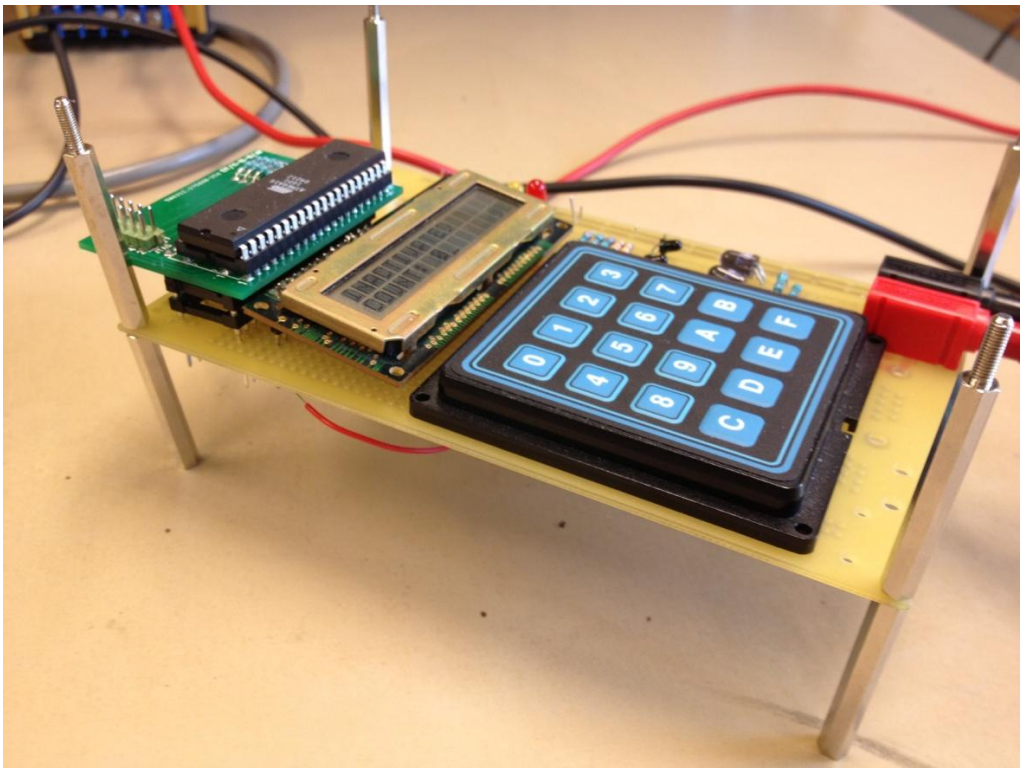
Digitala Projekt(EITF40) - Larm

Handledare: Bertil Lindvall

Erik Oredsson, I-09

Sara Sellin, I-09

2012-05-08



1. SAMMANFATTNING

I denna rapport presenteras vårt projekt att bygga ett huslarm från grunden bestående av två IR-sensorer som inte bara utlöser larmet när de ger utslag, utan även håller koll på hur många som är i lokalen. Vi använder oss av en AVR-processor till vilken vi kopplar på olika komponenter. Genom programmet AVR Studio programmerar vi processorn i programspråket C.

