Temperaturregleringssystem

Ett arbete i kursen Digitala Projekt vid LTH vårterminen 2002

Stefan Nilsson d98sn@efd.lth.se
Karl Torpel d98kt@efd.lth.se

Inledning:

Bakgrund:

Behovet av temperaturreglering av hela system ökar hela tiden. Antingen genereras för mycket värme eller också behöver det tillföras värme. För att reglera temperaturen sätter man in värmeelement, kylfläktar mm som förutom att reglera temperaturen ofta genererar störande ljud. För att minska ljudnivån är det därför viktigt att ingen komponent i systemet körs mer än nödvändigt för att uppnå maximal effekt.

Vi ville studera reglering av ett system och valde därför ett tämligen enkelt, men aktuellt system; en dator.

Varje komponent i datorn genererar värme, CPU, grafikkort, hårddiskar mm. Denna värme stängs dessutom in i en låda med tämligen dålig ventilation. Vi ville reglera ett antal fläktar, dels på komponenterna direkt, men även allmänna ”lådfläktar” som samverkar med varandra på ett smart sätt. Indata kommer från ett antal temperatursensorer placerade på lämpliga ställen i datorlådan.

Kravspecifikation:

Konstruktionen skall:

- kunna reglera 8 fläktar.
- ta in information om systemet från 8 temperatursensorer.
- kunna läsa en av användaren specificerad relationsdatabas som beskriver hur de olika fläktarna kan påverka systemet, samt gränsvärden för temperatursensorerna.
- reglera fläktarna individuellt och utifrån systemets totala kylningskrav.
- varje fläkt skall regleras steglöst för att alltid säkerställa lägsta möjliga ljudnivå med uppfyllt kylningskrav.
- aldrig köra en reglerare med mer effekt än vad som precis krävs för att klara gränsvärdena i relationsdatabasen.
- alltid köra en reglerare precis så fort att gränsvärdena i relationsdatabasen klaras.
- ge kontinuerlig information om systemets status på en LCD.
- spara relationsdatabasen i icke-flyktigt minne
- automatisk återläsa relationsdatabasen vid eventuellt strömbortfall/omstart
- (i mån av tid) relationsdatabasen skall kunna uppdateras direkt från ett program på en dator via en seriekoppling.
- (i mån av tid) kontinuerligt överföra information till dator över seriekoppling i loggningssyfte.

Genomförande:

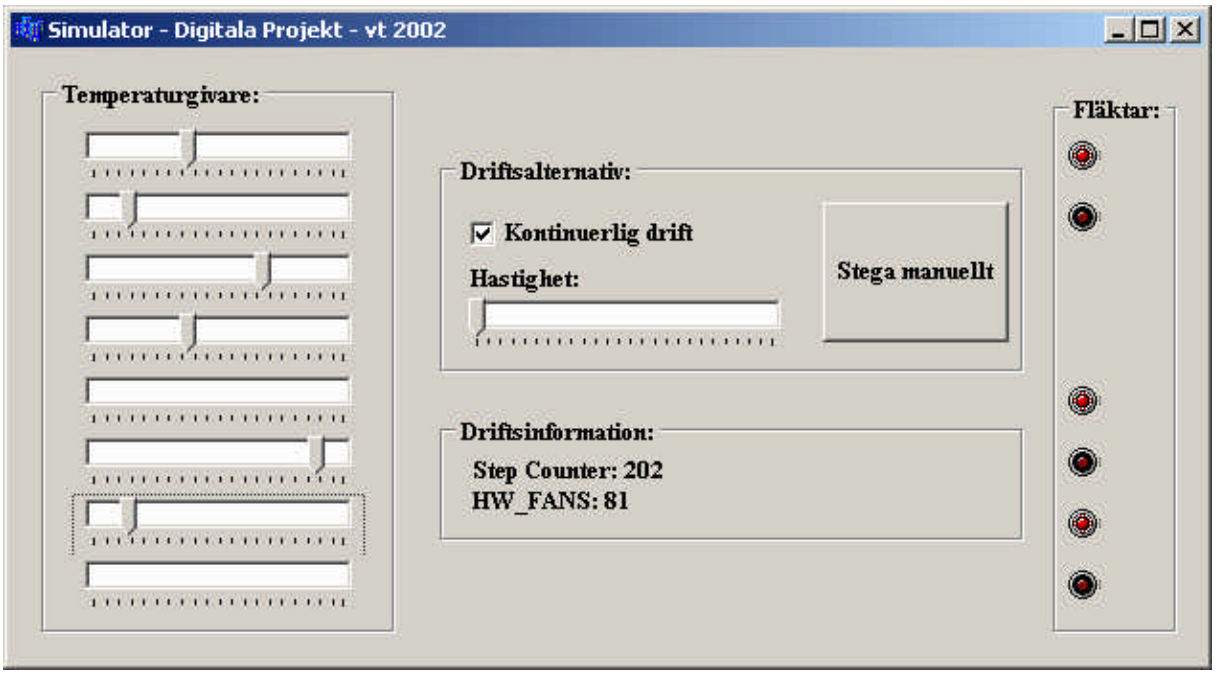
Teoretisk modell:

Varje temperatursensors aktuella temperatur antas bero på ett antal olika fläktar. Det kan t ex antas att temperaturen hos CPU:n direkt beror på hur mycket CPU-fläkten snurrar, men den beror även på hur varmt det är i själva lådan, dvs. på hur effektiv lådfläkten är. Varje temperatursensor kopplas därför till varje fläkt i hela systemet. Via en relationsdatabas kan det anges exakt hur mycket en viss fläkt kan tänkas påverka en viss temperatursensor, samt vilka gränsvärden en viss temperatursensor måste hållas inom. I exemplet ovan kan antas att CPU-temperaturen bero till 100% på CPU-fläkten och kanske till 25% på lådfläkten. Genom att med jämna mellanrum beräkna hur fort varje sensor tycker att varje fläkt skall snurra, och sedan sätta hastigheten på aktuell fläkt till maxhastigheten bland de beräknade hastigheterna, säkerställs att alla temperatursensorers krav uppfylls med så snäv marginal som möjligt. Kyleffekten maximeras och ljudnivån minimeras.

Styrningen av fläktarna sker medelst pulsning av driftsströmmen. Pulsningen sköts med en modifierad form av PWM (Pulse Width Modulation). Längden av pulserna bestämmer hur snabbt fläkten snurrar.

Simulator:

För att testa temperaturregleringsalgoritmen utvecklades en enkel simulator i Windows-miljö. Här kan anges vad temperatursensorerna ger för utsignal och vid körning studera hur pulserna till reglerarna ser ut.



Simulatorn i drift med 6 temperaturgivare och 6 fläktar inkopplade.

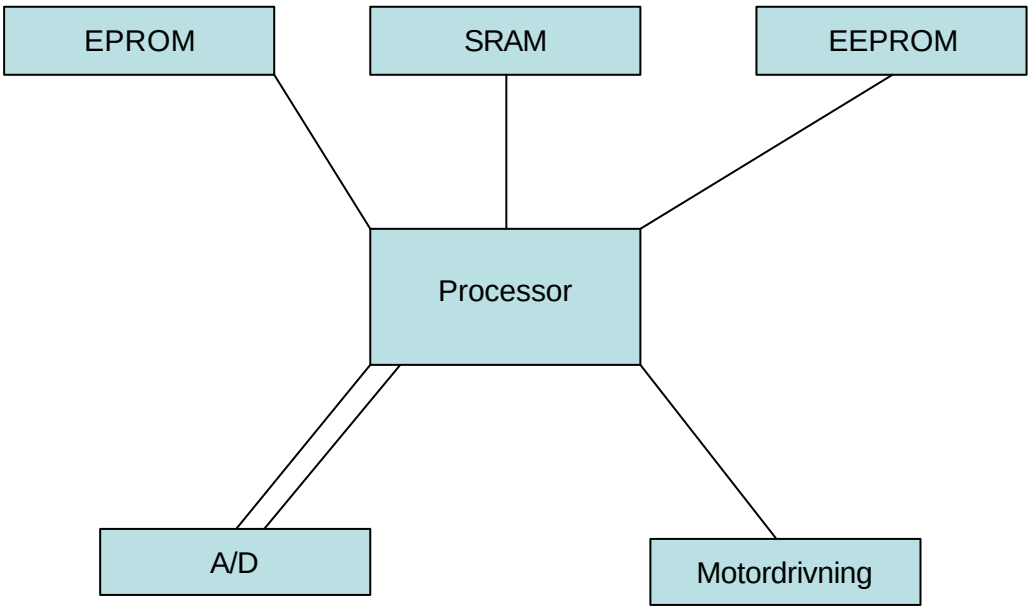
Praktisk modell:

Till projektet behövdes:

- Motorola 68008
- SRAM för datavariabler under körning
- EPROM för programkoden
- A/D-omvandlare för omvandling av de analoga temperaturgivarnas värde till digitala värden
- EEPROM för lagring av aktuella inställningar för de olika temperatursensorerna och fläktarna
- LCD för åskådliggörande av aktuell driftsinformation såsom temperaturer, fläkthastigheter mm
- Motorstyrningsenhet för drivning av fläktarna
- PAL, 2 stycken för adresserings- och styrsignaler
- Fläktar, upp till 8 stöds av konstruktionen
- Temperatursensorer, upp till 8 stöds av konstruktionen
- Klocka, en kristall plus ett antal diskreta komponenter användes för att bygga en enkel klockkrets
- PC-interface, i mån av tid var det önskvärt att bygga ett PC-interface så att man direkt från en dator kunde ändra inställningar samt ta in övervakningsinformation för loggning

Motorstyrning:

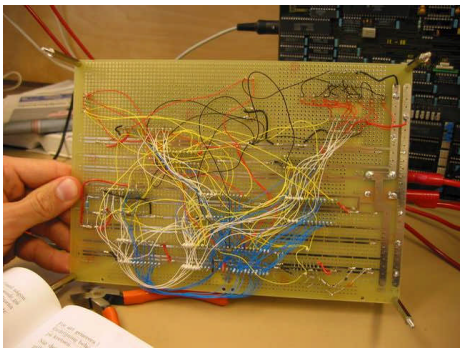
Motorstyrningsenheten byggdes med hjälp av en krets med 8 d-vippor, och en krets med 8 Darlingtonkopplade transistorer i. I CPU:n räknas en intern variabel hela tiden upp ett steg modulu 255. För varje fläkt finns det sedan beräknat hur snabbt den skall snurra i procent räknat. Detta värde omvandlas till ett värde mellan 0 och 255, och sedan är fläkten på så länge loop-variabelns värde är mindre än aktuellt värde, för att sedan stängas av resten av tiden upp till 255.



Principskiss över de olika enheterna

Utförande:

Konstruktionen byggdes i små steg, där allt kunde verifieras efter varje ny komponent. I första steget kopplades processor, PAL1, EPROM samt SRAM ihop och PAL1:an programmerades. Efter en del rättningar i PAL1:an konstaterades att första delen fungerade fint. Att lägga till EEPROM:et var heller inga problem. Det fungerade fint på en gång. Steg tre blev d-vipporna till motorstyrningen. Här lades även till en lysdiod för varje ”motor-utgång” så att motorernas styrsignaler kunde åskådliggöras på ett smidigt sätt. Detta fungerade utmärkt och nästa steg blev PAL2:an och A/D:n. Detta gick inte riktigt lika smidigt utan interrupt:en strulade lite. Efter att ha pausat med LCD:n och fått den att funka, fixades dock även interrupt:en till slut efter diverse omprogrammeringar av PAL2:an.



Baksidan av kortet under arbetets gång

Efter att ha fått all hårdvara att fungera tillfredsställande lades nu all energi på programmeringen. Programstrukturen kan sammanfattas enligt följande:

Huvudprogram:

1: Initiera displayen
2: Läs in relationsdatabasen från EEPROM
3: Beräkna initiala fläktvärden
4: Huvudloop:
5: För varje fläkt:
6: Kolla om den skall vara av eller på vid aktuellt stegvärde
7: Sätt fläktarna till rätt värden genom att ändra d-vipporna
8: Om det är dags att göra en temperaturavläsning:
9: Starta temperaturavläsning för aktuell temperatursensor genom att initiera A/D:n
10: Sätt vilken sensor som skall läsas av nästa gång
11: Öka stegvärdet modulu 255
12: Huvudloop slut

Interrupt-program:

101: Läs in värdet från A/D:n för den sensor som adresserats på rad 9 i huvudprogrammet
102: Uppdatera datastrukturerna för aktuell temperatursensor
103: Beräkna motorsignal för samtliga aktiverade fläktar baserat på samtliga aktiverade temperatursensorer

