

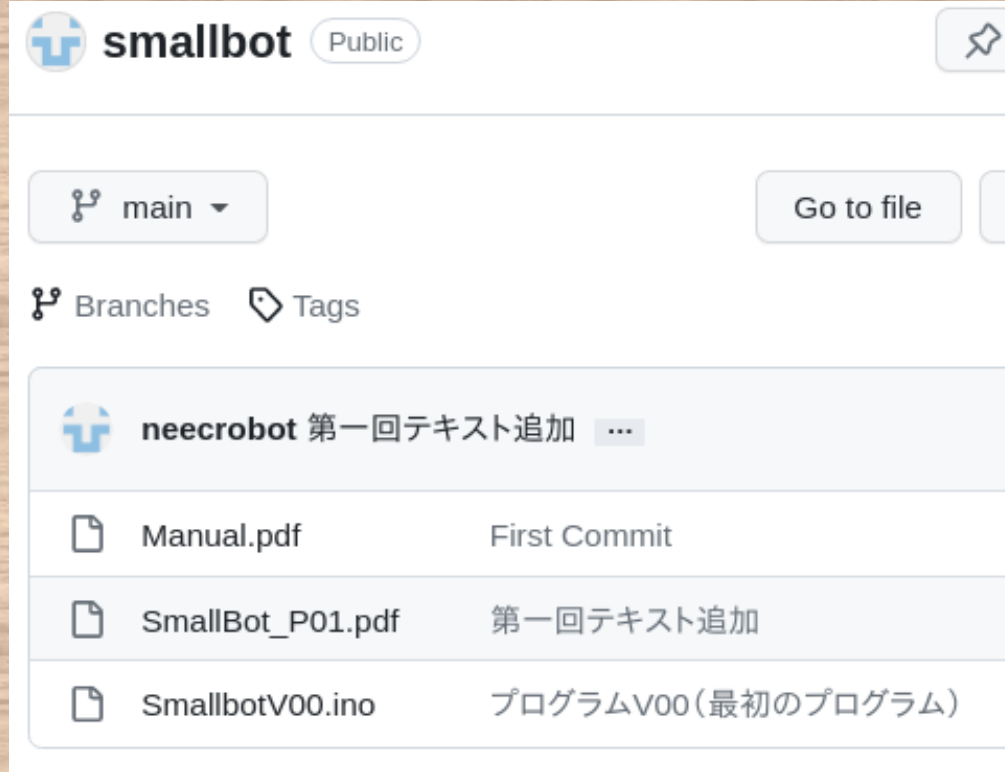
ロボットプログラミング#2



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

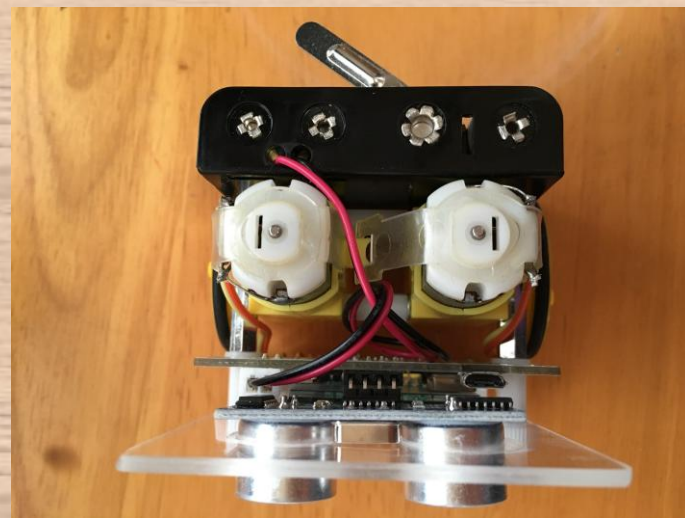
マニュアルやプログラム類は githubに置いてあります



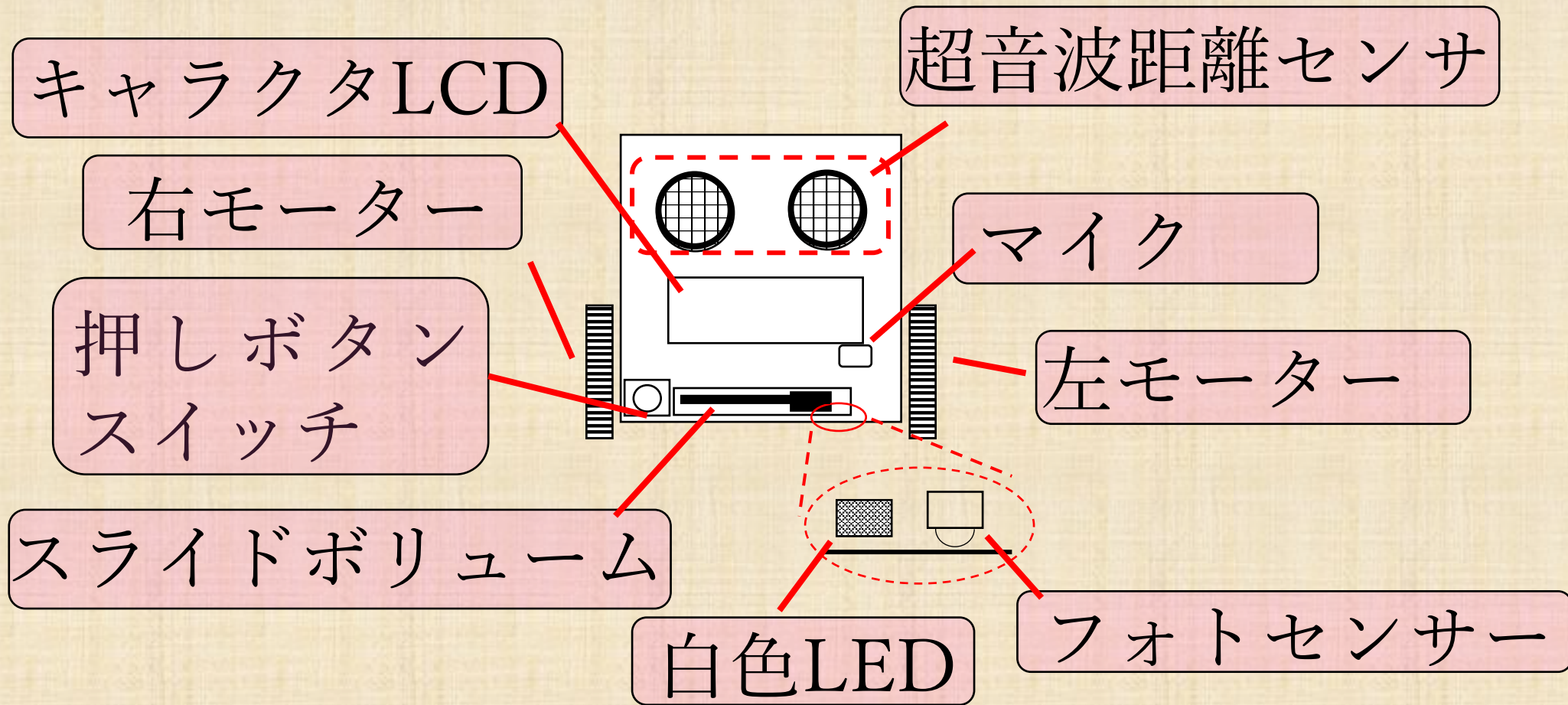
置いてあるファイル
はクリックして
ダウンロード可

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

SmallBotについて

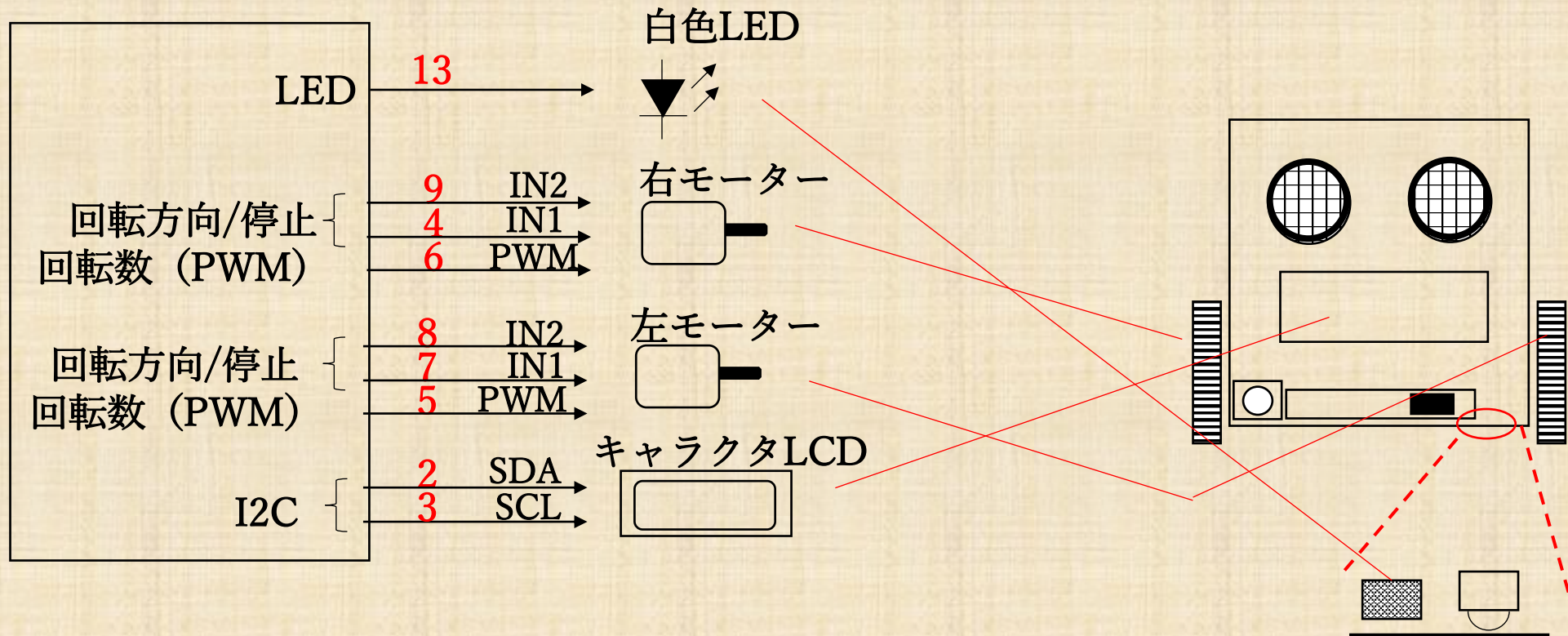


SmallBotの主要部品配置



出力信号の接続（参考）

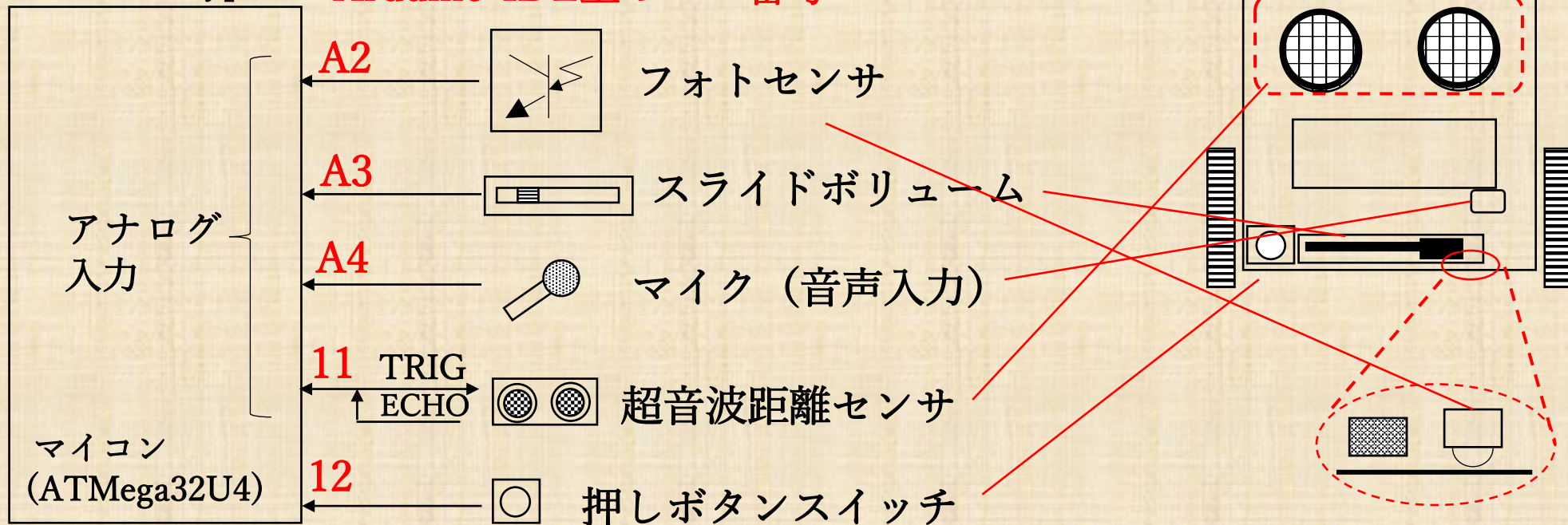
RDC-104 type II **Arduino-IDE上のピン番号**



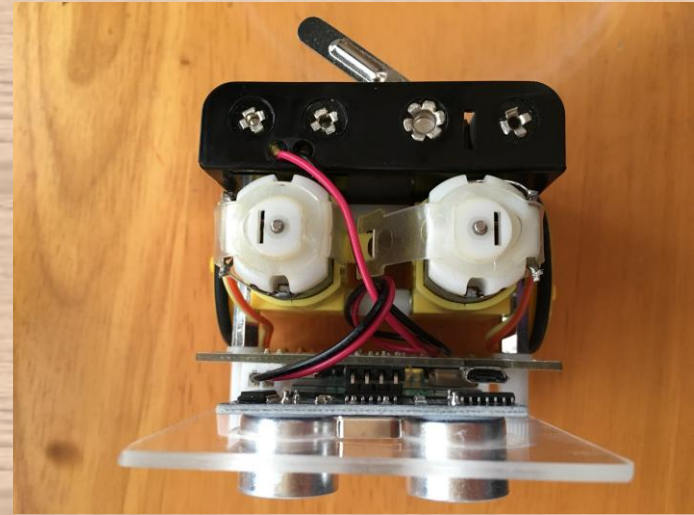
入力信号の接続（参考）

RDC-104 type II

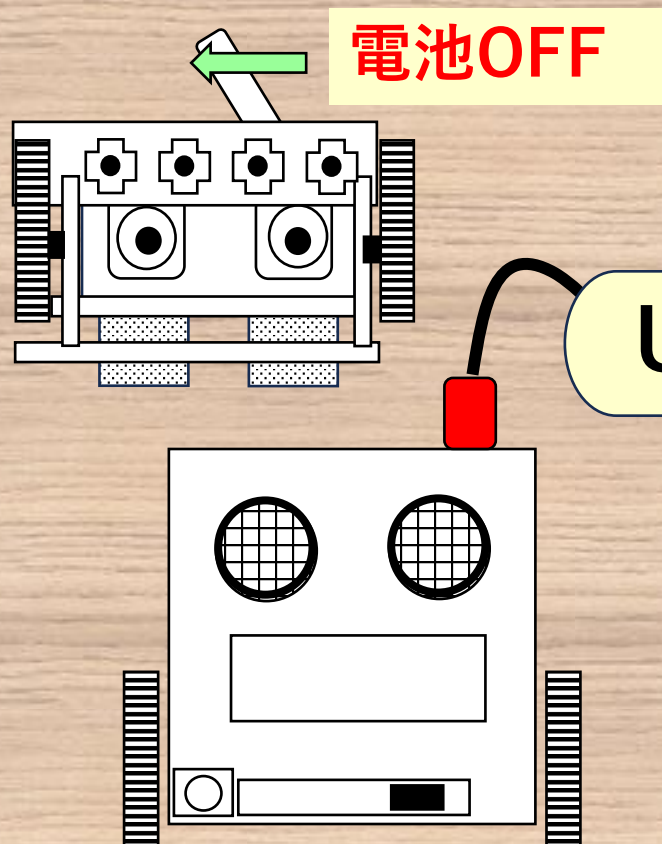
Arduino-IDE上のピン番号



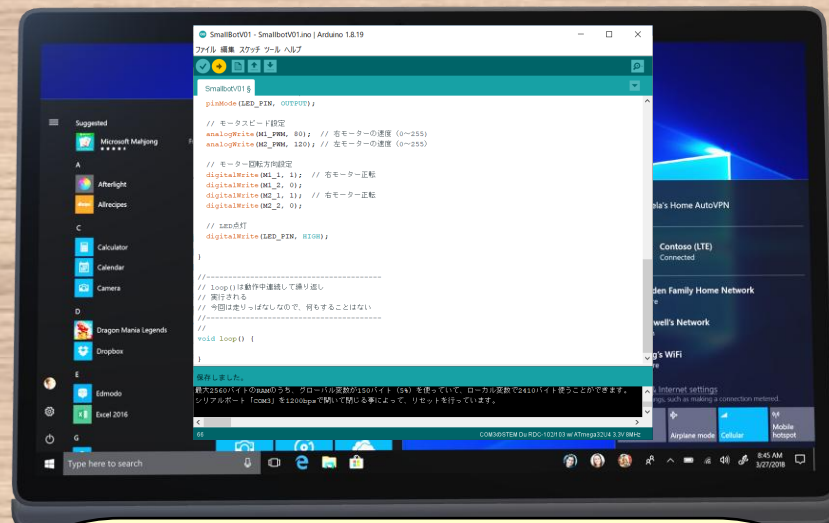
書き込んで動かすまでの手順 のおさらい



PCと接続

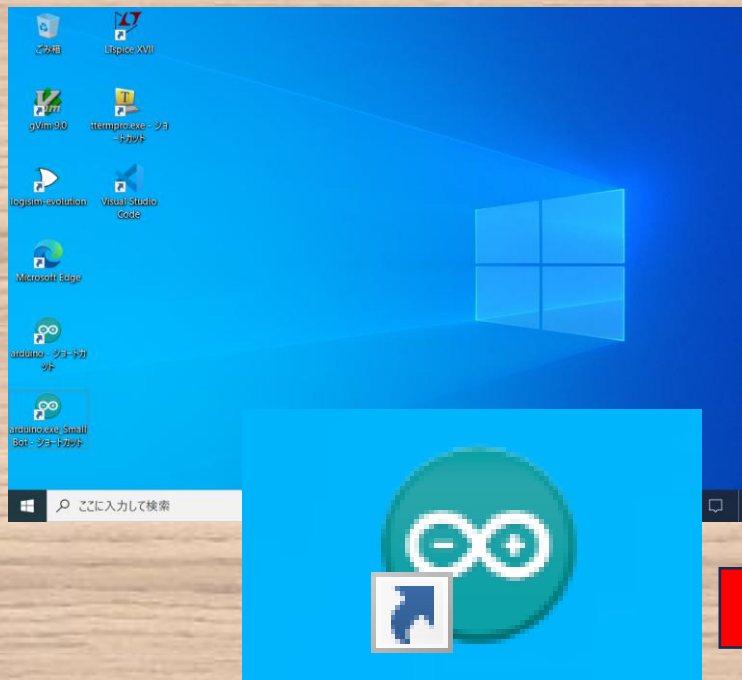


USB



COMデバイスに
見える

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:

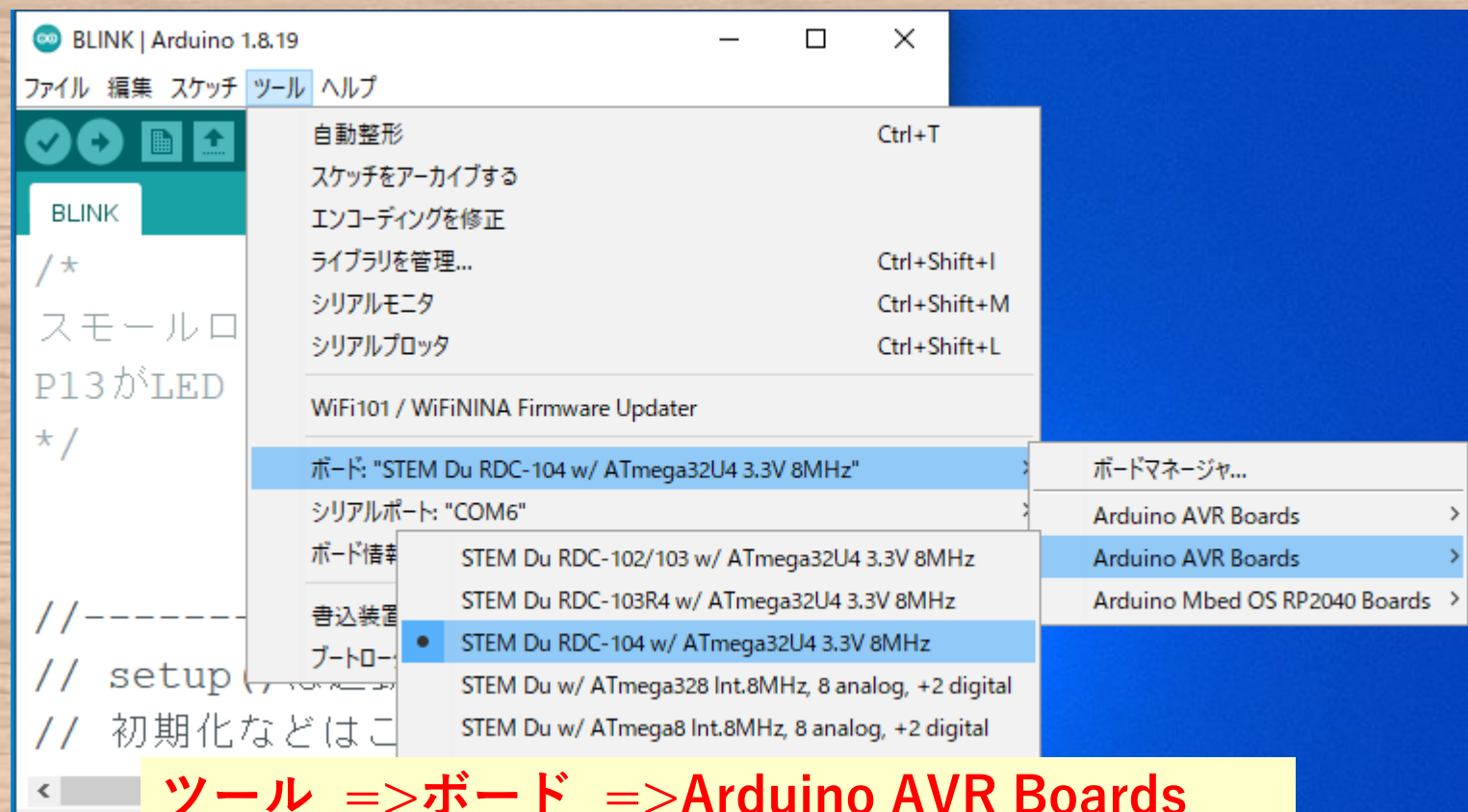
```
SmallbotV01 $  
  
pinMode(LED_PIN, OUTPUT);  
  
// モータースピード設定  
analogWrite(M1_PWM, 80); // 右モーターの速度 (0~255)  
analogWrite(M2_PWM, 120); // 左モーターの速度 (0~255)  
  
// モーター回転方向設定  
digitalWrite(M1_1, 1); // 右モーター正転  
digitalWrite(M1_2, 0);  
digitalWrite(M2_1, 1); // 右モーター正転  
digitalWrite(M2_2, 0);  
  
// 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

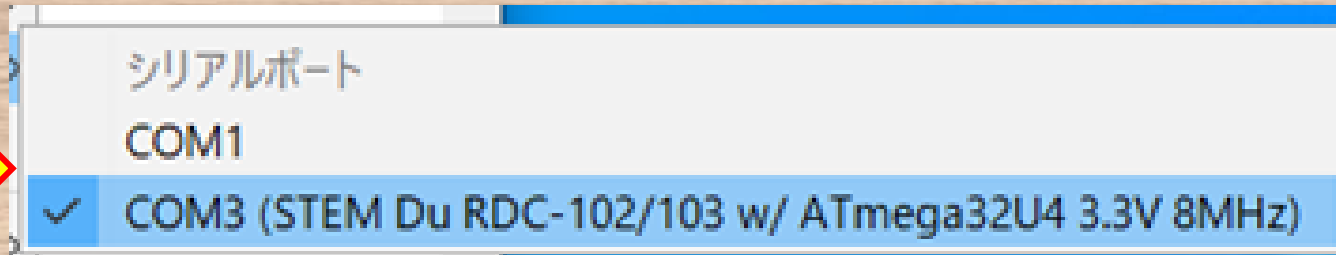


COM(シリアル) ポートの番号はPC環境によって異なる

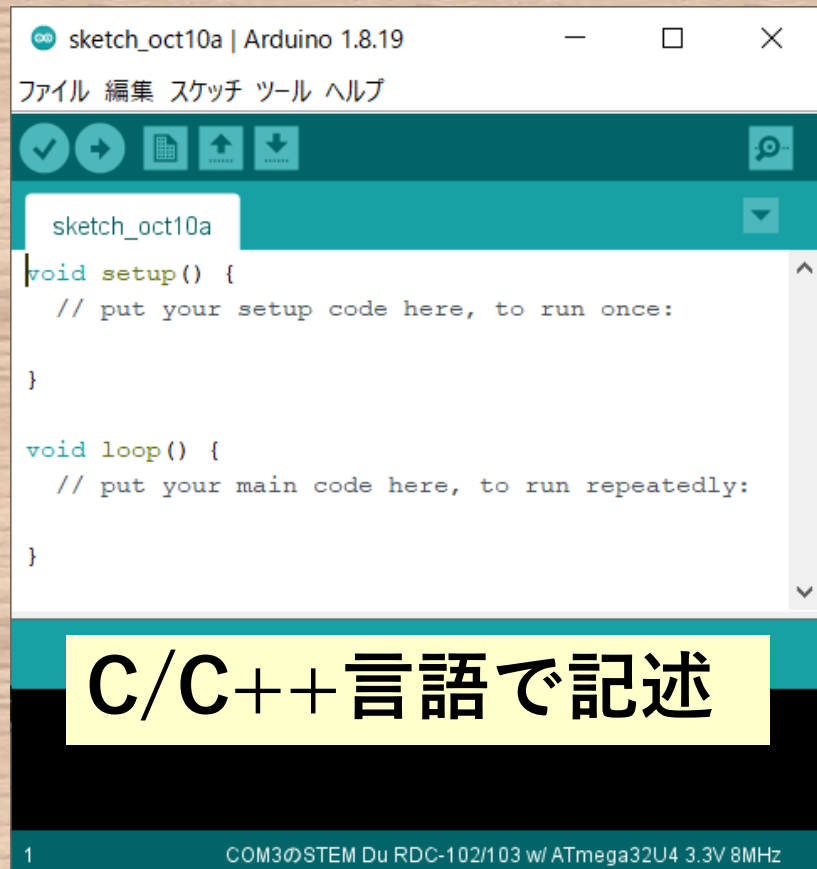
ツール

=>シリアルポート

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



プログラムの記述



```
void setup() {
```

起動時最初に
一回だけ実行される

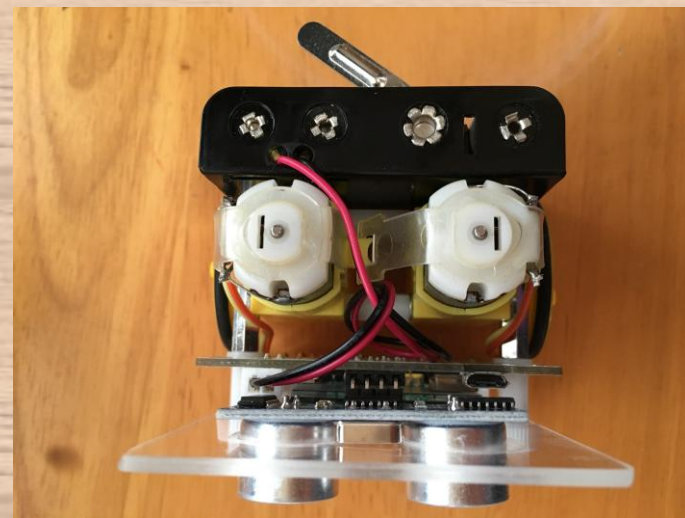
```
}
```

```
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** 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