INNEHÅLL

2. INLEDNING

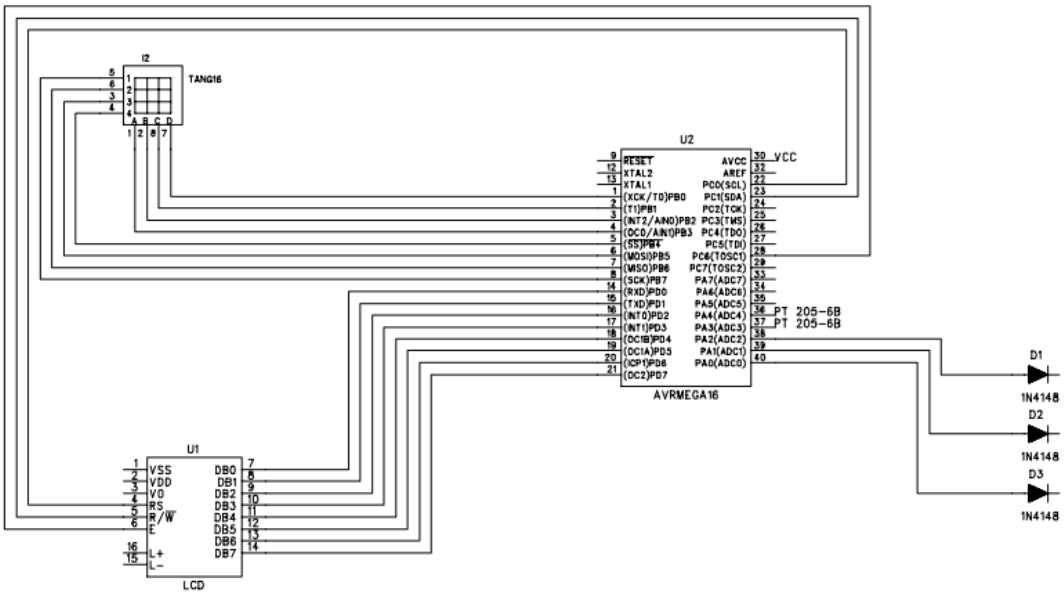
Projektets uppgift är att designa en fungerande prototyp och programmera en mjukvara som integrerar på ett korrekt sätt. Syftet är att få en ökad förståelse i programspråket C och hur en kretskonstruktion fungerar. Vi valde att konstruera en larmanläggning som tar hänsyn till om lokalen är tom eller inte. Idén var att man inte skulle kunna aktivera larmet då det är personer inne i lokalen. Detta för att underlätta för användare så att enbart sista man ut ur lokalen har möjlighet att sätta igång larmet.

3. KRAVSPECIFIKATION

Larmsystemet ska kunna övervaka exempelvis en arbetsplats eller ett hem. Kraven som ställs på prototypen och dess funktioner är:

- Larmet ska kunna hantera 1-2 analoga IR-sensorer för övervakning rörelser.
- Larmet ska kunna hantera en LCD-display.
- Systemet ska kunna aktiveras/avaktiveras av en 4-siffrig PIN-kod
- PIN-koden ska kunna läsas av från en numerisk knappsats.
- Systemet ska kunna räkna antalet personer inne i lokalen.
- Är lokalen inte tom ska en gul lysdiod lysa och antalet personer ska visas på displayen.
- Är larmet aktiverat ska en röd lysdiod lysa och ”ACTIVATED” ska visas på displayen.
- Är larmet inaktiverat ska en grön lysdiod lysa och ”INACTIVATED” ska visas på displayen.
- Larmet ska enbart kunna aktiveras när lokalen är tom.
- När larmet är aktiverat och sensorerna detekterar rörelse ska larmet utlösas, det vill säga, en röd lysdiod blinka.

4. BLOCKSCHEMA



Figur 1: Blockschema för larmsystemet

5. HÅRDVARA

5.1 PROCESSOR

Som processor till vårt larmsystem använde vi oss av en AVR ATMega16. Denna processor har en inbyggd intern klocka på 8 MHz. Det finns 4 portar (A, B, C och D) där de 32 programmerbara I/O pinnarna är uppdelade på. Port A har en inbyggd A/D för att omvandla analoga signaler. Processorn har ett inbyggt flashminne på 16KB och ett internt minne på 1KB. För att vi ska kunna kommunicera med processorn kopplar vi på en JTAG-enhet på protypen och använder programmet AVR Studio 4.

5.2 KNAPPSATS

Vi använde oss av en knappsats med 16 knappar från 0-9 och A-F, uppdelade på fyra rader och fyra kolumner. Dessa kopplade vi direkt till processorn på port C.

5.3 LCD-DISPLAY

För att larmet ska kunna skriva ut önskat meddelande har vi använt oss av en SHARP Dot-Matrix LCD Units som är en alfanumerisk display med möjlighet att skriva 2x16 tecken. LCDn använder sig av ASCII standard för inmatning av tecken.

5.4 LYSDIODER

För att visa om larmet är aktiverat/aktiverat och om lokalen är tom eller inte använder vi tre färgade lysdioder: gul, grön och röd.

5.5 SENSORER

Vi använder oss av två sensorer PT-204-6b som svarar mot två IR-lysdioder. När strålen bryts, då bryts även signalen in till processen och beroende på i vilken ordning dessa sensorer ger utslag får vi information om det är någon som går in i lokalen eller ut. På detta sätt vet vi om det finns personer i lokalen eller ej. Är larmet aktiverat kommer utslag av någon av sensorerna utlösa larmet.

