

電子サイコロV2(GLCD版)

概要

LEDを7個使用した、1個型の電子サイコロを以前に製作したことがありますが、今回は、GLCDを使用し、2個型のサイコロを製作してみました。

動作原理

GLCDの制御(使い方)についてはGLCDライブラリを参照してください。

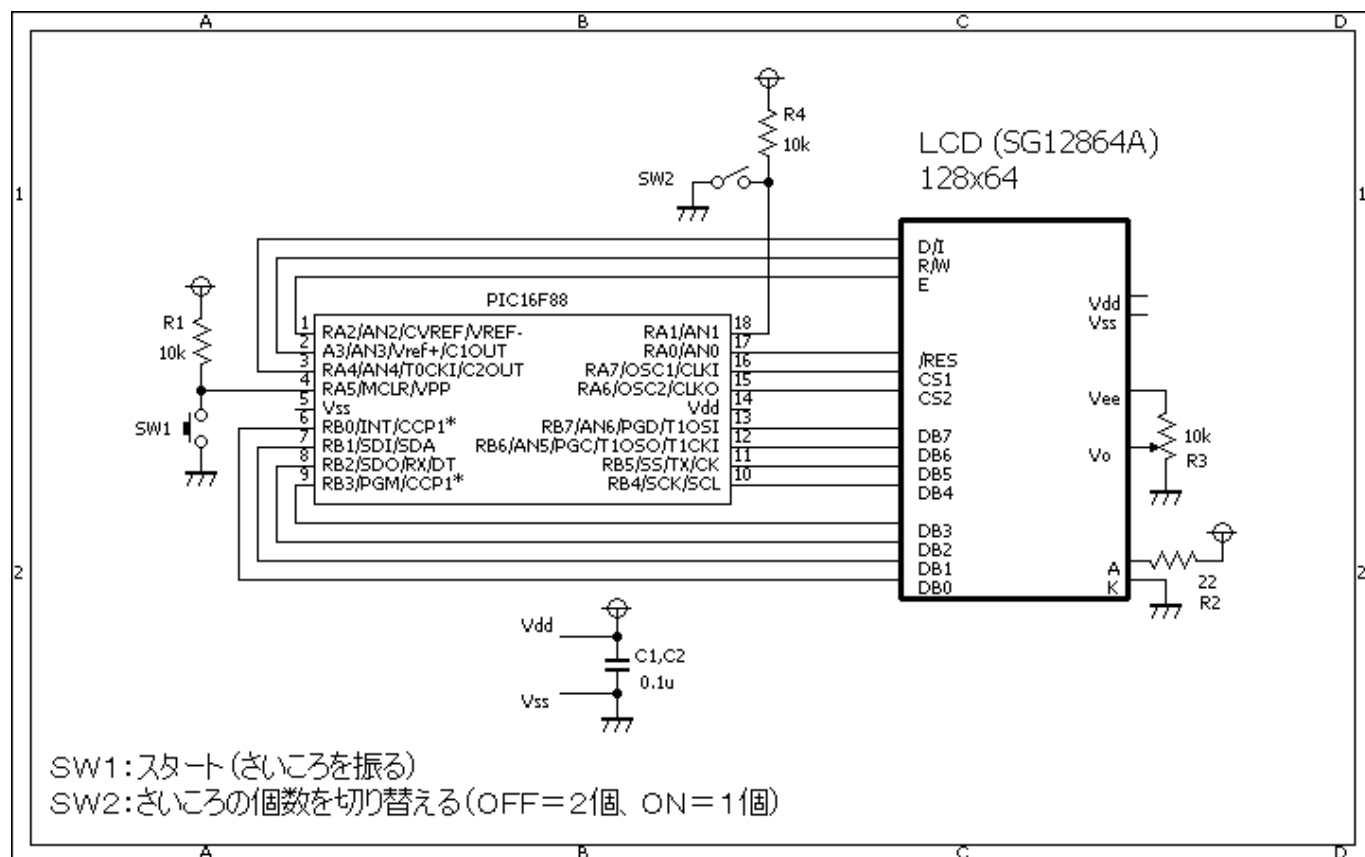
<処理の流れ>

1. 入出力ポートの設定およびGLCDの初期化(フォントの登録含む)を行います。
2. SW1を押下して乱数のシード(種)を設定します。(起動時1回のみ)
シードを設定しないといつも同じ系列で乱数が発生してしまいます。
これを防ぐためにSW1の押下待ちの間、シード値をインクリメントします。
そして押下時に、srand関数でシード値を設定します。
3. SW1が押下されるとrand関数で乱数を発生させ、サイコロの目の数を決定します。
※SW2がOFFの時には、サイコロ2個、ON時には、サイコロ1個とします。
※rand関数は、0-32767の数値を戻すので、この戻り値を5462で割り、0-5の値を得ます。
4. サイコロをGLCDに表示します。
※Glcd_Circle関数で、サイコロの個々の目を描画します。
※Glcd_Rectangle関数で、サイコロの個々の縁を描画します。
5. 3.に戻ります。

