# 演習実験　教材

2022/6/13 実施分

問 1~問 5 を実行しましょう。

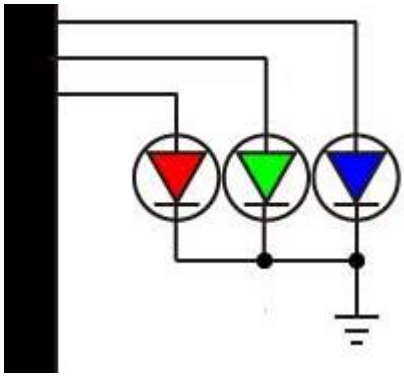
問 1．　3 色 LED を制御しましょう。

【使用機材】

3 色 LED、抵抗

3 色 LED は以下のような配線構造になっています。

LED を利用するには抵抗を付ける必要がありますが、手元に抵抗が 1 つしか準備されていません。



[1]

以下のプログラムが実行できるように配線を完成させてください。

【プログラム】

```
//問 1. 3 色 LED の制御

//ピン番号
int redLedPin    = 9;   //赤色 LED ピン番号
int blueLedPin   = 10; //青色 LED ピン番号
int greenLedPin = 11; //緑色 LED ピン番号

//明るさレベル(0~255 の 256 段階)
int redLedBrt    = 255;   //赤色 LED
int blueLedBrt   = 255; //青色 LED
int greenLedBrt = 128; //緑色 LED
```

```
void setup() {
   //ピンモードを出力に設定
   pinMode(redLedPin,OUTPUT);
   pinMode(blueLedPin,OUTPUT);
   pinMode(greenLedPin,OUTPUT);
}//続く
void loop() {
   //LED を順次点灯
   analogWrite(redLedPin,redLedBrt);
   delay(1000);
   analogWrite(blueLedPin,blueLedBrt);
   delay(1000);
   analogWrite(greenLedPin,greenLedBrt);
   delay(1000);
   analogWrite(redLedPin,0);
   delay(1000);
   analogWrite(blueLedPin,0);
   delay(1000);
   analogWrite(greenLedPin,0);
   delay(1000);
   //LED を 1 色ずつ点灯
   analogWrite(redLedPin,redLedBrt);
   delay(500);
   analogWrite(redLedPin,0);
   analogWrite(blueLedPin,blueLedBrt);
   delay(500);
   analogWrite(blueLedPin,0);
   analogWrite(greenLedPin,greenLedBrt);
   delay(500);
   analogWrite(greenLedPin,0);
   delay(1000);

}
```

【ヒント】

コモンピン（3 つの LED すべてとつながっているピン）は一番長いピンです。それ以外のピンは Arduino に接続をすることで確かめていきましょう。

問２．　ボタンを使いこなしましょう。

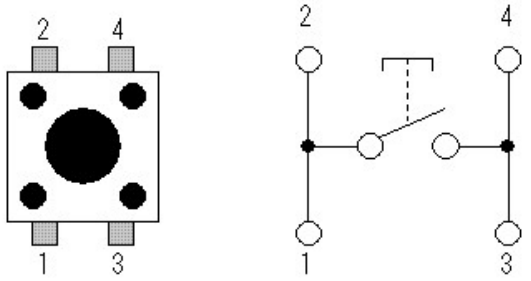
問 1 のコードを変えずに、ボタンを押している間だけ LED が消灯している回路を完成させてください。

【使用機材】

3 色 LED、抵抗、ボタン

【ボタンについて】

タクトボタンの構成は以下の通りです。




a）配線の接続　　b）内部接続

【プログラム】

問 2 のプログラムは問 1 と共通です。

【ヒント】

プルアップ回路・プルダウン回路について調べてみましょう。

問３．　角度・加速度センサを使いこなそう

SPI 通信を使ったセンサの回路を完成させましょう。完成したら実行し、シリアルモニタ(メニュータブのツールにあり)で角度と加速度が正しく取れていることを確認しましょう。

【使用機材】

角度加速度センサ ADXL345 ※動作電圧は 3.3V

【SPI 通信とは？】

SPI 通信はマイクロコントローラ（Arduino 等）と周辺 IC（センサ等）におけるシリアル通信の 1 つです。フォーマットや原理が簡単なため、高速で通信できます。4 本のワイヤーを接続することで通信を行います。電源と合わせて計 6 本のワイヤーを接続します。同時に複数の周辺 IC を接続することができます。

- SCKL (Serial Clock) :クロック同期を取る
- SS (Serial Select) :Master-Slave 関係におけるマスター(親）を決める
- MOSI (Master Out – Slave In) :マスター→スレーブへのデータ送信
- MISO (Master In – Slave Out) :スレーブ→マスターへのデータ送信

