国際開発農学実験実習24

IoT *センサー実習*

• 温湿度モニタリング機器を 作ってみよう

- 国際情報農学研究室
- 教授：溝口勝
- TA：

高草木和史 （2021）

野田坂秀陽 （2022）

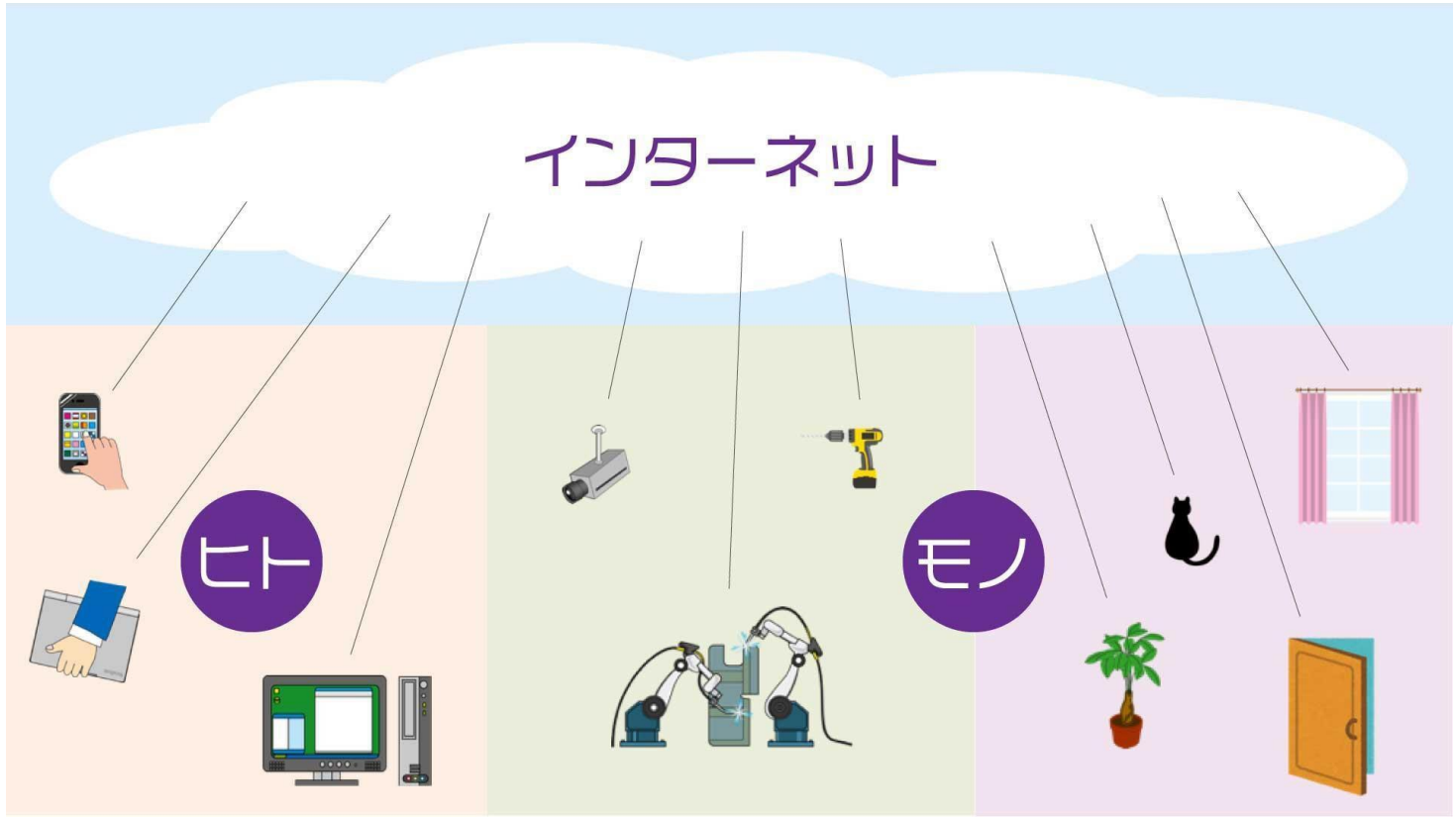
野田坂秀陽／上坂粋芳 （2023）

野田坂秀陽（2024）

そもそもIoTとは?

- Internet of Things:
  - モノがインターネット経由で通信することを意味する
  - モノのインターネット

以前は、インターネットはコンピュータ同士を接続するためのものだった



IoTを支える３技術

・**デバイス技術**

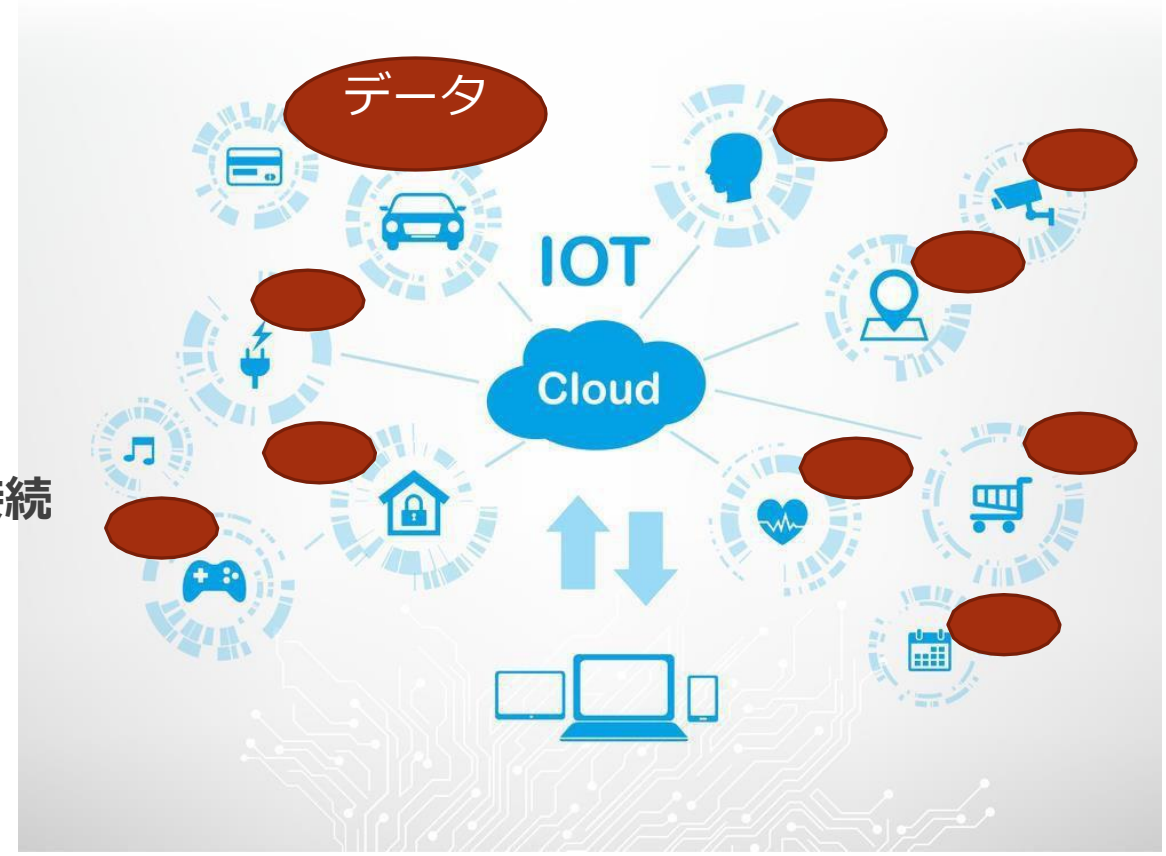
　　モノからデータを得る技術

・**システム技術**

　　データを送受信してクラウドに接続

・応用技術

　　データの分析と処理を行う



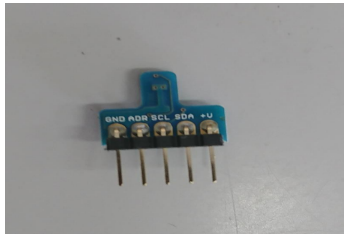
IoT実習の目的

IoTデバイスの仕組みの概要を理解し、
自分で作品を制作できるようになる。

今日やること

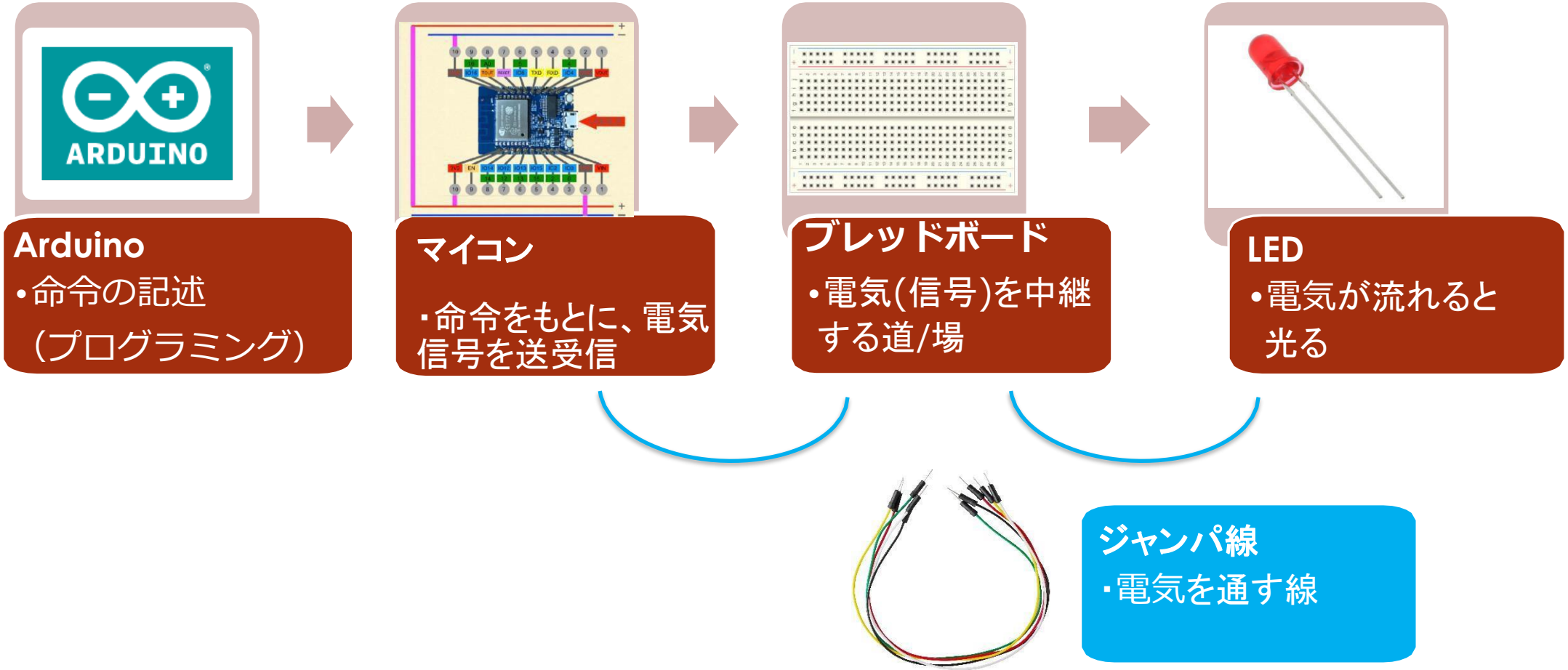
1. **LEDをチカチカ点灯させる。（略してLチカ）**
    1. 明るさ、点灯・消灯のタイミングなどを制御するプログラムを書く。

