

デジタル入力…タクトスイッチ（ボタン）を使う

《何ができる？》

ボタンのオン・オフの入力信号をプログラム内で検知できれば、それをクラウドに送り商品購入サービスにつなげることができるようになる。

（図は Amazon Dash ボタン）



デジタル入力に必要な関数

<code>pinMode (pin, mode)</code>	
ピンの動作を入力か出力に設定する。	
【パラメータ】	<code>pin</code> : 設定したいピンの番号 <code>mode</code> : INPUT、OUTPUT、INPUT_PULLUP
【戻り値】	なし
<code>digitalRead (pin)</code>	
指定したピンの値を読み取る。その結果は HIGH または LOW となる。	
【パラメータ】	<code>pin</code> : 読みたいピンの番号
【戻り値】	HIGH または LOW

《サンプル①》 タクトスイッチのオン（HIGH）・オフ（LOW）情報をシリアルモニタに出力する。


```
1 void setup() {  
2   Serial.begin(115200);  
3   pinMode(17, INPUT);  
4 }  
5  
6 void loop() {  
7   int b = digitalRead(17);  
8   Serial.println(b);  
9   delay(10);  
10 }
```

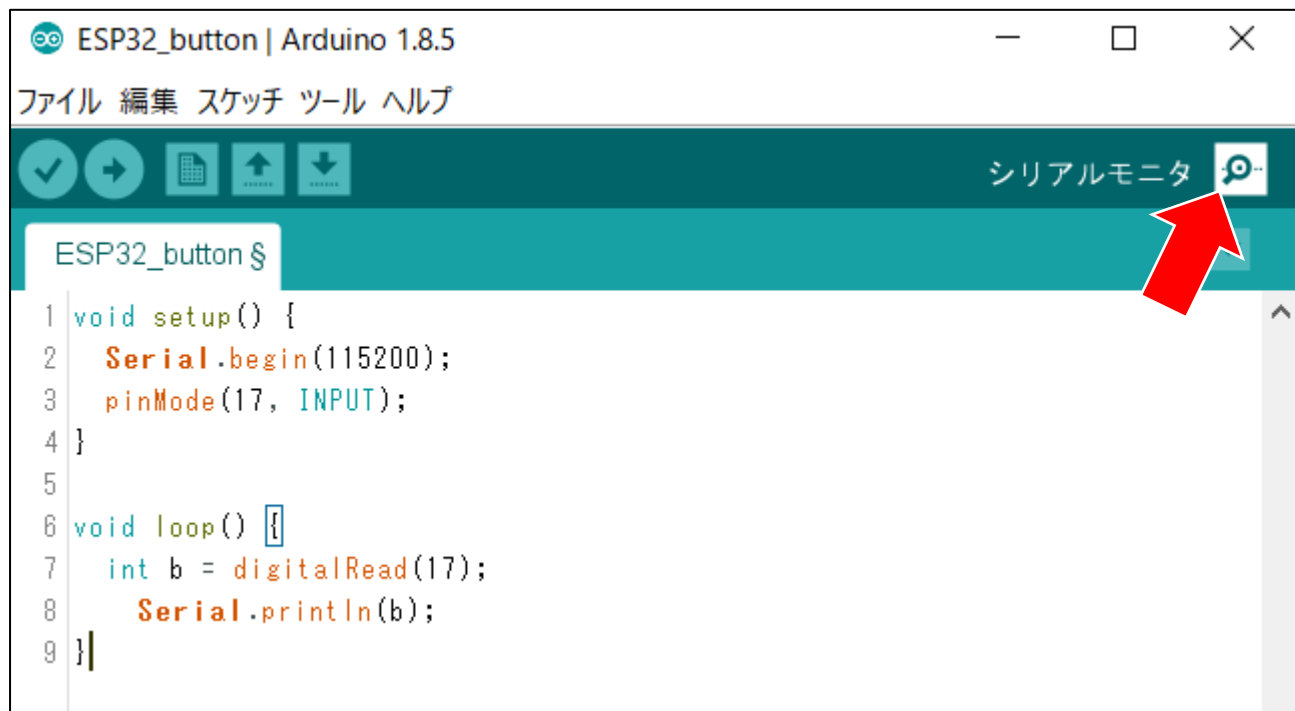
シリアル出力に必要なシリアル通信関数

<code>Serial.begin (speed)</code>	
シリアル通信のデータ転送レートを bps (baud) で指定する。bps はビット/秒。コンピュータと通信する際は、次のレートから 1 つを選ぶ。 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200	
【パラメータ】	<code>speed</code> : 転送レート (int)

