```
/************************************************
RTC clock with thermo & moisture meter by PIC12F1829
  LCD display and serial output with internal clock
          By nobcha all right reserved

  Reffer to  05/22/2014 PIC16F1827+RTC8564NB
                        + SHT-11 + 9600bps Serial
  Reffer to  11/01/2014 PIC16F1827 + RTC8583 + sw

  11/02/2014 PIC16F1829 + LCD + RTC8583 + SHT-11

-------------------------------------------
        RA0: LCD RS bit                 0
        RA1: LCD EN bit                 0
        RA2: Clock out from INT 1
        RA3: SW1 SET_sw                 1
        RA4: SW2 UP_sw                  1
        RA5: Mon LED                    0
        RB4: SDA                        1
        RB5: UART RX                    1
        RB6: SCL                        1
        RB7: UART TX                    0
        RC0: SHT-11 SCK                         1
        RC1: SHT-11 DATA pulling up by 103      1
        RC2-5: SC1602は4ビットモードで接続      0
        RC6: CCCP4:Charge pump source           0
        RC7: SW3 Down SW                        1
-----------------------------------------------

        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 16MHz

         Development Circumstance
         MPLAB IDE V8.86   HiTECH C V9.83
************************************************/
#define _XTAL_FREQ 16000000
#define PIC_CLOCK 16000000

#include <htc.h>
#include <stdio.h>
#include "lcd.h"
#include "mssp_i2c.h"
#include "sht.h"
```

```
#include "uart.h"

//-----------------------------------
#define RTC8583 0xA0                    // RTC address
#define CTRL 0x00                       // RTC register notation START/STOP
#define SECOND 0x02
#define MINUTE 0x03
#define HOUR 0x04
#define DAY 0x05
#define MONTH 0x06
#define TIMER 0x07                      // 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 RA3
#define Down_sw RC7
#define Up_sw RA4
//-----------------------------------

#define Heart_beat LATA5

_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
        );

// Grobal variables & data
unsigned char i ;
char wkdname[4];
unsigned char set_pos, clk1hzint=0, PBuf=0;
char Msg1[]  = "RTC+RT+RH SER";
char Str[8];

//--------------------
// Default/Max/Min table
unsigned char RTC_DATA[7][3]={{0x14,0x30,0x05},       // Year->Day
```

```
                        {0x12,0x12,0x01},   // month:6 stacked weekday
                        {0x31,0x31,0x01},   // day:5 stacked year
                        {0x23,0x23,0x00},   // hour:4 24hour:0
                        {0x59,0x59,0x00},   // minute:3
                        {0x50,0x59,0x0},    // second:2
                        {0x0,0x6,0x0}};     // Weekday->Month
                                            // Default,Max,Min

// LCD cursor position
// ..._.._.._  2014.10.12
// ._.._.._.._ 23:59:59Sun
char LCD_POS[7]={0x03,0x06,0x09,0x41,0x44,0x47,0x49};
//-----------------------

unsigned char i, j, zero_sup,disp_data ;
unsigned char bdat_L,bdat_H, bdat_CRC;
int ADValHR ,ValHR10,ADValTmp,wdat ;

// 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);
void putch(unsigned char);

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

// main
void main(){

//-------------------------------
        PORTA = 0b00100000;             // PORT clear
        ANSELA = 0b00000000;            // all digital
        //
        TRISA = 0b00011100;
/*              RA0: LCD RS bit     0
                RA1: LCD EN bit     0
```

```
                RA2: Clock out from INT 1
                RA3: SW1 SET_sw / reset 1
                RA4: SW2 UP_sw          1
                RA5: Mon LED            0
*/
        PORTB = 0b00000000;                     // PORT clear
        ANSELB = 0b00000000;                    // all digital
        //
        TRISB = 0b01110000;
/*      RB4: SDA                1
                RB5: UART RX            1
                RB6: SCL                1
                RB7: UART TX            0
                                                        */

        LATC =  0b00000000;                     // LATC RESET
        TRISC = 0b10000010;                     //

/*      RC0: SHT-11 SCK                                 0
                RC1: SHT-11 DATA pulling up by 103  1
                RC2-5: SC1602は4ビットモードで接続              0
                RC6: CCCP4:Charge pump source       0
                RC7: SW3 Down SW                                1

                                                        */
        ANSELC = 0b00000000;                    // all digital
//-----------------------------------------------
//
        OSCCON = 0b01111000;
/*                      | ^^^^-------------IRCF16MHz
                      |^-----------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 = 0x00000000;
        CM2CON0 = 0x00000000;
//      _delay_ms(200);
//      _delay_ms(200);
        _delay_ms(200);
        _delay_ms(200);

//CCP4 PWM initializing (83.8KHz on RB3 @16MHz)
        CCP4CON = 0b00001111;           // use PWM mode
        CCPR4L = 0x0c;                  // duty is 50%
        CCPTMRS = 0;                    // Select TMR2
//  TMR2 initilizing
        T2CON = 0b00000100;             // POSTSCALE 1:1 ,TMR2 ON ,
PRESCALE 1:1
        _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
        _delay_ms(20);
        mssp_init();
