DataBoard 4680®

Bygg ut ABC 80 och ABC 800/Facit DTC med DataBoard 4680! För mät- och styr-applikationer.

SATTCO

Sattco AB Dalvägen 10, S-171 36 Solna Tel. 08-734 00 40 Telex S-11588

Teckningar: Siv Våg
Bilder: Sattco, Luxor, Dataindustrier, Computer Press
Granskning och bearbetning: Lennart S.Å. Bergström, Computer Press AB

ISBN 91-40-20486-3 1 2 3 4 5 6 7 8 9 10

LiberTryck Stockholm 1982 208629

Förord

Användningsområdet för de populära bordsdatorerna ABC80, ABC800 och FACIT DTC kan avsevärt utökas med hjälp av DataBoard 4680-systemet, för mät-, styr- och reglertillämpningar. För detta krävs att datorn, med hjälp av en kabel, anslutes till en expansionsenhet i vilken expansionskort av olika typ kan placeras.

DataBoard-systemet är uppbyggt på ett stort antal modulära systemkomponenter kring en generell buss, "4680-bussen". Denna modulära uppbyggnad gör att nya komponenter kan införas utan omkonstruktion av hela systemet och förenklar dessutom både service och felsökning. Varje modul är uppbyggd på ett "Europa-kort" av standardformat, 100 x 160 mm, med buss och I/O-kontakt på respektive kortsida. Dessa moduler kan pluggas in i en expansionsenhet med 4680-bussen i bakplanet, varför ett system av önskad storlek och kapacitet enkelt kan byggas upp. Med DataBoard 4680 får man ett väl utprovat system som har utvecklats och tillverkats i Sverige.

I kapitel 1 beskrivs 4680-bussen och filosofin bakom uppdelningen av denna buss i separata minnes- och I/O-sidor. Figurer i detta kapitel visar även stiftlayouten för datorns anslutningskontakt och expansionskortens busskontakt. Vidare beskrivs generella kommandon för styrning av kort, hur dessa adresseras samt hur expansionsenhetens bakplan byglas för kort som använder interruptfunktionen. I appendix B visas motsvarande figurer för ABC80.

Kapitel 2 uppfriskar glömda kunskaper om de talsystem som man bör behärska för att kunna använda expansionskorten till fullo. Ett BASIC-program som omvandlar mellan de binära och decimala talsystemen kan även återfinnas här.

Kapitel 3 presenterar expansionskort för digital signalering. Den praktiska användningen av varje kort visas med hjälp av ett eller flera program.

Kapitel 4 presenterar expansionskort för analog signalering, dvs kort av typ analog-digital/digital-analog omvandlare, osv.

Om man vill utöka minnet i sin dator är kapitel 5 av intresse eftersom detta behandlar ett antal intressanta minneskort.

Den som är intresserad av automatisk mätning återfinner i kapitel 6 ett avsnitt om IEC-bussen och hur denna kan användas med ABC80/800/DTC. I ett antal programexempel visa hur automatisk mätning kan utföras med hjälp av dator och programmerbara räknare, multimetrar, oscilloskop och LF synthesizer.

Som avslutning beskrivs i kapitel 7 tre intressanta kortmoduler: färgvideo RAM, asynkront serieinterface och prototypkort.

I slutet av boken återfinnes fyra appendix, som innehåller minneskarta för ABC800, hur ABC80 skall byglas för interrupt och stiftlayout för buss-, I/O- och minneskortkontakt för ABC80. Appendix C innehåller ABC80-versioner av de program som ingår i denna bok och appendix D ger slutligen litteraturreferenser för den som vill veta mera.

Denna bok är inte avsedd som någon uppslagsbok för problemlösningar. Dess huvuduppgift är att presentera lämpliga expansionskort ur DataBoard 4680-systemet och ge programexempel på hur dessa kan programmeras och användas. Dessa program är skrivna för ABC800/DTC, medan motsvarande program för ABC80 finns samlade i appendix C.

En viktig avvikelse mellan de tecken som finns på datorns tangentbord och de program som ingår i denna bok måste påpekas. För samtliga programlistningar gäller att:

$ motsvaras av ¤ på datorns tangentbord
§ motsvaras av # på datorns tangentbord

För att få fullt utbyte av boken bör läsaren ha kunskaper i BASIC-programmering och digitalteknik. I vissa avsnitt är också kunskap i assemblerprogrammering en fördel.

Jag vill rikta ett varmt tack till alla medarbetare på SATTCO, PHILIPS, Dataindustrier AB och LUXOR, som gjort det möjligt för mig att skriva denna bok. Jag vill rikta ett speciellt tack till Lennart Bergström, Computer Press AB, för hans granskning och bearbetning av materialet.

Arboga, augusti 1982

Robert Wahlén

Innehållsförteckning

Figurförteckning

1. DataBoard 4680-bussen

4680-Bussen är en generell buss som används av ABC80/800/DTC för att hantera externa minnes- och I/O-kort. Busskontakten finns tillgänglig på baksidan av ABC80/800/DTC. Bussen är uppdelad i två separata halvor:

en **I/O-sida**
och en **minnessida**

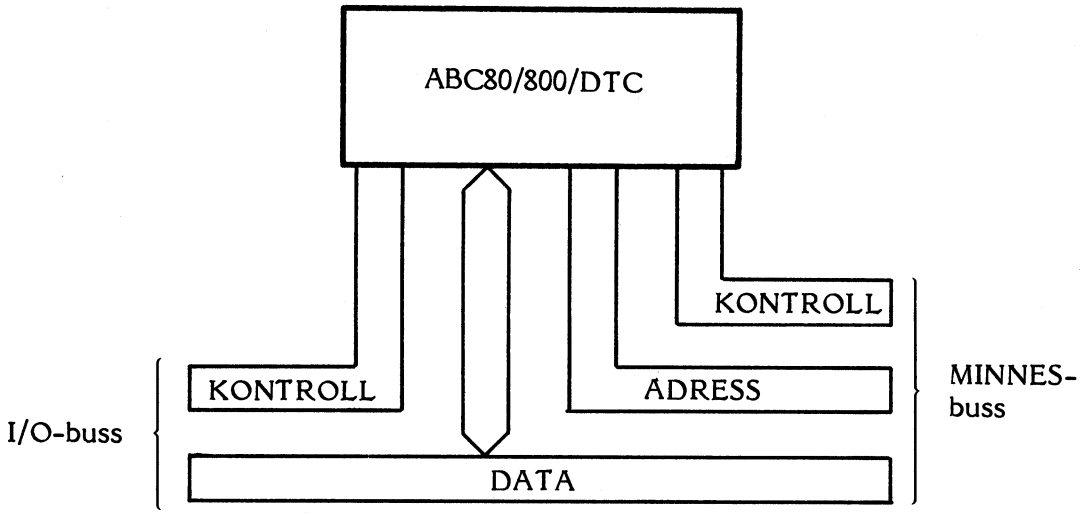


Fig. 1.1 4680-Bussen är uppdelad i en I/O-sida och en minnessida.

4680-Bussen förgrenar sig i en expansionslåda till en minnessida och en separat I/O-sida, där ledarna parallellkopplas med hjälp av ett bakplan till flera kortplatser, fig. 1.2. Kabeln anslutes till en särskild position, mitt mellan dessa sidor, i expansionslådan.

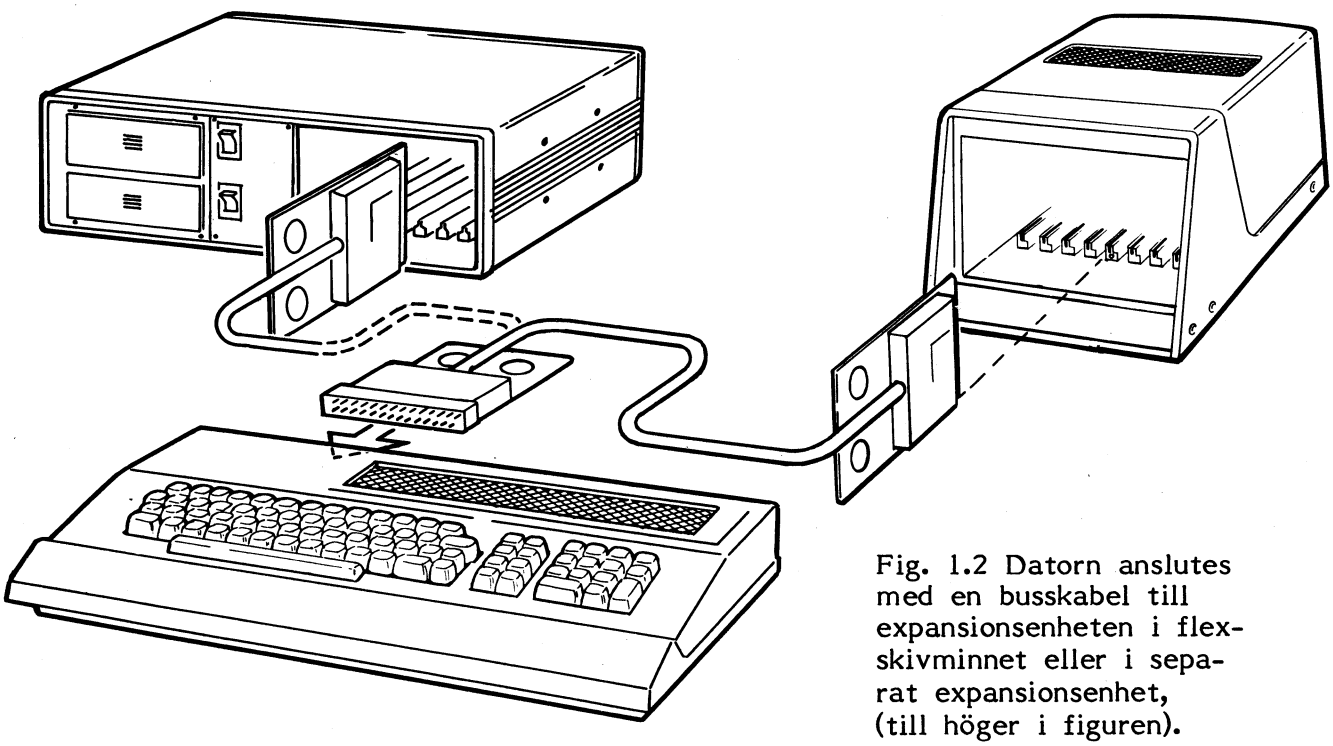


Fig. 1.2 Datorn anslutes med en busskabel till expansionsenheten i flexskivminnet eller i separat expansionsenhet, (till höger i figuren).

Signaleringen är av parallellsnitt genom särskilda kontrollsignaler som skickas till respektive sida (I/O- eller minnes-sidan).

Kortmodulerna som placeras i expansionsenheten konstrueras med i stort sett samma anslutningssnitt till 4680-bussen som kontakten från datorn. Skillnaderna kan ses i fig. 1.6, fig. 1.7 och fig. 1.8 som visar den fysiska utformningen av kontakterna till bakplanet.

Korten i sig själva består av tre huvudelement:

* Adressavkodning
* Bussdrivning (tri-state typ)
* Logik för klarsignalering

Adressavkodning

Alla kort på I/O-sidan har samtidig tillgång till samtliga styrsignaler vilket gör det nödvändigt att konstruera en mekanism genom vilken datorn kan aktivera ett och endast ett av dessa kort som då kan styras med de gemensamma styrsignalerna på bussen. Denna mekanism kallas **kortval**. Genom att öppna eller kortsluta 12 stift i en DIP-sockel på kortet, s.k. bygling, så kan man tilldela varje kort en egen "nyckel", dvs en adress. Hur man gör detta framgår av avsnittet "Adressering av I/O- och minneskort" nedan.

Bussdrivning

Bussdrivningen är av tri-state typ vilket medför att kortets signalledningar kan förläggas i ett aktivt tillstånd eller ett flytande tillstånd gentemot bussen. Detta senare tillstånd är nödvändigt för att flera kort skall kunna användas samtidigt på bussen. I samband med att ett kort väljes av datorn förlägges det valda kortets signalledningar i aktivt tillstånd medan alla andra kort flyter på bussen.

Logik för klarsignalering

Logiken för klarsignaleringen sköter styrningen av bussdrivningen samt håller ordning på vad som kommer och vad som skall sändas. De flesta signaler för klarsignaleringen (eller kontrollsignaleringen) är aktiva låga. Detta markeras i figurer med en asterisk (*) eller ett streck över signalen.

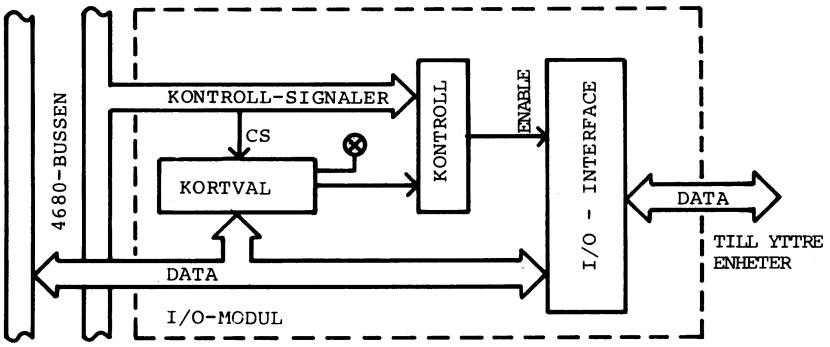


Fig. 1.3 I/O-kort för 4680-bussen.

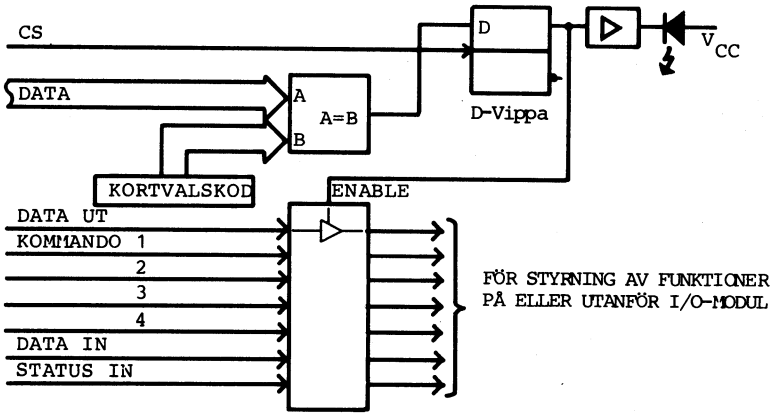


Fig. 1.4 Logik för klarsignalering.

Figuren 1.3 visar logiken på ett I/O-kort, med inkommande buss-signaler från vänster. Vid kortval lägger datorn ut en kod, dvs en 6-bitars adress, på databussen samtidigt som en styrsignal, "CS", genereras på en separat bussledning. CS-signalen medför att samtliga korts kortvals-logik jämför sin egen adress mot den adress som finns på databussen. Det kort som upptäcker likhet tänder en lysdiod för att markera att det nu är utvalt, och aktiverar samtidigt kortets logik. Figuren 1.4 visar detta i detalj.

Expansionsenhet LUXOR/FACIT

I expansionsenheten förgrenar sig 4680-bussen via bakplanet till:

4 kortplatser för I/O-kort
3 kortplatser för minnes-kort

Bakplanet är ett vedertaget uttryck för den bakre delen av expansionsenheten, alltså den del där korten trycks in och får kontakt med 4680-bussen.

Som synes i fig. 1.5 har bakplanet 8 kortplatser. Den 5:e kortplatsen från vänster är reserverad för kontakten från datorn. Den platsen är markerad med en lysdiod. Dioden har till uppgift att indikerar "power on" och visa var kontakten skall placeras. Bussen förgrenar sig sedan parallellt via bakplanet till de olika kortplatserna. Till vänster om kontakten skall I/O-kort placeras och till höger skall minneskort placeras. Fig. 1.9 visar hur signalerna förgrenar sig och går till de olika kortplatserna. Till vänster om kontakten från datorn och I/O-kortens kortplatser finns byglingar som skall ställas om när man använder kort med interrupt. Hur det fungerar beskrivs i avsnittet "Bygling för interrupt" nedan.

En finess som är värd att nämnas är att expansionsenhetens 4680-buss är densamma som utnyttjas av enkortsdatorerna i Databoard 4680-systemet. Dessa enkortsdatorer, (DataBoard 1001, 1003 och 1004), sätts in på samma plats som man annars ansluter busskontakten från ABC80/800/DTC. Expansionsenheten blir sålunda en självständig dator, som kan användas för de mest skiftande tillämpningar.

Fig. 1.6 visar stiftlayouten i den anslutningskontakt som förbinder expansionsenhetens bakplan med datorn, (ABC800/DTC). Figur 1.7 och 1.8 visar stiftlayouten hos kontakterna på I/O- respektive minnessidan i expansionsenheten. Motsvarande figurer för ABC80 kan återfinnas i Appendix B.

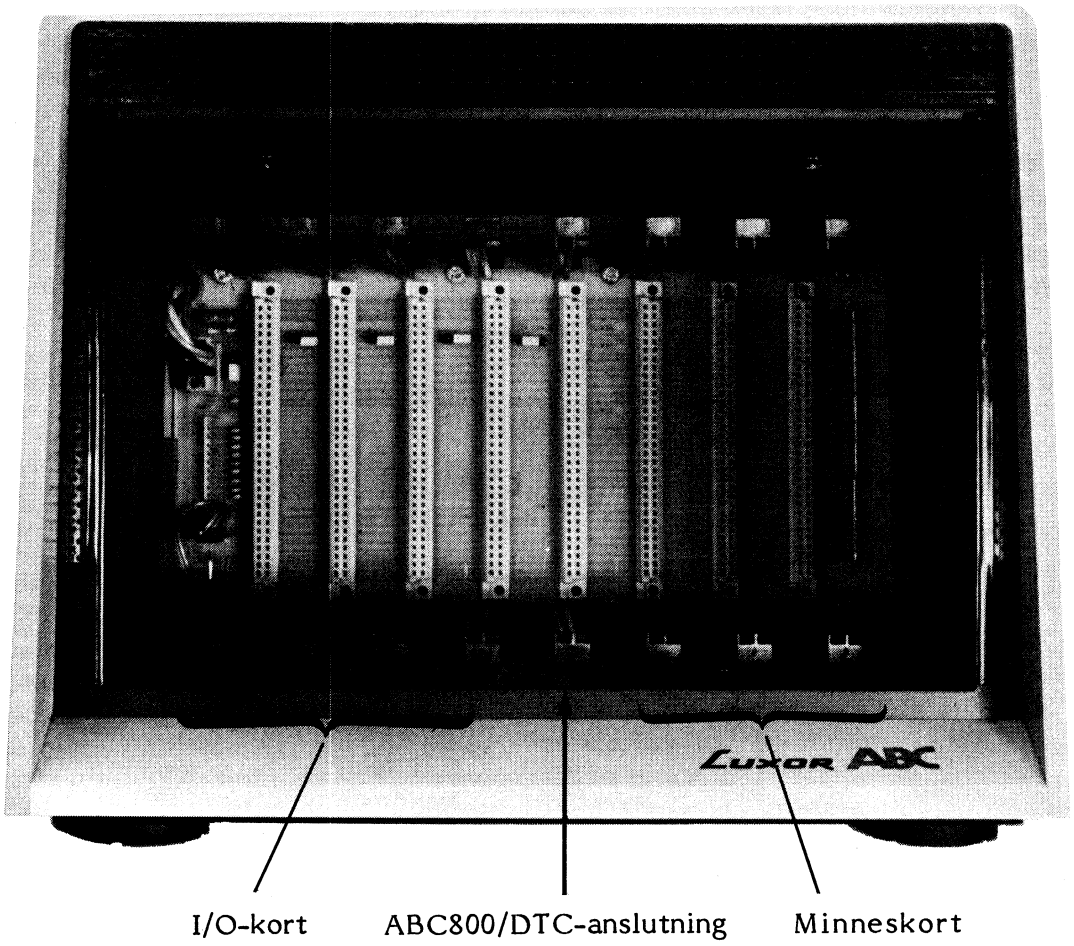


Fig. 1.5 Expansionsenhet LUXOR/FACIT.

		B	A		
	-12 V	.	1 .	-12 V	
	0 V	.	2 .	0 V	
	MEMWR*	.	3 .	RESIN*	
	XMEMFL*	.	4 .		
	Ø	.	5 .	INT*	
		.	6 .	D7	
		.	7 .	D6	
		.	8 .	D5	
		.	9 .	D4	
		.	10 .	D3	Databuss
		.	11 .	D2	
		.	12 .	D1	
	0 V	.	13 .	D0	
	A15	.	14 .		
	A14	.	15 .	RST*	
	A13	.	16 .	1 STAT*	(IN-port)
	A12	.	17 .	0 INP*	(IN-port)
	A11	.	18 .	5 C4*	(UT-port)
	A10	.	19 .	4 C3*	(UT-port)
	A9	.	20 .	3 C2*	(UT-port)
Adress-	A8	.	21 .	2 C1*	(UT-port)
buss	A7	.	22 .	0 OUT*	(UT-port)
	A6	.	23 .	1 CS*	(UT-port)
	A5	.	24 .	NMI*	
	A4	.	25 .	INP2*	
	A3	.	26 .	XINPSTB*	
	A2	.	27 .	XOUTPSTB*	
	A1	.	28 .	XM*	
	A0	.	29 .	RFSH*	
	MEMRQ*	.	30 .	RDY	
	+5 V	.	31 .	+5 V	
	+12 V	.	32 .	+12 V	

Fig. 1.6 Stiftlayout för anslutningskontakten från ABC800/DTC.

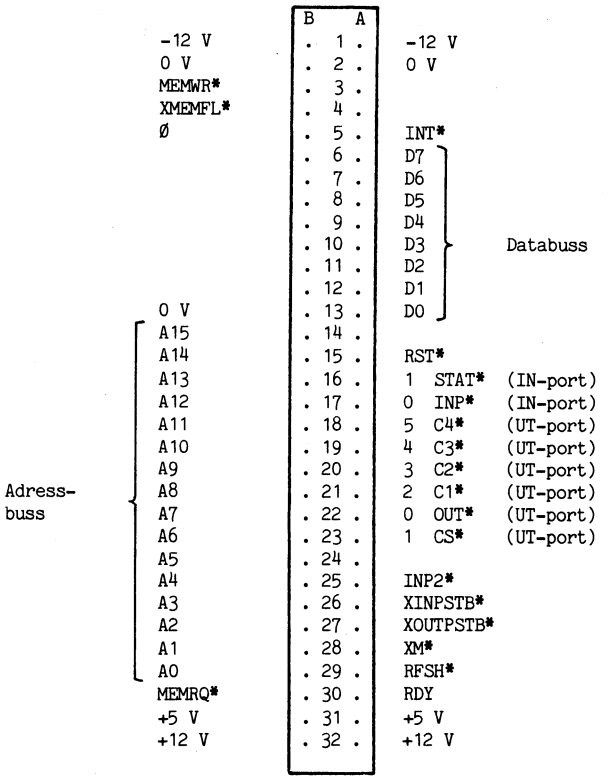


Fig. 1.7 Stiftlayout för kontakt på I/O-sida.

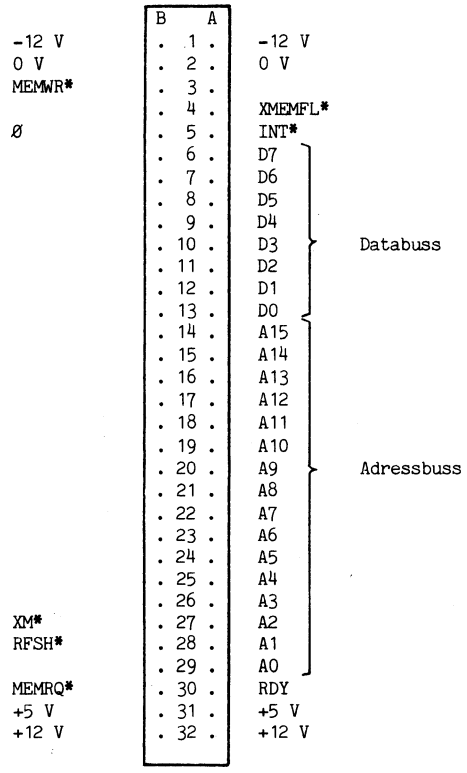


Fig. 1.8 Stiftlayout för kontakt på minnes-sida.

Ledningsdragning i bakplan från CPU-positionens A-sida

ABC 80	ABC 800	T	CPU	MINNE	I/O SPEC	I/O	SIGNAL	ANM
-	-	-	1A,1B	1A,1B	1A,1B	1A,1B	-12 V	
2A,2B	2A,2B	-	2A,2B	2A,2B	2A,2B	2A,2B	GND	
3A	3A	*	3A	-	-	-	RESIN	Lödyta vid CPU
-	-	-	4A	-	-	-	-	
5A	3B	*	5A	5A	5A	-	XMEMW	
6A	6A	0	6A	6A	6A	6A	D7	
7A	7A	0	7A	7A	7A	7A	D6	
8A	8A	0	8A	8A	8A	8A	D5	
9A	9A	0	9A	9A	9A	9A	D4	
10A	10A	0	10A	10A	10A	10A	D3	
11A	11A	0	11A	11A	11A	11A	D2	
12A	12A	0	12A	12A	12A	12A	D1	
13A	13A	0	13A	13A	13A	13A	D0	
14A	14A	*	14A	-	-	-	-	
15A	15A	*	15A	(15B)	15A	15A	RST	330 Ohm t. +5 V
16A	16A	*	16A	-	16A	16A	STAT	
17A	17A	*	17A	-	17A	17A	INP	
18A	18A	*	18A	-	18A	18A	C4	
19A	19A	*	19A	-	19A	19A	C3	
20A	20A	*	20A	-	20A	20A	C2	
21A	21A	*	21A	-	21A	21A	C1	
22A	22A	*	22A	-	22A	22A	UTP	
23A	23A	*	23A	-	23A	23A	CS	
24A	24A	*	24A	-	-	-	NMI	Lödyta vid CPU
25A	25A	*	25A	-	25A	25A	INP2	
26A	26A	*	26A	(4B)	26A	26A	XINPSTB	
27A	27A	*	27A	(26B)	27A	27A	XOUTSTB	
-	-	-	28A	27B	28A	28A	XM	Se anm 1
29A	29A	*	29A	8B,28B	29A	29A	RFSH	
30A	30A	*	30A	30A	30A	30A	RDY	
-	-	-	31A,31B	31A,31B	31A,31B	31A,31B	+5 V	
-	-	-	32A,32B	32A,32B	32A,32B	32A,32B	+12 V	

Anm 1 = Denna signal kan genereras på kabelkort

Ledningsdragning i bakplan från CPU-positionens B-sida

ABC 80	ABC 800	T	CPU	MINNE	I/O SPEC	I/O	SIGNAL	ANM
-	-	-	1A,1B	1A,1B	1A,1B	1A,1B	-12 V	
2A,2B	2A,2B	-	2A,2B	2A,2B	2A,2B	2A,2B	GND	
3B	-	*	3B	3B	3B	-	KILL DOG	Lödyta vid CPU
4B	4B	*	4B	4A	4B	-	MEMFL	
5B	5B	*	5B	5B	5B	-	CLOCK	
6B	6B	*	6B	6B	-	-	MUX	Gen. av mem map
7B	7B	*	7B	7B	-	-	RASST	Gen. av mem map
-	-	-	8B	-	-	-	-	
-	-	-	9B	-	-	-	-	
-	-	-	10B	-	-	-	-	
-	-	-	11B	-	-	-	-	
12B	12B	*	12B	23B	-	-	CSTOP	Gen. av mem map
13B	5A	*	13B	-	13B	A5	INT	
14B	14B	*	14B	14A	14B	-	A15	
15B	15B	*	15B	15A	15B	-	A14	
16B	16B	*	16B	16A	16B	-	A13	
17B	17B	*	17B	17A	17B	-	A12	
18B	18B	*	18B	18A	18B	-	A11	
19B	19B	*	19B	19A	19B	-	A10	
20B	20B	*	20B	20A	20B	-	A9	
21B	21B	*	21B	21A	21B	-	A8	
22B	22B	*	22B	22A	22B	-	A7	
23B	23B	*	23B	23A	23B	-	A6	
24B	24B	*	24B	24A	24B	-	A5	
25B	25B	*	25B	25A	25B	-	A4	
26B	26B	*	26B	26A	26B	-	A3	
27B	27B	*	27B	27A	27B	-	A2	
28B	28B	*	28B	28A	28B	-	A1	
29B	29B	*	29B	29A	29B	-	A0	
30B	30B	*	30B	30B	30B	-	MEMRD	
-	-	-	31A,31B	31A,31B	31A,31B	31A,31B	+5 V	
-	-	-	32A,32B	32A,32B	32A,32B	32A,32B	+12 V	

() = Ej inkopplad vid leverans
- = Ej inkopplad
0 = Förslag att ta bort denna terminering
* = Terminering 220 Ohm 680 pF i serie till GND

Fig. 1.9 4680-bussens bakplan.

Kommandon för styrning av kort

För att kommunicera med I/O-korten finns det två kommandon tillgängliga, **OUT** och **INP.**

Det första man bör göra i ett program som använder I/O-kort är att nollställa alla I/O-kort så att de kommer i rätt utgångsläge. Detta gör man med att skriva:

```
10 PRINT INP (7)
```

eller

```
10 Z = INP (7)
```

INP (7) skiljer sig därvid från övriga INP-kommandon, som har som uppgift att läsa data från en port. INP (7) sänder istället ut en reset (RST) signal som nollställer alla kort.

När man har nollställt alla kort skall man välja ut det kort man skall arbeta med. Det gör man genom att sända ut en CS-signal (Card Select). I basic skrivs detta:

```
20 OUT 1,<kortadress>
```

Se vidare nedan under avsnittet "Adressering av I/O- och minneskort".

Nu kan man börja att skicka ut data med kommandot:

```
30 OUT 0,Data
```

"Data" är här ett tal mellan 0 och 255. Om talet är större så skickas endast de åtta minst signifikanta bitarna ut på data-bussen. Detta beror på att 4680-bussen har en bredd av 8 bitar, (en Byte). För att skicka iväg både de åtta minst signifikanta bitarna och de åtta mest signifikanta bitarna skriver man:

```
40 OUT 0,Data
50 OUT 2,SWAP%(Data)
```

Man kan även slå ihop de två satserna till en och då blir det:

```
40 OUT 0,Data,2,SWAP%(Data)
```

Utöver dessa kommandon så finns det ytterligare några kommandon som skickar ut signaler till korten. Funktionen hos dessa kommandon är olika för olika kort varför måste man läsa specifikationerna för att förstå hur det aktuella kortet hanterar signalen. Kommandona är:

```
OUT 2,<Kommando C1>
OUT 3,<Kommando C2>
OUT 4,<Kommando C3>
OUT 5,<Kommando C4>
```

För att läsa inkommande data från bussen skriver man:

```
10 PRINT INP (0)
```

eller om vi tilldelar inläst data till en variabel:

```
10 A=INP (0)
```

		Bygel				
		B1	B2	B3	B4	B5
Kortplats	K1	-	X	X	X	X
"	K2	-	-	X	X	X
"	K3	-	-	-	X	X
"	K4	-	-	-	-	X

X = Bygel flyttas till slutet läge
- = Bygel flyttas till brutet läge

B1 är BRUTEN i det nedre läget
B2-B5 är BRUTNA i det högra läget

Observera! För att undvika kortslutning måste **minst en** bygel vara bruten.

Fig. 1.13 Byglingstabell för expansionsenhet, (för ABC80 se motsvarande figur i Appendix B).

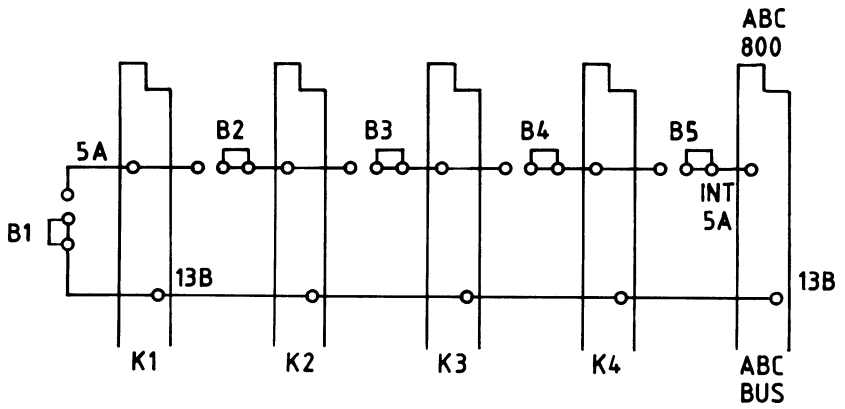


Fig. 1.14 Byglar för interrupt med LUXOR/FACIT expansionsenhet.

Bygling för interrupt på DATADISC

För att få fram interruptsignalen till kort på DATADISC måste man löda på bakplanet. På dess bakplan finns två lödöar, (utgående från datorns anslutningskontakt), som man kan få interruptsignal från. Vilken av dessa två man skall välja beror på vilken dator som används, (ABC80 eller ABC800/DTC). För att få interruptsignal i ABC80-tillämpningar skall man ansluta den lödö, som är ansluten till stift 13B på datorns anslutningskontakt, till den lödö, som är ansluten till stift 5A på önskad I/O-plats, (OBS ej 5A på datorns anslutningskontakt!). I ABC800/DTC-tillämpningar skall man ansluta den lödö, som är ansluten till stift 5A på datorns anslutningskontakt, till den lödö som är ansluten till stift 5A på önskad I/O-kortplats, se fig. 1.9.

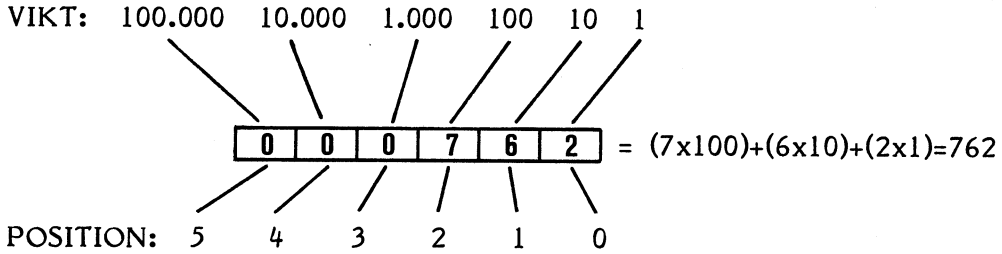
2. Talsystem

Decimala talsystemet

Det talsystem, som vi människor använder oss av i dagligt bruk, kallas för det decimala talsystemet. Det består av tio siffror: 0, 1, 2, 3, 4, 5, 6, 7, 8 och 9. Vi tycker att det är mest logiskt att använda tio siffror, men vi kunde lika gärna ha använt oss av 8, 12, 16 eller hur många som helst! Att vi människor använder ett talsystem med 10 som bas antas bero på att vi har tio fingrar och att dessa fingrar spelat en stor roll vid räknandet...

Låt oss ta ett decimalt tal, t.ex.: 762. I detta tal vet vi att 2 står för två ental, 6 för sex tiotal och 7 för sju hundratal. Detta ser vi på siffrornas placering i förhållande till varandra, dvs deras positioner. Det är alltså ett positionssystem. Vidare vet vi att varje position kan ha siffrorna 0 till 9, alltså tio siffror. Därav namnet decimalt talsystem, som betyder tio-system.

I talsystem har varje sifferposition en viss vikt. I det decimala systemet är vikterna 1, 10, 100, 1000, osv, som du ser i figuren nedan:



Vikterna är lika med talsystemets bas upphöjt till sifferpositionens nummer:

VIKT		
1	=	10^0
10	=	10^1
100	=	10^2
1000	=	10^3

POSITION

BAS

I ett talsystem med basen 2 får vi på motsvarande sätt vikterna 1, 2, 4, 8 osv, enligt:

1	=	2^0
2	=	2^1
4	=	2^2
8	=	2^3

I ett talsystem med basen 16 får vi vikterna 1, 16, 256, 4096 osv, enligt:

1	=	16^0
16	=	16^1
256	=	16^2
4096	=	16^3

* Ett talsystem med basen 2 kallas det **binära** talsystemet.
* Ett talsystem med basen 10 kallas det **decimala** talsystemet.
* Ett talsystem med basen 16 kallas det **hexadecimala** talsystemet.

I det decimala talsystemet använder vi oss av de tio siffrorna 0 till 9. Dvs lika många siffror som basen i talsystemet. I det binära talsystemet använder vi siffrorna 0 och 1, medan vi i det hexadecimala fallet, skulle vi behöva 16 siffror eftersom basen är 16. Men så många siffror har vi ju inte!? Ja, eftersom vi bara har 10 siffror av våra vanliga, så får vi lov att hitta på symboler för de resterande 6! Då har man helt enkelt valt de sex första bokstäverna i alfabetet, som med andra ord blivit siffror i det hexadecimala talsystemet!

* Binära talsystemet, bas 2, med siffror: 0 och 1.
* Decimala talsystemet, bas 10: 0, 1, 2, 3, 4, 5, 6, 7, 8, och 9.
* Hexadecimala talsystemet, bas 16: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Binära talsystemet

Det decimala talsystemet är mindre lämpat för maskinell bearbetning. Varje siffra måste kunna representeras med någon elektrisk storhet, t.ex en spänning. Med det decimala systemet skulle man då behöva ha 10 olika spänningar för att representera alla siffrorna 0 till 9. T.ex 1 Volt skulle representera siffran 1, 2 Volt siffran 2, osv. Men vad händer om maskinens matningsspänning varierar? Sjunker den 1 Volt blir alla siffrorna fel! Det är alltså ingen bra lösning. Det är därför man i sådana sammanhang uteslutande använder sig av det binära talsystemet. Där har man, som du kommer ihåg, bara två siffror: 0 och 1. Två siffror är det inga problem att representera. Man kan ju bestämma att om vi har ström i en ledare har vi siffran 1, medan ingen ström i ledaren betyder siffran 0. Eller också kan vi bestämma att 0 Volt representarar siffran 0 och 5 Volt siffran 1. Nu kan man låta spänningen sjunka till 3 Volt och ändå ha en "etta", medan på samma sätt kan "nollans" spänning få stiga till 1 Volt och vi fortfarande har en "nolla".

Vikterna i det binära talsystemet är 1, 2, 4, 8, 16, 64, 128, osv. Du ser att vikten fördubblas för varje position, så det är lätt att komma ihåg. Som i det decimala systemet är det positionen längst till höger som har lägsta vikt!

Position:	7	6	5	4	3	2	1	0
Vikt:	128	64	32	16	8	4	2	1
Binärt tal:	1	0	1	1	0	1	1	0

För att beräkna det binära talets decimala värde multiplicerar du siffran med dess vikt och adderar samtliga dessa produkter:

(1x128)+(0x64)+(1x32)+(1x16)+(0x8)+(1x4)+(1x2)+(0x1)=182

Program för omvandling mellan de decimala och binära talsystemen.

```
30  ! OMVANDLING AV DECIMALTAL TILL BINÄRTAL
40  ! OCH BINÄRTAL TILL DECIMALTAL
50  !
60  EXTEND : FLOAT
70  ; CHR$(12)
80  ;  TAB(20) " TALOMVANDLING "
90  ;  CUR(5,13) "1. OMVANDLA DECIMALTAL TILL BINÄRTAL"
100 ; CUR(7,13) "2. OMVANDLA BINÄRTAL TILL DECIMALTAL"
110 ; CUR(9,13) "3. AVSLUTA"
120 ; CUR(11,15) "VAD ÖNSKAS"; : INPUT Svar
130 IF Svar<1 OR Svar>3 THEN GOTO 10
140 ON Svar GOTO 180,220,260,10
150 !
160 ! DECIMALT-BINÄRT
170 !
180 Z$=FNBinär$ : ; CUR(19,10) Binär$ : GOTO 560
190 !
200 ! BINÄRT-DECIMALT
210 !
220 Z$=FNDecimal$ : ; CUR(19,10) Decimal$ : GOTO 560
230 !
240 ! AVSLUTA
250 !
260 ; CHR$(12)
270 END
280 !
290 DEF FNBinär$
300 !
310 ; CUR(15,0) SPACE$(80)
320 ; CUR(15,5) "ANGE DECIMALTALET (MAX 65535)"; : INPUT Decimal
330 Binär$=""
340 FOR I=15 TO 0 STEP -1
350 Binär$=Binär$+CHR$((Decimal AND 2üI)/2üI+48)
360 NEXT I
370 Binär$="TALET BLEV "+Binär$
380 RETURN Binär$
390 FNEND
400 !
410 DEF FNDecimal$
420 !
430 ; CUR(15,0) SPACE$(80)
440 ; CUR(15,5) "ANGE BINÄRTALET (MAX 1111 1111 1111 1111)"; :INPUT Binär$
450 I=1 : Antal=LEN(Binär$)
460 Decimal=0
470 WHILE I<Antal+1
480   Plats=INSTR(I,Binär$,"1")
490   Decimal=Decimal+2ü(Antal-Plats)
500 I=Plats+1
510 WEND
520 Decimal$=NUM$(Decimal)
530 RETURN Decimal$
540 FNEND
550 !
560 ! FLER TAL
570 !
580 ; CUR(15,0) SPACE$(80)
590 ; CUR(15,5) "FLER TALOMVANDLINGAR (J/N) <Ja>"; : INPUT Slask$
600 IF Slask$="" GOTO 10
```

```
610 IF CHR$(ASCII(Slask$) AND 95)="J" GOTO 10
620 GOTO 260
```

Några kommentarer till ovanstående program, som omvandlar mellan de decimala och binära talsystemen:

Rad 60-140
: hanterar val av funktion, (meny).

Rad 180
: skriver ut resultatet av beräkningarna gjorda på raderna 290-390 (omvandlingen decimal - binär).

Rad 220
: skriver ut resultatet av beräkningarna gjorda på raderna 430 - 540 (omvandlingen binär - decimal).

Rad 340 - 360
: utför omvandlingen decimal - binär. Genom att maska det inmatade talet med de sexton första jämna binära talen med "AND" och sedan dividera resultat "0" beroende på om det binära talet finns med eller ej. Adderar man sedan ihop ettorna och nollorna till en sträng (Binär$) så har man resultatet.

Rad 470 - 510
: omvandlar ett inmatat binärt tal till ett decimaltal. Genom att leta reda på ettorna i binärtalet och räkna ut deras värde (man vet ju var de står) kan man sedan addera ihop det till ett decimalt värde (decimal).

Bit, nibble och byte

"Binär siffra" heter på engelska: **binary digit,** vilket förkortas **bit.** Ett binärt tal består sålunda av ett antal bits. I datorsammanhang så har man benämningar även på grupper av bits:

* Fyra bits brukar kallas en **nibble,** (eng. uttalas "nibbel").
* Åtta bits brukar kallas en **byte,** (eng. uttalas "bajt").

Hexadecimala talsystemet

Det binära talsystemet är inte så lämpligt för oss människor. Det blir besvärligt att hålla reda på alla nollor och ettor och det blir svårt att få en uppfattning hur stort talet är. En hjälp i nöden är det hexadecimala talsystemet, som med det binära talet som grund representerar detta med ett mindre antal sifferpositioner.

Som du sett tidigare är vikterna 1, 16, 256, 4096, osv. Siffrorna vi använder oss av är våra vanliga 0 till 9 samt bokstäverna A, B, C, D, E och F, som har upphöjts till siffror i det hexadecimala talsystemet. Låt oss se ett exempel på ett HEX-tal:

9E

Vad är detta decimalt? (9 x 16) + (E x 1) = ? Det blev inte så enkelt att räkna ut eftersom vi måste veta "hur mycket E är". De första HEX-siffrorna är våra vanliga 0-9, varefter följer A-F i bokstavsordning:

1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
									10	11	12	13	14	15

HEX-siffrorna A-F:s decimala motsvarighet blir då som ovan, dvs E =14. Nu kan vi fortsätta beräkningen:

(9x16) + (14x1) = 158, dvs 9E = 158

Hur kan då HEX-talen hjälpa oss att hantera binära tal?

Med två HEX-siffror kan vi representera ett tal som det går åtta bits i det binära talsystemet för att representera. Eller med andra ord: En byte kan skrivas med två siffror i HEX-systemet! Låt oss så ta en byte och se hur vi kan skriva talet på det sättet.

Det binära talet 10110110 delar vi upp i nibbles: 1011 0110. Vi omvandlar därefter varje nibble var för sig till ett decimalt tal:

1011=(1x8)+(0x4)+(1x2)+(1x1)=11
0110=(0x8)+(1x4)+(1x2)+(0x1)= 6

Omvandla nu det decimala talet till HEX-tal och vi får då att

1011 = 11 = B
0110 = 6 = 6

Det binära talet 10110110 har nu omvandlats till det hexadecimala talet B6. Och visst är B6 lättare att hantera än 10110110?

Maskning av decimala tal.

När man maskar två tal så utför man en binär räkneoperation. För att förstå hur maskning går till bör man först göra om talen till binära och sedan jämföra dessa bit för bit. De olika maskningskommandona som man har tillgång till hos ABC80/800/DTC är NOT, AND, OR, XOR, (i fallande prioritetsordning förutom OR och XOR som har samma prioritet). Följande exempel redovisar hur det går till:

AND-operation:

	Decimaltal	Binärtal
	11	0000 1011
AND	5	0000 0101
	1	0000 0001

För att man skall få en etta vid AND måste båda bitarna vara ettställda. 1 AND 1 = 1; 1 AND 0 = 0; 0 AND 0 = 0.

OR-operation:

	11	0000 1011
OR	5	0000 0101
	15	0000 1111

Vid OR måste minst en bit vara ett för att vi skall få en etta, dvs 1 OR 1= 1; 1 OR 0 = 1; 0 OR 0 =0.

XOR-operation:

```
      11          0000 1011
XOR    5          0000 0101
-------------------------------
      14          0000 1110
```

Däremot skall vid XOR endast en bit vara ettställd för att ge en etta, dvs 1 XOR 1 = 0; 1 XOR 0 = 1; 0 XOR 0 = 0.

