

7セグLED(3桁)表示ユニット(I2C対応)

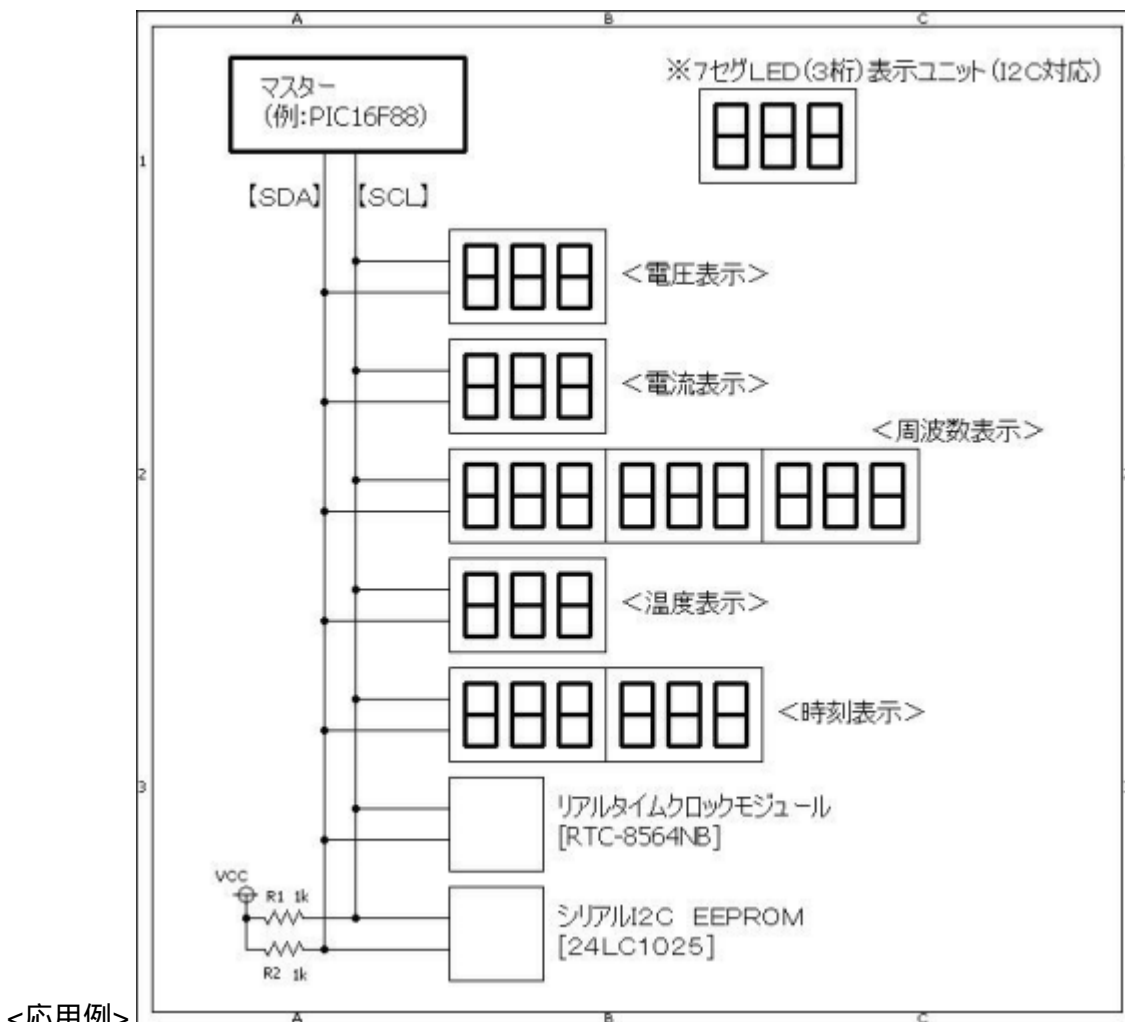
概要

7セグLEDは、LCDに比べて、文字が大きく、輝度が高いので、用途によっては、とても重宝するデバイスです。しかし、7セグLEDを制御するには、信号線を多く(1桁で9ピン)必要とするので、ピン数の少ないPICには不向きです。

