Rörelse- och temperaturkänsligt larmsystem

LUNDS UNIVERSITET

Gustav Christensen I15
Andreas Warvsten I15
Jesper Lundqvist I15
Projektarbete i kursen Digitala Projekt EITF11
Institutionen för Elektro- och informationsteknik
Handledare: Bertil Lindvall
2018–05–17

Abstract

In the course Digital Systems, Project Laboratory every group of students were given the opportunity to fabricate a digital product. The product was not on beforehand decided, instead we could freely choose the specific product we wanted to develop. The selected processor ATmega16 has a central role in our construction and is interconnected with the remaining components. The half year project has been very worthwhile in different ways. First of all the freedom to build the product which interested us the most. Second of all, we as a group have learnt more about digital communication and the close connection between hardware and software development in the industry.

Innehållsförteckning

1. Inledning

På grund av den oro som finns över att bli bestulen sina ägodelar valde vår grupp att konstruera ett larmsystem. Framför allt känner vi en oroa över att bli bestulna våra skoldatorer som vi är högst beroende av i vårt skolarbete. För att höja säkerhetsnivån och gardera oss om att våra datorer inte blir stulna ville vi att vårt larmsystem skulle vara både rörelse- och temperaturkänsligt samtidigt som vi kan fastställa när brottet skett utifall att det sker.

Digitala projekt, EITF11, är en halvårskurs anpassad för bland annat civilingenjörsstudenter för att skapa en bild av kopplingen mellan hårdvaru- och mjukvaruutveckling för att på så sätt få en bild av hur utvecklingsarbetet kan se ut i industrin. Varje grupp består av två-tre studenter som tillsammans ska bygga hårdvara och programmera tillhörande mjukvara. För att skapa förutsättningar för detta ges ett antal föreläsningar och laborationer under kursens första del.

2. Databladsreferenser

REF1: Encoder: 54C922
REF2: Processor: ATmega16
REF3: Temperatursensor: LM35
REF4: IR-sensorer: Parallax DevB
REF5: LCD-display: SHARP Dot-Matrix
REF6: Realtidsklocka: ICM 7170

3. Kravspecifikation

3.1. Definitioner

- **Aktiverat/deaktiverat larmsystem** innebär att PIR-sensorerna reagerar (eller inte reagerar) på rörelser och temperatursensorerna mäter (eller inte mäter) en temperatur större än 30 grader Celsius.

I scenariot där larmsystemet anses vara aktiverat kan två underscenarion uppstå:

- **Icke-utlöst larm** innebär att rörelse- och temperatursdetektorer är aktiva men att varken rörelse- eller temperatursaktiviteter har uppfattats av någon av detektorerna.
- **Utlöst larm** innebär att rörelse- och/eller temperatursdetektorerna har lyssnat av en rörelse eller temperaturskillnad och/eller att användaren har angivit felaktig kod inom tidsramen.

3.2. Grundläggande krav

Krav 01: Larmsystem ska ha två PIR-sensorer för detektion av rörelser i olika riktningar.
Krav 02: Larmsystem ska ha två temperatursensorer för detektion av temperaturskillnader.
Krav 03: Larmsystem ska ha en summer som ger ifrån sig ljud med en förutbestämd frekvens om larmet utlöses, annars inget ljud alls.
Krav 04: Larmsystem ska ha en encoder kopplad till knappsats och processor.
Krav 05: Larmsystem ska aktiveras då en fyrsiffrig kod skrivs in via knappsats.
Krav 06: Larmsystem ska avaktiveras då en korrekt fyrsiffrig kod skrivs in via knappsats.
Krav 07: Larmsystems fyrsiffriga kod vid aktivering/deaktivering ska kunna ändras.
Krav 08: Larmsystem ska ha en grön lysdiod som lyser ifall systemet är deaktiverat.
Krav 09: Larmsystem ska ha en röd lysdiod som lyser ifall systemet är aktiverat.

Krav 10: Larmsystem ska ge en sakenlig återkoppling via en inkopplad LCD-display innehållande information.

3.3. Användarfall

3.3.1. Användarfall A: Användare aktiverar larmsystem

Premiss: Larmsystem är deaktivt, prototyp är strömförsörjd och grön LED-lampa lyser.
Utfall:

1. Användare trycker ner A-knappen och anger önskad pinkod.
2. LCD-display visar tidsfördröjning 5 sekunder tills larmet aktiveras.
3. Användare aktar sig från prototyp under tidsfördröjningen tills larmet aktiveras.
4. LCD-display visar meddelande ”ACTIVATED”.
5. Larm är aktiverat och grön LED-lampa släcks medan röd LED-lampa tänds.