2. **温度センサーで温度を取得。**
    1. 取得した温度をクラウド上に自動アップロードし記録する温度計測デバイスを作る。



3. **応用編**
    1. 温度に応じてLEDの光り方を変えるプログラムを書く。

LEDをチカチカさせよう！



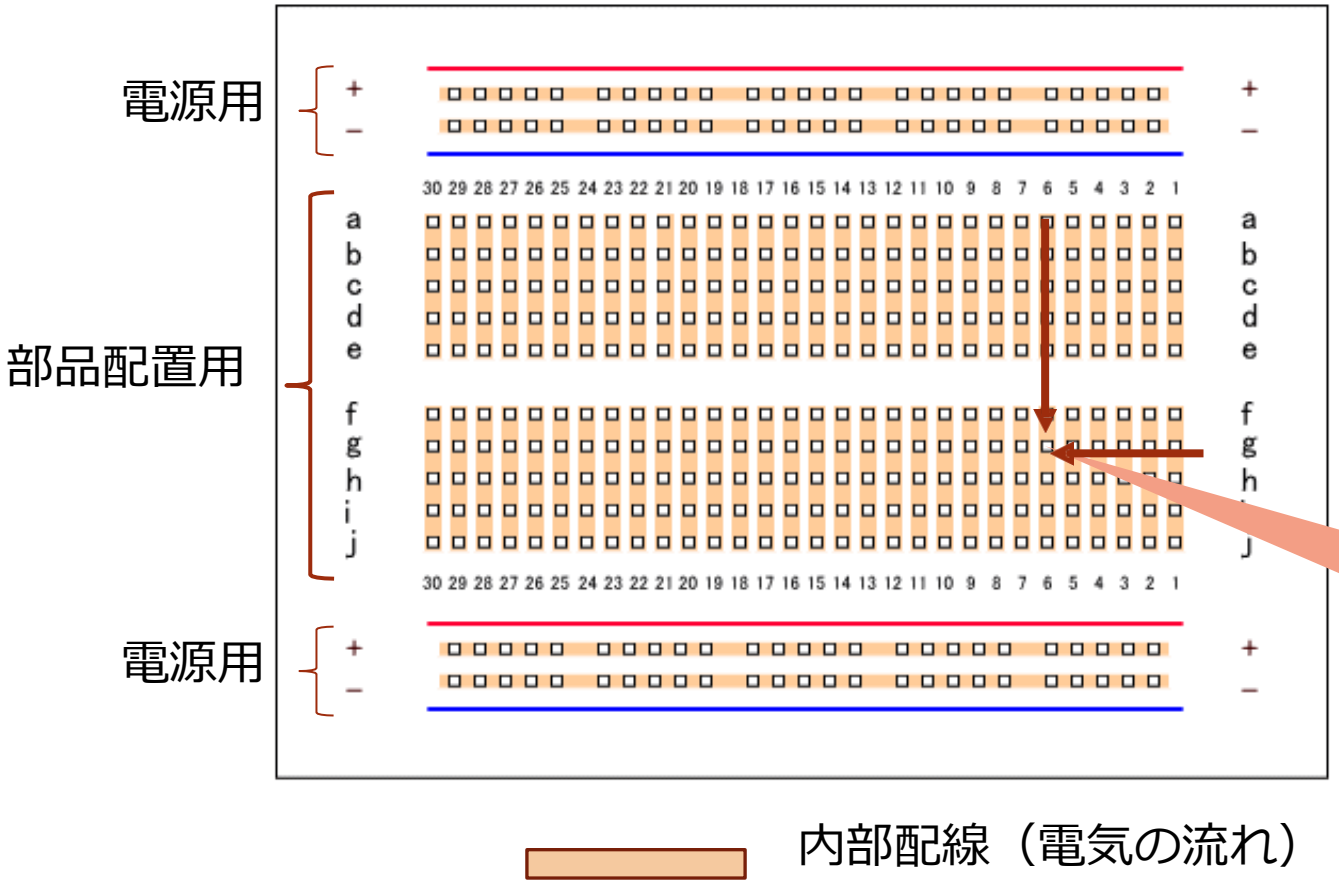
手順

1. ハード（機械）の制作

2. PCのソフトのダウンロード

3. ソフト（プログラム）の制作

ブレッドボードの配線



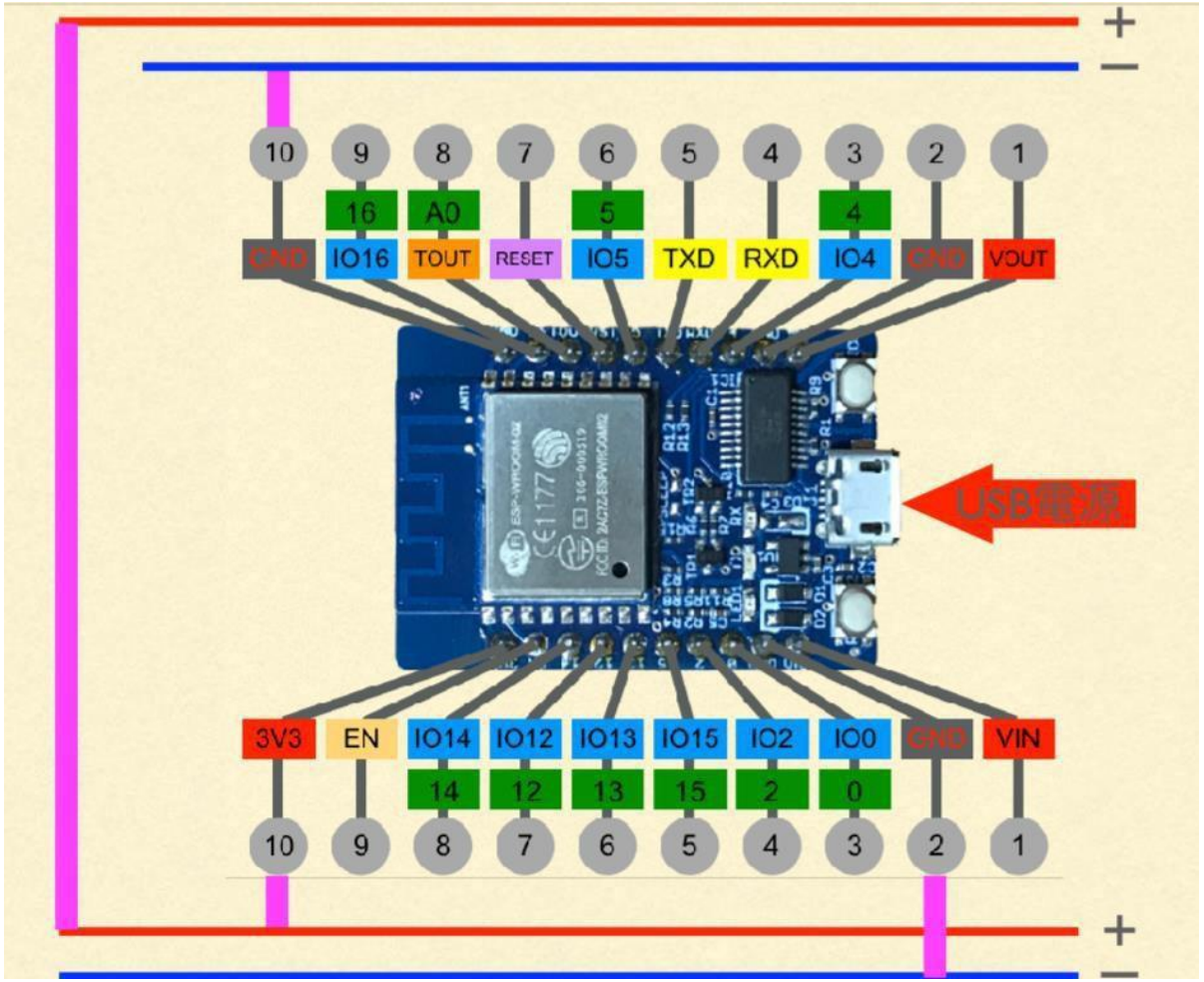
- 上から二列、下から二列は電源用。

  配線は横になっている

- 中の10列は部品を配置するためにある。配置する位置はアルファベットと数字によって**座標**で表せる

  配線は縦になっている。

開発ボードの配線

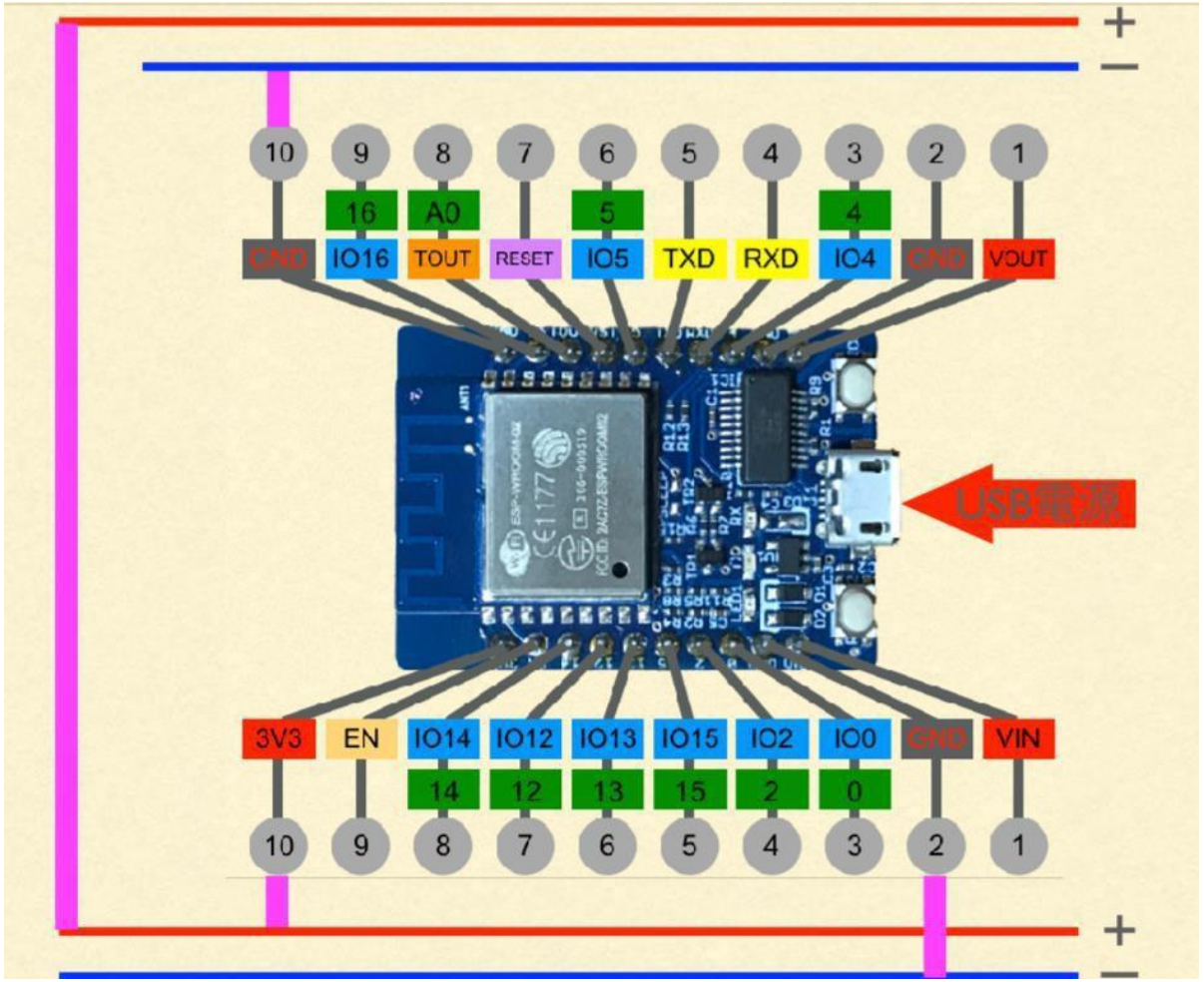


命令を処理したり記憶したりする場所

開発ボードのピンには、
それぞれ異なる機能がついている
今回のマニュアル内で
上段を↑、下段を↓で表す。

いまは、開発ボードと
ブレッドボードがつながっていないので電気を通す

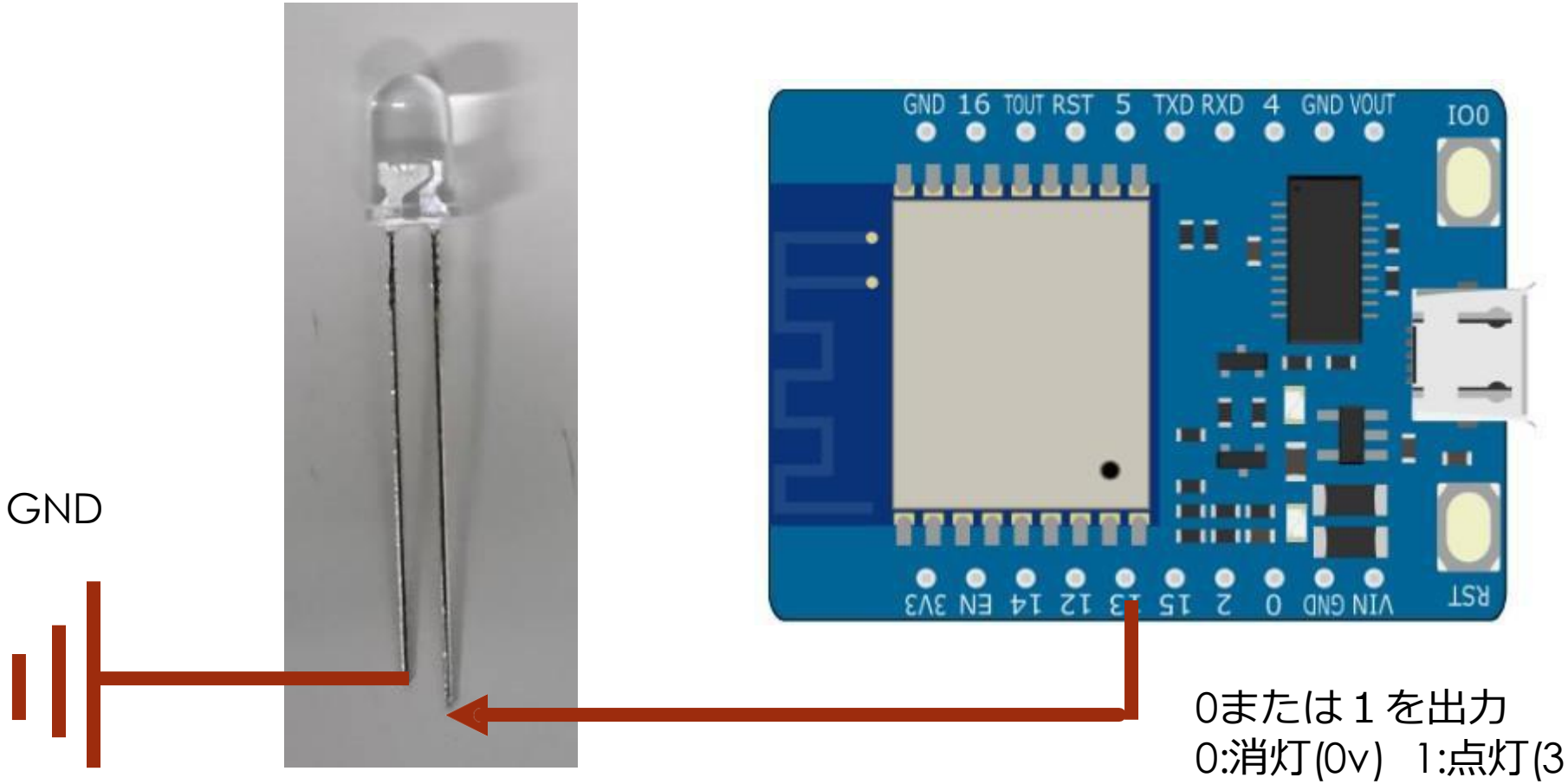
開発ボードの配線



ブレッドボード内の赤線沿いには、電圧がかかっている

青線沿いはGNDに接続しているので、電圧がかかってない状態

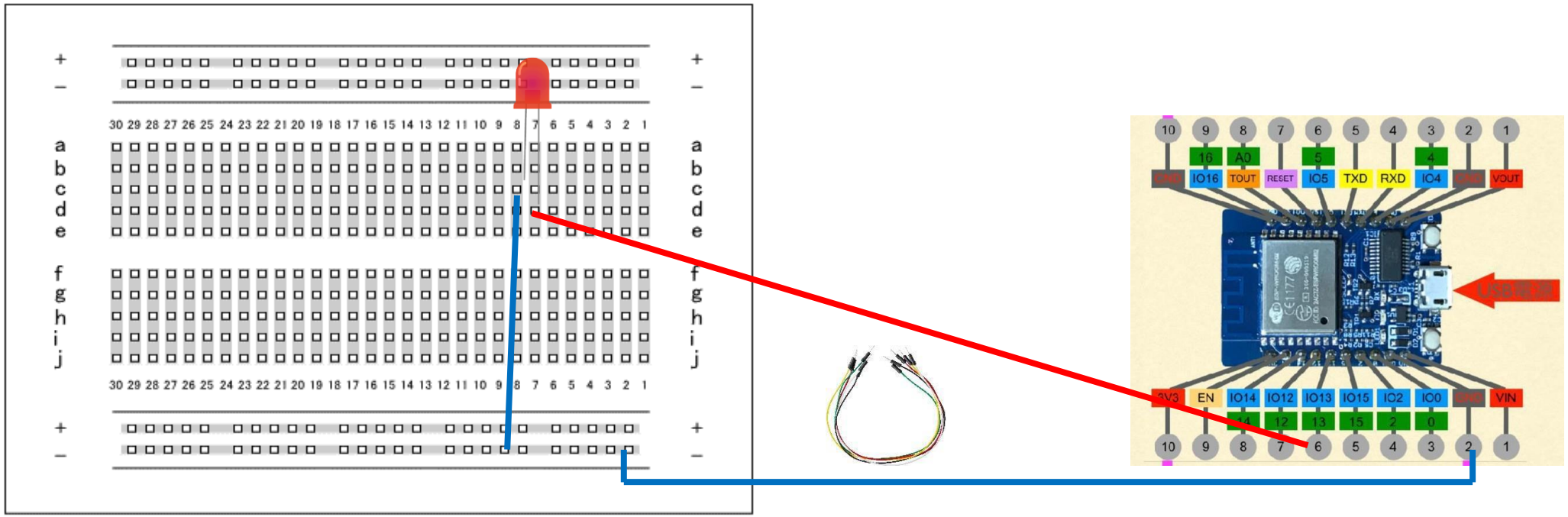
Lチカ回路のイメージ