SmallbotV00.ino

開く

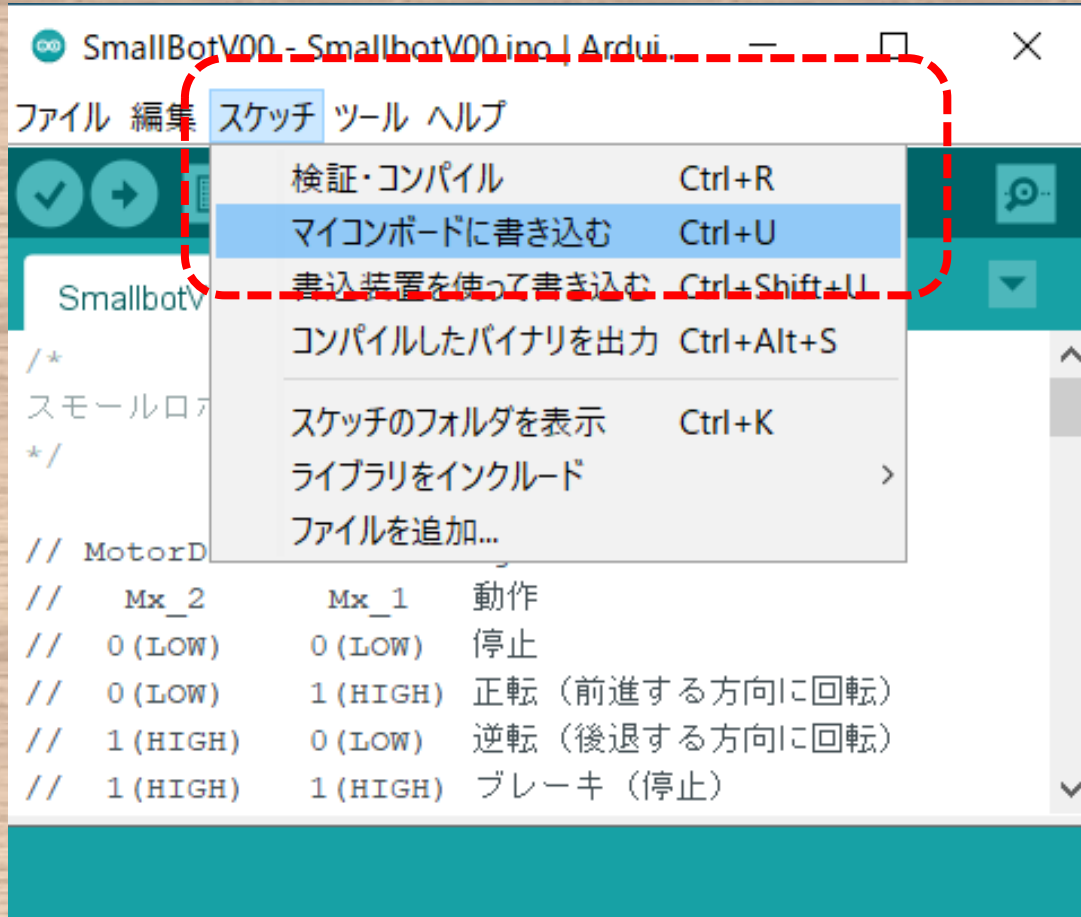
Code Blame 69 lines (56 loc) · 1.85 KB Code 55% faster with GitHub Copilot

```
1  /*
2  スモールロボット最初の一步
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
8
9
10
11
12
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
```

Arduno-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

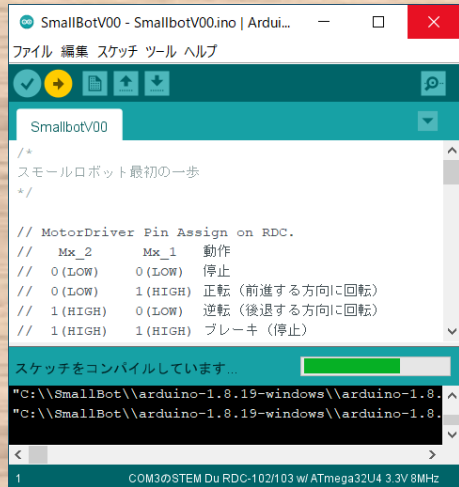


接続したSmallBotに書き込みを行う

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

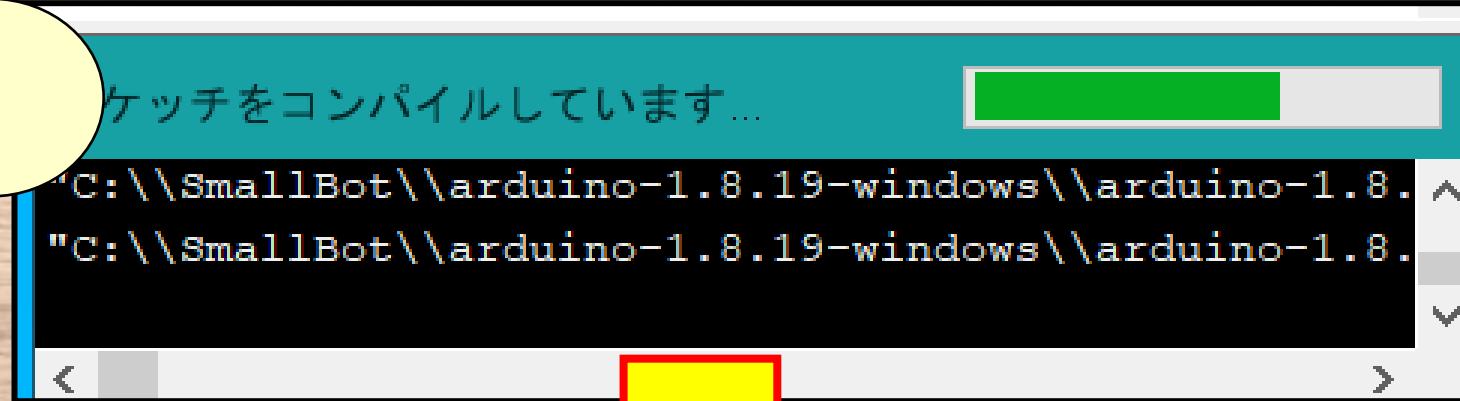
```
/*  
スモールロボット 最初の一步  
*/  
  
// MotorDriver Pin Assign on RDC.  
// Mx_2 Mx_1 動作  
// 0 (LOW) 0 (LOW) 停止  
// 0 (LOW) 1 (HIGH) 正転 (前進する方向に回転)  
// 1 (HIGH) 0 (LOW) 逆転 (後退する方向に回転)  
// 1 (HIGH) 1 (HIGH) ブレーキ (停止)
```

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.
```

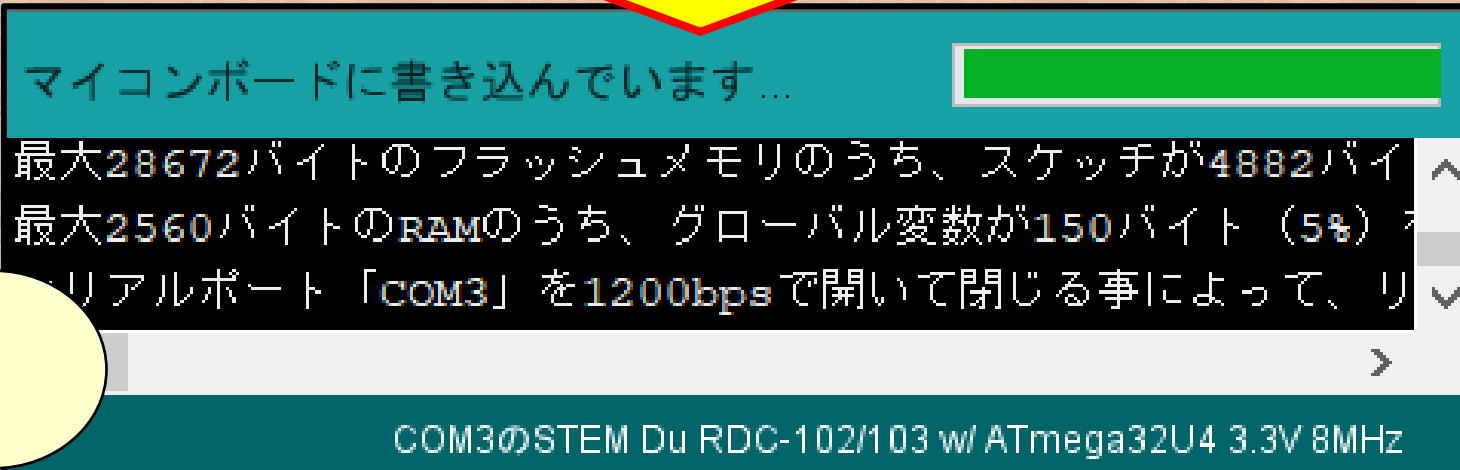
1 COM3のSTEM Du RDC-102/103 w/ ATmega32U4 3.3V 8MHz

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.
```

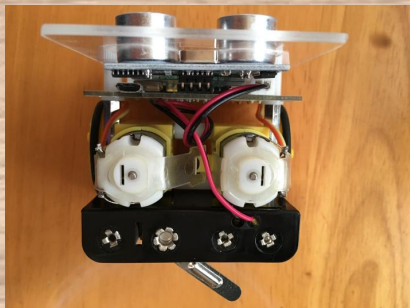


マイコンボードに書き込んでいます...

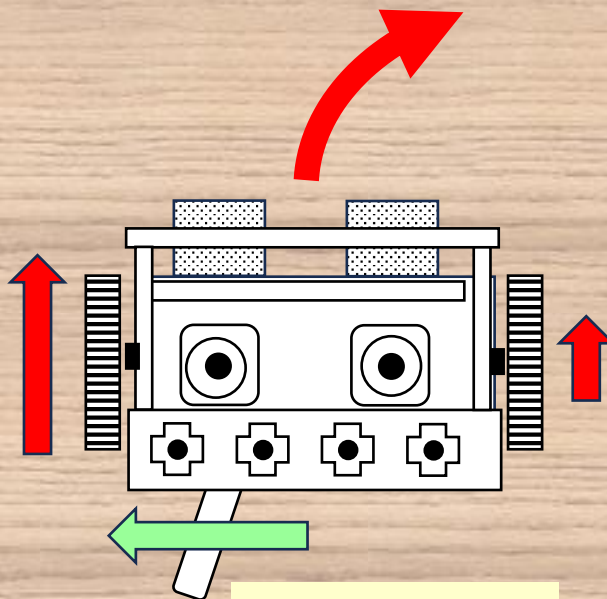
最大28672バイトのフラッシュメモリのうち、スケッチが4882バイト
最大2560バイトのRAMのうち、グローバル変数が150バイト (5%)
リアルポート「COM3」を1200bpsで開いて閉じる事によって、リ