3.3.2. Användarfall B: Användare deaktiverar larm

Premiss: Larmsystem är aktivt och utlöst, prototyp är strömförsörjd, röd LED-lampa lyser, summer är aktiv och LCD-display visar meddelande ”ENTER PIN:”.
Utfall A: Användare anger korrekt pinkod

1. Användare anger pinkod som angivits vid aktivering av larmsystem.
2. LCD-display visar meddelande ”DEACTIVATED”.
3. Larm är deaktiverat och grön LED-lampa tänds medan röd LED-lampa släcks och summer stängs av.
4. Tiden för larmutlösning och hur larm utlöses sparas och nås via D-knappen.

Utfall B: Användare anger inkorrekt pinkod

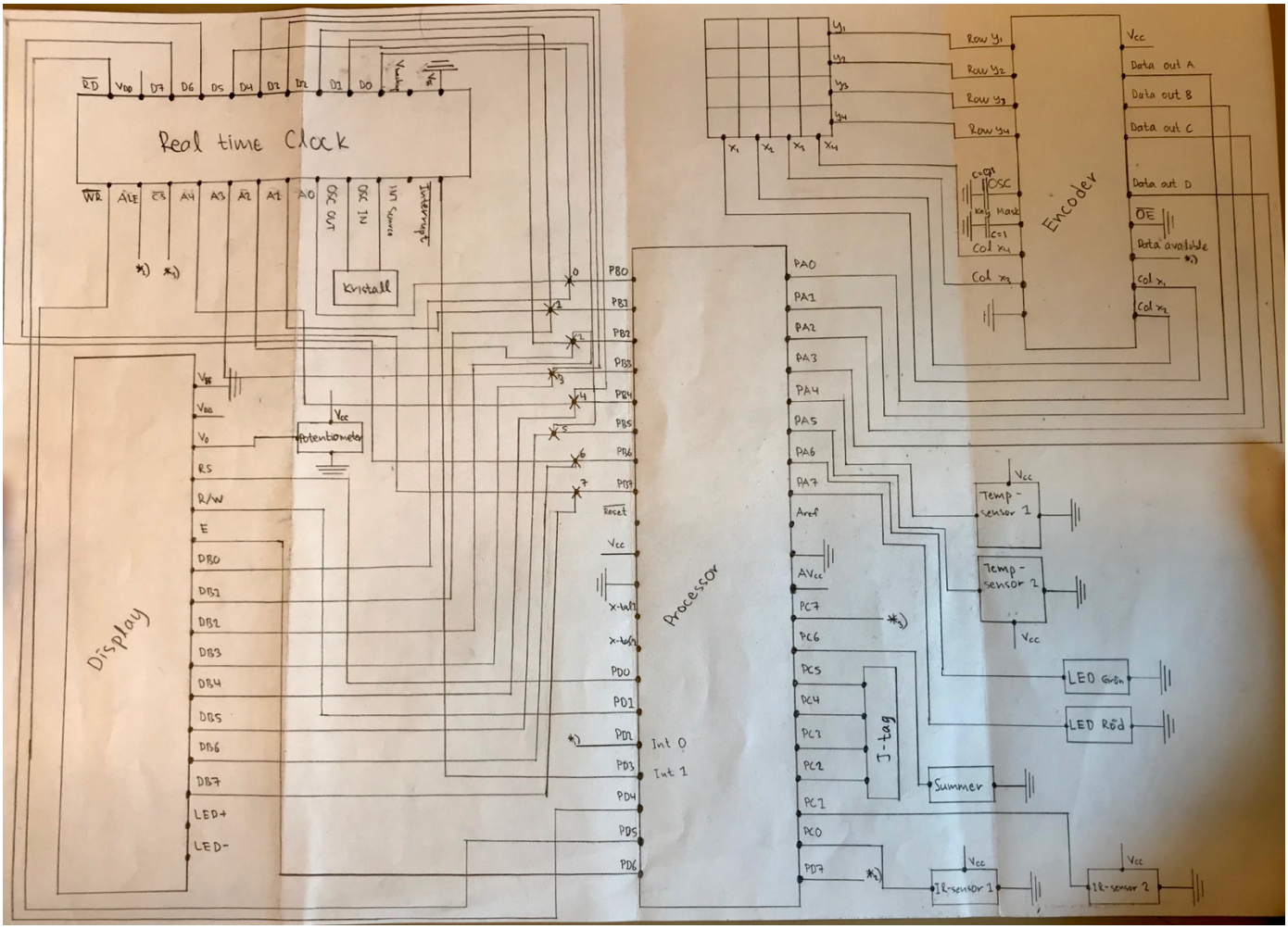
1. Användare anger inte den pinkod som angivits vid aktivering av larmsystem.
2. LCD-display visar meddelande ”WRONG PIN!” under en sekund.
3. LCD-display töms.
4. Användarfall B sker igen.

