# コードとデザイン

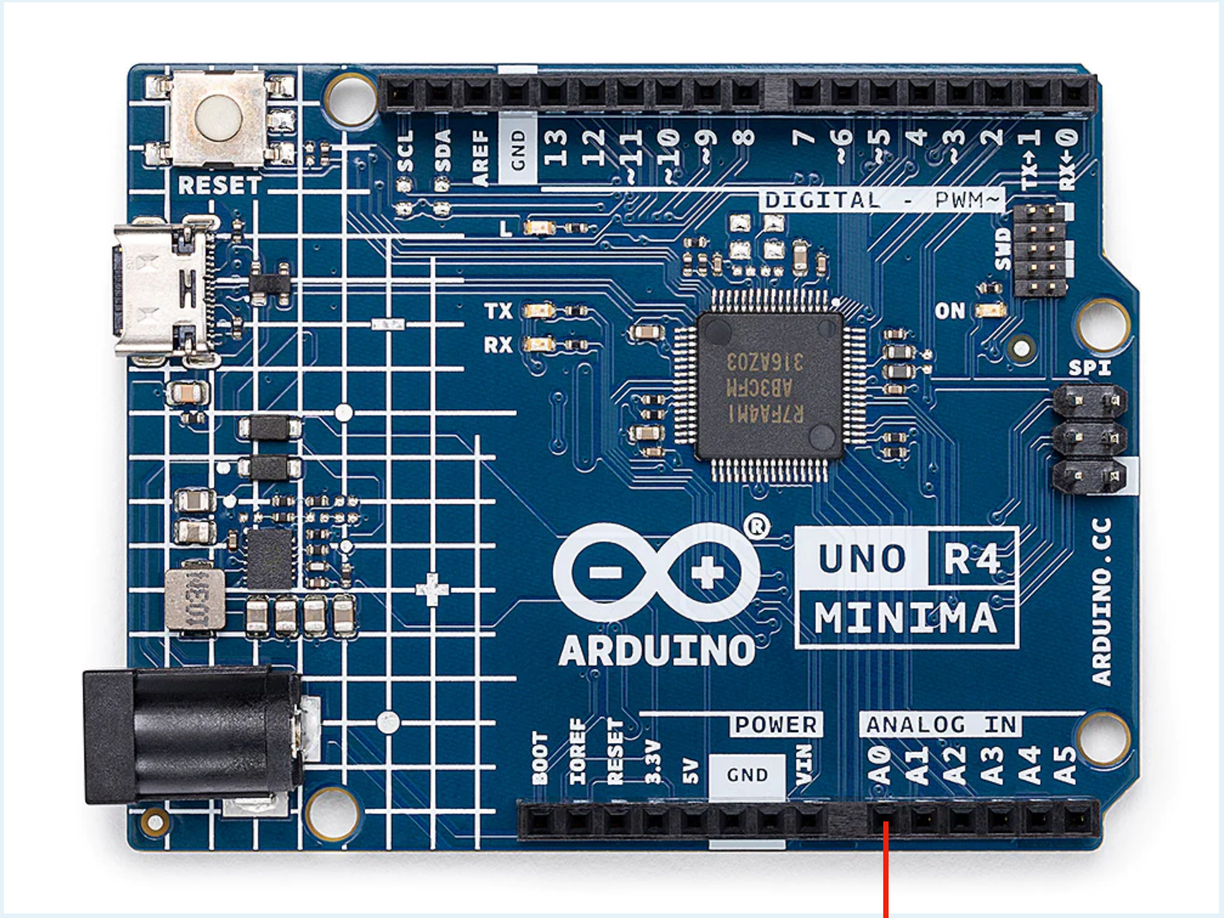
## 東京藝術大学 芸術情報センター開設科目 金曜4-5限 第8週

2024.06.07 松浦知也（matsura.tomoya@noc.geidai.ac.jp teach@matsuuratomoya.com）

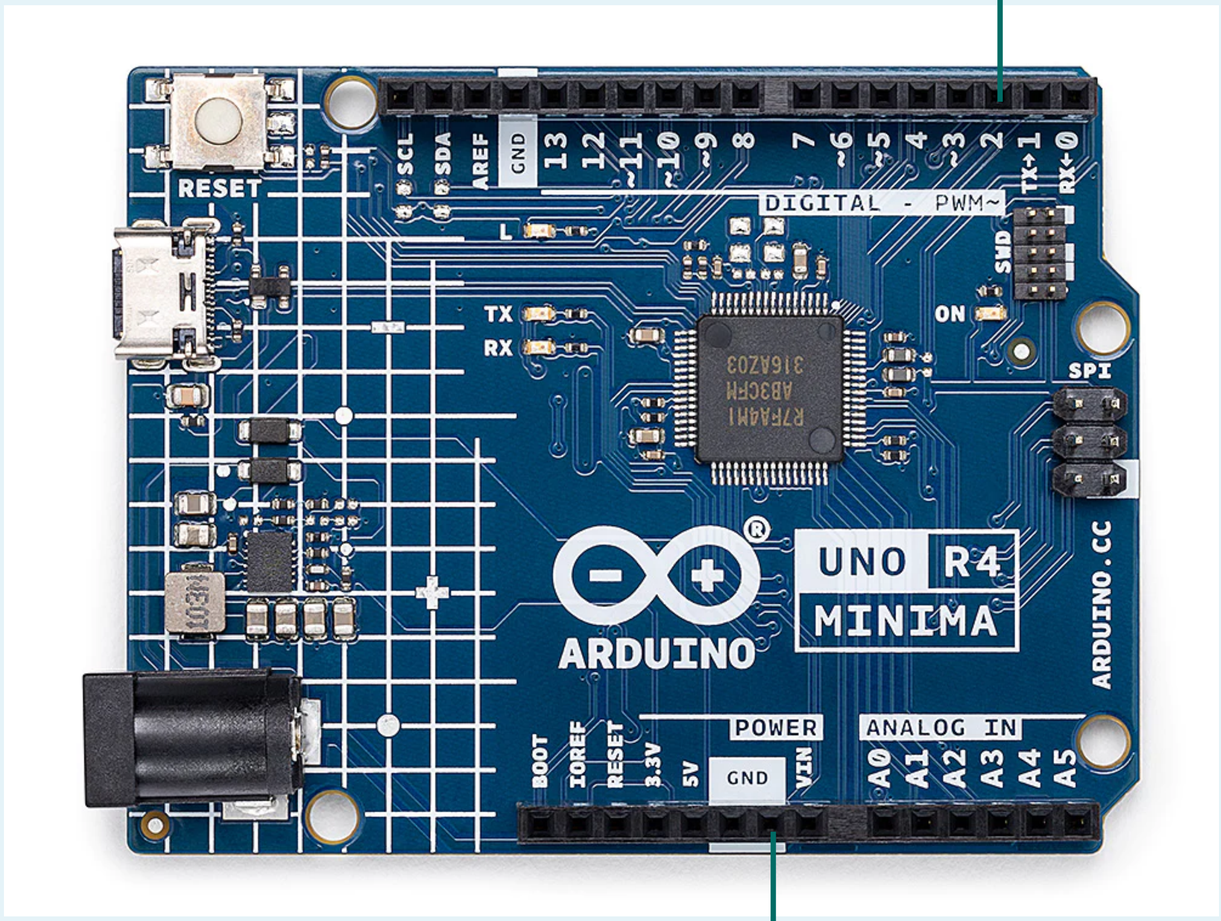
# 本日のスケジュール

- Arduinoでライブラリを使ってみよう
- 自分だけのコンピューター用入力装置を作ってみよう

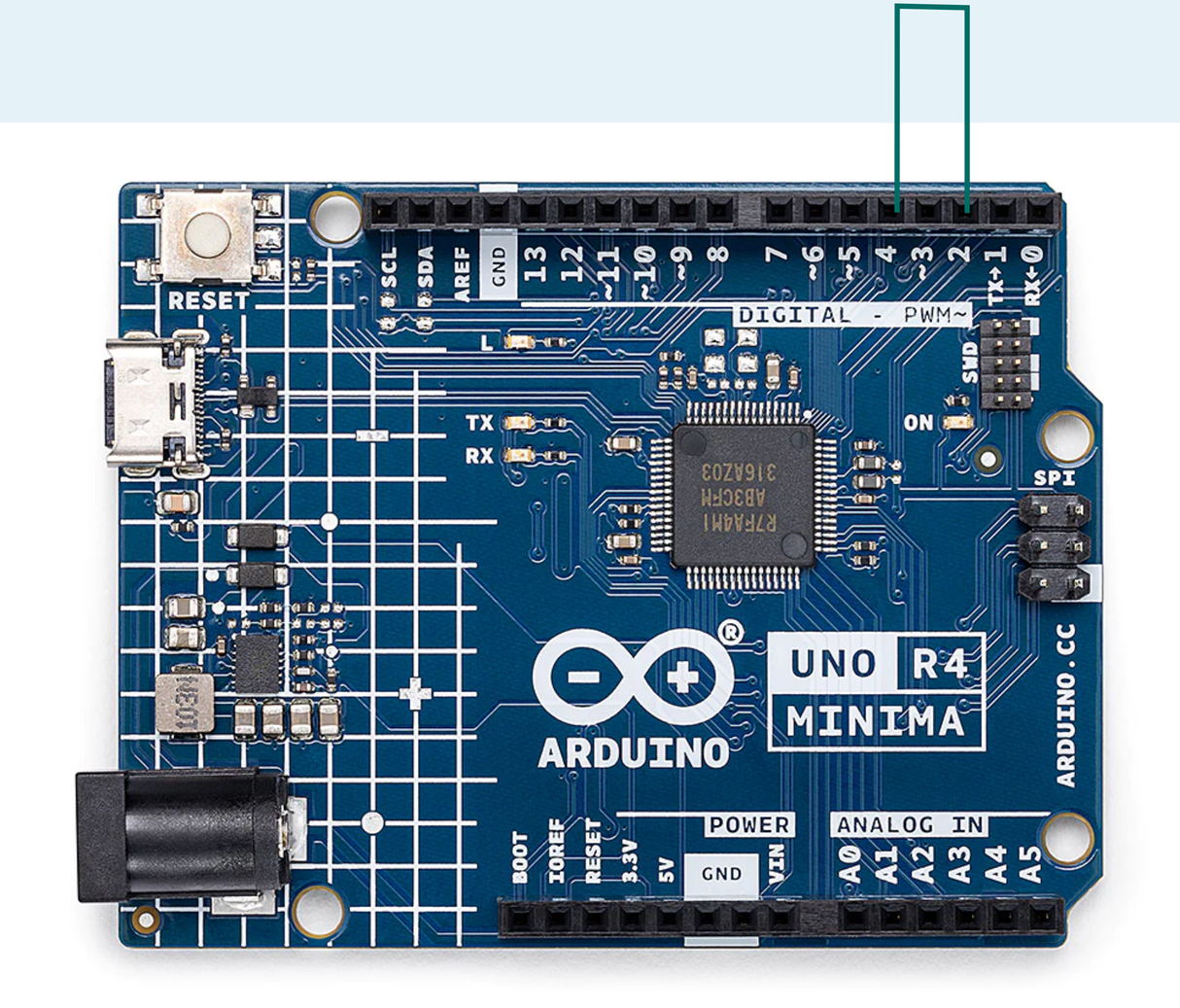
# Arduino これだけはやるな（R4版）




- 5V以上、0V以下の電圧を入出力ピンに加えない
- ピンが破壊される
