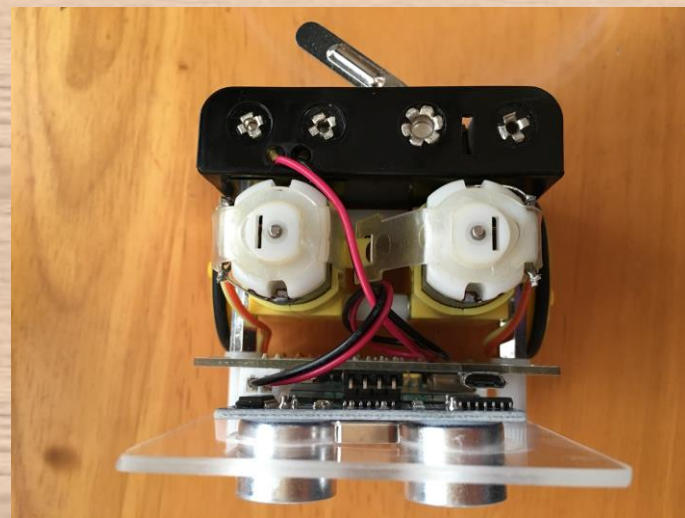


ロボットプログラミング



SmallBotを動かそう

SmallBotについて



マニュアルやプログラム類は githubに置いてあります



Manual.pdf

ハードウェアマニュアル

SmallBot_P01.pdf

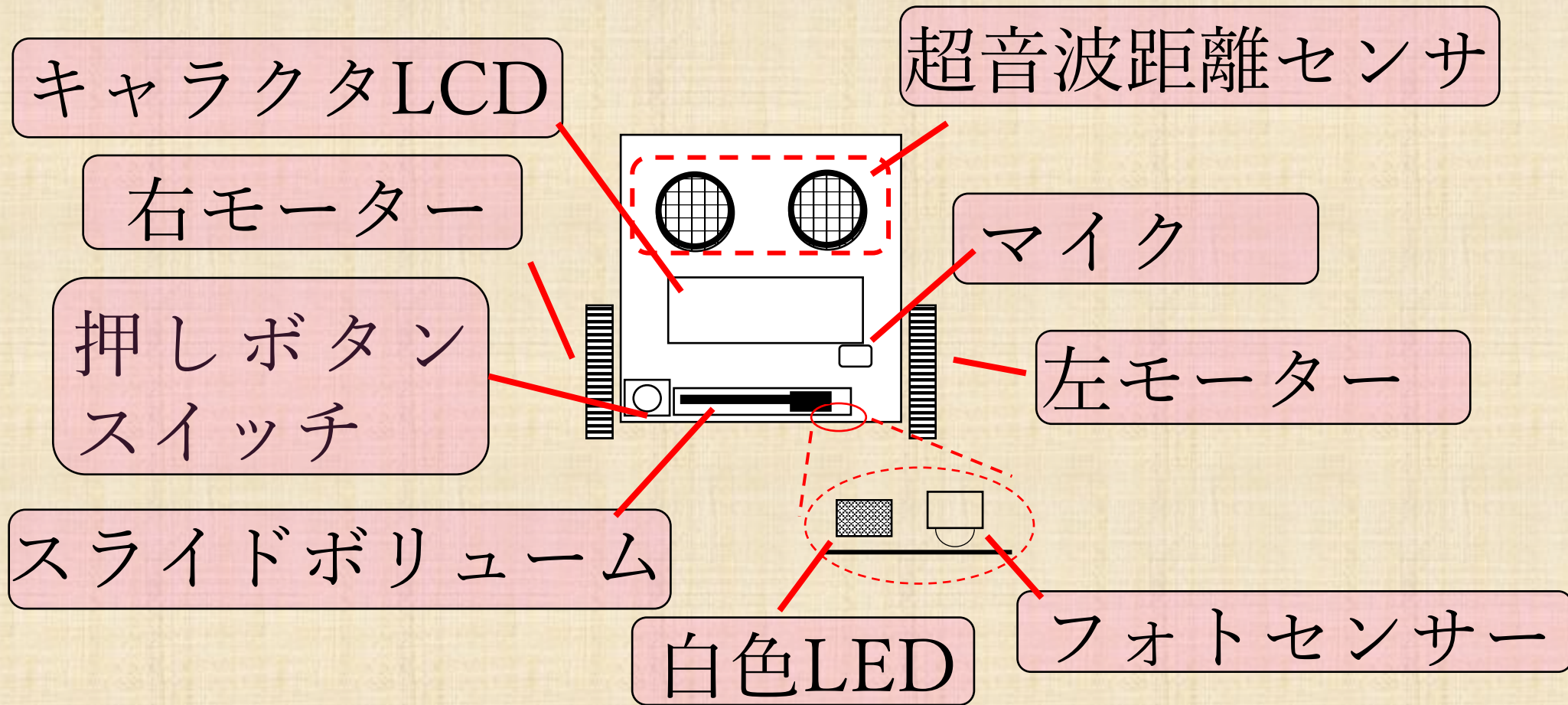
第一回テキスト

SmallbotV00.ino

サンプルプログラム

<https://github.com/neecrobot/smallbot>

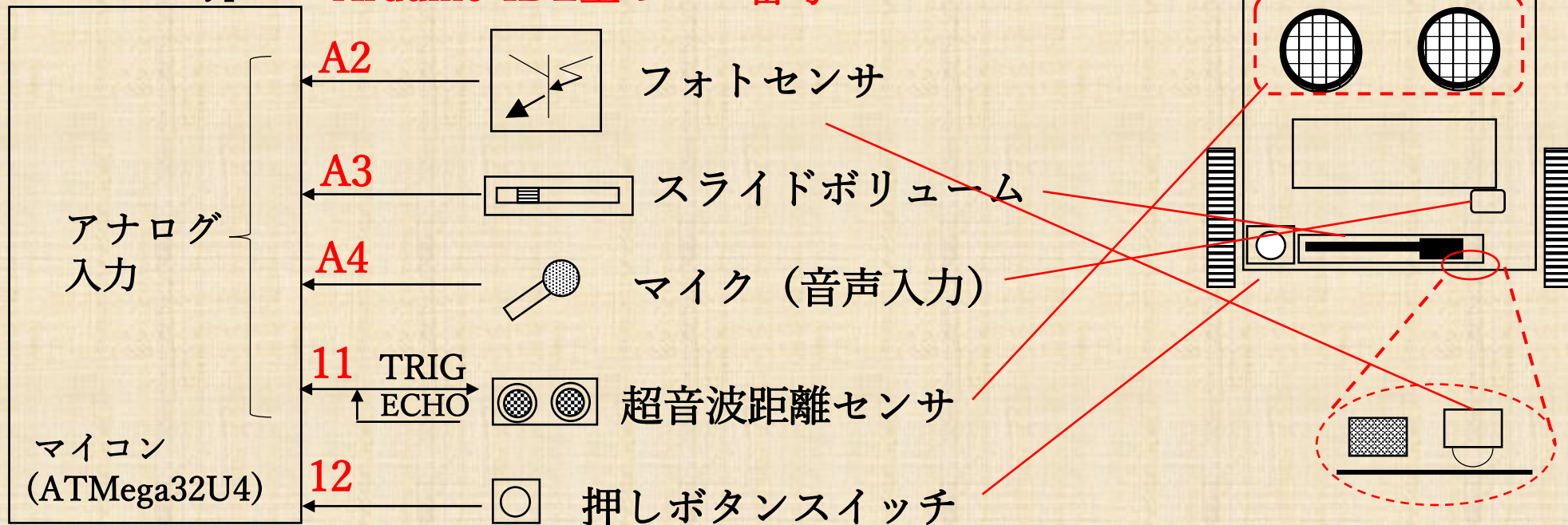
SmallBotの主要部品配置



入力信号の接続（参考）

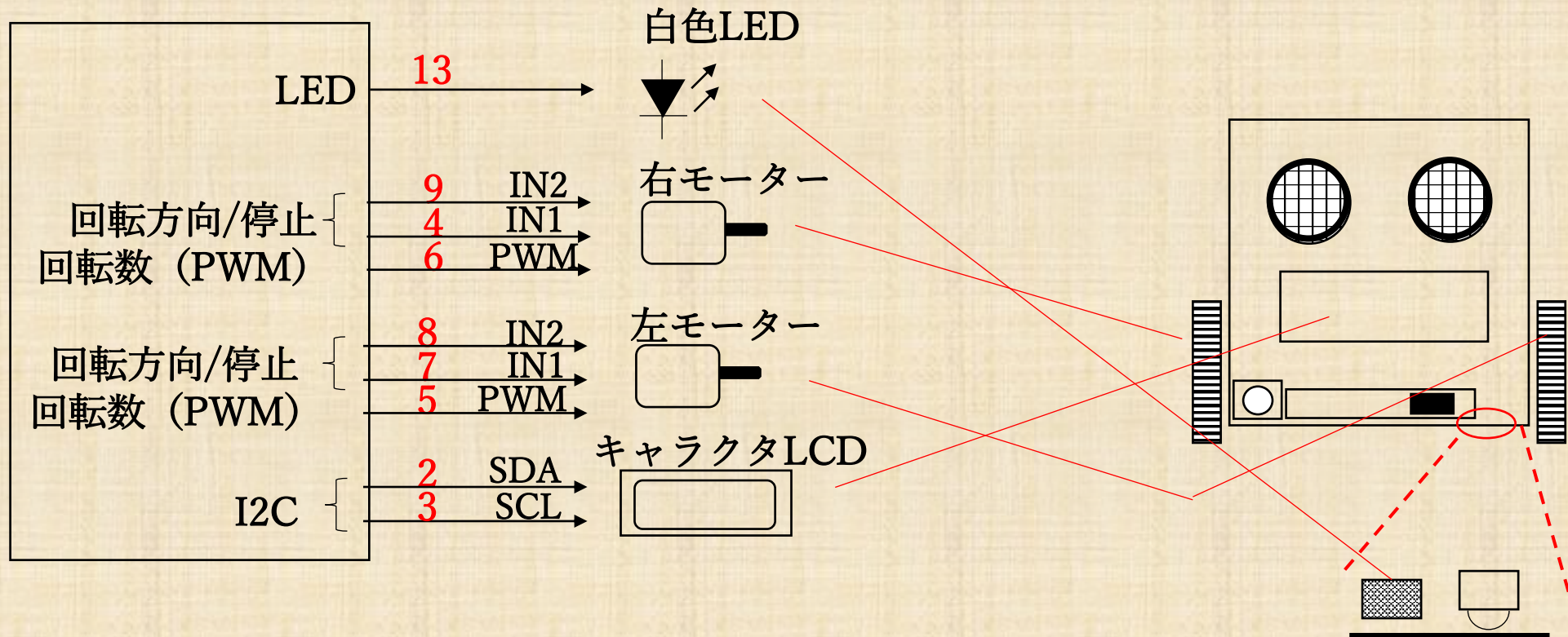
RDC-104 type II

Arduino-IDE上のピン番号

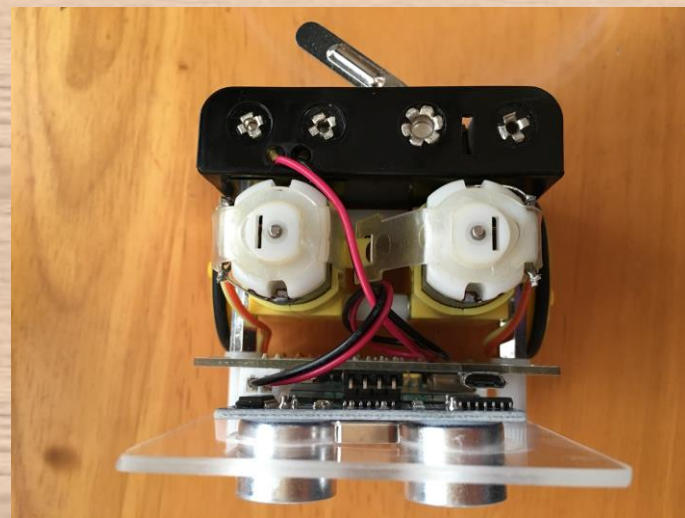


出力信号の接続（参考）

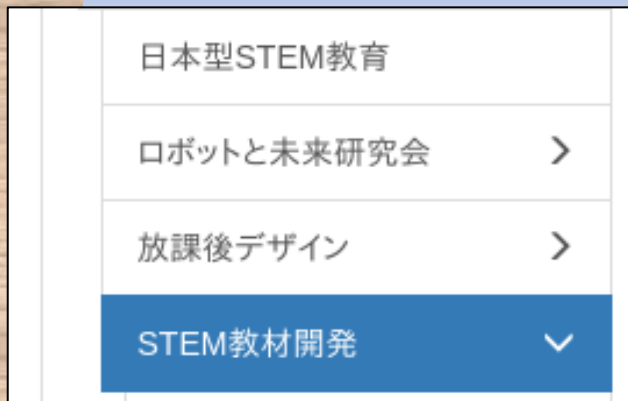
RDC-104 type II **Arduino-IDE上のピン番号**



SmallBot搭載マイコン用(RDC-104)の開発環境 インストール



SmallBot搭載マイコン用(RDC-104)の開発環境 インストール



「STEM プログラミング環境」で検索
=> 埼玉大学のSTEM教育研究センター

http://neo.stem-edulab.org/page_20200820030156/page_20201016230943

Arduino 1.8.19をインストール

Legacy IDE (1.8.X)