4. Hårdvara

4.1. Prototyp

Figur 1: Bild på larmsystemsprototypen

4.2. Kopplingsschema



Figur 2: Ritning av larmsystems kopplingsschema

Larmsystemet består av samtliga komponenter beskrivna i avsnitt 4.3–4.11

4.3. Knappsats

Larmsystemet använder en enkel knappuppsättning med hexadecimal inmatning uppdelat på fyra rader och kolumner. Rad 1: 0 till 3, rad 2: 4 till 7, rad 3: 8 till B, rad 4: C till F.

4.4. Encoder

Vid omvandling av information via knapptryck från knappsatsen till signaler in till processor används en encoder av modell 54C922. Se REF1 under avsnitt 2 ”Referenser” för vidare information.

4.5. Processor

Den använda processorn till larmsystemet heter ATmega16 High-Performance AVR 8-bit Microcontroller som har programmerats med hjälp av högnivåspråket C i utvecklingsmiljön *AtmelStudio*. Vid överföring av kod från dator till processor användes en J-Tag. För vidare information hänvisas läsaren till avsnitt 2 ”Referenser” och specifikt REF2.

4.6. Sensorer

4.6.1. IR-sensorer

För rörelsedetektion används två PIR-sensorer (Passive Infra-Red) av modell *#555–28027* som känner av förändringar i det infraröda spektrumet som avges av objekt i närhet av sensorns placering. Vid detektion av rörelse skickas en signal vidare in till processorn som utlöser larmet via summern. För vidare information om IR-sensorerna se avsnitt 2 ”Referenser” under REF4.

4.6.2. Temperatursensorer

För temperaturdetektion används två temperatursensorer. Dessa detektorer är anpassad för Celsiusskalan genom att dess spänningsutdata är proportionell mot Celsiustemperaturen. Ju högre temperatur desto högre spänning. På samma sätt som rörelsedetektorerna skickas en signal vidare till processorn som utlöser larmet via summern utifall att sensorerna känt av en temperatur över 30 grader Celsius. För vidare information se avsnitt 2 ”Referenser” under REF3.

4.7. Lysdioder

Larmsystemet består av två stycken lysdioder varav den ena är grön och den andra är röd. Lysdioderna används för att informera närvarande personer om larmsystemets status. Lyser den gröna lampan innebär det att larmsystem är deaktivt och lyser den röda lampan betyder det att larmet är aktivt. Se avsnitt 3.1 ”Definitioner” för information om (in)aktivitet.

4.8. Summer

Vid utlösande av larm ges en ljudlig notation till berörda parter. För att skapa denna ljudbild används en summer som en del av larmsystemet.

4.9. Kristall

För att öka precisionen hos larmsystems realtidsklocka används en kristall.

4.10. LCD-display

Som display används modellen *SHARP Dot-Matrix* där varje tecken består av 5x8 pixlar och två rader med plats för 8 tecken per rad. Displayen används för kommunikation med användaren i form av larmets aktivitetsstatus, instruktioner, återkoppling vid pinkodsinskrivning och tid för utlöst larm. Var god se avsnitt 2 ”Referenser” REF5 för vidare information.

4.11. Realtidsklocka

Komponenten används för att möjliggöra tidpunkt för utlösande av larm. Modellen heter *ICM 7170* och innehåller en reglerad oscillator som säkerställer frekvensstabilitet och låg energiförbrukning. Realtidsklockan har möjlighet att bestämma tiden mellan intervallet 1/100 sekund och 99 år. För mer information, se tillhörande datablad i avsnitt 2 ”Referenser” under REF6.

5. Mjukvara

Main-metoden består av en initieringsdel samt en evig while-loop från vilken prototypens funktioner kan kallas. Denna avbryts av två stycken interrupt, ett då en knapptryckning registreras och ett annat då klockan skall öka med en sekund.

De fyra funktionerna kallas A, B, C och D, och nås med respektive knapptryckningar enligt avsnitt 4.3 ”Knappsats”. A aktiverar larmet; B ställer in klockan; C visar vad klockan är; D berättar om larmet har utlösts och i så fall när och hur.