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

時計回りに回転する



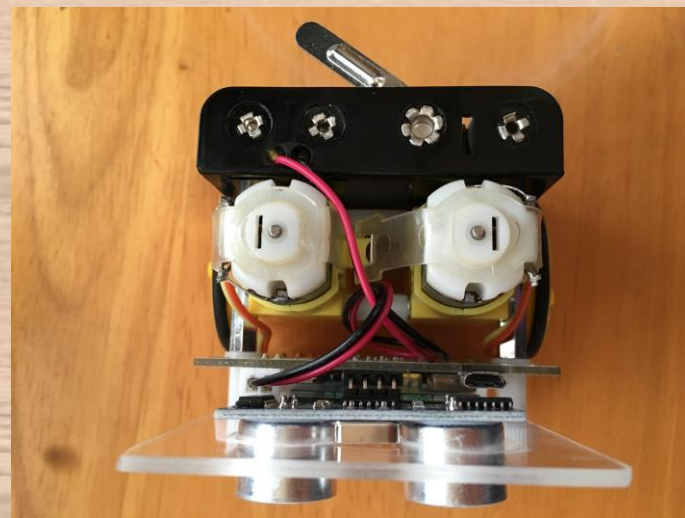
左モーター
速度 120



右モーター
速度 80

電池ON

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



開く

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

```
1  /*
2  スモールロボット最初の一步
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 //
15 #define M2_1 7
16 #define M2_2 8
17 #define M2_PWM 5
18
19 //
20 #define M2_1 7
21 #define M2_2 8
22 #define M2_PWM 5
```

Arduno-IDEにコピー
&ペーストできる

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

コピー&ペーストして動かしてみよう

```
Code Blame 69 lines (56 loc) · 1.85 KB Code 55% faster with GitHub Copilot
1  /*
2   スモールロボット最初の一步
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
```

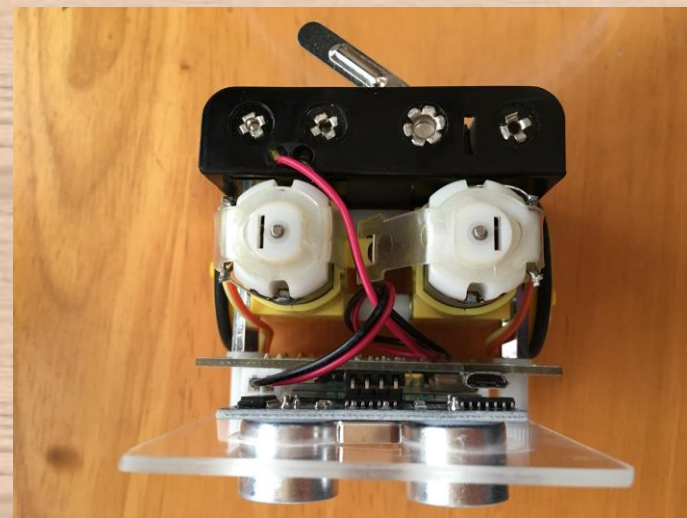
```
SmallBotV10 - SmallbotV10.ino | Arduino 1.8.19
ファイル 編集 スケッチ ツール ヘルプ
[Icons]
SmallbotV10
// 右モータ
#define M1_1    4
#define M1_2    9
#define M1_PWM  6

// 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     // スライダー
```

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
!= 等しくない

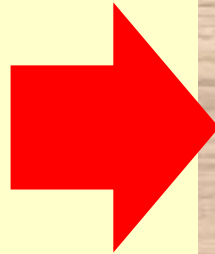
> 左辺 > 右辺
> = 左辺 ≥ 右辺

< 左辺 < 右辺
< = 左辺 ≤ 右辺

if() ... else if()... else

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

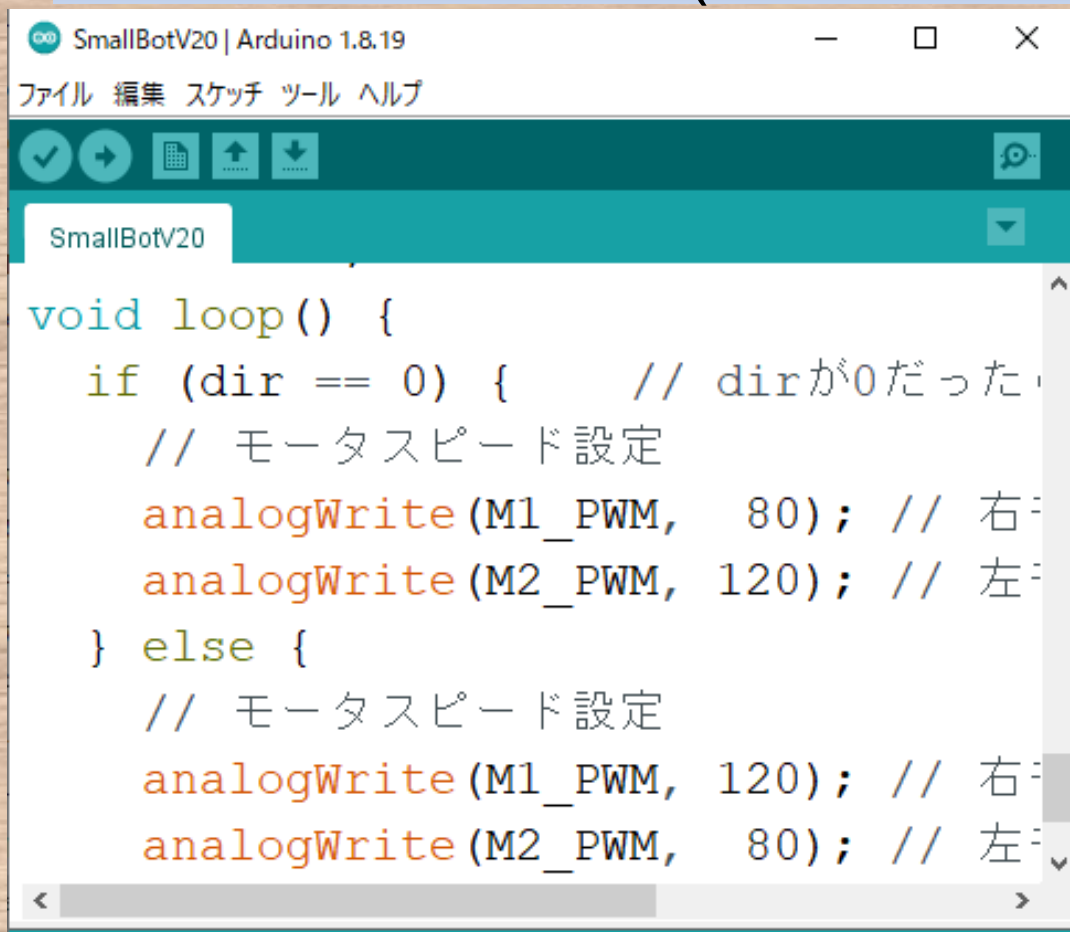
=> 3の時も3になってしまう



```
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

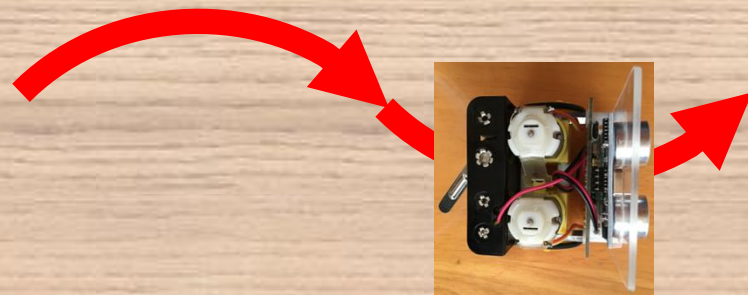
動かしてみよう (SmallbotV20.ino)



The screenshot shows the Arduino IDE interface with the file 'SmallBotV20' open. The code in the main editor area is as follows:

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

**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

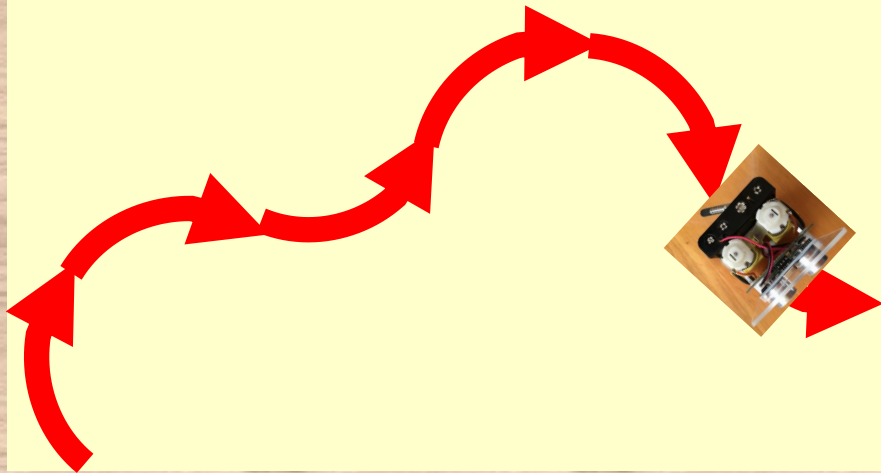
```
else {
  analogWrite(M1_PWM, 120);
  analogWrite(M2_PWM, 80);
  dir = 0; // 次は右回転
}
delay(1000); // 1秒待つ
}
```

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

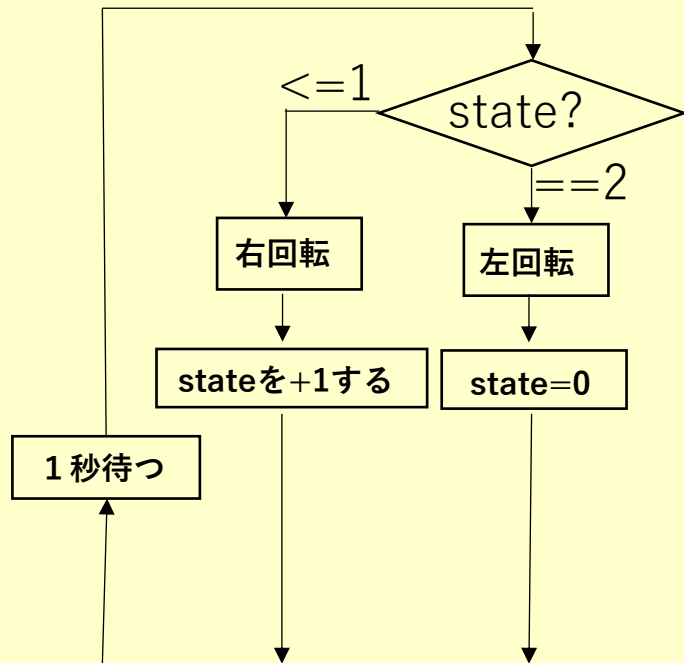
ヒント : 0 : 0,1,2

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);
}
```

A series of red arrows points from the code block towards a small electronic component, likely a motor or sensor module, which is shown in the bottom right corner.

