

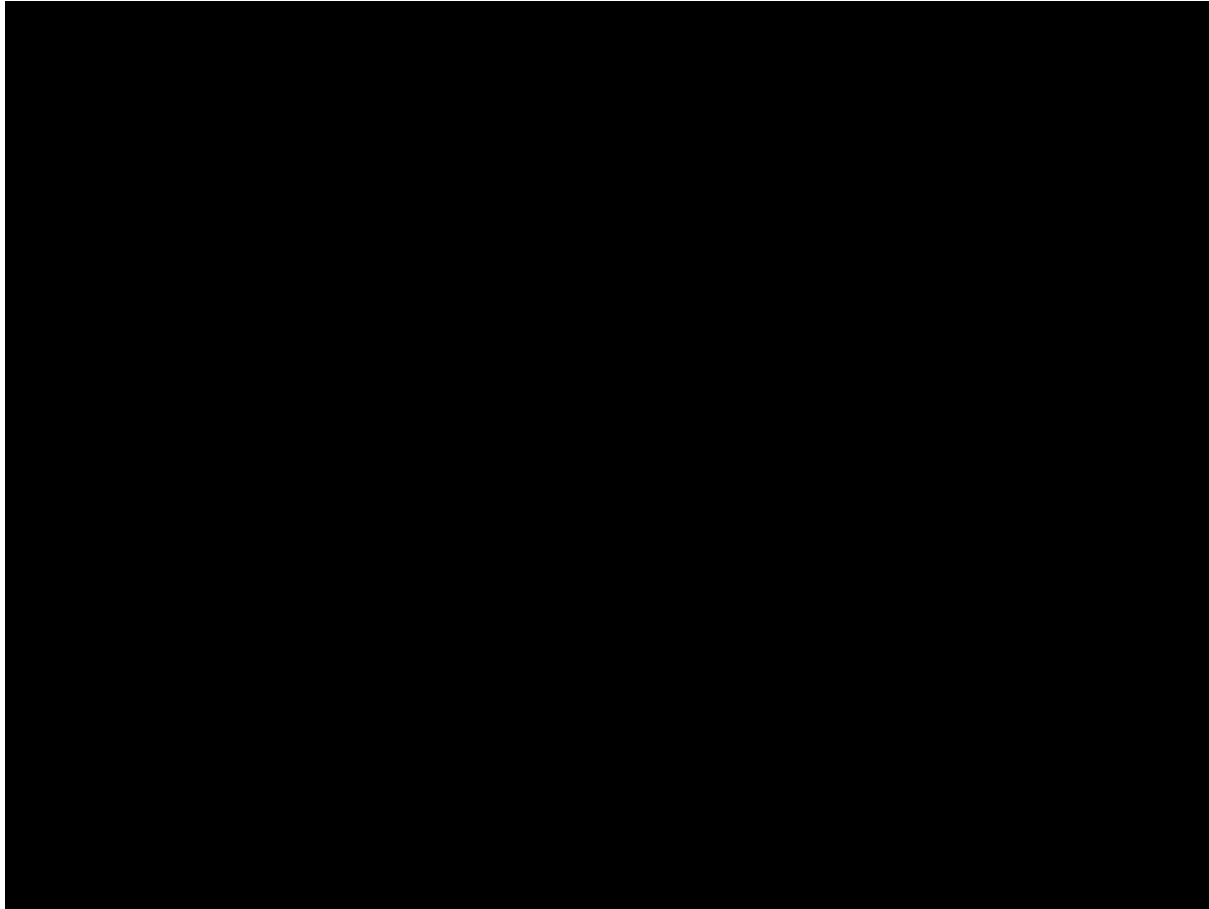
Arduinoではじめる 素晴らしいマイコンの世界



IT Technical Seminar V12
2011.05.20

株式会社オープン・トーン
根本

今日はこんなことをします！



今回の目的

- まずはマイコンを知ってもらおう
 - マイコンってなに？
 - Arduinoって一体・・・？
- マイコンって簡単ってことを知ってもらおう
 - 基礎講座
 - ちょっとだけ応用
- 素晴らしいマイコンの世界に興味を持っていたらだけば

おらっしゃ

あああ！！！！

∩ ∧ ∧

＼(°Д°)

＼c＼

o-、)～

u

さっそくですが・・・

マイコンってなんだろう？



CPUやメモリを1つのLSIチップに集積した回路

どんなことに使われている？

- 現代ではいろいろなものにマイコンが利用されている
- どんなものに利用されているかご存じ？



ところでArduinoって？

- アルドゥイーノと発音
- イタリア生まれ
- 入出力ポートを備えた基板
- C言語風のArduino言語とその統合開発環境から構成されるシステム

とにかく・・・ **開発環境一式が揃っているため、
誰でも簡単にマイコンの開発を
始めることが出来る！**

まずは準備を始めます！

必要なもの

- Arduino UNO
 - 3000円程度
- USBケーブル
 - 580円程度
 - Aオス-Bオス

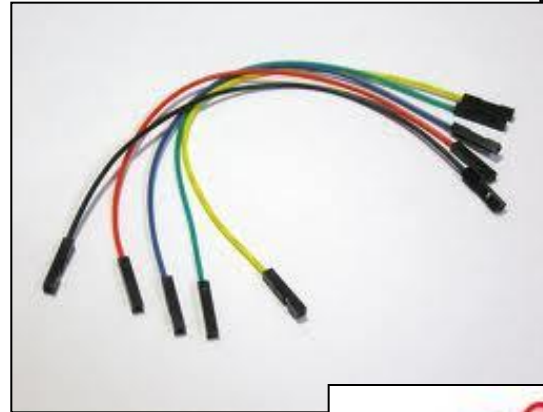


あったらいいもの

- ブレッドボード
 - 半田付けの必要がない
 - 250円程度



- ジャンパワイヤ
 - 柔軟性がある
 - 10本300円程度
 - オス・メスに注意



- 発光ダイオード
 - 10本300円程度



次は開発環境構築です！

開発環境をつくりましょう！

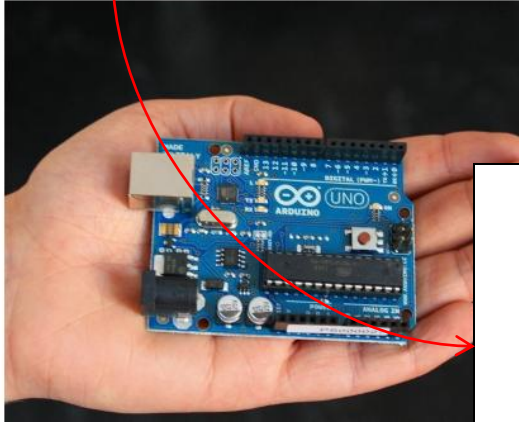
□ <http://arduino.cc/en/>

Main Site Blog Playground Forum Help | Sign in or Register



search

Buy Download Getting Started Learning Reference Hardware FAQ



Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

Download

Arduino 0022 (release notes), hosted by Google Code:

- + Windows
- + Mac OS X
- + Linux: 32 bit, 64 bit
- + source

Next steps

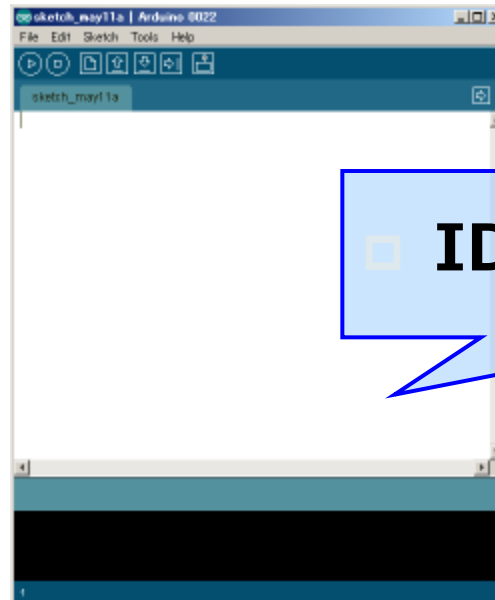
- Getting Started
- Reference
- Environment
- Examples
- Foundations
- FAQ

Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit) (64bit), Source

Downloadページに進み、使用OSにあったダウンロードリンクをクリック
(2011/5/20時点のバージョンは「Arduino 0022」)

インストール

- Arduino IDEのインストールと設定
 - インストールは、先ほどダウンロードした「arduino-0022.zip」を任意の場所に展開するだけ
 - 展開したファイルの中の「arduino.exe」を実行してIDEを起動する



□ **IDEを起動したところ**

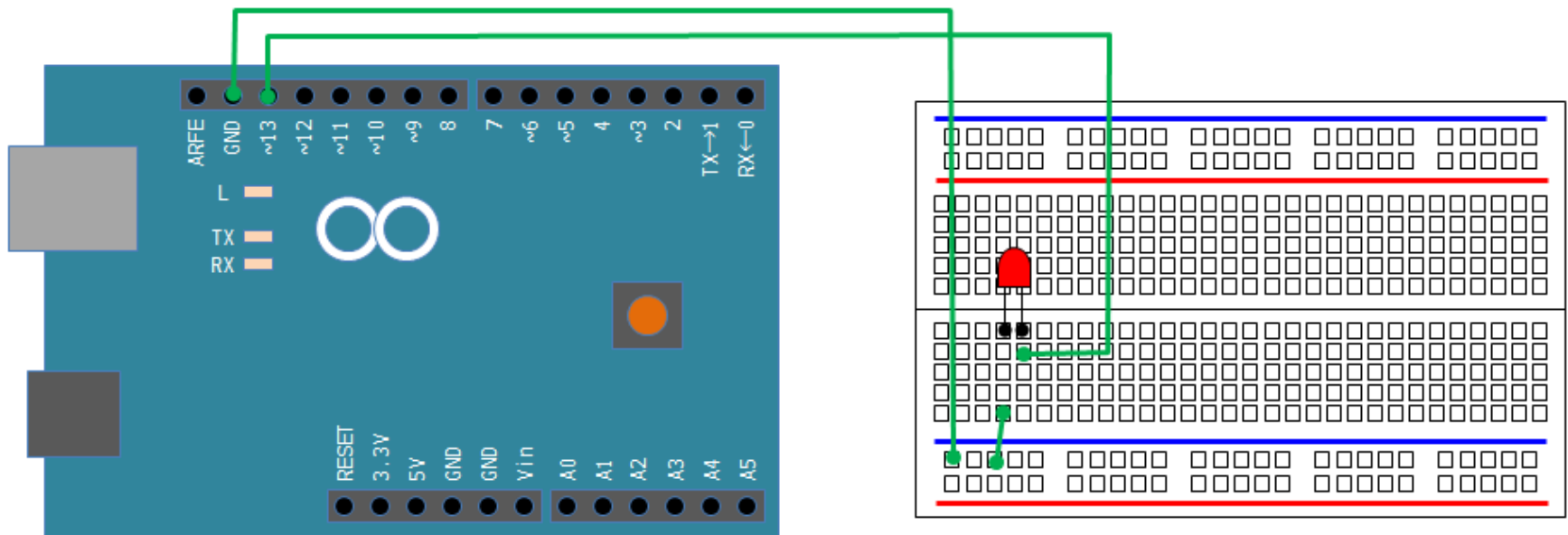
IDEの設定

- Arduino IDEのインストールと設定（続き）
 - Tools > Board > Arduino Unoを選択する
 - Tools > Serial Portで、USBケーブルを接続したシリアルポートを指定する