6. Genomförande

6.1. Planeringsfas

Under planeringsfasen fördes en diskussion i gruppen om vad för typ av projekt som upplevdes intressant att konstruera, vilka funktioner en framtida prototyp skulle kunna innehålla och hur tidsplanen skulle läggas upp för arbetet. Med utgångspunkt från en framtida prototyp och dess funktioner formulerades en kravspecifikation enligt avsnitt 3.2 ”Grundläggande krav”.

6.2. Förberedelsefas inför konstruktion

Innan konstruktion kunde påbörjas delades ansvarsområden upp sinsemellan samtliga medlemmar. Även om alla fick varsitt ansvarsområde fanns inga bestämda gränser för vem som gjorde vad utan alla var delaktiga i alla moment av arbetet. De tre ansvarsområdena inkluderar mjukvaru-, hårdvaru- samt rapportansvarig där den sistnämnda även inkluderar skapande av hemsida i html. En inventering av tillgängligt materialförråd gjordes för att säkerställa att önskade komponenter fanns tillgängliga. Ett kopplingsschema ritades på ett A3-papper för att underlätta vidare mjukvaru- och hårdvaruarbete och för att skapa sig en förståelse för kopplingen mellan samtliga komponenter.

6.3. Konstruktionsfas

Med hjälp av kopplingsschemat kunde både mjukvaru- och hårdvaruutvecklingen fortsätta. Byggandet av prototyp gjordes i enlighet med kopplingsschemat och varje komponent testades separat för att säkerställa att komponenter fungerade och var kopplade korrekt. Testning utfördes med hjälp av en logikpenna. Mjukvaruutvecklingen utfördes på LG-datorer i laborationssalar i E-huset. För djupare information angående mjukvaruutvecklingen var god se avsnitt 5 ”Mjukvara”.

6.4. Implementationsfas

Under implementationsfasen upptäcktes ett problem med realtidsklockan då denna komponent inte fungerade som planerat. Detta problem löstes genom att istället använda processorns interna realtidsklocka som vid testning fungerade enligt behov. Testning av prototyp enligt användarfallen gav positivt resultat förutom att problem uppstod vid test av temperatursensorer då det i praktiken var svårt att ändra temperaturen med tillgängliga hjälpmedel. Dock bör dessa fungera då de ger rimliga värden.

7. Diskussion och slutsats

Vi anser att hela arbetet har varit intressant men framför allt lärorikt på flera olika plan. Dels upplever vi det som att vi fått än större förståelse för komplexiteten i många av de teknikprodukter som vi stöter på i vardagen men även den nära kopplingen mellan hårdvaru- och mjukvaruutveckling. Men arbetsprocessen har inte fortskridit smärtfritt utan vi har flertalet gånger fått kämpa i motvind för att komma fram till en lösning. Exempelvis fick vi problem under ritningen av kopplingsschemat. Dels blev vi bestulna av/tappade bort vårt gamla kopplingsschema tragiskt nog. Men vi fick även problem hur vi skulle få alla portar på processorn att räcka till i och med alla komponenter vi valt ska ingå i larmsystemet. Men trots dessa motgångar och de motgångar vi stött på under programmeringsfasen lyckades vi till slut färdigställa vårt rörelse- och temperaturkänsliga larmsystem.

Innan starten av projektkursen hade vi bestämt att vi skulle bilda en grupp. Vi har alla arbetat tillsammans tidigare och tyckte att det kändes naturligt med arbets- och ansvarsfördelning även om samtliga medlemmar i gruppen var involverade i alla moment under arbetsprocessen.

Sammanfattningsvis så är vi nöjda med vår prototyp och hur vi lyckades tackla samtliga problem under arbetets gång, även om vi fick lite hjälp på traven ibland. Vi är övertygade om att projektet har gett oss en god förståelse för hur utvecklingsarbete och processer kan se ut ute i industrin och hur detta kan vara en fördel att förstå när vi möter på denna typ av arbete ute i näringslivet.

