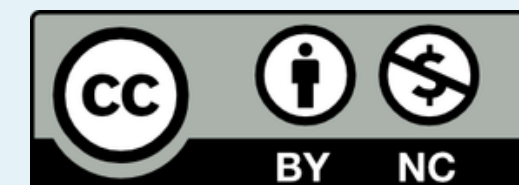


# コードとデザイン

東京藝術大学 芸術情報センター開設科目 金曜4-5限 第6週

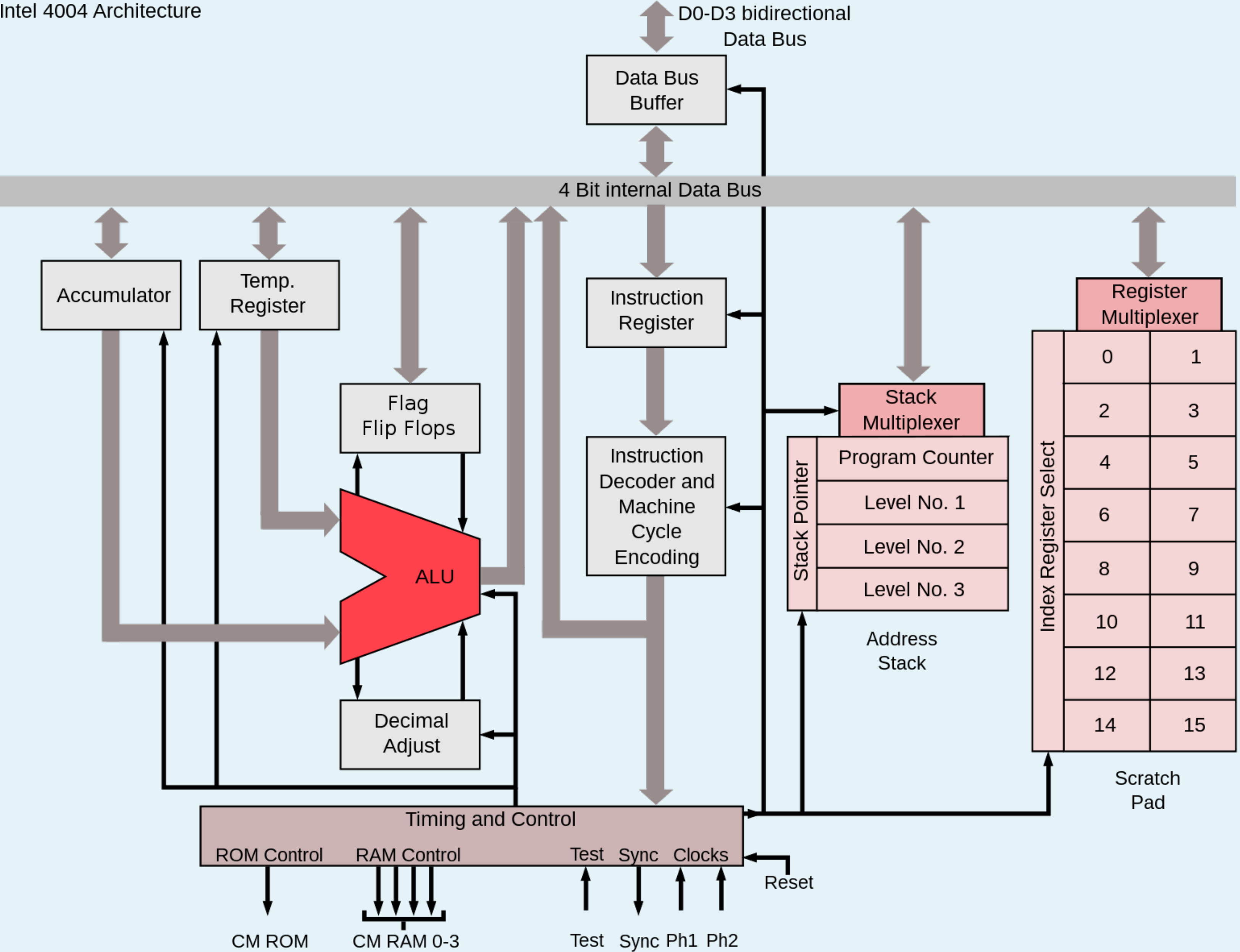
2024.05.24 松浦知也 ([matsura.tomoya@noc.geidai.ac.jp](mailto:matsura.tomoya@noc.geidai.ac.jp) [teach@matsuuratomoya.com](mailto:teach@matsuuratomoya.com))



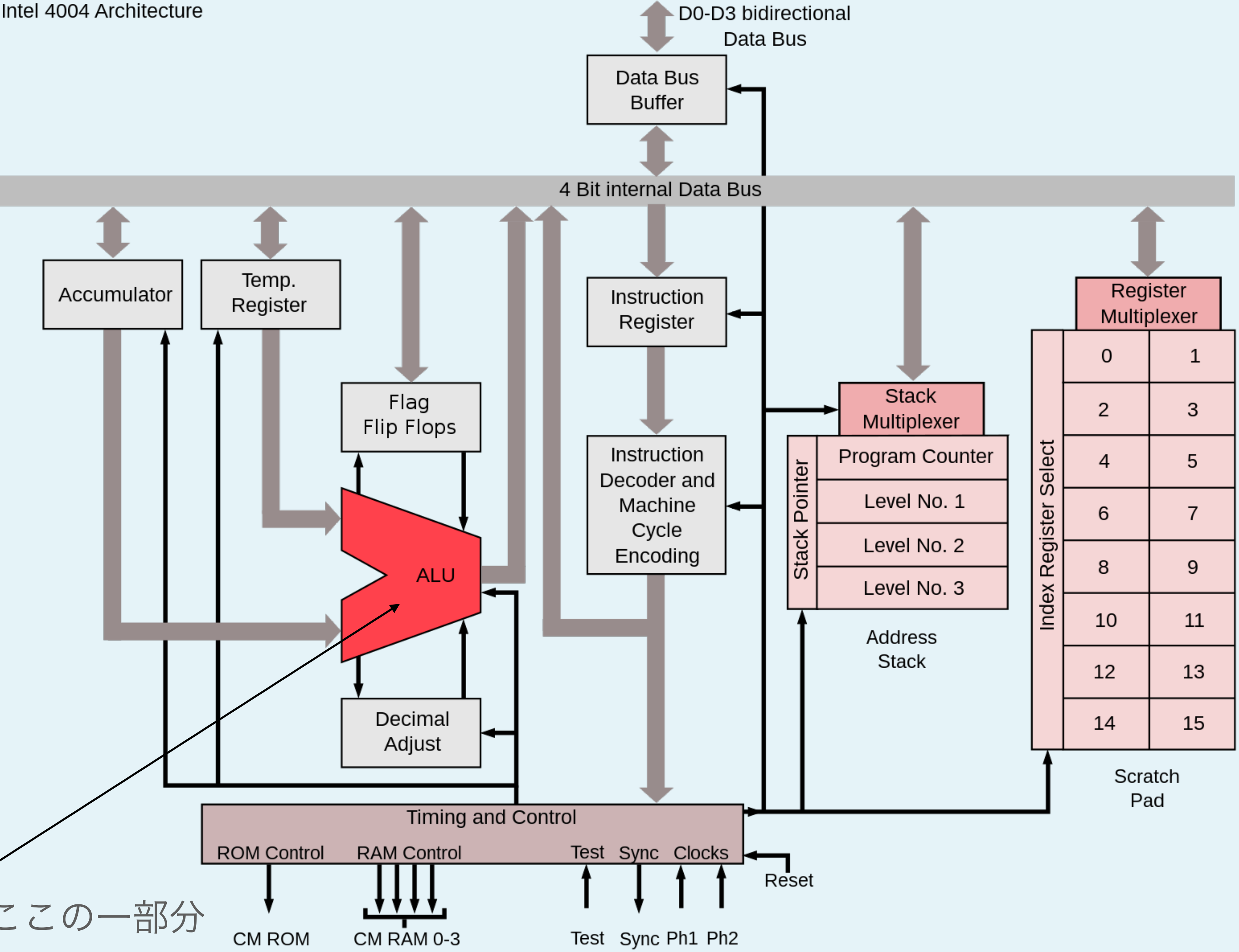
- 二進数で計算したものを最終的にどうやって人間の読めるフォーマットで出力する？

# 電卓とコンピューターの違い

Intel 4004 Architecture

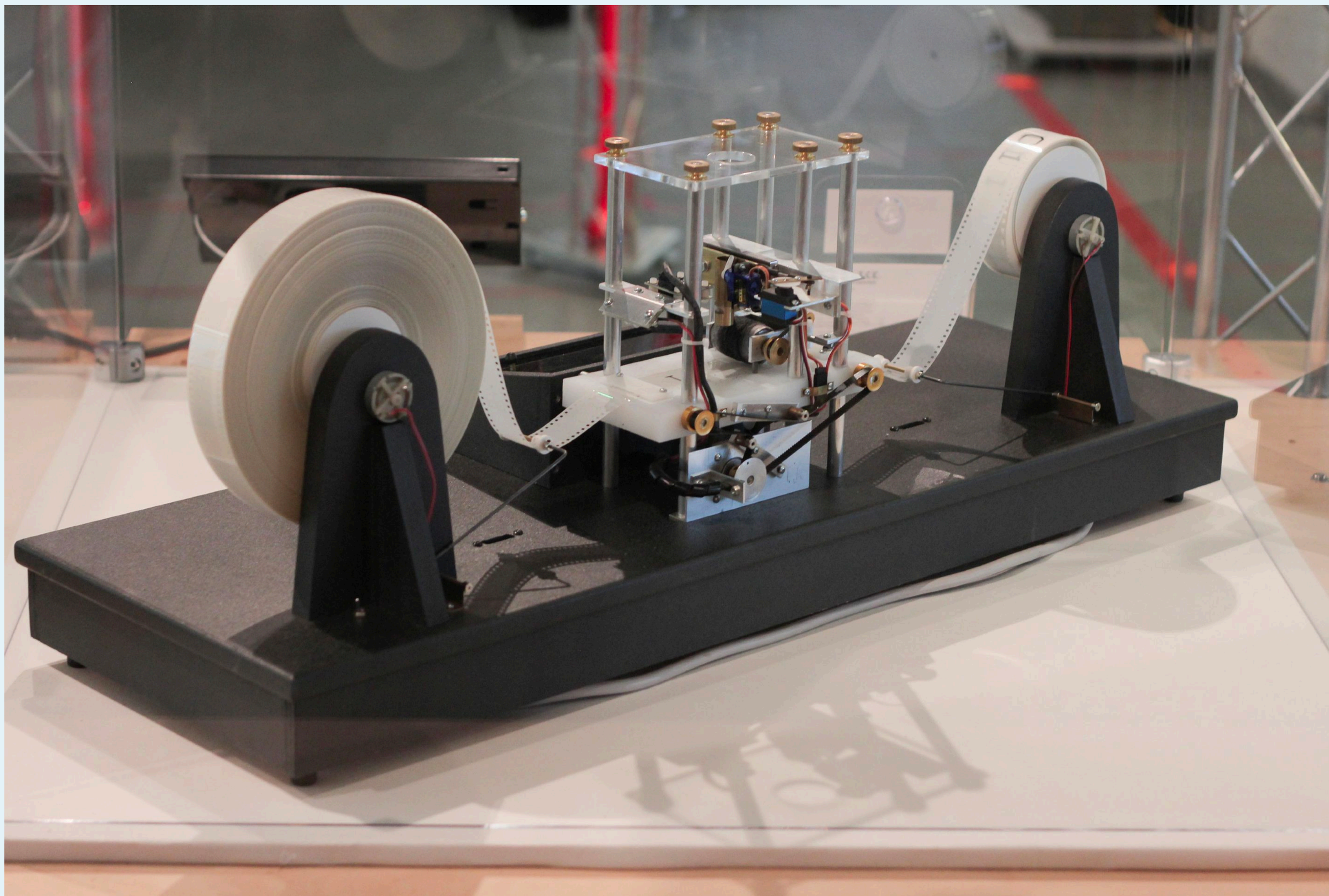






前回まで作ったのはここ的一部分





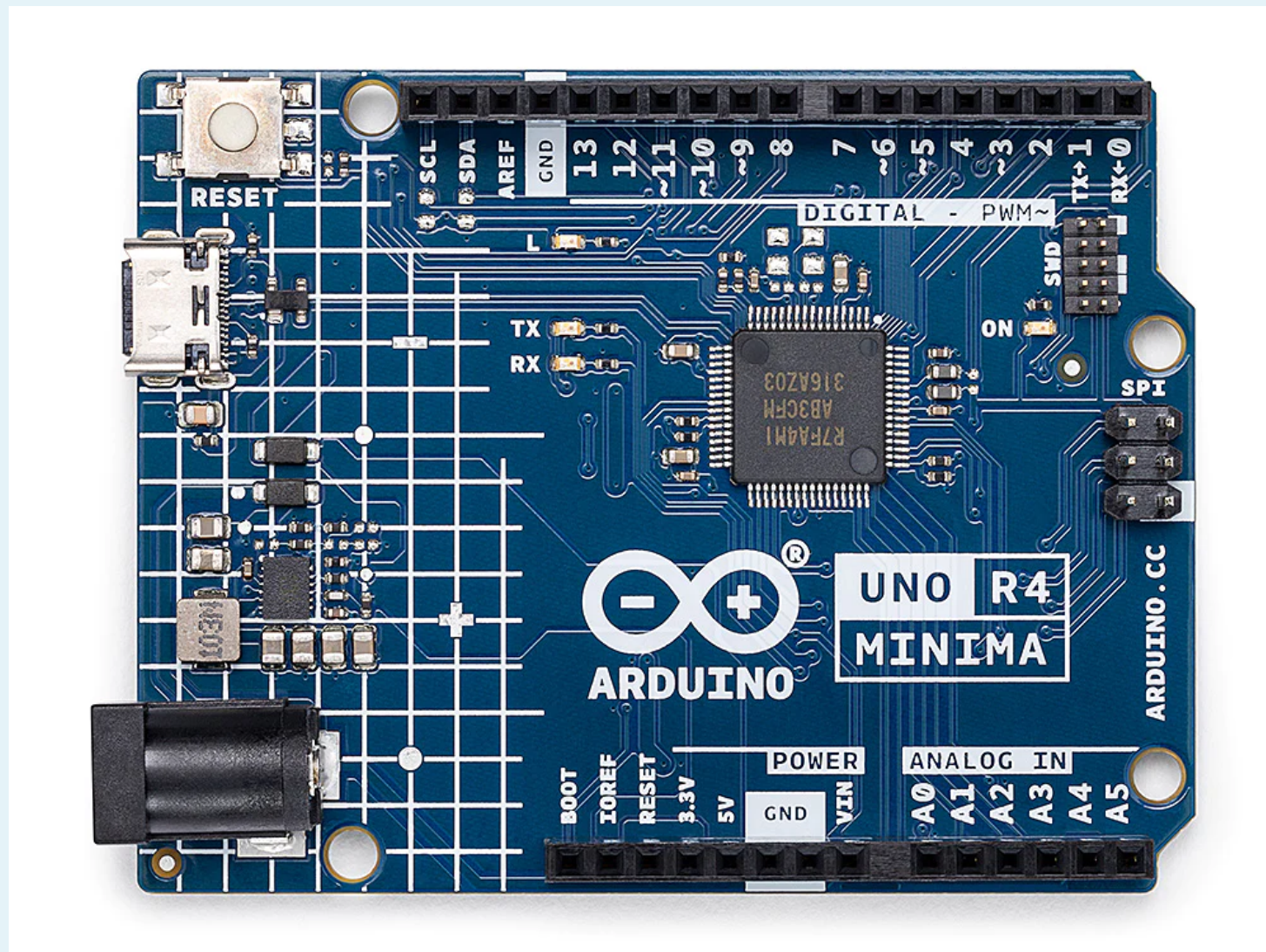
## チューリングマシンの物理モデル

Turing Machine, reconstructed by Mike Davey as seen at Go Ask ALICE at Harvard University, Rocky Acosta - Own work, CC-BY-3.0(2012), [https://en.wikipedia.org/wiki/Turing\\_machine#/media/File:Turing\\_Machine\\_Model\\_Davey\\_2012.jpg](https://en.wikipedia.org/wiki/Turing_machine#/media/File:Turing_Machine_Model_Davey_2012.jpg)



