
ESP-WROOM-02(ESP8266)

ちょっと使ってみた感じ

goji2100.com

*本書内の社名、製品名などは、一般に各社の商標または登録商標です。

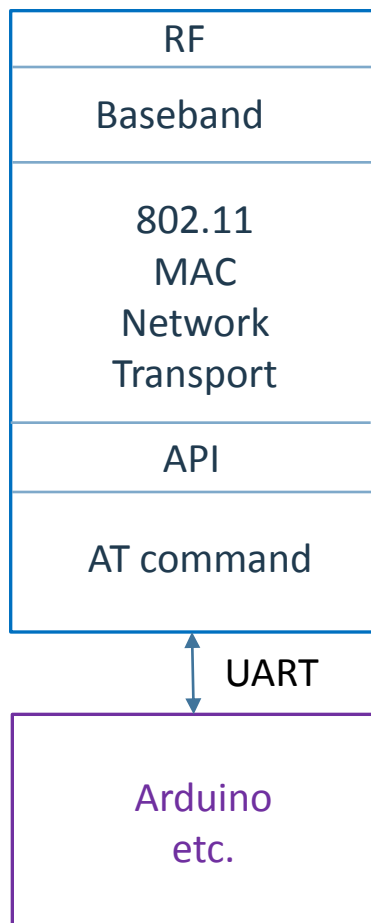


ESP-WROOM-02とは

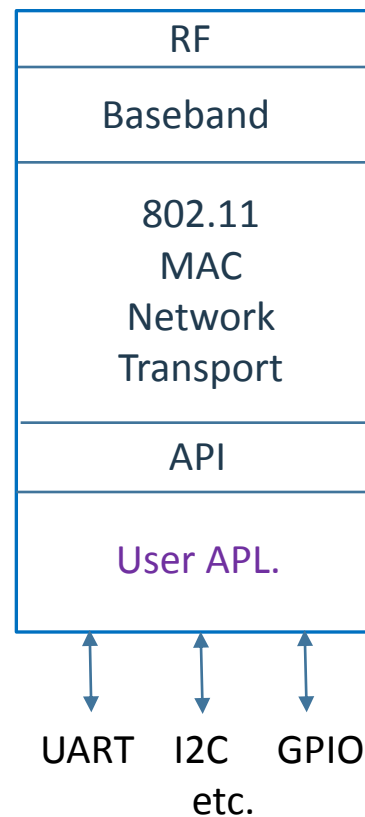
- ESP-WROOM-02は、TCP / IPプロトコルスタックを統合した32ビット低消費電力マイクロコントローラ（MCU）を搭載したWi-Fiモジュールです。
MCUには、HSPI、SDIO、UART、PWM、I2C、I2S、ADCなどの豊富なインターフェースモジュールが組み込まれています。
組み込み機器などに安価に低消費電力のワイヤレスインターネット接続機能を追加することができます。
- MicrochipのRN171やXBeeのWi-Fiモジュールと何が違う？
 - 内部MCUにコードを書くためのSDKが公開されている
 - Arduino IDEでもコードが書ける
 - **安い！（500円前後）**

海外で安いWi-Fiモジュールとして評判でしたが、技適マークが付いていないために国内では使用できませんでした。
WROOM-02には技適マークが付いていますので、日本国内でも安心して使用できる製品です。

