```
/*************************************************
  RTC clock by PIC12F1823
   LCD display and internal clock
            By nobcha all right reserved

  Ver 1.0 10/03/2011 for PIC16F1822  RTC
  Ver 1.1 10/10/2011 1Hz INT
  Ver 2.0 11/27/2011 PIC16F1823 4bits paralell LCD

   PIC12F1822 + LCD via I2C +RTC8564NB
        RC2-5:SC1602は4ビットモードとし、RC2ー5で接続します。
        RA0:LCD EN bit (enable)はRA0につなぎます。
        RA1:LCD RS bit (RS)はRA1につなぎます。
        RC1:SDA MSSP + Down_sw
        RC0:SCL MSSP
        RA3:UP_sw
        RA2:Set_sw
        RA4/CLKO: Charge pump source
        RA5: 1Hz periodical INT

        SC1602 pin connection via 4bit mode
                #1      Vdd=5V
                #2      Vss=GND
                #3      LCD contrust center of 2k VOL
                #4      RS      RA1
                #5      R/W     GND
                #6      E       RA0

                #11-14  DATA    RC2-5


     OSC INT 8MHz

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

         RTC-8564NB ADDRESS 0xA2

*************************************************/
#define _XTAL_FREQ 8000000
#define PIC_CLOCK 8000000


#include <htc.h>
#include <pic.h>
#include <stdio.h>
#include <string.h>
#include "delay.h"
```

```
#include "lcd.h"
#include "mssp_i2c.h"

#define RTC8564 0xA2                                    // RTC address
#define CTRL1 0x00                                      // RTC register
notation START/STOP
#define CTRL2 0x01
#define SECOND 0x02
#define MINUTE 0x03
#define HOUR 0x04
#define DAY 0x05
#define WEEKDAY 0x06
#define MONTH 0x07
#define YEAR 0x08
#define CKOUT 0x0D                                      // CLKOUT
#define CTRLT 0x0E                                      // TIMER CONTROL

#define second RTC_DATA[5][0]
#define minute RTC_DATA[4][0]
#define hour RTC_DATA[3][0]
#define day RTC_DATA[2][0]
#define month RTC_DATA[1][0]
#define year RTC_DATA[0][0]
#define weekday RTC_DATA[6][0]

#define Set_sw RA2
#define Down_sw RC1
#define Up_sw RA3

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

__CONFIG(
        WRT_OFF & PLLEN_OFF & STVREN_ON &  LVP_OFF
        );

unsigned char i ;
char wkdname[4];
unsigned char set_pos, clk1hzint=0;
char Msg1[9]  = "RTC TEST¥0";

unsigned char RTC_DATA[7][3]={{0x11,0x30,0x05},
                              {0x10,0x12,0x01},
                              {0x10,0x31,0x01},
                              {0x18,0x23,0x00},
                              {0x59,0x59,0x00},
```

```
                                1823_rtc_main
                                  {0x59,0x59,0x00},
                                  {0x1,0x06,0x00}};
                                                  //
year,month,day,hour,minute,second,weekday
char LCD_POS[7]={0x03,0x06,0x09,0x41,0x44,0x47,0x49};

/*******************************
 *  RTCへデータ1文字出力
 *******************************/
void rtc_write(unsigned char reg_no, unsigned char data){
        i2c_writeto(RTC8564);                   // RTCアドレスを
WRITE MODE で OPEN
        i2c_write(reg_no);                      // register select
        i2c_write(data);                        // DATA バイトを送る
        i2c_stop();                             // stop コンディショ
ン
        }

/*******************************
 *  RTCからデータ1文字取得
 *******************************/
char rtc_read(unsigned char reg_no){
        char data;
        i2c_writeto(RTC8564);                   // RTCアドレスをREAD
MODE で OPEN
        i2c_putbyte(reg_no);                    // register select
        i2c_readfrom(RTC8564);                  // RTCアドレスをREAD
MODE で REOPEN
        data=i2c_read();                        // DATA バイトを取得
する
        i2c_stop();                             // stop コンディショ
ン
        return data;
}

/*******************************
 *  LCDへBCDデータ2文字表示
 *******************************/
void lcd_hex(unsigned char bcd){
        lcd_putch(((bcd>>4)&0x0F) | 0x30);
        lcd_putch((bcd&0x0F) | 0x30);
}

/*******************************
 *  weekday数字から文字へ変換
 *******************************/
void lcd_wkd(unsigned char wkday){
        unsigned char wkdname[4]={"BAD¥x0"};
```

```
        switch(wkday){
                case 0: strcpy(wkdname,"Sun");
                                break;
                case 1: strcpy(wkdname,"Mon");
                                break;
                case 2: strcpy(wkdname,"Tue");