NOT-operation:

```
NOT   11          0000 1011
-------------------------------
      244         1111 0100
```

Till skillnad från AND, OR och XOR opererar NOT-operatorn enbart på en enda siffra, dvs NOT 1 = 0; NOT 0 = 1.

Om man vill kontrollera att en bit är satt och vill ha svaret "1" (sant) eller "0" falskt, så dividerar man resultatet av en maskning med AND, med värdet av den bit man vill ha kontrollerad.

Om man vill kontrollera att bit fyra är satt i talet 35 så skriver man:

```
PRINT (35 AND 2ü4)/2ü4
```

dvs datorn utför följande binära AND-operation:

```
      35          0010 0011
AND   16          0001 0000
-------------------------------
       0          0000 0000
```

Svaret blev 0, dvs bit fyra är inte satt.

Vi kallar den minst signifikanta biten, (den längst till höger) för bit noll. Då stämmer positionen och potensen överens.

Om man vill undersöka bit fem istället, så skriver man:

```
PRINT (35 AND 2ü5)/2ü5
```

dvs datorn utför följande binära AND-operation:

```
      35          0010 0011
AND   32          0010 0000
-------------------------------
      32          0010 0000
```

Svaret blev 1, dvs bit fem är satt.

3. Digital signalering

I följande avsnitt presenteras kortmoduler för digital signalering. Kommandon och signalsnitt i denna serie är i många fall kompatibla, vilket underlättar programmering och utbytbarheten mellan modulerna.

Till varje kortmodul återfinns ett eller flera programexempel som visar hur modulen kan programmeras och användas.

4005

2x8 Open-collector TTL-utgångar
1x8 TTL-ingångar

4006

4x8 Tri-state TTL-utgångar
2x8 TTL-ingångar

4008

16 Opto-ingångar för digital signalering i svår miljö.

4011

Som 4008 men med annat inspänningsområde.

4085

36 TTL-ingångar
2 TTL-utgångar

4095

16 Opto-utgångar

4103

16 Relä-utgångar med växlande kontakt.

3.1 Digital signalering med I/O-kort 4005 och 4006

För digital signalering finns I/O-korten 4005 och 4006, med följande specifikationer:

4005

Utgångar: 16 TTL Open-collector
Ingångar: 8 TTL

4006

Utgångar: 32 TTL tri-state
Ingångar: 16 TTL

Digitalt I/O-kort 4005 med 16 Utgångar och 8 Ingångar

Det digitala I/O-kortet 4005 har 16 Open collector utgångar. På utgångarna kan man maximalt ansluta 24 V, samt maximalt driva 125 mA. Utgångarna styrs i grupper om 8. Se fig. 3.5.

Ingångarna är av normalt TTL-snitt, vilket innebär att motsvarande bit nollställes om en spänning 0-1 V finns på ingången, eller ettställes om en spänning 4-5 V finns på ingången. Alla bitar är ettställda efter reset (RST) signalen. Alla ingångar avläses samtidigt.

4005 kan användas till att styra mindre reläer, lampor, lysdioder samt liknande utrustning. 4005 kan även läsa av tumhjul, switchar och liknande. Man kan också ansluta TTL-kompatibel elektronik. Om man t.ex. låter utgångarna styra tumhjul så kan man avläsa upp till 16x2 = 32 st tumhjul.

4005 har open-collector utgångar. En vanlig TTL-krets har utgångar av totempåletyp, figur 3.1(a), där utsignalen tas ut mellan två seriekopplade transistorer. Lägger TTL-kretsen ut en logisk etta på utgången leder den övre transistorn och den undre spärrar, medan det motsatta förhållandet råder när kretsen lägger ut en logisk nolla.

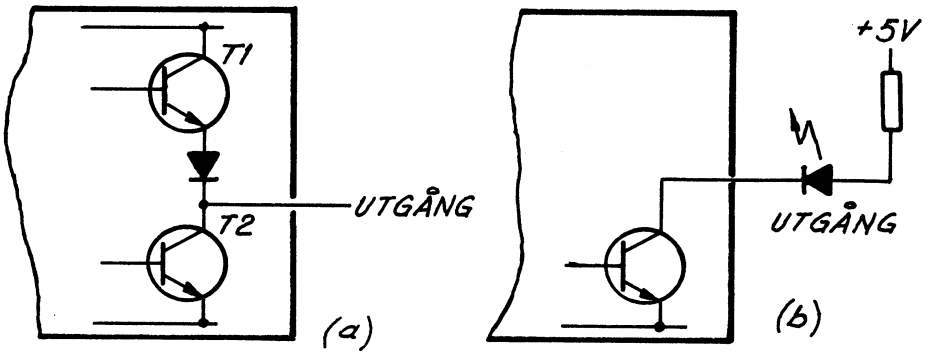


Fig. 3.1 (a) TTL Totempåleutgång, (b) TTL Open-collector utgång.

En TTL-krets med open-collector utgång saknar den övre av dessa transistorer, figur 3.1(b). En sådan utgång kan man således använda som en "elektronisk strömbrytare", genom att vi lägger ut logiska nollor och ettor på utgången.

När vi kopplar in kortet ligger nollor på alla utgångar vilket medför att transistorerna (="strömbrytarna") är ledande. Ansluter vi utgången till en lysdiod i serie med ett motstånd upp till +5 V kommer således lysdioden att lysa. För att öppna en strömbrytare, dvs göra utgångstransistorn spärrande skall man lägga ut en etta på den utgång man vill bryta.

Utgångarna sitter i grupper om åtta och styrs av binärtal. Detta innebär att man styr genom att skicka ut heltal som ligger mellan 0 och 255 med OUT-kommandot. För att veta vad man ettställer och nollställer måste man således göra om det decimala heltalet till ett binärt tal.

Exempel:

Ett kommando "OUT 0,57" innebär att ett binärt tal, som motsvarar det decimala talet 57 läggs ut på grupp 1, utgång 0-7.

57 = 00111001

Vi ser att utgång 0, 3, 4 och 5 blir ettställda medan övriga blir nollställda.

De åtta ingångarna avläses samtidigt och bildar ett heltal mellan 0 och 255. De inledningar som har en spänning 0-1 V ger en logisk nolla medan de som har en spänning 4-5 V ger en logisk etta. De ledningar som ej är anslutna ger logiska ettor.

Digitalt I/O-kort 4006 med 32 Utgångar och 16 Ingångar

Ett blockschema för 4006 kan återfinnas i figur 3.8. De 32 utgångarna (uppdelade i fyra grupper om 8 bitar i vardera), på 4006 skiljer sig från 4005 genom att de är av tri-state typ. Det innebär att utgångarna kan anta tre lägen : ettställd, nollställd eller frikopplad, "flytande". Man kan frikoppla en eller alla grupper genom att lägga + 5 V på tri-state kontrollen för gruppen man vill ha frikopplad. Man kan inte styra tri-state kontrollen från ABC80/800/DTC genom kontrollsignaler utan man måste göra en hårdvarukoppling för att kunna styra den från datorn. Detta är mycket användbart när man ska ha flera grupper som styr samma signal.

De 16 ingångarna fungerar som på 4005, men om man vill använda de åtta mest signifikanta insignalerna väljer man det vid kortval (CS) genom att sända ut kortets adress plus 128, dvs OUT 1,128+<kortadress>.

4006 kan användas för styrning av yttre digitala enheter och liknande samt avläsning av omkopplare, tumhjul, etc. In- och utgångarna är TTL-kompatibla.

Program skrivna för max 16 utgångar och max 8 ingångar kan användas på både 4005 och 4006.

Till skillnad från 4005 har 4006 32 st tri-state TTL utgångar. Detta medför att man med OUT-kommandona kan sätta utgångarna till + 5V (logisk etta) eller 0 V (logisk nolla). Tri-state läget innebär att varje utgång har tre lägen hög, låg och frikopplad. Utgångarna är kopplade i fyra grupper om åtta bitar i varje grupp och varje grupp styrs av en tri-state signal. Om man inte vill använda tri-state funktionen kopplas tri-state styrsignalen (TS) till jord. Tri-state funktionen används bl.a. när två eller fler grupper ligger parallellt anslutna till samma buss. Tri-state styrsignalerna finns i I/O-kontakten och genom att koppla dessa till en utport kan vi styra grupperna programvarumässigt. Efter en reset (RST) signal ligger alla utgångar låga (0 V).

Ingångarna finns i två grupper om åtta stycken i varje. Man kan bara använda en grupp i taget, sedan måste en omställning av kortvalet ske för att kunna använda den andra. Om man väljer kort på vanligt vis, t.ex: OUT 1,3 erhålls insignaler från grupp 1. Om man däremot väljer kort och adderar 128 till kortadressen, t.ex: OUT 1,3+128 får man insignaler från grupp 2.

I/O-Kommandon till kort 4005.

BASIC	SIGNAL	FUNKTION
INP (7)	RST	Nollställer alla I/O-kort. En programrad av typ: 10 Slask=INP (7) bör alltid inleda ett program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kortet med adressen <Kortval>. Exempel: 20 OUT 1,2 som väljer ut kortet med adressen 2 och tänder lysdioden på kortet.
INP (0)	INP	Läser de 8 ingångarna från kortet och tolkar inläst data som ett heltal. Exempel: 30 A=INP (0) tilldelar variabeln A de 8 bitar som lästs från inporten.
OUT 0,<Data>	OUT	<Data> skickas ut till grupp 1, utgång 0-7. Exempel: 40 OUT 0,32 ettställer utgång 5 och nollställer 0-4, 6 och 7, (2ü5=32)
OUT 2,<Data>	C1	<Data> skickas ut till grupp 2, utgång 8-15. Exempel: 50 OUT 2,255 ettställer utgångarna 8 - 15.

I/O-Kommandon till kort 4006.

BASIC	SIGNAL	FUNKTION
INP (7)	RST	Nollställer alla I/O-kort. En programrad av typ: 10 Slask=INP (7) bör alltid inleda ett program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kortet med adressen <kortval>. Exempel: 20 OUT 1,3 väljer ut kortet med adress 3 och grupp 1 (insignaler 0-7).
OUT 1,<Kortval+128>	CS	Väljer kortet med adressen <Kortval+128> plus att man väljer grupp 2 för läsning, insignaler 8-15. Exempel: 20 OUT 1,3+128 väljer ut kortet med adressen 3 och grupp 2 (insignaler 8-15).
INP (0)	INP	Läser värdet på den valda läsgruppen 1 eller 2, enligt tidigare kortval. Exempel: 30 Indata=INP (0) läser in data från vald grupp till variabeln "Indata".

OUT 0,<Data>	OUT	Variabeln <Data> läggs ut på grupp 1, signal 0 - 7. Exempel: 40 OUT 0,16 ettställer utgång 4, och nollställer 0 - 3 och 5 - 7, (2ü4=16).
OUT 2,<Data>	C1	Variabeln <Data> läggs ut på grupp 2, signal 8 - 15. Exempel: 50 OUT 0,255 ettställer utgångarna 8 - 15.
OUT 3,<Data>	C2	Variabeln <Data> läggs ut på grupp 3, utgångarna 16 - 24.
OUT 4,<Data>	C3	Variabeln <Data> läggs ut på grupp 4, utgångarna 25 - 31.

INSTALLATION AV 4005 OCH 4006.

Val av adress.

För att bestämma adress för kortet se kapitel 1: "Adressering av kort". I exempel nedan används adress 2 för kort 4005 och adress 3 för kort 4006.

Anslutning av yttre enheter.

Anslutning av de yttre enheterna på korten sker på I/O-kontakten. I/O-kontakten är den som sitter närmast lysdioden.

Montering av 4005 och 4006 i expansionsenheten.

Slå av spänningen först! Korten vänds så att I/O-kontakten kommer utåt och komponentsidan åt höger. Både 4005 och 4006 skall placeras i I/O-delen på expansionsenheten.

Kontroll av adressen.

Efter montering av ett kort i expansionsenheten bör man kontrollera att kortet har byglats till rätt adress, genom att skriva:

```
OUT 1,<Kortadress>
```

<kortadress> avser den adress som kortet har getts. För att testa kort 4005 eller 4006 med t.ex adressen 2, ge kommandot:

```
OUT 1,2
```

Om adressen är riktig skall lysdioden nere till höger på kortet tändas. Om man adresserar något annat kort skall I/O-kortets lysdiod släckas.

Styrning av enstaka utgångar i en grupp.

Ofta vill man bara styra en utgång i taget medan man vill ha de andra utgångarna opåverkade efter operationen. Detta kan åstadkommas genom maskning. Eftersom maskning för ett- respektive noll-ställning återkommer ganska ofta i program, som använder sig av korten 4005 eller 4006 för digital signalering, är det lämpligt att definiera funktioner för detta. Funktionerna nedan stämmer för båda korten om uppkoppligen är av samma typ som i figurerna 3.2 och 3.3 nedan.

Funktion för nollställning:

```
10 DEF FNNoll (Sistut,port)=Sistut AND (65535-2üport)
```

Funktion för ettställning:

```
20 DEF FNEtt (Sistut,port)=Sistut OR 2üport
```

Funktionerna anropas med två variabler som parametrar. Variabeln "Sistut" anger det tal som sist lades ut på den aktuella gruppen. Första gången ett OUT-kommando exekveras är "Sistut" = 0, andra gången skall "Sistut" vara samma värde som det som sist lades ut på den gruppen. Variabeln "Port" anger vilken utgång i gruppen man vill ändra.

Programexemplet nedan illustrerar, med hjälp av de funktioner vi definierat ovan, hur man kan styra utsignaler med I/O-korten 4005 och 4006. Fig. 3.2 och 3.3 visar uppkoppling av 4005 respektive 4006 för test av detta program. Vi använder oss av lysdioder för att visa tillståndet hos utgångarna, (strömbrytarna är inte nödvändiga för test av detta program).

Exempel 1: Styrning av utgångarna hos I/O-kort 4005 och 4006.

Programmet frågar efter vilken grupp det gäller och sedan vilken utgång som skall ändras. Slutligen frågar programmet om utgången skall ettställas eller nollställas varefter den utpekade utgången antar det önskade värdet utan att de övriga utgångarna påverkades.

```
10  ! STYRNING AV UTSIGNALERNA PÅ KORT 4005 OCH 4006
20  ! STYRNINGEN SKER INDIVIDUELLT
30  ! EN UTGÅNG PÅVERKAS PER GÅNG
40  EXTEND : INTEGER
50  !
60  ! FUNKTIONERNA
70  !
80  DEF FNNoll(Sistut,Port)=Sistut AND (65535-2üPort)
90  DEF FNEtt(Sistut,Port)=Sistut OR 2üPort
100 !
110 Slask=INP(7) ! NOLLSTÄLLNING AV I/O-KORTEN
120 OUT 1,2 ! KORTVAL 4005: 3 KORTVAL 4006: 2
130 ; CHR$(12)
140 INPUT 'VILKEN GRUPP ?'Grupp
150 IF Grupp<1 OR Grupp>4 GOTO 140
160 IF Grupp=1 Grupp=0
170 INPUT 'VILKEN UTGÅNG I GRUPPEN 0-7 ?'Port
180 IF Port<0 OR Port>7 GOTO 170
190 INPUT 'ETTSTÄLLNING ELLER NOLLSTÄLLNING AV PORTEN ?'Läge
200 IF Läge<0 OR Läge>1 GOTO 190
210 IF Läge=0 THEN Sistut=FNNoll(Sistut,Port)
220 IF Läge=1 THEN Sistut=FNEtt(Sistut,Port)
230 Ut=Sistut
240 OUT Grupp,Ut
250 GOTO 130
260 END
```

Kommentarer till ovanstående program:

40 : Långa variabelnamn och heltal väljs.
80 : Definition av nollställning.
90 : Definition av ettställning.
110 : Nollställning av I/O-korten.
120 : Kortval.
160 : "Grupp" anpassas till OUT-kommandona.
210 : Test om nollställning och "Sistut" får sitt värde.
220 : Test om ettställning och "Sistut"får sitt värde.
240 : Omställningen mellan ett och noll. "Ut" = utgången som skall ändras. "Grupp" = den grupp som skall få utgången ändrad.

Läsning av ingångarna hos I/O-kort 4005 och 4006.

Om man vill veta tillståndet hos en ingång i en grupp kan vi också här definiera en funktion, på samma sätt som vi gjorde för att styra utgångarna.

```
30 DEF FNEnin (Port)=(INP(0) AND 2üPort)/2üPort
```

Variabeln "Port" anger vilken av ingångarna 0 - 7 som skall läsas. Funktionen ger svaret 1 (sant) eller 0 (falsk). Programexempel 2 visar hur funktionen fungerar.

Programexempel 2 illustrerar hur man kan läsa insignaler med I/O-korten 4005 och 4006. Fig. 3.2 och 3.3 visar uppkoppling av 4005 respektive 4006 för test av detta program. Vi använder oss av strömbrytare för att simulera insignaler till kortet, (lysdioderna är inte nödvändiga för detta program).

Exempel 2: Läsning av insignaler med I/O-kort 4005 och 4006.

Programmet börjar med att fråga vilken ingång som skall läsas, och läser därefter alla åtta ingångarna (i grupp 1 eler grupp 2 på 4006 beroende på CS) och svarar om en viss bit är satt eller ej.

```
10  ! LÄSNING AV EN INGÅNG PÅ KORTEN 4005 OCH 4006
20  !
30  EXTEND : INTEGER
40  !
50  ! FUNKTIONEN
60  !
70  DEF FNEnin(Port)=(INP(0) AND 2üPort)/2üPort
80  !
90  Slask=INP(7) ! NOLLSTÄLLNING AV I/O-KORTEN
100 OUT 1,2 ! TÄNK PÅ ATT ADDERA 128 OM GRUPP 2 PÅ 4006
    SKALL ANVÄNDAS
110 !
120 ; CHR$(12)
130 INPUT 'VILKEN INGÅNG SKALL LÄSAS 0-7 ?'Port
140 IF Port<0 OR Port>7 GOTO 130
150 ; : ; : ; 'INGÅNGEN ÄR ';
160 IF FNEnin(Port)=0 THEN ; 'NOLLSTÄLLD' : ; : ;
170 IF FNEnin(Port)=1 THEN ; 'ETTSTÄLLD' : ; : ;
180 GOTO 130
190 END
```

Kommentarer till ovanstående program:

30 : Val av långa variabler och heltal.
70 : Definition av funktionen som kollar en viss bit.
90 : Nollställning av alla I/O-kort.
100 : Kortval, grupp 1. Om grupp 2 skall läsas måste 128 adderas till kortadressen.
160 : Kontrollerar om porten är nollställd.
170 : Kontrollerar om porten är ettställd.

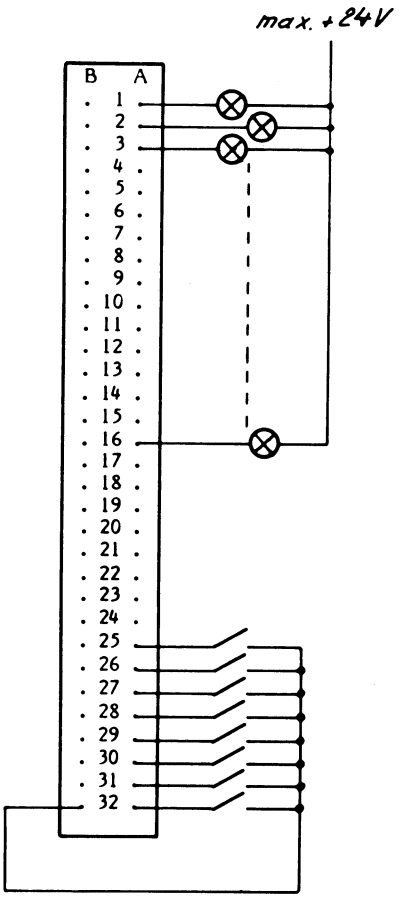


Fig. 3.2 Experimentuppkoppling för digitalt I/O-kort 4005.

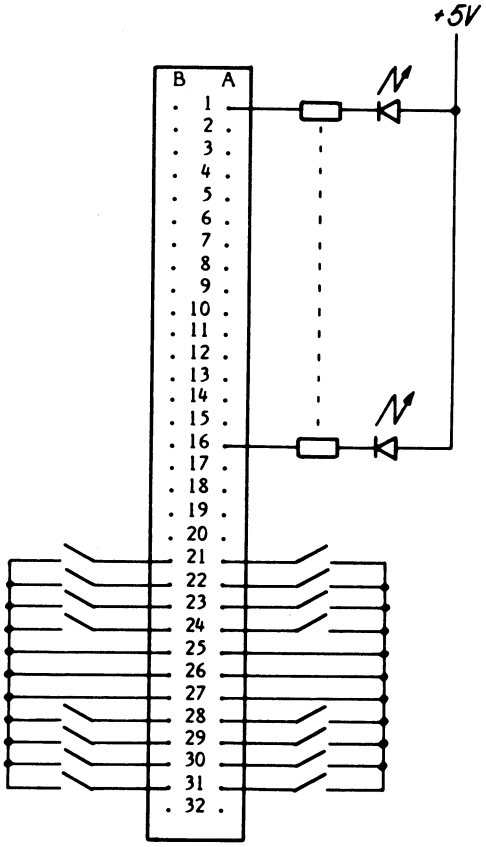


Fig. 3.3 Experimentuppkoppling för digitalt I/O-kort 4006.

Exempel 3: Avläsning av tumhjul och styrning av analog spänning.

Detta program läser av 3 stycken tumhjul med hjälp av det digitala I/O-kortet 4006 och styr sedan D/A-kortet 4083 så att vi får en analog utsignal som motsvarar det talvärde som ställts in på tumhjulen. Programmet läser av tumhjulen och tolkar det inlästa värdet som ett tal med en heltalssiffra och två decimaler. Om vi t.ex. har ställt in 534 på tumhjulet så tolkas detta av programmet som talet 5.34. Detta talvärde styr sedan kanal 1 på D/A-kortet 4083, som då ställer in sig på en analog utsignal av 5.34 VDC.

Uppkoppling av exemplet skall utföras som visas i fig 3.4. På D/A-kortet 4083 skall en voltmeter anslutas till stift 2B och 12B på I/O-kontakten och stift 7B och 14B förbindas, (se även fig. 4.2). Programmet går i en evighetsslinga efter uppstartandet varför man när som helst kan ändra spänningen på D/A-kortet med tumhjulen.

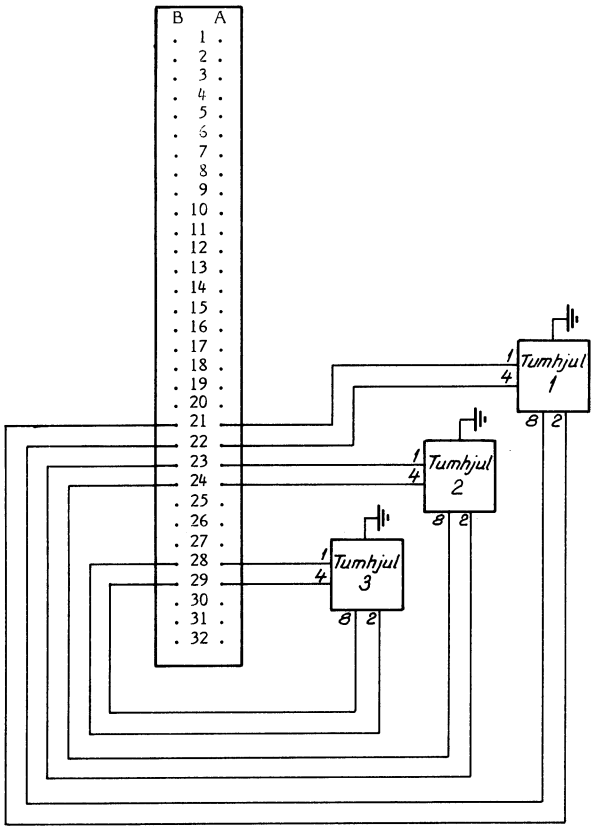


Fig. 3.4 Uppkoppling för styrning av analog utspänning med tumhjul.

```
10  ! ANALOG SPÄNNINGS-STYRNING
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  !
60  ! RUTIN FÖR AVLÄSNING AV TUMHJUL
70  !
80  ; CHR$(12)
90  ; TAB(20) " ANALOG SPÄNNINGSSTYRNING "
100 ; CUR(6,0) "Ställ in önskat värde på tumhjulet och tryck
          RETURN ";
110 GET Slask$
120 IF Slask$<>CHR$(13) GOTO 100
```

```
130 !
140 ! AVLÄSNING AV TUMHJUL
150 !
160 OUT 1,3 ! KORTVAL SAMT VAL AV GRUPP 1
170 Grupp1=INP(0) ! INLÄSNING AV FÖRSTA GRUPPEN (0-7)
180 !
190 OUT 1,3+128 ! KORTVAL SAMT VAL AV GRUPP 2
200 Grupp2=INP(0) ! INLÄSNING AV ANDRA GRUPPEN (8-15)
210 !
220 Första$=NUM$(Grupp2 XOR 255)
230 Andra$=NUM$((Grupp1 XOR 255)/16)
240 Tredje$=NUM$((Grupp1 XOR 255) AND 15)
250 Värde.=VAL(Första$+"."+Andra$+Tredje$)
260 !
270 ! RUTIN FÖR UTSPÄNNINGEN
280 !
290 OUT 1,11 ! VAL AV D/A-KORTET 4083
300 OUT 0,Värde.*4095/10,2,SWAP%(Värde.*4095/10) ! VÄRDET
        TILL D/A-KORTET
310 !
320 GOTO 140 ! TILLBAKA FÖR ATT LÄSA ETT NYTT VÄRDE
```

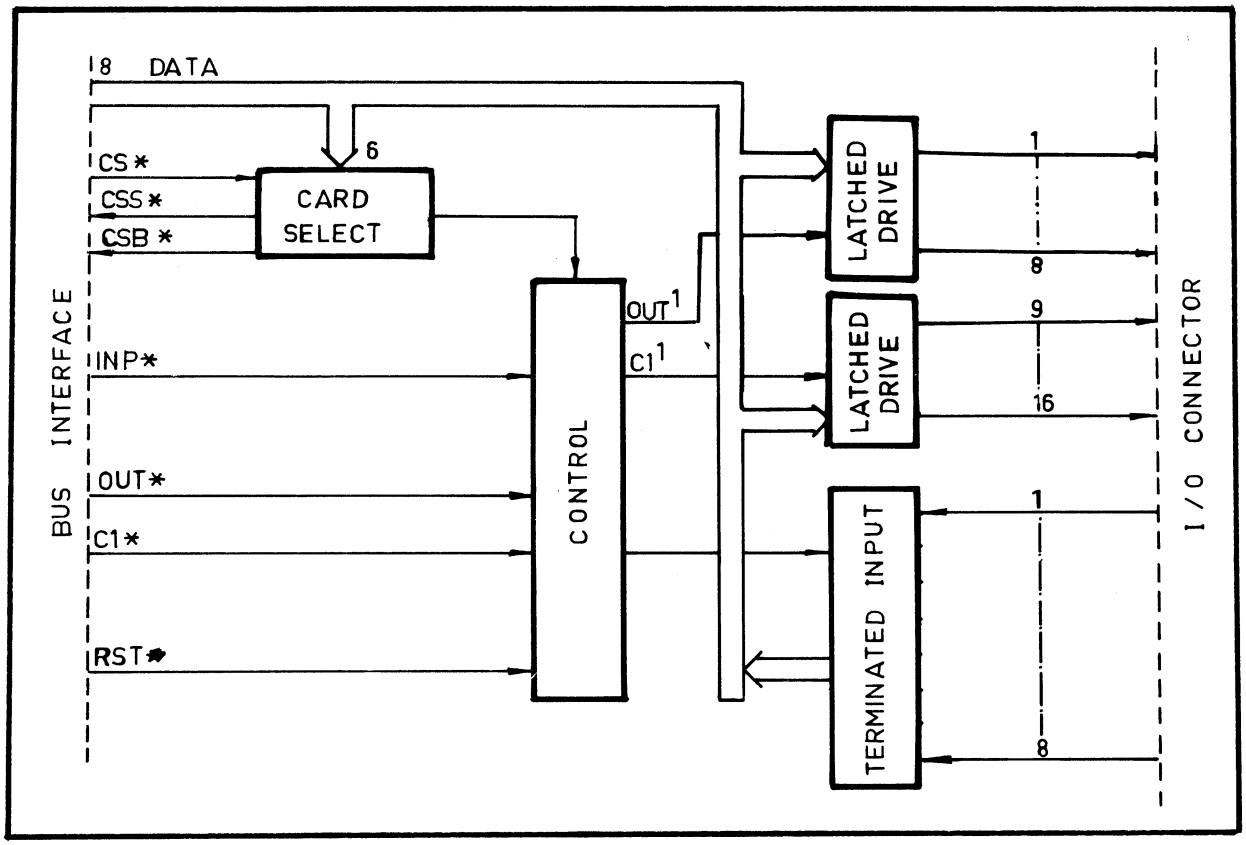


Fig. 3.6 Blockschema för digitalt I/O-kort 4005.

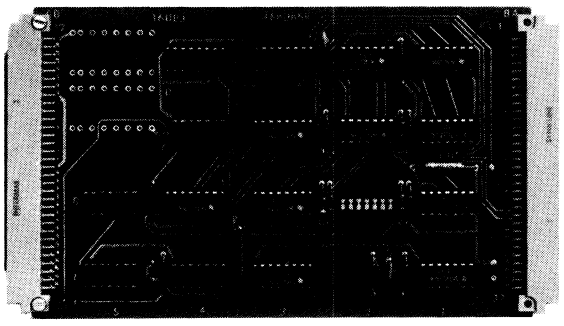


Fig. 3.5 Digitalt I/O-kort 4005.

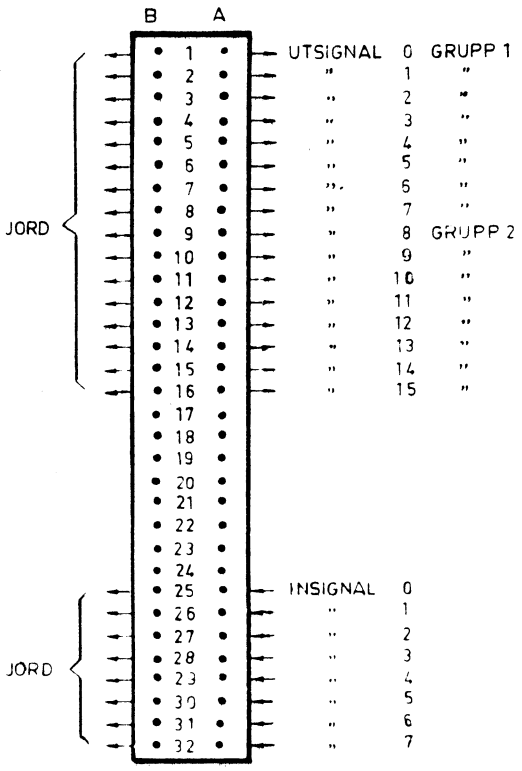


Fig. 3.7 Stiftlayout för I/O-kontakt hos 4005.

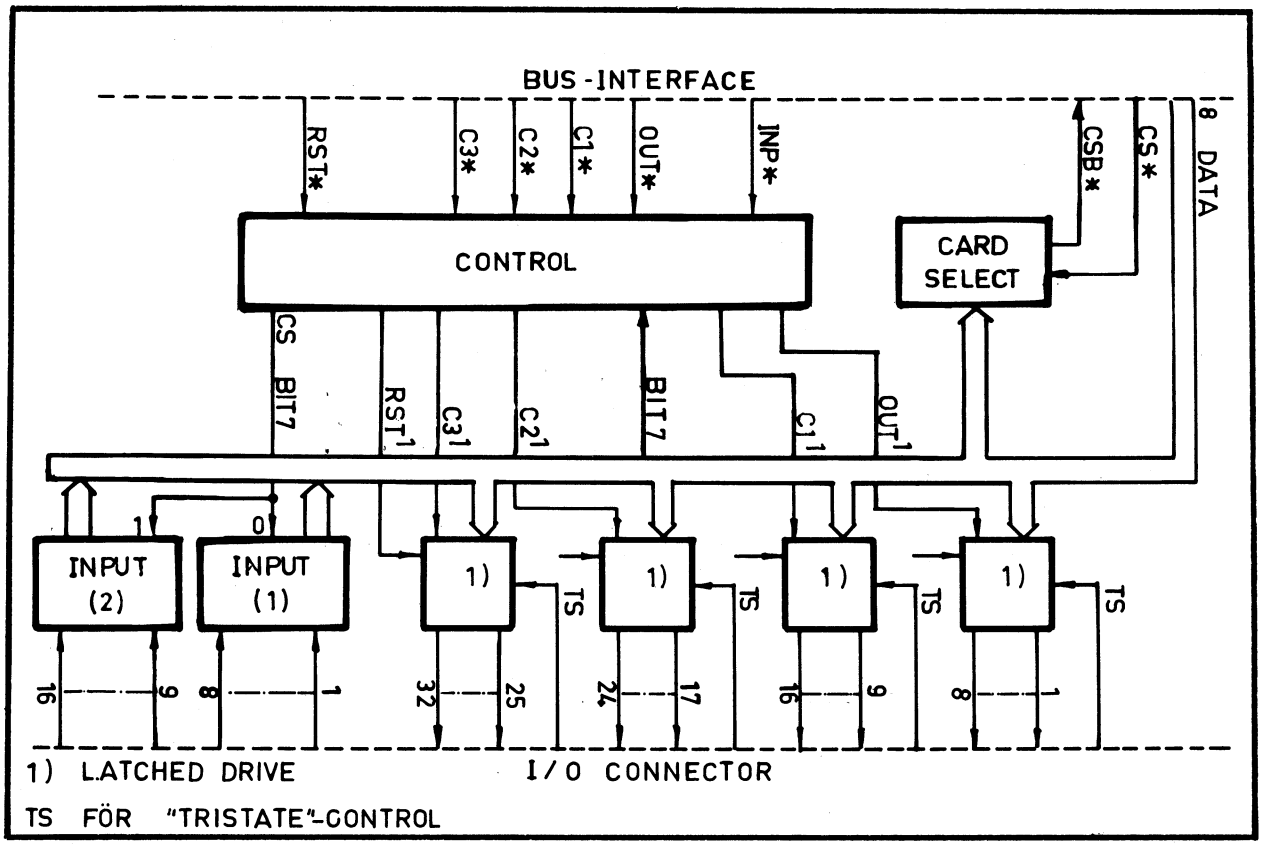


Fig. 3.9 Blockschema för digitalt I/O-kort 4006.

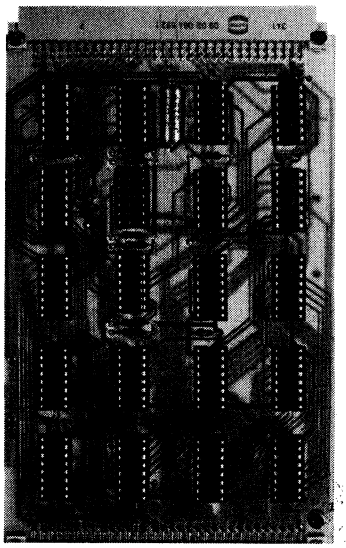


Fig. 3.8 Digitalt I/O-kort 4006.

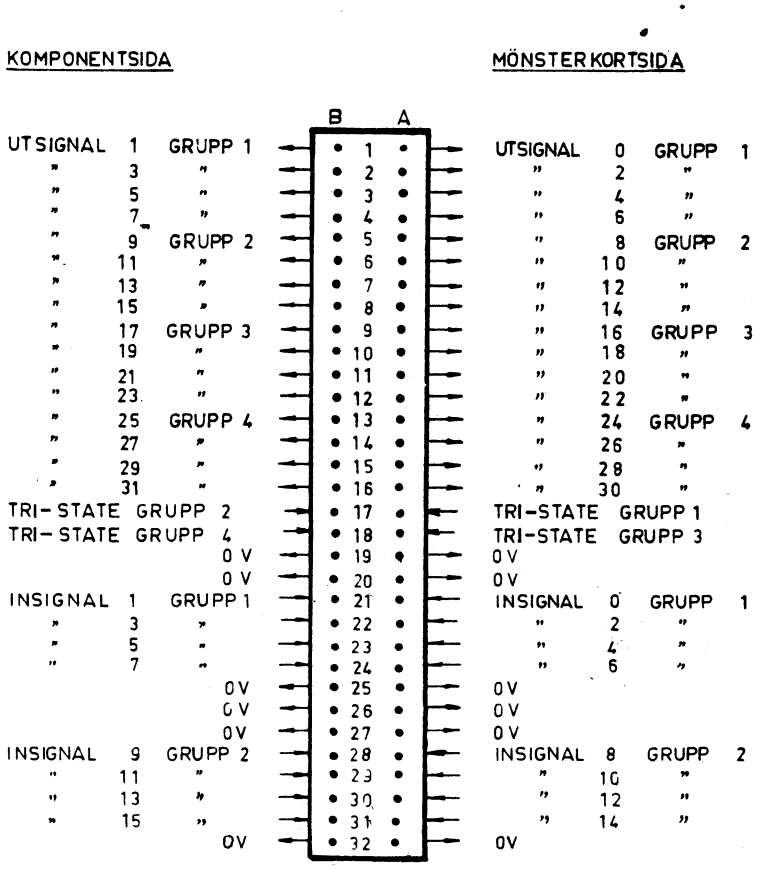


Fig. 3.10 Stiftlayout för I/O-kontakt hos 4006.

3.2 Digital signalering med 4008/4011 och 4095 OPTO-kort

För digital signalering med galvanisk separation finns I/O-korten 4008/4011, fig. 3.16, och 4095, fig. 3.19, med följande specifikationer:

4008/4011

16 Opto-ingångar
Insignal: Logisk "0": 10-25 mA eller 5-12 V, (4008)
Logisk "0": 10-50 mA eller 5-24 V, (4011)
Logisk "1": 0 mA eller 0 V, (4008/4011)

4095

16 Opto-utgångar
Utgångar: Max spänning 24 V, ström 0,8 A med termisk överströmsskydd.
Restspänning vid tillslag: 2 V

Digitalt 16 ingångars kort 4008/4011 med galvanisk separation.

4008/4011 har 16 stycken optokopplare på ingångarna, fig. 3.11. De är av typen CQY 80, (eller motsvarande) med en isolationsspänning på 4 kV. Med hjälp av dessa blir datorn galvaniskt skild från insignalerna. Detta inebär att man slipper ifrån problemet med gemensam jordning vid inkoppling av instrument. Risken för att spänningstransienter, induktionsspänning etc., orsakade av elektriska motorer, åsknedslag eller dylikt skall fortplanta sig in i datorn och störa eller skada den känsliga elektroniken är minimal. Skillnaden mellan korten 4008 och 4011 är att den logiska nollan på 4008 ligger på en spänning mellan 5 och 12 V medan den på 4011 ligger mellan 5 och 24 V. Den logiska ettan ligger i båda fallen på 0 V.

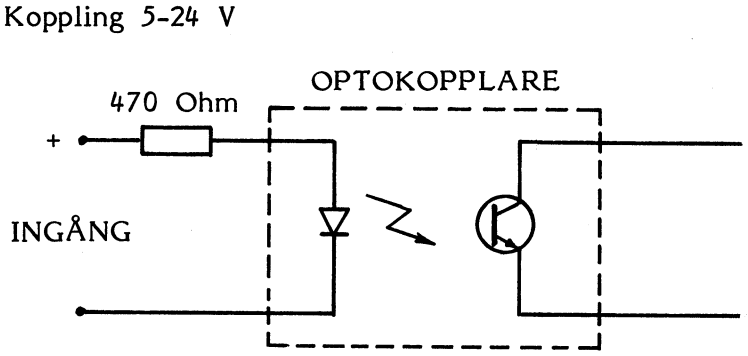


Fig. 3.11 Optokopplare på en ingång på kort 4008/4011.

Digitalt 16 utgångars kort 4095 med galvanisk separation.

Med 4095 kan man styra 16 optoisolerade **utgångar,** fig. 3.12. Man slipper således, som i fallet med 4008/4011 en gemensam jordning mellan datorn och uppkopplingarna utanför. Optokopplarna på 4095 klarar av att driva maximalt 24 V, 0.8 A och är termiskt skyddade mot överström. 4095 är programkompatibelt med utgångarna på det digitala I/O-kortet 4005. Med 4103 (16 Reläutgångar) är kortet både program- och kontaktkompatibelt.

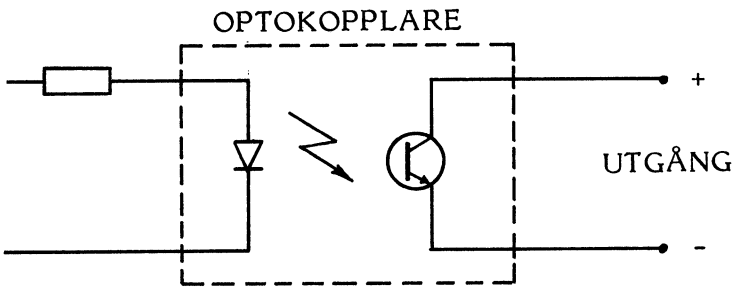


Fig. 3.12 Optokopplare på en utgång på kort 4095.

I/O-kommandon för 4008/4011.

Basic	Aktiverad signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP (7) bör inleda varje program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kortet med adressen <Kortval> och på 4008/4011 grupp 1 för läsning. Exempel: 20 OUT 1,4 väljer kort med adressen 4 och grupp 1 för läsning.
OUT 1,<Kortval+128>	CS	Väljer kortet med adressen <Kortval> och på 4008/4011 grupp 2 för läsning. Exempel: 20 OUT 1,4+128 väljer kort med adressen 4 och grupp 2 för läsning.
INP (0)	INP	Läser åtta insignalerna från grupp enligt tidigare val med OUT 1,<Kortval>. Exempel: 30 Data=INP (0) medför att variabeln "Data" tilldelas värdet av de åtta lästa insignalerna.

I/O-kommandon för 4095.

INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP (7) bör inleda varje program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kortet med adressen <Kortval>. Exempel: 20 OUT 1,7 väljer kort med adressen 7.
OUT 0,<Data>	OUT	De åtta minst signifikanta bitarna i heltalsvariabeln "Data" läggs ut på utgångarna 0-7. Exempel: Kommando 30 OUT 0,8 gör utgång 3 (2ü3=8) ledande, medan utgång 0-2 och 4-7 bryts.
OUT 2,<Data>	C1	De åtta minst signifikanta bitarna i heltalsvariabeln "Data" läggs ut på utgångarna 8-15. Exempel: 40 OUT 2,255 gör utgångarna 8-15 ledande (kortslutna).
OUT 4,0	C3	Bryter alla utgångar 0-15.

Installation av 4008/4011.

Val av adress.

Se kapitel 1: "Adressering av I/O- och minneskort" för val av adress hos kort som skall monteras i expansionsenheten. I nedanstående exempel används adress 4 för 4008 och adress 9 för 4095.

Anslutning av yttre enheter.

Yttre enheter ansluts till I/O-kontakten (kontakten närmast lysdioden).

Insättning av 4008/4011 och 4095.

Slå av spänningen först! Korten placeras i I/O-delen på expansionsdelen. Korten vänds så att I/O-kontakten kommer utåt och komponentsidan åt höger.

Kontroll av adressen.

Efter montering av ett kort i expansionsenheten bör man kontrollera att kortet har byglats till rätt adress, genom att skriva:

```
OUT 1,<Kortadress>
```

<Kortadress> är adressen för aktuellt I/O-kort. Lysdioden nere till höger skall tändas om byglingen är korrekt utförd. Om därefter annan adress väljes släcks lysdioden.

Läsning av enstaka ingångar i en grupp

Eftersom man vid inläsning läser åtta ingångar i taget och man oftast bara är intresserad av en ingång i taget måste man maska för att få reda på svaret. En funktion som utför detta är:

```
10 DEF FNMask(Port)=(INP (0) AND 2ü(Port-1))/2ü(Port-1)
```

Man anropar funktionen med en variabel som innehåller portnumret (1-8). Variabeln pekar på den insignal som skall läsas. Vilken grupp som skall läsas har bestämts tidigare med ett kortvals-kommando. Den här funktionen ger svaret 1 eller 0.

Exempel 1: Rinnande ljus med 4095

Programmet tänder två lampor åt gången i sådan följd att det ser ut som om ljuset 'rinner'. Se fig. 3.13 för uppkopplingen.

```
10  ! RINNANDE LJUS MED HJÄLP AV 4095
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,7
60  !
70  ! EVIGHETS LOOP
80  FOR Loop=0 TO 7
90    IF Loop=0 THEN OUT 0,2ü0+2ü7
100     IF Loop>0 THEN OUT 0,2üLoop+2ü(Loop-1)
110 FOR Vänta=0 TO 200 : NEXT Vänta
120 NEXT Loop
130 GOTO 80
140 END
```

Kommentarer till ovanstående program: (siffror anger programrad)

30 : Sätter långa variabelnamn och heltalsbasic.
40 : Nollställer alla I/O-kort.
50 : Väljer kortet med adressen 7, (=4095)
90 : Tänder lampa 0 och 7 om "Loop" = 0.
100 : Tänder två lampor när "Loop" > 0.

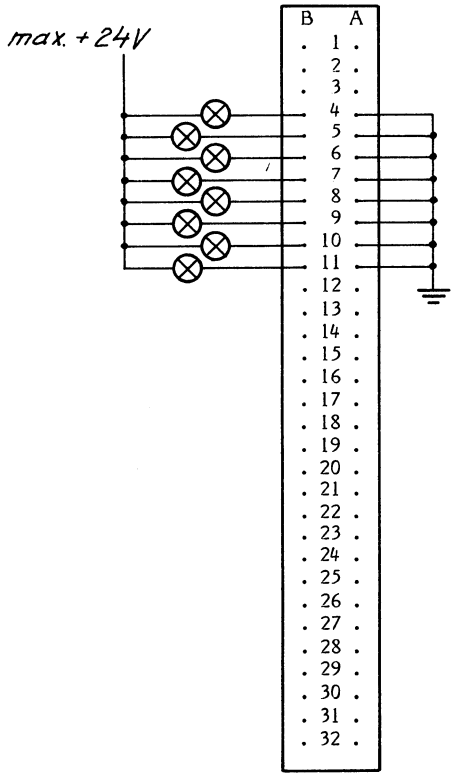


Fig. 3.13 Experimentuppkoppling för rinnande ljus.

