1 演習の目的

実験を通して、フルカラー LED OSTA5131A の使い方と仕組みの習得を目的とする。

2 演習の使用部品

2.1 図 1 の電子部品 (フルカラー LED OSTA5131A) を次のような点から調べなさい。

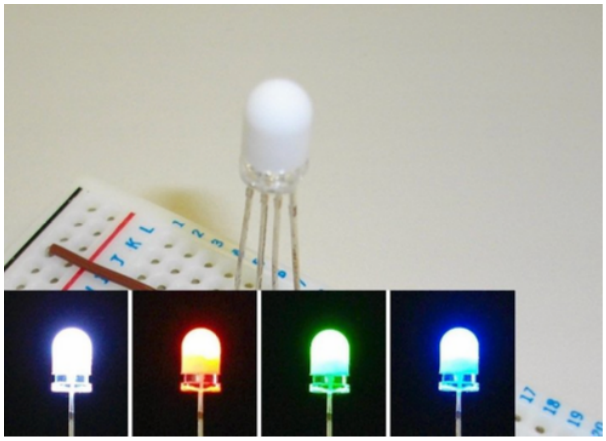

図 1: LED ダイオード

2.1.1 どのような部品か

オプトサプライのフルカラー RGBLED。発光色を混ぜるとフルカラーを表現可能となる [1] 。

2.1.2 どのような仕組みか

フルカラー LED には、それぞれ赤、緑、青で発光する半導体の小さな板（LED チップ）が入っており、それぞれの LED チップに流す電流の大きさを変えてそれぞれの色の光の強度を変え、3 色の混合割合を変えると、発光色を変化させる 。

2.1.3 どのような入力を取り扱うのか

電流を入力として取り扱う。

2.1.4 入力に応じて出力がどう変化するのか（データシートや仕様書を参考に）

OptoSupply によると、赤色は 2.8V ぐらいを加えると最大光量に達成するが、緑と青色は、3.8V ぐらいを加えると最大光量に達成する [2] 。 図 2 は、電圧が増加すると光量が増加する。グラフの形としては指数関数的に増加している。

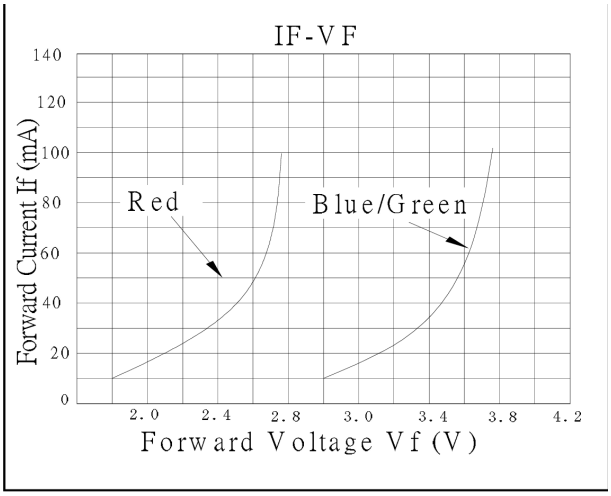

図 2: LED ダイオードグラフ

2.1.5 どのようなピンアサイン（各ピンの役割）か

OptoSupply によると、LED のピンアサインは、緑が 1 番、青が 2 番、GND が 3 番, 赤が 4 番である [2] 。 図 3 は、ピンアサインを示している。

