LABORATION B3

P R O J E K T

KONSTRUKTION AV ETT

MIKRODATORSYSTEM

LABORATION B3

PROJEKT

KONSTRUKTION AV ETT MIKRODATORSYSTEM

Detta projekt kommer att ge dig kunskap i hur man konstruerar ett mikrodatorsystem. Systemet skall bestå av processor, minne, serie- och parallellportar. Kretsar från 74HC-familjen skall användas för att generera nödvändiga styrsignaler.

Observera att laborationen kräver nogranna förberedelser innan konstruktionen sätts samman.

Följande dokument måste vara tillgängliga under arbetets gång

Hårdvarubeskrivning för **MD09, ML1, ML2 ML3**.
MOTOROLA Datablad för MC6809-familjen
Datablad över diverse kretsar från 74HC-familjen
Datablad för ROM 2764
Datablad för RWM 6116
Datablad för MAX232

Konstruktören (DU!) står förhållandevis fritt att lösa denna uppgift så länge specifikationen nedan följs.

Först beskrivs uppgiften i stora drag innan den fullständiga specifikationen ges. Sedan beskrivs utvecklingssystemet kortfattat. Slutligen ges hänvisningar till hur ett konstruktionsförfarande *kan* (i ditt fall, *SKA*) gå till.

Det är i mån om tid möjligt att utöka detta projekt. Detta är utmärkt med en stärna (*) i texten.

UPPGIFT

Denna konstruktionsuppgift går ut på att konstruera ett fristående mikrodatorsystem som består av följande: Processor, minne, en serieport och två parallellportar. Detta system skall användas för att styra en display med sex siffror, där man kan visa tid temperatur och dyligt, liknade de man kan se på fasader utomhus. I denna uppgift skall laborationskortet **ML3** användas som display (se hårdvarumanualen för **MD09, ML1, ML2, ML3**).

Systemet skall i första hand kunna styras externt via den seriella porten, varifrån man får uppgifter om vad som skall visas på displayen (t.ex att klockan sätts till 21:53:47 eller att temperaturen sätts till -12.3°C). Utvecklingssystemet (**MD09** och **PC/ETERM**) skall användas för att skicka denna information till din konstruktion.

* Beroende av tidåtgång för konstruktionen kan även klockan kunna ställas lokalt i systemet med hjälp av två tangenter på samma sätt som man ställer in billiga batteridrivna klockor.

Program- och avbrottsrutiner skall konstrueras och slutligen "prommas" för att uppnå de funktioner som är beskrivna ovan.

SPECIFIKATION

Här följer en specifikation av uppgiften. Konstruktionen skall byggas på ett "Single Europa"-kort, med kretsar från MC6809-familjen och 74HC-familjen.

HÅRDVARA

För konstruktionsuppgiften skall användas följande hårdvara:

MC6809	CPU
MC6821	Parallellport
MC6850	Serieport
2764	ROM 8kByte
6116	RAM 2kByte
MAX232	RS232C-driver
	Diverse Logik
* MC6840	TIMER

Adressavkodniig

Adressavkodningen skall vara ofullständig och systemets olika komponenter skall ha följande startadresser:

ROM	$E000
RWM	$C000
PIA	$A000
ACIA	$8000
ACIA	$8010
* TIMER	$8020

Parallellport

Parallellporten (PIA MC6821) skall användas för display. Anslutning för display sker via två st. 10-poliga flatkabelkontakter. Studeras beskrivningen för **ML3** så framgår det att displayet behöver åtta plus sex styrsignaler.

* Detta innebär att PIA:n har två lediga programerbara in/utportar och dessa skall kopplas till var sin tangent för att kunna ställa klockan. Utöver detta skall tangentnertryckningar generera avbrott till processorn.

Serieport

Serieporten (ACIA MC6850) skall ha ett RS232C snitt med BAUDRATE på 4800 BAUD, 8 databitar, ingen paritet, 1 stoppbit. Extern anslutning skall vara 2x5-polig stiftlist. En krets från MAXIM (MAX232) skall användas som drivare / mottagare. Sändar- och mottagarklocka skall genereras från E-klockan.

