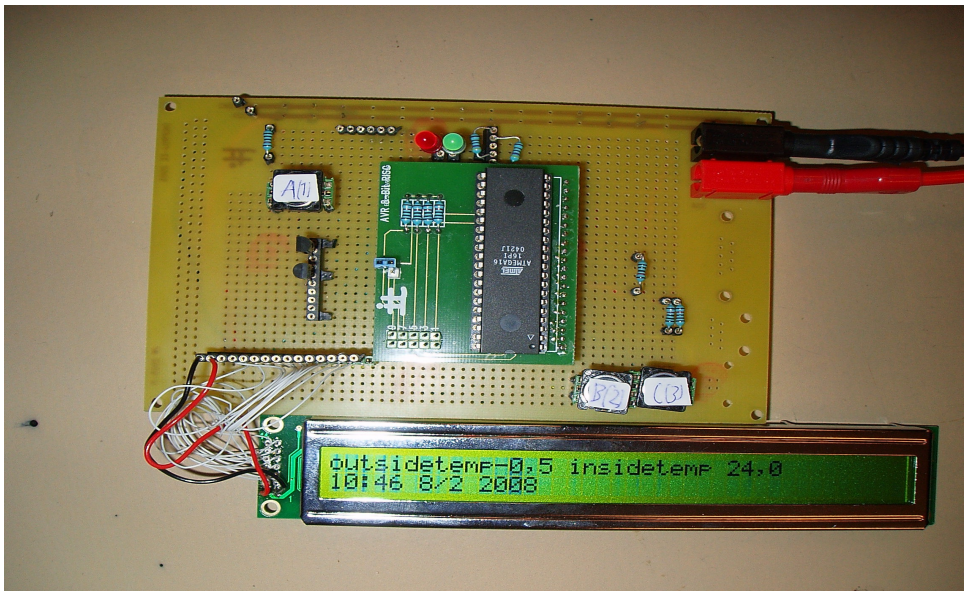
Digitala projekt
VT1 2007
Väderstationen

Av: Andreas Cremon E04
grupp 15

26/2-2007



Abstract:

Today's weatherstations with plugin features are expensive and doesn't have expansion possibility's
that I want. The logical solution to the problem was to build one that had the features and precision that was desirable. A good start for this project was to have a expandable base to build on witch was provided by the course “Digitala projekt” that had the Atmel AVR Atmega16 as one of the recommended microprocessor. The base of the weahterstation that was built around the mega16 has the ability to read outside and inside temperature and present it on a lcd-display and it also has the ability to present max and min temperture (with date) further more it has the possibility to expand to a numerous different devices such as pressuresensor and vindsensor.

Innehållsförteckning:

Bilagor:

Inledning:

Intresset för väder och vind är något som är väldigt stort i Sverige och därmed också ett stort intresse för mätapparatur inom vädersektorn. Väderstationer med grundutrustning och möjlighet att expandera antalet sensorer är oftast omotiverat dyra. För att komma förbi den höga kostnaden så bestämde jag mig för att bygga en väderstation med många expansionsmöjligheter.

Detta grundades på en Atmel AVR Atmega16 samt en lcd-display med 40x2 tecken vilket ger en bas som är kraftfull nog för att hantera en mångfald av sensorer minst lika många som de avancerade väderstationer som säljs i dagsläget. Den stora displayen ger möjligheter att skiva ut mycket data på samma gång och det är en klar fördel i expansionsynpunk.

Eftersom tiden på projektet inte påbjuder så kommer inte alla expansionsenheter kunna färdigställas utan bygget får koncentreras till att färdigställa den bas de ska dockas i.
Under bygget kommer jag dock att ha under åtanke att det kan komma att anslutas fler enheter i framtiden.

