ガスにゃんのプログラミングチュートリアル

東京工業高等専門学校 専攻科
電気電子工学専攻 山田恭平

本体のマイコン ESP8266 にプログラムを書き込む手順を解説します．開発環境のセットアップから解説しますので，開発経験があれば適宜読み飛ばしてください．作業環境は Windows 10 です．

Arduino IDE のインストール

以下の URL にアクセスして Arduino IDE のインストーラを入手します．
Windows Installer をクリックします．
https://www.arduino.cc/en/Main/Software

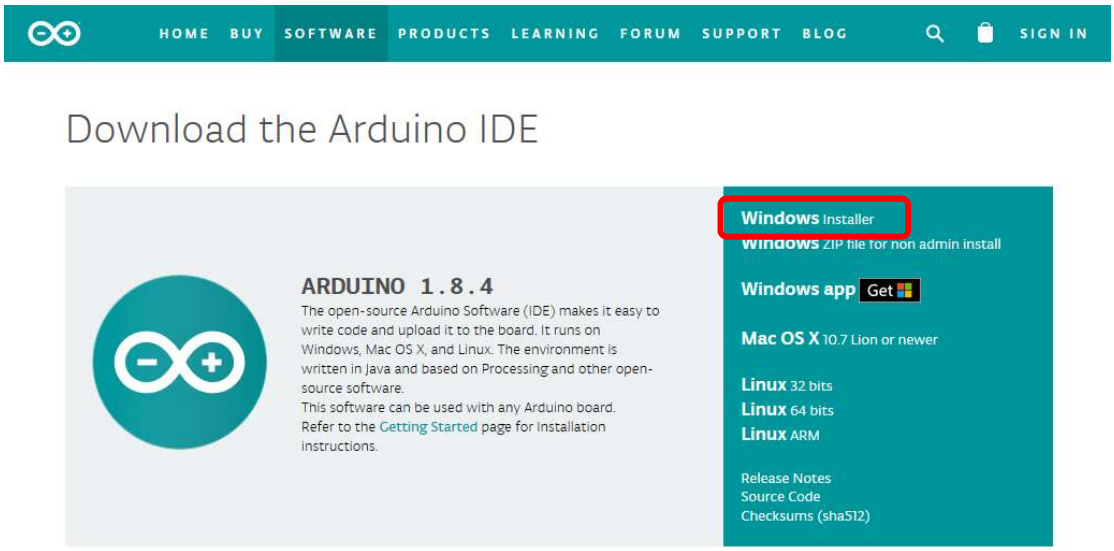


図 1　Arduino IDE ダウンロードページ

寄付(contribution)を募る画面が表示されます．寄付をしないのであれば just download をクリックするとダウンロードが始まります．ダウンロードされるファイルを実行します．

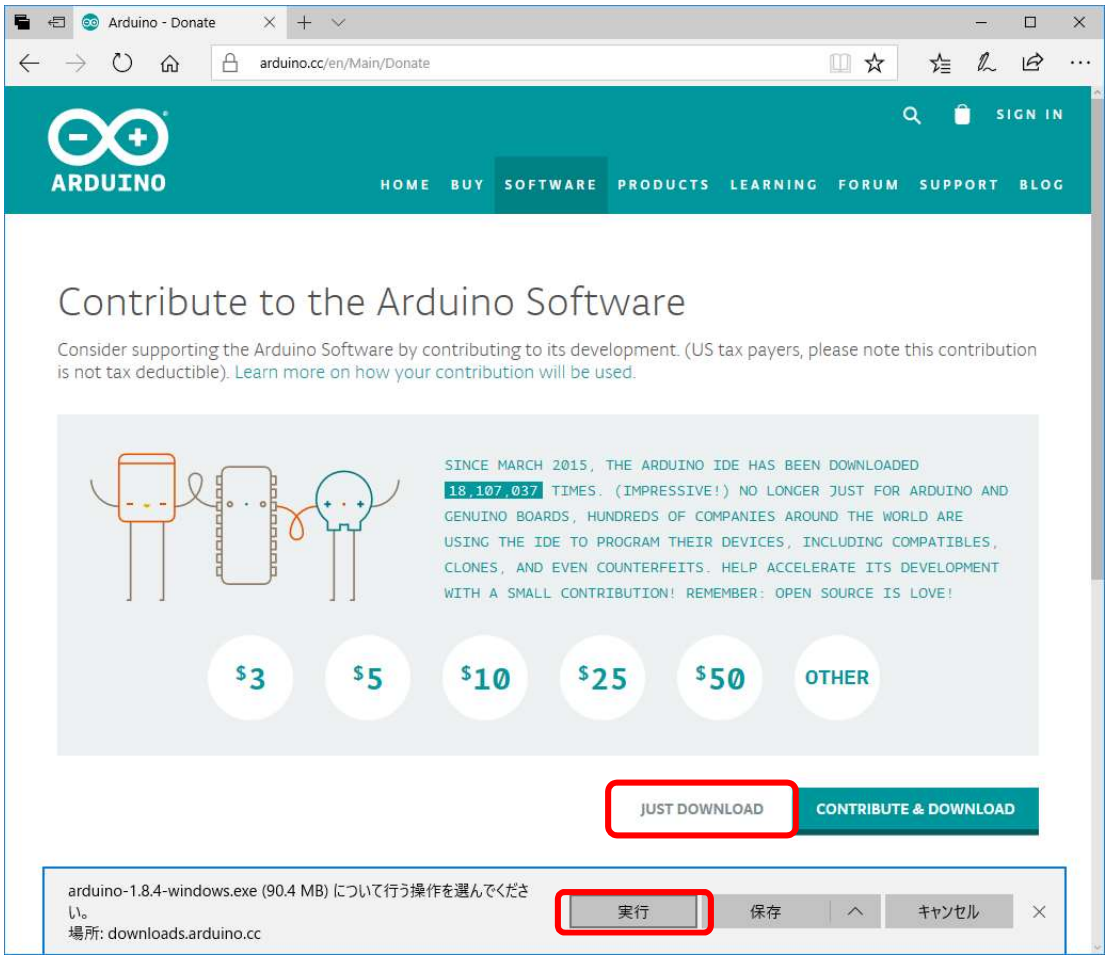


図 2　Arduino IDE ダウンロード方法

ライセンスに同意します.

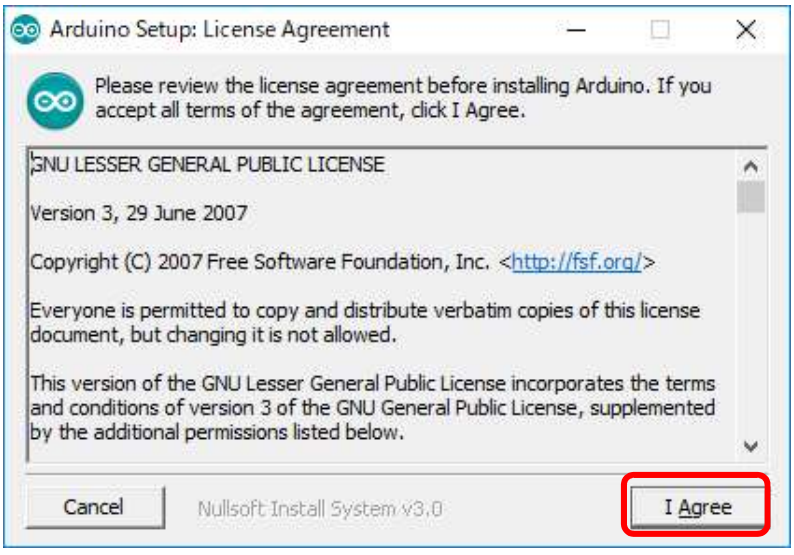


図 3　Arduino IDE ライセンス同意

インストールオプションすべてにチェックをつけて次へ進みます.