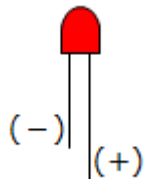
以上で、開発環境構築は完了です。

いよいよ、部品の組み立てとプログラムです！

ArduinoにLEDを配線

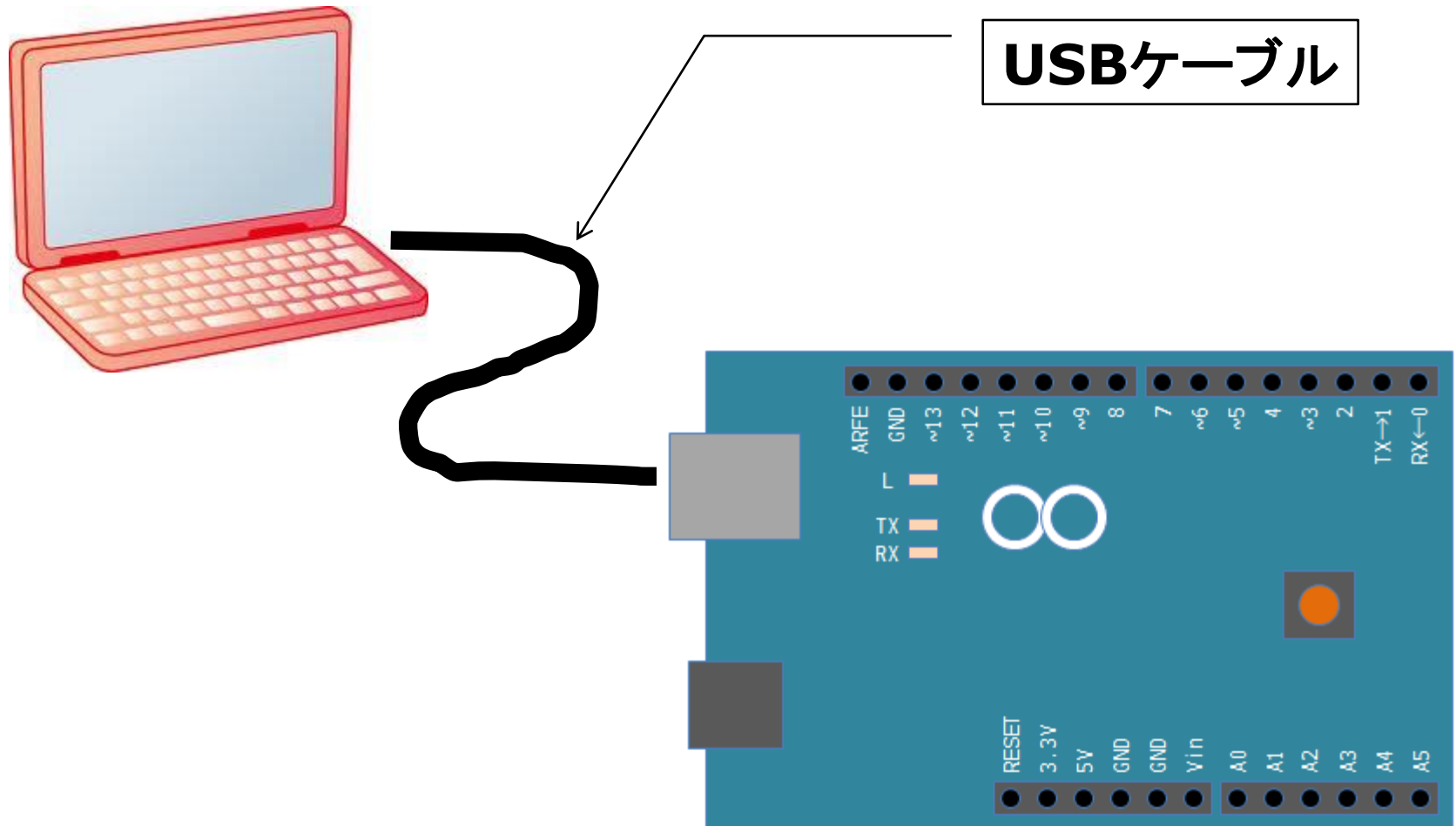


1. LEDのプラスとArduinoのPIN13をジャンプワイヤでつなげる。
2. LEDのマイナスとArduinoのGNDをジャンプワイヤでつなげる。



※LEDは足が長いほうがプラスで、短いほうがマイナスです。

配線したArduinoとPCを接続



スケッチしてみよう！

□ Arduino IDEを起動

□ 次のスケッチを記述してみよう！

```
#define LED_PIN 13
#define INTERVAL 1000

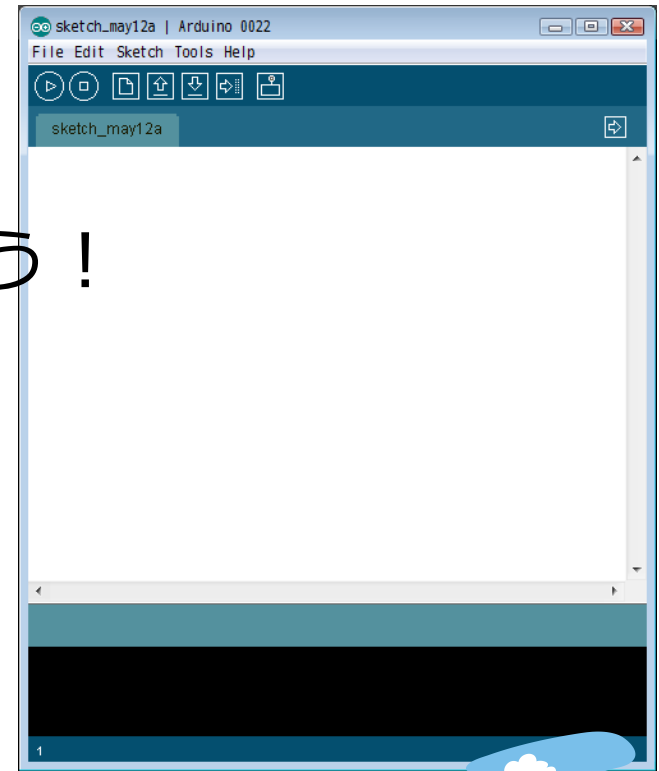
int value = LOW;

void setup() {
  pinMode(LED_PIN, OUTPUT);
}

void loop() {

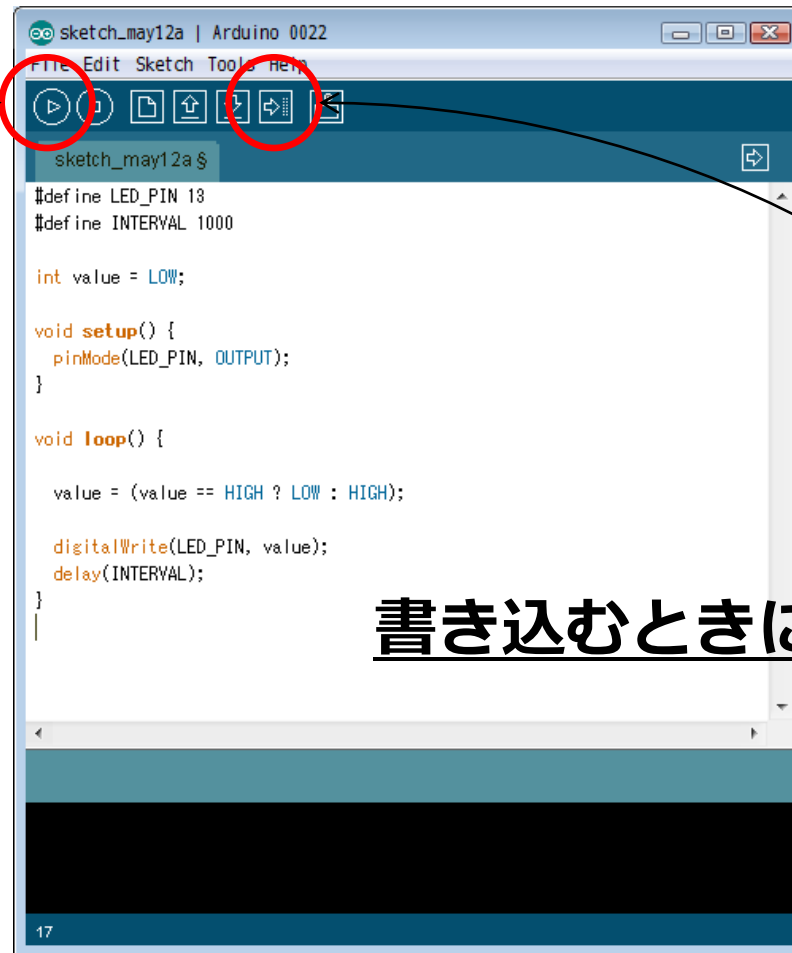
  value = (value == HIGH ? LOW : HIGH);

  digitalWrite(LED_PIN, value);
  delay(INTERVAL);
}
```



VerifyとUpload

データに誤りがなければ、規格に従っているかなどを検査する

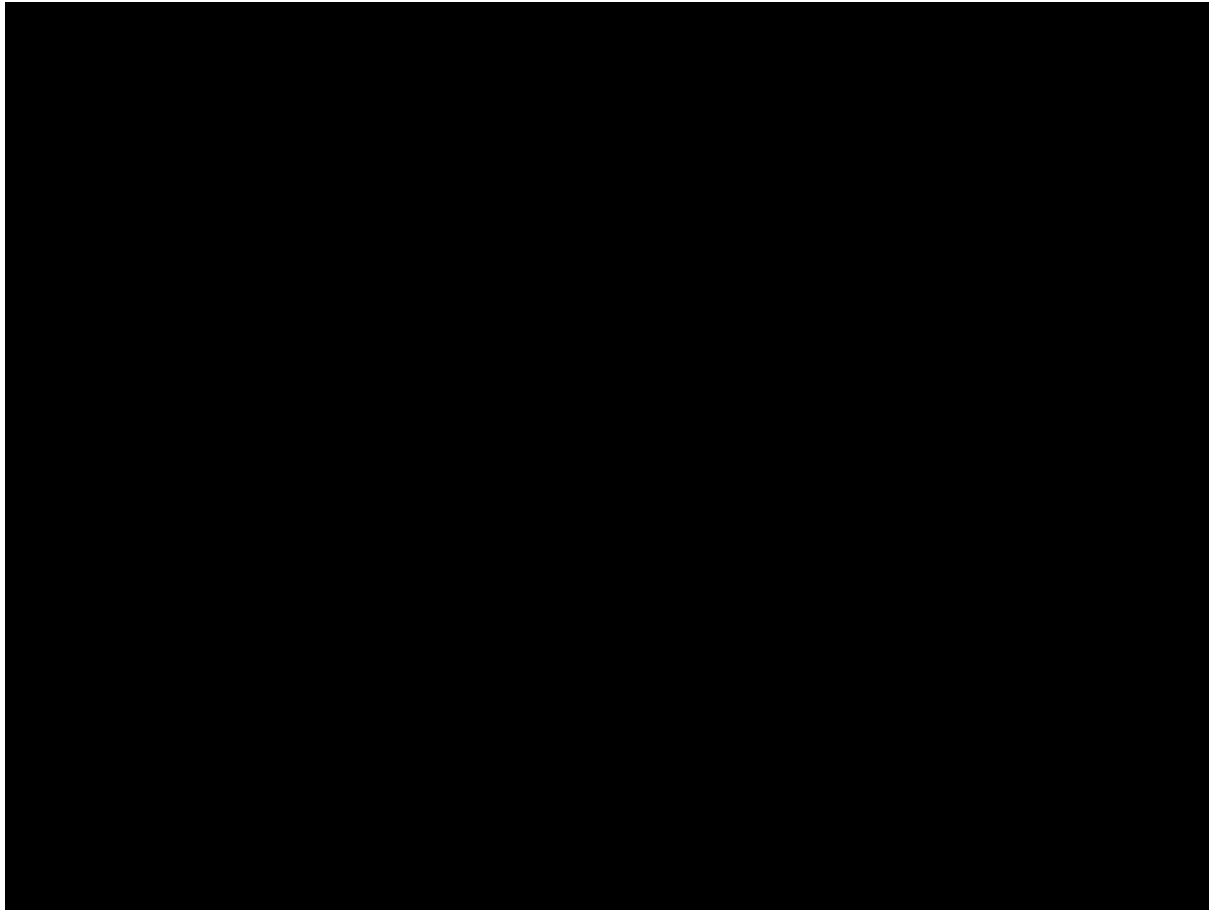


ArduinoにUploadする

書き込むときにはここをClick !