そこで、2本だけの信号線で複数接続可能な、I2C対応(スレーブ)の「7セグLED表示器」として、ユニット化すれば使いやすいのではと考えました。

<仕様>

- I2Cで制御可能とする。
- 表示桁数は、3桁とする。
- 接続台数は、8台まで可能とする。

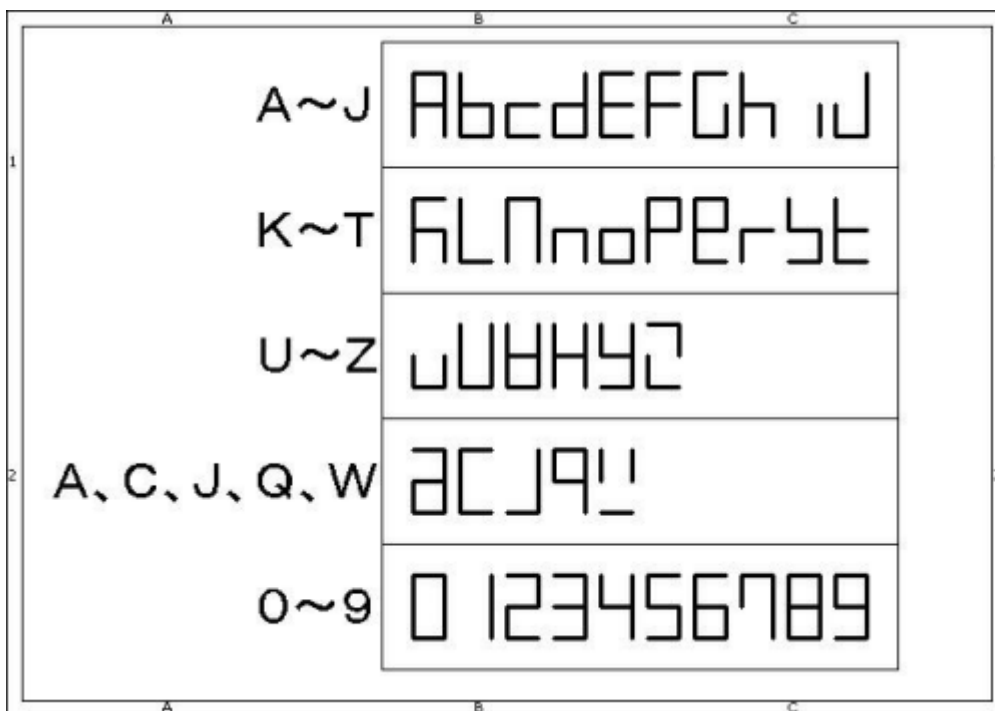
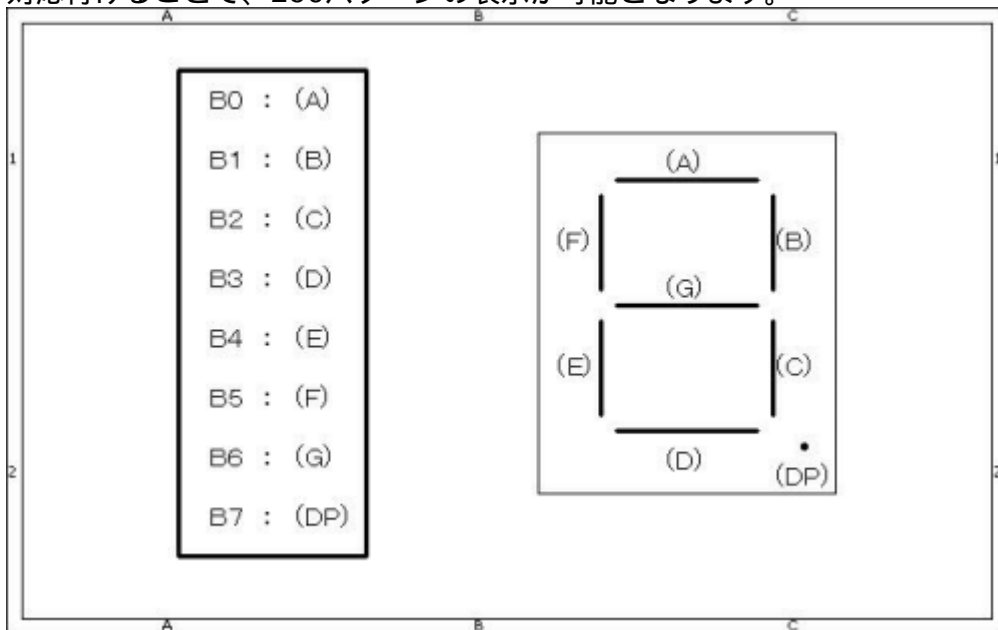


動作原理

I2Cのスレーブ機能としての、基本的な構造は、前回製作した「LCDモニタ」(I2C対応)と同じです。<メモ

リ構造> 7セグLED(3桁)のデータを設定するために、3バイトのメモリを使用します。0x00(7セグ1用(向かって、左側)) 0x01(7セグ2用(向かって、中央)) 0x02(7セグ3用(向かって、右側))

