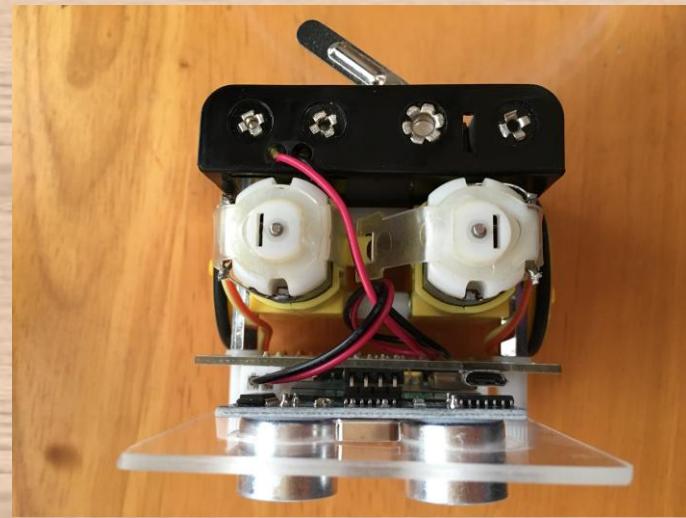
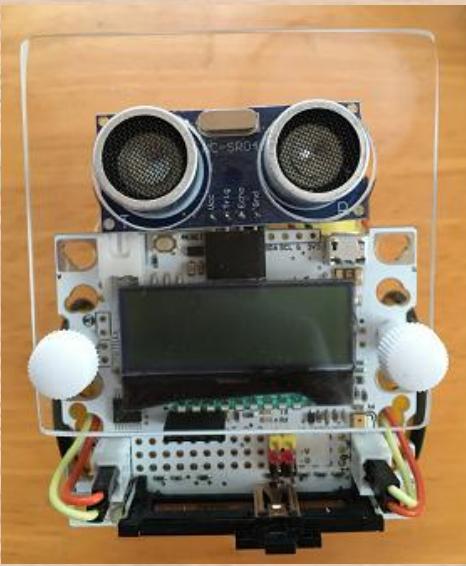


ロボットプログラミング



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

The screenshot shows a GitHub repository named 'smallbot'. The repository is public. On the left, there's a dropdown menu for the 'main' branch, a 'Branches' button, and a 'Tags' button. Below these are three file entries:

- A file named 'Manual.pdf' with a blue icon, circled in red.
- A file named 'SmallBot_P01.pdf' with a blue icon, circled in red.
- A file named 'SmallbotV00.ino' with a blue icon.

To the right of each circled file, there is Japanese text describing its purpose:

- 'Manual.pdf' is described as 'ハードウェアマニュアル' (Hardware Manual).
- 'SmallBot_P01.pdf' is described as '第一回テキスト' (First Text).
- 'SmallbotV00.ino' is described as 'サンプルプログラム' (Sample Program).

Red lines connect the circled file names to their respective descriptions on the right.

