簡易パルス幅測定器

概要

周期的なパルスであれば、周波数カウンタでも1サイクルの幅を求めることができます。 1サイクルの幅(T)=1秒÷周波数(Hz)

しかし、この方法では、次のような問題があります。

- 単発のパルスを測定することが出来ない。
- 非常に長いパルス(例えば、1時間、2日間など)を測定することが出来ない。
- パルス信号の立ち上がりや立下りにおける幅を求めることが出来ない。

そこで、今回は、これらの問題を解決すべく、“ 簡易パルス幅測定器 “ を製作しました。

<仕様>

- 精度(最小単位)は、0.1msecとします。
- 測定モードは、4種類から選択することが出来ます。
  - パルスの立上り[↑]-立下り[↓]までの時間
  - パルスの立上り[↑]-立下り[↓]-立上り[↑]までの時間
  - パルスの立下り[↓]-立上り[↑]までの時間
  - パルスの立下り[↓]-立上り[↑]-立下り[↓]までの時間
- 測定時間の最大は、約59時間とします。

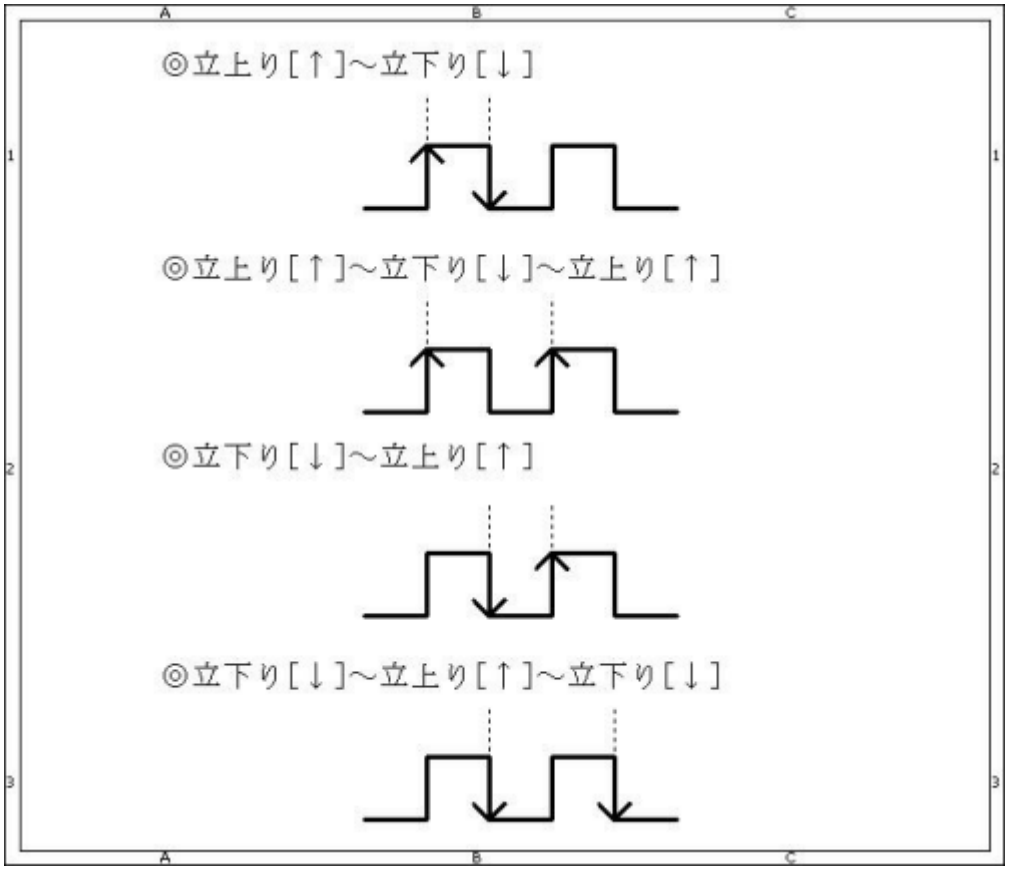
動作原理

<基準時間(0.1msec)> 精度の最小単位である□0.1msecの基準時間を得ます。

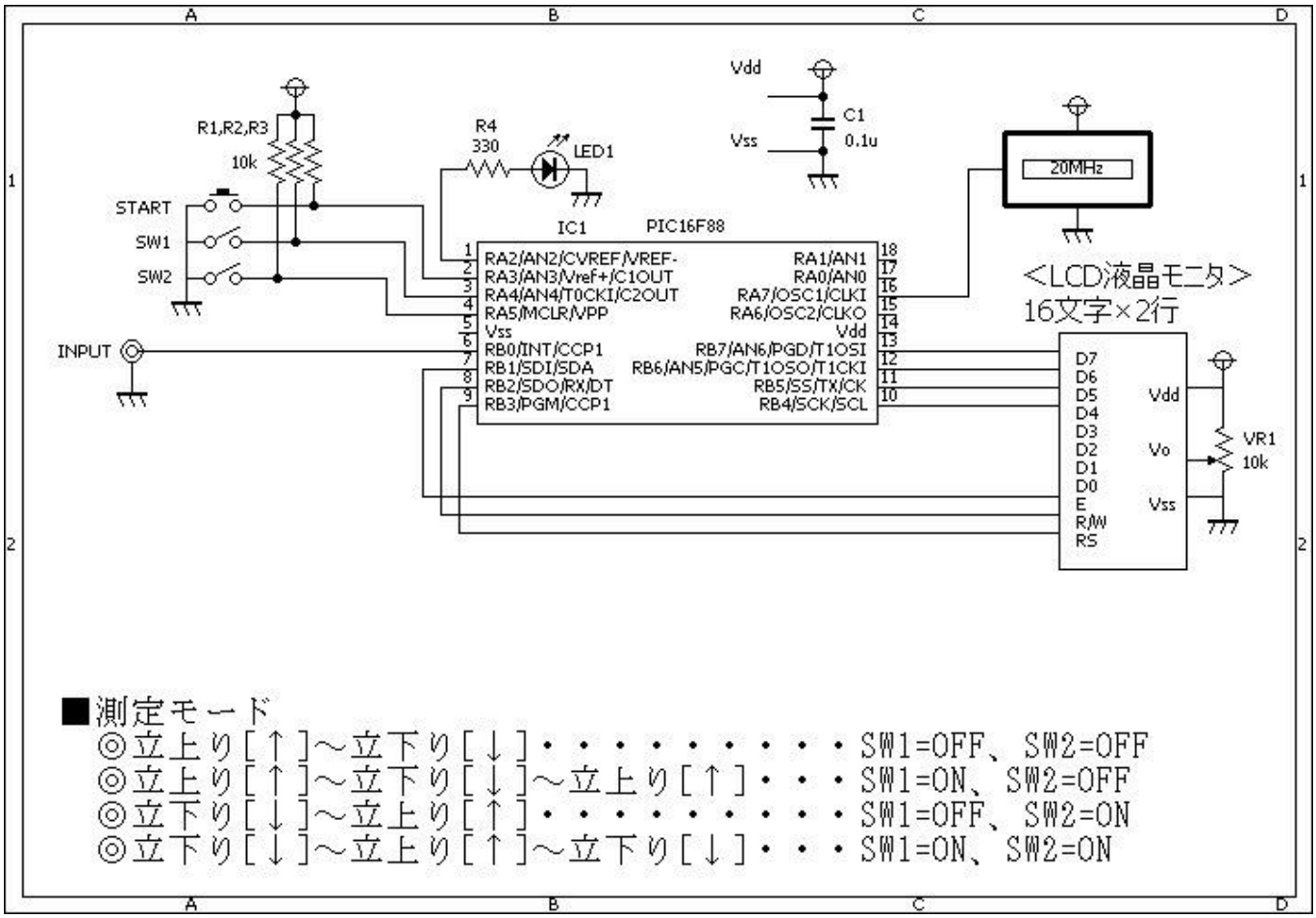
- 精度の高い、クリスタルオシレータ(20MHz)を使用します。
  ※測定結果の精度に最も影響を与える部分です。
- CCPモジュールをコンペアモードで使用し、0.1msecの割り込みを発生させます。
- 測定時間の最大は、約59時間(214748364.7msec)になります。(これはlong変数の最大値です)