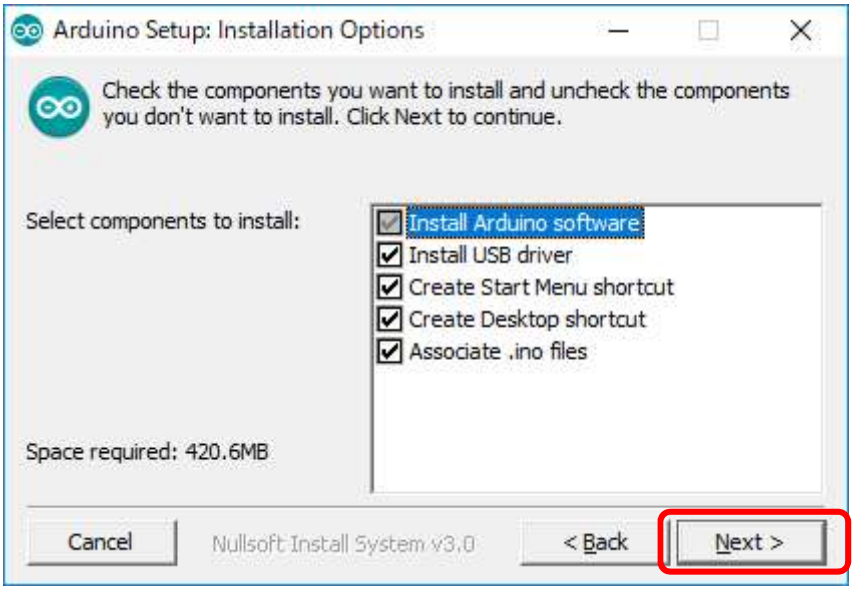


図 4　インストールオプション

特にこだわりがなければ，インストールディレクトリはそのままでインストールを実行します.

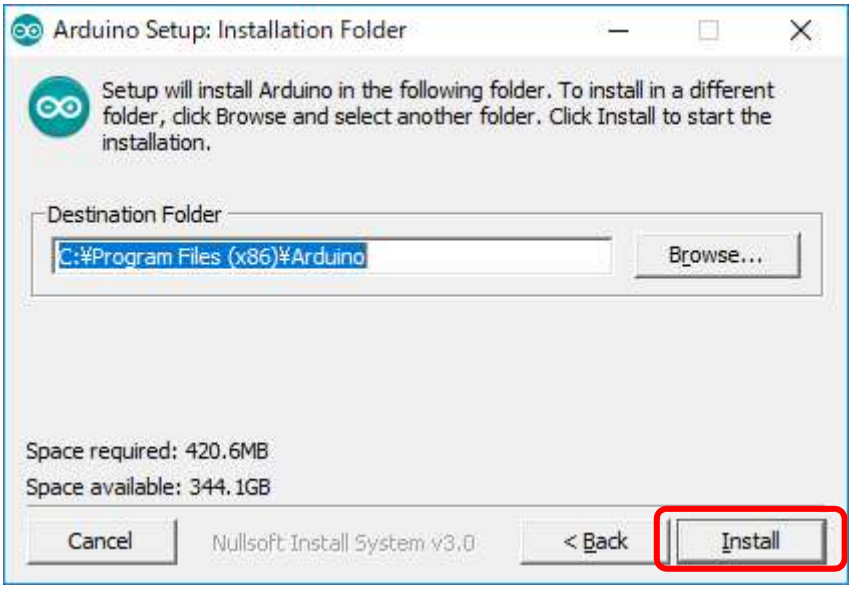


図 5　インストールディレクトリの指定

インストールの途中でドライバのインストールに同意が求められるので，同意します．
完了したら close します．

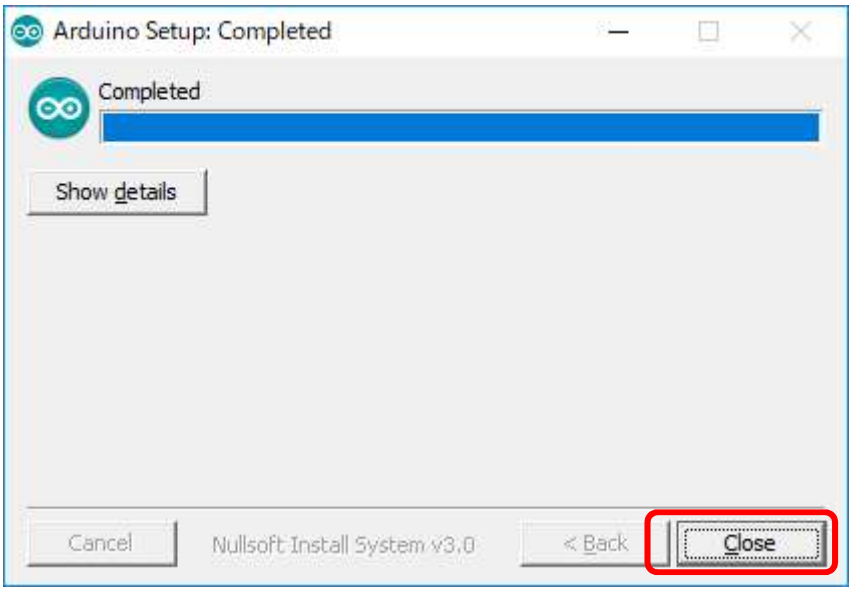


図 6　Arduino IDE インストールの完了

ESP8266 ボードの追加

初期状態の Arduino IDE では ESP8266 の開発環境が用意されていないので，追加する作業を行います．まず，インストールした Arduino IDE を起動します．
メニューバーから次のように選択して，ボードマネージャを表示します．

“ツール” → “ボード” → “ボードマネージャ...”