                                break;
                case 3: strcpy(wkdname,"Wed");
                                break;
                case 4: strcpy(wkdname,"Thu");
                                break;
                case 5: strcpy(wkdname,"Fri");
                                break;
                case 6: strcpy(wkdname,"Sat");
                                break;
                default:strcpy(wkdname,"BAD");
        }
        lcd_puts(wkdname);
}

/*********************************
 *  RTCへデータ列書き込み
 *********************************/
void rtc_writestr(void){
        rtc_write(CTRL1,0x20);                                  // Stop
        rtc_write(CTRL2,0x03);                                  // AE=1:No
Alarm INT,TIE=1 TF>0
        rtc_write(CTRLT,0x00);                                  // Timer
Control
        rtc_write(CKOUT,0x83);                                  //
7:FE=1,10:FD=11 (1Hz)
        rtc_write(YEAR,year);                                   // lower 2
year value
        rtc_write(MONTH,month);                                 // SEP
        rtc_write(DAY,day);                                             //
        rtc_write(HOUR,hour);                                   //
        rtc_write(MINUTE,minute);                               //
        rtc_write(SECOND,second);                               //

        rtc_write(WEEKDAY,weekday);
//      rtc_write(CTRL1,0x00);                                  // Start
}

/*********************************
 *  RTCからデータWORKへ取り込み
 *********************************/
void rtc_readstr(void){
//      rtc_write(CTRL1,0x20);                                  //
```

```
        year=rtc_read(YEAR);                                    // lower 2
year value
        month=rtc_read(MONTH)&0x1F;                             //
        day=rtc_read(DAY)&0x3F;                                 //
        hour=rtc_read(HOUR)&0x3F;                               //
        minute=rtc_read(MINUTE)&0x7F;                           //
        second=rtc_read(SECOND)&0x7F;                           //
        weekday=rtc_read(WEEKDAY)&0x07;                         //
        rtc_write(CTRL1,0x00);                                  // Start
}

/*******************************
 *  RTCデータをLCDへ書き込み
 *******************************/
void rtc_disp(void){
        lcd_goto(0x00);                                         // 1st line
        lcd_hex(0x20);                                          //
20**/**/**
        lcd_hex(year);
        lcd_putch(0x2F);
        lcd_hex(month);
        lcd_putch(0x2F);
        lcd_hex(day);

        lcd_goto(0x40);                                         // 2nd line
        lcd_hex(hour);                                          // **:**:**
BAD
        lcd_putch(0x3A);
        lcd_hex(minute);
        lcd_putch(0x3A);
        lcd_hex(second);
        lcd_putch(0x20);
        lcd_wkd(weekday);
}

// 2*BCD to Bynary
unsigned char bctob(unsigned char b2){
        unsigned char b3;
        b3=((b2>>4)*10)+(b2&0x0f);
        return (b3);
}

// Bynary to 2*BCD
unsigned char btobc(unsigned char b2){
        unsigned char b3;
        b3=((b2/10)&0x0f)<<4 | (b2%10)&0xf ;
        return (b3);
}
```

```
// data increment and MAX check
void data_inc(unsigned char set_pos ){
        unsigned char bynary;
        bynary=bctob(RTC_DATA[set_pos][0]);
        bynary++;
        RTC_DATA[set_pos][0]=btobc(bynary);
        if(RTC_DATA[set_pos][0]>(RTC_DATA[set_pos][1])){
                RTC_DATA[set_pos][0]=RTC_DATA[set_pos][2];
        }
}

// data decrement and 0 check
void data_dec(unsigned char set_pos){
        unsigned char bynary;
        if(RTC_DATA[set_pos][0]==RTC_DATA[set_pos][2]){
                RTC_DATA[set_pos][0]=RTC_DATA[set_pos][1]+1;
        }
        bynary=bctob(RTC_DATA[set_pos][0]);
        bynary--;
        RTC_DATA[set_pos][0]=btobc(bynary);
}

/*****************************************
* MSSP initialize
*****************************************/
void mssp_init(void){
       /* SSP1CON1 REGISTERS */
       SSPEN      = 1;   //Enables Serial Port Mode
       SSPM3      = 1;   ////////
       SSPM2      = 0;   //I2C Master Mode
       SSPM1      = 0;   // clock= Fosc/(4*(SSP1ADD+1))
       SSPM0      = 0;   ///////

       /* SSPCON2 REGISTERS */
       SSP1CON2   = 0x00;
