

徳島大学 大学開放実践センター 公開講座

## 無線で動くロボットを作ろう 第4回

---



徳島大学技術支援部

徳島大学社会産業理工学研究部総合技術センター

辻 明典

E-mail: [a-tsuji@is.tokushima-u.ac.jp](mailto:a-tsuji@is.tokushima-u.ac.jp)

# 講座日程

## ▶ 無線で動くロボットを作ろう

▶ 講師：辻 明典(徳島大学技術支援部)

桑折 範彦(徳島大学名誉教授)

川上 博(徳島大学名誉教授)

▶ 曜日・時間：土曜日 10時00分～11時30分

▶ スケジュール：

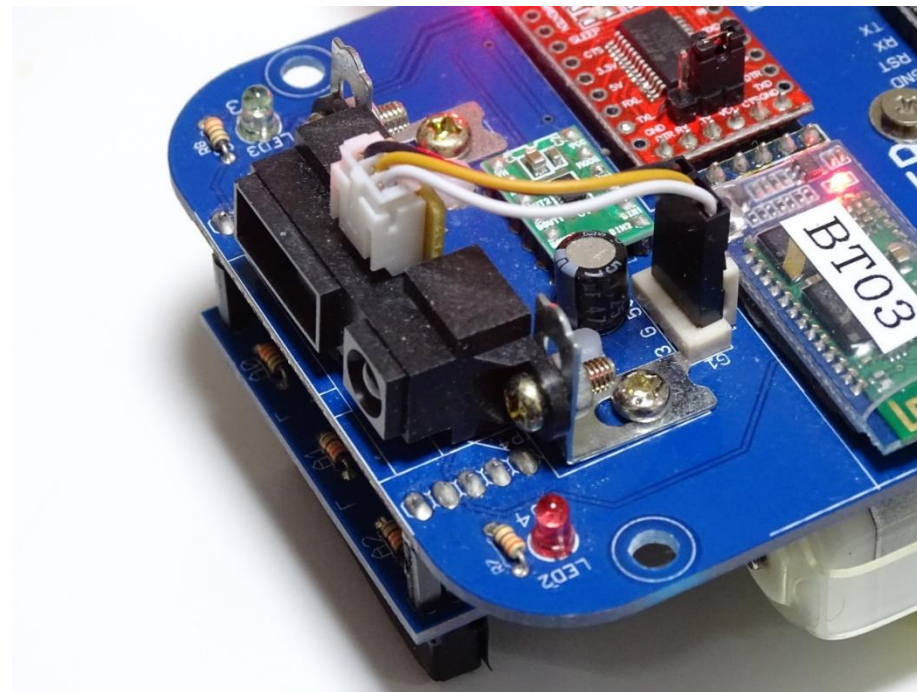
- ① 10/7 概要, ロボットの開発環境
- ② 10/14 ロボットのモーター1 (基本動作)
- ③ 10/21 ロボットのモーター2 (応用動作)
- ④ 10/28 **ロボットのセンサー1 (距離センサ, 有線・無線通信)**
- ⑤ 11/11 ロボットのセンサー2 (フォトリフレクタ)
- ⑥ 11/18 ロボットの制御1 (モータ・センサの協調動作)
- ⑦ 11/25 ロボットの制御2 (ライントレース)

# 本日の予定

- ▶ 距離センサ
  - 距離センサの取り付け
  - 距離の計測(有線)
- ▶ Bluetooth無線通信
  - 通信準備
  - 距離の計測(無線)
- ▶ 関数をつくる
  - 距離センサの例
- ▶ 距離を検出する
  - if文で判定

講座資料(スライド, サンプルスケッチ等)

