PULSMÄTARE

EITF11 - DIGITALA PROJEKT - VT15

Grupp 9: Emma Albertz, Mathilde Hagander & Alexandra Mansner
Handledare: Andreas Johansson & Bertil Lindvall

Abstract

As part of the course Digital Projects EITF11 at the Faculty of Engineering, Lund University, a prototype of a pulse monitor has been constructed. As the user applies a finger over two diodes and presses the button the user's pulse is measured. This report describes the process for reaching the final result. The underlying work for this report consists of three main parts. Firstly, research was made on how a human pulse could be registered through the finger tip, followed by the construction of hardware. Finally, the software was developed. In this report the theory behind the pulse monitor and its functions are described, as well as the construction and implementation of hardware and software. The result and its strengths and weaknesses are also discussed.

Contents

1. Inledning

I detta projekt konstrueras en pulsmätare och projektet ingår i kursen EITF11 Digitala projekt. Syftet med projektet är att få ökad kunskap i hur konstruktion av prototyper fungerar och hur dess tillhörande dokumentation tas fram.

1.1 Bakgrund

En pulsmätare mäter användarens puls, det vill säga antal hjärtslag per minut. Pulsen beskriver hur hårt hjärtat måste jobba för att för att förse kroppen med syre. Bland annat ålder samt fysisk och psykisk ansträngningsgrad påverkar pulsen. Ju hårdare kroppen arbetar desto mer energi krävs och desto högre är pulsen. Pulsen kan vara intressant i flera olika sammanhang, exempelvis för att bestämma energiförbrukning, arbetsintensitet och konditionsförbättring.

1.2 Syfte

Syftet med projektet är att bygga en pulsmätare som med hjälp av användarens finger ska kunna meddela användaren om dess puls.

2. Produktbeskrivning

Teorin som pulsmätaren grundar sig på är att infrarött ljus reflekteras i människans blodceller. Användaren placerar sitt finger över både en diod som emitterar infrarött ljus in i fingret och en fotodiod som tar upp infrarött ljus. Dioderna ligger tätt intill varandra så att fotodioden kan fånga upp det infraröda ljus som reflekteras i blodcellerna i användarens finger. Ju fler blodceller användaren har i fingret vid ett visst ögonblick, desto mer ljus reflekteras in i fotodioden. Detta resulterar i att användarens puls representeras av mängden ljus som fotodioden fångar upp. Pulsmätaren ska sedan kunna avgöra när en viss mängd ljus motsvarade ett pulsslag, vilket görs med hjälp av en Schmitt-Trigger som beskrivs senare i rapporten.

Både IR-dioden och fotodioden är omslutna av mörkt gummi. Detta för att allt ljus från IR-dioden ska koncentreras till att emitteras in i fingret och för att risken att fotodioden får ljus från andra riktningar än från reflektioner i fingret ska minimeras.

3. Kravspecifikation

- Skärmen ska vara på och tömd när ström kopplas in.
- Dioden som emitterar infrarött ljus ska sättas på när ström kopplas in.
- En pulsräknare ska räkna antalet pulser som registreras.
- Röd LED ska lysa då ett pulsslag registreras.
- Vid knapptryck ska en mätning påbörjas.
- Vid knapptryck ska grön LED sättas på.
- Vid knapptryck ska displayen tömmas.
- Vid knapptryck ska pulsräknaren nollställas.
- En mätning ska vara tio sekunder lång.
- När mätningen avslutas ska grön LED släckas.
- Antal slag per minut ska visas på displayen vid mätningens slut.

4. Hårdvara

De olika komponenterna kopplades ihop genom lödning och virning. Se bilaga 1-4 för produktens kopplingsscheman.

4.1 Processor

Processorn som används i pulsmätaren är ATmega16 8-bitars mikrokontroller med ett 16 kb flashminne och 40 I/O pinnar, varav 32 finns till förfogande för programmering. Se bilaga 4 för processorns kopplingsschema.

4.1.1 Port A

Används för att skicka databitar till skärmen.

4.1.2 Port B

Används för att ta emot pulsslagen från pulsräknaren och för att skicka instruktioner till grön LED.

4.1.3 Port C

I/O pinnarna C2-C5 är reserverade för JTAG. I övrigt används porten för att ta emot instruktioner från knappen.

