

UpDownカウンタ(PIC16F628)

概要

これは無線仲間に依頼されて作成したものです。彼は金物関連の“ものづくり”をされている方で、ある製品を製造するときのカウントをしたいとのことでした。仕様は、次のような内容です。

- 一万台(max19999)までカウントアップ出来ること↑(up)
- カウントダウンも出来ること↓(down)
- “0”クリア出来ること↕(clear)
- 電源を切ってもカウント値を記憶していること。

動作原理

7セグメントのLEDの点灯方式は、ダイナミック点灯としました。但、常に3つのスイッチ(up/down/clear))を監視しなければなりませんのでダイナミック点灯の処理はタイマー割り込み(約1msec)の中で行いました。メインの処理では、これらの3つのスイッチを常に監視し、何れかが押されるとその処理を行いカウンタ値をダイナミック点灯処理に引き渡します。ダイナミック点灯の周期を1msecにしましたので殆どチラツキはありません。尚、7セグメントLEDは4桁しかありませんので、1万台の時には単一LEDの点灯により行いました。

- 1万未満の時には、単一LEDは点滅状態
- 1万以上の時には、単一LEDは点灯状態

カウンタ値の保存は、PICのEEPROMを利用します。PICの規格では、内蔵EEPROM消去/書き込みサイクル10,000,000回と書いてありますので、今回の場合、単純な計算では、500ヶ月(1ヶ月あたり最大2万個の製造として)となりますので十分だと思われます。

※EEPROMとは? EEPROMは、データを格納するためのメモリです。特徴は電源がOFFになっても記憶内容が消えることが無い不揮発性メモリになっているということです。従って、書き換えが出来てしかもずっと取っておきたいパラメータなどを格納しておくのに便利です。