<https://goo.gl/K44cPc>



# 距離センサの取り付け

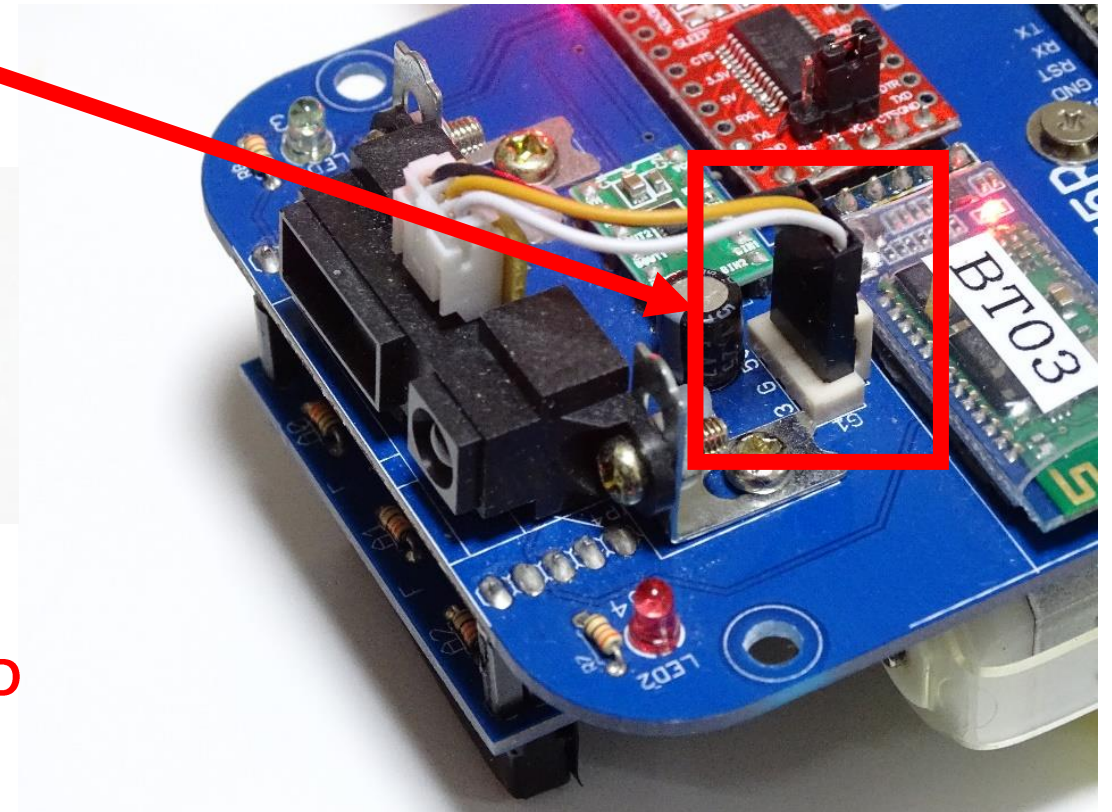
- ▶ ロボットに距離センサを取り付け
    - ・ Lアングルとネジ、ナットを使い距離センサを固定
    - ・ センサのコネクタ(3ピン)をロボットに接続
- ※ コネクタの取り付け向きに注意