# Arduino入門

# Arduino



+



規格化された入出力のボード

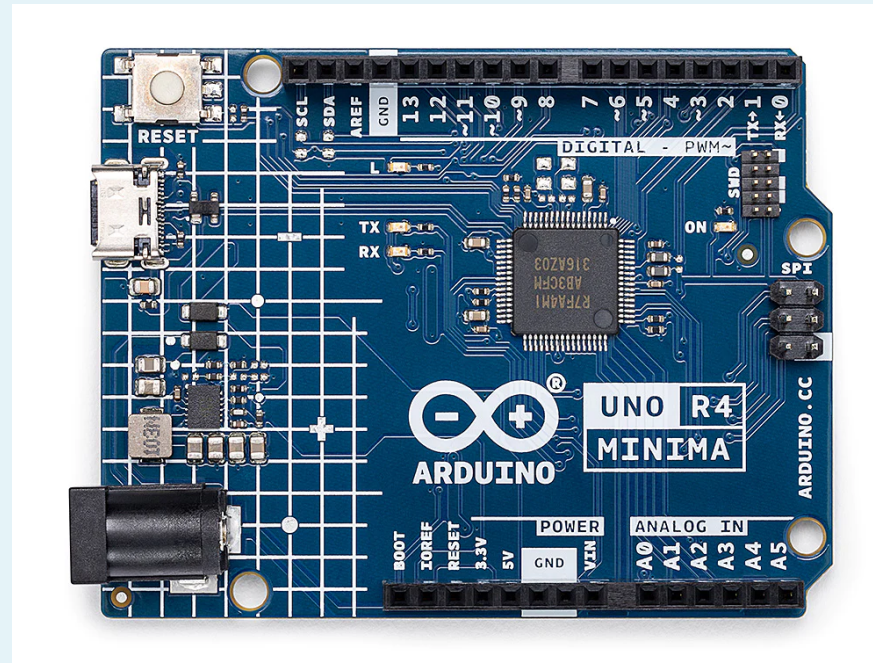
簡単 + 複数のボードに対応したプログラミング環境

C、C++を簡易化した言語を使用

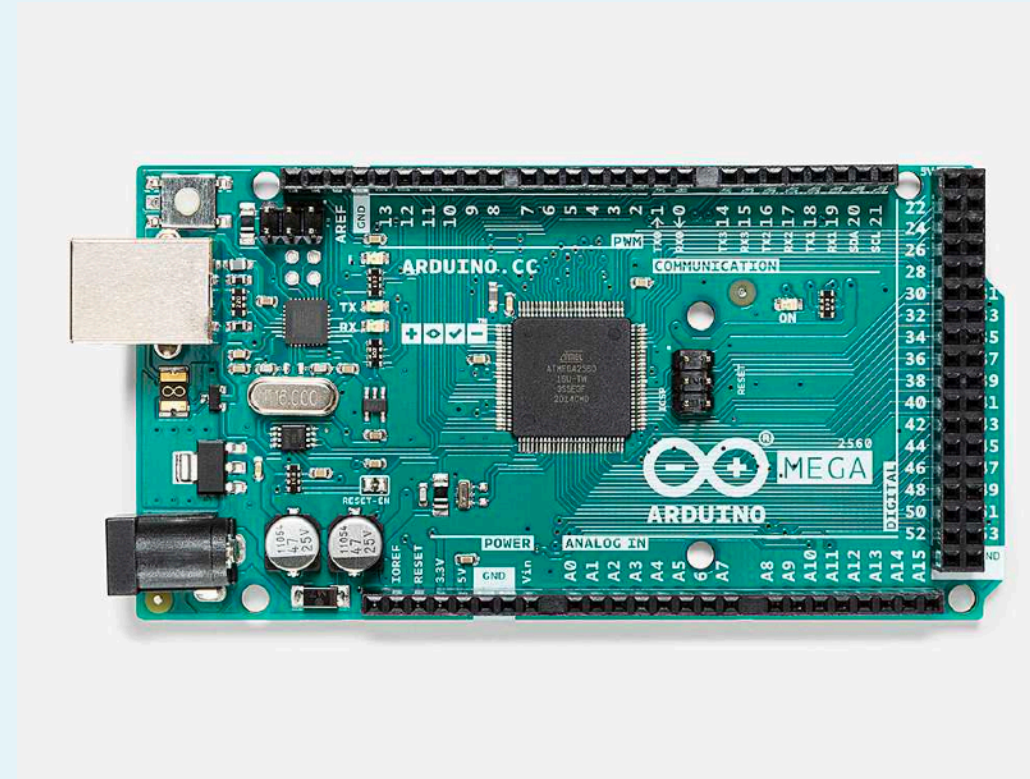
拡張子は.ino



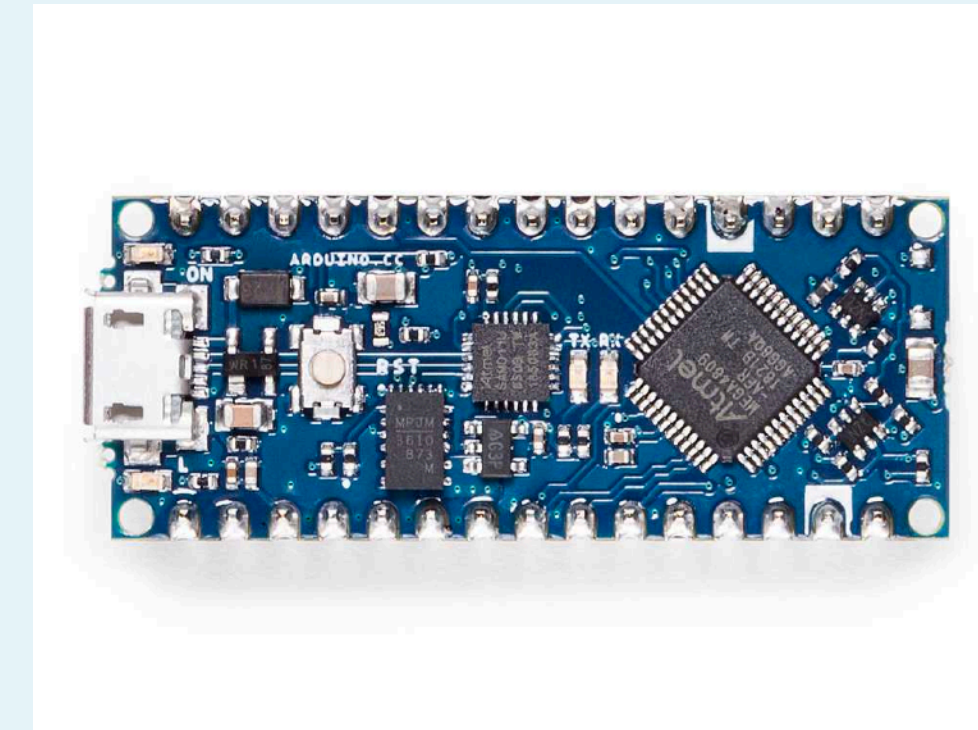
## Arduino公式シリーズ



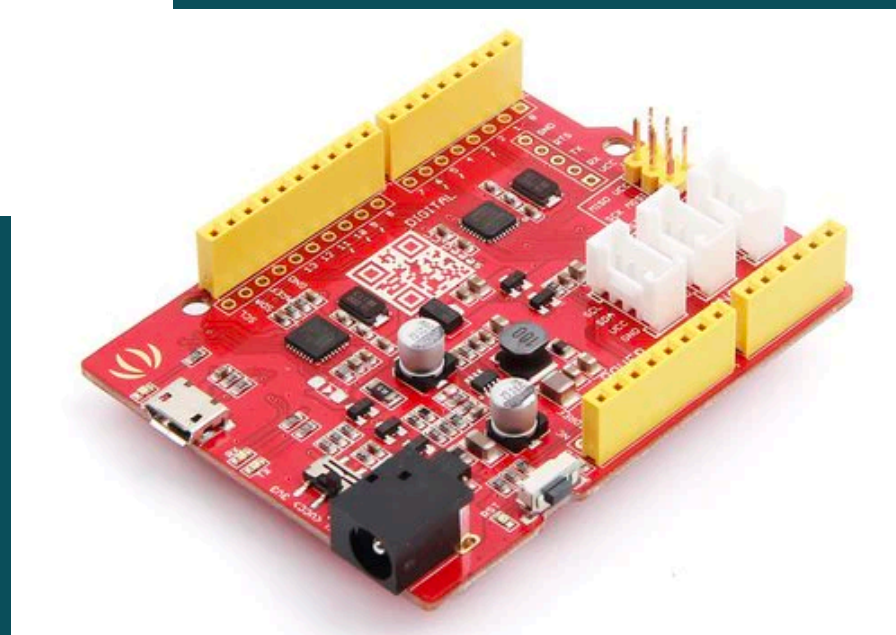
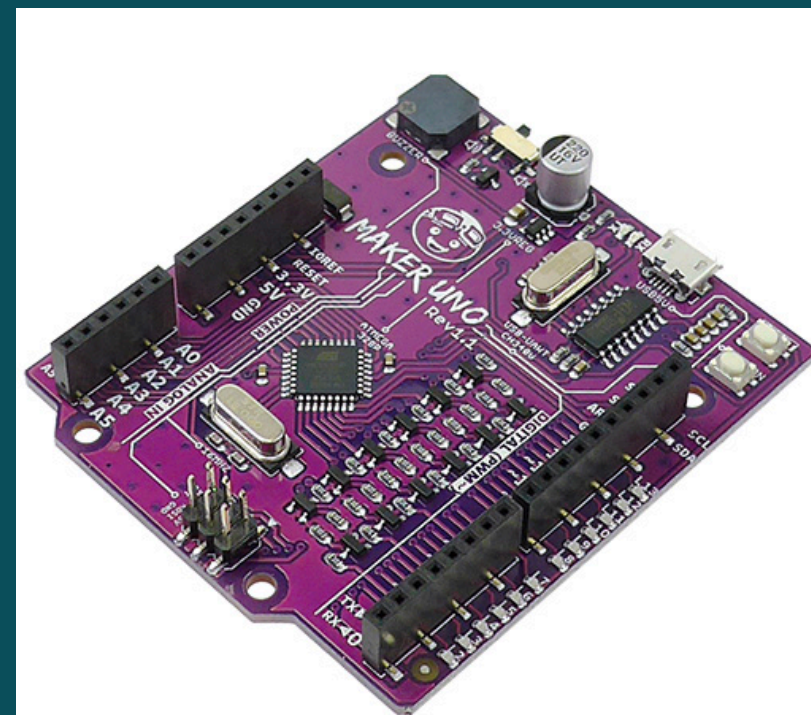
Arduino UNO(R4 minima)