- 普通にプログラムを書き込めるけど、チップが異常に熱くなったりする
- 5V以上の電圧を加えるのは電源供給用のVINピン（7~24V）のみ




- pinMode(OUTPUT)になっているピンをGNDとショートしてはいけない
- ピンから過電流が流れる

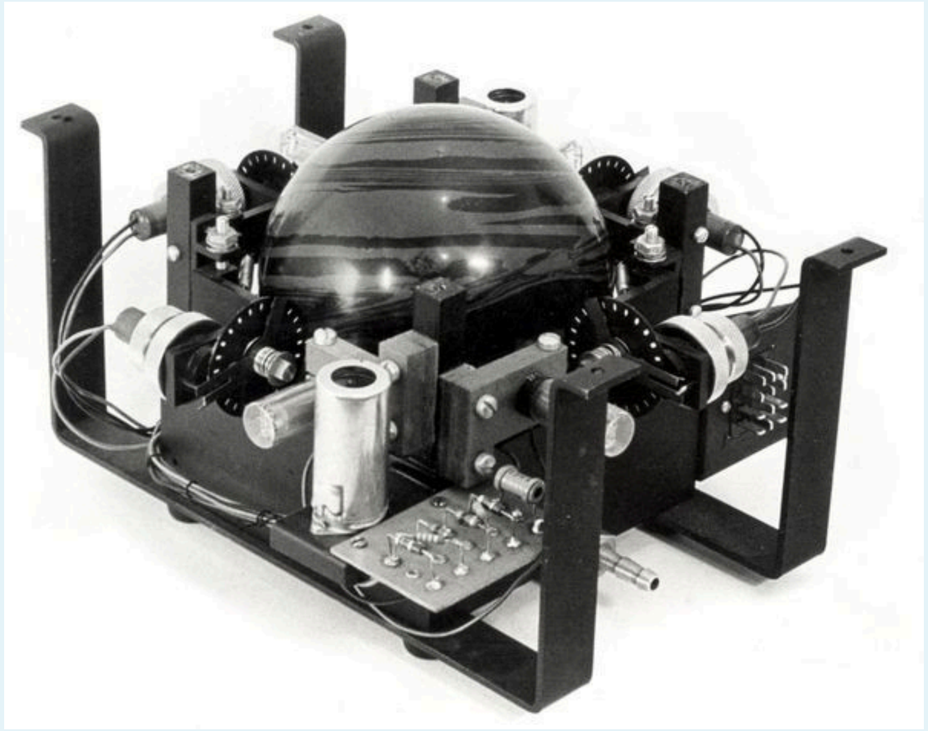



- 両方OUTPUTの状態で、片方がHIGH、もう片方がLOWの状態で、ピン同士がショートしても同じことが起きるので注意

# おかしい時に起こりがちなこと

- USBを挿したのに電源ランプが光らない（どこかでショートしている）
- USBを挿した瞬間PC側で「このアクセサリは使用できません」みたいなエラーが出る（やはりショートしていて電流を過剰に消費している）
- 変な匂いがする（どこかのパーツが過熱している）