8. Bilagor

8.1. Källkod

```
/*
* GccApplication1.c
*
* Created: 2018-04-20 15:19:59
* Author : ine15jlu
*/

#define F_CPU 8000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#define greenDiode PA6
#define redDiode PA7
#define summer PC6
#define E_D PD6
#define RW_D PD1
#define RS_D PD0
#define RD_C PD4
#define WR_C PD5
#define ALE_C PD7
#define CS_C PC7


char sensor1 = 0;
char sensor2 = 0;
uint16_t temp1 = 0;
uint16_t temp2 = 0;
unsigned int larmStatus = 0;
unsigned int newInput = 0;
char val;
char pinCode[4];
char pinCompare[4];
unsigned int wrongPin = 0;
unsigned int sec = 0;
unsigned int min = 0;
unsigned int hour = 0;
unsigned int triggerIR1 = 0;
unsigned int triggerIR2 = 0;
unsigned int triggerTemp1 = 0;
```

```
unsigned int triggerTemp2 = 0;
unsigned int secTrig = 0;
unsigned int minTrig = 0;
unsigned int hourTrig = 0;
volatile uint16_t TimerOverFlowCount;


/* Instructions for reading the keypad */
char readKeypad(char code){
		if(code == 0b00000000){
				return '0';
				}else if(code==0b00000001){
				return '1';
				}else if(code==0b00000010){
				return '2';
				}else if(code==0b00000011){
				return '3';
				}else if(code==0b00000100){
				return '4';
				}else if(code==0b00000101){
				return '5';
				}else if(code==0b00000110){
				return'6';
				}else if(code==0b00000111){
				return '7';
				}else if(code==0b00001000){
				return '8';
				}else if(code==0b00001001){
				return '9';
				}else if(code==0b00001010){
				return 'A';
				}else if(code==0b00001011){
				return 'B';
				}else if(code==0b00001100){
				return 'C';
				}else if(code==0b00001101){
				return 'D';
				}else if(code==0b00001110){
				return 'E';
				}else if(code==0b00001111){
				return 'F';
		}
}

ISR(TIMER0_OVF_vect)
{
		//each time the timer0 overflows, this ISR will be executed to indicate timer0
overflow
		//Increment the counter value by this operation.
		TimerOverFlowCount++;

		if(TimerOverFlowCount>= 33250)
		{
				sec++;
		if (sec == 60) {
```

```
                        min++;
                        sec = 0;
                }
                if (min == 60) {
                        hour++;
                        min = 0;
                }
                if (hour == 24) {
                        hour = 0;
                }
                        TimerOverFlowCount=0; //reset the timerOverFlow count back to
0.
                }

}

int main(void)
{
                INIT();
                timerINIT();
                ADMUX |= 1<<REFS0;
                ADCSRA |= (1<<ADEN) | (0<<ADPS2) | (1<<ADPS1) | (0<<ADPS0);
                sei();
                val = 0xff;
                startDisplay();
                setDisplay();
                clearDisplay();
                writeWelcome();
                _delay_ms(2000);
                writeSelect();

                while (1)
                {
                        while(larmStatus){
                                alarmActive();
                        }

                        while(newInput == 1) {
                                char input = readKeypad(val);
                                newInput = 0;
                                if (input == 'A') {
                                        functionA();
                                }
                                if (input == 'B') {
                                        functionB();
                                }
                                if (input == 'C') {
                                        functionC();
                                }
                                if (input == 'D') {
                                        functionD();
                                }
                        }
                }
```

```
}

ISR(INT0_vect) {
		val = PINA & 0b00001111;
		newInput = 1;
}

ISR(INT1_vect) {
		sec++;
		if (sec == 60) {
				min++;
				sec = 0;
		}
		if (min == 60) {
				hour++;
				min = 0;
		}
		if (hour == 24) {
				hour = 0;
		}
}

/*Sätter initialtillstånden*/
void INIT(){
		DDRB = 0b11111111; // vi sätter databussen ut
		DDRA = 0b11000000; // vi sätter A6, A7 ut, resten in
		DDRD = 0b11110011; // vi sätter RS. RW, E ut
		DDRC = 0b11000000; // C6, C7 ut, resten in
		PORTD = 0b00001100; // vi sätter RS till 0 och E till 0 även interupt 1
		GICR = 0b11000000;
		MCUCR = 0b0001111;
}

void timerINIT(){
		TCNT0 = 0;
		TimerOverFlowCount= 0;

		//Enable the Timer0 overflow interrupt flag
		TIMSK|=(1<<TOIE0);

		//Start the Timer0
		TCCR0|= (1<<CS00); //since we don't want any prescaling.
}

void rtcINIT() {
		PORTC &= ~_BV(CS_C);
		PORTD |= _BV(WR_C);
		PORTD |= _BV(RD_C);

		PORTB = 0b00010001;

		PORTD |= _BV(ALE_C);
		_delay_us(5);
		PORTD &= ~_BV(ALE_C);
```

```