4.1.4 Port D

Används för att skicka instruktioner från processorn till skärmen som påverkar hur skärmen behandlar de databitar som skickas på port A.

4.2 Display

Displayen som används för att skriva ut användarens puls är en LCD alfa-numerisk teckendisplay vid namn SHARP Dot-Matrix.

4.3 Operational Amplifier

Två OP-förstärkare används för att förstärka pulssignalen så att den blir observerbar. Förstärkarna TL064CN och LM324N utnyttjas för att få olika förstärkningsnivåer beroende på var de placeras i kretsen.

4.4 Schmitt-Trigger

En Shcmitt-Trigger med namn 74HC14 används för att omvandla den vågformade pulsen till pulser i form av en etta eller nolla. När den vågformade pulsen överstiger 2,5 V skickar Schmitt-Triggern en etta och när pulsen understiger 1,6 V skickar den en nolla.

4.5 Potentiometer

En potentiometer används för att reglera förstärkningen av den vågformade pulsen. Den har reglerats noggrant så stor så att den överstiger Schmitt-Triggerns gräns på 2,5V vid ett pulsslag, men samtidigt så liten så att endast den större vågformen i ett pulsslag överstiger 2,5V och inte den mindre "förpulsen". Förstärkningen är nu reglerad så att den har fungerat bra för ett tiotal olika personer.

4.6 Knapp

En knapp används för att användaren ska kunna bestämma när en mätning ska börja. På så sätt kan användaren prova pulsmätaren och se till att pulsen är regelbunden genom att observera den röda LED:en innan mätningen påbörjas. Med knappen kan även mätningens timer, som är inställd på tio sekunder, startas.

4.7 LED

En röd LED vid namn TLLR5400 utnyttjas för att visa när ett pulsslag registreras. En grön LED vid namn TLLG5400 används för att visa när en mätning pågår.

4.8 Emitterande IR-diod

Dioden som används för att emittera infrarött ljus är TSUS5400.

4.9 Fotodiod

Fotodioden som används för att ta upp infrarött ljus är SFH203FA.

4.10 JTAG

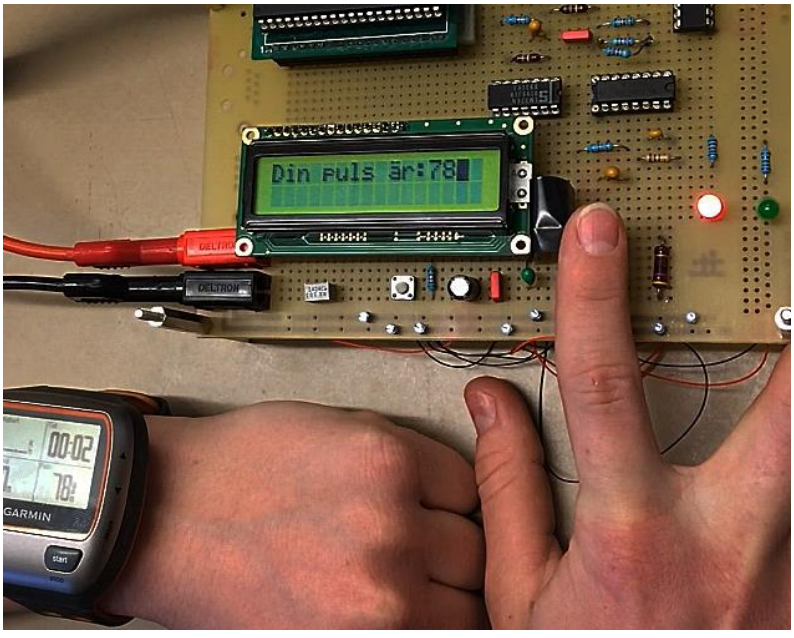
Joint Test Action Group som används för att koppla upp sig mot processorn och kunna exekvera och ladda mjukvara.

4.11 Voltregulator

En voltregulator med namn L7805ACV används för att omvandla inkommande spänning på 8V till 5V. Detta för att säkerställa att det alltid är precis 5V som kommer in i kretsen. Se bilaga 1 för kopplingsschema.

4.12 Övrigt

För att få konstruktionen att fungera som önskat används även flertalet resistorer och kondensatorer. Se bilaga 1-4.