2 Metod:

Det första som gjordes var att en kravspecifikation gjordes *(bilaga 1)* efter det så var nästa naturliga steg att välja komponenter vilka listas under rubriken hårdvara. När komponenterna var bestämda ritades ett kretsschema över hur alla komponenter skulle placeras för att få en fungerande konstruktion detta gjorde i programmet Power logic vilket visas i *(Figur 2.1)*.
Vidare så realiserades denna konstruktion med hjälp av lödning och virning på ett kopplingskort med öar.
När hårdvarudelen var färdigställd återstod endast mjukvara vilket skrivs med hjälp av programmet
AVR studio 4 i programmeringsspråket C tillvägagångsätter beskrivs under rubriken Mjukvara nedan.

(Figur 2.1)

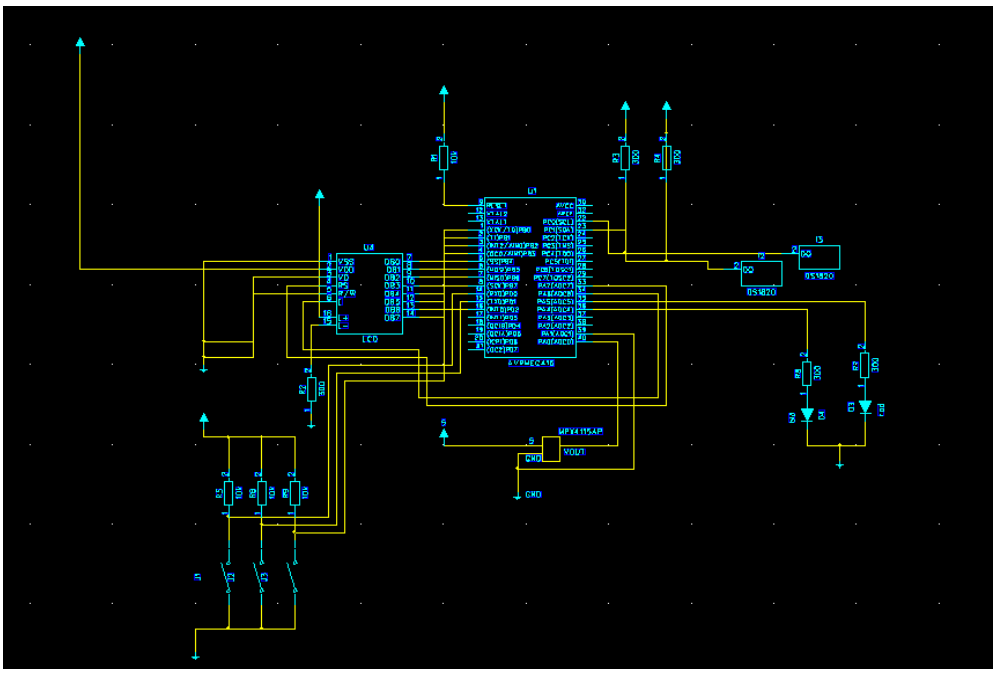
2.1 Hårdvara:

- DS18s20 är den temperatursensor som används det är en digital temperatursensor där kommunikationen med endast en tråd vilket kan vara lämpligt när sensorn skall placeras utomhus och det då krävs en längre kabel som ej bör vara för tjock.
  Förutom att det kan vara en fördel med att det inte behövs så mycket kabel så är det en fördel i jämförelse med analoga sensorer att den analoga signalen ej färdas någon längre sträcka och där med minimeras störningarna som kan uppkomma t.ex. vid analoga mätningar där en temperatursensor är fäst på en lång kabel. Sensorna har en upplösning på 0,5 grader och ett fel på ±0,5 grader i intervaller -10 till +85.

- Atmel AVR Atmega16 blev den giva processorn då denna var den minsta processor som erbjöds och uppgiften inte krävde så mycket processorkraft.

- En 40x2 teckens lcd-display användes mest på grund av att det fanns en sådan tillgänglig samt att det är en bra display om utökning av väderstatioen sker.

  Utöver de delar som nämns ovan användes ett antal resistorer samt två lysdioder för markering av understigna gränsvärden.
  Det färdiga resultatet av hårdvaruuppkopplingen blev enligt *(Figur 2.1.1)*



(Figur 2.1.1)

2.2 **Mjukvara:**

- Klockan realiserade genom att det genererades ett interupt varje sekund och då kördes en kod som uppdaterade klockan genom att räkna upp sek och om det behövdes min,timme,dag och år.