6. MJUKVARA

Programmet skrevs i programspråket C i AVR Studio 4, där systemet konstant går genom en while-loop. I varje loop kontrollerar mjukvaran om någon av sensorerna eller knappsatsen har gett utslag. På så vis har vi hela tiden koll på vad som händer med hårdvaran.

Det som kollas vid varje loop är följande:

- Om larmet är inaktiverat kollar sensorerna om det har gått in någon i lokalen, och i så fall uppdateras antalet, dvs vår variabel ”count”. Finns det folk i lokalen tänder vi den gula lampan.
- Om larmet är inaktiverat och lokalen är tom kontrollerar larmsystemet om knappsatsen används och sparar undan eventuell inmatning i en vektor. När rätt PIN-kod är inmatad aktiveras larmet.
- Om larmet är aktiverat, kontrolleras om någon av sensorerna ger utslag och i så fall utlöses larmet. Även här kontrolleras om knappsatsen registrerar något knapptryck. När rätt PIN-kod är inskriven inaktiveras larmet.

För att avgöra om en person går in i lokalen eller ut ur lokalen använder vi kombinationen av utslagen från sensorerna. Ger sensor1 utslag först (samtidigt som sensor2 inte ger utslag) beror det på att någon är på väg in i lokalen. Följs de sedan av nedanstående kombination kan vi räkna upp antalet personer i lokalen. På liknande sätt kan vi minska antalet personer i lokalen. While-loopen hjälper oss sedan att uppdatera antalet personer och skriva ut antalet på skärmen.

Person går in i lokal		
Tid	Sensor1	Sensor2
	■	
	■	■
		■

Person går ut lokal		
Tid	Sensor1	Sensor2
		■
	■	■
	■	

Se programkoden för att få utförligare information om hur programmet är uppbyggt.

7. KONSTRUKTIONS PROCESS

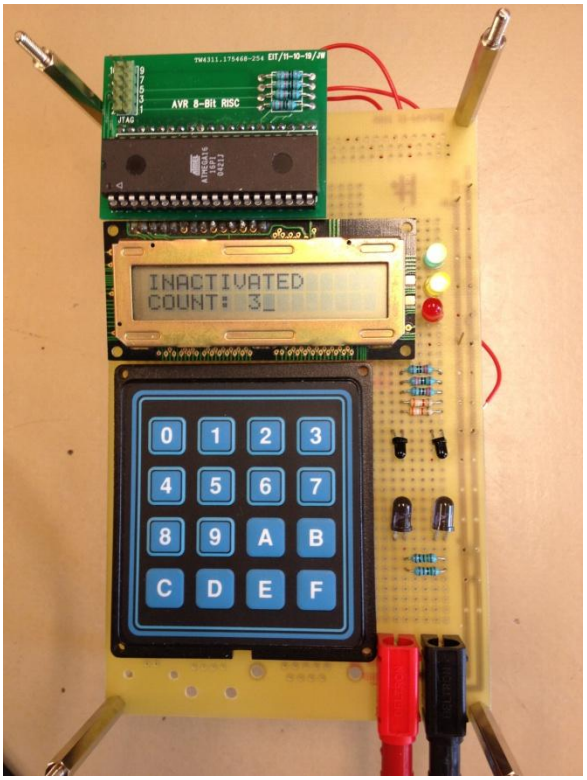
Inledningsvis ritade vi upp ett kopplingsschema enligt diverse datablad och byggde en prototyp efter denna. När vi löddat och virat färdigt hårdvaran kopplades processorn in i datorn med hjälp av en JTAG-enhet. För att underlätta att felsöka i programkoden valde vi att implementerades varje komponent för sig under processens gång.