# 入力を考える：雑なマウスを作る

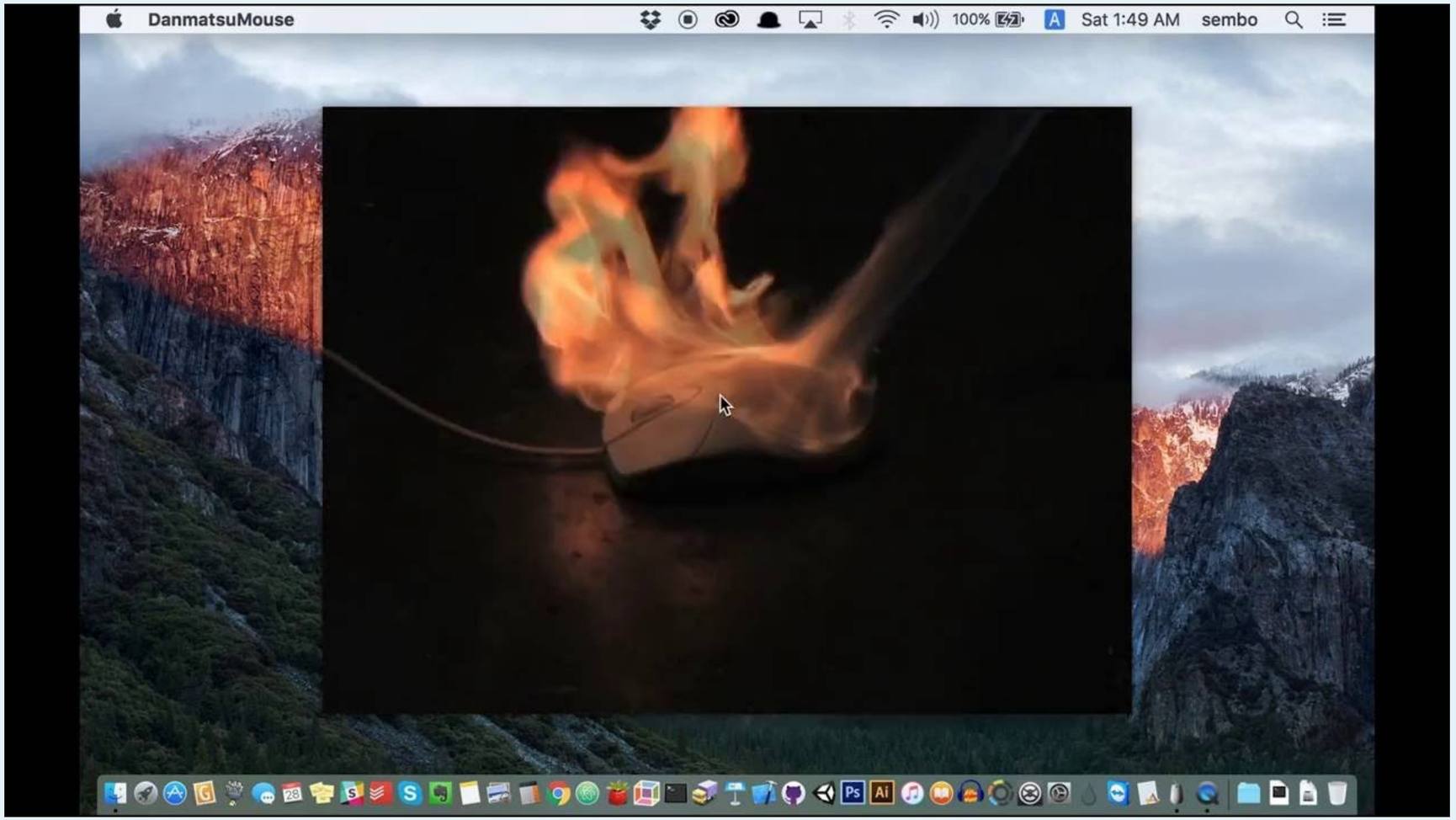


DATAR trackball(1952) - Computer History Museum, Courtesy of J. Vardalas
https://www.computerhistory.org/revolution/input-output/14/350/1881

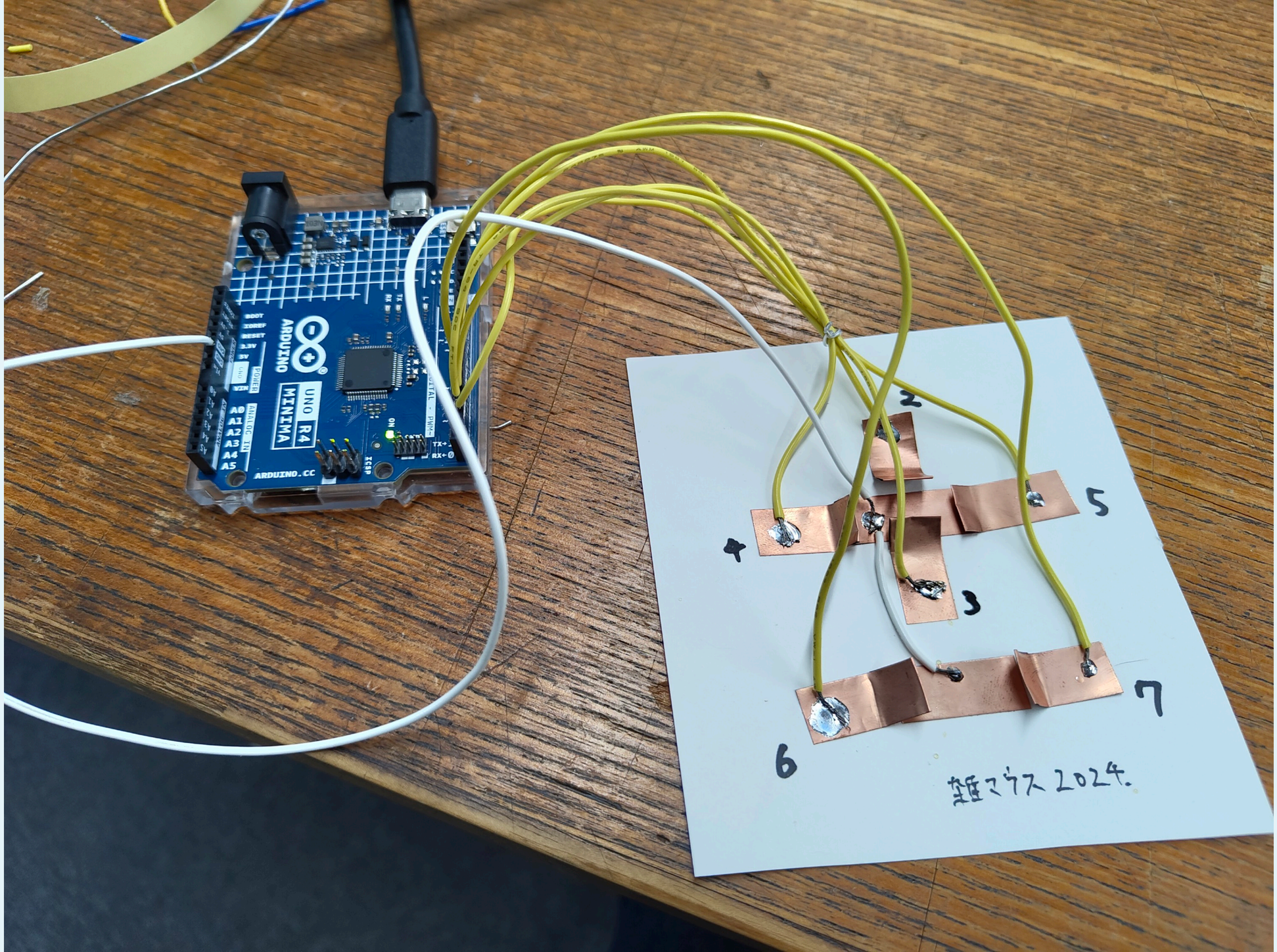
Prototype Engelbart mouse (replica), Computer History Museum, © Mark Richards