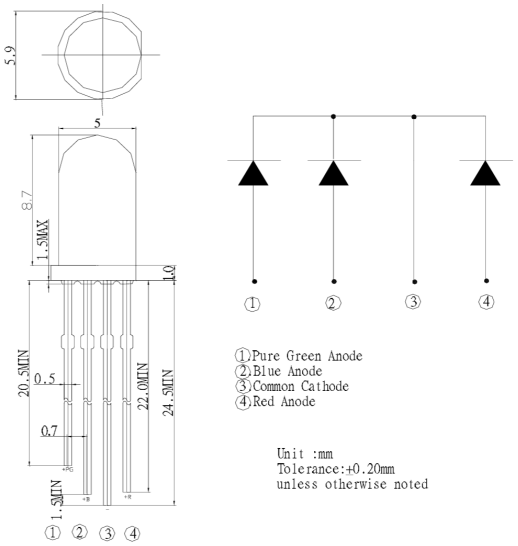

図 3: LED ダイオードピンアサイン

2.1.6 正しい動作の条件，範囲は何か

秋月電子通商によると、以下の通りとなる [1] 。

- 種別:砲弾型
- 色:赤・緑・青
- ドミナント波長:赤 635nm・緑 525nm・青 470nm
- ドミナント波長赤:635nm
- ドミナント波長緑:525nm
- ドミナント波長青:470nm

- 光度:赤 2000mcd・緑 7000mcd・青 2500mcd
- 光度赤:2000mcd
- 光度緑:7000mcd
- 光度青:2500mcd
- 順電圧:赤 2V・緑 3.6V・青 3.6V
- 順電圧赤:2V
- 順電圧緑:3.6V
- 順電圧青:3.6V
- 順電流 max.:赤 30mA・緑 30mA・青 30mA
- 順電流 max.赤:30mA
- 順電流 max.緑:30mA
- 順電流 max.青:30mA
- 逆電圧:赤 5V・緑 5V・青 5V
- 逆電圧赤:5V
- 逆電圧緑:5V
- 逆電圧青:5V
- 許容損失 max.:赤 75mW・緑 105mW・青 105mW
- 許容損失 max.赤:75mW
- 許容損失 max.緑:105mW
- 許容損失 max.青:105mW
- 半減角:30°
- 動作温度 min.:−30℃
- 動作温度 max.:85℃
- 構成:カソードコモン
- 端子部形状:ピン
- 実装タイプ:スルーホール
- 長さ:8.7mm
- 径:5mm

3 課題内容

3.1 フルカラー LED を光らせる

3.1.1 実験その 1（動作確認）

アナログ出力は以下の表 4.2 のようになる。 表に従いフルカラー LED の各ピンへのアナログ出力値に対してフルカラー LED が何色に発光するかを確認しなさい。 あわせて 表 1 の空欄箇所を埋めなさい。

アナログ出力			状態
赤	緑	青	
255	255	255	白
0	0	0	黒
255	0	0	赤
			緑
			水色
255	0	255	
255	100	0	

表 1: フルカラー LED の発光色のアナログ出力値

回路図

図 4 は、実験 1 の回路図を示す。 フルカラー LED の各ピンにアナログ出力を入力し、フルカラー LED を光らせる。 間に抵抗を入れて、電流を制限して LED を保護する。

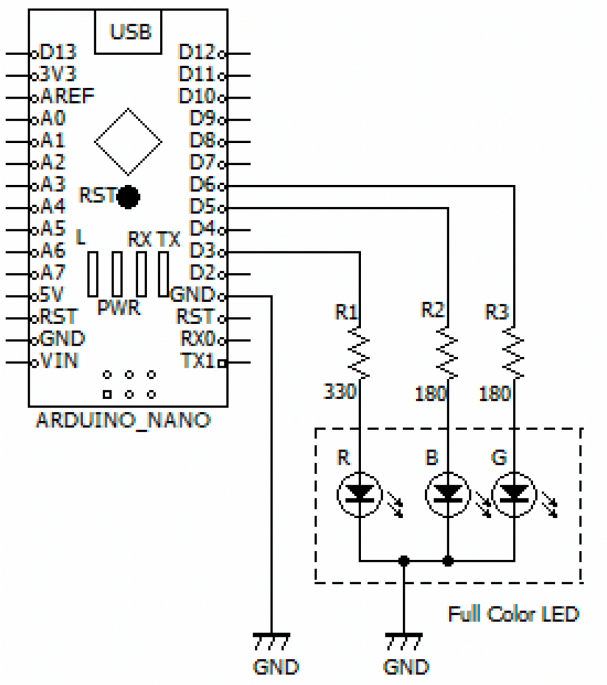

図 4: 実験 1 回路図

プログラム

リスト 1 は、実験 1 で使用したアナログ出力値を切り替えるプログラムを示している。

```
1  void setup(){
2    pinMode(3,OUTPUT);
3    pinMode(5,OUTPUT);
4    pinMode(6,OUTPUT);
5  }
6
7  void loop(){
8    analogWrite(3,255); // R
9    analogWrite(5,255); // B
10   analogWrite(6,255);  // G
11   delay(1000);
12 }
```