距離センサ  
Lアングル x2  
ネジ x4  
ナット x4

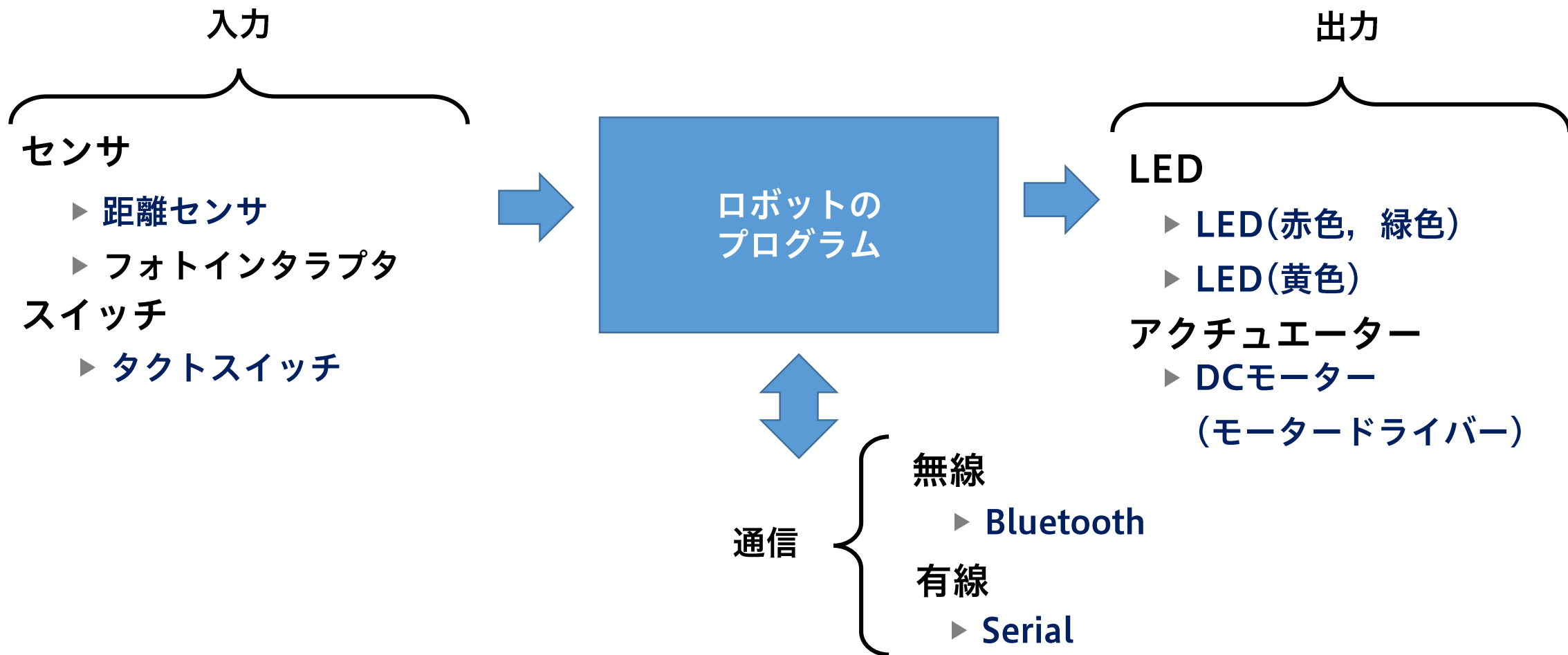


VCC (黒): 5V  
GND (橙): GND  
Vo (白): A3



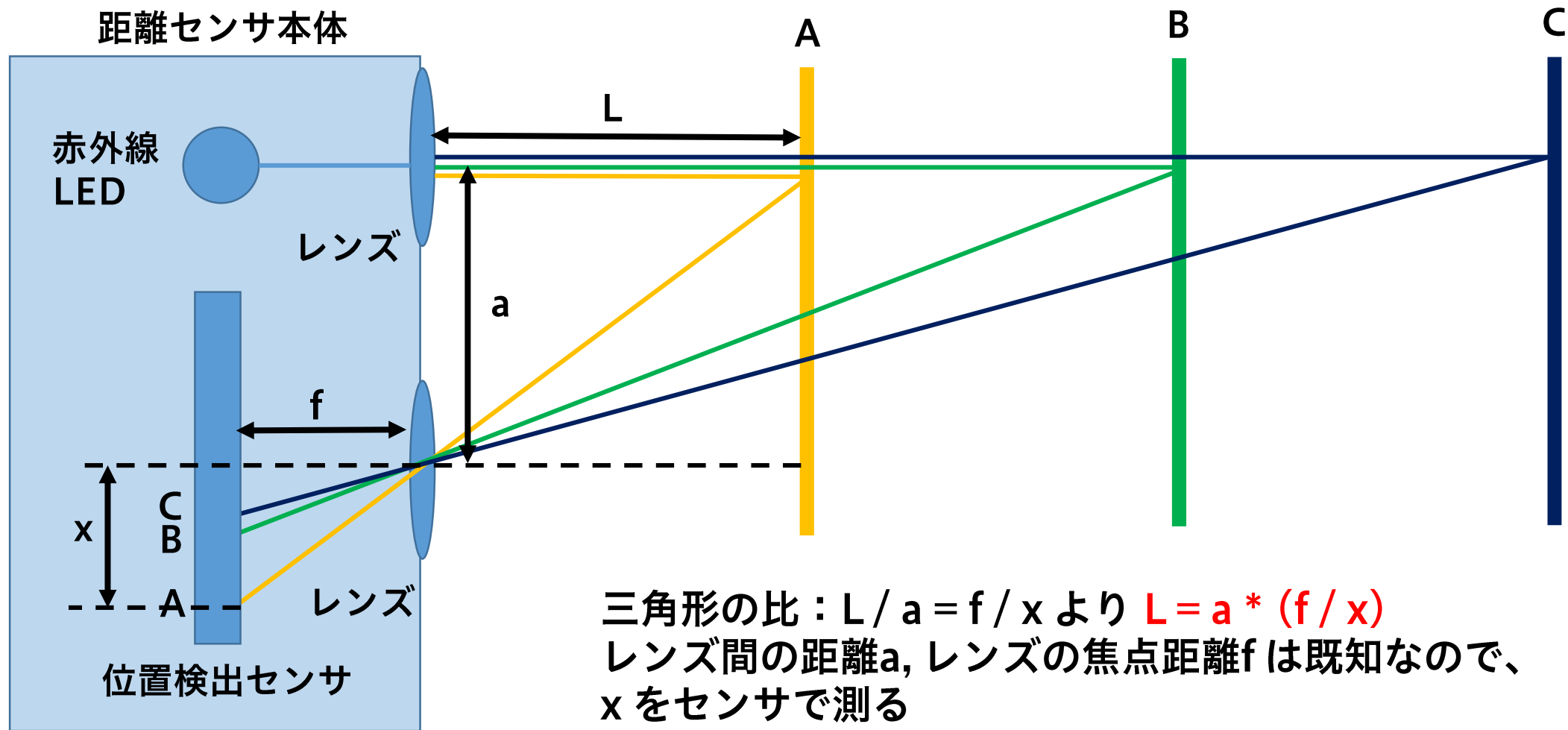
# ロボットのプログラミング

- ▶ ロボットのプログラミング：センサで周囲の状況を知り，マイコンで考え，モーターを動かし行動



# 距離センサの計測原理

- ▶ 赤外光を使い、三角測量の原理を用いて物体との距離を計測
  - Sharp, GP2Y0A21 : 計測可能な距離 10cm～80cm



