ESXAR

デベロッパーズマニュアル

Ver.20160821

PUBLIC DOMAIN

ピン配置

RC1 PIN ASSIGN			モード切替機能
LED R / WAKE UP / MOTORB INB /SD CS	D0	GPIO16	
SERVO/DIGITAL IN	D1	GPIO5	
LED G USBSW	D2	GPIO4	
	D3	GPIO0	L:WRITE H:RUN
	D4	GPIO2	H:FLASH BOOT L:NOT VALID
	D5	GPIO14	Wi-Fi切り替え
SPEAKER	D6	GPIO12	
	D7	GPIO13	タイマー切り替え
LED B	D8	GPIO15	H:SD BOOT L:NORMAL

		TOUT/ADC	

キット部品表 小さな部品は多めに入っています

種類	PARTS	規格	今回キットの内容物
基板をつくるもの			
	基板	ESCAR	Ver.20160617
	ステンシル		
	プラ版		別途ご用意ください
プログラムや動作を設定するもの			
	USBシリアル	CH340モジュール	
	ジャンパーケーブル	メスメス	5本+空コネクタ
	ジャンパピン	3ピン	ジャンパブロック付
	ジャンパピン	2ピン	ジャンパブロック付
	ジャンパピン	使用しないように変更になりました	ジャンパブロック付
	色付きヘッダピン	9ピン	ブレッドボード用
	色付きヘッダピン	2ピン	ブレッドボード用
	ヘッダピン	5ピン	プログラムピン
スナップ部品			
	LED	フルカラーLED	
	タクトスイッチ	リセットとINPUT	2つ
	ブザー		
リフロー用部品			
	ESP-WRPPM-02	ESP8266	別にお買い求めください
	Mini360	DC-DCコンバータ	
	チップ抵抗	4.7kΩ	
	チップ集合抵抗	4.7kΩ	
	チップ集合抵抗	330Ω	
	TLP222A	フォトリレー	
	コネクタ	USB　メス	
オプションの部品			
	（コンデンサ）		オプション
	（スピーカ）		オプション
	（外部端子）		オプション
	（オーディオコネクタ）		オプション
実験用部品			
	3mmビスナット	3セット	またはプラリベット
	ジャンパーケーブル	オスオス	3本
	ブレッドボード		
	光センサー	CDS　GL5516 0.5MΩ 明抵抗：5～10kΩ　暗抵抗：0.5MΩ	
	温度センサー	サーミスタ　10kΩr NTC-MF52AT　10K ±5% B=3950 ±1%	
	リードスイッチ		
	扇風機		別にお買い求めください
	ACアダプタ		別にお買い求めください
	ライト		別にお買い求めください
	LED		
	抵抗		

ESP8266 とは？

小さな Wi-Fi モジュールです。このモジュールを使うことで、既存のマイコンや電子工作などを Wi-Fi に接続することができます。このモジュール自体にもマイコン機能があり、これにスイッチやセンサーを繋げてデータを収集したり、ライトや動くものをつなげて Wi-Fi 経由で操作したりできます。
このようなモジュールはこれがはじめて、というわけではないのですが、既存のものに比べて以下の特徴があります。

- 技適が通っている！
  - 今まで、海外製の同じようなモジュールで面白そうなものがあっても日本では使えませんでした。どうして？ 電波をある程度以上出力するものは、総務省の「技術基準適合証明」（技適）というものを取得しないと日本では使えません。
  - 技適の対象は、電波を出すものまるごとでないといけないという建前があり、今までは Wi-Fi を使うものはノート PC まるごとだとかで証明を受けないといけませんでした。でもそれだと USB ドングル形式の Wi-Fi アダプタなどが使えなくなってしまいますよね。ということでモジュールで取得する道が少しずつ拓かれてきました。
  - ESP8266 は電源と通信の配線、簡単な通信方法だけで Wi-Fi が使えます。このような使いやすいモジュールが技適を通っておおっぴらに使えるということ、更に自分でモジュール自体にプログラムもでき、内部のプログラムを書き換えても技適の適用内ということで、大きなエポックメーキングとなりました。