- LCD-displayen styrs med 8 st pinnar för data Enable och PS eftersom dessa pinnar inte kommer att användas till något speciellt även fast väderstationen expanderar så programmerades de på fasta pinnar.

- Delar av koden för att styra DS18s20 togs färdigt då denna kod är onödig att ha som anledning till att försena projektet då det mest består av att kod och det är lätt att få tag på delar av sådan kod.

Diskussion och resultat:

Huvudsyftet med projektarbete att skapa en bas för framtida expandering uppnåddes helt klart. Förändringar av funktioner och utseende gjordes under projektets gång vilket enligt mig är en naturlig gång för att kunna utveckla en väl fungerande produkt då ingen är fullkomlig och förändringar kompenserar för detta. Det enda som jag anser hade kunnat vara bättre är upplösning och noggrannheten på de digital sensorerna DS18s20 det skulle kunna göras med en pt-100 eller likande analog variant dock kommer det in problem med mätstörningar och upplösningen på A/D omvandlingen samt att det blir en mer avancerad uppkoppling.
De delar som kan sättas på i framtiden är tycksensorn som är hårdvaruförbredd samt en vindmätare som jag hade tänkt konstruera med hjälp av tre ultraljudselement som placeras i en triangel och fungerar genom att en sänder ut ljud och och sen tas tiden tills de andra två har tagit emot ljudet då ljudet går snabbare hade det medvind till ultraljudselementet och vise versa. De tre olika elementen turas om att vara sändare vilket ger mätvärden som anger vindhastighet och rikting i 3 riktiningar. Med hjälp av de datan kan man ge en exakt vindrikting och hastighet.

Appendix A C koden:

vader.c

```
#include <inttypes.h>
#include <avr/io.h>
#define F_CPU 8000000UL
#include <util/delay.h>
#include <stdlib.h>
#include "1wiremin.h"
#include <ctype.h>
#include <avr/interrupt.h>

void setKlocka();
int klocka=0;                           // koll om klockan är inställd
        int sec=0;
        int min=0;                              // vilken minut
        int hour=0;                             //vilken timme
        int day=1;                              // vilken dag i månaden
        int days=1;                             // vilken dag på året 1-365
        int month=1;                    // vilken månad
        int year=2007;                  // vilket år 2007 och framåt

        double inmaxtemp=0;
        int     inmaxmin=0;
        int inmaxhour=0;                        //vilken timme maxvärdet inträffar
        int inmaxday=0;                         // vilken dag i månaden     maxvärdet inträffar
        int inmaxmonth=0;                       // vilken månad maxvärdet inträffar
        int inmaxyear=2007;                     // vilket år maxvärdet inträffar

        double inmintemp=30;
        int     inminmin=0;
        int inminhour=0;                        //vilken timme minvärdet inträffar
        int inminday=0;                         // vilken dag i månaden     minvärdet inträffar
        int inminmonth=0;                       // vilken månad minvärdet inträffar
        int inminyear=2007;                     // vilket år minvärdet inträffar

        double maxtemp=34;
        int     maxmin=0;
        int maxhour=0;                          //vilken timme maxvärdet inträffar
        int maxday=0;                           // vilken dag i månaden     maxvärdet inträffar
        int maxmonth=0;                 // vilken månad maxvärdet inträffar
        int maxyear=2007;                       // vilket år maxvärdet inträffar

        double mintemp=1;
        int minmin=0;
        int minhour=0;                          //vilken timme minvärdet inträffar
        int minday=0;                           // vilken dag i månaden     minvärdet inträffar
        int minmonth=0;                 // vilken månad minvärdet inträffar
        int minyear=2007;                       // vilket år minvärdet inträffar

        int outcold=0;
        int incold=0;
        double in=0;
        double ut=0;
        int setkl=0;
        int busy=0;
//********************** skriver ut en bokstav***********************************************

void writeChar(char Xv) {
```

```
	DDRB = 0xff;
	DDRA=_BV(PB6)|_BV(PA7);				// sätter E och RS
	PORTA|=_BV(PA6)|_BV(PA7);			//   -"-   -"-
	PORTB=Xv;					// lägger ut bokstaven char
	PORTA&=~_BV(PA6);				// sänker E
//	_delay_ms(20);
	delayus(2000);
	PORTB=0;					// nollställer b portarna

}

// ******************************delay*****************************************************
void  delayGT(uint16_t gt){
	uint16_t i=1;
	uint16_t k=1;
	for(i = 0; i < gt; i++){
		for(k; k < gt; k++){
			asm  volatile ("nop");

		}
	}
}

//******************************skriva ut int****************

void intPrint(int value){


if (value / 10) intPrint(value / 10);
writeChar('0' + value % 10);


	}

//**********************kolla knapp A (1) retunerar 1 om nertryckt 0 om inte**********************
int knappA(){
	DDRD =0x00;
	char data= PIND;
	char test=data & 0b00000001;
	if(test!=0b00000001){
			return 1;

		}
	else{
		return 0;
	}
}
//**********************kolla knapp B (2) retunerar 1 om nertryckt 0 om inte**********************
int knappB(){
	DDRD =0x00;
	char data= PIND;
	char test=data & 0b00000010;
	if(test!=0b00000010){
			return 1;

		}
	else{
		return 0;
	}
}
```

```
//**********************kolla knapp C (3) retunerar 1 om nertryckt 0 om inte**********************
int knappC(){
	DDRD =0x00;
	char data= PIND;
	char test=data & 0b00000100;
	if(test!=0b00000100){
				return 1;

		}
	else{
		return 0;
	}
}

//*******************************skriv ut datum light*****************************************
void datePrintLight(int min,int hour,int day,int mouth,int year){
intPrint(hour);
writeChar(':');
intPrint(min);
writeChar(' ');

intPrint(day);
writeChar('/');
intPrint(month);
writeChar(' ');
intPrint(year);
}
//*******************************updatering av max och min temperatur*****************
void updatetemp(double in,double ut){

if(in<=inmintemp){   // sätter ny inmin om temperaturen inomhus är minst hitills
	inmintemp=in;
	inminmin=min; // samt byter datum och tid för tillfället
	inminhour=hour;
	inminday=day;
	inminmonth=month;
	inminyear=year;

}


if(ut<=mintemp){  //sätter ny outmin om temperaturen utomhus minst hitills
	mintemp=ut;
	minmin=min;// samt byter datum och tid för tillfället
	minhour=hour;
	minday=day;
	minmonth=month;
	minyear=year;
}
if(ut>=maxtemp){//sätter ny outmax om temperaturen utomhus max hitills
	maxtemp=in;
	maxmin=min;
	maxhour=hour;
	maxday=day;		// samt byter datum och tid för tillfället
	maxmonth=month;
	maxyear=year;
}
```

```
if(in>=inmaxtemp){// sätter ny inmax om temperaturen inomhus är minst hittills
        inmaxtemp=in;
        inmaxmin=min;
        inmaxhour=hour;          // samt byter datum och tid för tillfället
        inmaxday=day;
        inmaxmonth=month;
        inmaxyear=year;
}
}
//**********************skriva ut en sträng ****************************************

void writeStr(char *str) {
        while(*str){
          writeChar(*str++);
      }

}


//***************************omvandling av temp*****************************
double change(uint8_t x){
double res=0;

if(x>63){
        intPrint(x);
        x=~x;
        res=x*0.5;
        return res*0.001;
        }

else{
        double res=x*0.5;
        return res;
        }

return res;

}


//*******************************skriva ut double****************
void doublePrint(double ain){
        int bin=ain*10;
        int cin=ain;
        intPrint(cin);
        writeChar(',');
        intPrint(bin % 10);
}