【戻り値】	なし
<code>Serial.println(pindata, format)</code>	
データの末尾にキャリッジリターン (ASCII コード 13 あるいは '¥r') とニューライン (ASCII コード 10 あるいは '¥n') を付けて送信する。	
【パラメータ】	<code>data</code> : すべての整数型と String 型 <code>format</code> : data を変換する方法を指定する (省略可)
【戻り値】	送信したバイト数 (byte)

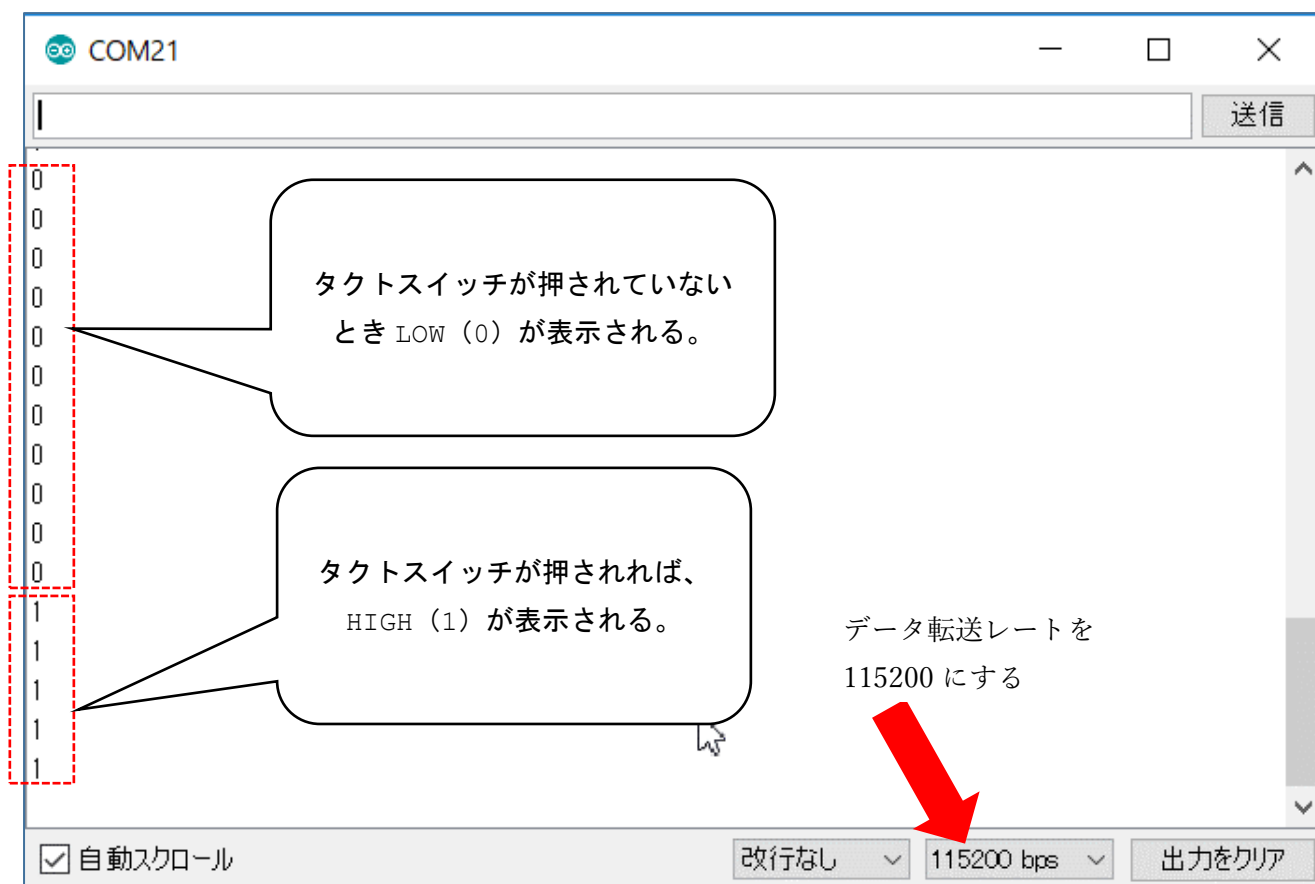
シリアル出力は、以下の方法で表示させることができる。