Exempel 2: Inläsning av data från optoingångarna på 4008/4011

Programmet läser ingångarna och presenterar därefter tillståndet hos de 16 ingångarna i binär form på bildskärmen. Se fig. 3.14 för uppkopplingen.

```
10  ! LÄSNING AV EN INGÅNG
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  !
60  ! DEFINITION SOM MASKAR UT EN BIT
70  !
80  DEF FNMask(Port)=(INP(0) AND 2ü(Port-1))/2ü(Port-1)
90  ; CHR$(12)
100 FOR Loop=1 TO 8
110 OUT 1,4
120 Binär1$=Binär1$+NUM$(FNMask(Loop))
130 OUT 1,4+128
140 Binär2$=Binär2$+NUM$(FNMask(Loop))
150 NEXT Loop
160 ; : ; 'GRUPP 1 : ' ; : ; Binär1$
170 ; : ; 'GRUPP 2 : ' ; : ; Binär2$
```

```
180 GET Slask$ : GOTO 100
190 END
```

Kommentarer till ovanstående program:

30 : Val av långa variabelnamn och heltal.
40 : Nollställer alla I/O-kort.
80 : Funktion som maskar ut en bit.
110 : Kortval, adresserar kort med adressen 4 och grupp 1.
120 : Gör om decimaltalet till ett binärt.
130 : Kortval, väljer grupp 2.
140 : Gör om decimaltalet till ett binärt.

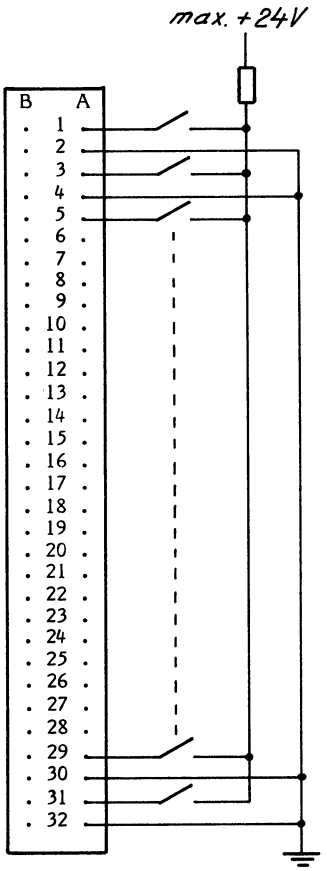


Fig. 3.14 Experimentuppkoppling för läsning av optoingångar.

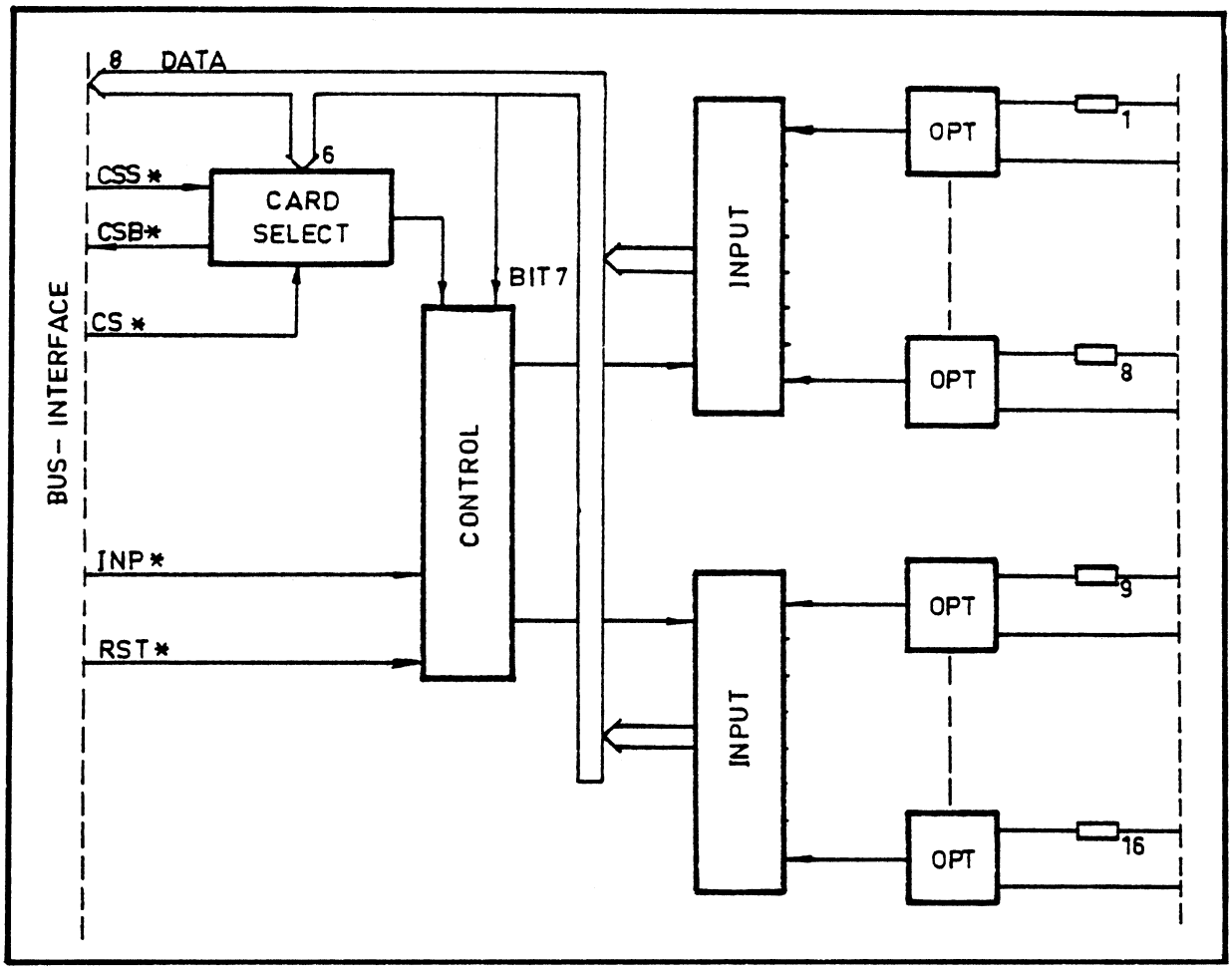


Fig. 3.16 Blockschema för Optokort 4008/4011

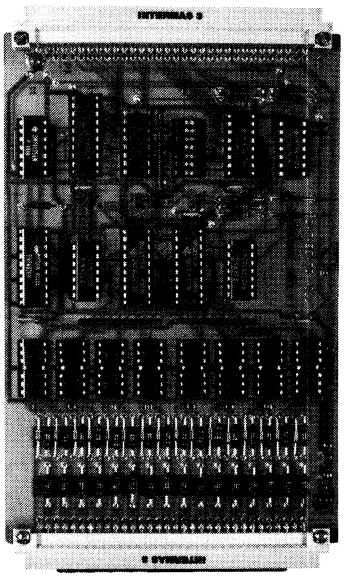


Fig. 3.15 Optokort 4008/4011

KOMPONENTSIDA MÖNSTERKORTSIDA

B	A					
1		INGÅNG	1	ANSLUTES	TILL	+
2		ÅTER	1	"	"	−
3		INGÅNG	2	"	"	+
4		ÅTER	2	"	"	−
5		INGÅNG	3	"	"	+
6		ÅTER	3	"	"	−
7		INGÅNG	4	"	"	+
8		ÅTER	4	"	"	−
9		INGÅNG	5	"	"	+
10		ÅTER	5	"	"	−
11		INGÅNG	6	"	"	+
12		ÅTER	6	"	"	−
13		INGÅNG	7	"	"	+
14		ÅTER	7	"	"	−
15		INGÅNG	8	"	"	+
16		ÅTER	8	"	"	−
17		INGÅNG	9	"	"	+
18		ÅTER	9	"	"	−
19		INGÅNG	10	"	"	+
20		ÅTER	10	"	"	−
21		INGÅNG	11	"	"	+
22		ÅTER	11	"	"	−
23		INGÅNG	12	"	"	+
24		ÅTER	12	"	"	−
25		INGÅNG	13	"	"	+
26		ÅTER	13	"	"	−
27		INGÅNG	14	"	"	+
28		ÅTER	14	"	"	−
29		INGÅNG	15	"	"	+
30		ÅTER	15	"	"	−
31		INGÅNG	16	"	"	+
32		ÅTER	16	"	"	−

Fig. 3.17 Stiftlayout för I/O-kontakt hos 4008/4011

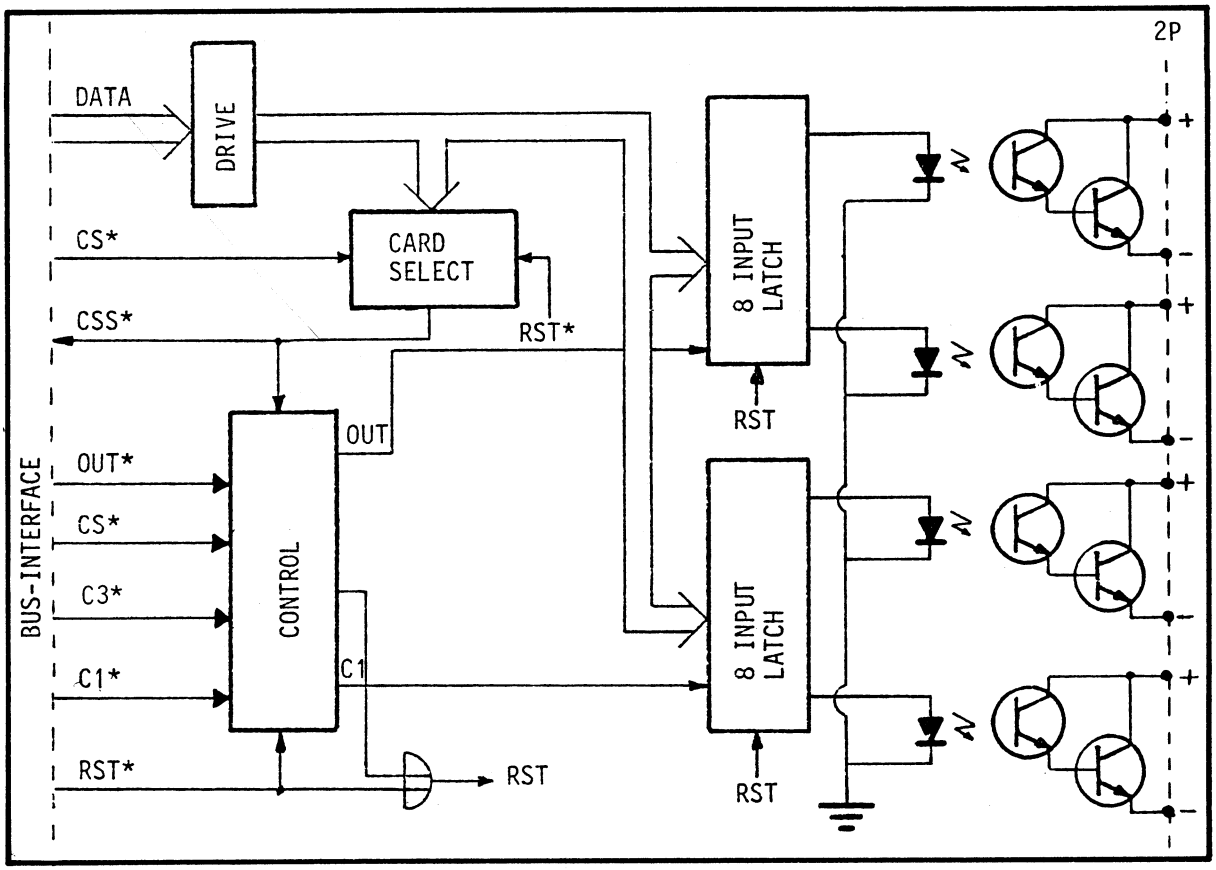


Fig. 3.19 Blockschema för Optokort 4095

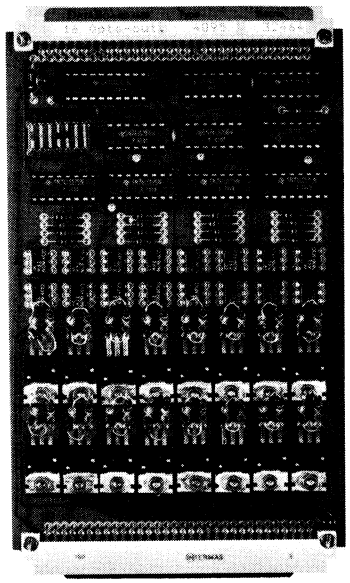


Fig. 3.18 Optokort 4095

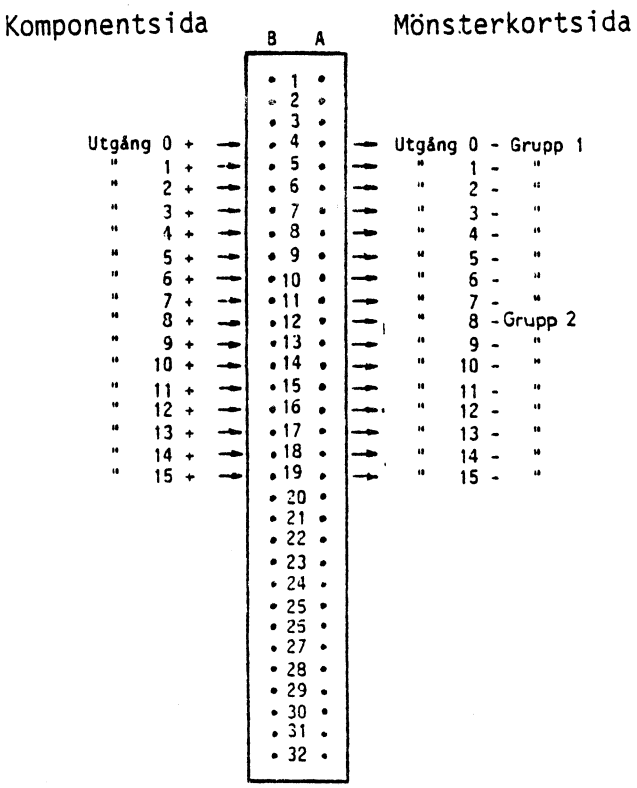


Fig. 3.20 Stiftlayout för I/O-kontakt hos 4095

3.3 Digital signalering med I/O-kort 4085

4085

32+4 TTL-ingångar
2 TTL-utgångar

Digitalt I/O-kort 4085 med 32+4 Ingångar och 2 utgångar.

4085 har 32 ingångar av normal TTL-typ. Ingångarna är ordnade i fyra 8-bitars grupper. Varje grupp har en styringång (STB) för val av lagring i vippor eller direkt avläsning av insignalerna. Det finns också fyra interrupt (INT) ingångar som kan användas för att generera avbrott till datorn. För att detta skall fungera måste dess kortplats byglas på bakplanet. Se kapitel 1 som visar hur bygling skall utföras. Om inte interrupt-ingångarna skall användas till att generera avbrott i datorn behöver inte bygeln flyttas och ingångarna kan användas som vanliga TTL-ingångar. Avbrottsrutinen i datorn måste vara skriven i assembler. En avbrottsrutin skriven i assembler kan du hitta i exempel 4 nedan.

De två utgångarna på 4085 är standard TTL-utgångar, som var och en kan driva tio stycken TTL-ingångar.

4085 är speciellt lämpat för att avläsa många digitala insignaler. Om de två utgångarna styr en multiplexer som kopplar in 36 insignaler åt gången kan man avläsa 36x4 = 144 stycken insignaler.

INSTALLATION AV 4085.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exempel nedan används adress 6 för kort 4085.

Anslutning av yttre enheter.

Anslutning av de yttre enheterna på korten sker på I/O-kontakten. I/O-kontakten är den som sitter närmast lysdioden.

Montering av 4085 i expansionsenheten.

Slå av spänningen först! Kortet vänds så att I/O-kontakten kommer utåt och komponentsidan åt höger. 4085 skall placeras i I/O-delen på expansionsenheten.

Kontroll av adressen.

Efter montering av kortet i expansionsenheten bör man kontrollera att kortet byglats till rätt adress, genom att skriva:

```
OUT 1,<Kortadress>
```

<kortadress> avser den adress som kortet har getts. För att testa kort 4085 ge kommandot:

```
OUT 1,6
```

Om byglingen är riktig skall lysdioden nere till höger på kortet tändas. Om därefter annat kort väljes, med t.ex. kommando OUT 1,0, skall lysdioden släckas.

I/O-kommandon för 4085.

Basic	Aktiverad signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: kommando 10 Slask=INP (7) bör inleda alla program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: Med kommando 20 OUT 1,6 väljer vi kortet med adressen 6.
OUT 0,<Ingrupp+Utg>	OUT	<Ingrupp> är ett tal mellan 0 och 3 (bit 0 och bit 1) och väljer ingångsgrupp för läsning med INP-kommandot. <Utg> är något av talen 0,4,8,12 (bit 2 och bit 3) och styr utgångarna A och B. Exempel: 30 OUT 0,0 väljer grupp 0 för läsning samt nollställer utgångarna 0 och 1. Exempel: 40 OUT 0,2+4 väljer grupp 2 för läsning samt ettställer utgång 0 och nollställer utgång 1. Exempel: 50 OUT 0,1+4+8 väljer grupp 1 för läsning samt ettställer utgång 0 och 1.
INP (0)	INP	Läser vald ingångsgrupp enligt tidigare OUT-kommando. Exempel: 60 In=INP (0) tilldelar variabeln "In" inlästa data från vald ingångsgrupp.
INP (1)	STAT	Läser de fyra interrupt-ingångarna. Bit 0 i inlästa data motsvarar ingång 0, bit 1 motsvarar ingång 1, bit 2 har ingång 2 och bit 3 har ingång 3. Övriga bitar 4-7 blir ettställda. Signallogiken betyder "1" = passiv ingång, dvs. inget interrupt begärt. "0" = "aktiv ingång, dvs interrupt är begärt. Exempel: 70 IF INP (1)=240 THEN ; "INGÅNGARNA 0-3 ÄR NOLLSTÄLLDA"
OUT 5,<Int>	C4	Väljer interruptingångar. Bit 0 kontrollerar INT 0, bit 1 kontroller INT 1, osv. Signallogiken innebär: "1" = interrupt tillåtet. "0" = interrupt bortkopplat. Exempel: 80 OUT 5,2 innebär att avbrott är tillåtet för INT 1. Övriga interruptingångar är bortkopplade.

Styringångarna STB 0-3.

4085/ har fyra strobe-ingångar, (STB), som vardera styr 8 ingångar (en grupp). Med STB-signalen kan t.ex. en yttre enhet lagra insignaler i en vippa för senare avläsning från ABC80/800/DTC. Då STB går låg lagras ingångsvärdet i vipporna, fig. 3.21 nedan. Om STB ligger hög följer vippornas utgångar sina ingångar, dvs de åtta signaler som ligger på ingångarna förpassas direkt in på databussen. Skall STB ej användas för låsning av insignaler kan den bara lämnas öppen så antar den högt läge.

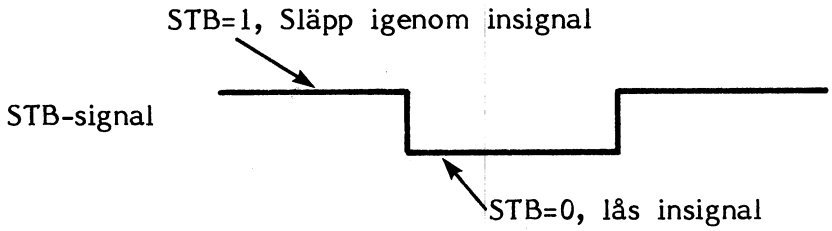


Fig. 3.21 STB-signalen för styrning av ingångsvippor på kort 4085.

Ingångarna INT FAS 0-3.

INT FAS används för att bestämma vilken flank som avbrottet skall ske på, se fig. 3.22 nedan. Om man vill ha avbrott på positiv (stigande) flank, dvs när INT-signalen går från låg till hög nivå, skall motsvarande INT FAS-ingång byglas till jord. Vill man däremot ha avbrott på negativ (fallande) flank, dvs när signalen går från hög nivå till låg nivå, skall motsvarande INT FAS-ingång lämnas öppen, eftersom denna ledning är ansluten till + 5 V via ett pull-up motstånd på kortet.

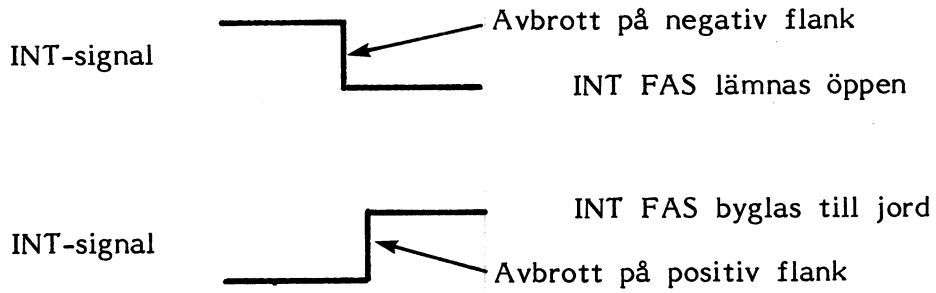


Fig. 3.22 INT FAS-signalen användes för att bestämma på vilken flank ett avbrott skall ske.

Exempel på användning.

När man använder 4085 och skall byta insignaler får man inte glömma att utgångarna ändras med samma kommando. För att undvika den sortens fel bör man ha aktuellt utgångsvärde i en variabel t.ex "Utg" och en variabel för val av ingångar t.ex "Ingrupp". När det då är dags att välja en ny ingrupp skriver man:

```
50 OUT 0,<Utg+Ingrupp>
```

Där "Utg" får anta värdena 0,4,8,12 och "Ingrupp" får innehålla värdena 0,1,2,3.

När man läser interruptingångarna får man ett värde som motsvarar värdet på interruptingångarna (bit 0-3) plus talet 240. 240 kommer från de övriga bitarna (4-7) i samma byte, som alla är ettställda.

Exempel 1: Styrning av utgångarna A och B.

Nedanstående program är ett exempel på hur de två utgångarna A och B kan styras. Uppkoppling enligt fig. 3.23.

```
10  ! STYRNING AV DE TVÅ UTSIGNALERNA PÅ 4085
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,6
60  ; : ; 'UTSIGNAL A='; : ; Lägea
70  ; 'UTSIGNAL B='; : ; Lägeb
80  ; : ; : INPUT 'SIGNAL A='Lägea
90  IF Lägea<0 OR Lägea>1 GOTO 80
100 INPUT 'SIGNAL B='Lägeb
110 IF Lägeb<0 OR Läge>1 GOTO 100
120 OUT 0,4*Lägea+8*Lägeb
130 GOTO 60
140 END
```

Kommentarer till ovanstående program: (tal hänvisar till programrad)

```
30  : Val av långa variabelnamn och heltal.
40  : Nollställer alla I/O-kort.
50  : Val av kort med adress 6.
60  : Utskrift av läget på utport A
70  : Utskrift av läget på utport B
80  : Frågar vad som skall skickas ut på port A
100 : Frågar vad som skall skickas ut på port B
120 : Skickar ut signalerna till portarna (A och B)
```

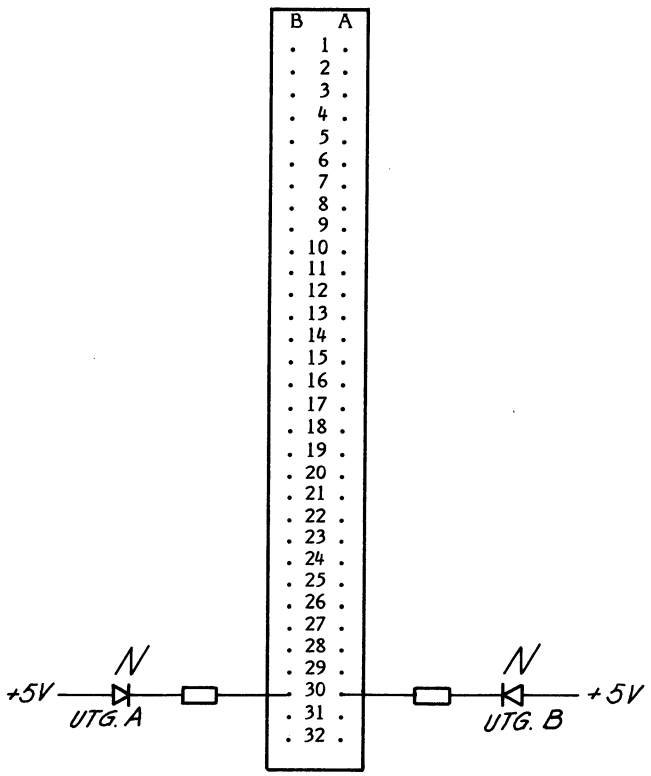


Fig. 3.23 Experimentuppkoppling för styrning av utgång A och B.

Exempel 2: Läsning och presentation av 32 ingångar.

Nedanstående program är ett exempel på hur man kan läsa 32 ingångar och presentera värdet. Uppkoppling enligt fig. 3.24.

```
10  ! LÄSNING AV 32 INSIGNALER MED 4085
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,6
60  ; CHR$(12)
70  INPUT 'VILKEN GRUPP (0-3) ?'Grupp
80  IF Grupp<0 OR Grupp>3 GOTO 70
90  OUT O,Grupp
100 ; : ; 'DECIMALTALET AV INLÄSNINGEN' INP(0)
110 ; 'BINÄRTALET: ';
120 FOR I=0 TO 7
130   ; (INP(0) AND 2üI)/2üI;
140 NEXT I
150 ; : ; : GOTO 70
160 END
```

Kommentarer till ovanstående program: (tal hänvisar till programrad)

30 : Val av långa variabelnamn och heltal.
40 : Nollställer alla I/O-kort.
50 : Väljer kort med adressen 6.
70 : Val av grupp för inläsning.
90 : Adresserar vald grupp.
100 : Skriver ut decimaltalet av inläsningen.
130 : Räknar ut binärtalet.

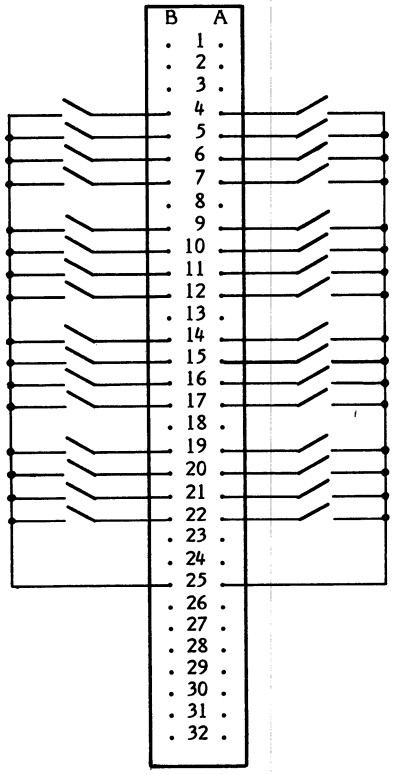


Fig. 3.24 Experimentuppkoppling för läsning av 32 ingångar.

Exempel 3: Avläsning av de fyra interruptingångarna.

Nedanstående program är ett exempel på hur man kan läsa de 4 interruptingångarna. Uppkoppling enligt fig. 3.25.

```
10  ! LÄSNING AV INTERRUPT SIGNALERNA
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,6
60  ; CHR$(12)
70  ; : ; 'DECIMALTALET AV INLÄSNINGEN' INP(1)-240
80  ; 'BINÄRTALET: ';
90  FOR I=3 TO 0 STEP -1
100   ; (INP(1) AND 2üI)/2üI;
110 NEXT I
120 ; : ; : ; : ; 'TRYCK RETURN FÖR NÄSTA VÄRDE'; : GET Slask$
130 ; : ; : GOTO 70
140 END
```

Kommentarer till ovanstående program: (tal hänvisar till programrad)

30 : Val av långa variabler och heltal.
40 : Nollställer alla I/O-kort.
50 : Väljer kort med adressen 6.
70 : Skriver ut de fyra interrupt ingångarna i decimal form.
100 : Räknar ut binärtalet av värdet.

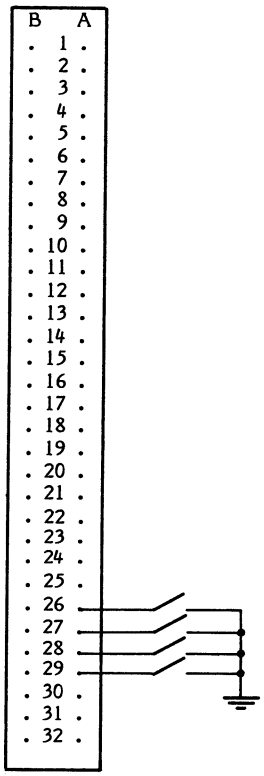


Fig. 3.25 Experimentuppkoppling för avläsning av INT-ingångar.

Exempel 4: Extern avbrottshantering.

Vid externa interrupt är alltid de åtta minst signifikanta bitarna i interruptvektorn 255. De åtta mest signifikanta bitarna finns i I-registret och kan anta varje värde mellan 0 och 255. I-registret kan laddas efter egen önskan men då måste man tänka på att flytta alla interna hoppadresser som interruptvektorerna pekar på. I ABC800/DTC ligger dessa på adress 65456 till 65476, (I-registret innehåller 255). I ABC80 finns det bara två vektorer, en för tangentbordet och en för bandspelarinterfacet: 52 respektive 54 (I-registret är noll). Om man använder tangentbordsinterrupt på ABC800/DTC behöver inte hopplistan flyttas, men på ABC80 måste man flytta den. I-registret laddas i detta exempel med 128. Vid externa interrupt måste I-registet och interrupttabell flyttas ner, detta för att man inte kan lägga in någon hoppadress på 65535 och 65536 (minnet slutar på 65535). I-registret måste alltid ändras i initieringsrutinen. I detta exempel görs även kortval, tillåter interrupt på alla kanaler och sätter en pekare som styr byglingen för interruptingång 0 och 1. Programmet är gjort så att man lätt kan ändra programmet och göra sina egna interruptrutiner. Om man vill göra detta skall man först ta bort allt mellan "HÄR BÖRJAR DEN EGNA RUTINEN" och "HÄR SLUTAR DEN EGNA RUTINEN" och därefter kan man lägga in sin egen rutin där. Om den nya rutinen använder några andra register än de som sparas vid början av interrupt-rutinen skall även dessa sparas och sedan lagras tillbaka före return. I A-registret finns det uppgifter om vilken ingång som gav interrupt, den bit som är nollställd indikerar ett interrupt på motsvarande ingång.

Programmet tar hand om två olika typer av interrupt. Den ena typen är "långa interrupt", dvs när man vill ha ett interrupt när signalen går ner (eller upp) och sedan ytterligare ett interrupt när signalen går upp igen (eller ner). Den andra typen är interruptsignaler som är kortare än ca 5 uS och som bara skall ge en enda interruptsignal. Interrupt-ingång 0 och 1 är programmerade för att ta hand om interrupt av den första typen. Byglingen av int fasen sker med hjälp av utgångarna på kortet. Den siffra som lagras i minnescell 65024 håller ordning på utgångarnas läge. Interruptingång 2 och 3 är programmerade för att ta hand om interrupt av den andra typen.

Uppkoppling till programmet enligt fig. 3.26.

```
INTERR    ZPROG ********** EXTERNT INTERRUPT **********
;
;
          ORG  32768
;
;INITIERINGS RUTIN SOM FLYTTAR ALLA INTERRUPT VEKTORER
;
BEGIN                        HÄR BÖRJAR INITIERINGSRUTINEN
          LD   A,128
          LD   I,A
          LD   A,6
          OUT  (1),A
          LD   A,12
          OUT  (0),A
          LD   A,15
          OUT  (5),A
          LD   HL,65024
          LD   (HL),3
          RET            HÄR SLUTAR INITIERINGSRUTINEN
;
```

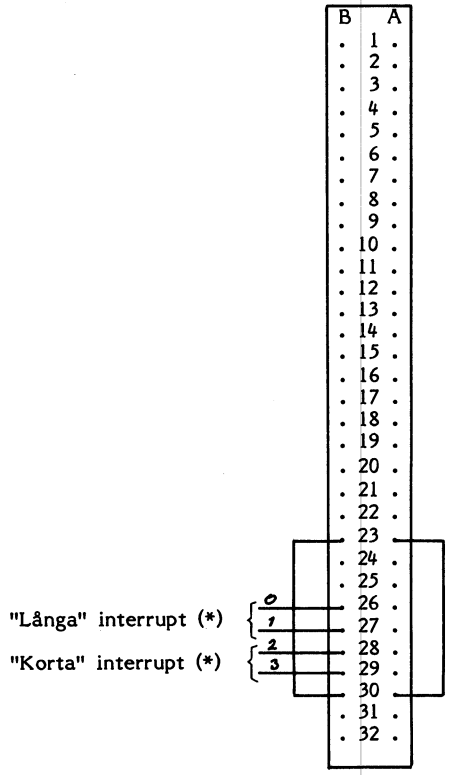
```
; INTERRUPT VEKTORERNA
;
          ORG  32768+176
;
; DART INTERRUPT VEKTORER
;
          DEFW T917
          DEFW T917
          DEFW T891
          DEFW T919
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
;
; SIO INTERRUPT VEKTORER
;
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T917
;
; CTC INTERRUPT VEKTORER
;
          DEFW T917
          DEFW T917
          DEFW T917
          DEFW T14994
;
; VEKTOR FRÅN DE EXTERNA INTERRUPTEN
;
          ORG  32768+255
;
          DEFW EXTERN
;
; HOPPLISTA FÖR INTERRUPTEN
;
EXTERN    JP   ÅTGÄRD
;
T917      JP   917
;
T891      JP   891
;
T919      JP   919
;
T14994    JP   14994
;
ÅTGÄRD
          PUSH AF              SPARAR REGISTER FÖR
          PUSH HL              ÅTERHOPPET
          PUSH BC
          PUSH DE
          LD   A,6
          OUT  (1),A
          LD   A,0
          OUT  (5),A
          IN   A,(1)
```

```
BÖRJAN                                  HÄR BÖRJAR DEN EGNA RUTINEN
          XOR   255                     INVERTERAR A-REGISTRET
          BIT   0,A                     KOLLAR VILKEN INGÅNG SOM GETT
          JR    NZ,KANAL1               INTERRUPT
          BIT   1,A
          JR    NZ,KANAL2
          BIT   2,A
          JR    NZ,KANAL3
          BIT   3,A
          JR    NZ,KANAL4
KANAL1
          LD    HL,65024                KONTROLLERAR LÄGET OCH ÄNDRAR
          LD    A,(HL)                  INT FAS
          BIT   0,A
          JR    NZ,NOLLA1
          SET   0,A                     SÄTTER INT FAS
          LD    (HL),A                  SPAR RESULTATET
          JR    ÅTGÄRD1
NOLLA1
          AND   254                     NOLLAR INT FAS
          LD    (HL),A                  SPAR RESULTATET
          JR    ÅTGÄRD1
KANAL2
          LD    HL,65024                KONTROLLERAR LÄGET OCH ÄNDRAR
          LD    A,(HL)                  INT FAS
          BIT   1,A
          JR    NZ,NOLLA2
          SET   1,A                     SÄTTER INT FAS
          LD    (HL),A                  SPAR RESULTATET
          JR    ÅTGÄRD2
NOLLA2
          AND   253                     NOLLAR INT FAS
          LD    (HL),A                  SPAR RESULTATET
          JR    ÅTGÄRD2
KANAL3
          LD    HL,TEXT3                SKRIVER UT OM DET BLEV INTERRUPT
          LD    BC,9                    PÅ INGÅNG 2
          CALL  11                      UTSKRIFT PÅ SKÄRM
          JR    AVSLUT
KANAL4
          LD    HL,TEXT4                SKRIVER UT OM DET BLEV INTERRUPT
          LD    BC,9                    PÅ INGÅNG 3
          CALL  11                      UTSKRIFT PÅ SKÄRM
          JR    AVSLUT
ÅTGÄRD1
          CALL  UT                      SÄNDER UT ÄNDRINGEN PÅ INT FAS
          LD    HL,TEXT1                SKRIVER UT OM DET BLEV INTERRUPT
          LD    BC,9                    PÅ INGÅNG 0
          CALL  11                      UTSKRIFT PÅ SKÄRM
          JR    AVSLUT
ÅTGÄRD2
          CALL  UT                      SÄNDER UT ÄNDRINGEN PÅ INT FAS
          LD    HL,TEXT2                SKRIVER UT OM DET BLEV INTERRUPT
          LD    BC,9                    PÅ INGÅNG 1
          CALL  11                      UTSKRIFT PÅ SKÄRM
          JR    AVSLUT
;
TEXT1     DEFM  ' KANAL 1 '
TEXT2     DEFM  ' KANAL 2 '
TEXT3     DEFM  ' KANAL 3 '
TEXT4     DEFM  ' KANAL 4 '
```

```
;
UT
            RLCA                    ÄNDRAR INT FAS TILL NYTT LÄGE
            RLCA
            OUT   (0),A
            RET                     HÄR SLUTAR DEN EGNA RUTINEN
AVSLUT
            LD    A,15
            OUT   (5),A
            EI
            POP   DE                LADDAR TILLBAKA VÄRDENA SOM
            POP   BC                SPARADES VID UTHOPPET
            POP   HL
            POP   AF
            RETI
            END   BEGIN
```

När man har assemblerar programmet ovan och gjort om det till en basic-fil får man programmet nedan som kan länkas in i ett annat basic-program.

```
10  INTEGER : EXTEND
15  POKE 65292,0,130 : POKE 65328,0,130 ! HÖJER GOLVET 0,5k BYTE
16  !
17  ! ASSEMBLERRUTINEN I SIFFROR
18  !
20  POKE 32768,62,128,237,71,62,6,211,1,62,12
30  POKE 32778,211,0,62,15,211,5,33,128,255,54
40  POKE 32788,3,201
50  POKE 32944,4,129,4,129,7,129,10,129,4,129
60  POKE 32954,4,129,4,129,4,129,4,129,4,129
70  POKE 32964,4,129,4,129,4,129,4,129,4,129
80  POKE 32974,4,129,4,129,4,129,4,129,13,129
90  POKE 33023,1,129,195,16,129,195,149,3,195,123
100 POKE 33033,3,195,151,3,195,146,58,245,229,197
110 POKE 33043,213,62,6,211,1,62,0,211,5,219
120 POKE 33053,1,238,255,203,71,32,12,203,79,32
130 POKE 33063,26,203,87,32,40,203,95,32,47,33
140 POKE 33073,128,255,126,203,71,32,5,203,199,119
150 POKE 33083,24,45,230,254,119,24,40,33,128,255
160 POKE 33093,126,203,79,32,5,203,207,119,24,41
170 POKE 33103,230,253,119,24,36,33,152,129,1,9
180 POKE 33113,0,205,11,0,24,80,33,161,129,1
190 POKE 33123,9,0,205,11,0,24,69,205,170,129
200 POKE 33133,33,134,129,1,9,0,205,11,0,24
210 POKE 33143,55,205,170,129,33,143,129,1,9,0
220 POKE 33153,205,11,0,24,41,32,75,65,78,65
230 POKE 33163,76,32,49,32,32,75,65,78,65,76
240 POKE 33173,32,50,32,32,75,65,78,65,76,32
250 POKE 33183,51,32,32,75,65,78,65,76,32,52
260 POKE 33193,32,7,7,211,0,201,62,15,211,5
270 POKE 33203,251,209,193,225,241,237,77
280 Start=CALL(32768)
```



(*) Förklaring ges på sidan 53.

Fig. 3.26 Experimentuppkoppling för interrupt.

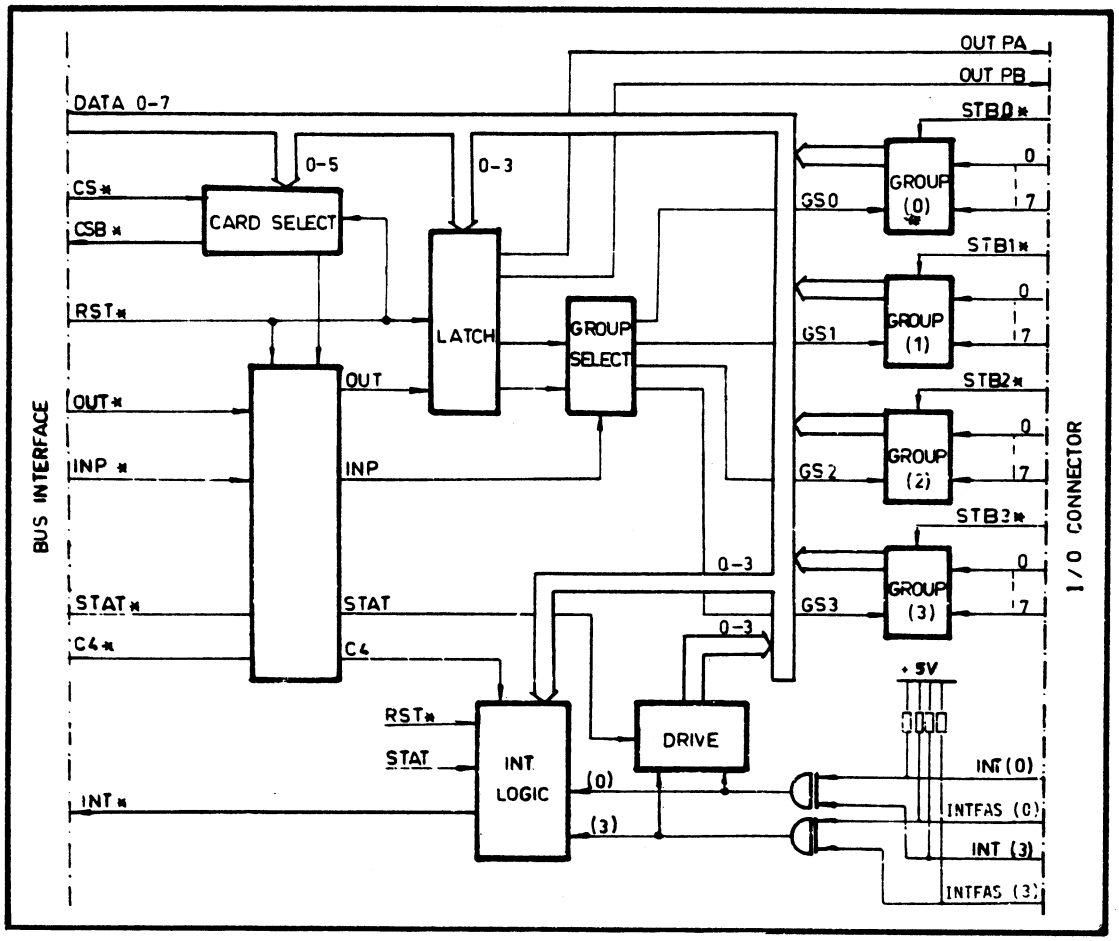


Fig. 3.28 Blockschema för digitalt I/O-kort 4085.

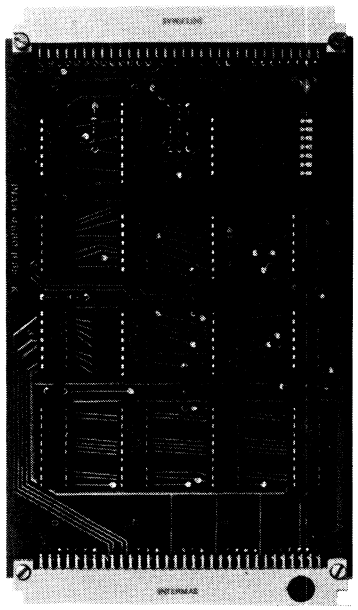


Fig. 3.27 Digitalt I/O-kort 4085.

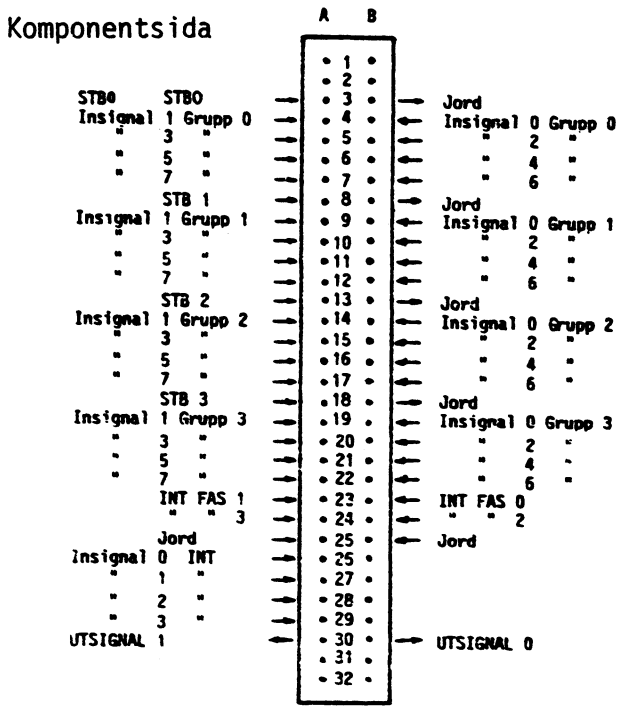


Fig. 3.29 Stiftlayout för I/O-kontakt

3.4 Digital signalering med reläkort 4103

4103

16 reläutgångar med växlande kontakt.
Relä av typ RH-5 med följande specifikationer:
Kontaktspänning max 110 V
Kontaktström max 1 A
Bryteffekt max 20 W
Drivspänning 5 V
Från/tillslagstid 1 ms

4103 har 16 stycken reläutgångar med växlande kontakter, se figuren nedan:

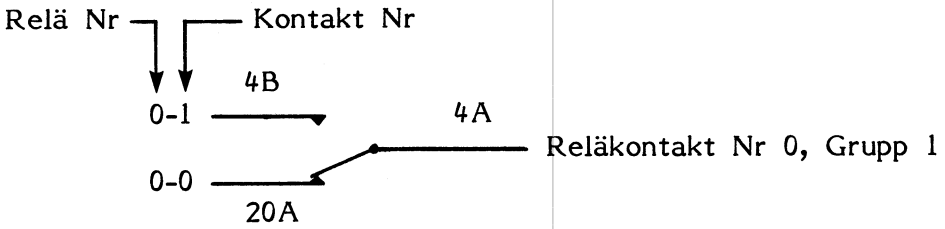


Fig. 3.30 Reläkontaktnumrering. I figuren visas kontakternas läge vid ej draget relä.