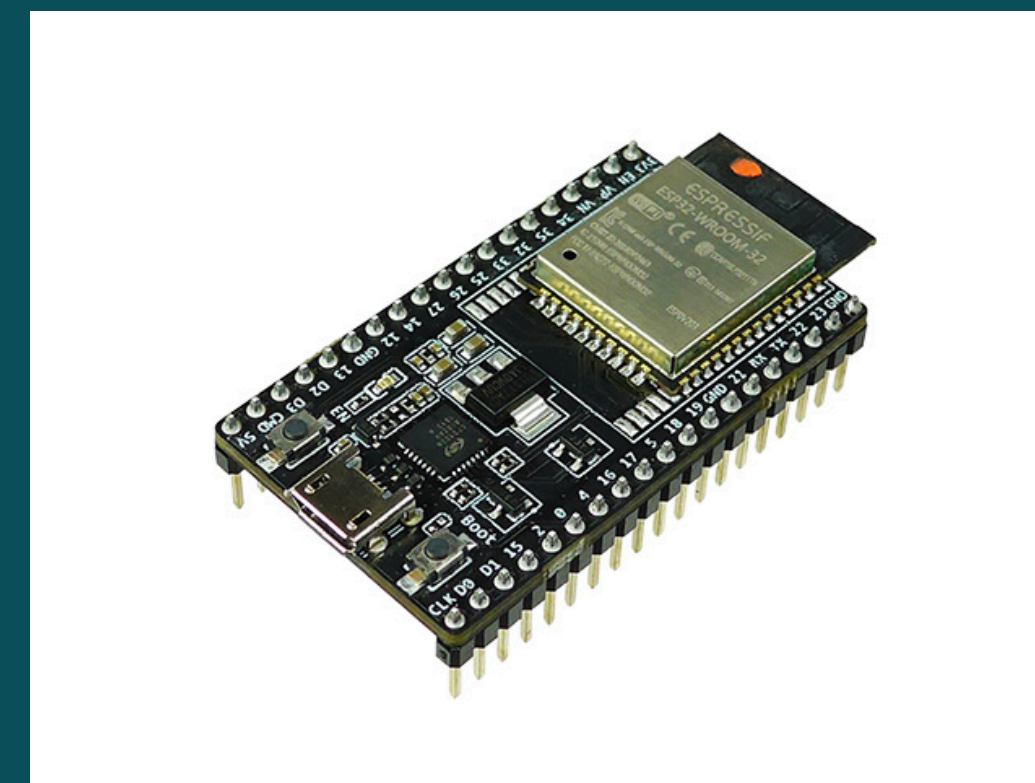
Arduino Mega



Arduino Nano Every

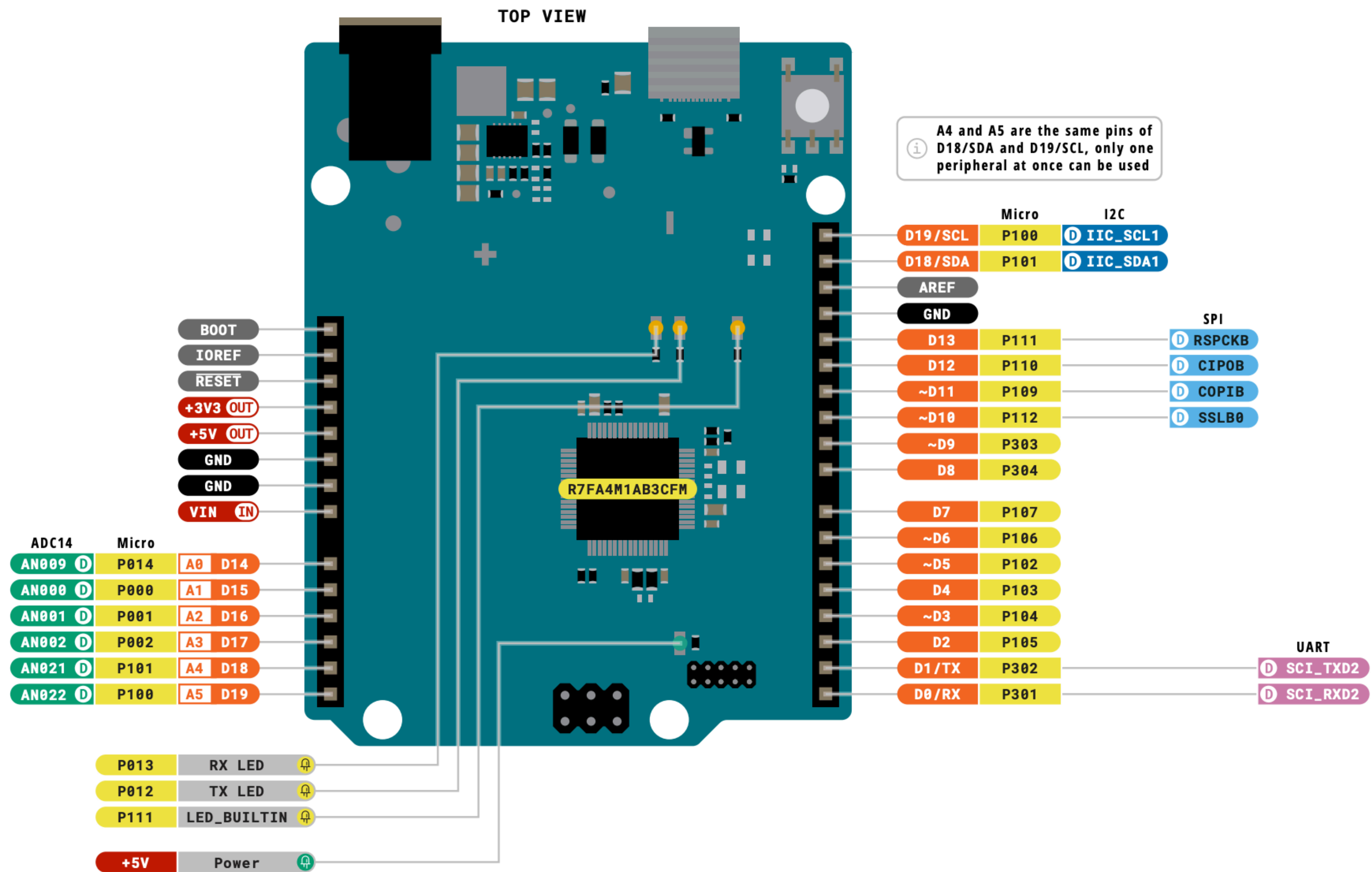


ピン配置互換性

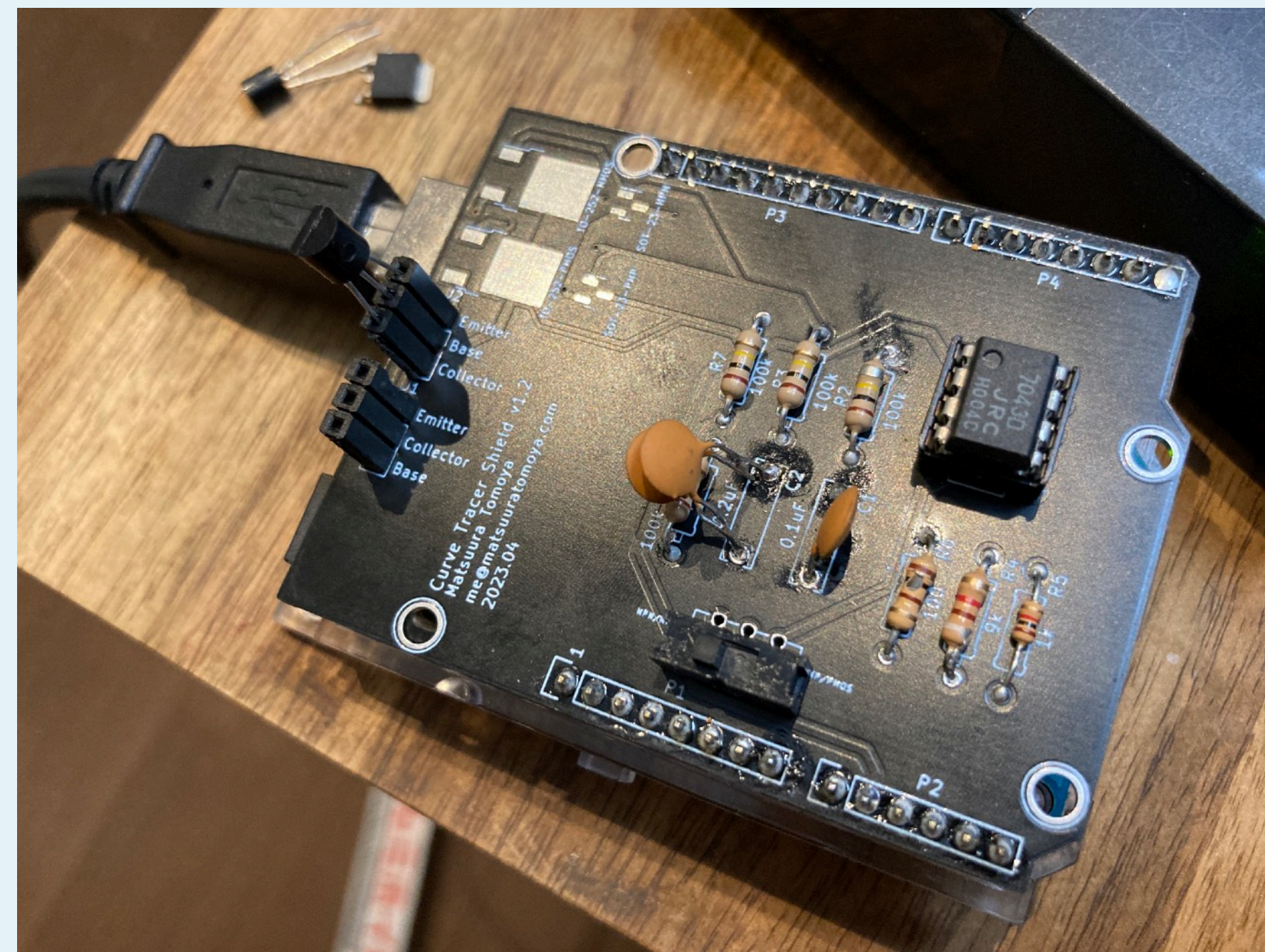
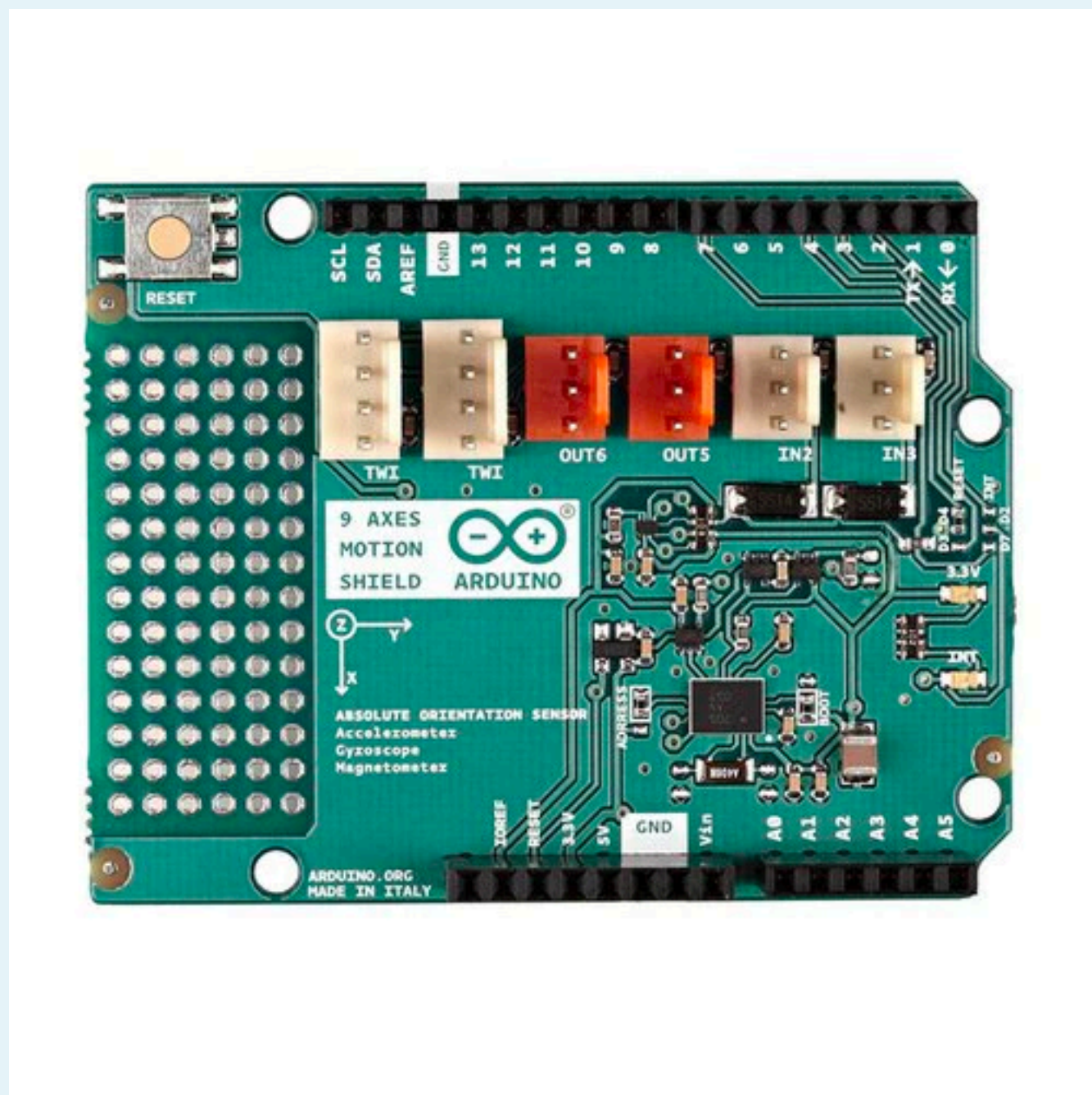


Arduino IDEでの開発対応



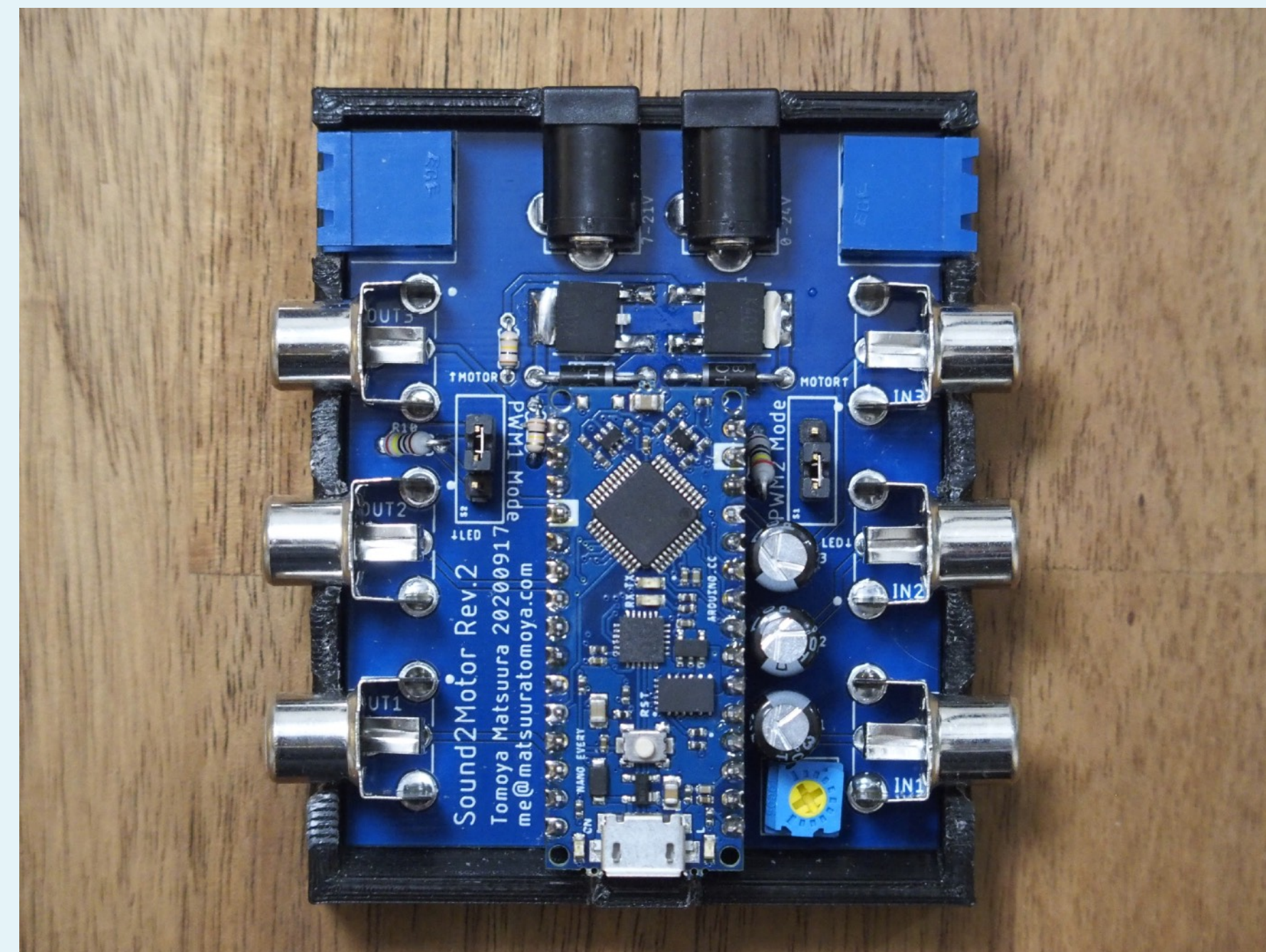
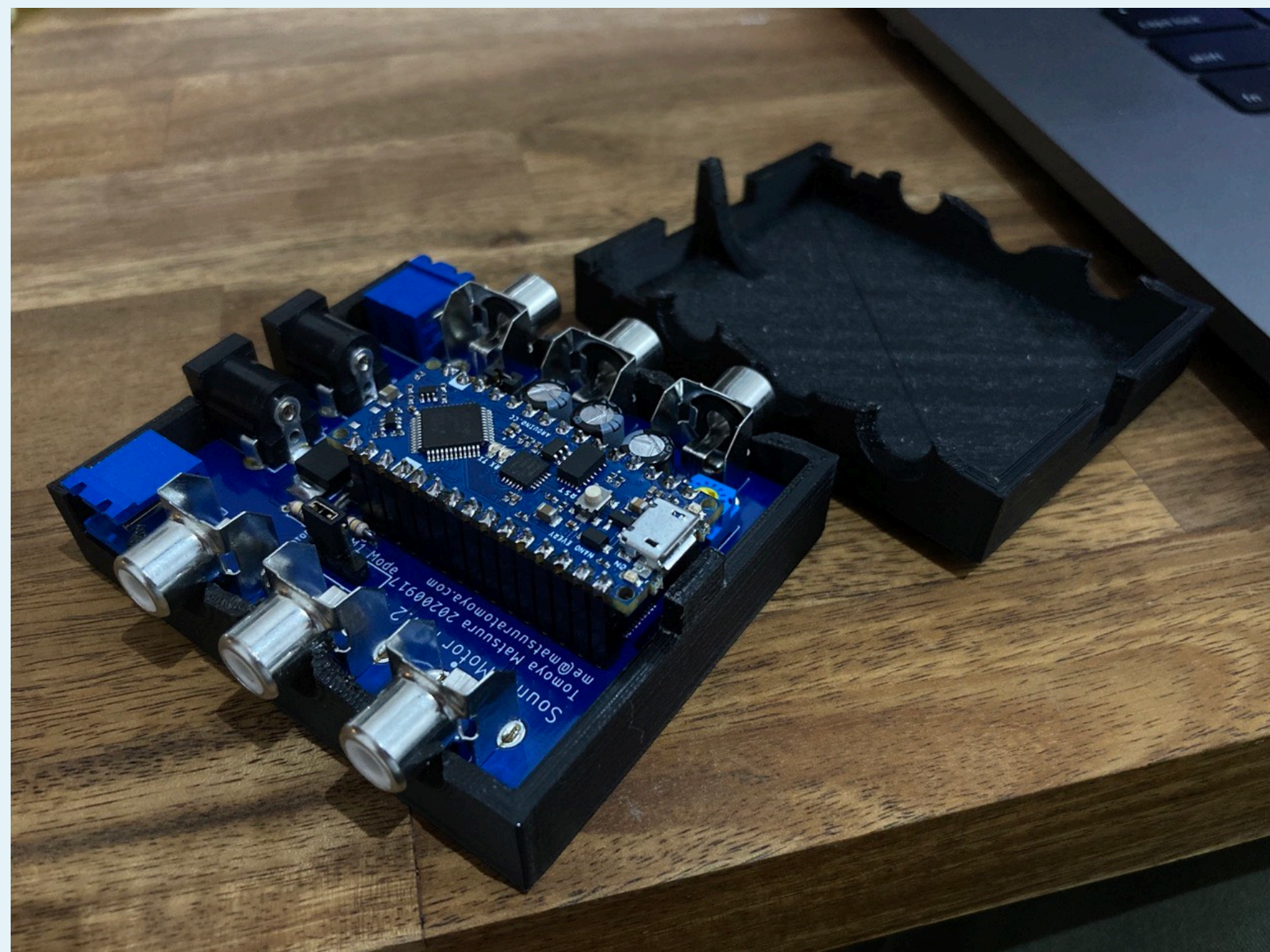






