

# ロボットプログラミング#2



## SmallBotを動かそう

- ・前回の復習
- ・マクロ(`#define`)の利用
- ・ライントレースさせてみよう

# 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. Below it are links for 'Branches' and 'Tags'. The main area lists several files:

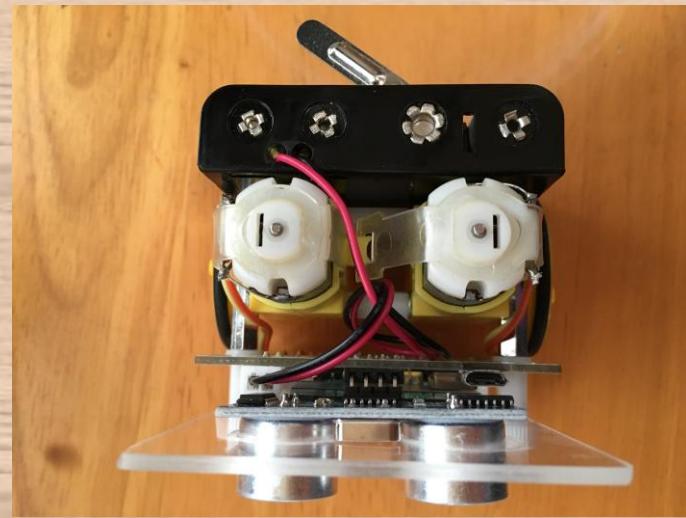
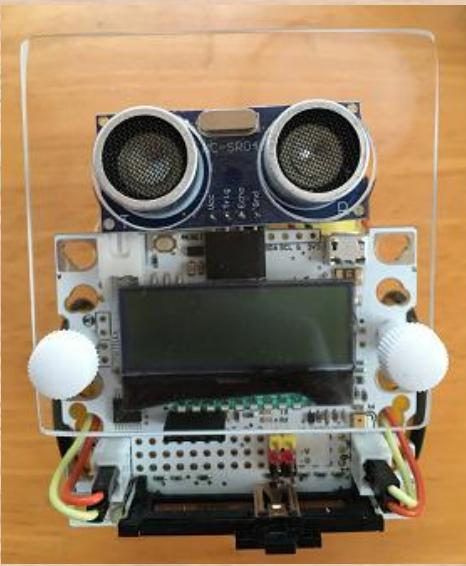
- neecrobot 第一回テキスト追加 ...
- Manual.pdf First Commit
- SmallBot\_P01.pdf 第一回テキスト追加
- SmallbotV00.ino プログラムV00(最初のプログラム)



