簡易電源タイマー(切り忘れ防止)

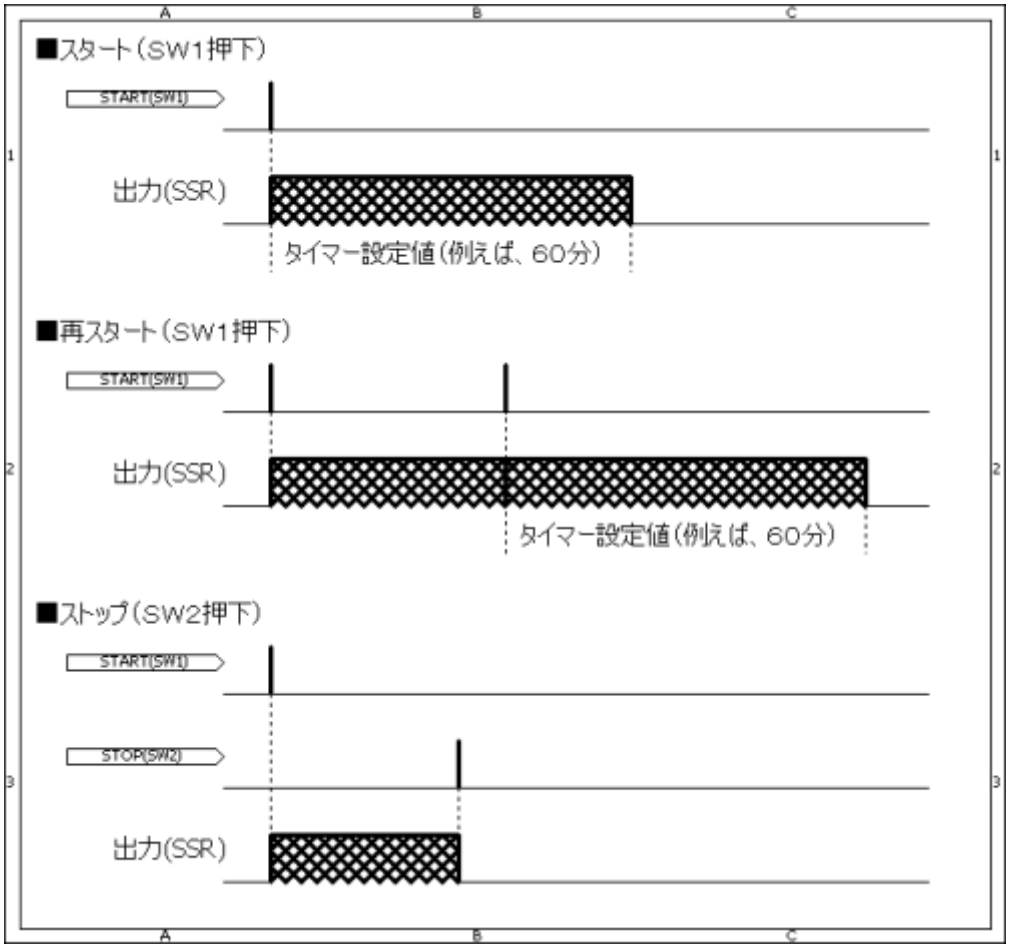
概要

半田ごてを使用していて、うっかり電源を切り忘れてしまい、翌日にヒヤッとしたときが、過去に幾度かあります。 そこで今回は、電源の切り忘れを防止するための“電源タイマー“を製作しみました。

<仕様>

- タイマーの設定値は、1分~999分まで設定出来ます(EEPROMに記憶)
- 商用電源(100V)をSSR(ソリッドステートリレー)を使用してON/OFFさせます。
- スタートスイッチを押下すると、設定した時間だけSSRをONします。
- 動作中に、スタートスイッチを押下すると、再スタートします。
- ストップスイッチを押下するとSSRをOFFします。
- タイムアウト直前の1分間は、1秒周期でブザーを鳴らせます。

<タイミングチャート>



動作原理(ハードウェア)

◎タイマー値の表示 3桁の7セグLED(カソードコモン)を簡易接続方式で使用します。 ※通常はトランジスタなどを利用して制御します)

◎商用電源(100V)のON/OFF 秋月電子で販売している、「ソリッド・ステート・リレー(SSR)キット」を使用します。

動作原理（ソフトウェア）

◎メイン処理(タイマーモード)