5. Mjukvara

Huvudprogrammet består av en uppstartsfas, en while-loop som exekverar metoder ifall knappen tryckts ner samt ett avbrott. Se bilaga 5 för källkod.

5.1 Uppstartsfas

I uppstartsfasen ställs först processorns olika DDR-portar in beroende på om det är signaler som ska tas emot eller skickas ut. Samtliga portar anges som utgående förutom den kopplad till knappen vars signaler ska tas emot. Displayen sätts på, töms på innehåll och dess markör aktiveras. Avbrott tillåts genom metoden sei(). Timern ställs in enligt normala portoperationer för en 16-bit timer, tilldelas ett startvärde och ställs in så att "overflow interrupts" tillåts. Räknaren ställs in så att den läser in signaler som kommer från pulsräknaren för en 8-bit timer och nollställs.

5.2 While-loop

En oändlig while-loop används för att ständigt kontrollera om knappen är nedtryck eller inte. När knappen trycks ner anropas en metod som rensar displayen och tänder den gröna LED:en. Samtidigt startas både timern och räknaren.

5.3 Avbrott

Efter tio sekunder genereras ett avbrott i form av ett "overflow interrupt". Då multipliceras räknarens värde med sex och sparas som användarens puls. Denna skrivs ut på displayen i form av "Din puls är: XXX". Grön LED släcks och timern och räknaren återställs. Efter detta är systemet redo för en ny mätning ifall knappen trycks ner på nytt.

6. Metod

6.1 Planering

Projektgruppen samlades i början av kursen och lade upp en plan för hur projektet skulle framskrida. Alla gruppmedlemmar var överens om att först och främst bestämma vilken prototyp som skulle konstrueras och sedan komma igång med arbetet så fort som möjligt för att undvika stress i slutet av kursen. Det bestämdes även att arbetsfördelningen i gruppen skulle vara så jämn som möjligt och alla att skulle vara insatta i alla delar av arbetet.

6.2 Hårdvarukonstruktion

Efter att ha bestämt att en pulsmätare skulle byggas konstruerades ett kopplingsschema. Eftersom själva pulsmätaren inte var en färdig komponent utan den skulle konstrueras från grunden användes en preliminär kopplingsplatta. På detta sätt var det lättare att prova sig fram samt lägga till och ta bort komponenter under arbetets gång.

Ett oscilloskop användes för att till en början kunna se att signalen överhuvudtaget var en puls. Denna del av arbetet var den mest tidskrävande eftersom det var svårt att få en stabil pulssignal och förstärka den tillräckligt mycket för att processorn skulle kunna bearbeta den. Efter att

signalen ansågs acceptabel användes oscilloskopet för att ställa in Schmitt-Triggern till att omvandla pulssignalen till en rektangelpuls som sedan skulle skickas in till processorn.

När pulsmätaren fungerade som den skulle flyttades komponenterna från den preliminära kopplingsplattan till en annan kopplingsplatta där komponenterna kopplades samman med hjälp av lödning och virning. Slutligen, för att se så att hårdvaran fungerade testades konstruktionen sedan i programmet AtmelStudio.

6.3 Mjukvarukonstruktion

När hårdvaran fungerade som den skulle påbörjades kodskrivandet. Koden skrevs i programmet AtmelStudio och programmeringsspråket som användes var C-kod. Koden fördes över från datorn till processorn med hjälp av JTAG:en. Under arbetets gång genomfördes tester för att kontrollera att koden var korrekt. Debuggern i AtmelStudio användes för att upptäcka fel i koden samt var felen hade uppstått.

När knappen på pulsmätaren trycks ner ska pulsmätningen börja och efter 10 sekunder ska pulsen skrivas ut på skärmen. En 1024 prescaler används för att bryta ner grundfrekvensen på 4 MHz till cirka 3,9 kHz. Genom att ange timerns startvärde som det decimala värdet 26 473 tar det 10 sekunder att med denna frekvens nå timerns maxvärde på 65 535. Då genereras ett avbrott i form av ett "overflow interrupt".