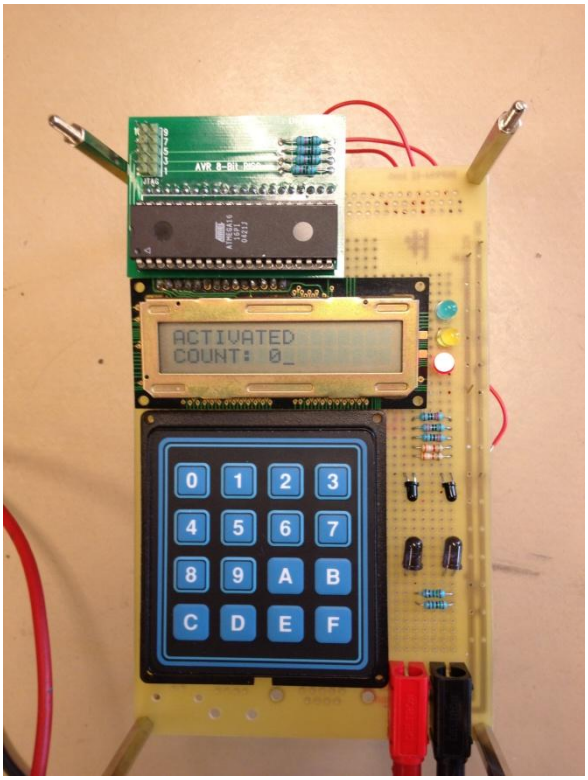
Först programmerades LCD-displayen för att testa att skriva ut ett givet kommando. När vi fick detta att fungera skrev vi även en funktion för att skriva ut om larmet är aktiverat eller inaktiverat samt antalet personer i lokalen. För att inte systemet skulle vara för snabbt och skriva över text var vi tvungna att lägga till delays.

Knappsatsen kopplades till samtliga pinnar på PORTB och programmerades så de lyssnade på rätt PIN-kod. För att lyckas få rätt knapp att registrera rätt siffra lät vi först systemet läsa den av raden i 4x4-matrisen, där första raden returnerade 10, andra returnerade 20 och så vidare. Därefter avgjordes vilken kolumn knappen befann sig på. Fanns den på första kolumnen adderades 1 till det returnerade talet, andra kolumnen adderade 2, och så vidare. Det resulterade talet översättas sedan till rätt siffra, exempelvis ger ”11” (första raden och första kolumnen) en nolla.

För att få lysdioderna att lysa lät vi pinne 0, 1 respektive 2 på PORTA lyssna på kommandon. Dessa funktioner användes bland annat för att visa när systemet är aktiverat eller inaktiverat och om lokalen inte är tom.

Det som var mest tids krävande var att få funktioner som lyssnade på vilken ordning sensorerna registrerade rörelse för att avgöra om personen var på väg in eller ut ur lokalen, och dessutom att skriva en programkod för att räkna upp/ner antalet personer i lokalen.





8. SLUTSATS

När vi byggt larmsystemet efter det första kretsschemat var det en komponent som vi inte lyckades få kontakt med. Efter många om och men, insåg vi att detta berodde på kallödning. Detta löste vi genom att lödda om komponenten. Ett annat problem vi stötte på under processens gång var att vi hade för högt motstånd till sensorerna, så vi bytte från 10K till 3,7K.

Ett problem som tog oss väldigt lång tid att upptäcka var att vi hade kopplat LCD-displayen på pinne 2 på PORTC. Denna port fick vi inte använda då den redan var upptagen av JTAG-enheten. Därför satte vi istället vårat ”E” på pinn 6.

I slutskedet av programmeringen upptäckte vi att när vi försökte skriva ut meddelande på LCD-display lyckades den skriva över bokstäverna och enbart varannan bokstav syntes. När vi satte in ”delays” i programkoden rättade timingen till sig och larmsystemet var färdigt.

Trots dessa problem gick projektet relativt smärtfritt trots att allt var nytt för oss. Vi har dels lärt oss att bygga en prototyp av givna komponenter även att programmera med programspråket C.

9. C-KOD

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
   unsigned char num;

int count = 0; // antal i lokalen
char larmTriggered = 0; //om larmet är utlöst
char larmActivated = 0; //om larmet är aktiverat
int sensor1 = 0;
int sensor2 = 0;
int in[3] = { 1, 2, 3};
int ut[3] = { 3, 2, 1};
int temp[3] = {0, 0, 0};
int var;
int temp2[3] = {0, 0, 0};
int vectorCount = 0;
int code[4] = {1,2,3,4};
int vectorCount2 = 0;
int tempCode[4] = {0, 0, 0, 0};
int temp3 = 10;
char val;

int main(){

		DDRA = 0xE7;
		PORTA = 0x00;
		DDRC = 0xFF;
		DDRD = 0xFF;
		DDRB = 0x0F;
		PORTB = 0xF0;

		disp_init();
		disp_home();
		writeInActivated();
		disp_secondLine();
		writePeopleCount();


		while(1) {
			countPeople();
			if (larmActivated == 0) {
				light_green();
				noLight_red();
				if (count != 0) {
					light_yellow();
```