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

キャラクタLCD

右モーター

超音波距離センサ

押しボタン
スイッチ

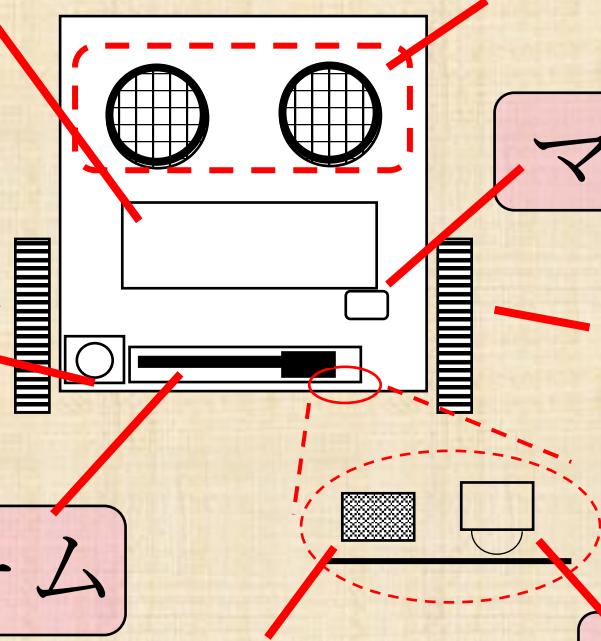
マイク

スライドボリューム

左モーター

白色LED

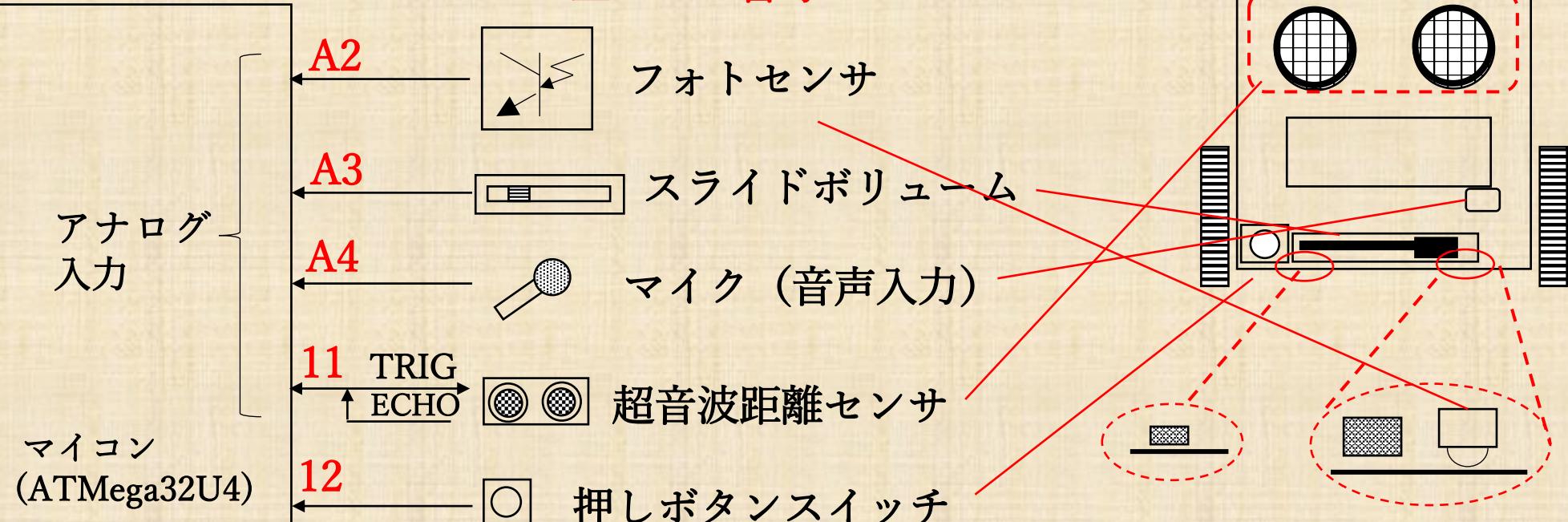
フォトセンサー



入力信号の接続（参考）

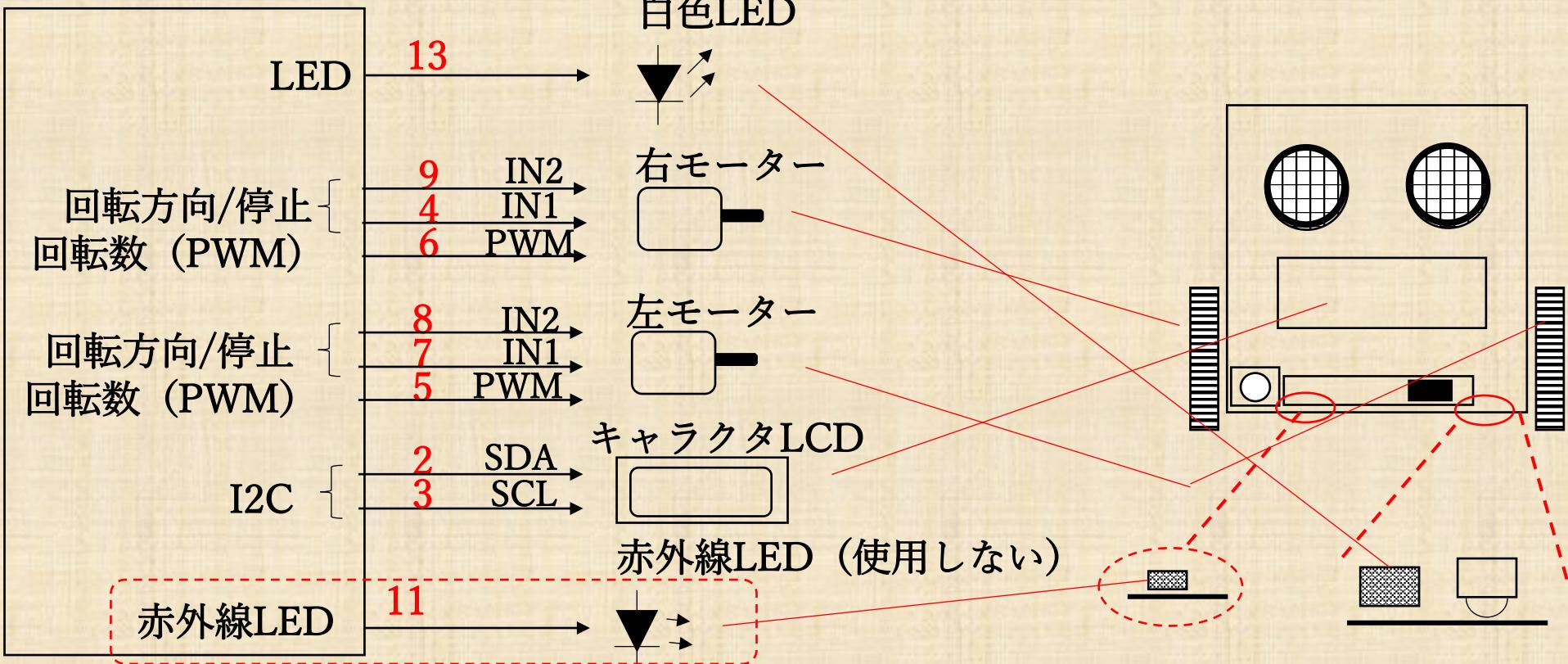
RDC-104 type II

Arduino-IDE上のピン番号

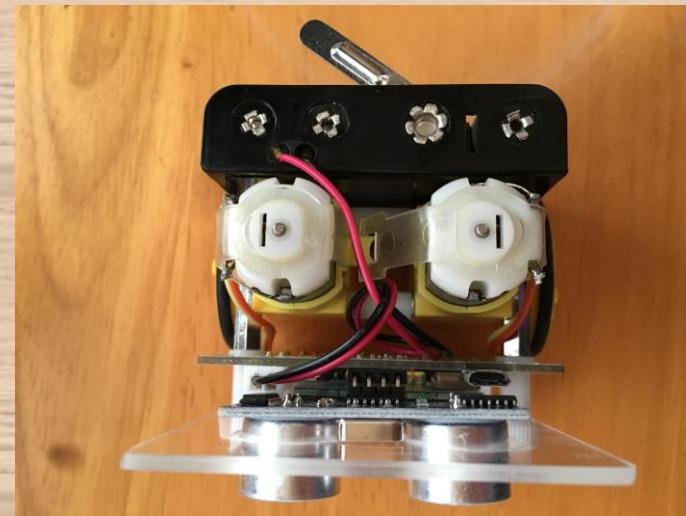


出力信号の接続（参考）

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



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

The screenshot shows the 'STEM Du CH552' software interface. On the left, there is a sidebar with the following menu items:

- 日本型STEM教育
- ロボットと未来研究会 >
- 放課後デザイン >
- STEM教材開発 <-- Selected item, highlighted in blue

The main content area has a header: 'STEM教育研究センター 埼玉大学教育学部野村研究室'. Below the header is a logo featuring a green dome with the word 'STEM' in white, surrounded by icons for Engineering, Mathematics, Technology, Science, and Art. The main content section is titled 'STEM Duコントローラのプログラミング環境' (Programming Environment for STEM Du Controller). It contains a list of updates:

- 2022/5/7 macos monterey (12.x.x) で STEM Du ESP32をインストールできるようにボードマネジャーからのインストールファイルを更新しました。
- 2022/1/2 Arduino 1.8.19での動作確認をしました。
- 2022/1/2 STEM Duライブラリを変更し、MPU6050ライブラリに関連するコードを削除しました。102, 103でMPU6050を使う場合は、別途ライブラリをインストールする必要があります。
- 2021/6/10 104までのコントローラについてボードマネジャーからのインストールに対応しました。
- 2021/6/10 ESP32についてボードマネジャーからのインストールに対応しました。
- 2021/6/9 Apple M1マシンのためのファイルを追加しました。
- 2021/6/9 最新のArduino 1.8.15での動作確認をしました。

Below the update list, there is a step-by-step guide: 'ステップ1: デバイスドライバのインストール' (Step 1: Install device driver).

「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.

A screenshot of the Arduino Software (IDE) download options page. The page has a teal header with the text "DOWNLOAD OPTIONS". Below this, there are two main download links: "Windows Win 7 and newer" and "Windows ZIP file". A red circle highlights the first link. To the right of these links is a "Get" button with a Windows logo. Below these links are links for Linux (32-bit and 64-bit), ARM 32-bit, and ARM 64-bit. Further down are links for Mac OS X (10.10 or newer), Release Notes, and Checksums (sha512).

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

[Windows app](#) Win 8.1 and later Get

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

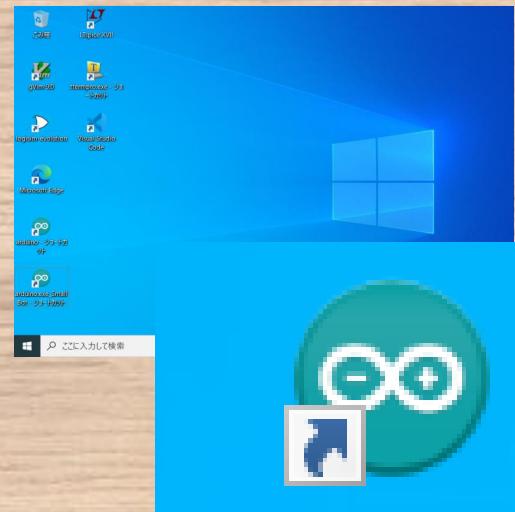
[Release Notes](#)
[Checksums \(sha512\)](#)

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

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

Arduino IDEの起動



A screenshot of the Arduino IDE window titled "sketch_oct19a | Arduino 1.8.19". The menu bar includes ファイル, 編集, スケッチ, ツール, ヘルプ. The code editor shows the following code:

```
void setup() {  
  // put your setup code here, to run  
  // once the sketch starts  
}  
  
void loop() {  
  // put your main code here, to run  
  // repeatedly  
}
```

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「追加のボードマネジャーの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】

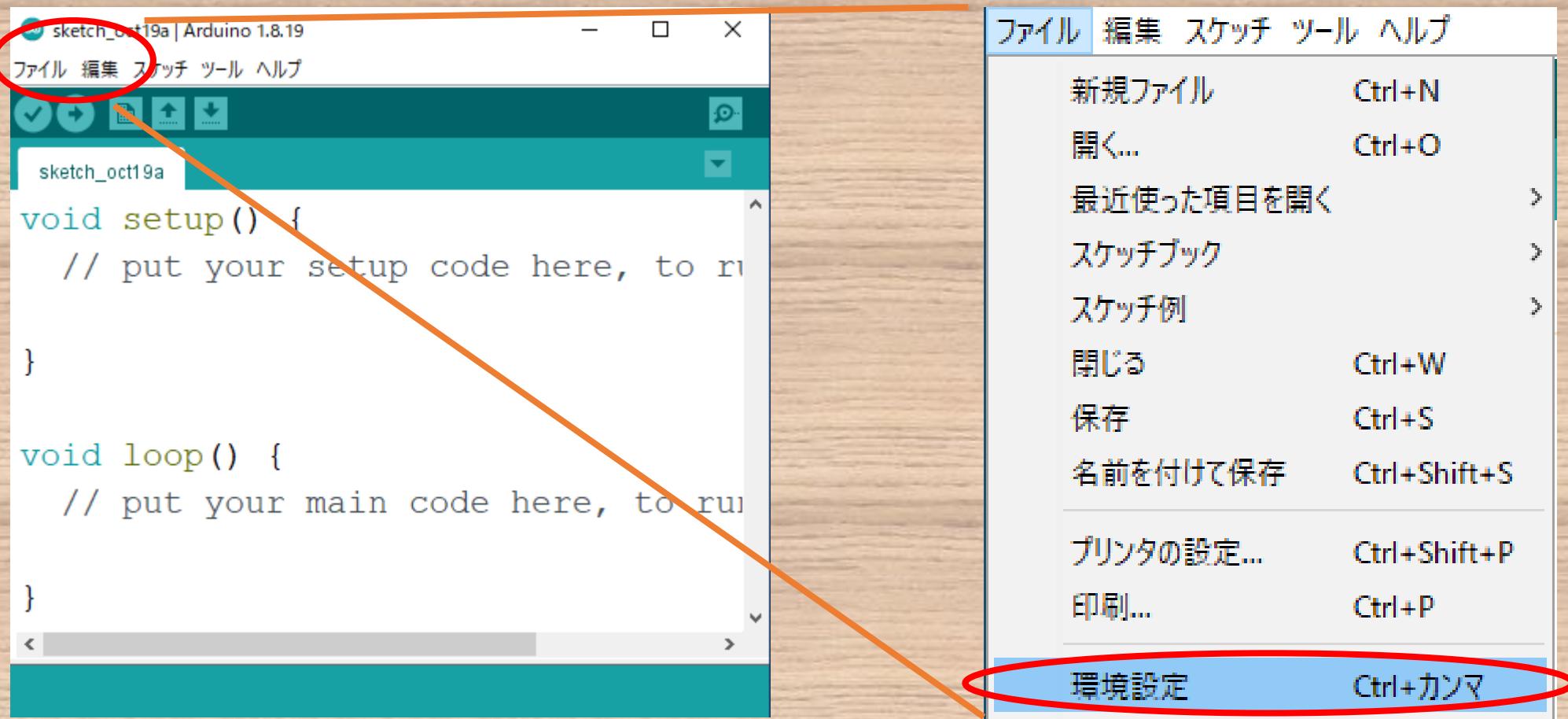
https://github.com/tairo/STEMDu_ArduinoIDE/releases/download/v1.0/package_stemdu-esp32_index.json