- VCC :電源
- GND :接地

【プログラム】[2]

```
#include <SPI.h>

// XYZ レジスタ用のテーブル(6byte)
uint8_t RegTbl[6];

void setup() {
  // SPI の初期化
  // ※自動的に「SCK、MOSI、SS のピンの動作は OUTPUT」となり「SS は HIGH」となる
  SPI.begin();
  // SPI 転送モード
  // クロック位相(CPOL) = 1 クロック極性(CPHA) = 1
  SPI.setDataMode(SPI_MODE3);
  // SPI 送受信用のビットオーダー(MSBFIRST)
  SPI.setBitOrder(MSBFIRST);

  // DATA_FORMAT(データ形式の制御)
  digitalWrite(SS, LOW);
    // DATA_FORMAT のアドレス
```

```
    SPI.transfer(0x31);
    //「最大分解能モード」及び「±16g」(0x0B == 1011)
    SPI.transfer(0x0B);
    //「10bit 固定分解能モード」及び「±16g」にする場合 (0x03 == 0011)
    // SPI.transfer(0x03);
  digitalWrite(SS, HIGH);

  // POWER_TCL(節電機能の制御)
  digitalWrite(SS, LOW);
    // POWER_CTL のアドレス
    SPI.transfer(0x2d);
    //   測定モードにする
    SPI.transfer(0x08);
  digitalWrite(SS, HIGH);

  Serial.begin(9600);
}

void loop() {
  // XYZ データの取得
  digitalWrite(SS, LOW);
    // XYZ の先頭アドレス(0x32)に移動する
    // ※複数バイトを読み込む際に必要なマルチバイト・ビットを加算
    SPI.transfer(0x32 | 0x40 | 0x80);
    // 6byte のデータを取得する
    for (int i = 0; i < 6; i++) {
      RegTbl[i] = SPI.transfer(0x00);
    }

  digitalWrite(SS, HIGH);

  //  データを各 XYZ の値に変換する(LSB 単位)
  int16_t x   = ((int16_t)RegTbl[1] << 8) | RegTbl[0];
  int16_t y   = ((int16_t)RegTbl[3] << 8) | RegTbl[2];
  int16_t z   = ((int16_t)RegTbl[5] << 8) | RegTbl[4];

  //  各 XYZ 軸の加速度(m/s^2)を出力する
  Serial.print("X : ");
  Serial.print( x * 0.0392266 );
  Serial.print(" Y : ");
  Serial.print( y * 0.0392266 );
  Serial.print(" Z : ");
  Serial.print( z * 0.0392266 );
  Serial.println(" m/s^2");

  delay(100);
}
```

【ヒント】

以下、SPI.h の公式リファレンスから引用です。

*SPI 通信で使われるピンは次のとおりです。*

*「中略」*

*Arduino Uno（ATmega168/328 を搭載するボード）: 10（SS）、11（MOSI）、12（MISO）、13（SCK）*

問４．　温湿度・気圧センサを使いこなそう

I2C 通信を使ったセンサの回路を完成させましょう。完成したら実行し、シリアルモニタ(メニュータブのツールにあり)で各指標が正しく取れていることを確認しましょう。

【使用機材】

角度加速度センサ ADXL345

【I2C 通信とは？】

- I2C 通信は SPI 通信同様に、マイクロコントローラ（Arduino 等）と周辺 IC（センサ等）におけるシリアル通信の１つです。Master がクロックを操作し、Slave 側がクロックに合わせてデータを送受信する仕組みです。

  I2C 通信は 2 本のワイヤーを接続することで通信を行います。電源と合わせて計 4 本のワイヤーを接続します。複数台繋ぐときも信号線が２本でいいことが特徴です。

  - SCL (Serial Clock) :クロック同期を取る。クロック線。
  - S DA (Serial Data) :データの送出を行う。データ線。

  - VCC :電源
  - GND :接地

【プログラム】（スケッチ例:bme680test を改変）

```
/***************************************************************************
  This is a library for the BME680 gas, humidity, temperature & pressure sensor

  Designed specifically to work with the Adafruit BME680 Breakout
  ----> http://www.adafruit.com/products/3660

  These sensors use I2C or SPI to communicate, 2 or 4 pins are required
  to interface.

  Adafruit invests time and resources providing this open source code,
  please support Adafruit and open-source hardware by purchasing products
  from Adafruit!

  Written by Limor Fried & Kevin Townsend for Adafruit Industries.
  BSD license, all text above must be included in any redistribution

 ***************************************************************************/

#include <Wire.h>
#include <SPI.h>
```

```
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"


#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME680 bme; // I2C

void setup() {
  Serial.begin(9600);
  while (!Serial);
  Serial.println(F("BME680 test"));

  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }

  // Set up oversampling and filter initialization
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms
}

void loop() {
  if (! bme.performReading()) {
    Serial.println("Failed to perform reading :(");
    return;
  }
  Serial.print("Temperature = ");
  Serial.print(bme.temperature);
  Serial.println(" *C");

  Serial.print("Pressure = ");
  Serial.print(bme.pressure / 100.0);
  Serial.println(" hPa");

  Serial.print("Humidity = ");
  Serial.print(bme.humidity);
  Serial.println(" %");

  Serial.print("Gas = ");
  Serial.print(bme.gas_resistance / 1000.0);
  Serial.println(" KOhms");

  Serial.print("Approx. Altitude = ");
```

```
    Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
    Serial.println(" m");

    Serial.println();
    delay(2000);
}
```