置いてあるファイル  
はクリックして  
ダウンロード可

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

# SmallBotについて



# 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

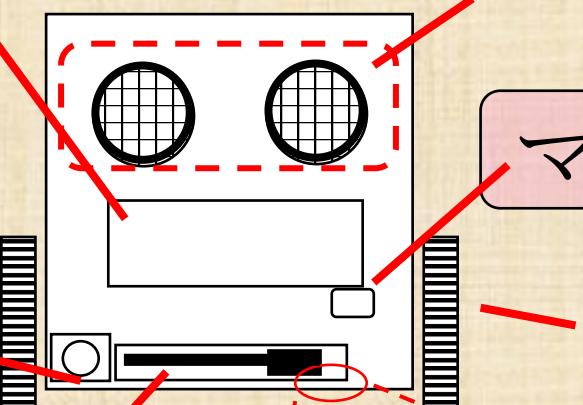
マイク

スライドボリューム

左モーター

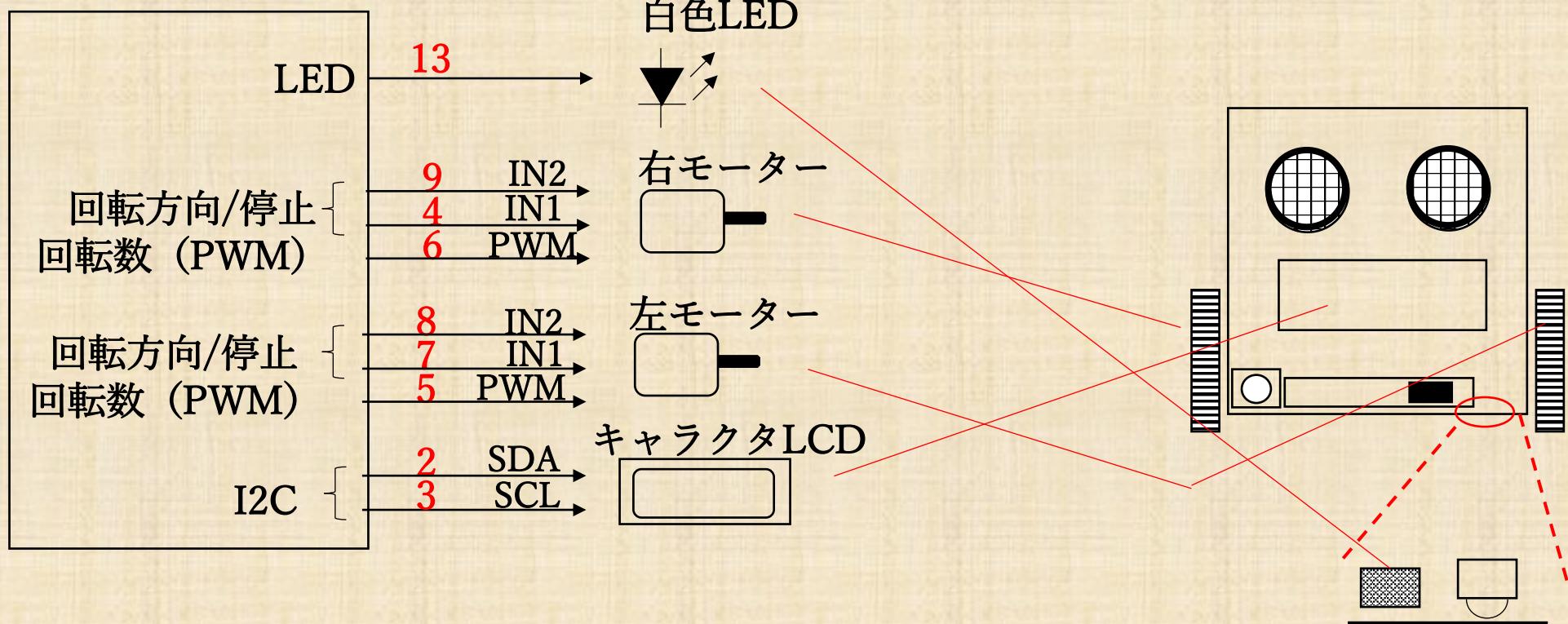
白色LED

フォトセンサー



# 出力信号の接続（参考）

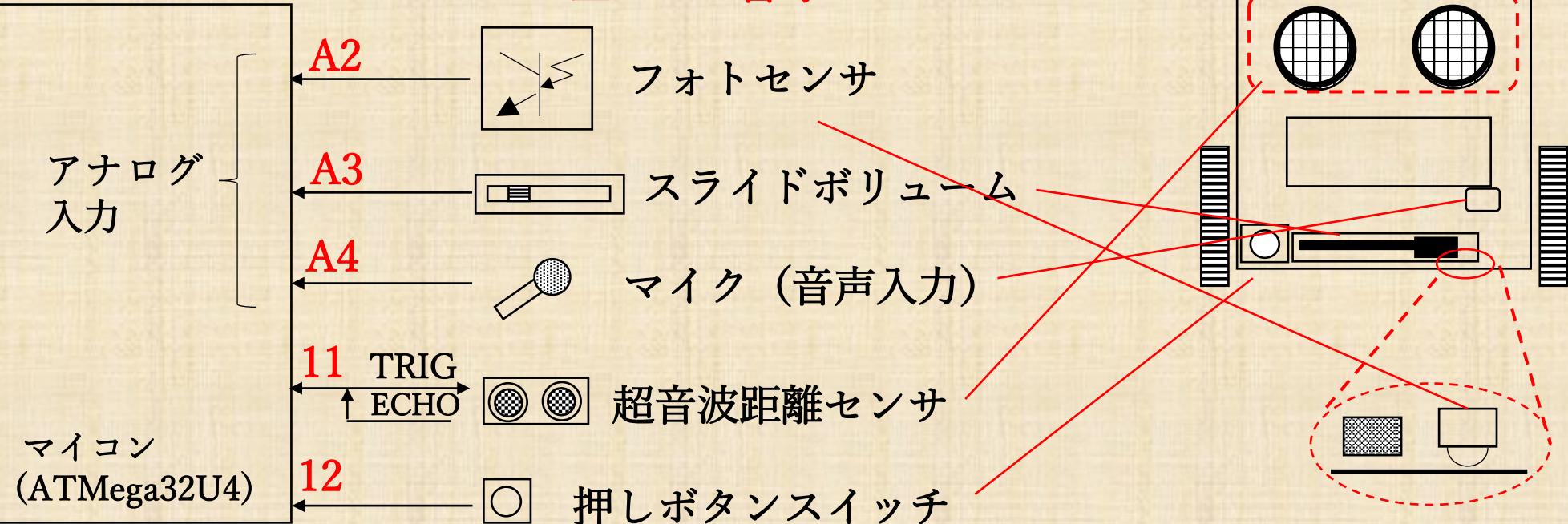
RDC-104 type II **Arduino-IDE上でのピン番号**



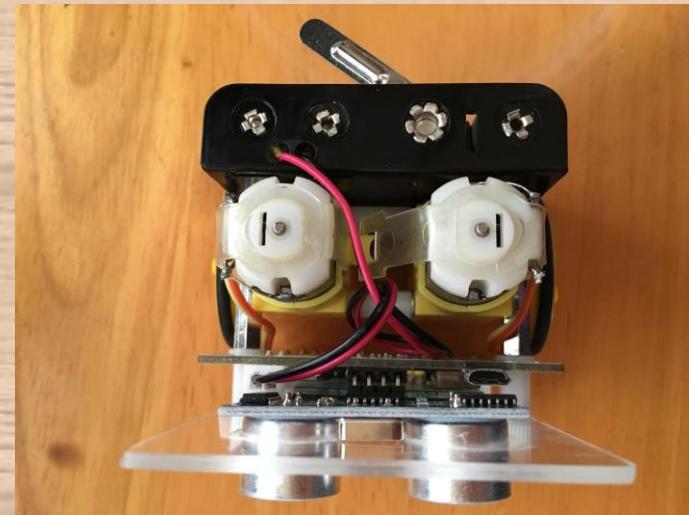
# 入力信号の接続（参考）

RDC-104 type II

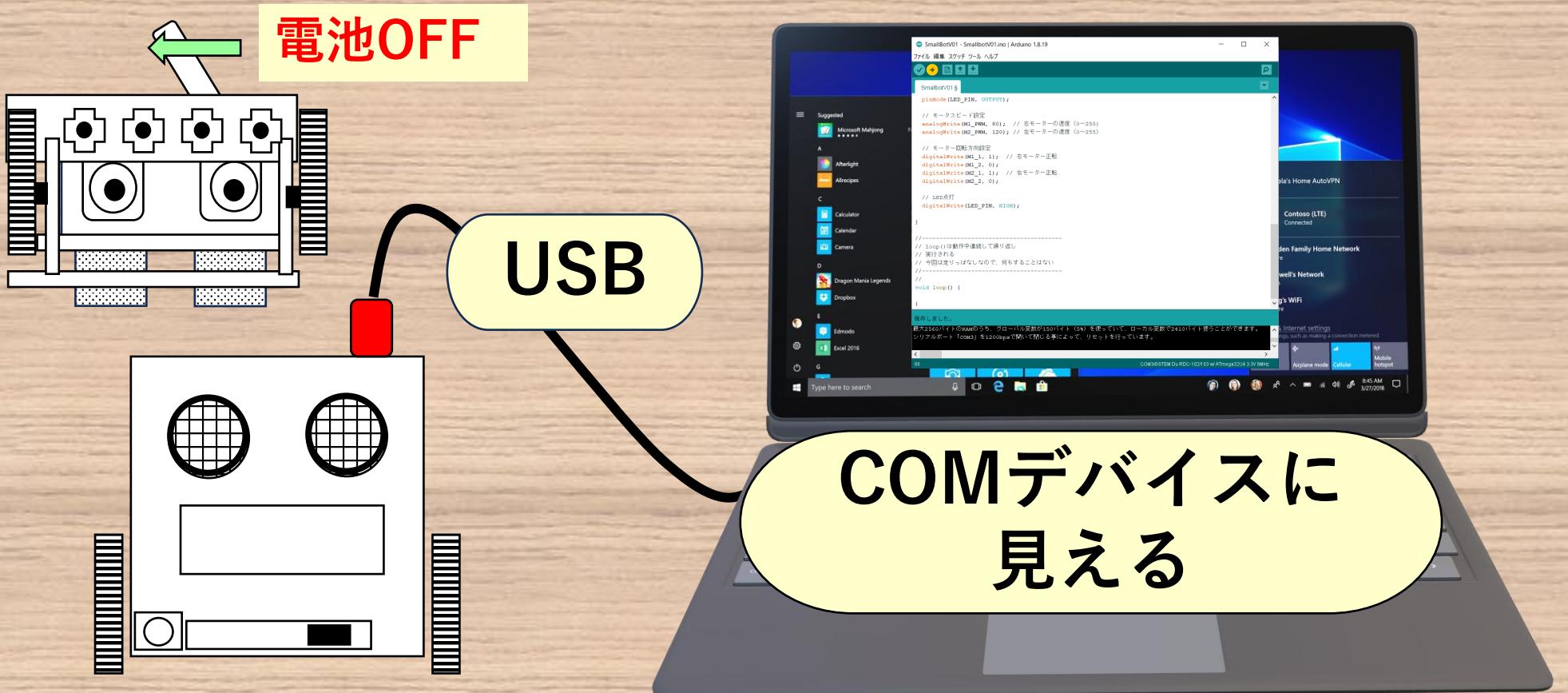
Arduino-IDE上のピン番号



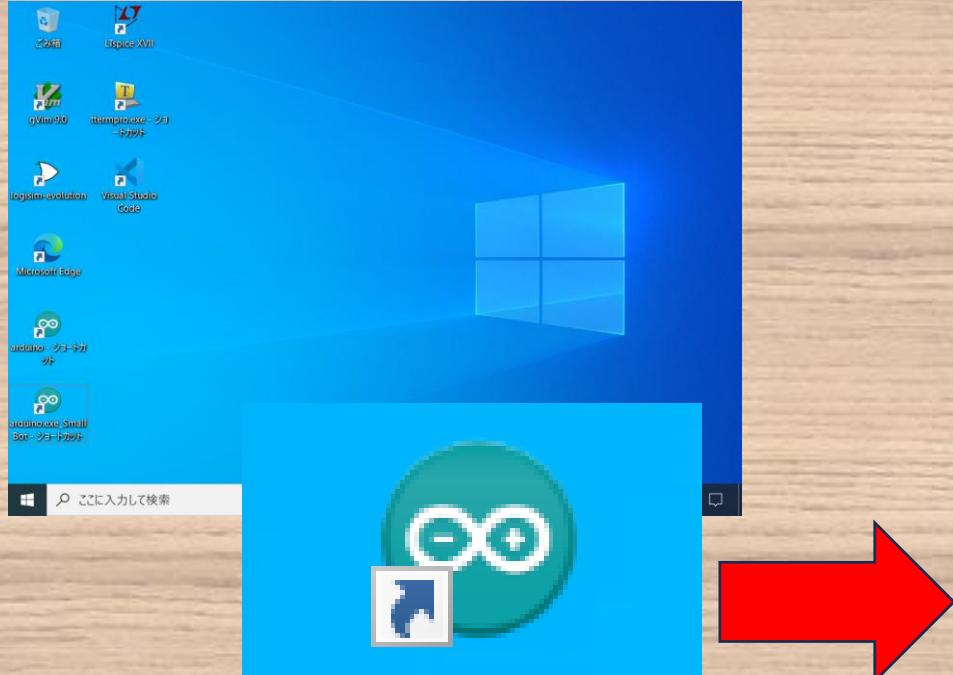
# 書き込んで動かすまでの手順 のおさらい



# PCと接続



# 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。

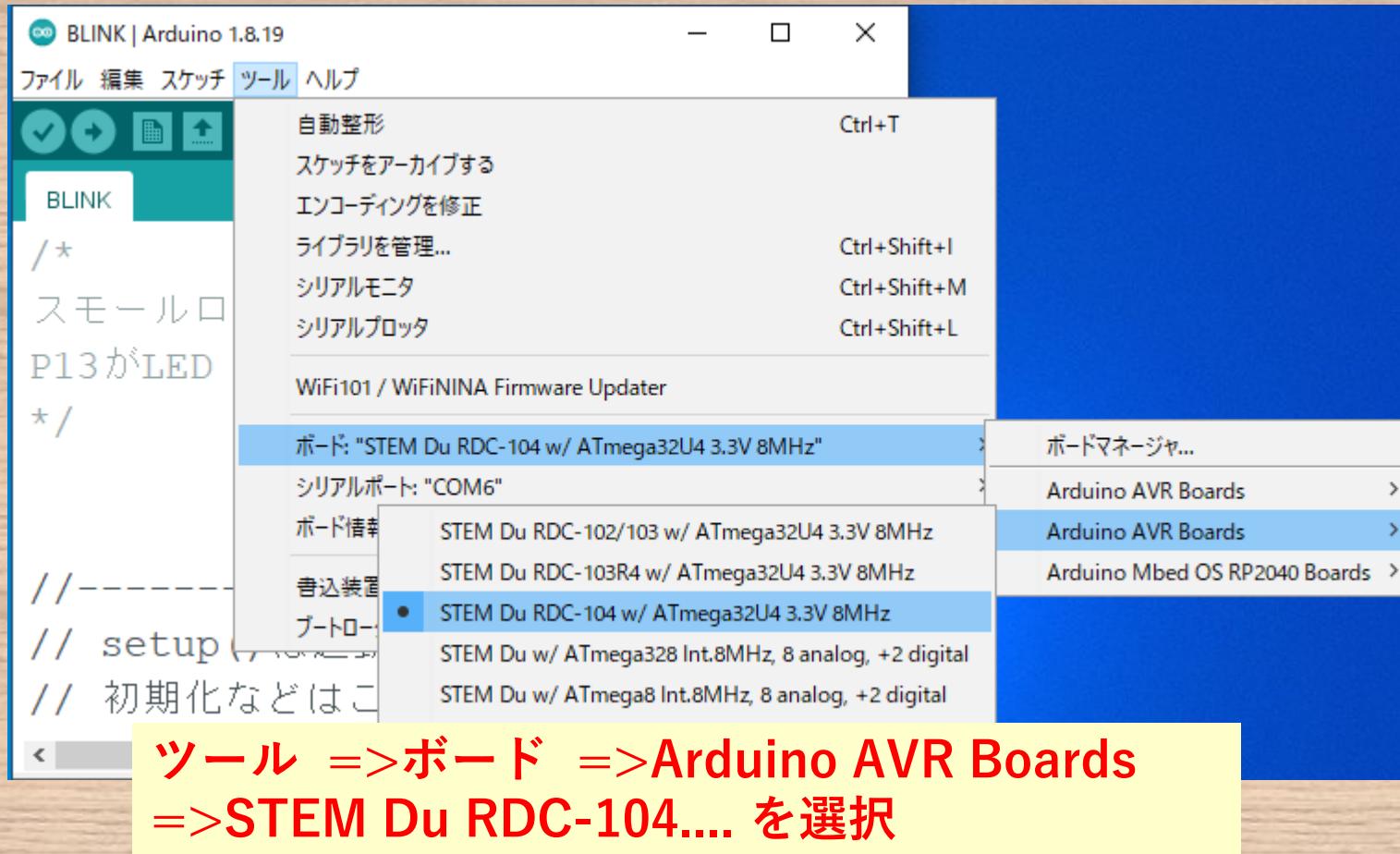
最大2560バイトのRAMのうち、グローバル変数が150バイト (5%) を使っていて、ローカル変数で2410バイト使うことができます。

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。

66

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

# ターゲットの選択



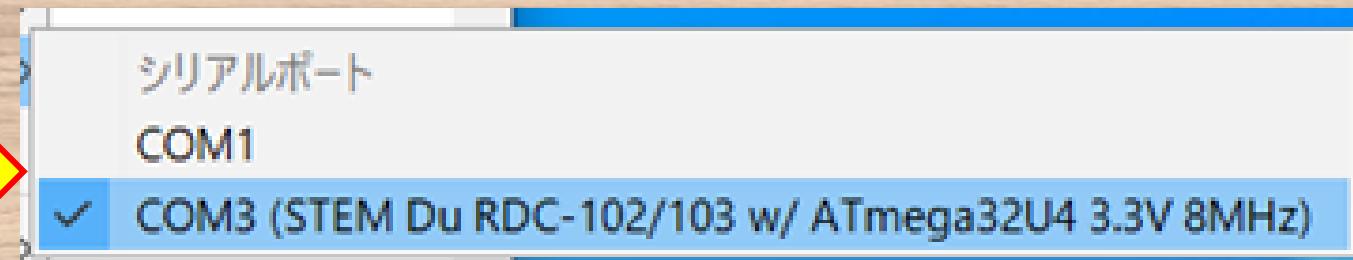
ツール => ボード => Arduino AVR Boards  
=> STEM Du RDC-104.... を選択