Arduino IDE の右上のシリアルモニタのアイコンをクリックする。



以下のシリアルモニタ画面が表示されるので、`Serial.begin(speed)` で示したデータ転送レート（ここでは 115200）を選ぶ。

データ転送レートが一致すれば、以下の様な出力が表示される。

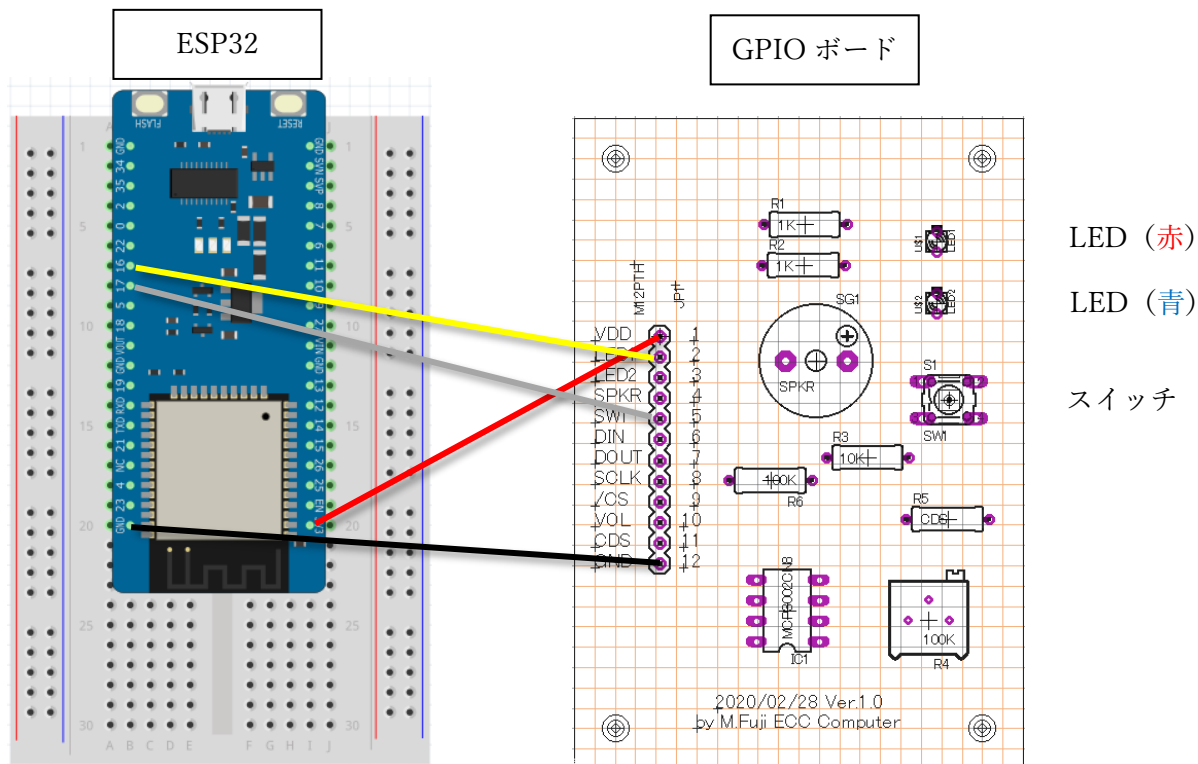


演習では、タクトスイッチの配線を行い、上記サンプルを動かしてみよう。

第2回 演習

【第2回の目標】

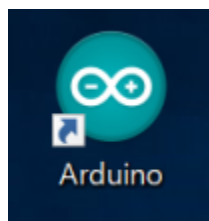
ESP32 ボードにタクトスイッチをつなぎ、スイッチのオン・オフを検知できるようにする。