		PORTB = 0b00011100;

		PORTD &= ~_BV(WR_C);
		_delay_us(5);
		PORTD |= _BV(WR_C);

		PORTB = 0b00010000;

		PORTD |= _BV(ALE_C);
		_delay_us(5);
		PORTD &= ~_BV(ALE_C);

		PORTB = 0b00001000;

		PORTD &= ~_BV(WR_C);
		_delay_us(5);
		PORTD |= _BV(WR_C);
}

void waitForInput(){
		while(newInput == 0){
		}
}

void greenDiodeOn(){
		PORTA |= _BV(greenDiode);
}

void greenDiodeOff(){
		PORTA &= ~_BV(greenDiode);
}

void redDiodeOn(){
		PORTA |= _BV(redDiode);
}

void redDiodeOff(){
		PORTA &= ~_BV(redDiode);
}

void summerOn(){
		PORTC |= _BV(summer);
}

void summerOff(){
		PORTC &= ~_BV(summer);
}

uint16_t analogReadTemp1(){
		ADMUX = 0b11000100;
		_delay_ms(20);
		ADCSRA|=(1<<ADSC);
```

```
		while ( !(ADCSRA & (1<<ADIF)));
		ADCSRA|=(1<<ADIF);
		return (ADC);
}

uint16_t analogReadTemp2(){
		ADMUX = 0b11000101;
		_delay_ms(20);
		ADCSRA|=(1<<ADSC);
		while ( !(ADCSRA & (1<<ADIF)));
		ADCSRA|=(1<<ADIF);
		return (ADC);
}

void setDisplay(){
		PORTB = 0b00000010;
		PORTD |= _BV(E_D);
		PORTD &= ~_BV(E_D);
		_delay_us(100);
		PORTB = 0b00111100;
		PORTD |= _BV(E_D);
		PORTD &= ~_BV(E_D);
		_delay_us(100);
}

void startDisplay(){
		_delay_ms(100);
		PORTB = 0b00001101;
		PORTD |= _BV(E_D);
		PORTD &= ~_BV(E_D);
}

void clearDisplay(){
		PORTB = 0b00000001;
		PORTD |= _BV(E_D);
		PORTD &= ~_BV(E_D);
		_delay_ms(2);
}

void changeCursor(){
		char location = 0x40;
		if(location < 0b01111111){
				PORTB = 0b10000000 | location;
				PORTD |= _BV(E_D);
				PORTD &= ~_BV(E_D);
				_delay_us(100);
		}
}

void writeChar(char ch){
		PORTD |= _BV(RS_D); // Sätter RS hög
		PORTB = ch;
		_delay_us(100);
		PORTD |= _BV(E_D); // Sätter E hög
```

```
		_delay_us(100);
		PORTD &= ~_BV(E_D); // Sätter E låg
		PORTD &= ~_BV(RS_D); // Sätter RS låg
		_delay_us(100);
}

void writeWelcome(){
		clearDisplay();
		writeChar('W');
		writeChar('E');
		writeChar('L');
		writeChar('C');
		writeChar('O');
		writeChar('M');
		writeChar('E');
		writeChar('!');
}

void writeActivate(){
		clearDisplay();
		writeChar('A');
		writeChar('C');
		writeChar('T');
		writeChar('I');
		writeChar('V');
		writeChar('A');
		writeChar('T');
		writeChar('E');
		writeChar(' ');
		writeChar('A');
		writeChar('L');
		writeChar('A');
		writeChar('R');
		writeChar('M');
		writeChar(',');
}

void writeWrongCode(){
		clearDisplay();
		writeChar('W');
		writeChar('R');
		writeChar('O');
		writeChar('N');
		writeChar('G');
		writeChar(' ');
		writeChar('P');
		writeChar('I');
		writeChar('N');
		writeChar('!');
}

void writeActivated(){
		clearDisplay();
		writeChar('A');
```

```
                writeChar('C');
                writeChar('T');
                writeChar('I');
                writeChar('V');
                writeChar('A');
                writeChar('T');
                writeChar('E');
                writeChar('D');
}

void writeDeactivated(){
                clearDisplay();
                writeChar('D');
                writeChar('E');
                writeChar('A');
                writeChar('C');
                writeChar('T');
                writeChar('I');
                writeChar('V');
                writeChar('A');
                writeChar('T');
                writeChar('E');
                writeChar('D');
}

void writeEnterPin(){
                clearDisplay();
                writeChar('E');
                writeChar('N');
                writeChar('T');
                writeChar('E');
                writeChar('R');
                writeChar(' ');
                writeChar('P');
                writeChar('I');
                writeChar('N');
                writeChar(':');
                writeChar(' ');
}

void writeSelect(){
                clearDisplay();
                writeChar('S');
                writeChar('E');
                writeChar('L');
                writeChar('E');
                writeChar('C');
                writeChar('T');
                writeChar(' ');
                writeChar('F');
                writeChar('U');
                writeChar('N');
                writeChar('C');
                writeChar('T');
                writeChar('I');
```

```
                writeChar('O');
                writeChar('N');
                writeChar(':');
                changeCursor();
                writeChar('A');
                writeChar(',');
                writeChar(' ');
                writeChar('B');
                writeChar(',');
                writeChar(' ');
                writeChar('C');
                writeChar(' ');
                writeChar('O');
                writeChar('R');
                writeChar(' ');
                writeChar('D');


}

void writeCountdown(){
                clearDisplay();
                writeChar('5');
