ネットワーク分野　研究室ローテーション

# Raspberry PiとProcessingによるIoTデバイス作成演習

## 第２回
## Processing入門・ディジタル入出力
## アナログーディジタル変換とセンサ

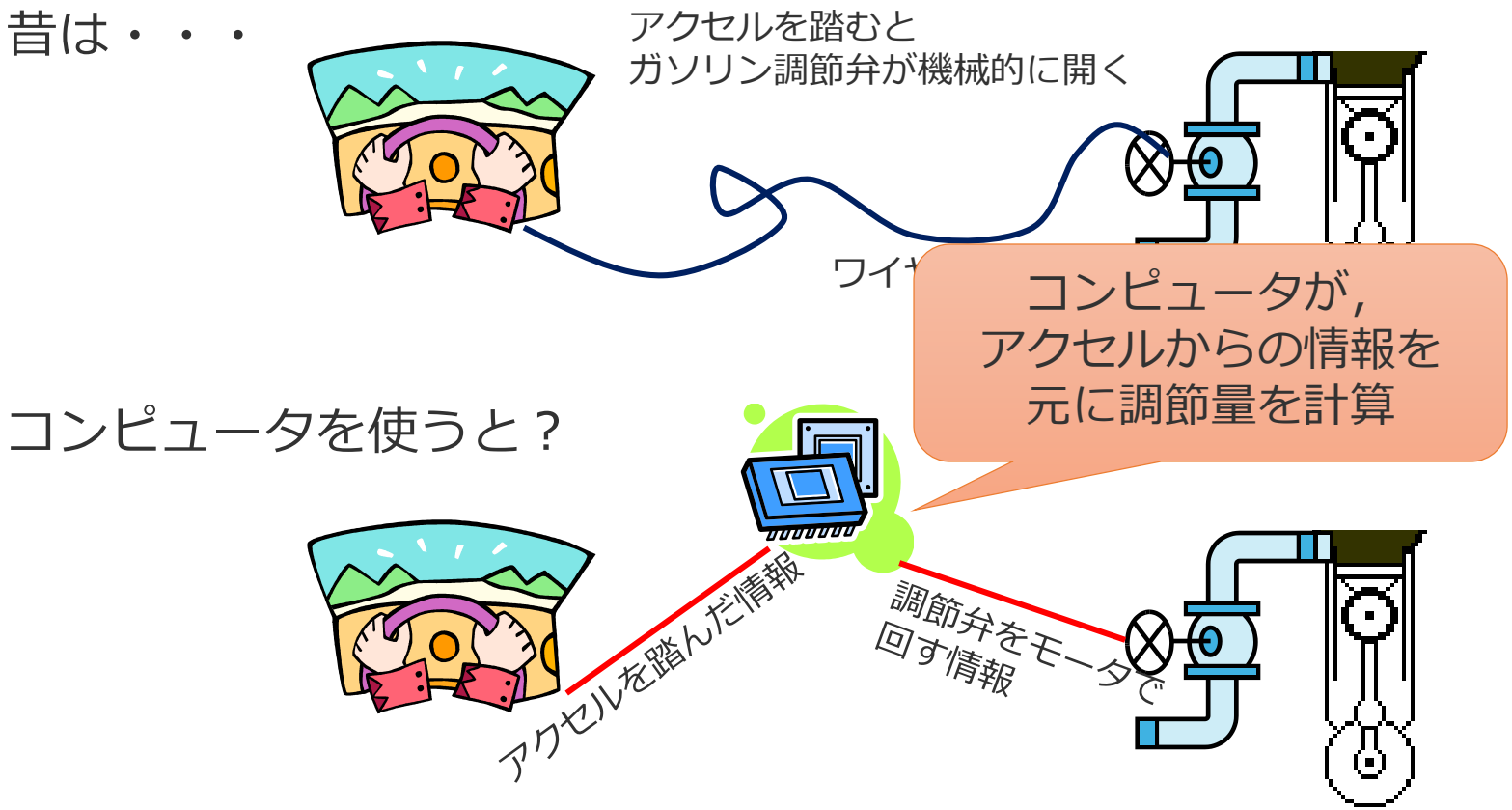
編集履歴
2016/12/12 福嶋
2017/04/18 前田・福嶋
2018/05/31 前田

# はじめに(1/3)

- 身の周りにはコンピュータがたくさん
  - –パソコン・携帯・テレビ・自動車...
- なぜ何でもコンピュータが使われる？

# はじめに(2/3)

- コンピュータは制御の頭脳
  - 例：自動車の制御



# はじめに(3/3)

- コンピュータによる制御
  - 実際には，コンピュータ上で動く「プログラム（ソフトウェア）」が制御している
    - プログラム(program) : 「手順書き」の意味