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or newer  [Get](#)

Linux 32 bits

Linux 64 bits

Linux ARM 32 bits

Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#)

[Checksums \(sha512\)](#)

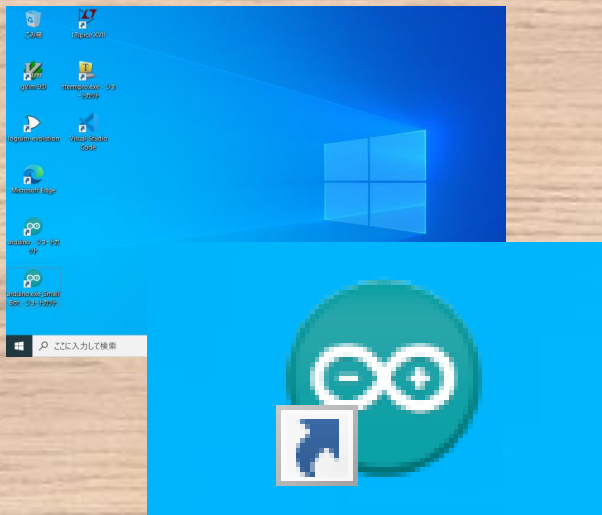
DOWNLOAD OPTIONS

Windows Win 7 and newer

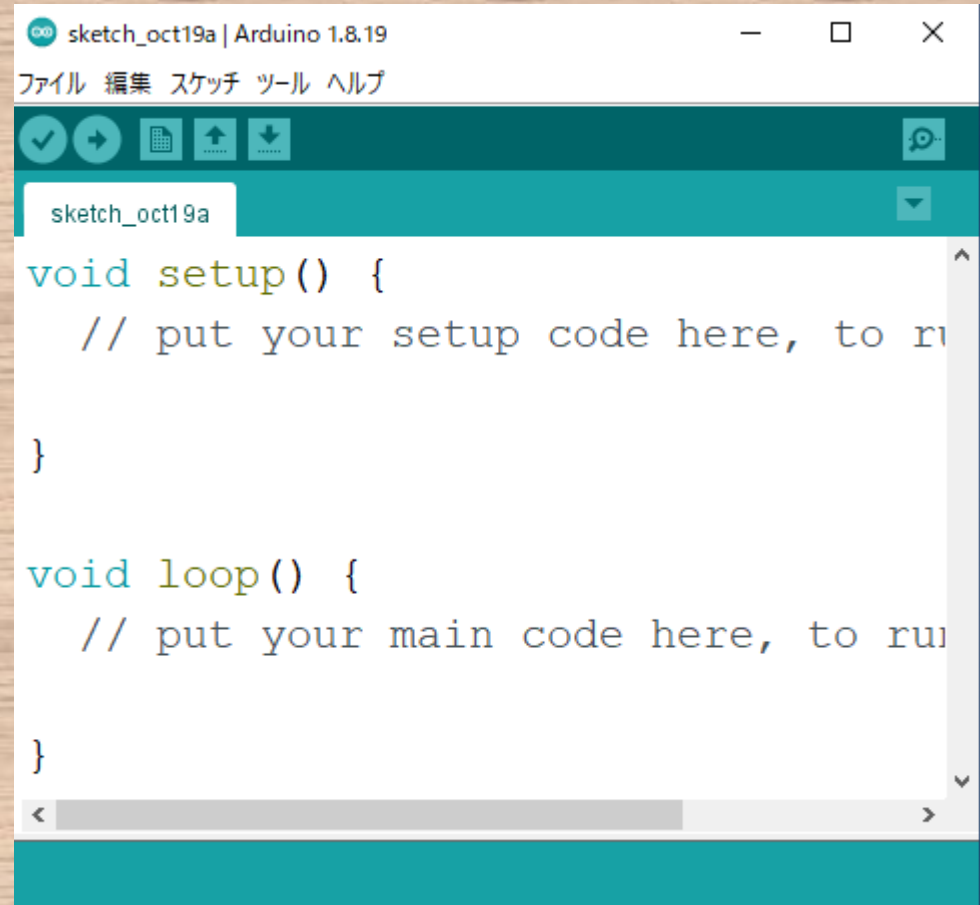
Windows ZIP file

<https://www.arduino.cc/en/software>

Arduino IDEの起動



既存のプログラムが開いたら
ファイル=>新規ファイル



ボードマネージャのURLをコピー

STEM教育研究センター 埼玉大学教育学部野村研究室



OK

キャンセル

2. 上の図のように「追加のボードマネージャのURL」の右の欄に、使用するSTEM Duマイコンボードのバージョンに合わせて、次のいずれかのURLをコピー&ペーストします。

【STEM Du RDC-ESP32コントローラの場合】

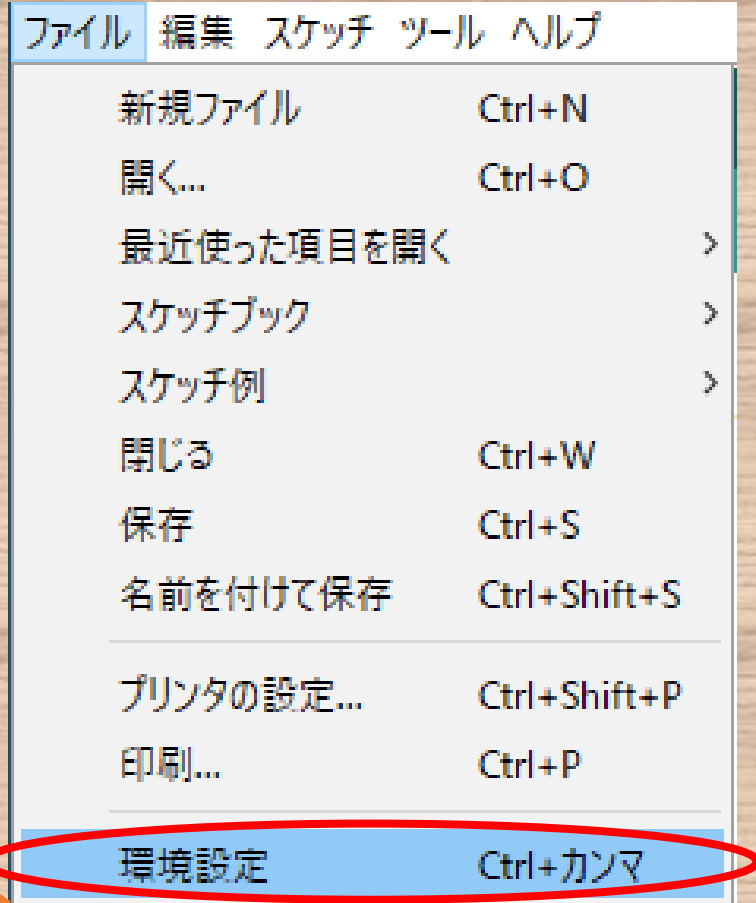
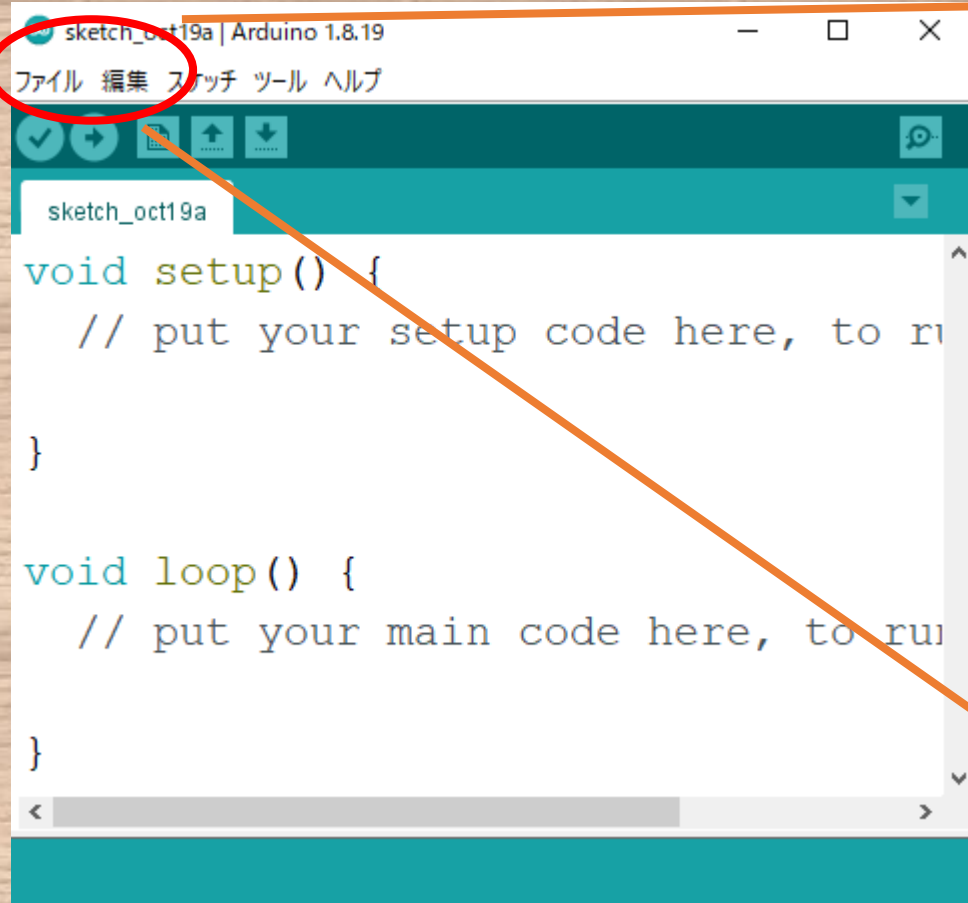
https://github.com/tairo/STEMDu_ArduinoIDE/releases/download/v1.0/package_stemdu-esp32_index.json