<測定モード> スイッチ(SW1□SW2)によって、次の4つの測定モードから、一つを選択します。

- パルスの、立上り[↑]-立下り[↓]の時間を測定します。
- パルスの、立上り[↑]-立下り[↓]-立上り[↑]の時間を測定します。
- パルスの、立下り[↓]-立上り[↑]の時間を測定します。
- パルスの、立下り[↓]-立上り[↑]-立下り[↓]の時間を測定します。


◎立上り[↑]～立下り[↓]
◎立上り[↑]～立下り[↓]～立上り[↑]
◎立下り[↓]～立上り[↑]
◎立下り[↓]～立上り[↑]～立下り[↓]

回路図


R1,R2,R3
10k
START
SW1
SW2
INPUT
R4
330
LED1
Vdd
C1
Vss
0.1u
20MHz
IC1
PIC16F88
RA2/AN2/CVREF/VREF-
RA3/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
<LCD液晶モニタ>
16文字×2行
D7
D6
D5
D4
D3
D2
D1
D0
E
R/W
RS
Vdd
Vo
Vss
VR1
10k
■測定モード
◎立上り[↑]～立下り[↓]・・・・・・・・・・SW1=OFF、SW2=OFF
◎立上り[↑]～立下り[↓]～立上り[↑]・・・SW1=ON、SW2=OFF
◎立下り[↓]～立上り[↑]・・・・・・・・・・SW1=OFF、SW2=ON
◎立下り[↓]～立上り[↑]～立下り[↓]・・・SW1=ON、SW2=ON