Reläerna är arrangerade i två grupper om åtta i varje. Reläerna i de två grupperna kan styras individuellt med två OUT-instruktiner, (se nedan). Om så önskas kan den interna drivspänningen (+5 V) till reläerna lätt ändras till en yttre drivspänning. Detta sker genom att ändra läget på en bygel. 4103 är program och anslutningskompatibelt, med avseende på till- repsktive frånslag, med kort 4095. 4103 är även programkompatibelt med 4005, då med tanke på utgångarna. Reläerna på kortet är av typen RH-5, som klarar 100 V max som kontaktspänning och en maximal kontaktström av 1 A. Reläerna har en bryteffekt på 20 W maximalt med en omslagstid på 1 mS.

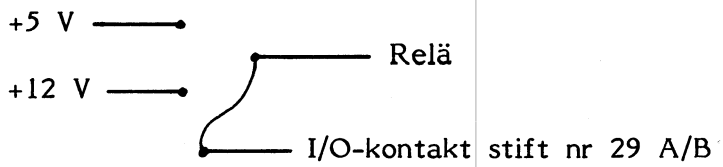
INSTALLATION AV RELÄKORT 4103.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exempel nedan används adress 8 för kort 4103.

Ändring av relädrivspänning.

Om man av någon anledning vill använda en extern drivspänning för reläerna i stället för den interna 5 V drivspänningen skall byglingen i position 3E flyttas till understa läget se nedan:



Anslut den yttre drivspänningen till stift 29A och/eller 29B, jorden skall placeras på stift 2A och/eller 2B på I/O-kontakten (den närmast lysdioden).

Anslutning av yttre enheter.

Anslutning av yttre enheter till kortet sker via I/O-kontakten, som är den kontakt som sitter närmast lysdioden.

Montering av 4103 i expansionsenheten.

Slå av spänningen först! Kortet vänds så att I/O-kontakten kommer utåt och komponentsidan åt höger. 4103 skall placeras i I/O-delen på expansionsenheten.

Kontroll av adressen.

Efter montering av kortet i expansionsenheten bör man kontrollera att kortet byglats till rätt adress, genom att skriva:

OUT 1,<Kortadress>

Där <kortadress> avser den adress som kortet har getts. För att testa kort 4103 ge kommandot:

OUT 1,8

Om byglingen är riktig skall lysdioden nere till höger på kortet tändas. Om därefter något annat kort väljes, med t.ex. kommando OUT 1,0 skall lysdioden släckas.

I/O-Kommandon för 4103.

Basic	Aktiverad signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP (7) bör inleda alla program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: 20 OUT 1,8 väljer ut kortet med adressen 8.
OUT 0,<Data>	OUT	De åtta minst signifikanta bitarna av heltalsvariabeln <Data> läggs ut till grupp 1, relä 0-7. Exempel: 30 OUT 0,4 sluter reläkontakt 2 (2ü2=4) mot 2-1, medan övriga kontakter i samma grupp sluts mot (nr) -0. Se fig. 3.28 och 3.32
OUT 2,<Data>	C1	De åtta minst signifikanta bitarna av heltalsvariabeln <Data> läggs ut som data till grupp 2, relä 8-15. Exempel: 40 OUT 2,255 medför att alla reläer i grupp 2 sluts mot kontakter (nr)-1. Se även fig. 3.28 och 3.32.
OUT 4,0	C3	Släpper alla 16 reläer. Exempel: 50 OUT 4,0 medför att alla reläer (grupp 1 och 2) sluts till läge (nr)-0.

Exempel 1: Rinnande ljus med reläkort 4103

Programmet tänder och släcker åtta lysdioder så att det ser ut som en ljuspunkt som förflyttar sig längs raden av lysdioder, ett s.k. rinnande ljus. Uppkoppling enligt fig. 3.29.

```
10  ! RINNANDE LJUS MED HJÄLP AV 4103
20  !
30  EXTEND : INTEGER
40  !
50  Slask=INP(7) ! RESET
60  OUT 1,8 ! KORTVAL
70  !
80  FOR Loop=0 TO 7
90    IF Loop=0 OUT 0,2ü0+2ü7
100     IF Loop>0 OUT 0,(2üLoop)+(2ü(Loop-1))
110     FOR Vänta=0 TO 200 : NEXT Vänta
120 NEXT Loop
130 GOTO 80
140 END
```

Kommentarer till ovanstående program: (Tal hänvisar till programrader)

30 : Tillåter långa variabelnamn och sätter heltalsmod.
40 : Nollställer alla I/O-kort.
50 : Väljer kort med adressen 8.
90 : Tänder diod noll och sju om "Loop"=0.
100 : Tänder två dioder när "Loop" är större än noll.
110 : Väntar en stund.

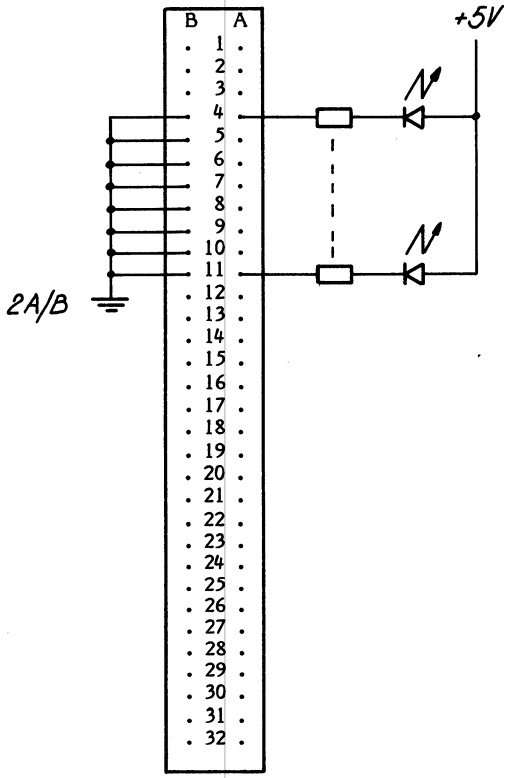


Fig. 3.31 Experimentuppkoppling för rinnande ljus med reläkort 4103.

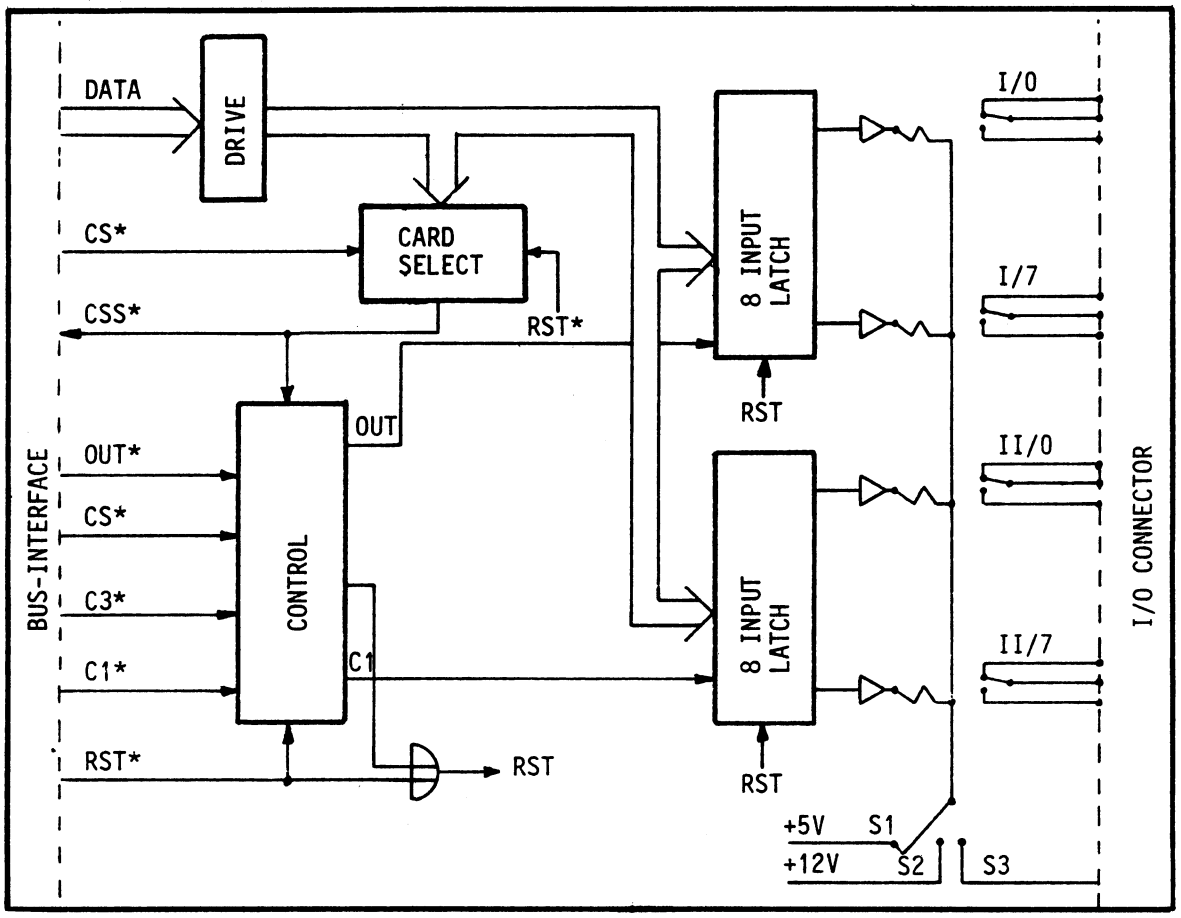


Fig. 3.33 Blockschema för reläkort 4103.

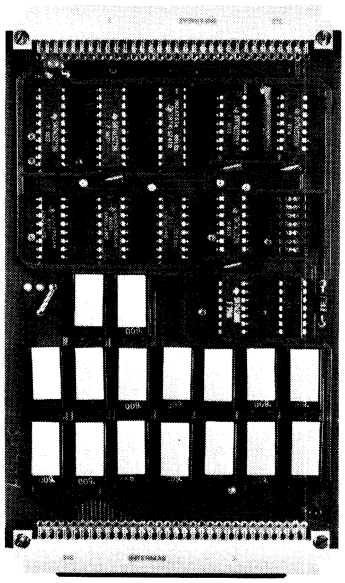


Fig. 3.32 Reläkort 4103.

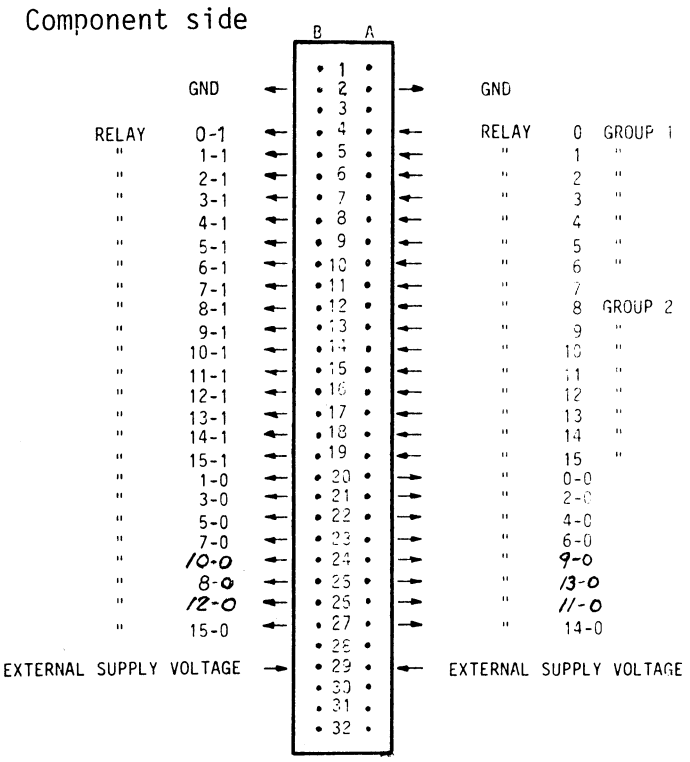


Fig. 3.34 Stiftlayout för I/O-kontakt hos 4103.

4. Analog signalering

I följande avsnitt presenteras kortmoduler för analog signalering, som digital-till-analog och analog-till-digital omvandlare samt PT100 mätsystem.

Till varje kortmodul återfinns ett eller flera programexempel som visar hur modulen kan programmeras och användas.

4083	2 x 12 bitars digital-till-analog omvandlare.
4084	4 x 8 bitars digital-till-analog omvandlare.
4115	32 ingångars analog-till-digital omvandlare.
4021	PT100 mätsystem för temperaturmätning.
4022	Multiplexer för anslutning av upp till 12 PT-100 givare.

4.1 Digital/analogomvandlare 4083 och 4084

4083

2 x 12 bitars digital till analog-omvandlare.
Omvandlingstid 5 µs.
Upplösning: cirka 0,025 % av maxvärdet.

4084.

4 x 8 bitars digital till analogomvandlare.
Omvandlingstid 5 µs.
Upplösning: cirka 0,4 % av maxvärdet.

4083 och 4084 omvandlar digitala signaler från datorn till analoga utsignaler. Med 4083 kan man styra två analoga utsignaler med 12 bitars upplösning medan 4084 har fyra analoga utgångar med 8 bitars upplösning. På korten finns två referensspänningar, -5 V och -10 V. Man kan även ansluta en extern referensspänning från -12 V till +12 V. Detta gör att man har fyra möjliga utsignalområden:

0 till 10 V (-10 V referens)
0 till 5 V (-5 V referens)
0 till referens (yttre referens)
0 till 20 mA (max 500 ohms belastning)

Upplösningen för 4083 är maxutslag/4095 vilket motsvarar ungefär 0.025% av maxvärdet. 4084 har en upplösning på maxutslag/255 som motsvarar ungefär 0.4% av maxvärdet. Båda har en omvandlingstid på ungefär 5 mikrosekunder.

Exempel på användningsområden är:

Styrning av funktionsgeneratorer.
Registrering av resultat på XY-skrivare.
Digital syntesgenerering.
Analog visning på tavelinstrument.

INSTALLATION AV DAC-KORT 4083 OCH 4084.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exempel nedan användes adress 11 för kort 4083 och adress 12 för kort 4084.

Anslutning av yttre enheter.

Anslutning av yttre enheter till kortet sker via I/O-kontakten. I/O-kontakten är den som sitter närmast lysdioden.

Montering av 4083 och 4084 i expansionsenheten

Slå av spänningen först! Korten vänds så att I/O-kontakten kommer utåt och komponentsidan åt höger. Både 4083 och 4084 skall placeras i I/O-delen på expansionsenheten.

Bygling för val av referens.

Byglingarna görs i anslutningskontakten. Observera att endast en bygling per kanal får göras!

Byglingar för 4083.

Kanal 1

Område:

0 till 10 V bygla 7A till 14B. Utsignal på stift 12B.
0 till 5 V bygla 4A till 14B. Utsignal på stift 12B.
0 till 20 mA bygla 4A till 14B. Utsignal på stift 3B.
0 till yttre referens, anslut den yttre referensen på stift 14B. Utsignal på stift 12B.

Kanal 2

Område:

0 till 10 V bygla 7B till 28B. Utsignal på stift 18B.
0 till 5 V bygla 4B till 28B. Utsignal på stift 18B.
0 till 20 mA bygla 4B till 28B. Utsignal på stift 27B.
0 till yttre referens, anslut den yttre referensen på stift 28B. Utsignal på stift 18B.

Bygligar för 4084.

Område 0 till 10 V.

Kanal 1: Bygla 29B till 7B. Utsignal på stift 5B.
Kanal 2: Bygla 29B till 13B. Utsignal på stift 11B.
Kanal 3: Bygla 29B till 17B. Utsignal på stift 19B.
Kanal 4: Bygla 29B till 25B. Utsignal på stift 23B.

Område 0 till 5 V.

Kanal 1: Bygla 27B till 7B. Utsignal på stift 5B.
Kanal 2: Bygla 27B till 13B. Utsignal på stift 11B.
Kanal 3: Bygla 27B till 17B. Utsignal på stift 19B.
Kanal 4: Bygla 27B till 25B. utsignal på stift 23B.

Område 0 till 20 mA.

Kanal 1: Bygla 27B till 7B. Utsignal på stift 3B.
Kanal 2: Bygla 27B till 13B. Utsignal på stift 9B.
Kanal 3: Bygla 27B till 17B. Utsignal på stift 15B.
Kanal 4: Bygla 27B till 25B. Utsignal på stift 21B.

Område 0 till yttre referens.

Kanal 1: Anslut yttre referens till 7B. Utsignal på stift 5B.
Kanal 2: Anslut yttre referens till 13B. Utsignal på stift 11B.
Kanal 3: Anslut yttre referens till 19B. Utsignal på stift 17B.
Kanal 4: Anslut yttre referens till 25B. Utsignal på stift 23B.

Utsignalerna tas mellan angivna stift och signaljord.

Kontroll av adress

Efter montering av kortet i expansionsenheten bör man kontrollera att kortet byglats till rätt adress, genom att skriva:

OUT 1,<Kortadress>

Där <Kortadress> avser den adress som kortet har getts. För att testa kort 4083 ge kommando:

Praktisk användning av 4083.

4083 använder tolv bitar, som motsvarar ett tal mellan 0 och 4095. Men eftersom databussen bara kan sända åtta bitar åt gången, dvs. ett tal mellan 0 och 255, så måste man skicka talet i två omgångar. De åtta minst signifikanta bitarna skickar man till port 0 eller 3 beroende på kanal. De fyra mest signifikanta bitarna skickas till port 2 eller 4, även det beroende på kanal. För att hantera detta har man stor hjälp av instruktionen SWAP%(Data), som beskrivits i ett tidigare avsnitt. SWAP% kommandot förenklar signaleringen betydligt, som vi lättast ser med ett exempel:

```
OUT 3,<Data>,4,SWAP%(<Data>)
```

<Data> är det digitala tal som sänds ut för omvandling till ett analogt tal, i kanal 2. Det som händer när datorn utför denna sats är att de åtta minst signifikanta bitarna i <Data> sänds till port 3 varefter de två byte, som talet består av, byter plats i talet <Data> med hjälp av SWAP% kommandot. De åtta mest signifikanta bitarna blir då de åtta minst signifikanta bitarna som sedan läggs ut på port 4.

Ett exempel : <Data>=4035 blir i binär form: 0000 1111 1100 0011. Om vi låter SWAP-kommandot operera på detta tal får vi då: 1100 0011 0000 1111. Omvandlingstiden är i basic ungefär 250 µS ,om man skriver kommandot som ett sammansatt kommando, som ovan. Om man istället delar upp den raden i två rader så får man en betydligt längre omvandlingstid. Vill man ha mindre omvandlingstider måste man skriva programmet i assembler. Då kan man nå ner till omvandlingstider kring 5 µS. I program senare i detta avsnitt visas hur man kan göra detta.

Praktisk användning av 4084.

4084 arbetar med åtta bitars upplösning (ett tal mellan 0 och 255) och fyra kanaler. Upplösningen är alltså något sämre än 4083 men istället har man fyra kanaler till sitt förfogande. Att 4084 har åtta bitars upplösning innebär att man inte behöver bekymra sig över två portar när man skall lägga ut data på en kanal, utan med 4084 räcker det med ett OUT-kommando.

Exempel 1: Strömstyrning med D/A-kortet 4083.

Programmet styr strömutgångarna på kort 4083. Experimentuppkopplingen visas i fig. 4.1 nedan.

```
10  ! STRÖMSTYRNING MED HJÄLP AV D/A-KORTET 4083
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,11
60  ; CHR$(12)
70  INPUT 'VILKEN KANAL SKALL STÄLLAS ?'Kanal
80  IF Kanal<1 OR Kanal>2 GOTO 70
90  ;
100 INPUT 'ANGE STRÖMMEN I mA (0-20) ?'Strömmen
110 IF Strömmen<0 OR Strömmen>20 GOTO 100
120 !
130 Utdata=4095/10*Strömmen/2
140 !
150 IF Kanal=1 OUT 0,Utdata,2,SWAP%(Utdata)
160 IF Kanal=2 OUT 3,Utdata,4,SWAP%(Utdata)
170 ; : ; : ; : GOTO 70
180 END
```

OUT 1,11

För att testa kort 4084 ge kommando:

OUT 1,12

Om byglingen är riktig skall lysdioden nere till höger på kortet tändas. Om därefter annat kort väljes, med t.ex. kommando OUT 1,0 skall lysdioden släckas.

I/O-Kommandon för 4083.

Basic	Aktiverad signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP (7) bör inleda alla program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: 20 OUT 1,11 väljer ut kort med adressen 11 och tänder lysdioden på kortet.
OUT 0,<Data>	OUT	Bitarna (0-7) i <Data> läggs ut för omvandling som de åtta minst signifikanta bitarna (0-7) på kanal 1.
OUT 2,<Data>	C1	Bitarna 0-3 i <Data> läggs ut för omvandling som de fyra mest signifikanta bitarna (8-11) till kanal 1. Exempel: 30 OUT 0,Data,2,SWAP%(Data) sänder ut innehållet i <Data> för omvandling i kanal 1.
OUT 3,<Data>	C2	Fungerar likadant som OUT 0,<Data> men sänder ut data till kanal 2 istället för kanal 1.
OUT 4,<Data>	C3	Fungerar likadant som OUT 2,<Data> men sänder ut data till kanal 2 istället för kanal 1. Exempel: 40 OUT 3,4095,4,SWAP%(4095) ger max utslag på kanal 2.

I/O-kommandon för 4084.

OUT 0,<Data>	OUT	Sänder ut de åtta minst signifikanta bitarna av <Data> för omvandling till kanal 1. Exempel: 30 OUT 0,255 Sänder ut talet 255 till kanal 1 och omvandlar det till en analog signal. 255 ger max utslag.
OUT 2,<Data>	C1	Samma som OUT 0,Data men för kanal 2.
OUT 3,<Data>	C2	Samma som OUT 0,Data men för kanal 3.
OUT 4,<Data>	C3	Samma som OUT 0,Data men för kanal 4.

Kommentarer till ovanstående program:

30 : Val av långa variabler och heltalsbasic.
40 : Nollställning av alla I/O-kort.
50 : Kortval. Här adresseras kortet med adressen 12.
130 : Formeln för datan som skall skickas ut.
150 : Data skickas ut till kanal 1.
160 : Data skickas ut till kanal 2.
170 : Data skickas ut till kanal 3.
180 : Data skickas ut till kanal 4.

Exempel 8: Styrning av spänningsutgångarna hos kort 4084.

Uppkoppling till programmet nedan ses i fig. 4.4. Programmet styr spänningsutgångarna på 4084.

```
10  ! SPÄNNINGSSTYRNING MED HJÄLP AV D/A-KORTET 4084
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,12
60  ; CHR$(12)
70  INPUT 'VILKEN KANAL SKALL STÄLLAS ?'Kanal
80  IF Kanal<1 OR Kanal>4 GOTO 70
90  ;
100 INPUT 'ANGE SPÄNNINGEN I V (0-10) ?'Spänningen.
110 IF Spänningen.<0 OR Spänningen.>10 GOTO 100
120 !
130 Utdata=255/10*Spänningen.
140 !
150 IF Kanal=1 OUT 0,Utdata
160 IF Kanal=2 OUT 2,Utdata
170 IF Kanal=3 OUT 3,Utdata
180 IF Kanal=4 OUT 4,Utdata
190 ; : ; : ; : GOTO 70
200 END
```

Kommentarer till ovanstående program:

30 : Val av långa variabelnamn och heltalsbasic.
40 : Nollställning av alla I/O-kort.
50 : Kortval. Här adresseras kortet med adressen 12.
130 : Formeln för data som skall skickas ut.
150 : Data skickas ut till kanal 1.
160 : Data skickas ut till kanal 2.
170 : Data skickas ut till kanal 3.
180 : Data skickas ut till kanal 4.

Exempel för kort 4084.

Exempel 7: Styrning av strömutgångarna hos kort 4084.

Programmet nedan styr strömutgångarna på kort 4084. Uppkoppling enligt fig. 4.3.

```
10  ! STRÖMSTYRNING MED HJÄLP AV D/A-KORTET 4084
20  !
30  EXTEND : INTEGER
40  Slask=INP(7)
50  OUT 1,12
60  ; CHR$(12)
70  INPUT 'VILKEN KANAL SKALL STÄLLAS ?'Kanal
80  IF Kanal<1 OR Kanal>4 GOTO 70
90  ;
100 INPUT 'ANGE STRÖMMEN I mA (0-20) ?'Strömmen
110 IF Strömmen<0 OR Strömmen>20 GOTO 100
120 !
130 Utdata=255/10*Strömmen/2
140 !
150 IF Kanal=1 OUT 0,Utdata
160 IF Kanal=2 OUT 2,Utdata
170 IF Kanal=3 OUT 3,Utdata
180 IF Kanal=4 OUT 4,Utdata
190 ; : ; : ; : GOTO 70
```

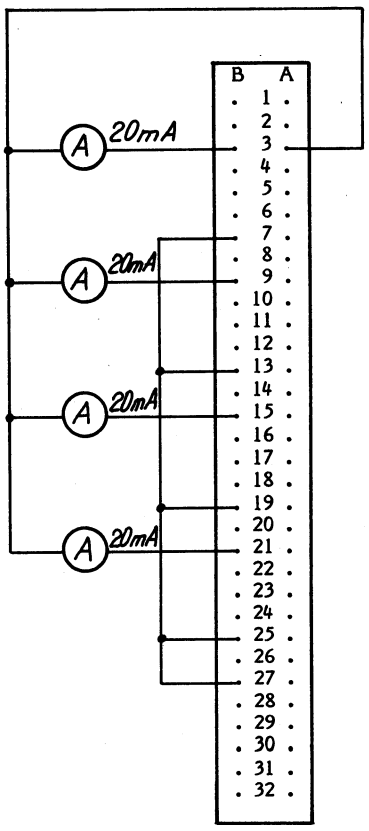


Fig. 4.3 Experimentuppkoppling för styrning av 4084 strömutgångar.

Exempel 6 är en tillämpning av assemblerprogrammet i exempel 5. BASIC-programmet i exempel 6 bygger upp en tabell för generering av en sinussignal som läggs upp i minnet där sedan assemblerprogrammet läser i den genererade tabellen och skickar ut den till kanal 1.

Exempel 6: Generering av sinusvåg

```
10  ! PROGRAMMET GENERERAR EN SINUSSIGNAL MED
      D/A KORTET 4083
20  !
30  EXTEND : INTEGER
40  Start=65024
50  !
60  ! ***** ASSEMBLER-RUTINEN 5 ******
70  !
80  POKE 65024,6,50,33,255,251,35,126,211,0,35
90  POKE 65034,126,211,2,16,246,24,239
100 !
110 Slask=INP(7)
120 OUT 1,11
130 !
140 FOR Loop.=0. TO 2.*PI STEP 2.*PI/50.
150   !
160   ! ***** GENERERING AV SINUSSIGNALEN *****
170   !
180   Sinus=(.5+SIN(Loop.)/2)*4095
190   !
200   ! ***** UPPLÄGGNING AV VÄRDENA *****
210   !
220   POKE 64512+Plus,Sinus,SWAP%(Sinus)
230   !
240   Plus=Plus+2
250 NEXT Loop.
260 Slask=CALL(Start)
270 END
```

Kommentarer till ovanstående program:

30 : Val av långa variabelnamn och heltalsbasic.
40 : Variabel som anger var assemblern läggs in. OBS Det räcker inte med att ändra variabeln för att flytta assemblerprogrammet i minnet.
50 : Objektkoden av assemblerprogrammet.
60 : Objektkoden av assemblerprogrammet.
110 : Nollställning av alla I/O-kort.
120 : Kortval. Här adresseras kortet med adressen 11.
180 : Formeln för funktionen som skall genereras.
220 : Uppläggning i minnet av de genererade värdena.
260 : Anrop av assemblerprogrammet som sänder ut funktionen.

Exempel 3: Kanal 1

Objektkod	Mnemonics
123	LD A,E
211 0	OUT (0),A
122	LD A,D
211 2	OUT (2),A
201	RET

Exempel 4: Kanal 2

123	LD A,E
211 3	OUT (3),A
122	LD A,D
211 4	OUT (4),A
201	RET

Exempel 5 läser 50 värden från minnet. Det första värdet finns i minnescell 64512 och 64513, medan det sista finns i 64612 och 64513. Varje värde består av två byte eftersom 4083 kräver tolv bitar. När första byte är inläst skickas den ut på port 0 varefter HL ökas med ett, (det register som håller reda på var det aktuella värdet skall hämtas). När detta är gjort läses det nya värdet och skickas ut till port 2. Därefter räknas B-registret ner med ett och kontrolleras att det inte är noll. Är B-registret noll så börjar proceduren om från början, om inte så läses nästa värde och skickas ut till kanal 1.

Exempel 5:

Objektkod		Mnemonics			Rad
6	50	START:	LD	B,50	1
33	255 251		LD	HL,64511	2
35		LOOP:	INC	HL	3
126			LD	A,(HL)	4
211	0		OUT	(0),A	5
35			INC	HL	6
126			LD	A,(HL)	7
211	2		OUT	(2),A	8
16	246		DJNZ	LOOP	9
24	239		JR	START	10

Kommentarer till ovanstående program:

1 : Laddar B registret med 50.
2 : Laddar HL registret med 64511.
3 : Ökar värdet i HL med 1. HL=HL+1
4 : Laddar A registret med innehållet i den minnescell som HL pekar på. Här laddas de åtta minst signifikanta bitarna som sänds ut till kanal 1.
5 : Skickar ut värdet i A registret till port 0. De åtta minst signifikanta bitarna.
6 : Ökar HL registret med 1. HL=HL+1
7 : Laddar de fyra mest signifikanta bitarna till A registret.
8 : Sänder ut de fyra mest signifikanta bitarna till port 2.
9 : Minskar B registret med ett. B=B-1. Om B inte blir lika med noll så är inte perioden färdig. Då hoppar processorn till label LOOP och tar nästa värde (två byte).
10 : B-registret blev noll och då skall en ny period börja. Processorn hoppar till label START och det hela börjar om.

```
160 IF Kanal=2 OUT 3,Utdata,4,SWAP%(Utdata)
170 ; : ; : ; : GOTO 70
180 END
```

Kommentarer till ovanstående program:

30 : Val av långa variabelnamn och heltalsbasic.
40 : Nollställning av alla I/O-kort.
50 : Kortval. Kortet med adressen 11 väljs i detta fall.
130 : Formeln för beräkning av värdet som skall sändas ut.
150 : Skickar ut data till kanal 1.
160 : Skickar ut data till kanal 2.

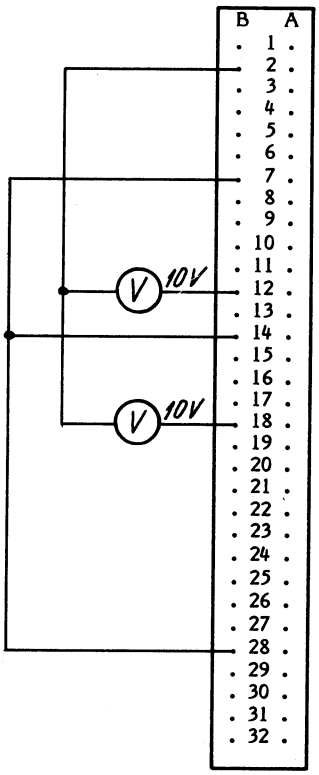


Fig 4.2 Experimentuppkoppling för styrning av 4083 spänningsutgångar.

Styrning av 4083 från assembler.

För att kunna använda 4083 till att generera diverse olika periodiska funktioner så bör man använda assembler. Ett lätt sätt att generera en sinuskurva är att först skriva ett program i basic som lägger upp en sinustabell i minnet (Se exempel 6). Sedan läser man av den tabellen från assemblerprogrammet och skickar ut den till D/A-kortet. Exempel 3 och 4 är två prov på hur man kan skicka med ett värde till en assemblerrutin som i sin tur skickar ut värdet till D/A-kortet. Det man bör tänka på i detta fall är att talet som skickas med till rutinen hamnar i DE-registret. De åtta mest signifikanta siffrorna hamnar i register D och de åtta minst signifikanta siffrorna i E.

Kommentarer till ovanstående program: (Tal hänvisar till programrad)

30 : Val av långa variabelnamn och heltalsbasic.
40 : Nollställning av alla I/O-kort.
50 : Kortval. Kortet med adressen 11 väljs i detta fall.
130 : Formeln för beräkning av värdet som skall sändas ut.
150 : Sänder ut data till kanal 1.
160 : Sänder ut data till kanal 2.

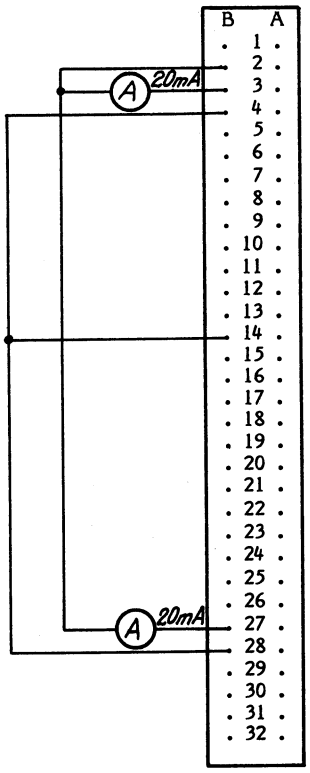


Fig. 4.1 Experimentuppkoppling för styrning av 4083 strömutgångar.

Exempel 2: Styrning av spänningsutgångarna på kort 4083.

Följande program styr spänningsutgångarna på kort 4083. Experimentuppkopplingen visas i fig. 4.2.

```
10 ! SPÄNNINGSSTYRNING MED HJÄLP AV D/A-KORTET 4083
20 !
30 EXTEND : INTEGER
40 Slask=INP(7)
50 OUT 1,11
60 ; CHR$(12)
70 INPUT 'VILKEN KANAL SKALL STÄLLAS ?'Kanal
80 IF Kanal<1 OR Kanal>2 GOTO 70
90 ;
100 INPUT 'ANGE SPÄNNINGEN I V (0-10) ?'Spänningen.
110 IF Spänningen.<0 OR Spänningen.>10 GOTO 100
120 !
130 Utdata=4095/10*Spänningen.
140 !
150 IF Kanal=1 OUT 0,Utdata,2,SWAP%(Utdata)
```

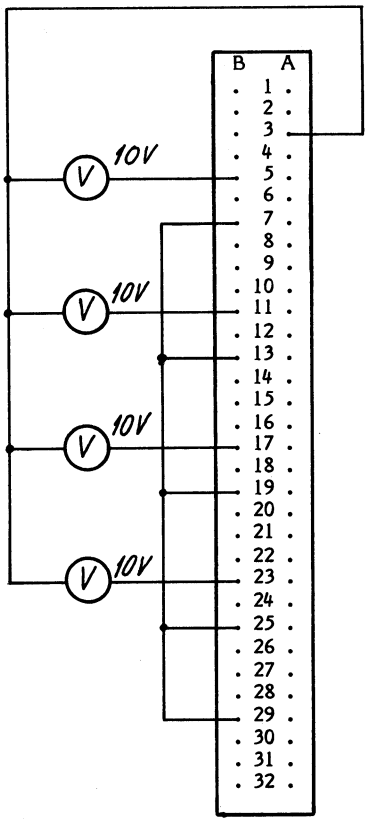


Fig. 4.4 Experimentuppkoppling för styrning av 4084 spänningsutgångar.

Styrning av 4084 från assembler.

Att styra 4084 från assembler är betydligt lättare eftersom styrningen till varje kanal sker med åtta bitar på 4084 mot tolv bitar på 4083. Principen för styrning av 4083 och 4084 är densamma så med små ändringar kan man göra om programmen. T.ex exempel 4 ser ut så här för 4084:

Objektkod	Mnemonics
123	LD A,E
211 0	OUT (0),A
201	RET

Genom att ändra på nollan på rad två så kan man enkelt byta kanal. En nolla betyder kanal 1, en två kanal 2, en trea kanal 3 och en fyra kanal 4.

Programmet för att generera en funktion ser ut så här:

Objektkod		Mnemonics		
6	50	START	LD	B,50
33	255 251		LD	HL,64511
35		LOOP	INC	HL
126			LD	A,(HL)
211	0		OUT	(0),A
16	250		DJNZ	LOOP
24	243		JR	START

Programmet blir mindre och snabbare än motsvarande för 4083 men upplösningen blir sämre.

Enklare funktioner kan man skriva direkt i assembler utan att behöva lägga upp en funktionstabell i minnet. Ett sådant program, som genererar en trekantvåg, kan återfinnas nedan.

Objektkod		Mnemonics			Rad
62	12	START	LD	A,12	1
211	1		OUT	(1),1	2
62	0		LD	A,0	3
211	0	RÄKNAUPP	OUT	(0),A	4
60			INC	A	5
254	255		CP	255	6
32	249		JR	NZ,RÄKNAUPP	7
211	0	RÄKNANER	OUT	(0),A	8
61			DEC	A	9
254	0		CP	0	10
32	249		JR	NZ,RÄKNANER	11
24	240		JR	RÄKNAUPP	12

Kommentarer till ovanstående program:

1 : Laddar register A med 12.
2 : Kortval, sänder ut innehållet i register A till port 1.
3 : Laddar register A med 0.
4 : Sänder ut innehållet i register A till port 0, (kanal 1).
5 : Ökar innehållet i register A med ett.
6 : Jämför innehållet i A med 255, om lika sätts ZERO-flaggan.
7 : Om inte ZERO-flaggan blev satt hoppar processorn till RÄKNANER.
8 : Om ZERO-flaggan blev satt, läggs innehållet i A ut på port 0.
9 : Minskar värdet i A med ett.
10 : Jämför innehållet i A med 0, om lika sätts ZERO-flaggan.
11 : Om inte ZERO-flaggan blev satt hoppar processorn till RÄKNANER.
12 : Om ZERO-flaggan blev satt börjar processorn om från början.

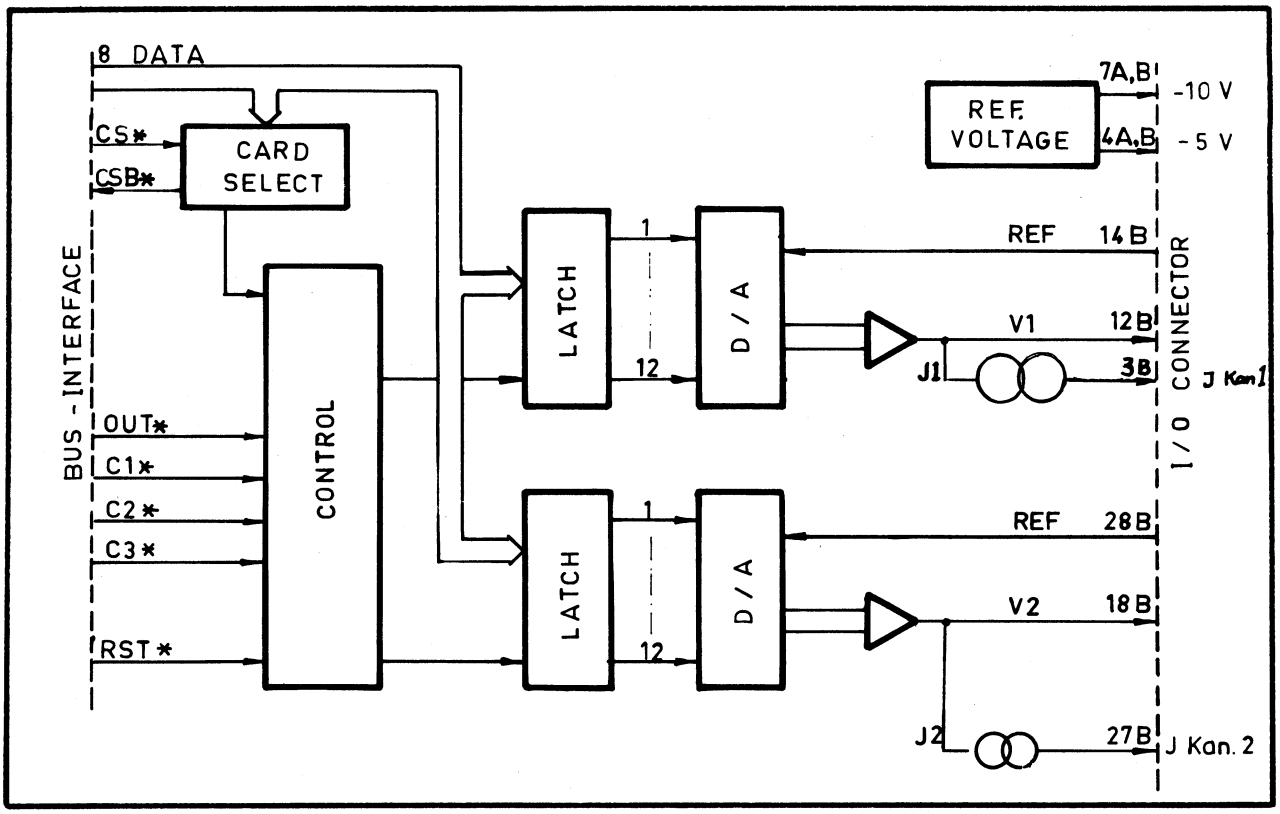


Fig. 4.6 Blockschema för 4083.

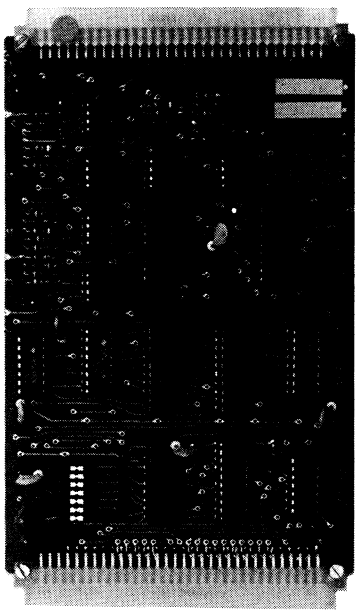


Fig. 4.5 D/A-omvandlare 4083.

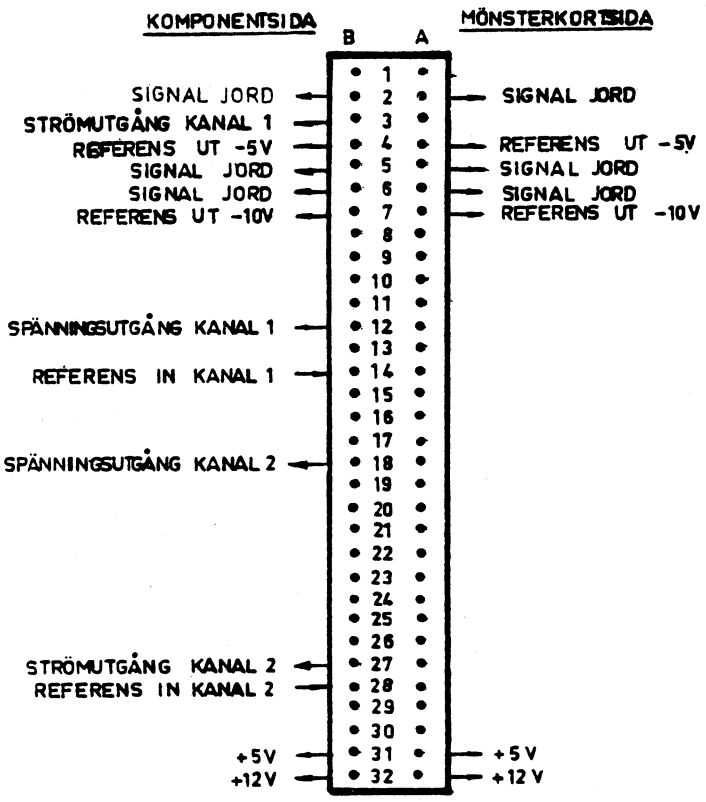


Fig. 4.7 Stiftlayout för 4083 I/O-kontakt.