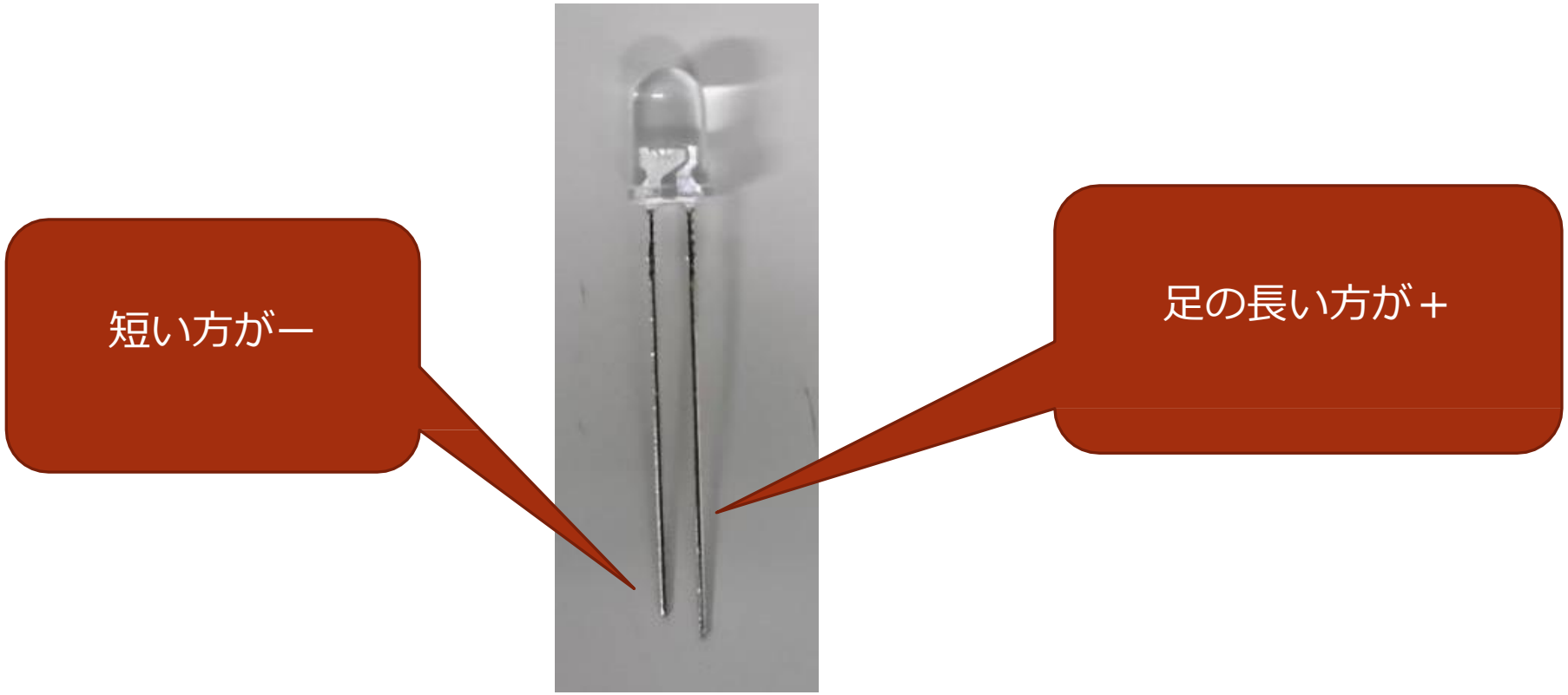
0または１を出力
0:消灯(0v)　1:点灯(3.3v)

準備：開発ボードとブレッドボードの配線

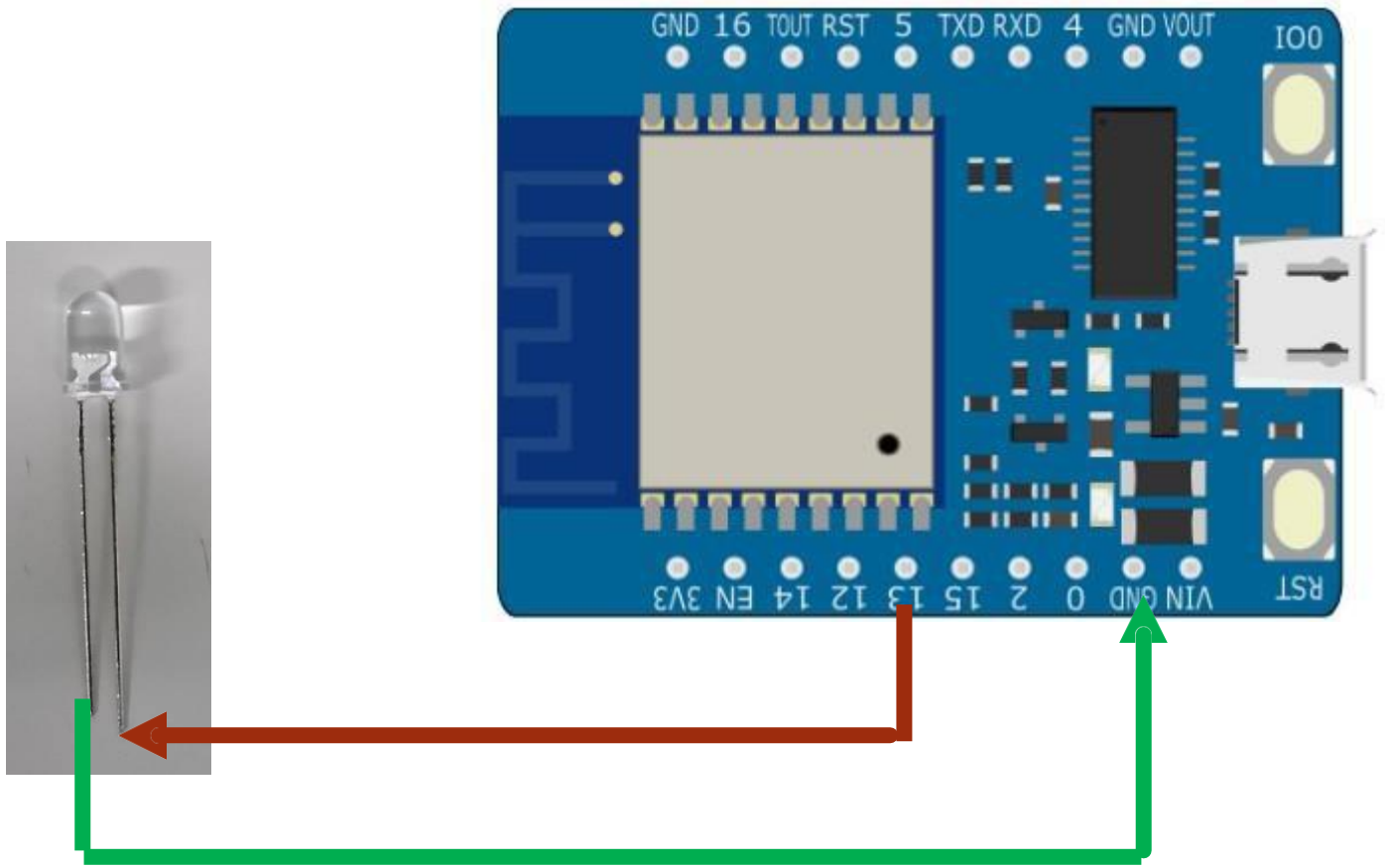
青：LEDの足の短い方（マイナス極）
赤：LEDの足の長い方（プラス極）
ジャンパ線の色は図と一致しなくてOK



LEDの極性



回路のイメージ


GND 16 TOUT RST 5 TXD RXD 4 GND VOUT
IO0
RST
VIN GND 0 2 15 13 12 14 EN 3V3

1.PCのソフトのダウンロード

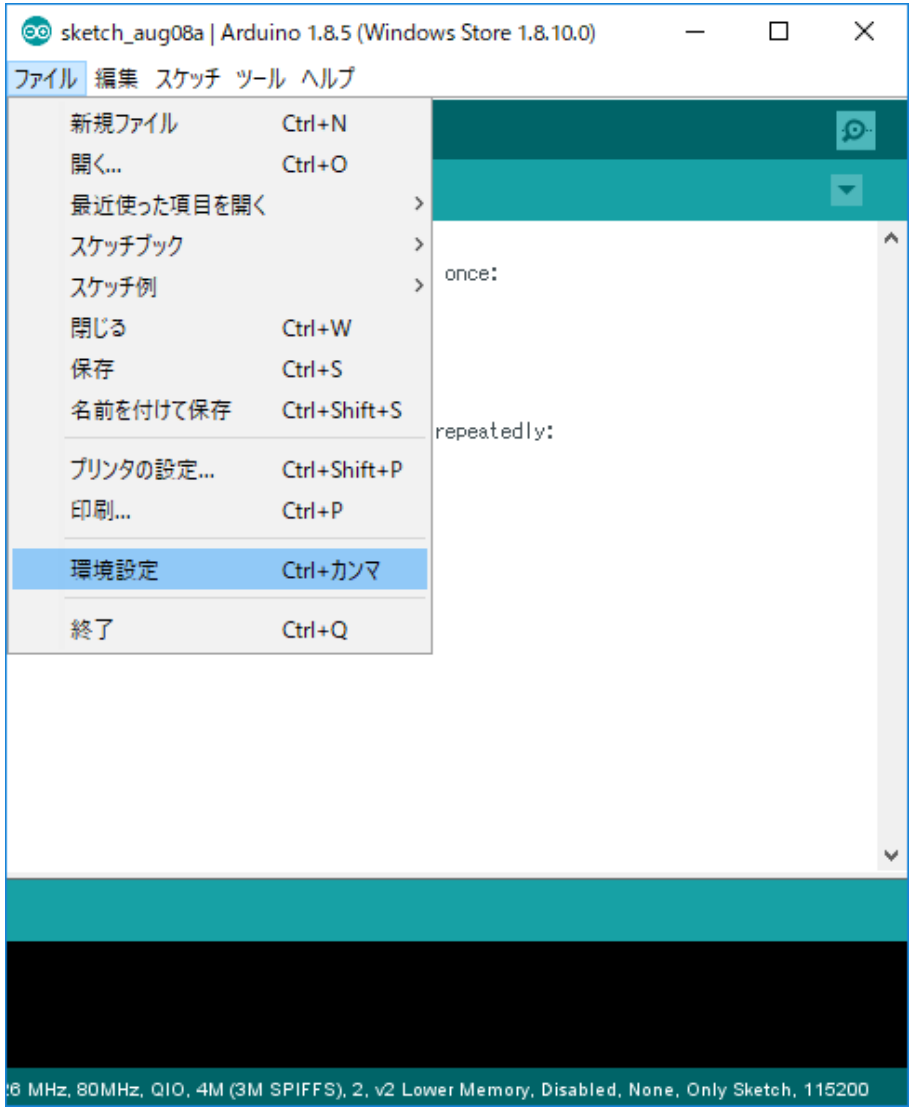
Arduino IDEのダウンロードをArduinoのwebページで行います。

https://www.arduino.cc/

1. [SOFTWARE] メニューの[DOWNLOADS] を選択
2. [Previous Releases] → **[Previous Release (1.8.13)] をクリック**
3. [1.8.13]の中で自分のバージョンをダウンロード
4. Contribute to the Arduino Software の画面はお金を払いたくない場合は、右下の[JUST DOWNLOAD]をクリックしましょう。
5. ダウンロードができたらArduinoを開きましょう。
6. ダウンロードしたインストーラのファイル名は　Arduino-1.8.12-windows.exeのような名前です。