Kablage: En seriekabel skall kunna anslutas mellan denna stiftlist och **MD09**:s serieport (anslutning K5).

PROGRAMVARA

Programvaran skall *alltid* visa något på displayen (tid, temperatur o dyl.). Den skall växelvis visa tid och tempratur i vardera 3 sekunder. Programvaran skall också känna av om någon ny tid eller temperatur ankommer på serieporten och i så fall visa denna tid/tempratur. Se för övrigt "PROTOKOLL" nedan.

* Programvaran skall känna av tangentnedtryckningar för att kunna ställa klockan lokalt. Systemet skall själv uppdatera klockan varje sekund. Om en ny tid anländer på den seriella porten skall denna tid i förtsättningen användas, trots man själv ställt klockan med tangenterna.

PROTOKOLL

Protokollet som skall användas för att överföra tid och tempratur visas nedan. Det består av ett datablock och ett svarsblock.

DATABLOCK till systemet.

1	2	3						9

Byte 1	SOH	($01)	Start på block
Byte 2	T	($54)	Data är tid
	C	($43)	Data är grader
Byte 3-8	Data		ASCII-kod för tid/tempratur. Mest signifikantab yte överförs först
Byte9	EOT	($04)	Slut på block

SVARSBLOCK från systemet

1

Byte 1	ACK	($06)	Datablock korrekt mottagits
	NAK	($15)	Fel i mottaget datablock
	DC1	($11)	Datablock korrekt mottagits. Jag är utrustad med TIMER

Det är inga tidskrav på datablockets utsträckning

INLEDNING

När man har ett bra utvecklingssystem som **MD09** och **PC/ETERM** är det enkelt att konstruera ett nytt mikrodatorsystem. **MD09** och **PC/ETERM** blir i detta fallet ett "VÄRDSYSTEM" och konstruktionen ett "MÅLSYSTEM".

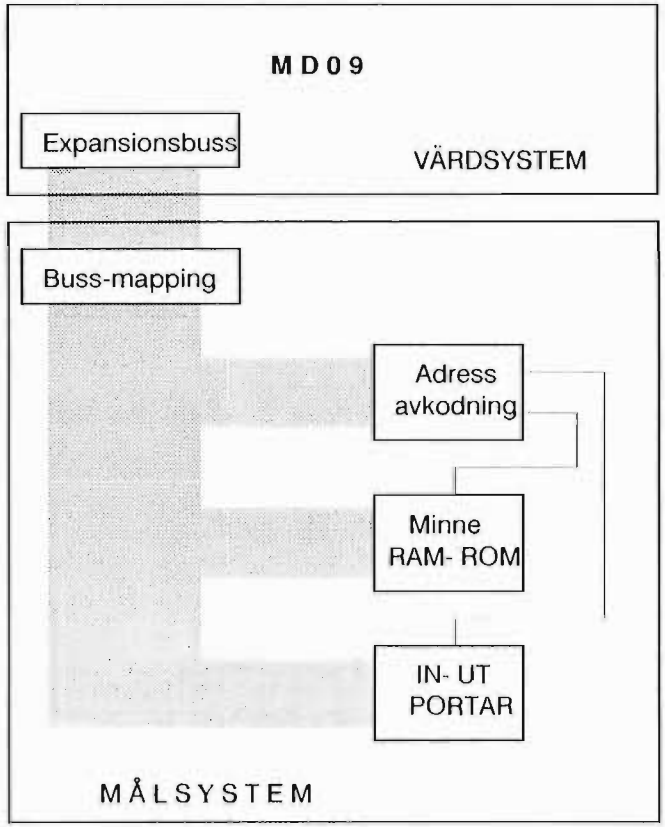
Genom att bygga upp konstruktionen bit för bit via **MD09**:s expansionsbuss har man hela tiden full kontroll på sin konstruktion. Se figur 1 på nästa sida. En arbetsgång som beskrivs nedan kan vara till hjälp.

Innan konstruktionen påbörjas bör du sätta dig inn i hur **MD09**:s expansionsbuss fungerar. Hur du kan expandera denna utan att "krocka" med andra befintliga enheter på **MD09**. Studera noggrant sid 25 i hårdvarubeskrivningen för MD09.