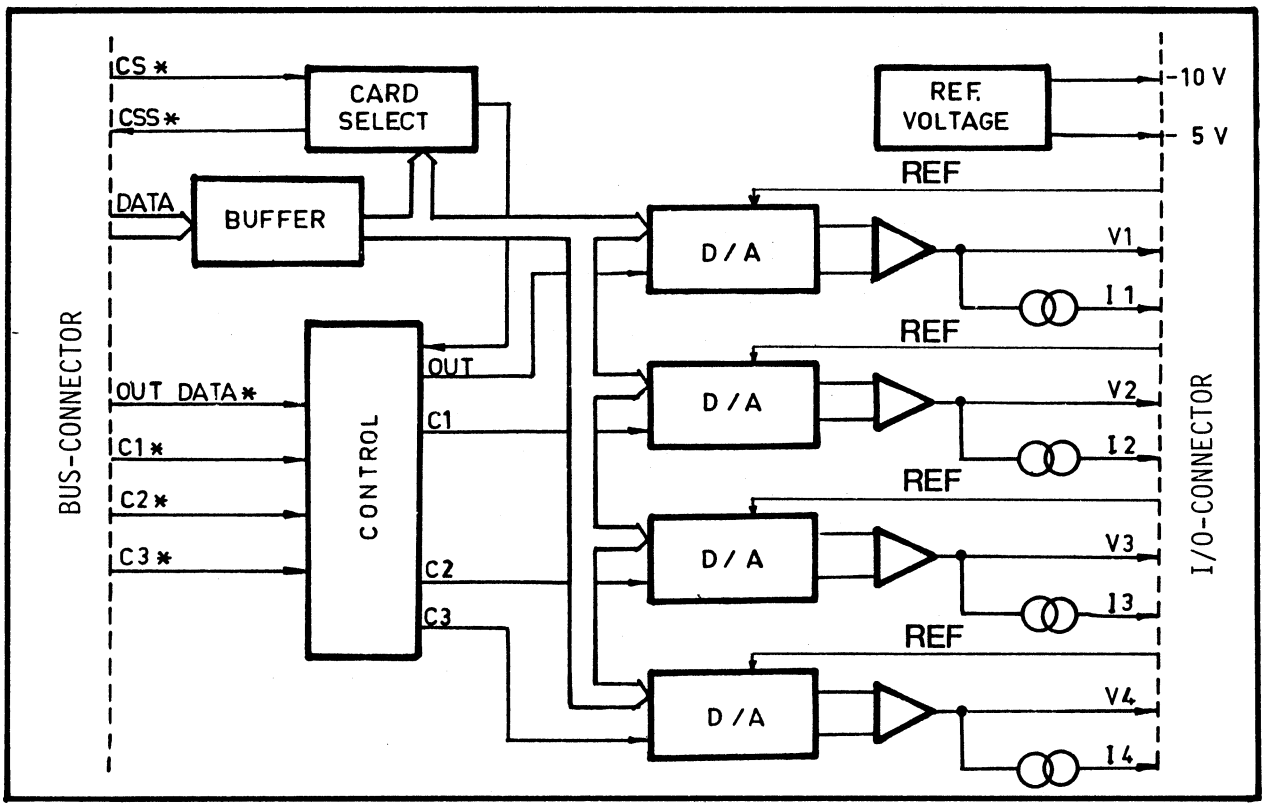


Fig. 4.9 Blockschema för 4084.

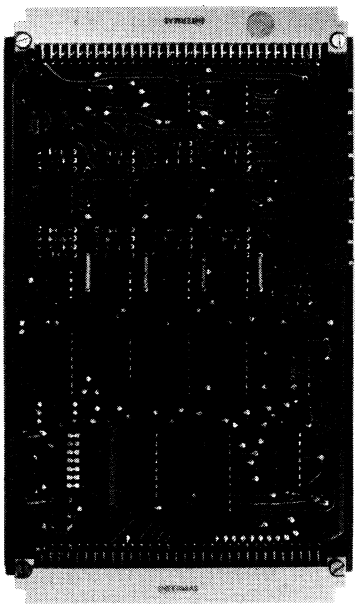


Fig. 4.8 D/A-omvandlare 4084.

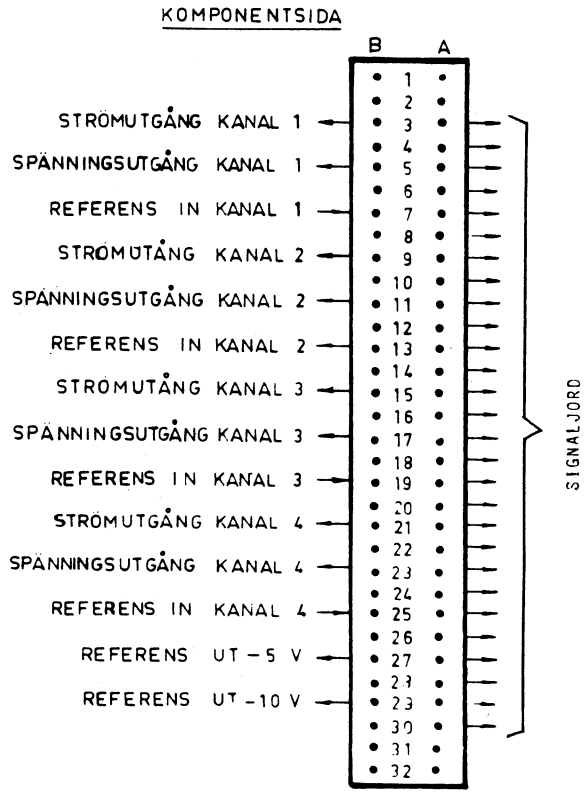


Fig. 4.10 Stiftlayout för 4084 I/O-kontakt.

4.2 Analog/digitalomvandlare 4115

4115 är en flexibel A/D-omvandlare, med 32 kanaler, som genom bygling kan kombineras till :

* 32 Enkla ingångar
* 16 Differentiella ingångar
* 16 Enkla ingångar och 8 differentiella ingångar
* 8 Differentiella ingångar och 16 enkla ingångar

Upplösningen för kortet kan varieras mellan 8 eller 12 bitar genom styrning från programvaran, liksom val av inspänningsområdet, (-5 till +5 V eller 0 till +10 V). Genom att använda en "sample-and-hold" krets med kort samplingstid minimerar man inverkan av eventuellt brus som skulle kunna störa A/D-omvandlingen. Tiden för omvandlingen är beroende på vilken upplösning som valts. Om man vill ha åtta bitars upplösning får man en omvandlingstid på ungefär 16 µS, dock max 27 µS. För tolv bitars upplösning blir motsvarande tider: normalt ca 25 µS och max 40 µS. När omvandlingen är klar sätts en status-flagga för att indikera detta. Stabiliseringstiden för multiplexern före samplingen är maximalt 30 µS. Den tiden är beroende på spänningsskillnaden mellan den nya och den gamla spänningen.

Användningsområdet för 4115 omfattar alla tillämpningar där analoga signaler hanteras. Med en lämplig givare kan de flesta fysikaliska storheter mätas, man kan t.ex använda den för mätning av temperatur med en termistor, eller mätning av ljus med en fotoresistor. Ansluter man en trådtöjningsgivare kan man mäta kraft, moment eller läge. Man kan även använda den som datalogger och transcientrecorder m.m.

INSTALLATION AV ADC-KORT 4115.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exempel nedan används adress 13 för 4115.

Anslutning av yttre enheter.

Anslutning av yttre enheter till kortet sker via I/O-kontakten. I/O-kontakten är den som sitter närmast lysdioden. Mellan digital jord och skyddsjord bör finnas ett filter som skydd mot statisk elektricitet. A/D-omvandlaren kan ta skada om spänningen försvinner på datorsidan medan signalsidan fortfarande har spänning. För att skydda A/D-omvandlaren mot detta så kopplas lämpligen ett 5 kohm motstånd i serie med varje ingång. Till analog jord skall ej något motstånd anslutas.

Bygling av ingångar.

Tal inom paranteser anger nummer för respektive kanaler.

1. 32 enkla ingångar. (0 - 31)

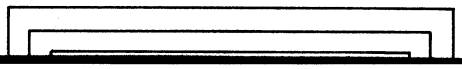
Bygla 20B till 25B
Bygla 19B till 25B

Ett par programexempel i ABC800/DTC basic:

Exempel 1: Läs ingång 1 och 3 från ADC-kort 4115

Programmet läser ingång 1 och 3 från A/D-kortet. Både differentiella och enkla ingångar går att läsa med programmet. För experimentuppkopplingar med enkla ingångar se fig. 4.11a och fig. 4.11b för differentiella ingångar.

```
10  ! LÄSNING AV INGÅNG 1 OCH 3 PÅ 4115
```



	32 kanaler enkla		16 kanaler diff.		8 kanaler diff. 16 kanaler enkla		16 kanaler enkla 8 kanaler diff.	
Ansl. nr.	Pin A Kanal	Pin B Kanal	Pin A Kanal	Pin B Kanal	Pin A Kanal	Pin B Kanal	Pin A Kanal	Pin B Kanal
1	0	8	0	0	0	0	0	8
2	1	9	1	1	1	1	1	9
3	2	10	2	2	2	2	2	10
4	3	11	3	3	3	3	3	11
5	4	12	4	4	4	4	4	12
6	5	13	5	5	5	5	5	13
7	6	14	6	6	6	6	6	14
8	7	15	7	7	7	7	7	15
9	16	24	16	16	16	24	16	16
10	17	25	17	17	17	25	17	17
11	18	26	18	18	18	26	18	18
12	19	27	19	19	19	27	19	19
13	20	28	20	20	20	28	20	20
14	21	29	21	21	21	29	21	21
15	22	30	22	22	22	30	22	22
16	23	31	23	23	23	31	23	23
17								
18								
19								
20								
21								
22								
23								
24								
25								

Kommentar: Pin 25A=25B=0 V analogt

4.3 Mätning av temperatur med PT100-kort 4021 och multiplexer 4022

4021

PT100 mätsystem mäter temperatur med Platina-resistanser.

4022

Multiplexer för anslutning av upp till 12 PT100 givare.

Temperaturmätning med PT100

De båda korten 4021 och 4022 används för att mäta temperatur med PT100 givare. 4021 omvandlar den analoga spänningen från givaren till en 12 bitars digital signal. Kortet kan användas ensamt och man får då en kanal för temperaturmätning med PT100 givare. Kopplas kortet parallellt med ett eller flera 4022 kort erhålles 12 kanaler för varje 4022 kort som kopplas till systemet. 4022 är ett multiplexer kort som genom ett kommando sänder en kanal till 4021 för omvandling.

PT100-kort 4021.

4021 levererar en växelström på ca 1 mA 200 Hz till PT100 givaren. Spänningen över givaren går in på en multiplicerande A/D-omvandlare. Skillnaden mellan denna och en intern referensspänning detekteras synkront och styr en upp/ner räknare till dess att PT100-signalen balanserats bort, se fig. 4.16. Vid balans är signalen in på detektorn noll, varvid inverkan av detektorvariationer minimeras. Omvandlingen är av "tracking" typ och följer hela tiden mätsignalen. Därav finns det inget "start conversion" kommando. Om man använder 4022 krävs dock en insvängningstid varje gång man byter insignal. Insvängningstiden är ungefär 2 sek, men om man förinställer kortet med det tidigare värdet kan man få ner insvängningstiden till ungefär 0.2 sek. Mätområdet för kortet ligger mellan -273 till +300 grader Celsius, vilket motsvarar 0 - 200 Ohm. Värdet in till datorn blir mellan 0 och 4095. Upplösningen blir då 1/4096 av det totala mätområdet, dvs ungefär 50 milliohm (bättre än 0.2 grader). Före användning måste kortet kalibreras, vilket sker i programvaran.

Multiplexer-kort 4022.

Genom multiplexern kopplas strömmen till en givare i taget när den kanalen väljs med ett kommando. Signalen från givarna förstärks på multiplexerkortet, varvid mätområdet bestäms av multiplexerkortet genom att en del av referensen dras ifrån innan mätsignalen går vidare till omvandlarkortet. Den del som bestämmer mätområdet sitter på en utbytbar modul. Som standard sitter en modul med området -50 till +150 grader celsius. Den modulen ger en upplösning på ungefär +/-0.05 grader celsius. Kortet är gjort så att man kan kalibrera det med programvara. Genom att montera in minst två kalibreringsmotstånd på kortet (platserna är märkta R2, R3, R4 och R5, kanal 12 - 15), som man vet den exakta resistansen hos, kan man sedan läsa av dessas resistans från kortet och beräkna den räta linjens ekvation. Med hjälp av denna ekvation kan man sedan beräkna resistansen på PT100 givarna med stor noggrannhet.

Installation av PT100-kort 4021.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exemplen nedan används adress 9 för 4021.

Anslutning av PT100 givare.

För anslutning av PT100 givare se fig. 4.16 och fig. 4.18. Anslut strömmatningen till stift 2 och till jord (0 V). Spänningsavkänningen ansluts till förstärkaren på stift 5, medan minussidan på avkänningen ansluts till jord (0 V). Dessutom skall förstärkarutgången (stift 6) byglas till D/A-omvandlarens ingång (stift 1).

Kalibrering.

För att kortet skall visa rätt måste det först kalibreras. Man börjar med att mäta upp två resistanser på en noggrann ohmmeter, (som i exemplet nedan blev R1=9.94 och R2=98.76). Därefter ansluter man ett motstånd i taget och läser av kortet, (som i detta fall gav U1=185 för R1 och U2=1847 för R2). Med hjälp av dessa fyra värden kan man bestämma konstanterna i ekvationen för en rät linje (Y=K*X+L). Det är den ekvationen som man sedan använder för att beräkna resistansen på PT100 givaren. Konstanterna erhålles ur:

K=(R2-R1)/(U2-U1)

L=((U2*R1)-(U1*R2))/(U2-U1)

Med ovanstående talvärden får vi:

K=(98.76-9.94)/(1847-185)=0.0534

L=((1847*9.94)-(185*98.76))/(1847-185)=0.0533

Den slutgiltiga ekvationen ser i detta fall ut på följande sätt:

(Resistansen)=0.0534*(Invärde)+0.0533

(Invärde) är det tal som läses från 4021. När man väl har konstanterna i ekvationen bestämda kan man koppla in sin PT100 givare.

Beräkning av temperaturen.

Beräkning av temperaturen sker efter formeln:

Temperaturen= A*(B+(C*Resistansen/(D-Resistansen)))

(Resistansen) erhålles från ekvationen ovan. Konstanterna i temperaturekvationen återfinnes nedan:

A = 0.997861 om "Resistansen" > 100 ohm
A = 1 om "Resistansen" < 100 ohm
B = -245.93
C = 6195.2
D = 2619.1

Installation av 4021 och 4022.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. I exemplen nedan används adress 9 för 4021 och adress 10 för 4022.

Anslutning av korten till varandra.

När man ansluter 4021 och 4022 skall fyra anslutningar göras mellan korten. Se fig. 4.18 och fig. 4.21.

Stift 1 "Signal in" ansluts till stift 1 på multiplexern
Stift 2 "Ström ut" ansluts till stift 2 på multiplexern
Stift 3 "Referens" ansluts till stift 3 på multiplexern
Jord (0 V) ansluts till jord (0 V) på multiplexern

Kalibrering av 4021 och 4022.

Kalibrering av 4021 och 4022 går i princip till likadant som kalibrering av 4021 ensam. Skillnaden ligger i att 4022 har fyra kanaler (kanal 12 - 15) där man kan ansluta kalibreringsmotstånd. Platserna för kalibreringsmotstånd är märkta R2, R3, R4 och R5. Man måste använda minst två motstånd. Sedan läser man av värdena på de använda kanalerna och räknar ut den räta linjens ekvation i programvara, (se under avsnitt "Kalibrering" ovan). I början på programmet tilldelar man variabler det exakta värdet på motstånden. Se exempel 2 nedan.

Anslutning av PT100 givarna.

Anslutning av PT100 givarna visas i fig. 4.20. Exempel: Anslutning av en PT100 givare till kanal 0. Fig. 4.20 stift 4A ansluts till 5B samt till ena benet på PT100 givaren, medan stift 4B ansluts till 5A samt till andra benet på PT100 givaren.

4021 kommandon.

Basic	**Aktiverad signal**	**Funktion**
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP(7) bör inleda alla program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: 20 OUT 1,9 väljer ut kortet med adressen 9.
INP (0)	INP	Läser de åtta minst signifikanta bitarna av det omvandlade värdet.
INP (1)	STAT	Läser de fyra mest signifikanta bitarna av det omvandlade värdet. Exempel: 20 Invärde=SWAP%(INP(1) AND 15)+INP(0) läser in värdet från PT100 givaren till variabeln "Invärde".
OUT 0,<Utvärde>	OUT	Förställer räknarens åtta minst signifikanta bitar.

OUT 2,<Utvärde>	C1	Förställer räknarens fyra mest signifikanta bitar. Exempel: 30 OUT "0,2048,2,SWAP%(2048) förställer räknaren med 2048.

Kommandon för 4022

Basic	Signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP(7) bör inleda alla program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: 20 OUT 1,10 väljer ut kortet med adressen 9.
OUT 0,<Kanal>	OUT	Väljer kanaladress som skall sändas över till 4021. D0 D1 Adress till PT100 givarna 0-11 D2 eller 12-15 för kalibrering. D3 D4 Sänder värdet från vald adress. D5 Används ej D6 Används ej D7 Används ej Exempel: 40 OUT 0,23 väljer kanal 7 och sänder värdet till 4021.
OUT 0,0	OUT	Nollställer alla kanaler. Skall stå före val av ny kanal.

Exempel 1: 4021 med en PT100 givare.

Programmet är ett litet exempel på hur man kan använda 4021 med en PT100 givare. Uppkoppling sker enligt installationsanvisningarna för 4021. Programmet mäter temperaturen i en evighetsslinga.

```
10  ! TEMPERATURMÄTNING MED KORT 4021 & PT100 GIVARE
20  !
30  EXTEND : INTEGER : SINGLE : DIGITS 3
40  OUT 1,9 ! KORTVAL
50  Slask=INP(7)
60  GOSUB 190 ! UTSKRIFT AV TEXT
70  !
80  ! LÄSNING AV GIVAREN
90  !
100 Invärde=(INP(1) AND 15)*256+INP(0)
110 !
120 Resistans.=.0534*Invärde+.0533 ! RÄKNAR UT
    RESISTANSEN PÅ PT100-GIVAREN
130 !
140 Temperatur.=.997861*(-245.93+((6195.2*Resistans.)/
    (2619.1-Resistans.)))
150 !
160 ; CUR(12,43) Temperatur.
170 GOTO 80
180 !
```

```
190 ! UTSKRIFT AV TEMPERATUREN
200 !
210 ; CHR$(12)
220 ; CUR(12,10) "TEMPERATUREN ÄR FÖR NÄRVARANDE :'CELSIUS"
230 RETURN
```

Kommentarer till ovanstående program:

100 : Läser värdet på PT100 givaren.
120 : Räknar om värdet till resistans. Formeln fås genom kalibreringen. Värdena på denna rad stämmer förmodligen inte.
140 : Omvandlar resistansen till temperatur.

Exempel 2: 4021 och 4022 sammankopplade.

Detta program mäter temperaturen på 12 ställen. Uppkoppling enligt installationsanvisningarna för 4021 och 4022 ovan. Programmet är självkalibrerarande om man anger värdena på resistorerna som sitter på R2 och R3. Resistorn som sitter på R2 skall ge ett talvärde till "Resistans.(12)" och R3 till "Resistans.(13)".

```
10  ! MÄTNING AV TEMPERATUR MED 4021 OCH 4022
20  !
30  EXTEND : INTEGER : SINGLE : DIGITS 3
40  Slask=INP(7)
50  DIM Resistans.(14),Temperatur.(14),Invärde(14)
60  ; CHR$(12)
70  ; CUR(3,5) ' TEMPERATURMÄTNING MED PT100, 4021 OCH 4022 '
80  !
90  ! SÄTTER VÄRDENA PÅ KALIBRERINGS-RESISTORERNA
100 !
110 Resistans.(12)=148.93
120 Resistans.(13)=98.78
130 !
140 ! LÄSER AV KALIBRERINGS-RESISTORERNA FÖR BERÄKNING
    AV DEN RÄTA LINJEN
150 !
160 FOR Kalloop=12 TO 13
170   OUT 1,10 ! VAL AV KORT 4022 (MULTIPLEXERN)
180   OUT 0,0
190   OUT 0,16+Kalloop ! LÄGGER UT EN KANAL'S VÄRDE TILL 4021
200   !
210   OUT 1,9 ! VAL AV KORT 4021 (KONVERTERAREN)
220   FOR Vänta.=0. TO 2000. : NEXT Vänta. ! VÄRDET STABILISERAR SIG
230 Invärde(Kalloop)=SWAP%(INP(1) AND 15)+INP(0)
240 NEXT Kalloop
250 !
260 ! RÄKNAR UT EKVATIONEN FÖR DEN RÄTA LINJEN
270 !
280 K.=(Resistans.(12)-Resistans.(13))/(Invärde(12)-Invärde(13))
290 L.=Resistans.(13)*Invärde(12)-Resistans.(12)*Invärde(13)
300 L.=L./(Invärde(12)-Invärde(13))
310 !
320 ! LÄSER ALLA KANALER
330 !
340 FOR Loop=0 TO 11
350   ; CUR(Loop+5,18) '=>'
360   OUT 1,10 ! VAL AV MULTIPLEXERKORTET
370   OUT 0,0
380   OUT 0,16+Loop ! LÄGGER UT EN KANAL TILL OMVANDLINGSKORTET
390  !
```

```
400  OUT 1,9 ! VAL AV KONVERTERINGSKORTET
410  OUT 0,Invärde(Loop),2,SWAP%(Invärde(Loop)) ! TIDIGARE
     VÄRDE TILL RÄKNAREN
420  FOR Vänta.=0. TO 2000. : NEXT Vänta.
430  Invärde(Loop)=SWAP%(INP(1) AND 15)+INP(0)
440  !
450  Resistans.(Loop)=K.*Invärde(Loop)+L.
460  Temperatur.(Loop)=.997861*(-245.93+(6195.2*Resistans.(Loop)/
     (-2619.1-Resistans.(Loop))))
470  ; CUR(Loop+5,18) 'KANAL ' Loop,Temperatur.(Loop) '" CELSIUS"
480 NEXT Loop
490 GOTO 320 ! LÄSER ALLA KANALER EN GÅNG TILL
```

Kommentarer till ovanstående program:

110 : Sätter ett av värdena för kalibrationsresistorerna.
120 : Sätter ett av värdena för kalibrationsresistorerna.
160-240 : Läser av värdet på kalibreringsresistorerna.
260-300 : Räknar ut ekvationen för den räta linjen.
340-480 : Läser av alla kanaler och omvandlar dem till grader.

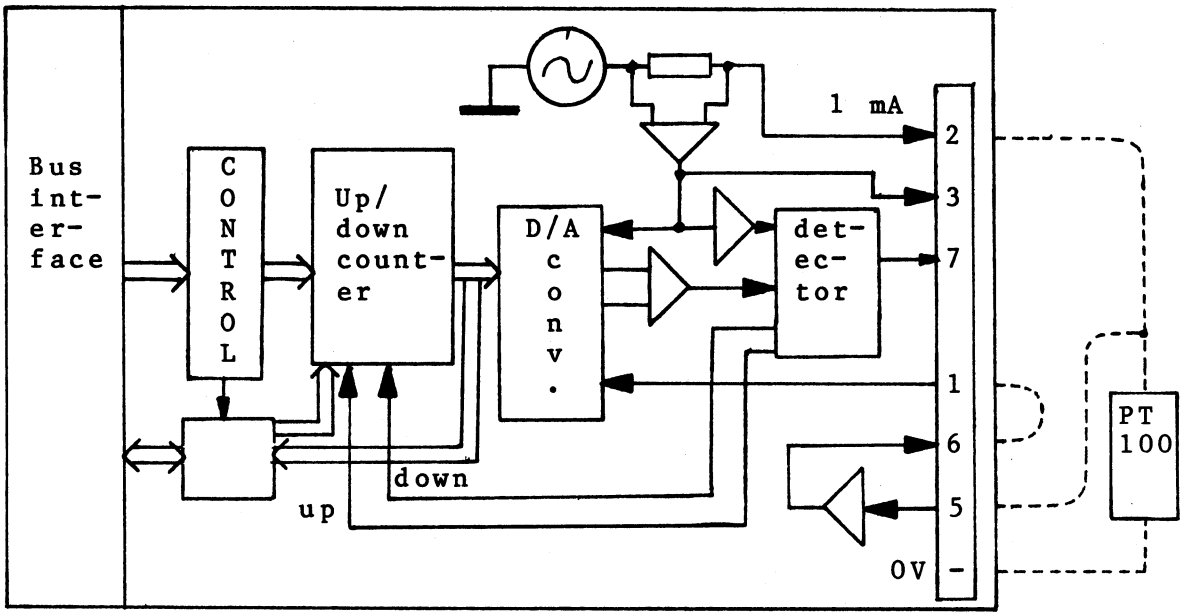


Fig. 4.16 Blockschema för PT100-kort 4021.

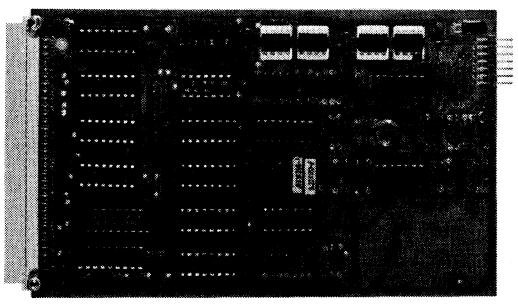
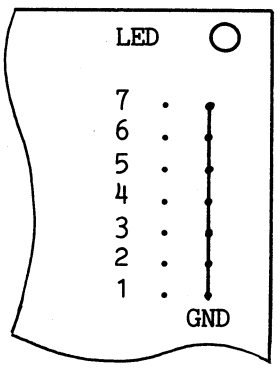


Fig. 4.17 PT100-kort 4021.



I/O Kontaktpinnar

UT Test-utgång
UT Från förstärkare
IN Till förstärkare
NC Ingen anslutning
UT Referensspänning
UT Current drive
IN Signalspänning

Fig. 4.18 4021 I/O-kontakt till PT-100.

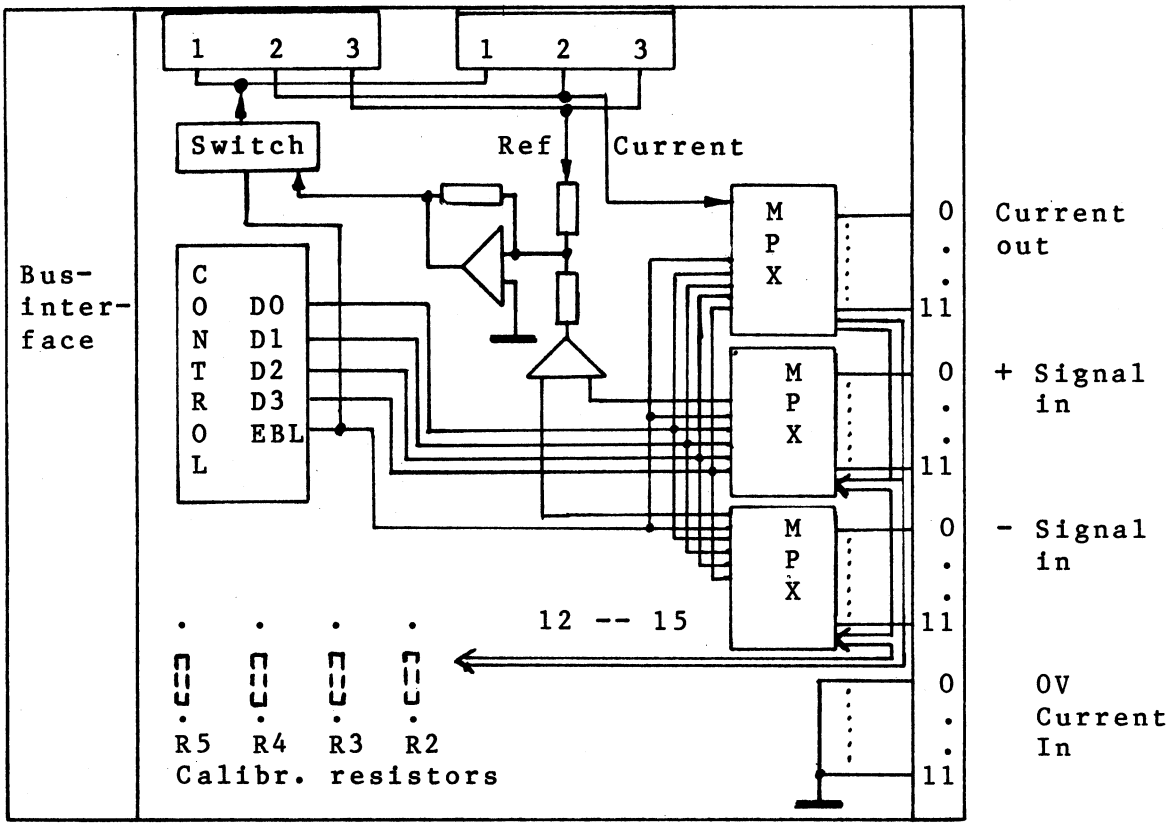


Fig. 4.19 Blockschema för multiplexerkort 4022.

		B	A	
	NC	.	1 .	NC
	NC	.	2 .	NC
	NC	.	3 .	NC
KANAL 0	Ström in (-)	.	4 .	Ström ut (+)
	Signal (+)	.	5 .	Signal (-)
KANAL 1	Ström in (-)	.	6 .	Ström ut (+)
	Signal (+)	.	7 .	Signal (-)
KANAL 2	Ström in (-)	.	8 .	Ström ut (+)
	Signal (+)	.	9 .	Signal (-)
KANAL 3	Ström in (-)	.	10 .	Ström ut (+)
	Signal (+)	.	11 .	Signal (-)
KANAL 4	Ström in (-)	.	12 .	Ström ut (+)
	Signal (+)	.	13 .	Signal (-)
KANAL 5	Ström in (-)	.	14 .	Ström ut (+)
	Signal (+)	.	15 .	Signal (-)
KANAL 6	Ström in (-)	.	16 .	Ström ut (+)
	Signal (+)	.	17 .	Signal (-)
KANAL 7	Ström in (-)	.	18 .	Ström ut (+)
	Signal (+)	.	19 .	Signal (-)
KANAL 8	Ström in (-)	.	20 .	Ström ut (+)
	Signal (+)	.	21 .	Signal (-)
KANAL 9	Ström in (-)	.	22 .	Ström ut (+)
	Signal (+)	.	23 .	Signal (-)
KANAL 10	Ström in (-)	.	24 .	Ström ut (+)
	Signal (+)	.	25 .	Signal (-)
KANAL 11	Ström in (-)	.	26 .	Ström ut (+)
	Signal (+)	.	27 .	Signal (-)
	NC	.	28 .	NC
	NC	.	29 .	NC
	NC	.	30 .	NC
	NC	.	31 .	NC
	NC	.	32 .	NC

Fig. 4.20 4022 I/O-kontakt till PT-100 givare.

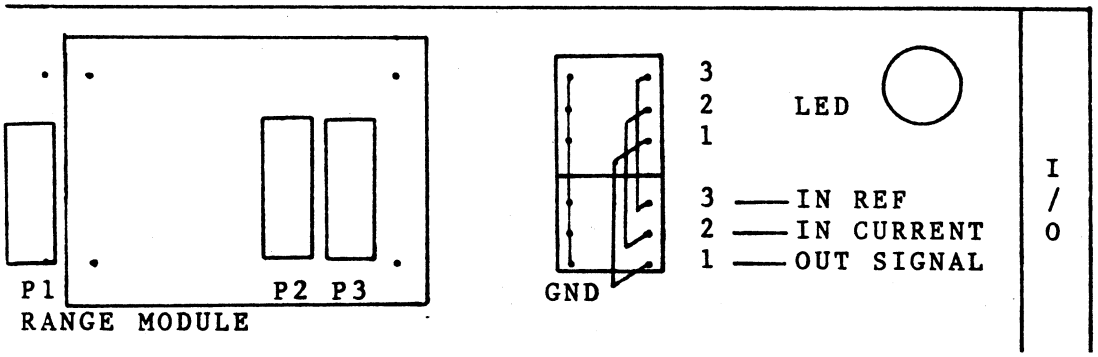


Fig. 4.21 4022 I/O-kontakt till 4021 PT100-kort.

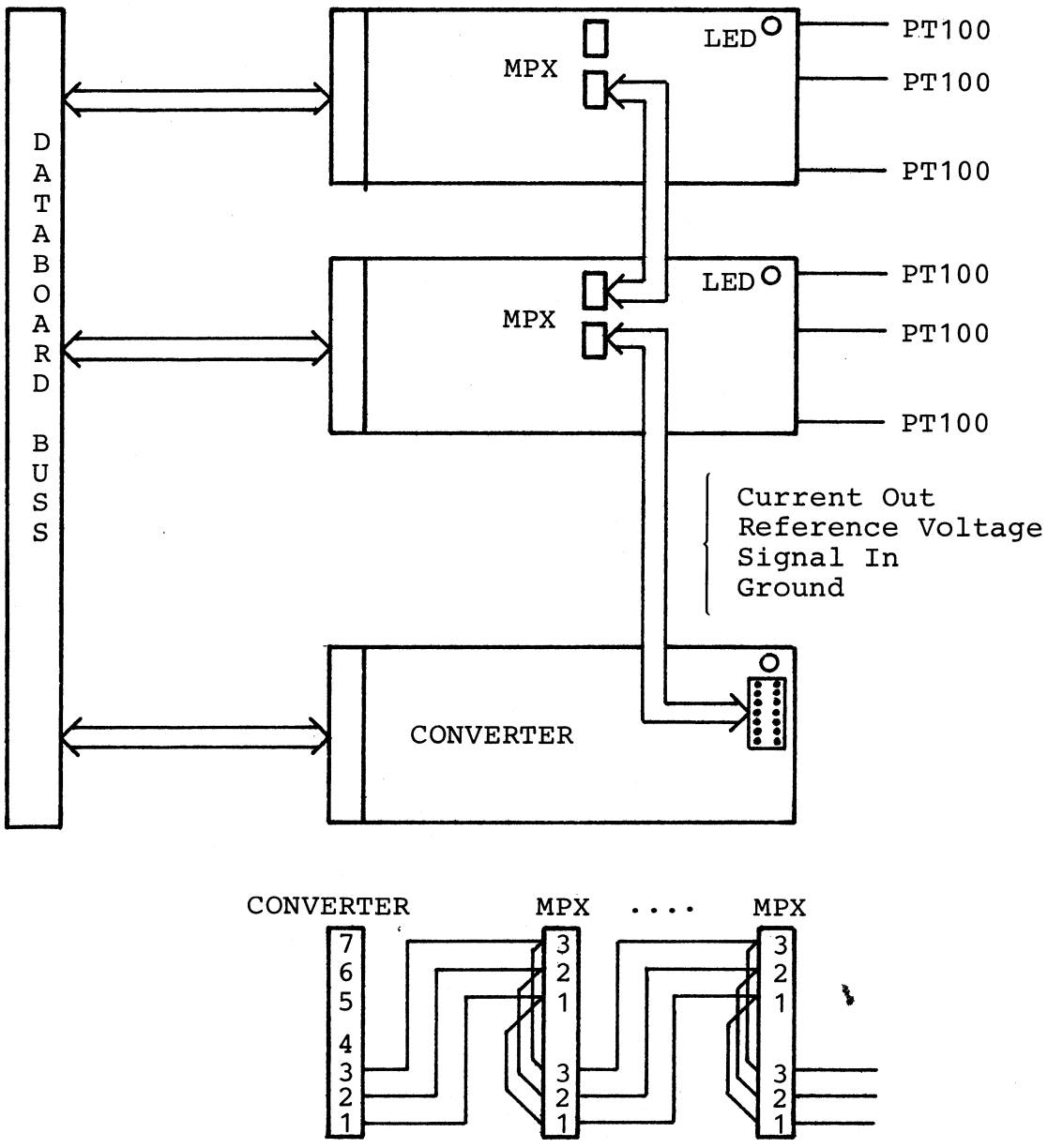


fig. 4.22 Uppkoppling med 4021 och en eller flera 4022.

5. Minnesexpansion på ABC80/800/DTC

2004
8/16/32k ByteWyde RAM/EPROM minneskort

3061
8/16/32k EPROM minneskort

3032
8k EPROM minneskort

2056
16k RAM minneskort

Minnet på ABC80 och ABC800/DTC kan byggas med ett antal olika minneskort. ABC800/DTC kan dock bara byggas ut med minneskortet 2004, som är det enda PROM-kort som använder "XM*"-signalen som kopplar bort det interna minnet när PROM-kortet blir adresserat. PROM-korten som används med ABC800/DTC skall placeras över RAM-arean. Detta för att ABC800/DTC är redan fullt utbyggd, (ABC800 C har 1 kB ledigt utrymme under bildminnet, se minneskartan för ABC800/DTC) och korten är konstruerade så att när en adress läggs ut, som finns på PROM-kortet, kopplas det interna minnet bort. Skulle då PROM-kortet överlappa en väsentlig minnesarea, t.ex Basictolken skulle datorn aldrig komma åt denna. Om man lägger kortet över RAM-minnet händer inget annat än att det tillgängliga RAM-minnet minskar. Detta medför att man även i ABC800/DTC kan göra egna assemblerrutiner som man kan lägga i PROM-kapslar och använda som på ABC80. Installationen av de olika PROM-korten skiljer sig inte mycket åt. Här nedan beskrivs hur korten 2004, 3032, 3061 och 2056 installeras. De generella byglingsförfarandet när det gäller basadressen för kortet kan återfinnas i kapitel 1: "Adressering av I/O- och minnes-kort".

Bygling av minneskort 2004. (För ABC80/800/DTC)

2004 har två oberoende sidor som kan fyllas med ROM- eller RAM-kapslar av storleken 1k, 2k eller 4k. De EPROM- och RAM-kapslar som kan väljas är:

	EPROM	RAM
1k	2758	MK4118, MK4801
2k	2716	MK4802, TMN2016, TMM5117
4k	2732	

eller motsvarande. Notera att till detta kort användes s.k. "ByteWyde" statiska RAM, dvs en adress i en minneskapsel ger en byte data, (ett ROM är alltid "ByteWyde").

Vardera sidan rymmer maximalt 16 kbyte. Båda sidorna adresseras oberoende av varandra. Adresseringspluggens fyra första byglingar gäller C-raden på kortet och de sista fyra gäller A-raden på kortet.

	rad C				rad A			
	1	2	3	4	5	6	7	8
1k	32k	16k	8k	4k	32k	16k	8k	4k
2k	32k	16k	--	8k	32k	16k	--	8k
4k	32k	16k	--	--	32k	16k	--	--

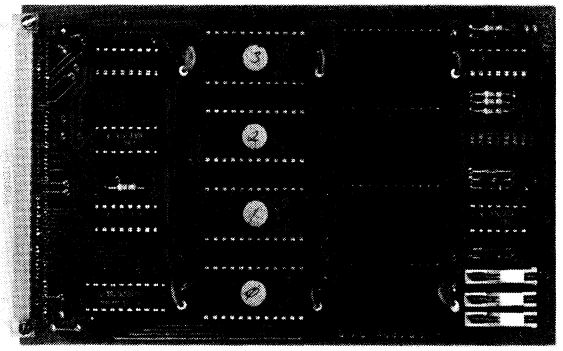


Fig. 5.2 Minneskort 3061

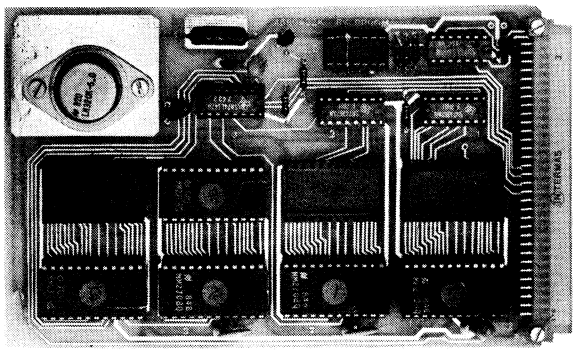


Fig. 5.3 Minneskort 3032

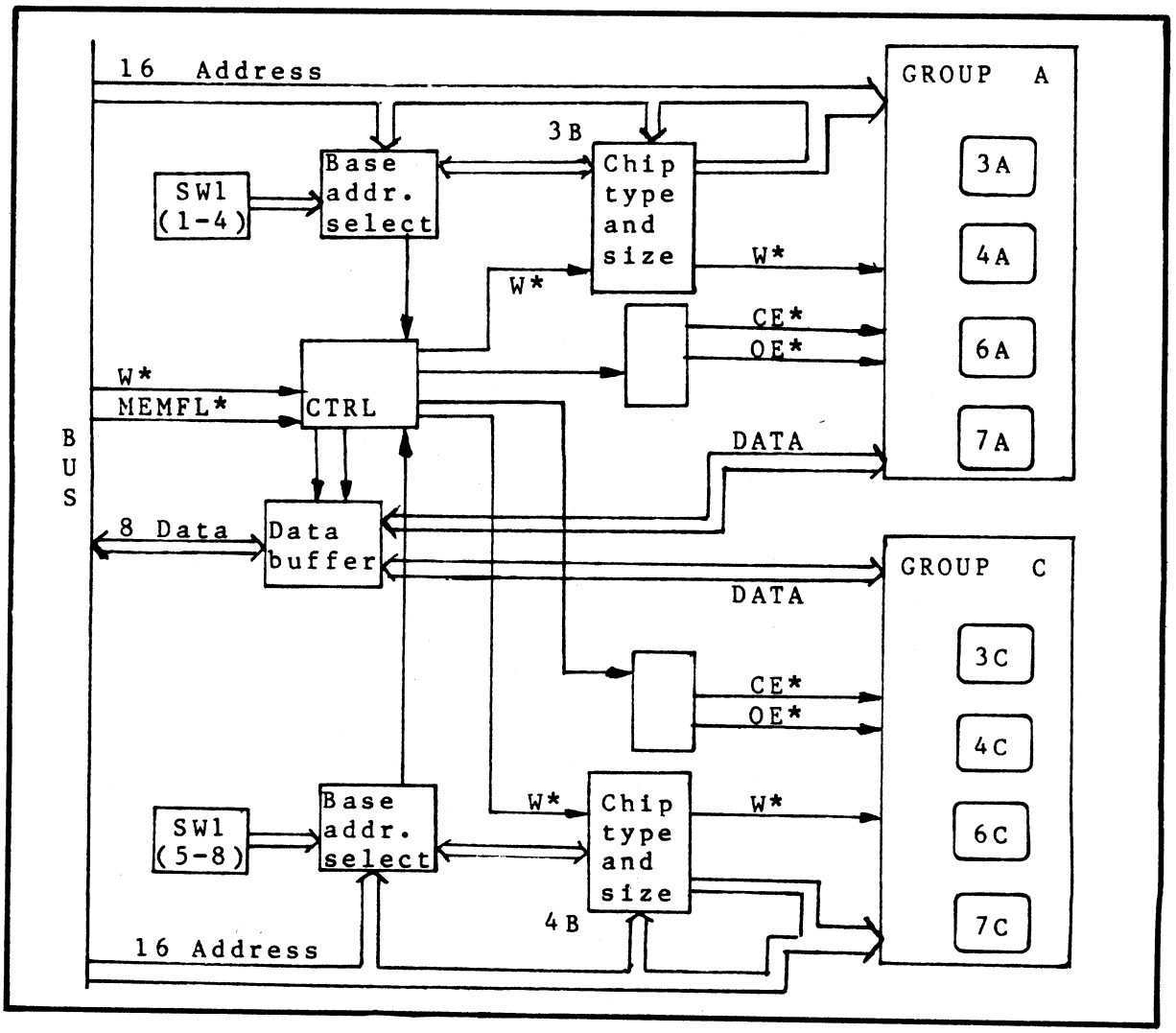


Fig. 5.4 Blockschema för minneskort 2004

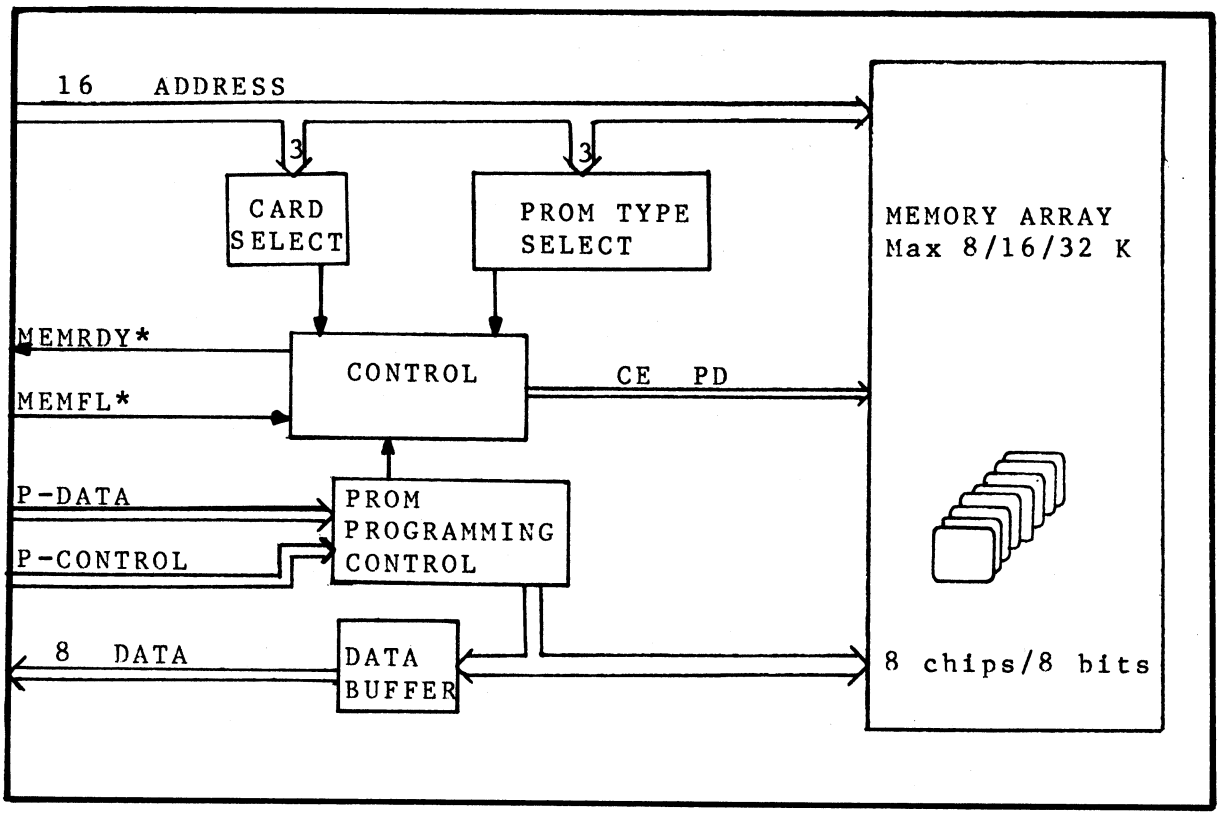


Fig. 5.5 Blockschema för minneskort 3061

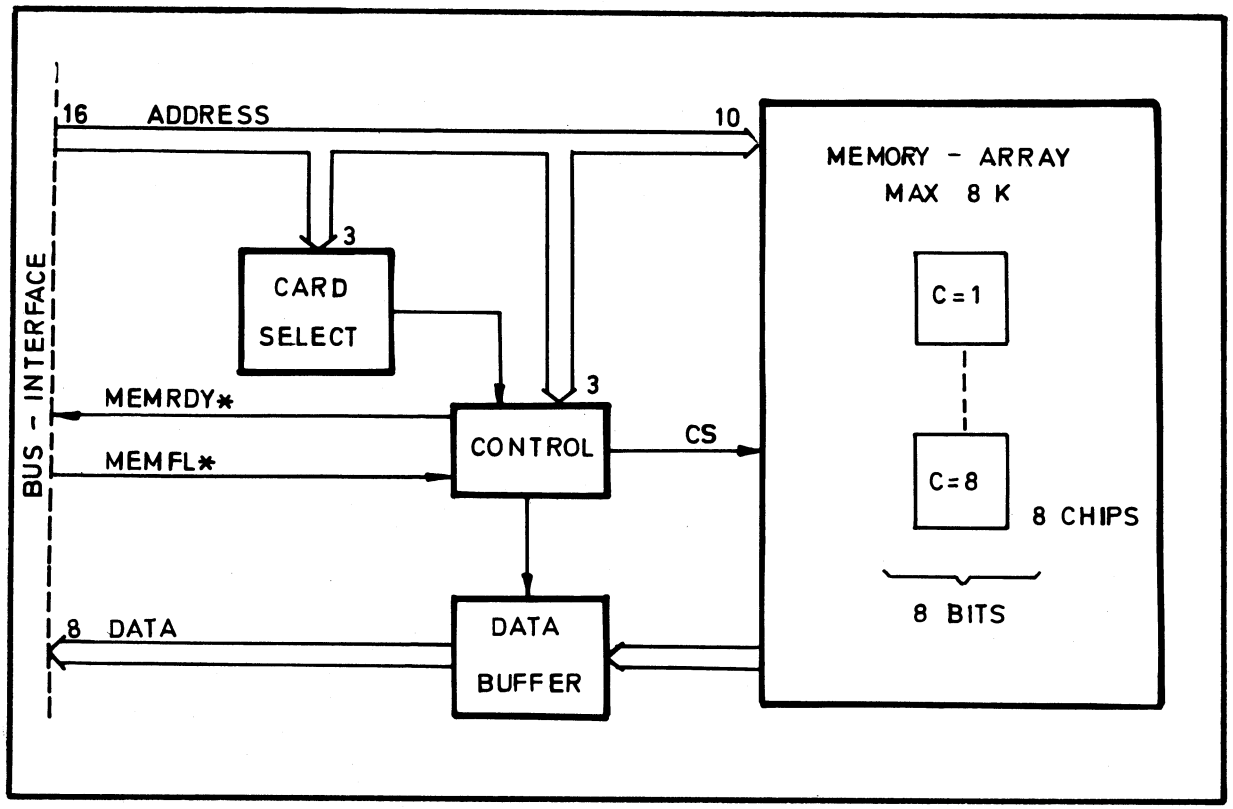


Fig. 5.6 Blockschema för minneskort 3032

Expansion av RAM-minnet hos ABC80.