7. Resultat och diskussion

Projektet resulterade i en pulsmätare som uppfyller samtliga krav i kravspecifikationen och som visar en närmast exakt puls vid jämförelse med andra pulsklockor. Däremot påverkas pulsmätarens prestanda mer än förväntat av hur fingret placeras över dioderna och av rörelser under mätningen. För bästa resultat bör fingret inte ligga med hårt tryck över dioderna och användaren bör fokusera på att sitta still under mätningens gång.

Både en styrka och en svaghet med pulsmätaren är att en mätning är 10 sekunder lång. Detta innebär att användaren inte behöver sitta en hel minut för att få se sin puls på displayen. Den kortare tiden minskar även risken för att felaktiga rörelser registreras som pulsslag under mätningens gång. Dock innebär en kortare mättid på 10 sekunder att pulsen måste beräknas genom att multipliceras med sex. Det medför att pulsen som visas på displayen alltid kommer vara ett tal i sexans multiplikationstabell och inte kan bli mer exakt än så. Det leder även till att om en störning uppstår kommer detta få större påverkan på pulsens värde. En längre mättid medför däremot en ökad risk för att störningar uppstår men får inte lika stor påverkan på slutresultatet. I detta projekt prioriterades användarvänligheten och minimering av risken för störningar vilket är varför den kortare mättiden har valts.

Den största svårigheten under projektet var konstruktionen av hårdvaran som krävde mer efterforskning, komponenter och tid än förväntat. Då målet med pulsmätaren var att den skulle kunna mäta en verklig persons puls med hjälp av fingertoppen och inte mäta en simulerad puls blev hårdvarukonstruktionen en stor del av projektet som helhet och krävde många omkopplingar, tillkommande komponenter och felsökningar. Då projektgruppen ursprungligen

hade lite kunskap på området kring hårdvarukonstruktion blev denna del av projektet, utöver den mest utmanande, också den mest lärorika.

Även mjukvarukonstruktionen var en utmaning då tidigare erfarenhet av att läsa och förstå datablad samt att programmera i C saknades. Med erfarenhet av javaScript och med hjälp av litteratur, handledare och webben blev utmaningen överkomlig.

8. Slutsats

Projektet har givit en inblick i hur mycket arbete som ligger bakom utvecklandet av nya produkter och en bättre förståelse för utvecklingsarbetets gång. Det har ökat förståelsen för hur mycket som kan gå fel och därmed vikten av att göra arbetet med felsökning effektivt. Detta eftersom projektet till stor del bestod av att bygga den analoga komponent som mäter pulsen och denna process var mycket svårare än förutspått. Ibland var det svårt att avgöra om komponenterna som valts inte var tillräckligt bra eller om någon felkoppling hade gjorts. Vid felsökning användes till största delen oscilloskop och voltmeter.

Lärdomar som gruppens medlemmar kommer ta med sig är tidskomplexiteten och det mångsidiga arbete som ligger bakom framtagningen av en produkt. Trots en del hinder på vägen så lyckades gruppen med att ta fram en pulsmätare som fungerar enligt de krav som ställts. Vägen dit var långt ifrån spikrak och mycket hjälp från handledare har i perioder krävts. Även om det vissa dagar känts mindre hoppfullt har arbetet till största del varit väldigt roligt.

9. Källförteckning

AVR Libc 1.6.7: http://www.eit.lth.se/fileadmin/eit/courses/edi021/avr-libc-user-manual/index.html

Datablad för ATMega16
http://www.eit.lth.se/fileadmin/eit/courses/edi021/datablad/Processors/ATmega16.pdf

Datablad för SHARP Dot-Matrix
http://www.eit.lth.se/fileadmin/eit/courses/edi021/datablad/Display/LCD.pdf

Datablad för OP-förstärkare TL064CN
http://www.st.com/web/en/resource/technical/document/datasheet/CD00000487.pdf

Datablad för OP-förstärkare LM324N
http://www.eit.lth.se/fileadmin/eit/courses/edi021/datablad/Analog/opamp/lm124.pdf