# Example0401: 距離センサの動作確認

- ▶ 距離センサ(A3)の動作確認
- ▶ シリアルモニタで確認
  - 通信速度を38,400bpsに設定

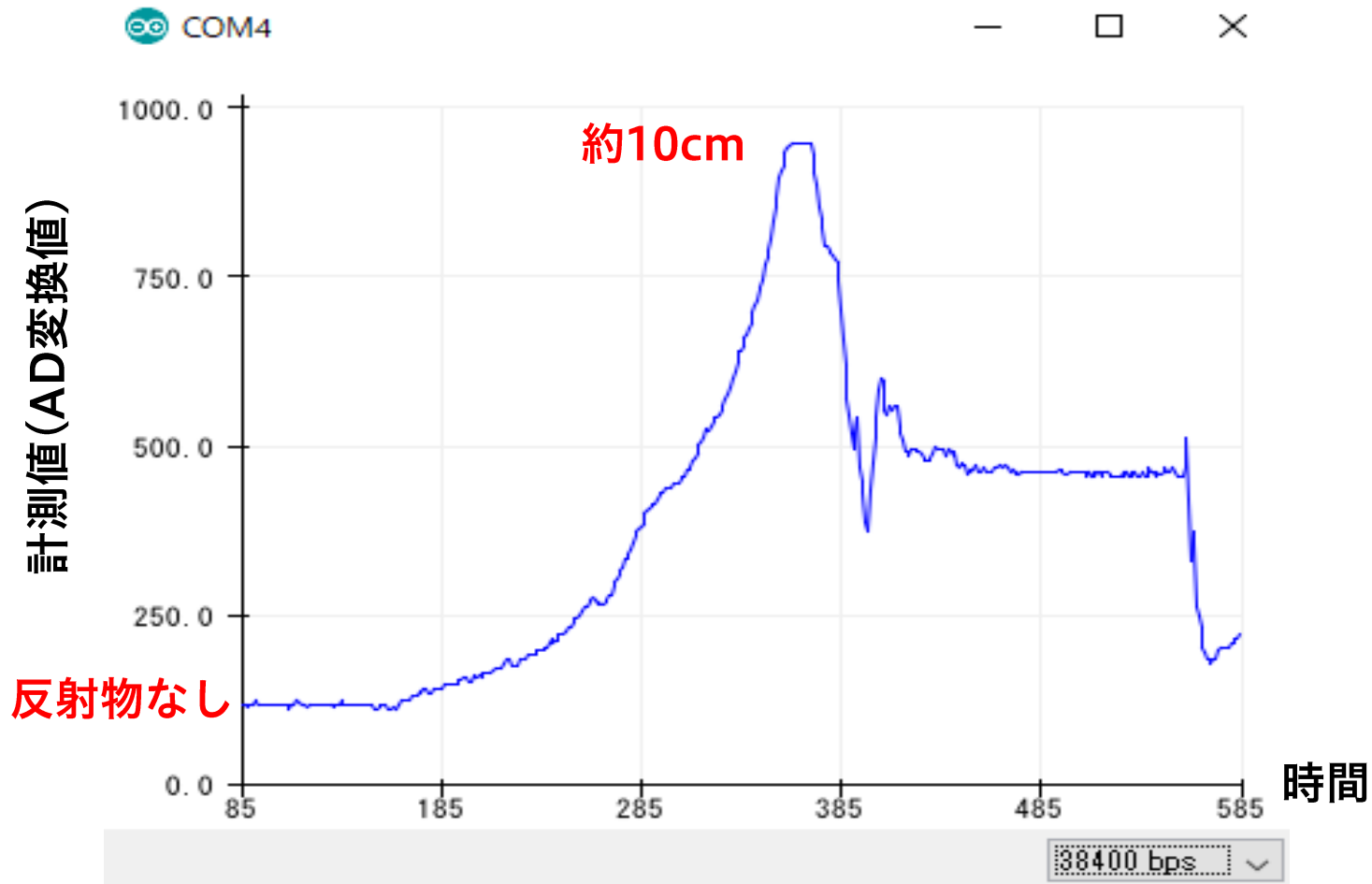
```
// Example 0401
void setup() {
  Serial.begin(38400);
  delay(1000);
}

void loop() {
  int sensorValue = analogRead(A3);
  Serial.println(sensorValue);
  delay(50);
}
```