Till ABC80 finns det ett antal olika kort för expansion av RAM-minnet. Av dessa kan nämnas t.ex. 2056, ett 16 kbyte RAM-kort.

2056 16k RAM-minne

2056 är bestyckat med 16kB dynamiskt RAM vid leveransen. Det enda som behöver utföras före montering är val av basadressen för kortet. Det sker med hjälp av två omkopplare nere i högra hörnet på kortet. Deras inställning skall vara:

omkopplare 14	omkopplare 15	minnesarea
1	1	0 - 16k
0	1	16 - 32k
1	0	32 - 48k
0	0	48 - 64k

När detta är gjort kan kortet placeras i **minnesdelen** på expansionslådan.

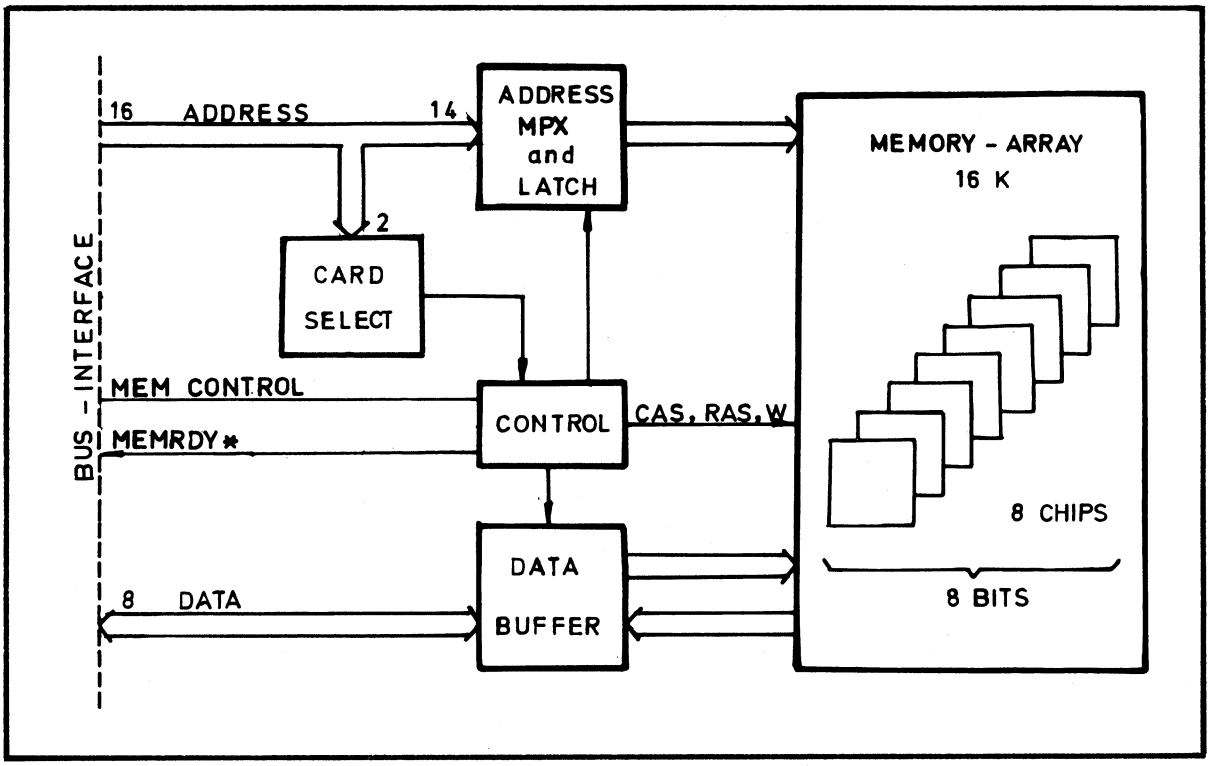


Fig. 5.7 Blockschema för minneskort 2056

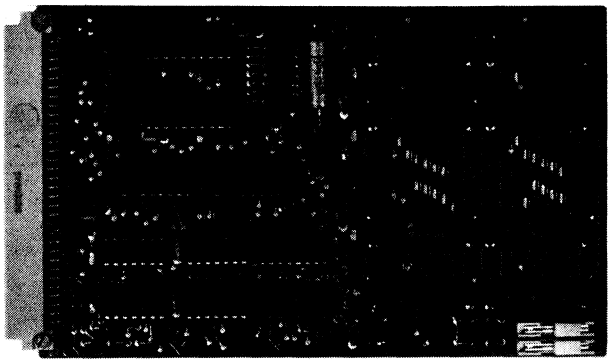


Fig. 5.8 Minneskort 2056

6. IEC-bussen

IEC-bussen är en internationell standardardiserad, (fastställd 1975), buss för att styra och läsa mätinstrument. GPIB och IEEE 488 är två andra bussar, som i stort överensstämmer med IEC-bussen. Skillnaden ligger i anslutningsdonens utformning. IEC-bussen har fått ett snabbt erkännande bland instrumenttillverkarna vilket har resulterat i att det i dag finns ett stort utbud med IEC-kompatibla instrument. Bland instrumenttillverkare som erbjuder sådana finner vi bl.a. Philips, Hewlett-Packard, Fluke och Rodhe & Schwartz. Bland de fördelar en sådan internationell standardiserad buss för med sig kan nämnas:

Man är inte beroende av någon tillverkare utan kan välja den kvalite och prisklass som passar bäst.

Man kan enkelt byta/utöka systemet med andra instrument eftersom de passar in på bussen utan några interface.

Små resurser krävs för att hålla systemet igång.

På IEC-bussen kan man ansluta upp till 15 instrument med en total kabellängd av 20 m. Kontakterna på anslutningskabeln har både en hane och en hona så att bussen går att förlänga från varje instrument. Genom detta undviker man en stor anhopning kablar på anslutningen till datorn. Skillnaden mellan IEC-bussen och IEEE-bussen sitter i kontakten. IEC-bussen använder sig av den 25 poliga "Canon D" kontakten, medan IEEE 488 bussen använder sig av den 24 poliga "Micro Ribbon" kontakten. Stiftplaceringen i dessa två kontakter kan ses i fig. 6.1. De flesta instrumenttillverkare väljer den 24 poliga "Micro Ribbon" kontakten. Båda dessa kontakter är "överpoliga" eftersom av 24/25 poler använder IEC-bussen endast 16 ledningar. De övriga är kopplade till jord. Fördelningen av de 16 ledarna är att 8 används till dataöverföring, 3 ledningar för handskakning samt 5 ledningar för själva busshanteringen.

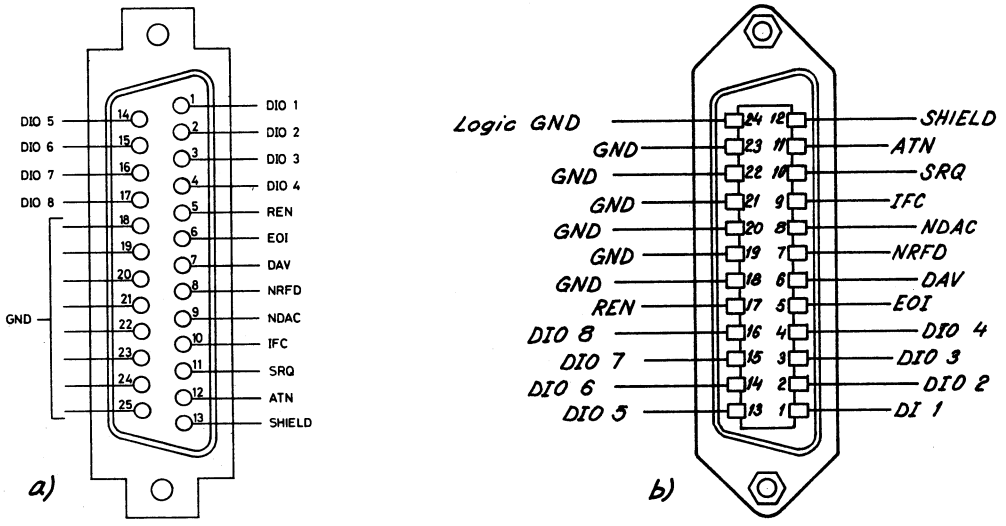


Fig. 6.1 Stiftlayout för (a) IEC-kontakten (b) IEEE-kontakten.

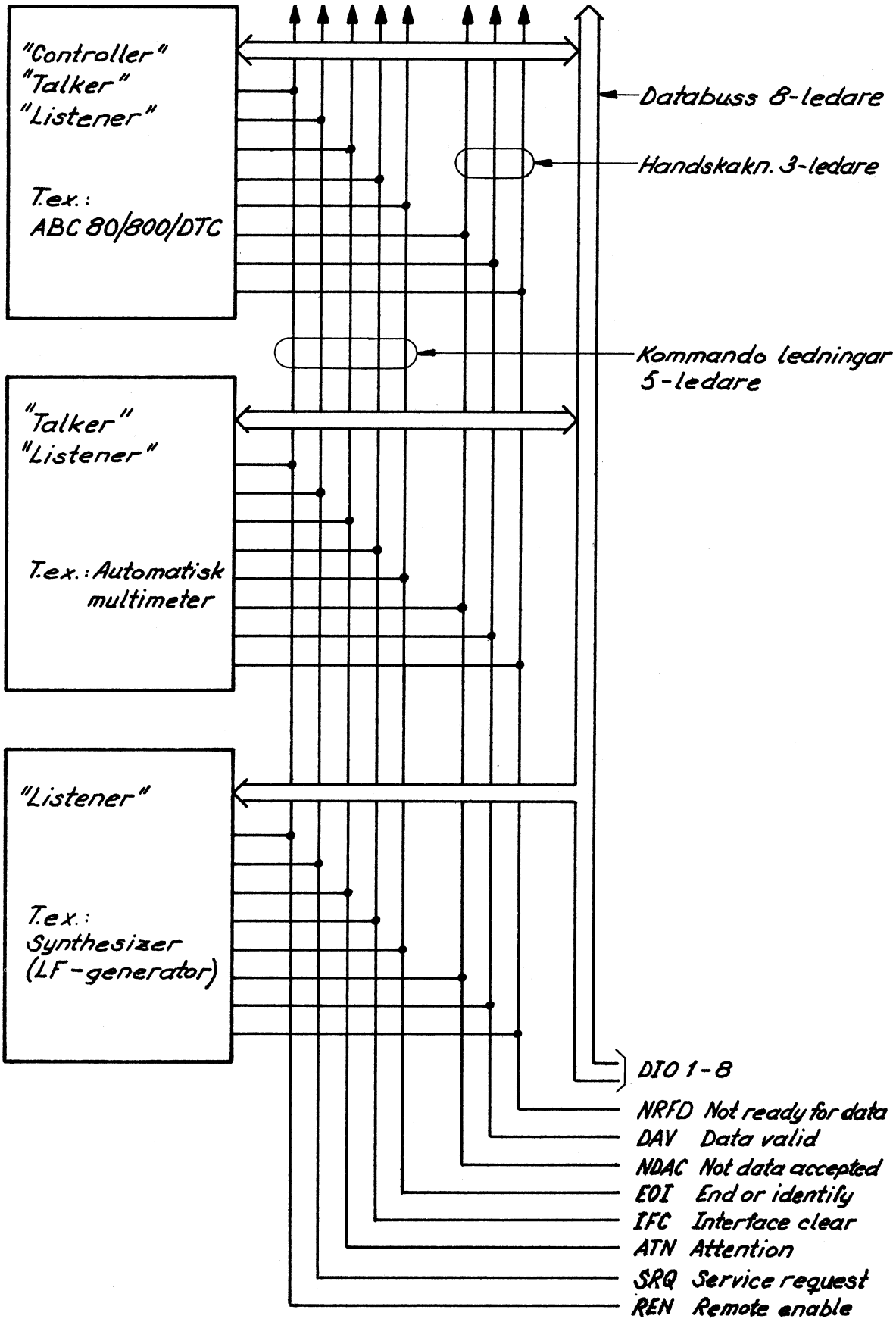


Fig. 6.2 IEC-bussens struktur.

Varje system består av ett antal instrument som har **Listener** funktion och/eller **Talker** funktion samt en enhet som fungerar som **Controller, Talker** och **Listener.**

En **Talker** fungerar som sändare av data.

En **Listener** fungerar som mottagare av data och kommandon.

En **Controller** fungerar som samordnande organ. Det innebär att den styr hela förloppet genom IEC-bussen.

För att det hela skall fungera måste givetvis varje enhet ha en egen adress. Ett instrument med både "Talker" och "Listener" har två olika adresser. Finns bara en av dessa två funktioner finns också bara en adress. Val av adress sker med en omkopplare på baksidan av instrumentet. Man ställer bara in de 5 minst signifikanta bitarna i den 7 bitar långa adressen, fig. 6.9. De två högsta bitarna anger om det är en "Talker"-adress eller en "Listener"-adress. Det är "Controllerns" uppgift att tala om vem som skall vara "Talker" och vilken/vilka som skall vara "Listeners". I systemet kan bara en "Talker" vara verksam åt gången medan flera listeners kan ta emot den information som sänds.

IEC-Bussen på ABC80/800/DTC.

För att få tillgång till IEC-bussen på ABC80/800/DTC måste man ansluta en 4025 IEC-bussanpassning till 4680-bussen. Genom detta kort får ABC80/800/DTC fasta adresser nämligen :

Talker adress: U
Listener adress: 5

Man behöver också en kabel från 4025 IEC-bussanpassningen till första instrumentet, en IEC-kabel 4680-IEC eller en 4680-IEEE. Man behöver även en drivrutin till kortet. Vill man sedan utöka systemet krävs det bara en ytterligare kabel mellan instrumenten. För att utöka systemet behövs således inte någon ytterligare modifiering av datorn.

Installation av 4025.

Val av adress.

Se kapitel 1 för val av adress hos kort som skall monteras i expansionsenheten. Kortadressen för IEC-kortet skall alltid vara 49.

Montering av 4025 i expansionsenheten.

Slå först av spänningen! 4025-kortet vänds så att I/O-kontakten kommer utåt. Komponentsidan skall vara vänd åt höger så att lysdioden hammnar nere till höger. 4025 skall placeras i I/O-delen på expansionsenheten.

Kontroll av adress.

Efter montering av kortet i expansionsenheten bör man kontrollera att kortet byglats till rätt adress genom att skriva:

```
OUT 1,49.
```

Är kortet rätt byglat skall lysdioden nere till höger på kortet tändas. Om därefter annat kort väljes, med t.ex kommando OUT 1,0, skall lysdioden slockna.

Uppstartning.

Anslut IEC-kabeln till kontakten på kortet. Ladda sedan drivrutinen. (På ABC80 finns drivrutinen på PROM, på ABC800/DTC laddas drivrutinen från en flexskiva). Sedan är systemet klart för körning.

Programmering av IEC-bussen.

Programmeringen av IEC-bussen är beroende på vilka instrument man har. Därför måste man studera manualen för varje instrument så att man vet hur det fungerar och kan programmeras.

När man installerat kortet, drivrutinen och kopplat in de instrument man skall använda är dags att börja programmera IEC-bussen. Man måste börja med att initiera drivrutinen och reservera en buffert i datorn. Detta gör man genom att skriva:

```
10 OPEN "IEC:" AS FILE X
```

Där X är filnumret (mellan 0 och 255). I och med att man initierar drivrutinen sätter man datorns "Talker" adress till U och "Listener" adressen till 5. När programmet är färdig skall man stänga filen efter sig. Detta gör man med:

```
5000 CLOSE X
```

Där "X" är samma filnummer som när man öppnade filen. Kommandon och data överförs till instrumenten med kommandot CMD. Kommandot skall åtföljas av parametrar som innehåller vem som skall tala och vem/vilka som skall lyssna samt ett "?"-tecken för att nollställa alla gamla "Listener". Ett exempel:

```
100 CMD "?U1","F125.5A10.0D.5W2";CHR$(3)
```

Raden börjar med att nollställa alla "Listeners" med parametern "?". Sedan väljs ABC80/800/DTC till "Talker" med parametern "U". Med "1" väljs instrumentet som har "Listener"-adress 1. Parameterraden efter kommat är data till instumentet med "Listener"-adress 1, i detta fall PHILIPS synthesizer PM 5190. Raden betyder : sätt frekvensen 125.5 kHz med topp-topp spänningen 10.0 V och en offset-spänning på 0.5 V DC samt fyrkantsvåg. CHR$(3) som står sist indikerar bara slut på kommandotexten. Det är parameterraden efter kommat som är speciell för varje instrument. När datorn fungerar som "Listener" läser man in informationen med kommandot IEC$. En rad som läser in 18 tecken från IEC-bussen kan se ut så här:

```
120 ; IEC$(18)
```

eller

```
120 In$=IEC$(18)
```

Men för man skall kunna läsa in något med IEC$ måste man först definiera en "Talker" och sedan datorn till "Listener". Så en komplett inläsning ser ut på följande sätt:

```
120 CMD "?V5"
130 In$=IEC$(18)
```

Det antal tecken man läser in är givetvis beroende på vilket instrument vi använder oss av.

Philips IEC-buss instrument

I exemplen här nedan används några av PHILIPS IEC-buss instrument, nämligen:

PHILIPS PM 2528 Automatisk Multimeter
PHILIPS PM 6671 Räknare
PHILIPS PM 5190 Synthesizer LF
PHILIPS PM 3310 Oscilloskop

För att exemplen skall bli begripliga följer här en beskrivning av hur man programmerar och läser av dessa instrument.

PHILIPS PM 2528 Automatisk multimeter.

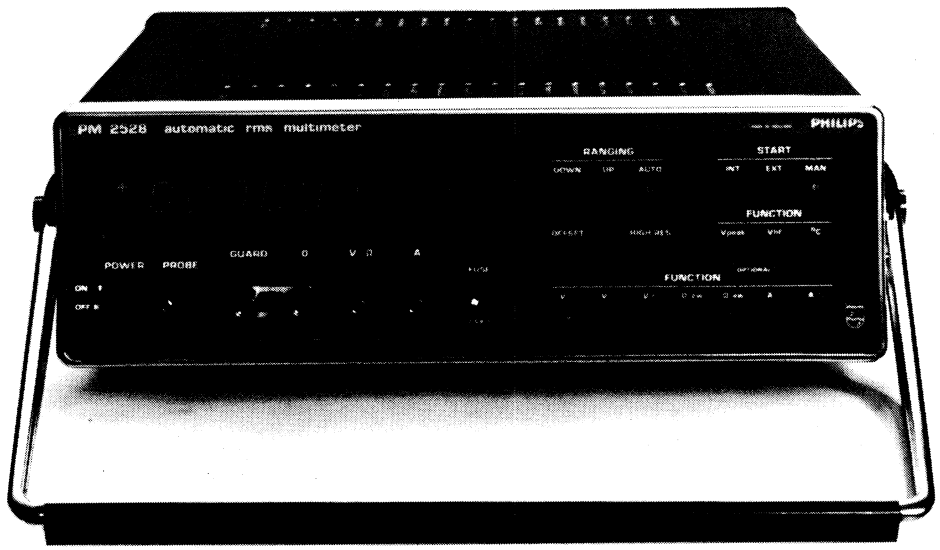


Fig. 6.3 PHILIPS PM 2528 Automatisk multimeter.

Programmering av multimetern.

Funktion	Kod	Beskrivning
V DC	F00	Sätter multimetern i läge för att mäta
V AC	F01	
V AC med offset	F02	
Ohm 2W	F03	
Ohm 4W	F04	
A DC	F05	
A AC	F06	
'C	F07	
Vhf	F08	
Vpeak t	F09	
Vpeak b	F10	
Vpeak t&b	F11	

Område	R0	Automatiskt områdesval.
	R1	Lägsta området.
	..	
	..	
	..	
	R8	Högsta området. I exemplen i boken används bara R0 varför områdena R1-R8 är i detta fall ointressanta. (För den intresserade står det i manualen för multimetern)
Start modell	T0	Intern start.
	T1	Extern start via IEC-bussen.
	T2	Extern start via IEC-bussen eller via BNC-kontakten på baksidan.
Start kommandon	E1	Startar mätningen.
	GET	Group Execute Trigger, startar mätningen.
Data klar förfrågan	D0	SRQ ej tillåten. Inget SRQ när nytt mätvärde är tillgängligt för läsning. D1 SRQ tillåten. SRQ när nytt mätvärde finns tillgängligt för läsning.
Mäthastighet	S0	Normal mäthastighet. Ungefär 100mS.
	S1	Hög mäthastighet. Ungefär 20 mS.
Upplösning	H0	Normal upplösning på mätvärdet.
	H1	Hög upplösning på mätvärdet.
Offset läge	O1O1	Offset spänning är kompenserad.
	O0O0	Offset spänning är ej kompenserad. OBS! Skall vara O-Ett-O-Ett, EJ Noll-Ett-Noll-Ett!

Läsning av Multimetern

Data som sänds ut omfattar 12 tecken.

Tecken nr	**Data**
1	Det först tecknet kan vara +,- eller mellanslag. +/- indikerar någon av funktionerna: V DC, A DC, 'C, V eller V b. Mellanslag indikerar: V AC, V AC, Ohm 2W, Ohm 4W, A AC, V b och Vhf.
2	En siffra.
3	En siffra eller ett decimalkomma.
4	En siffra eller ett decimalkomma.
5	En siffra eller ett decimalkomma.
6	En siffra eller ett decimalkomma.
7	En siffra.
8	En siffra.
9	E. Betecknar första bokstaven i ordet EXPONENT.
10	+/- Betecknar polariteten på exponenten 10ü+/-E.
11	0,3,6 Värdet på exponenten. 10ü+/-0, 10ü+/-3 osv.
12	ETX (End of TeXt). Indikerar slut.

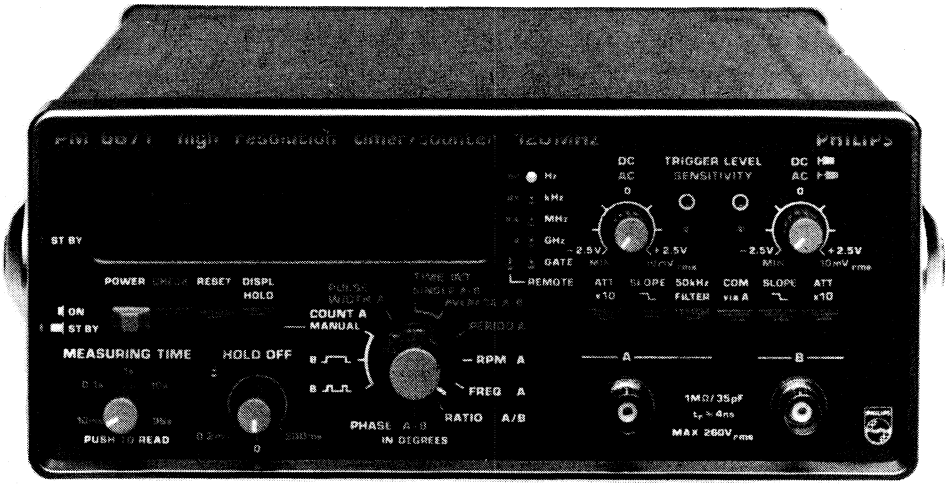


Fig. 6.4 PHILIPS PM 6671 Räknare

Programmering av räknaren.

Funktion	**Kod**	**Beskrivning.**
COUNT A	F9	Räknar A.
COUNT A START OR STOP BY B	F4	Räknar A, start och stopp styrs av B.
COUNT A GATED BY B	F2	Räknar A styrt av B
FREQUENCY A	F15	Ger frekvensen på A.
PERIOD A	F13	Ger periodlängden på A.
TIME INTERVAL A-B SINGLE	F6	Ger tidsintervallet mellan A och B på en period.
TIME INTERVAL A-B AVERAGE	F11	Ger medelvärdet på tidsintervallet mellan A och B.
PULS DURATION A	F12	Ger A's pulsvaraktighet.
RATIO A/B	F7	Ger förhållandet A/B.
PHASE A-B	F3	Ger fasförskjutningen mellan A och B.
RPM	F14	Ger antal perioder per minut.
HOLD OFF	H0	Stänger av HOLD OFF.
	H1	Sätter på HOLD OFF.
INPUT CHANNEL	A	Inmatningskanal.
FREQUENCY AVERAGE	G1	Ger medelvädet på frekvensen på ingång E.
ARMING (OR EXT RESET)	G0	Arming eller extern nollställning. Beroende på omkopplare på baksidan av instrumentet.
DISPLAY HOLD	T1	En mätning.
	T0	Repeterande mätningar.
MEASURING TIME	M...	Anger hur lång tid mätningen skall ske. Ett exempel M10E-1 anger en mät tid på 1 sek. Mät tiden kan varieras mellan 0.01 S till 99 S.
DEVICE CLEAR	D	Programmerar instrumentet till defaultvärdena som är: F15, H0, T0, G0, A, S0, M30E-2.
PROGRAM OUT	P	Sänder ut vad räknaren har blivit programmerad till. Exempel: F13H0T1G1AS1M30E-1, som sänds till datorn.
SERVICE REQUEST		
DISABLE	S0	Ej tillåten.
ENABLE	S1	Tillåten.

Tolkning av mätresultatet.

Resultatet av mätningen ges på följande form:

FFOXXXXXXXXXXE+/-XDD

Där "FF" betecknar funktionskoden som är:

CA	:	COUNT A
CP	:	COUNT A GATED BY B
PH	:	PHASE A-B
SS	:	COUNT A START/STOP BY B
FA	:	FREQUENCY A
FB	:	FREQUENCY B
FC	:	FREQUENCY C
TI	:	SINGLE TIME INTERVAL
RA	:	RATIO A/B
PA	:	PERIOD A
TA	:	TIME INTERVAL AVERAGE
PD	:	PULSE DURATION
RM	:	RPM
MT	:	MEASURING TIME

"O" indikerar "Overflow", medan ett mellanslag markerar "icke overflow".

"X" ger mantissan i det utsända mätvärdet.

"E" står för exponent och plus (+) eller minus (-) tecknet för positiv respektive negativ exponent. "X" anger beloppet av exponenten.

"DD" anger avslutningstecken "CR" och "LF".

PHILIPS PM 5190 Synthesizer.

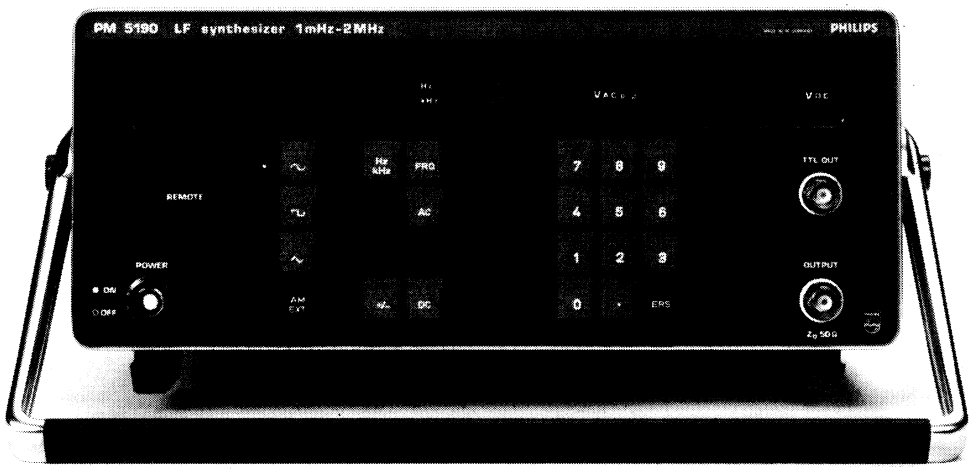


Fig. 6.5 PHILIPS PM 5190 Synthesizer.

Programmering av synthesizer.

Programmeringen av PM 5190 är betydligt lättare än de andra instrumenten. Man har bara fyra parametrar som behöver anges. Först en parameter som anger frekvensen, sedan en som talar om topp-topp spänningen, och eventuellt en DC offset spänning. Sist skall man bara ange vilken form man önskar på kurvan.

Frekvens.

Frekvensen skall anges på följade sätt:

FXXX.XXX

Där F anger att det som kommer är värdet av frekvensen och XXX.XXX är frekvensen i kHz.

Topp-topp spänning.

När man anger topp-topp spänningen börjar man med ett "A" som står för AC. Sedan kommer tre siffror som anger topp-topp spänningen. Den första siffran får bara vara ett (1) eller noll (0). Det måste också finnas en decimalpunkt före 1:a, 2:a eller 3:e siffran. Kommandot blir sålunda:

AXX.X

Offset-spänning.

Offset-spänningen måste alltid sättas tillsammans med topp-topp spänningen. Ett "D", (som står för DC), måste finnas i position 6 räknat från "A", annars ingnorerar instrumentet instruktionen. Om värdet är ett positivt tal behöver inte plus (+) tecknet anges. Kommandot blir sålunda:

AXX.XD-XX

Vågform.

Vågformen anges med ett W, som skall åtföljas av en siffra mellan 1 och 5 som anger vågformen enligt tabellen nedan:

1 = Sinusvåg
2 = Fyrkantsvåg
3 = Trekantsvåg
4 = Sinusvåg/AM ext.
5 = Trekantsvåg/AM ext.

Ett exempel:

Man vill ha en sinusvåg med frekvensen 12.5 kHz och topp-topp spänningen på 10 V men ingen offset spänning.

Kommandot blir sålunda:

```
70 CMD "?U1","F12.5A10.0D00W1"
```

PM 5190 kan inte användas som "Talker".

********** **Programexempel** **********

Exempel 1: Användning av Philips multimeter PM 2528.

Programmet nedan visar hur man kan använda en automatisk multimeter på IEC-bussen. Eftersom Philips PM 2528 multimeter kan vara både "Talker" och "Listener" kan man programmera den till vad man skall mäta och i vilket

mätområde man skall mäta. Programmet visar hur man kan välja mätfunktion på multimetern och sedan sända det uppmätta resultatet till datorn. När mätvärdet kommer till datorn skrivs det ut på skärmen och en ny mätning kan ske, eller om så önskas en återgång till programmets huvudmeny för mätning av en annan funktion. På rad 360 programmeras multimetern för mätning av den valda funktionen. Där programmeras även följande funktioner:

- Automatiskt val av mätområde (R0)
- Start av mätning från datorn med Busy-signalen (T1)
- Ingen SRQ (D0)
- Hög hastighetsläge. Integrationstid 20 mS. (S1)
- Högupplösning på mätresultatet. (H1)

De nu nämnda funktionerna kan ej ändras om man kör programmet, men det går givetvis att ändra i programmet på rad 360. På rad 380 sänder man ut en Busy-signal så att instrumentet börjar mäta.

```
10  ! MÄTNING MED EN AUTOMATISK MULTIMETER
20  !
30  EXTEND : INTEGER
40  OPEN 'IEC:' AS FILE 49
50  ; CHR$(12)
60  ; TAB(5) ' MÄTNING MED PHILIPS AUTOMATISKA MULTIMETER PM 2528 '
70  ; CUR(5,20) ' 1. MÄTNING AV LIKSPÄNNING'
80  ; CUR(6,20) ' 2. MÄTNING AV VÄXELSPÄNNING'
90  ; CUR(7,20) ' 3. MÄTNING AV VÄXELSPÄNNING'
100 ; CUR(8,20) " 4. MÄTNING AV MOTSTÅND MED TVÅ TRÅDIG PROB"
110 ; CUR(9,20) " 5. MÄTNING AV MOTSTÅND MED FYR TRÅDIG PROB"
120 ; CUR(10,20) ' 6. MÄTNING AV LIKSTRÖM'
130 ; CUR(11,20) ' 7. MÄTNING AV VÄXELSTRÖM'
140 ; CUR(12,20) ' 8. MÄTNING AV TEMPERATUR'
150 ; CUR(13,20) ' 9. MÄTNING AV HÖGFREKVESSPÄNNING'
160 ; CUR(14,20) '10. MÄTNING AV SPÄNNINGSTOPPAR'
170 ; CUR(15,20) '11. MÄTNING AV SPÄNNINGSBOTTNAR'
180 ; CUR(16,20) '12. MÄTNING AV TOPP-TOPP SPÄNNING'
190 !
200 ; CUR(18,15); : INPUT 'VILKET ALTERNATIV ÖNSKAS'(Alternativ
210 !
220 ! INLÄSNING AV KODEN FÖR OMRÅDET
230 !
240 RESTORE
250 FOR Loop=1 TO Alternativ
260   READ Kod$,Enhet$
270 NEXT Loop
280 DATA F00,VOLT,F01,VOLT,F02,VOLT,F03,OHM,F04,OHM,F05,AMPERE
290 DATA F06,AMPERE,F07,GRADER C,F08,VOLT,F09,VOLT,F10,VOLT,F11,VOLT
300 !
310 ! MÄTNING AV DET VALDA OMRÅDET
320 !
330 ! STÄLLER MULTIMETERN I VALT LÄGE + AUTOMATISK SKALNING,
      EXTERN START
340 ! INGEN SRQ,SNABB MÄTNING OCH HÖG UPPLÖSNING PÅ MÄTVÄRDET
350 !
360 CMD "?U6",Kod$+"R0T1D0S1H1"
370 !
380 CMD "?U6"+CHR$(8) ! STARTAR EN MÄTNING
390 !
400 CMD "?V5" ! DEFINIERAR MULTIMETER SOM TALKER OCH DATORN
    SOM LISTENER
410 !
```

```
420 Svar$=IEC$(12) ! LÄSER VÄRDET FRÅN BUSSEN
430 !
440 ! UTSKRIFT AV SVARET
450 !
460 ; CUR(20,5) 'SVARET AV MÄTNINGEN BLEV ' Svar$ ' ' Enhet$
470 ; CUR(22,20); : INPUT 'NY MÄTNING ELLER ÅTERGÅNG (M/Å) <M>'
      Fråga$
480 !
490 ! MASKAR SMÅ BOKSTÄVER TILL STORA
500 !
510 IF CHR$(ASCII(Fråga$) AND 95)='Å' GOTO 10
520 GOTO 380
530 END
```

Exempel 2: Användning av Philips räknare PM 6671.

Programexemplet för räknaren liknar programmet för multimetern. Detta beror på att båda instrumenten kan användas som "Talker" och "Listener". Det som skiljer programmen åt är behandlingen av svaret från instrumentet. Räknaren sänder även med information om vilken sorts mätning den utförde, den indikerar om värdet är för stort plus några kontroll tecken efter talet. Dessa tecken tar programmet bort. Om det blir för stort tal så indikeras detta. En annan sak som skiljer programmen åt är att man måste ange hur lång tid instrumentet skall mäta. Den fasta programmeringen av räknaren är defaultvärdena förutom valet av funktionen och inställning av tiden:

- Hold off avstängd
- Arming eller Externt reset beroende på en switch
- Kanal A
- Display hold repeterande
- Ingen SRQ

Vill man ändra på detta gör man det på rad 340. I definitionen på raderna 530-580 tar man bort de ej önskade tecknen som sänds från räknaren.

```
10  ! MÄTNING MED PHILIPS PM 6671 RÄKNARE
20  !
30  EXTEND : INTEGER
40  OPEN 'IEC:' AS FILE 49
50  ; CHR$(12)
60  ; TAB(5) ' MÄTNING MED PHILIPS RÄKNARE PM 6671 '
70  ; CUR(5,20) ' 1. COUNT A (MANUEL)'
80  ; CUR(6,20) ' 2. COUNT A START/STOP BY B'
90  ; CUR(7,20) ' 3. COUNT A GATED BY B'
100 ; CUR(8,20) ' 4. FREQUENCY A'
110 ; CUR(9,20) ' 5. PERIOD A'
120 ; CUR(10,20) ' 6. TIME INTERVAL A-B SINGLE'
130 ; CUR(11,20) ' 7. TIME INTERVAL A-B AVERAGE'
140 ; CUR(12,20) ' 8. PULSE DURATION A'
150 ; CUR(13,20) ' 9. RATIO A/B'
160 ; CUR(14,20) '10. PHASE A-B'
170 ; CUR(15,20) '11. RPM'
180 ; CUR(16,20) '12. ÄNDRING AV MÄTTID. NU :' Tid$ ' Sek'
190 !
200 ; CUR(18,15); : INPUT 'VILKET ALTERNATIV ÖNSKAS',Alternativ
210 IF Alternativ=12 ; CUR(20,10); : INPUT 'HUR LÅNG MÄT-
    TID ÖNSKAS (0.01 - 99 Sek) 'Tid$ : GOTO 50
220 !
230 ! INLÄSNING AV KODEN FÖR OMRÅDET
240 !
```

```
        2,8,100,"19.9"
530 DATA 10,2,4,100,"19.9",10,2,4,100,"19.9",10,2,4,100,"19.9",
        3,3,4,100,"19.9"
540 DATA 1,3,4,100,"19.9",12,2,4,4,"19.9",8,2,8,100,"19.9",5,2,
        8,100,"19.9"
550 DATA 3,2,4,100,"19.9",3,2,4,100,"19.9",3,2,4,100,"19.9",
        3,2,4,100,"19.9"
560 DATA 8,2,4,100,"19.9",8,2,4,4,"19.9",3,3,8,100,"19.9",
        12,2,8,100,"19.9"
570 DATA 10,2,4,100,"19.9",10,2,4,100,"19.9",10,2,8,100,"19.9",
        8,2,8,100,"19.9",10,2,8,100,"19.9",12,2,8,100,"19.9"
580 DATA 8,2,2,4,"19.9"
590 DATA 0,0,0,0,"0.00"
600 DATA 0,0,0,0,"0.00"2,4,4,"19.9",3,3,8,100,"19.9",12,2,8,100,"19.9"
```

Exempel 4: Uppmätning av en förstärkare.

Fig. 6.6 visar uppkopplingen av instrumenten och förstärkaren. Instrumenten som används i detta exempel är PHILIPS PM 5190 Synthesizer, PHILIPS PM 2528 Automatiska multimeter och PHILIPS PM 6671 Räknare.

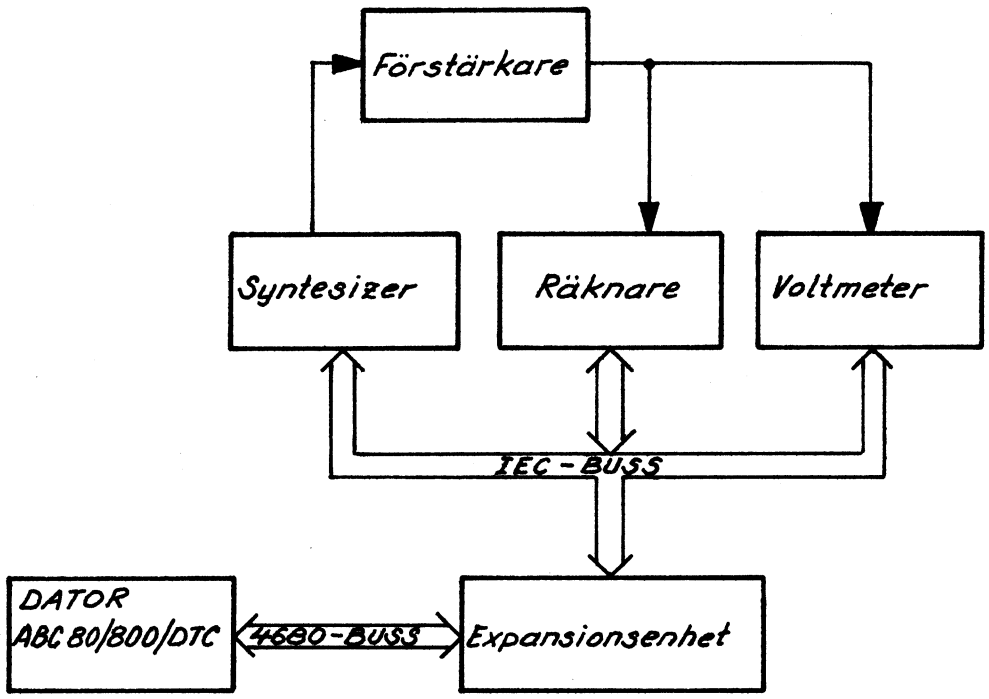


Fig. 6.6 Experimentuppkoppling för mätning på förstärkare.