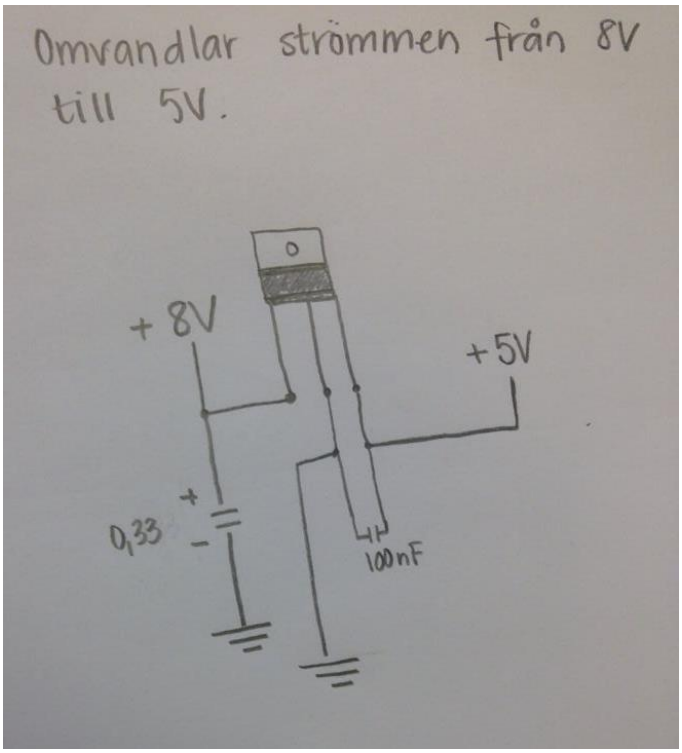
Datablad för Schmitt-Trigger 74HCT14
http://www.nxp.com/documents/data_sheet/74HC_HCT14.pdf

Datablad för voltregulator L7805ACV
http://www.farnell.com/datasheets/1805459.pdf

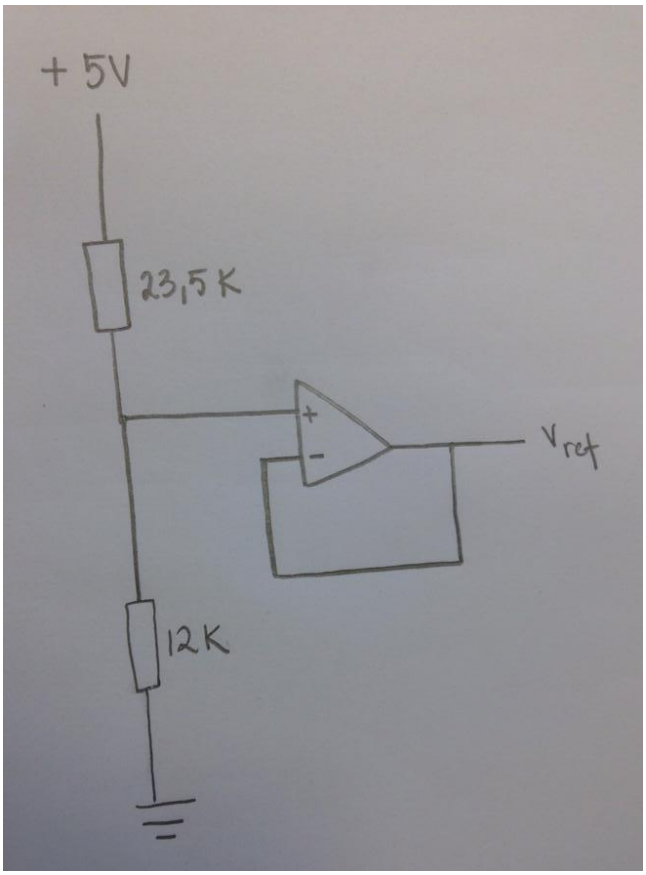
Pettersson, Gertrud. 1994. Råd och riktlinjer för rapportskrivning vid linjerna D, E och F. http://www.eit.lth.se/fileadmin/eit/courses/edi021/PDF_files/rad.pdf (Hämtad 2015-04-22).