<バイトデータの各ビットと7セグの各エレメントとの対応表> このようにビットとエレメントを1対1に対応付けることで、256パターンの表示が可能となります。



<表示例>

<メイン処理の流れ>

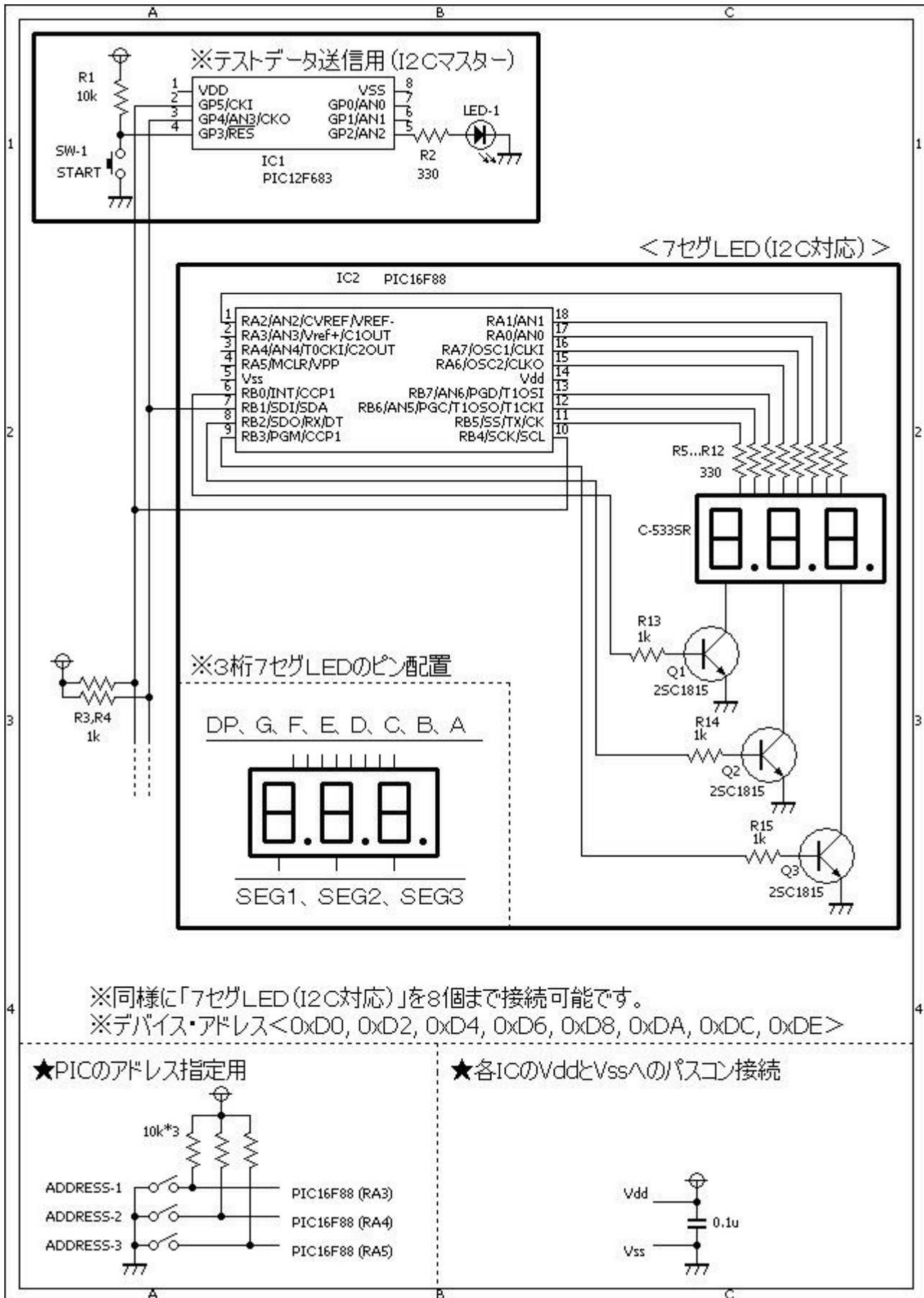
1. 0x00のメモリのデータを7セグLED1へ1msecの間だけ表示する。
2. 0x01のメモリのデータを7セグLED2へ1msecの間だけ表示する。
3. 0x02のメモリのデータを7セグLED3へ1msecの間だけ表示する。
4. 1.へ戻る。

<マスター側から、スレーブ側への、7セグLEDデータ出力> マスター側から、アドレス(0x00~0x02)のメモリに、出力データを書き込みます。

```
Soft_I2C_Start();
```

```
Soft_I2C_Write(0xD0);  
Soft_I2C_Write(0x00);  
Soft_I2C_Write(0b00111111); // 7セグ1(データは、 “ 0”)  
Soft_I2C_Write(0b00000110); // 7セグ2(データは、 “ 1”)  
Soft_I2C_Write(0b01011011); // 7セグ3(データは、 “ 2”)  
Soft_I2C_Stop();
```