少し趣向を凝らして、サイコロの周りや最上位の段を、線(-)や四角()のマークが流れるようにしました。

回路図

GLCDライブラリで使用した回路にSW2を追加しただけです。



ソースコード

[electronic_dice.c](http://elechobby.jp/electronic_dice.c)

```
//*****
*
/*
< 電子サイコロ >
*/
//*****
*

#define SW1 PORTA.F5
#define SW2 PORTA.F1

/* System 5x8 (char #32 to #128) */
const unsigned short MyFontSystem5x8[] = {
    0x00,0x00,0x00,0x00,0x00, /* Espace 0x20 */
    0x00,0x00,0x4f,0x00,0x00, /* ! */
    0x00,0x07,0x00,0x07,0x00, /* " */
    0x14,0x7f,0x14,0x7f,0x14, /* # */
    0x24,0x2a,0x7f,0x2a,0x12, /* $ */
    0x23,0x13,0x08,0x64,0x62, /* % */
    0x36,0x49,0x55,0x22,0x20, /* & */
    0x00,0x05,0x03,0x00,0x00, /* ' */
    0x00,0x1c,0x22,0x41,0x00, /* ( */

```

```

0x00,0x41,0x22,0x1c,0x00, ///  

0x14,0x08,0x3e,0x08,0x14, ///  

0x08,0x08,0x3e,0x08,0x08, ///  

0x50,0x30,0x00,0x00,0x00, ///  

0x08,0x08,0x08,0x08,0x08, ///  

0x00,0x60,0x60,0x00,0x00, ///  

0x20,0x10,0x08,0x04,0x02, ///  

0x3e,0x51,0x49,0x45,0x3e, ///  

0x00,0x42,0x7f,0x40,0x00, ///  

0x42,0x61,0x51,0x49,0x46, ///  

0x21,0x41,0x45,0x4b,0x31, ///  

0x18,0x14,0x12,0x7f,0x10, ///  

0x27,0x45,0x45,0x45,0x39, ///  

0x3c,0x4a,0x49,0x49,0x30, ///  

0x01,0x71,0x09,0x05,0x03, ///  

0x36,0x49,0x49,0x49,0x36, ///  

0x06,0x49,0x49,0x29,0x1e, ///  

0x00,0x36,0x36,0x00,0x00, ///  

0x00,0x56,0x36,0x00,0x00, ///  

0x08,0x14,0x22,0x41,0x00, ///  

0x14,0x14,0x14,0x14,0x14, ///  

0x00,0x41,0x22,0x14,0x08, ///  

0x02,0x01,0x51,0x09,0x06, ///  

0x3e,0x41,0x5d,0x55,0x1e, ///  

0x7e,0x11,0x11,0x11,0x7e, ///  

0x7f,0x49,0x49,0x49,0x36, ///  

0x3e,0x41,0x41,0x41,0x22, ///  

0x7f,0x41,0x41,0x22,0x1c, ///  

0x7f,0x49,0x49,0x49,0x41, ///  

0x7f,0x09,0x09,0x09,0x01, ///  

0x3e,0x41,0x49,0x49,0x7a, ///  

0x7f,0x08,0x08,0x08,0x7f, ///  

0x00,0x41,0x7f,0x41,0x00, ///  

0x20,0x40,0x41,0x3f,0x01, ///  

0x7f,0x08,0x14,0x22,0x41, ///  

0x7f,0x40,0x40,0x40,0x40, ///  

0x7f,0x02,0x0c,0x02,0x7f, ///  

0x7f,0x04,0x08,0x10,0x7f, ///  

0x3e,0x41,0x41,0x41,0x3e, ///  

0x7f,0x09,0x09,0x09,0x06, ///  

0x3e,0x41,0x51,0x21,0x5e, ///  

0x7f,0x09,0x19,0x29,0x46, ///  

0x26,0x49,0x49,0x49,0x32, ///  

0x01,0x01,0x7f,0x01,0x01, ///  

0x3f,0x40,0x40,0x40,0x3f, ///  

0x1f,0x20,0x40,0x20,0x1f, ///  

0x3f,0x40,0x38,0x40,0x3f, ///  

0x63,0x14,0x08,0x14,0x63, ///  

0x07,0x08,0x70,0x08,0x07, ///  

0x61,0x51,0x49,0x45,0x43, ///  

0x00,0x7f,0x41,0x41,0x00, ///  