```
                        } else {
                                noLight_yellow();
                                checkPinCode();
                        }
                }
                if (larmActivated == 1) {
                        checkLarm();
                        light_red();
                        noLight_green();
                        checkPinCode();
                        if (larmTriggered == 1) {
                                _delay_ms(500);
                                noLight_red();
                                _delay_ms(500);
                        }
                }
        }
}

void set_pin(char port, char pin, char state){
        char set = 1 << pin;
        if(port == 'A'){
                set &= PORTA;
                if(set && !state){ //ändra från 1 -> 0
                        PORTA ^= set;
                }
                if(set == 0 && state){ //ändra från 0 -> 1
                        set = 1 << pin;
                        PORTA ^= set;
                }
        } else if(port == 'B'){
                set &= PORTB;
                if(set && !state){ //ändra från 1 -> 0
                        PORTB ^= set;
                }
                if(set == 0 && state){ //ändra från 0 -> 1
                        set = 1 << pin;
                        PORTB ^= set;
                }
        } else if(port == 'C'){
                set &= PORTC;
                if(set && !state){ //ändra från 1 -> 0
                        PORTC ^= set;
                }
                if(set == 0 && state){ //ändra från 0 -> 1
                        set = 1 << pin;
                        PORTC ^= set;
                }
        } else if(port == 'D'){
                set &= PORTD;
```

```
                        if(set && !state){ //ändra från 1 -> 0
                                PORTD ^= set;
                        }
                        if(set == 0 && state){ //ändra från 0 -> 1
                                set = 1 << pin;
                                PORTD ^= set;
                        }
                }
}
```

Lysdioder _____

```
//Tänder grön lysdiod
void light_green() {
        set_pin('A', PA0, 1);
}

//Tänder gul lysdiod
void light_yellow() {
        set_pin('A', PA1, 1);
}

//Tänder röd lysdiod
void light_red() {
        set_pin('A', PA2, 1);
}

//Släcker grön lysdiod
void noLight_green() {
        set_pin('A', PA0, 0);
}

//Släcker gul lysdiod
void noLight_yellow() {
        set_pin('A', PA1, 0);
}

//Släcker röd lysdiod
void noLight_red() {
        set_pin('A', PA2, 0);
}
```

LCD-skärm _____

```
//Skriver ut ACTIVATED på skärmen
void writeActivated(){
        disp_writeCh('A');
        disp_writeCh('C');
```

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

//Skriver ut INACTIVATED på skärmen
void writeInActivated(){
		disp_writeCh('I');
		disp_writeCh('N');
		disp_writeCh('A');
		disp_writeCh('C');
		disp_writeCh('T');
		disp_writeCh('I');
		disp_writeCh('V');
		disp_writeCh('A');
		disp_writeCh('T');
		disp_writeCh('E');
		disp_writeCh('D');
}

//Skriver ut COUNT: "antal personer i lokalen" på skärmen
void writePeopleCount(){
		disp_writeCh('C');
		disp_writeCh('O');
		disp_writeCh('U');
		disp_writeCh('N');
		disp_writeCh('T');
		disp_writeCh(':');
		disp_writeCh(' ');
		disp_writeNum(count);
}

//Tillkallas när vi vill skriva kommando
void write_cmd(char val) {
		PORTD=val;
		_delay_ms(5);
		set_pin('C', PC1, 0); //RW
		set_pin('C', PC0, 0); //RS väntar på kommando
		_delay_ms(5);
		set_pin('C', PC6, 1); //E går till hög
		_delay_ms(5);
		set_pin('C', PC6, 0); //E går till låg
}

//Tillkallas när vi vill skriva ut en bokstav/siffra/tecken
void disp_writeCh(char val) {
```

```
		PORTD=val;
		set_pin('C', PC1, 0); //RW
		set_pin('C', PC0, 1); //RS visar på skärm
		_delay_ms(5);
		set_pin('C', PC6, 1); //E går till hög
		_delay_ms(5);
		set_pin('C', PC6, 0); //E går till låg
}

//Tillkallas när vi vill skriva nummer
void disp_writeNum(int number) {
		if (number < 10) {
			if (number == 0) {
				disp_writeCh('0');
			}
			if (number == 1) {
				disp_writeCh('1');
			}
			if (number == 2) {
				disp_writeCh('2');
			}
			if (number == 3) {
				disp_writeCh('3');
			}
			if (number == 4) {
				disp_writeCh('4');
			}
			if (number == 5) {
				disp_writeCh('5');
			}
			if (number == 6) {
				disp_writeCh('6');
			}
			if (number == 7) {
				disp_writeCh('7');
			}
			if (number == 8) {
				disp_writeCh('8');
			}
			if (number == 9) {
				disp_writeCh('9');
			}
		} else {
			int num1 = number/10;
			disp_writeNum(num1);
			int num2 = number%10;
			disp_writeNum(num2);
		}
}
```

```
void disp_clear() {
		write_cmd(0x01); // clear display
		_delay_ms(5);
		write_cmd(0x38); //functions set
		_delay_ms(5);
}

void disp_init() {
		disp_clear();
		write_cmd(0x0F); //display on
		_delay_ms(1);
		write_cmd(0x06); //Entry mode set
		_delay_ms(1);
}

void disp_home(){
		write_cmd(0x03); //flyttar markören hem
		_delay_ms(5);
}

void disp_secondLine() {
		write_cmd(0xC0); //Byter rad
		_delay_ms(5);
}
```