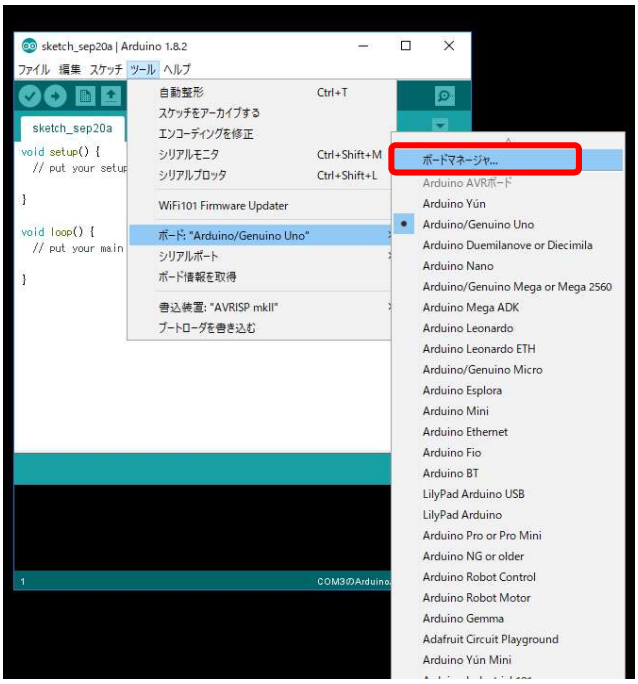


図 7　ボードマネージャの起動

ボードマネージャで次頁の図 8 のように実行します

1．上部の検索ボックスに“esp8266”と入力する．
2．“esp8266 by ESP8266 Community” の最新版を選択肢する
3．インストール
4．インストールが終了したら，閉じる

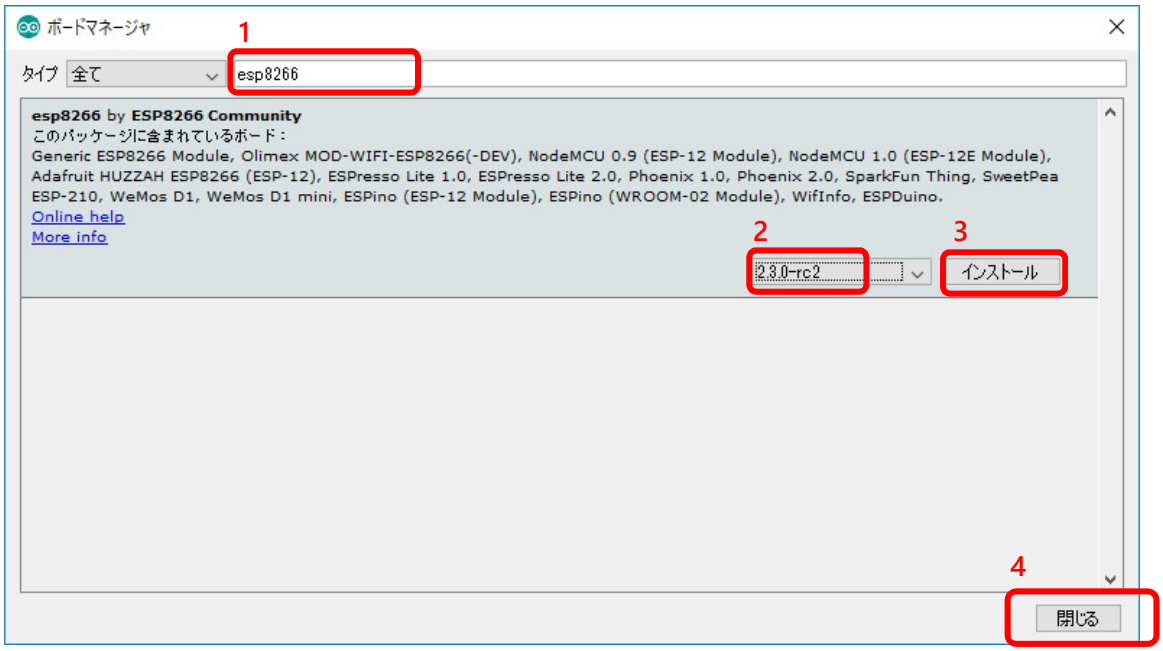


図 8 ボードマネージャの操作

ボードが追加されたのでメニューバーから次のようにボードの設定を変更します.

“ツール” → “ボード” → “Generic ESP8266 Module”

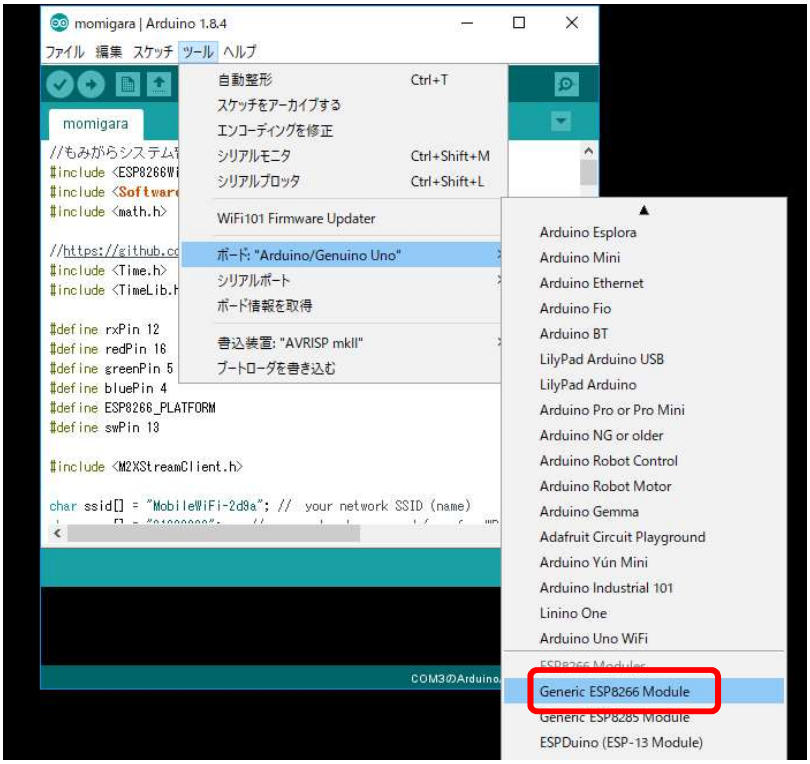


図 9　ボード設定の変更

ライブラリのインストール

ArduinoIDE に，必要なライブラリ 2 つをインストールする．

- 温湿度センサ（AM2320）用ライブラリ
- 空気質センサ（MQ135）補正自作ライブラリ

Arduino IDE を起動し，メニューバーから次のように選択して，ライブラリマネージャを表示する．

“スケッチ” → “ライブラリをインクルード” → “ライブラリを管理...”

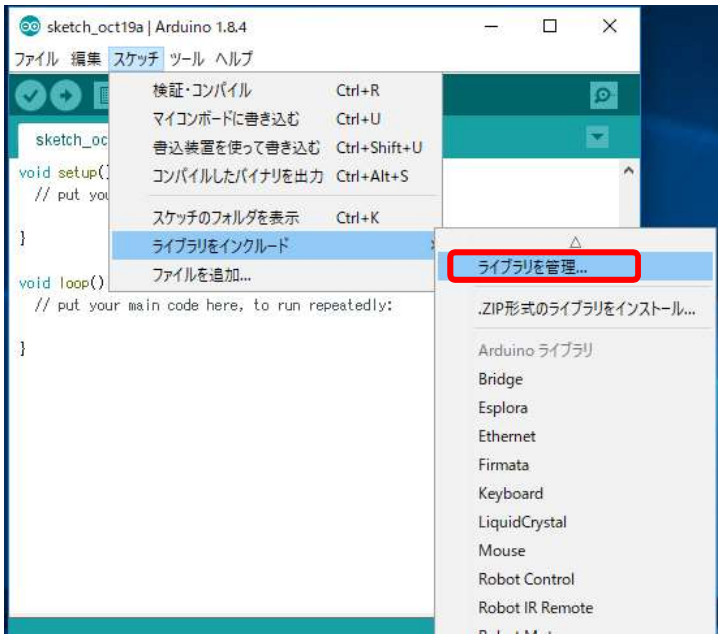


図 10　ライブラリマネージャの起動

ライブラリマネージャで次のように実行します

1．上部の検索ボックスに“dht”と入力する
2．“DHT sensor library”の最新版を選択肢する
3．インストール
4．インストールが終了したら，閉じる