//*******************************clear disp*********************************************
void clearDisp(){
        DDRB = 0xff;
        DDRA =_BV(PA6)|_BV(PA7);
        PORTA=_BV(PA6);                         // op 6 i datablad clear
        PORTB=_BV(PB0);                         // B= 0000 0001
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;
}
```

```
//*******************************ta hand om temperatur*****************************************

void tempOp(double i, double u ){

        if(u<0.4){
        outcold=1;
        }
        if(u>0.4){
        outcold=0;
        }
        if(i<25){
        incold=1;
        }
        if(u>25){
        incold=0;
        }
        in=i;
        ut=u;

        clearDisp();
        updatetemp(i,u);
        writeStr("outsidetemp");
        if(u<0.4){
        writeChar('-');
        doublePrint(u*1000);
        }
        if(u>0.4){                                      //skirver ut utetemperaturen
        doublePrint(u);
        }
        writeStr(" insidetemp ");
        doublePrint(i);     // skriver ut innetemperaturen



}

//***************************skriver ut max och min*****************************************

void printMaxMin(){
        clearDisp();

        writeStr("insidetemperature");
        writeStr(" max ");
        doublePrint(inmaxtemp);
        writeChar(' ');
        datePrintLight(inmaxmin,inmaxhour, inmaxday,inmaxmonth,inmaxyear);
        writeStr(" min ");
        doublePrint(inmintemp);
        writeChar(' ');
        datePrintLight(inminmin,inminhour, inminday,inminmonth,inminyear);

        while(knappB()==0){
        }
        clearDisp();
        writeStr("outsidetemperature");
        writeStr(" max ");
        doublePrint(maxtemp);
        writeChar(' ');
        datePrintLight(maxmin,maxhour,maxday,maxmonth,maxyear);
        writeStr("  min ");
```

```
		if(mintemp>=0.4){
		doublePrint(mintemp);
		}
		if(mintemp<=0.4){
		writeChar('-');
		doublePrint(mintemp*1000);
		}
		writeChar(' ');
		datePrintLight(minmin,minhour,minday,minmonth,minyear);
		while(knappB()==0){
		}
		clearDisp();
		tempOp(in,ut);
		writeStr("	");
		datePrintLight(min,hour,day,month,year);



}
//************************************hantering av knapp 1*************************************

void knapp1(){
		busy=1;
		if(klocka==0){
		setKlocka();				//	Kollar om klocka är ställd om inte går den till
		inställning
		}
		else{
		printMaxMin();				// hoppar till max och minutskrift
		}
		busy=0;
}



//************************************klocka*************************************************
		*********
void clock(){
//		sec++;
		if(sec==60){
		min++;
		sec=0;
		}
		if(min==60){
			hour++;							//byter timme
			min=0;
		}
		if(hour==24){
			days++;
			day++;						// byter dag
			hour=0;
		}


		if(year!=2008 && year!=2012 && year!=2016){
			if(sec==0 || setkl==1){
				if(days==32 || days== 60 || days== 91|| days==121|| days== 152|| days== 182|| days==
		213|| days== 244|| days== 274|| days== 305|| days== 335|| days== 366 ){
				month++;
				day=1;
```

```
                }
        }                                                                       // byter månad och år när
    det inte är skottår
        if (days==366){
        year++;
        day=1;
        days=1;
        month=1;
        }
    }
    if(year==2008 || year==2012 || year==2016){
        if(sec==0|| setkl==1){
            if(days==32 || days== 61 || days== 92|| days==122|| days== 153|| days== 183|| days==
    214|| days== 245|| days== 275|| days== 306|| days== 336|| days== 366 ){
            month++;
            day=1;
            }
        }                                                                       // byter månad och
    år när det är skottår
        if (days==367){
        year++;
        day=1;
        days=1;
        month=1;

        }
    }
    if(busy!=1){
    clearDisp();
    tempOp(in,ut);
    writeStr("       ");
    datePrintLight(min,hour,day,month,year);
    delayGT(2020);
    }

                                                    // ska skriva ut tid och datum när det är färdigt



}


//**********************************inställning av klocka**************************************

void setKlocka(){
    setkl=1;
    while(1){ // while 1.1
        if(klocka==0){
            clearDisp();
            datePrintLight(min,hour,day,month,year);
            klocka=9;
        }
        if(knappB()==1){
            year++;
            if(year>=2015){
            year=2007;
            }
            clearDisp();
            datePrintLight(min,hour,day,month,year);
            delayGT(2020);
```

```
                }
        delayGT(20202);

                if(knappC()==1){
                        delayGT(20202);
                        while(1){ // while 1.2
                                if(knappB()==1){
                                hour=24;
                                clearDisp();
                                clock();
                                datePrintLight(min,hour,day,month,year);
                                delayGT(2020);
                                }
                                if(knappC()==1){
                                        delayGT(20202);
                                        while(1){ // while 1.3
                                                if(knappB()==1){
                                                        min=60;
                                                        clearDisp();
                                                        clock();
                                                        datePrintLight(min,hour,day,month,year);
                                                                delayGT(2020);
                                                }
                                                if(knappC()==1){
                                                        delayGT(20202);
                                                        while(1){// while 1.3