【STEM Du RDC-104までのコントローラの場合】

https://github.com/tairo/STEMDu_ArduinoIDE/releases/download/v1.0/package_stemdu-avr_index.json

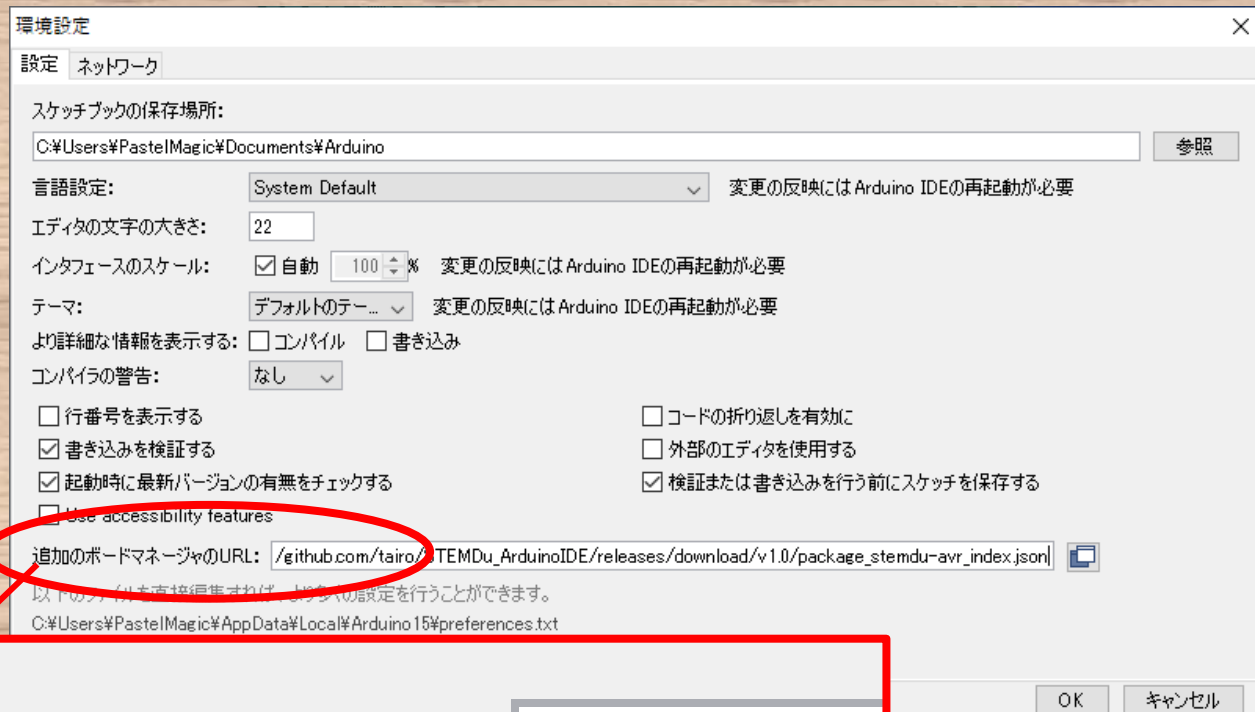
https://github.com/tairo/STEMDu_ArduinoIDE/releases/download/v1.0/package_stemdu-avr_index.json

ボード（RDC-104）の追加



ボードマネージャのURL追加

コピーしたURLを
ペースト



追加のボードマネージャのURL: /github.com

ボードマネージャを起動

ツール ヘルプ

自動整形

Ctrl+T

スケッチをアーカイブする

エンコーディングを修正

ライブラリを管理...

Ctrl+Shift+I

シリアルモニタ

Ctrl+Shift+M

シリアルプロッタ

Ctrl+Shift+L

WiFi101 / WiFiNINA Firmware Updater

ボード: "STEM Du RDC-104 w/ ATmega32U4 3.3V 8MHz"

シリアルポート

ボードマネージャ...

Arduino AVR Boards

ツール => ボード => ボードマネージャ

STEM Du AVRをインストール

ボードマネージャ

"STEM"で検索

タイプ 全て

STEM

STEM Du AVR

by Tairo Nomura

このパッケージに含まれているボード：

STEM Du 102, STEM Du 103, STEM

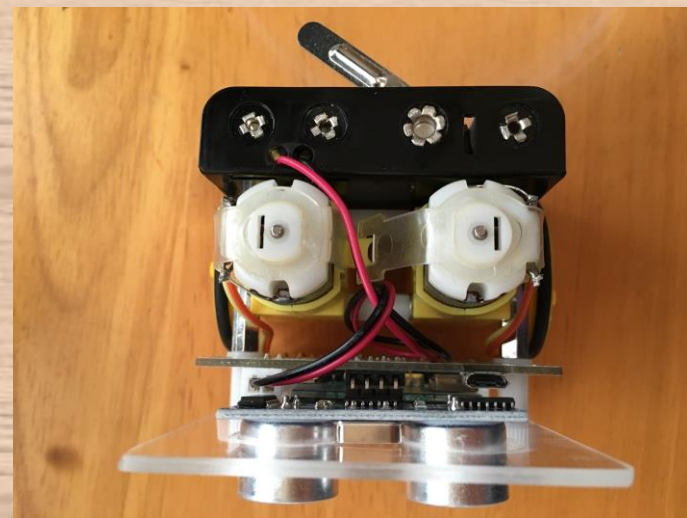
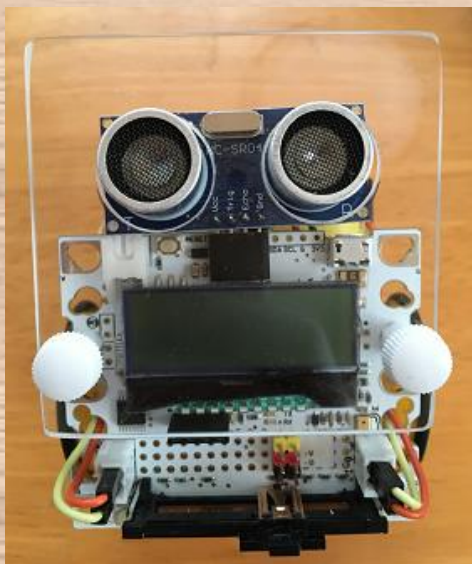
[More Info](#)