# 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/.....



# プログラムの記述

The screenshot shows the Arduino IDE interface. The title bar says "sketch\_oct10a | Arduino 1.8.19". The menu bar includes "ファイル 編集 スケッチ ツール ヘルプ". Below the menu is a toolbar with icons for upload, download, and other functions. The code editor contains the following code:

```
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 "loop" code block to the explanatory text on the right.

C/C++言語で記述

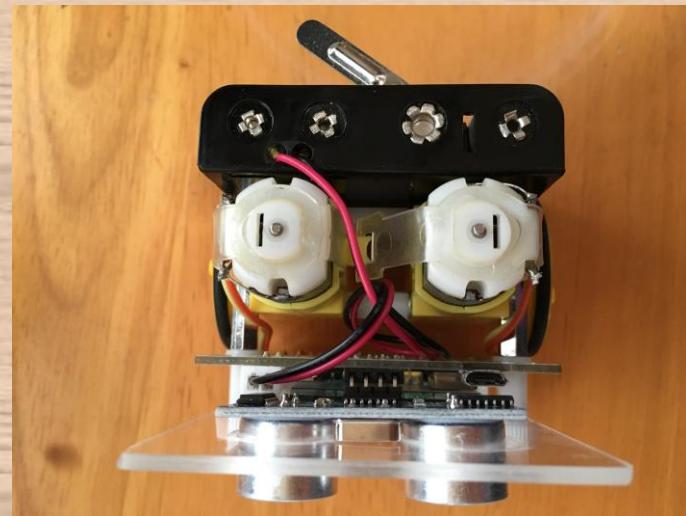
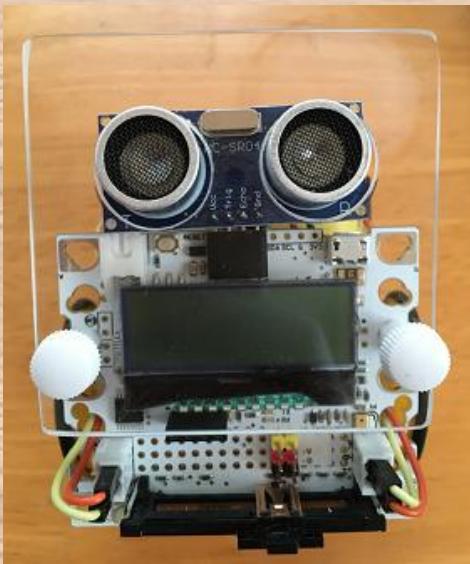
`void setup() {`

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

# モーターを動かそう



# モータ駆動サンプル (SmallbotV00.ino)

smallbot Public

main 1 branch 0 tags

neecrobot #defineを使ったモーター駆動サンプル

|                  |                       |
|------------------|-----------------------|
| Manual.pdf       | First Commit          |
| SmallBot_P01.pdf | 第一回テキスト更新             |
| SmallbotV00.ino  | #defineを使ったモーター駆動サンプル |
| SmallbotV10.ino  |                       |

SmallbotV00.ino

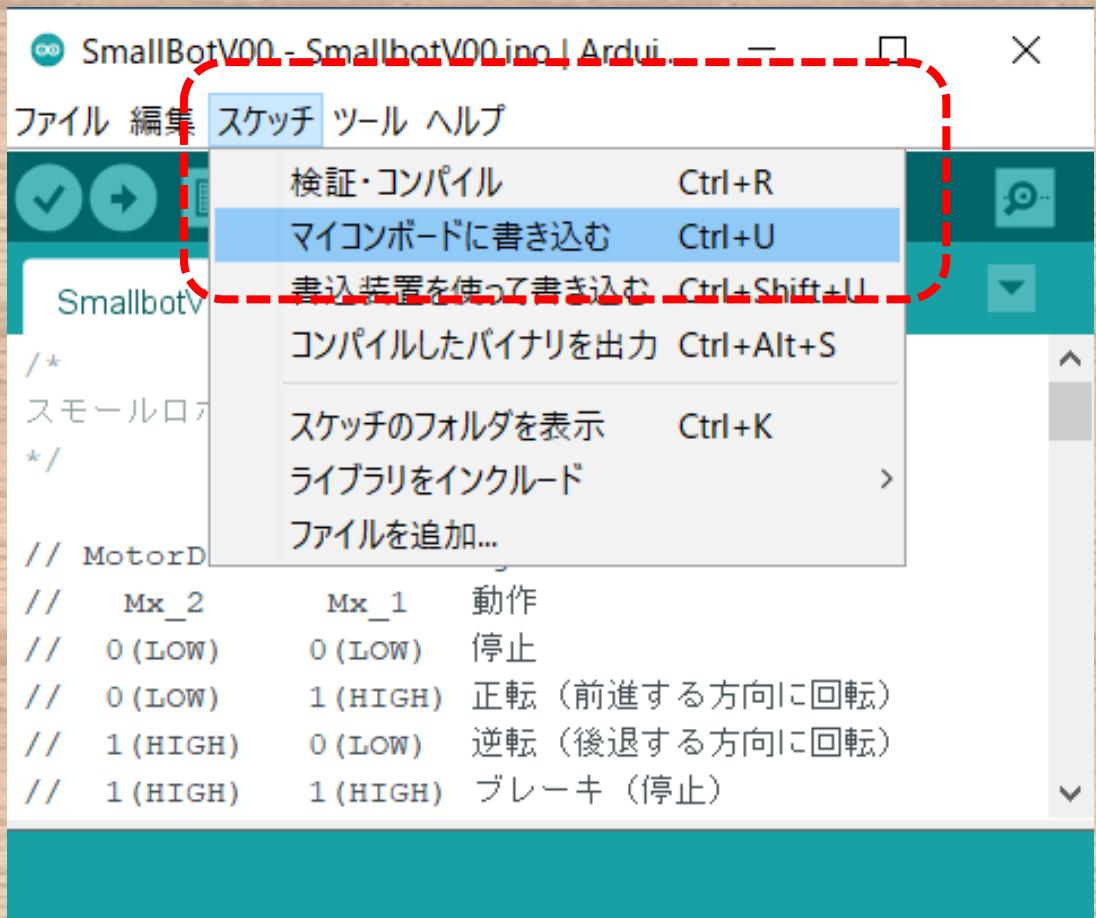
Code Blame 69 lines (56 loc) · 1.85 KB Code 55% faster with GitHub Codeship