リスト 1: アナログ出力値を切り替えるプログラム

- プログラムの概要

アナログ出力値を切り替えて、フルカラー LED を光らせるプログラムである。

- プログラムの説明
  - 1–5 行目: ピンの設定
    - 2, 3, 4 行目でそれぞれ赤、青、緑のピンを出力に設定している。
  - 7–12 行目：プログラムの動作
    - 8, 9, 10 行目でそれぞれ赤、青、緑のアナログ出力値を 255 に設定している。
    - ここの値を手動で変更し、フルカラー LED の発光色を変更できる。

実験結果

表 2 は、フルカラー LED の発光色のアナログ出力値の結果を示している。 全てのピンに 255 を入力したときに白色に発光するため、光の三原色である赤、緑、青の光を混ぜると白色になる現象と同一なため、他の色も光の三原色を利用して色の表現が可能になるとわかる。 また、赤色については、かなり他の色に比べて強く色が発色した。

アナログ出力			状態
赤	緑	青	
255	255	255	白
0	0	0	黒
255	0	0	赤
0	255	0	緑
0	255	255	水色
255	0	255	紫色
255	100	0	黄色

表 2: フルカラー LED の発光色のアナログ出力値の結果

考察

赤色については、かなり他の色に比べて強く色が発色する。 節 2.1.4 より、グラフを参照すると、電圧が増加すると光量が増加する。 そのときに、赤色は 2.8V ぐらいを加えると最大

光量に達成するが、緑と青色は、3.8V ぐらいを加えると最大光量に達成する。 そのため、赤色は他の色に比べて強く発色すると考えられる。

3.1.2 発展その 1(ランダム関数の導入)

1 秒ごとに、フルカラー LED の発光色がランダムに変わるようにしなさい。

回路図・プログラム

回路図は実験その 1 と同じものを使用する。 リスト 2 は、ランダムで LED を光らせるプログラムを示している。

```
1  void setup(){
2    pinMode(3,OUTPUT);
3    pinMode(5,OUTPUT);
4    pinMode(6,OUTPUT);
5  }
6
7  void loop(){
8
9    int r = random(0,255);
10   int b = random(0,255);
11   int g = random(0,255);
12
13   analogWrite(3,r); // R
14   analogWrite(5,b); // B
15   analogWrite(6,g);  // G
16   delay(1000);
17 }
```

リスト 2: ランダムで LED を光らせるプログラム

- プログラムの概要

アナログ出力値を切り替えて、フルカラー LED を光らせるプログラムである。 乱数を用いて、フルカラー LED の発光色をランダムに変更する。

- プログラムの説明
  - 1–5 行目: ピンの設定
    - 2, 3, 4 行目でそれぞれ赤、青、緑のピンを出力に設定している。
  - 7–17 行目：プログラムの動作
    - 9–11 行目でそれぞれの発色をランダムに設定している。
    - 13–15 行目でそれぞれの色のアナログ出力値を設定している。
    - 16 行目で 1 秒待機している。

実験結果・考察

乱数を利用したため、様々な色を表現できるようになった。 Delay を無くすと色がすばやく点滅し、ほとんど白色に見える。 Delay を行うことにより、Delay を行う前までの処理を一時停止して出力を切らないまま止める。

3.1.3 発展その 2(map 関数の導入)

好きな色を 3 色選び、一定時間ごとにフルカラー LED が選んだ色に少しずつ変わるようにしなさい。 例: 青(B) → (中間色 M) →赤(R) → (中間色 Y) →緑(G) → (中間色 C) →青(B)