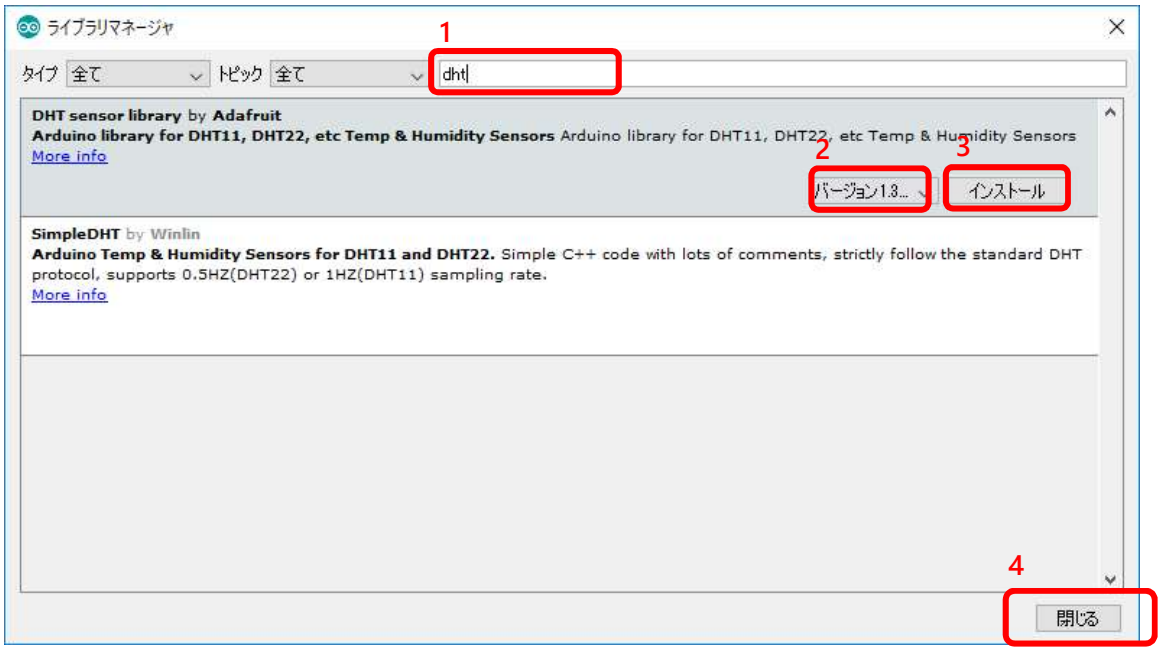


図 10 ライブラリマネージャの操作

次に，MQ135 用ライブラリを読み込みます．こちらのサイトで公開されている zip ファイル”MQ135_NNCO2Estimator.zip”をダウンロードしてください．
http://mitolab.sakura.ne.jp/mitoWiKi/index.php?title=MQ135

ダウンロードしたら, Arduino IDE で.ZIP 形式のライブラリをインストールを選択します.

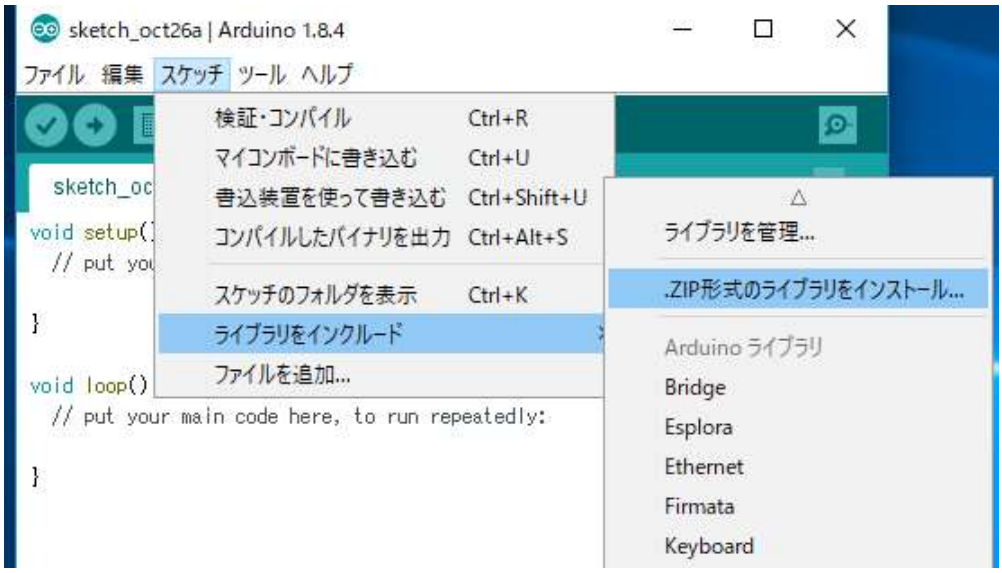


図 11　ZIP 形式ライブラリのインストーラの起動

ダウンロードした”MQ135_NNCO2Estimator.zip”を開くとインストールされます.

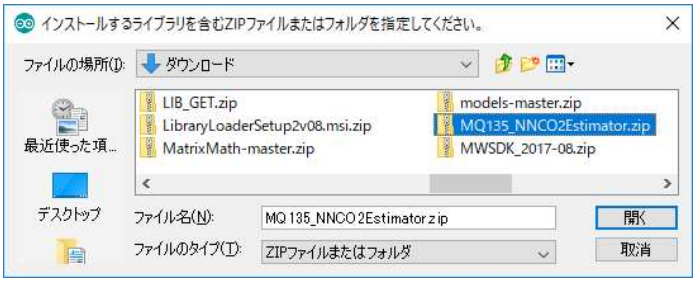


図 12 インストールライブラリの選択

インストールされたライブラリには，サンプルプログラムが付属しています．これを開くにはスケッチ例の MQ135 NN CO2 estimator の GasNyan_sample を選択します.