プログラムを変えれば制御方式も変えられるし，
違う機器だって制御できる！

プログラムこそがコンピュータ制御の本体だ！

# 演習全体の流れ



- 第２回
  - Processing入門
  - ディジタル入出力
  - アナログ入力とセンサ
  - ネットワーク
- 第３回
  - 画像処理
  - 音声出力
- 第４回
  - 自習
- 第５回
  - 発表

# プログラミング：Processing

# プログラミングとは



- コンピュータに指示を与えるための命令書を作ること

- プログラミング言語
  - 手順をコンピュータに教える言語
  - 例：C言語，Java

今回はProcessingというプログラミング言語を使う

# プログラムの構造



- 構造
  - 変数
    - 値を記憶しておくところ
  - 順次実行
    - 上から順番に実行する
  - 条件分岐
    - 「もしAならばBそうでないならC」を実行する
  - 繰り返し
    - 同じ処理を繰り返し実行する
  - 関数
    - 命令をまとめたもの

# 変数



- プログラムで値を記憶しておくところ
  - 変数には型が存在
  - 型で決められた値を記憶しておくことが可能
- 型
  - int：整数
    - Int a = -1;
    - Int a = １; //全角はだめ
  - float：実数
    - Float b = -1.2;
  - String：文字列
    - String c = “aaaa”;
    - String c= “1.2”; //これも文字列
  - Boolean：真偽
    - Boolean d = true;

# 順次実行



- プログラムは上から実行される

```
//GPIO14を出力に設定
GPIO.pinMode(outPinNum, GPIO.OUTPUT);
//GPIO15を入力に設定
GPIO.pinMode(inPinNum, GPIO.INPUT);
```

- ちなみに
  - //は実行されないコメント文

# 条件分岐



- If-else文
  - –条件が真ならば処理１，偽ならば処理２を行う

```
If(条件)
{
    処理１
}
else
{
    処理２
}
```

# 繰り返し



- for文
  - 初期化式：初期化する式
  - 条件式：条件が真の間実行
  - 変化式：処理が終わるごとに実行される

```
for(初期化式;条件式;変化式)
{
    処理
}
```

# 関数



- 決められた通りの処理を実行して結果を返す一連の命令
  - processは関数
  - 処理Aと処理Bを実行する
  - 1を結果として返す

```
int process()
{
    処理A;
    処理B;
    return 1;
}
```

# 関数



- 同じ名前の関数は一つだけ
  - だめ

```
int process()
{
    処理B;
    return 1;
}

int process()
{
    処理A;
    return 1;
}
```

# Processingとは



- 電子アートとビジュアルデザインのための プログラミング言語
- 初心者が学習するのに適した言語
- Javaを簡単にしたような言語
  - 内部はJavaで作成されている

- 特別な関数
  - draw
    - 常時実行されている関数
  - mouceClicked
    - マウスクリックされる毎に1回実行される関数
  - setup
    - drawよりも前に最初に一回だけ実行される関数
    - 基礎編の演習はこれだけを使った

# 書き方



- 文末はセミコロン；
- 全角文字は使えない
  - –コメントと""の中だけ可能

# ディジタル回路

# アナログ・ディジタルとは



- アナログ（連続値）
  - 例：
    - アナログ時計（針があるやつ）
    - 電気・光量・温度・水量・湿度
- ディジタル（離散値）
  - 例：
    - ディジタル時計（針がないやつ）
    - そろばん
      - 上段のおはじきと下段のおはじきの位置
    - **コンピュータ**
      - **０と１だけ**

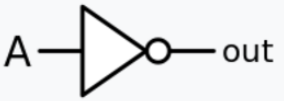
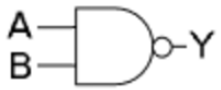
# コンピュータとディジタル回路 20

- コンピュータはデジタル
  - ０と１の組み合わせしか扱えない
- コンピュータと何かの装置をつなぐためにはデジタルな入出力を受けつける回路が必要
  - 「秒速10回転でモーターを右に回して」と指示するのには「**10110101**0の電気信号を流せ」としか言えない
    - **10　　110　　1010**
    - **右か左か　秒・分・時　　2進数で10**
- デジタル回路・論理回路
  - 2進数を扱う回路