- スタートスイッチ(SW1)が、押下されるとSSRをONにし、カウント変数(count_msec)に、設定値(set_up_count)をセットします。ドットフラグ(dot_flg)をONにします。
- ストップスイッチ(SW2)が、押下されるとSSRをOFFにします。ドットフラグ(dot_flg)をOFFにします。
- カウント変数(count_msec)が、“ 0 ”になるとSSRをOFFにします。ドットフラグ(dot_flg)をOFFにします。
- タイムアウト直前の1分間は、1秒周期でブザーを鳴らせます。

◎メイン処理(設定値変更モード)

- “ 分 “ インクリメントスイッチ(SW1)が、押下されると設定値(set_up_count)を“ 1分 ”インクリメントします。
- “ 分 “ デクリメントスイッチ(SW2)が、押下されると設定値(set_up_count)を“ 1分 ”デクリメントします。

◎メイン処理(その他)

- カウント変数(count_msec)から、“ 分 “ を求め、表示データ(seg_data)に設定します。

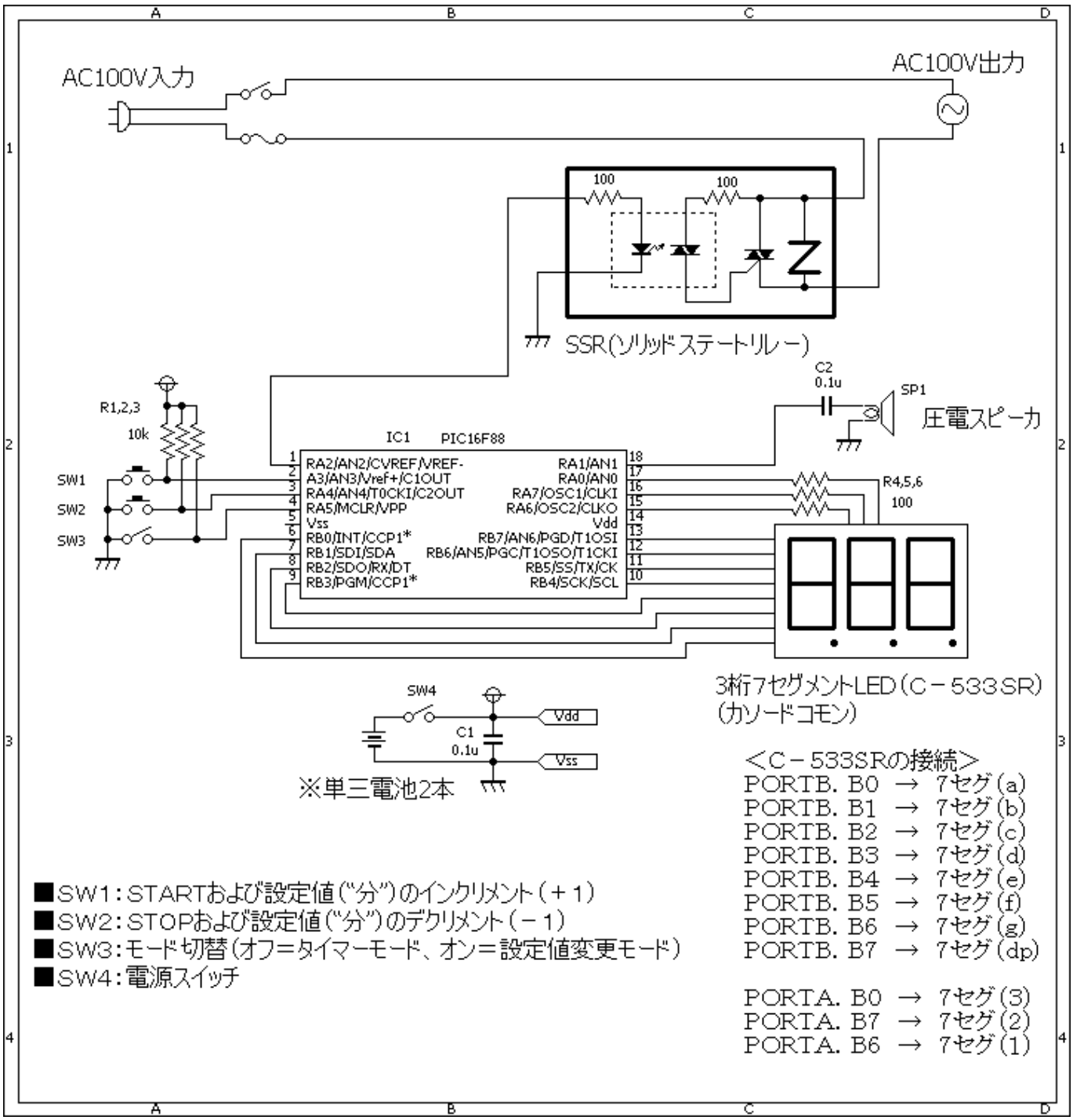
◎割り込み処理

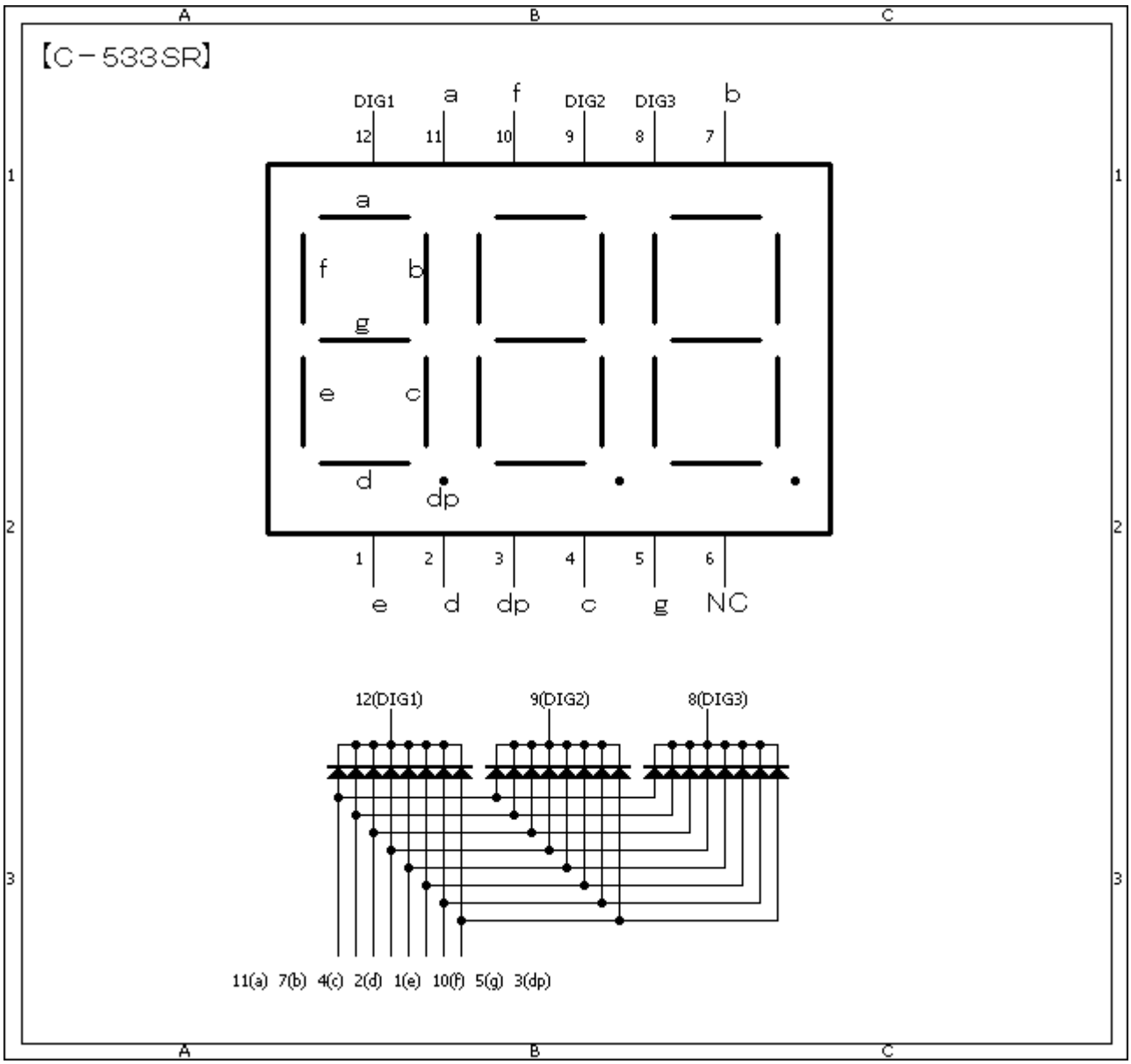
- TIMER2を使用して1msecの割り込みを発生させます。
- ダイナミック点灯処理を呼び出します。
- カウント変数(count_msec)をデクリメントします。
- ブザー用のビットをON/OFFします。