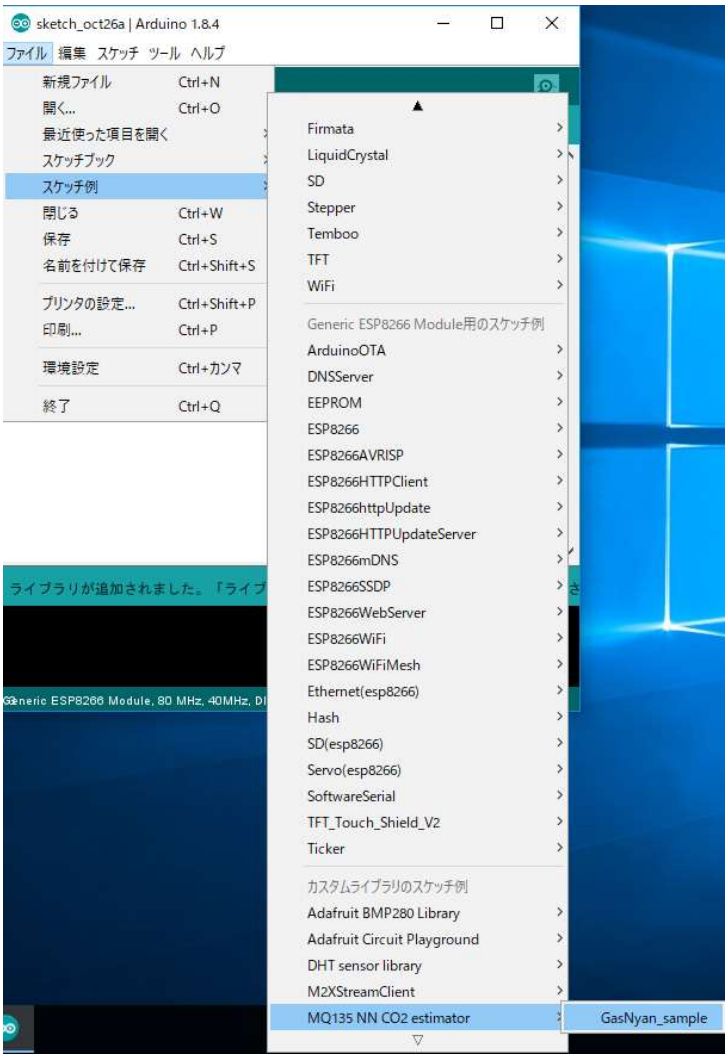


図 13　サンプルプログラムの起動

書き込み

本体底面の書き込みピンからシリアル通信することで書き込みます．書き込み時の電源は，通常動作時と同じ USB 電源を必要とします．したがって，次のような配線を行います．

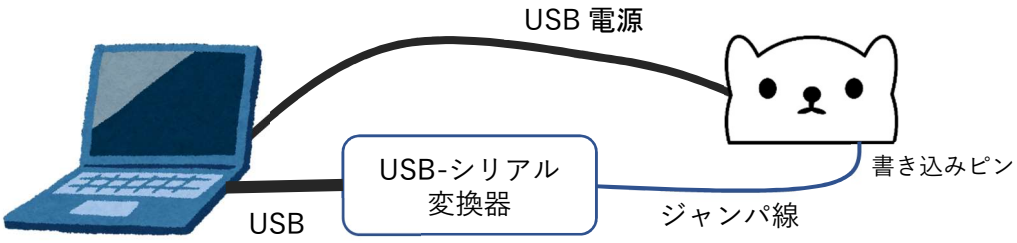


図 14　プログラム書き込み構成

具体的な作業手順は以下のとおりです．

(1) USB-シリアル変換器を PC に接続し，ドライバをインストール
(2) Arduino IDE で書き込みたいプログラムを開く
(3) シリアルポートの設定
(4) 本体底面の書き込みピンに接続
(5) モード切替ピンにより，書き込みモードに設定
(6) 電源を接続
(7) 書き込み
(8) モード切替ピンにより，動作モードに設定

順に解説します．

（1）USB-シリアル変換器を PC に接続し，ドライバをインストール

書き込みに用いる USB-シリアル変換器は特に指定はないので，各自用意してください．変換器によってドライバのインストール方法が異なるため，変換 IC の印字などから情報を得てインストールしてください．

（2）Arduino IDE で書き込みたいプログラムを開く

ここでは，サンプルプログラム GasNyan_sample を使います．前頁で示した方法で開いてください．

（3）シリアルポートの設定

(1)で接続したシリアル変換器の番号に設定します．書き込みがうまくできなければ，この項目を変更してみてください．

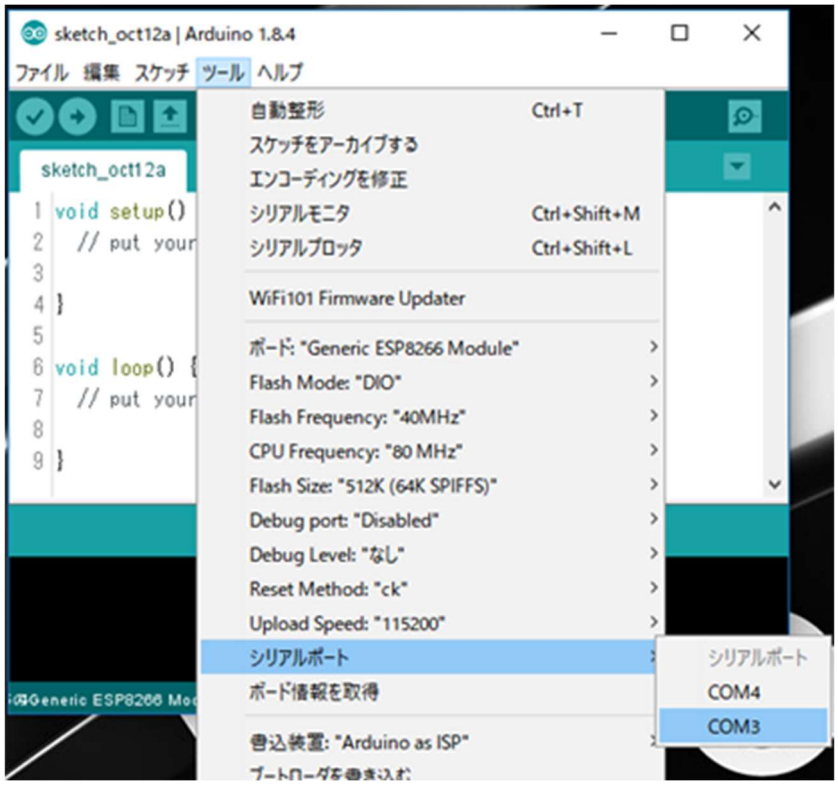


図 15　シリアルポートの設定方法

（4）本体底面の書き込みピンに接続

写真のようにシリアル変換器を接続します.