で、出来たのがこんな感じ



じゃあソースコードを見てみましょう！

スケッチの基本とは？

```
#define LED_PIN 13  
#define INTERVAL 1000
```

```
int value = LOW;
```

```
void setup() {  
  pinMode(LED_PIN, OUTPUT);  
}
```

```
void loop() {  
  
  value = (value == HIGH ? LOW : HIGH);  
  
  digitalWrite(LED_PIN, value);  
  delay(INTERVAL);  
}
```

setup()
開始時に一度だけ呼ばれる

loop()
この処理が呼ばれ続ける



チカチカ光らせるスケッチ

```
#define LED_PIN 13  
#define INTERVAL 1000
```

```
int value = LOW;
```

```
void setup() {  
  pinMode(LED_PIN, OUTPUT);  
}
```

```
void loop() {
```

```
  value = (value == HIGH ? LOW : HIGH);
```

```
  digitalWrite(LED_PIN, value);  
  delay(INTERVAL);  
}
```

PIN13をOUTPUTとして指定

デジタル出力

指定したミリ秒待機します。

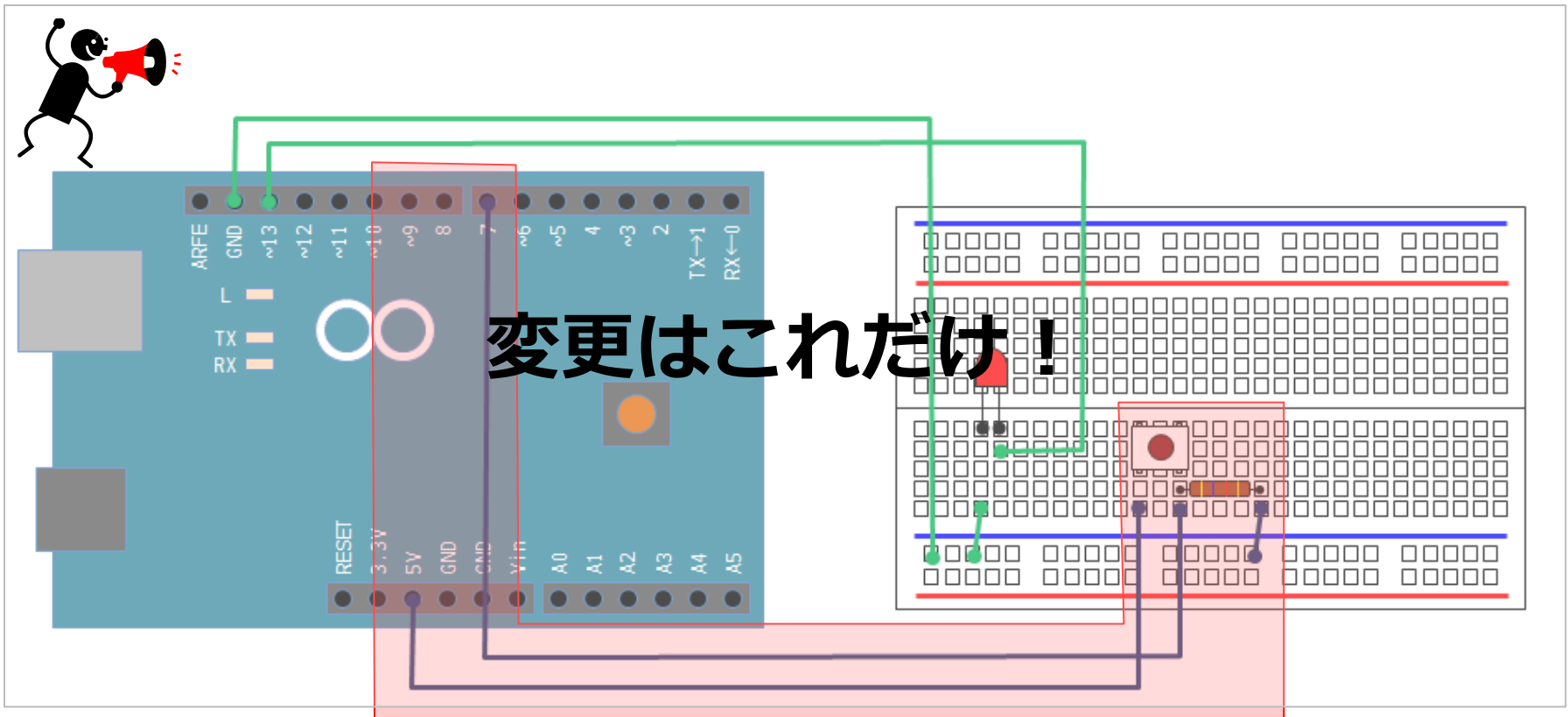


簡単でしょ？



**これがArduino版
Hello Worldです！**

では入力も加えましょう！



1. スイッチの片足側を、電源（5V）へつなげる。
2. スイッチのもう片足側は抵抗を挟み、入力するPINとなるPIN7へつなげる。
3. 抵抗を通した先は、GNDへつなげる。

スケッチも少しだけ修正

```
#define LED_PIN 13  
#define BTN_PIN 7  
#define INTERVAL 1000
```

```
int value = LOW;
```

```
void setup() {  
  pinMode(LED_PIN, OUTPUT);  
  pinMode(BTN_PIN, INPUT);  
}
```

```
void loop() {
```

```
  if (digitalRead(BTN_PIN) == HIGH || value == HIGH) {  
    value = (value == HIGH ? LOW : HIGH);
```

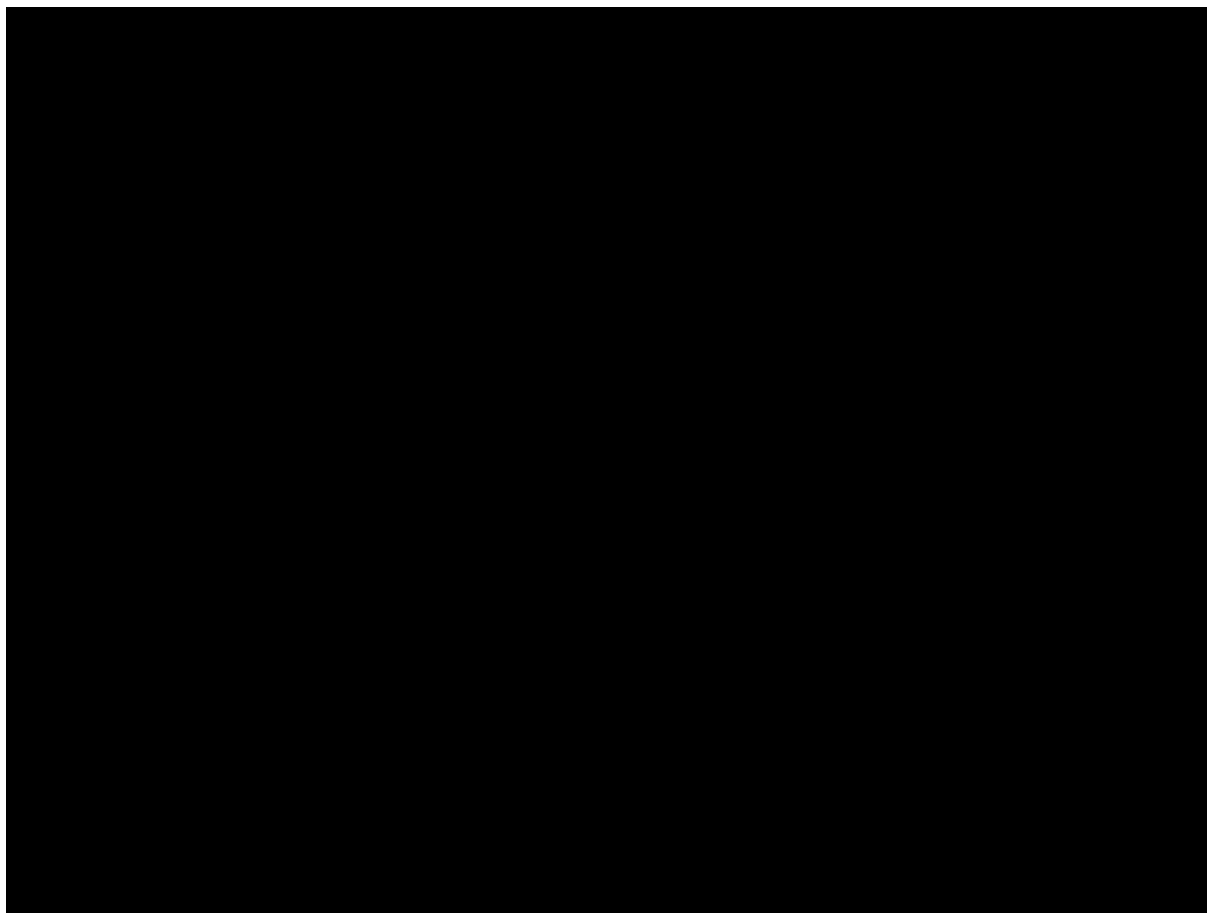
```
    digitalWrite(LED_PIN, value);  
    delay(INTERVAL);
```

```
  }  
}
```

赤字が追加した箇所です。



で、出来たのがこんな感じ



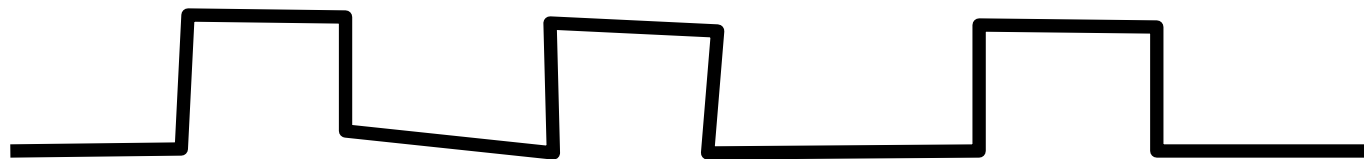
簡単でしょ？

とりあえず入出力完了。
興味を持ってもらえたか？

もう少しだけ基礎にお付き合いください

今までの件は . . .