上記実体配線図を元に、5本のジャンパーコードの配線を行う。接続するピンは以下の一覧の通り。

- ① ジャンパーコード (黄) ESP32 の I016 - GPIO ボード LED1 (2 番ピン)
- ② ジャンパーコード (白) ESP32 の I017 - GPIO ボード SW1 (5 番ピン)
- ③ ジャンパーコード (赤) ESP32 の 3V3 - GPIO ボード VDD (1 番ピン)
- ④ ジャンパーコード (黒) ESP32 の GND - GPIO ボード GND (12 番ピン)

【2. Arduino スケッチのサンプルプログラムをプログラム実行】

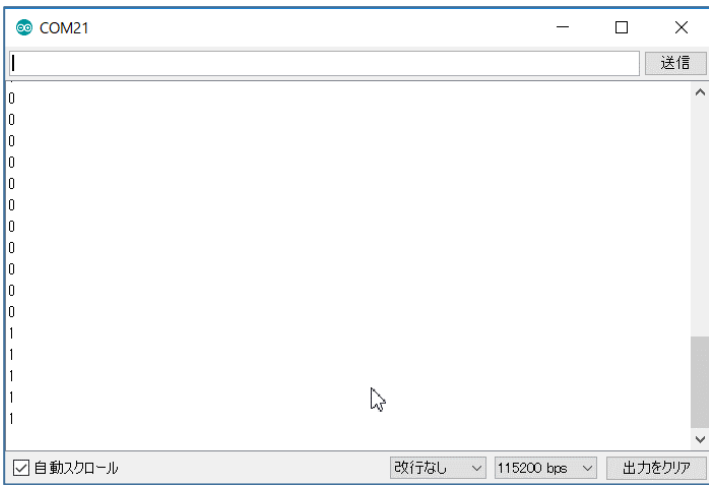


デスクトップのアイコンをダブルクリックして
Arduino IDE を起動する。

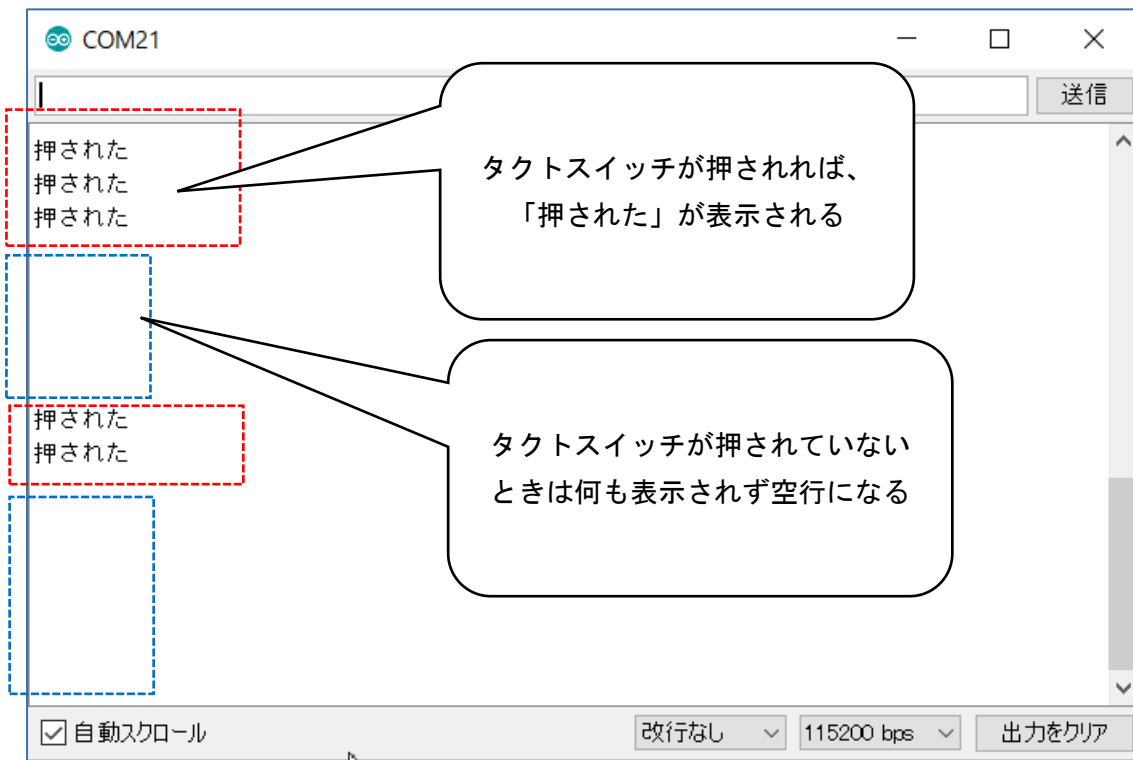
【課題 04】プロジェクト名「kad04_Switch」

《サンプル①》を打ち込んで、動作確認をする

```
1 void setup() {  
2   Serial.begin(115200);  
3   pinMode(17, INPUT);  
4 }  
5  
6 void loop() {  
7   int b = digitalRead(17);  
8   Serial.println(b);  
9   delay(10);  
10 }
```



タクトスイッチを押すと、シリアルモニタの表示が「1」になり、タクトスイッチを押さないときは、表示が「0」になるのを確認する。うまく動いているのを確認したら、スケッチ（Arduino プログラム）を書き加えて、以下のようなシリアルモニタの表示になるようにせよ。



【課題 05】 プロジェクト名「kad05_SwitchLED」

I017 番ピンに接続したタクトスイッチを押したら、I016 番ピンに接続した LED が点灯するスケッチを作成せよ。

【課題 06】 プロジェクト名「kad06_SwitchSOS」