【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

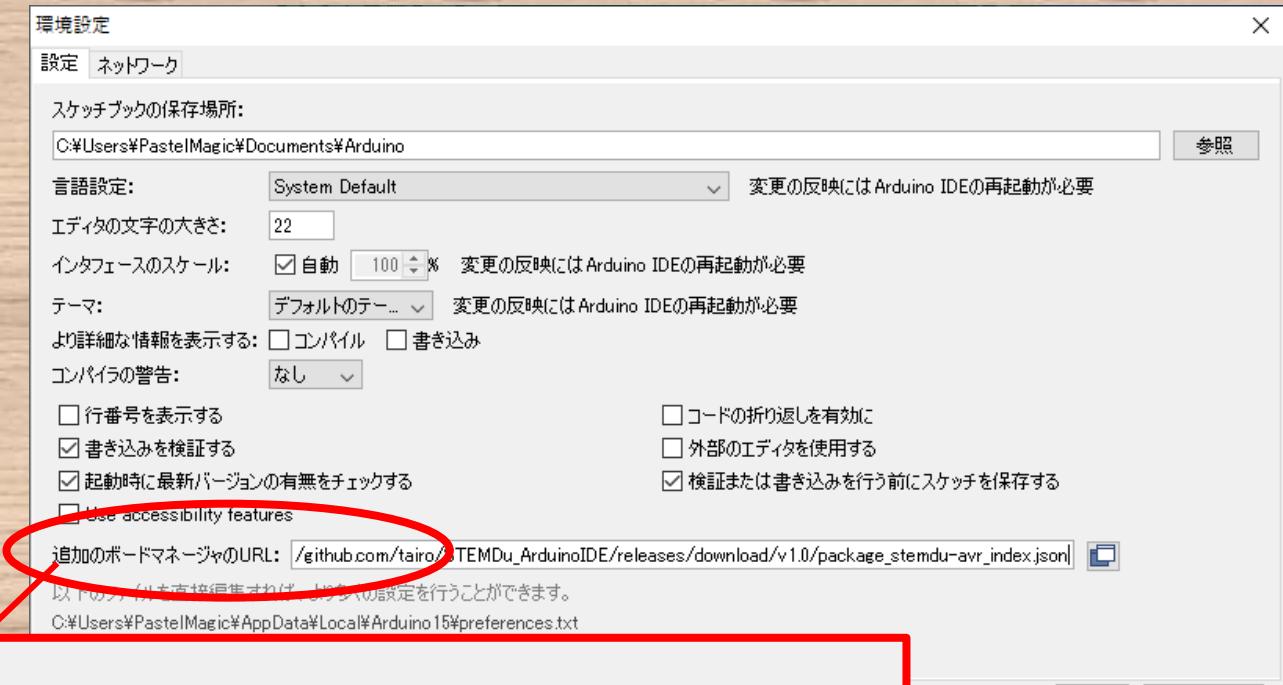
https://github.com/tairo/STEMDu_ArduinoIDE/releases/download/v1.0/package_stemdu-avr_index.json

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.c

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

シリアルプロット

Ctrl+Shift+L

WiFi101 / WiFiNINA Firmware Updater

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

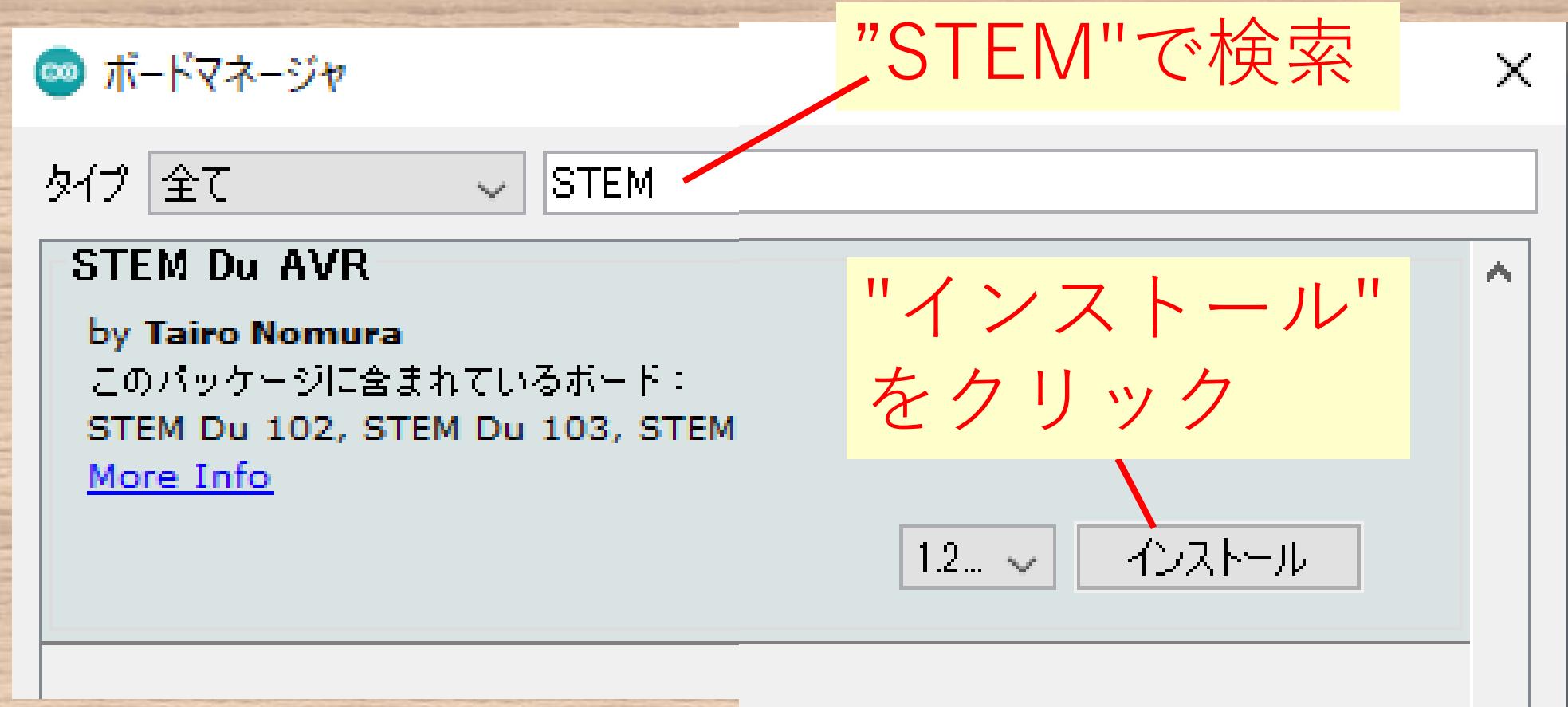
ボードマネージャ...

シリアルポート

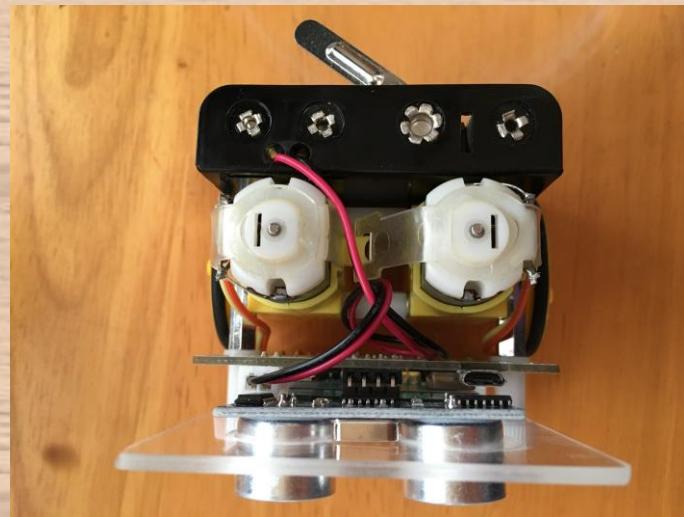
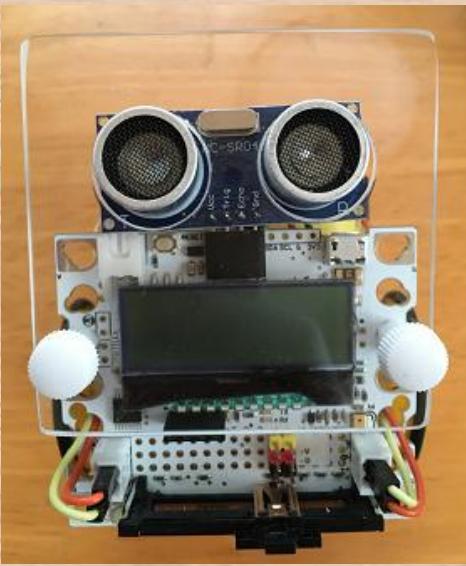
Arduino AVR Boards