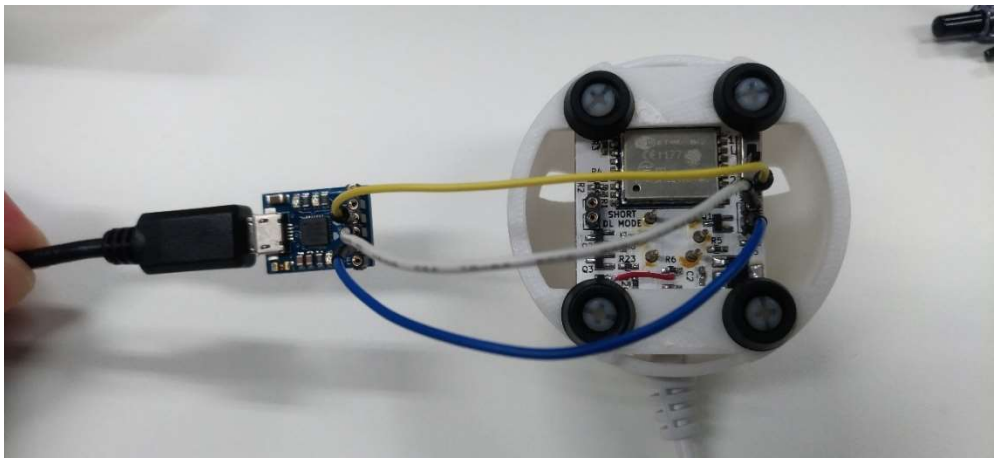


図 16　シリアル変換器との接続

変換器の送信(Tx) – デバイスの受信(Rx)

変換器の受信(Rx) – デバイスの送信(Tx)

変換器の GND – デバイスの GND

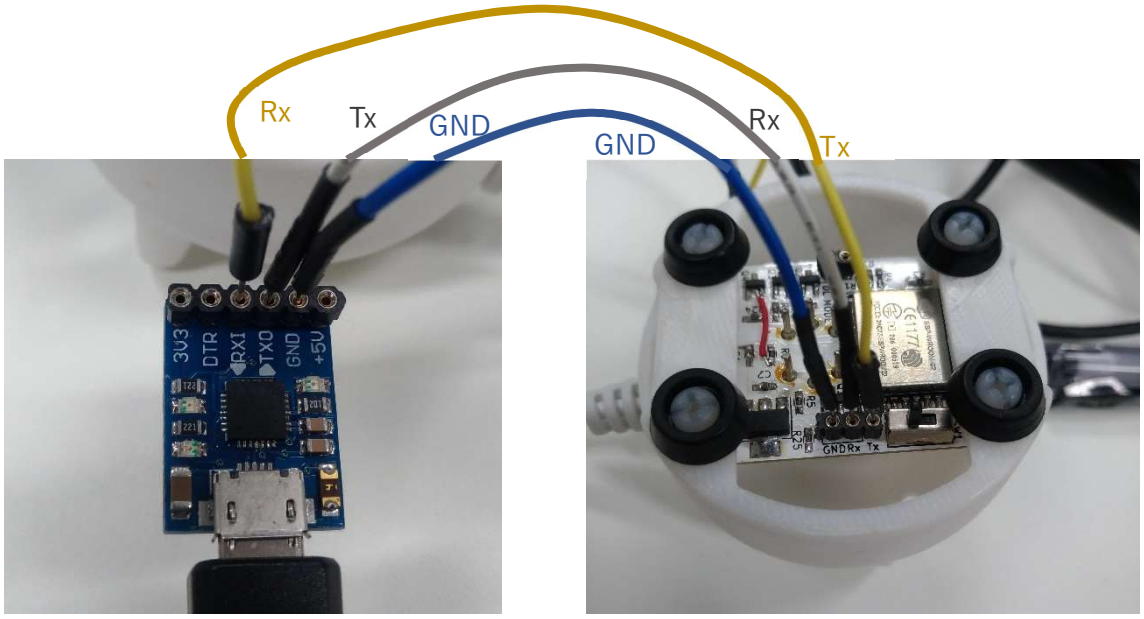


図 17　シリアル通信ピンの配置

（5）モード切替ピンにより，書き込みモードに設定

基板上に「SHORT DL MODE」と記されたピンがあるので，このピンをショートさせて書き込みモードに設定します．ESP は起動時に IO0 がプルアップされていれば通常起動モード(Flash boot mode)で動作し，IO0 がプルダウンされている場合に書き込みモード(UART download mode)で動作します．本基板では，写真左方向の 2 ピンをショートさせると書き込みモードの設定となります．

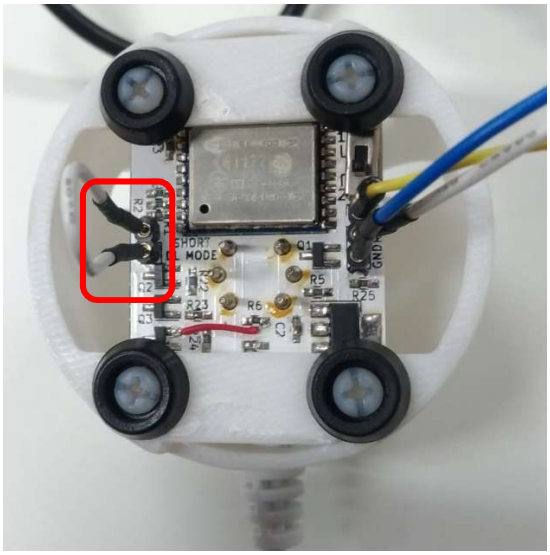


図 18　ジャンパピンによる書き込みモードへの切り替え

（6）電源を接続

この状態で電源を接続すると通常動作はせず，書き込み待機状態となります．書き込み待機状態では，LED の全色が点灯するため本体が白く発光します．

図 19 電源の接続

（7）書き込み

Arduino IDE 上で，左上の「→」ボタンで書き込みが実行されます．

図 20 書き込みの実行

書き込みが成功すると，デバイスは動作を開始します．

（8）モード切替ピンにより，動作モードに設定

書き込みが完了したら，デバイス底面に接続したジャンパ線をすべて外します．

ライブラリの解説

本ライブラリは，

- MQ135，温度，湿度センサの過去 24 時間のログを蓄積
- 蓄積したセンサログデータから現在の二酸化炭素濃度を推測

の，大きく 2 つの機能を持つクラス MQ135_NNCO2Estimator を提供します．

プログラムでの使用方法を解説します．まず，setup()や loop()の外でインスタンスを生成します．インスタンス名は任意ですが，ここではmq135 として説明します．

```
MQ135_NNCO2Estimator mq135;
```

生成されるインスタンスの機能の模式図を図 21 に示します．
(1),(2),(3)の関数でセンサのデータを逐次，入力します．
すると自動でログデータを管理し，過去 24 時間の平均値が計算されます．
(4)の関数を使うと，最新のデータを入力した時点の二酸化炭素を推測して，戻り値として出力します．
つまり，ユーザは(1),(2),(3),(4)の関数を使うのみで，二酸化炭素濃度を取得できます．

