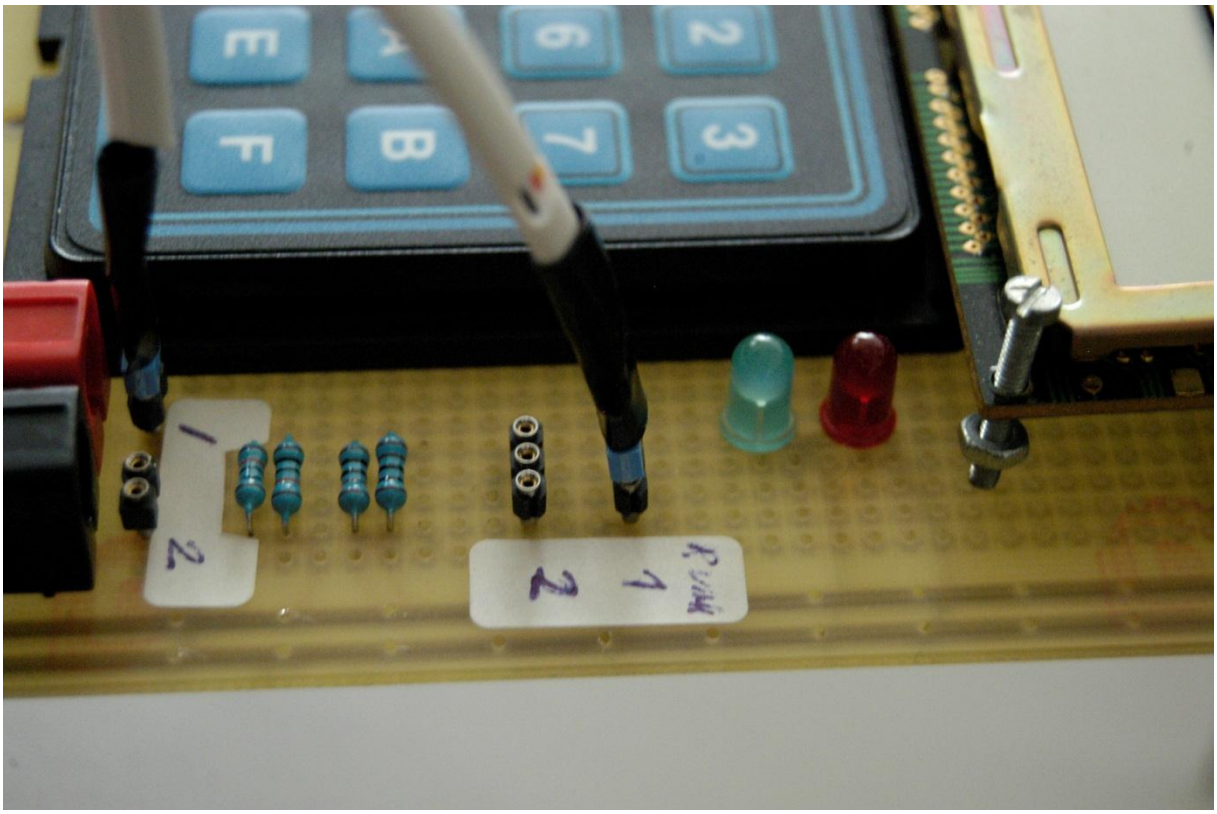
LUNDS TEKNISKA HÖGSKOLA

Kyl- & fryslarm

Ett projekt i EITF11

Jan Babor och Oscar Ågren
Handledare: Bertil Lindvall
5/16/2011



Introduktion

Syftet

I det stora hela har kursen Digitala Projekt krävt att studenten ska lära och förstå hur hårdvara integreras tillsammans med mjukvara. Det har gjorts genom att testa kunskaper inom kretskonstruktion och programmering för att i slutligen få ut en färdig prototyp.

Mål

I den här rapporten behandlas ett projekt som har till mål att skapa en prototyp som kan övervaka en kyl- och frysanordning. Anordning ska i huvudsak bevaka två kylda rum där varje rum övervakas med en temperatursensor och dörrsensor. I prototypen ska operatören ha möjlighet att bestämma gränsvärden för både temperatur och tid för hur länge en dörr får vara öppen innan larmet utlöses.