Tangentbord _____

```
//Läser av vilken rad som trycks in på PIN-kodsterminalen
int checkRow() {
		DDRB = 0x0F;
		PORTB = 0xF0;
		val = PINB & 0xF0;
		if (val == 0xE0) {
			return checkCol(10);
		}
		if (val == 0xD0) {
			return checkCol(20);
		}
		if (val == 0xB0) {
			return checkCol(30);
		}
		if (val == 0x70) {
			return checkCol(40);
		}
		return 0;
}

//Läser av vilken kolumn som trycks in på PIN-kodsterminalen
int checkCol(int x) {
		set_pin('B', PB0, 0);
```

```
		set_pin('B', PB1, 0);
		set_pin('B', PB2, 0);
		set_pin('B', PB3, 1);
		val = PINB & 0xF0;
		if (val == 0xF0) {
			return (x + 1);
		}
		set_pin('B', PB0, 0);
		set_pin('B', PB1, 0);
		set_pin('B', PB2, 1);
		set_pin('B', PB3, 0);
		val = PINB & 0xF0;
		if (val == 0xF0) {
			return (x + 2);
		}
		set_pin('B', PB0, 0);
		set_pin('B', PB1, 1);
		set_pin('B', PB2, 0);
		set_pin('B', PB3, 0);
		val = PINB & 0xF0;
		if (val == 0xF0) {
			return (x + 3);
		}
		set_pin('B', PB0, 1);
		set_pin('B', PB1, 0);
		set_pin('B', PB2, 0);
		set_pin('B', PB3, 0);
		val = PINB & 0xF0;
		if (val == 0xF0) {
			return (x + 4);
		}
}

//Översätter vilken siffra som trycks in på PIN-kodsterminalen. OBS de "orelevanta"
returnerar 9. Trycks ingen knapp ner returneras 10.
int checkButton() {
		int button = checkRow();
		if (button == 11) {
			return 9;
		}
		if (button == 12) {
			return 1;
		}
		if (button == 13) {
			return 2;
		}
		if (button == 14) {
			return 3;
		}
		if (button == 21) {
```

```
                return 4;
        }
        if (button == 22) {
                return 9;
        }
        if (button == 23) {
                return 9;
        }
        if (button == 24) {
                return 9;
        }
        if (button == 31) {
                return 9;
        }
        if (button == 32) {
                return 9;
        }
        if (button == 33) {
                return 9;
        }
        if (button == 34) {
                return 9;
        }
        if (button == 41) {
                return 9;
        }
        if (button == 42) {
                return 9;
        }
        if (button == 43) {
                return 9;
        }
        if (button == 44) {
                return 9;
        }
        return 10;
}
```