- 安い！
  - いくら使いやすいモジュールで、日本国内で使えても1つ 1 万円だとか、入手するのに 1000 個買わないといけないとかでは馬鹿らしいですよね。中国などでは性能はどうなんだか怪しいけれど、日本よりもはるかに安い値段で流通しているのに・・・・。今回 ESP8266 は数百円で、それもパーツ販売店などで 1 つから購入できます！！！！
- さて、この期待の ESP8266 ですが、残念ながらコストの関係で今回のセットには ESP8266 は入ってません。別途お買い求めください！
  - 売っているところ
  - 秋月電子通商
  - スイッチサイエンス
  - Amazon
  - ほか

USB シリアルアダプタを使う

PC に付属の USB シリアルアダプタを接続をします。

PC にドライバをインストール

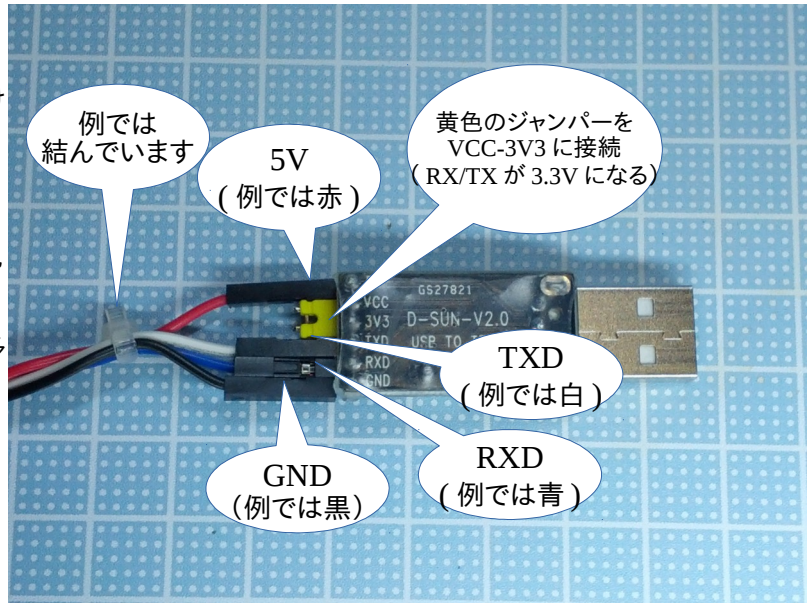
MS-Windows、および Linux (Ubuntu Linux 15.04 64bit 版でチェックしました) の場合は接続するだけで自動でインストールされます。なお、MS-Windows で認識しない場合は Windows Update を試してみてください。

Mac　OSX の場合は公式ページ http://www.wch.cn/download/CH341SER_MAC_ZIP.html からドライバをダウンロードしてインストールします。10.8 以降ではセキュリティとプライバシーの設定で、「すべてのアプリケーションを許可」にします。更に 10.10 以降ではターミナルから"sudo nvram boot-args="kext-dev-mode=1"と実行して再起動します。(野良署名付きドライバ: http://blog.sengotta.net/signed-mac-os-driver-for-winchiphead-ch340-serial-bridge/)

取り付けたシリアルポートのデバイスネームを確認してください。もしわからない場合は巻末トラブルシューティングの「シリアルポートのデバイスネーム」を参照してください。