KONSTRUKTIONSFÖRFARANDE

En av de viktigaste reglerna när man konstruerar är att **dokumentera** sin konstruktion. Varje ändring eller ny detalj man inför skall dokumenteras och dateras. Detta sparar mycket tid senare i konstruktionsfasen när problemen kommer (inte *om* problemen kommer, utan *NÄR* problemen kommer).

För att lösa komplexa uppgifter som detta bör man gå till väga på följande sätt som beskrivs i olika steg nedan. Stöter du på problem i hur du skall lösa uppgiften så kan man alltid "snegla" på andra ritningar, t.ex. **MD09**:s .



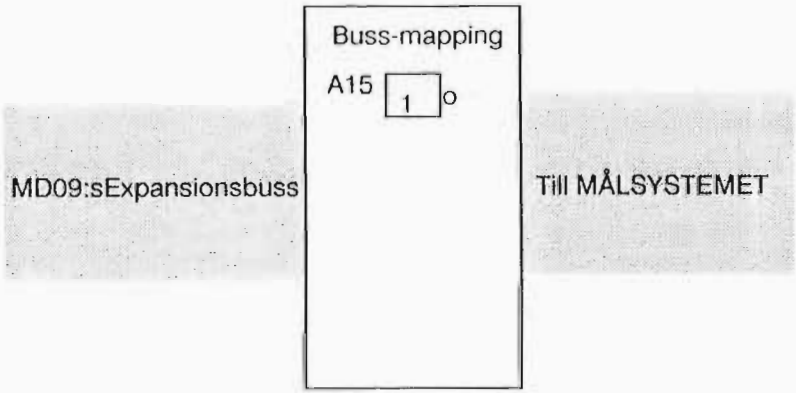
Figur 1 Utvecklingsmiljö

Steg 1

Du bör ha hela (åtminstonde en större del) konstruktionen klar på papper innan du kopplar upp. Du måste "lägga ut" dina kretsar och anslutningar på kortet så att ledningsdragningen blir så liten som möjligt. Å andra sidan är det lättare att vira när det är mycket utrymme mellan kretsarna. Använd därför "hela" kortet.

Steg 2

Koppla **MD09**:s expansionsbuss till målsystemet. Dra separat kraft fram till målsystemet då **MD09**:s expansionsbuss inte kan belastas mycket. *Observera att målsystemets adresser måste "mappas" över i ett annat adressområde.* Ta



Figur 2 Mappningslogik

hänsyn till detta redan nu. **MD09**:s PROM,RAM och I/O-area är i adressområdet $c000 - $ffff, och då målsystemet använder området $8000 - $ffff kan adressbit A15 inverteras för att "mappa" över målsystemets adresser till adressområdet $0000 - $7fff. Se figur 1 och figur 2. På detta sätt kommer du att adressera tex målsystemets PIA:n på adress $2000, sedd från **MD09**.

Steg 3

Konstruera därefter adressavkodning för målsystemet. Använd monitorns funktioner och gör "testloopar" på samma sätt som vid laboration B2 för att mäta dina CHIP SELECT-signaler och kontrollera att dessa är riktiga.

Steg 4

Koppla sedan in RWM:et. Använd monitorns rutiner på samma sätt som vid laboration B2 för att kontrollera att du kan skriva / läsa i detta.

Steg 5

Fortsätt lämpligen med periferikretsarna och då dessa fungerar på bussen är hårdvarukonstruktionen strax klar.

Steg 6

Huruvida du väljer att utveckla programmen i **MD09** eller i målsystemet är upp till dig, men var uppmärksam på att värdsystemet är en känd, fungerande miljö, vilket din konstruktion kanske inte är än.

Steg 7

När programvaran är uttestad och även testad i målsystemet är det dags att bränna PROM som testas i målsystemet. Denna sista fasen kan ofta vara tidsödande då det är svårt med monitorns hjälp att spåra "prommade"-program.
Avslutningsvis frikopplas målsystemet helt och hållet och du har ditt mikrodatorsystem.