◎ダイナミック点灯処理

- 呼び出される毎に、7セグLEDの桁を順次切り替えて、表示データ(seg_data)の値を点灯させます。
- 1桁目のドットをON/OFFします。

回路図


AC100V入力
AC100V出力
100
100
SSR(ソリッドステートリレー)
C2
0.1u
SP1
圧電スピーカ
R1,2,3
10k
IC1 PIC16F88
SW1
SW2
SW3
RA2/AN2/CVREF/VREF-
A3/AN3/VreF+/C1OUT
RA4/AN4/T0CKI/C2OUT
RA5/MCLR/VPP
Vss
RB0/INT/CCP1*
RB1/SDI/SDA
RB2/SDO/RX/DT
RB3/PGM/CCP1*
RA1/AN1
RA0/AN0
RA7/OSC1/CLKI
RA6/OSC2/CLKO
Vdd
RB7/AN6/PGD/T1OSI
RB6/AN5/PGC/T1OSO/T1CKI
RB5/SS/TX/CK
RB4/SCK/SCL
R4,5,6
100
3桁7セグメントLED(C－533SR)
(カソードコモン)
SW4
C1
0.1u
Vdd
Vss
※単三電池2本
＜C－533SRの接続＞
PORTB. B0 → 7セグ(a)
PORTB. B1 → 7セグ(b)
PORTB. B2 → 7セグ(c)
PORTB. B3 → 7セグ(d)
PORTB. B4 → 7セグ(e)
PORTB. B5 → 7セグ(f)
PORTB. B6 → 7セグ(g)
PORTB. B7 → 7セグ(dp)
PORTA. B0 → 7セグ(3)
PORTA. B7 → 7セグ(2)
PORTA. B6 → 7セグ(1)
■SW1：STARTおよび設定値("分")のインクリメント(＋1)
■SW2：STOPおよび設定値("分")のデクリメント(－1)
■SW3：モード切替(オフ＝タイマーモード、オン＝設定値変更モード)
■SW4：電源スイッチ