# シリアルプロッタで確認

- ▶ 距離センサの前に反射物を置き、遠くから近くへ徐々に近づける



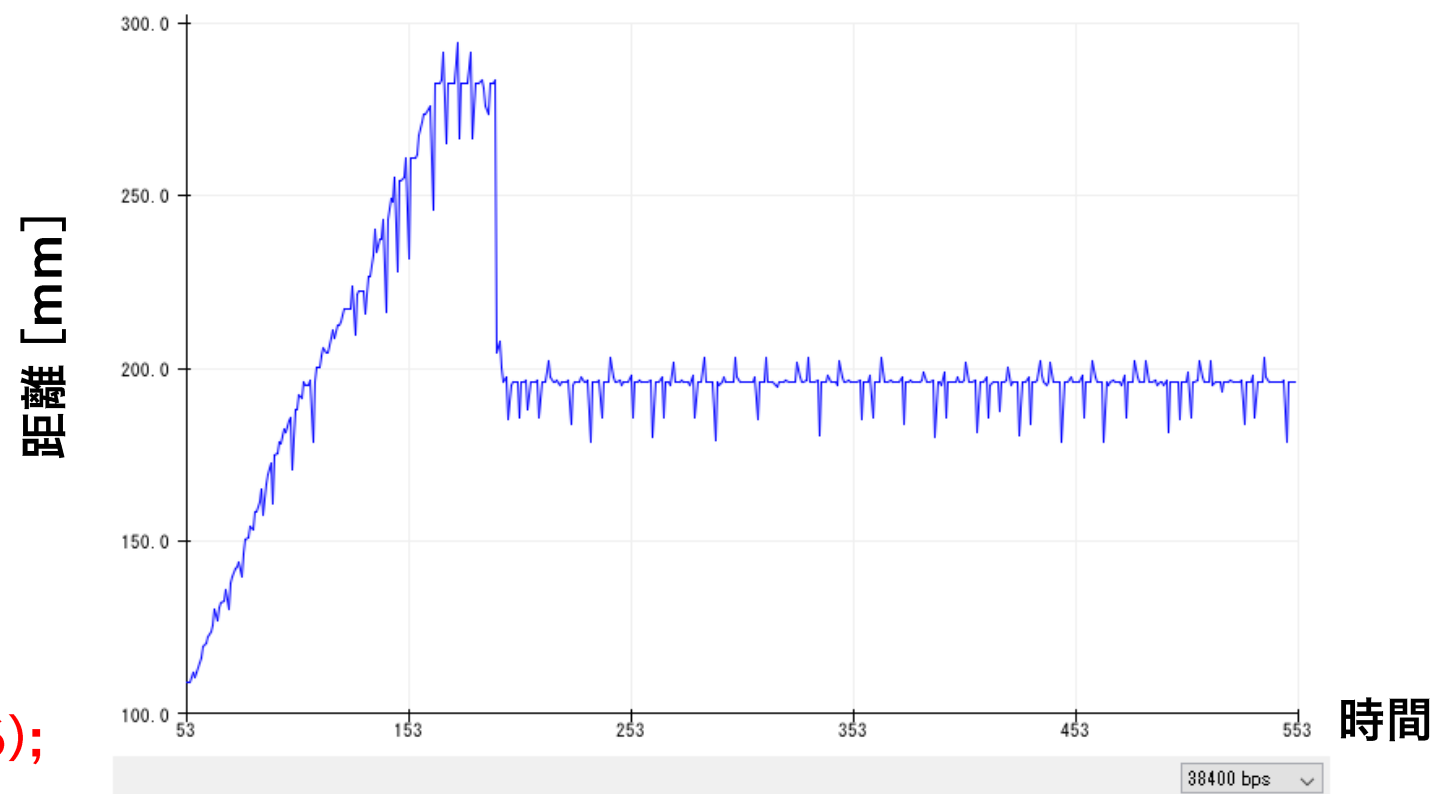


# Example0402: 距離の計測

- ▶ 計測値を距離に変換して表示
  - 距離の算出は「付録」に

```
// Example0402
void setup() {
  Serial.begin(38400);
  Serial.flush();
}

void loop() {
  int sensorValue = analogRead(A3);
  float dist = 322549.9 *
               pow(sensorValue, -1.25356);
  Serial.println(dist);
  delay(50);
}
```



