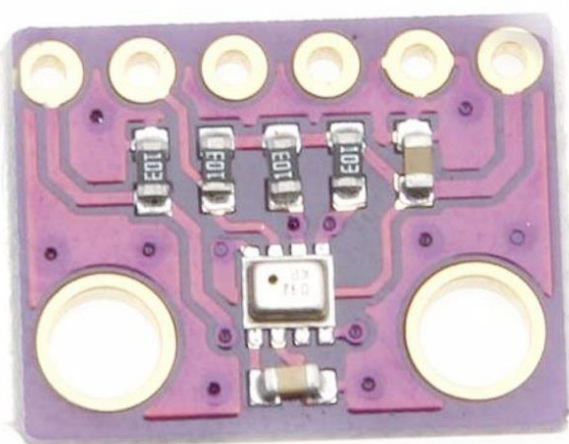


## ・Bosche BME280 センサーを使う

《何ができる?》前回まで、ボリューム（可変抵抗）や CdS センサ（光センサ）を使って、アナログ入力の値を Ambient クラウドサービスにアップロードした。ボリューム（可変抵抗）も光センサ（CdS センサ）もアナログ入力から値を取得している。アナログ入力を用いれば様々なアナログセンサーの値を取得することができる。

ただしアナログ入力は、センサーごとに入力ピン割り当てる。また ESP32 のアナログ入力の値は、ボードによってまちまちとなり、適切なセンサー値を得ることが難しい場合がある。

今回は、アナログ入力を使わずシリアル通信（SPI または I<sup>2</sup>C）を行うセンサーボード BME280 で温度・湿度・大気圧のデータを取得して、Ambient クラウドにアップロードしグラフによる可視化を行いたいと思う。



### 【目標】

シリアル通信（SPI）アナログ値を取得する方法を知る。WiFi ステーション（クライアント）接続で、インターネット・サイトに接続し、シンプルな国産 IoT プラットフォーム「Ambient」を利用して、データのエントリと可視化を行う。