シールドと呼ばれる拡張基板を組み合わせることができる  
(自分でも作れる)





音量に応じてLEDとモーターを起動するためのデバイス（2020年）

Arduino Nano Every用の拡張基板として実装



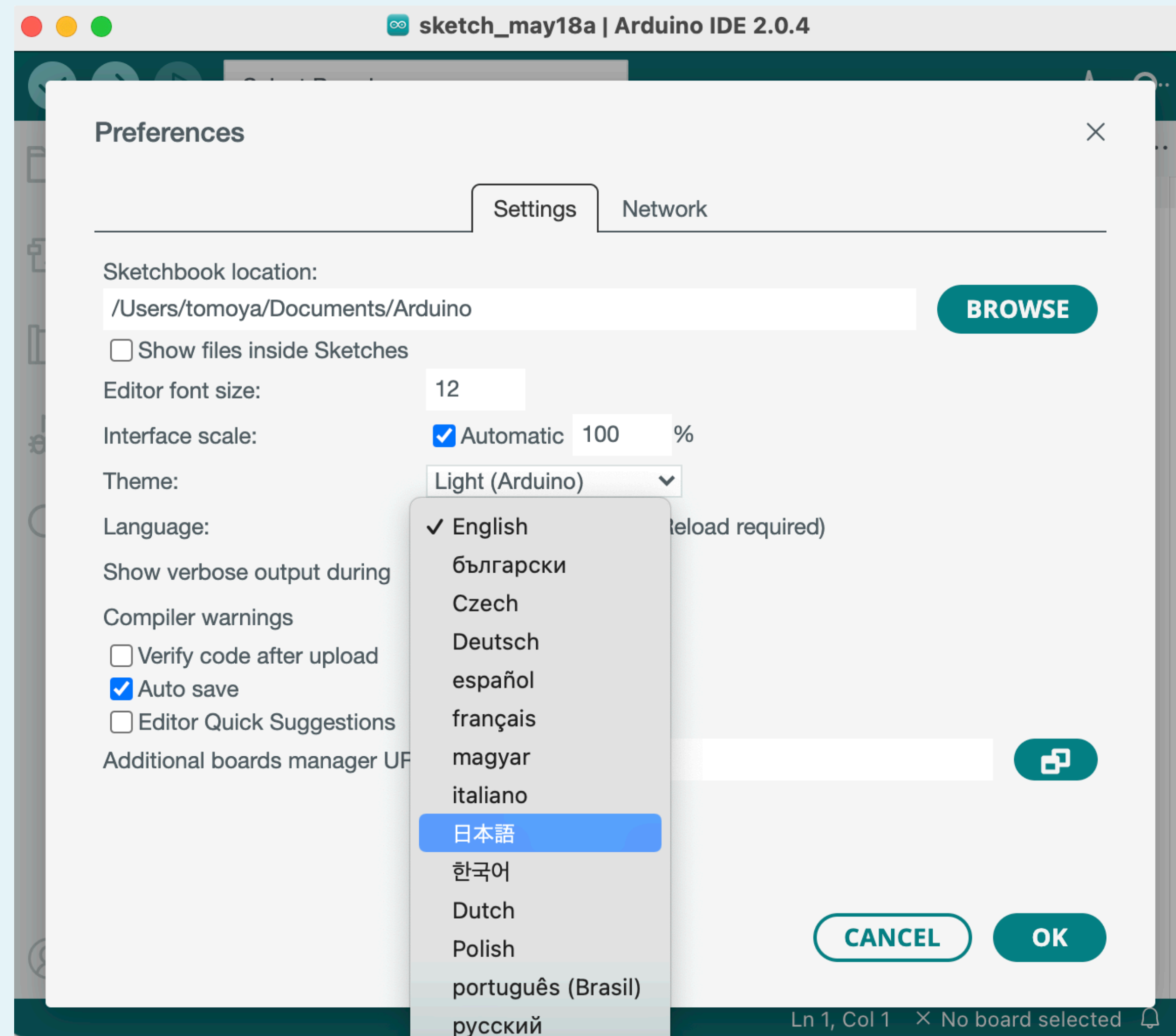


梅田哲也『時報』（福岡市美術館 「うたの起源」 2020年）

# Arduinoのプログラム入門

# そもそも：プログラミングとは

- コンピューターの特徴は、データだけでなく自分を動かす命令=プログラムも二進数のデータで記憶装置に読み書きできる点（プログラム内蔵方式）
- 二進数のまま命令データを記述するのは大変なので、まず人でも読み書きできるような人工言語=プログラミング言語の文章=ソースコードとして記述してから、二進数のフォーマットに変換=コンパイルして使う
- プログラミング言語は自然言語（英語や日本語）と違って厳密なルールに従って記述する必要がある

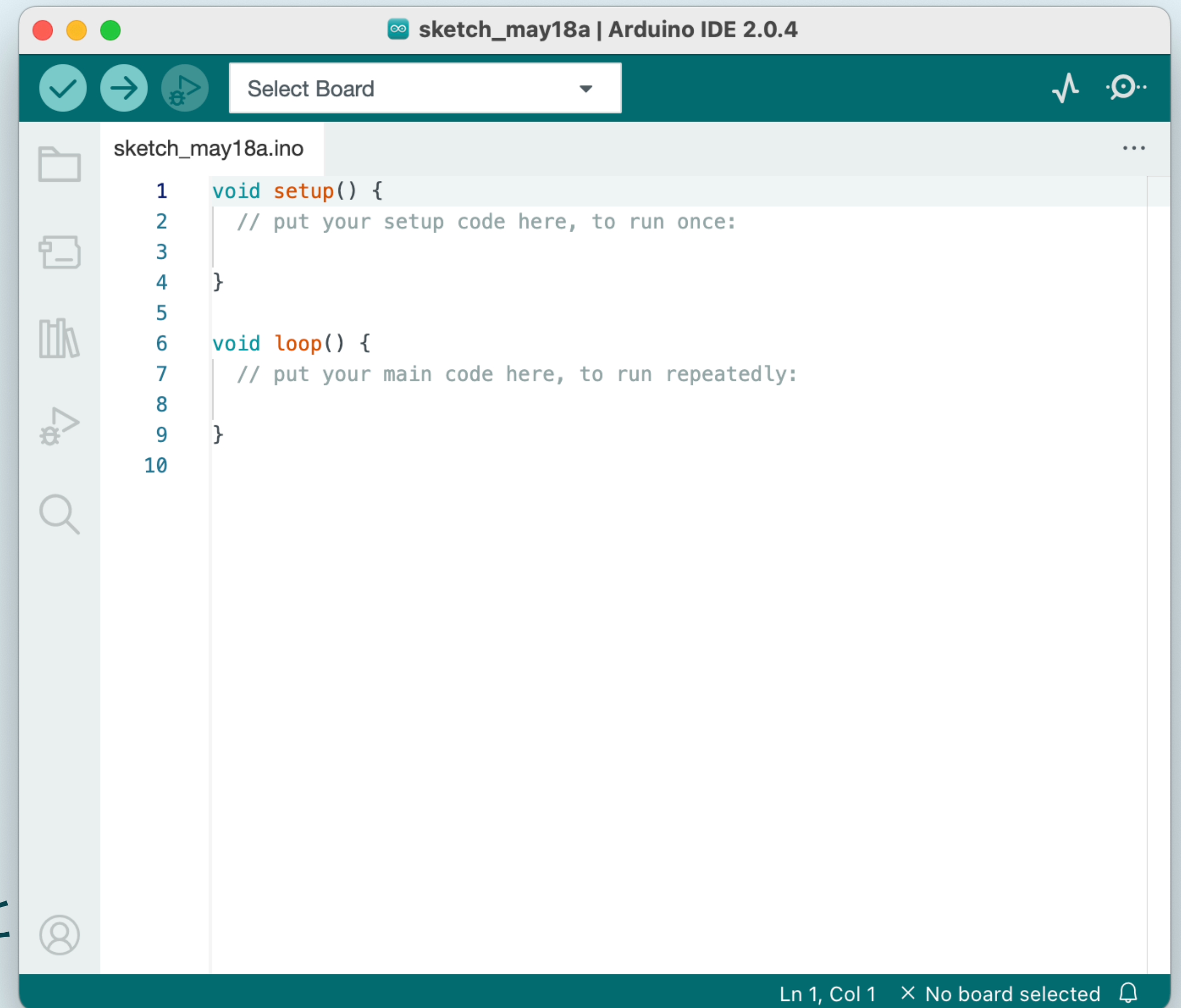


環境設定(Cmd+,)⇒Languageで日本語化できます



# スケッチ

- Arduinoのソースコードのファイルは「スケッチ」と呼ばれる
- Arduino IDE.appを立ち上げると自動で日付が入ったスケッチが開かれる
- ドキュメント/Arduino 以下に保存されるので、AMCの端末を使用する方は適宜USBメモリなどに保存すること



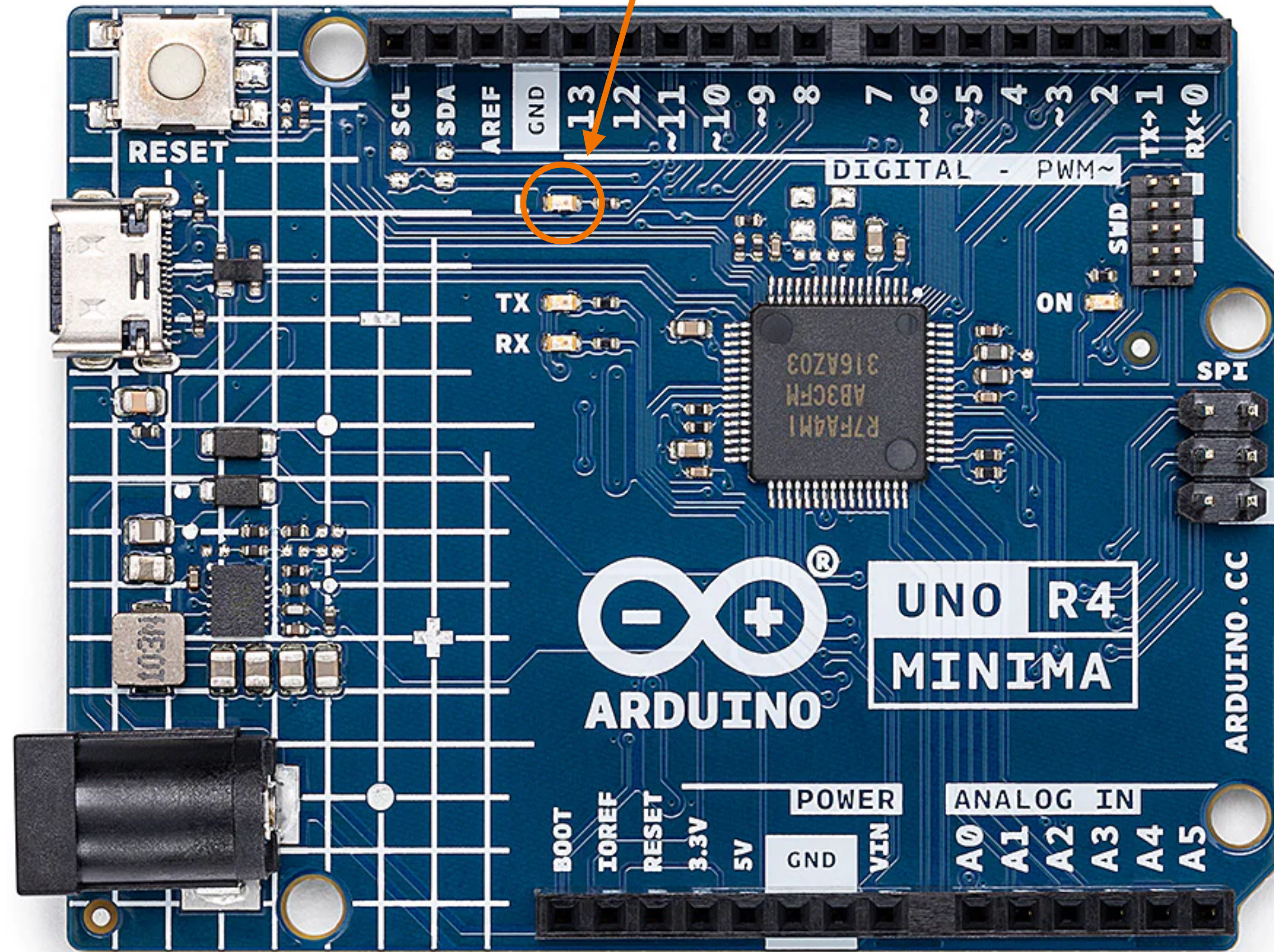
# スケッチ

```
void setup() {  
  pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(LED_BUILTIN, HIGH);  
  delay(1000);  
  digitalWrite(LED_BUILTIN, LOW);  
  delay(1000);  
}
```

とりあえず写経してみましょう



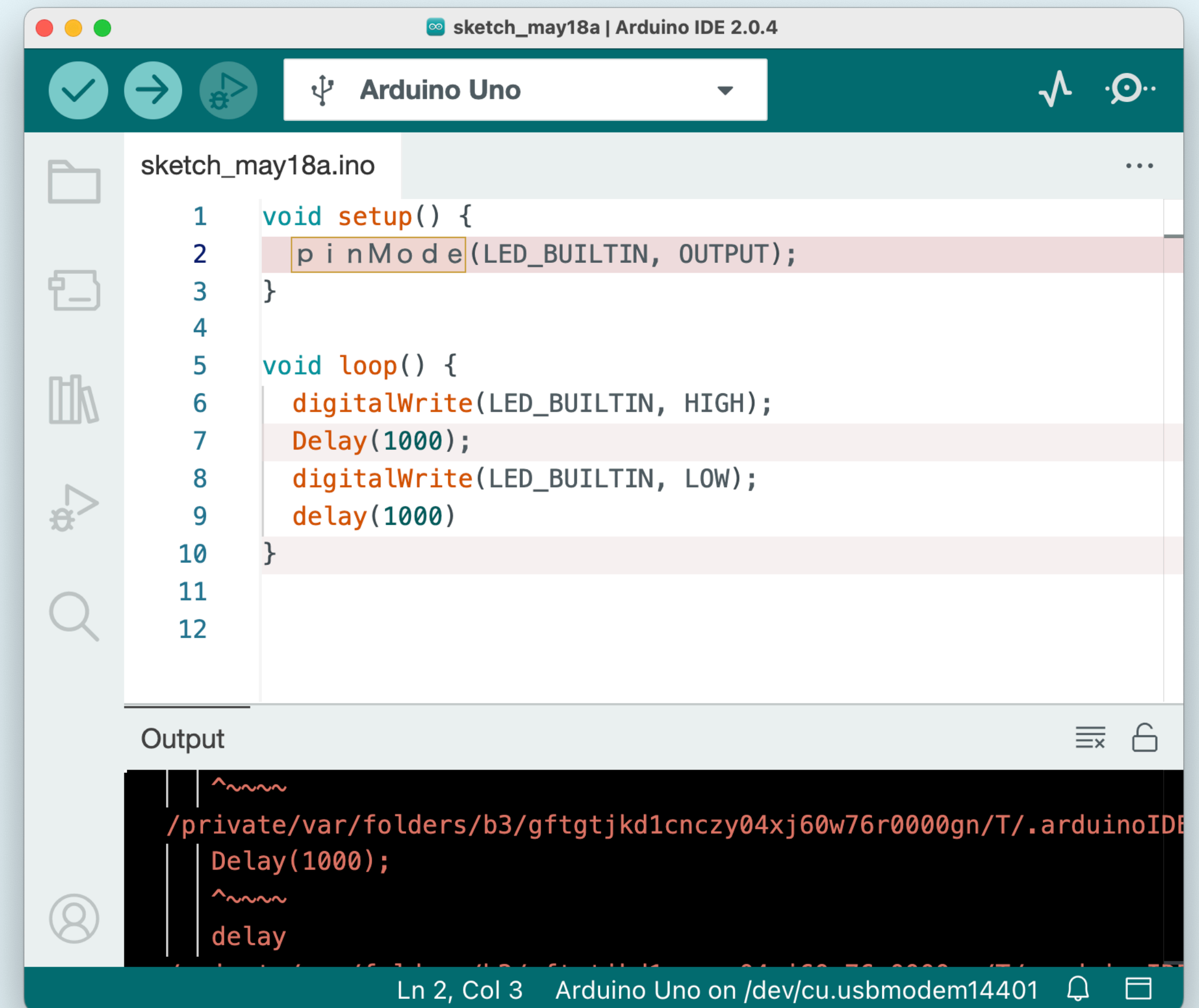
ここが1秒おきに点滅する





# C (C++) 言語のポイント

- 全角でタイプしない
- 大文字小文字を間違えない
- 行末のセミコロンを忘れない
- 2つ以上のスペースは全て1つのスペースと同じ  
(これも見やすさ重視)
- 改行はあってもなくても処理には関係ない (見やすさ重視で好きに)
- //以降はコメントアウト



```
sketch_may18a.ino
1 void setup() {
2   pinMode(LED_BUILTIN, OUTPUT);
3 }
4
5 void loop() {
6   digitalWrite(LED_BUILTIN, HIGH);
7   Delay(1000);
8   digitalWrite(LED_BUILTIN, LOW);
9   delay(1000)
10 }
11
12
```