10. Bilagor

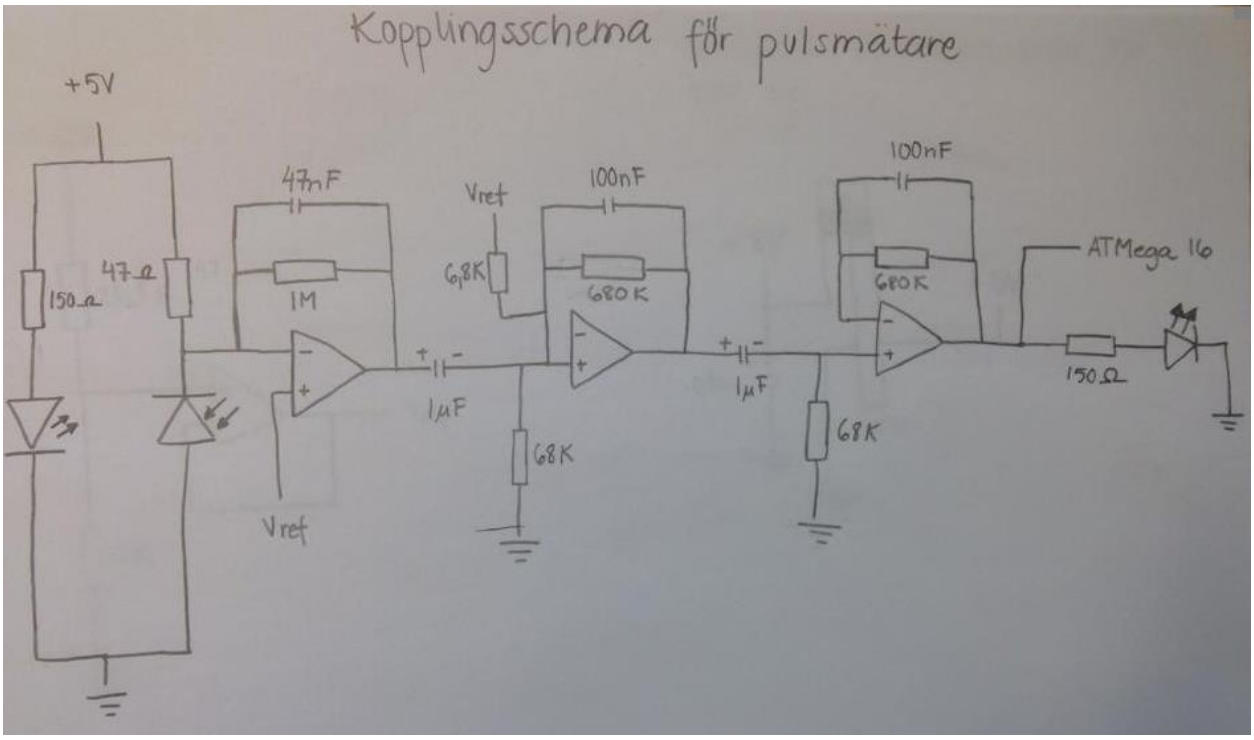
10.1 Bilaga 1 – Kopplingsschema omvandlare