ツール => ボード => ボードマネージャ

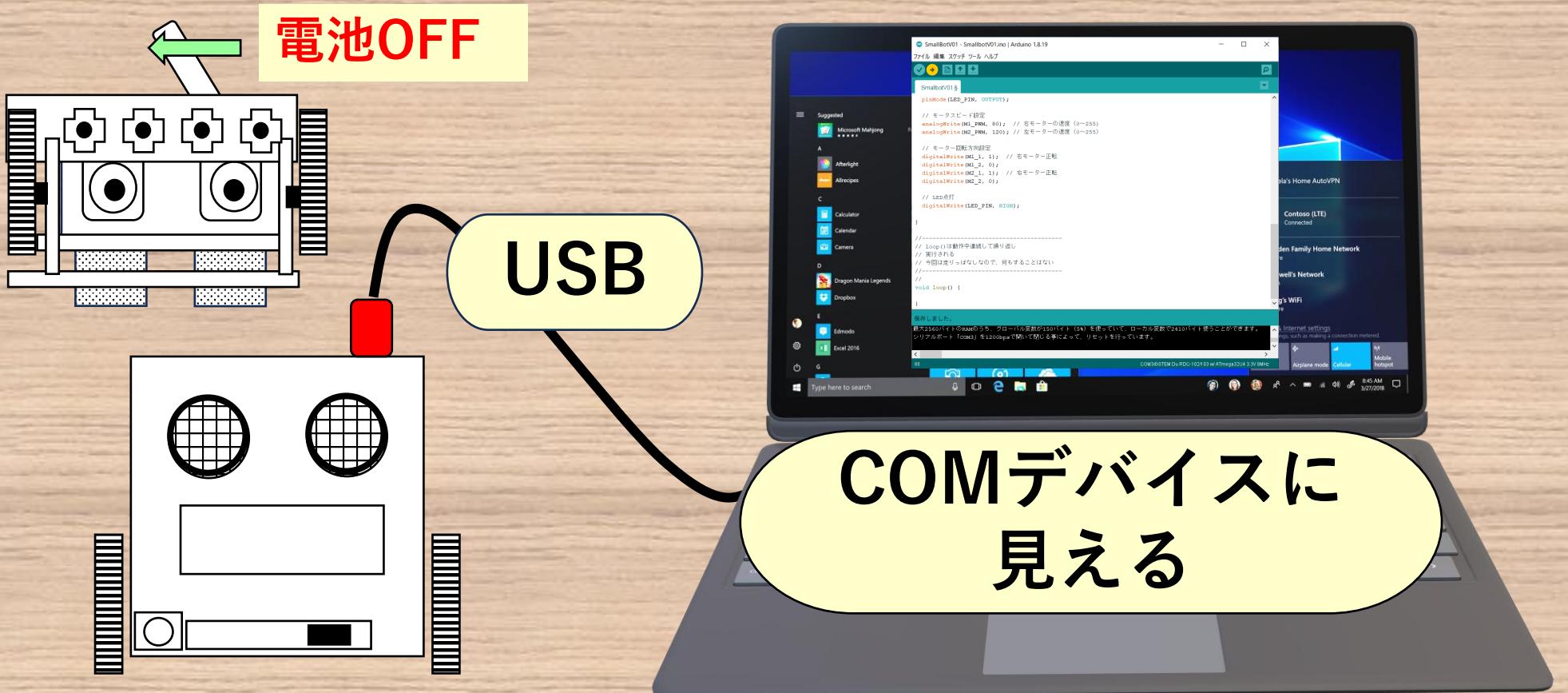
STEM Du AVRをインストール



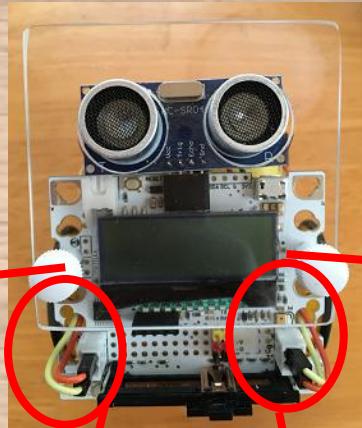
書き込んで動かすまでの手順



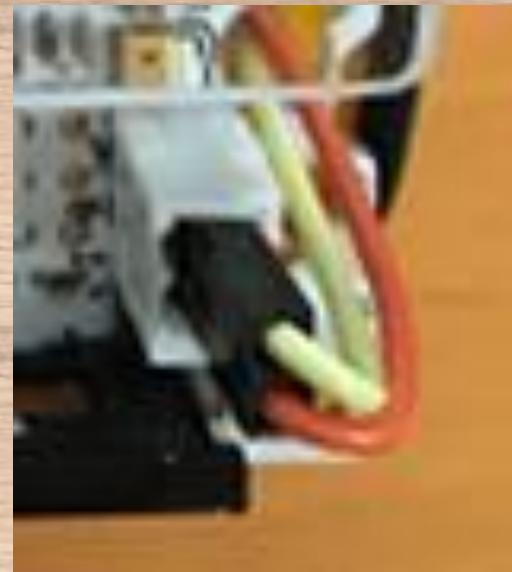
PCと接続



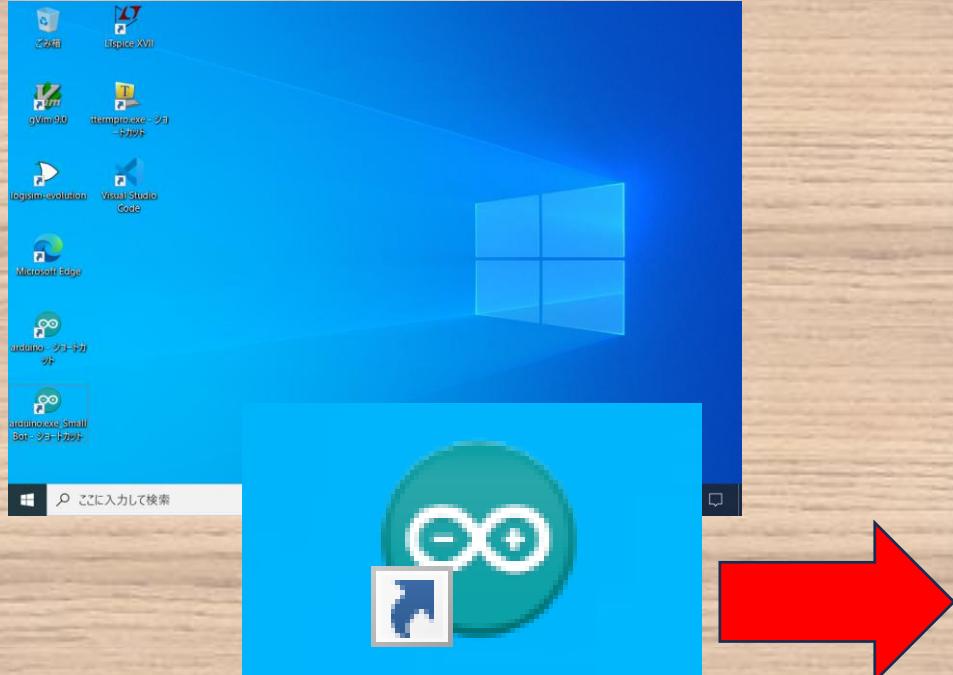
USB電源で動いてしまう時



赤か黄色のどちらかを抜く



Arduino IDEの起動



The screenshot shows the Arduino IDE interface. The title bar reads 'SmallBotV01 - SmallbotV01.ino | Arduino 1.8.19'. The menu bar includes 'ファイル' (File), '編集' (Edit), 'スケッチ' (Sketch), 'ツール' (Tools), and 'ヘルプ' (Help). The main area displays the following C++ code:

```
SmallBotV01 $
```

```
pinMode(LED_PIN, OUTPUT);

// モータスピード設定
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); // 左モーター正転
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() {

}