Innehållsförteckning

Process

Kravspecifikation

Vårt larmsystem är avsett för kyl- och frysrum. Dess funktionalitet består i att övervaka ett eller flera rum.

Funktionella krav (per rum)

- En analog sensor avläser temperaturen.
- För varje sensor definieras ett lågvärde och ett högvärde i grader.
- Hög- respektive låglarm utlöses när den analoga sensorn passerar respektive gräns.
- En digital sensor avläser huruvida dörren till rummet är stängd.
- För varje rum definieras en tidsperiod för hur länge dörren får vara öppen.
- Om dörren är öppen längre än angiven tidsperiod utlöser systemet ett dörr larm.
- Samtliga larm och larmtider skall loggas.
- Möjlighet att kvittera ett larm, då kan det inte larma igen förrän det har återgått (d.v.s. dörren stängd eller temperaturen har återgått till börvärde)
- Möjlighet att larma av rummet. Inga larm kommer att gå igång.

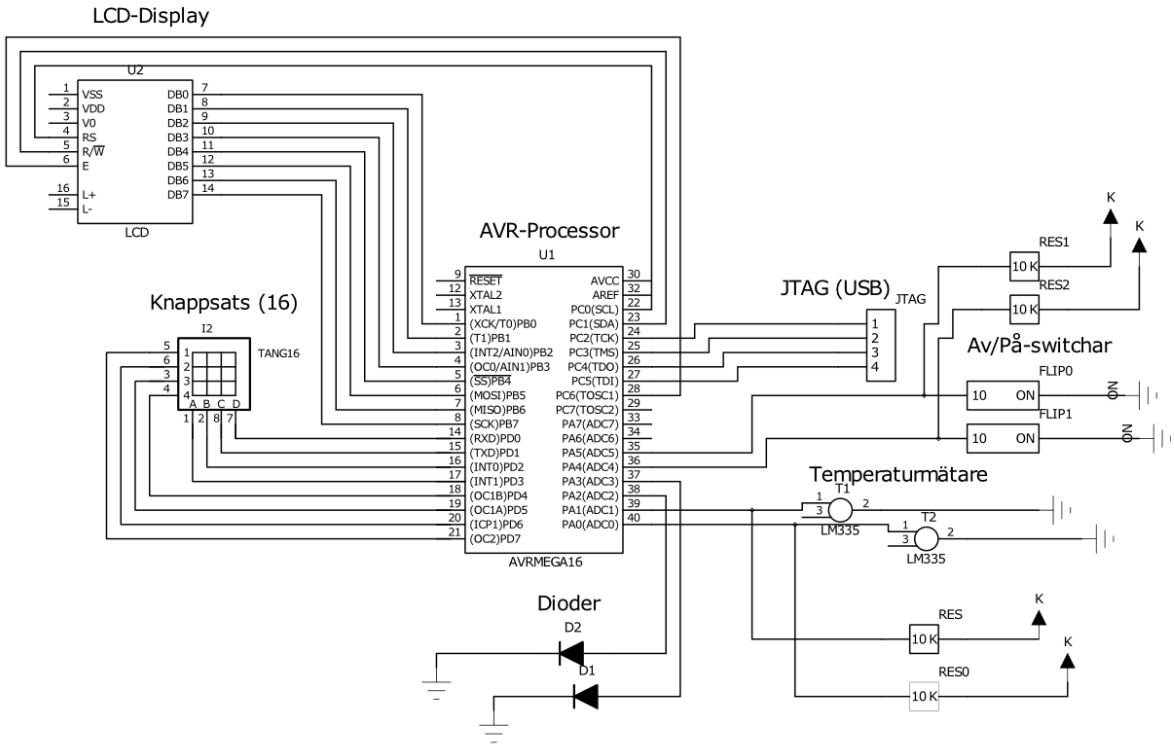
Icke-funktionella krav

- Tiden från sensor till aktivt larm skall vara under en sekund.
- Om flera larm är aktiva så skall temperaturlarm prioriteras före dörr larm (högst prioritet visas på display).
- Om användargränssnittet inte tillåter kvittering av specifikt larm så skall det larm som syns på displayen kvitteras.