```
/*
スモールロボット最初の一歩
モータの接続はM1が右、M2が左になっています。
明るさセンサの横にあるLED13を点灯することで、環境光の影響を減らします。
*/
// 右モータ
#define M1_1    4
#define M1_2    9
#define M1_PWM  6
// 左モータ
#define M2_1    7
#define M2_2    8
#define M2_PWM  5
```

Arduino-IDEにコピー  
&ペーストできる

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

# 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 left sidebar shows the sketch structure: "SmallbotV00" and its contents. The main code area contains the following C++ code:

```
/*
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... [progress bar]
```

The status bar at the bottom indicates "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.  
"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 (58)

リアルポート「COM3」を1200bpsで開いて閉じる事によって、リ

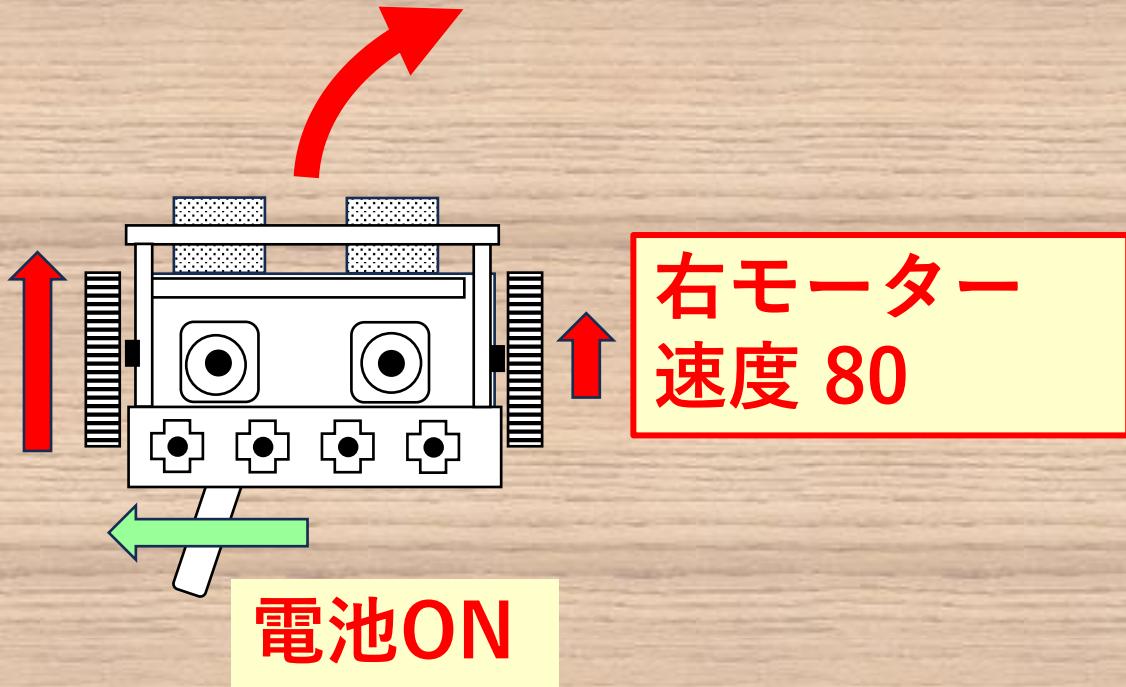
書き完了

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

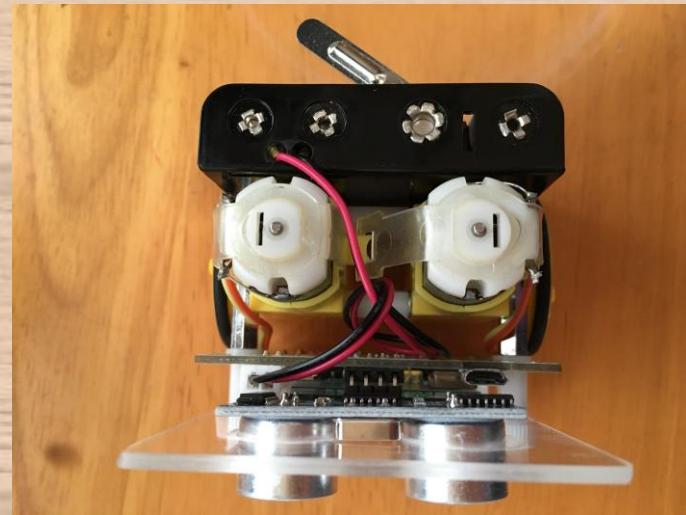
# 時計回りに回転する



左モーター  
速度 120



# 文字列の置き換えマクロ (#define) の使い方



# #defineの動作

```
pinMode(4, OUTPUT);  
pinMode(9, OUTPUT);  
pinMode(6, OUTPUT);
```

# define 置換前 置換後

```
#define M1_1      4  
#define M1_2      9  
#define M1_PWM    6  
pinMode(M1_1, OUTPUT);  
pinMode(M1_2, OUTPUT);  
pinMode(M1_PWM, OUTPUT);
```

"M1\_1"が"4"に置き換えられ、  
PinMode(4,OUTPUT)になる

# #defineを使ったサンプル (SmallbotV10.ino)

smallbot Public

main 1 branch 0 tags

neecrobot #defineを使ったモーター駆動サンプル

|                  |                       |
|------------------|-----------------------|
| Manual.pdf       | First Commit          |
| SmallBot_P01.pdf | 第一回テキスト更新             |
| SmallbotV00.ino  | プログラムV00（最初のプログラム）    |
| SmallbotV10.ino  | #defineを使ったモーター駆動サンプル |

**SmallbotV10.ino**

Code Blame 69 lines (56 loc) · 1.85 KB Code 55% faster with GitHub Codeship

```
/*
スモールロボット最初の一歩
モータの接続はM1が右、M2が左になっています。
明るさセンサの横にあるLED13を点灯することで、環境光の影響を減らします。
*/
// 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）

// #define M2_1
// #define M2_2
// #define M2_PWM

#define M2_1 7
#define M2_2 8
#define M2_PWM 5
```

開く

Arduno-IDEにコピー  
&ペーストできる

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

# コピー＆ペースして動かしてみよう

Code    Blame    69 lines (56 loc) · 1.85 KB    Code 55% faster with GitHub CodeSync

```
1  /*
2   スモールロボット最初の一歩
3   モータの接続はM1が右、M2が左になっています。
4   明るさセンサの横にあるLED13を点灯することで、環境光の影響を減らします。
5  */
6
7  // MotorDriver Pin Assign on RDC.
8  // Mx_2      Mx_1    動作
9  // 0(LOW)    0(LOW)  停止
10 // 0(LOW)    1(HIGH) 正転（前進する方向に回転）
11 // 1(HIGH)   0(LOW)  逆転（後退する方向に回転）
12 // 1(HIGH)   1(HIGH) ブレーキ（停止）
13
14 // 右モータ
15 #define M1_1      4
16 #define M1_2      9
17 #define M1_PWM    6
18
19 // 左モータ
20 #define M2_1      7
21 #define M2_2      8
22 #define M2_PWM    5
```

SmallBotV10 - SmallbotV10.ino | Arduino 1.8.19

ファイル 編集 スケッチ ツール ヘルプ

SmallbotV10