```

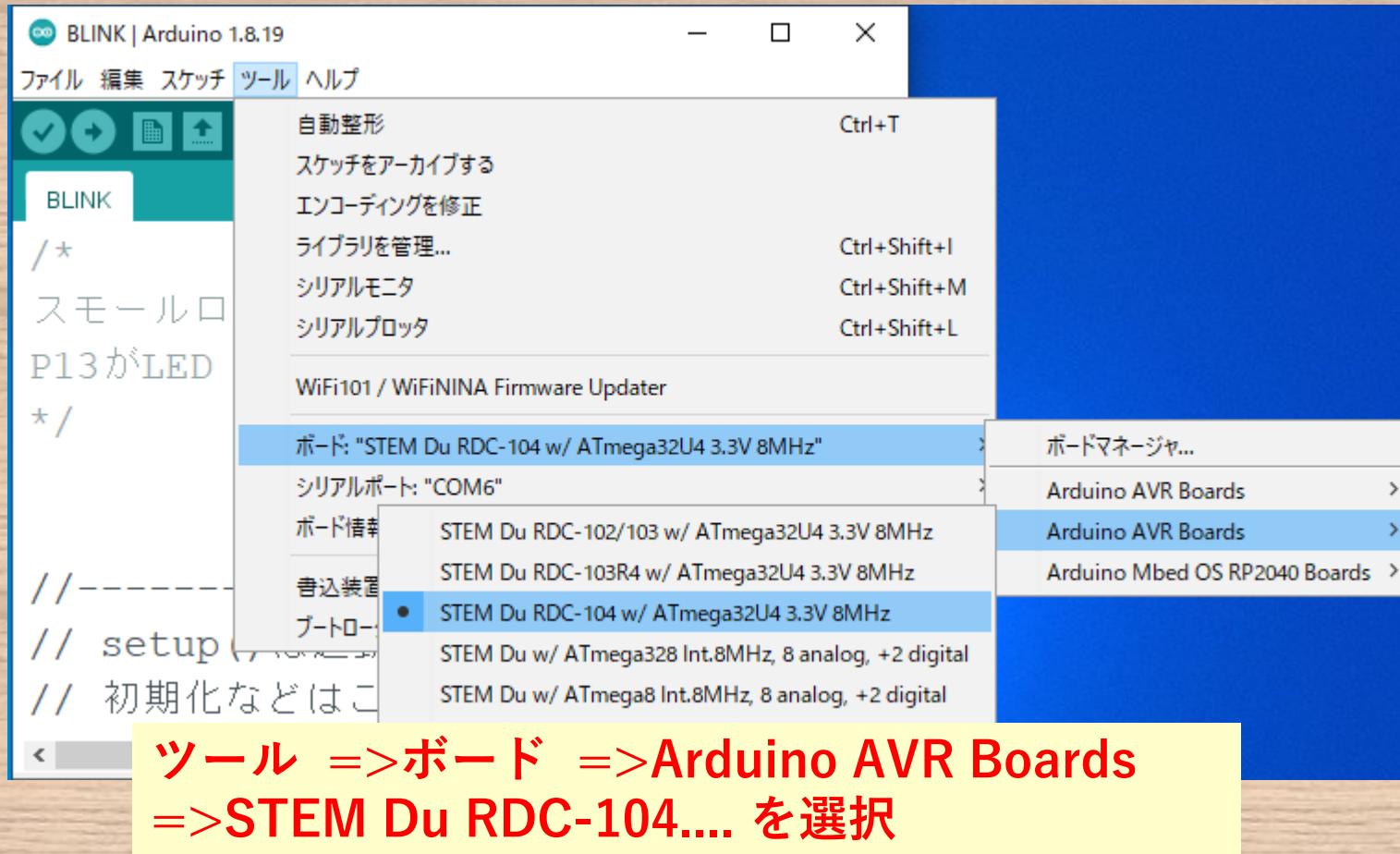
保存しました。

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。

COM3のSTEM Du RDC-102103 w/ ATmega32U4 3.3V 8MHz

ターゲットの選択

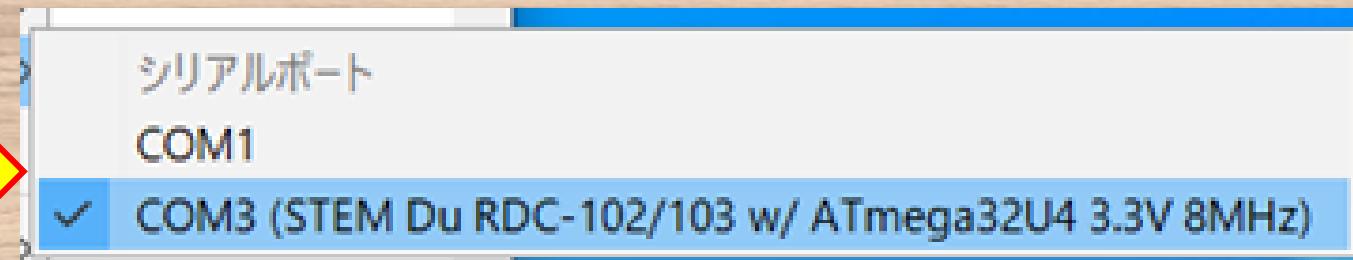


COM（シリアル）ポートの選択



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/.....



プログラムの記述 (setup()やloop()を消去しないこと)

The screenshot shows the Arduino IDE interface with a sketch titled "sketch_oct10a". The code editor contains the following:

```
sketch_oct10a | Arduino 1.8.19
...
ファイル 編集 スケッチ ツール ヘルプ
...
sketch_oct10a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

A red arrow points from the "C/C++言語で記述" callout box to the code editor area.

C/C++言語で記述

The diagram consists of two side-by-side code snippets with explanatory text boxes.

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

A red arrow points from the "起動時最初に" callout box to the first code snippet. Another red arrow points from the "繰り返し実行される" callout box to the second code snippet.

とりあえずLED点滅

```
BLINK | Arduino 1.8.19
ファイル ファーム 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る  
// 初期化などはここでを行う  
//-----  
void setup() {  
    // LEDのピン  
    pinMode(13, OUTPUT);  
}  
  
//-----  
// loop()は繰り返し実行される  
//-----  
void loop() {  
    digitalWrite(13,1); // ON  
    delay(500); // 500msウェイト  
    digitalWrite(13,0); // OFF  
    delay(500); // 500msウェイト  
}
```

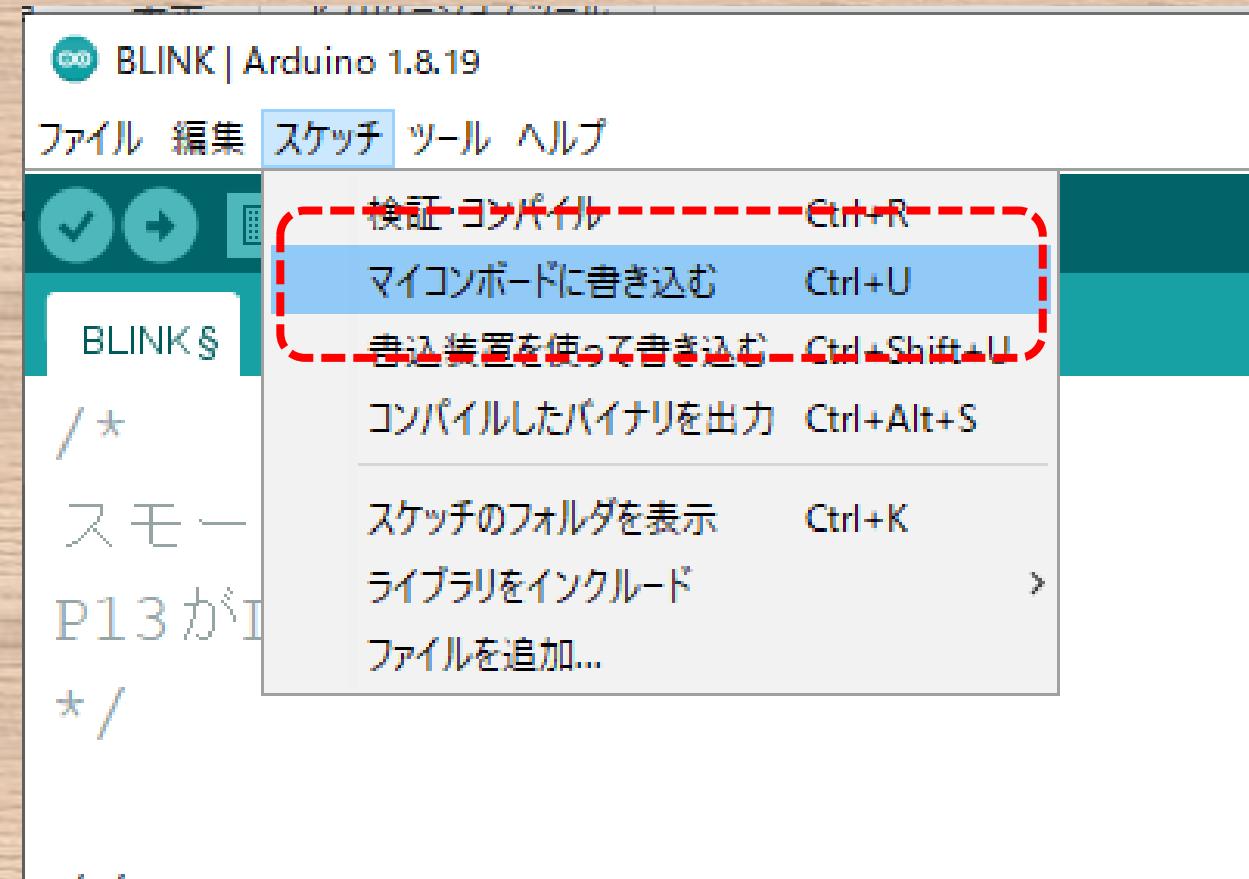
コンパイルが完了しました。
最大2093056バイトのフラッシュメモリのうち、スケッチが52308バイト（2%）を
最大262144バイトのRAMのうち、グローバル変数が10224バイト（3%）を使っています。

"//"
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 フレーム 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
< >
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
< >
1 COM3のSTEM Du RDC-104 w/ ATmega32U4 3.3V 8MHz
```



ボードへの書き込みが完了しました。

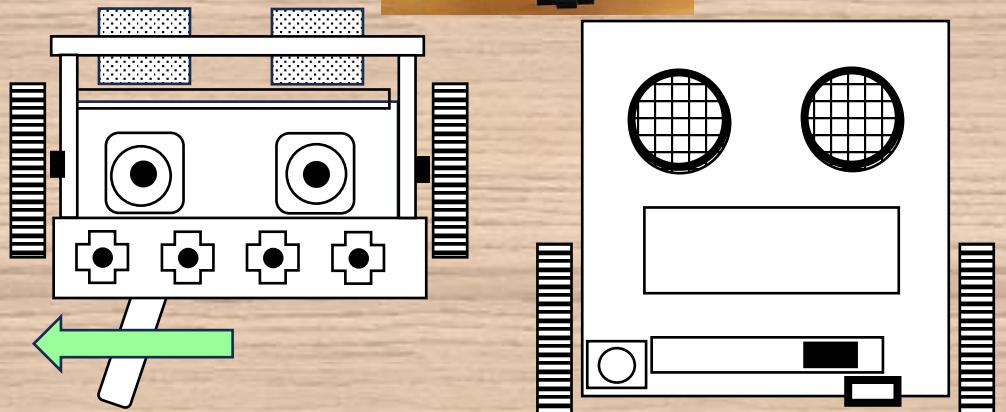
最大28672バイトのフラッシュメモリのうち、
最大2560バイトのRAMのうち、グローバル変数