# デジタル回路



- ０と１の組み合わせの2入力を入れて０と１の1出力を返す回路
- 組み合わせることで，2進数の入力を2進数の出力に変換できる
- デジタル回路では論理値を電圧として表現
  - 1：真、True = 5V(3.3)，High
  - 0：偽、False = 0V(GND)，Low

| 論理 | 論理式 | 回路記号（MIL記号） |
|---|---|---|
| NOT | $\overline{A}$ |   |
| OR | $A + B$ |   |
| AND | $A \cdot B$ |   |
| XOR | $A \oplus B$ |   |
| NOR | $\overline{A + B}$ |   |
| NAND | $\overline{A \cdot B}$ |   |

# ０・１の様々な表現



**0**

- 偽
- False
- Low
- ０Ｖ（GND）

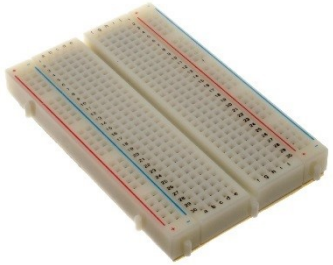
**1**

- 真
- True
- High
- 5V(3.3V)

# ブレッドボードって？

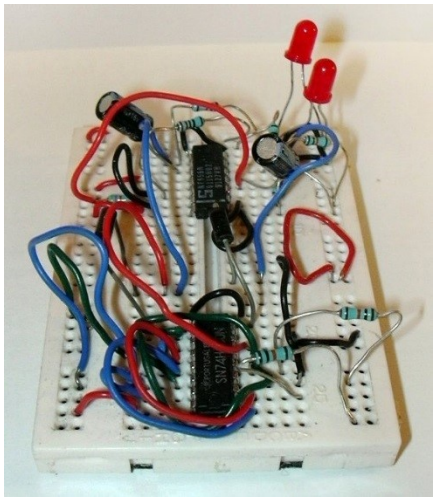


- 試作・実験用の電子回路のこと
- 電子工作が簡単に出来る
- 基盤半田ごていらず（英語だとソルダーレスブレッドボード）



before

- いろんなセンサとラズパイをこれでつないでIoTをする

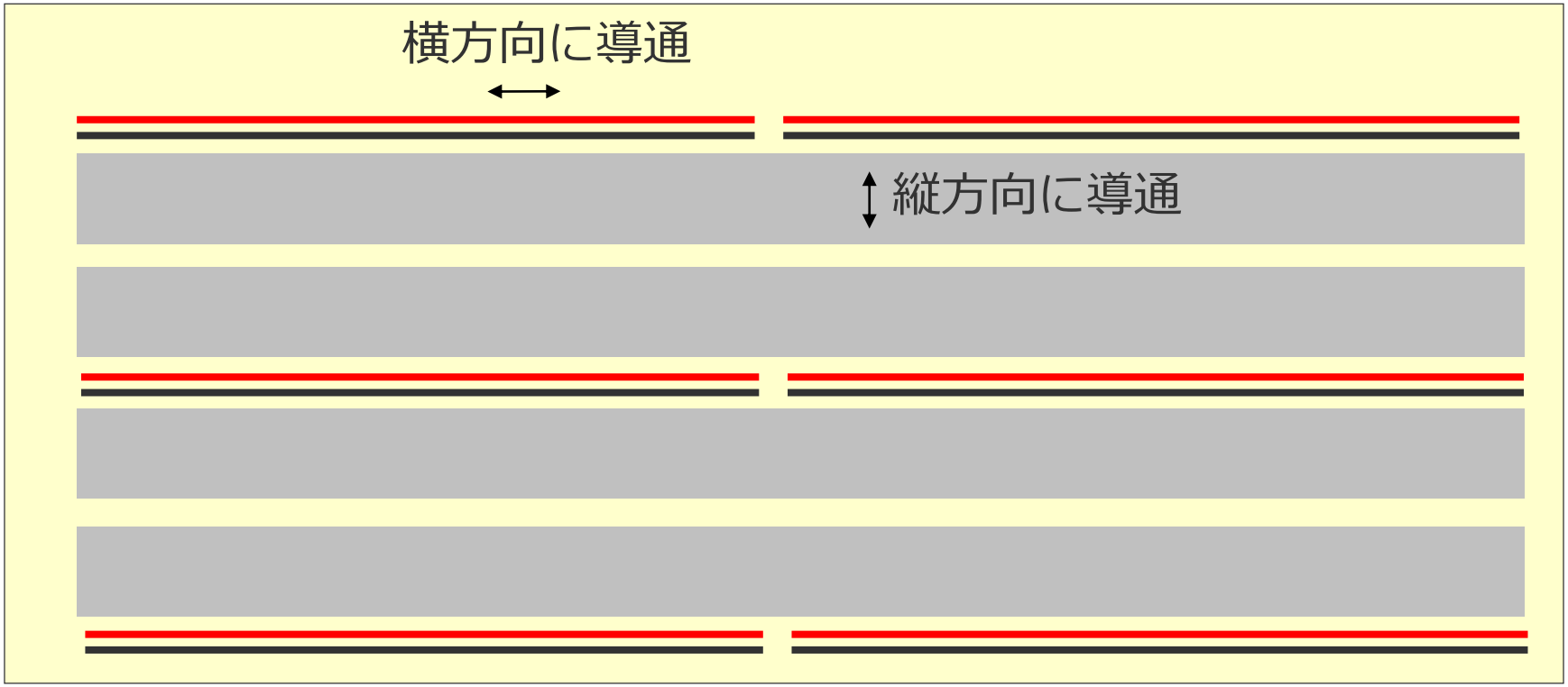


After

# ブレッドボード

- ルール
  - 赤のところには，５V
  - 黒のところには，GND

電源線（赤：5V，黒:GND)



# ディジタル入出力

# 概要



- 達成課題
  - Lチカ
- Lチカとは
  - LED（ライト）をチカチカさせる（光らせること）
  - 電子回路の基本
- 具体的には
  - Raspberry Pi上でProcessingでプログラミング
    Raspberry Piとブレッドボードを接続する

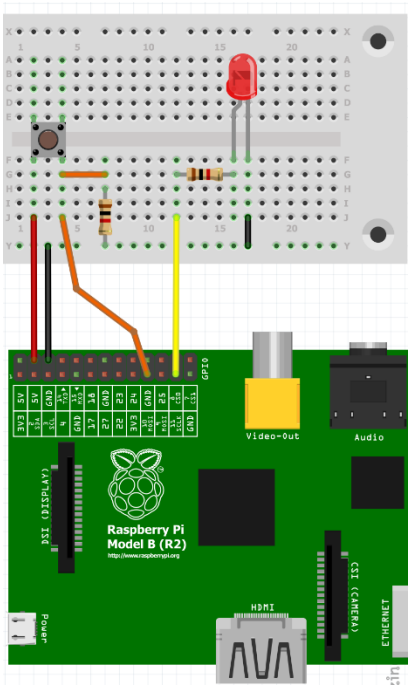
次から詳細の説明

# Raspberry PiとGPIO



- GPIO（General Purpose Input/Output）
  - Raspberry Piは，High, Lowの電圧の入出力を受けとる端子が存在