ボードマネージャの追加

- 左図のように「ファイル」メニューから、環境設定画面を開く
- チェックボックスの下の、追加のボードマネージャのＵＲＬを追加します。

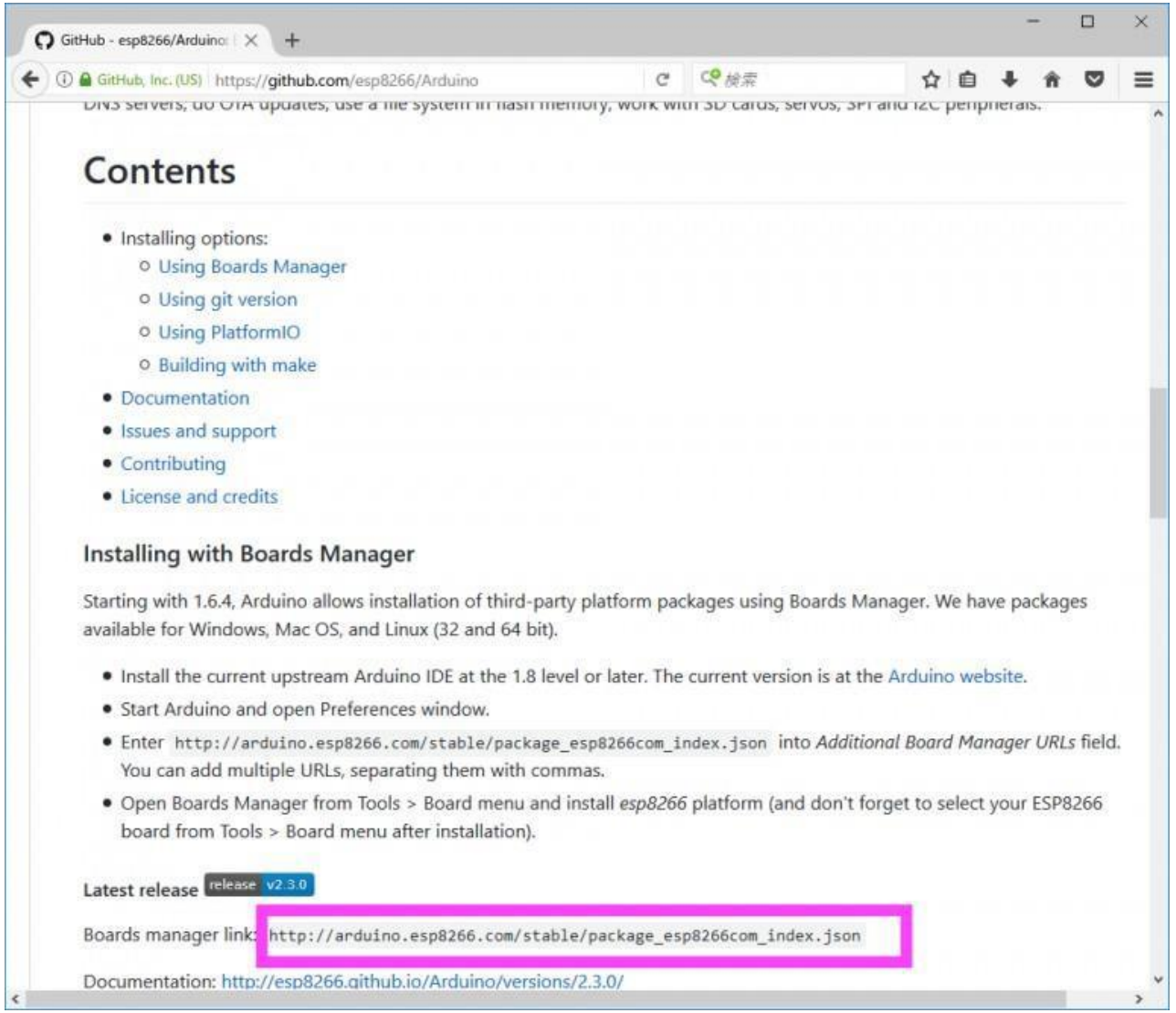


追加のボードマネージャ

↧ https://github.com/esp8266/Arduino
を開いて下の方にスクロールすると、ボードマネージャのリンクがあるのでコピーして、

前頁の“追加のボードマネージャのURL”のところにペーストします。

ＯＫボタンを押して環境設定を終了します。



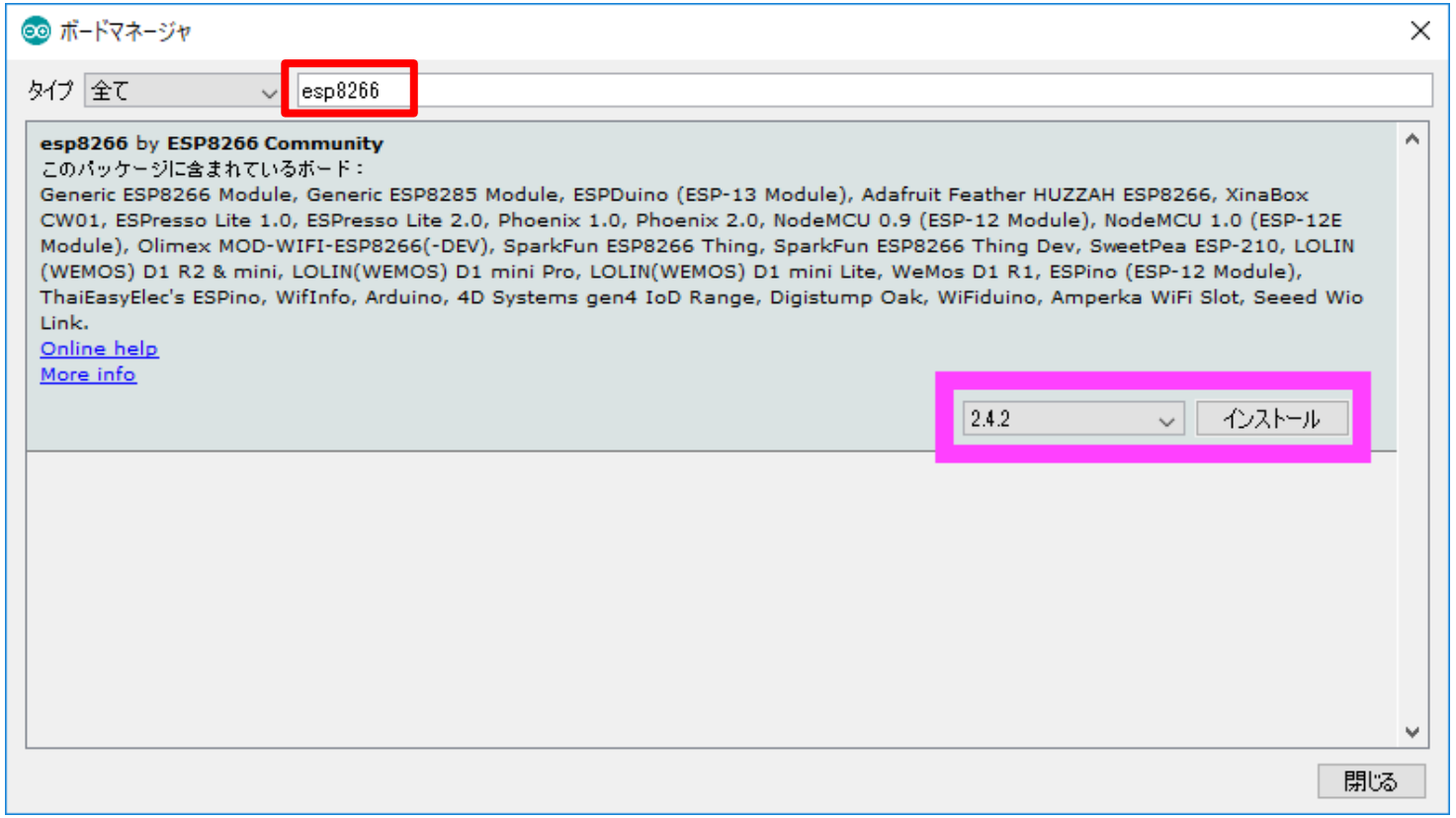
ライブラリの追加

- ［ツール］→［ボード］→［ボードマネジャ］で

  ボードマネージャ画面を開く

- 上の検索欄に"esp8266"を記入する

- 紫枠のようにversion2.4.2に変更しインストールを行う。

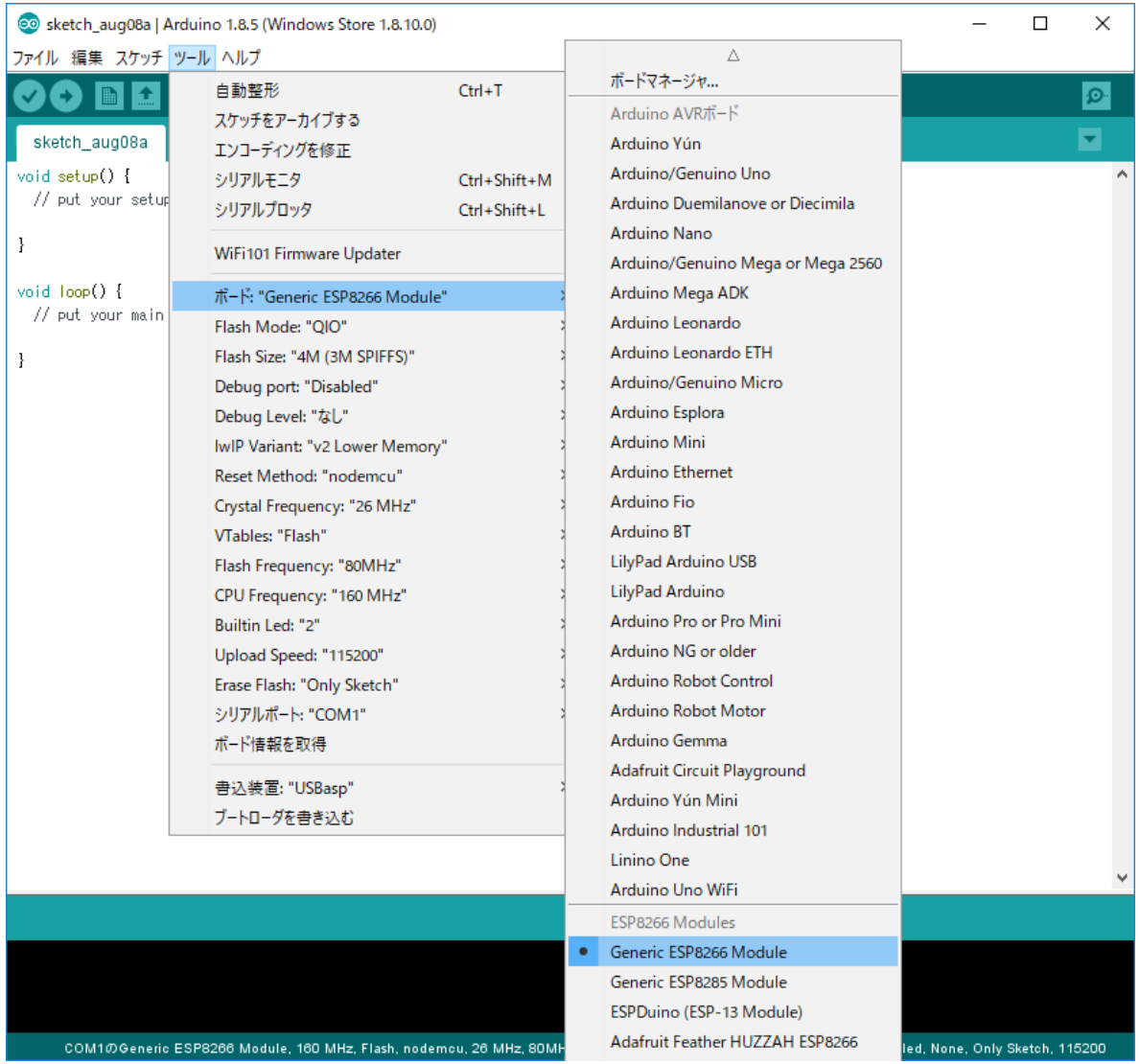


ボードの選択

「ツール」→「ボード」

で下の方にスクロールすると、

[Generic ESP8266 Module]という項目が現れるのでそれを選択する

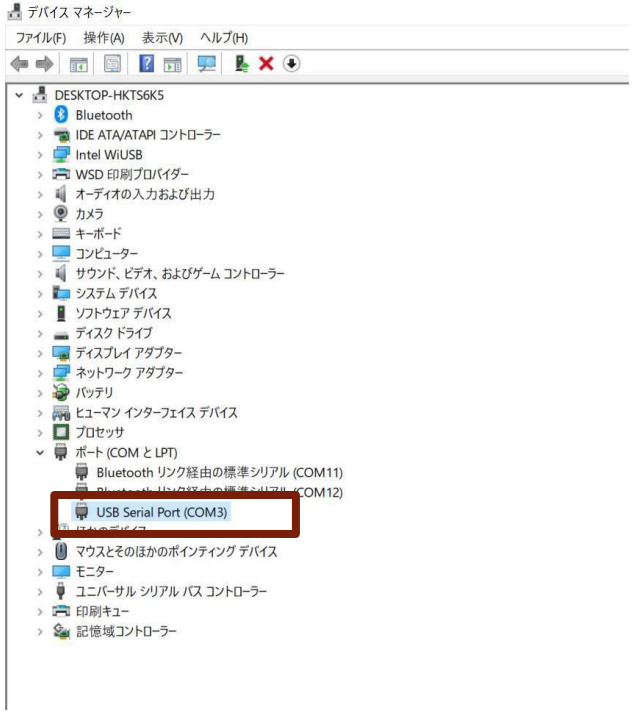


接続の確認

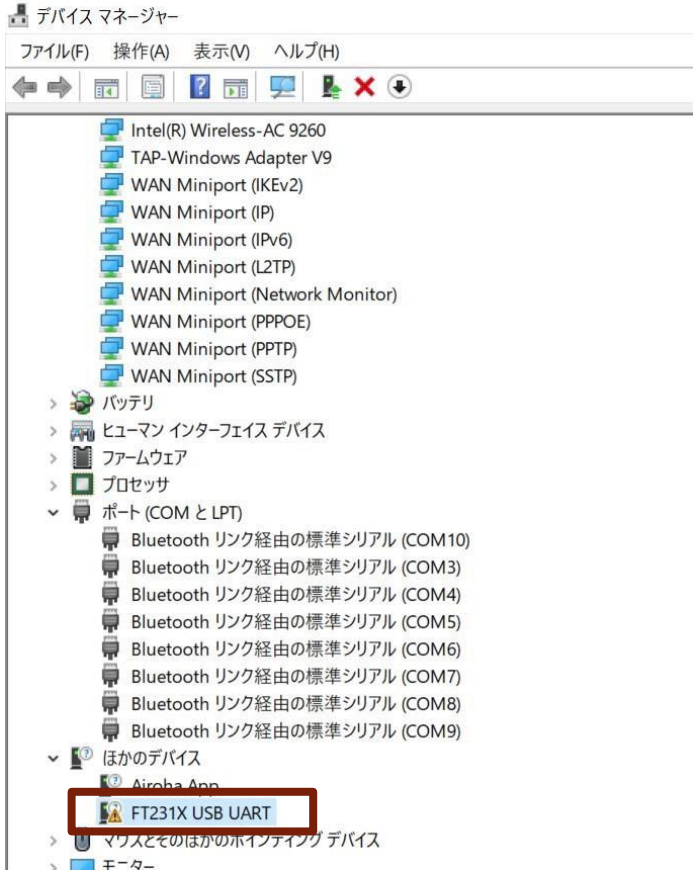
- 開発ボードとパソコンをつないで、シリアルポートを選択する。
- そのとき追加前から増えたシリアルポートを選択する。
- シリアルポートが追加されない場合は、
  - 環境設定のURL確認
  - ツールの設定が正しくなっているか確認
  - ケーブルを交換する