```

0x30 */

0x40 */

0x50 */

```

0x02,0x04,0x08,0x10,0x20, /// \ */
0x00,0x41,0x41,0x7f,0x00, /// ] */
0x04,0x02,0x01,0x02,0x04, /// ^ */
0x40,0x40,0x40,0x40,0x40, /// _ */
0x00,0x00,0x03,0x05,0x00, /// ` 0x60 */
0x20,0x54,0x54,0x54,0x78, /// a */
0x7F,0x44,0x44,0x44,0x38, /// b */
0x38,0x44,0x44,0x44,0x44, /// c */
0x38,0x44,0x44,0x44,0x7f, /// d */
0x38,0x54,0x54,0x54,0x18, /// e */
0x04,0x04,0x7e,0x05,0x05, /// f */
0x08,0x54,0x54,0x54,0x3c, /// g */
0x7f,0x08,0x04,0x04,0x78, /// h */
0x00,0x44,0x7d,0x40,0x00, /// i */
0x20,0x40,0x44,0x3d,0x00, /// j */
0x7f,0x10,0x28,0x44,0x00, /// k */
0x00,0x41,0x7f,0x40,0x00, /// l */
0x7c,0x04,0x7c,0x04,0x78, /// m */
0x7c,0x08,0x04,0x04,0x78, /// n */
0x38,0x44,0x44,0x44,0x38, /// o */
0x7c,0x14,0x14,0x14,0x08, /// p 0x70 */
0x08,0x14,0x14,0x14,0x7c, /// q */
0x7c,0x08,0x04,0x04,0x00, /// r */
0x48,0x54,0x54,0x54,0x24, /// s */
0x04,0x04,0x3f,0x44,0x44, /// t */
0x3c,0x40,0x40,0x20,0x7c, /// u */
0x1c,0x20,0x40,0x20,0x1c, /// v */
0x3c,0x40,0x30,0x40,0x3c, /// w */
0x44,0x28,0x10,0x28,0x44, /// x */
0x0c,0x50,0x50,0x50,0x3c, /// y */
0x44,0x64,0x54,0x4c,0x44, /// z */
0x08,0x36,0x41,0x41,0x00, /// { */
0x00,0x00,0x77,0x00,0x00, /// | */
0x00,0x41,0x41,0x36,0x08, /// } */
0x08,0x08,0x2a,0x1c,0x08, /// <- */
0x08,0x1c,0x2a,0x08,0x08, /// -> */
0xff,0xff,0xff,0xff,0xff, /// □ 0x80 */
0x06,0x09,0x09,0x06,0x00 /// // oC 0x81
};

```

```

//*****
*

```

```

void Switch0Ncheck()
{
    while (Button(&PORTA, 5, 1, 0) == 0)
        ;
    while (Button(&PORTA, 5, 1, 1) == 0)
        ;
}