ここにケーブルを指すことで抜き差してブレッドボードと接続




イメージ図

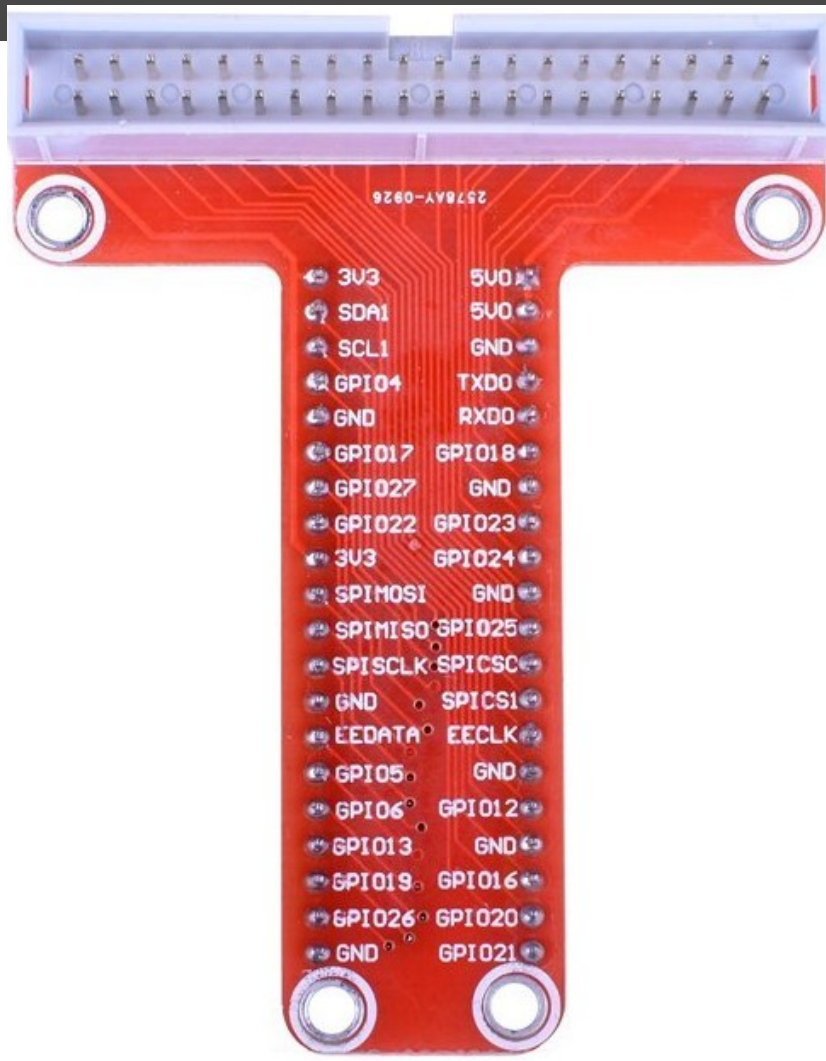
# GPIO（詳細）



| | ピン番号 | | |
|---|---|---|---|
| 3.3V | 1 | 2 | 5V |
| GPIO 2 | 3 | 4 | 5V |
| GPIO 3 | 5 | 6 | GND |
| GPIO 4 | 7 | 8 | GPIO 14 |
| GND | 9 | 10 | GPIO 15 |
| GPIO 17 | 11 | 12 | GPIO 18 |
| GPIO 27 | 13 | 14 | GND |
| GPIO 22 | 15 | 16 | GPIO 23 |
| 3.3V | 17 | 18 | GPIO 24 |
| GPIO 10 | 19 | 20 | GND |
| GPIO 9 | 21 | 22 | GPIO 25 |
| GPIO 11 | 23 | 24 | GPIO 8 |
| GND | 25 | 26 | GPIO 7 |
| ID_SD | 27 | 28 | ID_SC |