ESP-WROOM-02 使い方



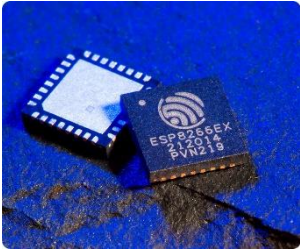
Wi-Fiモジュールとして使う



Wi-Fiモジュール付MCUとして使う

AT Command: <https://cdn.sparkfun.com/datasheets/Wireless/WiFi/Command Doc.pdf>

ESP8266EX SoC



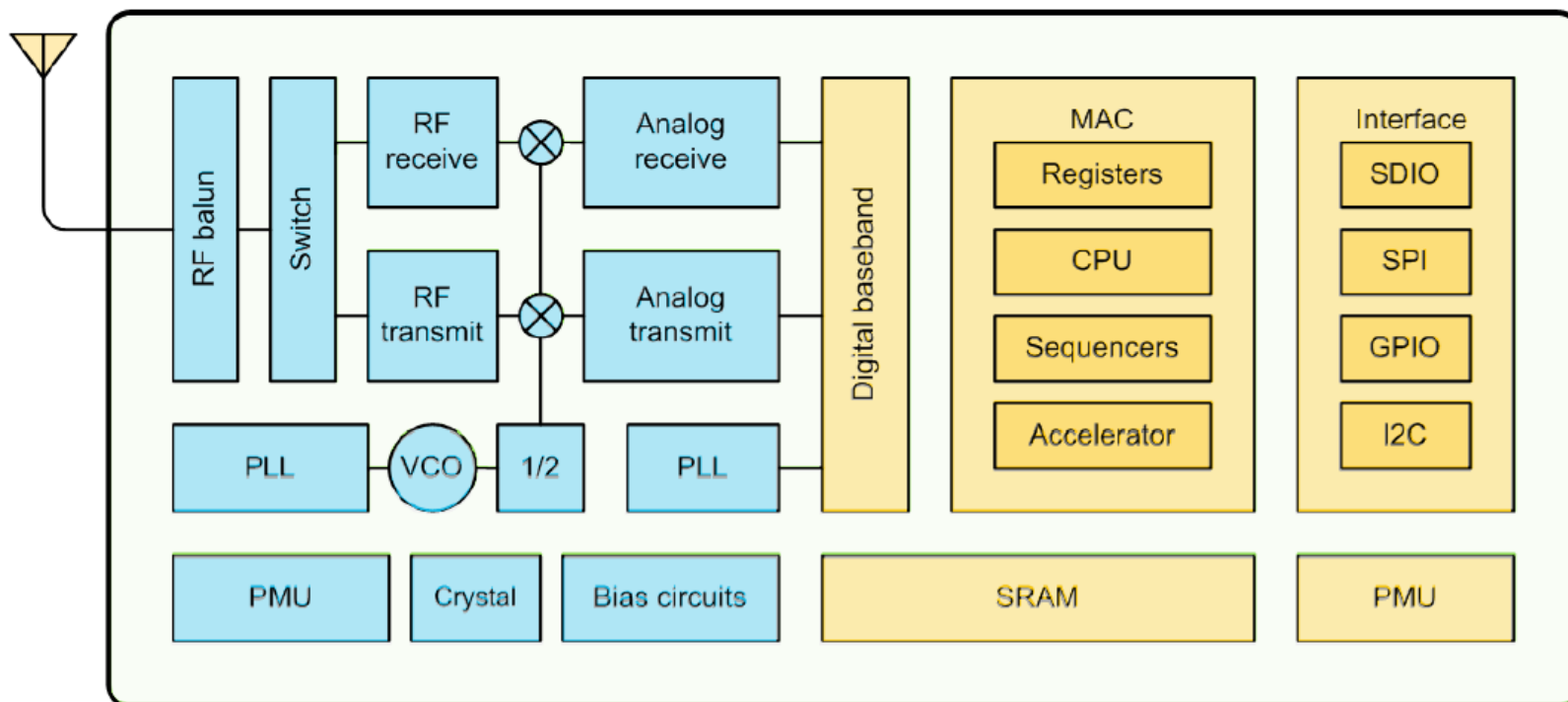
Features

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU aggregation and 0.4s guard interval
- Deep sleep power <10uA, Power down leakage current < 5uA
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)
- +20dBm output power in 802.11b mode
- Operating temperature range -40C ~ 125C
- FCC, CE, and TELEC certified