### 【1. ESP32 と電子工作部品との接続】

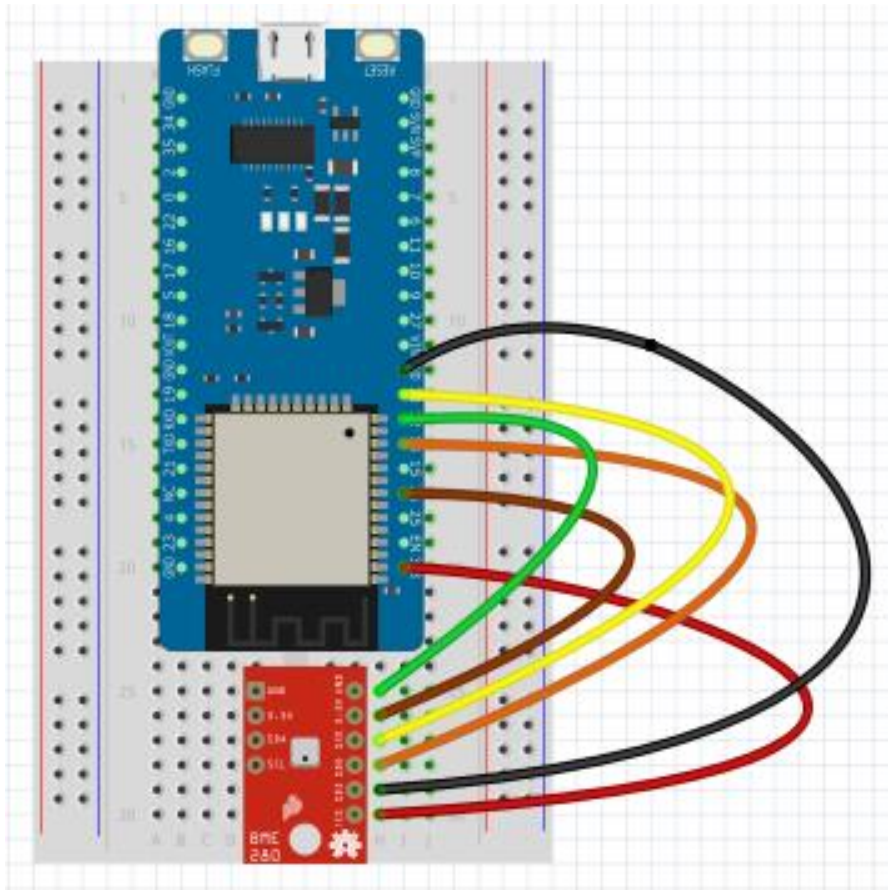
#### 1. 1. 必要な部品

パーツ名	必要個数
Bosch BME280 センサー	1 個
ジャンパーコード	6 本（赤色、茶色、橙色、緑色、黄色、黒色 各 1 本ずつ） ※上記の色が揃わなければ別の色でも良い

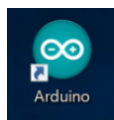
以下の接続を行う。以下の実体配線図を元に接続し、動作を確認すること。

- ① BME280 センサーをブレッドボード 1（ESP32 が載っている方）の g25 から g30 に差し込む。BME280 基板の裏のシルク印刷で VCC と印字されたピンが g30 に刺さるようにする。

- ② ジャンパーワイヤー（赤色）をブレッドボード1のi130へ、もう片方をESP32の3V3ピンへ。
- ③ ジャンパーワイヤ（黒色）をブレッドボード1のi29へ、もう片方をESP32のGNDピンへ。
- ④ ジャンパーワイヤ（橙色）をブレッドボード1のi28へ、もう片方をESP32のIO14ピンへ。
- ⑤ ジャンパーワイヤ（黄色）をブレッドボード1のi27へ、もう片方をESP32のIO13ピンへ。
- ⑥ ジャンパーワイヤ（茶色）をブレッドボード1のi26へ、もう片方をESP32のIO26ピンへ。
- ⑦ ジャンパーワイヤ（緑色）をブレッドボード1のi25へ、もう片方をESP32のIO12ピンへ。



## 【2. Arduino スケッチのサンプルプログラムを実行】



デスクトップのアイコンをダブルクリックして  
Arduino IDE を起動する

### 【課題 33】プロジェクト名「kad33\_BME280」

のマイコン側プログラミングを行う前に、BME280 SPI 接続用のライブラリをダウンロードして libraries フォルダに格納しておく必要がある。

以下の github サイトでライブラリをダウンロードする。

[https://github.com/mgo-tec/ESP32\\_BME280\\_SPI](https://github.com/mgo-tec/ESP32_BME280_SPI)

GitHub, Inc. [US] | [https://github.com/mgo-tec/ESP32\\_BME280\\_SPI](https://github.com/mgo-tec/ESP32_BME280_SPI)

Search or jump to... Pull requests Issues Marketplace Explore

mgo-tec / ESP32\_BME280\_SPI Watch 1 Star 3 Fork 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights

BME280 library for SPI interface ( for Arduino - ESP32 ).

3 commits 1 branch 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find File Clone or download Gitpod

mgo-tec Fix Sample Sketch Latest commit a6d769c on 16 Aug 2017

examples/ESP32_BME280_SPI_sample01	Fix Sample Sketch	2 years ago
src	New Library Upload!	2 years ago
LICENSE	Initial commit	2 years ago
README.md	New Library Upload!	2 years ago
keywords.txt	New Library Upload!	2 years ago
library.properties	New Library Upload!	2 years ago

