Programkod i C

```
/*
 * GccApplication1.c
 *
 * Created: 2018-04-12 09:07:28
 * Author : ine14hab
 */

#define F_CPU 8000000UL
#define F_SCL 100000UL


//LCD macros

#define LCD_ENABLE 1
#define LCD_RW 6
#define LCD_RS 4
#define LCD_DELAY 50

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdio.h>
#include <util/twi.h>
#include <string.h>

#include "rtc_mcp7940m.h"
#include "i2c_master.h"



//ATTRIBUTES

int tempIn;
int tempOut;

int maxTempIn;
int minTempIn;
int averageTempIn;

int maxTempOut;
int minTempOut;
int averageTempOut;

int tempVecIn[100]; // annan typ av deklaration i c tror jag...
int tempVecOut[100];
int tempIndexIn; // håller reda på om hur pass fylld vektorn är
int tempIndexOut; // håller reda på om hur pass fylld vektorn är

int outOrIn; //håller reda på om vi ska mäta ute- eller innetemperatur
int minOrMax; //håller reda på om vi ska uppdatera värde på max eller
min
int interTemp;
int change;

int MAX = 0; // godtyckliga värden
```

```
int MIN = 250;

volatile char val;
volatile char knappflagg = 0;
volatile char ticked = 0;

volatile unsigned char albin;
volatile int albin1;

uint8_t set_sec = 0;
uint8_t set_min = 0;
uint8_t set_h = 12;
uint8_t set_day = 5;
uint8_t set_date = 17;
uint8_t set_month = 5;
uint8_t set_year = 18;

uint8_t get_sec;
uint8_t get_min;
uint8_t get_h;
uint8_t get_day;
uint8_t get_date;
uint8_t get_month;
uint8_t get_year;


//METHODS
void initDiods(void);

void displayChar(char);
void displayString(char[]);
void initDisplay(void);
void displayOnOff(void);
void configureDisplay(void);
void clearDisplay(void);
void setRow(char);

void initThermal(void);
void print(int);
void printTemp(int);
void print_calc_time(uint8_t);
int calcTemp(int);
void warningLow(void);
void warningHigh(void);
void maxTempSetUp(void);
void minTempSetUp(void);
void increaseMaxTemp(void);
void decreaseMaxTemp(void);
void increaseMinTemp(void);
void decreaseMinTemp(void);
void clearVectors(void);
void insertTempIn(int);
void insertTempOut(int);
void printAverageTempIn(void);
void printAverageTempOut(void);

void enable_interupt(void);
void set_up_tick(void);
```

```
void setUpKeyboard(void);
void buttonPressed(void);

void printTime(void);


//DIODS

void initDiods() {
          DDRC |= (1<<PC6);
          DDRC |= (1<<PC7);
}

// DISPLAY

//Sätter DDR (Data Direction Registers) för att kontrollera displayen
void initDisplay() {
          DDRD |= (1<<LCD_ENABLE)|(1<<LCD_RW)|(1<<LCD_RS);
          DDRB = 0xFF;
}

//Konfigurerar displayen
void configureDisplay()
{
          _delay_ms(LCD_DELAY);
          PORTD |= (1<<LCD_ENABLE);
          PORTB = 0x3F;
          PORTD &= ~(1<<LCD_ENABLE);
          PORTD |= (1<<LCD_ENABLE);
}

//Sätter på displayen
void displayOnOff()
{
          _delay_ms(LCD_DELAY);
          PORTD |= (1<<LCD_ENABLE);
          PORTB = 0x0F;
          PORTD &= ~(1<<LCD_ENABLE);
          PORTD |= (1<<LCD_ENABLE);

}

//Rensar display på tecken, se "ClearDisplay"
void clearDisplay()
{
          _delay_ms(LCD_DELAY);
          PORTD |= (1<<LCD_ENABLE);
          PORTB = 0x01;
          PORTD &=~ (1<<LCD_ENABLE);
          PORTD |= (1<<LCD_ENABLE);

}

//Skriver siffror och bokstäver på displayen
void displayChar(char c)
{
          _delay_ms(10);
```

```
            PORTD |= (1<<LCD_RS); //E och RS sätts till 1, övriga 0
            PORTD &= ~(1 << LCD_RW); // write mode
            PORTD  |= (1<<LCD_ENABLE);
            PORTB = c;
            PORTD &=~ (1<<LCD_ENABLE);
            PORTD &= ~(1<<LCD_RS);
            PORTD |= (1<<LCD_ENABLE);
            _delay_ms(10);
}

//Skriver meningar på displayen
void displayString(char str[]) {

             uint8_t i = 0;

             while(str[i] != '\0') {
                        displayChar(str[i]);
                        i++;
             }
}

//Bestämmer vilken rad vi ska skriva på
void setRow(char row)
{
            PORTD |= (1<<LCD_ENABLE);
            PORTB = row;
            PORTD &=~ (1<<LCD_ENABLE);
            PORTD |= (1<<LCD_ENABLE);
            _delay_ms(LCD_DELAY);
}


// TEMPERATURE

//Initierar temp sensorer
void initThermal() //se sida 202 i handbok
{
            ADMUX |= 0b01000000;
            ADMUX &=~ (1<<ADLAR);
            ADCSRA |=
(1<<ADEN)|(1<<ADIE)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);
            ADCSRA |=(1<<ADSC);
}

// Interrup för temperatursensor
ISR(ADC_vect)
{
            //int albin;
            if((interTemp > 0)) { //kanal 0
                        albin=0;
                        albin = ADCL;
                        tempIn = (albin-140)/2.75;
                        if (tempIn > MAX)
                        {
                                    MAX = tempIn;
                        }
                        if (tempIn < MIN)
                        {
```

```
                                MIN = tempIn;
                        }
                        ADMUX |= 0b01000001;
                        interTemp = 0;