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

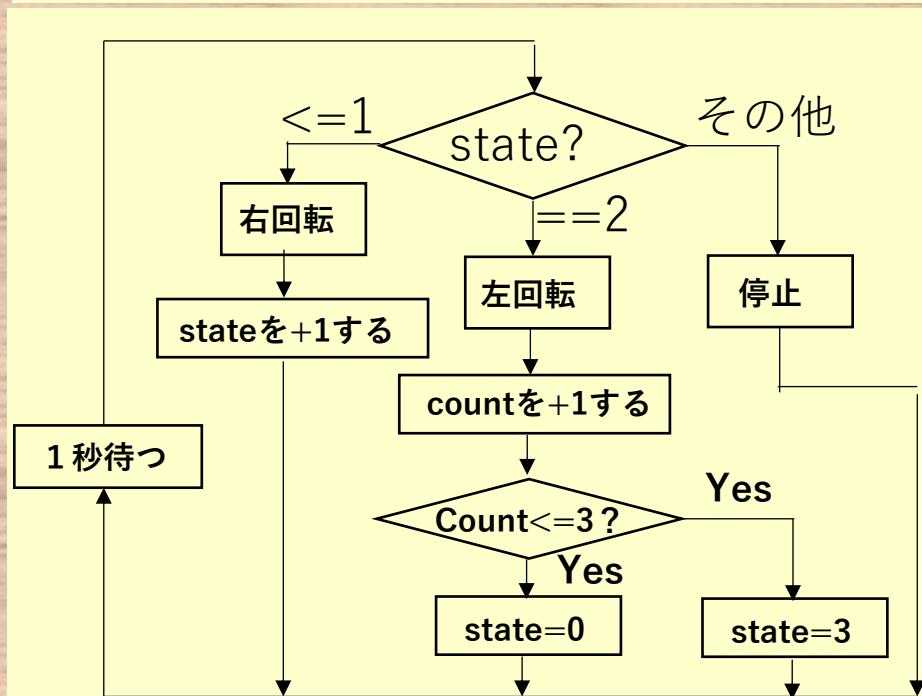
```
int state = 0;
void loop() {
  if (state <= 1) //0,1の時
    <右回転>
    state = state + 1;
}
```

```
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>
```

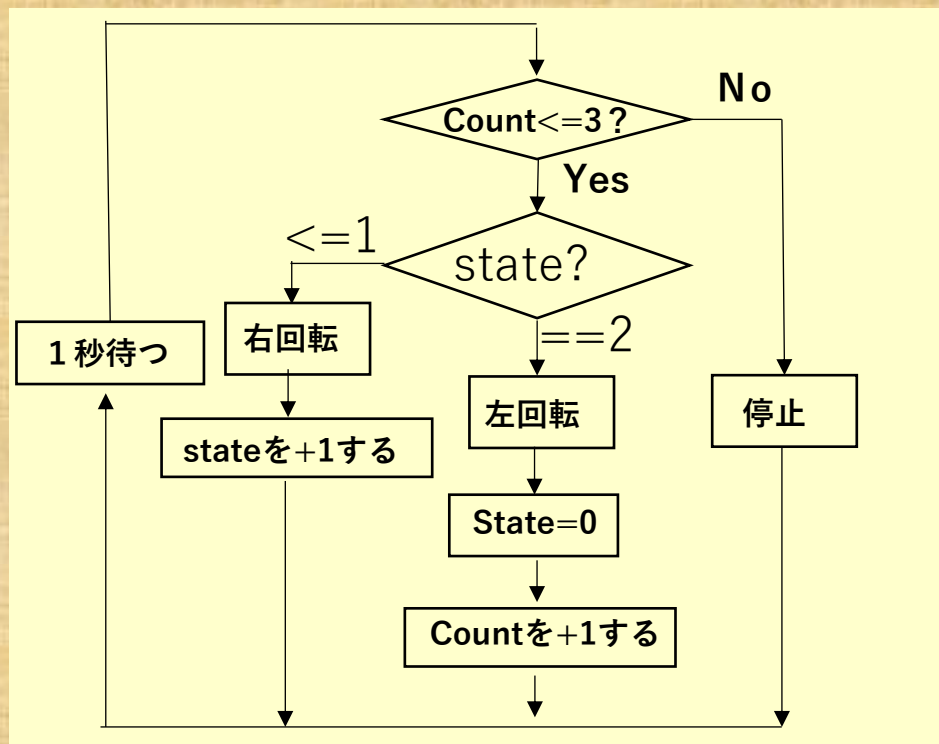
```
    }
```

```
    delay(1000); // 1秒待つ
```

```
}
```

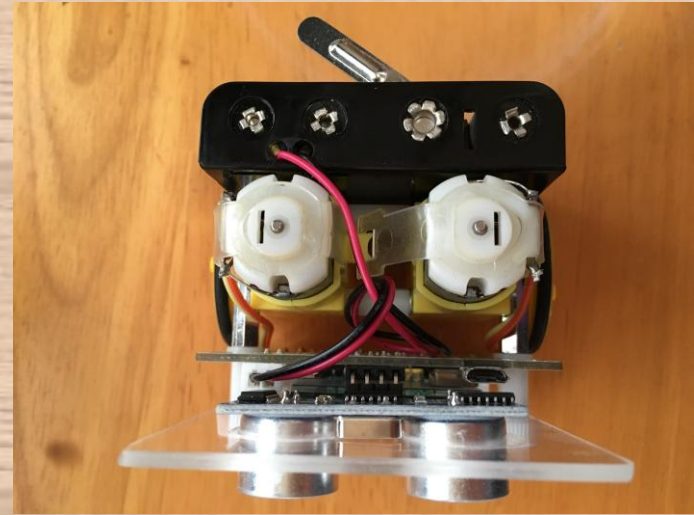
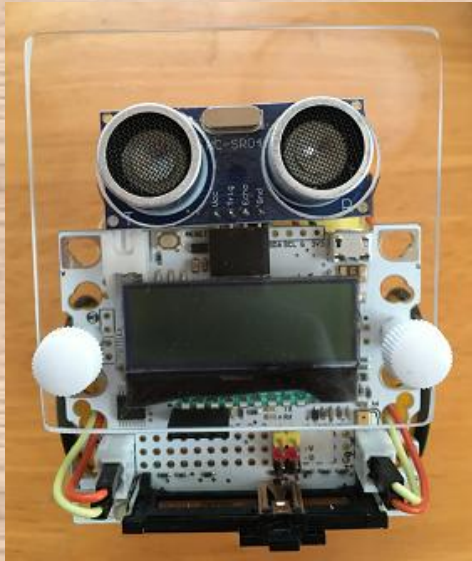
考え方の例－ 3-2