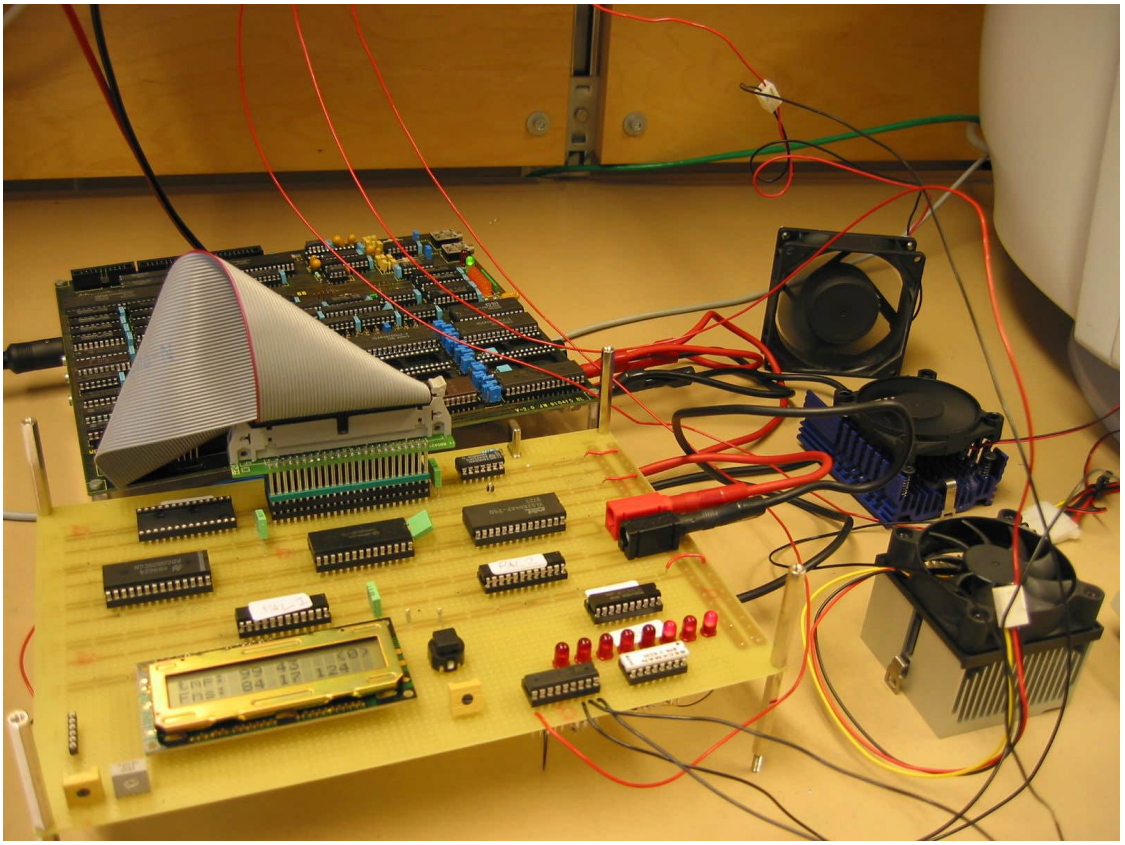
Efter mjukvaruimplementeringen fungerade det mesta utan större problem.

Resultat:

De teorier som utarbetades under utvecklingsfasen visade sig fungera riktigt bra även i praktiken. Det finns dock ett par problem som stötts på under konstruktionens gång:

- Det är svårt att köra fläktar långsamt med pulsning. Fläktarna tenderar att behöva kickstartas (vevas igång manuellt) då de skall snurra långsamt. Är de väl igång går det dock bra.
- Fläktarna har inte kopplats in i ett verkligt system, och det är därför svårt att uppskatta deras kylningseffekt vid olika varvtal. Det har här antagits ett linjärt samband mellan pulslängd och kylningseffekt.
- I konstruktionen har temperatursensorerna ersatts av potentiometrar för enklare handhavande och testning av apparaturen. Det är svårt att förutsäga exakt hur temperatursensorerna hade reagerat. Även här har det t ex antagits ett linjärt samband mellan temperatur och resistans.

Vi hann dessvärre inte med vår planerade utökning i form av PC-kommunikation. Det var synd eftersom det hela lite grann bygger på att man kan ställa in de olika parametrarna efterhand. Som det är nu måste man bränna om EEPROM:et med ny information så fort man kopplar in en ny fläkt t ex.



Slutresultat efter många timmars slit

Sammanfattning:

Kursen har varit mycket intressant och vi har lärt oss en hel del. Vi har båda två lite erfarenheter från diverse mikrokontrollers, så därför tyckte vi att det var skoj att bygga allt från början med databuss, adressbuss mm. Det var kul att resultatet blev så pass bra som det blev. Det var synd att vi inte hann implementera PC-gränssnittet då detta hade givit konstruktionen betydligt större praktisk nytta.