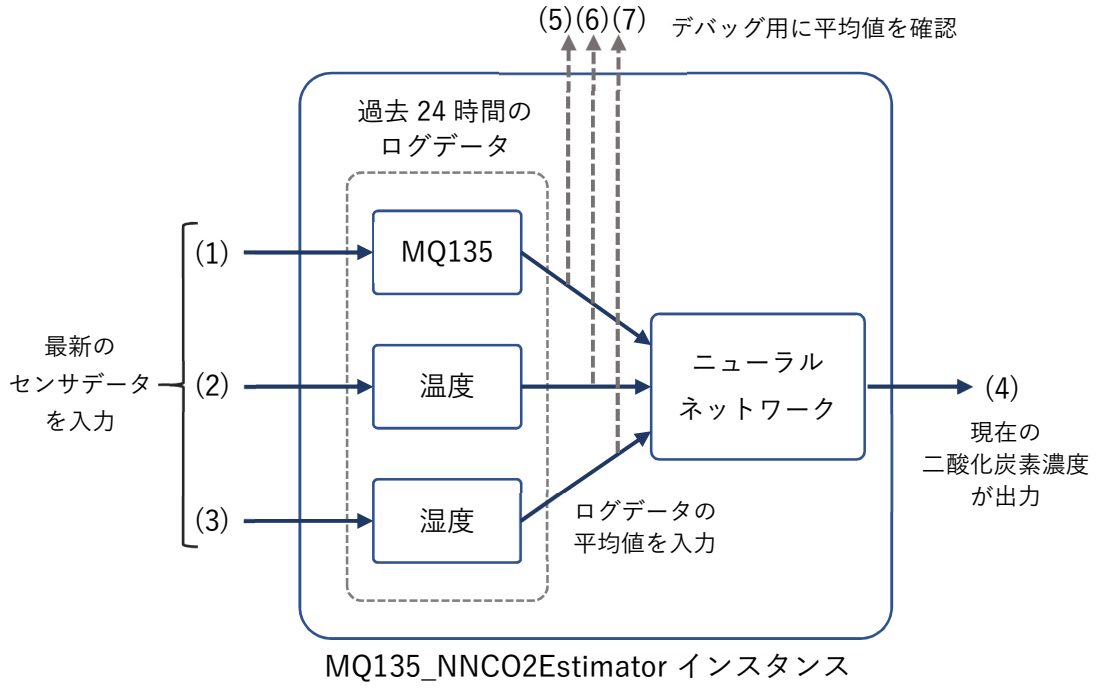


図 21　ライブラリの内部処理

インスタンス名を mq135 とすると，(1)-(7)の処理は次の関数で実行されます

(1) mq135.adc.update(float datum);

引数：float datum　MQ135 の最新の adc 値

戻り値：true(成功)，false(失敗)

(2) mq135.temperature.update(float datum);

引数：float datum　最新の温度値[℃]

戻り値：true(成功)，false(失敗)

(3) mq135.humidity.update(float datum);

引数：float datum　最新の相対湿度[%]

戻り値：true(成功)，false(失敗)

(4) mq135.estimate(void);

戻り値：float 最新の二酸化炭素濃度の推測結果

以下はデバッグ用に，最新データから過去 x 秒間の平均値を計算する関数です．0.1 秒から 24 時間(86400 秒)まで対応しています．移動平均を知りたいときに便利です．

(5) mq135.adc.secAverage (float timeRangeSec);

引数：float timeRangeSec 平均値の範囲[s]

戻り値：float 指定範囲の平均値

(6) mq135. temperature.secAverage (float timeRangeSec);

引数：float timeRangeSec 平均値の範囲[s]

戻り値：float 指定範囲の温度平均値[℃]

(7) mq135. humidity.secAverage (float timeRangeSec);

引数：float timeRangeSec 平均値の範囲[s]

戻り値：float 指定範囲の湿度平均値[%]

API の仕様に要望があれば連絡ください．できるだけ対応します．

回路の仕様

本デバイスには，入力用の素子として

- 温湿度センサ AM2320
- 空気質センサ MQ135
- スライドスイッチ

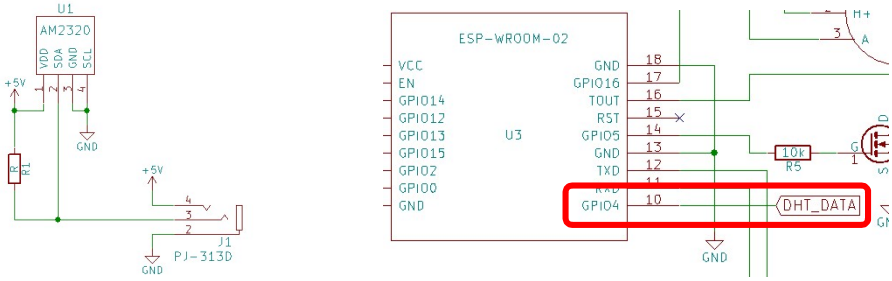
出力用の素子として

- RGB LED 5 個

が搭載されています．それぞれの実装と，プログラムからの利用方法ついて解説します．

温湿度センサ AM2320

デバイスの尻尾部分に配置され，通信ピンが ESP-WROOM-02 の GPIO4 に接続されています．このセンサは，SCL ピンの接続により通信方式を I²C もしくはシリアル通信を設定しますが，SCL ピンを GND に接続することでシリアル通信に設定しています．



(a) 尻尾部の基板　　(b) マザーボード

図 22　温湿度センサの回路

空気質センサ MQ135

MQ135 はガス濃度によって抵抗値が変化するセンサです．本デバイスでは，MQ135 と抵抗で 3.3V を分圧し，抵抗の両端電圧を ESP-WROOM-02 のアナログ入力ピン TOUT に入力しています．ESP-WROOM-02 のアナログ入力電圧の範囲が 0-1.0V なので[1]，使用域でこれを超過しないよう，分圧抵抗に 10kΩ を採用しています．また，ヒーターの加熱をプログラムから ON/OFF できるように N-ch FET を搭載しています．GIPO5 を HIGH にすると加熱され，LOW にすると加熱されないという動作です．

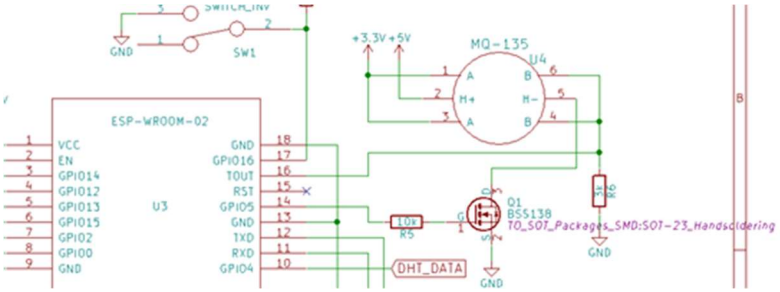


図 23　MQ135 の回路

スライドスイッチ

GPIO16 にスライドスイッチが接続されています．現状のサンプルプログラムでは使用しておらず，将来のために拡張性を持たせたものです．アプリケーション開発者が任意の機能を割り当てることを想定していて，用途としては動作モードの切替などが考えられます．

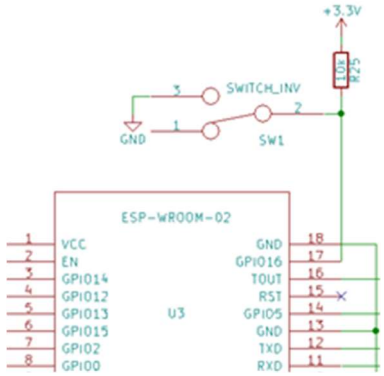


図 24　スライドスイッチの回路

RGB LED

5 個搭載された RGB LED は同じ色ごとに並列接続され，FET で色ごとにスイッチングします．GPIO12-14 を HIGH にすると，対応する色が発光します．スイッチング用の出力を PWM にすることで，フルカラーを表現します．回路図での RGB の接続表記に誤りがあり，各色に対応するスイッチング用の端子は以下の通りです．