< >

1 COM3のSTEM Du RDC-104 w/ ATmega32U4 3.3V 8MHz

書き込み完了

電池から電源を供給



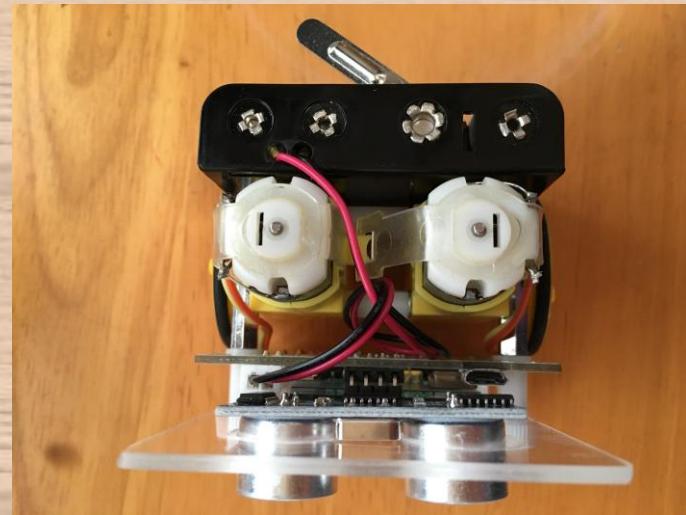
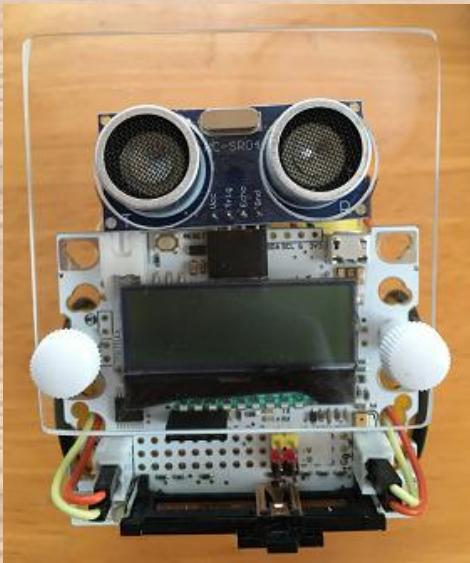
電池ON

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);
```

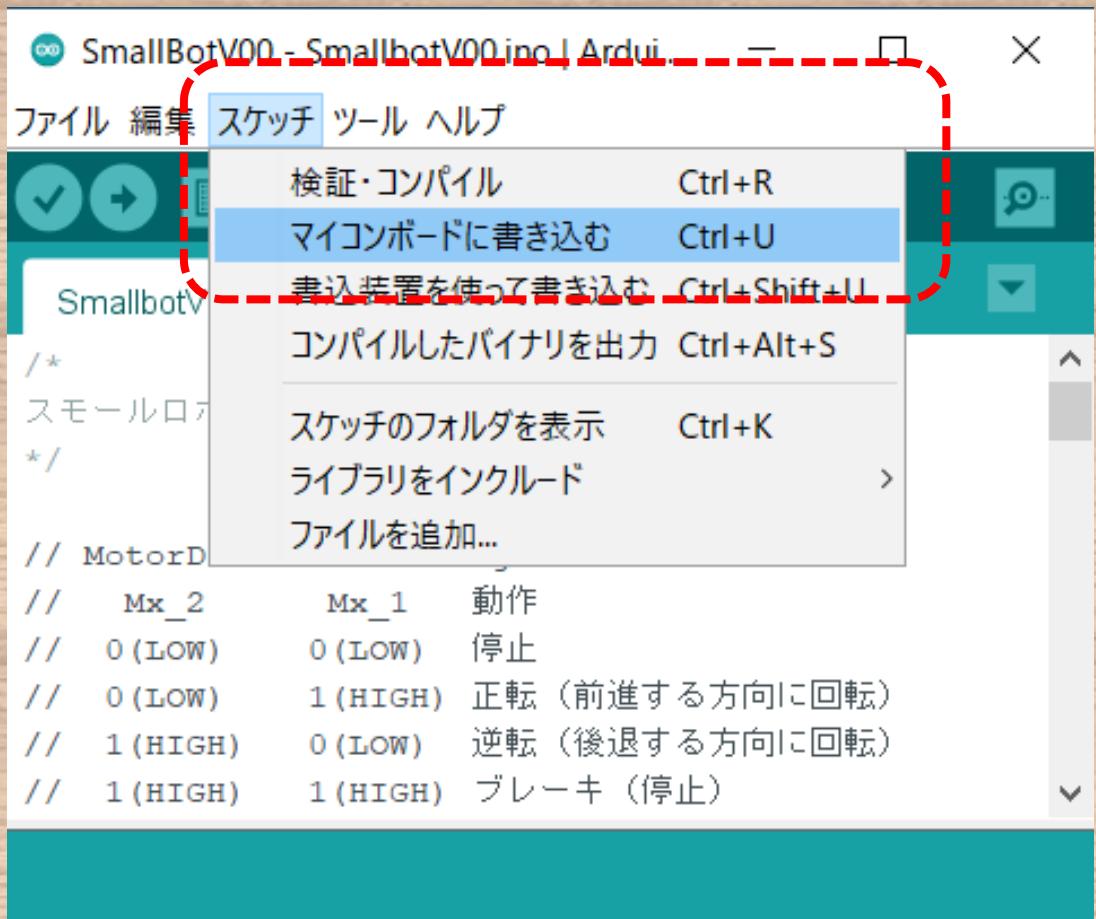
```
// 右モーターの速度 (0~255)  
analogWrite(6, 80);  
// 左モーターの速度 (0~255)  
analogWrite(5, 120);  
// 右モーター正転  
digitalWrite(4, 1);  
digitalWrite(9, 0);  
// 左モーター正転  
digitalWrite(7, 1);  
digitalWrite(8, 0);  
}
```

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



The screenshot shows the Arduino IDE interface. The title bar reads "SmallBotV00 - SmallbotV00.ino | Arduino". The menu bar has "ファイル" (File), "編集" (Edit), "スケッチ" (Sketch), "ツール" (Tools), and "ヘルプ" (Help). A red dashed box highlights the "スケッチ" (Sketch) menu item, which is currently selected and highlighted in blue. The submenu under "スケッチ" includes "検証・コンパイル" (Verify/Compile) with Ctrl+R, "マイコンボードに書き込む" (Upload to Board) with Ctrl+U (which is also highlighted with a red dashed box), "書き込み装置を使って書き込む" (Upload Using Programmers) with Ctrl+Shift+U, "コンパイルしたバイナリを出力" (Output Compiled Binary) with Ctrl+Alt+S, "スケッチのフォルダを表示" (Show Sketch Folder) with Ctrl+K, "ライブラリをインクルード" (Include Library), and "ファイルを追加..." (Add File). The main code editor window displays the following C++ code:

```
/*  
 *  
 * スマートロボット  
 */  
  
// MotorD  
  
// 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)
```

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

The screenshot shows the Arduino IDE interface. The title bar says "SmallBotV00 - SmallbotV00.ino | Arduino...". The menu bar includes "ファイル", "編集", "スケッチ", "ツール", and "ヘルプ". A toolbar with icons for save, upload, and other functions is visible. The main area shows the code for "SmallbotV00". The code defines pin assignments for a motor driver and includes comments about forward and reverse directions. Below the code, it says "スケッチをコンパイルしています..." (Compiling sketch...) with a progress bar.