# Bluetooth無線通信

- ▶ Bluetooth規格: 近距離無線通信規格、2.4 GHz帯、数m～数十m
    - 用途：無線マウス、オーディオ、スマートフォンなど
  - ▶ ロボットの通信に利用
    - Bluetoothバージョン2.0準拠、SPP(シリアルポートプロファイル)
- 有線通信を無線通信に置き換え  
(無線でプログラムの書き換えはできません)



USBケーブル

有線シリアル通信



Bluetooth親機

Bluetooth子機

無線シリアル通信

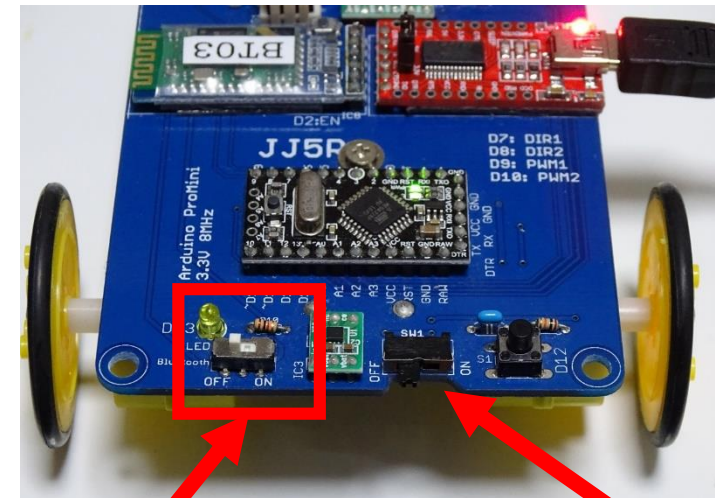
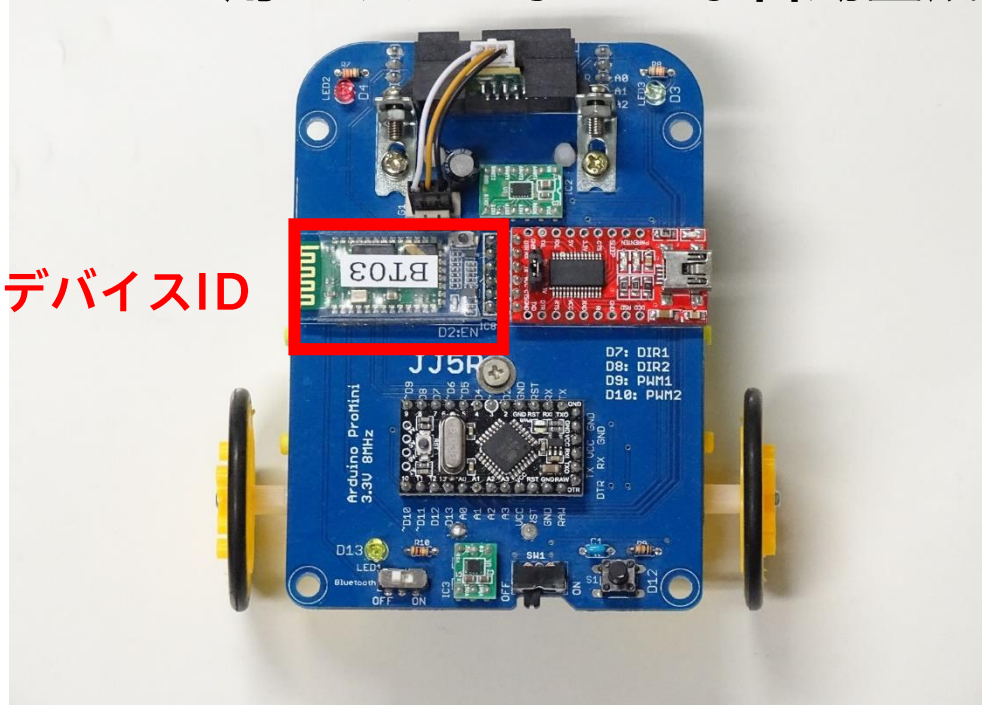
# Bluetoothのペアリング

- ▶ 別紙を参考に設定を行う
- ▶ **BluetoothデバイスID (BT0x)の確認**
- ▶ **BluetoothスイッチをON**にして電源スイッチを入れる
- ▶ 親機と子機をペアリング、PINコード：**1234**
- ▶ Bluetooth用シリアルポートが自動生成(発信と着信の2ポートあり、発信を利用)

USBのシリアルポート：  
Bluetoothのシリアルポート：

各自の使用するポートを記入

デバイスID



Bluetoothスイッチ

電源スイッチ

# Example0402: 無線シリアル通信の確認

## ▶ プログラム書き込み(有線のみ)

- ロボットにUSBケーブルを接続
- Arduino: シリアルポート(USB)を選択
- プログラム(Example0401)を書き込む(**Bluetoothスイッチは必ずOFF**)
- USBケーブルを抜く