回路図



ソースコード

7segLed_i2c.c

```
//*****  
*  
/*  
  < 7セグ□□□□□□対応 ) >  
*/  
//*****  
*  
  
#define      SW_ADDR1      PORTA.F3  
#define      SW_ADDR2      PORTA.F4  
#define      SW_ADDR3      PORTA.F5  
  
#define      ADDR_BASE     0xD0  
  
#define      SEG1          PORTB.F0  
#define      SEG2          PORTB.F2  
#define      SEG3          PORTB.F3  
  
#define      ON            1  
#define      OFF           0  
  
#define      DATA_SIZE    3  
  
//*****  
*  
  
static unsigned   short   data[DATA_SIZE], pnt, flg, tmp, stat;  
  
//*****  
*  
  
void  i2c_Write(unsigned short dat)  
{  
    while (SSPSTAT.BF == 1)  
        ;  
    while (1) {  
        SSPCON.WCOL = 0;  
        SSPBUF = dat;  
        if (SSPCON.WCOL == 1)  
            continue;  
        //  
        SSPCON.CKP = 1;  
        return;  
    }  
}  
  
//*****  
*
```

```
void i2c_Handler()
{
    stat = SSPSTAT;
    tmp = SSPSTAT & 0b00101101;
    //
    if (tmp == 0b00001001) { //書き込みモード、デバイスアドレス
        tmp = SSPBUF;
        flg = 0;
        pnt = 0;
        return;
    }
    if (tmp == 0b00101001) { //書き込みモード、データ
        if (flg == 0) {
            pnt = SSPBUF;
            flg = 1;
            return;
        }
        if (flg == 1) {
            data[pnt] = SSPBUF;
            pnt++;
            return;
        }
    }
    if (tmp == 0b00001100) { //読み込みモード、デバイスアドレス
        i2c_Write(data[pnt]);
        pnt++;
        return;
    }
    if (tmp == 0b00101100) { //読み込みモード、データ□□□□□
        i2c_Write(data[pnt]);
        pnt++;
        return;
    }
    if (tmp == 0b00101000) { //読み込みモード、データ□□□□□□□□
        tmp = SSPBUF;
        SSPCON = 0x36;
        return;
    }
}

//*****
*

void interrupt()
{
    if (PIR1.SSPIF == 1) {
        PIR1.SSPIF = 0;
        //
        i2c_Handler();
    }
}
```

```
//*****  
*  
void ByteToBit(unsigned short number, char *output)  
{  
    short cnt;  
    //  
    for (cnt = 0; cnt < 8; cnt++) {  
        if ((number & 0x80) != 0)  
            output[cnt] = '1';  
        else  
            output[cnt] = '0';  
        number <<= 1;  
    }  
    output[8] = 0x00;  
}  
  
//*****  
*  
void segDataSet(unsigned short dat)  
{  
    PORTA.F2 = (dat.F0 == 1) ? 1 : 0; // A  
    PORTA.F1 = (dat.F1 == 1) ? 1 : 0; // B  
    PORTA.F0 = (dat.F2 == 1) ? 1 : 0; // C  
    PORTA.F7 = (dat.F3 == 1) ? 1 : 0; // D  
    PORTA.F6 = (dat.F4 == 1) ? 1 : 0; // E  
    PORTB.F7 = (dat.F5 == 1) ? 1 : 0; // F  
    PORTB.F6 = (dat.F6 == 1) ? 1 : 0; // G  
    PORTB.F5 = (dat.F7 == 1) ? 1 : 0; // DP  
}  
  
//*****  
*  
void segDataDisplay()  
{  
    segDataSet(data[0]);  
    SEG1 = ON;  
    Delay_ms(1);  
    SEG1 = OFF;  
    //  
    segDataSet(data[1]);  
    SEG2 = ON;  
    Delay_ms(1);  
    SEG2 = OFF;  
    //  
    segDataSet(data[2]);  
    SEG3 = ON;  
    Delay_ms(1);  
}
```

```
    SEG3 = OFF;
}

//*****
*

void main()
{
    unsigned short cnt, tmp;
    //
    CMCON = 0b00000111; //コンパレータは使用しない。
    ANSEL = 0b00000000; //A/Dコンバータは使用しない。
    OSCCON = 0b01110000; //クロックは内臓8MHzを使用する。
    TRISA = 0b00111000; //PORTAを設定する。
    TRISB = 0b00010010; //PORTBを設定する。
    OPTION_REG.NOT_RBPU = 0; //PORTBをプルアップする。
    //□□□を設定する。
    SSPSTAT.SMP = 1;
    SSPSTAT.CKE = 1;
    SSPCON.WCOL = 0;
    SSPCON.SSP0V = 0;
    SSPCON.SSPEN = 1;
    SSPCON.CKP = 1;
    SSPCON.SSPM0 = 0;
    SSPCON.SSPM1 = 1;
    SSPCON.SSPM2 = 1;
    SSPCON.SSPM3 = 0;
    SSPADD = ADDR_BASE + ((SW_ADDR3 == 1) ? 8 : 0) + ((SW_ADDR2 == 1) ?
4 : 0) + ((SW_ADDR1 == 1) ? 2 : 0);
    PIE1.SSPIE = 1;
    PIR1.SSPIF = 0;
    //
    pnt = 0;
    flg = 0;
    data[0] = 0b11111111;
    data[1] = 0b11111111;
    data[2] = 0b11111111;
    SEG1 = OFF;
    SEG2 = OFF;
    SEG3 = OFF;
    //
    tmp = 0b00000001;
    for (cnt = 0; cnt < 8; cnt++) {
        SEG1 = ON;
        segDataSet(tmp);
        Delay_ms(100);
        SEG1 = OFF;
        tmp = tmp << 1;
    }
    tmp = 0b00000001;
    for (cnt = 0; cnt < 8; cnt++) {
```

```

    SEG2 = ON;
    segDataSet(tmp);
    Delay_ms(100);
    SEG2 = OFF;
    tmp = tmp << 1;
}
tmp = 0b00000001;
for (cnt = 0; cnt < 8; cnt++) {
    SEG3 = ON;
    segDataSet(tmp);
    Delay_ms(100);
    SEG3 = OFF;
    tmp = tmp << 1;
}
// 割り込み(全体)の設定
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
while (1) {
    segDataDisplay();
}
}

//*****
*
```