"インストール"
をクリック

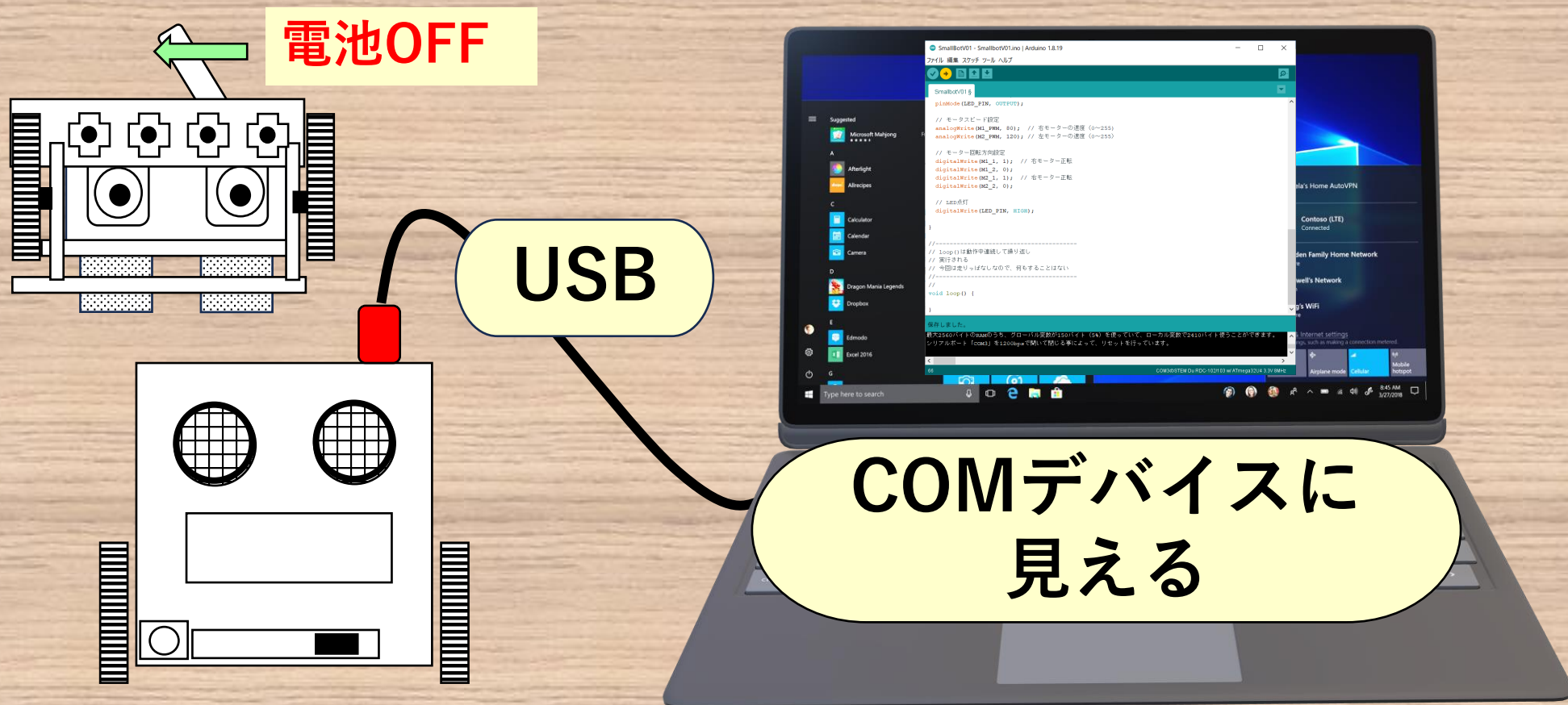
1.2... ▾

インストール

書き込んで動かすまでの手順

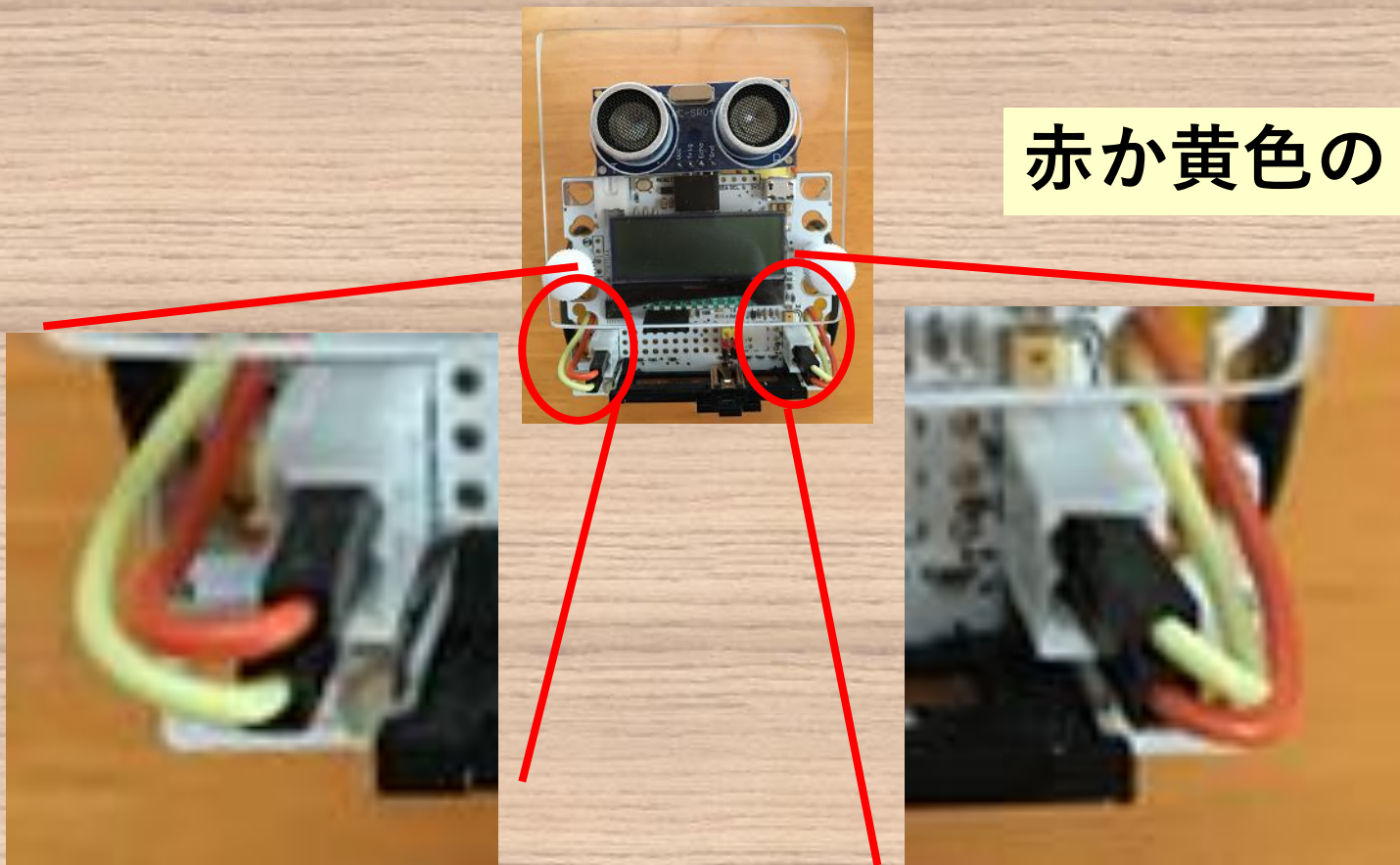


PCと接続

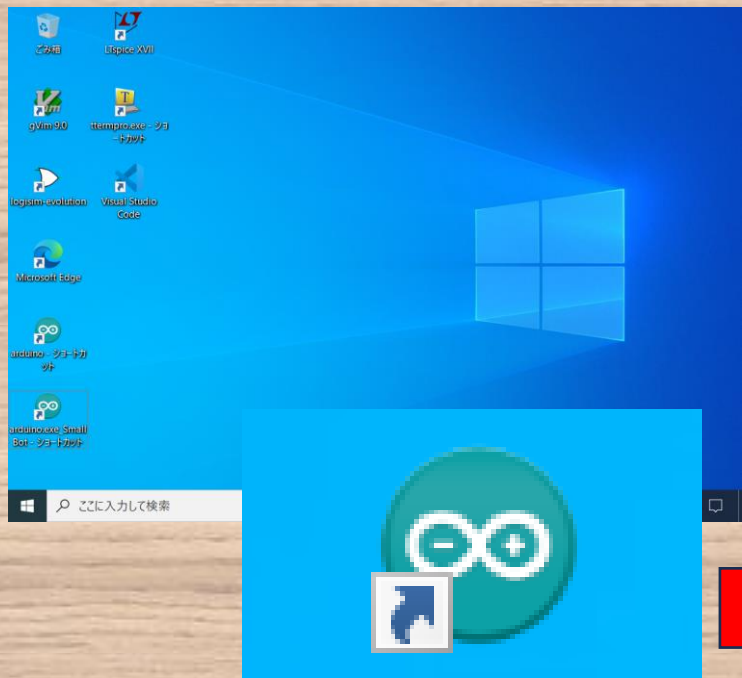


USB電源で動いてしまう時

赤か黄色のどちらかを抜く



Arduino IDEの起動

A screenshot of the Arduino IDE interface. The title bar reads 'SmallBotV01 - SmallbotV01.ino | Arduino 1.8.19'. The menu bar includes 'ファイル', '編集', 'スケッチ', 'ツール', and 'ヘルプ'. The toolbar contains icons for opening, saving, and running. The main text area shows the following code:

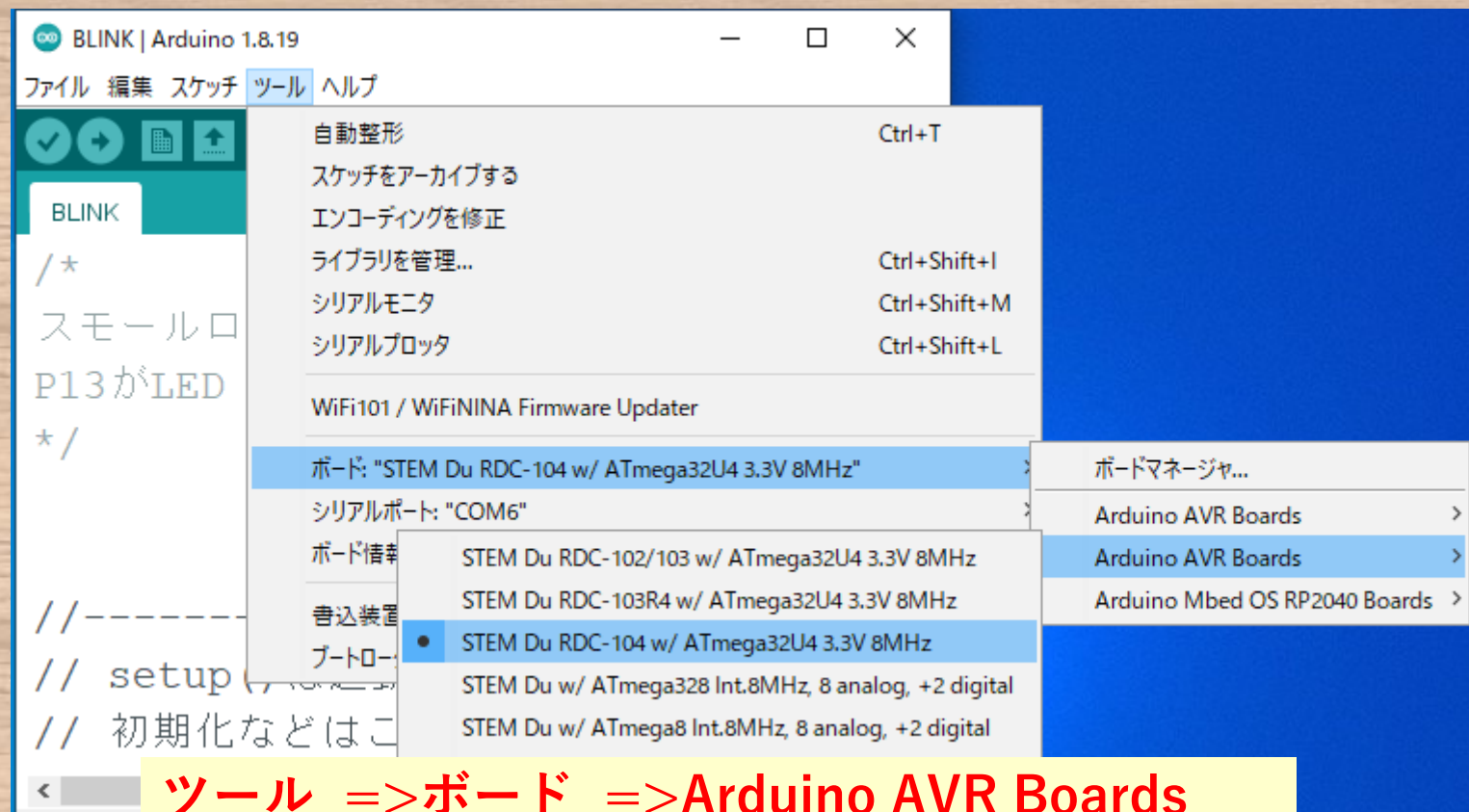
```
SmallbotV01 $  
  
pinMode(LED_PIN, OUTPUT);  
  
// モータースピード設定  
analogWrite(M1_PWM, 80); // 右モーターの速度 (0~255)  
analogWrite(M2_PWM, 120); // 左モーターの速度 (0~255)  
  
// モーター回転方向設定  
digitalWrite(M1_1, 1); // 右モーター正転  
digitalWrite(M1_2, 0);  
digitalWrite(M2_1, 1); // 右モーター正転  
digitalWrite(M2_2, 0);  
  
// LED点灯  
digitalWrite(LED_PIN, HIGH);  
  
}  
  
//-----  
// loop() は動作中連続して繰り返し  
// 実行される  
// 今回は走りっぱなしなので、何もすることはない  
//-----  
//  
void loop() {  
  
}
```

保存しました。

最大2560バイトのRAMのうち、グローバル変数が150バイト（5%）を使っていて、ローカル変数で2410バイト使うことができます。シリアルポート「COM3」を1200bpsで開いて閉じる事によって、リセットを行っています。

66 COM3のSTEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz

ターゲットの選択



**ツール => ボード => Arduino AVR Boards
=> STEM Du RDC-104.... を選択**

COM (シリアル) ポートの選択

**COM(シリアル) ポートの番号はPC
環境によって異なる**

ツール

=>シリアルポート

=>STEM Du RDC-102/103 w/.....

ツール ヘルプ

- 自動整形
- スケッチをアーカイブする
- エンコーディングを修正
- ライブラリを管理...
- シリアルモニタ
- シリアルプロッタ

WiFi101 / WiFinINA Firmware Updater

ボード: "STEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz"

シリアルポート: "COM3 (STEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz)"

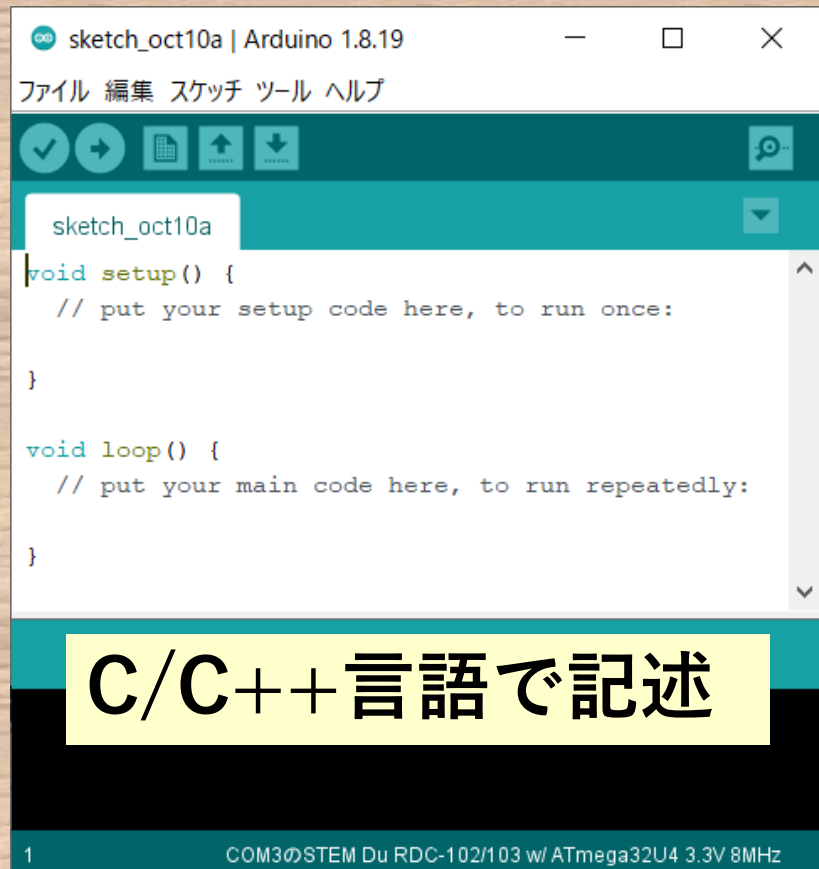


シリアルポート

- COM1
- ✓ COM3 (STEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz)