ESXAR との接続

USB シリアルアダプタから、ESXAR への接続をします。



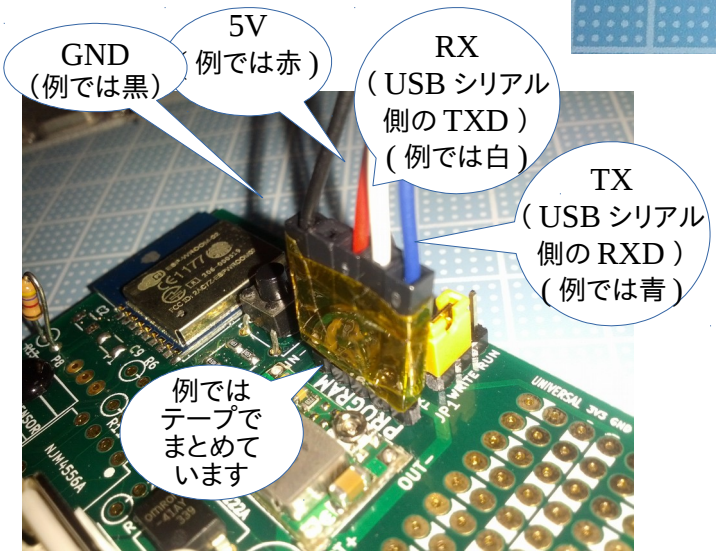
まず、USB シリアルアダプタにケーブルを取り付けます。

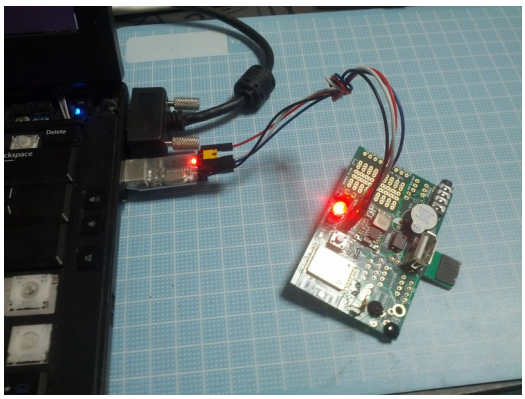
写真では赤黒青白のケーブルを使っていますがキットに付属しているケーブルは都合により色は決まっていません。適宜お持ちのケーブルに併せて色は変更してください。

(なお、慣例で GND は黒、5V は赤にすることが多いのでもし付属のケーブルにこの色が含まれていたらなるべく合わせてください。)

す。

ESXAR には写真のように接続します。





プログラムしよう!!

開発のしかた

Arduino IDE という開発環境を使ってコンパイル・書き込みをします。(なお、この方法を使うと前ページの AT コマンドの動作は上書きされて使えなくなります。)

http://arduino.cc より、開発環境 Version 1.6.5 をダウンロード、インストールしてください。

開発環境より、「ファイル」-「環境設定」で、Additional Boards Manger URLs:に

http://arduino.esp8266.com/staging/package_esp8266com_index.json

を入力し、OK を押します。

「ツール」-「ボード」-「Boards Manager...」で「esp8266 by ESP8266 Community」

と出てくるので「Install」を押します。

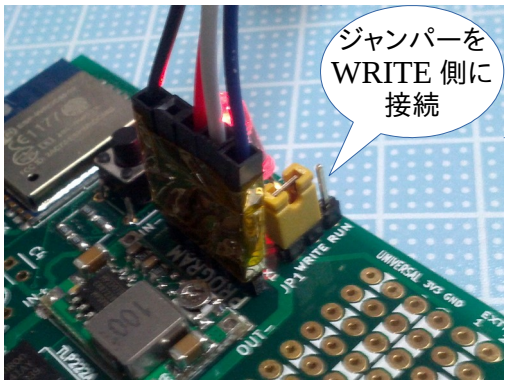
インストールが終わると、「ツール」-「ボード」に「Generic ESP8266 Module」が出てくるのでそれを選択します。「ツール」ー「ポート」を、使用する USB シリアルアダプタのものに変更しておきます。わからない場合は巻末のトラブルシューティングを参照してください。

ESP8266 のブートローダーモードと LED の接続

起動時に GPIO がどのレベルになるかで、ESP8266 の動作モードが以下のように変わります。

UART Download Mode	GPIO0 Lo GPIO2 Hi GPIO15 Lo
Normal	GPIO0 Hi GPIO2 Hi GPIO15 Lo
SD-Card Boot	GPIO0 Lo GPIO2 Lo GPIO15 Hi

写真のようにジャンパーを WRITE 側にしてから電源を入れることで、UART Download Mode となり、シリアルアダプタ(UART)からプログラミングができます。