Programmet består av två delar, en för uppmätning av förstärkaren och en för utvärdering av densamma. Uppmätningen går till så att synthesizern sänder ut en signal till förstärkaren. På utgången av förstärkaren sitter multimetern och räknaren inkopplade där multimetern mäter utspänningen och räknaren mäter frekvensen. Mätvärdena sparas i variabler, som kommer till användning under utvärderingen. När man sedan kommer till denna kan man få frekvensgången utritad på skärmen och/eller alla värden utskrivna på en datalista där man kan också kan se hur mycket förstärkaren förvränger signalen. På rad 220 ställer man synthesizerns utgångsspänning, (den raden kan behöva ändras beroende på vilken ingång man använder på förstärkaren). Om denna spänning ändras måste man även ändra på rad 730 och 850. I detta exempel sänds 0.5 VAC ut, men om man på rad 220 ändrar till t.ex 1.0 VAC måste man dividera med 1 istället för 0.5.

```
10  ! UPPMÄTNING AV EN FÖRSTÄRKARES FÖRSTÄRKNING
20  !
30  !
40  EXTEND : INTEGER
50  DIM Volt.(210),Frekvens.(210)
60  OPEN 'IEC:' AS FILE 49
70  !
80  ; CHR$(12)
90  ; TAB(15) ' UPPMÄTNING AV FÖRSTÄRKARE '
100 ; CUR(5,5) 'Koppla synthesizern till förstärkarens ingång,räknaren'
110 ; CUR(6,5) 'och multimeter till högtalar utgångarna.'
120 ; CUR(8,10) 'Tryck på RETURN för att starta mätningen';
130 GET Return$ : IF Return$<>CHR$(13) GOTO 120
140 ; : ; : ; : ;
150 !
160 ! START AV MÄTNINGARNA
170 !
180 CMD '?U6','F01R0T0D0S1H1' ! PROGRAMMERAR VOLTMETERN
190 !
200 CMD '?U*','F15H0G0T0AS0M30E-2'+CHR$(3) ! PROGRAMMERAR
    RÄKNAREN
210 !
220 CMD '?U1','A0.50D00W1'+CHR$(3) ! SYNTHESIZERNS SPÄNNING
230 !
240 ! LOOPEN SOM TESTAR FÖRSTÄRKAREN
250 !
260 FOR Testfrekvens.=.10001 TO 20.1 STEP .1
270   Index=Index+1
280   CMD '?U1','F'+NUM$(Testfrekvens.)+CHR$(3)
290   CMD '?J5'
300   Slask$=IEC$(18)
310   Frekvens.(Index)=VAL(MID$(IEC$(18),4,13))
320   CMD '?U6'+CHR$(8)
330   CMD '?V5'
340   Volt.(Index)=VAL(LEFT$(IEC$(12),11))
350   ; CUR(20,0) 'MÄTNING NR' Index " Volt.(Index) ' V';
360   ; Frekvens.(Index) ' Hz'
370 NEXT Testfrekvens.
380 !
390 ! UTVÄRDERING AV RESULTATEN
400 !
410 ; CHR$(12)
420 ; TAB(7) ' UTVÄRDERING AV RESULTATEN '
430 ; CUR(5,10) '1. GRAFISK UTSKRIFT PÅ BILDSKÄRMEN'
440 ; CUR(7,10) '2. VÄRDENA UTSKRIVNA PÅ EN PRINTER'
450 ; CUR(9,10) '3. AVSLUTNING'
460 ; CUR(13,5); : INPUT 'VAD ÖNSKAS 'Valutv
```

```
470 IF Valutv<1 OR Valutv>3 GOTO 410
480 !
490 ON Valutv GOSUB 550,790,520
500 !
510 GOTO 390
520 CLOSE 49
530 END
540 !
550 ! GRAFISK UTSKRIFT PÅ SKÄRMEN
560 !
570 ; CHR$(12) : FGPOINT 0,0,0 : FGFILL 239,239 ! Rensar skärm
580 FGCTL 3 ! VÄLJER FÄRGGRUPP 3
590 FGPOINT 39,39,1 ! SÄTTER ORIGO
600 FGLINE 239,39 ! DRAR X-AXELN
610 FGPOINT 39,39
620 FGLINE 39,239 ! DRAR Y-AXELN
630 ; CUR(0,20) 'FÖRSTÄRKARENS FÖRSTÄRKNING I dB'
640 ; CUR(21,0) 'Frekvensen05k';
650 ; TAB(46) '10k15k20k'
660 ; CUR(0,6) '20 dB'
670 ; CUR(5,6) '15 dB'
680 ; CUR(10,6) '10 dB'
690 ; CUR(15,7) '5 dB'
700 ; CUR(19,7) '0 dB'
710 !
720 FOR Loop=1 TO 200
730   FGPOINT 39+Loop,39+20*LOG10(Volt.(Loop)/.5)
740 NEXT Loop
750 !
760 ; CUR(23,0) 'TRYCK PÅ EN TANGENT FÖR ATT ÅTERVÄNDA'
        ; : GET Slask$
770 RETURN
780 !
790 ! LISTAR UT VÄRDENA PÅ PRINTERN
800 !
810 OPEN 'PR:VSA32V72.5' AS FILE 1
820 ; §1,'FREKVENS INFREKVENS UTSPÄNNING
        UTFÖRSTÄRKNING FÖRSTÄRKNING I dB'
830 FOR Loop=100 TO 20000 STEP 100
840   ; §1,TAB(5) Loop;TAB(19) Frekvens.(Loop/100)
        ;TAB(34) Volt.(Loop/100);
850   ; §1,TAB(49) Volt.(Loop/100)/.5
        ;TAB(64)20*LOG10(Volt.(Loop/100)/.5)
860 NEXT Loop
870 CLOSE 1
880 RETURN
```

Exempel 5: Avläsning av PHILIPS PM 3310 Oscilloskop.

Ett av de mera sofistikerade instrumenten som kan kopplas in på IEC-bussen är PHILIPS PM 3310 oscilloskop. Oscilloskopet är ett av de vanligare instrumenten som används inom elektroniken idag. Det finns dock mycket få oscilloskop som går att koppla till IEC-bussen.

Nedanstående program är ett exempel på hur man kan läsa av ett oscilloskop på IEC-bussen och göra om hexkoden från oscilloskopet till klartext. Programmet läser även av ett valbart minne, (av de fyra som finns på PM 3310) för kopiering till datorns bildskärm. Det måste dock då finnas högupplösningsgrafik i datorn. Programmet är uppbyggt så att det först läser av oscilloskopets inställning, varefter datorn ritar ut den av oscilloskopet

uppmätta kurvan på bildskärmen om så önskas. Om man inte vill rita upp kurvan, utan istället ställa om oscilloskopet till ett annat mätområde så är det bara att svara nej på frågan om man vill ha någon kurva uppritad. Efter ändringen kan man få den nya kurvan uppritad.

```
10  ! ANVÄNDNING AV OSCILLOSKOP PÅ IEC-BUSSEN
20  !
30  ! ANVÄNT OSCILLOSKOP ÄR PHILIPS PM 3310
40  !
50  ; CHR$(12) : FGPOINT 0,0,0 : FGFILL 239,239 : FGCTL 3
60  EXTEND : INTEGER
70  OPEN 'IEC:' AS FILE 49
80  DIM Front$(17),Data$=810
90  Hexa$='0123456789ABCDEF'
100 !
110 ! LÄSER INSTÄLLNINGEN AV OSCILLOSKOPET
120 !
130 CMD '?U(',CHR$(27)+'0MT00'+CHR$(3)
140 CMD '?H5'
150 Inställning$=IEC$(39)
160 !
170 ! INLÄSNING AV FRONT$(Loop)
180 !
190 ; CUR(0,10) ' AVLÄSNING AV PHILIPS PM 3310 '
200 FOR Loop=1 TO 17
210   RESTORE 800
220   FOR I=1 TO Loop
230     READ Plats,Antal
240   NEXT I
250   ON Loop RESTORE 850,910,980,850,910,980,1100,1230,1360,
        1440,1490,1560,1600,1640,850,850,850
260   !
270   IF Loop<15 THEN Front$(Loop)=FNLäsa$(MID$(Inställning$,Plats,Antal))
280   IF Loop>14 THEN Front$(Loop)=MID$(Inställning$,Plats,Antal)
290   ; CUR(Loop+1,10) Loop '.' Front$(Loop)
300 NEXT Loop
310 !
320 ! KOPIERING AV BILD FRÅN OSCILLOSKOP TILL DATORN
330 !
340 ; CUR(20,15) 'SKALL NÅGON KURVA UTRITAS (J/N)';
350 INPUT Svar$
360 IF CHR$(ASCII(Svar$) AND 95)<>'J' GOTO 660
370 ; CUR(22,15) 'ANGE VILKET MINNE (K=ACCU, L=STO 1,
        M=STO 2, N=STO 3) ';
380 INPUT Minne$
390 IF ((ASCII(Minne$)) AND 95)<75 OR ((ASCII(Minne$))
        AND 95)>78 GOTO 370
400 !
410 ! INLÄSNING AV DATA TILL KURVAN
420 !
430 CMD '?U(',CHR$(27)+'0R'+LEFT$(Minne$,1)+'00'+CHR$(3)
440 CMD '?H5'
450 Datain$=IEC$(804)
460 !
470 ! UTRITNING AV KURVAN PÅ BILDSKÄRMEN
480 !
490 ; CHR$(12) : FGPOINT 0,0,0 : FGFILL 239,239 : FGCTL 3
500 Räknare=-1
510 FGPOINT 0,0,0
520 FOR Loop=37 TO 754 STEP 3 ! LÄSER IN 240 PUNKTER
530   Slask$=MID$(Datain$,Loop,2)
```

```
540   Värde=(INSTR(1,Hexa$,LEFT$(Slask$,1))-1)*16
550   Värde1=INSTR(1,Hexa$,RIGHT$(Slask$,2))-1
560   Värde=Värde+Värde1
570   IF Värde>=120 Värde=Värde-126 ELSE IF Värde<120
        Värde=Värde+128
580   IF Värde>239 Värde=239 ELSE IF Värde<0 Värde=0
590   Räknare=Räknare+1
600   FGPOINT Räknare,Värde,1
610 NEXT Loop
620 GOTO 340
630 !
640 ! NY INSTÄLLNING
650 !
660 CLOSE 49
670 ; CUR(24,10) 'SKALL NY INSTÄLLNING LÄSAS (J/N)' ; : INPUT Svar$
680 IF CHR$((ASCII(Svar$) AND 95))='J' GOTO 10
690 END
700 DEF FNLäsa$(Byte$)
710   WHILE Jämför$<>Byte$
720     READ Jämför$,Inställ$
730   WEND
740   Jämför$=""
750   RETURN Inställ$
760 FNEND
770 !
780 ! POSITIONSNUMMRET OCH ANTALET TECKEN SOM LÄSES IN
        FRÅN PM 3310
790 !
800 DATA 4,2,7,1,8,1,10,2,13,1,14,1,16,2,19,2,22,1
810 DATA 23,1,25,1,26,1,28,1,29,1,31,2,34,2,37,2
820 !
830 ! DATA FÖR BYTE 3&4, 7&8 KANAL A OCH B'S INSTÄLLNING
840 !
850 DATA D4,"10 mV/RUTA",C4,"20 mV/RUTA",84,
        "50 mV/RUTA",8C,"0.1 V/RUTA"
860 DATA CC,"0.2 V/RUTA",44,"0.5 V/RUTA",64,
        "1 V/RUTA",60,"2 V/RUTA"
870 DATA 50,"5 V/RUTA",D0,"10 V/RUTA",E0,
        "20 V/RUTA",24,"50 V/RUTA"
880 !
890 ! DATA FÖR BYTE 5&9
900 !
910 DATA 2,"KALIBRERAD OCH ANVÄNDER 1:10 PROBE"
920 DATA 3,"KALIBRERAD OCH ANVÄNDER 1:1 PROBE"
930 DATA A,"EJ KALIBRERAD OCH ANVÄNDER 1:10 PROBE"
940 DATA B,"EJ KALIBRERAD OCH ANVÄNDER 1:1 PROBE"
950 !
960 ! DATA FÖR BYTE 6&10
970 !
980 DATA 0,"PÅSLAGEN, ADD/BINV, NOLLSTÄLLD OCH I LÄGE AC"
990 DATA 1,"PÅSLAGEN, ADD/BINV, NOLLSTÄLLD OCH I LÄGE DC
1000 DATA 2,"PÅSLAGEN, ADD/BINV OCH I LÄGE AC
1010 DATA 3,"PÅSLAGEN, ADD/BINV OCH I LÄGE DC
1020 DATA 8,"PÅSLAGEN, NOLLSTÄLLD OCH I LÄGE AC
1030 DATA 9,"PÅSLAGEN, NOLLSTÄLLD OCH I LÄGE DC
1040 DATA A,"PÅSLAGEN OCH I LÄGE AC
1050 DATA B,"PÅSLAGEN OCH I LÄGE DC
1060 DATA F,"AVSLAGEN
1070 !
1080 ! DATA FÖR BYTE 11&12 RECURRENT
1090 !
```

```
1100 DATA D4,"5 nS",C4,"10 nS",84,"20 nS",8C,"50 nS",
        CC, "0.1 uS",44,"0.2 uS"
1110 DATA 64,"0.5 uS",60,"1 uS",50,"2 uS",D0,"5 uS",
        E0,"10 uS",24,"20 uS"
1120 DATA 3C,"50 uS",38,"0.1 mS",30,"0.2 mS",70,"0.5 mS",
        78,"1 mS",18,"2 mS"
1130 DATA 98,"5 mS",88,"10 mS",0C,"20 mS",2C,"50 mS",A8,
        "0.1 S",90,"0.2 S"
1140 !
1150 ! DATA FÖR BYTE 11&12 ROLL-MODE
1160 !
1170 DATA D0,"0.5 S",E0,"1 S",24,"2 S"
1180 DATA 3C,"5 S",38,"10 S",30,"20 S",70,"30 S",78,
        "60 S",18,"120 S"
1190 DATA 98,"360 S",88,"900 S",0C,"1800 S",2C,"3600 S"
1200 !
1210 ! DATA FÖR BYTE 13&14
1220 !
1230 DATA 10,"RECURRENT MODE OCH TRIGGAR",
        18,"RECURRENT MODE OCH TRIGGAR INTE"
1240 DATA 20,"MULTIPLE MODE OCH MINNENA UPPFRÄSCHADE"
1260 DATA 28,"MULTIPLE MODE OCH VÄNTAR PÅ TRIGGSIGNAL
1280 DATA 40,"SINGLE SHOT OCH ACCU UPPFRÄSCHAT"
1290 DATA 48,"SINGLE SHOT OCH VÄNTAR PÅ TRIGGER"
1310 DATA 80,"ROLL MODE FÄRDIGT",88,"ROLL MODE EJ FÄRDIGT"
1330 !
1340 ! DATA FÖR BYTE 15 VISADE REGISTER
1350 !
1360 DATA F,"INGET",E,"ACCU",D,"STO 1",C,"ACCU OCH STO 1",
        B,"STO 2",A,"ACCU OCH STO2"
1370 DATA 9,"STO 1 OCH STO 2",8,"ACCU, STO 1 OCH STO 2",7,
        "STO 3",6,"ACCU OCH STO 3
1380 DATA 5,"STO 1 OCH STO 2",4,"ACCU, STO 1 OCH STO 3",3,
        "STO 2 OCH STO 3"
1390 DATA 2,"ACCU, STO 2 OCH STO 3",1,"STO 1, STO 2 OCH STO 3"
1400 DATA 0,"ACCU, STO 1, STO 2 OCH STO 3"
1410 !
1420 ! DATA FÖR BYTE 16 INVERTERADE REGISTER
1430 !
1440 DATA 0,"STO 1, STO 2 OCH STO 3",2,"STO 2 OCH STO 3",4,
        "STO 1 OCH STO 3"
1450 DATA 6,"STO 3",8,"STO 1 OCH STO 2",A,"STO 2",C,"STO 1",
        E,"INGET"
1460 !
1470 ! DATA FÖR BYTE 17
1480 !
1490 DATA 0,"Y*5, DOTTAR, X=A & Y=B",1,"Y*5, DOTTAR, X=t"
1500 DATA 2,"Y*5, INGA DOTTAR, X=A & Y=B",3,"Y*5,
        INGA DOTTAR, X=t"
1510 DATA 4,"Y*1, DOTTAR, X=A & Y=B",5,"Y*1, DOTTAR, X=t"
1520 DATA 6,"Y*1, INGA DOTTAR, X=A & Y=B",7,"Y*1,
        INGA DOTTAR, X=t"
1530 !
1540 ! DATA FÖR BYTE 18
1550 !
1560 DATA 0,"ACCU REN, MINNESLÅS",4,"ACCU REN",8,"MINNESLÅS",
        C,"WRITE"
1570 !
1580 ! DATA FÖR BYTE 19
1590 !
1600 DATA 1,"ACCU",2,"STO 1",4,"STO 2",8,"STO 3"
```

```
1610 !
1620 ! DATA FÖR BYTE 20 EJ PLOT MODE
1630 !
1640 DATA 1,"STO 1, STO 2 OCH STO 3",3,"STO 2 OCH STO 3",
        5,"STO 1 OCH STO 2"
1650 DATA 7,"STO 3",9,"STO 1 OCH STO 2",B,"STO 2",D,"STO 1",
        F,"INGET"
1660 !
1670 ! DATA FÖR BYTE 20 PLOT MODE
1680 !
1690 DATA 0,"STO 1, STO 2 OCH STO 3",2,"STO 2 OCH STO 3",4,
        "STO 1 OCH STO 2"
1700 DATA 6,"STO 3",8,"STO 1, STO 2",A,"STO 2",C,"STO 1",
        E,"INGET"
```

Exempel 6: Avancerad användning av PHILIPS PM 3310 Oscilloskop.

Med nedanstående program kan man styra alla funktionerna hos PM 3310 Oscilloskop. Man kan läsa av frontpanelen, programmera om frontpanelen, läsa av vad som står i varje minne och applicera en egen kurva från datorn till vilket minne som helst. Trots sin mängd av funktioner är programmet lätt att använda. När man startar upp programmet får man upp en meny på bildskärmen. Det första man skall göra då är att öppna IEC-bussen för sändning och mottagning, vilket görs med kommandot <OP>. När man skall avsluta en körning med programmet skall man stänga IEC-bussen vilket görs med kommandot <CL>. Efter det att man öppnat IEC-bussen har man 12 olika kommandon att använda. De är LF, DF, ÄF, SF, LR, SR, DR, LD, SD, DD, TD, och BD.

<LF>
läser av inställningen på frontpanelen och skriver ut koden för inställningen. Förklaring till koden står i manualen för instrumentet.

<DF>
skriver ut koden i tolv delar om två bytes i varje del. Framför varje del står vilka byte det är, därefter en förkortning som talar om vilken funktion som koden framför berör. AT.A (B) = Attenuator A (B), MO.A (B) = Mode channel A (B), BA.T = Time base, MO.T = Time base mode, RE = Display and invert register, MO.D = Disply mode, SE.R = Register select for scaling/plot and save, TR+ = Trigger delay och DEL = delay.

<ÄF>
ändrar den tidigare inställningen. Först skrivs koden ut som under <DF> varefter en cursor placeras framför koden. Då har man tre alternativ att svara. Antingen slår man in en ny kod för den funktionen (2 byte) utan avslutande RETURN eller så slår man RETURN direkt då cursorn placerar sig framför raden och då får den raden samma betydelse som det som stod där förut. Det sista alternativet är om man har slagit fel tidigare då trycker man "<" och så får man börja slå om allt från början. Trycker man "<" på första raden så återvänder man till menyn.

<SF>
när man kört <ÄF> ger man kommandot <SF> för att skicka ut data till instrumentet.

<LR>
läser ett register från PM 3310 till datorn efter att datorn frågat vilket register som skall läsas. 0 = ACCU, 1 = STO 1, 2 = STO 2, 3 = STO 3.

<SR>
skriver data från datorn till ett register på PM 3310. Här frågas också till vilket register data skall överföras till. Samma värden som under <LR> gäller.

<DR>
skriver ut inställningen och data på skärmen i kodform.
<LD>
läser en datafil från skivan.
<SD>
skriver data på en fil på flexskivan.
<DD>
ritar ut en kurva på skärmen med tecknet "*".
<TD>
tillverkar data för senare utskick till ett register på PM 3310. Man kan välja mellan sinus, fyrkant, triangel och en egen kurva. Den egna kurvan lägger man in i programmet före körning.
<BD>
räknar ut max och min värde för inneliggande data.

```
10  ! PROGRAMMMERING OCH LÄSNING AV PHILIPS PM 3310
20  !
30  INTEGER : ; CHR$(12)
40  !
50  DIM D$=600,I$=900,Ö$=100,M(24),D(256),L$(13)=6
60  F$="MTCCBBD03798100C381E003310"
70  R$="RK00207B02100F1250000120"
80  !
90  ! ---------- LÄS KOMMANDO
100 !
110 ; CUR(0,8) " PROGRAMMERING AV PHILIPS PM 3310 "
120 GOSUB 380
130 ; CUR(21,15); : INPUT "KOMMANDO ? "K$ : IF LEN(K$)<2 GOTO 130
140 ; CUR(21,0) SPACE$(80)
150 K=(INSTR(1,"LF,DF,ÄF,SF,LR,SR,DR,LD,SD,OP,DD,TD,BD,CL",
        LEFT$(K$,2))+5)/3
160 ON K GOSUB 190,560,690,880,1050,1150,2090,1440,2380,
        2630,250,2270,2730,3410,300
170 GOTO 90
180 !
190 ; CUR(21,15) SPACE$(65)
200 ; CUR(21,30) "?????"
210 RETURN
220 !
230 ! --------- ÖPPNAR IEC-BUSSEN
240 !
250 OPEN "IEC:" AS FILE 49
260 CMD "?U2",CHR$(27)+"0DD0"+CHR$(3) : E4=1 : RETURN
270 !
280 ! --------- STÄNGER IEC-BUSSEN
290 !
300 CLOSE 49 : E4=0 : RETURN
310 !
320 ! --------- IEC-BUSSEN HAR EJ ÖPPNATS
330 !
340 ; CUR(21,15) SPACE$(65)
350 ; CUR(21,30) "**** IEC:EJ ÖPPNAT!!!! ****" CHR$(7)
360 RETURN
370 !
380 ! ---------- UTSKRIFT AV MENYN
390 !
400 ; CUR(2,10) "LF = LÄS FRONT"
410 ; CUR(3,10) "DF = DISPLAY FRONT"
420 ; CUR(4,10) "ÄF = ÄNDRA FRONT"
430 ; CUR(5,10) "SF = SKRIV FRONT"
```

```
440 ; CUR(7,10) 'LR = LÄS REGISTER FRÅN PM 3310'
450 ; CUR(8,10) 'SR = SKRIV -"-       TILL     -"-'
460 ; CUR(9,10) 'DR = DISPLAY -"- (INST&DATA)'
470 ; CUR(11,10) 'LD = LÄS DATA FRÅN DISK'
480 ; CUR(12,10) 'SD = SKRIV -"- TILL -"-'
490 ; CUR(14,10) 'DD = DISPLAY DATA PÅ SKÄRM'
500 ; CUR(15,10) 'TD = TILLVERKA DATA'
510 ; CUR(16,10) 'BD = BEARBETA -"-'
520 ; CUR(18,10) 'OP = OPEN IEC:'
530 ; CUR(19,10) 'CL = CLOSE -"-'
540 RETURN
550 !
560 ! ---------- LÄS FRONT
570 !
580 IF E4=0 GOTO 340
590 ; CUR(21,15) SPACE$(65)
600 CMD "?U(",CHR$(27)+"0MT00"+CHR$(3)
610 CMD "?H5" : F$=""
620 FOR I=1 TO 13
630   I1$=LEFT$(IEC$(3),2)
640   F$=F$+I1$
650 NEXT I
660 ; CUR(21,30) F$
670 RETURN
680 !
690 ! ------- DISPLAY FRONT
700 !
710 ; CHR$(12)
720 RESTORE 3920
730 FOR I=1 TO 13
740   READ L$(I)
750 NEXT I
760 RESTORE 3940
770 FOR I=1 TO LEN(F$) STEP 2
780   READ L$
790   ; L$(I/2+1) TAB(8) L$ TAB(15) MID$(F$,I,2) ""
800 NEXT I
810 IF K<>3 RETURN
820 ; CUR(22,10) 'TRYCK PÅ EN TANGENT FÖR RETURN TILL
        MENYN'; : GET G$
830 ; CHR$(12)
840 RETURN
850 !
860 ! PROGRAMMERING AV FRONTPANELEN
870 !
880 GOSUB 690
890 ; CUR(1,20);
900 C=0
910 FOR I=1 TO LEN(F$)
920   GET Ö$
930   IF Ö$="<" GOTO 1010
940   IF Ö$=CHR$(13) AND (I AND 1)=1 ; CUR(I/2+1,20)
        MID$(F$,I,2) : Q9=1
950   IF Q9=1 Q9=0 : I=I+1 : GOTO 990
960   IF Ö$=CHR$(13) AND (I AND 1)=0 GOTO 920
970   F$=LEFT$(F$,I-1)+Ö$+RIGHT$(F$,I+1) : C=1
980   PUT Ö$
990   IF (I AND 1)=0 ; CUR(I/2+1,20);
1000 NEXT I
1010 ; : IF C GOTO 880
1020 ; CHR$(12)
```

```
1030 RETURN
1040 !
1050 ! ---------- SET FRONT
1060 !
1070 IF E4=0 GOTO 340
1080 CMD "?U(",CHR$(27)+"0"+F$
1090 ; CUR(21,15) SPACE$(65)
1100 ; CUR(21,30) "ÖVERFÖRT "
1110 RETURN
1120 !
1130 ! ---------- LÄS DATA FRÅN PM3310 REGISTER
1140 !
1150 GOSUB 1340
1160 IF E4=0 GOTO 340
1170 CMD "?U(",CHR$(27)+"0R"+CHR$(75+Ö)+"00"
1180 CMD "?H5"
1190 ; CHR$(12)
1200 R$=""
1210 FOR I=1 TO 12
1220   I$=IEC$(3)
1230   R$=R$+LEFT$(I$,2)
1240 NEXT I
1250 D$=""
1260 FOR I=1 TO 256
1270   I$=IEC$(3)
1280   D$=D$+LEFT$(I$,2)
1290 NEXT I
1300 ; CUR(21,15) SPACE$(65)
1310 ; CUR(21,30) "LÄST REGISTER " ASCII(MID$(R$,2,1))-75
1320 RETURN
1330 !
1340 ! --------- VAL AV REGISTER
1350 !
1360 ON ERROR GOTO 1360
1370 ; CUR(21,15) SPACE$(65)
1380 ; CUR(21,30);
1390 INPUT "VILKET REGISTER SKALL BEHANDLAS (0-3) ? "Ö$
1400 Ö=VAL(Ö$)
1410 IF Ö<0 OR Ö>3 GOTO 1360
1420 RETURN
1430 !
1440 ! ------------ DISPLAY REGISTER DATA
1450 !
1460 I1=0 1470 FOR I=3 TO 15 STEP 6
1480   GOSUB 2000
1490   I1=I1+1
1500   G.(I1)=G.
1510 NEXT I
1520 G.(4)=FNH1(MID$(R$,21,1))*1000+FNH1(MID$(R$,22,1))*100
1530 G.(4)=G.(4)+FNH1(MID$(R$,23,1))*10+FNH1(MID$(R$,24,1))
1540 IF MID$(R$,19,1)="F" G.(4)=-G.(4)
1550 FOR I=7 TO 20
1560   M(I)=FNH1(MID$(R$,I,1))
1570 NEXT I
1580 ; CHR$(12)
1590 GOSUB 1850
1600 ; "****** REGISTER PARAMETERS *******"
1610 ; : ; "KANAL ";
1620 IF M(7) AND 2 ; "A ELLER B(1 SVEP)"; ELSE ; "A OCH B
       (2 SVEP)";
1630 ; ""; : IF M(7) AND 1 ; "JÄMN" ELSE ; "UDDA"
```

```
1640 IF (M(8) AND 8)=0 ; "ADD";
1650 IF (M(14) AND 8)=0 ; "B INVERTERAD";
1660 ;
1670 IF (M(8) AND 4)=0 ; "----- SETTING KANAL A -----"
1680 I1=7
1690 GOSUB 1920
1700 IF (M(14) AND 4)=0 ; "----- SETTING KANAL B -----"
1710 I1=13
1720 GOSUB 1920
1730 ; "========== TIDBAS ========"
1740 ; "TIDBAS: " G.(3) "SEK./RUTA"
1750 ; "MODE:"; : IF M(13) AND 8 ; "ROLL";
1760 IF M(13) AND 4 ; "SINGLE";
1770 IF M(13) AND 2 ; "MULTIPLE;"
1780 ;
1790 ; "DELAY" G.(4)
1800 ; CUR(22,10) 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
1810 GET G$
1820 ; CHR$(12)
1830 RETURN
1840 !
1850 ! -------- DISP.PARAM.BYTE
1860 !
1870 FOR I=1 TO LEN(R$) STEP 2
1880   ; MID$(R$,I,2) " ";
1890 NEXT I
1900 ; : RETURN
1910 !
1920 ! ----- DISP.KANAL SETTING
1930 !
1940 ; "FÖRSTÄRKNING" G.(I1/6) "VOLT / RUTA"
1950 IF M(I1) AND 1 ; "EJ KALIBRERAD" ELSE ; "KALIBRERAD"
1960 IF M(I1+1) AND 1 ; "DC KOPPLAD" ELSE ; "AC KOPPLAD"
1970 IF M(I1+1) AND 2 ; "ZERO"
1980 RETURN
1990 !
2000 ! ------- OMVANDLING AV 4 BYTE ASCII TILL FLOAT G.(3)
2010 !
2020 G=VAL(MID$(R$,I+1,1))
2030 IF MID$(R$,I,1)="1" G=-G
2040 G.=VAL(MID$(R$,I+2,2))*.01*10.üG
2050 RETURN
2060 !
2070 ! --------- SKRIV DATA TILL PM 3310
2080 !
2090 GOSUB 1340
2100 IF E4=0 GOTO 340
2110 CMD "?U("
2120 R$="R"+CHR$(75+Ö)+RIGHT$(R$,3)
2130 FOR I=1 TO 24 STEP 2
2140   CMD ,MID$(R$,I,2)+CHR$(3)
2150 NEXT I
2160 FOR I=1 TO 512 STEP 2
2170   CMD ,MID$(D$,I,2)+CHR$(3)
2180 NEXT I
2190 ; CHR$(12)
2200 ; R$ : ; D$
2210 ; "SKRIVIT TILL REGISTER " ASCII(MID$(R$,2,1))-75
2220 ; : ; 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
2230 GET G$
2240 ; CHR$(12)
```

```
2250 RETURN
2260 !
2270 ! ---------- DISPLAY DATA
2280 !
2290 ; CHR$(26,12)
2300 FOR I=i TO 512 STEP 2
2310   ; CUR(10-(FNH(MID$(D$,I,2))+5)/10,3+I/16) "*";
2320 NEXT I
2330 ; CUR(22,0) 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
2340 GET G$
2350 ; CHR$(12)
2360 RETURN
2370 !
2380 ! --------- LÄS FRÅN DISK
2390 !
2400 GOSUB 2500 : IF K=0 RETURN
2410 INPUT §1R$
2420 D$=""
2430 FOR I=0 TO 5
2440   INPUT §1Ö$
2450   D$=D$+Ö$
2460 NEXT I
2470 CLOSE 1
2480 RETURN
2490 !
2500 ! --------- VAL AV FIL
2510 !
2520 ; CUR(21,15) SPACE$(65)
2530 ; CUR(21,30);
2540 INPUT "FILNAMN ? "S$
2550 IF LEN(S$)>8 S$=LEFT$(S$,8)
2560 S$=S$+".DSP"
2570 ON ERROR GOTO 2600
2580 OPEN S$ AS FILE 1
2590 RETURN
2600 IF K=9 K=0 : ; CUR(21,30) "FILEN" S$ "FINNS EJ !!"
                ERRCODE : RETURN
2610 PREPARE S$ AS FILE 1 : RETURN
2620 !
2630 ! --------- SKRIV TILL DISK
2640 !
2650 GOSUB 2500
2660 ; §1 R$
2670 FOR I=0 TO 4
2680   ; §1 MID$(D$,1+I*100,100)
2690 NEXT I
2700 ; §1 RIGHT$(D$,501)
2710 CLOSE 1 : RETURN
2720 !
2730 ! -------- TILLVERKA DATA
2740 !
2750 ; CUR(21,15) SPACE$(65)
2760 ; CUR(21,30);
2770 INPUT "S=SINUS F=FYRKANT T=TRIANGEL E=E-FUNKT. ? "Ö$
2780 ON INSTR(1,"SFTE",Ö$)+1 GOTO 2770,2800,2930,3080,3260
2790 !
2800 ! -------- SINUS
2810 !
2820 U=100
2830 P=2
2840 ; CUR(21,15) SPACE$(65)
```

```
2850 ; CUR(21,30) "SINUSU-TOPP=" U "PERIODER=" P
2860 K.=P*2*PI/256
2870 D$=" "
2880 FOR T=0 TO 255
2890   D$=D$+FNH2$(U*(SIN(T*K.))+.5)
2900 NEXT T
2910 RETURN
2920 !
2930 ! -------- FYRKANT
2940 !
2950 U=100
2960 P=3
2970 ; CUR(21,15) SPACE$(65)
2980 ; CUR(21,30) "FYRKANTU-TOPP=" U "PERIODER=" P
2990 P0=256/(P*2)
3000 D$=" "
3010 FOR T=0 TO 255
3020   T1=T1+1
3030   IF T1>P0 T1=0 : U=U*(-1)
3040   D$=D$+FNH2$(U)
3050 NEXT T
3060 RETURN
3070 !
3080 ! --------- TRIANGEL
3090 !
3100 U=100
3110 P=3
3120 ; CUR(21,15) SPACE$(65)
3130 ; CUR(21,30) "TRIANGELU-TOPP=" U "PERIODER=" P
3140 P0.=256/(P*2)
3150 U1.=(2*U)/P0.
3160 V.=-U-U1.
3170 D$=" "
3180 FOR T=0 TO 255
3190   T1=T1+1
3200   IF T1>P0. T1=0 : U1.=U1.*(-1)
3210   V.=V.+U1.
3220   D$=D$+FNH2$(V.)
3230 NEXT T
3240 RETURN
3250 !
3260 ! --------- E-FUNKTION
3270 !
3280 U=100
3290 C.=.5
3300 T1.=.2
3310 ; CUR(21,15) SPACE$(65)
3320 ; CUR(21,30) "E-FUNKTIONU-TOPP=" U "TIDSKONSTANT=" C.
3330 C1.=-T1.*10/(256*C.)
3340 D$=" " : E.=2.71828
3350 FOR T=0 TO 255
3360   V.=U*E.ü(T*C1.)
3370   D$=D$+FNH2$(V.)
3380 NEXT T
3390 RETURN
3400 !
3410 ! -------- BEARBETA DATA BERÄKNA MAX & MIN
3420 !
3430 FOR I=1 TO 256
3440   D(I)=FNH(MID$(D$,I*2-1,2))
3450 NEXT I
```

```
3460 M0=500 : M2=-500
3470 FOR I=1 TO 256
3480   IF D(I)<M0 M0=D(I) : M1=I
3490   IF D(I)>M2 M2=D(I) : M3=I
3500 NEXT I
3510 ; CHR$(12)
3520 ; : ; "BERÄKNADE VÄRDEN:"
3530 ; "MAXVÄRDE =" M2 " ENHETER" M2*G.(1)/127 " VOLT EFTER ";
3540 ; M3 " ENHETER" M3*G.(3)/25.6 " SEK."
3550 ; "MINVÄRDE =" M0 " ENHETER" M0*G.(1)/127 " VOLT EFTER ";
3560 ; M1 " ENHETER" M1*G.(3)/25.6 " SEK."
3570 ; : ; 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
3580 GET G$
3590 ; CHR$(12)
3600 RETURN
3610 !
3620 ! ========== OMVANDL AV HEX I ASCII TILL INTEGER
3630 !
3640 ! --------- ETT TECKEN
3650 !
3660 DEF FNH1(Ö$)=INSTR(1,"0123456789ABCDEF",Ö$)-1
3670 !
3680 ! --------- TVÅ TECKEN
3690 !
3700 DEF FNH2(Ö$)=FNH1(LEFT$(Ö$,1))*16+FNH1(RIGHT$(Ö$,2))
3710 !
3720 ! --------- TVÅ TECKEN TILL HELTAL
3730 !
3740 DEF FNH(Ö$)=SWAP%(FNH2(Ö$))/256
3750 !
3760 ! ----- 4 BIT TILL ASCII
3770 !
3780 DEF FNH1$(Ö)
3790   Ö$=CHR$((Ö AND 15)+48)
3800   IF Ö$>"9" RETURN CHR$(ASCII(Ö$)+7) ELSE RETURN Ö$
3810 FNEND
3820 !
3830 ! ----- EN BYTE TILL TVÅ ASCII
3840 !
3850 DEF FNH2$(Ö)
3860   Ö1$=FNH1$((Ö AND 255)/16)
3870   RETURN Ö1$+FNH1$(Ö)
3880 FNEND
3890 !
3900 ! ---------- DATA FÖR FRONTPANELEN
3910 !
3920 DATA BYTE,3&4,5&6,7&8,9&10,11&12, 13&14
3930 DATA 15&16, 17&18, 19&20, 21&22, 23&24, 25&26
3940 DATA COD,AT.A,MO.A,AT.B,MO.B,BA.T
3950 DATA MO.T,RE ,MO.D,SE.R,  TR+,DEL,DEL,
```

7. Övriga kort

I detta avsnitt skall vi undersöka tre intressanta kort ur 4680-serien, som kan användas tillsammmans med ABC80/800/DTC:

2006	Färg-video RAM
4117	Asynkront serieinterface
5070	Prototypkort med I/O-buss interface

7.1 2006 Färg-video RAM

Beskrivning av 2006 Färg-video RAM.

Med 2006 kan ABC80 anslutas till en färg- eller svart/vit TV-monitor. Om anslutningen sker till en färgmonitor finns det sju färger att välja bland: röd, grön, gul, blå, magenta (violett) och cyan (ljus-blå). För att få en effektiv kontroll över färgen och grafiken finns det särskilda standardiserade kontrolltecken. Teckenuppsättningen är anpassad till ASCII-koden, med 32 kontrolltecken, 96 alfanumeriska och 64 grafiska tecken. Denna teckenuppsättning håller samma standard som VIEWDATA (som används för text-tv).

Videoutgångar finns både för färgsignaler (röd/grön/blå) och för svart/vit utsignal. Kortet kan även anpassas till olika marknader genom byte av teckengeneratorn, (TROM).

Videominne:

Bildskärmen har en storlek på 24 (rader) x 40 (tecken). Texten på skärmen lagras i två stycken 1x4 kB statiska RAM (2114). Eftersom bilden endast kräver 960 byte minnesutrymme (24x40), finns det alltså 96 byte som inte används.

Teckengenerator:

Teckengeneratorn styrs av signaler från videominnet. Datasignalen består av 7-bitars data till teckengeneratorn, som definierar ett punktmatrismönster i en teckenmatris. En teckenmatris är 6 punkter bred och 10 linjer hög. Datasignalen till teckengeneratorn består även av 1 bit som indikerar cursor (blinkning). En punktkolumn lämnas mellan varje tecken och en linje lämnas mellan varje textrad. Alfanumeriska tecken genereras i en matris som är 5 punkter bred och 9 linjer hög. Det medför att det finns plats för underslängar på tecknen, som t.ex "g". Alfanumeriska tecken är avrundade, dvs. en halv punkt inskjutes före eller efter en hel punkt i en diagonallinje i matrisen. Denna rundning ger bästa resultatet med radsprång. Alla de 64 grafiska tecknen kodas som en 2x3 matris, som fyller hela 6x10 punktmatrisen. Grafiska tecken kan antingen vara sammanhängande eller separerade. Ett tecken per mikrosekund skrivs ut på skärmen, vilket innebär en punktfrekvens på 6 MHz. Tidsstyrningen sker med en 6 MHz-klocka som sitter på kortet.

Styrtecken:

32 styrtecken används för att sätta färg på tecken och bakgrund, välja blinkning, grafik, dubbel höjd m.m. Se sammanställning nedan.

Styrtecken för 2006 färg-video RAM:

ASCII-kod	Funktion	Kommentar
129	Röd text	Skriver ut röd text på hela raden eller till dess att ett nytt styrtecken skrivs ut.
130	Grön text	Samma som ovan men med grön text.

131	Gul text	Samma som ovan men med gul text.
132	Blå text	Samma som ovan men med blå text.
133	Magenta text	Samma som ovan men med magenta text.
134	Cyan text	Samma som ovan men med cyan text.
135	Vit text	Samma som ovan men med vit text.
136	Blink start	Får följande text att blinka.
137	Blink slut	Upphäver tecknet 136.
140	Normal höjd	Texten skrivs i normal höjd. 24 textrader.
141	Dubbel höjd	Skriver efterföljande text med dubbel höjd. Tecknet 140 upphäver 141.
145	Röd grafik	Sätter efterföljande tecken i röd grafisk mod. Versaler förändras ej av detta styrtecken.
146	Grön grafik	Som tecken 145, men sätter grön grafisk mod
147	Gul grafik	Som tecken 145, men sätter gul grafisk mod
148	Blå grafik	Som tecken 145, men sätter blå grafisk mod
149	Magenta grafik	Som tecken 145, men sätter magenta grafisk mod
150	Cyan grafik	Som tecken 145, men sätter cyan grafisk mod
151	Vit grafik	Som tecken 145, men sätter vit grafisk mod
153	Anslutande grafik	Grafiska tecknen i anslutning till varandra.
154	Separerad grafik	Grafiska tecken separerande från varandra.
156	Svart bakgrund	Texten skrivs mot svart bakgrund.
157	Ny bakgrund	Den senast valda teckenfärgen läggs som bakgrundsfärg.
158	Grafik över styrtecken	Normalt blir det ett mellanslag där det står ett styrtecken. Med styrtecken 158 erhålls färgväxlingarna kant i kant. Hålet fylls med samma grafiksymbol som står till vänster om styrtecknet ifråga.
159	Blankt styrtecken	Upphäver styrtecken 158.

INSTALLATION AV FÄRGVIDEO-RAM 2006

Byglingar

På kortet sitter det även 8 byglingar S1-S8, som skall ställas innan kortet kan tas i bruk. Se blockschema i fig. 7.1.

Bygel	Funktion
S1 MEMREQ*	Om systemet använder sig av dynamiska RAM skall S1 vara öppen för att refresh signalen inte skall störa bildskärmen. I ABC80 tillämpningar skall sålunda S1 vara öppen.
S2 FÄRG	Bygel S2 skall vara sluten och S3 skall vara öppen när man vill använda RGB utgångarna.
S3 SVART/VIT	Bygel S3 skall vara sluten och S2 skall vara öppen när man vill använda den svart/vita utgången (den blå utgången på kortet).
S4 POSITIV SYNC	Bygel S4 skall vara sluten och S5 skall vara öppen för positiv sync.
S5 NEGATIV SYNC	Bygel S5 skall vara sluten och S4 skall vara öppen för negativ sync.
S6 RADSPRÅNG	Bygel S6 skall vara sluten och S7 skall vara öppen för radsprång (Interlace). Detta ger en stabil bild på en monitor med lång efterlysningstid på bildröret.
S7 EJ RADSPRÅNG	Bygel S7 skall vara sluten och S6 skall vara öppen för att koppla bort radsprånget.
S8 LÄS-SKYDD	Bygel S8 skall vara öppen om man vill förhindra läsning från videominnet.

Val av adress

Adresseringen av 2006 fungerar på samma sätt som vanliga minneskort med den skillnaden att bygel 6 betyder här 1k. Se kapitel 1: "Adressering av I/O- och minnes-kort", figur 1.10. Om bygel 3,4 och 6 bryts får kortet basadressen 8k+16k+1k=25k. Om kortet skall användas med printer-PROM skall adressen sättas till 16k.

Montering av 2006 i expansionsenheten.

Slå av spänningen först. Placera därefter 2006 färgvideo-RAM i minnessidan av expansionsenheten.

Programmering av 2006.

Det lättaste sättet att programmera 2006 är att man öppnar kortet som en fil. Det förutsätts då att det finns ett printer-PROM i datorn. Filen öppnas som en vanlig fil.

```
10 OPEN "PR:R" AS FILE 3
```

När detta är gjort kan man skriva på skärmen som det vore en vanlig fil.

```
20 ; §3,"BYGG UT ABC80/800/DTC MED DATABOARD 4680"
```

Texten som skrivs ut på skärmen kan styras med kommandona CUR och CHR$(..). Detta gör att man kan styra texten precis som på ABC80 men med några viktiga skillnader.

```
30 ; §3,CHR$(12)
```

Detta kommando rensar skärmen.

```
40 ; §3,CUR(11,19)"MITT I SKÄRMEN"
```

Detta kommando skriver ut "MITT I SKÄRMEN" mitt på bildskärmen. Man kan även rita samma grafik på bildskärmen som på ABC80. Det finns dock ytterligare en grafiktyp på 2006 som inte finns på ABC80, s.k. separerande grafik, (ABC80 har "anslutande" grafik). Båda grafiktyperna har samma stil men med den skillnaden att den separerande grafiktypen har en linje eller en punkts anstånd mellan varje dot. Satserna SETDOT, CLRDOT och DOT finns dock ej men med en enkel subrutin kan man göra de kommandona. Ett exempel på detta återfinns i exempel 1. Man kan givetvis även använda POKE-kommandot och lägga texten direkt i bildminnet, vilket dock är betydligt krångligare än att öppna skärmen som en fil. POKE-kommandot är dock en nödvändighet när man skall göra "SETDOT", "CLR-DOT" och "DOT" kommandona.

Styrtecknen på 2006

Styrtecknen sänds ut med kommandot CHR$(..). Styrtecknen är tal mellan 129 och 160. Ett exempel på en sats som sänder iväg några styrtecken och text kan se ut som följer:

```
50 ; §3,CHR$(132,157,131)"STYRTECKNEN LÖSER DET"
```

Denna rad skriver ut "STYRTECKNEN LÖSER DET" i gult på blå bakgrund.

Exempel 1: SETDOT-, CLRDOT- och DOT-kommandon.

Eftersom kommandona SETDOT, CLRDOT och DOT inte finns i PRINTER-PROMET får man göra en egen rutin som utför dessa kommandon. I det exempel som visas här nedan ligger dessa rutiner i subrutiner utom DOT som ligger i en definition. Subrutinerna anropas med koordinaterna för punkten. Före anropet av subrutinerna måste variablerna X% och Y% ha värdet för koordinaten där det antingen skall ske en SETDOT eller CLRDOT. Ett anrop till de funktionerna ses här nedan.

```
30  REM ***** ANROP TILL SETDOT-FUNKTIONEN *****
40  REM
50  X%=30% : Y%=30% : GOSUB 1000

80  REM ***** ANROP TILL CLRDOT-FUNKTIONEN *****
90  REM
100 X%=30% : Y%=30% : GOSUB 2000
```

Anropet till DOT-funktionen kan se ut enligt nedan.

```
150 IF FND%(Y%,X%) THEN ; "PUNKTEN ÄR TÄND"
```

X% och Y% är koordinaterna för punkten. I dessa fall kan X% anta värden mellan 0-79 och Y% kan anta värden mellan 0-71. Skärmen ser alltså precis ut som ABC80-skärmen. I programmet nedan är alla definitioner klara varför det är klart för användaren att mellan rad 10 och 990 lägga in sitt program som skall göra SETDOT och CLRDOT. Behövs mer utrymme är det bara att flytta subrutinerna längre ner.

```
1    REM ***** DOT-FUNKTIONER *****
2    REM
3    DEFFNP%(Y%,X%)=Y%/3%*128%-Y%/3%/8%*984%+X%/2%+16384%
4    DEFFNB%(Y%,X%)=B%(Y%-Y%/3%*3%,X%-X%/2%*2%)
5    DEFFND%(Y%,X%)=PEEK(FNP%(Y%,X%)) AND FNB%(Y%,X%)
6    FOR Y%=0% TO 2% : FOR X%=0% TO 1%
7    READ B%(Y%,X%)
8    NEXT X% : NEXT Y%
9    DATA 1,2,4,8,16,32,64
1000 REM
1010 REM ***** SETDOT *****
1020 REM
1030 P%=FNP%(Y%,X%)
1040 POKE P%,(PEEK(P%) OR FNB%(Y%,X%))
1050 RETURN
2000 REM
2010 REM ***** CLRDOT *****
2020 REM
2030 P%=FNP%(Y%,X%)
2040 POKE P%,(PEEK(P%) AND NOT FNB%(Y%,X%))
2050 RETURN
```

Kommentarer till ovanstående program:

```
   3 : Funktion som beräknar adress i bildminnet.
   4 : Funktion som beräknar aktuell bit.
   5 : Motsvarar DOT(Y%,X%).
 6-9 : Läser in bitmönstret till matris B%.
1000 : Subrutinen SETDOT.
2000 : Subrutinen CLRDOT.
```

Exempel 2: Styrning av teckenstorlek, bakgrundsfärg och blinkning.