       /* SSPCON3 REGISTERS */
       SSP1CON3   = 0x00;


       /* SSP1STAT REGISTERS */
       SMP        = 1;   //SPI MASTER MODE
       CKE        = 0;   //SMBus Specific Inputs Enabled

       //SSP1ADD    = 0x19;      //~75kHz
       //SSP1ADD    = 0x13;      //~100kHz
       //SSP1ADD    = 0x07;      //~400kHz
       SSP1ADD    = 0x50;
```

```
}

// Interrupt routine
void interrupt clk1hz(void){
        GIE=0;
        if ( IOCIF )  {
                IOCIF = 0;                                      // interrupt
on change
                IOCAF0 = 0;
                clk1hzint=1;                                    // 1Hz int
flag on
        }
}

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

        PORTA = 0b00000000;                                     // PORT
clear
        ANSELA = 0b00000000;                                    // all
digital
        //
        TRISA = 0b00111100;
        /*               ^---------------EN
                 ^---------------RS
                   ^---------------SEL_sw
                  ^---------------UP_sw
                 ^---------------CLOCK OUT
               ^---------------1Hz                        */
        PORTC = 0b00000000;                                     // PORT
clear
        ANSELC = 0b00000000;                                    // all
digital
        //
        TRISC = 0b00000011;
        /*               ^---------------SCL
                 ^---------------SDA+DOWN_sw
                   ^---------------DB4
                  ^---------------DB5
                 ^---------------DB6
               ^---------------DB7                      */



        OSCCON = 0b01110000;
        /*           ^^^^----------IRCF:8MHz
                      ^----------SCS:int  */
```

```
        OPTION_REG = 0b00000000;
        /*              ^------------Weak up disable
                       ^-----------TMR0CS:RA2/T0CKI
                        ^----------TMR0SE:H->L of RA2/T0CKI
                         ^---------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
        CM1CON0 = 0x00000111;
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);                    /*  これだけ待っても立ち上げ
不良になることある。再SW__ON  */
        __delay_ms(200);                    /*  LCD低電圧駆動ではかなり
の立ち上げ余裕必要  */

        lcd_init();                         // LCD initialize

        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        lcd_goto(0x10);                     // select first line
        __delay_ms(200);
        lcd_puts(Msg1);                     // No display for SC1602

        mssp_init();
        IOCAP = 0b000000001;                // Int on change RA0
positive
        IOCAF = 0b000000000;
        INTCON = 0b10001000;                // GIE:1, IOCIE:1

        rtc_readstr();                      //  get RTC data
        rtc_disp();                         //  display rtc status

        rtc_writestr();                     //  set RTC > 1Hz INT start

while(1){

        if(Set_sw==0){                                  // if RA3==0
```

```
Set mode
                __delay_ms(100);
                if(Set_sw==0){
                        set_pos=0;                      //
YEAR->Month->day->hour->minuts->second

                        rtc_readstr();                  // get RTC
data
                        while(set_pos<7){               //
set_pos:5->0
                                lcd_goto(LCD_POS[set_pos]);
                                __delay_ms(100);
                                while((PORTA&0x38)==0x38){}     //
Wait Switch on
                                __delay_ms(100);        // Wait a
time
                                if(Up_sw==0){ data_inc(set_pos);}
                                if(Down_sw==0){ data_dec(set_pos);}
                                if(Set_sw==0){  set_pos++;}
                                rtc_disp();             //  display
rtc scratch status
                        }

                }
                rtc_writestr();                         //  set RTC
                __delay_ms(100);
        }
        if(clk1hzint==1){
                clk1hzint=0;                            //  flag
clear
                rtc_readstr();                          //  get RTC
data
                rtc_disp();                             //  display
rtc status
        }

}
}
```