```
/*************************************************************
 Target MCU & clock speed: ATmega328P @ 1Mhz internal
 Name    : utils.c
 Author  : Insoo Kim (insoo@hotmail.com)
 Created : May 17, 2015
 Updated : May 17, 2015

 Description: Get system compile time & date and display on LCD 2*16
    Button toggling to turn on or off the backlight of LCD

 HEX size[Byte]:

 Ref:
    Donald Weiman    (weimandn@alfredstate.edu)
    http://web.alfredstate.edu/weimandn/programming/lcd/ATmega328/
      LCD_code_gcc_4d.html
 *************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/wdt.h>
#include <avr/sleep.h>
#include "externs.h"
#include "defines.h"
#include <util/delay.h>


//-----------------------------------
void sysClockTest()
{
    PORTB |= _BV(debug_PIN);
    _delay_ms(10);
    PORTB &= ~_BV(debug_PIN);
    _delay_ms(10);

}//sysClockTest

//-----------------------------------
void countButton()
{
    uint8_t menuCnt=0;
    uint8_t prevLoop=0, curLoop=0, lapse=0;
    uint8_t loopCnt=0;

    uint8_t val;
    uint8_t DONE=0;


    loopCnt=0;
    prevLoop=0;
    //visual cue to show being ready to get user input of menuCnt
    //blinkLED(1);
    //Get menuCnt by counting the button press
```

```
    //If pressing the button within 1 second of interval between each press,
    //  it will be accumulated as "menuCnt".
    //If the interval is over 1 sec, which is menuSelectInterval,
    //  then DONE is set to 1.
    while (!DONE)
    {
        loopCnt++;
        curLoop = loopCnt;
        lapse = curLoop - prevLoop;

        //menuSelectInterval is
        //  multiple of _delay_ms(halfSec/4)
        //  which is the unit delay of each while loop
        if (lapse > menuSelectInterval)
        {
            if (menuCnt != 0)
                DONE = 1;
            else
                //wait another 1 sec to get user's menuCnt
                ;
        }
        //if user has not chosen menuCnt > 0, and time lapse over 3sec,
        //forget about it, and call DONE as 2, and eventually go to sleep.
        if ((lapse > menuSelectInterval*2) && (menuCnt ==0))
            DONE = 2;

      //check if tactile_Switch pressed to 0
      //AVR equivalent to Arduino digitalRead(tactile_Switch_bit)
      val = tactile_Switch_port & _BV(tactile_Switch_bit);
      if (val == 0)
      {
        // for debounce
        _delay_ms(halfSec/2);
        menuCnt++;

        //Pressing the button, lap time calculation should be reset
        //to give 1 sec of time to choose menuCnt
        prevLoop = loopCnt;
      }//if(val == 0)

        // should be fast enough to catch button press frequency
        // menuSelectInterval is a multiple of times of _delay_ms(halfSec/4);
      _delay_ms(halfSec/4);
  }//while(!DONE)

    //menuCnt has been set within 3sec of a PCINT occurence
    //  then, play WDT count for a corresponding alarm period.