I017 番ピンに接続したタクトスイッチを押したら、I016 番ピンに接続した LED で SOS のモールス信号が 1 回のみ点滅するようなスケッチを作成せよ。

単音「・」 200 ミリ秒点灯

長音「ー」 600 ミリ秒点灯

消灯 200 ミリ秒

SOS のモールス信号「・ ・ ・ — — — ・ ・ ・」（ト ト ト ツー ツー ツー ト ト ト）

【課題 07】 プロジェクト名「kad07_Count」

I017 番ピンに接続したタクトスイッチを押したら、シリアルモニタに 1 2 3 …と 1 から改行された形でタクトスイッチを押すたびに 1 ずつカウントアップするスケッチを作成せよ。

普通にスケッチを記述するとチャタリングといって、少しでもスイッチを押しても 2 以上カウントアップされてしまう。押した状態をフラグ変数に持っておいてチャタリングを防止する必要がある。

ヒント（スケッチの枠組みのソース部分）	シリアルモニタ
<code>const int swPin = 17; //const...初期化した後代入されない</code>	1 スイッチがおされたときだけ
<code>int count = 0; //カウンタ変数</code>	2 カウンタの値が表示される
<code>int flag = 0; //フラグ初期値 0:押されていない状態</code>	3
	4
<code>void setup() {</code>	5
<code>Serial.begin(115200);</code>	6
<code>pinMode(swPin, INPUT); // Switch デジタル入力</code>	7
<code>}</code>	8
	9
<code>void loop() {</code>	10
<code>int b = digitalRead(swPin);</code>	11
<code>/*</code>	...
ここにフラグ変数 flag を活用して、スイッチがおされたときだけ	
カウンタ変数 count がインクリメントされ表示されるソースをかく	
<code>*/</code>	
<code>delay(10);</code>	
<code>}</code>	