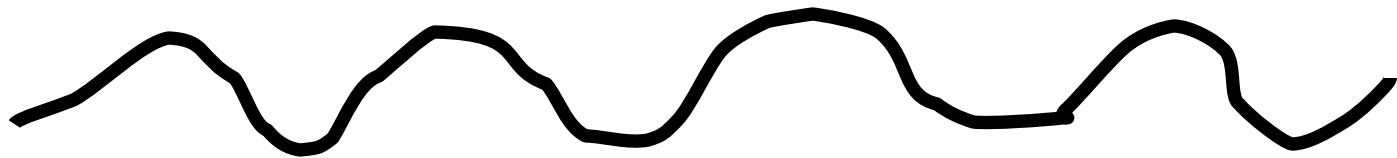
- 俗に言う、デジタル信号の入出力です。



- こんなイメージです。
- なので、LEDも付く・付かないのような制御。

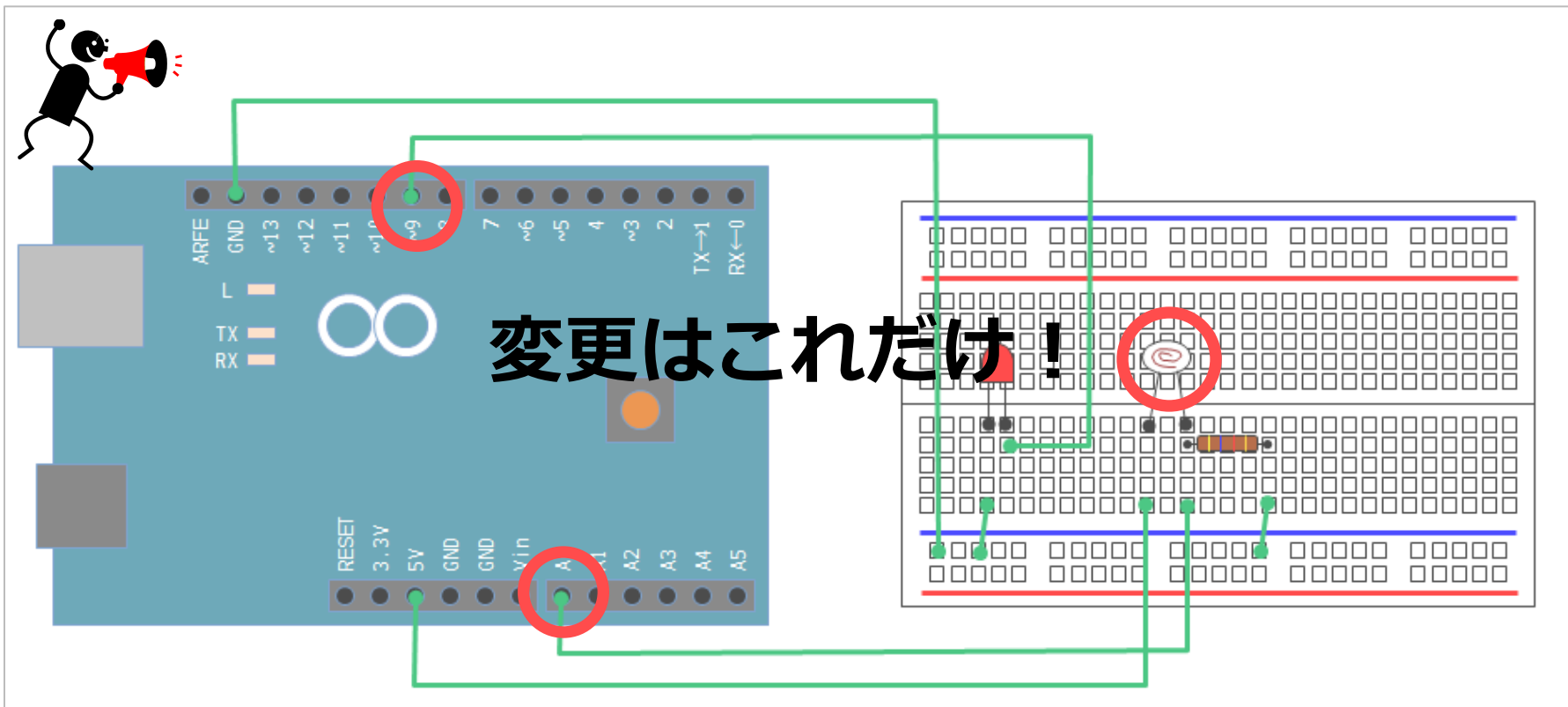
じゃあアナログ信号って？

□ アナログ入出力のご紹介



- こんなイメージです。
- なので、LEDも強弱をつけて制御できる。

スイッチからセンサーへ！



1. LEDの出力先のPINは、PIN13からPIN9へ変更
2. スイッチをセンサーへ変更
3. センサーの入力PINを、PIN7からPINA0へ変更

ついでに解説

```
#define LED_PIN 9  
#define SENSOR_PIN 0  
#define INTERVAL 10
```

```
int value = 0;
```

```
void setup() {  
  pinMode(LED_PIN, OUTPUT);  
}
```

```
void loop() {  
  value = analogRead(SENSOR_PIN);  
  analogWrite(LED_PIN, value / 4);  
  delay(INTERVAL);  
}
```

PIN9をOUTPUTとして指定

センサーの値を取得
(アナログ入力)