README.md

ダウンロードした ZIP ファイルを解凍して、フォルダを開く。

名前

ダウンロード > ESP32\_BME280\_SPI-master

名前

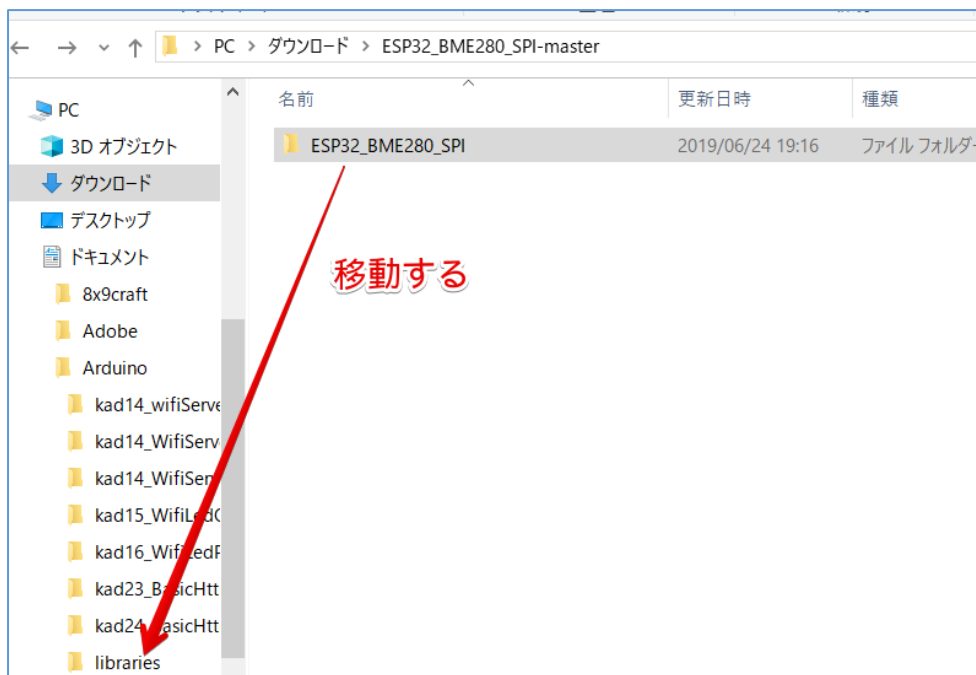
ESP32\_BME280\_SPI-master.zip

ESP32\_BME280\_SPI-master

ESP32\_BME280\_SPI-master

「ESP32\_BME280\_SPI-master」フォルダ内の「ESP32\_BME280\_SPI-master」フォルダを  
「ESP32\_BME280\_SPI」に書き換えて、ドキュメントフォルダ内の Arduino フォルダにある「Libraries」

フォルダに移動する。



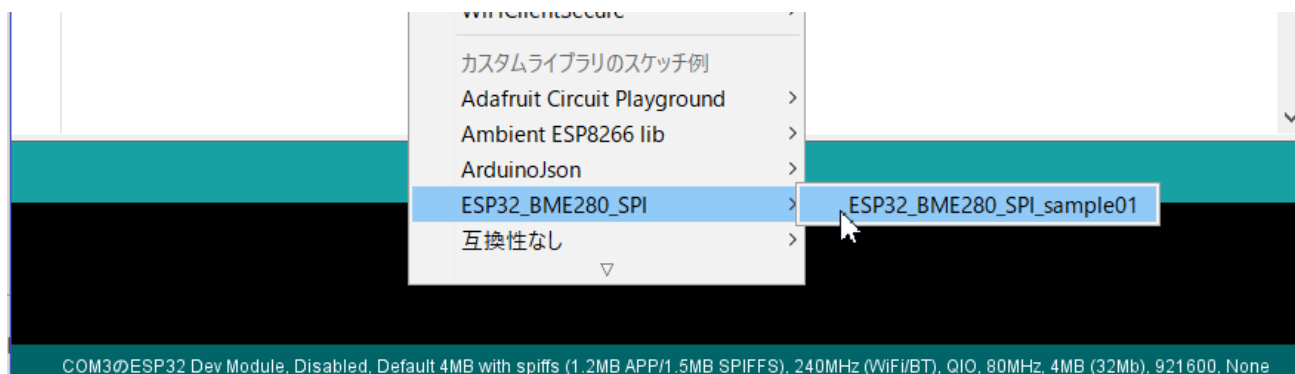
このとき、Arduino IDE を起動しているままならば、いったん Arduino IDE を終了させ、再度起動しなおす。