                                                                if(knappB()==1){
                                                                sec=60;
                                                                clearDisp();
                                                                clock();
                                                                datePrintLight(min,hour,day,month,year);

                                                                }
                                                                if(knappC()==1){
                                                                goto out;
                                                                }
                                                        }// while1.3
                                                }
                                        } //while 1.3
                                }
                        } // while 1.2
                }
        }// while 1.1

        out:{
        clearDisp();
        tempOp(in,ut);
        writeStr("       ");
        datePrintLight(min,hour,day,month,year);
        setkl=0;
        }

}

//*************************interuppt hantering*******************
ISR (TIMER1_COMPA_vect){
sec++;                          // räknar upp klockan
if(sec==15||sec==60||sec==30|| sec==45){              //uppdaterar klockan och sköter övergången till min day osv.
clock();
}
```

```
if(busy!=1){
if(sec==16||sec==1||sec==31|| sec==46){                    // uppdaterar temperaturen

                writeStr("1");                                  //******* läser av tempsensor inne och sparar i ina
                        setBit(0);
                        reset();
                        writebyte(0xCC);//          oberoende av adressen

                        writebyte(0x44);//          börja konvertera temperaturen
                        delayus(3000);
                        reset();
                        writebyte(0xCC);//          bry dig inte om adressen
                        writebyte(0xBE);//          vill läsa scratchpad
                        uint8_t ina=readbyte();//  läse;
                        writeStr("2");
                        setBit(1);                      // **********läser av tempsensor ute  och sparar i
                        reset();
                        writebyte(0xCC);//          oberoende av adressen

                        writebyte(0x44);//          börja konvertera temperaturen
                        delayus(3000);
                        reset();
                        writebyte(0xCC);//          bry dig inte om adressen
                        writebyte(0xBE);//          vill läsa scratchpad
                        uint8_t ute=readbyte();//  läse;
                tempOp(change(ina),change(ute)); // omvandlar värdena fårn tempsonsorn till riktiga tempvärden
                writeStr("        ");
                datePrintLight(min,hour,day,month,year); // skiver ut datum
        }

}
}

//**********************************initsiering av display**************************************
void initDisp(void){

        //_delay_ms(20);
        delayGT(2020);

        DDRB = 0xff;
        DDRA=_BV(PA6)|_BV(PA7);
        PORTA=_BV(PA6);                                                                 // op 1 in
        datasheet
        PORTB =_BV(PB4)|_BV(PB5);               // B= 0011 0000
        PORTA&=~_BV(PA6);
//      _delay_ms(6);
        delayus(6000);


        PORTA=_BV(PA6);                                                                 // op 2 in
        datasheet                       // B=0000110000
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);




        PORTA=_BV(PA6);                 // op 3 in datasheet  B= 0011 0000
```

```
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);


        PORTA=_BV(PA6);                                                         // op 4 in
        datasheet func set N=0=singe line display and F=0=5*8 chardisp
        PORTB =_BV(PB4)|_BV(PB5)|_BV(PB3);              // B= 0011 1000
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;



        PORTA=_BV(PA6);                                         // op 5 i datablad disp off
        PORTB =_BV(PB3);                                //  B= 0000 1000
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;



        PORTA=_BV(PA6);                                 // op 6 i datablad clear
        PORTB=_BV(PB0);                                 // B= 0000 0001
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;



        PORTA=_BV(PA6);                                         // op 7 i datablad mode set I/D=1 shift adress
        right S=0 don't shift whole diplay
        PORTB=_BV(PB1)|_BV(PB2);                        // B= 0000 0110 A= 1?00 000
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;


        PORTA=_BV(PA6);
        PORTB=_BV(PB1);                                 // B= 0000 0010
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;

        PORTA=_BV(PA6);                                                         // dips on
        PORTB =_BV(PB3)|_BV(PB2);                       //  B= 0000 1000
        PORTA&=~_BV(PA6);
//      _delay_ms(2);
        delayus(2000);
        PORTB=0;
        }

//*****************************init interrupt*****************************************
void initInte(){
```

```
	TCCR1A = 0;
	TCCR1B=0b00001100; // CRC mode för interrupt samt 256 prescaler
	TIMSK=0b00010000; // sätter interrupt

	OCR1AH=0b01111010;
	OCR1AL=0b00010010;

	sei();

}
//***************************huvudprogram************************************************

int main (void){

	initInte();		//initsierar interrupt
	initDisp();		// starta upp displayen



	while(1){


		if(knappA()==1){			// kontrollerar funktionsknappen
		knapp1();
		}

		if(outcold==1){
		DDRA|=_BV(PA5);			// tänder blå lampa

		PORTA|=_BV(PA5);
		}
		if(incold==1){
		DDRA|=_BV(PA4);			// tänder röd lampa

		PORTA|=_BV(PA4);
		}


	}

	return (0);
}
```