シリアルポートが読み込めないとき　1

上手くいくケース
　ポートに’USB Serial Port (COM番号)’が現れる



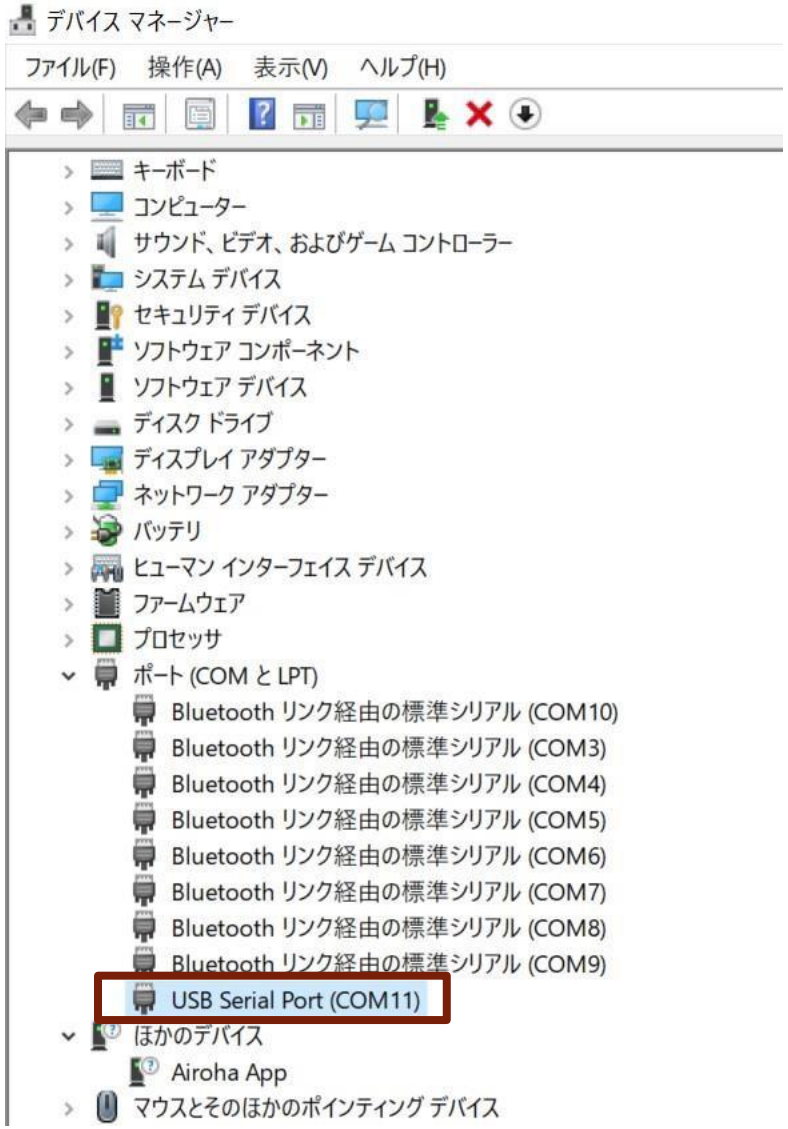
上手くいかないケース
　ポートに’USB Serial Port (COM番号)’が現れない



シリアルポートが読み込まれないとき

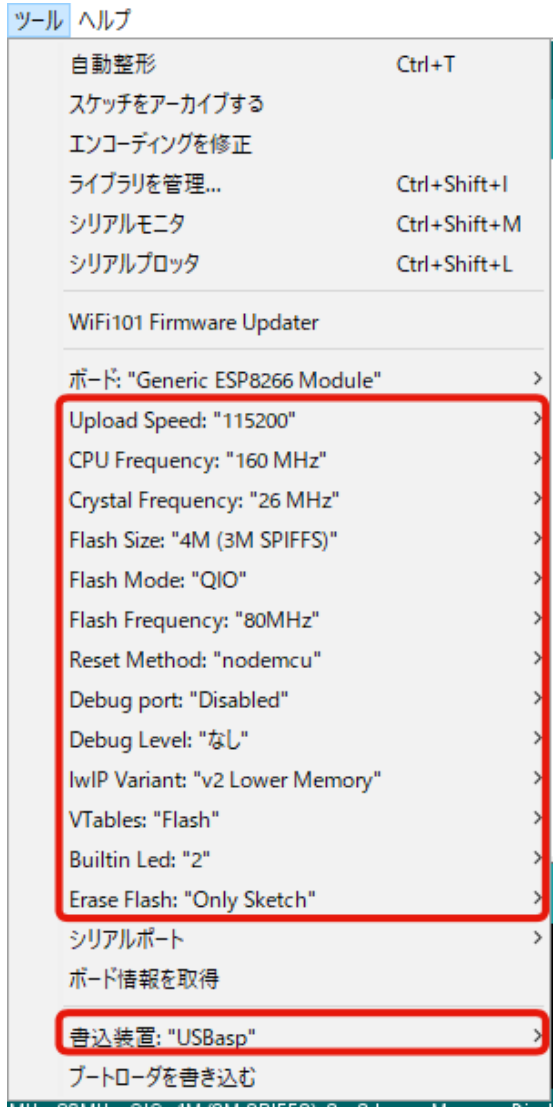
2

- https://www.monoxit.com/tecmat/windows/ftdiusb/
- なんのドライバーをインストールする必要があるかを、確認して手動で更新する
  - 例では、FTDI社のドライバーをインストールしている
- このエラーはパソコンの規格によって異なる
- ‘読み込まれないドライバ名　手動’ などとググってがんばる