BME280 のライブラリが利用可能になっているかと BME280 センサーの動作確認のため、サンプルスケッチを開いてみる。

[ファイル]–[スケッチ例]–[ESP32\_BME280\_SPI]–[ESP32\_BME280\_SPI\_sample01]を選ぶ。



<中略>



サンプルスケッチ「ESP32\_BME280\_SPI\_sample01」を開いたら、そのままでは保存できないので、[ファイル]- [名前をつけて保存...]で、「**kad33\_BME280**」というプロジェクト名に変えて保存する。

そのまま実行し、以下の様なシリアルモニタの表示となる。

シリアルモニタ	
-----	
Temperature = 27 °C	←5 秒おきに温度・湿度・大気圧を表示
Humidity = 41 %	
Pressure = 1012 hPa	
-----	
Temperature = 27 °C	
Humidity = 41 %	
Pressure = 1012 hPa	
-----	
Temperature = 32 °C	← BME280 を指で押さえると、気温よりも温度が上がる
Humidity = 47 %	← 湿度も少し上がる
Pressure = 1012 hPa	←大気圧は指で押さえても変わらないようだ
-----	
Temperature = 27 °C	
Humidity = 42 %	
Pressure = 1012 hPa	
-----	

上記のように温度・湿度・大気圧が適切な値ができれば BME280 センサーボードの接続は OK。

### 【課題 34】 プロジェクト名「kad34\_BME280\_Ambient」

課題 33 の温度・湿度・大気圧を Ambient クラウドサービスにアップロードするプログラムを作る。その後 Ambient サイトで温度・湿度・大気圧をグラフで可視化するようにする。チャンネル ID (ライトキーも) は前回の課題 31 「kad31\_cdsAmbient」を流用して、d1, d2, d3 にエントリする形にする。

d1, d2, d3 のデータソース(データ元)は、先ほどの課題「kad33\_BME280」を温度・湿度・大気圧の順でエントリする。

【ポイント】 Ambient は d1 から d8 まで 8 個のデータを同時にエントリできる。

d1～d3 は同時に send するようにせよ。そのほうが、通信量・省電力・リアルタイム性などが良い。

以下 loop() 関数のみヒントを提示する。

```
52 void loop() {
53     delay(5000); // 順序入れかえ最初に持ってくる
54
55     // ここから BME280 kad33 の bme_get() を使わず温度・湿度・大気圧の変数を float 型でセット
56     float temperature = bme280spi. [redacted];
57     float humidity = bme280spi. [redacted];
58     float pressure = bme280spi. [redacted];
59     // BME280 の温度・湿度・大気圧の値を d1, d2, d3 にセット
60     [redacted]
61     [redacted]
62     [redacted]
63     // Ambient クラウドに一斉送信
64     [redacted]
65
66     // BME280 の温度・湿度・大気圧の値をシリアルモニタに表示
67     Serial.printf("温度 = [redacted]");
68     Serial.printf("湿度 = [redacted]"); // % は半角で表示
69     Serial.printf("大気圧 = [redacted]");
70
71     delay(10000); // 追加
72 }
```

d1 温度 タイトル「温度」

- ・ float 型の温度のための変数(temperature など)に bme280spi.Read\_Temperature() を入れる
- ・ Serial.printf() 関数を使って小数第 1 位までの浮動小数点数を書式表示

d2 湿度 (新しくグラフを追加)

- ・ float 型の湿度のための変数(humidity など)に bme280spi.Read\_Humidity() を入れる
- ・ Serial.printf() 関数を使って小数第 1 位までの浮動小数点数を書式表示

d3 大気圧 (新しくグラフを追加)

