksum
```

```
 Serial.println("----------------------------------------------------");
 Serial.print("CMD Time : ");Serial.println (millis());
 Serial.println ("byte             : 00 01 02 03 04 05 06 07 08");
 Serial.print   ("cmd              :");
 for (int ico=0; ico<9 ;ico++ )
 {
  Serial.print (" ");
  if(cmd[ico] < 16)
    Serial.print ("0");
  Serial.print (cmd[ico],HEX);
 }
 Serial.println("");



 Serial.println ("signific         : FF CM HH LL TT SS Uh Ul CS");
 Serial.print   ("response         :");
 for (int ico=0; ico<9 ;ico++ )
   {
    //Serial.print ("%x", (unsigned char) response[ico]);
    Serial.print (" ");
    if(response[ico] < 16)
      Serial.print ("0");
    Serial.print (response[ico],HEX);
   }
  Serial.println("");


 //CO2 via pwm
 do {
  th = pulseIn(pwmPin, HIGH, 1004000) / 1000;
  tl = 1004 - th;
  CO2_ppm2 = 2000 * (th-2)/(th+tl-4);
  CO2_ppm5 = 5000 * (th-2)/(th+tl-4);
 } while (th == 0);


 //Serial.print(0.0);
 Serial.println("");
 //Serial.print(",");
 Serial.print("CO2_ppm serial   : ");Serial.print(CO2_ppm);
 Serial.println("");
 Serial.print("th               : ");Serial.println(th);
 Serial.print("tl               : ");Serial.println(tl);
 Serial.print("Temperatura      : "); Serial.println(Temperatura);
 Serial.print("Status           : "); Serial.println(sts);

 delay(5000);
}}
```