プログラムの記述

(setup()やloop()を消去しないこと)



```
void setup() {
```

起動時最初に
一回だけ実行される

```
}
```

```
void loop() {
```

繰り返し実行される

```
}
```

とりあえずLED点滅



```
BLINK | Arduino 1.8.19
ファイル 編集 スケッチ ツール ヘルプ

BLINK.g
/*
 * スモールロボットLED点滅
 * P13がLED
 */

//-----
// setup() は起動時最初に一回だけ実行される
// 初期化などはここで行う
//-----
//
void setup() {
  // LEDのピン
  pinMode(13, OUTPUT);
}

//
コンパイルが完了しました。
最大2093056バイトのフラッシュメモリのうち、スケッチが52308バイト (2%) を
最大262144バイトのRAMのうち、グローバル変数が10224バイト (3%) を使ってし
COM10 Raspberry Pi Pico, 2MB (no FS), 133 MHz, Small (0d) (standard), Disabled, Disabled, Disabled, Disabled, None, Pico SDK, iPod Only, Default (UF2)
```

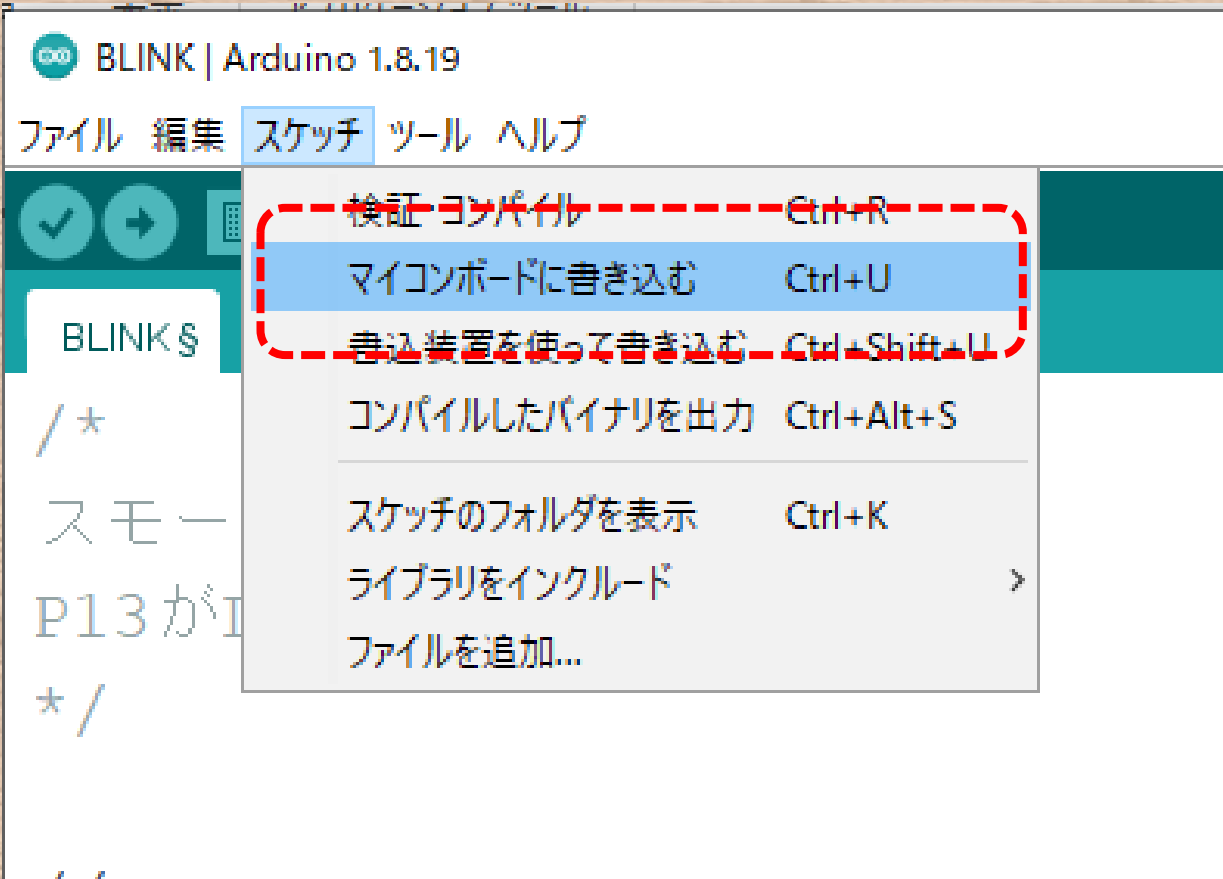
"//"**以下はコメント**
(入力不要)

```
void setup() {
  pinMode(13, OUTPUT);
}
```

```
void loop() {
  digitalWrite(13,1); // ON
  delay(500); // 500msウェイト
  digitalWrite(13,0); // OFF
  delay(500); // 500msウェイト
}
```

プログラム本体はすべて半角文字 (スペースも半角！)

ビルド&書き込み&実行

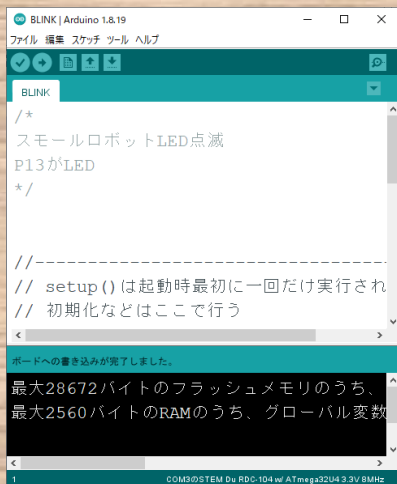


接続したSmallBotに
書き込みを行う

ビルドも自動的に行
われる

書き込み実行

ビルド中



```
BLINK [Arduino 1.8.19]
ファイル 編集 スケッチ ツール ヘルプ

BLINK
/*
 * スモールロボットLED点滅
 * P13がLED
 */

//-----
// setup() は起動時最初に一回だけ実行され
// 初期化などはここで行う
//-----

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // LEDを点灯
  delay(1000);                       // 1秒間待機
  digitalWrite(LED_BUILTIN, LOW);    // LEDを消灯
  delay(1000);                       // 1秒間待機
}
```

スケッチをコンパイルしています...

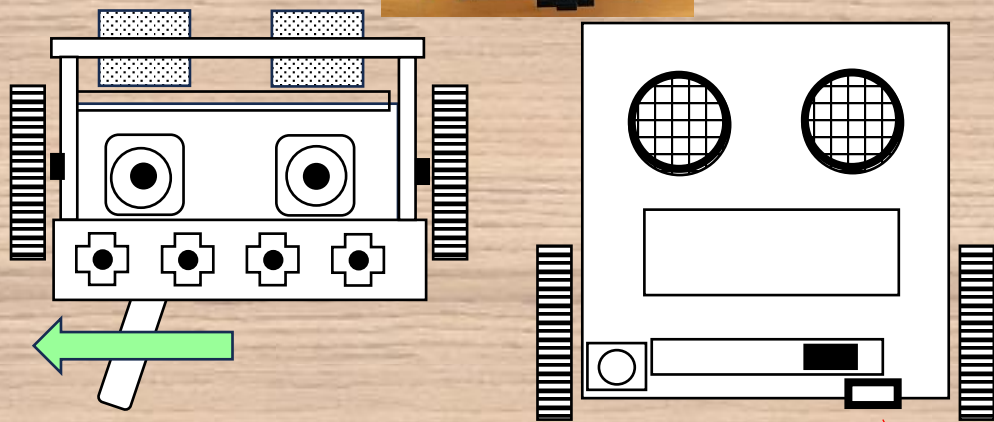


ボードへの書き込みが完了しました。

最大28672バイトのフラッシュメモリのうち、
最大2560バイトのRAMのうち、グローバル変数

書き込み完了

電池から電源を供給



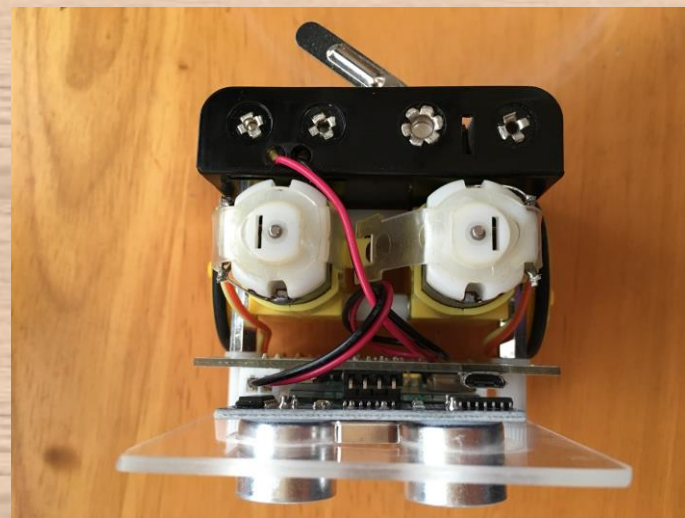
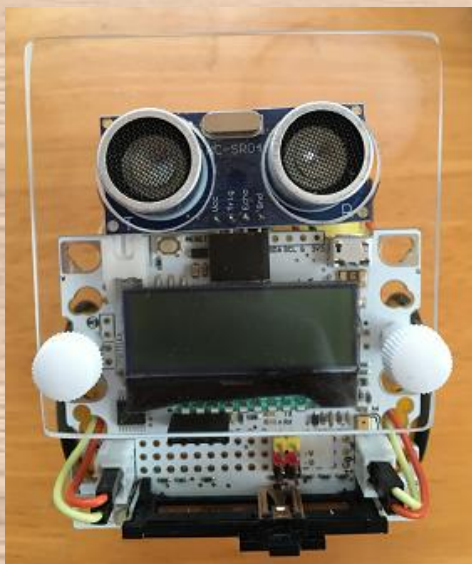
電池ON

白色LEDが点滅

基板上のLEDが1秒周期で点滅する

点滅の周期などを書き換えて動作を確認してみよう

モーターを動かそう



最初のロボットプログラム(V00.ino)

(setup()の中に記述)

```
void setup() {  
  // モータ関係の出力ピン  
  pinMode(4, OUTPUT);  
  pinMode(9, OUTPUT);  
  pinMode(6, OUTPUT);  
  pinMode(7, OUTPUT);  
  pinMode(8, OUTPUT);  
  pinMode(5, OUTPUT);  
  // デジタル出力ピン (LED)  
  pinMode(13, OUTPUT);
```