| GPIO 5 | 29 | 30 | GND |
| GPIO 6 | 31 | 32 | GPIO 12 |
| GPIO 13 | 33 | 34 | GND |
| GPIO 19 | 35 | 36 | GPIO 16 |
| GPIO 26 | 37 | 38 | GPIO 20 |
| GND | 39 | 40 | GPIO 21 |

色分け

| |
|---|
| 5V |
| 3.3V |
| GND |
| GPIO |
| GPIO(抵抗付) |
| I2C EEPROM |




Reference: https://tool-lab.com/make/raspberrypi-startup-22/

# LED



- LED
  - 発光ダイオード
  - 電圧をかけると光る
  - 足が長い方が＋
    足が短い方がー

# タクトスイッチ



- スイッチ
  - 押したら，オン
  - 通常，オフ

# 演習 1



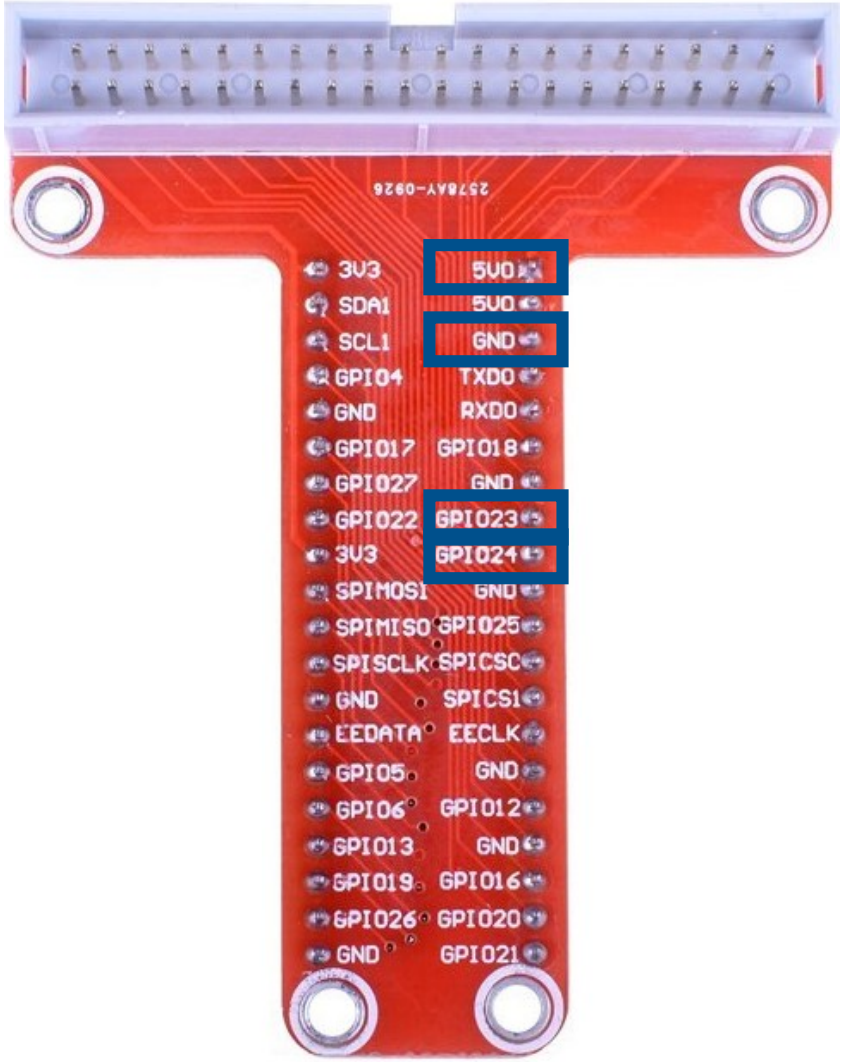
- Processingを起動
  - –Processingの位置ってどこ？（左上にあるP）
  - –ソースコードを開く