            } else { //kanal 1
                        albin1 = ADCL;
                        tempOut = (albin1-140)/2.75;
                         if (tempOut > MAX)
                         {
                                  MAX = tempOut;
                         }
                         if (tempOut < MIN)
                         {
                                  MIN = tempOut;
                         }
                        ADMUX |= 0b01000000;
                        interTemp = 1;
            }
            ADCSRA |= (1<<ADSC); //starta A/D
}

//Skriver ut info om temp, så som varningar etc
void print(int k)
{
            if(outOrIn >0)//Inomhus
            {
                        if(k>maxTempIn) // Varning, överskrider högsta
temepratur
                        {
                                  printTemp(k);
                                  _delay_ms(1000);
                                  displayChar(' ');
                                  warningHigh();
                        }
                        if(k < minTempIn) // Varning, underskriver minsta
temperatur
                        {
                                  printTemp(k);
                                  _delay_ms(1000);
                                  displayChar(' ');
                                  warningLow();
                        } else {
                                  printTemp(k); //ingen varning
                        }
            } else //Utomhus
            {
                        if(k>maxTempOut) //Varning, överskrider högsta
temperatur
                        {
                                  printTemp(k);
                                  _delay_ms(1000);
                                  displayChar(' ');
                                  warningHigh();
                        }
                        if(k < minTempIn) // Varning, underskriver minsta
temperatur
                        {
```

```
                                printTemp(k);
                                _delay_ms(1000);
                                displayChar(' ');
                                warningLow();
                        } else {
                                printTemp(k); //ingen varning
                        }
                }
}

//Skriver ut temperatur
void printTemp(int t)
{
                int ental = t % 10;
                t= t/10;
                int tiotal = t%10;
                t= t/10;
                int hundratal = t%10;

                if (hundratal!= 0) // ej säkert att det finns hundratal
                {
                int a = calcTemp(hundratal);
                displayChar(a);
                }
                if (tiotal!= 0) // ej säkert att det finns tiotal
                {
                int b = calcTemp(tiotal);
                displayChar(b);
                }
                int  c= calcTemp(ental); // säkert att det finns ett tal
                displayChar(c);

                displayChar(0b11011111); // tecken för celcius
                displayChar('C');
}

//Skriver ut tiden
void print_calc_time(uint8_t h)
{
                int ental = h%10;
                h= h/10;
                int tiotal = h%10;
                h= h/10;
                int hundratal = h%10;

                if (hundratal!= 0)
                {
                        int a = calcTemp(hundratal);
                        displayChar(a);
                }
                if (tiotal!= 0)
                {
                        int b = calcTemp(tiotal);
                        displayChar(b);
                } else
                {
                        displayChar('0');
                }
```

```
		int  c= calcTemp(ental);
		displayChar(c);
}

//Omvandlar temperaturen x till en siffra som kan generera x i
displayen
int calcTemp(int x)
{
		x += 0b00110000; // lägger till 49 här för att siffran ska
stämma överens med tabellen i displayens siffror. Motsvarar siffran i
bits
		return x;
}

//Varningsmeddelande för min-temp
void warningLow()
{
		PORTC|=(1<<PC6);
		setRow(0xC0);
		displayString("WARNING! Low temp!");
		_delay_ms(100);
		PORTC &=~(1<<PC6);
}

//Varningsmeddelande för max-temp
void warningHigh()
{
		PORTC |= (1<<PC6);
		setRow(0xC0);
		displayString("WARNING! High temp!");
		_delay_ms(100);
		PORTC &=~(1<<PC6);
}

//Skriver ut vad maxgränsen för temperaturen är för inne- eller
utemperatur
void maxTempSetUp()
{
		clearDisplay();
		displayString("Max limit for ");
		setRow(0xC0);

			if (outOrIn > 0)
			{
				displayString("indoor temp is now: ");
				setRow(0x94);
				printTemp(maxTempIn);
			}
			else
			{
				displayString("outdoor temp is now: ");
				setRow(0x94);
				printTemp(maxTempOut);
			}
}

//Skriver ut vad maxgränsen för temperaturen är för inne- eller
utemperatur
```

```
void minTempSetUp()
{
          clearDisplay();
          displayString("Min limit for ");
          setRow(0xC0);