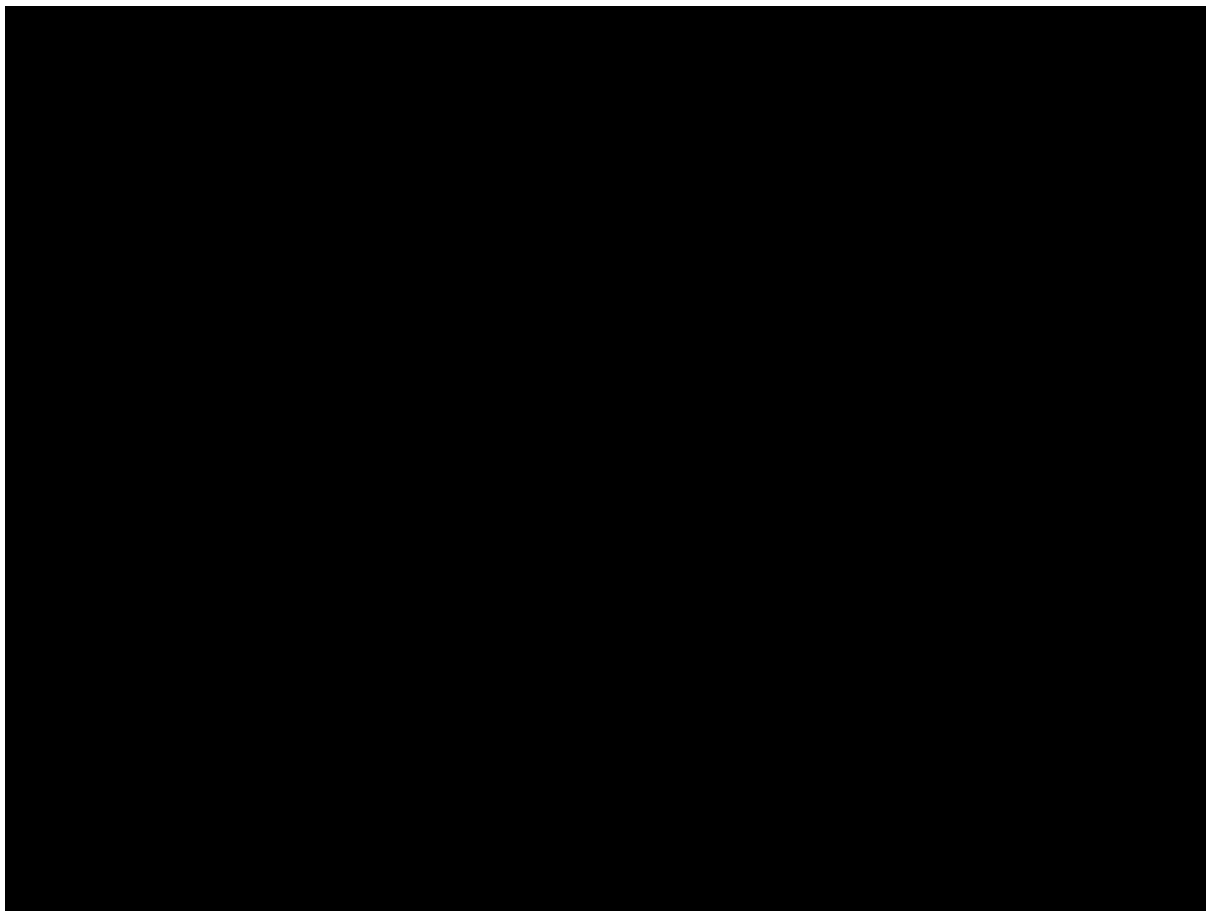
アナログ出力

※補足

analogWrite()でvalueを4で割る理由は、
入力は0～1024の値を取得するが、
出力は0～255の値までの為。



で、出来たのがこんな感じ



簡単でしょ？



この程度で基礎が終わり？

この程度です！

- Arduinoが出来ること
 - デジタル入力
 - デジタル出力
 - アナログ入力
 - アナログ出力
 - プログラム実行

これだけ・・・



でもこれだけ出来ると・・・

こんなことも可能です。



発想次第で夢が広がる（かな）

そんなArduinoで遊ぶ為に・・・

次からは
少～しだけスキルアップ！？



こんなの使ってみます。

何ができるかな？？？

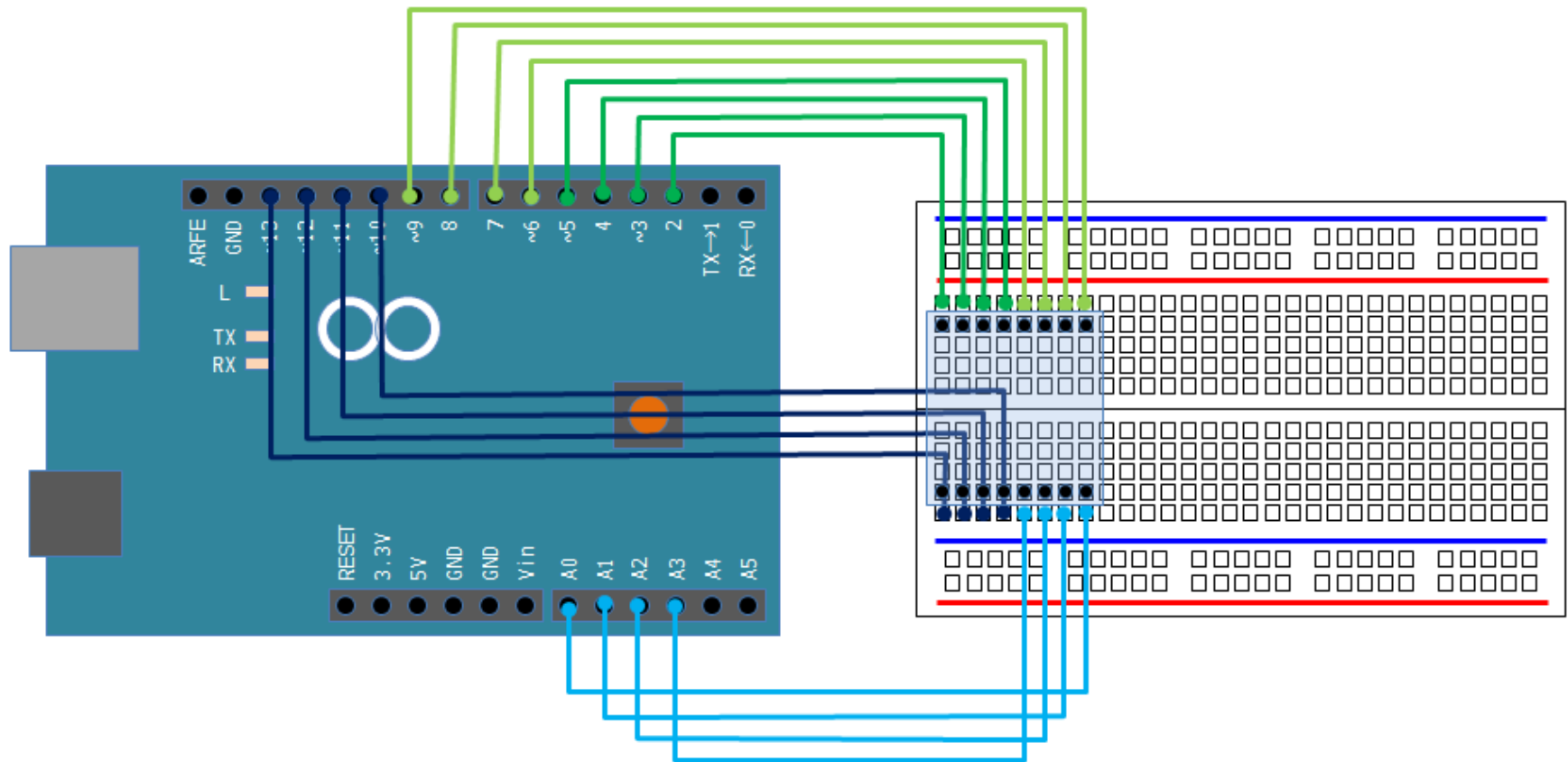


電光掲示板！



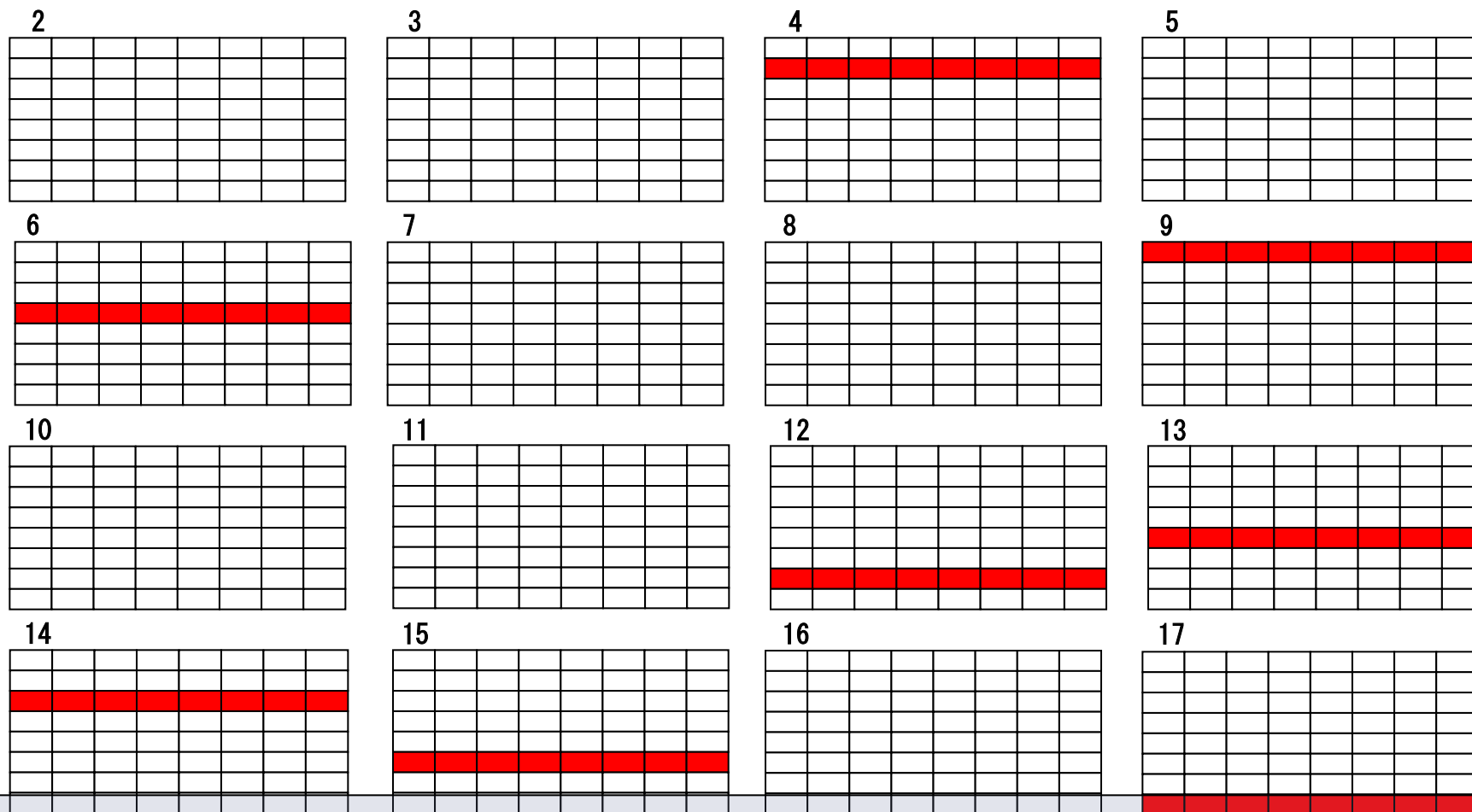
の子供見たいなもの
じゃあ早速つかってみましょう

わからないので . . .



PINとマトリクスのPINをつないでみましょう（汗）

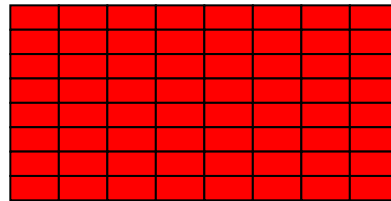
どうなるか確かめる



このPINに信号を送ると、この列がつくんだ！

ってことは！？

4,6,9,12,13,14,15,17のPINに信号を送ると、

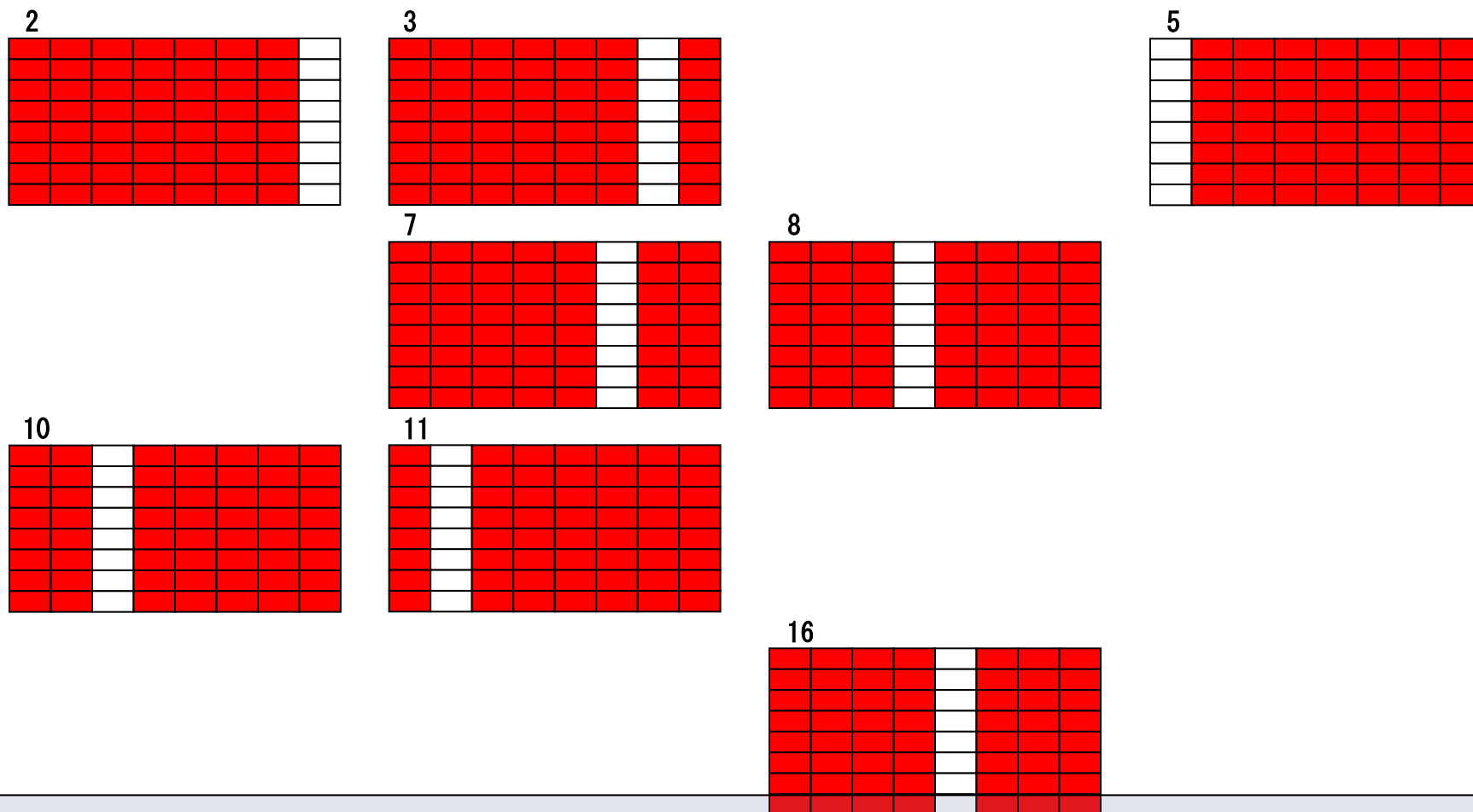


こうなるんだね！

では、

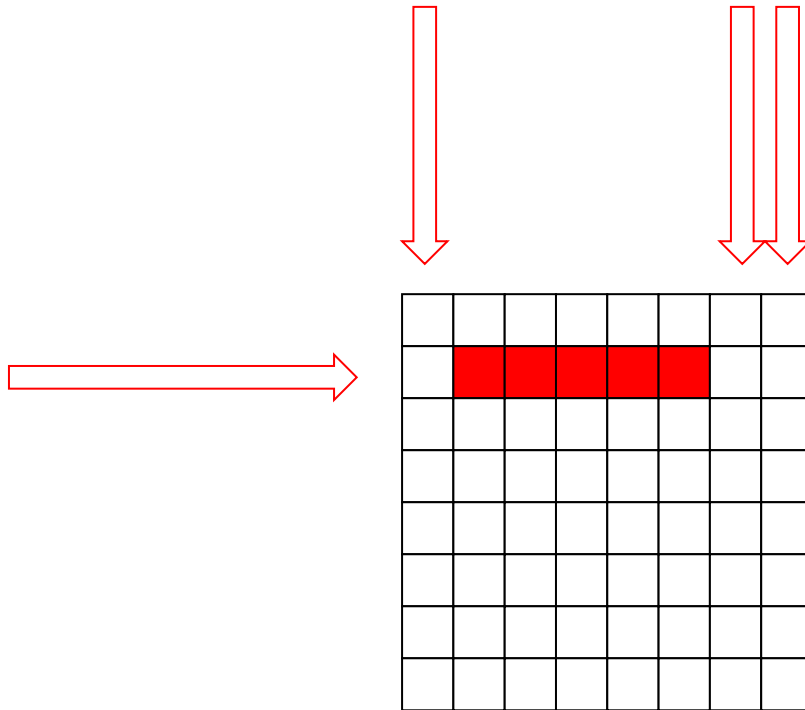
この状態で変化のなかったPINをもう一度試すと・・・

結果はこうなりました。

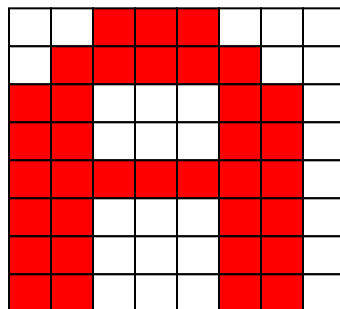


これでPINの役割がわかりました！

例えば . . .