回路図

7セグメントのLEDは、手持ちの適当な物(アノードコモン4桁)を利用しました。



```
//*****
*

#define          LED          PORTA.F2
#define          UP           (tmp.F3 == 0)
#define          DOWN        (tmp.F4 == 0)
#define          CLEAR       (tmp.F5 == 0)

//*****
*

static unsigned   char    segIndex, seg0, seg1, seg2, seg3, flg10000;

void    interrupt()
{
    if (INTCON.T0IF == 1) {
```

```
INTCON.T0IF = 0;
//
switch (segIndex) {
case 3:
    PORTB = seg3;
    PORTA.F0 = 1;
    PORTA.F1 = 1;
    PORTA.F6 = 0;
    PORTA.F7 = 1;
    segIndex = 0;
    break;
case 2:
    PORTB = seg2;
    PORTA.F0 = 1;
    PORTA.F1 = 1;
    PORTA.F6 = 1;
    PORTA.F7 = 0;
    segIndex++;
    break;
case 1:
    PORTB = seg1;
    PORTA.F0 = 0;
    PORTA.F1 = 1;
    PORTA.F6 = 1;
    PORTA.F7 = 1;
    segIndex++;
    break;
case 0:
    PORTB = seg0;
    PORTA.F0 = 1;
    PORTA.F1 = 0;
    PORTA.F6 = 1;
    PORTA.F7 = 1;
    segIndex++;
    break;
}
}
if (PIR1.TMR1IF == 1) {
    PIR1.TMR1IF = 0;
    if (flg10000 == 0)
        LED = ~LED;
    else
        LED = 1;
}
}

//*****
*

static unsigned char segmentTbl[13] = {
    0b00100001, // 0
```

```
0b11100111,    // 1
0b00110100,    // 2
0b10100100,    // 3
0b11100010,    // 4
0b10101000,    // 5
0b00101000,    // 6
0b11100001,    // 7
0b00100000,    // 8
0b10100000,    // 9
0b11111110,    // -
0b00000000,    // all-on
0b11111111,    // all-off
};

unsigned char ChangeSegData(unsigned char c)
{
    switch (c) {
        case '0':
            return (segmentTbl[0]);
            break;
        case '1':
            return (segmentTbl[1]);
            break;
        case '2':
            return (segmentTbl[2]);
            break;
        case '3':
            return (segmentTbl[3]);
            break;
        case '4':
            return (segmentTbl[4]);
            break;
        case '5':
            return (segmentTbl[5]);
            break;
        case '6':
            return (segmentTbl[6]);
            break;
        case '7':
            return (segmentTbl[7]);
            break;
        case '8':
            return (segmentTbl[8]);
            break;
        case '9':
            return (segmentTbl[9]);
            break;
        case '-':
            return(segmentTbl[10]);
            break;
        case '*':
```

```
        return(segmentTbl[11]);
        break;