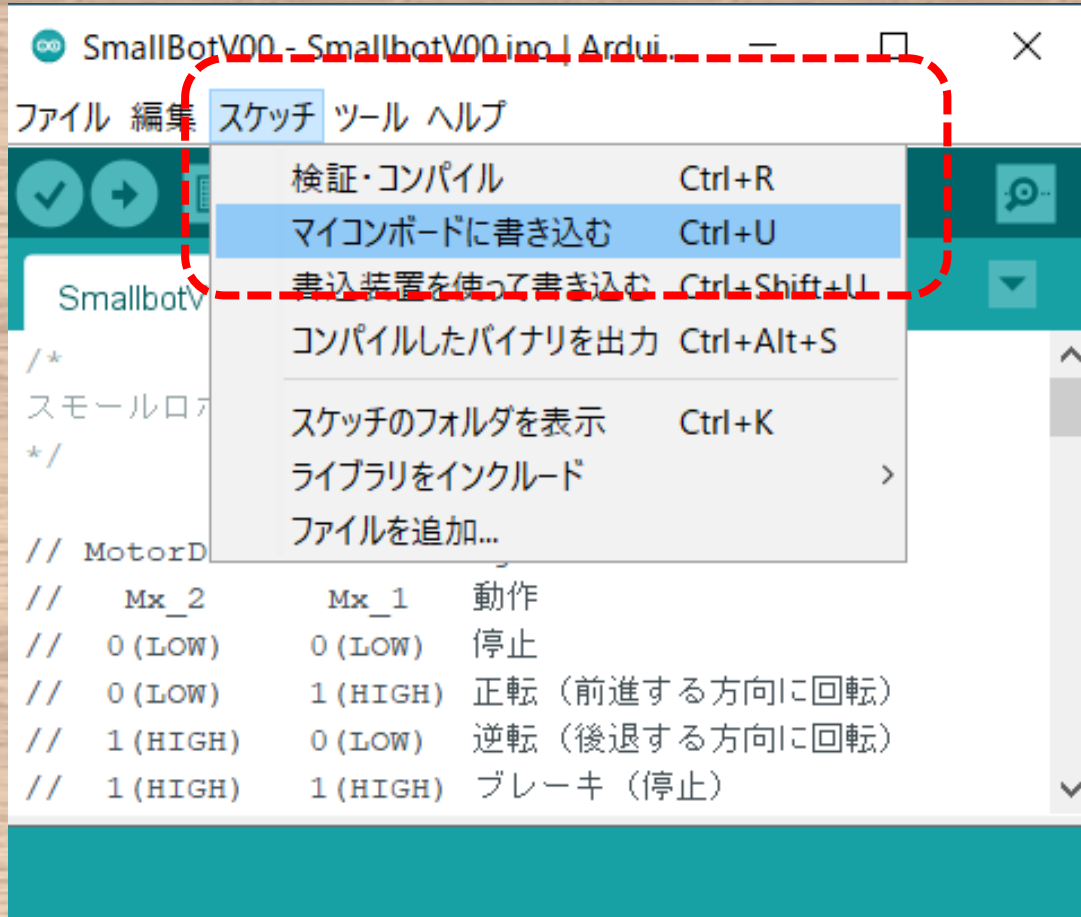
```
  // 右モーターの速度 (0~255)  
  analogWrite(6, 80);  
  // 左モーターの速度 (0~255)  
  analogWrite(5, 120);  
  // 右モーター正転  
  digitalWrite(4, 1);  
  digitalWrite(9, 0);  
  // 左モーター正転  
  digitalWrite(7, 1);  
  digitalWrite(8, 0);  
}
```

速度 80

速度 120

プログラム本体はすべて半角文字 (スペースも半角！)

ビルド&書き込み&実行

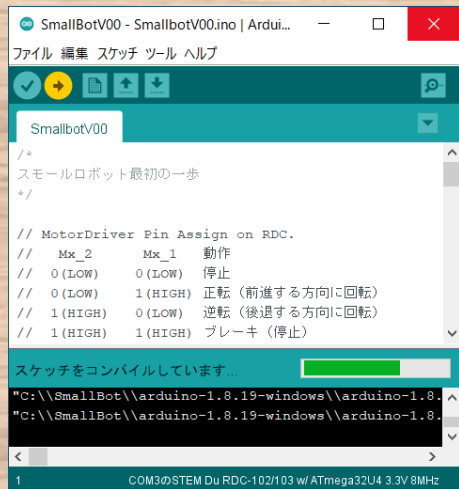


接続したSmallBotに書き込みを行う

ビルドも自動的行われる

書き込み実行

ビルド中



SmallBotV00 - SmallbotV00.ino | Ardui...
ファイル 編集 スケッチ ツール ヘルプ

SmallbotV00

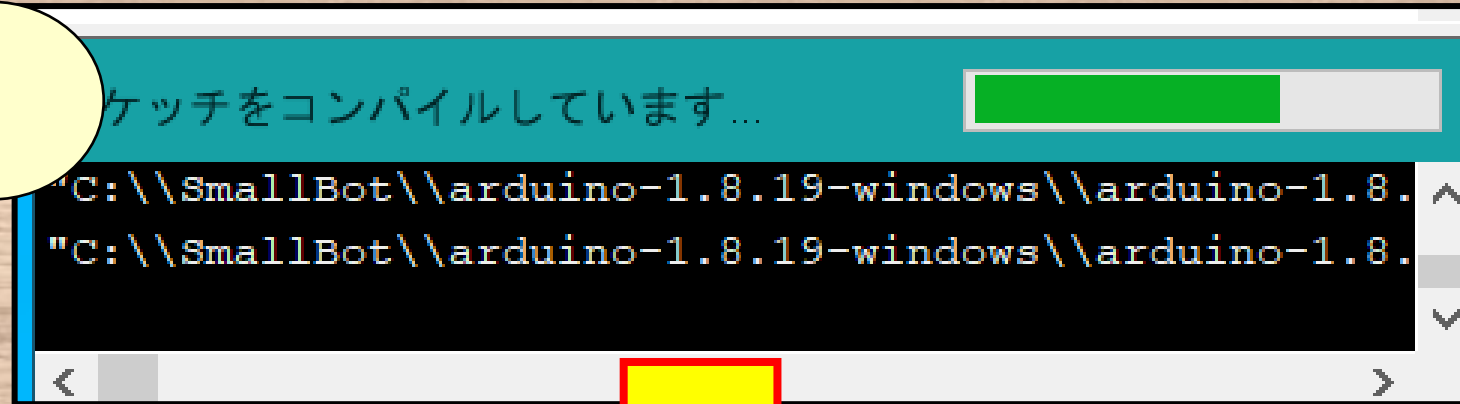
```
/*  
スモールロボット 最初の一步  
*/  
  
// MotorDriver Pin Assign on RDC.  
// Mx_2 Mx_1 動作  
// 0 (LOW) 0 (LOW) 停止  
// 0 (LOW) 1 (HIGH) 正転 (前進する方向に回転)  
// 1 (HIGH) 0 (LOW) 逆転 (後退する方向に回転)  
// 1 (HIGH) 1 (HIGH) ブレーキ (停止)
```

スケッチをコンパイルしています...

```
"C:\\SmallBot\\arduino-1.8.19-windows\\arduino-1.8.  
"C:\\SmallBot\\arduino-1.8.19-windows\\arduino-1.8.
```

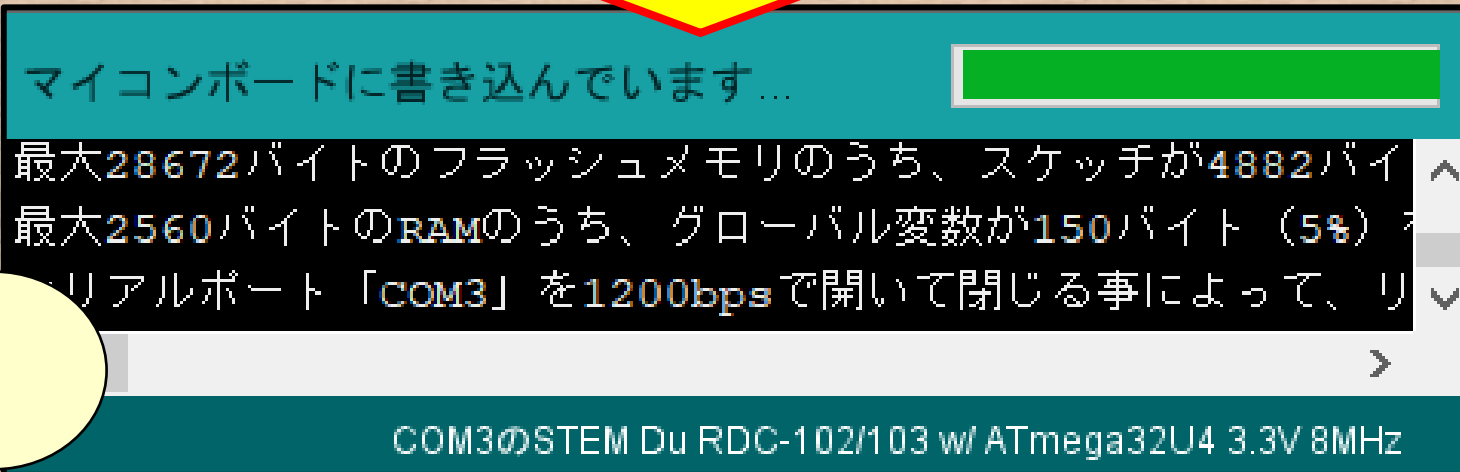
1 COM3のSTEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz

書込完了



スケッチをコンパイルしています...

```
"C:\\SmallBot\\arduino-1.8.19-windows\\arduino-1.8.  
"C:\\SmallBot\\arduino-1.8.19-windows\\arduino-1.8.
```

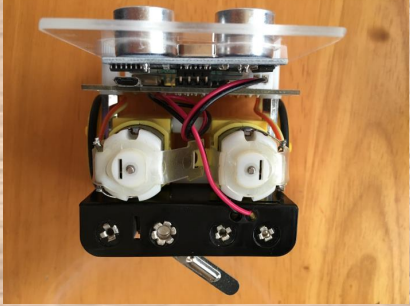


マイコンボードに書き込んでいます...

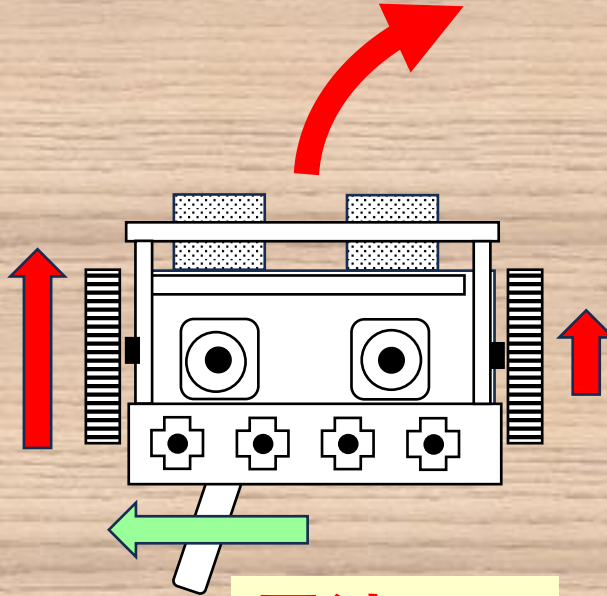
最大28672バイトのフラッシュメモリのうち、スケッチが4882バイト
最大2560バイトのRAMのうち、グローバル変数が150バイト (5%)
リアルポート「COM3」を1200bpsで開いて閉じる事によって、リ