PIN-Kod_____

```
//Lyssnar på PIN-kodsterminalen och om rätt siffra i rätt ordning slås in, läggs de in i en vektor. Vid rätt PIN-kod aktiveras larmet/inaktiverats larmet.
void checkPinCode(){
        _delay_ms(250);

        temp3 = checkButton();

        if (temp3 == 10) {
                return;
        }
```

```
		if (temp3 == 9) {
			vectorCount2 =0;
		}

		if (temp3 == 1 && vectorCount2 == 0) {
			tempCode[0] = 1;
			vectorCount2 = 1;

		} else if (temp3 == 2 && vectorCount2 == 1) {
			tempCode[1] = 2;
			vectorCount2 = 2;


		} else if (temp3 == 3 && vectorCount2 == 2) {
			tempCode[2] = 3;
			vectorCount2 = 3;

		} else if (temp3 == 4 && vectorCount2 == 3) {
			tempCode[3] = 4;
			vectorCount2 = 0;
			if (larmActivated == 1) {
				if (rightPinCode() == 1) { //Returnerar 1 om rätt kod
					larmActivated = 0;
					larmTriggered = 0;
					disp_clear();
					disp_home();
					writeInActivated();
					disp_secondLine();
					writePeopleCount();
					}
				} else {
					if (rightPinCode() == 1) {
						larmActivated = 1;
						larmTriggered = 0;
						disp_clear();
						disp_home();
						writeActivated();
						disp_secondLine();
						writePeopleCount();
				}
			}
		}
}

//Jämför om tempCode är rätt PIN-kod. Returnerar 1 om rätt PIN-kod.
int rightPinCode() {
		if (tempCode[0] == 1 && tempCode[1] == 2 && tempCode[2] == 3 &&
tempCode[3] == 4) {
			return 1;
```

```
		} else {
			return 0;
		}
}
```

Sensor _____

```
//Kollar sensorerna om larmet är inaktiverat
void checkSensor(){
		char sensor = PINA & 0x18;
		if (sensor == 0x10) {
			sensor1 = 1; //har ej brytits
			sensor2 = 0;
		}
		if (sensor == 0x08) {
			sensor2 = 1; //har ej brytits
			sensor1 = 0;
		}
		if (sensor == 0x18) {
			sensor1 = 1;  //har ej brytits
			sensor2 = 1;  //har ej brytits
		}
		if (sensor == 0x00) {
			sensor1 = 0;
			sensor2 = 0;
		}
		return;
}

//Kollar sensorerna om larmet är aktiverat
void checkLarm() {
		char sensorLarm = PINA & 0x18;
		if (sensorLarm != 0x18) {
			larmTriggered = 1; //Larmet utlöses
		}
}

//Räknar antal personer i lokalen
void countPeople() {
		checkSensor();
		inOrOut();
}

//Sätter kombination av aktiva sensorer till en variabel och lägger variabler in i en vektor,
detta för att sedan veta om personer går in i lokalen eller ut.
void inOrOut(){
		if (sensor1 == 1 && sensor2 == 1){
			var = 0;
		}
		if (sensor1 == 0 && sensor2 == 1){
```

```
                    var = 1;
                }
                if (sensor1 == 0 && sensor2 == 0){
                    var = 2;
                }
                if (sensor1 == 1 && sensor2 == 0){
                    var = 3;
                }
                if (var == 0 && vectorCount == 3) {
                    checkInOrOut();
                }
                if (var == 1 && vectorCount == 0) {
                    temp[0] = 1;
                    vectorCount++;
                }
                if (var == 2 && vectorCount == 1) {
                    temp[1] = 2;
                    vectorCount++;
                }
                if (var == 3 && vectorCount == 2) {
                    temp[2] = 3;
                    vectorCount = 3;
                }
                if (var == 3 && vectorCount == 0) {
                    temp[0] = 3;
                    vectorCount++;
                }
                if (var == 1 && vectorCount == 2) {
                    temp[2] = 1;
                    vectorCount = 3;
                }
}
```

//Kontrollerar om personen går in eller ut ur lokal genom att granska kombinationen i vektorn som skapades i föregående funktion. Obs accepterar ej negativt antal.

```
void checkInOrOut(){

                //Kollar om In
                if (temp[0] == 1 && temp[1] == 2 && temp[2] == 3) {
                    count++;
                    disp_clear();
                    disp_home();
                    writeInActivated();
                    disp_secondLine();
                    writePeopleCount();
                    vectorCount = 0;
                    temp[0] = 0;
                    temp[1] = 0;
                    temp[2] = 0;
                }
```

```
		//Kollar om Out
		if (temp[0] == 3 && temp[1] == 2 && temp[2] == 1) {
				count--;
				disp_clear();
				disp_home();
				writeInActivated();
				disp_secondLine();
				writePeopleCount();
				vectorCount = 0;
				temp[0] = 0;
				temp[1] = 0;
				temp[2] = 0;
		}

		if (count == -1) {
				count = 0;
		}
}
```