では「A」を表示させたい場合

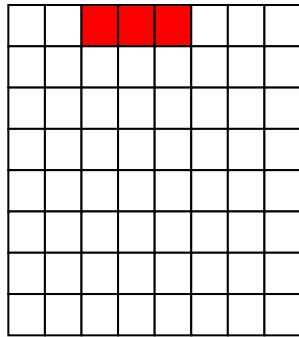


でも . . .

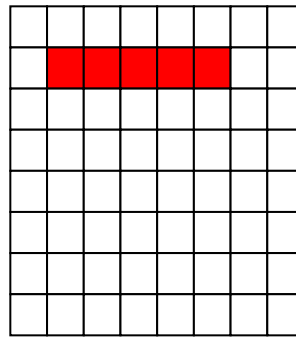
全部で16本のPINで、どう表現しているんだ???

実は「A」はこの集合体

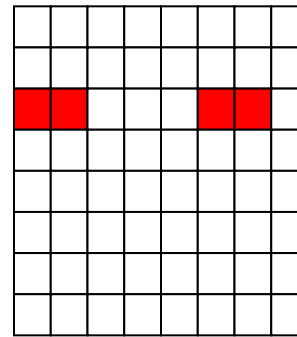
1列目



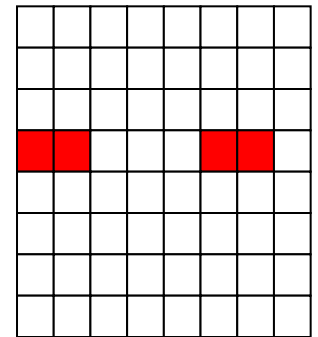
2列目



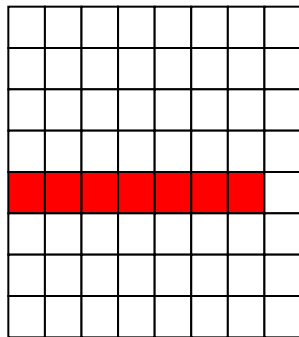
3列目



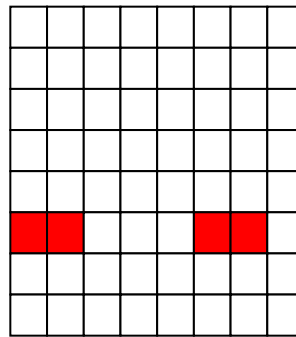
4列目



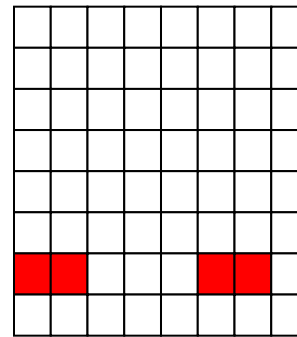
5列目



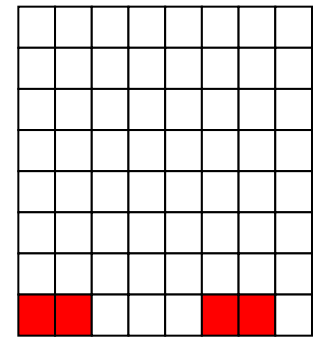
6列目



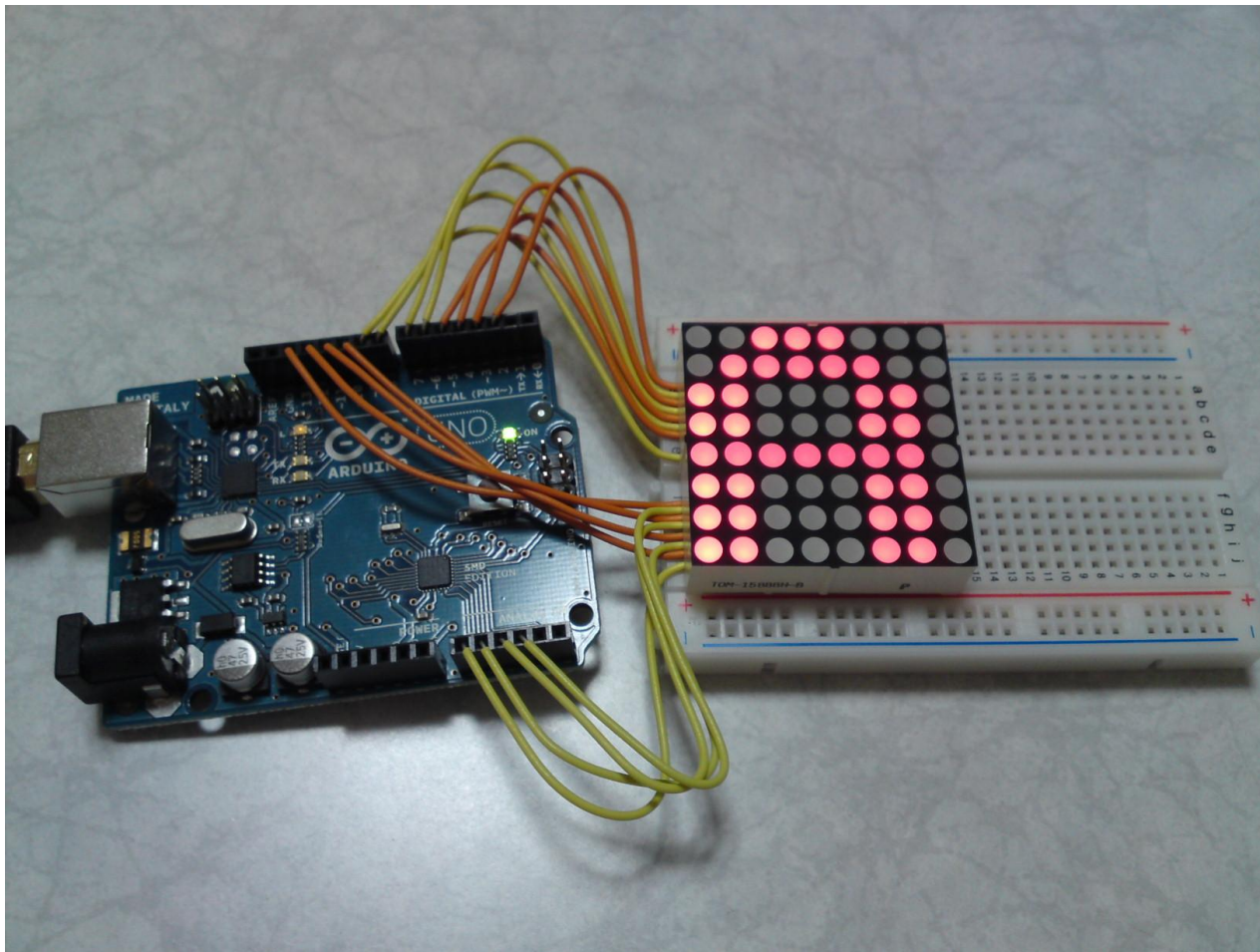
7列目



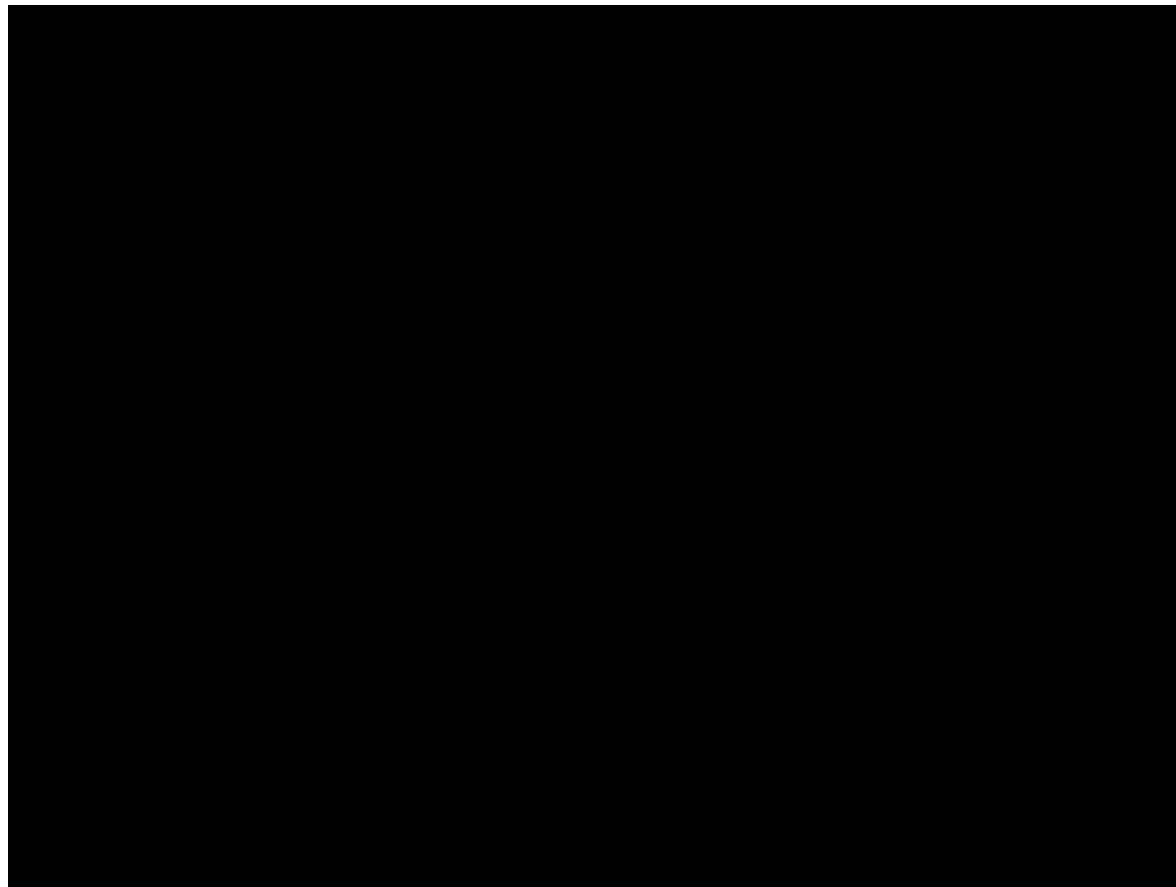
8列目



これを高速に連続させると



さらにこれを応用すると



じゃあ最後はOpentoneってことで



超お手軽に

Felicaを読んじゃう方法です！

必要なものは

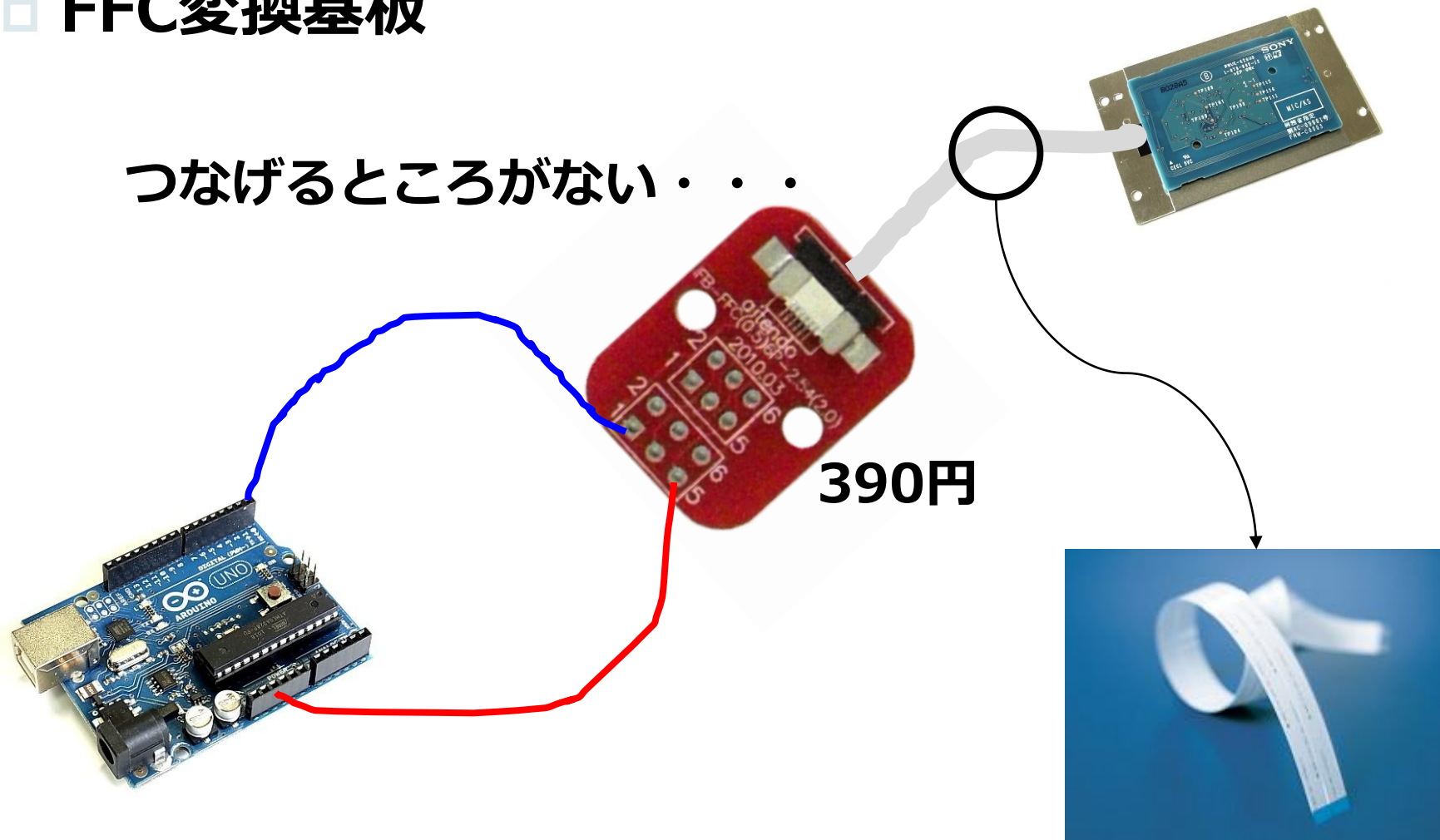
RC-S620S

- これ1つでFeliCaのリーダー
- SONYさんからライブラリが提供されている
 - ▣ これが超お手軽な最大の理由！

実は必要なものがもう一つ

□ FFC変換基板

つなげるところがない . . .



ライブラリをダウンロード

■ Arduino向けRC-S620/S制御ライブラリ

- http://blog.felicalauncher.com/sdk_for_air/?page_id=2699

SONY
make.believe

Developers' Blog 検索

FeliCa ホーム | Developers' Blog トップ

FeliCa 非接触ICカード技術 "FeliCa"

