```
/************************************************
  RTC clock by PIC12F1827
   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
  Ver 3.0 9/5/2012  PIC16F1827 4bits paralell LCD

   PIC12F1827 + LCD +RTC8564NB
        RA0-3:SC1602は4ビットモードとし、RA0－3で接続します。
        RB6:LCD EN bit (enable)はRB6につなぎます。
        RB7:LCD RS bit (RS)はRB7につなぎます。
        RB1:SDA1 MSSP1
        RB4:SCL1 MSSP1
        RB2:UP_sw
        RB3:Down_sw
        RA5:Set_sw
        RA6/CLKO: Charge pump source
        RB0: 1Hz periodical INT
        RA7:Heart beat  LED

        SC1602 pin connection via 4bit mode
                #1      Vdd=5V
                #2      Vss=GND
                #3      LCD contrust center of 2k VOL
                #4      RS      RB7
                #5      R/W     GND
                #6      EN      RB6

                #11-14  DATA    RA0-3

     OSC INT 8MHz

         Development Circumstance
         MPLAB IDE V8.85   HiTECH C V9.83

         RTC-8564NB ADDRESS 0xA2

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

#include <htc.h>
#include <stdio.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 RA5
#define Down_sw RB3
#define Up_sw RB2
#define Heart_beat LATA7
```

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

// Prototyping functions
void rtc_write(unsigned char, unsigned char);
char rtc_read(unsigned char );
void lcd_hex(unsigned char );
void lcd_wkd(unsigned char);
void rtc_writestr(void);
void rtc_readstr(void);
void rtc_disp(void);
unsigned char bctob(unsigned char );
unsigned char btobc(unsigned char );
void data_inc(unsigned char );
void data_dec(unsigned char );
void mssp_init(void);
void interrupt clk1hz(void);

// ---------------------------------------

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

        PORTA = 0b10000000;             // PORT clear
        ANSELA = 0b00000000;            // all digital
        //
        TRISA = 0b01110000;
        /*        |      ^--------------DB4
                  |     ^---------------DB5
                  |    ^----------------DB6
                  |   ^-----------------DB7
                  |  ^------------------COUNTER IN
                  | ^-------------------SEL_sw
                  |^--------------------CLOCK OUT
                  ^---------------------heart Beat
                                          */
        PORTB = 0b00000000;             // PORT clear
        ANSELB = 0b00000000;            // all digital
        //
        TRISB = 0b00011111;
        /*        |      ^--------------1hz INT
                  |     ^---------------SDA
                  |    ^----------------UP_sw
                  |   ^-----------------Down_sw
                  |  ^------------------SCL
                  | ^-------------------GATE
                  |^--------------------EN
                  ^---------------------RS
                                       */
//
        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;
        CM2CON0 = 0x00000111;
//      __delay_ms(200);
//      __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __delay_ms(200);
        __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(0x00);                 // select first line
        lcd_puts(Msg1);                 // Display for SC1602

        mssp_init();
        lcd_putch(0x40);
        rtc_readstr();                  //  get RTC data
        if(RTC_DATA[0][0]==0){          //  if no back up
                RTC_DATA[0][0]=0x12;    //  set default value
                RTC_DATA[1][0]=0x09;
                RTC_DATA[2][0]=0x05;
        }
        lcd_putch(0x40);
        rtc_disp();                     //  display rtc status

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

        INTCON = 0b10010000;            // GIE:1, INTE:1

while(1){

        if(Set_sw==0){                          // if RA3==0 Set mode
                __delay_ms(150);
                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(150);
                                while(Up_sw&Down_sw&Set_sw){}   // Wait Switch on
                                __delay_ms(150);                // 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
                Heart_beat ^= 1;                //  Toggle LED
                GIE=1;
```

```
                INTE=1;
        }

}
}

/*******************************
 *  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LCD表示
 *******************************/
void lcd_wkd(unsigned char wkday){
        switch(wkday){
                case 0: lcd_puts("Sun");
                                break;
                case 1: lcd_puts("Mon");
                                break;
                case 2: lcd_puts("Tue");
                                break;
                case 3: lcd_puts("Wed");
                                break;
                case 4: lcd_puts("Thu");
                                break;
                case 5: lcd_puts("Fri");
                                break;
                case 6: lcd_puts("Sat");
                                break;
                default:lcd_puts("BAD");
        }
}

/*******************************
 *  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){
        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);
        lcd_putch(0x3A);
        lcd_hex(minute);
        lcd_putch(0x3A);
        lcd_hex(second);
        lcd_putch(0x20);
        lcd_wkd(weekday);
}

// 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=(RTC_DATA[set_pos][0]&0xf0)/16*10+(RTC_DATA[set_pos][0]&0x0f);
        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=(RTC_DATA[set_pos][0]&0xf0)/16*10+(RTC_DATA[set_pos][0]&0x0f);
        bynary--;
        RTC_DATA[set_pos][0]=btobc(bynary);
}

/*************************************
* MSSP initialize
*************************************/
void mssp_init(void){
       /* SSP1CON1 REGISTERS */
```

```
    SSP1CON1bits.SSPEN      = 1;   //Enables Serial Port Mode
    SSP1CON1bits.SSPM3      = 1;   ////////
    SSP1CON1bits.SSPM2      = 0;   //I2C Master Mode
    SSP1CON1bits.SSPM1      = 0;   // clock= Fosc/(4*(SSP1ADD+1))
    SSP1CON1bits.SSPM0      = 0;   ////////

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


    /* SSP1STAT REGISTERS */
    SSP1STATbits.SMP       = 1;   //SPI MASTER MODE
    SSP1STATbits.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 ( INTF )  {
            INTF = 0;                       // interrupt on change
            clk1hzint=1;                    // 1Hz int flag on
    }
}
```