3 : 変数を二つ使って2と同じ動きにしてみよう

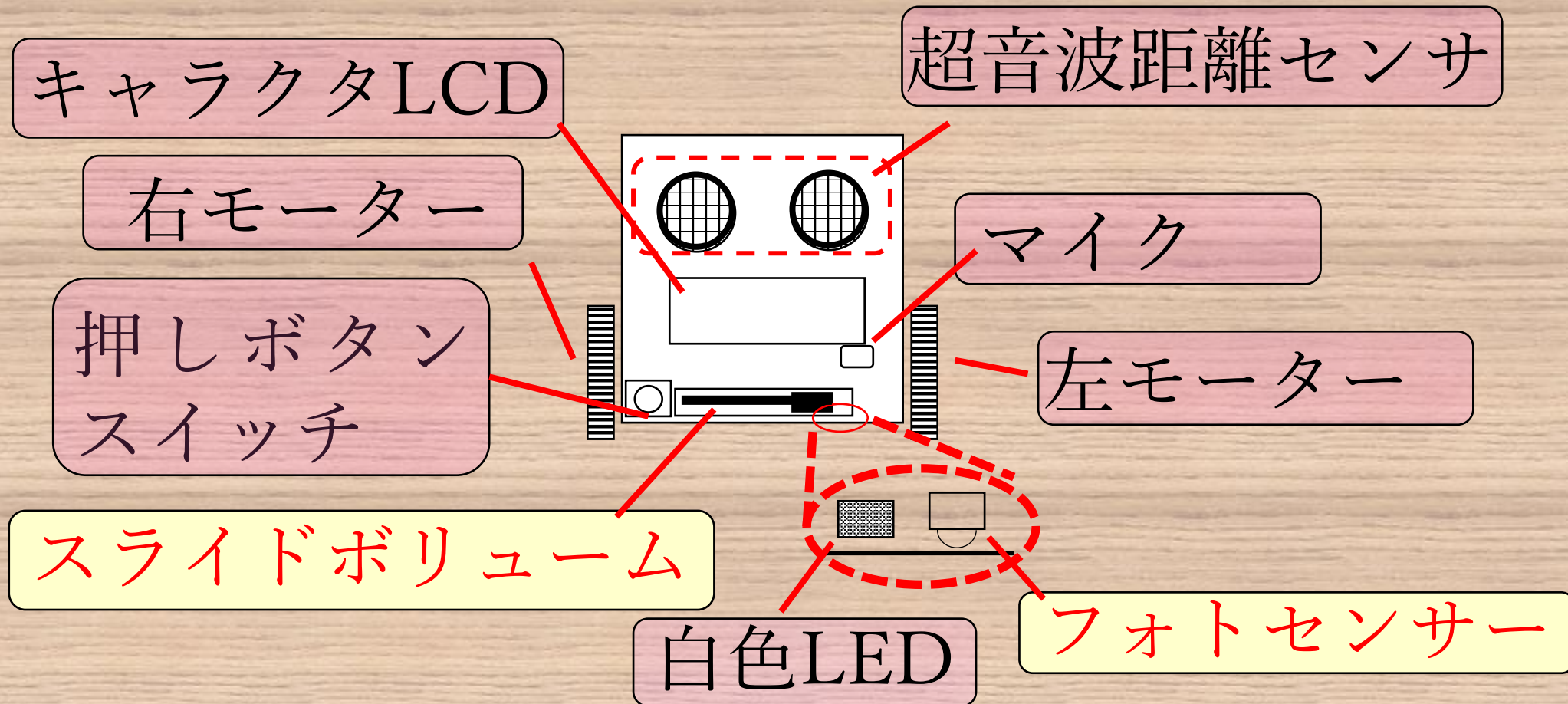


```
int state = 0;
int count = 0;
void loop() {
  if (count <= 3) { //0,1,2,3なら
    if (state <= 1) { // 0,1なら右回転
      <右回転>
      state = state + 1;
    } else if (state == 2) { //2なら左回転
      <左回転>
      state=0;
      count = count + 1; // 1 回分終わり
    }
    delay(1000);
  } else { 4回繰り返したら
    <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

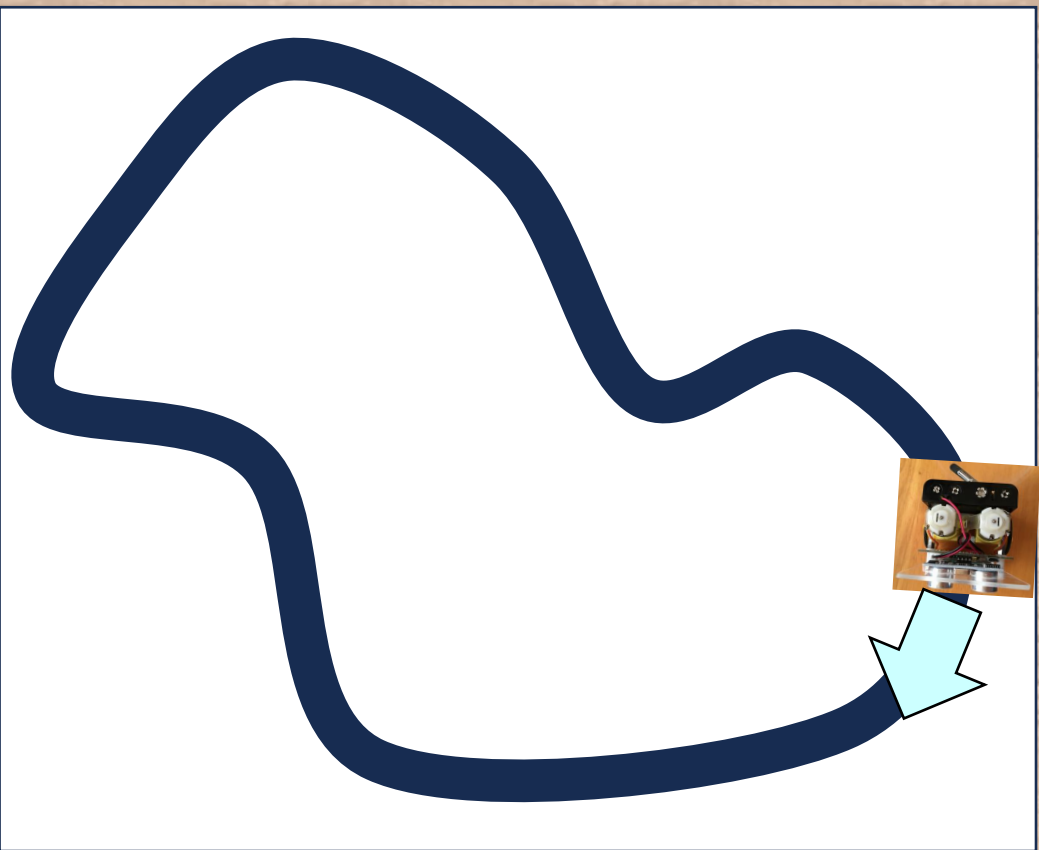
フォトセンサを使って ライントレースさせてみよう



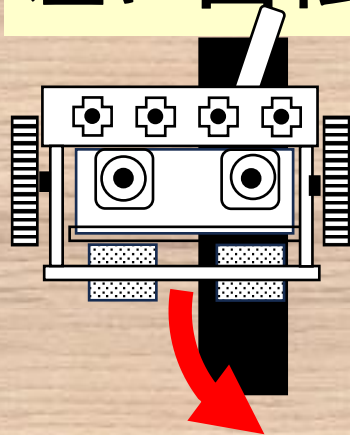
使用するセンサ類



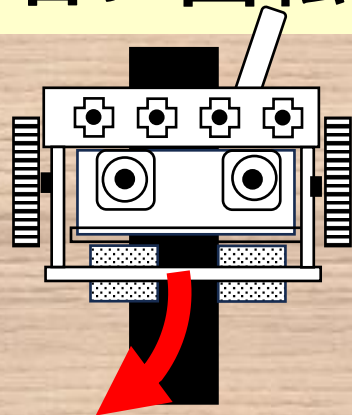
ライントレースロボット



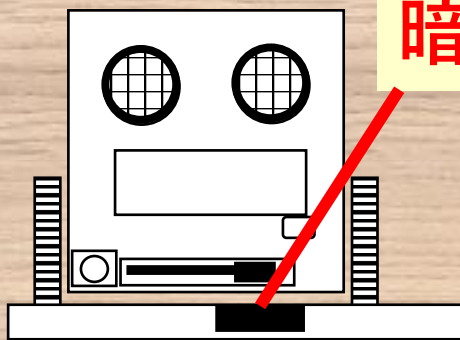
左に回転



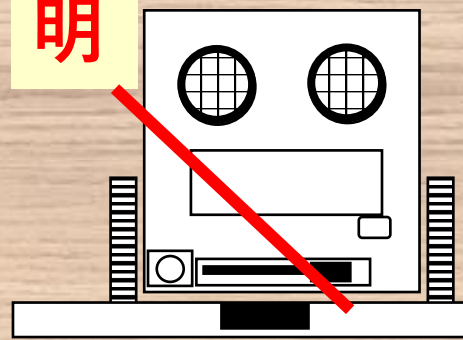
右に回転



暗

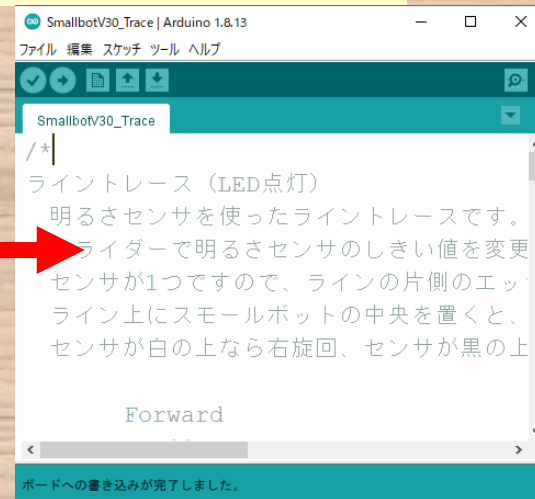
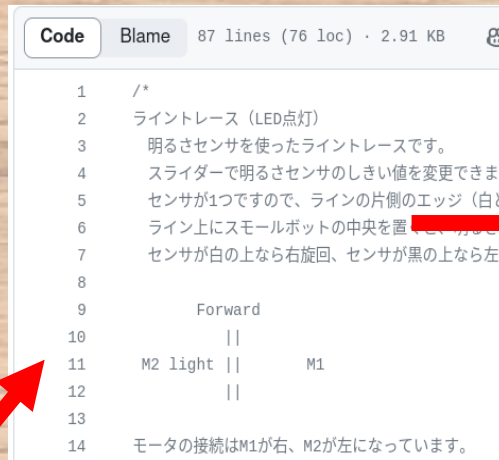
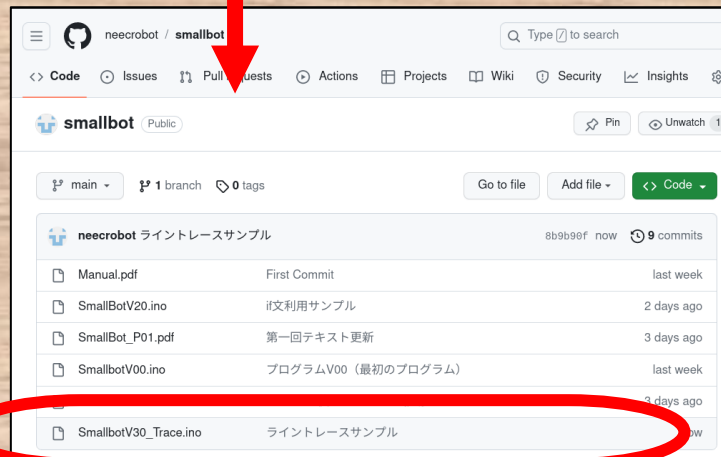


明



ライントレースサンプル (SmallbotV30_Trace.ino)

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