[*http://blog100.nimokichi.com/wp-content/uploads/2015/06/ESP8266_WROOM_WiFi_Module_Datasheet_EN_v0.3.pdf](http://blog100.nimokichi.com/wp-content/uploads/2015/06/ESP8266_WROOM_WiFi_Module_Datasheet_EN_v0.3.pdf)

ESP8266EX SoC

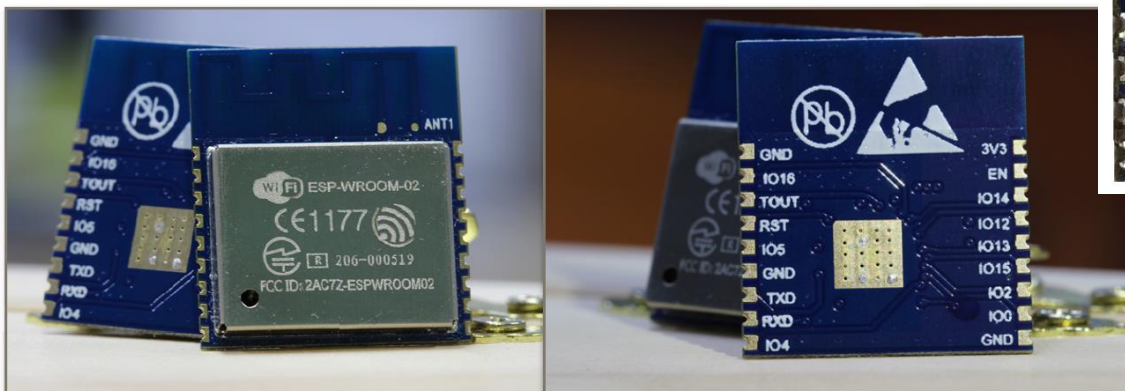
ESP8266EX Block Diagram



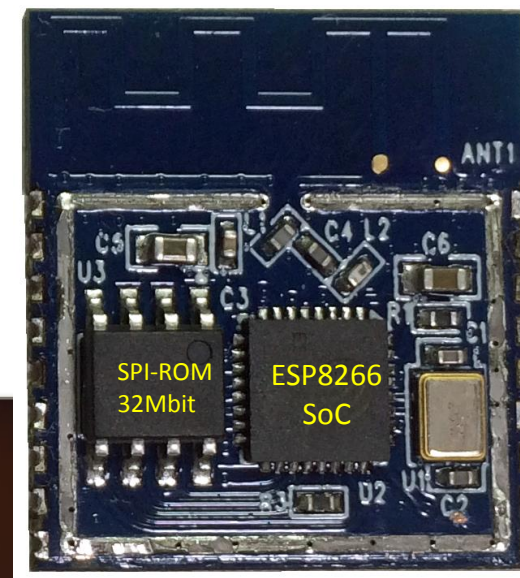
内部にはROMがありません、プログラムは外部のSPI-ROMからSRAMにロードされて実行されます。

*http://blog100.nimokichi.com/wp-content/uploads/2015/06/ESP8266_WROOM_WiFi_Module_Datasheet_EN_v0.3.pdf

ESP-WROOM-02 外観、内部



外観写真



シールドケース内部写真

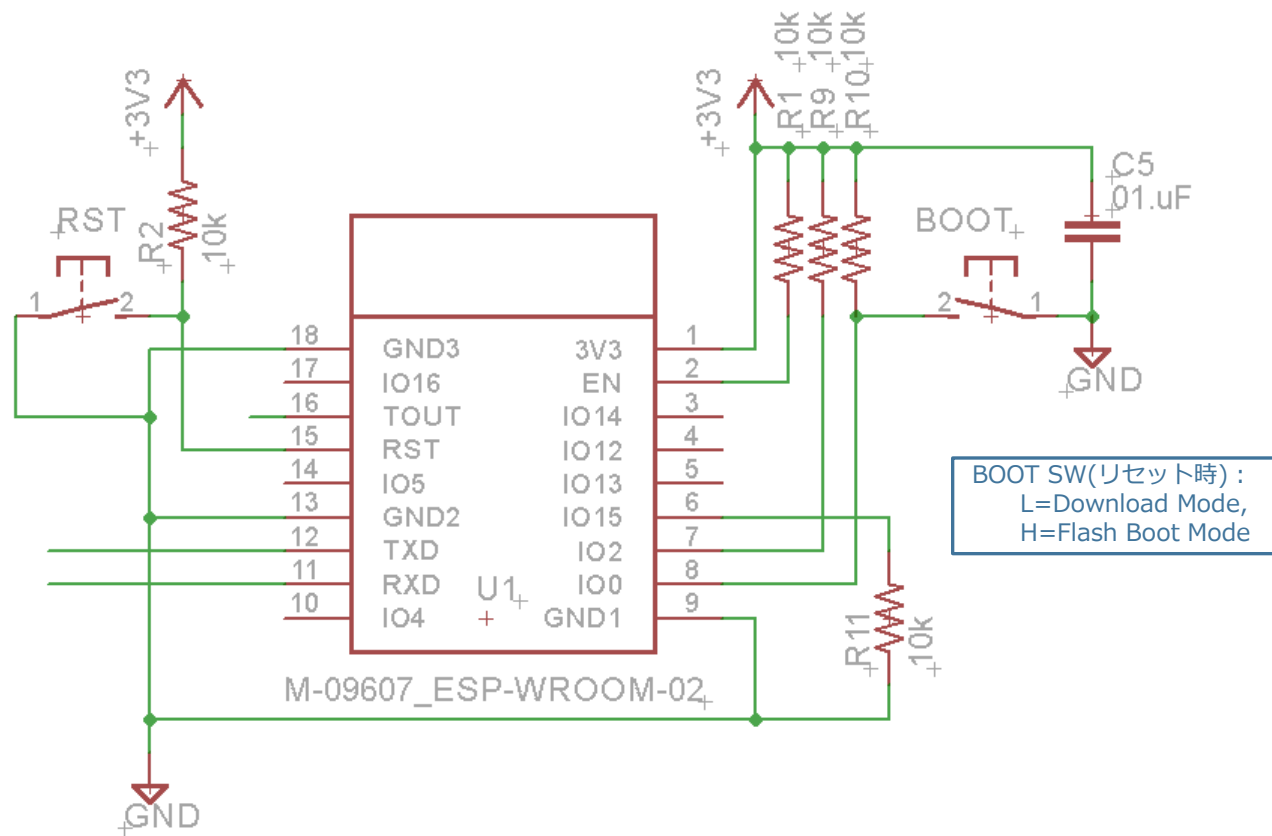
*http://blog100.nimokichi.com/wp-content/uploads/2015/06/ESP8266_WROOM_WiFi_Module_Datasheet_EN_v0.3.pdf