FeliCa Developers' Blog

FeliCaやNFCの技術情報、開発環境、イベント情報などについてお伝えします

「Arduino向けRC-S620/S制御ライブラリ」

ダウンロードご希望の方は、以下の「RC-S620/S制御ライブラリ」使用してください。ボタンをクリックするとダウンロードを開始します。

== Version 1.0.0.1 2010.11.26リリース版 ==

RC-S620/S制御ライブラリ

本規約は、ソニー株式会社(以下「ソニー」とします)が作成「RC-S620/S」制御ライブラリソフトウェアプログラム(以下「本関連資料」(以下「関連資料」とします)につき、ソニーがお客様に提供いたします。

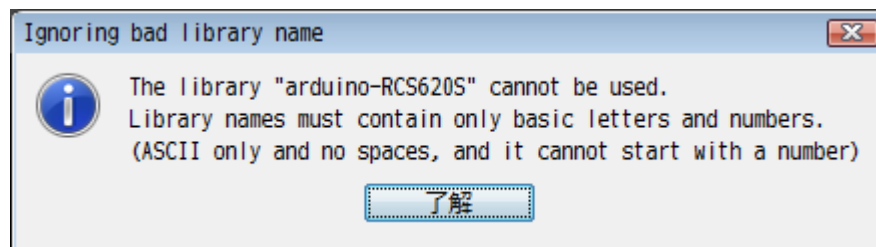
第6条(本ソフトウェア等の廃棄)
第4条の規定により使用権の許諾が解約された場合、お客様は当該使用権の許諾期間の終了した日から2週間以内に本ソフトウェア、関連資料およびそれらの複製物を廃棄するものとします。但し、第1条に基づき、第三者に頒布されたものはこの限りではありません。

第7条(その他)
1.お客様は、ソニーの事前の文書による承諾なくして、本規約上の地位ならびに本規約に基づく権利・義務の全部または一部を、第三者に譲渡し、承継させ、もしくは、担保の用に供してはならないものとします。
2.本規約は、日本国法に基づき解釈、適用されるものとします。
3.お客様は、本ソフトウェアおよび関連資料を日本国外に持ち出して使用する場合、適用ある輸出管理規制、法律または命令に従うものとします。
4.本規約に定めなき事項もしくは本規約の解釈に疑義を生じた場合、お客様はソニーにお問い合わせのうえソニーと誠意をもって協議し、解決するものとします。

規約に同意して、ダウンロードを開始します。

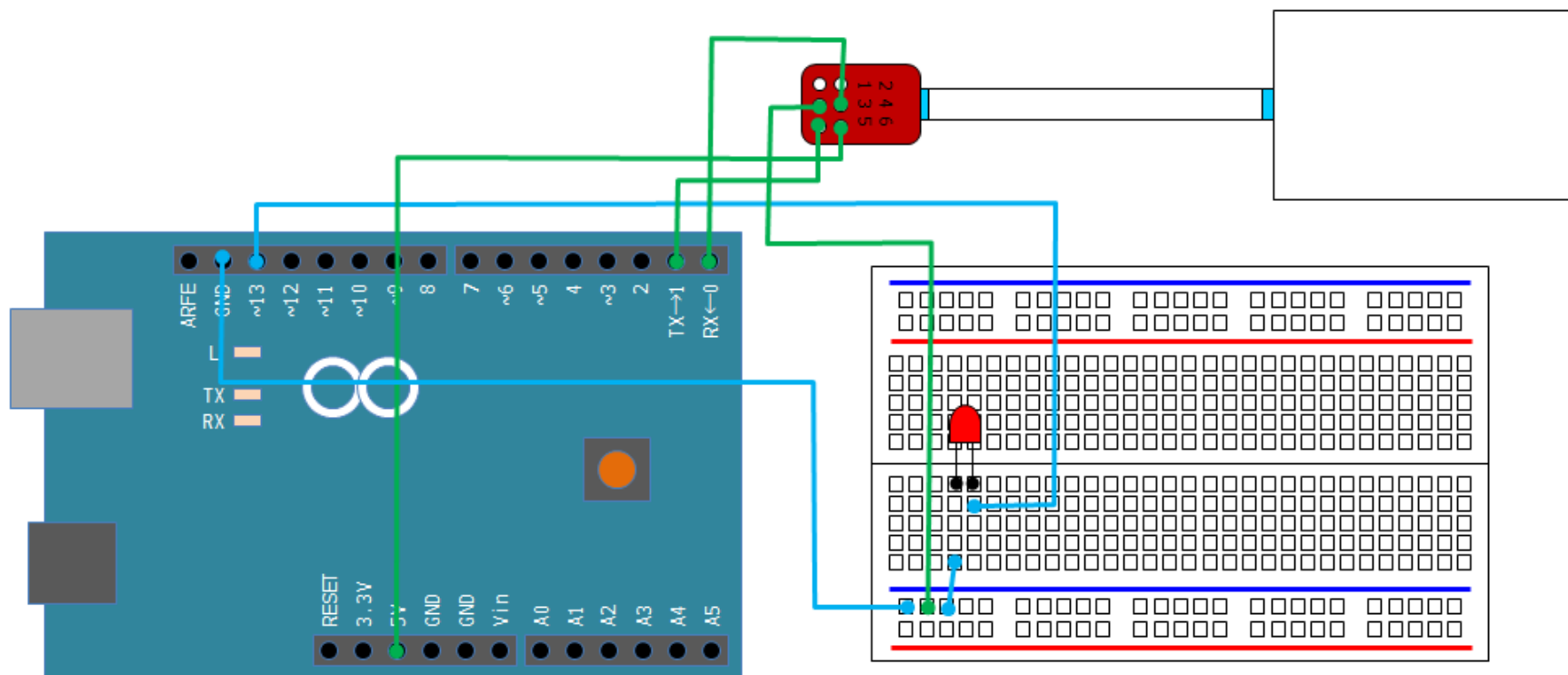
ライブラリ設定

- ダウンロードしたarduino-RCS620S.zipを解凍。
- 解凍したフォルダを[RCS620S]にリネーム
 - フォルダ名が[Arduino-RCS620S]の場合、以下のようなメッセージがArduino IDE起動時に表示されます。