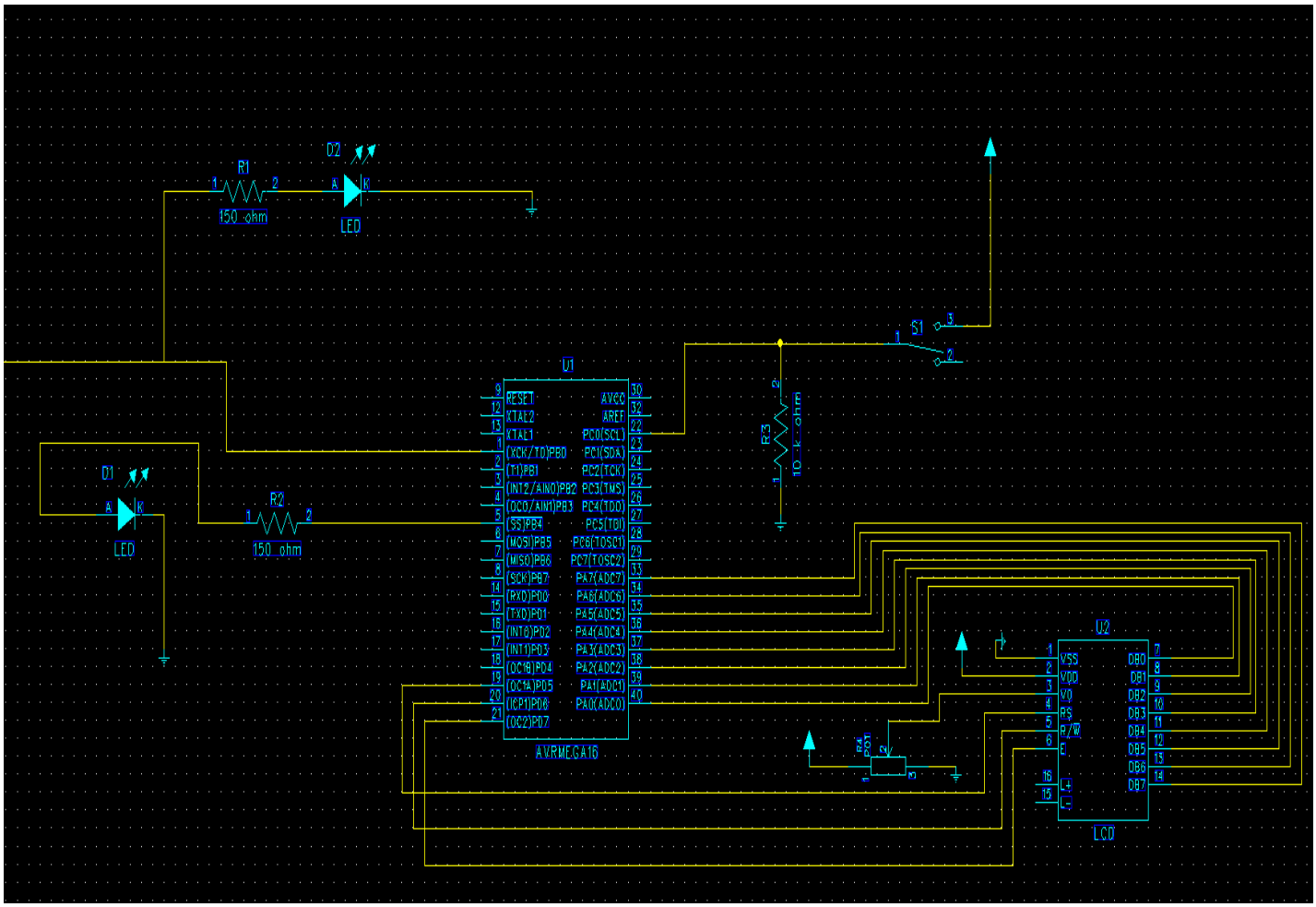
10.2 Bilaga 2 – Kopplingsschema V_{ref}



10.3 Bilaga 3 – Kopplingsschema pulsmätare



10.4 Bilaga 4 – Kopplingsschema processor



10.5 Bilaga 5 – Källkod

```
/*
 * Pulsmatare.c
 *
 * Created: 2015-04-13 14:33:40
 *  Author: digpi09
 */

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
unsigned char letter;
unsigned int pulse;

int checkButton()
{
	if(PINC & (1 << 0)) { //if button is pressed
		return 1;
	}
	return 0;
}

void pulseCountReady()
{	//Installations for 8-bit timer on T0:
	TCCR0 = 0x00; //Stops counter
	TCNT0 = 0xFF; //Setting 8-bit counter to read mode.
	TCNT0 = 0x00; //Clears counter

}

void resetCounter()
{
	TCCR0 = 0x00; //Stops counter
	TCNT0 = 0x00; //Clears counter
}

void timerReady()
{	//Installations for 16-bit timer on T1:
	TCCR1A = 0x00; //Normal port operation for 16-bit timer
	TCCR1B = 0x00; //Stops timer
	TCNT1H = 0x67; //Setting the 16-bit timer to 26473 (bit 8-15)
	TCNT1L = 0x69; //Setting the 16-bit timer to 26473 (bit 0-7)
	TIMSK = 0x04; //Enabling overflow interrupt
	TIFR = 0x04; //Enabling overflow interrupt
}

void resetTimer()
{
	TCCR1B = 0x00; //Stops timer
	TCNT1H = 0x67; //Setting the 16-bit timer to 26473 (bit 8-15)
	TCNT1L = 0x69; //Setting the 16-bit timer to 26473 (bit 0-7)
}
```

```
void startTimer()
{
	TCCR1B = 0x05; //Enables clock source, sets prescaler to 1024
}

void startCounter()
{
	TCCR0 = 0x07; //Starts counter
}

ISR(TIMER1_OVF_vect)
{	//When interruption occurs after 10 seconds:
	pulse = TCNT0*6; //Saves pulse per minute
	PORTB &= ~_BV(PB4); //Turns off green LED
	resetTimer();
	resetCounter();
	pulseToDisplay(pulse);
}

void numberHandler(int number)
{
	if(number == 0){
		writeLetter(0x30);
	}
	if(number == 1){
		writeLetter(0x31);
	}
	if(number == 2){
		writeLetter(0x32);
	}
	if(number == 3){
		writeLetter(0x33);
	}
	if(number == 4){
		writeLetter(0x34);
	}
	if(number == 5){
		writeLetter(0x35);
	}
	if(number == 6){
		writeLetter(0x36);
	}
	if(number == 7){
		writeLetter(0x37);
	}
	if(number == 8){
		writeLetter(0x38);
	}
	if(number == 9){
		writeLetter(0x39);
	}
}
```

```
void pulseToDisplay(int pulse)
{	//Writes pulsemessage and pulse on display