```

```
//*****
*

void  Glcd_Circle_Ex(short x, short y)
{
    static    short    cnt;
    //
    for (cnt = 0; cnt < 7; cnt++) {
        Glcd_Circle(x, y, cnt, 1);
    }
}

//*****
*

void  dice_proc(short num)
{
    static    int        r;
    //
    num = num * 55;
    Glcd_Rectangle(10 + num, 10, 60 + num, 60, 1);
    Glcd_Rectangle(10 + num + 2, 10 + 2, 60 + num - 2, 60 - 2, 1);
    Glcd_Rectangle(10 + num, 10, 60 + num, 60, 1);
    Glcd_Rectangle(10 + num + 2, 10 + 2, 60 + num - 2, 60 - 2, 1);
    //
    r = rand();
    switch (r / 5462) {
    case 0:
        Glcd_Circle_Ex(35 + num, 35);
        break;
    case 1:
        Glcd_Circle_Ex(20 + num, 50);
        Glcd_Circle_Ex(50 + num, 20);
        break;
    case 2:
        Glcd_Circle_Ex(20 + num, 50);
        Glcd_Circle_Ex(35 + num, 35);
        Glcd_Circle_Ex(50 + num, 20);
        break;
    case 3:
        Glcd_Circle_Ex(20 + num, 50);
        Glcd_Circle_Ex(50 + num, 20);
        Glcd_Circle_Ex(20 + num, 20);
        Glcd_Circle_Ex(50 + num, 50);
        break;
    case 4:
        Glcd_Circle_Ex(20 + num, 50);
        Glcd_Circle_Ex(50 + num, 20);
        Glcd_Circle_Ex(20 + num, 20);
        Glcd_Circle_Ex(50 + num, 50);
        Glcd_Circle_Ex(35 + num, 35);
    }
```

```

        break;
    case 5:
        Glcd_Circle_Ex(20 + num, 50);
        Glcd_Circle_Ex(50 + num, 20);
        Glcd_Circle_Ex(20 + num, 20);
        Glcd_Circle_Ex(50 + num, 50);
        Glcd_Circle_Ex(20 + num, 35);
        Glcd_Circle_Ex(50 + num, 35);
        break;
    }
}

//*****
*

void main()
{
    static unsigned int seed;
    static unsigned short x, y, tmp, lineMode, boxMode,
boxCnt;
    //
    OSCCON = 0b01110000; // クロックを8Mhzに設定する。
    ANSEL = 0b00000000; // □□□変換は使用しない。
    //ポートの設定
    TRISA = 0b00100010;
    TRISB = 0b00000000;
    PORTA = 0b00000000;
    PORTB = 0b00000000;
    //□□□□の初期化&フォントデータの設定&画面クリア
    Delay_ms(100);
    Glcd_Init(&PORTA, 7, 6, 4, 3, 0, 2, &PORTB); //
cs1, cs2, rs, rw, rst, en
    Delay_ms(100);
    Glcd_Set_Font(MyFontSystem5x8, 5, 8, 32);
    Glcd_Fill(0);
    Glcd_Write_Text("Electronic Dice", 0, 0, 1);
    //乱数の種を設定します。
    boxMode = 0;