- ・ float 型の大気圧のための変数(pressure など)に bme280spi.Read\_Pressure() を入れる

- ・ Serial.printf()関数を使って小数第1位までの浮動小数点数を書式表示

#### 【課題 34 実行結果】 シリアルモニタ

```

rst:0x10 (RTCWDT_RTC_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
WiFi connected
IP address: 192.168.1.80
温度 = 30.2 [°C]
湿度 = 42.5 [%]
大気圧 = 1013.3 [hPa]
温度 = 30.3 [°C]
湿度 = 42.2 [%]
大気圧 = 1013.3 [hPa]
温度 = 30.3 [°C]
湿度 = 42.1 [%]
大気圧 = 1013.3 [hPa]
温度 = 30.3 [°C]
湿度 = 42.2 [%]
大気圧 = 1013.2 [hPa]
温度 = 30.3 [°C]
湿度 = 42.1 [%]
大気圧 = 1013.3 [hPa]
温度 = 30.2 [°C]

```

温度・湿度・大気圧表示  
小数第1位まで表示

10+5 秒おきに表

以下 15 秒おきに表示を繰り返す

☐ 自動スクロール ☐ タイムスタンプを表示 LFのみ 115200 bps 出力をクリア

グラフを追加して3つのデータが同時に見られるようにしよう。

#### 【課題 34 実行結果】 Ambient グラフ

Ambient クラウド側のグラフは以下の3つになる。

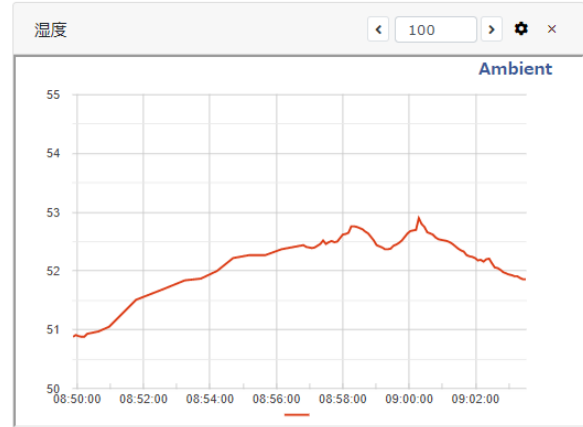
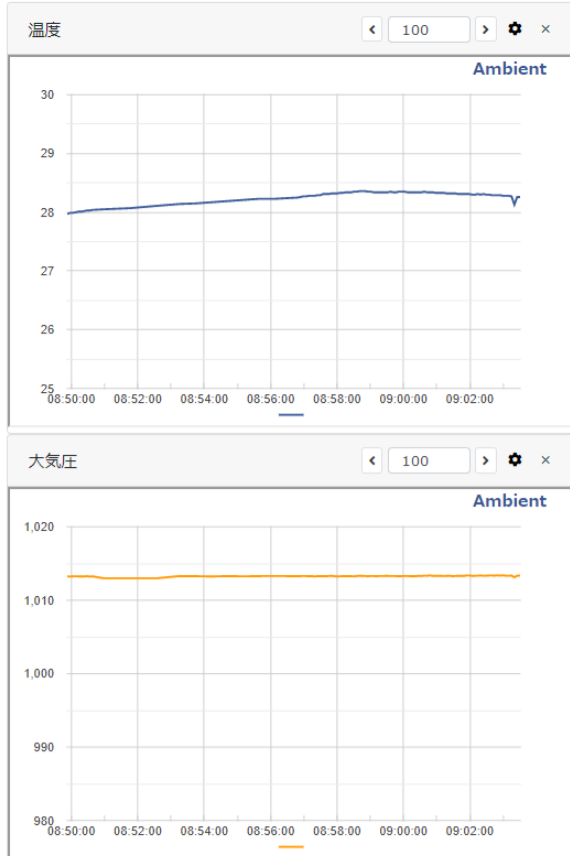
温度・湿度・大気圧のグラフのY軸を調整して、変化が分かりやすいように工夫してみよう。

(例：温度だと 25～35℃にする等)

