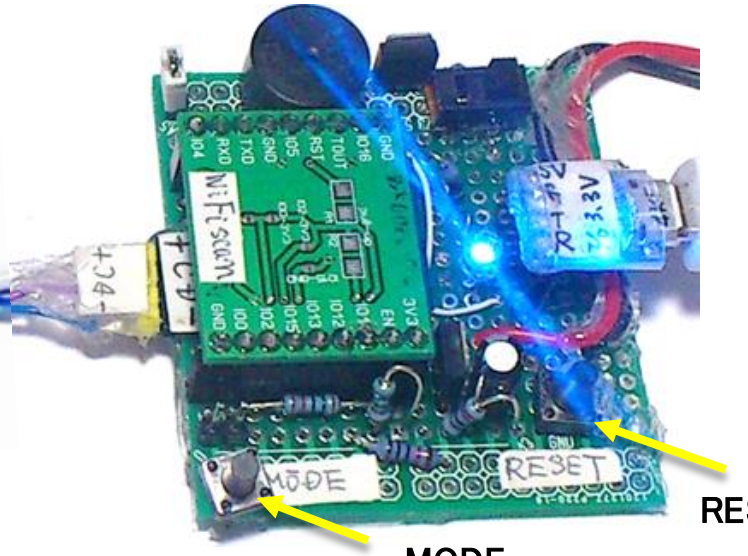
# 受信強度，　http サーバ設定による送信　（2モードスケッチ統合）

2017.3.13


受信強度測定




RESET

MODE

受信強度モード ： 電源ON，または“RESET” ボタン
サーバモード　　： 電源ON後，または“RESET” ボタン後，
2 秒以内に“MODE”ボタン

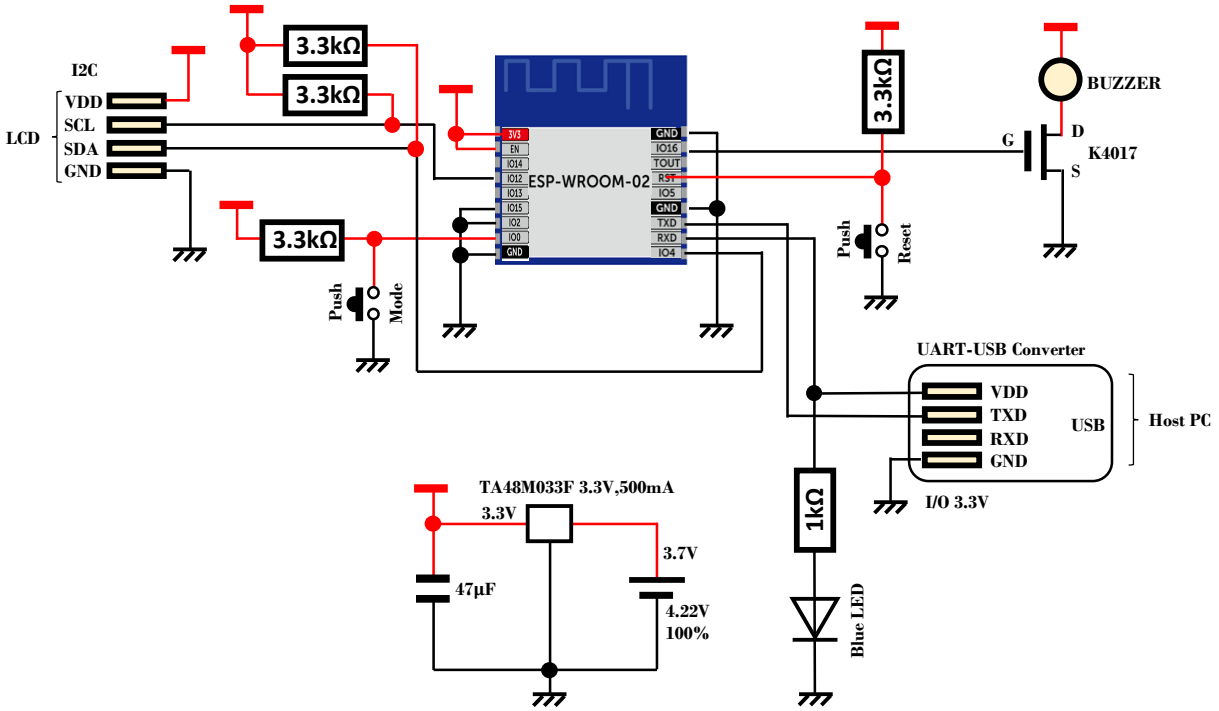

http サーバ起動


http データ送信


受信

参考にさせて頂いた他人様のページ　http://blog.goo.ne.jp/sim00/e/ab138be751d447bcdb0eecaaca232214

# 1つの装置に2つの機能を持たせる




他人様ESP8266の元図　http://qiita.com/umi_kappa/items/ac3d37db44a2dcfe71fd

```
/*  2017.2.17 (Fri.)
    WiFi_ESP8266_LCD_Inten_Smapho.ino
      ---- WiFi Radio Intensity Detector ----

    WiFi module WROOM-02 by aitendo
    LCD 1602D1      Strawberry-Linux &
    LCD AQM1602     Akizuki

  ref: http://blog.goo.ne.jp/sim00/e/ab138be751d447bcdb0eecaaca232214 */

#include <Wire.h>
#include "ESP8266WiFi.h"

#define ADDR   0x3e
#define C_Low  0x70
#define C_High 0x56   // 0x5c?

#define AQM1602
#ifdef  AQM1602
#define Cont     5   // Contrast AQM1602 Akizuki
#else
#define Cont    10   // Contrast 1602D1 Strawberry
#endif

//#define Cont     9   // Contrast

/*  ========= Trans Smapho http server ========= */
#define BUZZER 16
#define WIFI_SSID1 "**********"
#define WIFI_PSK1  "**********" // Univ.
#define WIFI_PSK2  "**********"      // Home

#define WIFI_SSID3 "**********"   // 0 ==> O (Oh---)
#define WIFI_PSK3  "**********"    // Private

#define DEST_HOST "api.fixer.io"
#define DEST_PORT 80
#define DEST_URL  "/latest?base=USD&symbols=JPY"

const char* ssid     ;
const char* password ;
boolean Ini_html_on = false;
//ブラウザからの初回HTTPレスポンス完了したかどうかのフラグ

WiFiServer server(80);
WiFiClient client;

uint8_t cmd_cr[] = {0xc0};  // C/R
uint8_t cmd_cl[] = {0x01};  // CLear Display
int ID0 = 0;         // = ID0

void setup() {
  uint8_t cmd_init1[] = {0x38, 0x39, 0x14};
  uint8_t cmd_init2[] = {C_Low | (Cont & 0x0f), C_High | (Cont >> 4 & 3), 0x6c};
  uint8_t cmd_init3[] = {0x38, 0x0d, 0x01};   // 0x0d => 0x0c ?

  uint8_t mess_1[]         = "Setup done";
  uint8_t select_APInten[] = "ON=AP, OFF=Inten";
  uint8_t start_server[]   = "start HTT Server";
  uint8_t point[]     = {0x2e};

  pinMode(BUZZER, OUTPUT);
  digitalWrite(BUZZER, LOW);

  pinMode(ID0, INPUT);
  Serial.begin(115200);
  delay(10);
