

周波数表示ユニット2(プログラム全面見直し)

概要

以前に作成しました、「周波数表示ユニット?U(真空管ラジオ用)」のプログラムを全面的に見直しました。ハードウェアはそのままです。大きな変更点は、次の通りです。

1. 以前は、正確な0.1秒を得るためにPIC内臓のCCPモジュールとTIMER1モジュールを使いましたが、
/ 今度は、シンプルにTIMER1モジュールのみで実現(関数化含む)しました。
2. コンパイル時にマクロを利用して機能を切り替えることが出来ます。(何れかを選択)
#define FC_TYPE_A *今回新規に追加した機能
#define FC_TYPE_B *以前の機能

<FC_TYPE_A> sw1: プリスケーラの切り替え

- sw1=11/1
- sw1=01/8

sw2: ゲートタイムの切り替え

- sw2=11秒
- sw2=00.1秒

<FC_TYPE_B> sw1: -455kHzの有無切り替え

- sw1=1-455Khz
- sw1=0-0Khz

sw2: 表示レンジの切り替え

- sw2=1Hz表示
- sw2=0kHz表示

ソースコード

RadioFC4.c

```
/*
<周波数カウンター>

コンパイル時にマクロを利用して機能を切り替えることが出来ます。

FC_TYPE_A
sw1: プリスケーラの切り替え
sw1=11/1
sw1=01/8
sw2: ゲートタイムの切り替え
```

```

□□□□□ sw2=1□1秒
□□□□□ sw2=0□0.1秒

□□FC_TYPE_B□
□□ sw1: -455kHzの有無切り替え
□□□□□ sw1=1□-455Khz
□□□□□ sw1=0□-0Khz
□□ sw2: 表示レンジの切り替え
□□□□□ sw2=1□Hz表示
□□□□□ sw2=0□kHz表示
*/

//*****
*

#define    FC_TYPE_A

#define    sw1    PORTB.F2
#define    sw2    PORTB.F3

//*****
*

void    interrupt()
{
    if (PIR1.TMR1IF == 1) {
        PIR1.TMR1IF = 0;
        //
        TRISA.F4 = 0;           // ゲートを閉める。
        PORTA.F4 = 0;
        T1CON.TMR1ON = 0;      // TIMER1を停止する。
    }
}

//*****
*

unsigned    long    FreqMeasurement100msec()
{
    unsigned    long    freq;
    //
    TRISA.F4 = 0;           // ゲートを閉める。
    PORTA.F4 = 0;
    // TIMER0の設定
    INTCON.T0IF = 0;
    TMR0 = 0;
    // TIMER1の設定
    TMR1L = 0xB0;
    TMR1H = 0x3C;
    //
    freq = 0;

```

```

// 割り込みを許可する。
INTCON.PEIE = 1;
INTCON.GIE = 1;
// 開始
T1CON.TMR10N = 1;
//
Delay_Cyc(2);
asm    nop;
asm    nop;
asm    nop;
asm    nop;
asm    nop;
asm    nop;
asm    nop;
//
TRISA.F4 = 1;           // ゲートを開ける。
// 測定
while (T1CON.TMR10N != 0) {
    if (INTCON.T0IF == 1) {
        INTCON.T0IF = 0;
        freq++;
    }
}
if (INTCON.T0IF == 1) {
    INTCON.T0IF = 0;
    freq++;
}
freq *= 256;
freq += TMR0;
return (freq);
}

// *****
*

void main()
{
    static unsigned long    freq, temp;    // 0...4294967295
    static unsigned char    buf[12];
    static unsigned short   cnt, prescaler, gateTime;
    // アナログの設定
    ANSEL = 0b00000000;    // → 今回は使用しない。
    // ポートの設定
    TRISA = 0b10111100;
    TRISB = 0b00001100;
    OPTION_REG.F7 = 0;    // PORTBをプルアップ設定する。
    // TIMER0の設定
    INTCON.T0IE = 0;
    INTCON.T0IF = 0;
    OPTION_REG.T0CS = 1;
    OPTION_REG.T0SE = 0;