また、LED の接続は以下の割り当てになっています。

- GPIO16　RED LED
- GPIO5　GREEN LED
- GPIO4　BLUE LED

LED をチカチカしよう

「ファイルー「スケッチの例」で「ESP8266」-「Blink」を選択します。出てくる BUILTIN_LED というのは

GPIO13 番のことなので、これを以下のように書き換えます。

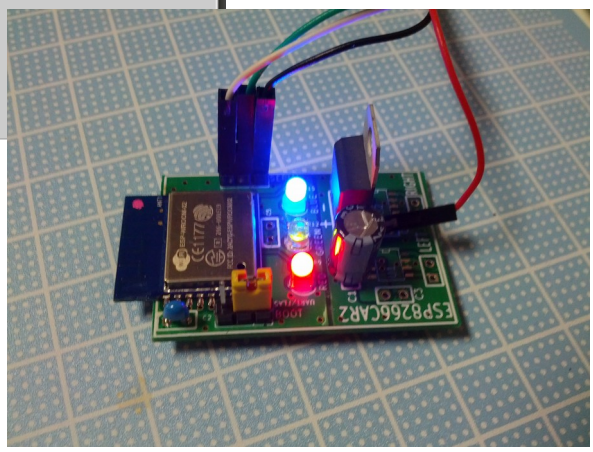
```
void setup() {
 pinMode(4, OUTPUT);
}

void loop() {
 digitalWrite(4, LOW);
 delay(1000);
 digitalWrite(4, HIGH);
 delay(2000);
}
```

緑 LED だけ点滅のプログラム

実行して緑 LED が光れば OK です。

Web から LED をチカチカしよう

サンプルの Advanced Web Server を基にして、Web ブラウザから3色の LED を ON／OFF できるようにしてみました。



```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

const char* ssid = "cnccafe";
const char* password = "freebeer";

ESP8266WebServer server(80);

const int Rled = 16;
const int Gled = 5;
const int Bled = 4;
void returnHTMLDisp(){
    char temp[300];
    int sec = millis() / 1000;
    int min = sec / 60;
    int hr = min / 60;

        snprintf ( temp, 300,

"<html>\
  <head>\
    <title>ESP8266 LED Web Brink</title>\
    <style>\
      body { background-color: #cccccc; font-family:
Arial, Helvetica, Sans-Serif; Color: #000088; }\
    </style>\
  </head>\
  <body>\
    <h1>LED CHANGED!</h1>\
    <a href=\"\/\">return</a>\
  </body>\
</html>",

    hr, min % 60, sec % 60
    );

    server.send ( 200, "text/html", temp );
}
void handleRoot(){
    char temp[500];
    int sec = millis() / 1000;
    int min = sec / 60;
    int hr = min / 60;

    Serial.println("root");
    snprintf ( temp, 500,

"<html>\
  <head>\
    <title>ESP8266 LED Web Brink</title>\
    <style>\
      body { background-color: #cccccc; font-family:
Arial, Helvetica, Sans-Serif; Color: #000088; }\
    </style>\
  </head>\
  <body>\
    <h1>Hello from ESP8266!</h1>\
    <a href=\"\/Ron\">RED ON</a>\
    <a href=\"\/Roff\">RED OFF</a><br>\
    <a href=\"\/Gon\">GREEN ON</a>\
    <a href=\"\/Goff\">GREEN OFF</a><br>\
    <a href=\"\/Bon\">BLUE ON</a>\
    <a href=\"\/Boff\">BLUE OFF</a>\
  </body>\
</html>",

    hr, min % 60, sec % 60
    );

    server.send ( 200, "text/html", temp );
}

void ledROn() {
    Serial.println("RED LED ON");
    digitalWrite(Rled, HIGH);
    returnHTMLDisp();   }
```