<参考> テスト用(PIC12F683)のプログラムです。

7segLed_i2c_master.c

```

//*****
*
/*
『7セグLEDのテストデータ送信用(マスター)』
*/
//*****
*

#define SW GPIO.F3
#define LED GPIO.F2

#define ON 1
#define OFF 0

#define ACK 1
#define NO_ACK 0

#define SEG_DATA_0 0b00111111
#define SEG_DATA_1 0b00000110
#define SEG_DATA_2 0b01011011
```

```
#define SEG_DATA_3      0b01001111
#define SEG_DATA_4      0b01100110
#define SEG_DATA_5      0b01101101
#define SEG_DATA_6      0b01111101
#define SEG_DATA_7      0b00100111
#define SEG_DATA_8      0b01111111
#define SEG_DATA_9      0b01101111
#define SEG_DATA_DP     0b10000000
#define SEG_DATA_ALL    0b11111111

//*****
*

void SwitchONcheck()
{
    while (Button(&GPIO, 3, 1, 0) == 0)
        ;
    while (Button(&GPIO, 3, 1, 1) == 0)
        ;
}

//*****
*

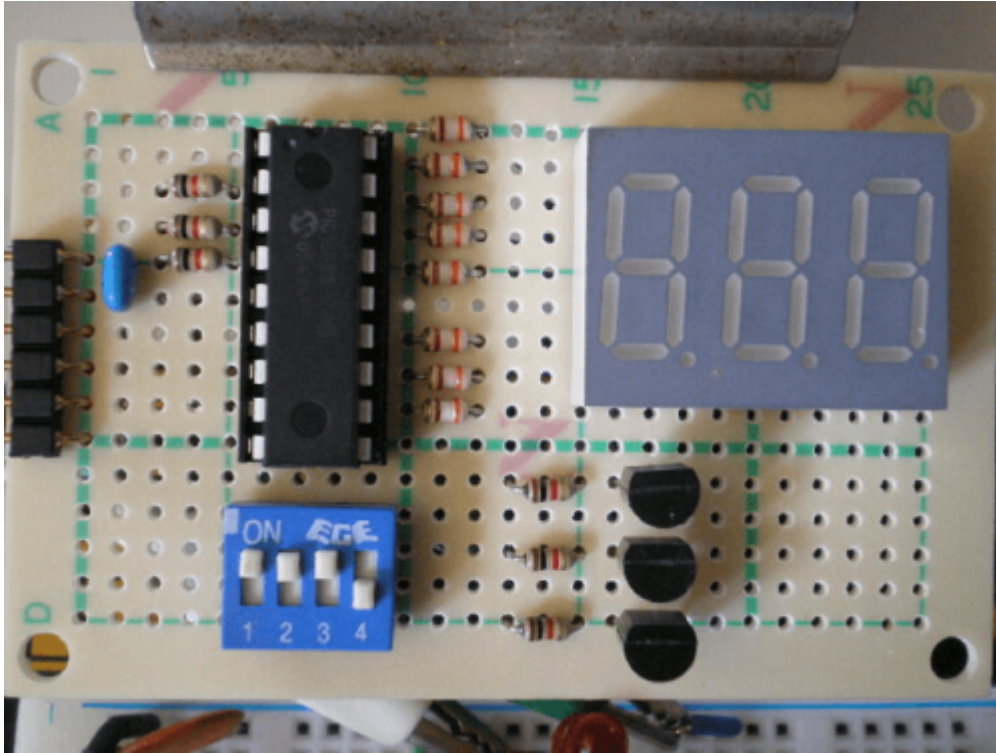
void main()
{
    unsigned short cnt, dat;
    //
    CMCON0 = 0b00000111;
    ANSEL.ANS0 = 0;
    ANSEL.ANS1 = 0;
    ANSEL.ANS2 = 0;
    ANSEL.ANS3 = 0;
    ADCON0.VCFG = 0;
    TRISIO = 0b00001011;
    OSCCON = 0b01110000;
    //
    for (cnt = 0; cnt < 10; cnt++) {