                _delay_ms(1000);
                clearDisplay();
                writeChar('4');
                _delay_ms(1000);
                clearDisplay();
                writeChar('3');
                _delay_ms(1000);
                clearDisplay();
                writeChar('2');
                _delay_ms(1000);
                clearDisplay();
                writeChar('1');
                _delay_ms(1000);
}

void writeSetTime(){
                clearDisplay();
                writeChar('S');
                writeChar('E');
                writeChar('T');
                writeChar(' ');
                writeChar('T');
                writeChar('I');
                writeChar('M');
                writeChar('E');
                writeChar(':');
}

void writeAlarmTrig(){
                clearDisplay();
                writeChar('A');
```

```
writeChar('L');
writeChar('A');
writeChar('R');
writeChar('M');
writeChar(' ');
writeChar('T');
writeChar('R');
writeChar('I');
writeChar('G');
writeChar('G');
writeChar('E');
writeChar('R');
writeChar('E');
writeChar('D');
changeCursor();
writeChar('A');
writeChar('T');
writeChar(' ');
displayTime(hourTrig, minTrig, secTrig);
_delay_ms(300);
waitForInput();
newInput = 0;
clearDisplay();
if(triggerIR1){
                writeChar('I');
                writeChar('R');
                writeChar('1');
                writeChar(':');
                writeChar('Y');
                writeChar('E');
                writeChar('S');
                } else {
                writeChar('I');
                writeChar('R');
                writeChar('1');
                writeChar(':');
                writeChar('N');
                writeChar('O');
                writeChar(' ');
}
writeChar(' ');
if(triggerIR2){
                writeChar('I');
                writeChar('R');
                writeChar('2');
                writeChar(':');
                writeChar('Y');
                writeChar('E');
                writeChar('S');
                } else {
                writeChar('I');
                writeChar('R');
                writeChar('2');
                writeChar(':');
                writeChar('N');
```

```
                writeChar('O');
        }
        changeCursor();
        if(triggerTemp1){
                writeChar('T');
                writeChar('1');
                writeChar(' ');
                writeChar(':');
                writeChar('Y');
                writeChar('E');
                writeChar('S');
                } else {
                writeChar('T');
                writeChar('1');
                writeChar(' ');
                writeChar(':');
                writeChar('N');
                writeChar('O');
                writeChar(' ');
        }
        writeChar(' ');
        if(triggerTemp2){
                writeChar('T');
                writeChar('2');
                writeChar(' ');
                writeChar(':');
                writeChar('Y');
                writeChar('E');
                writeChar('S');
                } else {
                writeChar('T');
                writeChar('2');
                writeChar(' ');
                writeChar(':');
                writeChar('N');
                writeChar('O');
        }
        _delay_ms(300);
        waitForInput();
        newInput = 0;
}

void writeNoTrig(){
        clearDisplay();
        writeChar('A');
        writeChar('L');
        writeChar('A');
        writeChar('R');
        writeChar('M');
        writeChar(' ');
        writeChar('W');
        writeChar('A');
        writeChar('S');
        writeChar(' ');
        writeChar('N');
```

```
		writeChar('O');
		writeChar('T');
		changeCursor();
		writeChar('T');
		writeChar('R');
		writeChar('I');
		writeChar('G');
		writeChar('G');
		writeChar('E');
		writeChar('R');
		writeChar('E');
		writeChar('D');
		_delay_ms(300);
		waitForInput();
		newInput = 0;
}

void displayTime(int h, int m, int s){
		unsigned int place1 = h / 10;
		unsigned int place0 = h - (place1 * 10);
		writeChar(readKeypad(place1));
		writeChar(readKeypad(place0));
		writeChar(':');
		place1 = m / 10;
		place0 = m - (place1 * 10);
		writeChar(readKeypad(place1));
		writeChar(readKeypad(place0));
		writeChar(':');
		place1 = s / 10;
		place0 = s - (place1 * 10);
		writeChar(readKeypad(place1));
		writeChar(readKeypad(place0));
}

void alarmActive(){
		sensor1 = PINC & 0b00000001;
		sensor2 = PINC & 0b00000010;
		temp1 = analogReadTemp1();
		temp2 = analogReadTemp2();
		triggerIR1 = 0;
		triggerIR2 = 0;
		triggerTemp1 = 0;
		triggerTemp2 = 0;
		if (sensor1 == 1 || sensor2 == 2 ||temp1 > 120 || temp2 > 120) {

			if(sensor1 == 1) {
				triggerIR1 = 1;