1wiremin.c

```
#include <avr/pgmspace.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdlib.h>
#include "1wiremin.h"
#define F_CPU 8000000UL
#include <util/delay.h>

uint8_t WBIT=0;
void write(uint8_t bit);
void writebyte(uint8_t byte);
uint8_t read();
uint8_t readbyte();
uint8_t reset();
void delayus(uint16_t us);

void setBit(uint8_t katt){
if(katt){                                       // ändrar tempsensor
WBIT= 1;
}
else {
WBIT=0;
}
}

void inline delayus(uint16_t us){
static uint16_t i;
for(i = 0; i < us; i++){                  // delayrutin ca 10+3*x us där x är antalet gånger
        //for(uint8_t k = 0; k < 1; k++){
                asm  volatile ("nop");

        //}
}
}


void write(uint8_t bit){
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
if(bit)
        delayus(DA);

else if(!bit)
        delayus(DC);

WDDR &= ~_BV(WBIT);
if(bit)
        delayus(DB);

else if(!bit)
        delayus(DE);

}



void writebyte(uint8_t byte){
```

```
uint8_t i;
for(i = 0; i < 8; i++){
        write(byte & 0x1);
        byte >>=1;
}

}
uint8_t readbyte(){
 uint8_t data = 0;
 uint8_t i;
 for(i = 0; i < 8; i++){
        data |= read() << i;
}
return data;
}

uint8_t read(){
static uint8_t bit = 0;
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
delayus(DA);
WDDR &= ~_BV(WBIT);
delayus(DE);

if(WPIN & _BV(WBIT))
        bit = 1;
else if(!(WPIN & _BV(WBIT)))
        bit = 0;
delayus(DF);


return bit;
}

uint8_t reset(){
static uint8_t bit = 0;
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
delayus(DH);
WDDR &= ~_BV(WBIT);
delayus(DI);
if(WPIN & _BV(WBIT))
        bit = 1;
else if(!(WPIN & _BV(WBIT)))
        bit = 0;
delayus(DJ);
return bit;
}


#include <avr/pgmspace.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdlib.h>
#include "1wiremin.h"
#define F_CPU 8000000UL
#include <util/delay.h>
/* howto read temperature from ds18s20
uint16_t gettemp(){
reset();
```

```
writebyte(0xCC);oberoende av adressen
writebyte(0x44);  börja konvertera temperaturen
delay_ms(3);
reset();
writebyte(0xCC);bry dig inte om adressen
writebyte(0xBE);vill läsa scratchpad
return readbyte();  läse

}

*/
uint8_t WBIT=0;
void write(uint8_t bit);
void writebyte(uint8_t byte);
uint8_t read();
uint8_t readbyte();
uint8_t reset();
void delayus(uint16_t us);

void setBit(uint8_t katt){
if(katt){
WBIT= 1;
}
else {
WBIT=0;
}
}

void inline delayus(uint16_t us){
static uint16_t i;
for(i = 0; i < us; i++){
        //for(uint8_t k = 0; k < 1; k++){
                asm  volatile ("nop");

        //}
}
}


void write(uint8_t bit){
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
if(bit)
        delayus(DA);

else if(!bit)
        delayus(DC);

WDDR &= ~_BV(WBIT);
if(bit)
        delayus(DB);

else if(!bit)
        delayus(DE);

}

void writebyte(uint8_t byte){

uint8_t i;
```

```
for(i = 0; i < 8; i++){
        write(byte & 0x1);
        byte >>=1;
}

}
uint8_t readbyte(){
 uint8_t data = 0;
 uint8_t i;
 for(i = 0; i < 8; i++){
        data |= read() << i;
}
return data;
}

uint8_t read(){
static uint8_t bit = 0;
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
delayus(DA);
WDDR &= ~_BV(WBIT);
delayus(DE);

if(WPIN & _BV(WBIT))
        bit = 1;
else if(!(WPIN & _BV(WBIT)))
        bit = 0;
delayus(DF);


return bit;
}

uint8_t reset(){
static uint8_t bit = 0;
WDDR |= _BV(WBIT);
WPORT &=  ~_BV(WBIT);
delayus(DH);
WDDR &= ~_BV(WBIT);
delayus(DI);
if(WPIN & _BV(WBIT))
        bit = 1;
else if(!(WPIN & _BV(WBIT)))
        bit = 0;
delayus(DJ);
return bit;
}
```