ツール設定の変更

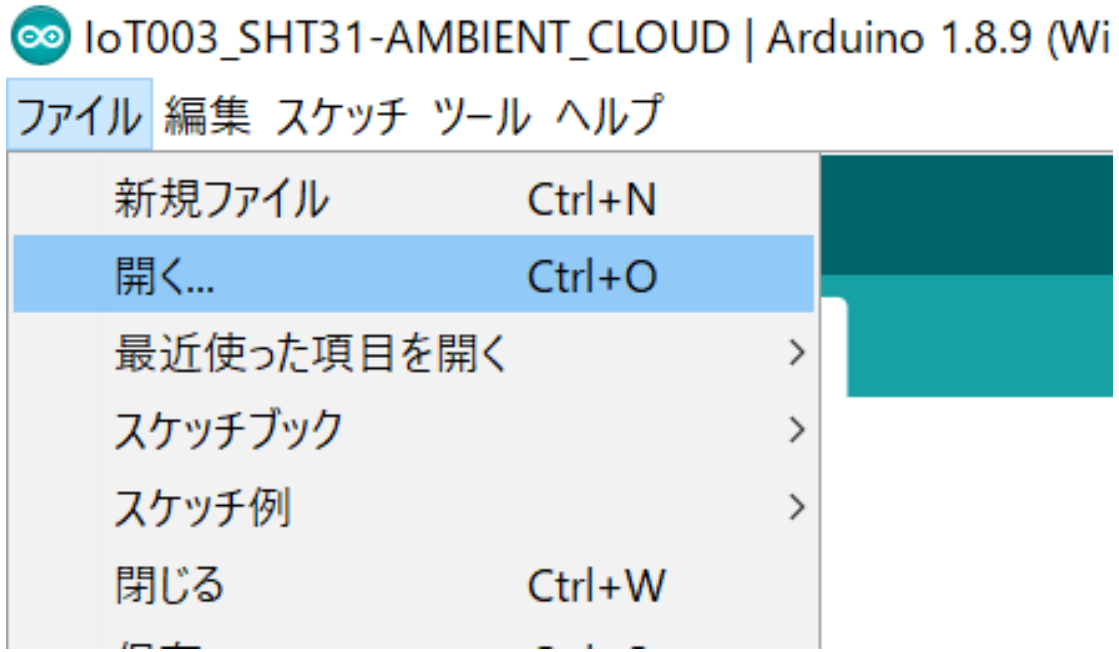
- ツールを選択して、赤枠と同じ設定に変更する



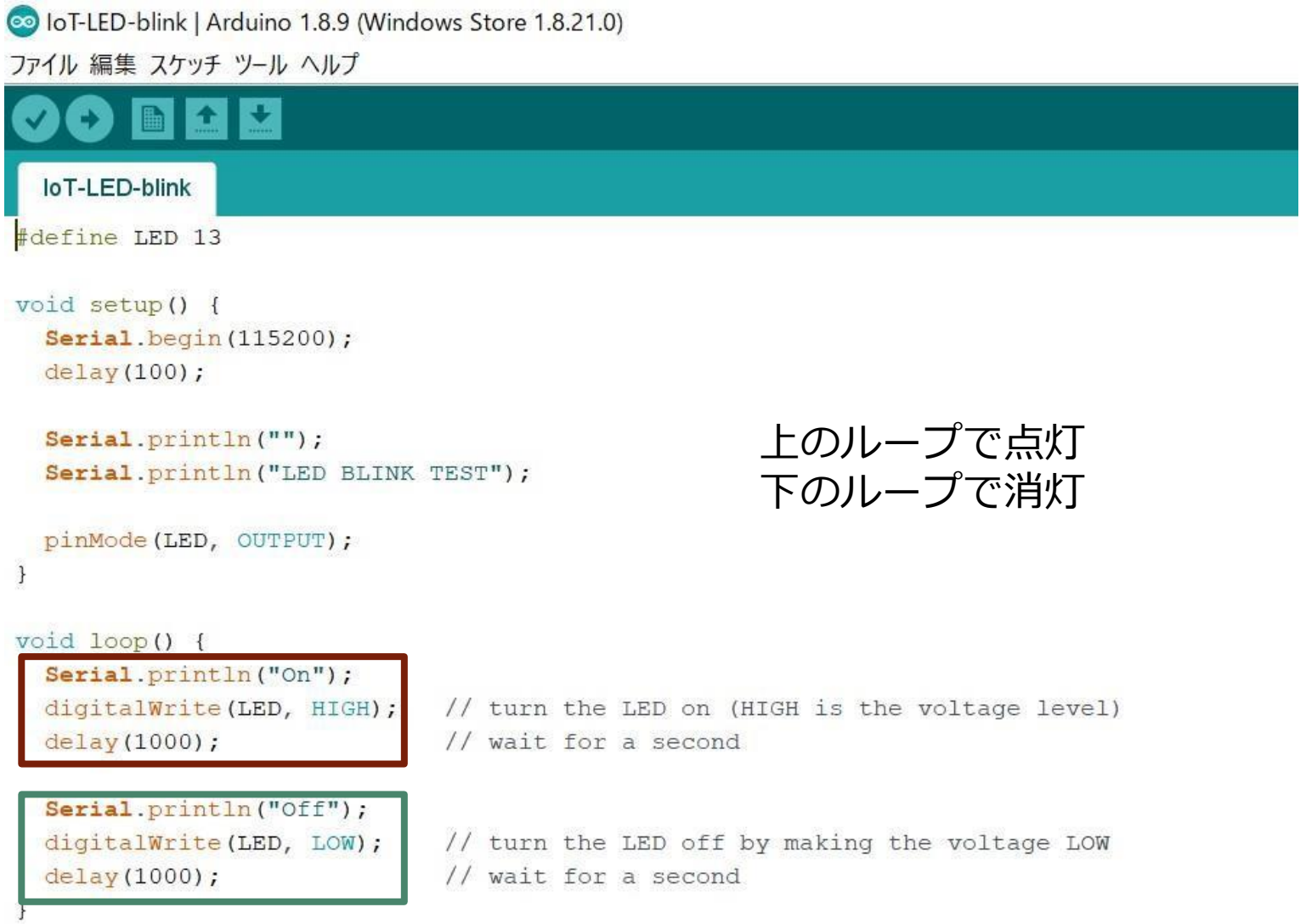
Arduinoのプログラムの構造

ソースコードの開き方

- IDEを起動して[ファイル]メニューの[開く]で開く。
- Lチカでは、デスクトップにある[ソースコード]のフォルダ IoT1-LEDを開いてください



Lチカのソースコード




```
#define LED 13

void setup() {
  Serial.begin(115200);
  delay(100);

  Serial.println("");
  Serial.println("LED BLINK TEST");

  pinMode(LED, OUTPUT);
}

void loop() {
  Serial.println("On");
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second

  Serial.println("Off");
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

上のループで点灯
下のループで消灯

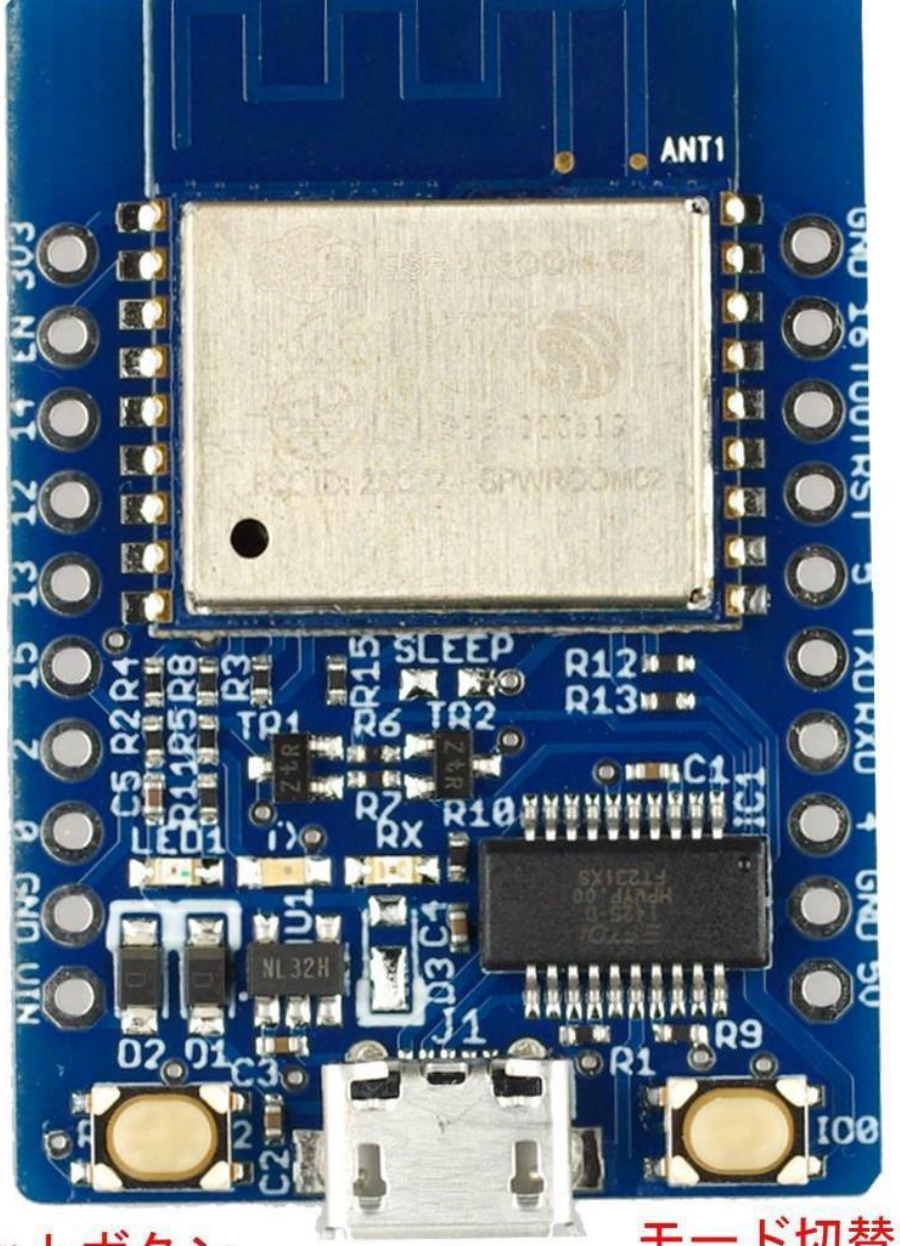
コンパイルと読込

矢印マークを
クリックすると
マイコンへの
書き込みが始まる


```
#define LED 13

void setup() {
  Serial.begin(115200);
  delay(100);

  Serial.println("");
  Serial.println("LED BLINK TEST");

  pinMode(LED, OUTPUT);
}

void loop() {
  Serial.println("On");
  digitalWrite(LED, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second

  Serial.println("Off");
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

コンパイルとプログラムの書き込み

- シリアルコンソールが開いていたら閉じる
- ESPr DeveloperにUSBケーブルを接続する
- ツールからシリアルポートを選択
- シリアルモニターを開き、Developerのリセットボタンを押す
- マイコンのFLASHボタンを押しながら、コンパイル＆プログラミングの書き込みボタンをクリック
- 書き込みが完了