    if (DONE)
    {
        if (DONE == 2)
            menuCnt = prevMenuCnt;

        //visual cue to notifiy user selected menuCnt
        //blinkLED(menuCnt);
        //_delay_ms(halfSec);
        //turnOnLCDpower();
```

```
        //lcd_dispON();
        prevMenuCnt = menuCnt;
        switch (menuCnt)
        {
            case 1:
                //lcd_dispWords();
                lcd_dispRealClock();
                _delay_ms(1000);
                break;
            case 2:
                _delay_ms(200);
                lcd_showDHT11();
                _delay_ms(2000);
                break;
            case 3:
                turnOnLCDBacklight();
                lcd_dispRealClock();
                _delay_ms(2000);
                turnOffLCDBacklight();
                break;
            case 4:
                //adjustClock();
                adjustMin();
                adjustHour();
                adjustSec();
                lcd_dispRealClock();
                _delay_ms(1000);
                break;
            case 5:
                lcd_dispAccumulatedTime();
                _delay_ms(2000);
                break;
            case 6:
                lcd_dispProgInfo();
                break;
            default:
                lcd_dispMenu();
        }//switch (menuCnt)
        //lcd_dispOFF();
        //turnOffLCDpower();

        //reset menuCnt
        menuCnt=0;

        //Enable watchdog timer interrupts
        // and begin counting for alarm
        //WDTCR |= _BV(WDTIE);
    }//if (DONE == 1)

}//countButton

void adjustHour()
{
    uint8_t DONE=0, val;
    uint8_t curLoop=0, preLoop=0, lapse=0;
    char strHour[3];
```

```
    while (!DONE)
    {
    // set cursor to start of first line
        lcd_write_instruction_4d(lcd_SetCursor | lcd_LineOne);
        _delay_us(DELAY_INST);                                  // 40 uS delay     ↵
          (min)
    // display the first line of information
        lcd_write_string_4d((uint8_t *)"Hour: ");
        itoa(hour, strHour, 10);
        lcd_write_string_4d((uint8_t *)strHour);
        lcd_write_string_4d((uint8_t *)" ");
        val = tactile_Switch_port & _BV(tactile_Switch_bit);
        _delay_ms(100);
        if (val == 0)
        {
            hour++;
            if (hour > 23)
                hour=0;
            preLoop = curLoop;
        }
        curLoop++;
        lapse = curLoop - preLoop;
        if (lapse > adjustTimeInterval)
            DONE = 1;
        _delay_ms(halfSec/4);
    }//while (!DONE)
}//adjustHour

void adjustMin()
{
    uint8_t DONE=0, val;
    uint8_t curLoop=0, preLoop=0, lapse=0;
    char strMin[3];

    while (!DONE)
    {
    // set cursor to start of first line
        lcd_write_instruction_4d(lcd_SetCursor | lcd_LineOne);
        _delay_us(DELAY_INST);                                  // 40 uS delay     ↵
          (min)
    // display the first line of information
        lcd_write_string_4d((uint8_t *)"Min: ");
        itoa(min, strMin, 10);
        lcd_write_string_4d((uint8_t *)strMin);
        lcd_write_string_4d((uint8_t *)" ");
        val = tactile_Switch_port & _BV(tactile_Switch_bit);
        _delay_ms(100);
        if (val == 0)
        {
            min++;
            if (min > 59)
                min=0;
            preLoop = curLoop;
        }
        curLoop++;
```

```
        lapse = curLoop - preLoop;
        if (lapse > adjustTimeInterval)
            DONE = 1;
        _delay_ms(halfSec/4);
    }//while (!DONE)

}//adjustMin

void adjustSec()
{
    uint8_t DONE=0, val;
    uint8_t curLoop=0, preLoop=0, lapse=0;
    char strSec[3];

    while (!DONE)
    {
    // set cursor to start of first line
        lcd_write_instruction_4d(lcd_SetCursor | lcd_LineOne);
        _delay_us(DELAY_INST);                                  // 40 uS delay     ↵
          (sec)
    // display the first line of information
        lcd_write_string_4d((uint8_t *)"sec: ");
        itoa(sec, strSec, 10);
        lcd_write_string_4d((uint8_t *)strSec);
        lcd_write_string_4d((uint8_t *)" ");
        val = tactile_Switch_port & _BV(tactile_Switch_bit);
        _delay_ms(100);
        if (val == 0)
        {
            sec++;
            if (sec > 59)
                sec=0;
            preLoop = curLoop;
        }
        curLoop++;
        lapse = curLoop - preLoop;
        if (lapse > adjustTimeInterval)
            DONE = 1;
        _delay_ms(halfSec/4);
    }//while (!DONE)

}//adjustSec
```