```c
/*
 * Eggtimer_v1.c
 *
 * Created: 2016-04-21 12:00:51
 *  Author: digpi05
 */

#include <avr/io.h> //kan skriva betäckningar som i databladet
#include <util/delay.h>
#include <avr/interrupt.h>


#define F_CPU 8000000UL   // 8 MHz konstant som definierar frekvensen

//------------------------------VARIABLES--------------------------------------

int buttonReset = 0;
int buttonTime = 0;
int celsius;
double rawVoltage;
double millivolts;
double counter; //tidräkning
int min;
int sec;
int sec1;
int sec2;

//flaggor
int first0;
int first1;
int first2;
int first3;
int first4;
int reached;

//------------------------------METHODS----------------------------------------

void setUp();
void setUpLCD();
void setPin(char port, char pin, char state);
int checkButton();
void enableButtonInterrupt();

void phase1();
void phase2();
void phase3();

void LCDcmd(char cmd);
void LCDwrt(char ch);
void LCDclear();
void backOnTrack();

void messageWelcome();
void messageunder100();
void messagePhase1();
void messagePhase2();
void messagePhase3();
void messageButton2();
```

```
void messagesunk();

void setUpTemp();
int readTemp();

//---------------------------------------SETUP-----------------------------------

void setUp() {
            DDRA=0b00001110;         //1 är utsignal, 0 är insignal eller används inte
            DDRB=0b11111111;
            DDRC=0b00000000;
            DDRD=0b00111000;

            enableButtonInterrupt();
            sei(); //enables interuptions

            counter=0; //tidräkning=0

            first0=0;
            first1=0;
            first2=0;
            first3=0;
            first4=0;
            reached=0;

            setPin('A', PA1, 0); //grön lampa släcks
            setPin('A', PA2, 0); //gul lampa släcks
            setPin('A', PA3, 0); //röd lampa släcks

            setUpLCD();
            setUpTemp();
            setUpTimer();
            celsius=readTemp(); //läser av temperaturen
}

void setUpLCD() {
            LCDclear();
            LCDcmd(0x38); //functional set
            LCDcmd(0x0F); //display on
            LCDcmd(0x01);
            LCDcmd(0x03); //entry mode set
            messageWelcome();
}


//--------------------------------HELP-METHODS-----------------------------------------

void setPin(char port, char pin, char state) {
            char set = 1 << pin; // bit shifts -> set = shifting 1 to the left by pin
bits -> sätter aktuell pin till 1
            if (port == 'A') {
                        set &= PORTA; //bitwise AND -> kräver att båda är 1+1=1, 1+0=0.
läs på datorn
                        if(set && !state) { // om set=1 och state=0
                                    PORTA ^= set; // ändrar set eller state från 1 -> 0?
Vi sätter PORTA till set. bitwise exclusive or -> 0+0=0, 1+0=1, 1+1=0
                        }
                        if(set==0 && state) { // om set=0 och state = 1 LIKA MED NOLL???
```

```
                                        set = 1 << pin;          //sätter aktuell pin till 1
                                        PORTA^=set;  //ändrar state från 0 -> 1
                        }
        } else if (port == 'B') {
                        set &= PORTB;
                        if(set && !state) {
                                        PORTB ^= set;
                        }
                        if(set == 0 && state) {
                                        set = 1 << pin;
                                        PORTB^=set;
                        }
        } else if (port == 'C') {
                        set &= PORTC;
                        if(set && !state) {
                                        PORTC ^= set;
                        }
                        if(set == 0 && state) {
                                        set = 1 << pin;
                                        PORTC^=set;
                        }
        } else if (port == 'D') {
                                        set &= PORTD;
                                        if(set && !state) {
                                        PORTD ^= set;
                                        }
                                        if(set == 0 && state) {
                                        set = 1 << pin;

                                        PORTD^=set;
                                        }
                        }
        }

        //--------BUTTONS-------

        int checkButton() {
                        char check = PINA & 0b00110000; //kollar PA4 och PA5, alla andra
kommer bli noll
                        if(check == 16) { //om reser var intryckt (2^4)
                                        return 1;

                        } else if (check == 32) { //om buttonTime var intryckt (2^5)
                                        return 2;
                        }
        }


        void enableButtonInterrupt() {
                        MCUCR = 0b00000011; //s 66, detta är för INT0
                        SREG=0x82;
                        GICR=(1<<INT0);
        }

        ISR(INT0_vect) { //Globalt button-interrupt.