ESP-WROOM-02 ピン仕様

| NO. | Pin Name | Function |
|-----|----------|--|
| 1 | 3V3 | 3.3V power supply (VDD) |
| 2 | EN | Chip enable pin. Active high. |
| 3 | IO14 | GPIO14; HSPI_CLK |
| 4 | IO12 | GPIO12; HSPI_MISO |
| 5 | IO13 | GPIO13; HSPI_MOSI; UART0_CTS |
| 6 | IO15 | GPIO15; MTDO; HSPICS; UART0_RTS |
| 7 | IO2 | GPIO2; UART1_TXD |
| 8 | IO0 | GPIO0 リセット時 : L=Download Mode、 H=Flash Boot Mode |
| 9 | GND | GND |
| 10 | IO4 | GPIO4 |
| 11 | RXD | UART0_RXD; GPIO3 |
| 12 | TXD | UART0_TXD; GPIO1 |
| 13 | GND | GND |
| 14 | IO5 | GPIO5 |
| 15 | RST | Reset the module |
| 16 | TOUT | It can be used to test the power-supply voltage of VDD3P3 (Pin3 and Pin4) and the input power voltage of TOUT (Pin 6). However, these two functions cannot be used simultaneously. |
| 17 | IO16 | GPIO16; can be used to wake up the chipset from deep sleep mode. |
| 18 | GND | GND |

*http://blog100.nimokichi.com/wp-content/uploads/2015/06/ESP8266_WROOM_WiFi_Module_Datasheet_EN_v0.3.pdf

ESP-WROOM-02 基本接続図



基本接続図

上記のWROOM-02のライブラリはnonNoiseさんのGitHub
*<https://github.com/nonNoise/AKIZUKI-Eagle-Library/tree/master>

Breakout Boardいろいろ

- モジュールは1.5mmピッチのためにBreakout Boardが便利



メーカー：マイクロテクニカ
モデル名：BBCOMP-ESPWROOM02



メーカー：スイッチサイエンス
モデル名：SSCI-023474 《フル版》
「フル版」、「シンプル版」がある



メーカー名：Aitendo
モデル名：WiFiモジュール変換基板（J-B） [IFB1518J-B]
「(A)」と「(B)」がある



メーカー名：Cerevo
モデル名：CDP-ESP8266

ソフトウェア開発環境 Arduino

- Arduino IDE 1.6.4以上でサポート (1.6.5 : 2015/9/6現在)

<https://github.com/esp8266/Arduino>

Installing with Boards Manager

Starting with 1.6.4, Arduino allows installation of third-party platform packages using Boards Manager. We have packages available for Windows, Mac OS, and Linux (32 and 64 bit).

- Install Arduino 1.6.5 from the [Arduino website](#). Arduino本家
- Start Arduino and open Preferences window.
- Enter `http://arduino.esp8266.com/stable/package_esp8266com_index.json` into *Additional Board Manager URLs* field. You can add multiple URLs, separating them with commas.
- Open Boards Manager from Tools > Board menu and install *esp8266* platform (and don't forget to select your ESP8266 board from Tools > Board menu after installation).

Available versions

Stable version updated Jul 23, 2015

Boards manager link: http://arduino.esp8266.com/stable/package_esp8266com_index.json

Documentation: [Reference](#)