赤: GPIO12,　　緑: GPIO14,　　青: GPIO13

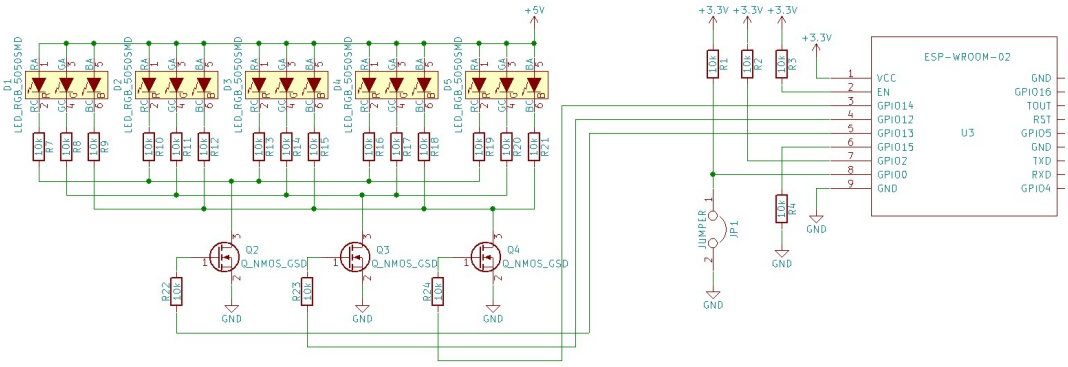
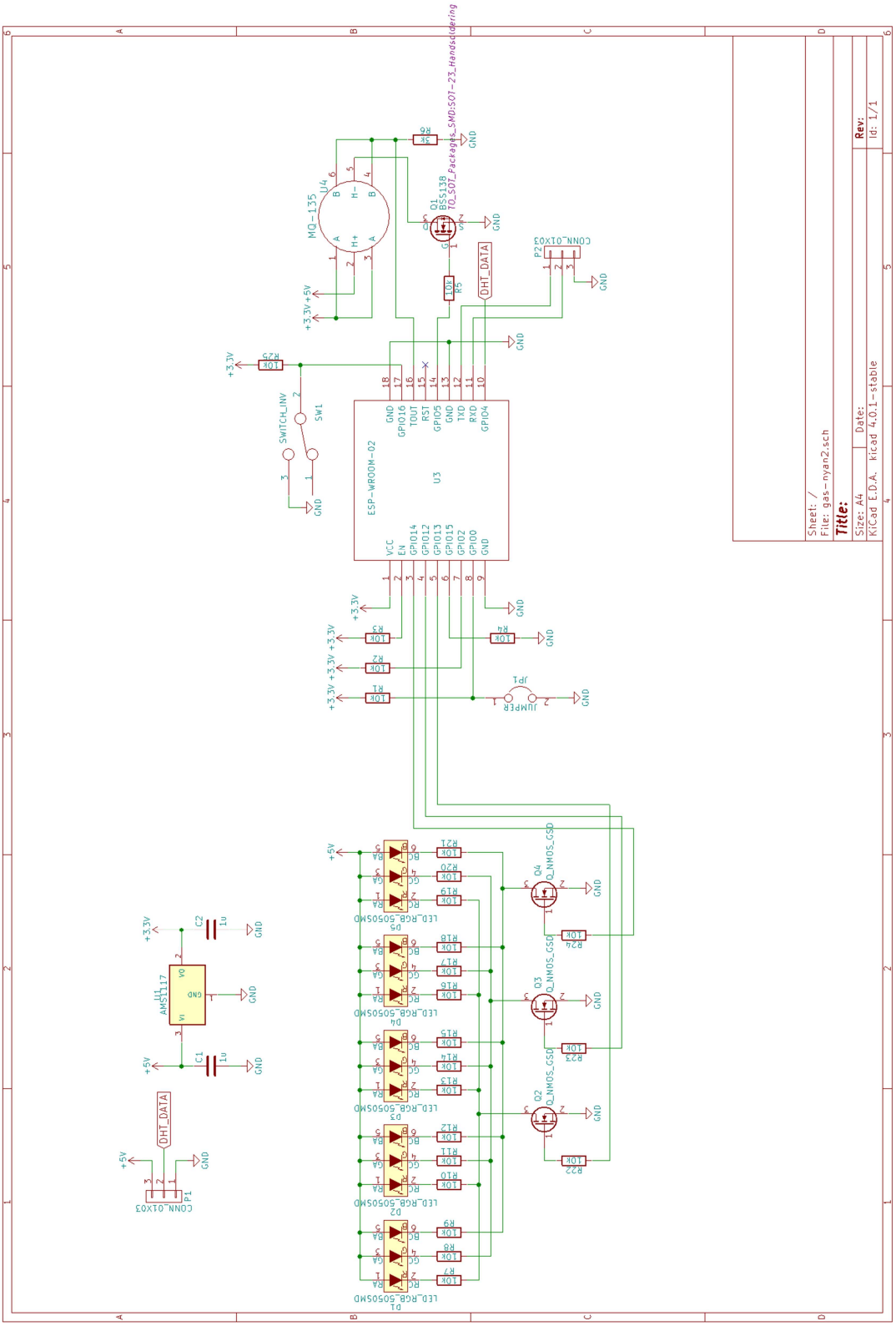


図 25　LED 部の回路

参考資料

[1] Espressif Systems IOT Team, ESP8266EX Datasheet Version 4.4, 2015

全体の回路図を以下に示します.



CO2 計測について補足

精度について

精度について十分な検証ができていませんが，現状±50%くらいの誤差だと思います．精度の検証には，機械工学科清水先生からお借りしている CO_2 ロガー(MCH-383SD)が使用可能です．

予熱について

本デバイスを起動して数時間は，センサの出力不安定でが信頼できるデータは取得できません．12 時間ぐらい動作させるとセンサの出力が安定するようです．したがって，計測したい前日の夕方くらいに設置する必要があります．

ログデータの保持について

ログデータから現在の CO_2 濃度を推測していますが，このデータは電源を切ると消えてしまいます．なので，何かのトラブルでリセットした場合，計測がしばらく途絶えてしまいます．もし実用上この仕様が致命的であれば，改善案を募集します．

通信・サーバの仕様案

- サーバへ送信する内容
  デバイスは以下の項目を，一定間隔（1 分?）で http post するようにする．

post パラメータ	内容	データ形式	送信例
co2	二酸化炭素濃度の平均	10 進数の文字	co2=530
temp	温度の平均	10 進数の文字	temp=23.6
hum	湿度の平均	10 進数の文字	hum=48.7
adc	MQ135 の ADC の平均	10 進数の文字	adc=456.4
mac_addr	MAC アドレス	16 進数の文字	mac_addr=1afe34a48ca3

- 個体管理の方法
  センサの個体番号をデバイスのプログラム内でひとつひとつ設定すると，デバイスの数だけプログラムの変更が必要になってしまう．これを避けるために，デバイスの MAC アドレスをデータ管理に利用することを提案する．

- サーバの機能
  - デバイスアップロード用 URL では，http psot されるデータを DB に保存する．
  - 閲覧用 URL では，google charts などを用いて DB のデータをグラフ化する
    - Bootstrap でレスポンシブデザインにする．
  - phpMyAdmin などの機能で，ログデータをダウンロードできるようにする．