Myチャンネル / IE3A00 (チャンネルID: 17220)



最新データ登録: 2020/8/3 9:03:30



電源電圧・温度・湿度・大気圧のグラフのY軸を自分なりに工夫して調整できたら、動作チェックするので先生に申告する。



### 【課題 35】 プロジェクト名「kad35\_BME280\_DeepSleepAmbient」

課題 34 で温度・湿度・大気圧を Ambient クラウドサービスにアップロードすることができたが、センサー値をアップロードしていないときも WiFi に接続し無駄な電気を消費している。

電池などで駆動させることも考えて、DeepSleep を行うようにしてみる。

以下の記事にある「温湿度センサデバイスのプログラム」を元に、【課題 34】「kad34\_BME280\_Ambient」とソースを組み合わせてまずは 60 秒おきに DeepSleep するようにする。

ESP32 ではじめる IoT デバイス開発：一般記事 | gihyo.jp … 技術評論社

[https://gihyo.jp/dev/column/01/iot/2019/esp32\\_iot?page=2](https://gihyo.jp/dev/column/01/iot/2019/esp32_iot?page=2) (←記事の 2 ページ目)



図3 温湿度センサデバイスの回路図

デバイスは単3乾電池3本で駆動します。電池電圧を抵抗で分圧してESP32のADコンバータ（SENSOR\_VNピン）に加え、温度、湿度と合わせて測定するようにしました。

#### 温湿度センサデバイスのソフトウェア

温湿度センサデバイスのプログラムは次のようになります。

```
#include
#include "Adafruit_Si7021.h"
#include

#define TIME_TO_SLEEP 60 // 測定周期(秒)

const char* ssid = "ssid"; // Wi-FiルーターのSSID
const char* password = "password"; // Wi-Fiルーターのパスワード

WiFiClient client;
Ambient ambient;

unsigned int channelId = 100; // AmbientのチャンネルID
const char* writeKey = "writeKey"; // ライトキー

Adafruit_Si7021 sensor = Adafruit_Si7021();

#define BATTERY 39 // バッテリー電圧を測るピン

void setup(){
  unsigned long starttime = millis();
  Serial.begin(115200);
  while (!Serial);

  WiFi.begin(ssid, password); // Wi-Fiネットワークに接続する
  while (WiFi.status() != WL_CONNECTED) { // 接続したか調べる
    delay(500);
  }
```

記事サイトをスクロールすると左の図のような「温湿度センサデバイスのプログラム」が見つかる。

これを元ソースとして【課題 35】「kad35\_BME280\_DeepSleepAmbient」を作ると良い。

このプログラム、なぜか1行目、3行目が#include 命令のみでヘッダライブラリの指定が無い。追加修正しよう(2行目もBME280のヘッダに変更する必要がある)。

DeepSleep の期間を指定するTIME\_TO\_SLEEP マクロが60秒となっているが、確認するには長い為実行時には15秒程に短縮して良い。

Ambient のチャンネル ID, ライトキー WiFi のSSID、パスコードは書き換える。

この記事では、センサーはBME280を使っておらずSi7021というデジタル温湿度センサーを用いている。この部分をBME280 センサーのコードに差し替えるとうまく動作する。

【課題 35 実行結果】 シリアルモニタ

COM3

送信

```
rst:0x5 (DEEPSLEEP_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
config: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
.温度 = 27.4 [°C]
湿度 = 48.5 [%]
大気圧 = 1013.2 [hPa]
ets Jun  8 2016 00:22:57
rst:0x5 (DEEPSLEEP_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
config: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
.温度 = 27.4 [°C]
湿度 = 48.4 [%]
大気圧 = 1013.2 [hPa]
```

温度・湿度・大気圧表示  
小数第1位まで表示

指定時間後再度立ち上がる

温度・湿度・大気圧表示  
小数第1位まで表示

以下指定時間おきに表示を繰り返す

☐ 自動スクロール ☐ タイムスタンプを表示

出力をクリア