```
// 右モータ
#define M1_1      4
#define M1_2      9
#define M1_PWM    6

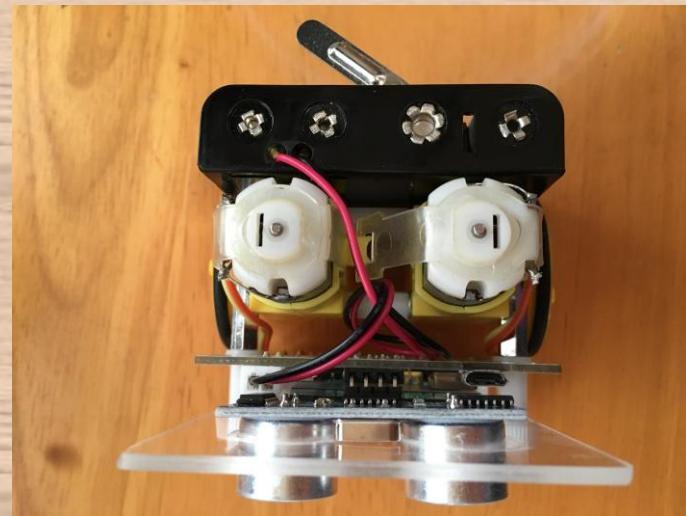
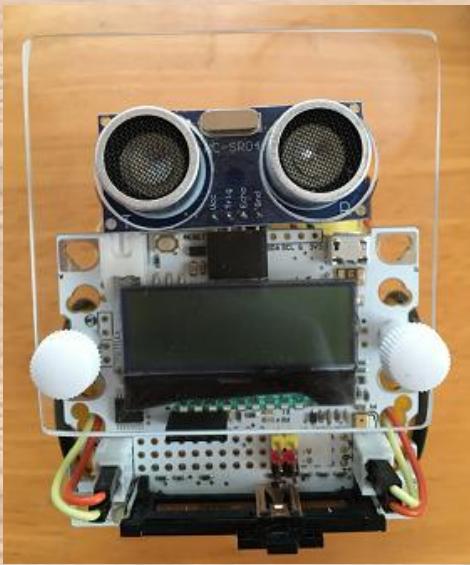
// 左モータ
#define M2_1      7
#define M2_2      8
```

ボードへの書き込みが完了しました。

# SmallBot用の定義例

|                    |    |              |
|--------------------|----|--------------|
| #define M1_1       | 4  | // 右モータ回転方向  |
| #define M1_2       | 9  |              |
| #define M1_PWM     | 6  | // 右モーター速度   |
| #define M2_1       | 7  | // 左モータ回転方向  |
| #define M2_2       | 8  |              |
| #define M2_PWM     | 5  | // 左モーター速度   |
| #define PING_PIN   | 11 | // 超音波測距センサ  |
| #define BUTTON_PIN | 12 | // 押しボタンスイッチ |
| #define LED_PIN    | 13 | // LED       |
| #define PHOTO      | A2 | // フォトセンサ    |
| #define SLIDER     | A3 | // スライダ      |

# C言語をちょっとだけ



# まずはこれだけ

- 整数型変数 (int型)

<宣言>

```
int data;
```

```
int data = 0; // 初期化あり
```

<代入>

```
data = 3;
```

```
data = data+4;
```

- 条件判断と分岐

< if 文>

```
if (data == 3) {
```

条件成立時の処理

```
} else {
```

条件不成立時の処理

```
}
```

$==$  等しい

$\neq$  等しくない

$>$

$\geq$

左辺 > 右辺

左辺  $\geq$  右辺

$<$

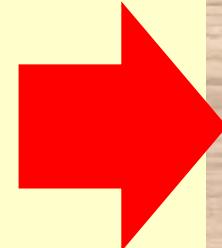
$\leq$

左辺 < 右辺

左辺  $\leq$  右辺

# `if() ... else if()... else`

```
if (data == 3) {  
    data = 2;  
}  
  
if (data == 2) {  
    data = 3;  
}
```



=> 3の時も3になってしまう

```
if (data == 3) {  
    data = 2;  
} else if (data == 2) {  
    data = 3;  
} else {  
    data = 0;  
}
```

3のときは 2  
2のときは 3  
それ以外なら 0

# 動かしてみよう (SmallbotV20.ino)

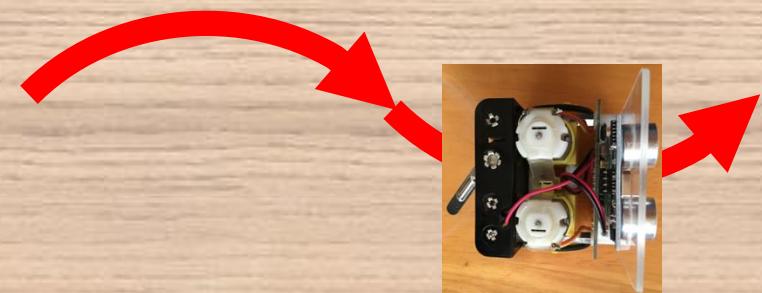
SmallBotV20 | Arduino 1.8.19

ファイル 編集 スケッチ ツール ヘルプ

SmallBotV20

```
void loop() {
    if (dir == 0) {      // dirが0だった時
        // モータスピード設定
        analogWrite(M1_PWM, 80); // 右回り
        analogWrite(M2_PWM, 120); // 左回り
    } else {
        // モータスピード設定
        analogWrite(M1_PWM, 120); // 右回り
        analogWrite(M2_PWM, 80); // 左回り
    }
}
```

GithubのV20.ino  
をコピー&ペースト  
して実行してみよう



# 変数とif文の利用 (V20.inoから抜粋)

左右交互に繰り返し

```
int dir = 0;  
void loop() {  
    if (dir == 0) { // dirが0  
        analogWrite(M1_PWM, 80);  
        analogWrite(M2_PWM, 120);  
        dir = 1; // 次は左回転  
    }  
}
```

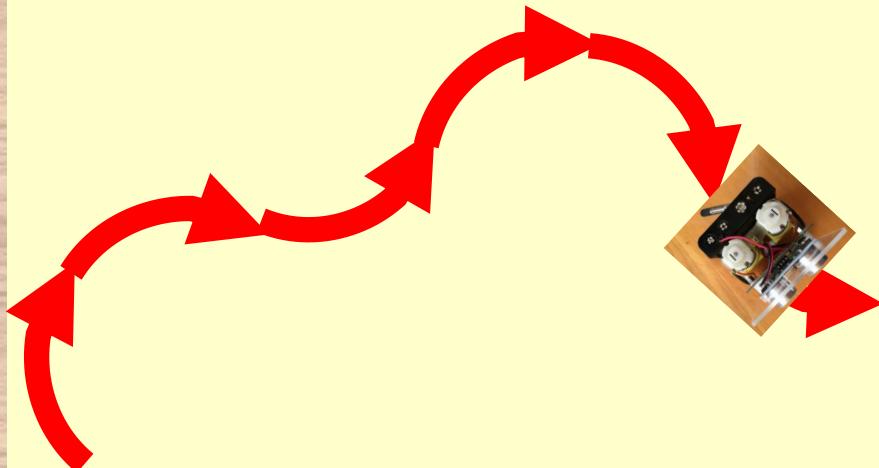
変数dirが0なら  
右回転に設定  
dirの値を1にする

```
else {  
    analogWrite(M1_PWM, 120);  
    analogWrite(M2_PWM, 80);  
    dir = 0; // 次は右回転  
}  
delay(1000); // 1秒待つ  
}
```

変数dirが0でないなら  
左回転に設定  
dirの値を0にする

# 改造してみよう

1：右に2秒回転  
左に1秒回転  
を繰り返すようにしてみよう  
ヒント：“>” や “>=” の利用

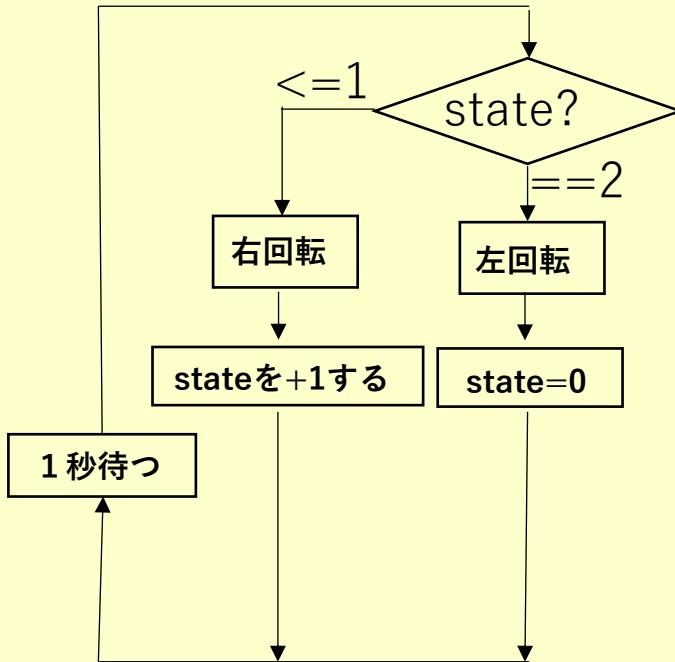


2：1の動きを4回行った  
後、**停止**するように  
してみよう  
ヒント：0,1,2, 3,4,5, … ,12

3：変数を二つ使って2と  
同じ動きにしてみよう  
ヒント： 0 : 0,1,2  
1 : 0,1,2  
…

# 考え方の例－1

1：右に2秒回転  
左に1秒回転  
を繰り返すようにしてみよう