                        _delay_ms(1);
                        int a = checkButton();
                        if(a==1) {
```

```
                                buttonReset=1;
                        } else if (a==2) {
                                buttonTime=1;
                        }
                }

//---------------------------------------PHASES-----------------------------------------
-----------

void phase1() {
                setPin('A', PA1, 1);
}

void phase2() {
                setPin('A', PA1, 1);
                setPin('A', PA2, 1);
}

void phase3() {
                setPin('A', PA1, 1);
                setPin('A', PA2, 1);
                setPin('A', PA3, 1);
}



                //--------------------------LCD-----------------------------------------

void LCDcmd(char cmd) {
                _delay_ms(50);
                setPin('D', PD3,1); //R/W=1
                setPin('D',PD5,1); //RS=1
                setPin('D', PD4,1); //E=1
                PORTB=cmd;
                setPin('D', PD3,0);//R/W=0
                setPin('D',PD5,0); //RS=0

                setPin('D', PD4,0);//E=0 data läses av

                setPin('D', PD3,1); //R/W=1
                setPin('D',PD5,1); //RS=1
                setPin('D', PD4,1); //E=1
                }

void LCDwrt(char ch) { // RS alltid 1
                _delay_ms(50);
                setPin('D', PD3,1); //R/W=1
                setPin('D',PD5,1); //RS=1
                setPin('D', PD4,1); //E=1
                PORTB=ch;
                setPin('D', PD3,0); //R/W=0

                setPin('D', PD4,0); //E=0 data läses av

                setPin('D', PD3,1); //R/W=1
                setPin('D', PD4,1); //E=1
}
```

```
void LCDclear() {
            LCDcmd(0x02); //cursor home
            for(int a=0;a<40;a++) {
                        LCDwrt(' ');
            }
            LCDcmd(0xC0);
            for(int a=0;a<20;a++) {
                        LCDwrt(' ');
            }
            LCDcmd(0x02);
}

//-----------------------------------MESSAGES-------------------------------------

void messageWelcome() {
            LCDwrt('W');
            LCDwrt('E');
            LCDwrt('L');
            LCDwrt('C');
            LCDwrt('O');
            LCDwrt('M');
            LCDwrt('E');
            _delay_ms(20000);
            LCDcmd(0x01); //clear display
}

void messageunder100() {
            LCDclear();
            LCDwrt('T');
            LCDwrt('E');
            LCDwrt('M');
            LCDwrt('P');
            LCDwrt(' ');
            LCDwrt('H');
            LCDwrt('A');
            LCDwrt('S');
            LCDwrt(' ');
            LCDwrt('N');
            LCDwrt('O');
            LCDwrt('T');
            LCDwrt(' ');
            LCDcmd(0xC0);
            LCDwrt('R');
            LCDwrt('E');
            LCDwrt('A');
            LCDwrt('C');
            LCDwrt('H');
            LCDwrt('E');
            LCDwrt('D');
            LCDwrt(' ');
            LCDwrt('1');
            LCDwrt('0');
            LCDwrt('0');
            LCDwrt(' ');
            LCDwrt('D');
            LCDwrt('E');
            LCDwrt('G');
            LCDcmd(0x00);
```

```
            }

void messageOver100() {
            LCDclear();
            LCDwrt('T');
            LCDwrt('E');
            LCDwrt('M');
            LCDwrt('P');
            LCDwrt(' ');
            LCDwrt('H');
            LCDwrt('A');
            LCDwrt('S');
            LCDcmd(0xC0);
            LCDwrt('R');
            LCDwrt('E');
            LCDwrt('A');
            LCDwrt('C');
            LCDwrt('H');
            LCDwrt('E');
            LCDwrt('D');
            LCDwrt(' ');
            LCDwrt('1');
            LCDwrt('0');
            LCDwrt('0');
            LCDwrt(' ');
            LCDwrt('D');
            LCDwrt('E');
            LCDwrt('G');
            LCDcmd(0x00);
            }

void messagePhase1() {
LCDclear();
LCDwrt('Y');
LCDwrt('O');
LCDwrt('U');
LCDwrt('R');
LCDwrt(' ');
LCDwrt('E');
LCDwrt('G');
LCDwrt('G');
LCDwrt(' ');
LCDwrt('I');
LCDwrt('S');
LCDcmd(0xC0);
LCDwrt('S');
LCDwrt('O');
LCDwrt('F');
LCDwrt('T');
LCDwrt('-');
LCDwrt('B');
LCDwrt('O');
LCDwrt('I');
LCDwrt('L');
LCDwrt('E');
LCDwrt('D');
LCDcmd(0x00);
}
```

```
void messagePhase2() {
            LCDclear();
            LCDwrt('Y');
            LCDwrt('O');
            LCDwrt('U');
            LCDwrt('R');
            LCDwrt(' ');
            LCDwrt('E');
            LCDwrt('G');
            LCDwrt('G');
            LCDwrt(' ');
            LCDwrt('I');
            LCDwrt('S');
            LCDcmd(0xC0);
            LCDwrt('M');
            LCDwrt('E');
            LCDwrt('D');
            LCDwrt('-');
            LCDwrt('B');
            LCDwrt('O');
            LCDwrt('I');
            LCDwrt('L');
            LCDwrt('E');
            LCDwrt('D');
            LCDcmd(0x00);
}

void messagePhase3() {
            LCDclear();
            LCDwrt('Y');
            LCDwrt('O');
            LCDwrt('U');
            LCDwrt('R');
            LCDwrt(' ');
            LCDwrt('E');
            LCDwrt('G');
            LCDwrt('G');
            LCDwrt(' ');
            LCDwrt('I');
            LCDwrt('S');
            LCDwrt(' ');
            LCDcmd(0xC0);
            LCDwrt('H');
            LCDwrt('A');
            LCDwrt('R');
            LCDwrt('D');
            LCDwrt('-');
            LCDwrt('B');
            LCDwrt('O');
            LCDwrt('I');
            LCDwrt('L');
            LCDwrt('E');
            LCDwrt('D');
            LCDcmd(0x00);
}
void messagesunk(){
            LCDclear();
```

```
                LCDwrt('I');
                LCDwrt('N');
                LCDwrt('C');
                LCDwrt('R');