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

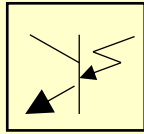
RDC-104 type II

Arduino-IDE上のピン番号

アナログ
入力

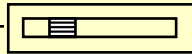
マイコン
(ATMega32U4)

A2



フォトセンサ

A3



スライドボリューム

A4



マイク (音声入力)

11

TRIG

ECHO

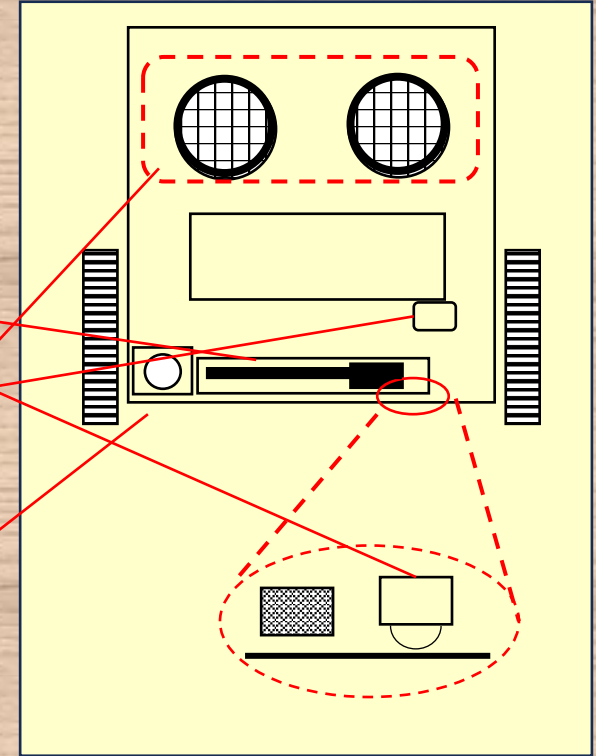


超音波距離センサ

12

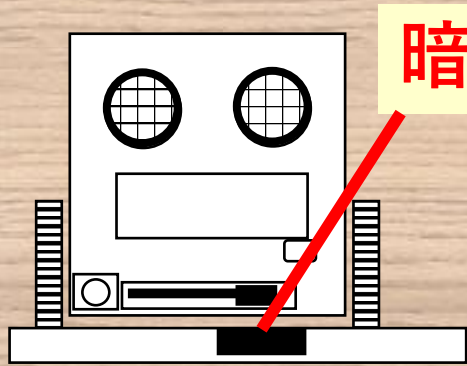


押しボタンスイッチ



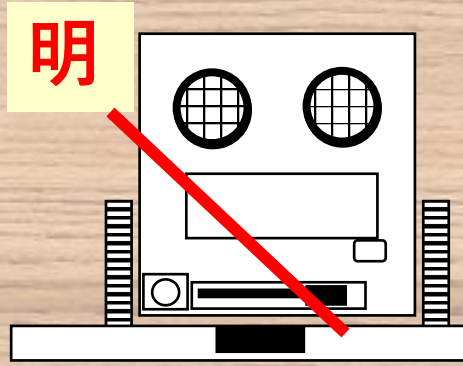
センサ類の値の取得と大小

(**analogRead()**の値は0~1023)



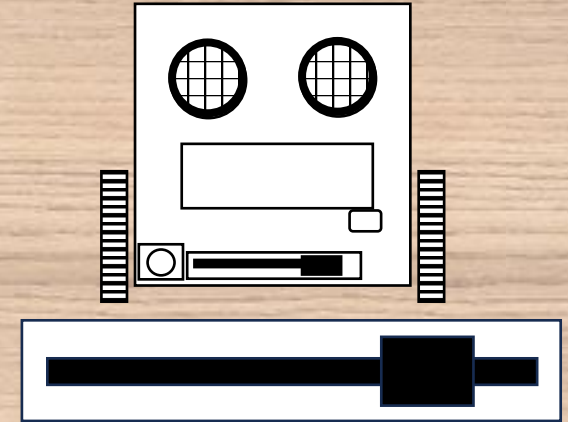
暗

フォトセンサ値小



明

フォトセンサ値大



スライダ値 小

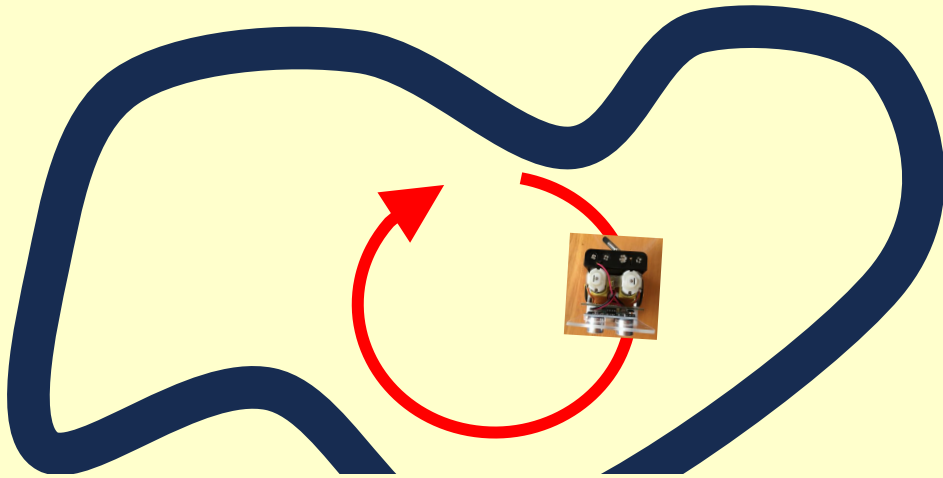


スライダ値 大

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

改造してみよう

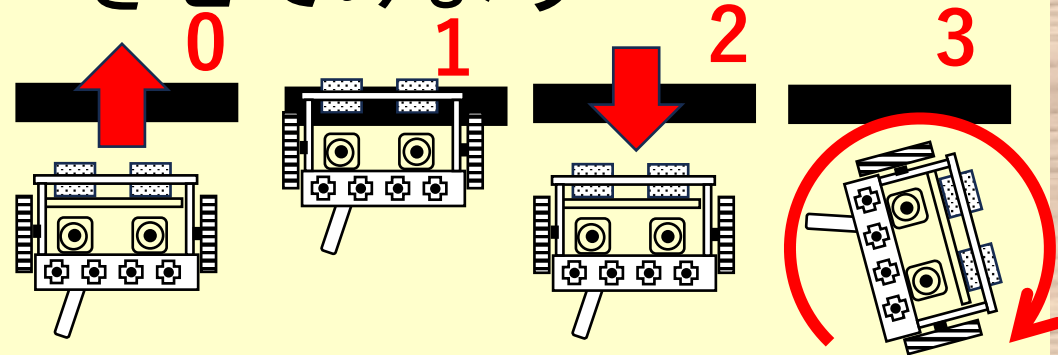
1 : 線が見つからない時
回転し続けずに線を探しに行くには？



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



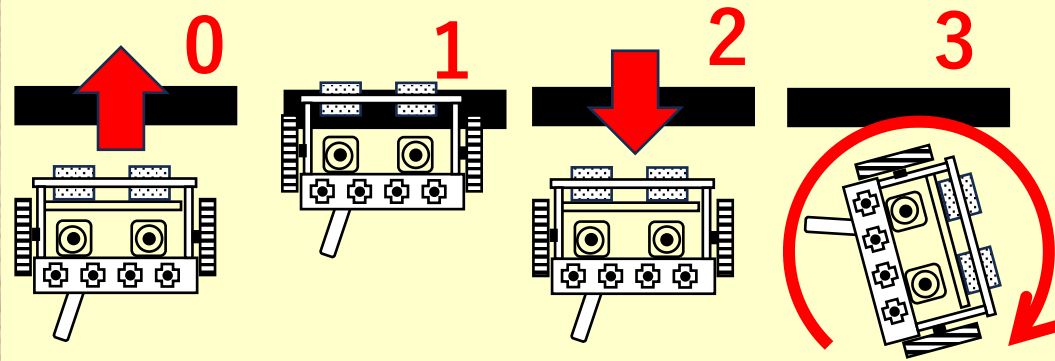
考え方の例－ 1

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
}
```