Output

```
~~~~~
/private/var/folders/b3/gftgtjkd1cnczy04xj60w76r0000gn/T/.arduinoIDB
Delay(1000);
~~~~~
delay
```

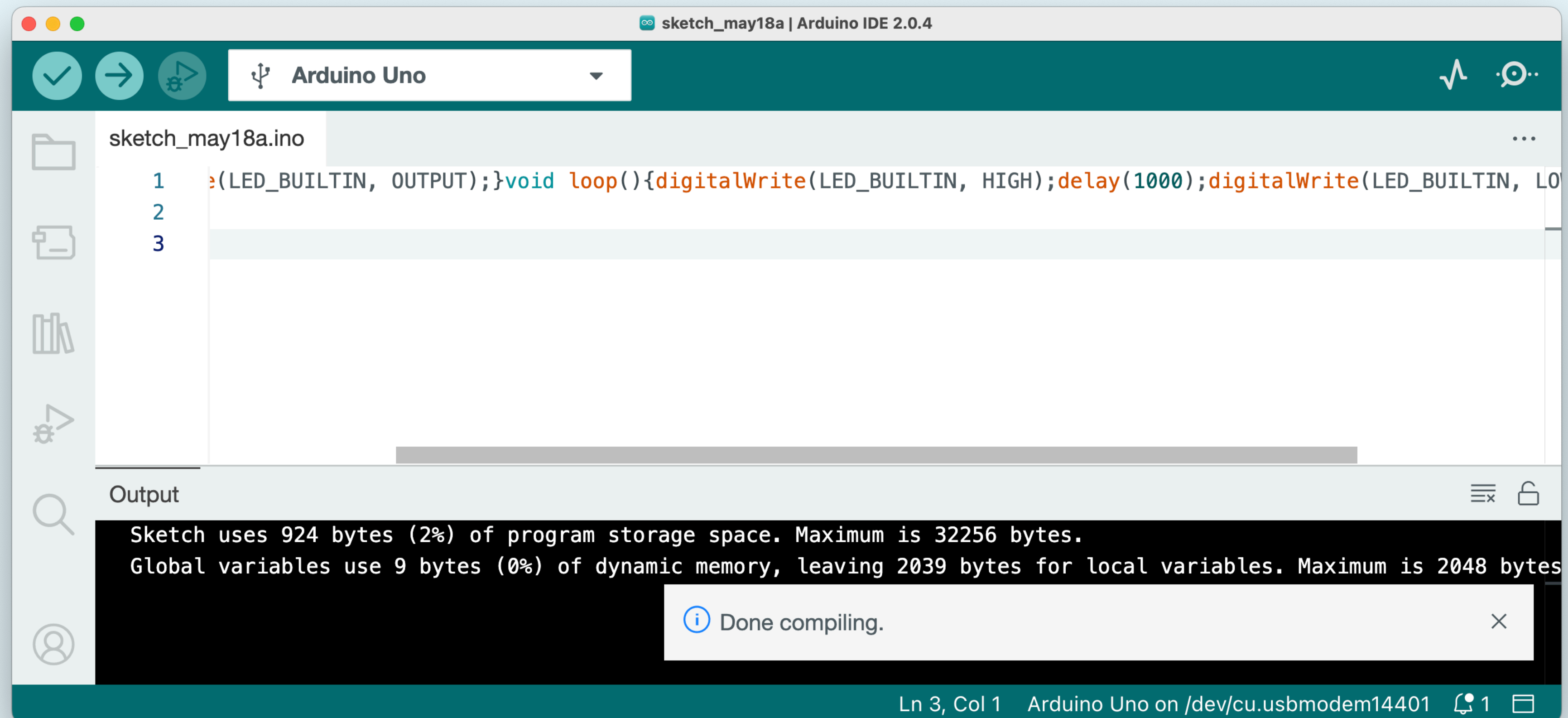
Ln 2, Col 3 Arduino Uno on /dev/cu.usbmodem14401



# 覚えておくと便利なショートカット

- `Cmd+⇐⇒` : 行頭行末へカーソルをジャンプ
- `Option+⇐⇒` : 単語単位でカーソルをジャンプ
  - `+Shift`と組み合わせて選択範囲を変える
- `Option+↑↓` : カーソルのある行/選択範囲を上下と入れ替える
- `Cmd+ /` : カーソルのある行or選択範囲をコメントアウト、コメント解除

Visual Studio Codeなど、モダンなエディタでは大体共通して使える機能



仕様を理解してれば、一見めちゃくちゃ読みづらくても動く

# 数値をいじってみよう

```
void setup() {  
    pinMode(LED_BUILTIN, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(500);  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(100);  
}
```

Delayの数値はミリ秒単位！

入力：センサーを使ってみよう



# シャープ 測距モジュール

## GP2Y0A02YK



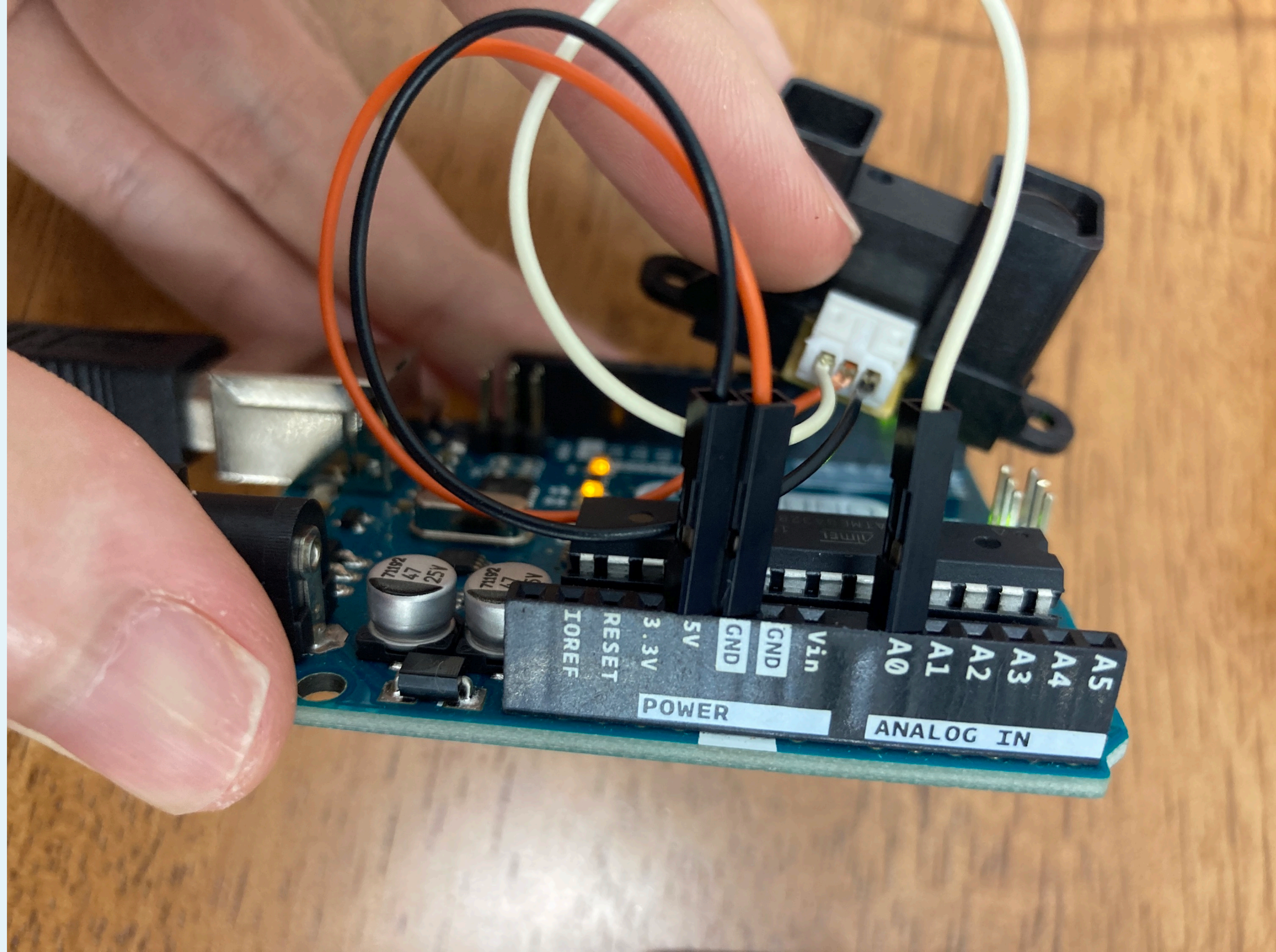
20cm~150cm測れる

白：距離に応じた電圧出力

オレンジ：GND

黒：電源(+5V)







```
void setup() {  
    pinMode(A0, INPUT);  
    Serial.begin(9600);  
}  
  
void loop() {  
    int distance = analogRead(A0);  
    Serial.println(distance);  
    delay(20);  
}
```