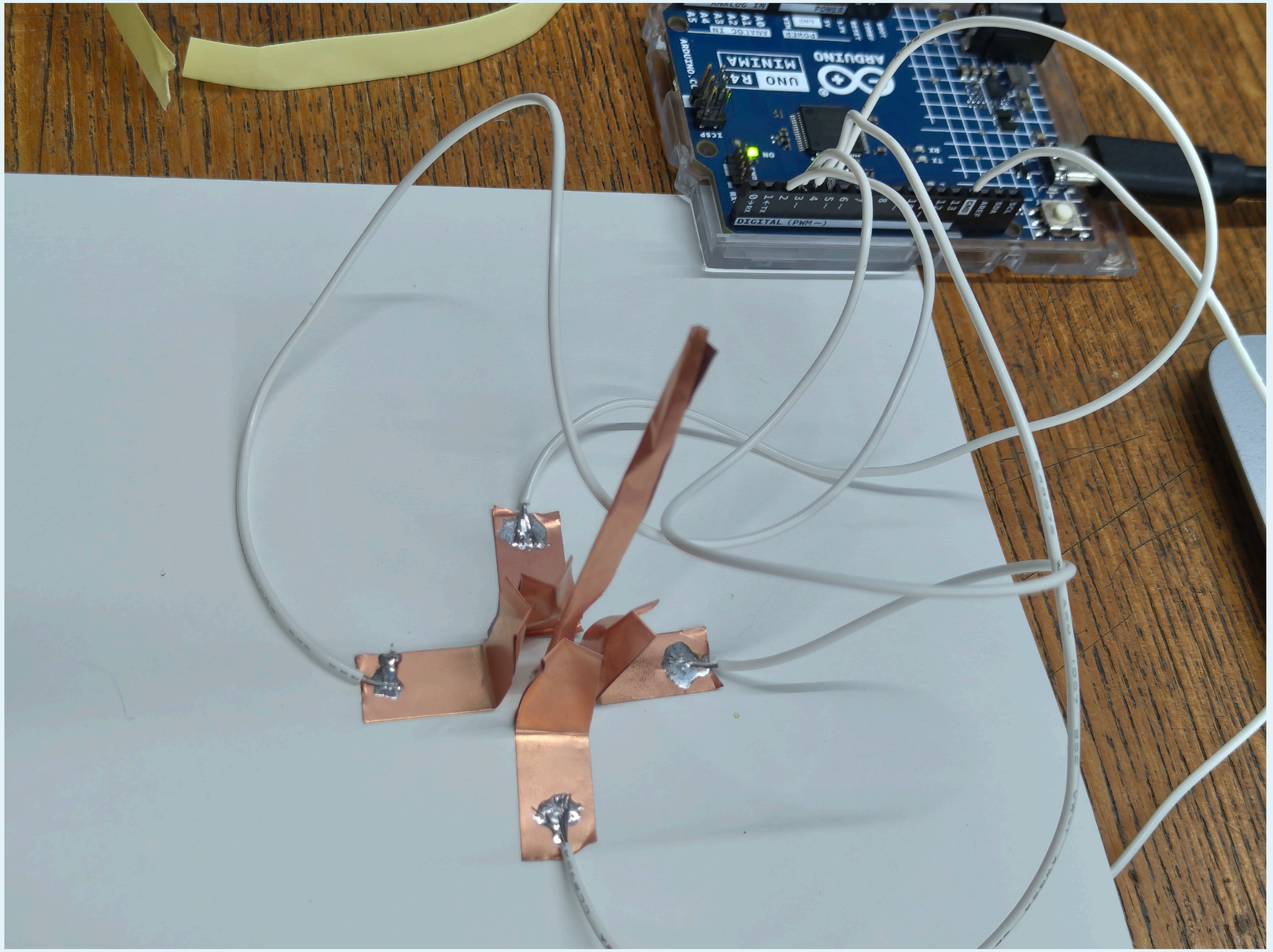
https://www.computerhistory.org/revolution/input-output/14/350/1546?position=3


1968 "Mother of All Demos" by SRI's Doug Engelbart and Team
https://www.youtube.com/watch?v=B6rKUf9DWRI



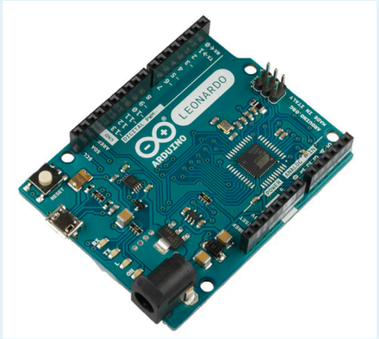

DanmatsuMouse(2007), exonemo,
http://exonemo.com/works/danmatsumouse/?ja

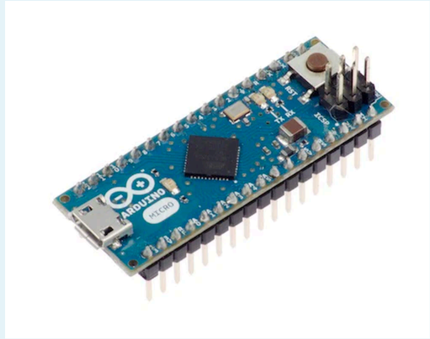

ARDUINO
UNO R4
MINIMA
ARDUINO.CC
2
5
4
3
7
6
雑さうス 2024.

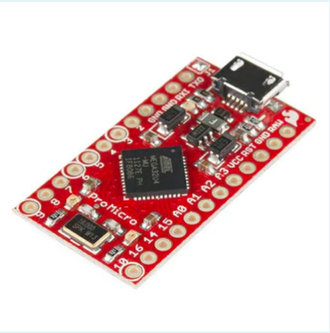

ARDUINO
UNO R4
MINIMA
DIGITAL (PWM~)

# Arduinoをマウスにする

- Arduino Uno R4はHIDライブラリを使用することで、USBマウスやキーボードとして認識させられる
- Arduino Uno R3以前はNG, R4以外だとArduino Leonardoというシリーズや、Arduino Micro、Pro Micro（非公式のバリエーション）など、ATmega32u4というCPUが乗っているものでのみ可能




https://kata0510.github.io/Lily58-Document/

実際、自作キーボードはPro Microで作られてるものが多い

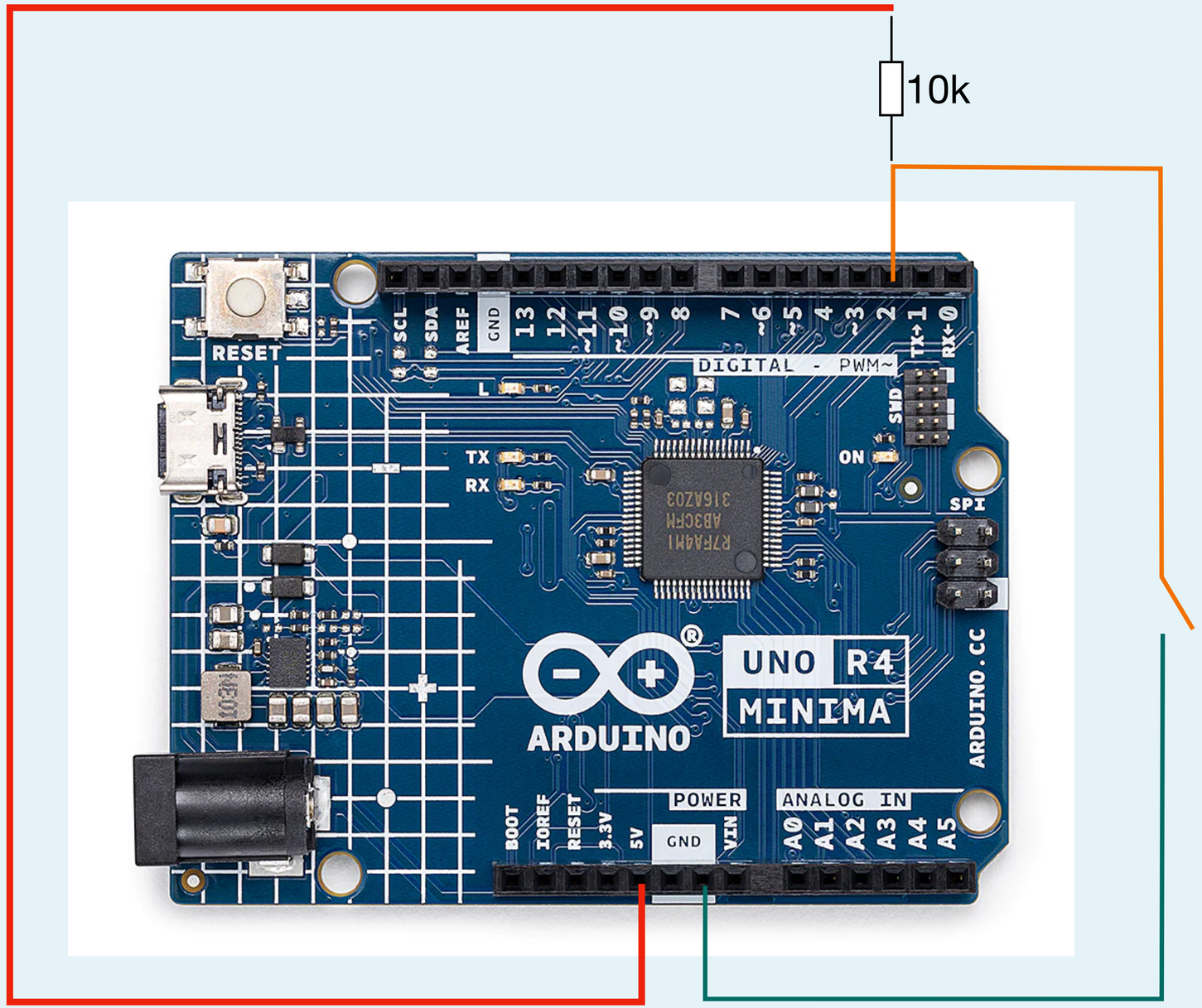
# マウスをArduino側から操作

```ino
#include <Mouse.h>

void setup() {
  Mouse.begin();
  delay(1000);
}

void loop() {
  Mouse.move(10,10);
  delay(1000);
  Mouse.move(-10,-10);
  delay(1000);
}
```