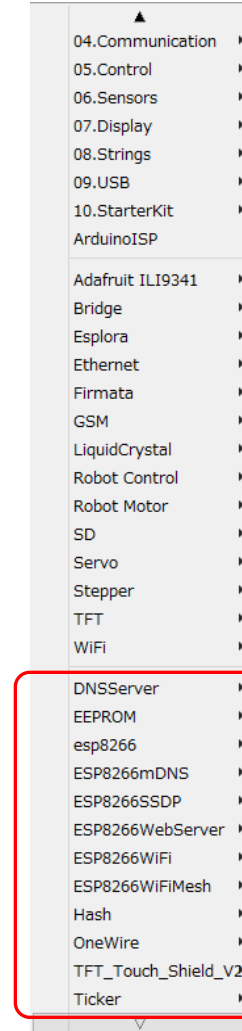
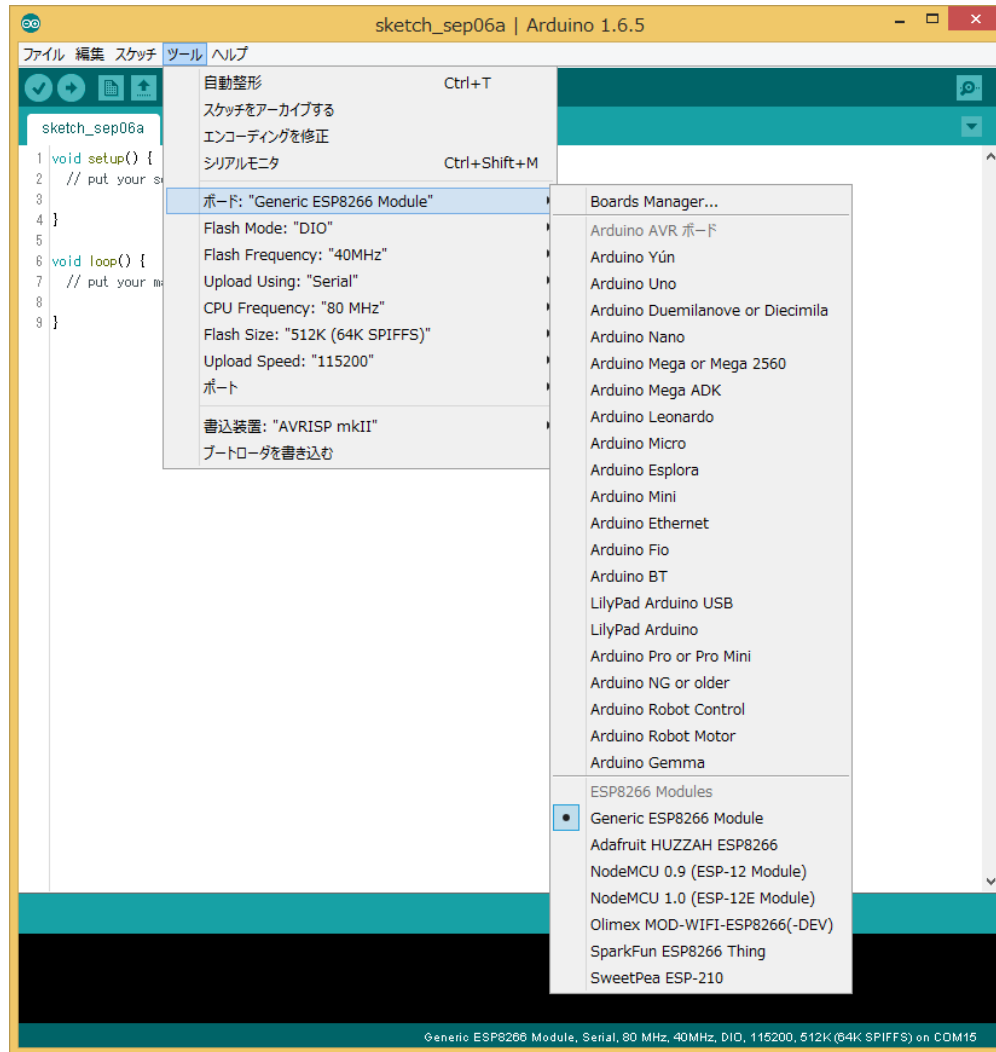
Staging version updated Aug 31, 2015

Boards manager link: http://arduino.esp8266.com/staging/package_esp8266com_index.json

Documentation: [Reference](#) 説明書

ソフトウェア開発環境 Arduino Board Manager

- Arduino IDE Board Manager (ESP-WROOM-02はGenericを選択)



スケッチの例が追加される

サンプル・スケッチ Blink

The screenshot shows the Arduino IDE interface with the Blink sketch loaded. The code is as follows:

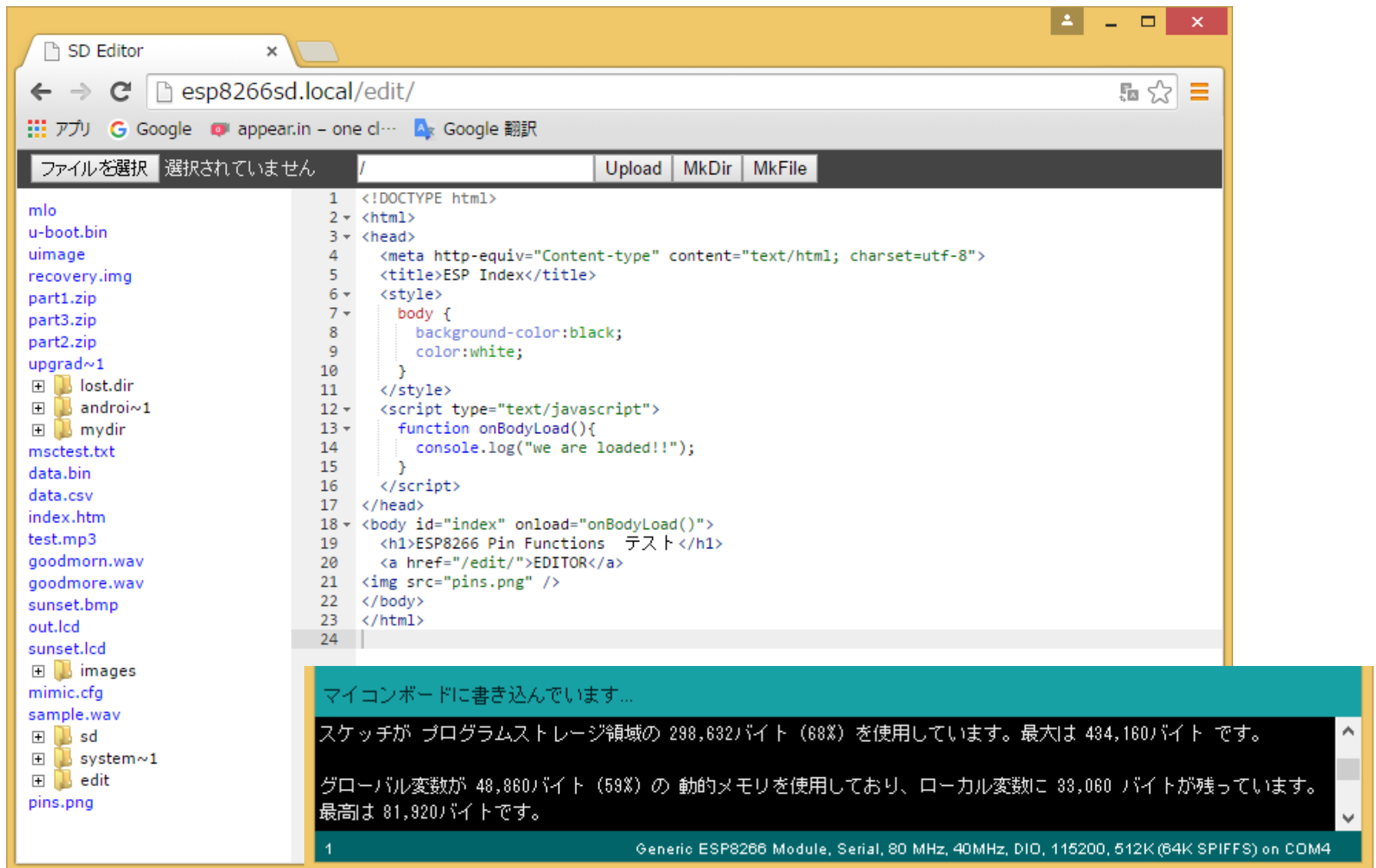
```
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4
5  Most Arduinos have an on-board LED you can control. On the Uno and
6  Leonardo, it is attached to digital pin 13. If you're unsure what
7  pin the on-board LED is connected to on your Arduino model, check
8  the documentation at http://www.arduino.cc
9
10 This example code is in the public domain.
11
12 modified 8 May 2014
13 by Scott Fitzgerald
14 */
15
16
17 // the setup function runs once when you press reset or power the board
18 void setup() {
19   // initialize digital pin 13 as an output.
20   pinMode(13, OUTPUT);
21 }
22
23 // the loop function runs over and over again forever
24 void loop() {
25   digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
26   delay(1000);           // wait for a second
27   digitalWrite(13, LOW); // turn the LED off by setting the output LOW
28   delay(1000);           // wait for a second
29 }
```