書き込んだらここをクリック





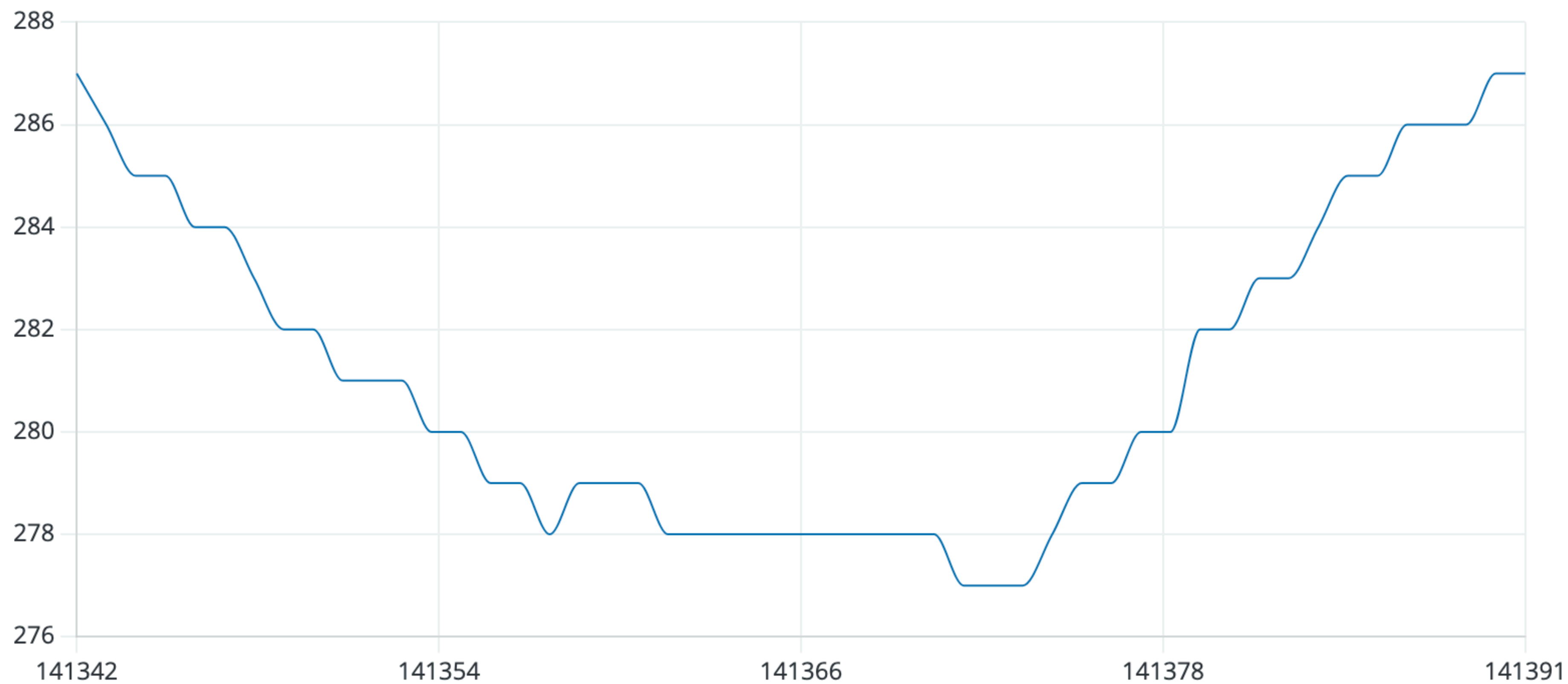


/dev/cu.usbmodem144401

☒ value 1

Interpolate ☒

STOP



Type Message

SEND

New Line

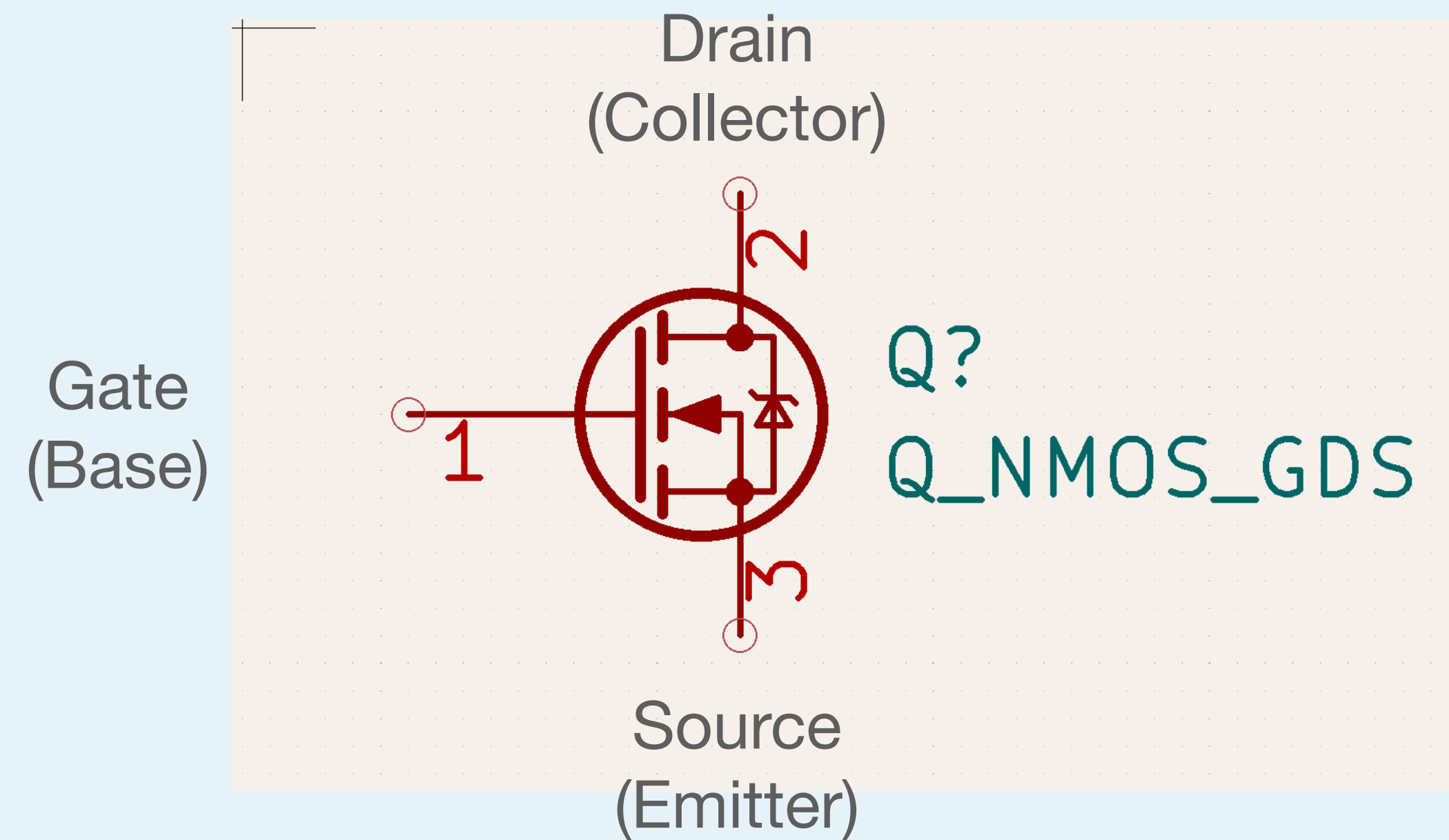


9600 baud



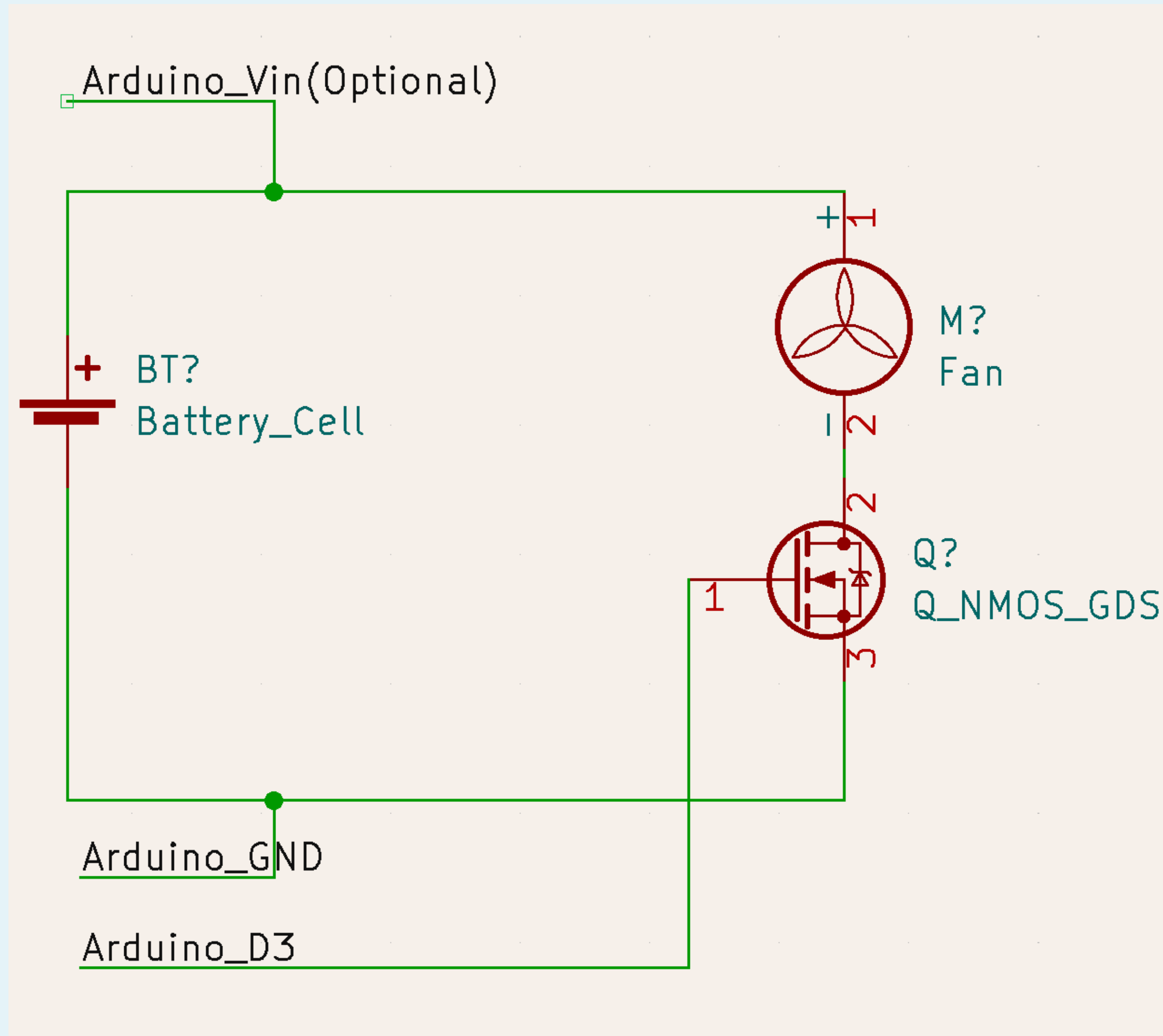
出力:電流消費の激しい物を扱う

# K2232 MOSFET



<https://akizukidenshi.com/catalog/g/gI-02414/>

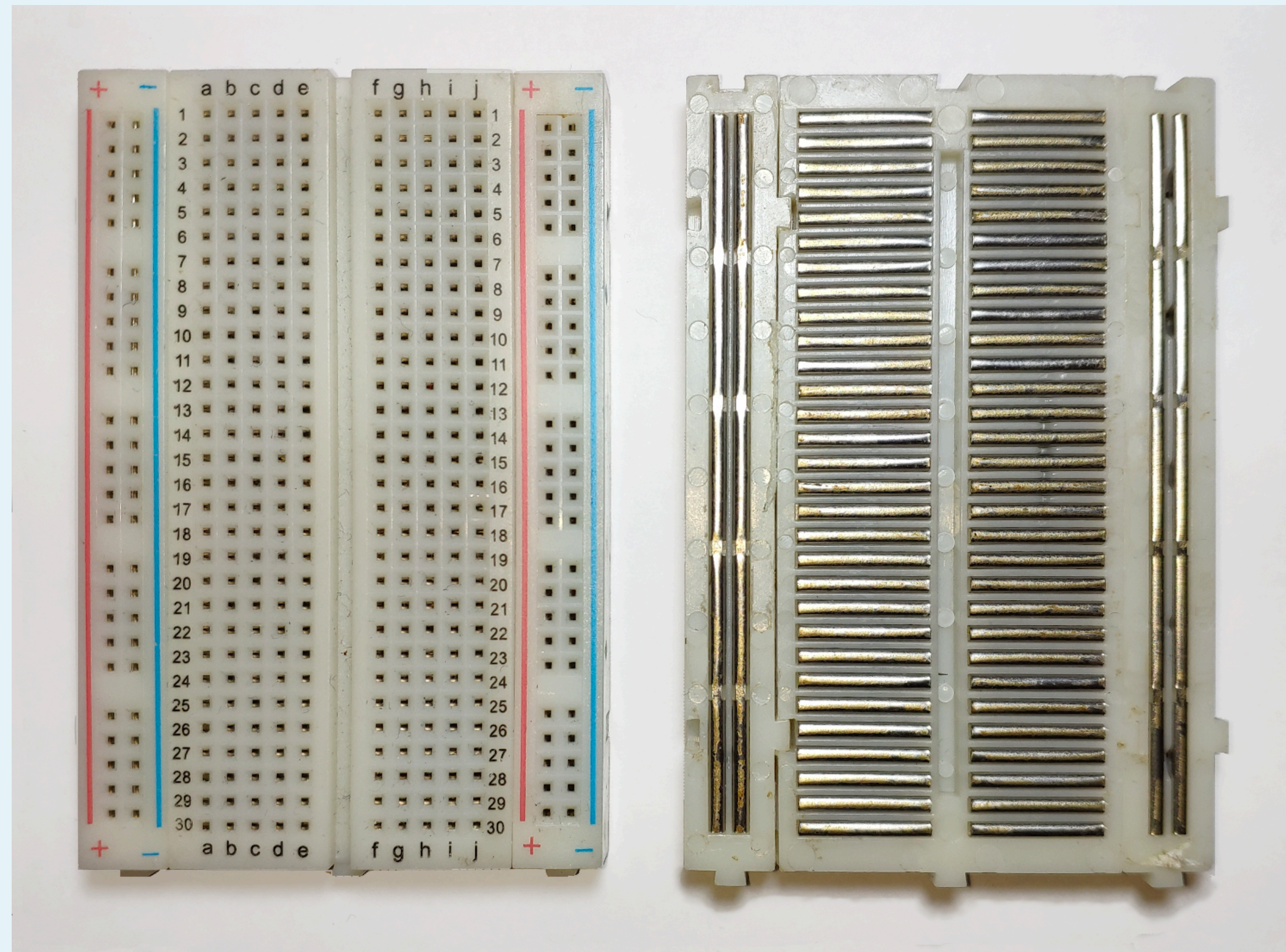