  //Wire.begin(4, 14);
  Wire.begin(4, 12);
  delay(40);

  // LCD Initialize
  command(cmd_init1, sizeof(cmd_init1));
  command(cmd_init2, sizeof(cmd_init2));
  delay(300);
  command(cmd_init3, sizeof(cmd_init3));

  delay(2000);
  clear_LCD();

  if (digitalRead(ID0)) {         //   HTTP Server mode
    write(start_server, sizeof(start_server));
    CR_LF();

    ssid     = WIFI_SSID3;
    password = WIFI_PSK3;

    WiFi.begin(ssid, password);
    Serial.println();
    Serial.println("start http server");
    while (WiFi.status() != WL_CONNECTED) {
      delay(1000);
      Serial.print(".");
      write(point, sizeof(point));
    }

    Serial.println("");
    Serial.println("WiFi connected");
    server.begin();
    Serial.println("Server started");
    Serial.println(WiFi.localIP());
    delay(1000);
    HTTP_server();
  }
  else {            // ID0 = 0 ==>Intensity WiFi Wave
    WiFi.mode(WIFI_STA);
    delay(100);
    Serial.println();
    Serial.println("Setup done");
    write(mess_1, sizeof(mess_1));
    delay(1000);
    WiFi_scan();
  }
}

void loop() {
  ;
}

// ********* HTTP Server ***********

void HTTP_server() {
  Disp_LCD();

  while (1) {
    if (Ini_html_on == false) {
      Ini_HTTP_Response();
    } else if (client.available()) {
      Serial.println("client.available()");
      Serial.print(client.read());
    }
    delay(100);//これは重要かも。これがないと動作しないかも。
  }
}

//      *****  初回ブラウザからのGET要求による
//      HTMLタグ吐き出しHTTPレスポンス  *******

void Ini_HTTP_Response()
{
  client = server.available();//クライアント生成
  delay(1);
  String req;

  while (client) {

    if (client.available()) {

      req = client.readStringUntil('¥n');
      Serial.println(req);
      if (req.indexOf("GET / HTTP") >= 0 || req.indexOf("GET /favicon") >= 0) {
        //ブラウザからリクエストを受信したらこの文字列を検知する
        //Google Chromeの場合faviconリクエストが来るのでそれも検出する
        Serial.println("-----from Browser FirstTime HTTP Request---------");
        Serial.println(req);
```

```
      //ブラウザからのリクエストで空行（¥r¥nが先頭になる）まで読み込む
       while (req.indexOf("¥r") != 0) {
         req = client.readStringUntil('¥n');
         //¥nまで読み込むが¥n自身は文字列に含まれず、捨てられる
         Serial.println(req);
       }
       req = "";