作例では 5 になってますが 15 に変えてください

```
void ledROff() {
    Serial.println("RED LED OFF");
    digitalWrite(Rled, LOW);
    returnHTMLDisp();   }
void ledGOn() {
    Serial.println("GREEN LED ON");
    digitalWrite(Gled, HIGH);
    returnHTMLDisp();   }
void ledGOff() {
    Serial.println("GREEN LED OFF");
    digitalWrite(Gled, LOW);
    returnHTMLDisp();   }
void ledBOn() {
    Serial.println("BLUE LED ON");
    digitalWrite(Bled, HIGH);
    returnHTMLDisp();   }
void ledBOff() {
    Serial.println("BLUE LED OFF");
    digitalWrite(Bled, LOW);
    returnHTMLDisp();
}
void handleNotFound(){
    String message = "File Not Found\n\n";
    message += "URI: ";
    message += server.uri();
    message += "\nMethod: ";
    message += (server.method() ==
HTTP_GET)?"GET":"POST";
    message += "\nArguments: ";
    message += server.args();
    message += "\n";
    for (uint8_t i=0; i<server.args(); i++){
        message += " " + server.argName(i) + ": " +
server.arg(i) + "\n";
    }
    server.send(404, "text/plain", message);
}
void setup(void){
    pinMode(Rled, OUTPUT);
    pinMode(Gled, OUTPUT);
    pinMode(Bled, OUTPUT);
    digitalWrite(Rled, LOW);
    digitalWrite(Gled, LOW);
    digitalWrite(Bled, LOW);
    Serial.begin(115200);
    WiFi.begin(ssid, password);
    Serial.println("");

    // Wait for connection
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.print("Connected to ");
    Serial.println(ssid);
    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());
    if (MDNS.begin("esp8266")) {
        Serial.println("MDNS responder started");
    }
    server.on("/",     handleRoot);
    server.on("/Ron",  ledROn);
    server.on("/Roff", ledROff);
    server.on("/Gon",  ledGOn);
    server.on("/Goff", ledGOff);
    server.on("/Bon",  ledBOn);
    server.on("/Boff", ledBOff);
    server.onNotFound(handleNotFound);
    server.begin();
    Serial.println("HTTP server started");
}
void loop(void){
    // Handle incoming connections
    server.handleClient();
}
```

センサーをつなげる

ESP8266 には、アナログの信号を読み取ることのできるポートが1つあります。これにセンサーを繋げてみましょう。例では CdS という光センサーをつなげてみました。例と同じように実験をするには、別途以下の部品を買い揃えてください。

- CdS 光センサー　明抵抗 10kΩ のものを使いましたが、明抵抗 50kΩ ぐらいの方が使いやすいでしょう。
- 1/4W 抵抗 22kΩ　　分圧甩

1.抵抗とともにセンサーホールに取り付けます。

.ESP8266 では 0-1023 の値で読み取ります。

しかしながらセンサーへ供給する電源電圧は 3.3V なのでうまくして信号変化が 0-1V に収まるようにしないといけません。ここでは CdS を使いましたが、分圧抵抗を同様にして工夫してみてください。

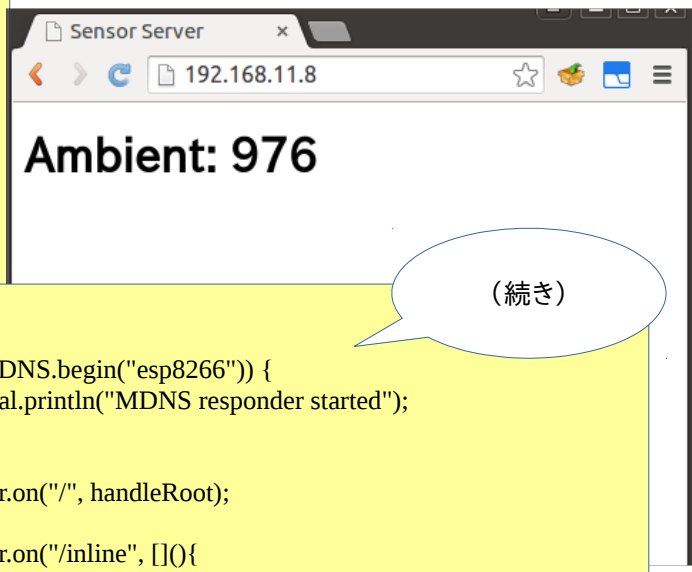
4.このスケッチを実行すると、明るさを表示する Web サーバとして動作します。このスケッチはサンプルプログラムの「HelloServer」を基に作成しました。

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>

extern "C" {
#include "user_interface.h"
}

const char* ssid = "cnccafe";
const char* password = "freebeer";