        SHTInit();

        init_UART();

        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
// Humidity reading
        SHTTSSeq();                             // TSシーケンス発行
        SHTWrite(0x05);                         // measure humidity コマンド送信

// 変換完了待ち
                while((SHT_PORT & (1<<bitDATA))) {} // A-D変換完了を待つ

                bdat_H = SHTRead(0);            // read high byte(ACK)
                bdat_L = SHTRead(0);            // read low byte(ACK)
                bdat_CRC = SHTRead(1);          // read CRC(NOACK)

                ADValHR = ((int)bdat_H << 8) | bdat_L;  // 16ビットに合成
                ValHR10 = CalcHR10(ADValHR);        // 工学数値へ変換

                sprintf(Str,"%d.%d%%",ValHR10/10,ValHR10%10);       // 文字列を作成

                lcd_goto(0x4B);
                lcd_puts(Str);                  // LCDへ表示
```

```
// Temperature reading
        SHTTSSeq();                             // TSシーケンス発行
        SHTWrite(0x03);                         // measure Temperature コマンド送信

// 変換完了待ち
                while((SHT_PORT & (1<<bitDATA))) {} // A-D変換完了を待つ

                bdat_H = SHTRead(0);            // read high byte(ACK)
                bdat_L = SHTRead(0);            // read low byte(ACK)
                bdat_CRC = SHTRead(1);          // read CRC(NOACK)

                ADValTmp = ((int)bdat_H << 8) | bdat_L;  // 16ビットに合成
                wdat = CalcTMP10() ;
        sprintf(Str,"%d.%d°C",wdat/10,wdat%10);          // 文字列を作成

                lcd_goto(0xA);
                lcd_puts(Str);                  // LCDへ表示

// Serial output
                printf("DATE:%c%c/%c%c/%c%c:%c%c:%c%c#",(month>>4)&0x0F|
0x30,month&0x0F|0x30,(day>>4)&0x0F|0x30,day&0x0F|0x30,(hour>>4)&0x0F|0x30,hour&
0x0F|0x30,(minute>>4)&0x0F|0x30,minute&0x0F|0x30,(second>>4)&0x0F|0x30,second&
0x0F|0x30);
                if(wdat>0) printf("RT=%d.%ddeg",wdat/10,wdat%10);
                else{
                        wdat=-wdat;
                        printf("RT=-%d.%d%%",wdat/10,wdat%10);
                }
                printf("RH=%d.%d%%¥n¥r",ValHR10/10,ValHR10%10);

                Heart_beat ^= 1;                //  Toggle LED

// INT release
                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(RTC8583);                   // 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(RTC8583);                   // RTCアドレスをREAD MODE で OPEN
        i2c_putbyte(reg_no);                    // register select
        i2c_readfrom(RTC8583);                  // 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);
}

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

/*********************************
 * 2*BCD to Binary
 *********************************/
unsigned char bctob(unsigned char b1){
        unsigned char b0;
        b0=(b1&0xf) + ((b1>>4)&0x0f)*10;
        return (b0);
}

/*********************************
 * 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(CTRL,0x80);                   // Stop
        day = (day&0x3F) | (( bctob(year)&0x3)<<6);     // Stacking year on day
        month = month | (weekday<<5);// Stacking weekday on month
        rtc_write(MONTH,month);                 //
        rtc_write(DAY,day);                     //
        rtc_write(HOUR,hour);                   //
        rtc_write(MINUTE,minute);               //
        rtc_write(SECOND,second);               //
        rtc_write(CTRL,0x00);                   // Start
}

/********************************
 * RTCからデータWORKへ取り込み
 ********************************/
void rtc_readstr(void){
        rtc_write(CTRL,0x80);                   // Stop

        month=rtc_read(MONTH);                  // w2w1w0m10 m4m2m1m0
        day=rtc_read(DAY);                      // y1y0d11d10 d4d2d1d0
        hour=rtc_read(HOUR);                    // 24_0_h11h10 h4h2h1h0
        minute=rtc_read(MINUTE);                // m14m12m11m10 m4m2m1m0
```

```
        second=rtc_read(SECOND);            // s14s12s11s10 s4s2s1s0
        weekday=(month&0xE0)>>5;            //

        if(((((rtc_read(DAY)>>6))&0x3)==0)&((day&0x3F)==1)&((month&0x1F)==1)&(hour==
0)&(minute==0)&(second==0))
                {year=btobc((bctob(year)&0xFC)+4)+(rtc_read(DAY)>>6);}      // if year
count up

        else {year=btobc((bctob(year)&0xFC)+(rtc_read(DAY)>>6));}
                                            // lower 2 year value
        rtc_write(CTRL,0x00);               // Start
        day=day&0x3F;                       // strip year data for INC/DEC
        month=month&0x1F;                   // strip week data for INC/DEC
}

/*******************************
 * RTCデータをLCDへ書き込み
 *******************************/
void rtc_disp(void){

        lcd_goto(0x00);                     // 1st line

        lcd_hex(year);                      // **/**/**
        lcd_putch(0x2F);
        lcd_hex(month);
        lcd_putch(0x2F);
        lcd_hex(day);
        lcd_putch(0x20);

        lcd_goto(0x40);                     // 2nd line

        lcd_hex(hour);

        lcd_putch(0x3A);
        lcd_hex(minute);
        lcd_putch(0x3A);
        lcd_hex(second);
        lcd_wkd(weekday);
}

// 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;
}

// atandard serial I/O
void     putch(unsigned char c ){
```

```
        put_UART(c);
        return;
}

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