ソースコード

power_supply_timer.c

```
//*****************************************************************************
*
/*
    ＜電源タイマー＞
*/
//*****************************************************************************
*
//■■■マクロ定義■■■
//7SEG-SELECT
#define     SEG_ON              0
#define     SEG_OFF             1
sbit        SEG1                at          PORTA.B6;
sbit        SEG1_Direction      at          TRISA.B6;
```

```
#define     SEG1_ON             SEG1 = SEG_ON
#define     SEG1_OFF            SEG1 = SEG_OFF
sbit        SEG2                at        PORTA.B7;
sbit        SEG2_Direction      at        TRISA.B7;
#define     SEG2_ON             SEG2 = SEG_ON
#define     SEG2_OFF            SEG2 = SEG_OFF
sbit        SEG3                at        PORTA.B0;
sbit        SEG3_Direction      at        TRISA.B0;
#define     SEG3_ON             SEG3 = SEG_ON
#define     SEG3_OFF            SEG3 = SEG_OFF
//7SEG-DATA
#define     SEG_DOT_ON          1
#define     SEG_DOT_OFF         0
#define     SEG_DATA            PORTB
#define     SEG_DATA_DOT        PORTB.B7
#define     SEG_DATA_Direction  TRISB
//switch
sbit        SW_START            at        PORTA.B3;
sbit        SW_STOP             at        PORTA.B4;
sbit        SW_UP               at        PORTA.B3;
sbit        SW_DOWN             at        PORTA.B4;
sbit        SW_MODE             at        PORTA.B5;
//SSR
#define     SSR_ON              1
#define     SSR_OFF             0
sbit        SSR                 at        PORTA.B2;
//BUZZER
sbit        BUZZER_BIT          at        PORTA.B1;
//other
#define     INPUT_MODE          1
#define     OUTPUT_MODE         0
#define     BYTE                unsigned  short
#define     WORD                unsigned  int
#define     DWORD               unsigned  long
#define     UN_LOCK             0
#define     LOCK                1
//*****************************************************************
*
//■■■関数&共有データ宣言■■■
extern      void        main();
extern      void        init_segment();
extern      void        init_timer();
extern      void        segment_disp();
extern      void        segment_set_data(short seg1, short seg2, short
seg3);
extern      void        interrupt();
extern      void        buzzer(short msec);
extern      DWORD       count_msec;
extern      short       dot_flg;
extern      BYTE        buzzer_flg;
//*****************************************************************
```

```
*
//■■■メイン関数■■■
void    main()
{
        DWORD   tmp, set_up_count;
        WORD    minute;
        BYTE    m1, m2, m3, mode, cnt, sec_new, sec_old;
        union {
                DWORD        _long;
                BYTE         _short[4];
        }       set_up_count_eeprom;
        //
        OSCCON = 0b01110000;        //クロックを8MHzに設定します。
        ANSEL  = 0b00000000;        //A/D変換モジュールは使用しません。
        TRISA  = 0b00111000;
        //
        mode = 0;
        SSR = SSR_OFF;
        count_msec = 0;
        dot_flg = SEG_DOT_OFF;
        //
        init_segment();
        init_timer();
        //カウント値を読み込む
        set_up_count_eeprom._short[0] = Eeprom_Read(0);
        set_up_count_eeprom._short[1] = Eeprom_Read(1);
        set_up_count_eeprom._short[2] = Eeprom_Read(2);
        set_up_count_eeprom._short[3] = Eeprom_Read(3);
        set_up_count_eeprom._long = ((set_up_count_eeprom._long <
60000) || (set_up_count_eeprom._long > 59940000)) ? 3600000 :
set_up_count_eeprom._long;
        set_up_count = set_up_count_eeprom._long;
        //割り込みを許可します。
        INTCON.PEIE = 1;
        INTCON.GIE = 1;
        //
        for (cnt = 0; cnt < 5; cnt++) {
                segment_set_data(10, 10, 10);
                buzzer(100);
                segment_set_data(11, 11, 11);
                Delay_ms(100);
        }
        //
        while (1) {
                //タイマースタート
                if ((SW_MODE == 1) && (SW_START == 0)) {
                        mode = 1;
                        SSR = SSR_ON;
                        dot_flg = SEG_DOT_ON;
                        count_msec = set_up_count;
                        sec_old = 60;
```

```
                buzzer(100);
        }
        //タイマーストップ
        if ((SW_MODE == 1) && (mode == 1) && (SW_STOP == 0)) {
                mode = 0;
                SSR = SSR_OFF;
                dot_flg = SEG_DOT_OFF;
                for (cnt = 0; cnt < 5; cnt++) {
                        segment_set_data(10, 10, 10);
                        buzzer(100);
                        segment_set_data(11, 11, 11);
                        Delay_ms(100);
                }
        }
        //タイムアウト
        if ((mode == 1) && (count_msec == 0)) {
                mode = 0;
                SSR = SSR_OFF;
                dot_flg = SEG_DOT_OFF;
                for (cnt = 0; cnt < 5; cnt++) {
                        segment_set_data(10, 10, 10);
                        buzzer(100);
                        segment_set_data(11, 11, 11);
                        Delay_ms(100);
                }
        }
        //設定値（分）のインクリメント
        if ((SW_MODE == 0) && (mode == 0) && (SW_UP == 0)) {
                set_up_count += 60000;
                set_up_count_eeprom._long = set_up_count;
                Eeprom_Write(0, set_up_count_eeprom._short[0]);
                Delay_ms(20);
                Eeprom_Write(1, set_up_count_eeprom._short[1]);
                Delay_ms(20);
                Eeprom_Write(2, set_up_count_eeprom._short[2]);
                Delay_ms(20);
                Eeprom_Write(3, set_up_count_eeprom._short[3]);
                Delay_ms(20);
        }
        //設定値（分）のデクリメント
        if ((SW_MODE == 0) && (mode == 0) && (SW_DOWN == 0)) {
                set_up_count -= 60000;
                if (set_up_count < 60000) {
                        set_up_count = 60000;
                }
                set_up_count_eeprom._long = set_up_count;
                Eeprom_Write(0, set_up_count_eeprom._short[0]);
                Delay_ms(20);
                Eeprom_Write(1, set_up_count_eeprom._short[1]);
                Delay_ms(20);
                Eeprom_Write(2, set_up_count_eeprom._short[2]);
```