       delay(10);//10ms待ってレスポンスをブラウザに送信
       //メモリ節約のため、Fマクロで文字列を囲う
       //普通のHTTPレスポンスヘッダ
       client.print(F("HTTP/1.1 200 OK¥r¥n"));
       client.print(F("Content-Type:text/html¥r¥n"));
       client.print(F("Connection:close¥r¥n¥r¥n"));// 1 行空行が必要
       //ここからブラウザ表示のためのHTML JavaScript吐き出し
       client.println(F("<!DOCTYPE html>"));
       client.println(F("<html>"));
       client.println(F("<font size=30>"));
       client.println(F("Hello World<br>"));
       client.println(F("0x07"));
       client.println(F("It is fine<br>"));    // 追加
       client.println(F("TAKAHASHI FRC Tokyo<br>"));    // 追加

       client.println(F("</html>¥r¥n"));
       delay(1);//これが重要！これが無いと切断できないかもしれない。
       client.stop();//一旦ブラウザとコネクション切断する。

       Serial.println("¥nGET HTTP client stop----------------------");
       req = "";
       Ini_html_on = false;  //一回限りの接続の場合、ここをtrueにする

       Blink_LCD();
     }
   }
  }
}

void Blink_LCD() {
  for (int i = 0; i < 4; i++) {
    clear_LCD();
    digitalWrite(BUZZER, HIGH);
    delay(300);
    Disp_LCD();
    digitalWrite(BUZZER, LOW);
    delay(300);
  }
}
void Disp_LCD() {
  uint8_t    ssi[] = WIFI_SSID3;
  uint8_t  hello[] = "Send Hello World";

  write(ssi, sizeof(ssi));
  CR_LF();    // CRLF
  write(hello, sizeof(hello));
}
```

```
// ********* scan WiFi Intensity  **********
void WiFi_scan() {
  uint8_t scan_start[] = "scan start -----       ";
  uint8_t scan_done[]  = "scan done ------        ";
  uint8_t scan_ssid[]  = "                        ";
  char    scan_inte[]  = "                        ";
  int Inten;
  int i, j;

  String data;
  while (1) {
    clear_LCD();
    delay(10);
    write(scan_start, sizeof(scan_start));
    Serial.println("scan start");
    int n = WiFi.scanNetworks();
    CR_LF();
    write(scan_done, sizeof(scan_done));
    delay(500);
    Serial.println("scan done");
    if (n == 0)
      Serial.println("no networks found");
    else
    {
      Serial.println();
      Serial.print(n);
      Serial.println(" networks found");
      Serial.println("-----------------------------");
      clear_LCD();
      delay(1);
      for ( i = 0; i < n; ++i) {
        Serial.print(i + 1);