COM3のSTEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz

時計回りに回転する



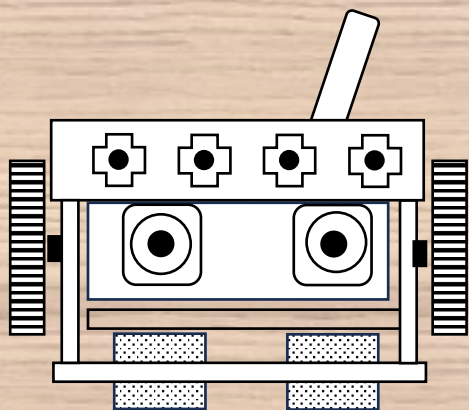
左モーター
速度 120



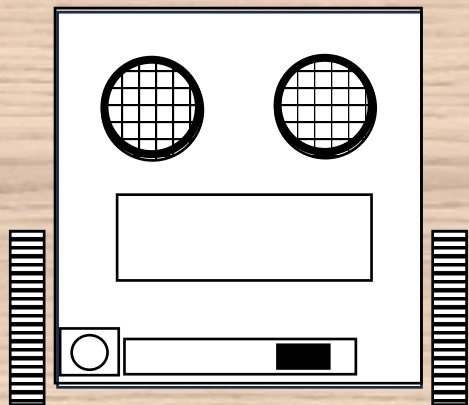
右モーター
速度 80

電池ON

練習



プログラムを書き換えて
いろいろな動きをさせてみよう



モーターを逆回転等させてみよう

(00:停止 11:ブレーキ)

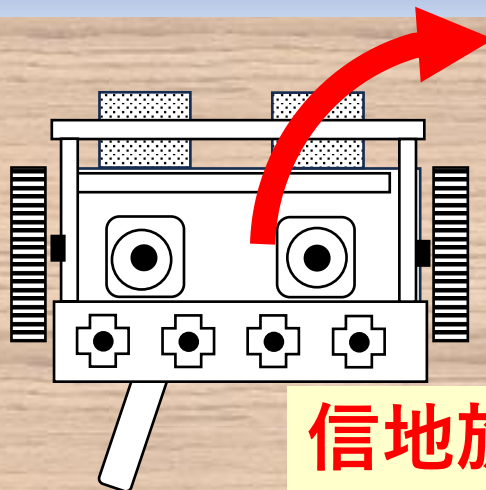
```
void setup() {  
  // モータ関係の出力ピン  
  pinMode(4, OUTPUT);  
  pinMode(9, OUTPUT);  
  pinMode(6, OUTPUT);  
  pinMode(7, OUTPUT);  
  pinMode(8, OUTPUT);  
  pinMode(5, OUTPUT);  
  // デジタル出力ピン (LED)  
  pinMode(13, OUTPUT);
```

```
// 右モーターの速度 (0~255)  
  analogWrite(6, 80);  
// 左モーターの速度 (0~255)  
  analogWrite(5, 120);  
// 右モーター逆転  
  digitalWrite(4, 0);  
  digitalWrite(9, 1);  
// 左モーター逆転  
  digitalWrite(7, 0);  
  digitalWrite(8, 1);  
}
```

プログラム本体はすべて半角文字 (スペースも半角！)

旋回

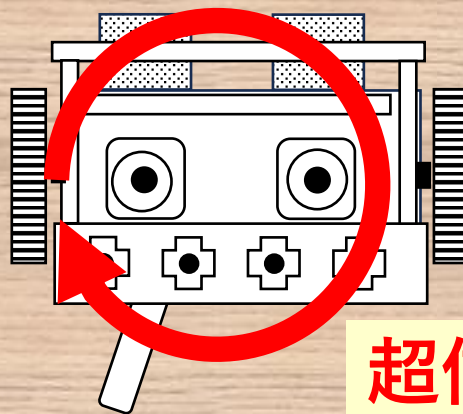
左モーター
速度 120



右モーター
速度 0

信地旋回

左モーター
速度 120

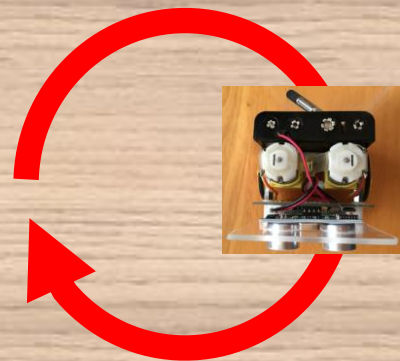


右モーター
速度 120

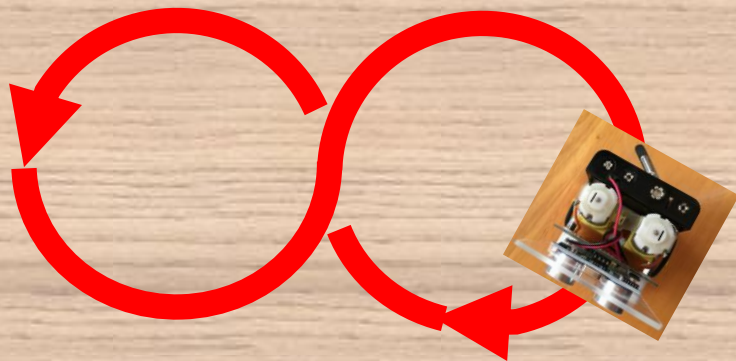


超信地旋回

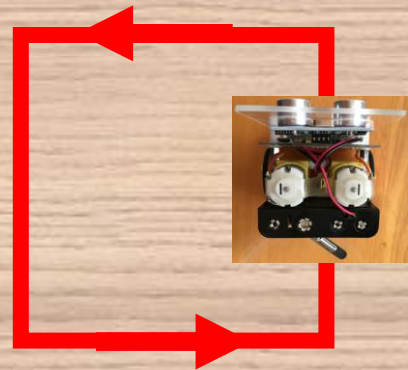
いろいろな動きをさせてみよう



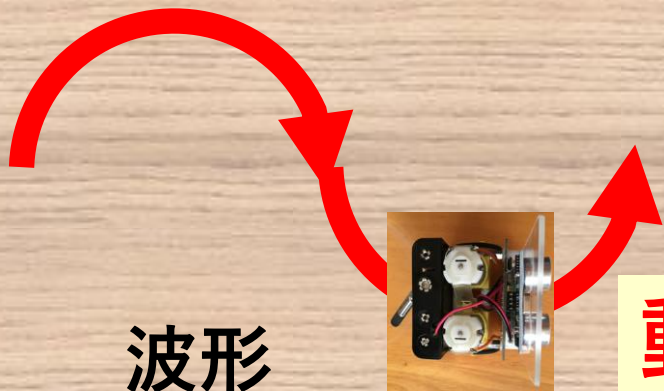
円



8の字

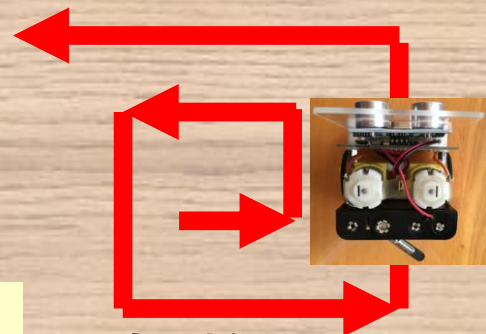


正方形



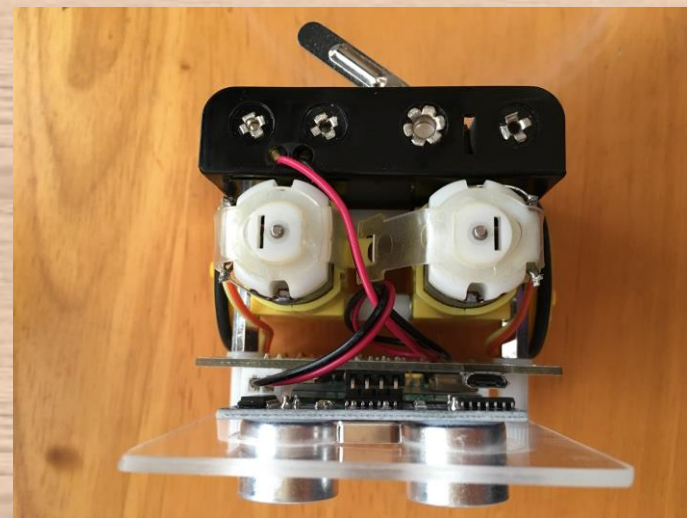
波形

動作パターン例



螺旋状

プログラム（ライブラリ）について

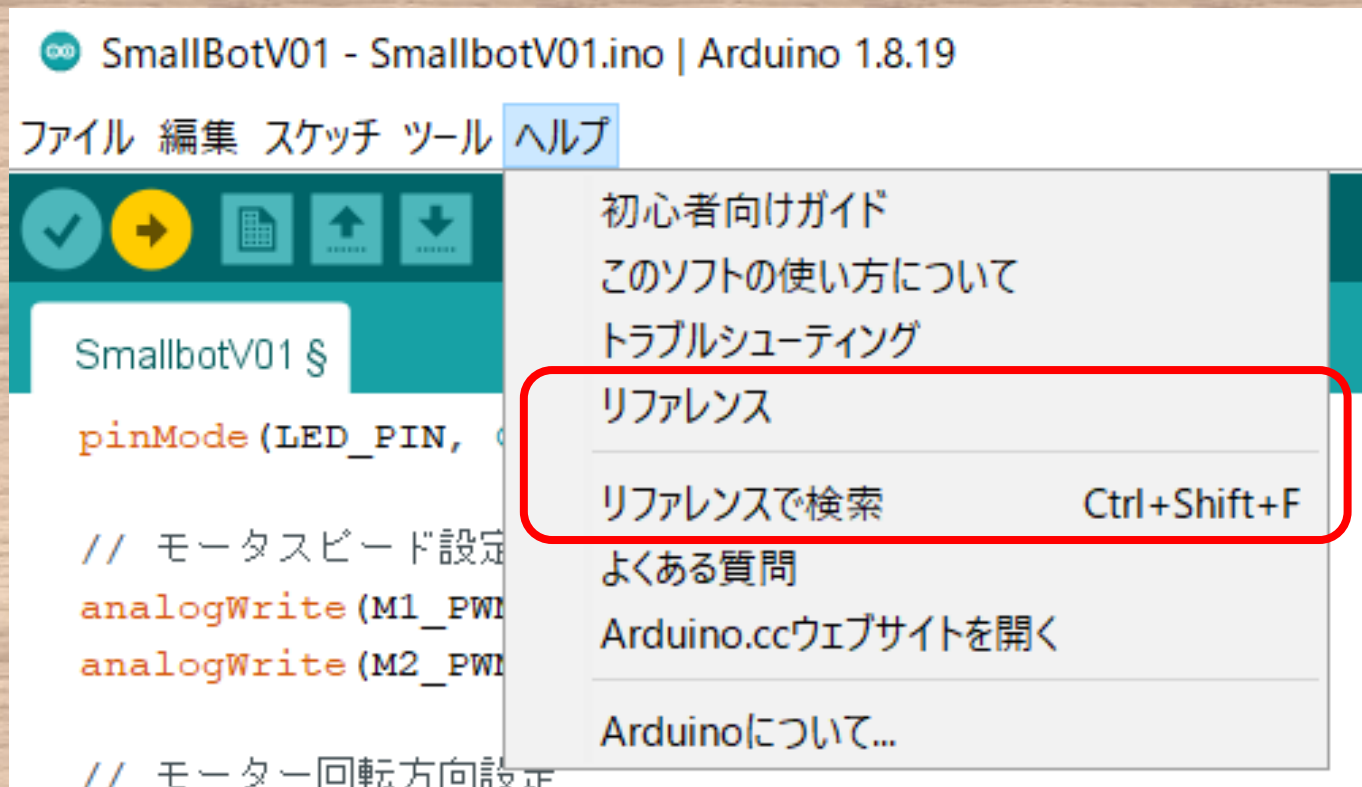


ライブラリ関数の呼び出し

```
void setup() {  
  // モータ関係の出力ピン  
  pinMode(4, OUTPUT);  
  pinMode(9, OUTPUT);  
  pinMode(6, OUTPUT);  
  
  pinMode(7, OUTPUT);  
  pinMode(8, OUTPUT);  
  pinMode(5, OUTPUT);  
  
  // デジタル出力ピン (LED)  
  pinMode(13, OUTPUT);
```

```
// 右モーターの速度 (0~255)  
  analogWrite(6, 80);  
// 左モーターの速度 (0~255)  
  analogWrite(5, 120);  
// 右モーター正転  
  digitalWrite(4, 1);  
  digitalWrite(9, 0);  
// 左モーター正転  
  digitalWrite(7, 1);  
  digitalWrite(8, 0);  
}
```