    /home/pi/exam-code/01_digital_in_out/digital_in_out/digital_in_out.pde
- GPIOから制御信号を出力してLEDを操作
- GPIOにリードスイッチで0・１を入力

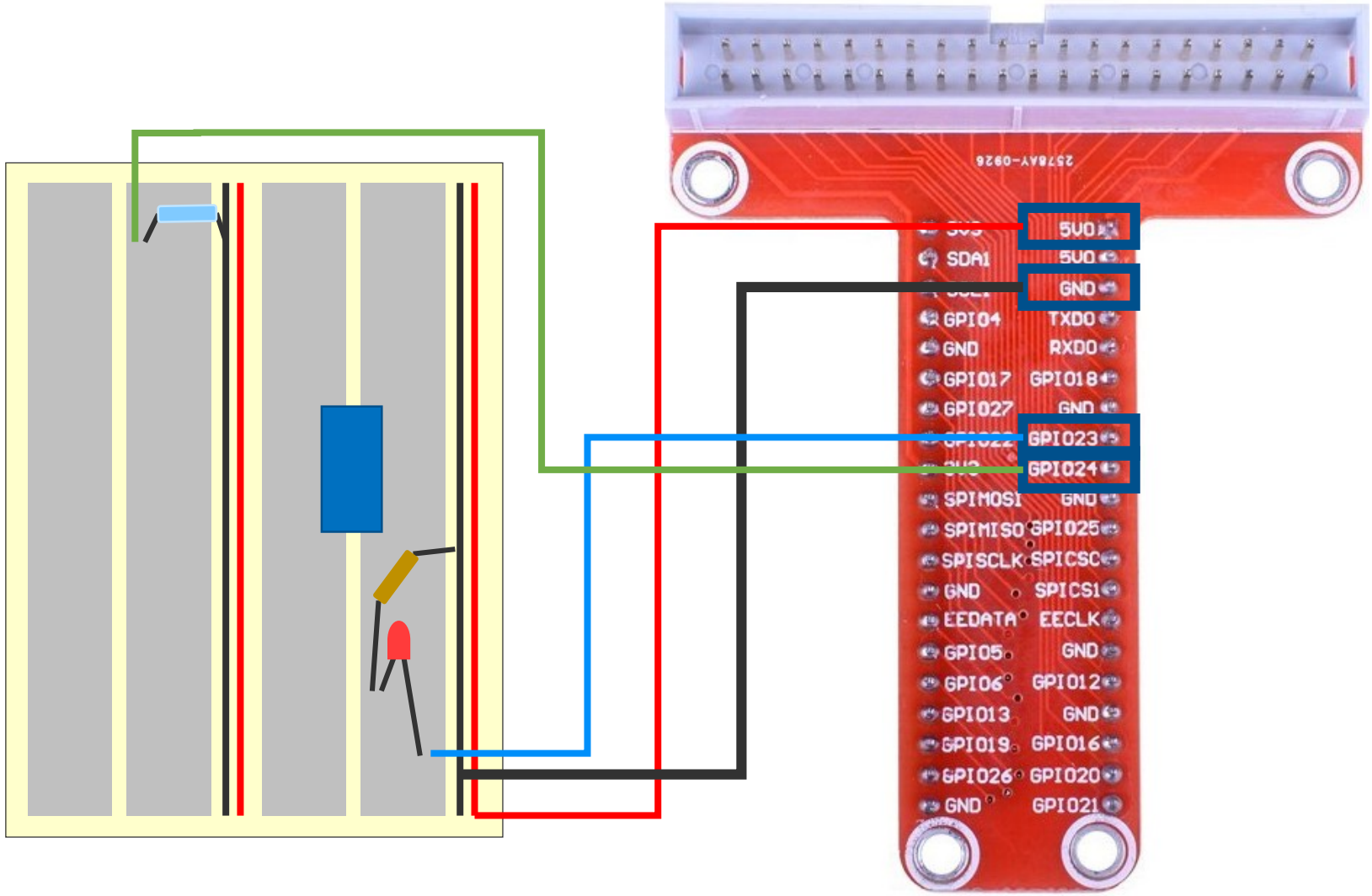
# 結線（文字）



- 5V（5VO）を赤に接続
- GNDを黒に接続
- GPIO23をLEDの抵抗が刺さっていないほうに接続
- GPIO24をスイッチの黒に刺さっていないほうに接続
- できたら，プログラムを実行



# 結線（図）


5V0
SDA1
GND
GPIO4
TXD0
GND
RXD0
GPIO17
GPIO18
GPIO27
GND
GPIO23
GPIO24
SPIMOSI
GND
SPIMISO
GPIO25
SPISCLK
SPICSC
GND
SPICS1
EEDATA
EECLK
GPIO5
GND
GPIO6
GPIO12
GPIO13
GND
GPIO19
GPIO16
GPIO26
GPIO20
GND
GPIO21

# 注意事項



- ショートすると再起動します
  - 5Vの線をRaspberry Piの基板や金属部分に接触させないように
  - 5VとGNDを直結しないように
  - 線の抜き差しはゆっくりと丁寧に
- リードスイッチはガラスなので割れます

# アナログーディジタル変換とセンサ

# 概要



- 達成課題
  - 温度センサの値を取得する
  - 温度などの取得したデータをWebに送信
- 具体的には
  - Raspberry Piと温度センサを接続
  - 送信したデータをWebで確認

次から詳細の説明

# アナログーディジタル変換



- アナログーディジタル変換とは