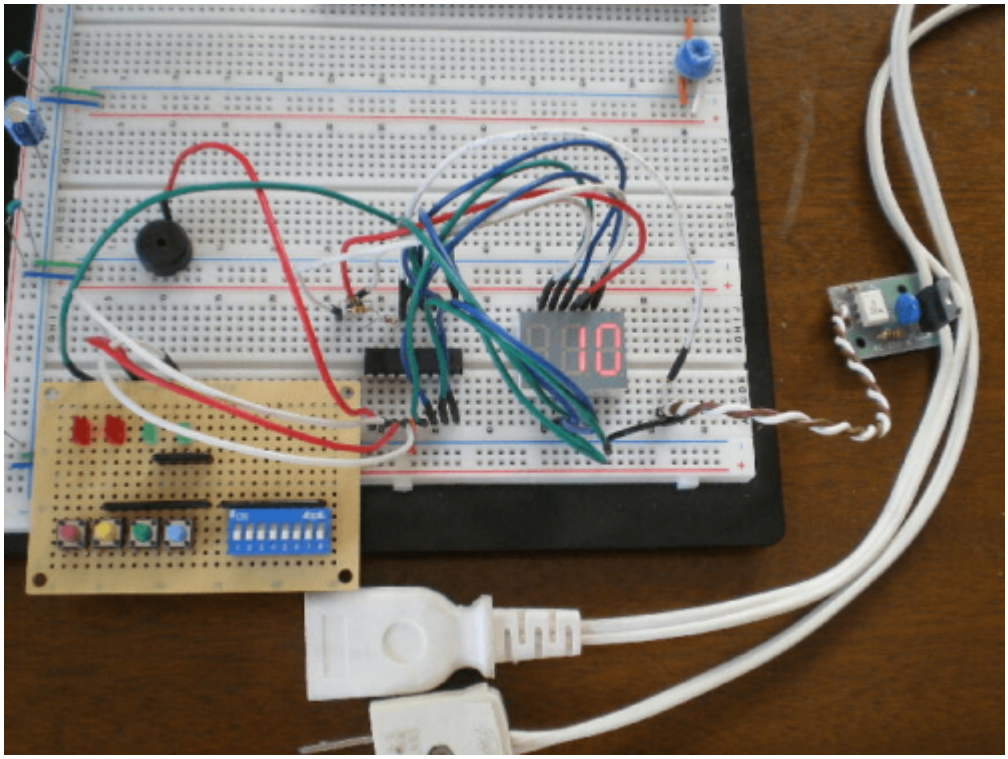
```
                        Delay_ms(20);
                        Eeprom_Write(3, set_up_count_eeprom._short[3]);
                        Delay_ms(20);
                }
                //タイムアウト直前の1分間
                if (mode == 1) {
                        tmp = count_msec;
                        minute = tmp / 60000;
                        if (minute == 0) {
                                sec_new = tmp / 1000;
                                if (sec_new != sec_old) {
                                        buzzer(100);
                                        sec_old = sec_new;
                                }
                        }
                        minute++;
                } else {
                        tmp = set_up_count;
                        minute = tmp / 60000;
                }
                //タイマー値の表示
                m1 = minute / 100;
                m2 = (minute % 100) / 10;
                m3 = minute % 10;
                if (m1 != 0) {
                        segment_set_data(m1, m2, m3);
                }
                if ((m1 == 0) && (m2 != 0)) {
                        segment_set_data(11, m2, m3);
                }
                if ((m1 == 0) && (m2 == 0)) {
                        segment_set_data(11, 11, m3);
                }
                //
                Delay_ms(100);
        }
}
//*********************************************************************
*
void    buzzer(short msec)
{
        buzzer_flg = 1;
        Vdelay_ms(msec);
        buzzer_flg = 0;
}
//*********************************************************************
*
//■■■セグメント初期化関数■■■
void    init_segment()
{
        SEG1_Direction = OUTPUT_MODE;
```

```
        SEG1_OFF;
        SEG2_Direction = OUTPUT_MODE;
        SEG2_OFF;
        SEG3_Direction = OUTPUT_MODE;
        SEG3_OFF;
        //
        SEG_DATA_Direction = 0b00000000;
        SEG_DATA = 0xFF;
}
//*****************************************************************************
*
//■■■タイマー初期化関数■■■
void    init_timer()
{
        T2CON.T2CKPS1 = 0;
        T2CON.T2CKPS0 = 0;
        T2CON.TOUTPS3 = 1;
        T2CON.TOUTPS2 = 1;
        T2CON.TOUTPS1 = 1;
        T2CON.TOUTPS0 = 1;
        TMR2 = 0;
        PIE1.TMR2IE = 1;
        PIR1.TMR2IF = 0;
        PR2 = 125;          //125=1msec/((1sec/8MHz)*4*16PS)
        T2CON.TMR2ON = 1;
}
//*****************************************************************************
*
//■■■セグメントデータ設定関数■■■
short   seg_dat[3] = {0, 0, 0};
//
void    segment_set_data(short seg1, short seg2, short seg3)
{
        seg_dat[0] = seg1;
        seg_dat[1] = seg2;
        seg_dat[2] = seg3;
}
//*****************************************************************************
*
//■■■割り込み関数■■■
DWORD   count_msec = 0;
BYTE    buzzer_flg = 0;
//
void    interrupt()
{
        if (PIR1.TMR2IF == 1) {
                PIR1.TMR2IF = 0;
                //
                segment_disp();
                if (count_msec > 0) {
                        count_msec--;
```