```

```
#ifdef FC_TYPE_A
    OPTION_REG.PSA = 1;
    OPTION_REG.PS0 = 0;
    OPTION_REG.PS1 = 0;
    OPTION_REG.PS2 = 0;
#endif
#ifdef FC_TYPE_B
    OPTION_REG.PSA = 0;
    OPTION_REG.PS0 = 0;
    OPTION_REG.PS1 = 1;
    OPTION_REG.PS2 = 0;
#endif
// TIMER1の設定
PIE1.TMR1IE = 1;
PIR1.TMR1IF = 0;
T1CON.T1CKPS0 = 1;
T1CON.T1CKPS1 = 1;
T1CON.TMR1ON = 0;
//
#ifdef FC_TYPE_A
    prescaler = 1;
    gateTime = 10;
#endif
#ifdef FC_TYPE_B
    prescaler = 8;
    gateTime = 1;
#endif
//
Lcd_Custom_Config(&PORTB,4,5,6,7,&PORTA,1,0,6);
Lcd_Custom_Cmd(LCD_CURSOR_OFF);
Lcd_Custom_Out(1, 1, "FreqCounterEx R1");
Delay_ms(1000);
Lcd_Custom_Cmd(LCD_CLEAR);
//
while (1) {
    freq = 0;
    for (cnt = 0; cnt < gateTime; cnt++) {
        freq += FreqMeasurement100msec();
    }
    // 補正
    freq = freq * prescaler;
    freq = freq * (10 / gateTime);
#ifdef FC_TYPE_B
    // 00000Hzの有無
    if (sw1 == 1) {
        freq -= 455000;
        Lcd_Custom_Out(2, 1, "[-455kHz][0.1s]");
    } else {
        Lcd_Custom_Out(2, 1, "          [0.1s]");
    }
}
// 表示レンジの切り替え
```

```
    if (sw2 == 1) {
        LongToStr(freq, buf);
        Lcd_Custom_Out(1, 1, &buf[3]);
        Lcd_Custom_Out(1, 9, "Hz [1/8]");
    } else {
        temp = freq / 1000;
        if ((freq - (temp * 1000)) > 500) {
            temp++;
        }
        LongToStr(temp, buf);
        Lcd_Custom_Out(1, 1, &buf[3]);
        Lcd_Custom_Out(1, 9, "kHz[1/8]");
    }
#endif
#ifdef FC_TYPE_A
    // 表示
    LongToStr(freq, buf);
    Lcd_Custom_Out(1, 1, &buf[3]);
    Lcd_Custom_Out(1, 9, "Hz");
    // プリスケラの切り替え
    if (sw1 == 1) {
        OPTION_REG.PSA = 1;
        OPTION_REG.PS0 = 0;
        OPTION_REG.PS1 = 0;
        prescaler = 1;
        Lcd_Custom_Out(2, 1, "[1/1]");
    } else {
        OPTION_REG.PSA = 0;
        OPTION_REG.PS0 = 0;
        OPTION_REG.PS1 = 1;
        prescaler = 8;
        Lcd_Custom_Out(2, 1, "[1/8]");
    }
    // ゲートタイムの切り替え
    if (sw2 == 1) {
        gateTime = 10;
        Lcd_Custom_Out(2, 7, "[1sec]  ");
    } else {
        gateTime = 1;
        Lcd_Custom_Out(2, 7, "[100msec]");
    }
}
#endif
}

//*****
*
```

動作確認

<FC_TYPE_A>プリスケラ(1/1)、ゲートタイム(1秒) これで1Hz単位での表示が可能となります。



<FC_TYPE_A>プリスケラ(1/8)、ゲートタイム(1秒) 精度8Hz



<FC_TYPE_A>プリスケラ(1/1)、ゲートタイム(0.1秒) 精度10Hz



<FC_TYPE_A>プリスケラ(1/8)、ゲートタイム(0.1秒) 精度80Hz



<FC_TYPE_B>プリスケラ(1/8)、ゲートタイム(0.1秒)□-455kHz□Hz表示 精度80Hz



<FC_TYPE_B>プリスケラ(1/8)、ゲートタイム(0.1秒)□-455kHz□kHz表示 精度80Hz



<FC_TYPE_B>プリスケラ(1/8)、ゲートタイム(0.1秒)□Hz表示 精度80Hz



<FC_TYPE_B>プリスケラ(1/8)、ゲートタイム(0.1秒) kHz表示 精度80Hz



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:pic16f88:58>

Last update: **2020/05/01 18:20**