    case ' ':
        return(segmentTbl[12]);
        break;
    }
}

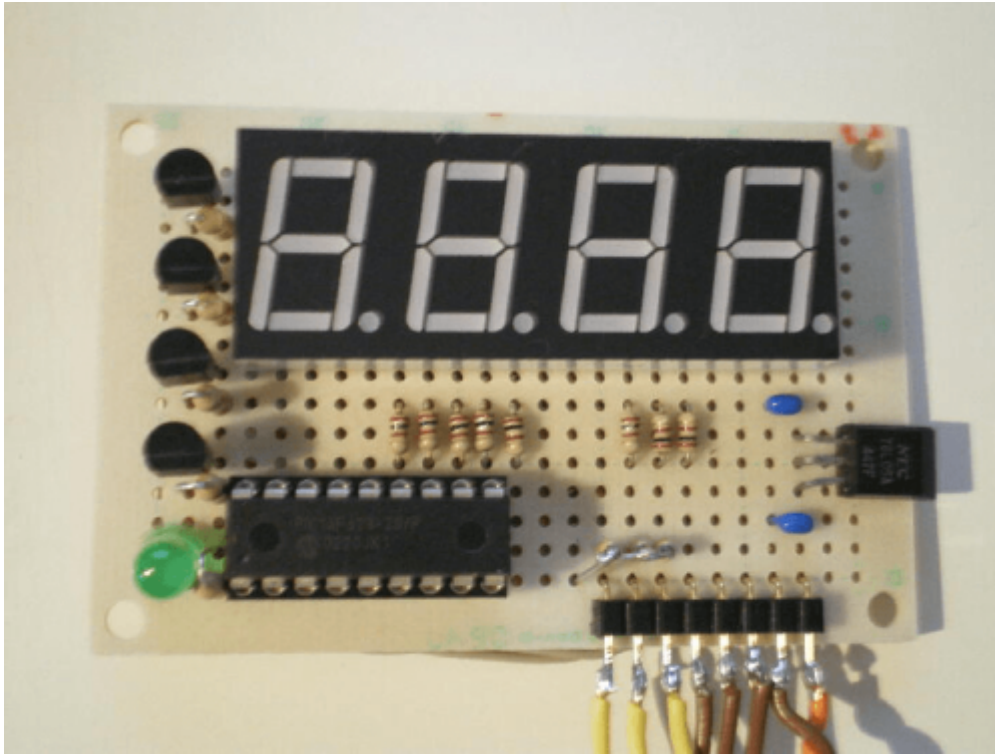
void main()
{
    static unsigned int cnt;
    static unsigned char buf[10], tmp, tmp1, tmp2;
    //
    CMCON = 0b00000111;           // コンパレータは使用しない。
    TRISA = 0b00111000;           // PORTAのモードを背呈する。
    TRISB = 0b00000000;           // PORTBのモードを背呈する。
    //
    INTCON.GIE = 0;                // 割り込みを無効にする。
    INTCON.PEIE = 1;               // Peripheral Interrupt Enable
    // タイマー0を設定する。
    INTCON.T0IE = 1;               // TMR0 Overflow Interrupt
    Enable
    INTCON.T0IF = 0;               // TMR0 Overflow Interrupt Flag
    OPTION_REG.T0CS = 0;
    OPTION_REG.PSA = 0;
    OPTION_REG.PS0 = 0;
    OPTION_REG.PS1 = 1;
    OPTION_REG.PS2 = 0;
    TMR0 = 0;
    // タイマー1を設定する。
    PIE1.TMR1IE = 1;              // TMR1 Overflow Interrupt
    Enable
    PIR1.TMR1IF = 0;              // TMR1 Overflow Interrupt Flag
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
    //
    segIndex = 0;
    // 保存されたデータを読み込む
    tmp1 = Eeprom_Read(0);
    Delay_ms(20);
    tmp2 = Eeprom_Read(1);
    Delay_ms(20);
    cnt = (tmp2 << 8) | tmp1;
    cnt = (cnt > 19999) ? 0 : cnt;
    // 表示データをセットする。
    IntToStr(cnt, buf);
    flg10000 = (buf[1] == '1') ? 1 : 0;
    seg3 = ChangeSegData(buf[2]);
    seg2 = ChangeSegData(buf[3]);
    seg1 = ChangeSegData(buf[4]);
}
```

```
seg0 = ChangeSegData(buf[5]);
//
INTCON.GIE = 1;    // これ以降の処理で割り込みを許可する。
//
while(1) {
    tmp = PORTA;
    if (UP | DOWN | CLEAR) {    // スイッチの押下をチェックする。
        if (UP) {
            if (cnt < 19999)
                cnt++;
        }
        if (DOWN) {
            if (cnt > 0)
                cnt--;
        }
        if (CLEAR) {
            cnt = 0;
        }
        // データを保存する。
        Eeprom_Write(0, cnt & 0xFF);
        Delay_ms(20);
        Eeprom_Write(1, (cnt >> 8) & 0xFF);
        Delay_ms(20);
        //
        while (UP | DOWN | CLEAR) {
            tmp = PORTA;
            Delay_ms(1);
        }
        // 表示データをセットする。
        IntToStr(cnt, buf);
        flg10000 = (buf[1] == '1') ? 1 : 0;
        seg3 = ChangeSegData(buf[2]);
        seg2 = ChangeSegData(buf[3]);
        seg1 = ChangeSegData(buf[4]);
        seg0 = ChangeSegData(buf[5]);
    }
}

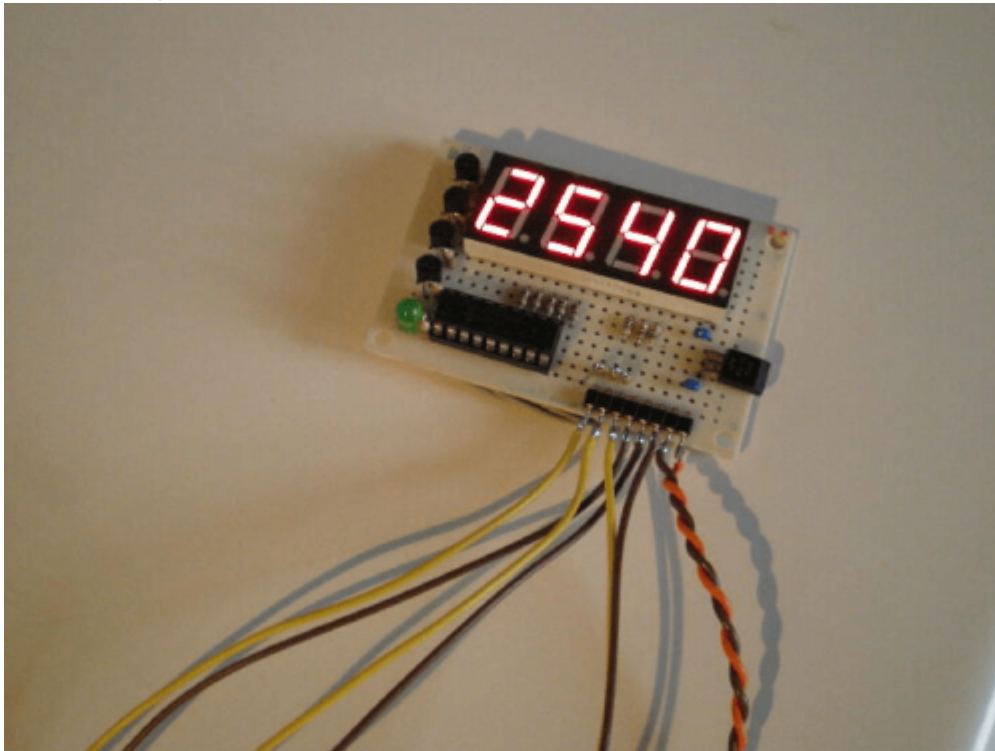
//*****
*
```