    boxCnt = 0;
    while (SW1 == 1) {
        seed++;
        //最下位の段を■が流れるように点滅させます。
        Glcd_Box(boxCnt * 4, 60, boxCnt * 4 + 3, 63, 2);
        switch (boxMode) {
            case 0:
                if (boxCnt < 32) {
                    boxCnt++;
                } else {
                    boxMode = 1;
                }
            break;
        }
    }
}

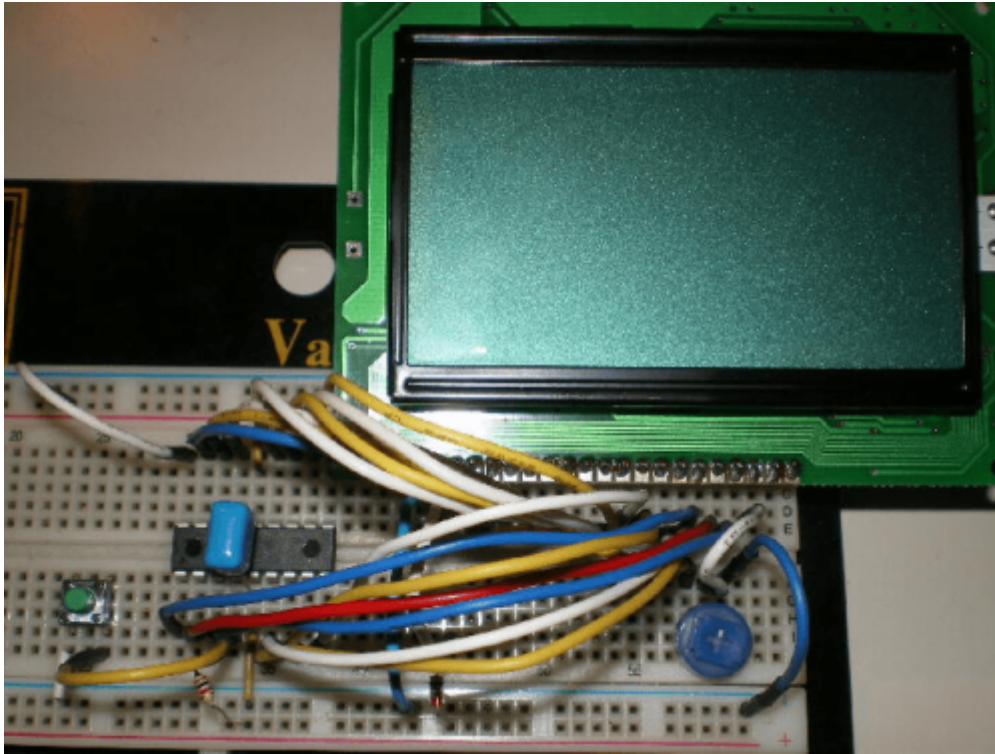
```

```
        case 1:
            if (boxCnt > 0) {
                boxCnt--;
            } else {
                boxMode = 0;
            }
            break;
    }
    Delay_ms(10);
}
srand(seed);
//
lineMode = 0;
x = 8;
y = 8;
boxMode = 0;
boxCnt = 0;
//
while (1) {
    //サイコロを振ります。
    while (SW1 == 1) {
        //最上位の段を■が流れるように点滅させます。
        Glcd_Box(boxCnt * 4, 0, boxCnt * 4 + 3, 3, 2);
        switch (boxMode) {
            case 0:
                if (boxCnt < 32) {
                    boxCnt++;
                } else {
                    boxMode = 1;
                }
                break;
            case 1:
                if (boxCnt > 0) {
                    boxCnt--;
                } else {
                    boxMode = 0;
                }
                break;
        }
    }
    //サイコロの周りを線が流れるように点灯させます。
    Glcd_Dot(x, y, 2);
    tmp = (SW2 == 1)? 117 : 62;
    switch (lineMode) {
        case 0:
            if (y < 62) {
                y++;
            } else {
                lineMode = 1;
            }
            break;
        case 1:
            if (x < tmp) {
                x++;
            } else {
                lineMode = 0;
            }
            break;
    }
}
```

```
        if (x < tmp) {
            x++;
        } else {
            lineMode = 2;
        }
        break;
    case 2:
        if (y > 8) {
            y--;
        } else {
            lineMode = 3;
        }
        break;
    case 3:
        if (x > 8) {
            x--;
        } else {
            lineMode = 0;
        }
        break;
    }
    Delay_ms(10);
}
SwitchONcheck();
Glcd_Fill(0);
//サイコロを表示します。
if (SW2 == 1) {
    dice_proc(0);
    dice_proc(1);
} else {
    dice_proc(0);
}
}
}

//*****
*
```