                LCDwrt('E');
                LCDwrt('A');
                LCDwrt('S');
                LCDwrt('E');
                LCDwrt(' ');
                LCDwrt('T');
                LCDwrt('H');
                LCDwrt('E');
                LCDcmd(0xC0);
                LCDwrt('T');
                LCDwrt('E');
                LCDwrt('M');
                LCDwrt('P');
                LCDcmd(0x00);
}


void messageButton2(){
                LCDclear();
                min = (int) counter/60;
                sec= (int) counter%60;
                sec1=sec/10;
                sec2=sec%10;
                LCDwrt('T');
                LCDwrt('I');
                LCDwrt('M');
                LCDwrt('E');
                LCDwrt(':');
                LCDcmd(0xC0);
                intToChar(min);
                LCDwrt(':');
                intToChar(sec1);
                intToChar(sec2);
                LCDcmd(0x00);
}

void intToChar(int number)
{
                if(number == 0){
                                LCDwrt(0x30);
                }
                if(number == 1){
                                LCDwrt(0x31);
                }
                if(number == 2){
                                LCDwrt(0x32);
                }
                if(number == 3){
                                LCDwrt(0x33);
                }
                if(number == 4){
                                LCDwrt(0x34);
                }
                if(number == 5){
```

```
                    LCDwrt(0x35);
            }
            if(number == 6){
                    LCDwrt(0x36);
            }
            if(number == 7){
                    LCDwrt(0x37);
            }
            if(number == 8){
                    LCDwrt(0x38);
            }
            if(number == 9){
                    LCDwrt(0x39);
            }
}

void backOnTrack() {
            if(first3==1) {
                    messagePhase3();
                    } else if (first2==1) {
                    messagePhase2();
                    } else if (first1==1) {
                    messagePhase1();
                    } else if(first4==1) {
                    messageOver100();
                    } else {
                    messageunder100();
             }
}


//----------------------------TEMP-------------------------------------------

void setUpTemp() {
            ADMUX=0b01000000;
            ADCSRA=0b11000011; //delat med 8. s 217
}

int readTemp() {
            rawVoltage=ADC;
            millivolts=(rawVoltage/1024)*5000;
            celsius=(int)((millivolts/10)-273);
            ADCSRA=0b11000011;
            return celsius;
}

//-------------------------------------TIME----------------------------

void setUpTimer() { //16 bit timer
            TCCR1A=0b00000000;
            TCCR1B=0b00000011; // Prescaler =64
            TCNT1H=0xFF;
            TCNT1L=0xFF;
            TIMSK=0x04; //enable overflow interrupt
            TIFR=0x04; //enable overflow interrupt
}
```

```
ISR(TIMER1_OVF_vect) { //timer för att mäta temperaturen och räkna tid.
65536/(8Mhz/64)=0,524288 s.
            celsius=readTemp();
            if(celsius>=17 || reached==1) { //ska vara 100 celsius eller någon gång nått
100 celsius.
                        counter=counter+0.524288;
            }
}


//-----------------------------------MAIN---------------------------

int main(void)
{
setUp();

while(1)    {
            if(celsius<17 && reached==0) { // ska vara 100 celsius

                                    if(first0==0) {
                                                LCDclear();
                                                messageunder100();;
                                    }
                                    first0=1;
                                    }

            if(celsius>=17 || reached==1) { // ska vara 100 celsius
                        reached = 1;
                        if(celsius<17) { // ska vara 100 celsius
                                    messagesunk();
                                    _delay_ms(8000);
                                    backOnTrack();
                        }
                        if(first4==0) {
                                    messageOver100();
                        }
                        first4=1;

                        if(counter>=30 && counter < 45 ) { //4-6 min -> 240-360 sek

                                    phase1();
                                    if(first1==0) {
                                                messagePhase1();
                                    }
                                    first1=1;
                        }
                        else if (counter >= 45 && counter < 60) { //6-10 min -> 360-600
sek
                                    phase2();
                                    if(first2==0) {

                                                messagePhase2();
                                    }
                                    first2=1;
                        } else if(counter>=60) { // > 10 min -> 600 sek
                                    phase3();
                                    if(first3==0) {
```

```
                                                messagePhase3();
                                    }
                                    first3=1;
                        }

            }

            if(buttonReset==1) {
                        setUp();
                        buttonReset=0;
            }
            if(buttonTime==1) {
                        messageButton2();
                        _delay_ms(6000);
                        buttonTime=0;
                        backOnTrack();
            }

}

}
```