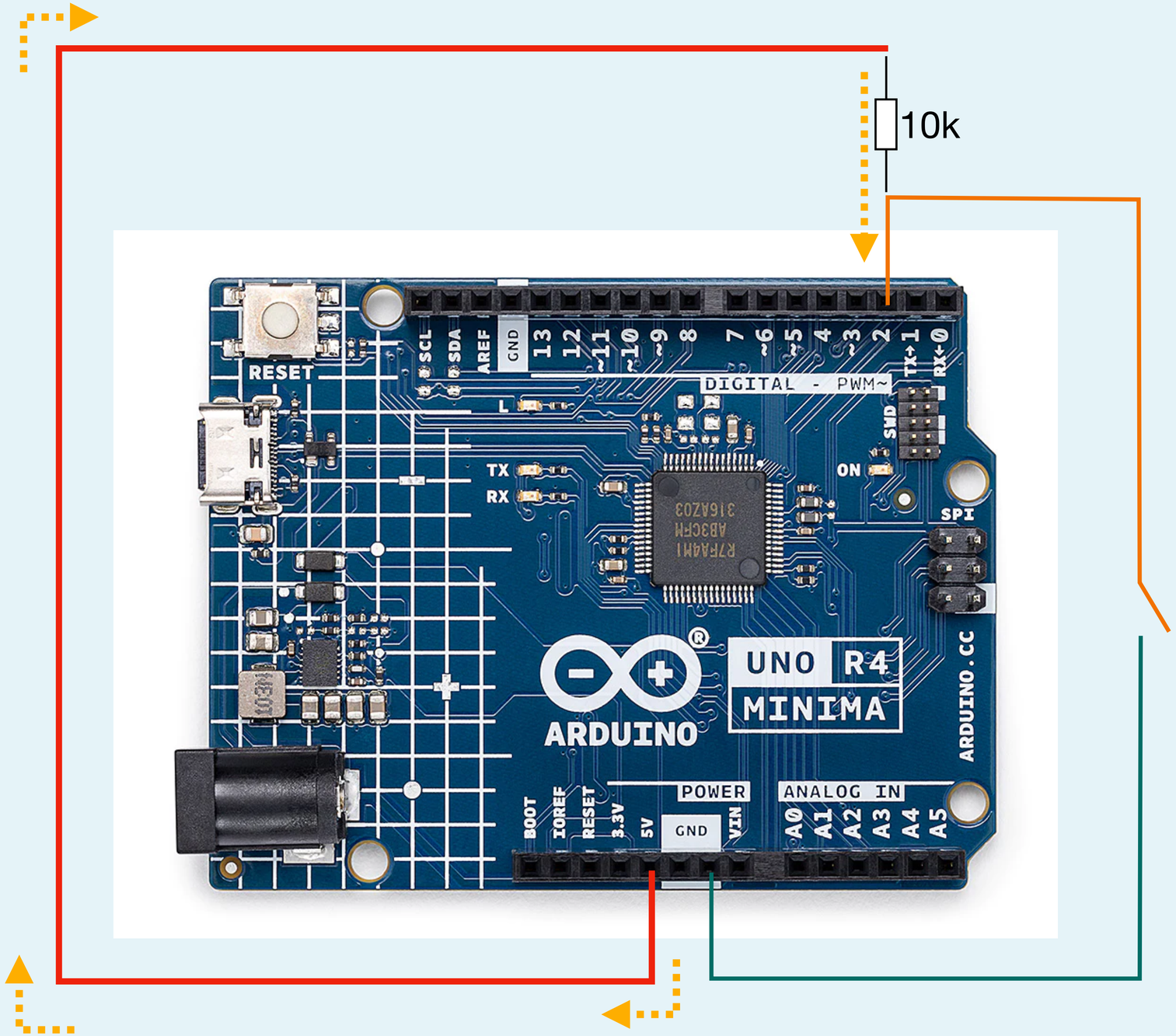
mouse_jiggle.ino

# シンプルなスイッチの制作方法

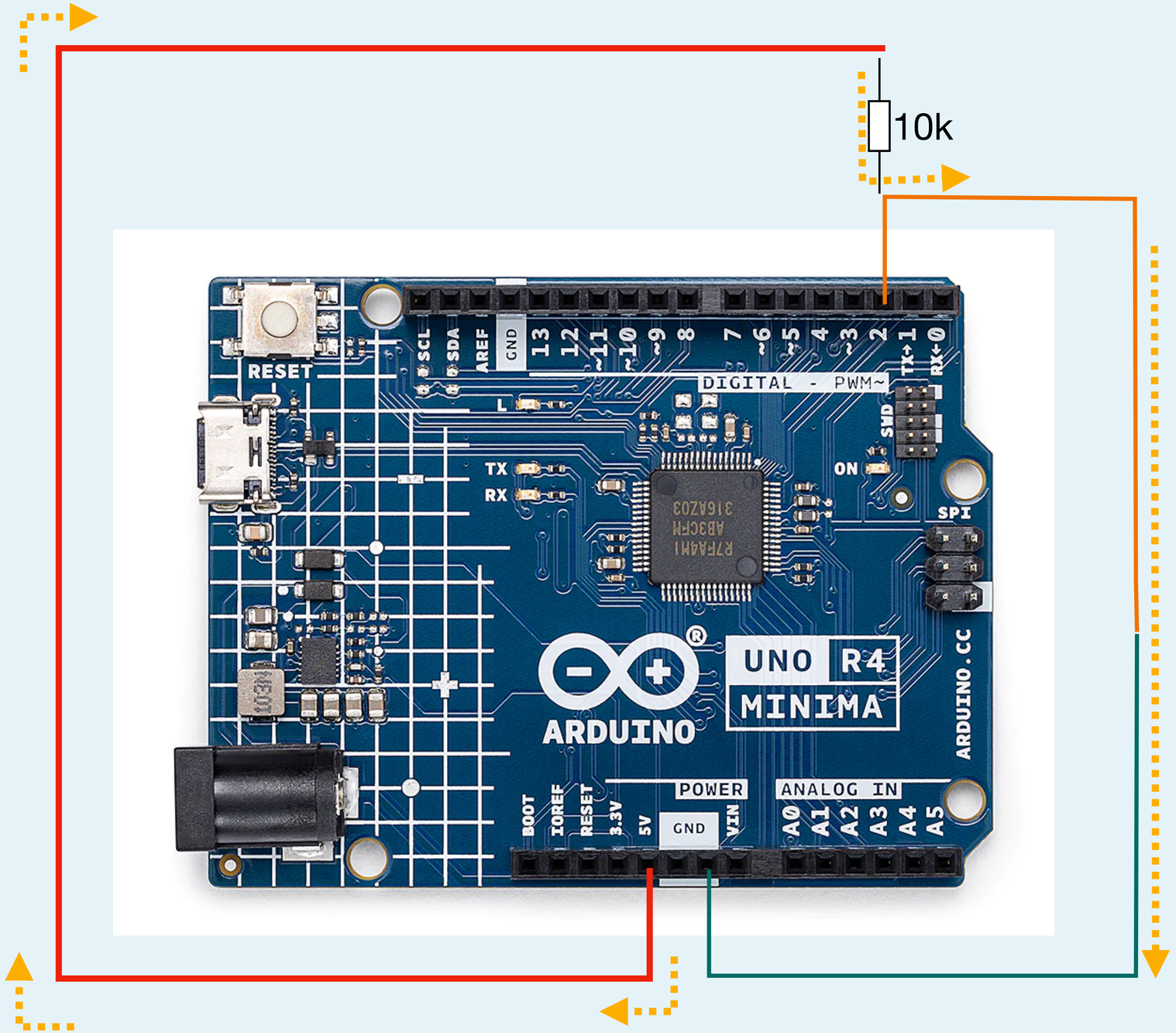
- pinMode(2,INPUT_PULLUP)のようにすると、2番ピンをワイヤー1本でスイッチとして使用できる
- 2番ピンから伸ばしたワイヤーを、GNDとショートさせると0(LOW)、してないときは1(HIGH)として判定
- INPUT_PULLUPしていないピンをGNDに直接繋げると過電流になるかもので、配線前にプログラムを書き込むこと
  - 前のプログラムでピンがOUTPUTのままになってるかも・・・