A warning dialog box is displayed in the foreground, indicating memory usage:

マイコンボードに書き込んでいます...
スケッチが プログラムストレージ領域の 201,082バイト (46%) を使用しています。最大は 434,160バイト です。
グローバル変数が 44,592バイト (54%) の 動的メモリを使用しており、ローカル変数に 37,328 バイトが残っています。最高は 81,920バイトです。

Generic ESP8266 Module, Serial, 80 MHz, 40MHz, DIO, 115200, 512K (64K SPIFFS) on COM4

サンプル・スケッチ SDWebServer



The screenshot displays the SD Editor web interface in a browser window. The address bar shows the URL `esp8266sd.local/edit/`. The interface includes a file explorer on the left, a code editor in the center, and a console window at the bottom.

File Explorer: Lists various files and folders such as `mlo`, `u-boot.bin`, `uimage`, `recovery.img`, `part1.zip`, `part3.zip`, `part2.zip`, `upgrad~1`, `lost.dir`, `androi~1`, `mydir`, `msctest.txt`, `data.bin`, `data.csv`, `index.htm`, `test.mp3`, `goodmorn.wav`, `goodmore.wav`, `sunset.bmp`, `out.lcd`, `sunset.lcd`, `images`, `mimic.cfg`, `sample.wav`, `sd`, `system~1`, `edit`, and `pins.png`.

Code Editor: Contains the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta http-equiv="Content-type" content="text/html; charset=utf-8">
5   <title>ESP Index</title>
6   <style>
7     body {
8       background-color:black;
9       color:white;
10    }
11  </style>
12  <script type="text/javascript">
13    function onBodyLoad(){
14      console.log("we are loaded!!");
15    }
16  </script>
17 </head>
18 <body id="index" onload="onBodyLoad()">
19   <h1>ESP8266 Pin Functions テスト</h1>
20   <a href="/edit/">EDITOR</a>
21   
22 </body>
23 </html>
24
```

Console Window: Displays the following messages:

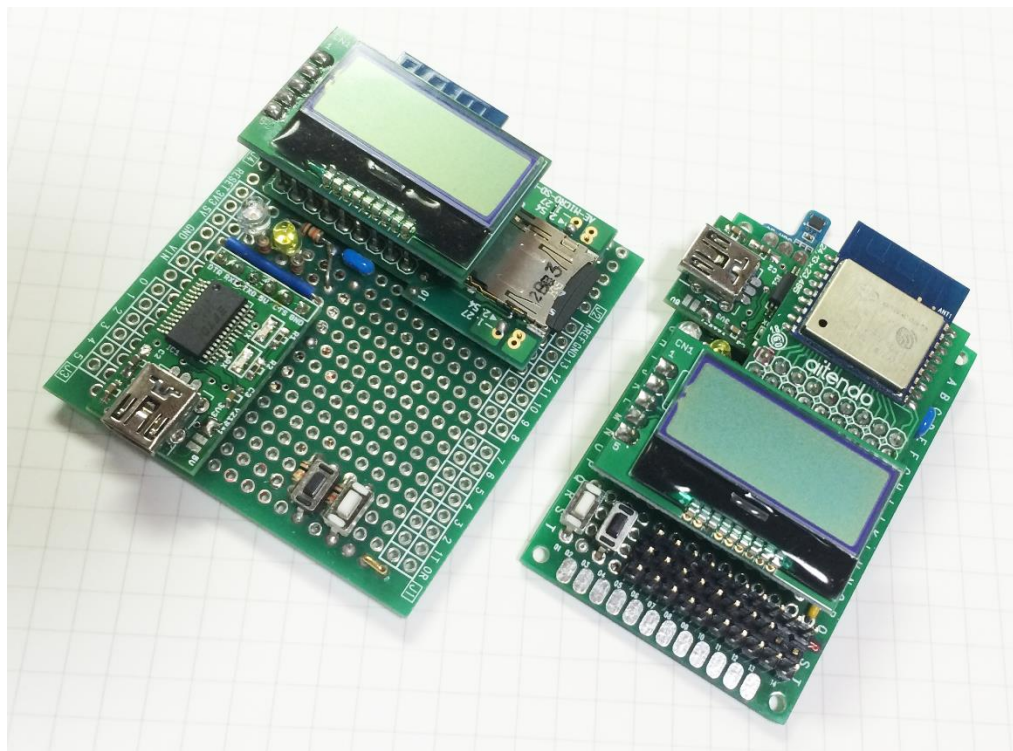
```
マイコンボードに書き込んでいます...
スケッチが プログラムストレージ領域の 298,632バイト (68%) を使用しています。最大は 434,160バイト です。
グローバル変数が 48,860バイト (59%) の 動的メモリを使用しており、ローカル変数に 33,060 バイトが残っています。
最高は 81,920バイトです。
```

The status bar at the bottom of the console window reads: `Generic ESP8266 Module, Serial, 80 MHz, 40MHz, DIO, 115200, 512K (64K SPIFFS) on COM4`.

[Ace: High performance code editor for the WEB.](#)

実装例

- ・ Aitendoブレイクアウト(B)を使用して実装した例



SDカード付き
ソフトテスト用

全部のピンを引出し
ポート構成確認用

IoTデバイス・デモ

• IoTデバイスの仕様(ESP-WROOM-02)

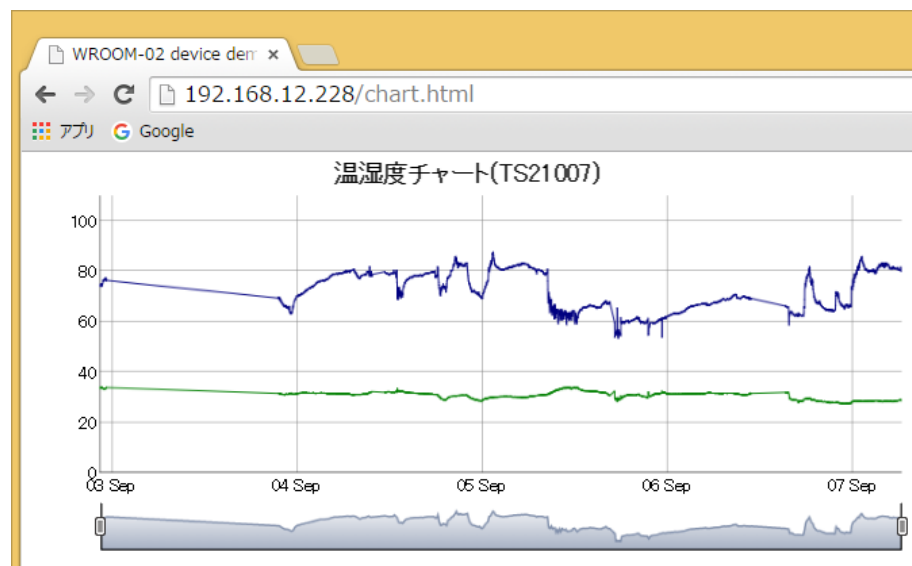
- 遠隔地に設置するデバイスに接続したセンサ（温度、湿度）からデータを読み取り、設定された間隔でhttp通信を使用してサーバに送出する
- 時刻合わせのために設定された間隔でntpサーバと時刻同期する

• サーバ (Intel® Edison)

- IoTデバイスから受信した観測データをデバイス毎にファイル保存する
- 保存したデータはWebブラウザで表示またはダウンロードできる(CSV形式)