ソースコード

pulse_measurement_v1.c

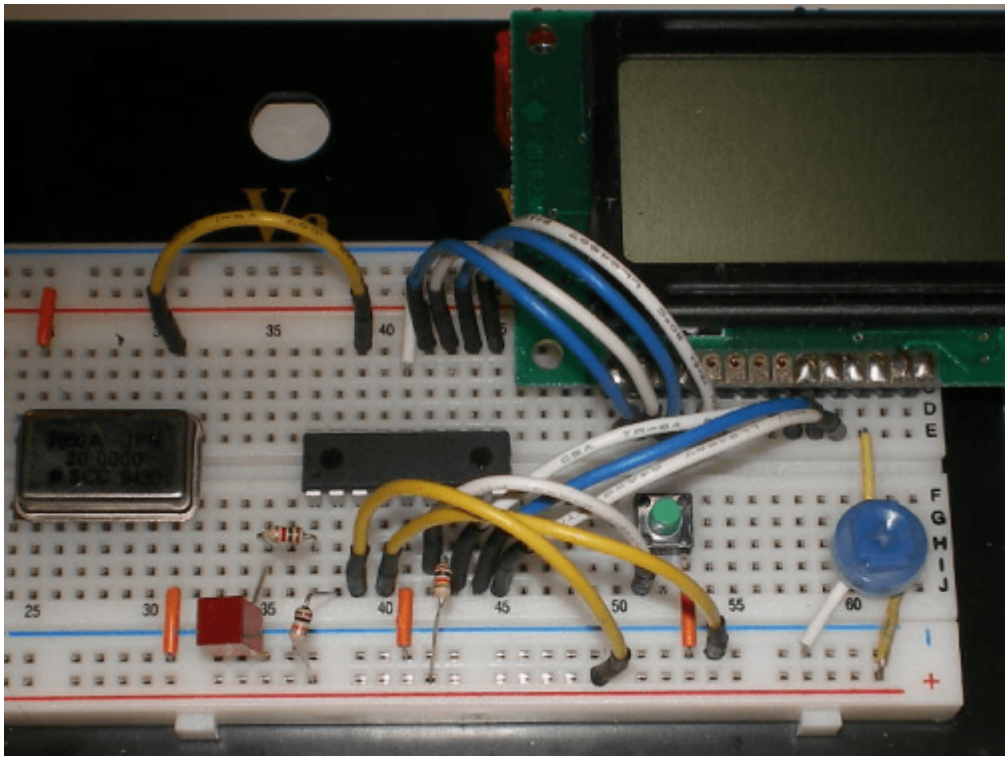
```
//*****************************************************************************
*
/*
    「簡易パルス測定器」
  ■測定モード
    ・立上がり→立下り
    ・立上がり→立下り→上がり
    ・立下り→立上がり
    ・立下り→立上がり→立下り
  ■測定精度□□□□□□□□
  ■測定時間＝最大約５９時間
*/
//*****************************************************************************
*

#define       LED             PORTA.F2

#define       SW_START    PORTA.F3
#define       SW1             PORTA.F4
#define       SW2             PORTA.F5

#define       INPUT         PORTB.F0

//*****************************************************************************
*

static unsigned    long    cnt;

void   interrupt()
{
    PIR1.CCP1IF = 0;
    //
    cnt++;
    LED = ~LED;
}

//*****************************************************************************
*

unsigned   long    measurement(short mode)
{
    TMR1L = 0;
    TMR1H = 0;
    cnt = 0;
    //
    switch (mode) {
    case 0:         //↑↓
```

```
        while (INPUT == 1)     //“０”になるまで待つ
            ;
        while (INPUT == 0)     //立上がりをチェック
            ;
        T1CON.TMR1ON = 1;      //カウント開始
        while (INPUT == 1)     //立下りをチェック
            ;
        T1CON.TMR1ON = 0;      //カウント停止
        break;
    case 1:         //↑↓↑
        while (INPUT == 1)     //“０”になるまで待つ
            ;
        while (INPUT == 0)     //立上がりをチェック
            ;
        T1CON.TMR1ON = 1;      //カウント開始
        while (INPUT == 1)     //立下りをチェック
            ;
        while (INPUT == 0)     //立上がりをチェック
            ;
        T1CON.TMR1ON = 0;      //カウント停止
        break;
    case 2:         //↓↑
        while (INPUT == 0)     //“１”になるまで待つ
            ;
        while (INPUT == 1)     //立下がりをチェック
            ;
        T1CON.TMR1ON = 1;      //カウント開始
        while (INPUT == 0)     //立上りをチェック
            ;
        T1CON.TMR1ON = 0;      //カウント停止
        break;
    case 3:         //↓↑↓
        while (INPUT == 0)     //“１”になるまで待つ
            ;
        while (INPUT == 1)     //立下がりをチェック
            ;
        T1CON.TMR1ON = 1;      //カウント開始
        while (INPUT == 0)     //立上がりをチェック
            ;
        while (INPUT == 1)     //立下りをチェック
            ;
        T1CON.TMR1ON = 0;      //カウント停止
        break;
    }
    //
    return (cnt);
}

//*****************************************************************************
*
```

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

//*****************************************************************************
*