  - アナログ（連続値）をディジタル（離散値）に変更すること
- なぜ必要
  - コンピュータで扱えるのはディジタル
  - センサ（例：温度センサ）から取れる値（電圧）はアナログ
  - コンピュータで扱うためには，アナログからディジタルから変換が必要

# ADC IC



- ADC IC
  - Analog-Digital Converter IC
  - アナログからディジタルに変換してくれるIC
  - 今回使うICはMCP3008
- 分解能
  - ADC ICがアナログをどのぐらいの段階でディジタルに分解できるのか
  - MCP3008は1024（=2^10：２進数10桁）段階に分解可能
  - 5V/2^10=約5 mVの単位で測定可能
  - 数値と電圧の関係
    - 0 => 0V
    - 1=> 0.005V
    - 20=>0.1V

# 温度センサ



- 温度センサ
  - 温度を電圧に変換してくれるセンサ
  - 電圧はアナログ
  - 今回使用するLM35DZは０度から１００度まで測れる
  - ０度での出力は０Ｖ
  - １度あがるごとに10mV上昇
    - ０度：０Ｖ
    - 20度：200mV = 0.2V

# 演習 2



- Raspberry PiにADC ICを接続
- ADC ICに温度センサを接続（接続済み）
- Processingで温度を取得（実行するだけ）
  - ソースコード

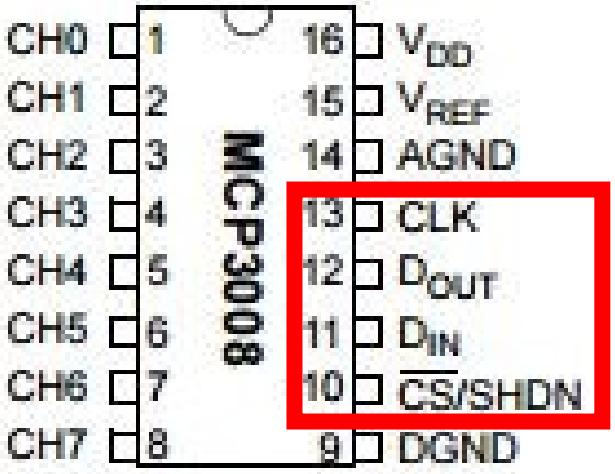
    /home/pi/exam-code/02_adc_ic/adc_ic/adc_ic.pde
- Processingで取得した値をWebに送信（コードに記述済み）
- Webブラウザで値を見る
  - http://localhost/temperature/