```
int state = 0;
void loop() {
    if (state <= 1) { // < 2 でもいい
        <右回転>
        state = state + 1;
    } else {
        <左回転>
        state = 0;
    }
    delay(1000);
}
```



# 考え方の例 – 2

2 : 1の動きを4回行った後、**停止**するようにしてみよう

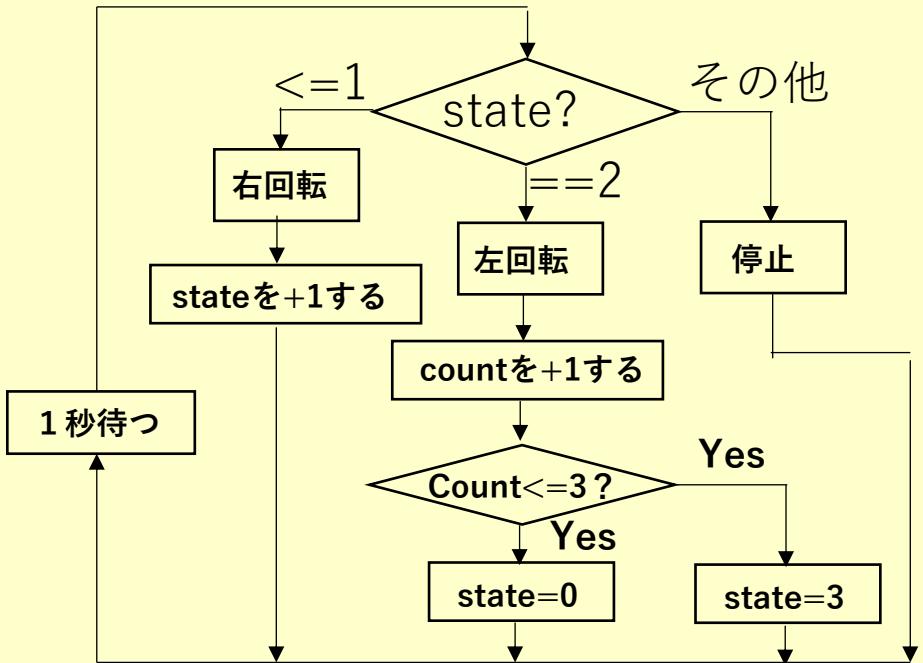
ヒント : 0,1,2, 3,4,5, … ,12  
右右左 右右左 … ,停止

```
int state = 0;  
void loop() {  
    if (state <= 1) //0,1の時  
        <右回転>  
        state = state + 1;  
}
```

```
else if (state == 2) { // 2の時  
    <左回転>  
    state = state + 1;  
} else if (state <= 4) { // 3,4の時  
    <右回転>  
    state = state + 1;  
} else if · · ·  
· · ·  
} else if (state == 12) { // 終了  
    <停止>  
}  
delay(1000);
```

# 考え方の例－3-1

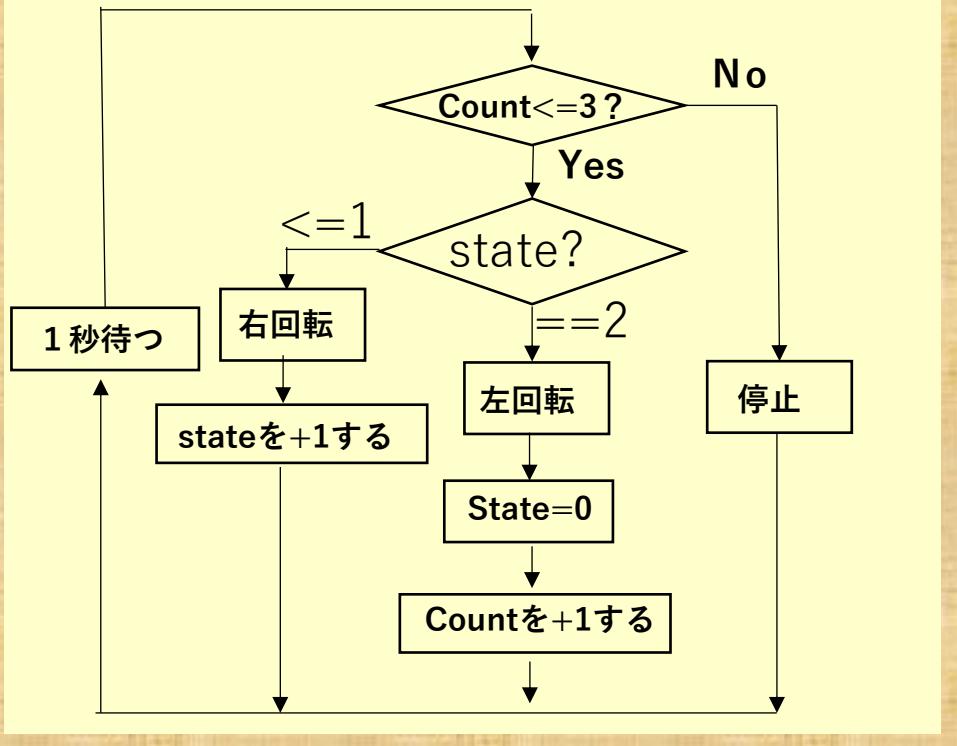
## 3：変数を二つ使って2と同じ動きにしてみよう



```
int state = 0;  
int count = 0;  
void loop() {  
    if (state <= 1) { // 0,1なら右回転  
        <右回転>  
        state = state + 1; // 1増やす  
    } else if (state == 2) { // 2なら左回転  
        <左回転>  
        count = count + 1; // 回数カウントを増やす  
        if (count <= 3) // 回数カウントが1,2,3なら  
            state = 0; // もう一回  
        else // 4になったら  
            state = 3; // 終わり  
    } else { // 4だったら  
        <モータ停止>  
    }  
    delay(1000); // 1秒待つ  
}
```

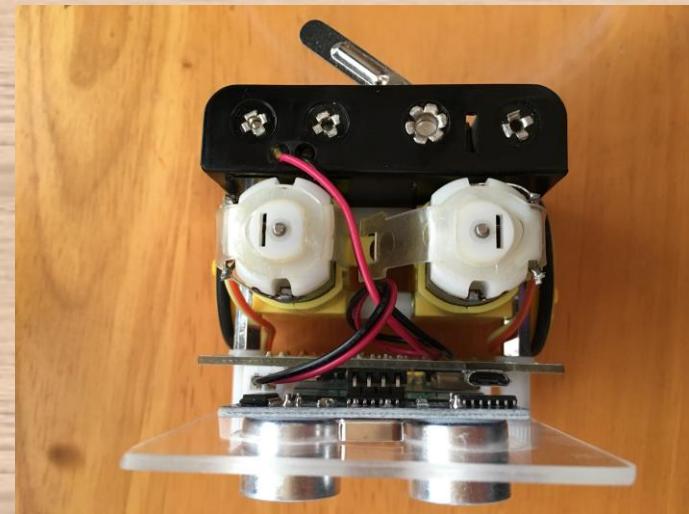
# 考え方の例－3-2

## 3：変数を二つ使って2と同じ動きにしてみよう