                    if (outOrIn > 0)
                    {
                              displayString("indoor temp is now: ");
                              setRow(0x94);
                              printTemp(minTempIn);
                    }
                    else
                    {
                              displayString("outdoor temp is now: ");
                              setRow(0x94);
                              printTemp(minTempOut);
                    }
}

//Ökar maximala temperaturgränsen innan varning skickas med 1 kelvin
void increaseMaxTemp()
{
          if (outOrIn > 0)
          {
                    maxTempIn += 1;
          }
          else
          {
                    maxTempOut += 1;
          }
          maxTempSetUp();
}

//Minskar maximala temperaturgränsen innan varning skickas med 1 kelvin
void decreaseMaxTemp()
{
          if (outOrIn > 0)
          {
                    maxTempIn -= 1;
          }
          else
          {
                    maxTempOut -= 1;
          }
          maxTempSetUp();
}

//Ökar minsta temperaturgränsen innan varning skickas med 1 kelvin
void increaseMinTemp()
{
          if (outOrIn > 0)
          {
                    minTempIn += 1;
          }
          else
          {
                    minTempOut += 1;
```

```
		}
		minTempSetUp();
}

//Minskar minsta temperaturgränsen innan varning skickas med 1 kelvin
void decreaseMinTemp()
{
		if (outOrIn > 0) // sänker gräns inne
		{
				minTempIn -= 1;
		}
		else //sänker gräns ute
		{
				minTempOut -= 1;
		}
		minTempSetUp();
}

//Raderar mätdata i vektorerna och återställer mätindex
void clearVectors()
{
		int k = 0;
		while (k < tempIndexIn)
		{ //alla värden sätts till 0
				tempVecIn[k]=0;
				k++;
		}
		tempIndexIn=0;

		k=0;
		while (k < tempIndexOut)
		{ //alla värden sätts till 0
				tempVecOut[k]=0;
				k++;
		}
		tempIndexIn=0;
}

//Sparar temperaturavläsningar i två vektorer för inne och ute
void insertTempIn (int j)
{
if (tempIndexIn==99)
{
		tempIndexIn= 0;
}
				tempVecIn[tempIndexIn] = j;
				tempIndexIn+=1;
}

void insertTempOut (int j)
{
		if (tempIndexOut==99)
		{
				tempIndexOut= 0;
		}
				tempVecOut[tempIndexOut] = j;
				tempIndexOut+=1;
}
```

```
//Skriver ut den genomsnittliga temperaturen inne
void printAverageTempIn()
{
          int sum = 0;
          int k = tempIndexIn;
          while(k > 0)
          {
                    sum +=tempVecIn[k];
                    k -= 1;

          }
          int averageTempIn = sum/(tempIndexIn); //oklart vad vi ska
dela på
          displayString("Average indoor ");
          setRow(0xC0);
          displayString("temp: ");
          printTemp(averageTempIn);
}

//Skriver ut den genomsnittliga temperaturen ute
void printAverageTempOut()
{
          int sum =0;
          int k =tempIndexOut;
          while(k > 0)
          {
                    sum +=tempVecOut[k];
                    k-= 1;

          }
          int averageTempOut = sum/(tempIndexOut-1);
          displayString("Average outdoor ");
          setRow(0xC0);
          displayString("temp: ");
          printTemp(averageTempOut);
}


//INTERUPT FOR AVERAGE

//Initiera tick
void set_up_tick() {
          GICR |=(1<<INT1);
          MCUCR|= (1<<ISC11) |(1<<ISC10);
}

void enable_interupt() {
          i2c_start(RTC_WRITE);
          i2c_write(0x07);
          i2c_write(0b11000000);
          i2c_stop();
}

ISR(INT1_vect) {
          ticked = 1;
}
```

```
// KEYBOARD

//Konfigurerar inställningarna för knappsatsen
void setUpKeyboard()
{
          MCUCR |= 0b00000011; //Sista ska vara 0 för att interrupt, se
sida 67
          GICR |= 0b01000000; //INT0, bit 6,ska vara satt till 1, sida
67, de andra ej relevanta
          DDRD &=~(1<<PD2); //Data available i encodern
}

//Interrupt keyboard
ISR(INT0_vect) {
          val=(PINA&0b00111100) >> 2; //Konfigurering av Data Output
från Encodern till MPU. A, B, C, D ska vara 1 ty vi ska läsa av dessa
pinnar, de är dessa som är relevanta
          knappflagg = 1;
}

//En knapp har trycks ned, läs av val
void buttonPressed() {

                    if(val==0x03) //Knapp 0, start-sida
                    {
                              clearDisplay();
                              printTime();
                    }

                    if(val==0x02) //Knapp 1, Skriv ut nuvarande
innetemperatur
                    {
                              clearDisplay();
                              displayString("Indoor temp: ");
                              print(tempIn);
                    }

                    if (val==0x01) //Knapp 2, skriver ut nuvarande
utetemperatur
                    {
                              clearDisplay();
                              displayString("Outdoor temp: ");
                              print(tempOut);
                    }

                    if (val==0x00) //Knapp 3, visar högst uppmätta temp
                    {