## ▶ 無線通信

- BluetoothスイッチをON
- ロボットの電源を入れる
- Arduino: シリアルポート(Bluetooth)を選択
- シリアルモニタまたはプロッタを起動

※ プログラムを書き込む時、Bluetoothスイッチは必ずOFF

※ 有線・無線の切り替え時：Arduinoのシリアルポート選択を変更

# スケッチの書き込みエラー

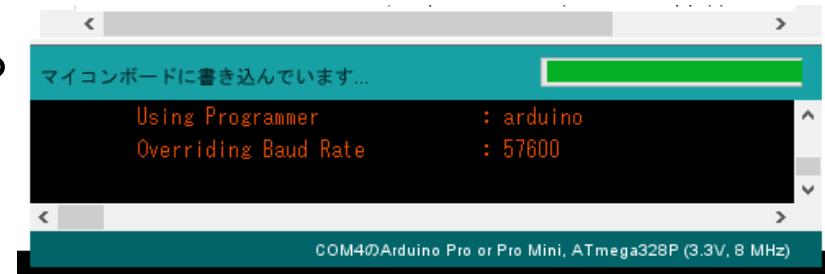
- ▶ USBシリアルポートとBluetoothポートを切り替えると、
  - Bluetoothのスイッチの入れ忘れ、切り忘れ
  - USBケーブルの接続忘れなどにより、シリアルポートが正しく選択されておらず、スケッチの書き込み中にエラーが発生することがよくある。
  
- ▶ **スケッチの書き込み中にエラーが発生した場合の対処**
  1. ArduinoのシリアルポートCOMx(USBポート)を確認
  2. Bluetoothスイッチの状態OFFを確認
  3. USBケーブルを差し直す（ロボットの電源入れ直し）
  4. Arduinoを終了して、再起動

# スケッチの書き込みエラー(よくあるケース)

## ▶ BluetoothスイッチをONのまま、書き込みをした。

- ・ マイコンボードに書き込んでいます。のまま応答なし。

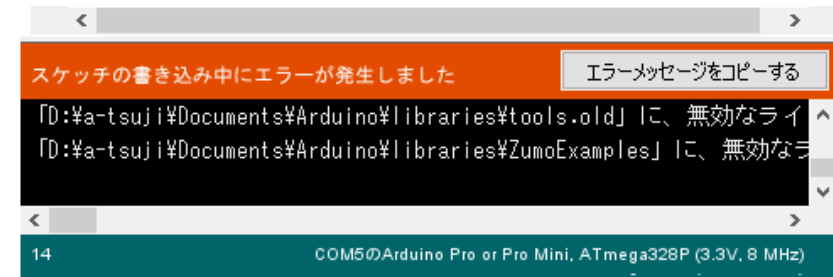
→ Bluetooth SWをOFFにして、再度、書き込み



## ▶ シリアルポートをUSBに変更せずに、書き込みをした。

- ・ スケッチの書き込みエラーが発生しました。

→ シリアルポートをUSBに設定して、再度、書き込み

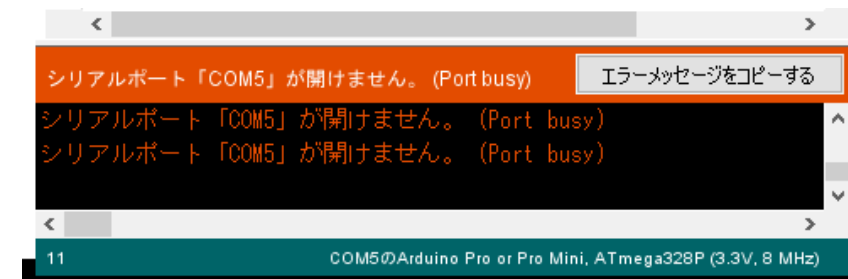


## ▶ Bluetoothシリアルポートに設定、通信中に書き込みをした。

- ・ シリアルポート(COMx)が開けません。(Port busy)

→ Arduinoを閉じて再起動

→ シリアルポートをUSBに設定



# Example0403: 関数を作る

## ▶ 距離を計測するコード

```
int sensorValue = analogRead(A3);  
float dist = 322549.9 *  
            pow(sensorValue, -1.25356);
```