リセットボタン

モード切替ボタン
押すとIO0:L
放すとIO0:H

書き込み完了

書き込みが完了すると下のところにお知らせが来る

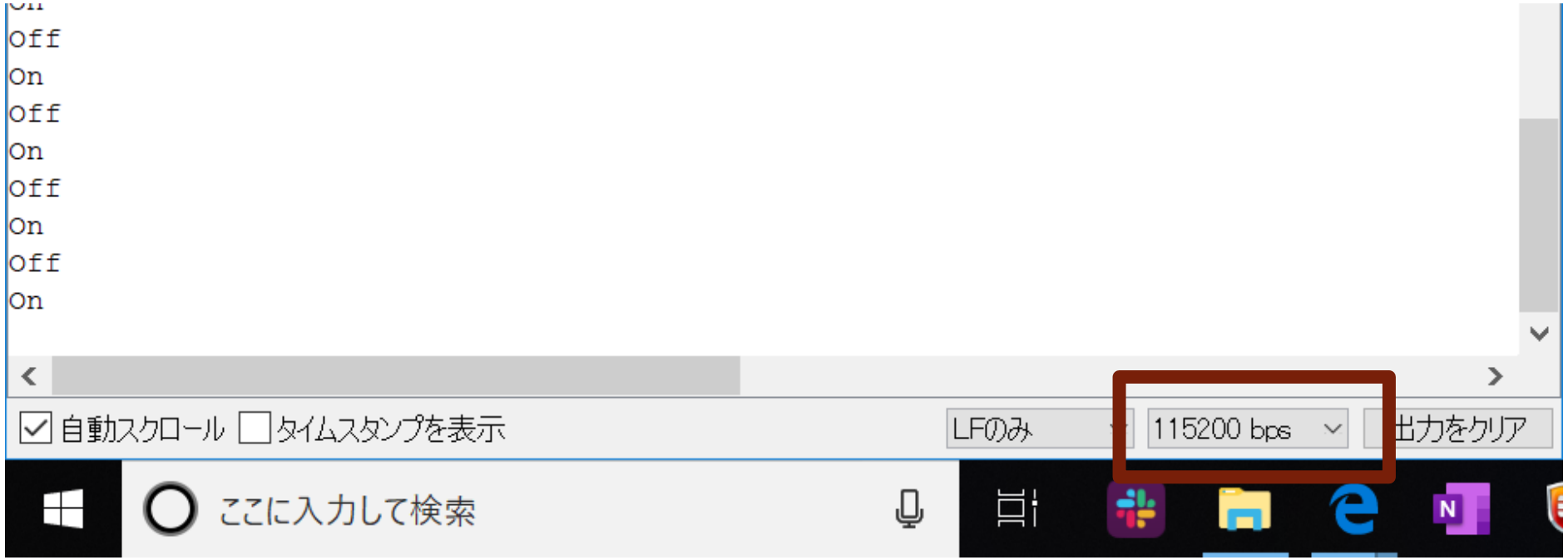
```
  Serial.println("Off");
  digitalWrite(LED, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

コンソール出力


COM10
On
Off
On
Off
On
Off
On
Off
LED BLINK TEST
On
Off
On
Off
On
Off
On
Off

ｺコンソール出力の方法

サリアルポートを開いて
右下の枠のところが、115200bpsになっているか確認



よくあるエラー

- シリアルポートが読み込めない
  - P.13-14をよく読む
- 書込装置をUSBaspに変更できない
  - Arudinoのバージョンを確認。1.8.12以下にする

応用編

点灯・消灯の間隔を変える。

Delayの（）内の数字を変える。（）内の数字は、動作を止める時間の長さをミリ秒で表している。点灯した状態でdelay(1000)にすると、点灯したまま1000ミリ秒＝1秒そのままになる。

明るさを変える

アナログ出力を使う。アナログ出力とは、スイッチのオンオフを高速で切り替えることで、疑似的に電圧の上下を再現する方法。

例えば、1秒間点灯しっぱなしの時と、0.01秒ごとにオンとオフを繰り返す時を比べると、後者は半分の時間が実際にはオフになっているので、明るさも半分くらいに見える。

やり方

digitalWrite(pin番号, HIGH) となっているところを、analogWrite（pin番号、明るさを0~255の間の数値で指定）に書き換える。

全体の構成

準備編

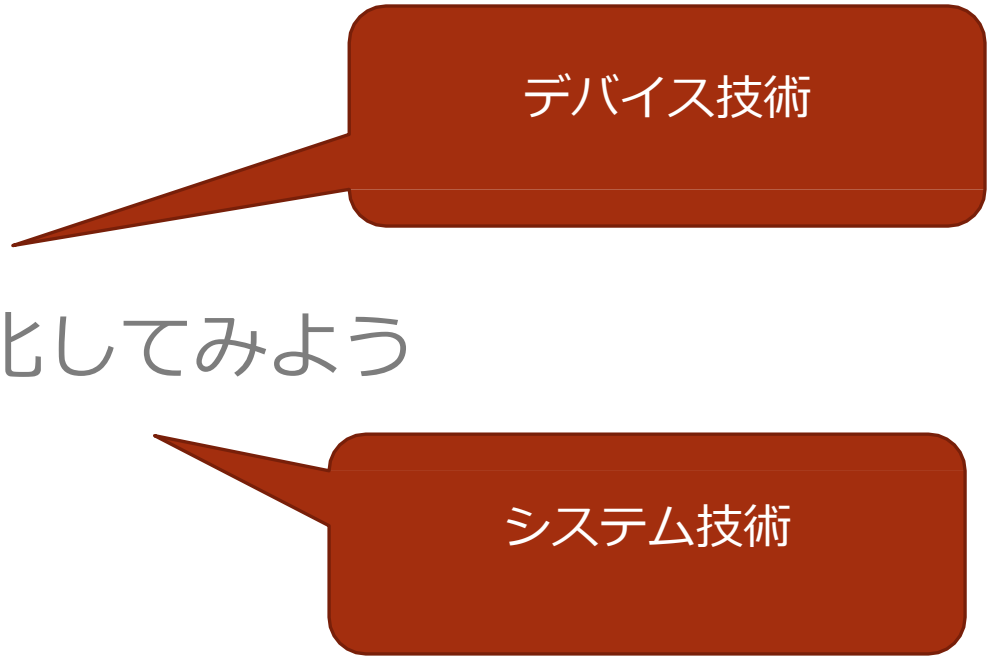
①Arduino IDEのインストール、環境設定

②Lチカをしてみよう

実践編

③温湿度の計測をしよう

④測った温湿度をグラフ化してみよう

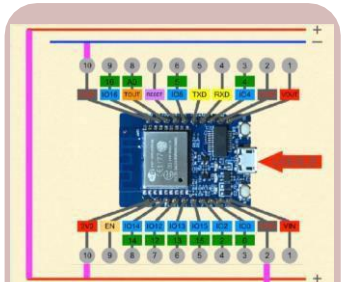


温湿度を測ろう！

⇒温湿度計から電気信号を受けとる！

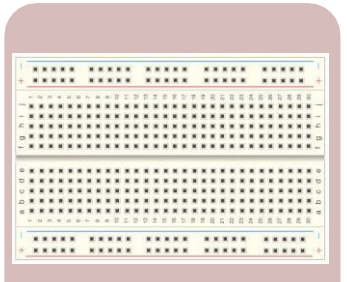
Arduino

- 温湿度の表示



開発ボード

- 温湿度計からの電気信号の翻訳



ブレッドボード

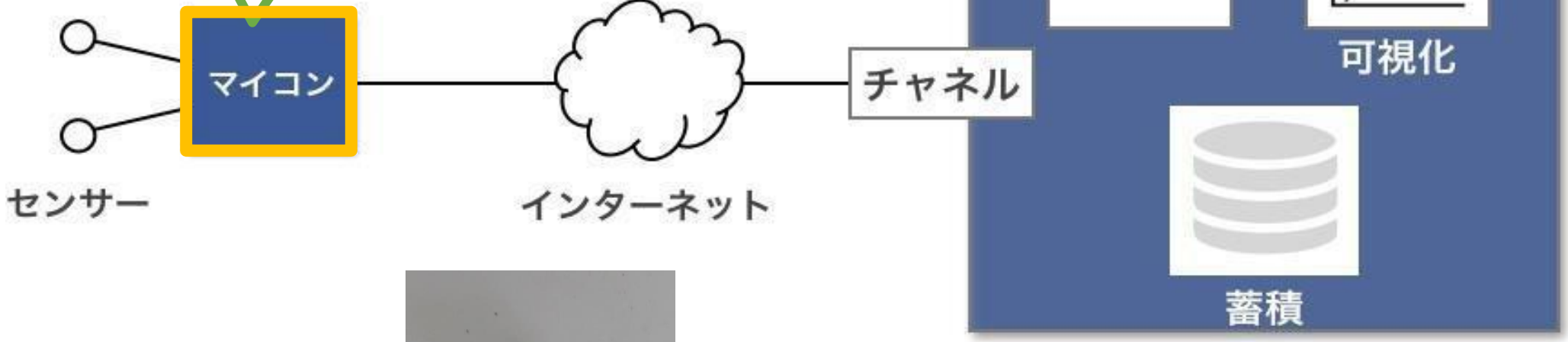
- 電気信号が通る道/場

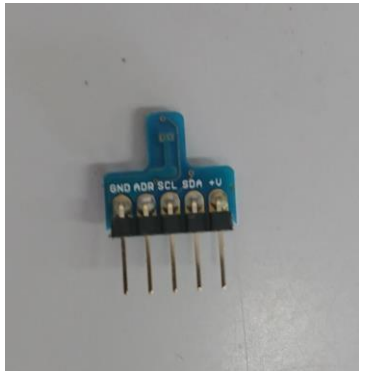
温湿度計

- 温湿度を測り電気信号にする

全体の構造

命令を処理する
（ESP-WROOM32,
switch science社）





温湿度センサー SHT31

Ambient社HPより抜粋

全体の構造

命令を処理する
（ESP-WROOM32,
switch science社）

Ambient

受信

可視化

蓄積

チャネル

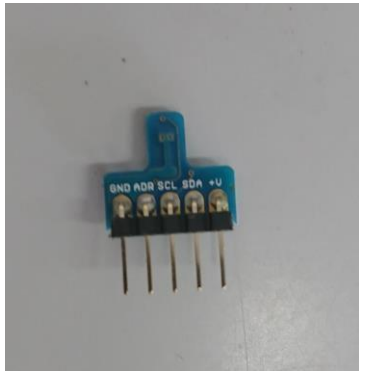
マイコン

データ

センサー

インターネット

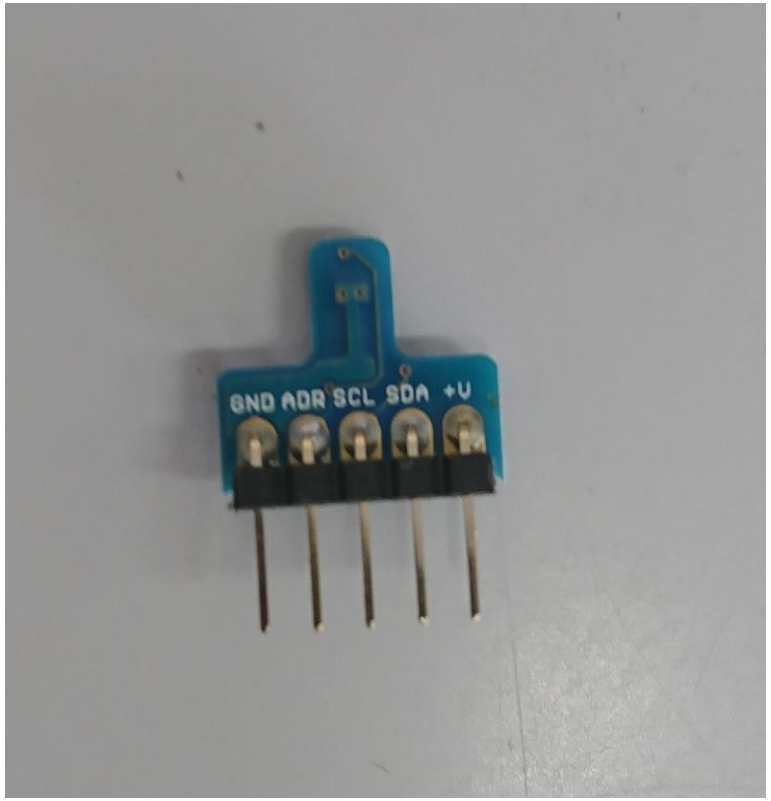
温湿度を計測



温湿度センサー SHT31

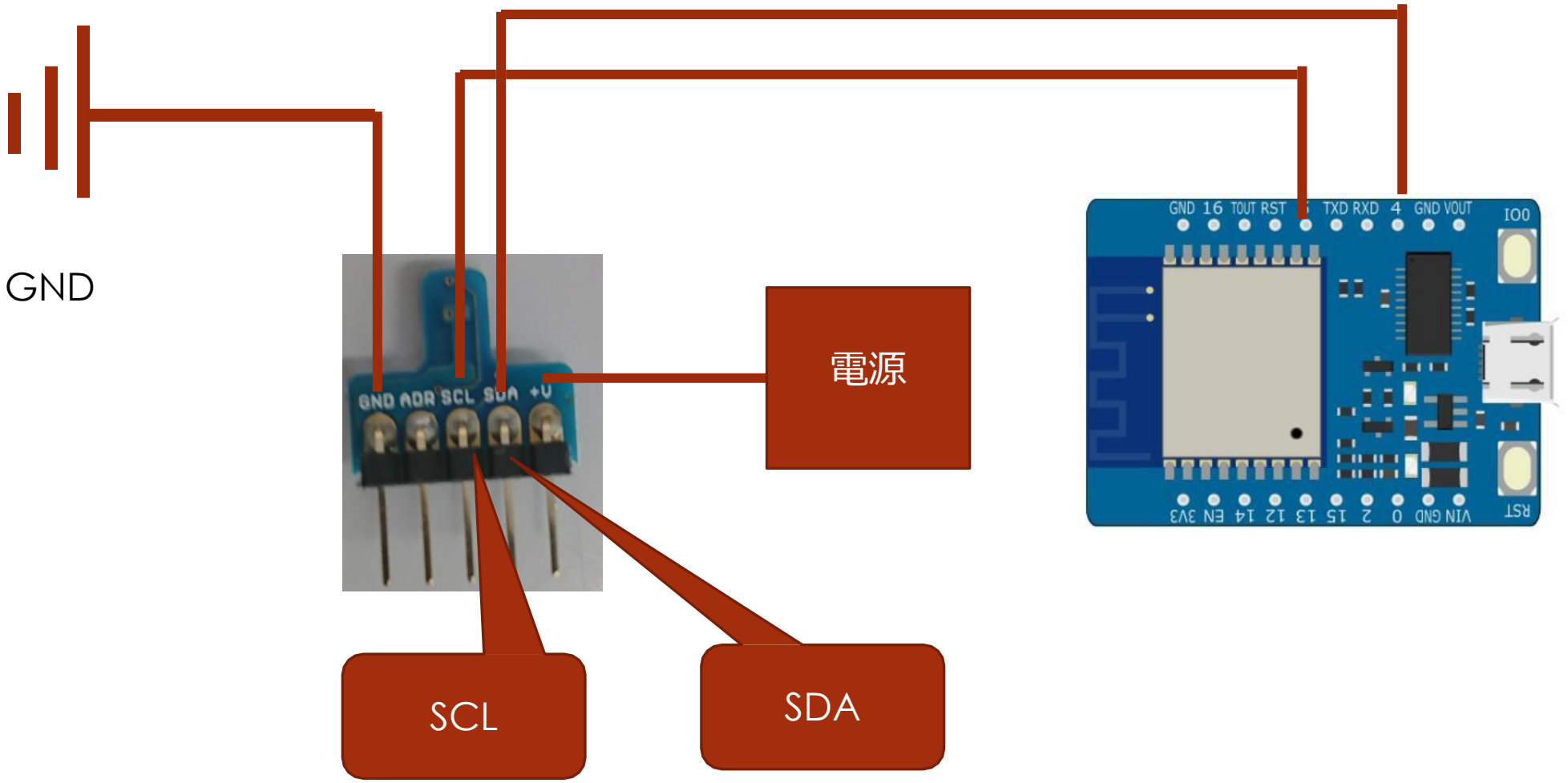
Ambient社HPより抜粋

温湿度センサーをつなげる



- SENSIRION社製
- 温度 -40℃～+125℃ 精度：±0.3℃（@0℃～90℃）
- 湿度 0%～100% 精度：±2%（@0℃～90℃）
- 価格 950円（秋月電子通商 http://akizukidenshi.com/catalog/g/gK-12125/）

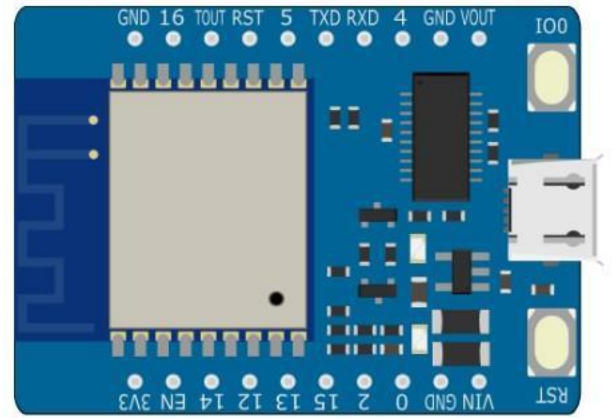
作成する回路


GND
GND ADR SCL SDA +V
電源
SCL
SDA
GND 16 TOUT RST TXD RXD 4 GND VOUT
IO0
RST

通信の仕方