【ヒント】

Arduino の側面や本体の文字を確認してみましょう。

問５．　ボタンで温湿度センサと加速度センサの動作を切り替えよう

通常状態で温湿度が、ボタンを押している間は加速度センサが
シリアルモニタで確認できるような回路を完成させましょう。
（発展：ここまでできたら LED でも押下状況を確認できるように回路を組み替えてみましょう。）

【プログラム】

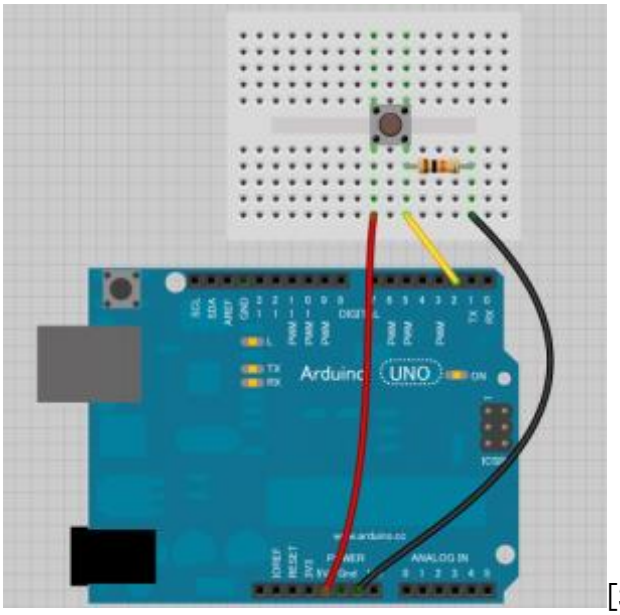
```
#include <SPI.h>
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"

//Button
#define switch_pin 3
//BME680
#define SEALEVELPRESSURE_HPA (1013.25)
Adafruit_BME680 bme;
// ADXL345_XYZ(6byte)
uint8_t RegTbl[6];

void setup() {
  //スイッチ・シリアルセットアップ
  pinMode(switch_pin,INPUT);
  Serial.begin(9600);
  //ADXL345 セットアップ
  SPI.begin();
  SPI.setDataMode(SPI_MODE3);
  SPI.setBitOrder(MSBFIRST);
  digitalWrite(SS, LOW);
  SPI.transfer(0x31);
  SPI.transfer(0x0B);
  digitalWrite(SS, HIGH);
  digitalWrite(SS, LOW);
  SPI.transfer(0x2d);
  SPI.transfer(0x08);
  digitalWrite(SS, HIGH);
  //BME680 セットアップ
  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }
  bme.setTemperatureOversampling(BME680_OS_8X);
```

```
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms
}

void loop() {
  // put your main code here, to run repeatedly:
  if(digitalRead(switch_pin)){
    loop_ BME680();
  }else{
    loop_ ADXL345();
  }
  delay(100);
}
void loop_BME680(){
    if (! bme.performReading()) {
    Serial.println("Failed to perform reading :(");
    return;
  }
  Serial.print("Temperature = ");
  Serial.print(bme.temperature);
  Serial.println(" *C");

  Serial.print("Pressure = ");
  Serial.print(bme.pressure / 100.0);
  Serial.println(" hPa");

  Serial.print("Humidity = ");
  Serial.print(bme.humidity);
  Serial.println(" %");

  Serial.print("Gas = ");
  Serial.print(bme.gas_resistance / 1000.0);
  Serial.println(" KOhms");

  Serial.print("Approx. Altitude = ");
  Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
  Serial.println(" m");
  delay(500);
}
void loop_ADXL345(){
  digitalWrite(SS, LOW);
  SPI.transfer(0x32 | 0x40 | 0x80);
  for (int i = 0; i < 6; i++) {
    RegTbl[i] = SPI.transfer(0x00);
  }
  digitalWrite(SS, HIGH);
```

```
    int16_t x   = ((int16_t)RegTbl[1] << 8) | RegTbl[0];
    int16_t y   = ((int16_t)RegTbl[3] << 8) | RegTbl[2];
    int16_t z   = ((int16_t)RegTbl[5] << 8) | RegTbl[4];

    Serial.print("X : ");
    Serial.print( x * 0.0392266 );
    Serial.print(" Y : ");
    Serial.print( y * 0.0392266 );
    Serial.print(" Z : ");
    Serial.print( z * 0.0392266 );
    Serial.println(" m/s^2");
}
```

【ヒント】

ボタンを使う際の基本の形は以下の図の通りです。この場合、ボタンを押下すると 1→0 になります。



[3]

引用 :

[1] h ttps://thunderblog.org/2018/12/led_common.html

[2] h ttps://www.petitmonte.com/robot/howto_adxl345_spi.html

[3] h ttps://mag.switch-science.com/2013/05/23/input_pullup/