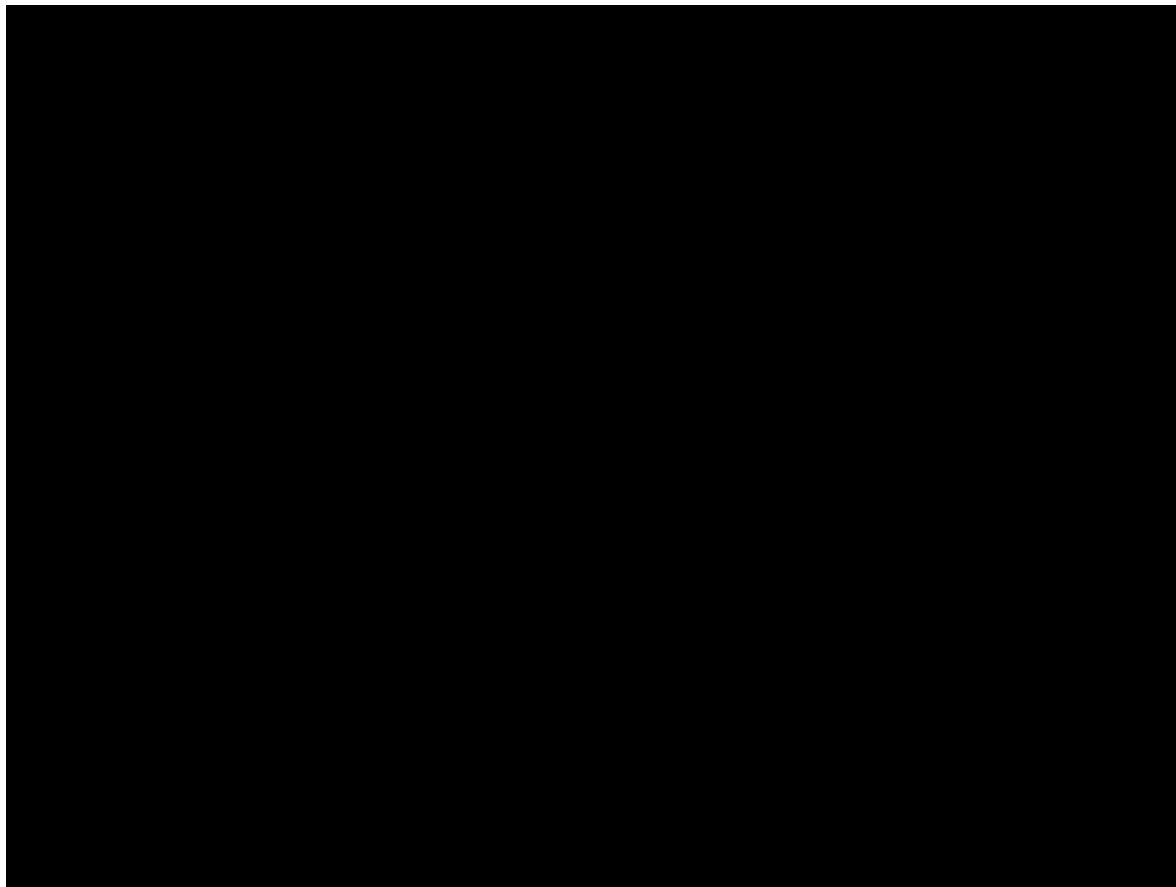


- Arduino IDEフォルダ内[libraries]フォルダへ[RCS620S]を移動

接続はこんな感じです。



サンプルプログラムの実行



あ、もう終わりです・・・

FeliCaを読んでいるかどうかわからん・・・

ちゃんと読み込んだときに

IDm(Felica製造ID)

PMm(Felica製造パラメータ)

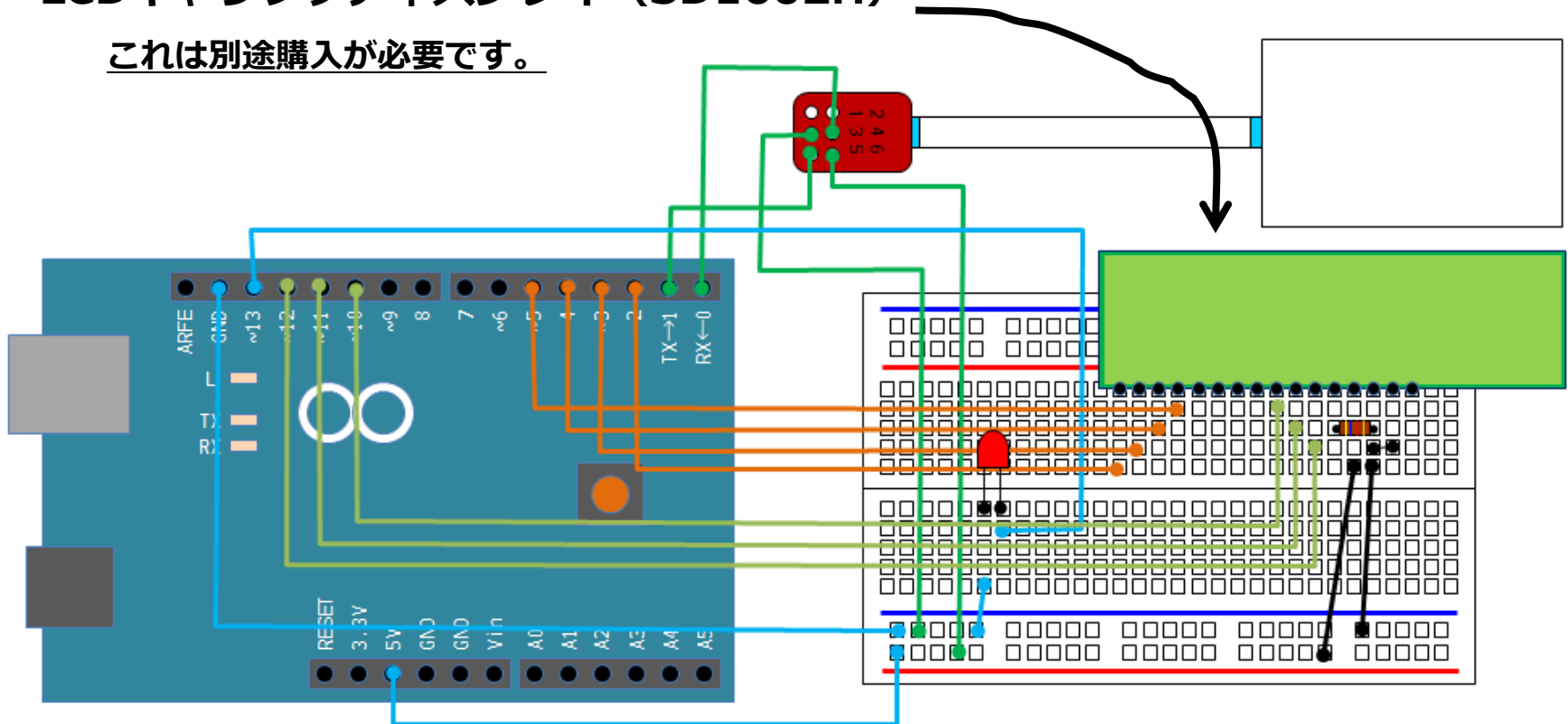
取得しています！

液晶ディスプレイに表示させて確認します

ちょっと複雑ですが

LCDキャラクタディスプレイ (SD1602H)

これは別途購入が必要です。



スケッチはexampleの中で、Felicaを読み込んだタイミングでLCD
キャラクタディスプレイに書き込む処理を追加します。

と言われても？？？ですよ

□ 簡単に説明します！

- キャラクタディスプレイ用ライブラリ (LiquidCrystal.h) をinclude

```
#include<LiquidCrystal.h>
```

- ライブラリの初期化 (グローバルで)

```
LiquidCrystal lcd(8, 7, 6, 5, 4, 3, 2);
```

- LCDの初期化と表示

```
lcd.begin(2, 16);
```

2列16桁って意味です。

```
lcd.clear();
```

```
lcd.setCursor(0, 1);
```

引数 1、桁番号 (0が1番目)
引数 2、列番号 (0が1番目)

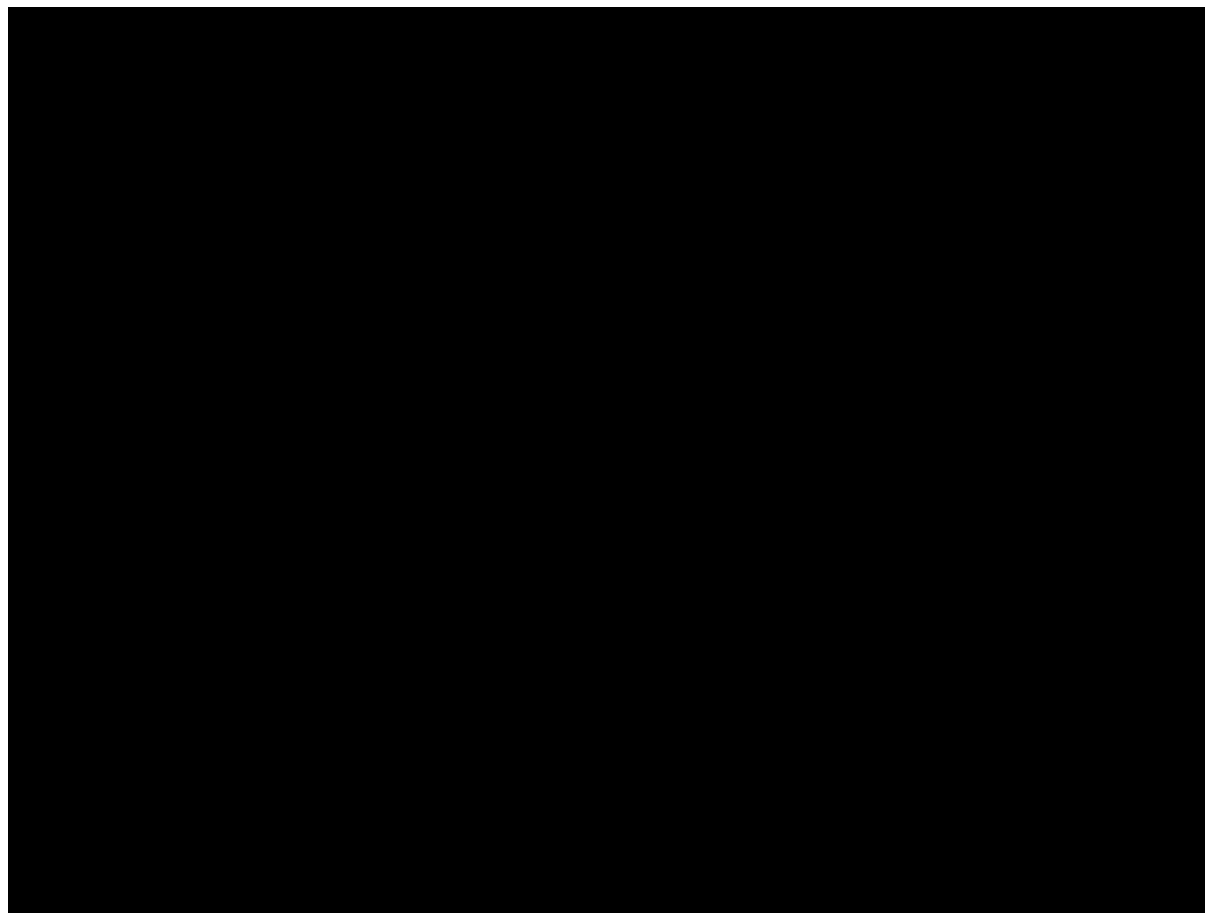
```
lcd.write(0x41);
```

表示する文字
0x41は「A」

- こんな感じです。
- 簡単でしたよね？



で、出来たのがこんな感じ



よりお手軽に始めるために

お店によってはエントリーキットなどが販売されていて、
何をしたいのかわからない人でもとっつきやすいかも！

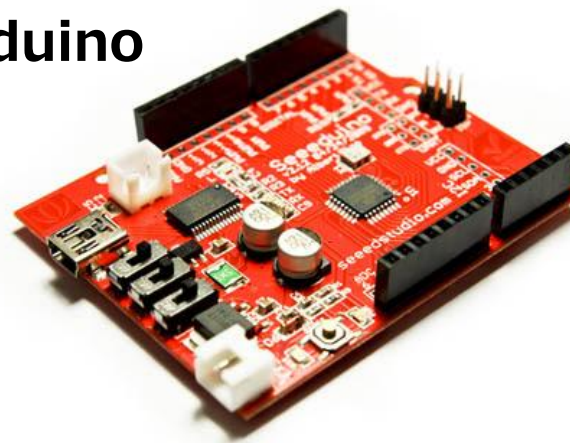


これは若松通商さんのエントリーキットです。

その他のArduino ?

- Arduinoはオープンソース

Seeeduino



Netduino



Seeeduino Film



**.NET Micro Frameworkを使っ
てソフトウェアを開発！**

最後に

- 日本語リファレンスもある
 - <http://www.musashinodenpa.com/arduino/ref/>
- 専門家でなくとも簡単に実現可能
- 何かを実現するためのインタフェース
- そして、自慢する為に！



ご静聴ありがとうございました。