```
/*
スモールロボット最初の一歩
*/
// 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-1.8.19\cores\arduino\WProgram.h"
"C:\SmallBot\arduino-1.8.19-windows\arduino-1.8.19\cores\arduino\Arduino.h"
< >
1 COM3のSTEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz
```

スケッチをコンパイルしています...

"C:\SmallBot\arduino-1.8.19-windows\arduino-1.8.19\cores\arduino\WProgram.h"
"C:\SmallBot\arduino-1.8.19-windows\arduino-1.8.19\cores\arduino\Arduino.h"



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

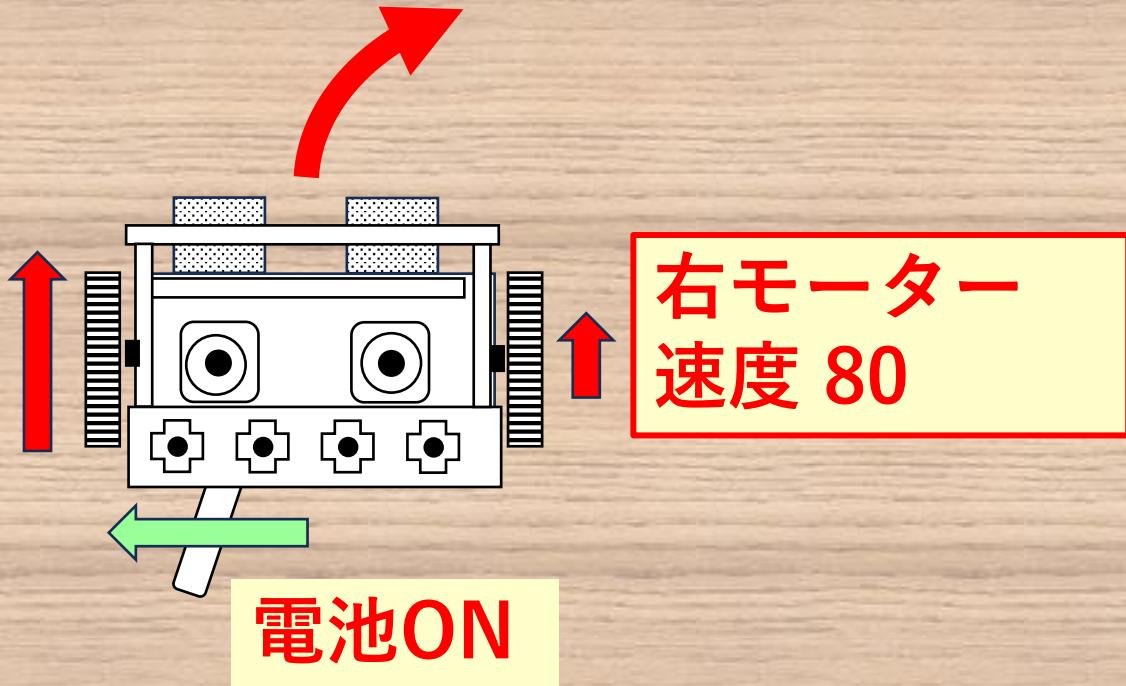
書き完了

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

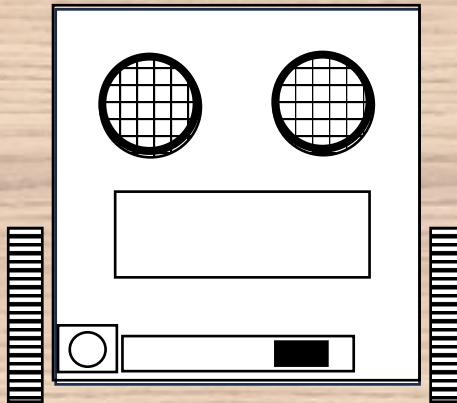
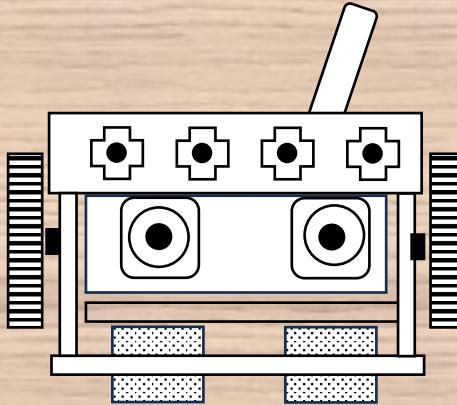
時計回りに回転する



左モーター
速度 120



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)

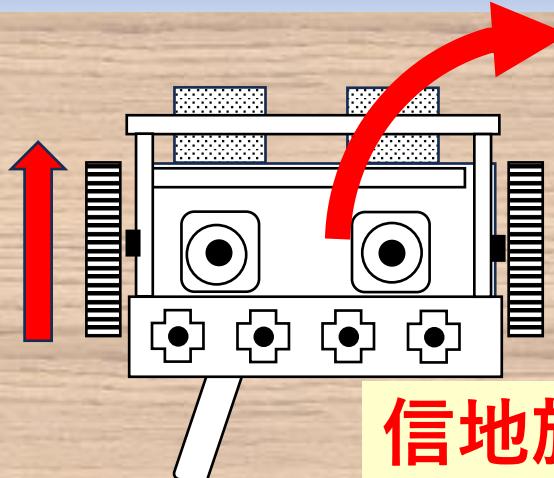
```
void setup() {  
    // モータ関係の出力ピン  
    pinMode(4, OUTPUT);  
    pinMode(9, OUTPUT);  
    pinMode(6, OUTPUT);  
    pinMode(7, OUTPUT);  
    pinMode(8, OUTPUT);  
    pinMode(5, OUTPUT);  
    // デジタル出力ピン (LED)  
    pinMode(13, OUTPUT);
```