トランジスタとほぼ同じだが、ゲートからソースには電流が流れない



※ファンから逆流する電流について考慮していないので本当は不適切です、今度補足

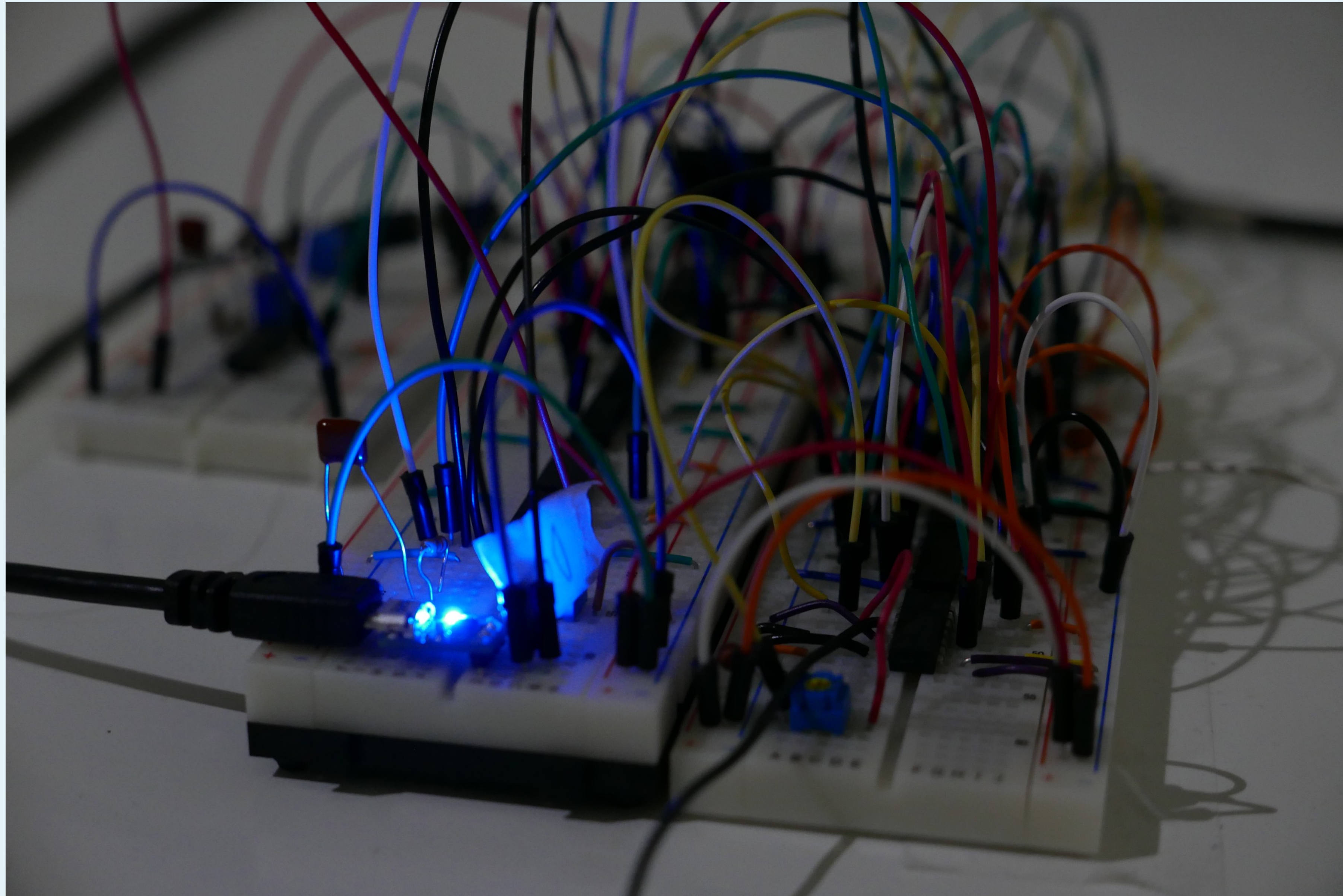


# ブレッドボード



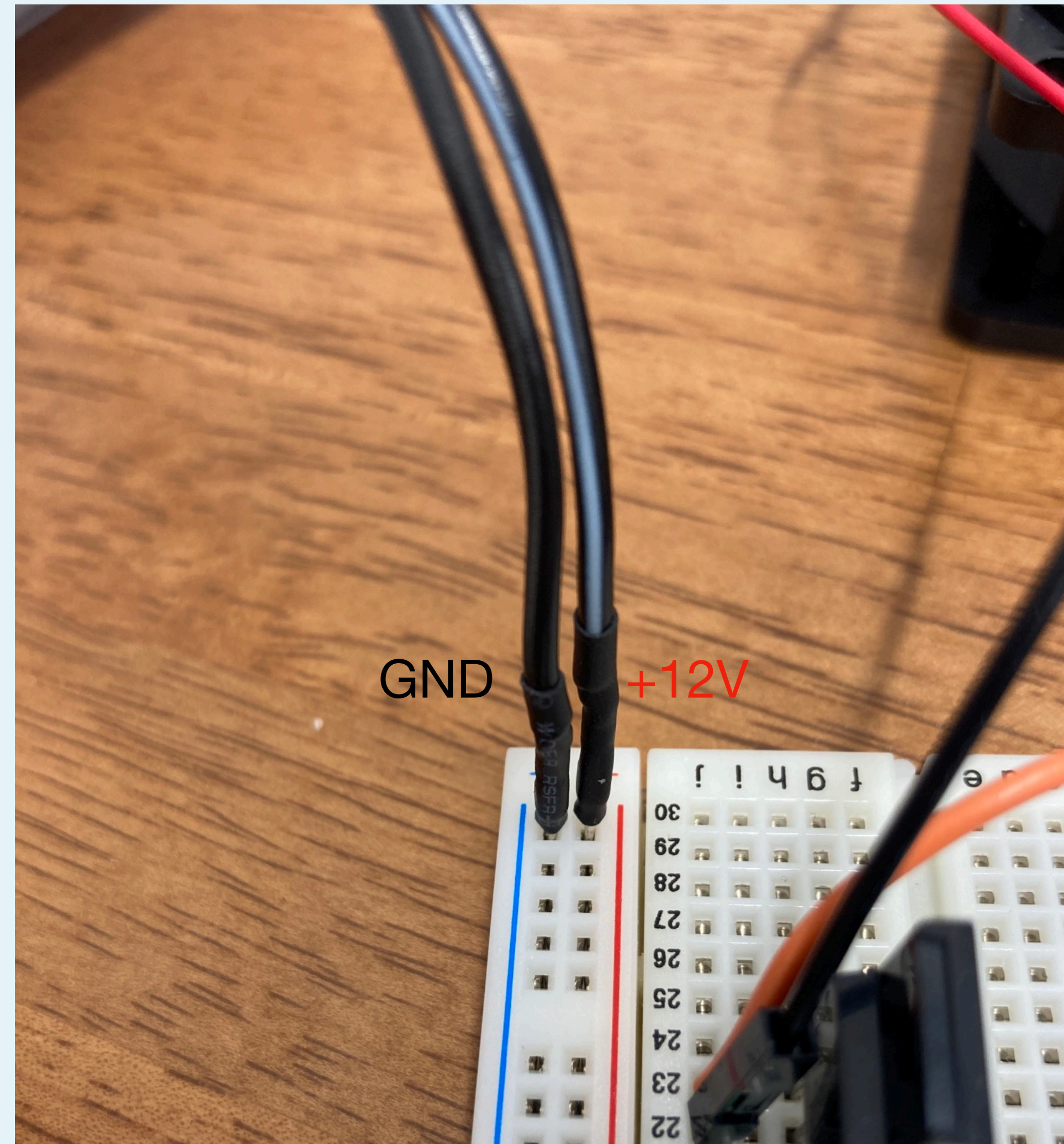
- はんだ付けなしで簡単に回路を組みたい人用





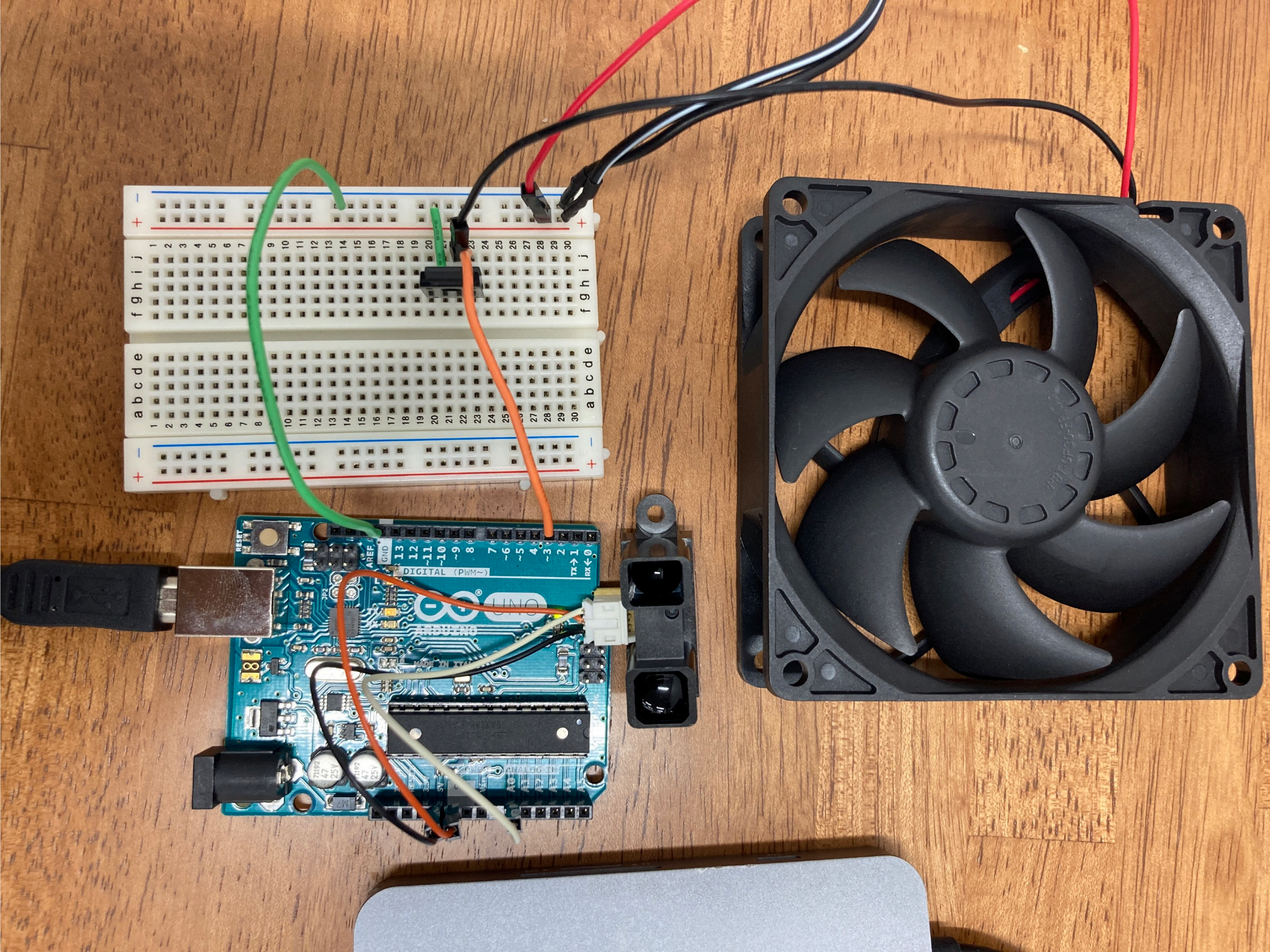
<https://matsuuratomoya.com/works/adm/>



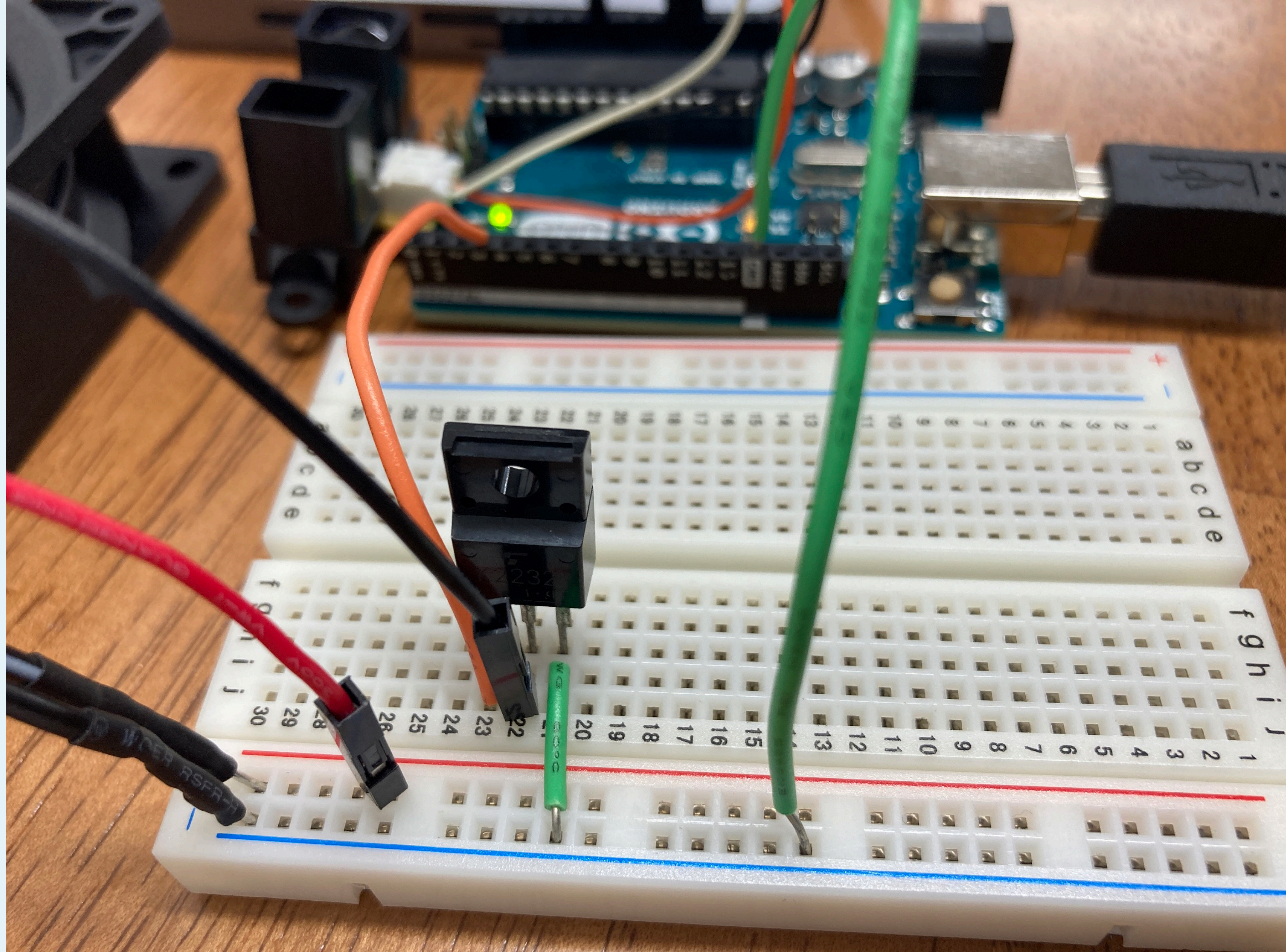


今回は、白い線が入ってる方が+  
(場合によりけりなので、きちんとテスターで確かめること)