                              clearDisplay();
                              displayString("Highest recorded");
                              setRow(0xC0);
                              displayString("temp: ");
                              printTemp(MAX);
                    }

                    if (val==0x07) //Knapp 4, visar lägst uppmätta temp
                    {
                              clearDisplay();
                              displayString("Lowest recorded");
```

```
                                setRow(0xC0);
                                displayString("temp: ");
                                printTemp(MIN);
                        }

                        if (val==0x06) //Knapp 5, visar average temperatur
inne
                        {
                                 clearDisplay();
                                 printAverageTempIn();
                        }

                        if (val==0x05) //Knapp 6 visar average temperaturer
ute
                        {
                                clearDisplay();
                                printAverageTempOut();
                        }

                        if(val==0x04) //Knapp 7, data cleared!
                        {
                                clearDisplay();
                                clearVectors();
                                displayString("Data cleared!");
                        }

                        if(val==0b1011) //Knapp 8, bestäm ute eller inne
                        {
                                clearDisplay();
                                displayString("You will now set: ");
                                setRow(0xC0);
                                if(outOrIn > 0)
                                {
                                        outOrIn = 0;
                                        displayString("outdoor
limits");
                                } else
                                {
                                        outOrIn = 1;
                                        displayString("indoor
limits");
                                }
                        }

                        if(val==0b1010) //Knapp 9, bestäm om max eller min
                        {
                                clearDisplay();
                                _delay_ms(50);
                                displayString("You will now set:");
                                setRow(0xC0);
                                if(minOrMax > 0)
                                {
                                        minOrMax = 0; //Sätter den
till det den inte var innan
                                        displayString("max limit");
                                } else
                                {
                                        minOrMax = 1;
```

```
                displayString("min limit");
        }
    }

    if(val==0x09) { //Knapp A, öka varningsgränsen med
1 celsius
        clearDisplay();
        if(minOrMax > 0)
        {
                increaseMinTemp();
        } else
        {
                increaseMaxTemp();
        }
    }

    if(val==0x08) { //Knapp B, minska varningsgränsen
med 1 celsius
        clearDisplay();
        if(minOrMax > 0)
        {
                decreaseMinTemp();;
        } else
        {
                decreaseMaxTemp();
        }
    }

    if(val==0b1111) { //Knapp C, växla mellan år, månad
och dag
        clearDisplay();
        displayString("You will now set:");
        setRow(0xC0);
        if (change == 0) {
                displayString("year");
                change = 1;
        } else if (change == 1) {
                displayString("month");
                change = 2;
        } else if (change == 2){
                displayString("day");
                change = 3;
        } else if (change==3){
                displayString("hour");
                change = 4;
        } else if(change==4){
                displayString("minute");
                change = 5;
        } else { //change är 5
                displayString("second");
                change = 0;
        }
    }

    if(val==0b1110) { //Knapp D, öka tid och datum
        clearDisplay();
        if (change == 1) {
                set_year+=1;
```

```
                                        displayString("Year is now:
");
                                        print_calc_time(set_year);
                                } else if (change == 2) {
                                        set_month+=1;
                                        displayString("Month is now:
");
                                        print_calc_time(set_month);
                                } else if (change == 3){
                                        set_day+=1;
                                        displayString("Date is now:
");
                                        print_calc_time(set_date);
