```
/**************************************************
  CPS sitch testing by PIC12F1822
   I2C LCD display and internal clock
             By nobcha all right reserved

  Ver 7.0 08/04/2011 for  PIC16F1822 CPS switch and I2C LCD
  Ver 8.0 08/18/2011 for PIC16F1822 CPS switch and ＬＥＤ SWITCH

   PIC12F1822 + LCD via I2C
   PIN Assign
                #7 LATA0:CPS plate
                #6 LATA1:Monitor LED
                #5 LATA2:Chopper switchng for LED
                #3 LATA4/SCL I2C clock for dubug
                #2 LATA5/SDA I2C data for debug

   TMR1 is counter for CPS

    OSC INT 4MHz

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

 CPS input is LA0 which is goingt to TMR1 input.
 LATA0 for DC boost switching
 LATA1 for monitor LED

  induc_cycle(on_off) : Inducter chopping 5uS on, 15uS off
    on_off:1 lighting
  dim_cycle(dim_level) : dim_level <50:turn off,
    dim_level-50= turn on cycles
  1 dim_cycle is about 1mS for CPS count cycle

  mode:0=bright, 1=blink with dimming, 2=dimming

**************************************************/
#define _XTAL_FREQ 4000000

#include <htc.h>
#include <pic.h>
#include <stdio.h>
#include "delay.h"
#include "lcd_i2c_func.h"
unsigned char Msg1[18] = "CPS         Gate";
unsigned char Msg2[5] = " 1mS";
unsigned char Msg4[4] = "cnt";


__CONFIG(
    FOSC_INTOSC & WDTE_OFF & PWRTE_ON & MCLRE_OFF & CP_OFF
    & CPD_OFF & BOREN_OFF & CLKOUTEN_OFF & IESO_OFF & FCMEN_OFF
        );

__CONFIG(
        WRT_OFF & PLLEN_OFF & STVREN_ON &  LVP_OFF
        );

unsigned char timeup , sw, sw_on=1, i;
short read_data, count_1, n_ave_freq, ave_freq=1100;

void cnt_setup(void){

        PORTA = 0b00110000;             // bit4,5:high
        TRISA = 0b00111100;             // io0:chop out,io1:mon,io2:cps,io3:in,io4:SCL,io5:SDA
        TMR1L = 0;                      // Clear Low Byte of TMR1
        TMR1H = 0;                      // Clear TMR1
        TMR1L = 0;                      // Clear TMR1

        TMR1IE=0;                       // TMR1 INT DISABLE
        TOIE=0;                         // TMR0 INT DISABLE
        PIE1=0b00000000;                // 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,TOIE 1,INTE 0,GPIE 0,TOIF 0,INTF 0,GPIF 0

        TMR1ON = 1;                     // TMR1ON 1
        CPSON = 1;                      // CPS start
}

/* Inducter short & open cycle 4MHz for 20uS  */
void    induc_cycle( char on_flag ) {
                LATA0 = on_flag;        // inducter short or open
                __delay_us( 5 ) ;       // 5uS shorting
                LATA0 = 0 ;             // inducter open
                __delay_us(12);         // 12uS open
}

/* Dimming cycle is composed on_off cycles  50times of 1mS on_off cycle */
void    dim_cycle( char dim_level ) {
                char m, on_off ;
                for (m=50;m>0;m--){             // 50 times count for dimming
                        dim_level -- ;
                        if (dim_level>50){
                                on_off = 1 ;
                            }
                        else { on_off = 0 ;}
                        induc_cycle( on_off ) ;
                        }
        }

void main(){
        unsigned char i, j, zero_sup,disp_data ;
        unsigned char dim=75, mode=0, dim_cont, blink;
        short decimal;


        PORTA = 0b00110000;                             // PORT clear
        ANSELA = 0b00000100;
        //                  ^-----------RA2 for CPS
        TRISA = 0b00111100;
        /*          ^---------------SDA
                     ^--------------SCL
                      ^-------------MCLR
                       ^------------CPS      */
        OSCCON = 0b01101010;
        /*          ^^^^------------IRCF:4MHz
                         ^----------SCS:int  */

        OPTION_REG = 0b10110001;
        /*              ^------------Weak up disable
                         ^----------TMROCS:RA2/TOCKI
                          ^---------TMROSE:H->L of RA2/TOCKI
                           ^--------PSA:assinged
                            ^^^-----PS:1:4  */

        INTCON=0;                                       // INT off
        T1CON = 0b11000000;                             // Timer1 CPS, T1CKPS:00, T1OSC DIS,T1SYNC, TMR1
off
//
        T1GCON = 0b00000000;
        PIR1 = 0b00000000;                              // ADIF 0,RCIF 0,TXIF 0,SSPIF 0,CCP1IF 0,TMR2IF
0,TMR1IF 0

        CPSCON0 = 0b10001101;                           // CPSON:1,CPSRM:low, CPSRNG:11,CPSOUT:0,TOXCS:1
        /*          ^---------------CPS on
                       ^^----------CPSRNG 11
                          ^--------TOXCS CPS  */
        CPSCON1 = 0b00000010;                           // CPS2

        PORTA = 0b00110010;
        /*          ^---------------SDA
                     ^--------------SCL
                        ^-----------LED on  */
        __delay_ms(200);
        __delay_ms(200);

        lcd_init();
```

```
        lcd_goto(0x00);                         // select first line
        lcd_str(Msg1);                          // "CPS count   Gate"

        lcd_goto(0x4C);                         // 2nd line 12th char