ESP8266WebServer server(80);

const int led = 13;

void handleRoot() {
  digitalWrite(led, 1);
  int ret;
  ret=system_adc_read();
  String message = "<!DOCTYPE html><title>Sensor Server</title>";
  message +="<p><h1>Ambient: ";
  message += ret;
  message +="</h1>";
  server.send(200, "text/html", me
  Serial.println(message);
  digitalWrite(led, 0);
}

void handleNotFound(){
  digitalWrite(led, 1);
  String message = "404 Not Found\n";
  server.send(404, "text/plain", message);
  digitalWrite(led, 0);
}

void setup(void){
  pinMode(led, OUTPUT);
  digitalWrite(led, 0);
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
```

実行結果

アナログ値を読むための
あらかじめおまじない

アナログ値を読んでいるところ



（続き）

```
  if (MDNS.begin("esp8266")) {
    Serial.println("MDNS responder started");
  }

  server.on("/", handleRoot);

  server.on("/inline", [](){
    server.send(200, "text/plain", "this works as well");
  });

  server.onNotFound(handleNotFound);

  server.begin();
  Serial.println("HTTP server started");
}

void loop(void){
  server.handleClient();
}
```

メールを送る

アナログ値を読んで、1024 になったらメールを送ります。メールの送信サーバにあわせて送信方法はプログラムを変更する必要があります。例に使ったメールサーバは、pop-before-smtp を使っているのでまず pop3 で接続し、その後 smtp でメールを送っています。自分のメールサーバがどのように動作するかは、telnet などで調べてください。

```
#include <ESP8266WiFi.h>
const char* ssid = "cnccafe";      //  your Wi-Fi ID
const char* password = "freebeer";    //  your Wi-Fi key
const char* host = "mail.example.net"; // your mail settings
const char* auth_user = "hogehoge";
const char* auth_pass = "password";
const char* mail_from = "hogehoge@example.net";
const char* rcpt_to = "hagehage@example.com";
const char* subject = "ESCAR通知メール";
const int smtpPort = 587;
const int pop3Port = 110;
int value = 0;
int beforeValue = 1024;
extern "C" {
#include "user_interface.h"
}
int ret;

WiFiClient client;

void ReadLoop(){
  while(client.available()){
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }
}
void sendMail(){
    delay(5000);
    ++value;
    Serial.print("connecting to ");
    Serial.println(host);
    // Use WiFiClient class to create TCP connections
    if (!client.connect(host, pop3Port)) {
      Serial.println("connection failed");
      return;
    }
    delay(1000);
    client.print("USER ");
    client.println(auth_user);
    ReadLoop();
    delay(3000);
    client.print("PASS ");
    client.println(auth_pass);
    ReadLoop();
    delay(1000);
    client.print("QUIT");
    ReadLoop();
    Serial.println();
    Serial.println("closing pop3 connection");

    delay(3000);

    Serial.print("connecting to ");
    Serial.println(host);
    // Use WiFiClient class to create TCP connections
    if (!client.connect(host, smtpPort)) {
      Serial.println("connection failed");
      return;
    }
    client.print("ehlo ");
    client.println(host);
    ReadLoop();
    delay(1000);
    client.print("mail from:");
```