void   main()
{
    static    char    buf[16], mode;
    static    unsigned    long    msec;
    //
//    OSCCON = 0b01110000;
    CMCON  = 0b00000111;
    ANSEL  = 0b00000000;
    TRISA  = 0b11111011;
    TRISB  = 0b00000001;
    // TIMER1の設定
    PIE1.TMR1IE = 0;
    PIR1.TMR1IF = 0;
    T1CON.T1CKPS0 = 0;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 0;
    TMR1L = 0;
    TMR1H = 0;
    // CCPの設定
    PIE1.CCP1IE = 1;
    PIR1.CCP1IF = 0;
    CCP1CON.CCP1M3 = 1;
    CCP1CON.CCP1M2 = 0;
    CCP1CON.CCP1M1 = 1;
    CCP1CON.CCP1M0 = 1;
    CCPR1L = 0x7D;    // 0.1msec...(1÷20000000)*4*4*125
    CCPR1H = 0x00;    //
    //□□□の初期化
    Lcd_Custom_Config(&PORTB, 7, 6, 5, 4, &PORTB, 3, 2, 1);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(LCD_CLEAR);
    Lcd_Custom_Out(1, 1, "PulseMeasurement");
    Delay_ms(500);
    Lcd_Custom_Cmd(LCD_CLEAR);
    Lcd_Custom_Out(2, 13, "msec");
    //割り込みを許可する。
    INTCON.PEIE = 1;
    INTCON.GIE = 1;
    //
    while (1) {
```

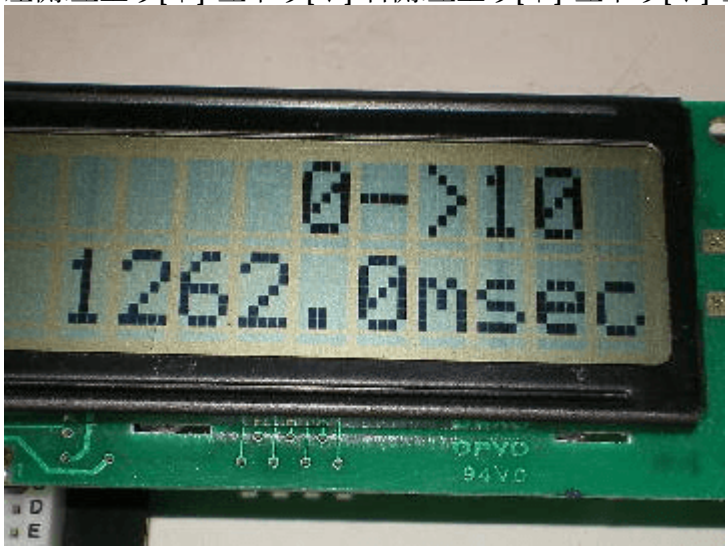
```
        if ((SW1 == 1) && (SW2 == 1)) {
            mode = 0;
            Lcd_Custom_Out(1, 11, "0->10 ");
        }
        if ((SW1 == 0) && (SW2 == 1)) {
            mode = 1;
            Lcd_Custom_Out(1, 11, "0->101");
        }
        if ((SW1 == 1) && (SW2 == 0)) {
            mode = 2;
            Lcd_Custom_Out(1, 11, "1->01 ");
        }
        if ((SW1 == 0) && (SW2 == 0)) {
            mode = 3;
            Lcd_Custom_Out(1, 11, "1->010");
        }
        LED = 0;
        Lcd_Custom_Out(1, 1, "     ");
        if (SW_START == 1) {
            continue;
        }
        while (Button(&PORTA, 3, 1, 1) == 0)
            ;
        //
        Lcd_Custom_Out(1, 1, "start");
        Lcd_Custom_Out(2, 1, "            ");
        msec = measurement(mode);
        LongToStr(msec, buf);
        buf[12] = 0x00;
        buf[11] = buf[10];
        buf[10] = '.';
        Lcd_Custom_Out(2, 1, buf);
    }
}

//*****************************************************************************
```

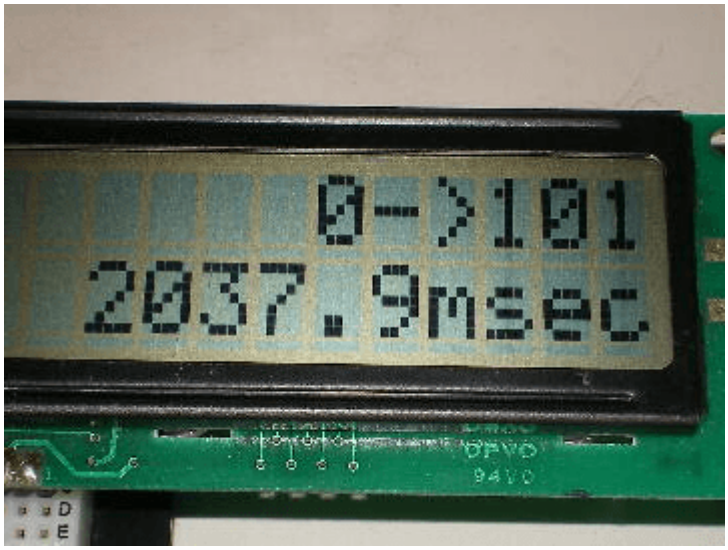
動作確認



時間の精度は、このクリスタルオシレータの精度に左右されます。

左側:立上り[↑]-立下り[↓] 右側:立上り[↑]-立下り[↓]-立上り[↑]





左側:立下り[↓]~立上り[↑] 右側:立下り[↓]~立上り[↑]~立下り[↓]