ハードウェア第 3 回レポート

坪井正太郎（101830245）

2020 年 9 月 9 日

1 課題 3-1

1.1 概要

この実験では，プログラム 1 を改変し，動作を確認することでシュミレータでの実験の基礎を体得する。

1.2 実験方法

ソースコード 1　program3-1

```
1  console.log("Start");
2  // onoff.Gpio ライブラリをロード
3  var Gpio = require("onoff").Gpio;
4  // ピン番号を指定して Gpio インスタンスを作成
5  // in / out は^^e2^^bc^^8a^^e2^^bc^^92に使うか出^^e2^^bc^^92に使うかの指定
6  1
7  var led = new Gpio(4, "out");
8  var button = new Gpio(12, "in");
9  // 下段 LED1: 4, 17, 27, 22
10 // 上段 LED1: 5, 6, 13, 19
11 // ボタン 1: 20，ボタン 2: 16，ボタン 3: 12
12 // 3 回までループ
13 for(i=0; i<3; i++){
14 // LED を 1 秒周期（点灯 0.5 秒，消灯 0.5 秒）で 5 回明滅
15 for(j=0; j<5; j++){
16 sleep(500);
17 led.writeSync(1);
18 sleep(500);
19 led.writeSync(0);
20 }
21 // 無限ループでボタン^^e2^^bc^^8a^^e2^^bc^^92を待つ（0: off, 1: on）
22 console.log("[Waiting button input]");
23 while(button.readSync() == 0) {
24 sleep(100);
25 }
```

```
26  console.log("Pushed!!");
27  }
28  console.log("Finished!");
```

もとのプログラムを改変して led の配置を変更，追加する。

ソースコード 2　program3-1d

```
1   console.log("Start");
2   var Gpio = require("onoff").Gpio;
3
4   var green_led1 = new Gpio(22, "out");
5   var green_led2 = new Gpio(19, "out");
6   var button = new Gpio(16, "in");
7   // 下段 LED1: 4, 17, 27, 22
8   // 上段 LED1: 5, 6, 13, 19
9   // ボタン 1: 20, ボタン 2: 16, ボタン 3: 12
10  for (i = 0; i < 3; i++) {
11    // LED を 1 秒周期（点灯 0.5 秒，消灯 0.5 秒）で 5 回明滅
12    for (j = 0; j < 5; j++) {
13      sleep(500);
14      green_led1.writeSync(1);
15      green_led2.writeSync(1);
16      sleep(500);
17      green_led1.writeSync(0);
18      green_led2.writeSync(0);
19    }
20    console.log("[Waiting button input]");
21    while (button.readSync() == 0) {
22      sleep(100);
23    }
24    console.log("Pushed!!");
25  }
26  console.log("Finished!");
```

1.3　実験結果

1 を実行した結果 LED が明滅し，ボタン入力が正しく受け取られた。

2 を実行した結果，明滅する LED が代わり，入力ボタンの位置も変わった。

2 課題 3-2-1

3 課題 3-2-2

4 課題 3-3

4.1 概要

複数の LED を同時に操作して，スイッチによる制御も行う。

4.2 実験方法

LED を操作するオブジェクトの配列を作成し，count の増減で LED の明滅順を制御する。スイッチ入力のタイミングを増やすために，スリープを分割している。

ソースコード 3　program3-3

```
1  console.log("Start");
2  var Gpio = require("onoff").Gpio;
3
4  var led1 = new Gpio(4, "out");
5  var led2 = new Gpio(17, "out");
6  var led3 = new Gpio(27, "out");
7  var led4 = new Gpio(22, "out");
8  var leds = [led1, led2, led3, led4]
9
10 var sw1 = new Gpio(20, "in");
11 var sw2 = new Gpio(16, "in");
12 // 下段 LED1: 4, 17, 27, 22
13 // 上段 LED1: 5, 6, 13, 19
14 // ボタン 1: 20, ボタン 2: 16, ボタン 3: 12
15 var count = 0;
16 var upCount = 1;
17 while (true) {
18   leds[count].writeSync(1);
19
20   for (i = 0; i < 5; i++) {
21     // スイッチ振り分け
22     if (sw1.readSync() != 0) {
23       console.log("stopped!!");
24       while (sw1.readSync() == 0) {
25         sleep(100);
26       }
27       console.log("restart!!");
28
29     } else if (sw2.readSync() != 0) {
```

```
30          console.log("reversed!!");
31          upCount = (upCount == 1) ? 3 : 1;
32        }
33        sleep(100);
34      }
35
36      leds[count].writeSync(0);
37
38      // upCount の値によって昇降切り替わる
39      count = (count + upCount) % 4;
40      console.log(count);
41  }
```

4.3 実験結果

明滅状態が並び順に推移されるようになった。SW1 を押したときには，推移が止まり入力待ち状態になった。SW2 を押したときには，推移の方向が逆順になった。

4.4 考察

スイッチ入力を 1 つでまとめると，タイミングが取れなくなるためスリープを分割して複数回入力を待つようにした。これによって若干入力感度が向上した。

5 課題 3-4

5.1 概要

5.2 実験方法

ソースコード 4　program3-4

5.3 実験結果

5.4 考察

6 課題 3-5