```
    client.println(mail_from);
    ReadLoop();
    delay(1000);
    client.print("rcpt to:");
    client.println(rcpt_to);
    ReadLoop();
    delay(1000);
    client.println("data");
    ReadLoop();
    delay(1000);
    client.print("From:");
    client.println(mail_from);
    ReadLoop();
    delay(1000);
    client.print("To:");
    client.println(rcpt_to);
    ReadLoop();
    delay(1000);
    client.print("subject:");
    client.println(subject);
    ReadLoop();
    delay(1000);
    client.println("");
    client.println("ONになりました！");
    ReadLoop();
    delay(1000);
    client.println(".");
    ReadLoop();
    delay(1000);
    client.println("quit");
    ReadLoop();
    Serial.println();
    Serial.println("closing connection");
}
void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(4, OUTPUT);
  // Connect to Wi-Fi
  // connect();
}
void connect(){
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
void loop() {
  ret=system_adc_read();
  Serial.println ( ret );
  if (1024 ==ret ) {
    if ( 1024 != beforeValue ){
      Serial.println ( "ON!" );
      digitalWrite(4, HIGH);
//      sendMail();
    }
  } else {
       digitalWrite(4, LOW);
  }
  beforeValue = ret;
  delay(500);
```

トラブルシューティング!!

動かなくなった!

ジャンパーを BOOT 側にしてプログラムを書き込みし、動作しますが、2回めの動作はジャンパーを戻してから電源を ON にしないといけません（ジャンパーを BOOT 側にしたままだとプログラム書き込み待機状態となる）。

Wi-Fi が接続できない

ESP8266 が接続できる Wi-Fi は 2.4GHz のみとなっています。5GHz 帯の Wi-Fi には接続できませんのでご注意ください。また、2.4GHz 帯のものも、電波が混雑していると接続が不安定になりがちです。

モーターコントロールが不安定

モータをつなげた場合、例えば FA130 モータなどは、ストールした時に 1000mA ぐらい流れたりします。そのために、ESP8266 への電源電圧が不安定となっていないかを確認しましょう。

また、モーターはノイズが多く出ます。対策としてコンデンサーをモータに直接つけるのも効果的です。

書き込みできない!

付属の USB シリアルアダプタは、USB ポートの 5V 電圧が足りないと問題が起こりやすくなります。その場合、PC の USB ポートに直接接続してみてください。

また、下記「シリアルポートのデバイスネーム」も参照してください。

シリアルポートのデバイスネーム

取り付けたシリアルポートのデバイスネームは、MS-Windows の場合は COM0 や COM12 のように、Mac OSX の場合は tty.なんとかかんとか のような名前、Linux の場合は ttyUSB0 や ttyUSB1 のような名前になります。もしわからない場合は、下記「ArduinoIDE で USB シリアルアダプタの確認と設定をする」を参照ください。

ArduinoIDE で USB シリアルアダプタの確認と設定をする

•「ツール」-「シリアルポート」で設定します。また、今システムで認識している USB ポートの確認もできます。認識しているものが一覧で表示されます。

•どれが USB シリアルアダプタのポートなのかわからない場合は、シリアルアダプタを取り付けたときと、取り外したときを比較して、取り付けた時に現れるシリアルポートを選択します。Mac の場合は 2 つ同時に現れますが、tty という名前のついているほうが正解です。

•なお、シリアルポートは何度も付け外ししていると番号が変わる場合があります。その際は同様に「ツール」-「シリアルポート」で選択しなおしてください。

Mac の場合の注意点

/dev/cu.usbserial 等になっている場合は、/dev/tty.usbserial を選択

Linux の場合の注意点

もし、シリアルデバイス(/dev/ttyUSB0　等)を使用する権限が無い場合は、一般ユーザで動かすために,利用ユーザを dialout グループに登録する。※デバイスファイルは dmesg コマンドなどで確認可能。

シリアル通信ソフト

MS-Windows の場合は Tera Term、Linux の場合は minicom が良く使われます。Mac OSX の場合は jerm というソフトがおすすめです（ダウンロードしてください）。jerm -b 115200 -p none -d 8 -s 1 -f none /tty/usb なんとか（調べたデバイス名）として起動します。

ちょっとよくわからないな・・・という方は、Arduino IDE をシリアル通信ソフトとして使うことができます。

Arduino IDE をシリアル通信ソフトとして使用する

「ツール」-「シリアルモニタ」で起動します。「115200bps」「CR および LF」として設定してください。

こわしちゃったよ!!

実験などをやっていると、なかなかうまくいかないことも多いと思います。パーツなどを壊してしまっても、遠慮せずにご相談ください。相談先は下記をご参照ください。

もっと知りたい!!

オープンフォースまでご遠慮無くお問い合わせください。

openforce@project2108.com

また、日本 Android の会ロボット部にて、月例で勉強会を行なっています。興味のある方は ML(メーリングリスト)にご登録ください。
http://groups.google.com/group/robot-android-group-japan-akb
ML では勉強会のアナウンスや部員が興味をもった様々な話題がなされています。