Krav i mån av tid

- När larm uppstår skall en diod tändas och en signal ljudas utöver det att displayen visar aktivt larm.
- Vid kvittering av aktivt larm så tystas ljudsignalen, men dioden lyser tills dess att samtliga larm återgått. Alternativt tre dioder (grön, gul, röd) som visar status på systemet.
- 4x4-knappsats som tillåter kvittering och av/på-larmning av specifika rum.

Hårdvara



Figur 1: Kopplingsschema skapat i PowerLogic

- **Skärm - SHARP Dot-Matix LCD Units**
  En LCD med möjlighet till att spara 4 x 40 tecken där varje tecken skrivs ut på en 5x7 pixel-matris. I prototypen är det inte nödvändigt med mer än 2x16 tecken.
  LCDn använder sig av ASCII standard för inmatning av tecken och tar emot tecknen med hjälp av en 8-bitars mikroprocessor.
  Skärmen ger även möjligheten att skriva på specifika adressrutor för uppdatering av enbart enskilda rutor, t.ex. en klocka i ena hörnet.

- **Processor - AVR ATMega16**
  En processor som uppfyller prototypens krav. Den erbjuder 32 programmerbara 1/0 pinnar, varav 8 har möjligheten till A/D omvandling. I projektet utnyttjas 29 pinnar. Samt ger möjligheten till att skriva ett 16 KB stort förprogrammerat program som lagras i ett inbyggt flash-minne. Processorn erbjuder även ett 512B EEPROM minne för ytterligare lagring samt ett internt minne på 1 KB. Det finns ett J-TAG gränssnitt för enkel integrering med AVR-studio som är programmet där all kod skrivs.

- **2 x Temperatur sensorer - LM335**
  LM335 är en temperatursensor som mäter i Kelvin. Den ger utslag för 10 mV/grad och har ett gränsområde mellan -40 C till 100 C där den har en inledande noggrannhet på 1 C vilket är tillräckligt för att uppfylla anläggningens krav. I projektet används inte justerings pinnen då det inte anses vara nödvändigt för att uppfylla kraven.

- **2 x av/på switchar - Sentrol magnetkontakt**
  En sensor som skapar en bruten krets som sluts när den befinner sig nära ett magnetfält. Komponenten är vital i prototypen för att den ska kunna registrera om dörrarna är öppna eller stängda.

- **2 x 10k Ohm + 2 x 1k Ohm resistorer**
  4 resistorer är nödvändiga att integrera för att magnetkontakterna och temperatur sensorerna ska fungera korrekt. 10 k Ohms resistorer kopplas in tillsammans med magnetkontakterna för att processorn ska läsa en 1:a när kretsen inte är sluten och en 0:a när kretsen är sluten(kontakt med ett magnetfält). 1 k Ohms resistorer kopplas in med temperatur sensorerna för att få en korrekt referens värde som processorn kan läsa.

- **Knapp sats - Grayhill 88BB2-072**
  En 4x4 matris knappsats med tangenterna 0-9 och A-F. 0-9 ska fungera som referenstangenter och A-F fungerar som Alternativ(meny) knappar.
  Knappsatsen är konstruerad så att signaler skickas kontinuerligt från processorn till varje knapprad och om en knapp är nedtryck går signalen vidare och registrerar en specifik kolumn till processorn, signalerna sätt ihop och bildar tillsammans den specifika nedtryckta knappen.

- **2 LED lampor**
  Röd - En alarm lampa
  Grön - En funktions lampa
  Beroende på hur lamporna blinkar kommer ska operatören ha möjligheten läsa av olika scenarion

- **1 MHz kristall**
  En extern kristall på 1 MHz monteras in för att skapa en mer precis klocka. Kristallen ger möjligheten till att få ut mer exakta avbrott och därigenom få en klocka som inte drar fel.

