```
                              628_cnt_pres_main
/************************************************
  Simple frequency counter read out prescaler
   1602 display By nobcha all right reserved

  Ver 0.1 08/25/2010  Ver0.2 02/15/2011
  Ver 2.0 08/20/2011
  Ver 3.0 11/19/2011  PIC16F683
  Ver 4.0 12/04/2011  PIC16F628A

   PIC16F683 + 4bitLCD + Prescaler read out
   PIN Assign
                #17,18,1,2 RA0-3:LCD DATA
                #16,15 CLKI,CLKO:12.8kHz OSC IN
                #13 RB7:Prescaler read out switching
                #11 RB5:Monitor LED
                #8 RB2:LCD RS
                #6 RB0:LCD ENABLE
                #4 RA5:MCLR
                #3 RA4:TMR0 INPUT

         SC1602 pin connection via 4bit mode
                #1      Vdd=5V
                #2      Vss=GND
                #3      LCD contrust center of 2k VOL
                #4      RS      RB0
                #5      R/W     GND
                #6      E       RB2

                #11-14  DATA    RA0-3

   TMR0 is counter with 1/64 prescaler
   TMR1 is gate time controller as set (65536-32000)
                10mS Time ->compensate int process delay 11
                65536 - 31989 = 33547

   CLK HS 12.8MHz

  Development Circumstance
  MPLAB IDE V8.73a   HiTECH C V9.82

  Counter data is put on to GPIO2 which ia TMR0 input.
  TMR1 is worked for 10mS gate. TMR0 has 1:64 prescaler.
  Overflow of TMR0 is counted on  count_1.
************************************************/
#define _LEGACY_HEADERS
#define _XTAL_FREQ 12800000
#define TOGGLE_pres RB7
#define TMR0_INPUT TRISB7
```

```
#define MON_LED RB5

#define ACM0802 1                       // 8 characters * 2 lines

#include <htc.h>
#include <pic.h>
#include <stdio.h>
#include "lcd.h"


__CONFIG(BORDIS & UNPROTECT & PWRTEN & WDTDIS & MCLREN & HS );

unsigned char timeup, count_1, pre_pad, tmr0_value ;
long read_data;

void cnt_setup(void){
        TMR0_INPUT=0;                   // RB7 disable
        TMR0 = 0 ;                      // TMR0 clear
        TMR1L = 0;                      // Clear Low Byte of TMR1
        TMR1H = 131;                    // Set 131*256 + 11
        TMR1L = 11;                     // Set (131*256) =33547=65536-31989

                                        // 1.0003 over 32000 -> 31989
        T0IF=0;                         // TMR0 flag off
        TMR1IF=0;                       // TMR1 flag off
        TMR1IE=1;                       // TMR1  INT ENABLE
        T0IE=1;                         // TMR0 INT ENABLE
        PIE1=0b00000001;                // ADIE 0,RCIE 0,TXIE 0,SSPIE
0,CCP1IE 0,TMR2IE 0,TMR1IE 1

        timeup=0;                       // Reset timeup flag
        count_1=0;                      // Reset overtime flag

        INTCON=0b01100000;              // GIE 0,PEIE 1,T0IE 1,INTE 0,GPIE
0,T0IF 0,INTF 0,GPIF 0

        T1CON = 0b00000001;             // T1RUN 1,T1CKPS 00,T1OSCDIS,T1SYNC
1,TMR1CS 0,TMR1ON 1
        TMR0_INPUT=1;                   // RB7 INPUT enable
        }

/* Prescaler value read out by padding pulses  */
unsigned char get_pres(char pres){      // GPIO5 is switch prescaler
padding
        pre_pad=0;                      // pres:Prescaler value
        tmr0_value=TMR0;
        while (TMR0==tmr0_value){
                TOGGLE_pres=1;          // Prescaler counting up
```

```
until TMW0 increment
                TOGGLE_pres =0;
                pre_pad++;
                }
        return(pres - pre_pad );
        }

void interrupt cnt_int(void){
        GIE=0;
        if(T0IF){
                count_1++;              // Timer 0 overflow occured
                T0IF=0;
                GIE=1;
                }
        if(TMR1IF){
                TMR0_INPUT=0;           // RB7 OUTPUT TMR0 input
disable
                timeup=1;               // Gate time over

                TMR1IF =0;
                }

        }

void main(){
        unsigned char i, j, zero_sup,disp_data ;
        long decimal;

        PORTA = 0b00000000;             // PORT clear
        PORTB = 0;

        TRISA=0b11110000;               // RA0-RA3:data
RA4:T0CKI,RA5:reset,RA6,7:CLOCK

        TRISB=0b00000000;               // RB0;LEDC EN,RB2:LED
RS,RB5:MON LED,RB7:PRESCALERTOGGLE

        OPTION = 0b00110101;            // PORTB pullup,INTEDG
0,T0CS T0CKI 1,ToSE1,PSA TIMER0,1/64

        CMCON = 0x7;                    // COMPALATOR RESET
        INTCON=0;                       // INT off
        T1CON = 0;                      // Timer1 off
        PIR1 = 0b00000000;              // ADIF 0,RCIF 0,TXIF
0,SSPIF 0,CCP1IF 0,TMR2IF 0,TMR1IF 0

        MON_LED = 1;                    // Montor LED on
        _delay(2000);                   // power on timing
```

```
        lcd_init();

        lcd_goto(0);                        // select first line

#ifdef ACM0802                              // ACM0805 is 8 characters *
2 lines
        lcd_puts("10mS MHz");
#else                                       // ELSE 16 characters *
2lines
        lcd_puts("Frequency   Gate");
#endif
        lcd_goto(0x4C);
        lcd_puts("10mS");
        MON_LED = 0;                        // Monitor LED off
        lcd_goto(0x40);                     // select second line
        _delay(200);
        cnt_setup();                        // counter initial set up
        _delay(10);
        j=0;

while(1){
        GIE=1;                                      // int enable

        if(timeup){                                 // wait for gate
time up
                timeup=0;
                GIE=0;                              // int disable

                read_data=2*8192*(long)(count_1);

read_data=((long)(tmr0_value)*64)+(long)(get_pres(64))+read_data ;
                                                    //
(tmr0*64+(count_1*64*256))+prescaler
                i=0;
                decimal = 100000;
                zero_sup=1;                         // Zero suppres flag
                lcd_goto(0x41);                     // Goto 2nd line
                while(i<6){                         // 5digit
                        disp_data = ((char)((read_data/decimal)%10))
| 0x30;
                                                    // Get digit data

                        if((disp_data==0x30)&zero_sup & i==0 ){
                                disp_data=0x20; // zero suppres
                                }
                        else{ zero_sup=0;}          // zero suppress
release
```

```
                        628_cnt_pres_main
                                                //
                        lcd_putch(disp_data );  // Display digit
                        decimal = decimal / 10; // 10->1
                        i++;
                        if(i==2){lcd_putch( 0x2e);}     // Display
DP
                        }
                lcd_puts("MHz");
                for(i=126;i>0;i++){
                        _delay(100);
                }
                cnt_setup();                    // Gate open Counter
start
                j++;
                MON_LED = (j>>4)&0b1;
                }
        }
}
```