動作確認

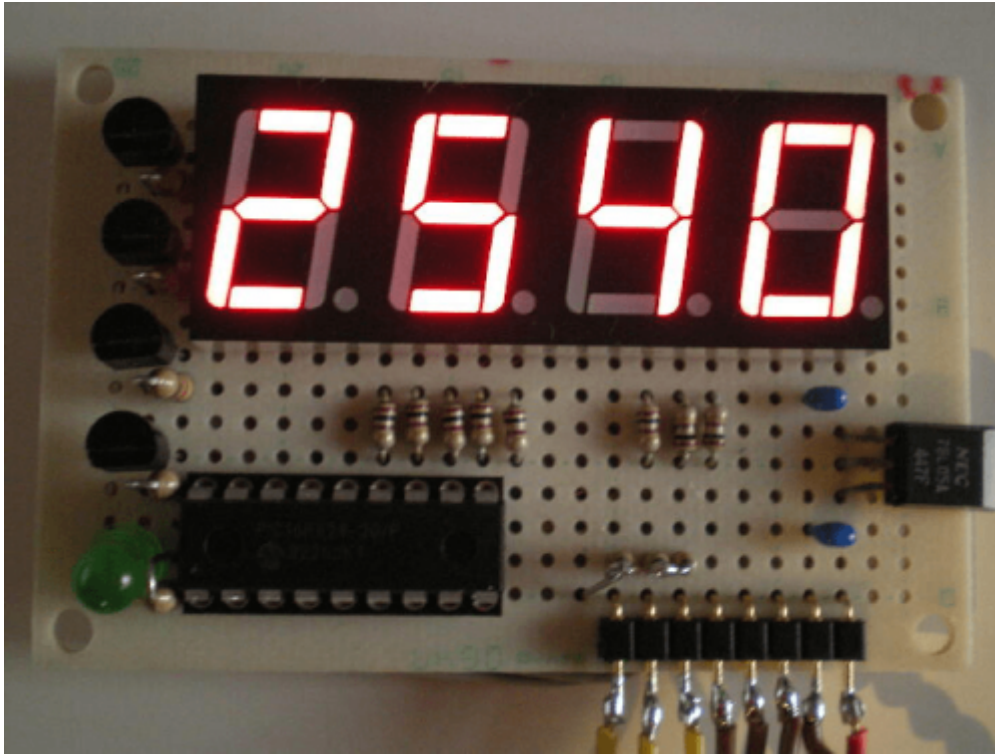
小さな蛇の目基板に実装しました。



端子には□Up/Down/Clearのスイッチを接続します。そして電源(8~12V)も。。。

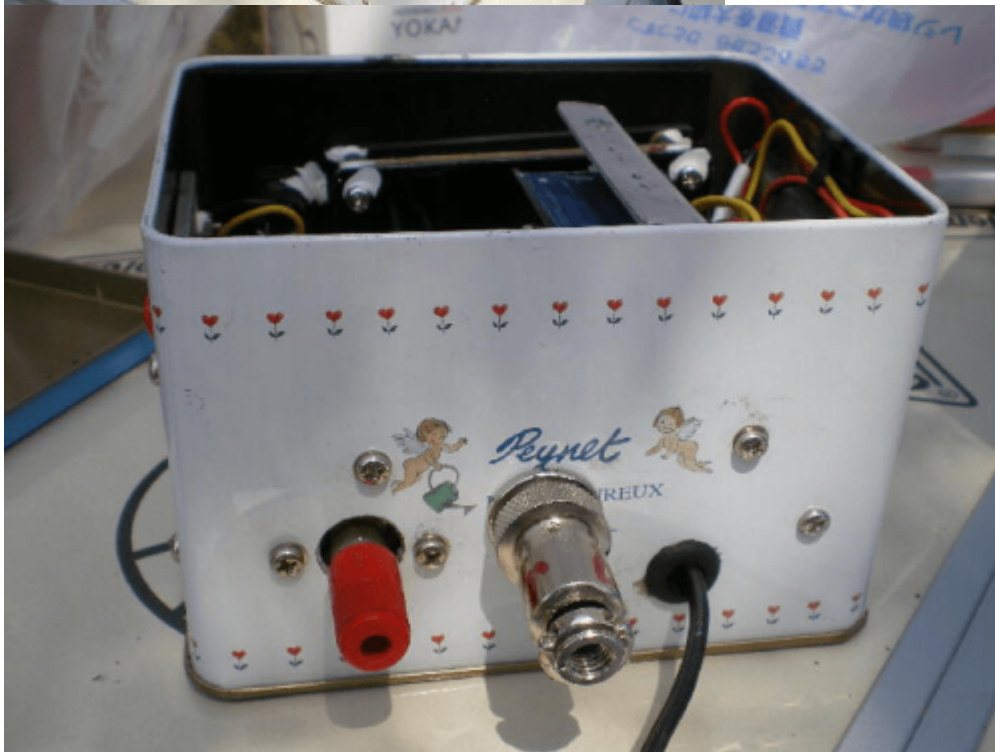
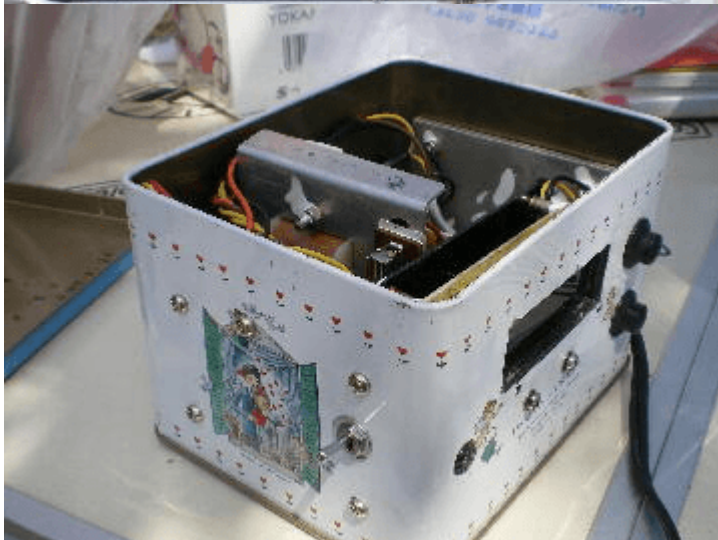
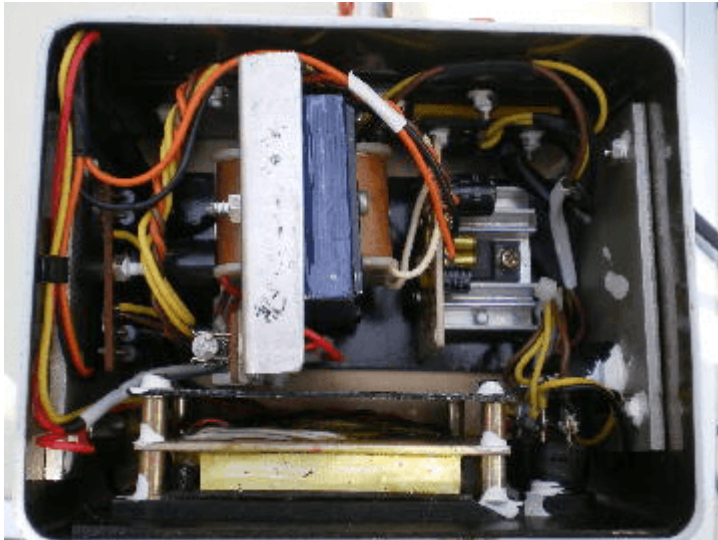


割と大き目の7セグメントLEDを使用しましたので見易いと思います。



これをケースに入れるのは、依頼者にお任せしました。





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
うごくといいな - <http://deepsky.jp/wiki/>

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:otherpic:161>

Last update: **2020/05/01 19:14**