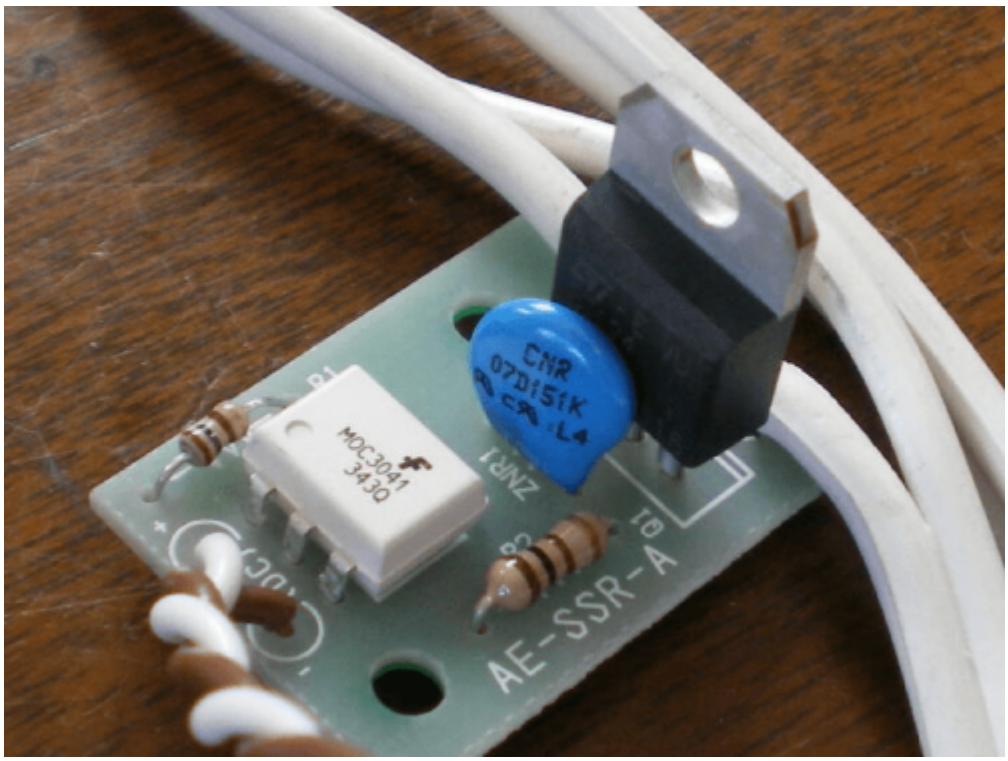
```
                }
                //
                if (buzzer_flg == 1) {
                        BUZZER_BIT = ~BUZZER_BIT;
                }
        }
}
//*****************************************************************************
*
//■■■セグメントデータ表示関数■■■
short   seg_tbl[12] = {
                0b00111111,         //0
                0b00000110,         //1
                0b01011011,         //2
                0b01001111,         //3
                0b01100110,         //4
                0b01101101,         //5
                0b01111101,         //6
                0b00100111,         //7
                0b01111111,         //8
                0b01101111,         //9
                0b01000000,         //-
                0b00000000          //space
        };
short   seg_cnt = 0;
short   dot_flg = 0;
int     dot_cnt = 0;
//
void    segment_disp()
{
        dot_cnt++;
        if (dot_cnt == 1000) {
                dot_cnt = 0;
        }
        //
        switch (seg_cnt) {
        case 0:
                SEG3_OFF;
                SEG_DATA = seg_tbl[seg_dat[0]];
                SEG1_ON;
                seg_cnt = 1;
                break;
        case 1:
                SEG1_OFF;
                SEG_DATA = seg_tbl[seg_dat[1]];
                SEG2_ON;
                seg_cnt = 2;
                break;
        case 2:
                SEG2_OFF;
                SEG_DATA = seg_tbl[seg_dat[2]];
```

```
                SEG3_ON;
                seg_cnt = 0;
                //
                if (dot_flg == SEG_DOT_OFF) {
                        return;
                }
                //
                switch (dot_cnt / 500) {
                case 0:
                        SEG_DATA_DOT = 0;
                        break;
                case 1:
                        SEG_DATA_DOT = 1;
                        break;
                }
                break;
        }
}
//*****************************************************************************
*
```