```
int state = 0;  
int count = 0;  
void loop() {  
    if (count <= 3) { // 0,1,2,3なら  
        if (state <= 1) { // 0,1なら右回転  
            <右回転>  
            state = state + 1;  
        } else if (state == 2) { // 2なら左回転  
            <左回転>  
            state = 0;  
        }  
        count = count + 1; // 1回分終わり  
    }  
    delay(1000);  
} else { // 4回繰り返したら  
    <モータ停止>  
}
```

# フォトセンサを使って ライントレースさせてみよう



# 使用するセンサ類

キャラクタLCD

右モーター

押しボタン  
スイッチ

スライドボリューム

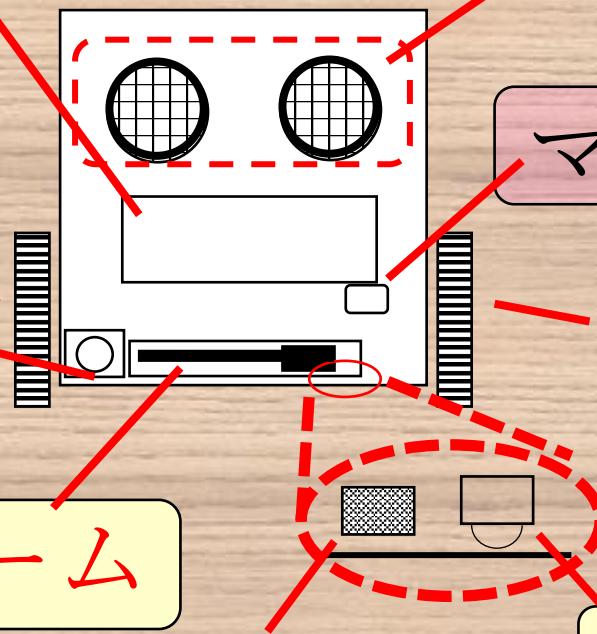
超音波距離センサ

マイク

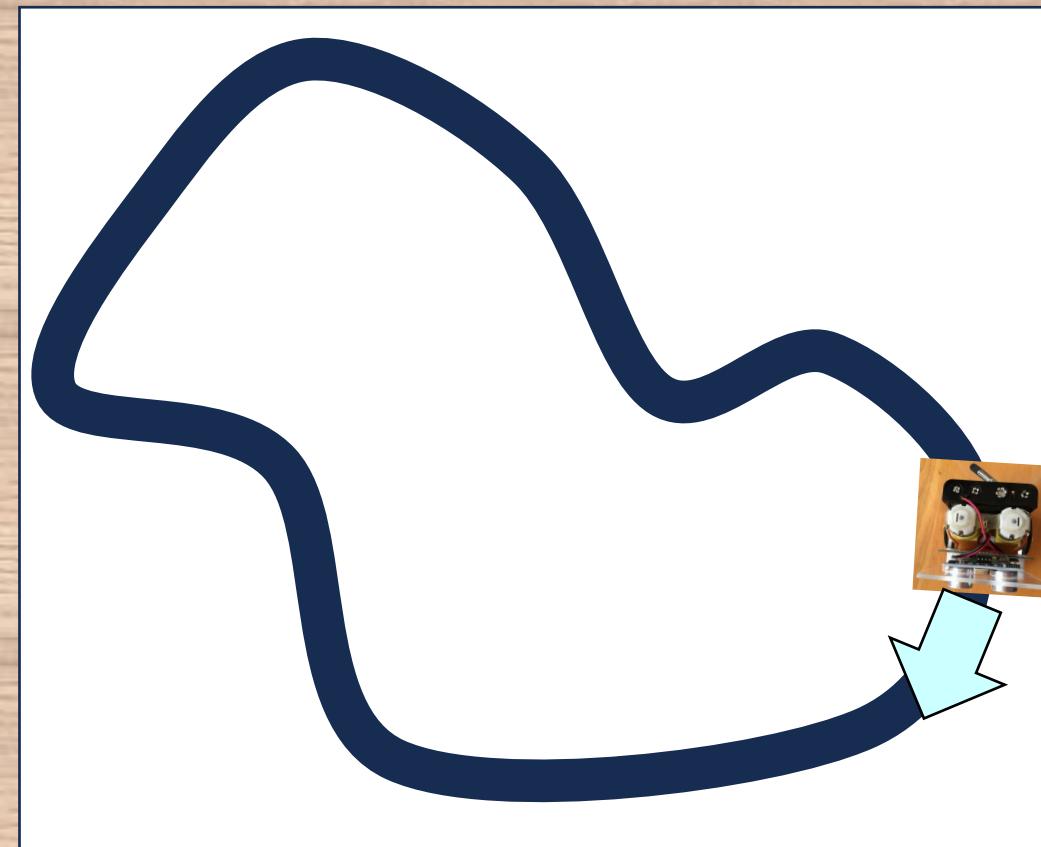
左モーター

フォトセンサー

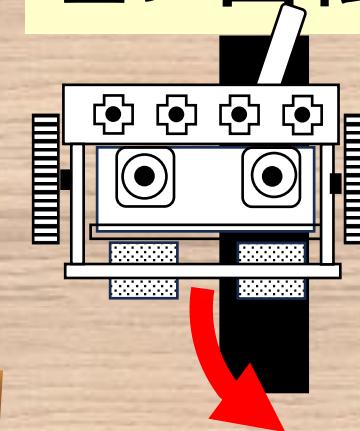
白色LED



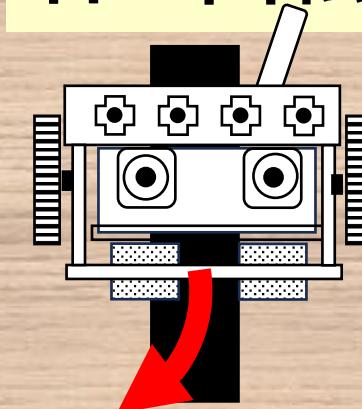
# ライントレースロボット



左に回転



右に回転



# ライントレースサンプル (SmallbotV30\_Trace.ino)

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

The GitHub repository page shows the file structure. A red circle highlights the 'SmallbotV30\_Trace.ino' file, and a red arrow points from it to the code editor window.

The code editor displays the Arduino sketch. The code is as follows:

```
/*
ライントレース(LED点灯)
明るさセンサを使ったライントレースです。
スライダーで明るさセンサのしきい値を変更できます
センサが1つですので、ラインの片側のエッジ(白と黒)
ライン上にスマートボットの中央を置くと、左回りセンサが白の上なら右旋回、センサが黒の上
センサが白の上なら左旋回、センサが黒の上なら右旋回
*/
Forward
|| M2 light || M1
||

モータの接続はM1が右、M2が左になっています。
```

The Arduino IDE shows the uploaded code. A red arrow points from the GitHub code editor to the Arduino IDE interface. The status bar at the bottom right of the IDE says "ボードへの書き込みが完了しました。" (Writing to board completed).



SmallbotV30\_Trace.ino

ライントレースサンプル

# ライントレースのポイント (SmallbotV30\_Trace.inoから抜粋)

```
int sliderval = 0; // スライダーの値
int photoval = 0; // フォトセンサの値
void loop() {
    sliderval = analogRead(SLIDER);
    photoval = analogRead(PHOTO);
    if (photoval > sliderval) { // 右旋回
        analogWrite(M1_PWM, 80);
        analogWrite(M2_PWM, 120);
    }
}
```

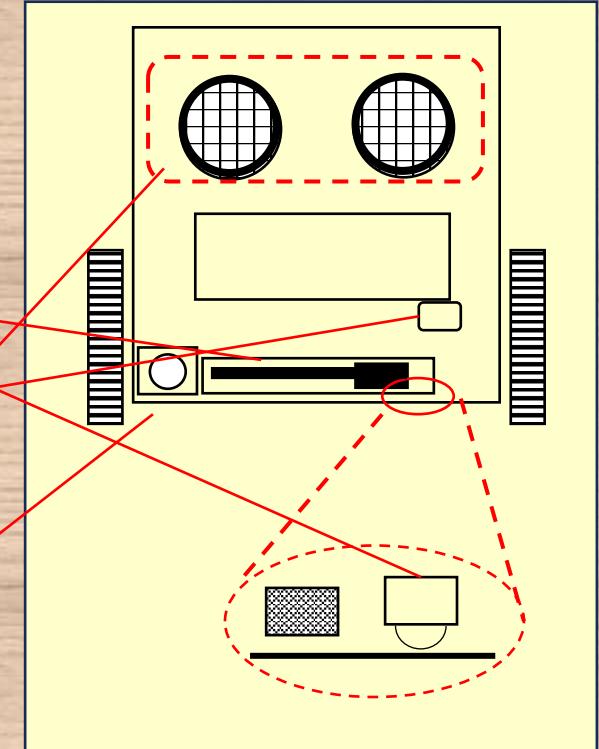
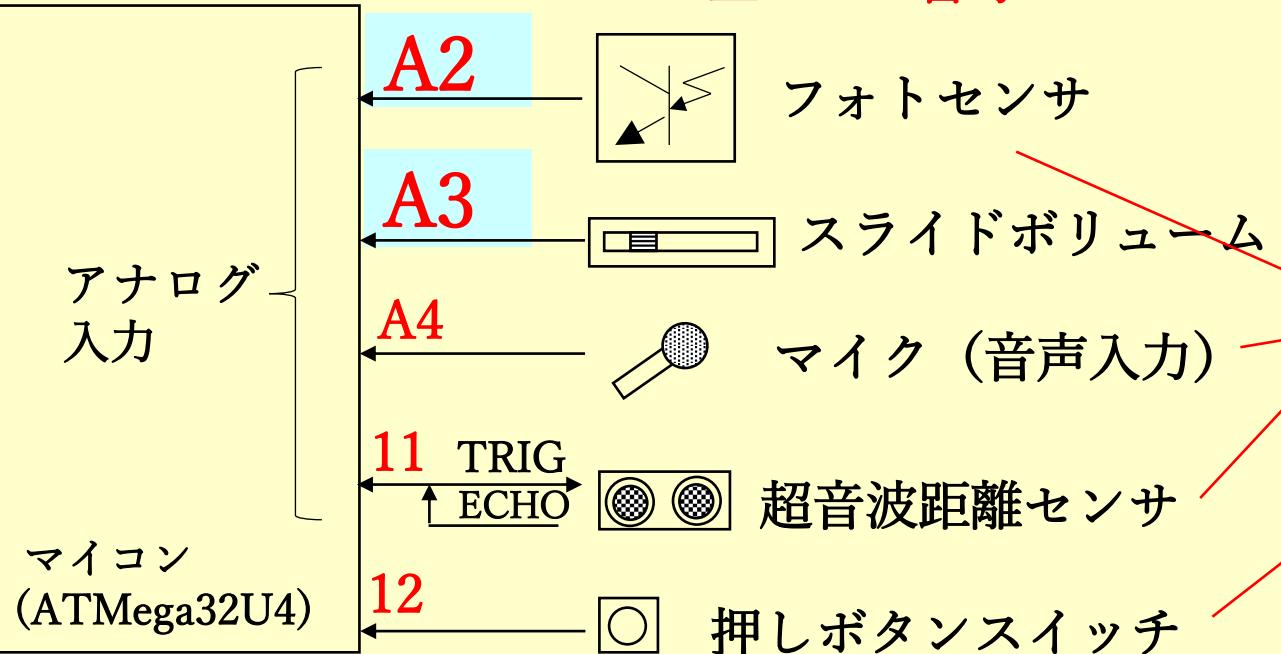
```
else { // 左旋回
    analogWrite(M1_PWM, 120);
    analogWrite(M2_PWM, 80);
}
delay(100); // 0.1秒現状維持
}
```

フォトセンサの値をスライダーの値と比較して、線の上か否かを決めている

# 入力信号の接続

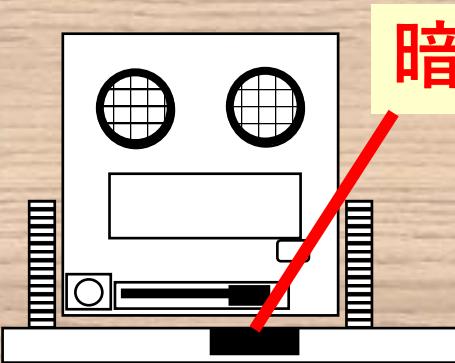
RDC-104 type II

Arduino-IDE上のピン番号

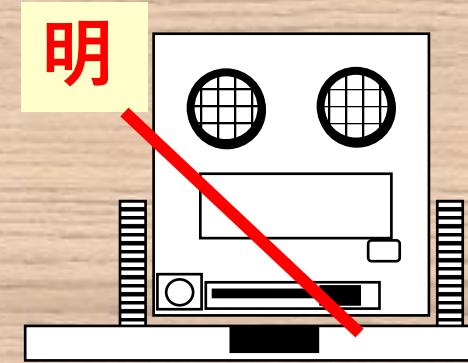


# センサ類の値の取得と大小

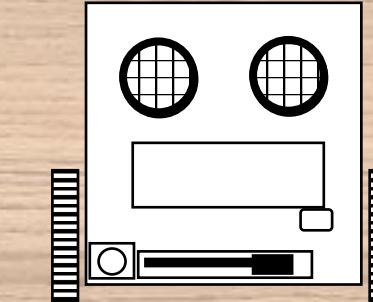
(analogRead()の値は0~1023)



フォトセンサ値小



フォトセンサ値大



スライダ値 小

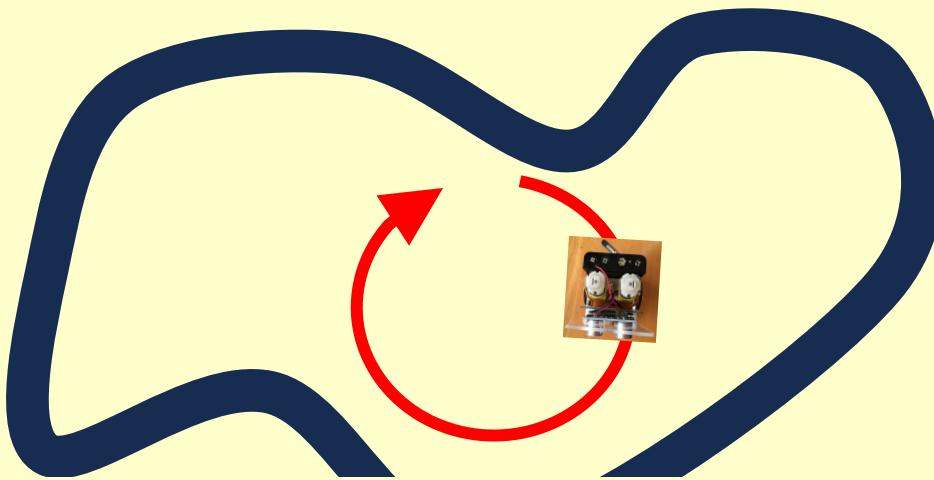


スライダ値 大