        Serial.print(": ");
        for (j = 0; j < 20; j++) {
          scan_ssid[j] = ' ';
        }
        for (int j = 0; j < 16; ) {
          if ( (WiFi.SSID(i))[j  - 1] != ' ') {
            scan_ssid[j] = (WiFi.SSID(i))[j];
            ++j;
          }
        }

        for (j = 0; j < 20; j++) {
          scan_inte[j] = ' ';
        }
```

```
        Inten = (WiFi.RSSI(i));
        Inten += 100;
        dtostrf(Inten, -5, 0, scan_inte);
        clear_LCD();
        write(scan_ssid, sizeof(scan_ssid));
        CR_LF();
        for (int j = 0; j < 16; ) {
          scan_ssid[j] = scan_inte[j] ;
          ++j;
        }
        write(scan_ssid, sizeof(scan_ssid));

        Serial.print(WiFi.SSID(i));
        Serial.print(" (");
        Serial.print(scan_inte);
        Serial.print(")");
        Serial.println((WiFi.encryptionType(i) == ENC_TYPE_NONE) ? " " : "*");
        delay(1000);
      }
    }
    Serial.println("");
  }
}


void CR_LF() {
  command(cmd_cr, sizeof(cmd_cr));
  delay(10);
}

void clear_LCD() {
  command(cmd_cl, sizeof(cmd_cl));
  delay(10);
}

void command(uint8_t *cmd, size_t len) {
  size_t i;
  for (i = 0; i < len; i++) {
    Wire.beginTransmission(ADDR);
    delayMicroseconds(30);
    Wire.write(0x00);
    Wire.write(cmd[i]);
    delayMicroseconds(30);
    Wire.endTransmission();
    delayMicroseconds(30);    // 26.3us
  }
}

void write(uint8_t *cmd, size_t len) {
  size_t i;
  for (i = 0; i < len; i++) {
    Wire.beginTransmission(ADDR);
    delayMicroseconds(30);
    Wire.write(0x40);
    delayMicroseconds(30);
    Wire.write(cmd[i]);
    delayMicroseconds(30);
    Wire.endTransmission();
    delayMicroseconds(30);    // 26.3us
  }
  delay(1);
}

void disp_demo() {
  uint8_t cmd_str1[]  = "WiFi LAN ESP8266";
  uint8_t cmd_str2[]  = "Takahashi2017215";
  uint8_t cmd_str3[]  = "LCD 1602D1 *****";
  uint8_t cmd_str4[]  = "Strawberry-Linux";
  while (1) {
    clear_LCD();
    write(cmd_str1, sizeof(cmd_str1));
    CR_LF();
    write(cmd_str2, sizeof(cmd_str2));
    delay(2000);
    clear_LCD();
    write(cmd_str3, sizeof(cmd_str3));
    CR_LF();
    write(cmd_str4, sizeof(cmd_str4));
    delay(2000);
  }
}
```