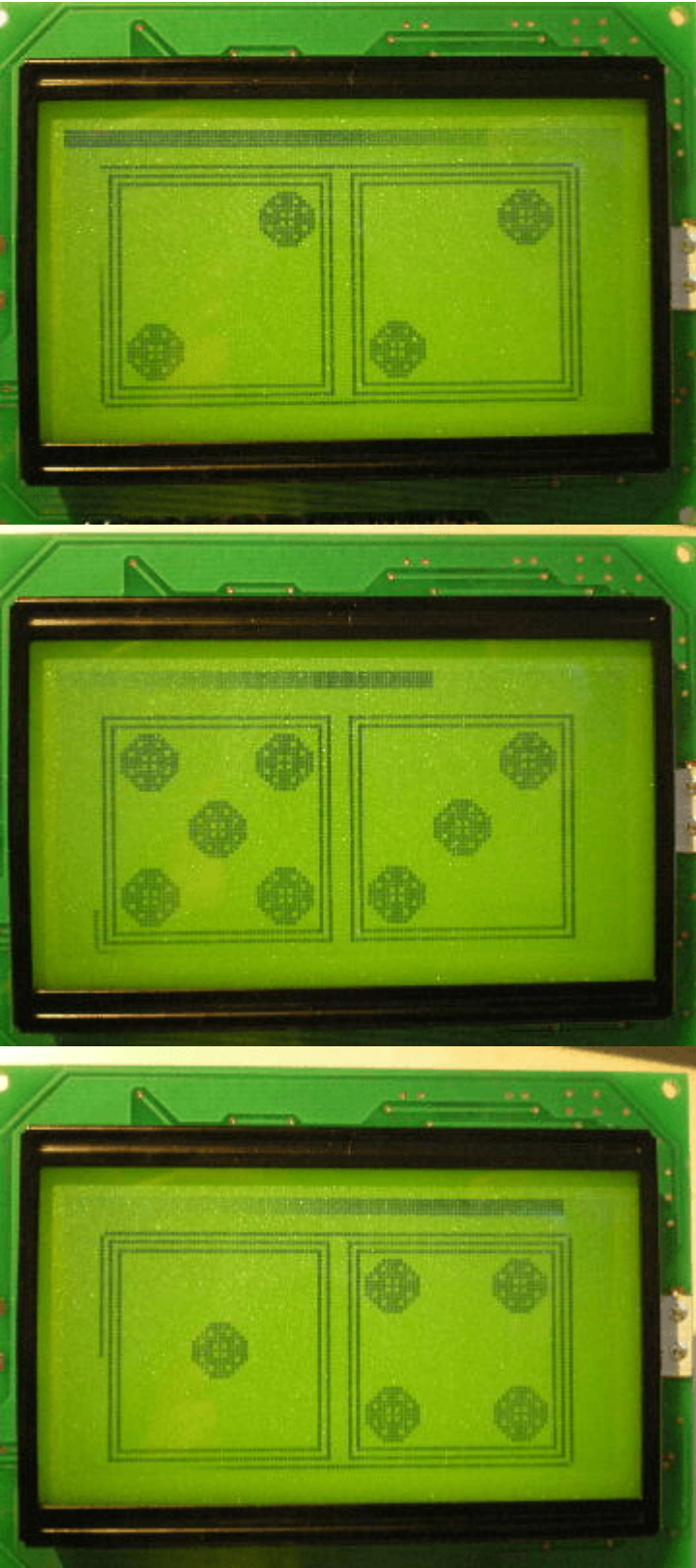
動作確認

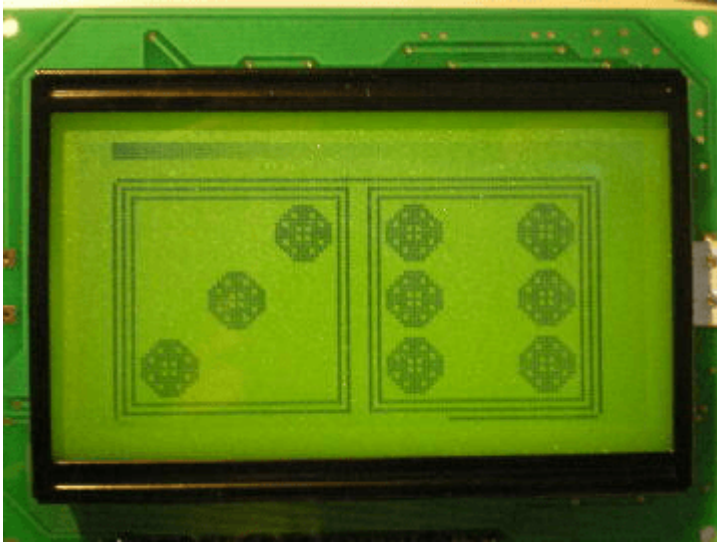


この時点で、SW1を押下すると、乱数の種が設定されます。

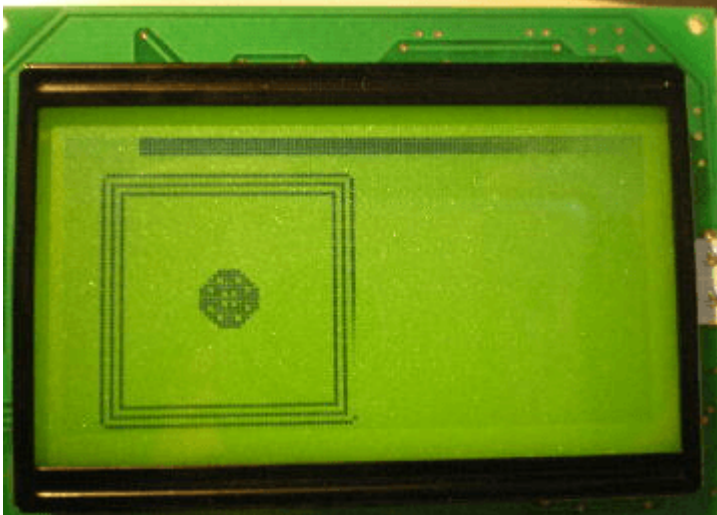
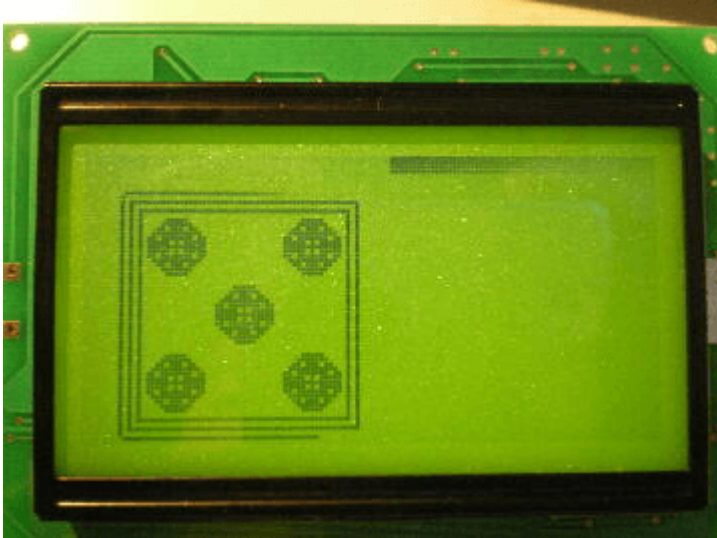