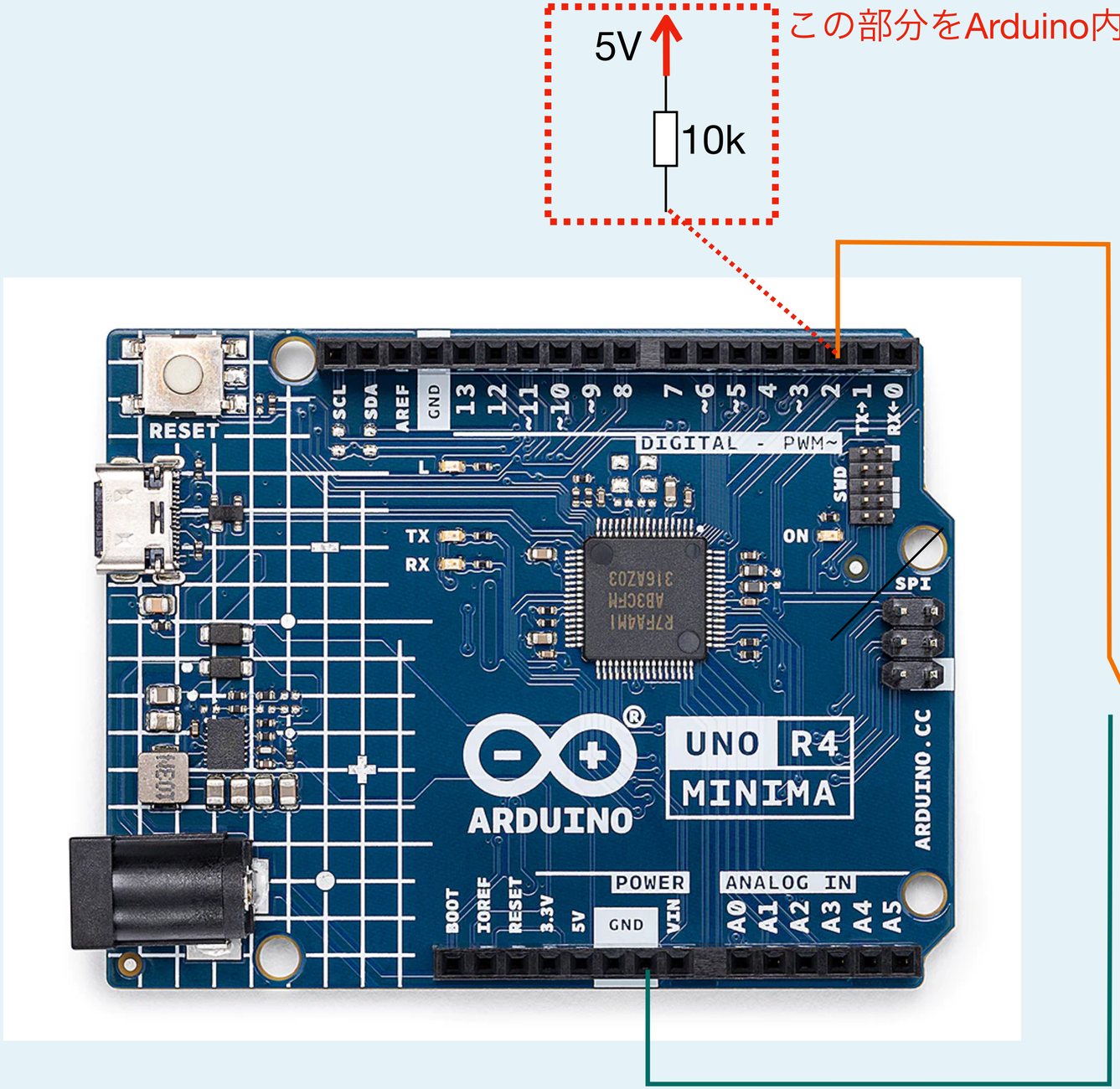

- INPUT_PULLUPを使用しない場合（INPUT）




- INPUT_PULLUPを使用しない場合（INPUT）
- スイッチが触れてない場合は2番ピンは5Vになる

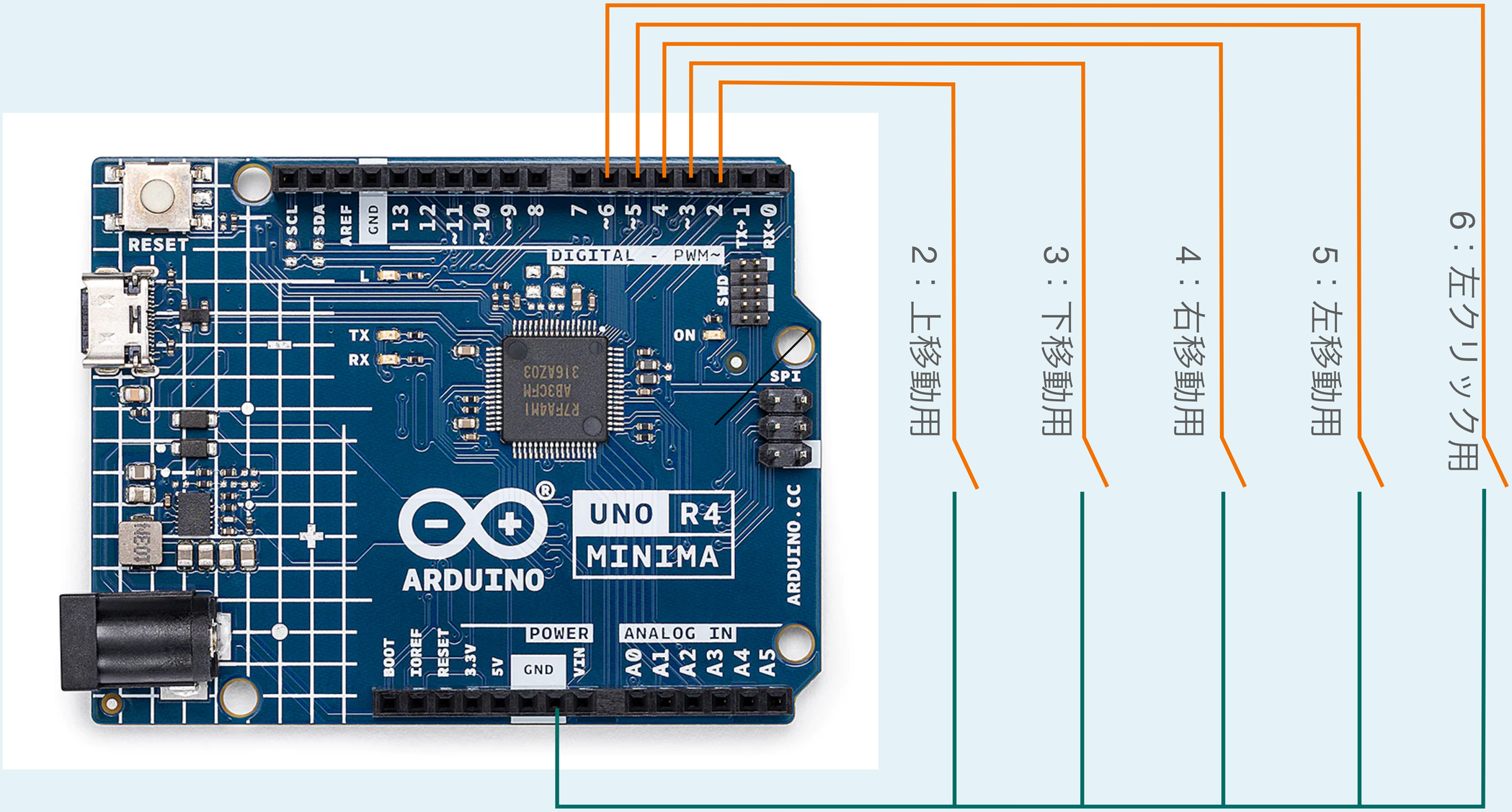



- INPUT_PULLUPを使用しない場合（INPUT）
- スイッチが触れてない場合は2番ピンは5Vになる
- スイッチが触れてる場合は電流は10k抵抗を介してGNDに流れるので0Vになる