                                } else if (change==4){
                                        set_h+=1;
                                        displayString("Hour is now:
");
                                        print_calc_time(set_h);
                                } else if(change==5){
                                        set_min+=1;
                                        displayString("Minute is now:
");
                                        print_calc_time(set_min);
                                } else { //change är 5
                                        set_sec+=1;
                                        displayString("Second is now:
");
                                        print_calc_time(set_sec);
                                }
                        }

                        if(val==0b1101) { //Knapp E, minska tid och datum
                                clearDisplay();
                                if (change == 1) {
                                        set_year-=1;
                                        displayString("Year is now:
");
                                        print_calc_time(set_year);
                                } else if (change == 2) {
                                        set_month-=1;
                                        displayString("Month is now:
");
                                        print_calc_time(set_month);
                                } else if (change == 3){
                                        set_day-=1;
                                        displayString("Date is now:
");
                                        print_calc_time(set_date);
                                } else if (change==4){
                                        set_h-=1;
                                        displayString("Hour is now:
");
                                        print_calc_time(set_h);
                                } else if(change==5){
                                        set_min-=1;
                                        displayString("Minute is now:
");
                                        print_calc_time(set_min);
```

```
                                } else { //change är 5
                                        set_sec-=1;
                                        displayString("Second is now:
");
                                        print_calc_time(set_sec);
                                }
                        }

                        if(val==0b1100) { //Knapp F, reboot time
                                clearDisplay();
                                //Stop clock
                                i2c_start(RTC_WRITE);
                                i2c_write(0x00);
                                i2c_stop();
                                //Start clock again, with new time
values
                                set_clock();
                                displayString("Time is updated!");
                                setRow(0x94);
                                printTime();
                        }
}


// TIME

//Skriver ut tid och datum
void printTime() {
          get_time(); //uppdatera tiden
          displayString("Date: ");
          print_calc_time(get_date);
          displayString("/");
          print_calc_time(get_month);
          displayString("/");
          print_calc_time(get_year);
          displayString(" ");

          setRow(0xC0);
          displayString("Time: ");
          uint8_t hour = 10*((0b00110000 & get_h)>>4) + (get_h &
0b00001111);
          print_calc_time(hour);
          displayString(":");
          uint8_t minutes = 10*((0b01110000 & get_min)>>4) + (get_min &
0b00001111);
          print_calc_time(minutes);
          displayString(":");
          uint8_t seconds = 10*((0b01110000 & get_sec)>>4) + (get_sec &
0b00001111);
          print_calc_time(seconds);

}


// MAINMETHOD
int main(void)
{
          maxTempIn =25;
```

```
        minTempIn =15;
        maxTempOut = 35;
        minTempOut = 0;

        val = 0;
        outOrIn = 1;
        minOrMax = 1;
        interTemp = 1;
        change = 0;

        _delay_ms(50);

        initDiods();

        initDisplay();
        displayOnOff();
        configureDisplay();
        clearDisplay();
        _delay_ms(50);
        displayString("Welcome!");

        setUpKeyboard();

        initThermal();

        i2c_init();
        set_clock();
        set_up_tick();
        enable_interupt();


        sei();

        while (1) {

                if(ticked==1){ //ta ny temperatur för medel
                        insertTempIn(tempIn);
                        insertTempOut(tempOut);
                        ticked = 0;
                }

                _delay_ms(10);

                if(knappflagg==1)
                {
                        buttonPressed();
                        knappflagg=0;
                }


        } //stänger while

} //stänger main
```