回路図・プログラム

回路図は実験その 1 と同じものを使用する。 リスト 3 は、グラデーションをかけて LED を光らせるプログラムを示している。

```
// ピンの指定
const int redPin = 3;
const int greenPin = 6;
const int bluePin = 5;
// ピンのセットアップ
void setup()
{
    pinMode(redPin, OUTPUT);
    pinMode(greenPin, OUTPUT);
    pinMode(bluePin, OUTPUT);
}

void loop()
{
    // グラデーションを行うループ
    fadeColor(0, 0, 255, 255, 0, 255, 2000);
    fadeColor(255, 0, 255, 255, 0, 0, 2000);
    fadeColor(255, 0, 0, 255, 255, 0, 2000);
    fadeColor(255, 255, 0, 0, 255, 0, 2000);
    fadeColor(0, 255, 0, 0, 255, 255, 2000);
    fadeColor(0, 255, 255, 0, 0, 255, 2000);
}

void fadeColor(int r1, int g1, int b1, int r2, int g2, int b2, int duration)
{
    // グラデーションの初期設定
    int steps = 255;
    // グラデーションを行う間隔を設定
    int stepDelay = duration / steps;

    // グラデーション開始
    for (int i = 0; i <= steps; i++)
    {
        // 0〜255 の切り替えを r1~r2 までの幅と同じスケールで変わっていくようにしてくれる
        int r = map(i, 0, steps, r1, r2);
        int g = map(i, 0, steps, g1, g2);
        int b = map(i, 0, steps, b1, b2);

        // 表示して
        analogWrite(redPin, r);
        analogWrite(greenPin, g);
        analogWrite(bluePin, b);
        // 短い間隔で止まる
        delay(stepDelay);
    }
}
```

リスト 3: グラデーションをかけて LED を光らせるプログラム

- プログラムの概要

アナログ出力値を切り替えて、フルカラー LED を光らせるプログラムである。 Map を用いて、綺麗なグラデーションを行なう、また、Delay を細かく行うことで、ちらつかずに滑らかに色を変える。

- プログラムの説明
  - 1–11 行目: ピンの設定
    - 2–4 行目でそれぞれ赤、青、緑のピン番号を設定している。
    - 8–10 行目でそれぞれ赤、青、緑のピンの出力を設定している。
  - 13–46 行目：プログラムの動作
    - 16–21 行目でそれぞれの発色のグラデーションを設定している。
    - 24–46 行目でグラデーションを行う関数を設定している。
    - 27 行目でグラデーションの初期設定を行っている。
    - 29 行目でグラデーションを行う間隔を設定している。
    - 32–44 行目でグラデーションを行う処理を行っている。
    - 35–37 行目で map 関数を用いて、範囲を揃えて再マッピングを行っている。
    - 40–42 行目でそれぞれの色のアナログ出力値を設定している。
    - 44 行目で Delay を行っている。

実験結果・考察

Map を用いると、綺麗にグラデーションを行える。 Map を用いると 0 255 の範囲と旧値と新値の範囲を揃えて再マッピングを行ない、上限までに到達する時間を揃えられるため、綺麗にグラデーションを行える。 Delay を細かく行うと、ちらつかずに滑らかに色を変えれる。

3.1.4 まとめ

LED は光の三原色を用いることでフルカラーを表示できる。 Random 関数を使うと色を複雑な順序で変えれる。 Map と細かい Delay を用いると綺麗なグラデーションを作れる。

参考文献

[1] 秋月電子通商, 「RGB フルカラー LED 5mm OSTA5131A カソードコモン」. 参照: 2024 年. [Online]. 入手先: https://akizukidenshi.com/catalog/g/g102476/

[2] OptoSupply, 「OSTA5131A-R/PG/B」. 参照: 2024 年. [Online]. 入手先: https://akizukidenshi.com/goodsaffix/OSTA5131A-RPGB.pdf