## ▶ 距離の計測をする関数

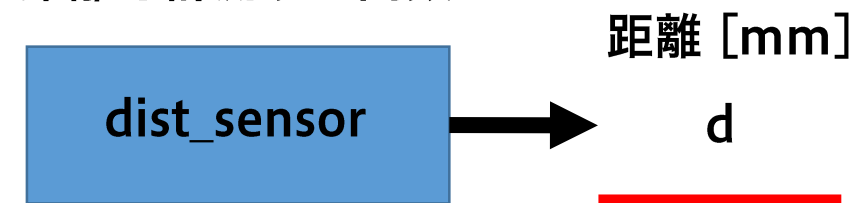
```
float d = dist_sensor();
```

### • dist\_sensor 関数 :

```
float dist_sensor() {  
    int sensorValue = analogRead(A3);  
    float dist = 322549.9 *  
                pow(sensorValue, -1.25356);  
    return dist;  
}
```

関数 → 距離だけ、わかればいい

距離を計測する関数



```
int sensorValue = analogRead(A3);  
float dist = 322549.9 *  
            pow(sensorValue, -1.25356);  
return dist;
```

# Example0404: 距離を検出する

- ▶ ロボットがある範囲の距離に入ったときにLEDを点灯



- ▶ if文を使う

- 指定した条件にあうかどうか判定
- 条件 :  $<$ ,  $>$ ,  $\leq$ ,  $\geq$ ,  $\&\&$ ,  $\|\|$ ,  $\text{==}$ ,  $\text{!=}$

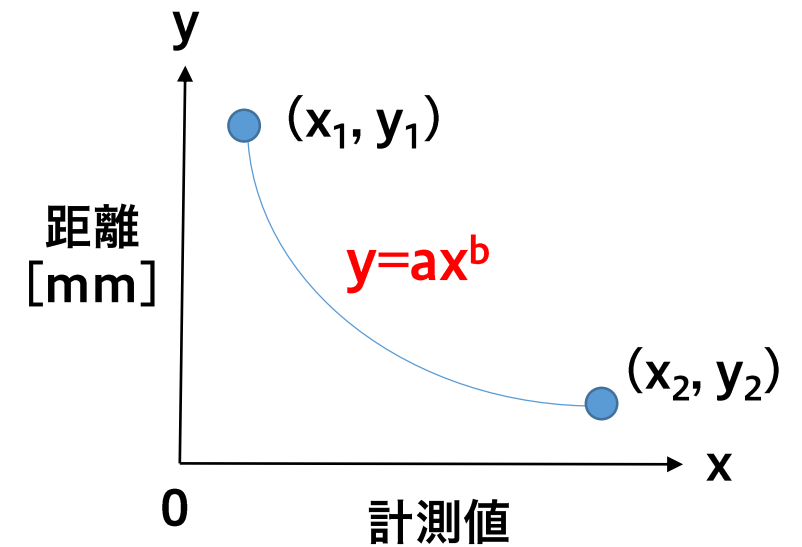
$x < a$	xはaより小さい(未満)	$x == a$	xはaと等しい	$x \geq a \ \&\& \ x < b$	xがa以上で、かつb未満 (両方の条件を満たす)
$x \leq a$	xはa以下(a含む)	$x != a$	xはaと等しくない	$x \geq a \ \ \  \ x < b$	xはa以上、または、b未満 (どちらか一方を満たす)
$x > a$	xはaより大きい				
$x \geq a$	xはa以上(a含む)				

- 例 : `if (x < 100) { . . . } // xが100未満`  
`else { . . . } // xは100未満以外`



## 付録: 距離特性の近似曲線

- ▶ 距離センサの計測結果の近似曲線を定義し、その係数を求めておき、計測値 $x$ がわかると距離 $y$ が求まる
- ▶ 近似曲線  $y=ax^b \cdots (1)$ 
  - 2点  $(x_1, y_1)$ ,  $(x_2, y_2)$  を通る曲線
  - $x_1, x_2$  は計測値 (0-1023)
  - $y_1, y_2$  は実際の計測距離 (mm)
- ▶  $(x_1, y_1)$ ,  $(x_2, y_2)$  を各々 (1) に代入し、 $b$  を求める。  
 $y_1 = ax_1^b$ ,  $y_2 = ax_2^b$  より、  
 $a = y_1 / x_1^b$ ,  $a = y_2 / x_2^b$  となり、  
 $y_1 / x_1^b = y_2 / x_2^b$  左辺、右辺を入れ替え  
 $y_1 / y_2 = x_1^b / x_2^b$  両辺 log をとる  
 $\log(y_1 / y_2) = b \log(x_1 / x_2)$
- ▶  $b = \log(y_1 / y_2) / \log(x_1 / x_2)$



$a$  は、(1) より、  
 $a = y / x^b$   
ここに  $(x_1, y_1)$  を代入して求める  
 $a = y_1 / x_1^b$

```
float dist = a * pow(sensorValue, b);
```