Mjukvara

Programmet skrivs i C-kod i AVR Studio 4. Programmet skrivs så att krav specifikationen uppfylls, det tillåts att det uppkommer enstaka fel. Men enbart så tolerant att prototypen fortfarande kan utför sin uppgift.
Programmet har en hemmaskärm som visas vid start. På hemskärmen kan operatören avläsa om något larm har utlösts och isåfall vilket. Vid hemskärmen kan operatören gå in i undermenyer:

- "B" - Stäng av
  A - Bekräfta avstängning
- "C" - Se klocka
  "C" - Ställ in tiden
- "D" - Display - Visar status för de rummen som är inkopplade
- "E" - Konfigurering
  1 till 9 - Bestäm rum för konfiguration sedan HÖG och LÅG temperatur
- "F" - Larm Meny
  "A" - Kvittera larm (om ej aktivt, okvitterat larm finns)
  "A" – Larma på systemet (Om systemet är avaktiverat)
  "A" - Larma av systemet (Om systemet är aktiverat)

I alla undermenyer backar programmet tillbaka till hemskärmen med kommandot "**B**".
Vid konfigurering av någon variabel bekräftas valet av "**C**" eller "**F**". Ex vid tids inställning bekräftas inmatningen med någon av dessa knappar.
Vid inmatning av negativa värden är "**A**" symbol för minustecken (Tryck **A3** = -3).
För att korrigera en felaktig inmatning raderas inmatningen med alltid med "**E**".

Konstruktions process

För att processen ska bli enkel att följa och felsöka i, blir det lämpligt att varje hårdvarukomponent installeras stegvis med noggrann implementering.

Först kopplas processorn tillsammans med J-TAG in för att se att processorn fungerar som den ska och att AVR-Studio får kontakt. Noggrannhet är viktigt för att inte processorn ska kortslutas.

Knappsatsen kopplas därefter in på port B. Den använder alla 8 pinnar på porten där fyra kommer ta emot signaler genom att använda pull-up-läget och de resterande fyra kommer skicka tre höga och en låg signal. Den låga pinnen alterneras hela tiden via avbrottsfunktionen. På så sätt kan en intryckt knapp avläsas genom matrislogik.

Prototypen använder sig av en intern klocka med ett programmerat avbrott. Avbrottet används för att läsa av knappar och ökar en global räknare, kallad ticks. Klockan och LED-lamporna beror i sin tur på räknaren.

LCD kopplas in på hela port D men behöver även 3 pinnar extra för speciella kommandon som kopplas in på port C och pinnar 0,1 och 6. Processorn skickar signaler via alla pinnar. Lämpligt skrivs

en metod som håller reda på klockan och visar den på LCD för att testa kopplingar och avbrott. Knappsatsen implementeras och en funktion för att ställa in klockan appliceras till programmet.

LED lampor kopplas in på port A och pinnarna 0 och 1. Processorn ska skicka signaler till lamporna. Metoder för hur lamporna ska blinka vid olika scenarion kodas och testas.

Det krävs exakt avbrott för att klockan ska fungera utan att dra fel. Det krävdes en extern frekvensmatare för att lösa det. En kristall på 1mhz tillsammans med två kondensatorer installerades på XTAL-pinnen. Processorns klocka ställdes om från intern till extern kristall.

Sist kopplas temperatur och magnetkontakter in på port A och pinnarna 2,3 respektive 4 och 5. Processorn programmeras för att ta emot signaler från temperatursensorn, som utnyttjar en A/D-omvandlare tillsammans med det externa referens värdet från pinne AREF, och magnetkontakten.

När alla komponenter och metoder för att styra dem fungerar återstår endast att implementera ett program som utnyttjar allt

Resultat

Programmets struktur

För att behålla översikten när mängden kod ökar krävs en tydlig struktur. C är inget objektorienterat språk, men genom att dela in relaterade funktioner i separata filer så kan en viss objektstruktur uppnås.