	pulseMessage(); //Writes pulsemessage "Din puls är:"
	int ten;
	int temp = pulse/100;
	if (temp > 0 ) {
		numberHandler(temp); //Writes the first number of pulse if it exists
		ten = pulse -100*temp;
	} else {
		ten = pulse;
	}
	temp = ten/10;
	numberHandler(temp); //Writes the first/(second) number of pulse
	temp = ten - temp*10;
	numberHandler(temp); //Writes the second/(third) number of pulse
}

void clear()
{
	//Clear display and return cursor to home
	EHigh();
	PORTA = _BV(PA0);
	ELow();
}

void writeReady()
{
	//Prepare to write text
	PORTD = PORTD | _BV(PD7); //Sets E to 1
	PORTD = PORTD | _BV(PD5); //Sets RS to 1
}

void writeLetter(char letter)
{
	//Writes a letter
	writeReady();
	PORTA=letter;
	ELow();
}

void pulseMessage()
{
	//Writes text before pulsenumber: "Din puls är:"
	writeLetter(0x44); //D
	_delay_us(10);
	writeLetter(0x69); //i
	_delay_us(10);
	writeLetter(0x6E); //n
	_delay_us(10);
	writeLetter(0x20); //
	_delay_us(10);
	writeLetter(0x70); //p
	_delay_us(10);
	writeLetter(0x75); //u
	_delay_us(10);
	writeLetter(0x6C); //l
	_delay_us(10);
	writeLetter(0x73); //s
```

```
	_delay_us(10);
	writeLetter(0x20); //
	_delay_us(10);
	writeLetter(0xE1); //ä
	_delay_us(10);
	writeLetter(0x72); //r
	_delay_us(10);
	writeLetter(0x3A); //:
	_delay_us(10);
}

void ELow()
{
	PORTD &= ~_BV(PD7); //Sets E to 0
}

void EHigh()
{
	PORTD = _BV(PD7); //Sets E to 1
}

void RSLow()
{
	PORTD &= ~_BV(PD5); //Sets RS to 0
}

void RSHigh()
{
	PORTD = _BV(PD5); //Sets RS to 1
}

void RWLow()
{
	PORTD &= ~_BV(PD6); //Sets RW to 0
}

void RWHigh()
{
	PORTD = _BV(PD6); //Sets RW to 1
}

void systemOn()
{	//Installations for the system
	DDRA = 0xFF; //Sets all to outgoing
	DDRD = 0xE0; //Sets D5-D7 to outgoing
	DDRB = 0x10; //Sets B4 to outgoing
	DDRC &= DDRC | ~_BV(PC0); //Sets C0 to ingoing
	EHigh();
	PORTA = 0x38; //Function Set on
	ELow();
	EHigh();
	PORTA = 0x0F; //Cursor on
	ELow();
	sei(); //Enables interruptions
	timerReady();
	pulseCountReady();
	clear();
}
```

```
int main(void)
{
systemOn();

	while(1) {
		if(checkButton()==1) {
			clear(); //Clears display
			PORTB = PORTB | _BV(PB4); //Turns green LED on
			startTimer(); //Starts timer
			startCounter(); //Starts counter
		}
	}
}
```