```
Start v1.0.0 - TS21005
Connecting to Buffalo-G-B960
.....
WiFi connected
IP address: 192.168.12.224
Starting UDP
Local port: 8888
NTP 00:00:00 - 15
NTP 15:04:30 - 14
Connect to 192.168.12.228
HTTP/1.1 200 OK
WEB 15:05:00 = 97 : [38] TS21005=2015/09/06 15:05:00,24.9,64.7
NTP 15:05:00 - 13
NTP 15:05:30 - 14
Connect to 192.168.12.228
HTTP/1.1 200 OK
WEB 15:06:00 = 97 : [38] TS21005=2015/09/06 15:06:00,24.6,62.4
NTP 15:06:01 - 13
NTP 15:06:30 - 13
```

デバイス側



蓄積データをブラウザで表示

[Dygraphs: Dygraphs is a fast, flexible open source JavaScript charting library.](#)

IoTデバイス・デモ ネットワーク構成

サーバ (Intel® Edison)



携帯電話
アクセスポイント



ブラウザ

ID:21005

ID:21007

ID:21008

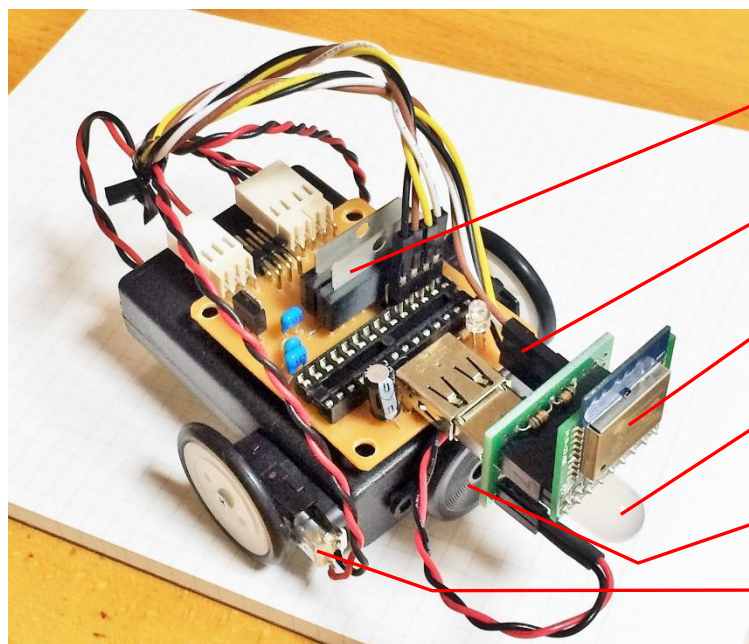


エッジ・デバイス

WiFiリモコン・デモ

・リモコン受信機の仕様(ESP-WROOM-02)

- 受信機はSoftAPモードで動作し、アクセスポイント無しで接続できる
 - 左右にギアモータを取付けた台車（バッテリーケース）に搭載し、iOSアプリからのコマンドを受信する
- コマンドには、左右のモータ回転速度、回転方向の制御、ヘッドライトの点灯、警笛の鳴動がある



フルブリッジ(TA7291Px2)

電池ボックス(単3x3)

ESP-WROOM-02

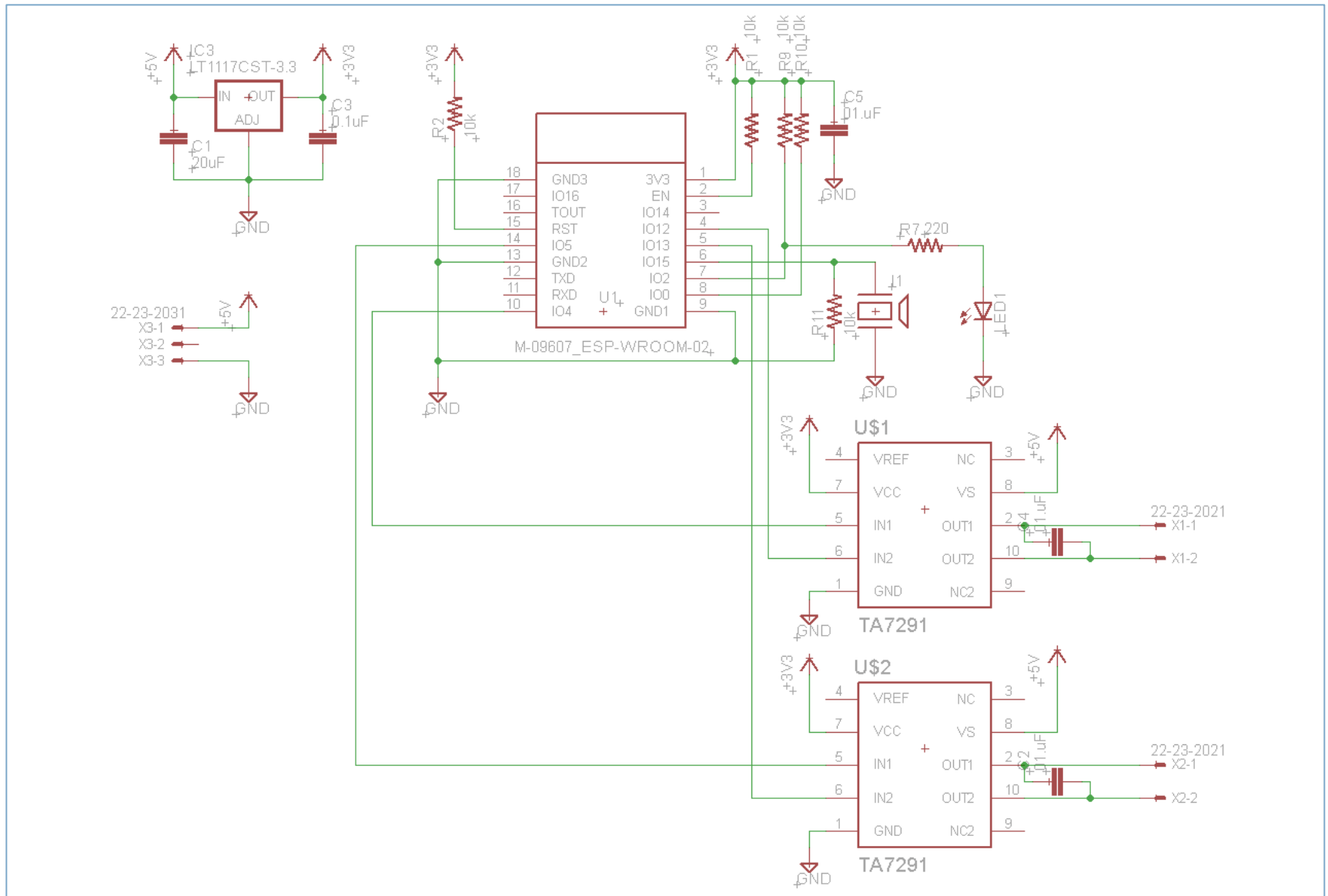
ヘッドライト(^-^;

クラクション(^-^;

モータ・ギヤ・ホイール(左右)

ちっちゃいものくらぶ：[マイコン直結で駆動できる超小型ギアモーターとプーリー・タイヤセット](#)

WiFiリモコン・デモ 受信側接続図



WiFiリモコン・デモ 受信側スケッチ