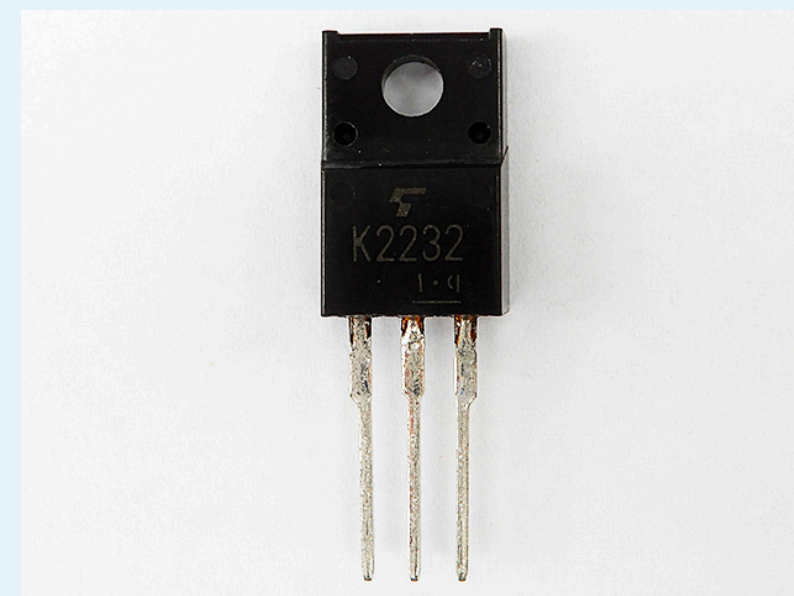




AC  
アダプタ

+

-



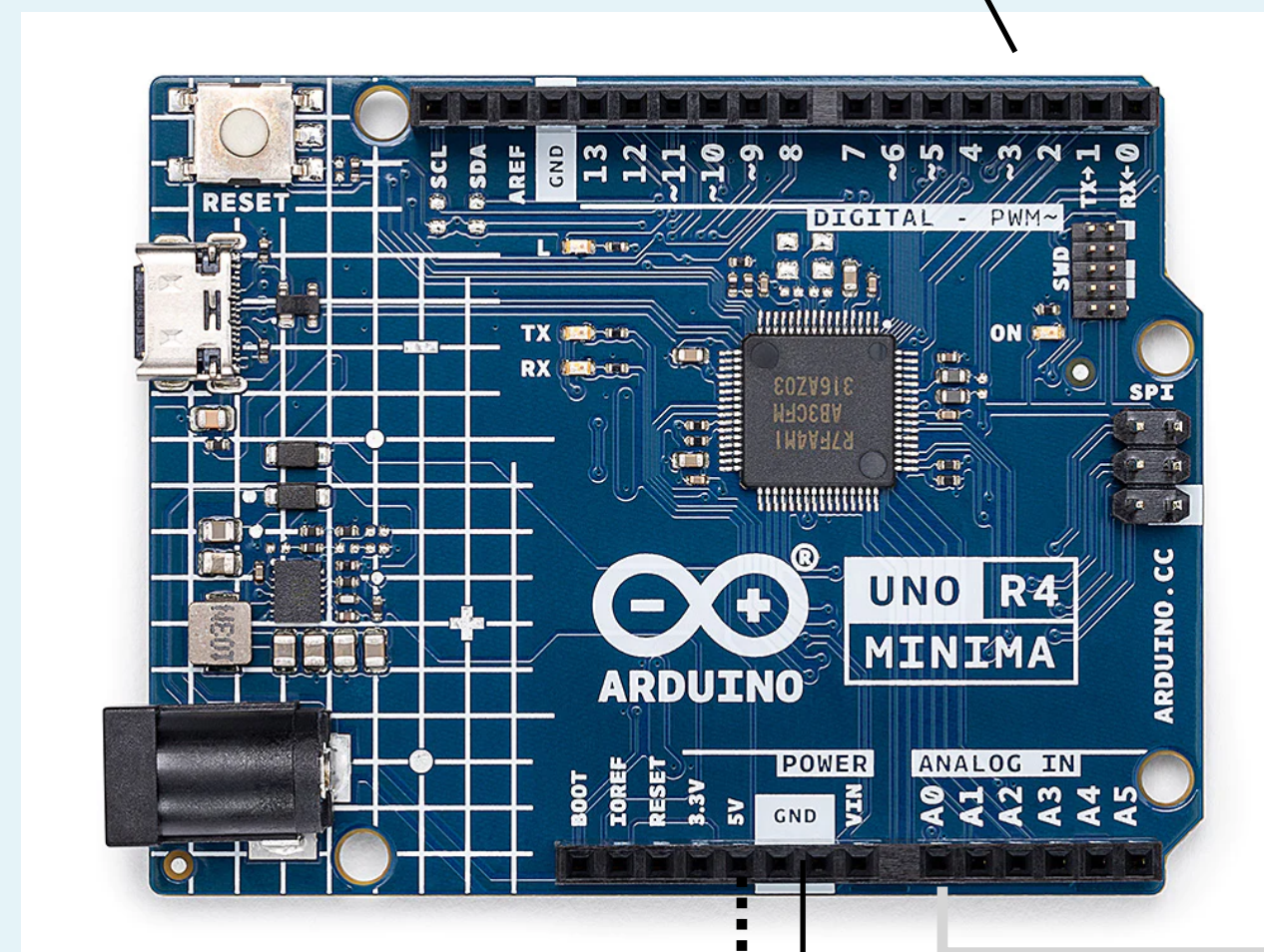
G D S

3

+

-

ファン



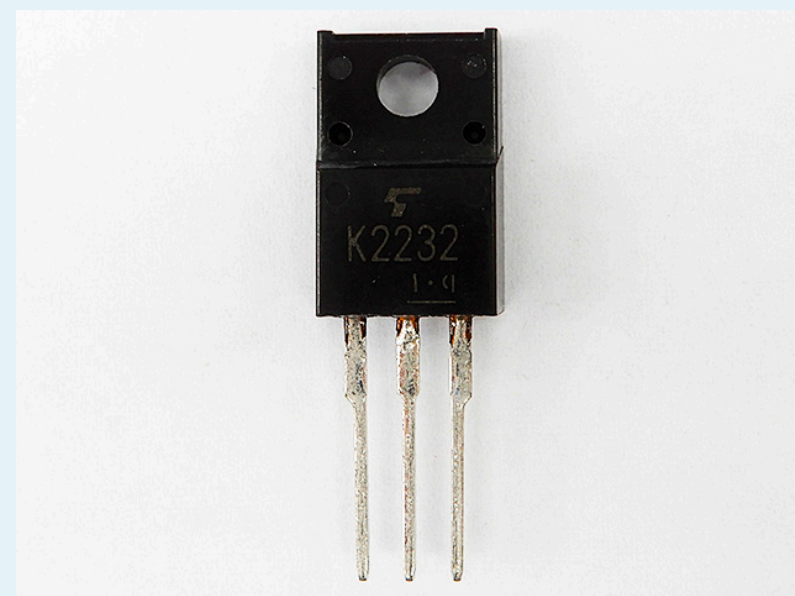
OUT

5V

GND

光センサ



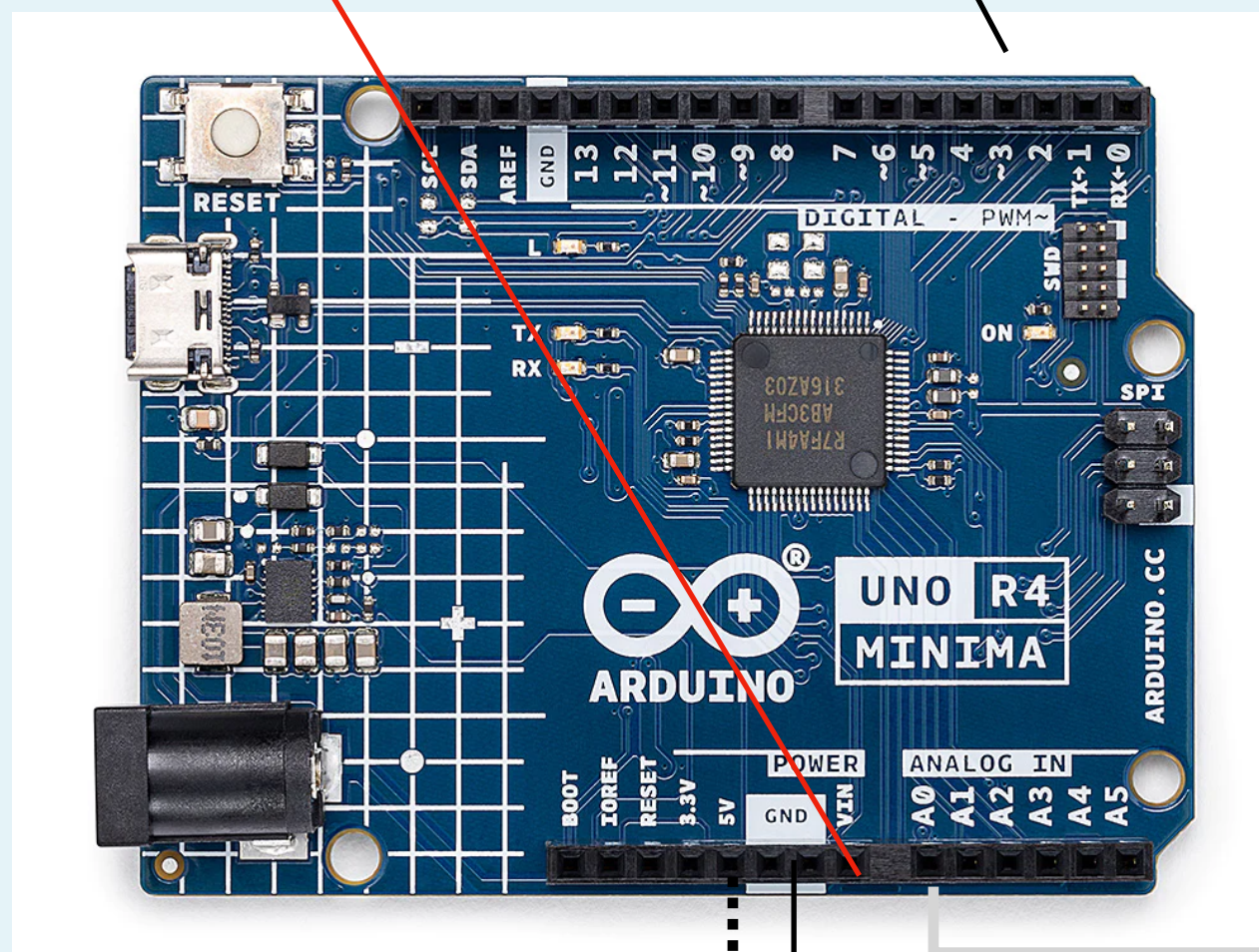


G D S

USB電源なしで駆動したければ、  
ACアダプタをVinにもつなぐ

AC  
アダプタ

ファン



3

OUT  
5V  
GND

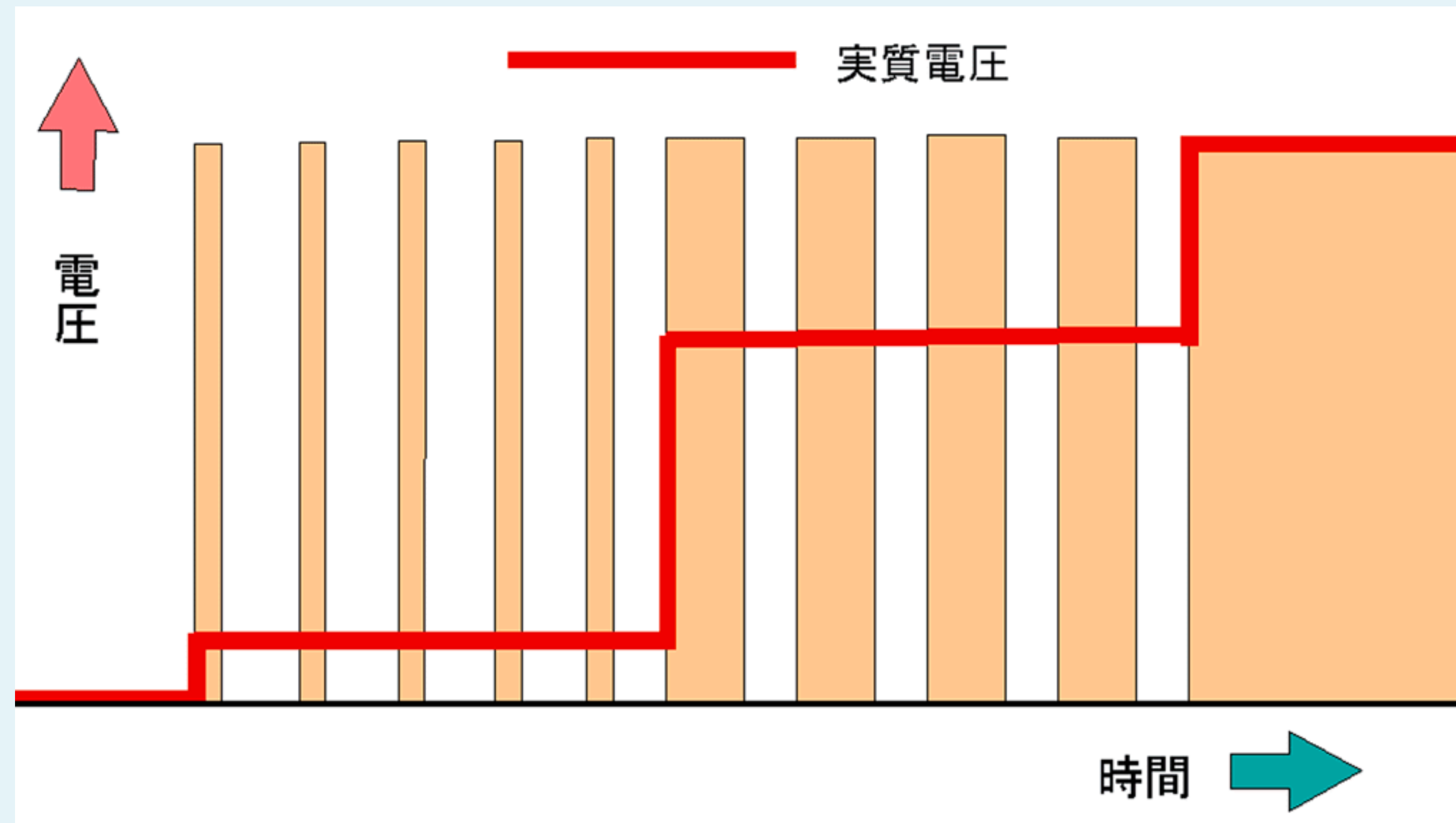
光センサ

```
void setup() {  
    pinMode(A0, INPUT);  
    pinMode(3, OUTPUT);  
    Serial.begin(9600);  
}  
  
void loop() {  
    int distance = analogRead(A0);  
    Serial.println(distance);  
    if(distance > 400) {  
        digitalWrite(3, HIGH);  
    } else {  
        digitalWrite(3, LOW);  
    }  
    delay(20);  
}
```

If文でオンオフ切り替え

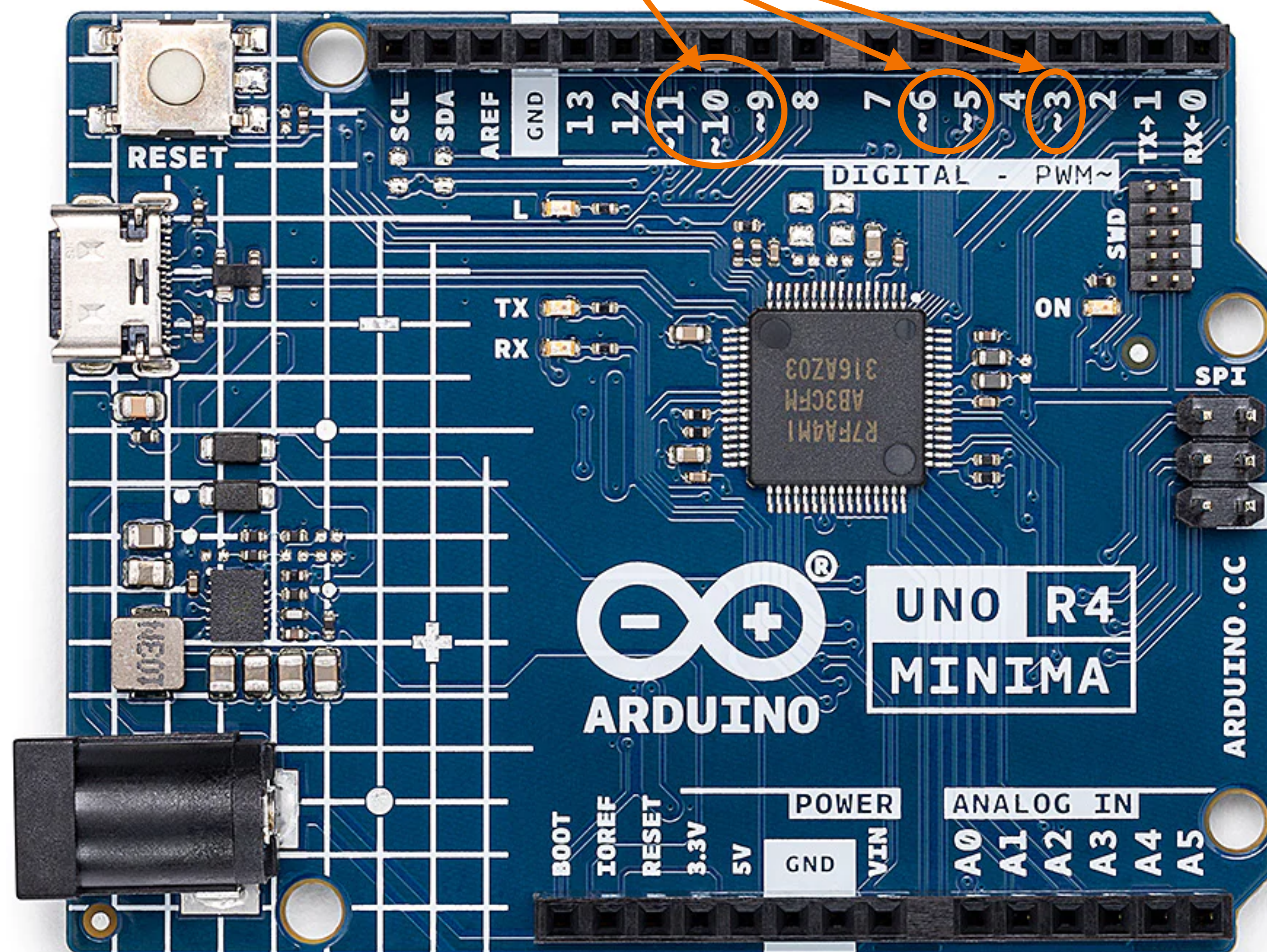


# PWM制御





analogWrite()でPWM出力が使用できるピン (~がついたもの)



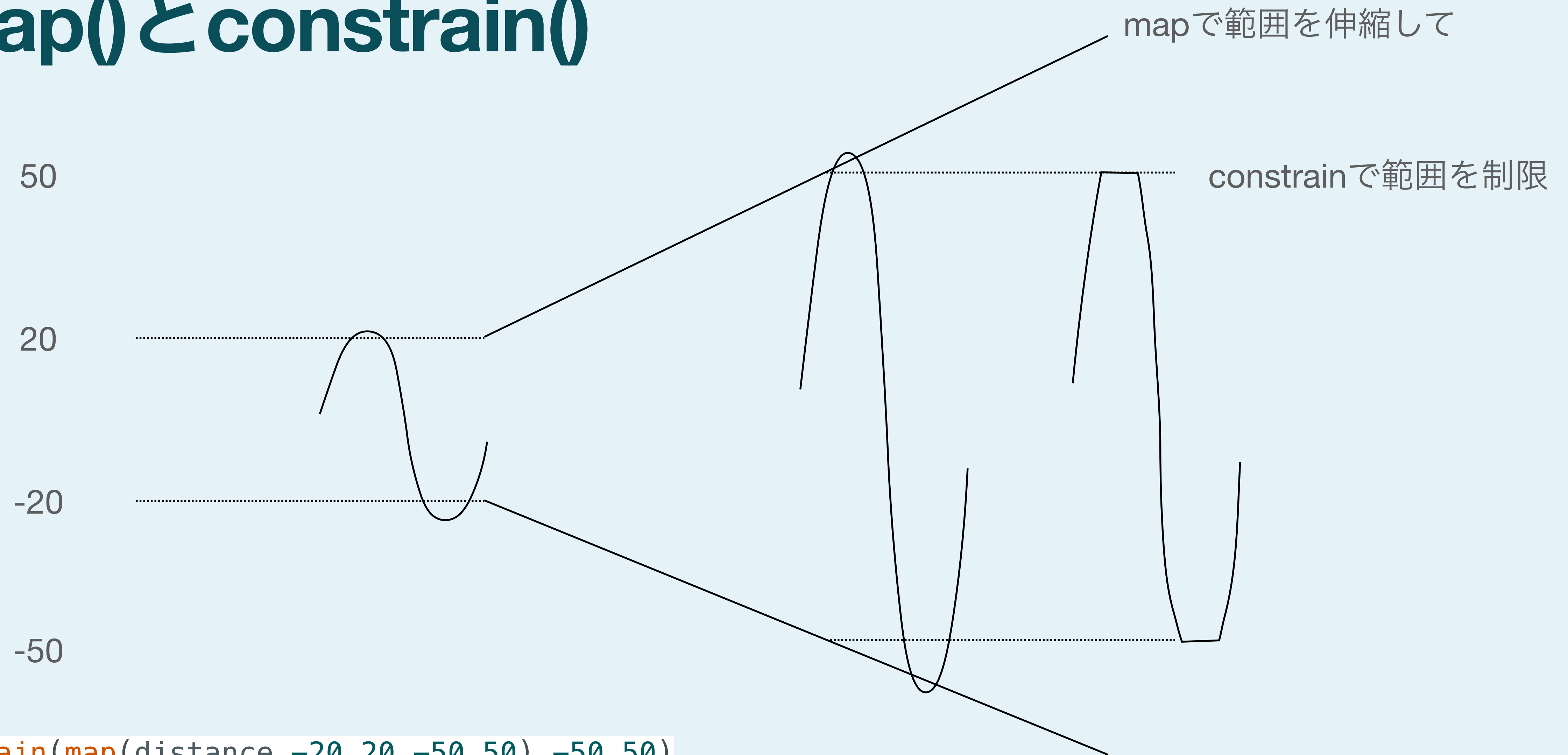


```
void setup() {  
  pinMode(A0, INPUT);  
  pinMode(3, OUTPUT);  
  Serial.begin(9600);  
}  
  
void loop() {  
  int distance = analogRead(A0);  
  int output = constrain(map(distance, 0, 500, 0, 255), 0, 255);  
  analogWrite(3, output);  
  Serial.print(distance);  
  Serial.print(',');  
  Serial.println(output);  
  delay(40);  
}
```

analogWriteを使って連続的に変化させる  
出力範囲は0~255なので範囲の変換が必要



# map() と constrain()



```
constrain(map(distance, -20, 20, -50, 50), -50, 50)
```

に対して、入力-30~30ぐらいの値が入ってきた時のイメージ



```
Serial.println(output);
```



```
Serial.print(output);  
Serial.print('\n');
```

Serial.println()はSerial.print()に改行(\n)の出力を加えたもの  
1行の中に複数の変数を出力したいときはprintの方を使おう



```
void setup() {  
  pinMode(A0, INPUT);  
  pinMode(3, OUTPUT);  
  Serial.begin(9600);  
}
```

```
void loop() {  
  int distance = analogRead(A0);  
  int output = constrain(map(distance, 0, 500, 0, 255), 0, 255);  
  analogWrite(3, output);  
  Serial.print(distance);  
  Serial.print(', ');  
  Serial.println(output);  
  delay(40);  
}
```

1行ごとにカンマ区切りでシリアルにprintすると、  
シリアルプロッタで複数の値を同時にプロットできる