ライブラリ関数などの説明



<https://www.arduino.cc/reference/en/> が開く

日本語の説明ページ例

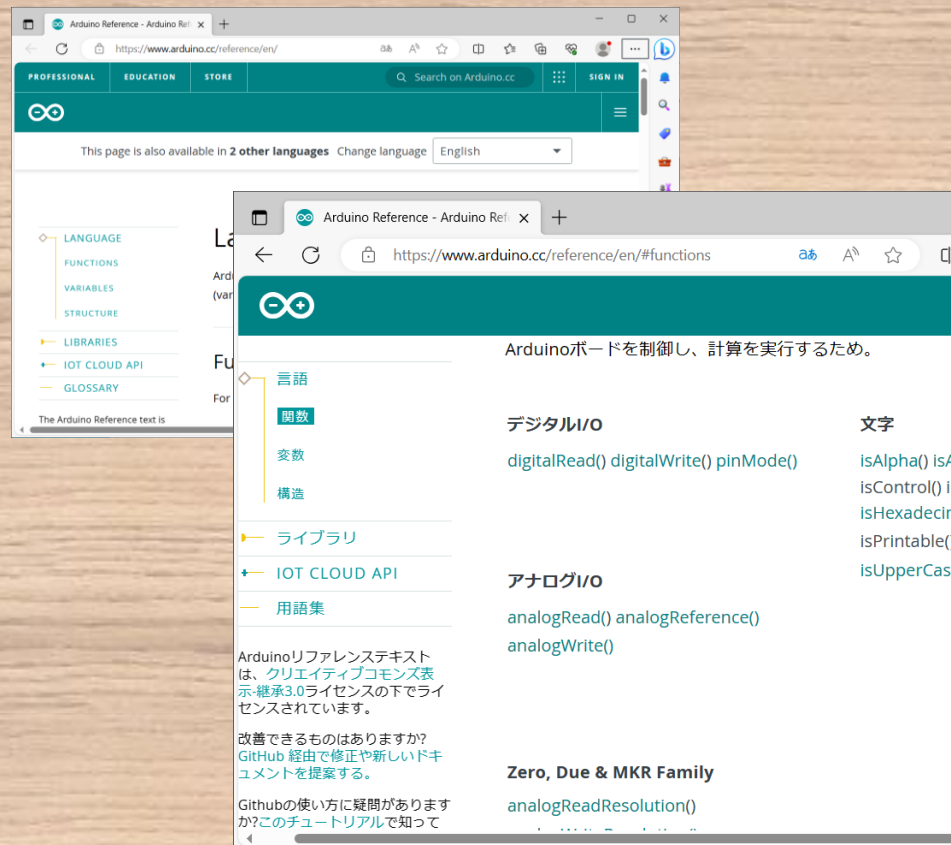
<http://www.musashinodenpa.com/arduino/ref/index.php>
など

[arduino リファレンス]
で検索してみると良い

※：公式なものではない
(間違っている可能性も)



翻訳機能を活用しても良い



デジタルI/O

`digitalRead()` `digitalWrite()` `pinMode()`

翻訳ミスがあるかもしれないので
原文（英語）も確認しておこう

アナログI/O

`analogRead()` `analogReference()`

`analogWrite()`

ライブラリ：まずはここから

`pinMode(pin, mode);`

ピン（端子）のモード設定

`digitalWrite(pin, value);`

デジタルピンの出力設定

`digitalRead(pin);`

デジタルピンの読み込み

`analogWrite(pin, value);`

アナログピンの出力設定

`analogRead(pin);`

アナログピンの読み込み

`delay(ms);`

指定した時間ウェイト
(1/1000秒：ミリ秒単位)

`pulseIn(pin, value);`

パルス幅計測(μ 秒単位)

`delayMicroseconds(us);`

指定した時間ウェイト (μ 秒単位)

pinMode(pin, mode);

ピン（端子）のモード設定

pin : ピン（端子）番号

mode : 動作モード

- **INPUT**
入力
- **OUTPUT**
出力
- **INPUT_PULLUP**
入力(無接続時にはHIGH)

記述例 :

```
pinMode(4, OUTPUT);
```

```
pinMode(11, INPUT);
```

```
pinMode(12, INPUT_PULLUP);
```

pinMode()で設定しないとき
(デフォルト) は
INPUTになっている

digitalWrite(pin, value);

デジタルデータ (HIGH/LOW) 出力

- pin: ピン(端子)番号
- value : 出力値
 - 0または1

pinMode(xxx, OUTPUT);
で出力設定したピン(端子)の状態設定

0: LOW 電圧が低い (0V) 状態

1: HIGH 電圧が高い (3.3V) 状態

例 :

```
// 右モーター正転
digitalWrite(4, 1);
digitalWrite(9, 0);
delay(1000);
// 右モーター逆転
digitalWrite(4, 0);
digitalWrite(9, 1);
// LED点灯
digitalWrite(13, 1);
```

digitalRead(pin); デジタルデータ (HIGH/LOW) 入力

・ pin: ピン (端子) 番号

pinMode(xxx, INPUT);
で出力設定したピン(端子)の状態読み

0: LOW 電圧が低い(0V) 状態
1: HIGH 電圧が高い(3.3V) 状態

例 :

```
int swval;  
swval=digitalRead(12);  
if (swval == 0) {  
    digitalWrite(13, 1);  
} else {  
    digitalWrite(13, 0);  
}
```

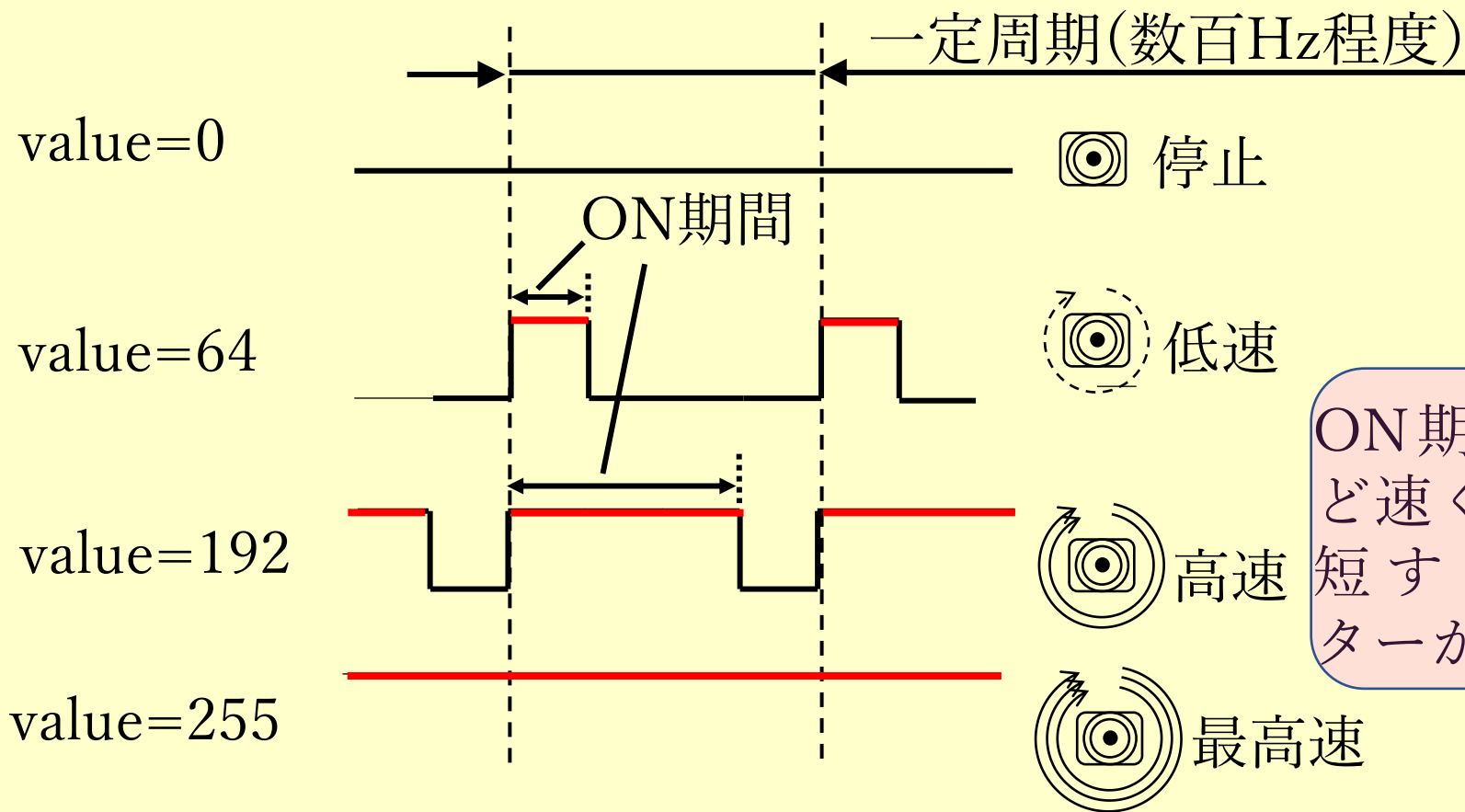
analogWrite(pin, value); PWM(パルス幅変調)出力

- pin: ピン（端子）番号
- value: 出力パルス幅
設定値: 0～255

pinMode(xxx, INPUT);
で出力設定したピン(端子)の
1 (HIGH)パルス幅比率を256
段階で設定

例 :
pinMode(6, OUTPUT);
// 出力幅のHigh比率は120/255
analogWrite(6, 120);

analogWrite()による出力変化と モーターの回転



ON 期間が長いほど速く回転するが、短すぎるとモーターが回らない

analogRead(pin); ピン（端子）の電圧を読み込み

- ・ pin: ピン（端子）番号

pinMode(xxx, INPUT);
で出力設定したピンの電圧
(0～3.3V)を デジタル値で返す
戻り値の範囲は0～1023
端子電圧 = $3.3 \div 1024 \times \text{戻り値}$

例：

```
sliderValue = analogRead(A3);  
if (analogRead(A2) > 200) {  
  . . .  
}
```

スライダの位置や、光センサ（フォトセンサ）による
明るさの判断などに利用している

delay(ms);

1m秒単位でウェイト（動作停止）

- ms: ウェイトする時間
(単位はm秒)

例：

```
digitalWrite(13,1); // LED ON  
delay(500); // 500msウェイト  
digitalWrite(13,0); // LED OFF  
delay(500); // 500msウェイト
```

pulseIn(pin, value, (Timeout));

パルス幅計測(μ 秒単位)

- pin: ピン（端子）番号
- value: 1 (HIGH)または0 (LOW)
- Timeout: タイムアウト
 μ 秒単位で指定(省略可)
- **value**
 - 1: 入力が0から1になった後、0に戻るまでの時間
 - 0: 入力が1から0になった後、1に戻るまでの時間

例:

```
pinMode(11, INPUT);  
duration=pulseIn(11, HIGH);
```

delayMicroseconds(us); 指定した時間ウェイト (μ秒単位)

- us: ウェイトする時間
(μ秒単位)

例 :

// 2μ秒ウェイト

```
delayMicroseconds(2);
```

// 1000μ秒(=1m秒)

// ウェイト

```
delayMicroseconds(1000);
```