```
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>

#define LED_Light 2
#define LED_Hone 15
#define PWM_LF 4
#define PWM_LR 12
#define PWM_RF 5
#define PWM_RR 13

const char ssid[] = "ESPCar";
const char pass[] = "ESP8266ap";

WiFiUDP udp;
unsigned int localPort = 10000;
const int OSC_PACKET_SIZE = 256;
char packetBuffer[OSC_PACKET_SIZE];

void setup() {
  Serial.begin(115200);
  WiFi.softAP(ssid, pass);
  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: "); Serial.println(myIP);

  Serial.println("Starting UDP");
  udp.begin(localPort);
  Serial.print("Local port: "); Serial.println(udp.localPort());

  pinMode(LED_Light, OUTPUT);
  pinMode(LED_Hone, OUTPUT);
  pinMode(PWM_LF, OUTPUT);
  pinMode(PWM_LR, OUTPUT);
  pinMode(PWM_RF, OUTPUT);
  pinMode(PWM_RR, OUTPUT);
  analogWrite(PWM_LF, 0);
  analogWrite(PWM_LR, 0);
  analogWrite(PWM_RF, 0);
  analogWrite(PWM_RR, 0);
}
```

```
void loop() {
  int rlen, NoData = 0;
  int Val_L = 0, Val_R = 0, Vrf_L = 0, Vrf_R = 0, Val_F = 0, Val_C = 0;
  while (1) {
    if (!(rlen = udp.parsePacket())) {
      if (++NoData > 50) {
        analogWrite(PWM_LF, 0); analogWrite(PWM_LR, 0);
        analogWrite(PWM_RF, 0); analogWrite(PWM_RR, 0);
      }
      delay(10);
      continue;
    }
    NoData = 0;
    udp.read(packetBuffer, (rlen > OSC_PACKET_SIZE) ? OSC_PACKET_SIZE : rlen);
    if (strncmp(&packetBuffer[0], "/osc/", 5) == 0) {
      switch (packetBuffer[5]) {
        case 'F':
          Val_F = packetBuffer[15];
          digitalWrite(LED_Light, Val_F);
          break;
        case 'C':
          Val_C = packetBuffer[15];
          analogWrite(LED_Hone, Val_C ? 512 : 0);
          break;
        case 'L':
          analogWrite(PWM_LF, 0); analogWrite(PWM_LR, 0);
          Val_L = packetBuffer[15];
          if (Val_L >= (64 - 8) && Val_L <= (64 + 8)) { Val_L = 64; }
          if (Val_L >= 64)
            analogWrite(PWM_LF, (Val_L - 64) * 16);
          else analogWrite(PWM_LR, (63 - Val_L) * 16);
          break;
        case 'R':
          analogWrite(PWM_RF, 0); analogWrite(PWM_RR, 0);
          Val_R = packetBuffer[15];
          if (Val_R >= (64 - 8) && Val_R <= (64 + 8)) { Val_R = 64; }
          if (Val_R >= 64)
            analogWrite(PWM_RF, (Val_R - 64) * 16);
          else analogWrite(PWM_RR, (63 - Val_R) * 16);
          break;
      }
      Serial.printf("F=%d, C=%d, L=%4d, R=%4d\n", Val_F, Val_C, Val_L, Val_R);
    }
    delay(10);
  }
}
```

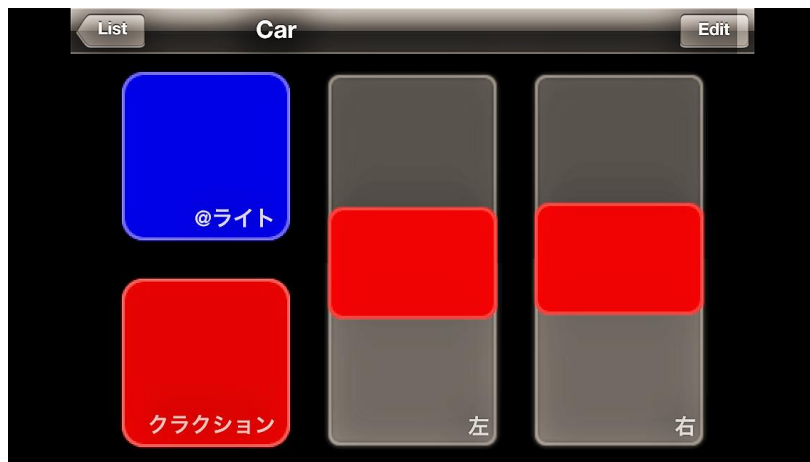
WiFiリモコン・デモ iOSC

・iOSCアプリ

– RECOTANA氏のiOSCを使用する

iPhineで操作するiOSCコントローラアプリです。

操作画面を用意されたテンプレートから選択し、配置されたボタンやスライダの操作状態をUDPパケットで送ります。



iOSCの操作画面

iOSC: recotana.com

UDPパケットの形式

- ヘッドライト(ボタン、オルタネイト動作)
`/osc/Fxx, ixx0123` ・ ・ 0123は4byteのバイナリ数値
0:ON、1:OFF
- 警笛(ボタン)
`/osc/Cxx, ixx0123` ・ ・ 0123は4byteのバイナリ数値
0:ON、1:OFF
- 左(スライダ)
`/osc/Lxx, ixx0123` ・ ・ 0123は4byteのバイナリ数値
0~127: 左スライダの位置
- 右(スライダ)
`/osc/Rxx, ixx0123` ・ ・ 0123は4byteのバイナリ数値
0~127: 右スライダの位置

iOSCからの制御データの形式

PICでもArduino Pinguino

- Pinguino : Arduino風の開発環境

PIC32 – PINGUINO



Pinguino

the first Arduino-compatible multi-target platform

The PIC32-Pinguino is an ARDUINO-like board with PIC32MX440F256H

Visit <http://pinguino.cc>
for all the latest information on pinguino software and hardware

INSTALLATION

Instruction resources –the pinguino.cc website

WINDOWS

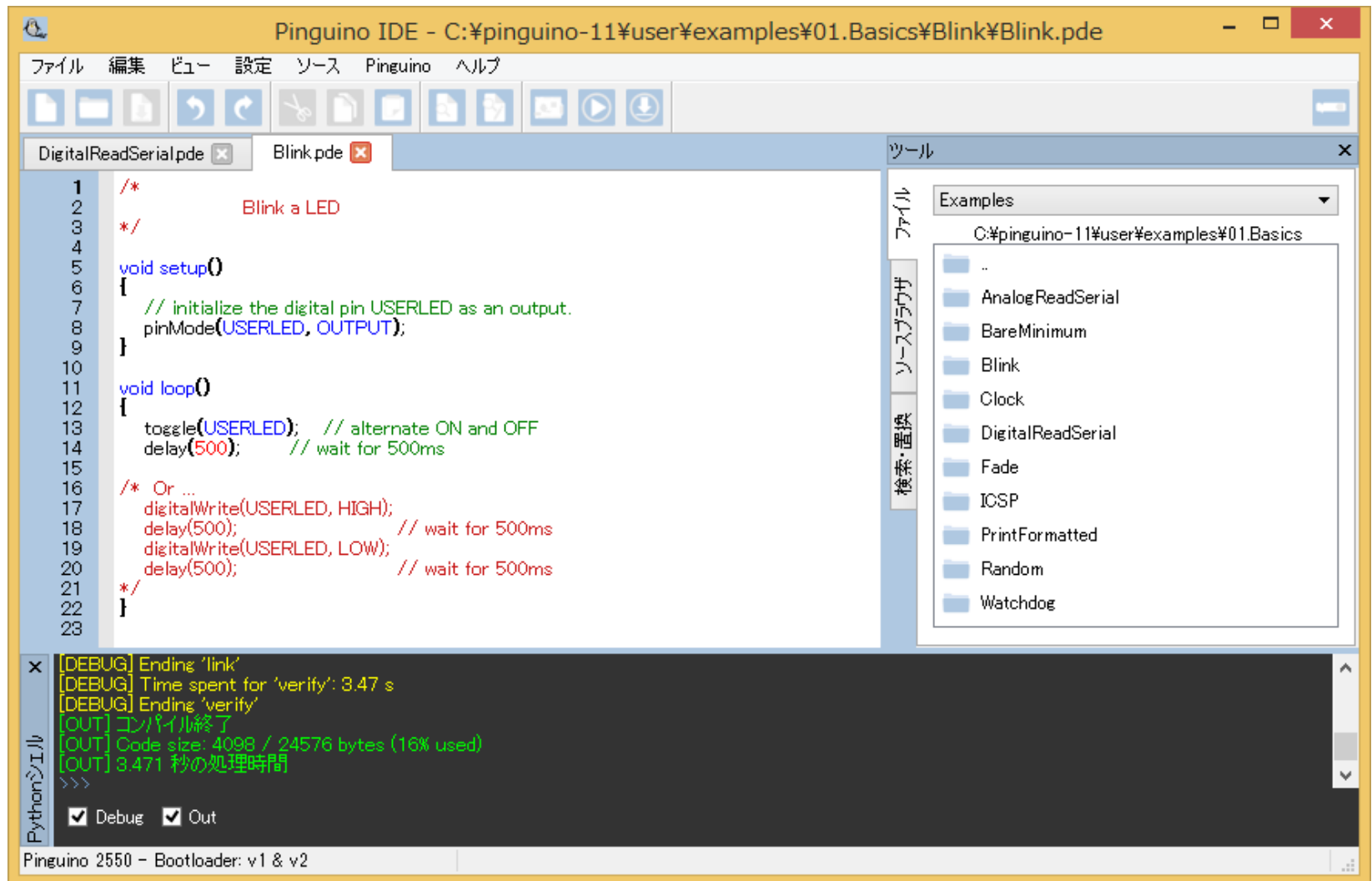
To download the IDE + compiler, go to this page:
<http://code.google.com/p/pinguino32/downloads/list>

Download the latest version for windows (most likely a single .exe file). Download it somewhere convenient and double-click on the .exe file.



8-bit : PIC18Fx450, PIC18Fx5(K)50, PIC18Fx6J50 and PIC18Fx7J53 family
32-bit : PIC32MX MIPS family
Pinguino: [PIC32 - Pinguino](#)

Pinguino IDE



Pinguino IDE: [Pinguino IDE download](#)