```
// 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

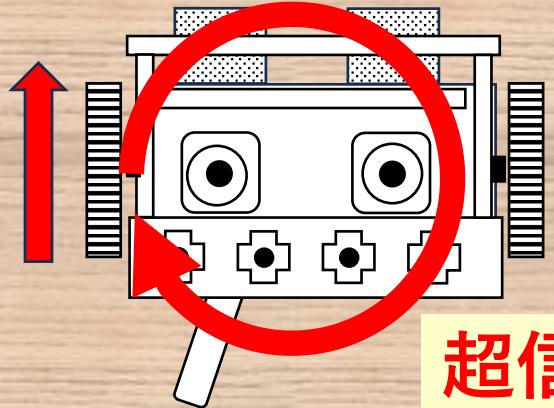
左モーター
速度 120



右モーター
速度 0

信地旋回

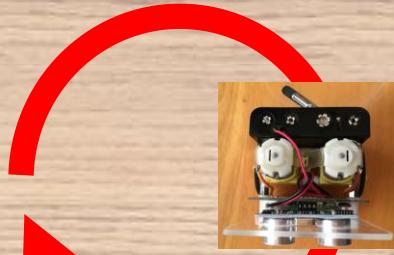
左モーター
速度 120



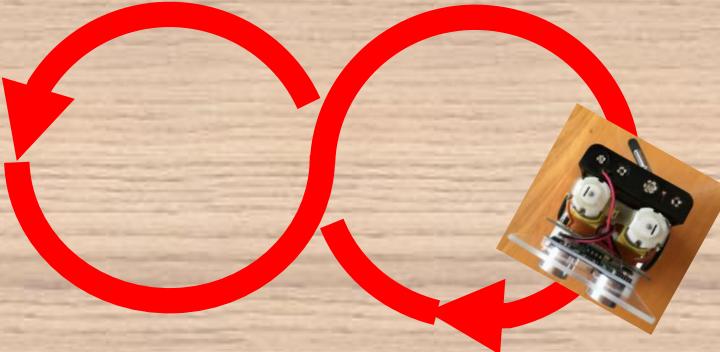
右モーター
速度 120

超信地旋回

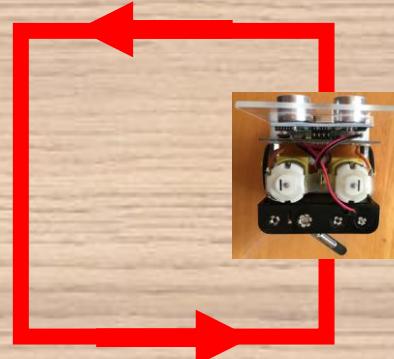
いろいろな動きをさせてみよう



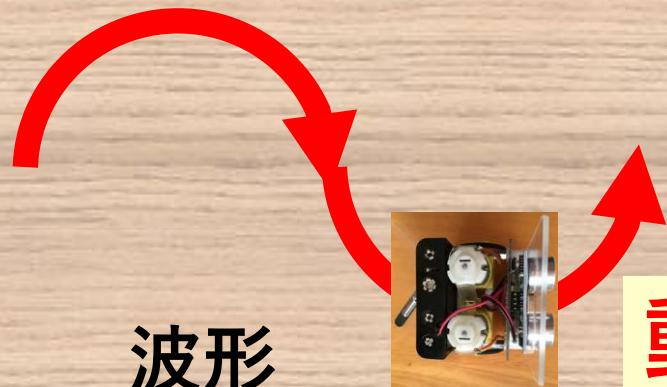
円



8の字



正方形



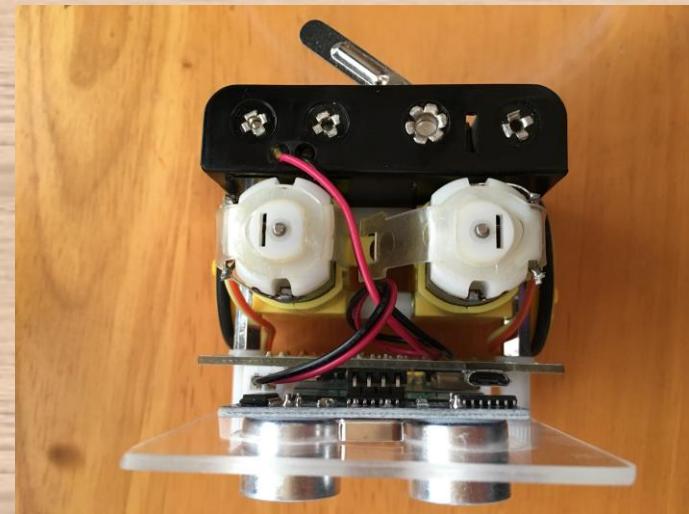
波形

動作パターン例



螺旋状

プログラム（ライブラリ）について

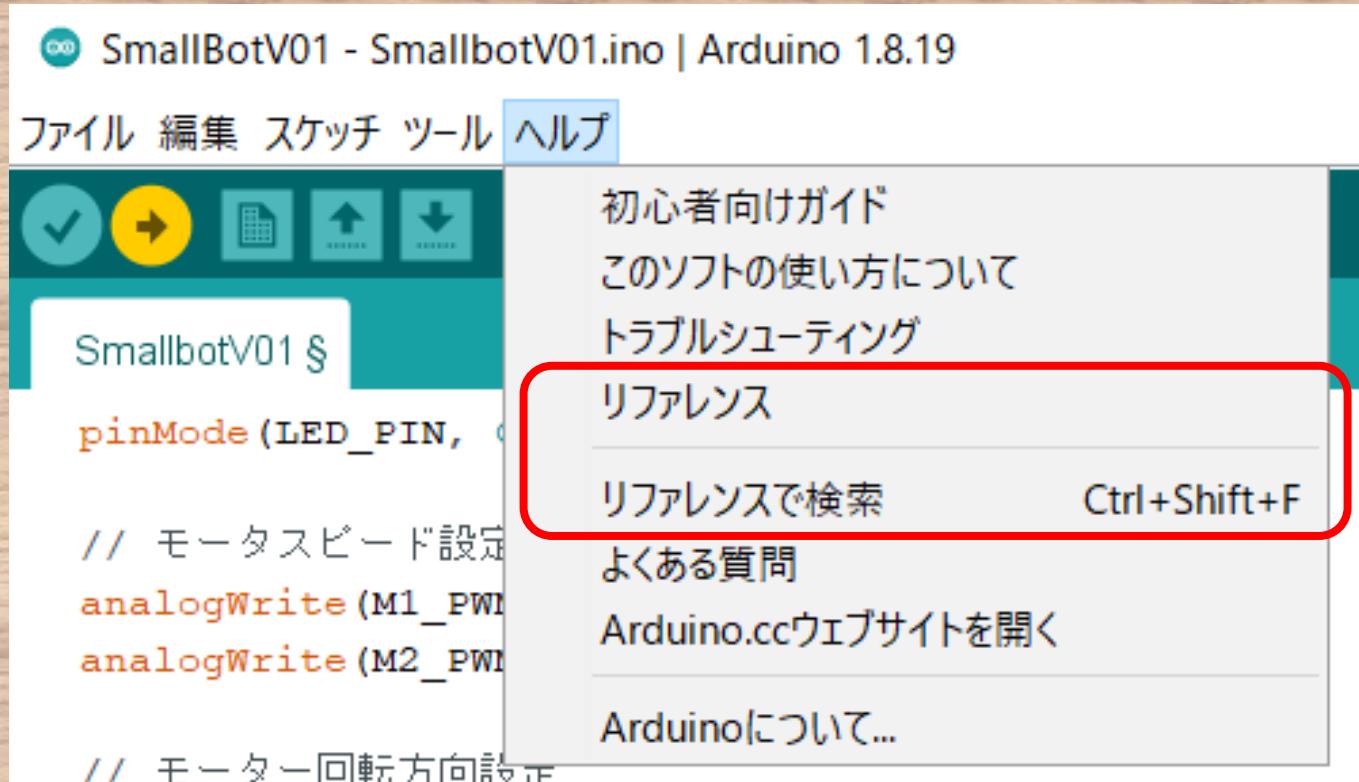


ライブラリ関数の呼び出し

```
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](http://www.musashinodenpa.com/arduino/ref/index.php)
など

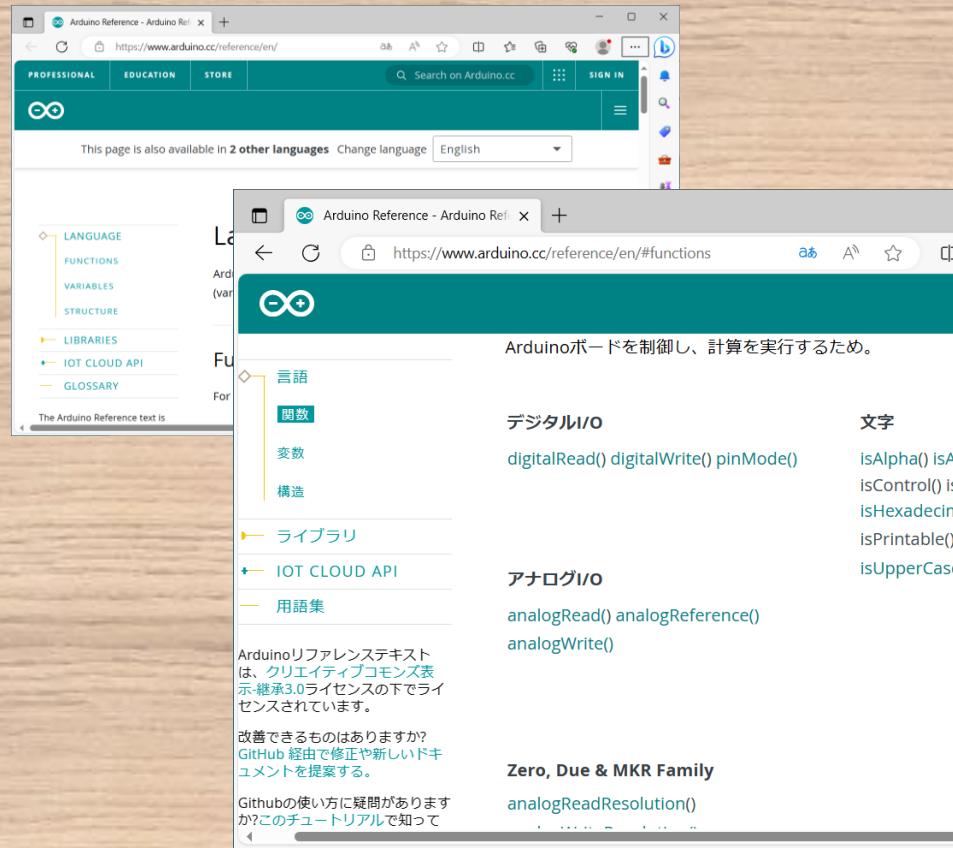
[arduino リファレンス]
で検索してみると良い

※：公式なものではない
(間違っている可能性も)

The screenshot shows a web browser displaying the Arduino Japanese Reference page at www.musashinodenpa.com/arduino/ref/index.php. The page title is "Arduino 日本語リファレンス". The content is organized into two main columns:

- Arduino言語**: Contains links to [setup\(\)](#) and [loop\(\)](#).
- 制御文**: Contains links to [if](#), [if else](#), [switch case](#), [for](#), [while](#), [do while](#), [break](#), and [continue](#).
- 標準ライブラリ**: Contains text about the standard library and user-submitted libraries, along with a link to [ライブラリの使い方](#).
- EEPROM**: Contains text about EEPROM memory on Arduino boards.

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

pulseIn(pin, value);

パルス幅計測(μ秒単位)

analogWrite(pin, value);

アナログピンの出力設定

analogRead(pin);

アナログルピンの読み込み

delay(ms);

指定した時間ウェイト
(1/1000秒 : ミリ秒単位)

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)の状態読み込み

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

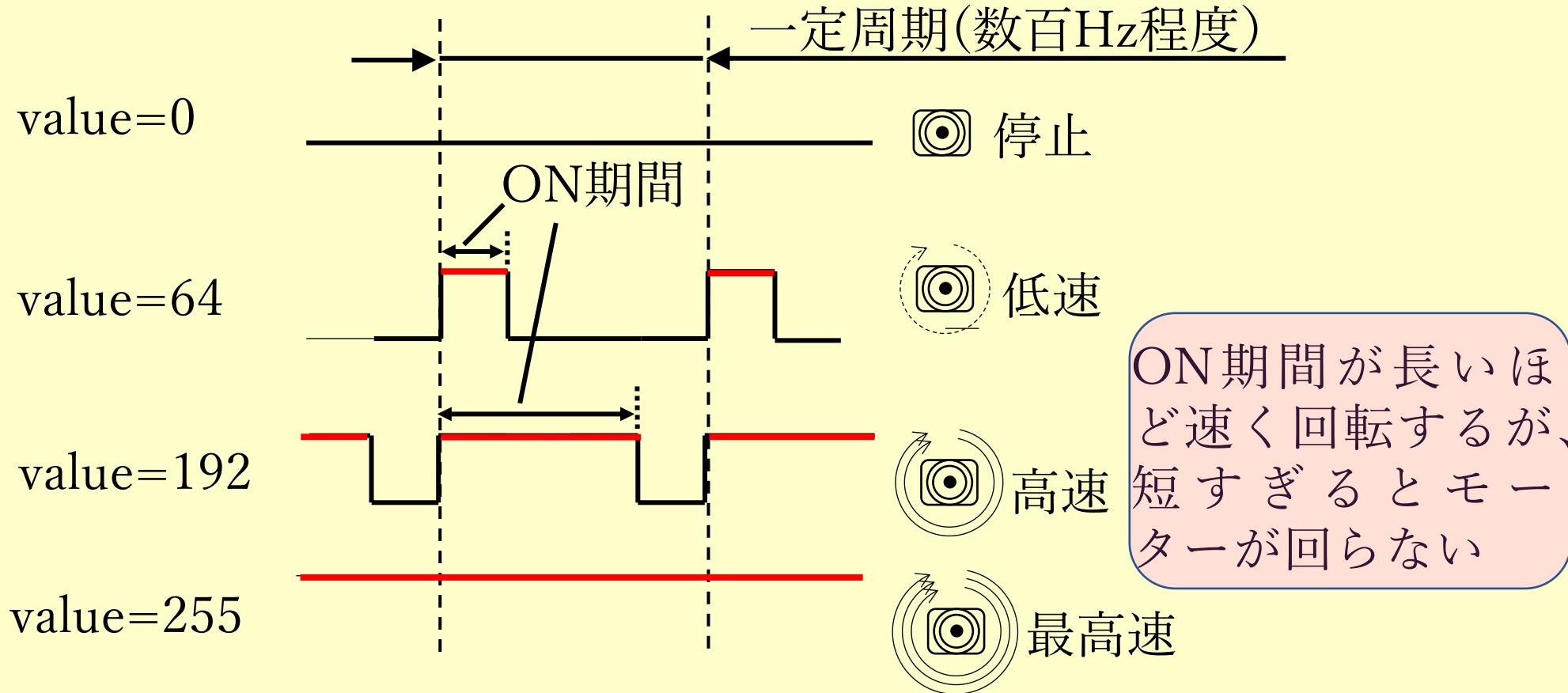
- pin: ピン (端子) 番号
- value: 出力パルス幅
設定値: 0~255

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



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);