- INPUT_PULLUPを使用すると、内部的に抵抗を介して5V電源に接続してくれる
- pinModeがOUTPUTになってる状態でこの接続をすると過電流になるので注意（先にプログラムを書いてから配線しよう）

INPUT_PULLUP
あり



# 余談

- なぜデジタルピンの0,1を使わず2から使用するのか？
- →0,1はシリアル通信にあらかじめ使用されている。Serial.println()した内容がUSB経由でArduino IDEに表示されているが、同じ内容がピンの方でも書き込まれているため、他の用途に使えない

```ino
//マウス操作用ライブラリの読み込み
#include "Mouse.h"
//ピン番号の指定
int up = 2;
int down = 3;
int left = 4;
int right = 5;
int click = 6;

int range = 5;              // 1ループあたりの移動距離


void setup() {
  // 抵抗を使わないためにピンをプルアップにする
  pinMode(up, INPUT_PULLUP);
  pinMode(down, INPUT_PULLUP);
  pinMode(left, INPUT_PULLUP);
  pinMode(right, INPUT_PULLUP);
  pinMode(click, INPUT_PULLUP);
  // initialize mouse control:
  Mouse.begin();
}
```

```ino
void loop() {
  // プルアップなので、スイッチが離れてる時は1、スイッチがGNDとショートされたときに0になる。わかりやすいように反転しておく。
  int upState = 1 - digitalRead(up);
  int downState = 1 - digitalRead(down);
  int rightState = 1 - digitalRead(right);
  int leftState = 1 - digitalRead(left);
  int clickState = 1 - digitalRead(click);

  // 同時押しなども考慮してマウスの移動位置を計算
  int x_distance = (rightState - leftState) * range;
  int y_distance = (downState - upState) * range;

  // 移動距離がゼロの時は何もしない
  if ((x_distance != 0) || (y_distance != 0)) {
    Mouse.move(x_distance, y_distance, 0);
  }
  // クリック用の処理
  if (clickState == 1) {
    // 既にクリックされてたら何もしない
    if (!Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.press(MOUSE_LEFT);
    }
  }
  else {
    // 前ループでクリックしてて今回離してたら解除してあげる
    if (Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.release(MOUSE_LEFT);
    }
  }
  delay(20);
}
```

Mouse_Basic.ino

```ino
//マウス操作用ライブラリの読み込み
#include "Mouse.h"
//ピン番号の指定
int up = 2;
int down = 3;
int left = 4;
int right = 5;
int click = 6;

int range = 5;             // 1ループあたりの移動距離


void setup() {
  // 抵抗を使わないためにピンをプルアップにする
  pinMode(up, INPUT_PULLUP);
  pinMode(down, INPUT_PULLUP);
  pinMode(left, INPUT_PULLUP);
  pinMode(right, INPUT_PULLUP);
  pinMode(click, INPUT_PULLUP);
  // initialize mouse control:
  Mouse.begin();
}

void moveMouse(int x_distance, int y_distance){
  // 移動距離がゼロの時は何もしない
  if ((x_distance != 0) || (y_distance != 0)) {
    //移動距離に乱数を足す
    int x = random(-10,10) + x_distance;
    int y = random(-10,10) + y_distance;
    Mouse.move(x, y, 0);
  }
}
```

```ino
void loop() {
  // プルアップなので、スイッチが離れてる時は1、スイッチがGNDとショートされたときに0になる。
  // わかりやすいように反転しておく。
  int upState = 1 - digitalRead(up);
  int downState = 1 - digitalRead(down);
  int rightState = 1 - digitalRead(right);
  int leftState = 1 - digitalRead(left);
  int clickState = 1 - digitalRead(click);


  // 同時押しなども考慮してマウスの移動位置を計算
  int x_distance = (rightState - leftState) * range;
  int y_distance = (downState - upState) * range;

  moveMouse(x_distance,y_distance);//自作関数の呼び出し
  // クリック用の処理
  if (clickState == 1) {
    // 既にクリックされてたら何もしない
    if (!Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.press(MOUSE_LEFT);
    }
  }
  else {
    // 前ループでクリックしてて今回離してたら解除してあげる
    if (Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.release(MOUSE_LEFT);
    }
  }
  delay(20);
}
```

Mouse_Drunk.ino

```ino
//マウス操作用ライブラリの読み込み
#include "Mouse.h"
//ピン番号の指定
int up = 2;
int down = 3;
int left = 4;
int right = 5;
int click = 6;

int range = 5;              // 1ループあたりの移動距離


void setup() {
  // 抵抗を使わないためにピンをプルアップにする
  pinMode(up, INPUT_PULLUP);
  pinMode(down, INPUT_PULLUP);
  pinMode(left, INPUT_PULLUP);
  pinMode(right, INPUT_PULLUP);
  pinMode(click, INPUT_PULLUP);
  // initialize mouse control:
  Mouse.begin();
}

void moveMouse(int x_distance, int y_distance){
  // 何もしなくても動き続ける
    int x = random(-10,10) + x_distance;
    int y = random(-10,10) + y_distance;
    Mouse.move(x, y, 0);

}
```

```ino
void loop() {
  // プルアップなので、スイッチが離れてる時は1、スイッチがGNDとショートされたときに0になる。
  // わかりやすいように反転しておく。
  int upState = 1 - digitalRead(up);
  int downState = 1 - digitalRead(down);
  int rightState = 1 - digitalRead(right);
  int leftState = 1 - digitalRead(left);
  int clickState = 1 - digitalRead(click);


  // 同時押しなども考慮してマウスの移動位置を計算
  int x_distance = (rightState - leftState) * range;
  int y_distance = (downState - upState) * range;

  moveMouse(x_distance,y_distance);//自作関数の呼び出し
  // クリック用の処理
  if (clickState == 1) {
    // 既にクリックされてたら何もしない
    if (!Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.press(MOUSE_LEFT);
    }
  }
  else {
    // 前ループでクリックしてて今回離してたら解除してあげる
    if (Mouse.isPressed(MOUSE_LEFT)) {
      Mouse.release(MOUSE_LEFT);
    }
  }
  delay(20);
}
```