信号２本で通信を行う SCL（シリアルクロック線）：同期をとるための信号線 SDA（シリアルデータ線）：SCLに同期してデータの転送に用いる信号線

SDA
（SCLがLowのときデータ変更）
（SCLがHighのときデータ転送）

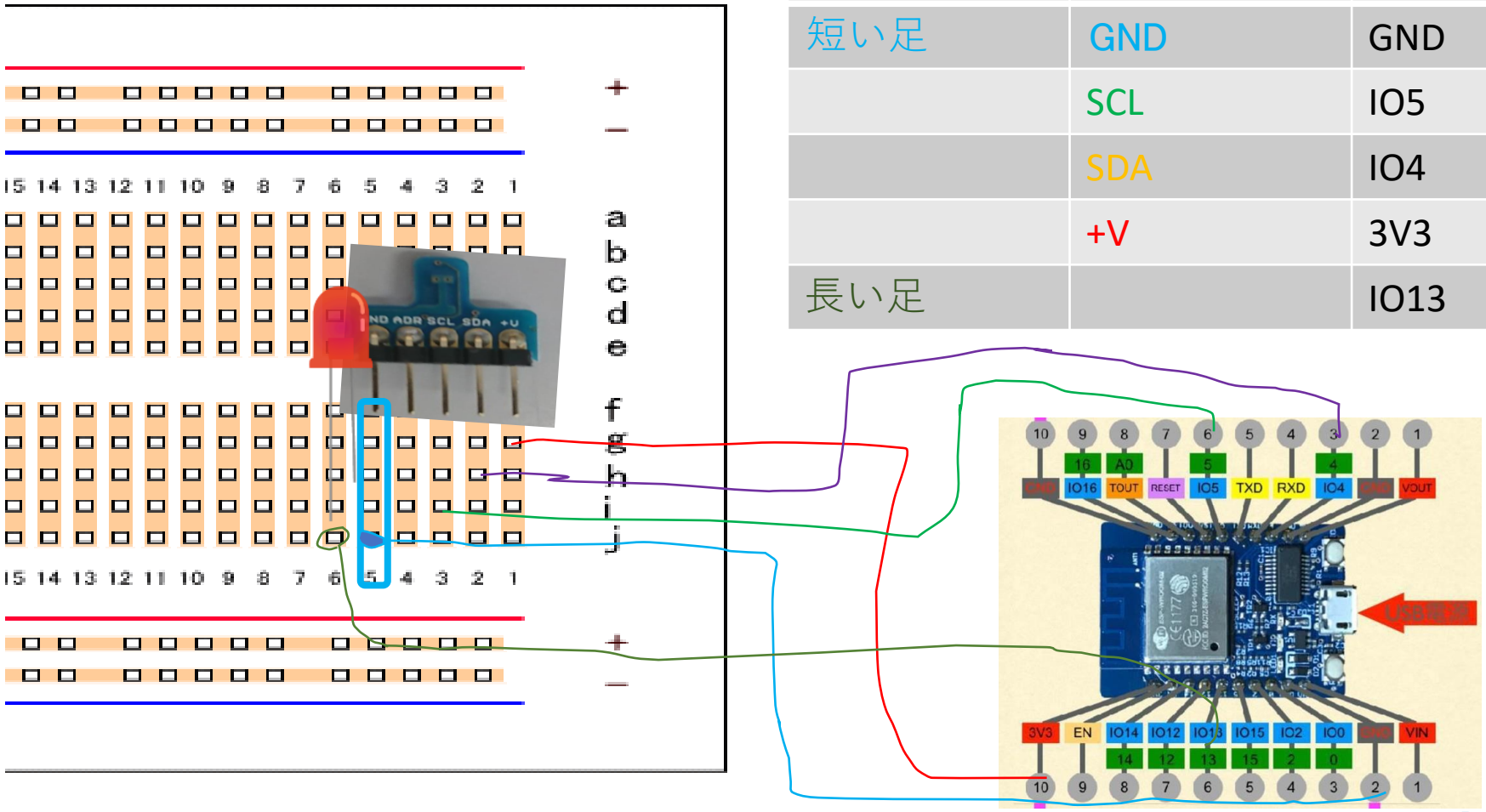


SCL
LowとHighの切り替えをする

準備：開発ボードとブレッドボードの配線

温度センサとArduinoのそれぞれのピンが以下のように繋がればOK！

LED	温度センサ	ラズパイ
短い足	GND	GND
	SCL	IO5
	SDA	IO4
	+V	3V3
長い足		IO13

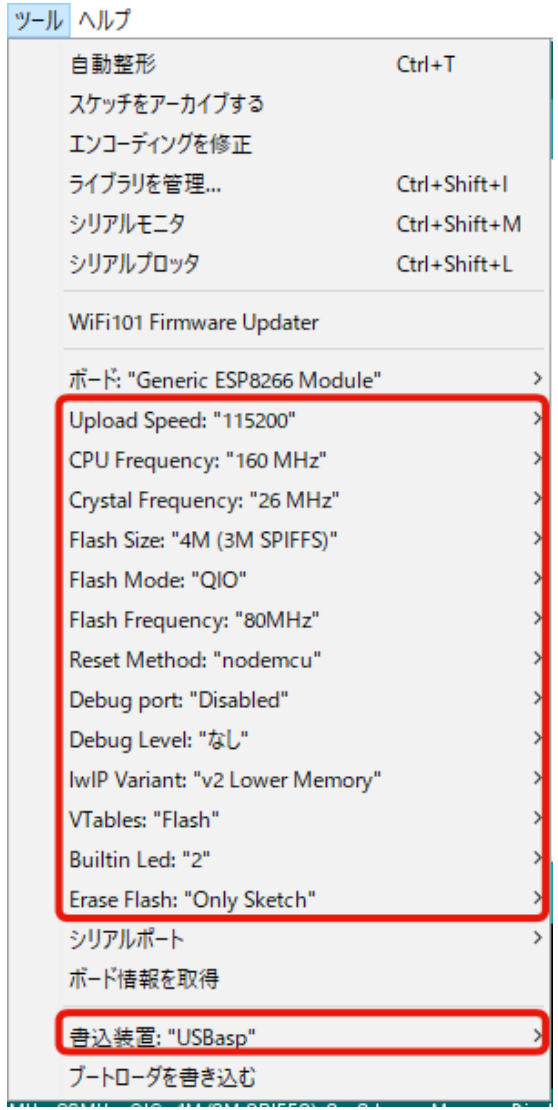


コンパイルと読込しよう

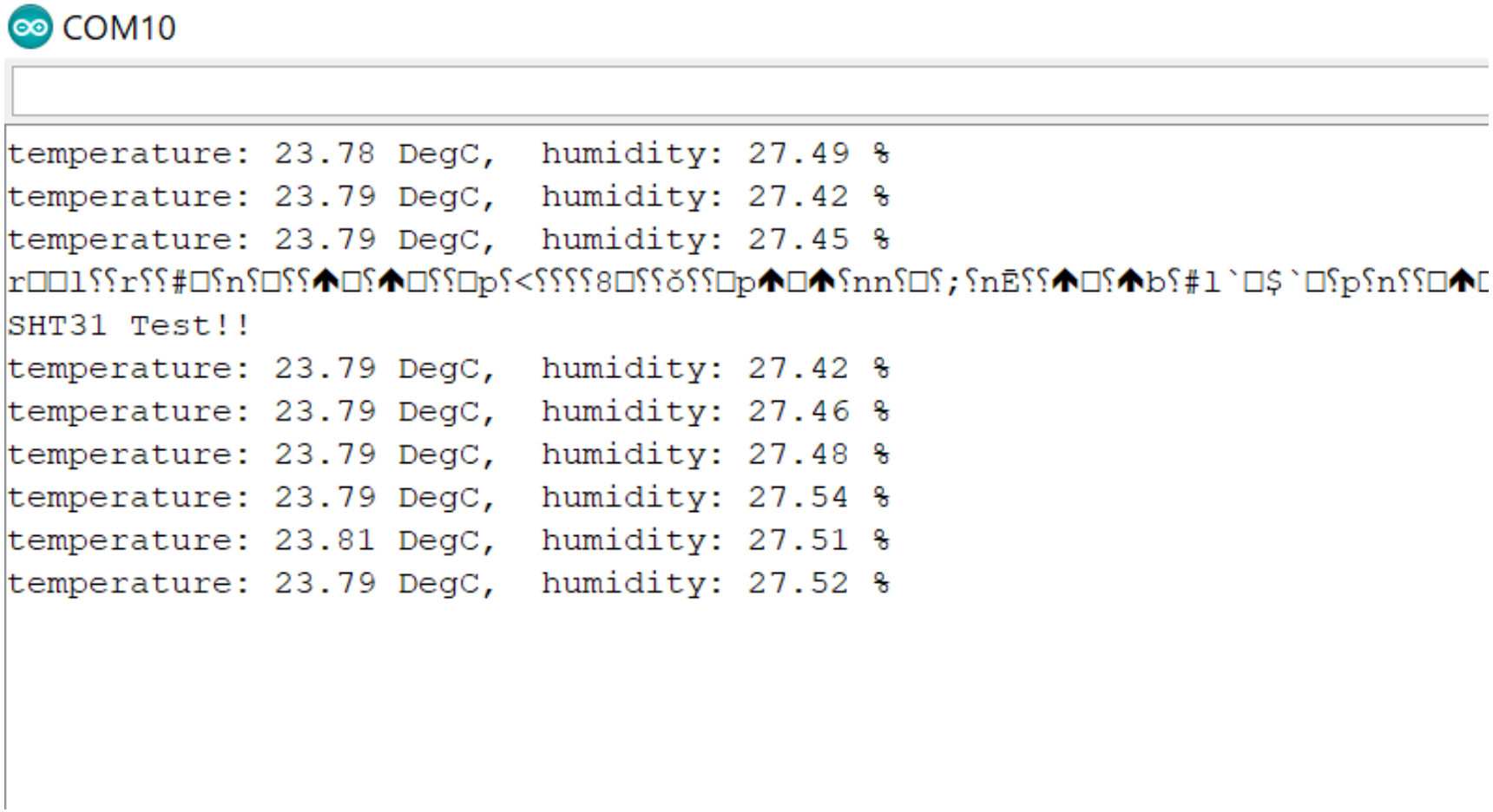
- [ソースコード]のIoT2_sht31を開きましょう。
- 矢印をクリックしてコンパイル＆読込をしましょう。

ツール設定の確認

- ツールを選択して、赤枠と同じ設定に変更する
- 設定がリセットされる可能性がある



温湿度測定のコンソール出力



全体の構成

準備編

①Arduino IDEのインストール、環境設定

②Lチカをしてみよう

実践編

③温湿度の計測をしよう

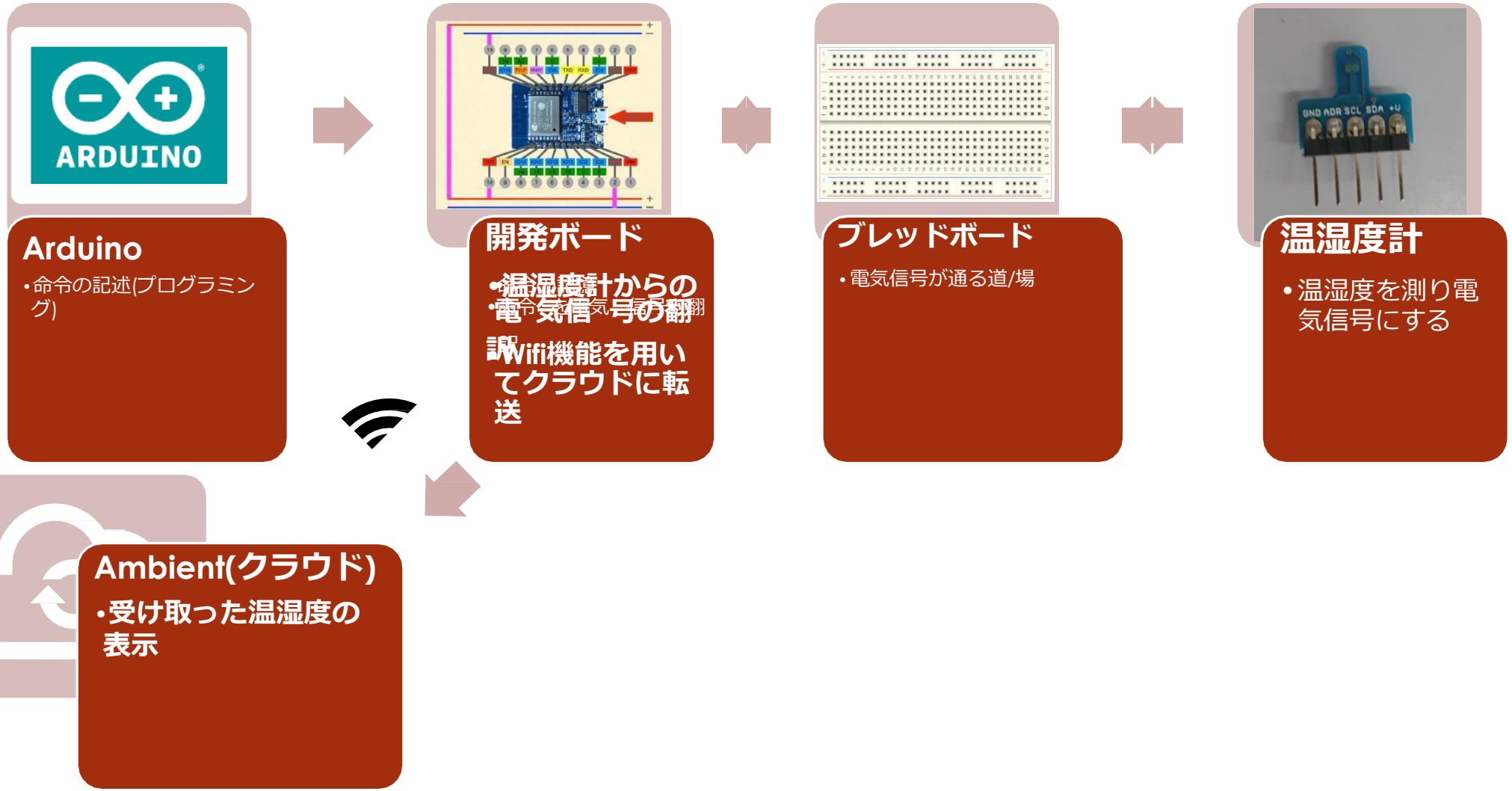
④測った温湿度をグラフ化してみよう

デバイス技術

システム技術

温湿度をクラウドにアップしよう！

⇒温湿度計から電気信号をクラウドに飛ばす！



全体の構造

命令を処理する
（ESP-WROOM32,
switch science社）

Ambient

受信

可視化

蓄積

チャネル

無線Wi-Fi

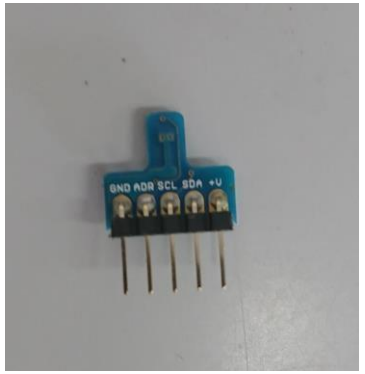
マイコン

データ

センサー

インターネット

温湿度を計測



温湿度センサー SHT31

クラウド（Ambient）上で
データを自動グラフ化

Ambient社HPより抜粋

Ambientの準備・設定

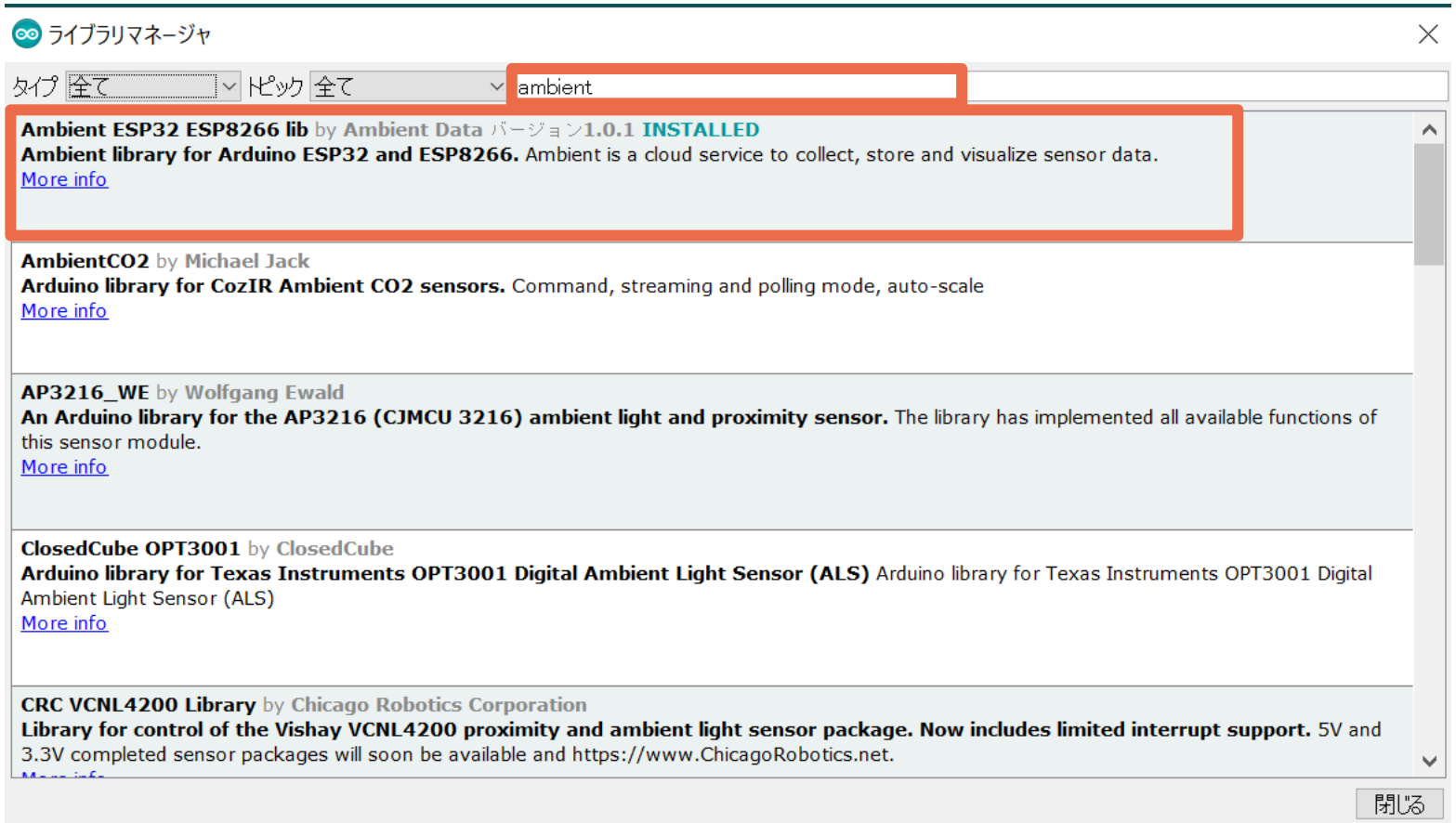
- ユーザ登録

https://ambidata.io/usr/signup.html

チャネルを作り、チャネルＩＤとライトキーを得る

Ambientライブラリのインストール

- 「ツール」メニューの「ライブラリを管理...」を選択し、ライブラリマネージャ
- 検索窓に、'ambient'を入力し、表示された「Ambient ESP32 ESP8266 lib」をインストールする



コードの変更

IoT3_SHT31_CLOUD | Arduino 1.8.10

ファイル 編集 スケッチ ツール ヘルプ

IoT3_SHT31_CLOUD

```
#include <ESP8266WiFi.h>
#include "Ambient.h"
#include "AE_SHT31.h"

#define LED 13
#define SDA 4
#define SCL 5


#define PERIOD 30

// ---------- customize here ----------
// WiFi Connection
const char* ssid = "wifiのSSIDに変更する";
const char* password = "wifiのパスワードに変更する";

// Ambient Channel Info
unsigned int channelId = チャネルIDに変更;
const char* writeKey = "writekeyに変更";
// ------------------------------------

WiFiClient client;
Ambient ambient;
// SHT31のアドレスを設定
AE_SHT31 SHT31 = AE_SHT31(0x45, SDA, SCL);
```

Ambientソースコードについて

- IoT3_SHT31_CLOUD を読み込む
- コンパイル＆書き込みボタンを押し、プログラミングを読み込む

最終的に得られるグラフ