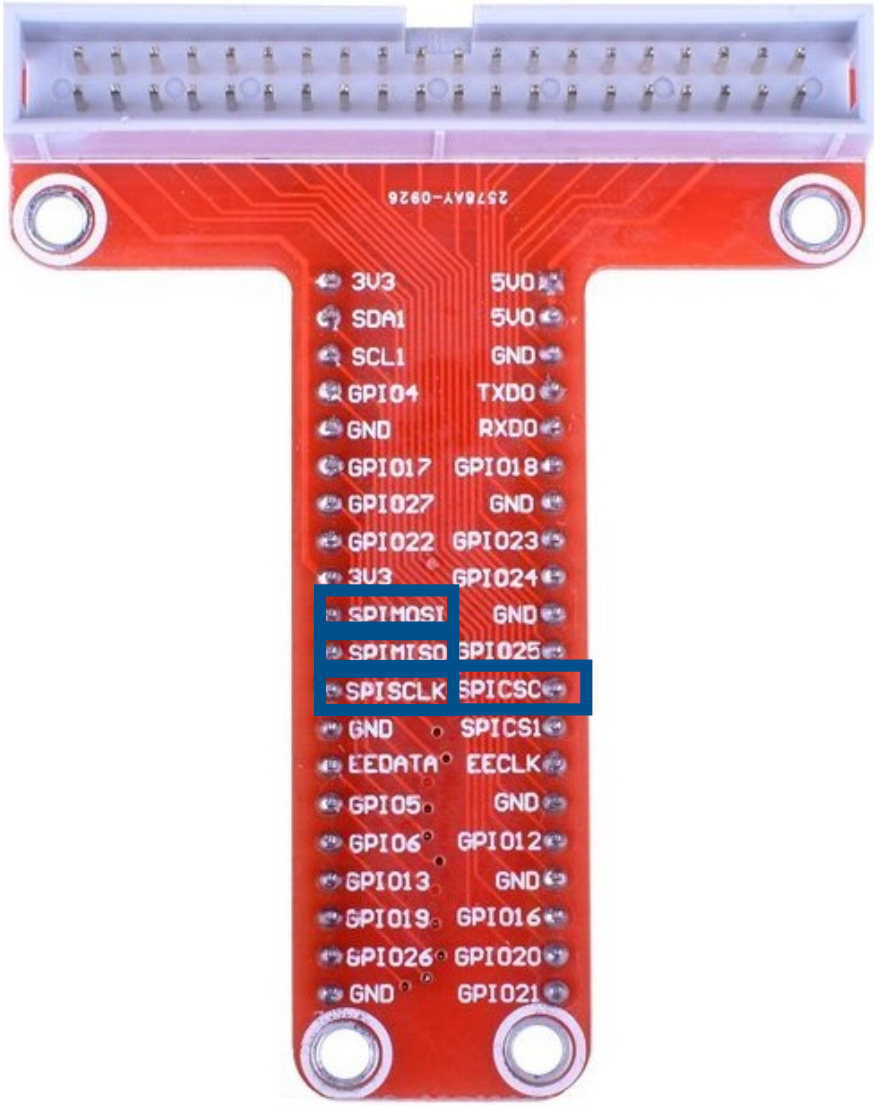
# 結線



## ICは，くぼんでる方が上

- ADC IC
  - CLK => SPISCLK
  - $D_{OUT}$ => SPIMISO
  - $D_{IN}$ => SPIMOSI
  - $\overline{CS}$/SHDN => SPICSC



# 課題

# 課題１（ディジタル入出力）



- LEDとリードスイッチの接続を変更する
  - リードスイッチ：GPIO24
  - LED：GPIO23
- プログラムも変更
  - 該当するGPIOに変更

```
//入力に使うGPIO番号
int inPinNum =24;
//出力に使うGPIO番号
int outPinNum =23;
```

# 課題２（アナログ入力とセンサ）45

- 温度を摂氏から華氏に変換して表示
  - 摂氏から華氏への変換式

$$°\mathrm{F} = \frac{9.0}{5.0}°\mathrm{C} + 32.0$$

  - 小数点で書くこと
- ディジタル値を温度に変換している部分
  - ディジタル値を電圧に変換して100倍すれば摂氏

```
float digital2Temp(int digital, float vref)
{
        float c = digital2Volt(digital, vref)*100;
        return c;
}
```

- 表示する方法
  - println(表示する値);
  - 文字列は＋でつなげることができる