1wiremin.h

```
#include <avr/pgmspace.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdlib.h>
void inline delayus(uint16_t us);
void setBit(uint8_t katt);

#define WDDR DDRC
#define WPORT PORTC
//#define WBIT 0                        // sensor sitter på pinne C0
#define WPIN PINC
#define DA 1                          //**************
#define DB 64/3  //64
#define DC 60/3
#define DD 10/3
#define DE 1                          // tider att fördröja
#define DF 55/3
#define DG 1
#define DH 480/3
#define DI 70/3
#define DJ 410/3                      //******************

void write(uint8_t bit);
uint8_t read();
uint8_t reset();
void writebyte(uint8_t byte);
uint8_t readbyte();
```

Grupp 15: Andreas Cremon

Kravspecifikation

Krav:

Väderstationen ska ha en sensor som ska kontrollera temperaturnen inomhus dels ha gränser som kontrollerar att temperaturen är över en viss gräns,dels ska det finnas en temperaturesensor som avläser temperaturen utomhus via kabel den sensorn ska även den bevakas.Då mingränsen för innetermometern nås ska det tändas en röd ledlampa. Då mingränsen för utomhussensorn är nåd så ska en blå diod tändas. Båda senorerna skall avläsas åt minstånde en gång på minut. Båda mingränserna för temperaturen skall vara möjliga att ändra till önskad nivå. Dessa båda temperaturena ska generera ett interupt vid underskridning av dom satta teperaturerna.
Max och min värden ska kunna skrivas ut via en knapptryckning(samt datum).
En (Lcd) Display ska presentera temperatur och datum/tid

I mån av tid:

Skall även en tryckmätare kopplas med avläsningar var 30 min som lagras en tabell av de senaste 48 värdena, eventuellt en indikator om trycket sjunker eller höjs.
En vindmätare kopplas in eventuellt en vindriktningsindikator.
displayen ska visa temperatur,tid/datum och vind, tryck (med knapptryckning)