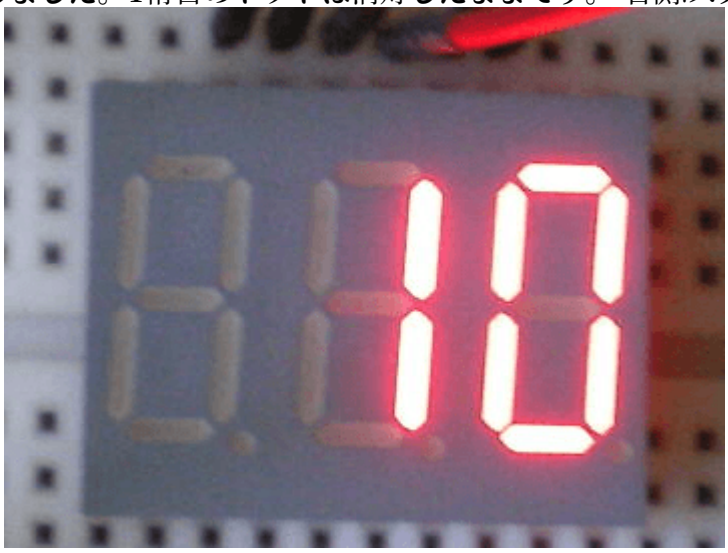
動作確認

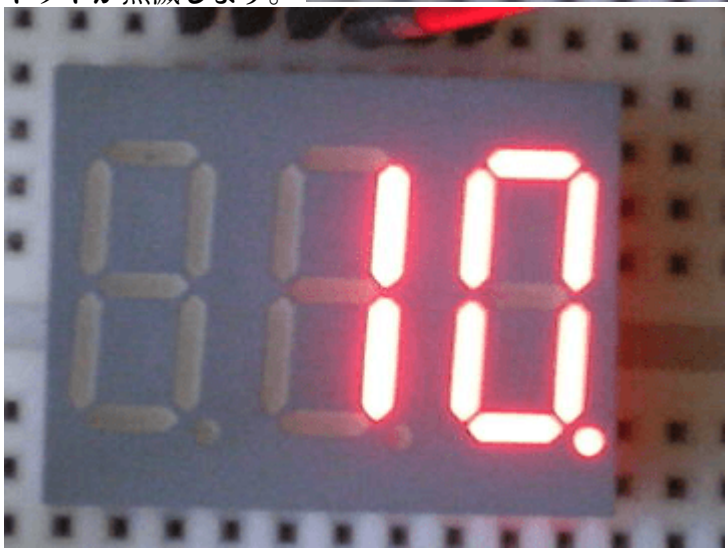


SSRには、秋月電子のキットを使用しました。全体を乾電池(3V)駆動させるため、制御入力側の抵抗を、330Ω□100Ωに変更しました。



左側:設定値を10分にしました。1桁目のドットは消灯したままです。 右側:スタート直後です。1桁目のドットが点滅します。





左側:SSRに扇風機を接続し、スタートさせました。扇風機が回っています。 右側:タイムアウトして、

扇風機が停止します。