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
    //
    Soft_I2C_Config(&GPIO, 4, 5); // SDA, SCL
    //
    while (1) {
        SwitchONcheck();
        //
        Soft_I2C_Start();
        Soft_I2C_Write(0xD0);
    }
}
```

```
Soft_I2C_Write(0x00);
Soft_I2C_Write(SEG_DATA_0);
Soft_I2C_Write(SEG_DATA_1);
Soft_I2C_Write(SEG_DATA_2);
Soft_I2C_Stop();
//
SwitchONcheck();
//
Soft_I2C_Start();
Soft_I2C_Write(0xD0);
Soft_I2C_Write(0x00);
Soft_I2C_Write(SEG_DATA_3);
Soft_I2C_Write(SEG_DATA_4);
Soft_I2C_Write(SEG_DATA_5);
Soft_I2C_Stop();
//
SwitchONcheck();
//
Soft_I2C_Start();
Soft_I2C_Write(0xD0);
Soft_I2C_Write(0x00);
Soft_I2C_Write(SEG_DATA_6);
Soft_I2C_Write(SEG_DATA_7);
Soft_I2C_Write(SEG_DATA_8);
Soft_I2C_Stop();
//
SwitchONcheck();
//
Soft_I2C_Start();
Soft_I2C_Write(0xD0);
Soft_I2C_Write(0x00);
Soft_I2C_Write(SEG_DATA_9);
Soft_I2C_Write(SEG_DATA_DP);
Soft_I2C_Write(SEG_DATA_ALL);
Soft_I2C_Stop();
//
for (cnt = 0; cnt < 10; cnt++) {
    LED = ON;
    Delay_ms(50);
    LED = OFF;
    Delay_ms(50);
}
}
}

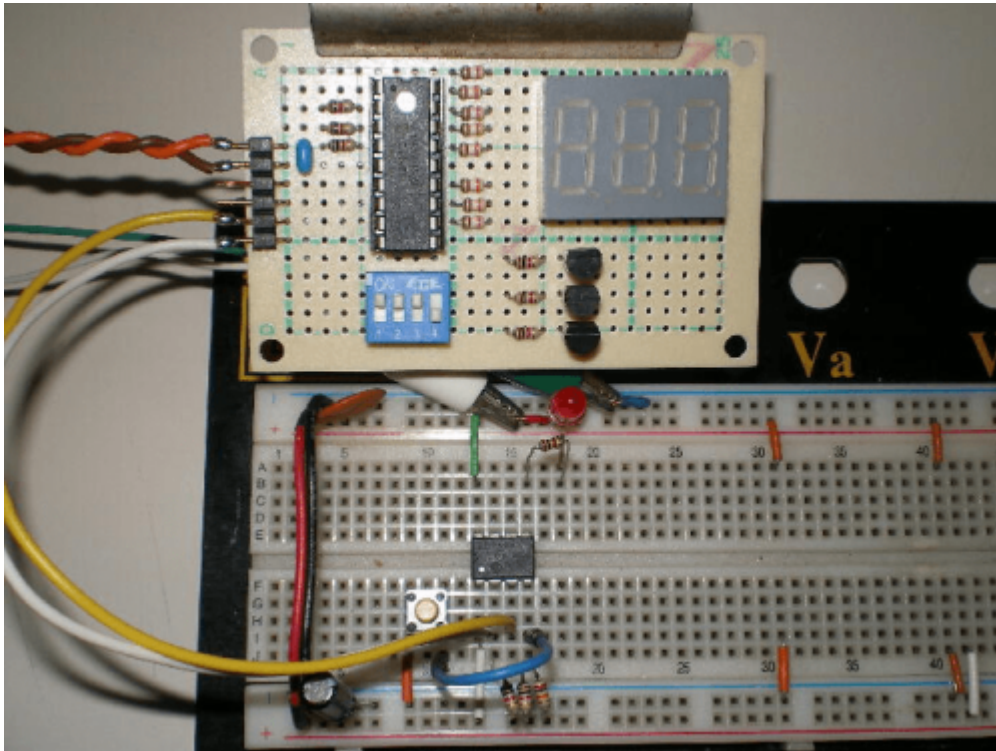
//*****
*
```