        lcd_str(Msg2);                          // "10mS";

        lcd_goto(0x40);                         // select second line
        __delay_us(200);

        cnt_setup();                            // counter initial set up
        __delay_us(10);

        __delay_ms(200);
        PORTA = 0b00110000;

        __delay_ms(200);
        PORTA = 0b00110010;

while(1){
        switch ( mode ) {                       // mode controling brink cycle
                case 0: blink=100;      dim=100;        // brink below 50 is meant dark
                        break;
                case 1: blink=75;                       // 75 draw 50 is 25 half lighting on
                        break;
                default: blink=100;                     // full turning on
        }

        LATA1 = 0;

        for ( i=0; i<49; i++){
                if (blink >50 ){ dim_cont=dim ; }
                else{ dim_cont=50 ; }
                for ( j=0; j<5; j++){
                        dim_cycle(dim_cont);
                }
                blink-- ;
        }
        LATA1 = 1;

        cnt_setup( ) ;                          // m-touch start
        if (blink >50 ){ dim_cont=dim; }
        else{ dim_cont=50;}                     // Goto 1mS dimmer cycle
        dim_cycle(dim_cont);


        read_data=(short)((TMR1L)+TMR1H*256) ;          // TMR1

        n_ave_freq = (ave_freq*7 + read_data )/8;       // ave_freq*15/16+read_data

        if ( n_ave_freq < 1020 )        {               // if long touching, FEQ declined below 180kHz
                sw=2;                                   // mode is changed because long touched
        }

                else if ( ( ave_freq - n_ave_freq ) > 2 ){      // if declining FRQ, sitch senced
                sw=1;
                ave_freq = n_ave_freq ;
                }
                else {                                  // not touched
                        ave_freq = n_ave_freq ;
                        sw=0;
                        sw_on=0;
                }

        if (sw==1 & sw_on==0){
                dim++;                                  // increase dim level
                sw_on=1;
        }
        if ( dim>100 ){ dim=52;}                        // dim set to under 100

        if (sw==2 & sw_on!=2){                          // if sw is long touch firstly
                mode ++;
                sw_on=2;
        }                                               //

        if ( mode>2 ){ mode=0;}
```

```
        if( RA3==0 ){                                   // debug mode

                lcd_goto(0x03);                         // select 1st line
                i=0;
                decimal = 10000;
                zero_sup=1;                             // Zero suppres flag
                while(i<5){                             // 5digit
                        disp_data = ((char)((read_data/decimal)%10)) | 0x30;
                                                        // Get digit data

                        if((disp_data==0x30)&zero_sup & i!=3 ){
                                disp_data=0x20;         // zero suppres
                                }
                        else{ zero_sup=0;}              // zero suppress release
                                                        //
                        lcd_data(disp_data );           // Display digit
                        decimal = decimal / 10;         // 10->1
                        i++;
                }
                lcd_str(Msg4);                          // "cnt"

                lcd_goto(0x043);                        // select 2nd line
                i=0;
                decimal = 10000;
                zero_sup=1;                             // Zero suppres flag
                while(i<5){                             // 5digit
                        disp_data = ((char)((ave_freq/decimal)%10)) | 0x30;
                                                        // Get digit data

                        if((disp_data==0x30)&zero_sup & i!=3 ){
                                disp_data=0x20;         // zero suppres
                                }
                        else{ zero_sup=0;}              // zero suppress release
                                                        //
                        lcd_data(disp_data );           // Display digit
                        decimal = decimal / 10;         // 10->1
                        i++;
                }

                lcd_goto(0x40);
                lcd_data(( (dim/10)%10 )| 0x30 );       // Display dimm
                lcd_data(( (dim)%10 )| 0x30 );          // Display dimm
                lcd_data(( (dim_cont/10)%10 )| 0x30 );  // Display dimm
                lcd_data(( (dim_cont)%10 )| 0x30 );     // Display dimm

                lcd_goto(0x49);                         // select second line 9th ch

                lcd_data(sw | 0x30 );                   // Display sw status
                lcd_goto(0x4B);                         // select second line 12th ch
                lcd_data(mode | 0x30 );                 // Display mode
        }

}
}
```