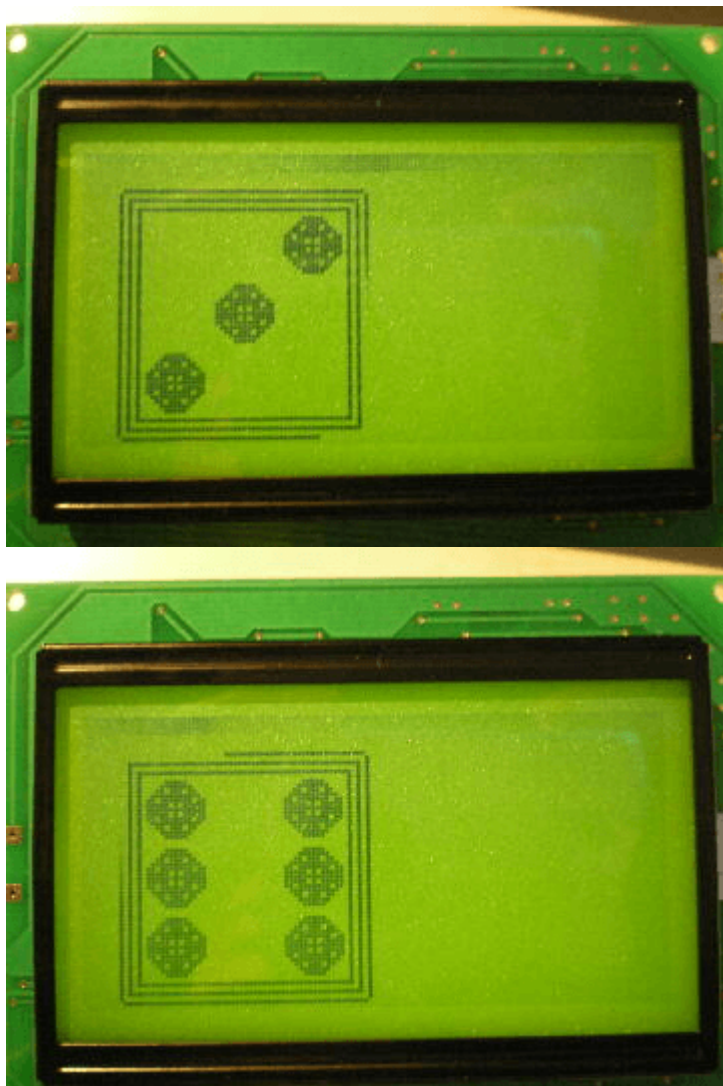
SW2がOFF時には、サイコロを2個表示します□ SW1を押下する毎に、サイコロの目の数が変わります。





SW2がON時には、サイコロを1個表示します□ SW1を押下する毎に、サイコロの目の数が変わります。





如何ですか? これをゲームなどに使えば、楽しくなりますね! 😊

著作権表示 copyright notice

このページは稲崎様の閉鎖したHPのコピーで、著作権は稲崎様にあります。詳細 This page is a copy of Mr. Inasaki's closed website, and the copyright is held by him. [Details](#)

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:119>

Last update: **2020/05/01 19:08**