Programfilerna är följande

Freeze.c – ”Huvudprogrammets fil, sköter initiering av alla andra filer, huvudloop och interrupt”
Alarm.c – ”Sköter larmhantering”
Key.c – ”Sköter knappsatsen”
LCD.c – ”Drivrutiner för LCD-displayen”
LED.c – ”Drivrutiner för LED-lamporna”
Menu.c – ”Sköter menysystemet”
Temp.c – ”Hanterar temperaturavläsning”
Switch.c – ”Sköter magnetswitchavläsningen”
Time.c – ”Sköter klockfunktioner”

Samtliga programfiler har även en header-fil. Utöver dessa filer finns även ports.h, en header-fil som definerar samtliga portar för såväl LCD som temperatur. Det underlättar att ha alla portar och pinnar på en överskådligt plats.

Alla programfiler förutom Freeze.c implementerar en initieringsmetod och en destruktionsmetod enligt samma namnschema,Freeze.c anropar exempelvis initLCD(); och destroyLCD() vid uppstart respektive avstängning. Detta gör det enkelt att undvika minnesläckor.

Fungerar prototypen

Prototypen fungerar utan några felaktigheter efter de första testerna. Det finns inga kända buggar.

Ouppnått mål

- Samtliga larm och larmtider skall loggas.

  Istället loggas det senaste hög-/låglarmstiden för varje rum. Dörrsensorlarmet är inte lika tidskritiskt så det återgår av sig själv och ingen tid loggas.

  Loggningen sker inte till EEPROM utan behålls i minnet tills dess att larmet har kvitterats.

Felaktigheter under hårdvaru konstruktionen

En processor kortslöts då pinnarna felvändes vid inkoppling och plus blev minus och vice versa. Processorn fick ersättas.

Ett annat misstag var att temperatursensorn kopplades fel på liknande sätt som processorn, vilket ledde till någon dags försening innan felet kunde lokaliseras.

Magnetkontakternas resistorer kopplades till GRD vilket ledde till att pinnarna alltid gav låg-signal oavsett om magnetkretsen var sluten eller inte.

Diskussion

Jämfört med Java-programmering på moderna x86-datorer så var detta en frisk omväxling. Att själv sköta avbrott och klockan var lärorikt, det är något vi annars tar för givet.

Felen som uppstod var inte ovälkomna utan nästan nödvändiga för att man verkligen skulle få förståelse för projektet. Skulle inga problem ha uppstått tror vi inte att vi skull ha den kunskapen vi nu besitter efter all felsökning.

Framtida justeringar

Om prototypen ska ha en högre funktionalitet på bevakade rum, kan det bli relevant att utöka loggningsfunktionaliteten. Till exempel logga mot ett datorsystem eller via sms så att larmet kan spridas vidare.

Appendix - källkod

En del av freezer.c (hela källkoden finns på hemsidan).

```
/// destroy:: clean up and free memory
void destroy(void){
   destroyTime();
   destroyLED();
   destroyLCD();
   destroyKeyboard();
   destroyTimers();
   destroyMenu();
   destroyAlarm();
}

/// main:: program loop
int main(void)
{
   for(;;)
   {
      // set time to 00:00:00
      initTime(0,0,0);
      // initiate the digital switches
      initSwitch();
      // light both leds
      initLED();
      // Initiate the LCD
      initLCD();
      // Initiate the keypad
      initKeyboard();
      // Start the timers and interrupt
      initTimers();
      // Initiate temperature reading
      initTemp();
      beginReadTemp();
      // Activate the alarm
      initAlarm();
      // Create the menu system
      initMenu();

      while(!getMenuState().shutdown)
      {
         if(time_ticks > timeTicksPerSecond){
            time_ticks -= timeTicksPerSecond;
            addSecond(getTime());
            // refresh the menu/display
            menu_show();
            // begin another temperature reading
            beginReadTemp();
            // update alarm status
            if(isTempInitialized())
               checkAlarm();
         }
         if(led_ticks > ledTicksPerSecond){
            led_ticks -= ledTicksPerSecond;
            LED_blink();
         }

         char key = readKey();
         if(key != NOKEY ){
            menu_handleKey(key);
         }
      }
      // rebooting...
      LCD_clear();
      LCD_putCharArray("    Goodbye!");
      cli();
      _delay_ms(500);
      destroy();
   }
   return 0;
}
```