			}
			if(sensor2 == 2) {
				triggerIR2 = 1;
			}
			if(temp1 > 120) {
				triggerTemp1 = 1;
			}
			if(temp2 > 120) {
```

```
                                        triggerTemp2 = 1;
                                }
                                hourTrig = hour;
                                minTrig = min;
                                secTrig = sec;
                                summerOn();
                                while(larmStatus){
                                        deactivate();
                                }
                        }

                        if(newInput == 1) {
                                char input = readKeypad(val);
                                newInput = 0;
                                if (input == 'A') {
                                        deactivate();
                                }
                        }
}

void deactivate(){
                                writeEnterPin();
                                _delay_ms(300);
                                waitForInput();
                                newInput = 0;
                                char pin = readKeypad(val);
                                pinCompare[0] = pin;
                                writeChar(pin);
                                for(int i = 1; i < 4; i++){
                                        waitForInput();
                                        newInput = 0;
                                        char pin = readKeypad(val);
                                        pinCompare[i] = pin;
                                        writeChar(pin);
                                }
                                for(int i = 0; i < 4; i++){
                                        if (pinCompare[i] != pinCode[i]){
                                                wrongPin = 1;
                                        }
                                }
                                _delay_ms(1000);
                                if(wrongPin){
                                        wrongPin = 0;
                                        writeWrongCode();
                                        _delay_ms(1000);
                                        clearDisplay();
                                } else {
                                        larmStatus = 0;
                                        summerOff();
                                        redDiodeOff();
                                        greenDiodeOn();
                                        writeDeactivated();
                                        _delay_ms(1000);
                                        writeSelect();
                                }
```

```
}

void functionA(){
		writeActivate();
		changeCursor();
		writeChar('P');
		writeChar('I');
		writeChar('N');
		writeChar(':');
		writeChar(' ');
		for(int i = 0; i < 4; i++){
			waitForInput();
			newInput = 0;
			char pin = readKeypad(val);
			pinCode[i] = pin;
			writeChar(pin);
		}
		_delay_ms(1000);
		writeCountdown();
		larmStatus = 1;
		greenDiodeOff();
		redDiodeOn();
		writeActivated();
		_delay_ms(1000);
		clearDisplay();
}

void functionB(){
		writeSetTime();
		_delay_ms(300);

		waitForInput();
		clearDisplay();
		writeChar('H');
		writeChar('O');
		writeChar('U');
		writeChar('R');
		writeChar(':');
		writeChar(' ');
		newInput = 0;
		char pin = readKeypad(val);
		unsigned int place1 = val;
		writeChar(pin);
		waitForInput();
		newInput = 0;
		pin = readKeypad(val);
		unsigned int place0 = val;
		writeChar(pin);
		hour = place1*10 + place0;

		waitForInput();
		clearDisplay();
		writeChar('M');
		writeChar('I');
```

```
		writeChar('N');
		writeChar(':');
		writeChar(' ');
		newInput = 0;
		pin = readKeypad(val);
		place1 = val;
		writeChar(pin);
		waitForInput();
		newInput = 0;
		pin = readKeypad(val);
		place0 = val;
		writeChar(pin);
		min = place1*10 + place0;

		waitForInput();
		clearDisplay();
		writeChar('S');
		writeChar('E');
		writeChar('C');
		writeChar(':');
		writeChar(' ');
		newInput = 0;
		pin = readKeypad(val);
		place1 = val;
		writeChar(pin);
		waitForInput();
		newInput = 0;
		pin = readKeypad(val);
		place0 = val;
		writeChar(pin);
		sec = place1*10 + place0;
		_delay_ms(1000);
		writeSelect();

}

void functionC(){
		while (!newInput){
			clearDisplay();
			displayTime(hour, min, sec);
			_delay_ms(100);
		}
		newInput = 0;
		writeSelect();
}

void functionD(){
		if(triggerIR1 || triggerIR2 || triggerTemp1 || triggerTemp2) {
			writeAlarmTrig();
		} else {
			writeNoTrig();
		}
		writeSelect();
}
```

8.2. Litteraturreferenser

Kernighan, B.W. & Ritchie D.M. (1988). ***The C Programming Language***. 2. uppl. Pearson