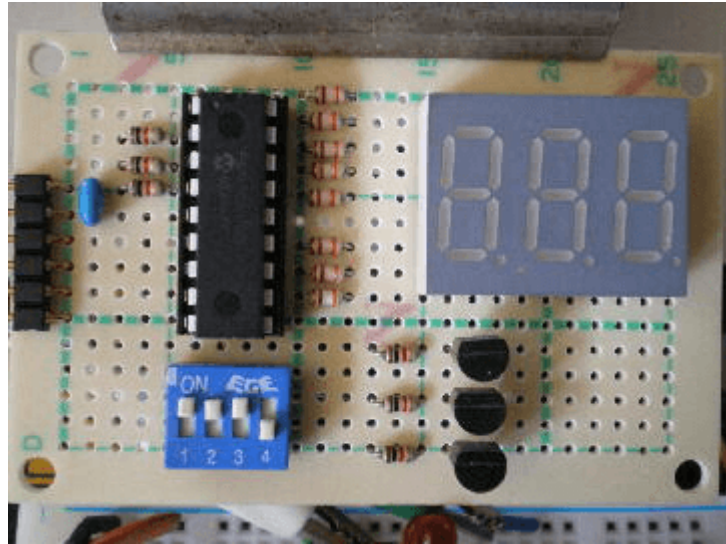
動作確認

久しぶりに、蛇の目基板に組み立てて見ました。(ディップスイッチの4は未使用です) 工夫すれば、もう少しコンパクトに仕上がると思います。

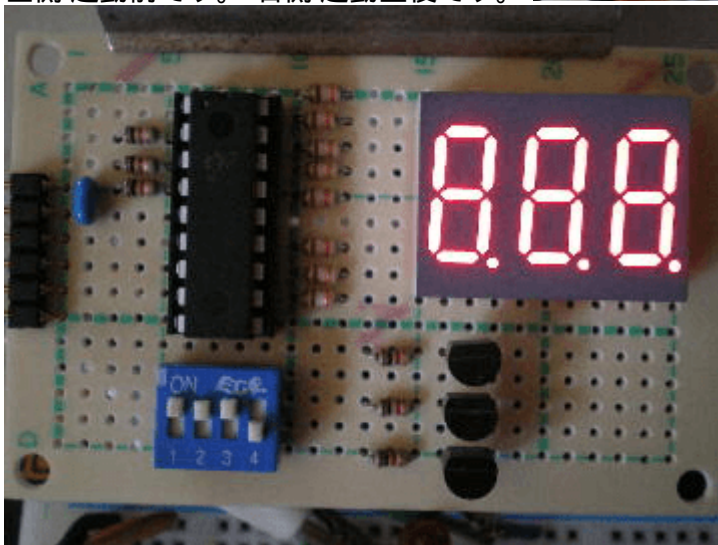


本ユニットへのテストデータ送信用のPIC12F683も含めた全体像です。<黄色線=SDA□白色線=SCL>

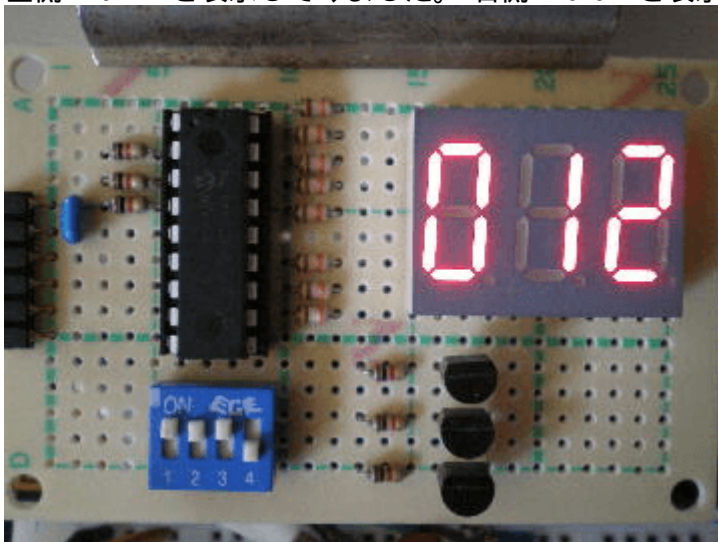


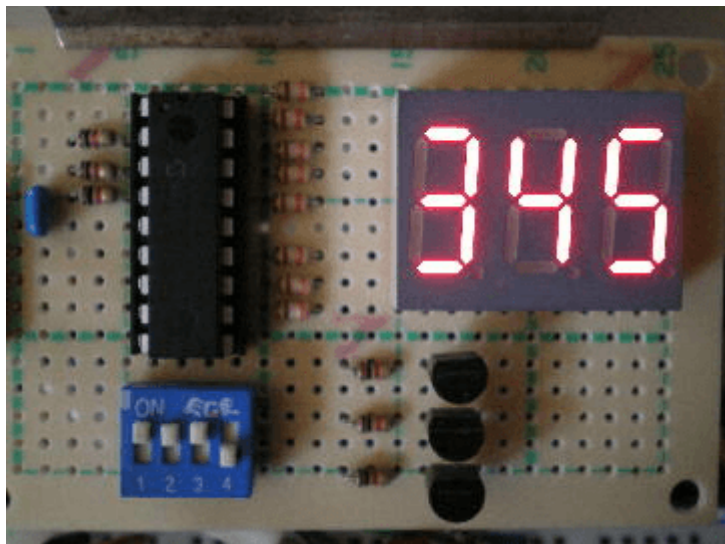


左側:起動前です。 右側:起動直後です。

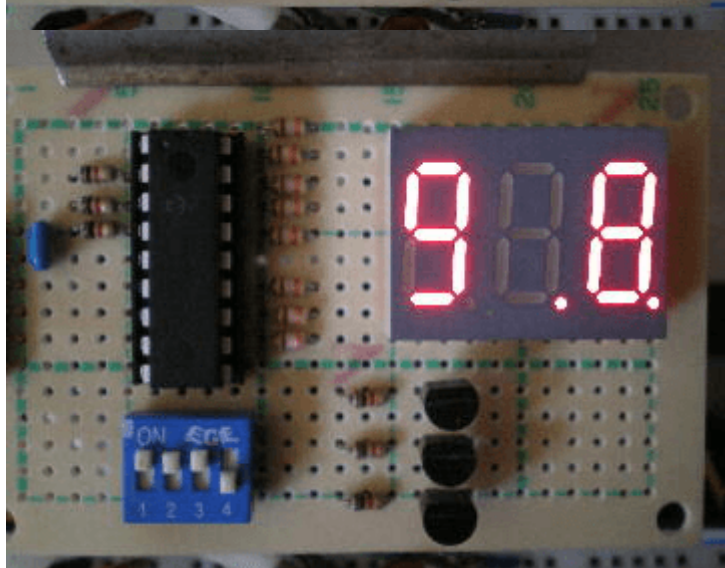
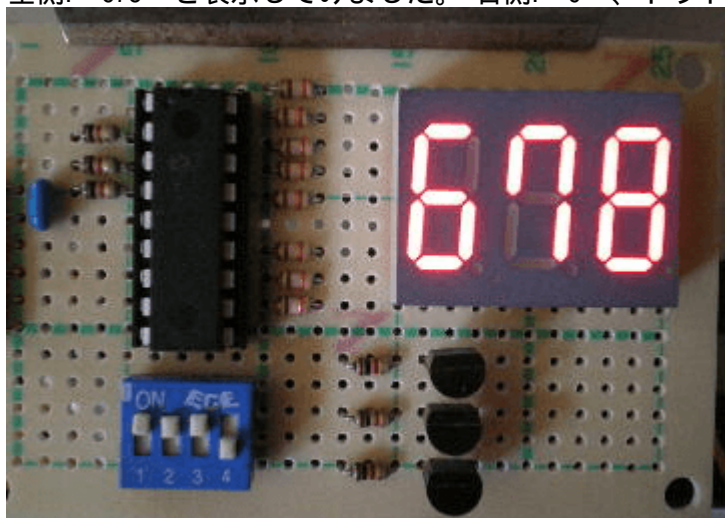


左側: “ 012 “ を表示してみました。 右側: “ 345 ” を表示してみました。





左側: “ 678 “ を表示してみました。 右側: “ 9 ”、ドット、全部を表示してみました。



如何ですか? これでピン数の少ないPICに、7セグLEDを複数接続することができますね



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://www.deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://www.deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:96>

Last update: **2025/10/17 14:29**