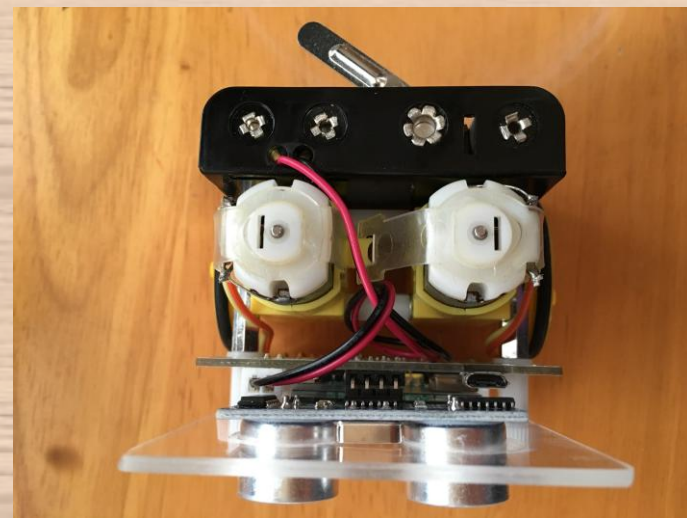
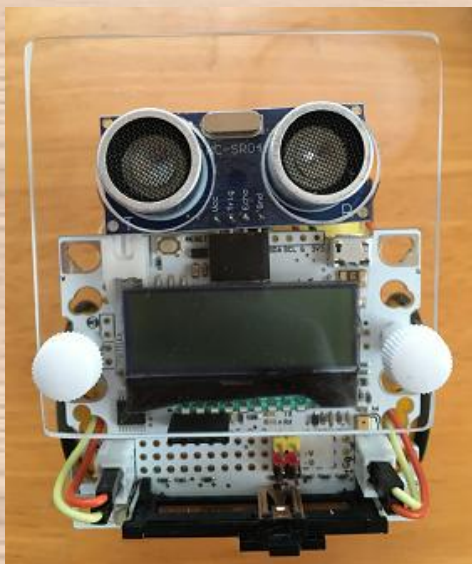
考え方の例－2



```
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