Mouse_Drunk.ino

```
//マウス操作用ライブラリの読み込み
#include "Mouse.h"

//ピン番号の指定
int up = 2;
int down = 3;
int left = 4;
int right = 5;
int lclick = 6;
int rclick = 7;

int range = 5;            // 1ループあたりの移動距離
void moveMouse(int x_distance, int y_distance){
  // 移動距離がゼロの時は何もしない
  if ((x_distance != 0) || (y_distance != 0)) {
    Mouse.move(x_distance, y_distance, 0);
  }
}
//クリック処理を関数として切り出す
void clickButton(int click_state, uint8_t button_type){
  if (click_state == 1) {
    // 既にクリックされてたら何もしない
    if (!Mouse.isPressed(button_type)) {
      Mouse.press(button_type);
    }
  }
  else {
    // 前ループでクリックしてて今回離してたら解除してあげる
    if (Mouse.isPressed(button_type)) {
      Mouse.release(button_type);
    }
  }
}
```

```
void setup() {
  // 抵抗を使わないためにピンをプルアップにする
  pinMode(up, INPUT_PULLUP);
  pinMode(down, INPUT_PULLUP);
  pinMode(left, INPUT_PULLUP);
  pinMode(right, INPUT_PULLUP);
  pinMode(lclick, INPUT_PULLUP);
  pinMode(rclick, INPUT_PULLUP);
  // initialize mouse control:
  Mouse.begin();
}

void loop() {
  // プルアップなので、スイッチが離れてる時は1、スイッチがGNDとショートされたときに0になる。
  // わかりやすいように反転しておく。
  int upState = 1 - digitalRead(up);
  int downState = 1 - digitalRead(down);
  int rightState = 1 - digitalRead(right);
  int leftState = 1 - digitalRead(left);
  int lclickState = 1 - digitalRead(lclick);
  int rclickState = 1 - digitalRead(rclick);

  // 同時押しなども考慮してマウスの移動位置を計算
  int x_distance = (rightState - leftState) * range;
  int y_distance = (downState - upState) * range;

  moveMouse(x_distance,y_distance);//自作関数の呼び出し
  // クリック用の処理
  clickButton(lclickState,MOUSE_LEFT);
  clickButton(rclickState,MOUSE_RIGHT);
  delay(20);
}
```

Mouse_RClick.ino

# 注意

- マウスが勝手に動き続けるようなプログラムを書いた場合、新しいプログラムを書き込むためにArduinoを接続したらマウスが勝手に動いて書き込めない…といった状況が起こりうる
- その場合、ArduinoのRESETボタンを2回素早くクリックすると、書き込み用のモードに変化するのでそれで対処
- また、SerialとUSB HID機能は同時に使うと挙動が怪しい（別の機器に繋いだときに動かなくなったりする）ので、デバッグが終わったら消すのが無難

# やってみよう

- 自作マウスでネットサーフィン/お絵描き/ゲーム
- 長押ししたら加速するように改造
- 前回使った距離センサなど、アナログピンを使うとどうなるか？

```
#include "Keyboard.h"


void setup() {
  // make pin 2 an input and turn on the pull-up resistor so it goes high
unless
  // connected to ground:
  pinMode(2, INPUT_PULLUP);
  Keyboard.begin();
}

void loop() {
  while (digitalRead(2) == HIGH) {
    // do nothing until pin 2 goes low
    delay(500);
  }
  delay(1000);

  Keyboard.press(KEY_LEFT_GUI);
  // Shift-Q logs out:
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.press('Q');
  delay(100);
  Keyboard.releaseAll();
  // enter:
  Keyboard.write(KEY_RETURN);


  // do nothing:
  while (true)
    ;
}
```

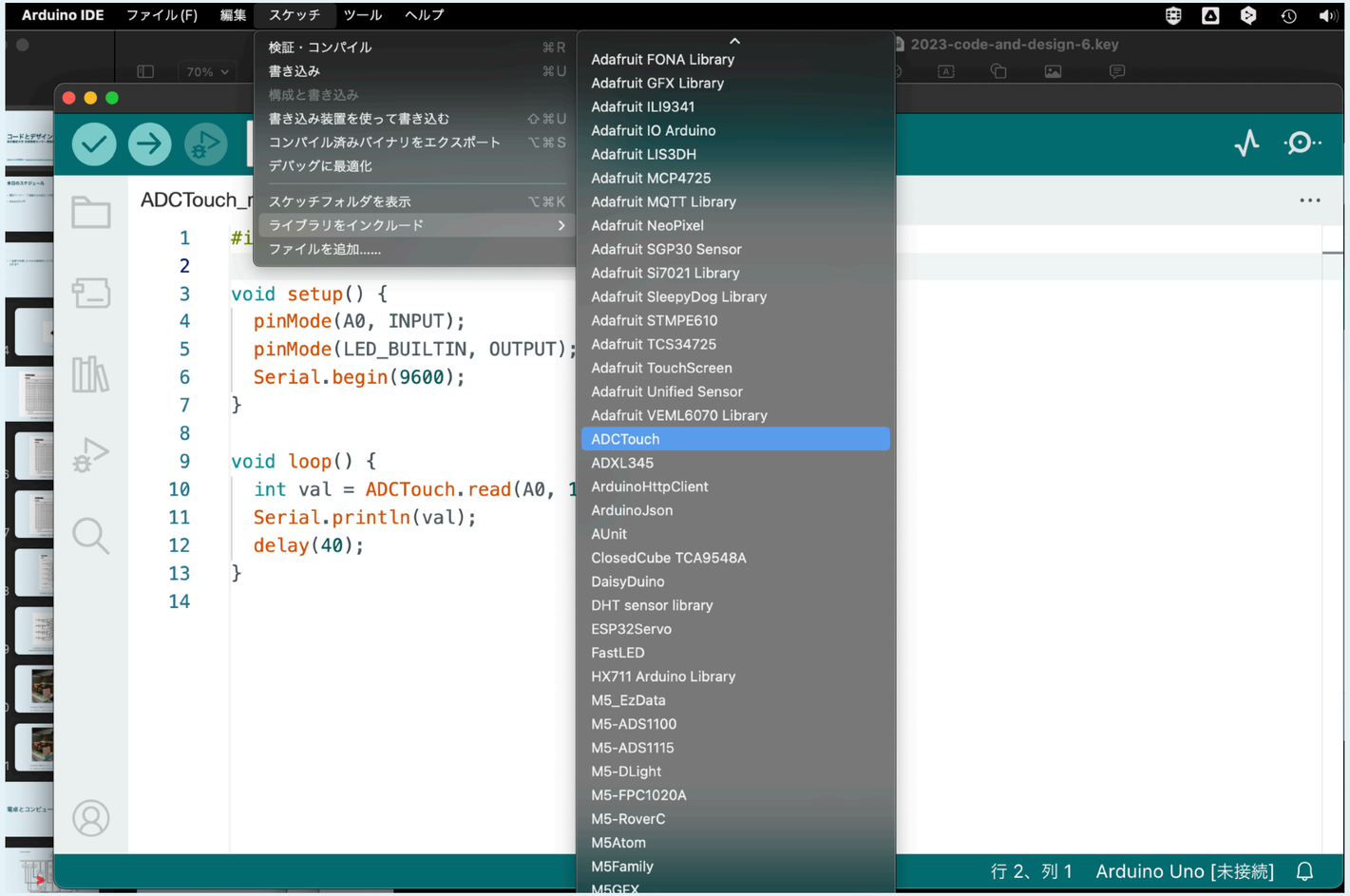
マウスと同様にキーボード操作の自動化もできる（2番ピンのスイッチを押すと強制ログアウトするキーボード）

Examples - 0.9USB - Keyboard - KeyBoardLogOut をmacOS用のみに簡略化

# おまけ：ワイヤー本でタッチセンサを作る（Arduino R3以前限定）

# 検索


※AMCの端末はシャットダウン時にインストールしたライブラリが消えます




スケッチ>ライブラリをインクルード で#includeするべきファイル名を自動で入れてくれる

```
#include <ADCTouch.h>

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

void loop() {
  int val = ADCTouch.read(A0, 100); //(使用するピン番号,平均を取るサンプル数)
  Serial.println(val);
  delay(40);
}
```

A0にワイヤを刺して、触ってみよう

シリアルプロッタで値の範囲を確認してみよう

ADCTouch_monitor.ino

スケッチのアップロード時や、
書き込み後の電源立ち上げ時（setup関数が走るとき）に、
ワイヤを触ってると範囲が狂うことがあるので注意

```
#include <ADCTouch.h>
//setup()とloop()で共通して使う値はグローバル変数(関数の外側)で定義しておく
int ref = 0;

void setup() {
  pinMode(A0, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  //繋げた導体の状態で初期値が変わるので、それを保存しておく必要がある
  ref = ADCTouch.read(A0, 100);
}

void loop() {
  //初期値分を引き算してオフセット
  int val = ADCTouch.read(A0, 100) - ref;
  if (val > 40) {
    digitalWrite(LED_BUILTIN, HIGH);
  } else {
    digitalWrite(LED_BUILTIN, LOW);
  }

  Serial.println(val);

  delay(40);
}
```