```
#define PHOTO A2  
photoval = analogRead(PHOTO);  
  
#define SLIDER A3  
sliderval = analogRead(SLIDER);
```

# 改造してみよう

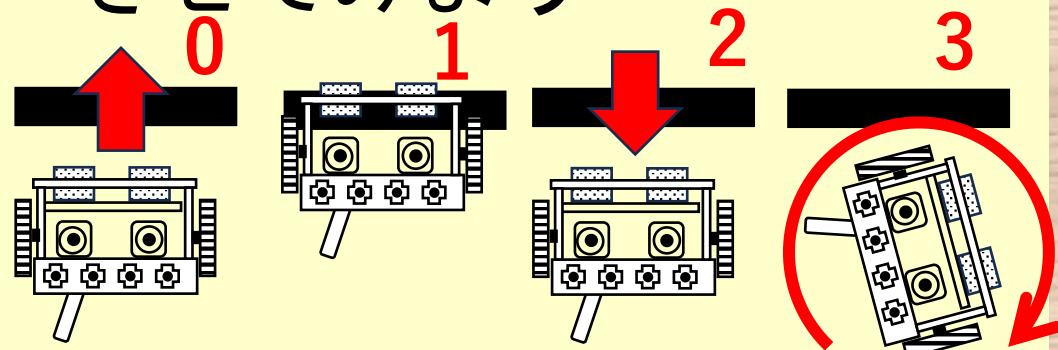
1：線が見つからない時  
回転し続けずに線を  
探しに行くには？



ヒント：途中で方向を変える

2：通常は直進

黒い線(停止線)を検出  
=>1秒間後退して  
1秒間右回転  
させてみよう



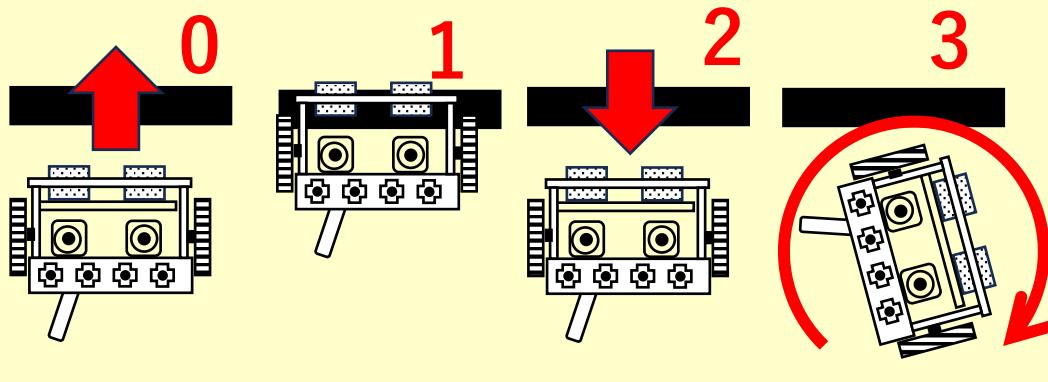
# 考え方の例 – 1

1 : 線が見つからない時  
回転し続けずに線を  
探しに行くには？

```
count = 0;  
void loop() {  
....  
if (photoval>sliderval) { //白地  
    count = count+1;  
if (count <= 20) { //0.1×20= 2秒  
    <右旋回>  
}
```

```
else if (count < 40) { //0.1×40= 4秒  
    <左旋回>  
} else {  
    count = 0; // 右旋回やりなおし  
}  
} else { // 黒地  
    count = 0;  
    <左旋回>  
}  
delay(100); //0.1秒  
}
```

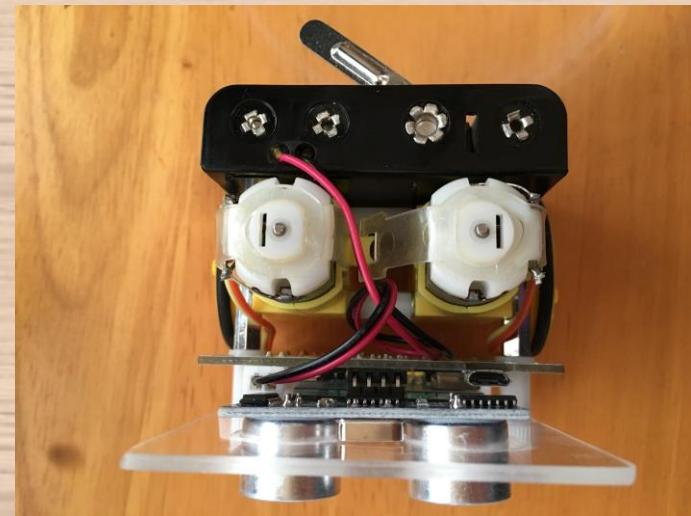
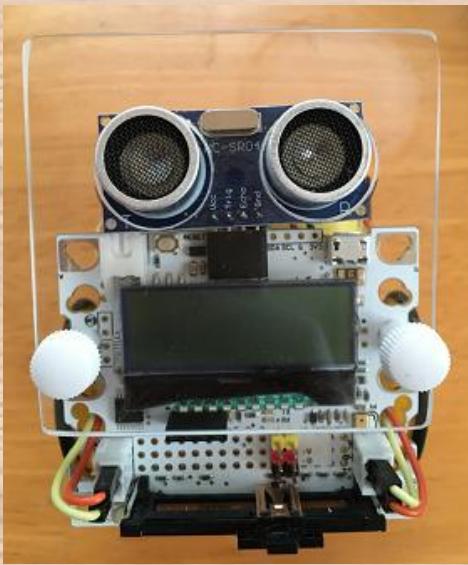
# 考え方の例 – 2



```
count = 0;  
void loop() {  
if (state == 0) { // 直進中  
if (photoval > sliderval) { // 白線  
<直進>  
} else { // 黒線
```

```
count = 10; // タイマ  
state = 2; // 後退  
}  
} else if (state == 2) {  
<後退>  
count = count-1;  
if (count == 0) {  
count = 10; state = 3;  
}  
. . . . .  
}
```

# お疲れ様でした



次回はマイクと超音波センサを使ってみましょう