Detta exempel visar hur man kan styra teckenstorlek, bakgrundsfärg, teckenfärg och blinkning. Se styrkodstabellen ovan vid körning av programmet.

```
10  REM ***** TEXTSTYRNING *****
20  REM
30  ; CHR$(12%)
40  OPEN "PR:R" AS FILE 3%
50  ; : ; : ;
60  ; "BAKGRUNDSFÄRG (129-135)"; : INPUT B%
70  ; "FÄRGEN PÅ TEXTEN (129-135)"; : INPUT T%
90  ; "DUBBEL TECKEN STORLEK (J/N)"; : INPUT D$
100 ; "BLINKNING (J/N)"; : INPUT B$
110 D$=CHR$(ASC(D$) AND 95%) : REM GÖR SMÅ BOKSTÄVER STORA
120 B$=CHR$(ASC(D$) AND 95%) : REM GÖR SMÅ BOKSTÄVER STORA
130 ; §3%,CHR$(12%)
140 FOR I%=0% TO 23%
150 ; §3%,CUR(I%,0%)CHR$(B%,157%,T%)
160 NEXT I%
170 ; §3%,CUR(11,10);
180 IF D$="J" THEN ; §3%,CHR$(141%);
```

```
190 IF B$="J" THEN ; §3%,CHR$(136%);
200 ; §3%,"TEST TEXT FÖR FÄRG-TV"
210 GOTO 50
220 END
```

Kommentarer till ovanstående program:

```
40  : Öppnar 2006 som en fil.
150 : Tv-skärmen får bakgrundsfärg B% och textfärg T%.
180 : Om dubbel höjd skriv styrtecken 141 framför text.
190 : Om blinkning skriv styrtecken 136 framför text.
```

Exempel 3: Simulerad kärnklyvning med färg-video RAM 2006.

```
10 REM SIMULERAD KÄRNKLYVNING MED FÄRGVIDEO RAM 2006
20 REM
30 ; CHR$(12%)
40 DEF FNP%(Y%,X%)=Y%/3%*128%-Y%/3%/8%*984%+X%/2%+16384%
50 DEF FNB%(Y%,X%)=D%(Y%-Y%/3%*3%,X%-X%/2%*2%)
60 DEF FND%(Y%,X%)=PEEK(FNP%(Y%,X%)) AND FNB%(Y%,X%)
70 FOR Y%=0% TO 2% : FOR X%=0% TO 1%
80     READ D%(Y%,X%)
90 NEXT X% : NEXT Y%
100 OPEN 'PR:R' AS FILE 1%
110 ; §1%,CHR$(12%)
120 FOR I%=3% TO 20%
130   ; §1%,CUR(I%,1%) CHR$(158%,132%,157%,147%);
140 NEXT I%
150 T1%=180% : RESTORE 570 : RANDOMIZE
160 ON ERROR GOTO 220
170 READ A%,B%,C%
180 FOR I%=B% TO C%
190   Y%=A% : X%=I% : GOSUB 590 : REM SETDOT
200 NEXT I%
210 GOTO 160
220 R1%=55% : K1%=15%
230 IF INP(56%)=215% Y1%=-1% : X1%=0%
240 IF INP(56%)=216% Y1%=1% : X1%=0%
250 IF INP(56%)=193% X1%=-1% : Y1%=0%
260 IF INP(56%)=196% X1%=1% : Y1%=0%
270 IF INP(56%)=209% X1%=-1% : Y1%=-1%
280 IF INP(56%)=195% X1%=1% : Y1%=1%
290 IF INP(56%)=218% X1%=-1% : Y1%=1%
300 IF INP(56%)=197% X1%=1% : Y1%=-1%
310 R1%=R1%+Y1% : K1%=K1%+X1%
320 IF R1%<11% R1%=10% : Y1%=1%
330 IF R1%>59% R1%=60% : Y1%=-1%
340 IF K1%<11% K1%=10% : X1%=1%
350 IF K1%>79% K1%=79% : X1%=-1%
360 IF INP(56%)=211% X1%=0% : Y1%=0%
370 IF FND%(R1%,K1%) ; CHR$(7%) ELSE 440
380 R%=(RND*50+10%) : K%=(RND*69+10%)
390 P%=P%+1%
400 Y%=R% : X%=K% : GOSUB 590 : REM SETDOT
410 Y%=R%+1% : X%=K% : GOSUB 590 : REM SETDOT
420 Y%=R% : X%=K%+1% : GOSUB 590 : REM SETDOT
430 Y%=R%+1% : X%=K%+1% : GOSUB 590 : REM SETDOT
440 Y%=R1% : X%=K1% : GOSUB 590 : REM SETDOT
450 ; §1%,CUR(0%,0%) CHR$(129%) "POÄNG" P%
```

```
460 ; §1%,CUR(0%,20%) T1% " SEKUNDER KVAR "
470 T%=T%+1% : IF T%=9% T1%=T1%-1% : T%=0%
480 Y%=R1% : X%=K1% : GOSUB 650 : REM CLRDOT
490 IF T1%=0% 510
500 GOTO 230
510 ; §1%,CUR(21%,0%) "SPELET ÄR SLUT DU HADE" P% " POÄNG "
520 IF P%>H% H%=P% : ; §1%,"DU HADE HÖGRE ÄN TOP SCORE" : GOTO 540
530 ; §1%,"TOP SCORE " H%
540 ; §1%,"TRYCK PÅ RETURN"
550 GET G$ : IF G$=CHR$(13%) P%=0% : GOTO 110 ELSE 550
560 DATA 1,2,4,8,16,64
570 DATA 30,40,41,31,39,42,32,38,43,33,39,42,34,40,41
580 REM
590 REM SETDOT Y%,X%
600 REM
610 P1%=FNP%(Y%,X%)
620 POKE P1%,(PEEK(P1%) OR FNB%(Y%,X%))
630 RETURN
640 REM
650 REM CLRDOT Y%,X%
660 REM
670 P1%=FNP%(Y%,X%)
680 POKE P1%,(PEEK(P1%) AND NOT FNB%(Y%,X%))
690 RETURN
```

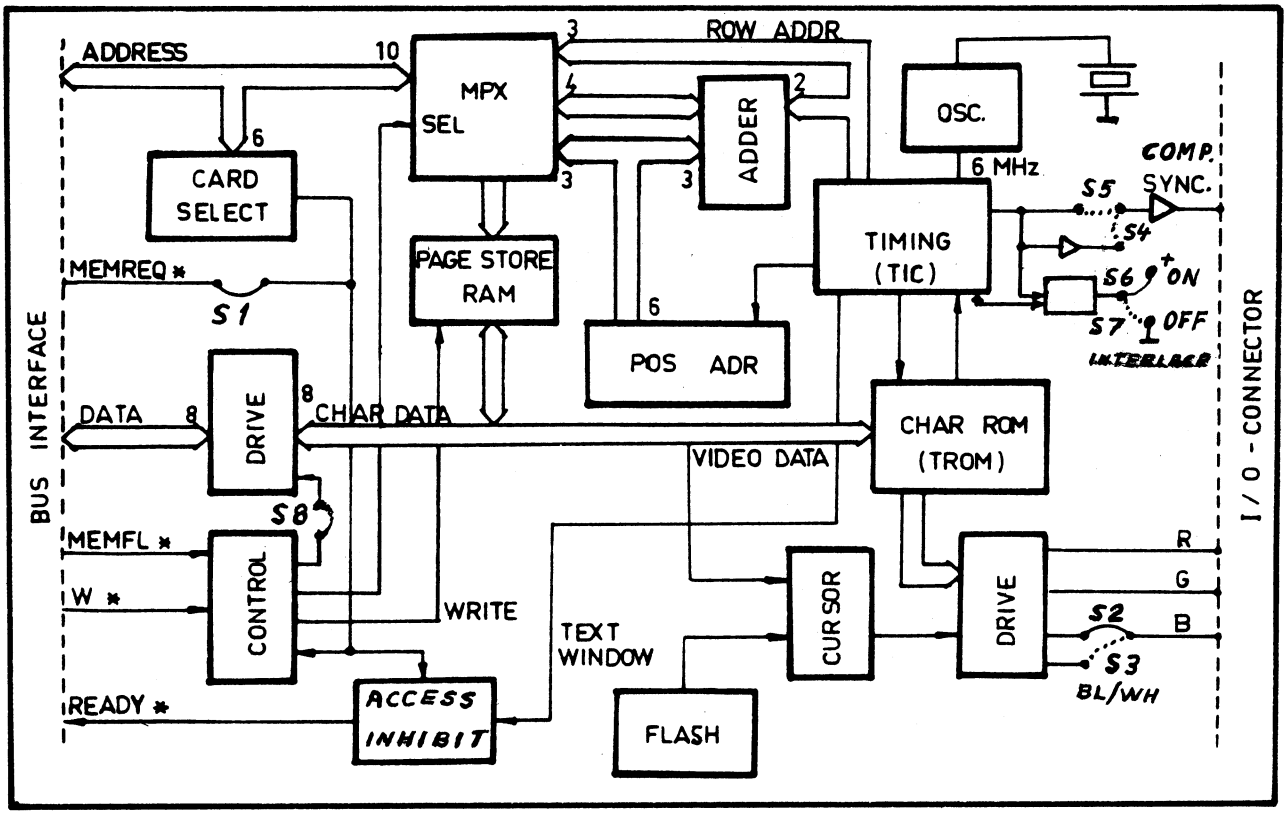


Fig. 7.1 Blockschema för 2006 färg-video RAM.

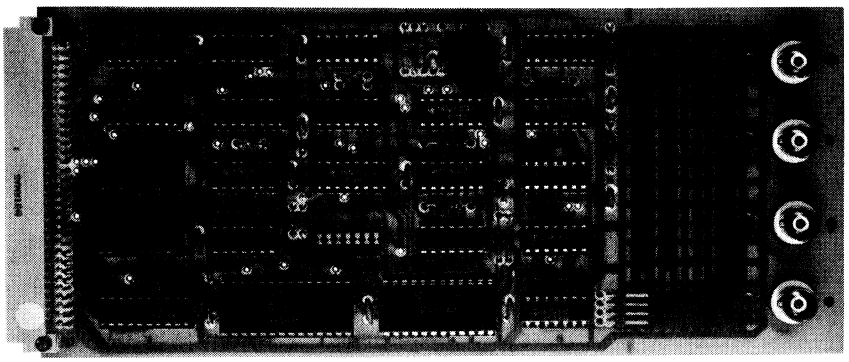


Fig. 7.2 2006 Färg-video RAM.

Fig. 7.3 Teckenuppsättning

SAA 5050 CHARACTER SET

Data bits	b_7	0	0	0	0	1	1	1	1
	b_6	0	0	1	1	0	0	1	1
b_4 b_3 b_2 b_1	b_5	0	1	0	1	0	1	0	1
0 0 0 0				0	P	―	p		
0 0 0 1		Alpha^{n} Red	Graphics Red	1	A	Q	a	q	
0 0 1 0		Alpha^{n} Green	Graphics Green	2	B	R	b	r	
0 0 1 1		Alpha^{n} Yellow	Graphics Yellow	3	C	S	c	s	
0 1 0 0		Alpha^{n} Blue	Graphics Blue	4	D	T	d	t	
0 1 0 1		Alpha^{n} Magenta	Graphics Magenta	5	E	U	e	u	
0 1 1 0		Alpha^{n} Cyan	Graphics Cyan	6	F	V	f	v	
0 1 1 1		Alpha^{n} White	Graphics White	7	G	W	g	w	
1 0 0 0		Flash	Conceal Display	8	H	X	h	x	
1 0 0 1		Steady	Contiguous Graphics	9	I	Y	i	y	
1 0 1 0			Separated Graphics	:	J	Z	j	z	
1 0 1 1				;	K	←	k	¼	
1 1 0 0		Normal Height	Black Background	<	L	½	l	‖	
1 1 0 1		Double Height	New Background	=	M	→	m	¾	
1 1 1 0			Hold Graphics	>	N	↑	n	÷	
1 1 1 1			Release Graphics	?	O	#	o	■	

Black represents display colour, and white represents background.

SAA 5052 CHARACTER SET

Bits b_7 b_6 b_5 / b_4 b_3 b_2 b_1 / Col / Row	0_0	0_1	0_2	0_3	1_0	1_1	1_2	1_3
	0	1	2	3	4	5	6	7
0 0 0 0 0				Ø	É	P	é	p
0 0 0 1 1	Alpha^{n} Red	Graphics Red	!	1	A	Q	a	q
0 0 1 0 2	Alpha^{n} Green	Graphics Green	"	2	B	R	b	r
0 0 1 1 3	Alpha^{n} Yellow	Graphics Yellow	#	3	C	S	c	s
0 1 0 0 4	Alpha^{n} Blue	Graphics Blue	¤	4	D	T	d	t
0 1 0 1 5	Alpha^{n} Magenta	Graphics Magenta	%	5	E	U	e	u
0 1 1 0 6	Alpha^{n} Cyan	Graphics Cyan	&	6	F	V	f	v
0 1 1 1 7	Alpha^{n} White **	Graphics White	'	7	G	W	g	w
1 0 0 0 8	Flash	Conceal Display	(	8	H	X	h	x
1 0 0 1 9	Steady **	Contiguous Graphics **	)	9	I	Y	i	y
1 0 1 0 10		Separated Graphics	*	:	J	Z	j	z
1 0 1 1 11			+	;	K	Ä	k	ä
1 1 0 0 12	Normal Height **	Black Background **	,	<	L	Ö	l	ö
1 1 0 1 13	Double Height	New Background	-	=	M	Å	m	å
1 1 1 0 14		Hold Graphics	.	>	N	Ü	n	ü
1 1 1 1 15		Release Graphics **	/	?	O	_	o	■

Control characters shown in columns 0 and 1 are normally displayed as spaces.

** These control characters are presumed before each row begins

Character rectangle

Black represents display colour.

White represents background.

7.2 4117 UART Asynkront serieinterface

Beskrivning av 4117 UART Asynkront serieinterface.

4117 är ett asynkront serieinterface som är anpassat till europeisk V24 standard, (RS 232 C). Kortet kan anslutas till utrustning som använder serieöverföring t.ex skrivare, andra datorer, modem, och vissa mätinstrument. Överföringshastighten är valbar mella 75 och 19200 baud. Sändnings- och mottagningshastigheten behöver inte vara lika utan dessa kan väljas olika. Man kan även välja mellan 5-8 bitars ordlängd, jämn eller udda paritet eller ingen alls samt en eller två stoppbitar (1,5 stoppbit vid 5 bitars ordlängd). Man kan även koppla en extern klocka till 4117 om så önskas. 4117 UART-kort skall helst användas tillsammans med PRINTER-PROM. Detta PROM sitter som standard i ABC800/DTC men ej i ABC80 där det får beställas separat, (om man inte har Luxors flexskivenhet ABC där det också ingår som standard). Genom att använda detta PROM slipper man ladda in en printerrutin varje gång kortet skall användas. 4117 är programkompatibel med tidigare UART-kort 4017 och 4017/1.

INSTALLATION AV UART-KORT 4117.

Val av adress.

För att bestämma adress för kortet se kapitel 1: "Adressering av I/O- och minneskort". Skall kortet användas tilsammans med printer-PROM skall adressen vara 60.

Bygling av överföringshastighet.

Överföringshastigheten väljs med byglingarna i position 5B. Hastigheten för sändning väljs med byglingarna mellan rad A och B. Mottagningshastigheten väljs med byglingarna mellan rad C och D. Kolumn 1-9 väljer intern klocka 19200-75 baud medan kolumn 10 väljer extern klocka, som i så fall skall anslutas till stift 20B på I/O-kontakten eller, om printerkabeln används, till stift 17 på Cannon-kontakten.

Bygling av paritet och val av antal stoppbitar.

Byglingen utföres på position 5D med byglingarna J1-J5.

J1-0 : Paritet genereras ej och kontrolleras inte heller.
J1-1 : Paritet genereras och kontrolleras.
J2-0 : Jämn paritet.
J2-1 : Udda paritet.
J3-0 : Två stoppbitar. (1.5 om 5 bitars ordlängd.)
J3-1 : En stoppbit.
J4 : Ordlängd, se tabell nedan.
J5 : Ordlängd, se tabell nedan.

J4	J5	Ordlängd
1	1	5 bitar
0	1	6 bitar
1	0	7 bitar
0	0	8 bitar

I tabellen ovan, för byglar J1-J5, betyder:

1 = Bygling monterad.
0 = Ingen bygling.

Val av CTS-fas och sändstopp.

Valet görs med byglingarna J6 och J7 i position 5C.

J6-0 : Ej sändstopp.
J6-1 : CTS ger sändstopp.
J7-0 : CTS hög ger sändstopp.
J7-1 : Inverterad CTS. 0 V ger sändstopp.

Märk att CTS används normalt för avkänning av DTR-signal. Exempelvis kan en "printer busy" signal stoppa sändningen från 4117.

Avbrott

Ring Signal kan sätta en vippa för att generera avbrott. Ring Signal selektiv interrupt enable måste vara valt (kommando C4). Anslut bygling J8 i position 3B för denna funktion. Observera att bakplanet måste vara byglat för interrupt för om denna funktion skall användas.

Anslutning av yttre enheter.

Anslutning av yttre enheter till kortet sker på I/O-kontakten. I/O-kontakten är den som sitter närmast lysdioden.

Montering av 4117 i expansionsenheten.

Slå av spänningen först. Kortet vänds så attt I/O-kontakten kommer utåt och komponentsidan åt höger. 4117 skall placeras i I/O-delen på expansionsenheten.

Kontroll av adressen.

Efter monteringen av kortet i expansionsenheten bör man kontrollera att kortet byglats till rätt adress, genom att skriva:

OUT 1,<Kortadress>

<Kortadress> avser den adress som kortet har getts. För att testa kort 4117 ge kommandot:

OUT 1,60

Om byglingen är riktig skall lysdioden nere till höger på kortet tändas. Om därefter annat kort väljes, med t.ex. kommando OUT 1,0, skall lysdioden släckas.

Kommandon till 4117

Basic	Aktiverad signal	Funktion
INP (7)	RST	Nollställer alla I/O-kort. Exempel: Kommando 10 Slask=INP(7) bör inleda varje program som använder I/O-kort.
OUT 1,<Kortval>	CS	Väljer kort med adressen <Kortval>. Exempel: 20 OUT 1,60 väljer kortet med adressen 60.
OUT 0,<Data>	OUT	Sänder ut data. Får göras när bit 1 i status-registret är 0. Exempel: 30 OUT 0,65 sänder ut 65 till den yttre enheten.
INP (0)	INP	Läser åtta bitar från den yttre enheten. Exempel 40 Data=INP(0) Läser in åtta bitar till <Data>.
INP (1)	STAT	Läser in status från kortet. Exempel: 50 Status=INP(1) Läser in åtta status-bitar till <Status>. Bitarna i <Status> betyder: D0: DR READY interrupt, aktiv låg. D1: TBR EMPTY, =0 när data får sändas. D2: TBR EMPTY interrupt, aktiv låg. D3: CTS/DCD aktiv låg eller hög beroende på bygel J7. D4: REC SPEC 3 D5: Ring Signal, aktiv låg. D6: Error. 0 vid Parity, Framing eller Overrun fel. D7: DR READY, =0 då tecken finns att hämta.
OUT 3,<Data>	C2	Aktiverar signaler. Exempel: 60 OUT 3,240 aktiverar signalerna: TTL OUT 2, RTS, SRTS och DTR. D4: TTL OUT 2 D5: RTS D6: SRTS D7: DTR
OUT 4,0	C3	Nollställer kort. Motsvarar signalen RST men endast på detta kort.
OUT 5,<Data>	C4	Aktiverar Interrupt. Exempel: 80 OUT 5,240 aktiverar signalerna: TTL OUT 4, RING SIGNAL Interrupt enable, RECEIVE Interrupt enable och SEND Interrupt enable. D4: TTL OUT 4 D5: RING SIGNAL Interrupt enable. D6: RECEIVE Interrupt enable. D7: SEND Interrupt enable.

Programmering av 4117.

Sändning av data.

1. Välj kort (CS).
2. Vänta till dess att mottagarna är klara.
3. Sänd data. (OUT)
4. Läs status. (STAT) Om D7=0 gå till punkt 2 annars till 4.

Mottagning av data.

1. Välj kort.
2. Läs status (STAT). Om D7=0 gå vidare annars punkt 2.
3. Läs data. (INP)
4. Gå till punkt 2

Exempel på användning av 4117 UART-kort:

Om man använder printer-PROM så har man en rutin som sköter alla signaler och man behöver då bara bry sig om vad man skall sända över till mottagaren. Om man till exempel vill lista ut ett program på en printer, som är kopplad till UART-kortet skriver man:

```
LIST PR:U+parametrar
```

Parametrarna talar om för datorn vilken bredd och längd man har på papperet plus fakta om skrivaren. Vad alla parametrarna betyder återfinns i bruksanvisningen för printer-PROM. I exemplen avänds en Epson skrivare och parametrarna är ställda därefter.

```
10  ! * SÄNDNING AV DATA MED 4117 OCH PRINTER-PROM *
20  !
30  OPEN "PR:USA32C72.5" AS FILE 1
40  FOR I=65 TO 93
50  ; §1,CHR$(I);
60  NEXT I
```

Pogrammet ovan skriver ut alfabetet på skrivaren.

```
10  ! * SÄNDNING AV DATA MED 4117 UTAN PRINTER-PROM *
20  !
30  Slask=INP(7)
40  OUT 1,60
50  A=13 : GOSUB 120
60  A=10 : GOSUB 120
70  FOR A=65 TO 93
80  GOSUB 100
90  NEXT A
100 END
110 !
120 ! ***** SÄND RUTIN *****
130 !
140 IF (INP(1) AND 2)=0 THEN OUT 0,A ELSE 140
150 RETURN
```

Programmet ovan utför samma sak som det första programmet men utan printer-PROM. På rad 120 väntar datorn på att ett tecken kan sändas.

```
10  ! ***** MOTTAGNING AV TECKEN MED 4117 *****
20  !
30  Slask=INP(7)
40  OUT 1,60
50  IF INP(1)<128 THEN A=INP(0) ELSE 40
60  ; CHR$(A)
70  GOTO 50
```

Programmet ovan läser in ett tecken från enheten som är ansluten till 4117 kortet. På rad 50 väntar datorn till dess att det finns ett tecken att hämta.

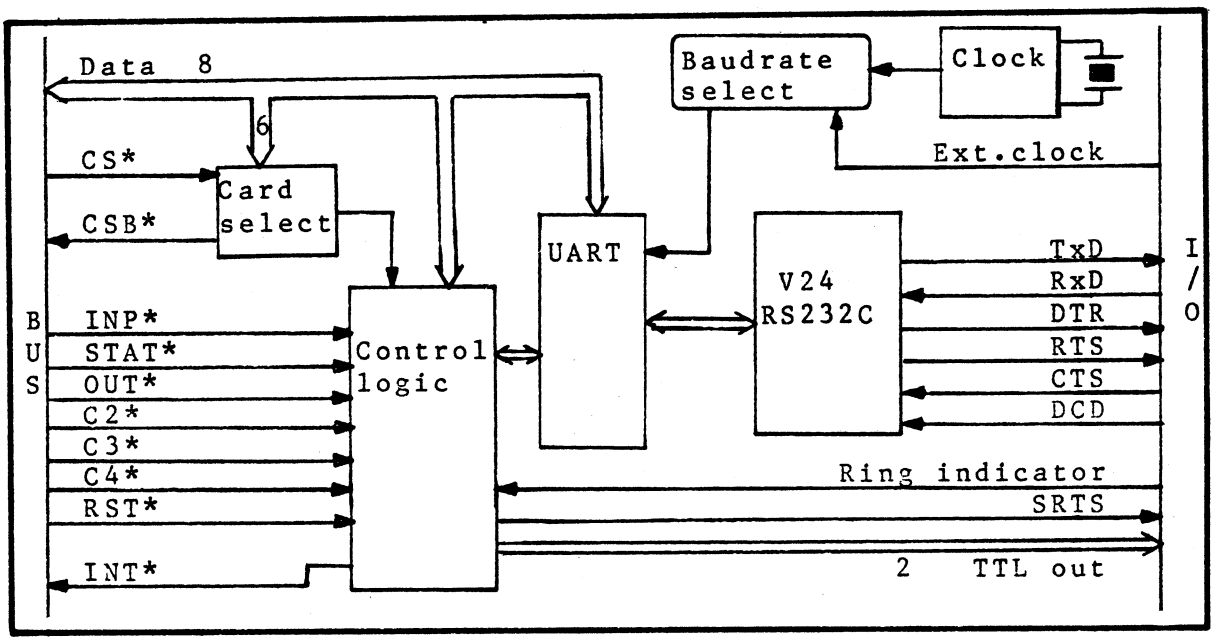


Fig. 7.4 Blockschema för 4117 UART-kort.

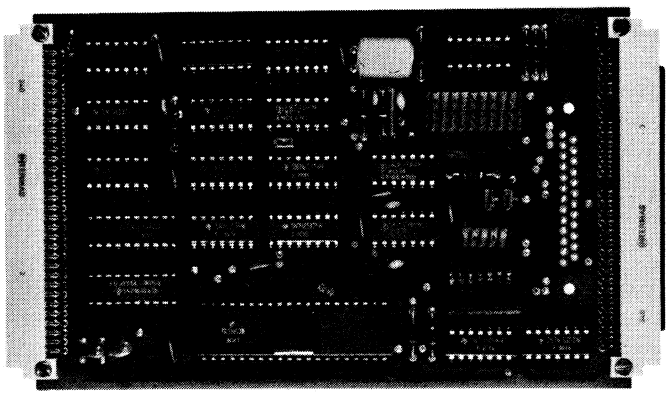


Fig. 7.5 4117 UART-kort.

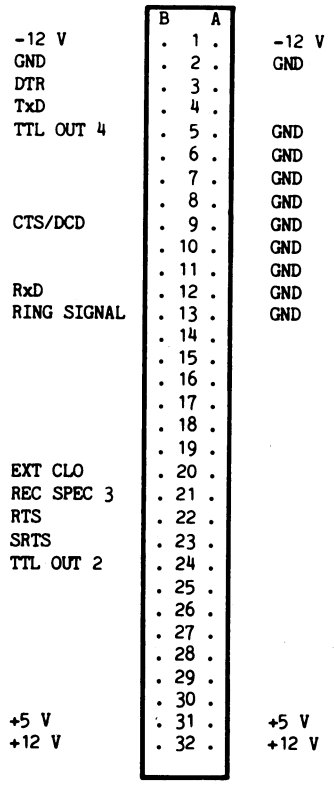


Fig. 7.6 Stiftlayout för I/O-kontakt.

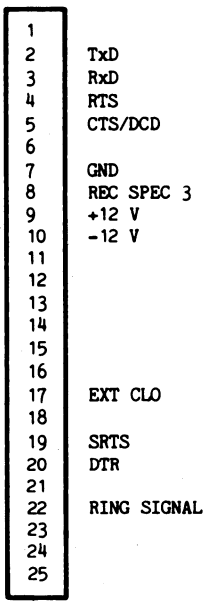


Fig. 7.7 Stiftlayout för Cannon-kontakt.

7.3 Prototypkort med I/O-buss interface

Beskrivning av 5070 prototypkort.

5070 erbjuder genom sin färdiga anpassning till 4680 en snabb väg för användaren som vill konstruera ett eget I/O-kort som ej ingår som standard i 4680 systemet.

Kortet innehåller bussanpassning för samtliga signaler. OP0-OP7 signalerna finns också framdragna på kortet och kan utnyttjas för internt samarbete mellan flera kort. Busssignalerna beskrivs i systemmanualen, där även den erforderliga drivningen och belastningen står beskriven. Kortet är förberett för virning på komponentsidan, där användaren disponerar virstift för anslutning av egen kretslogik till disponibla stift i respektive kontaktdon samt för anslutning, inne på kortet, till samtliga bussignaler. Kortet har plats för (12 + 4) 16 poliga IC-socklar. Fyra av dem kan utnyttjas för flexibelt polantal. Den färdiga konstruktionen bör prövas med hjälp av den I/O-testenhet som ingår i 4680-serien.

Kortet har standardformat för användning i expansionsenheten och är utrustad med standard kontaktdon för anslutning till bakplan och I/O.

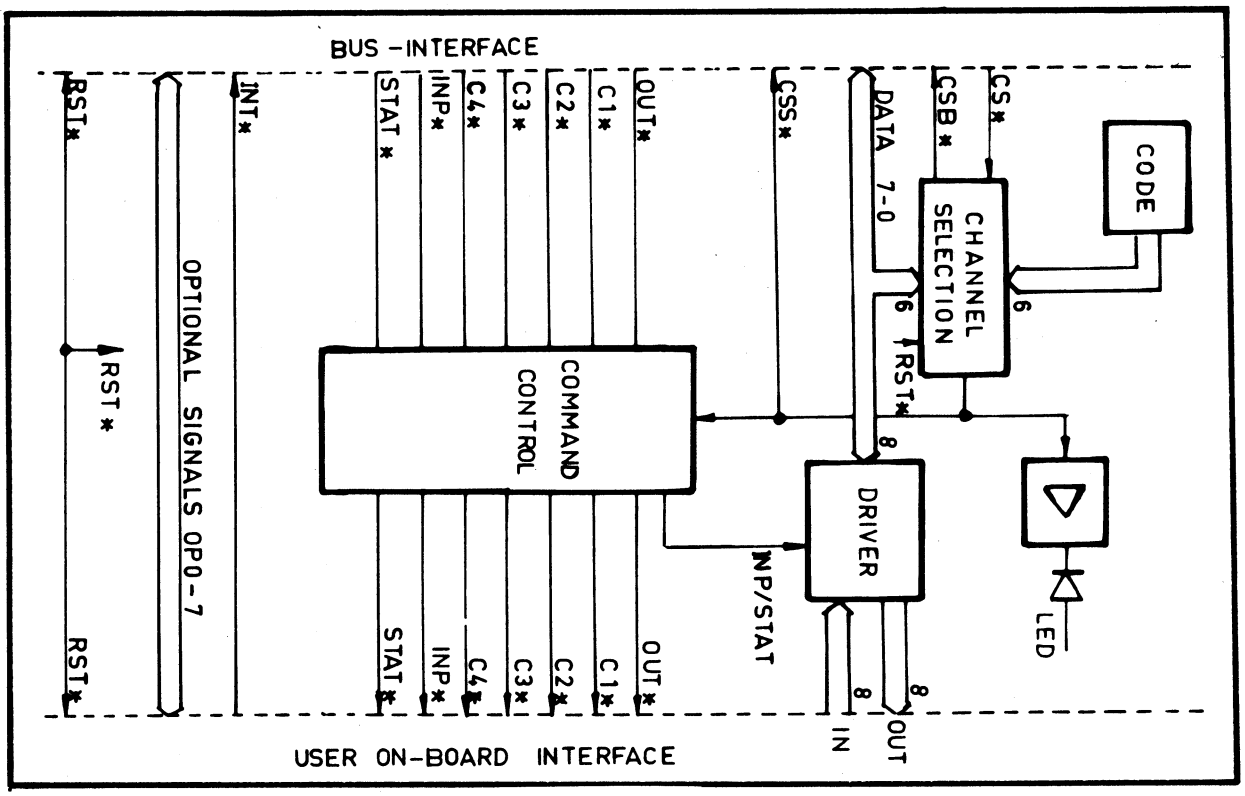


Fig. 7.8 Blockschema för prototypkort 5070.

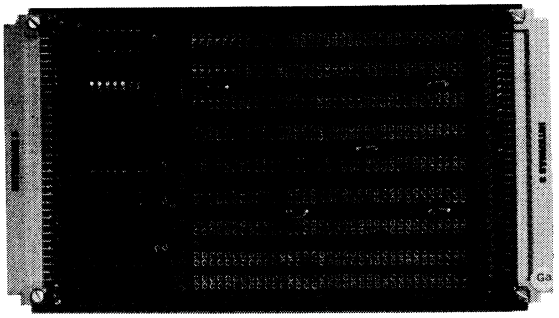
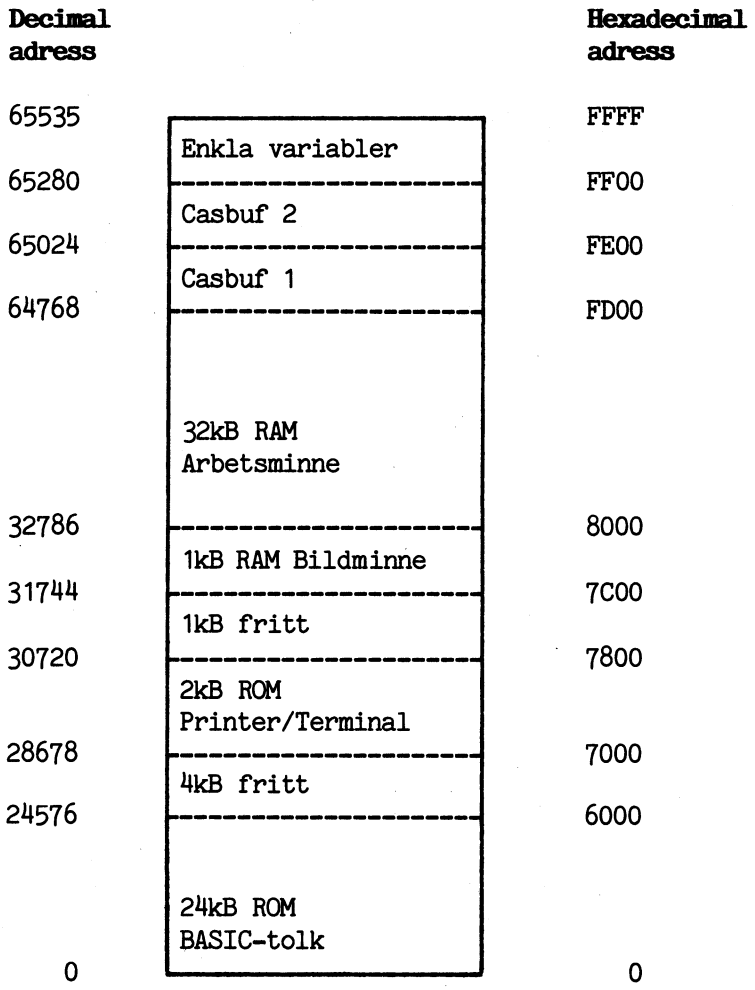


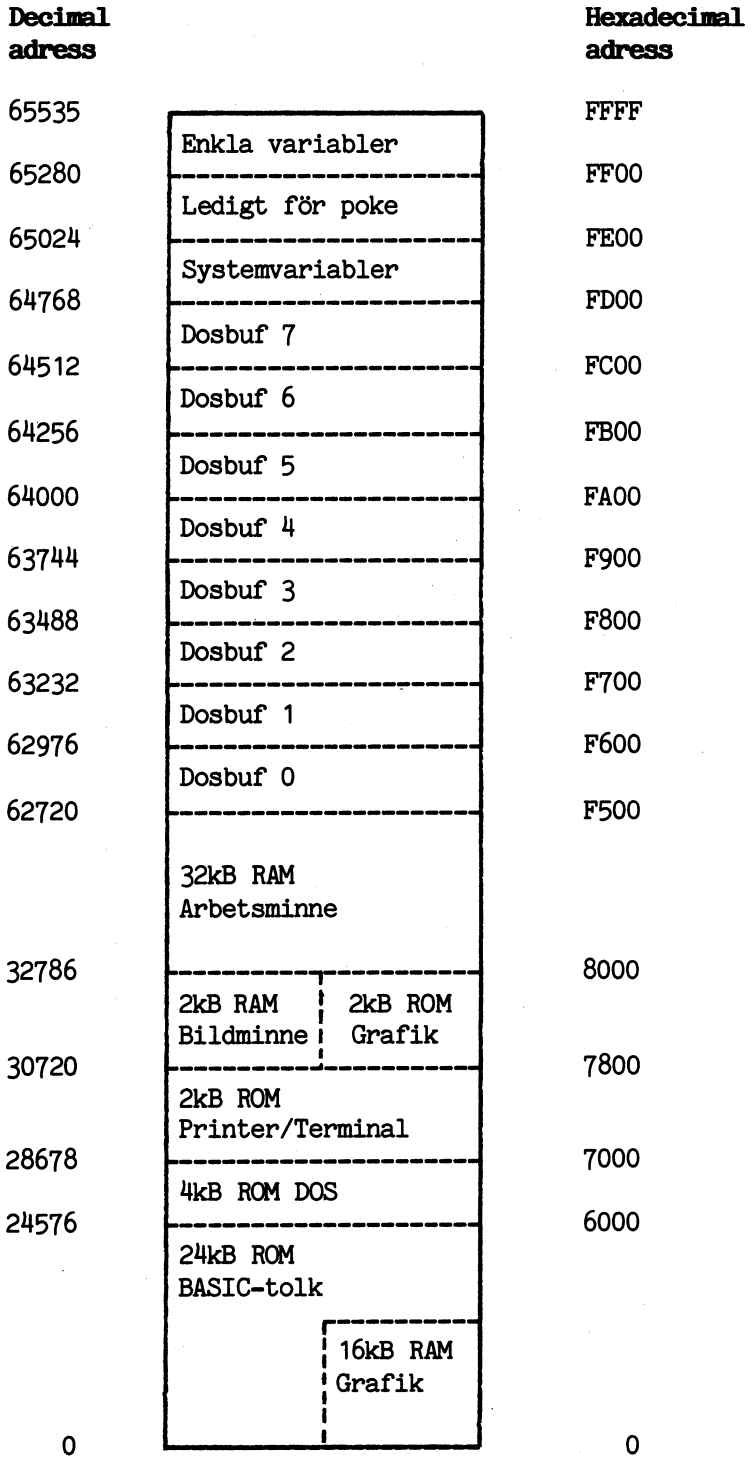
Fig. 7.9 Prototypkort 5070.

APPENDIX A

Minneskarta för ABC800 C, utan flexskivminne



Minneskarta för ABC800 MHR, med flexskivminne



APPENDIX B

Bygling för ABC80 interrupt

	Bygel				
	B1	B2	B3	B4	B5
Kortplats K1	X	-	-	-	-
" K2	X	X	-	-	-
" K3	X	X	X	-	-
" K4	X	X	X	X	-

X = Bygel flyttas till slutet läge
\- = Bygel flyttas till brutet läge

B1 är BRUTEN i det nedre läget, medan B2-B5 är BRUTNA i det högra läget
OBS! För att undvika kortslutning måste **minst en** bygel vara bruten.

Fig. B.1 Byglingstabell för expansionsenhet, (ABC80).

Stiftlayout för buss-, I/O- och minneskortkontakt hos ABC80

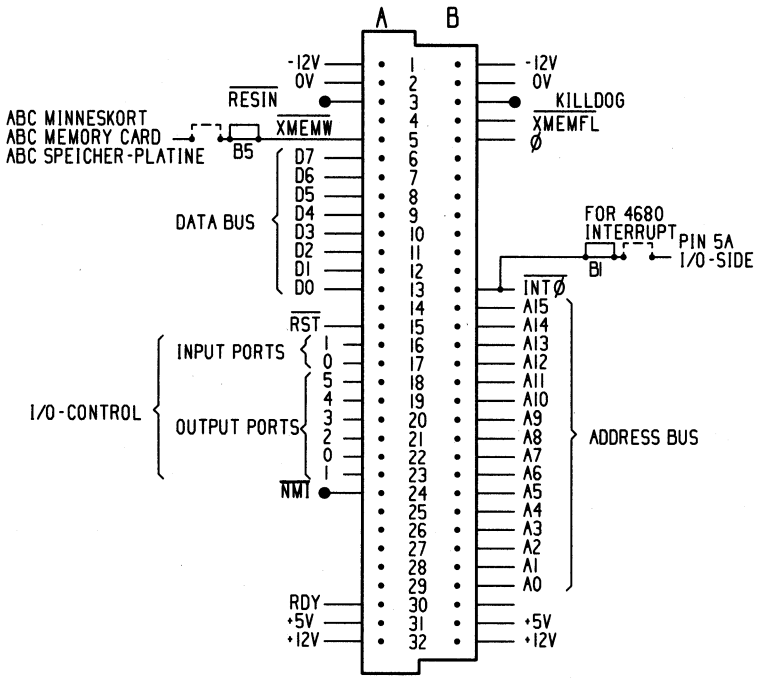


Fig. B.2 Stiftlayout för busskontakten från ABC80. (Motsvarar fig. 1.6)

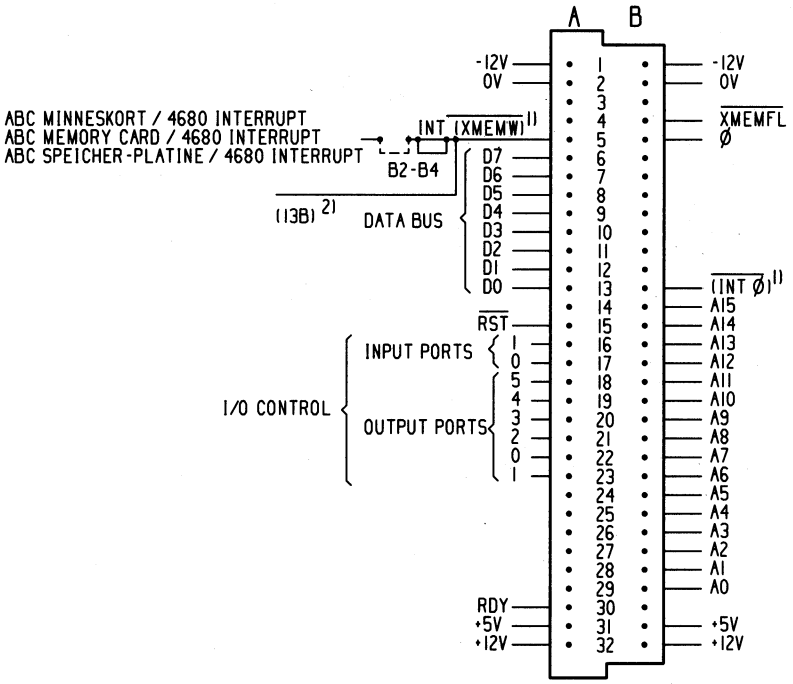


Fig. B.3 Stiftlayout för kontakt på I/O-sida, (motsvarar fig. 1.7).

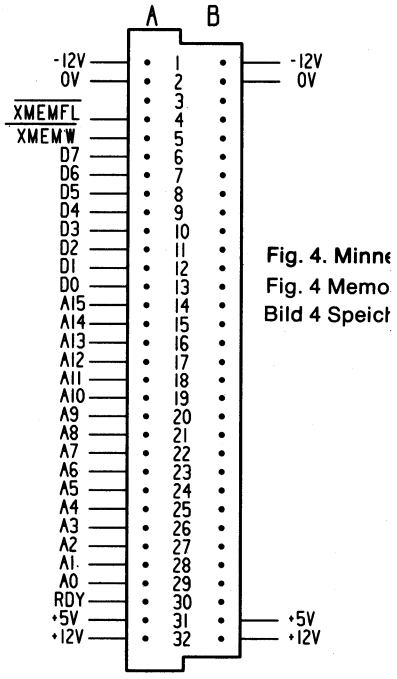


Fig. B.4 Stiftlayout för kontakt på minnes-sida, (motsvarar fig. 1.8).

APPENDIX C

ABC80 Program

De program som beskrivits i det föregående är avsedda för ABC800/DTC. Nedan följer motsvarande program för ABC80.

********** Kapitel 2 - Talsystem **********

Avsnitt 2.3 Omvandling mellan de decimala och binära talsystemen

Exempel 1: Program för omvandling mellan de decimala och binära talsystemen

```
10  REM ***** OMVANDLING *****
20  REM
30  REM OMVANDLING AV DECIMALTAL TILL
40  REM BINÄRTAL OCH BINÄRTAL TILL
50  REM DECIMALTAL
60  REM
70  ; CHR$(12%)
80  ; TAB(10%) " TALOMVANDLING "
90  ; CUR(5%,3%) "1. OMVANDLA DECIMALTAL TILL BINÄRTAL"
100 ; CUR(7%,3%) "2. OMVANDLA BINÄRTAL TILL DECIMALTAL"
110 ; CUR(9%,3%) "3. AVSLUTA"
120 ; CUR(11%,5%) "VAD ÖNSKAS"; : INPUT S%
130 IF S%<1% OR S%>3% GOTO 10
140 ON S% GOTO 180,220,260,10
150 REM
160 REM DECIMALT-BINÄRT
170 REM
180 GOSUB 290 : ; CUR(19%,5%) B$ : GOTO 550
190 REM
200 REM BINÄRT-DECIMALT
210 REM
220 GOSUB 410 : ; CUR(19%,5%) D$ : GOTO 550
230 REM
240 REM AVSLUTA
250 REM
260 ; CHR$(12%)
270 END
280 REM
290 REM SUBRUTINEN DEC-BIN
300 REM
310 ; CUR(15%,0%) SPACE$(80%)
320 ; CUR(15%,5%) "ANGE DECIMALTALET (MAX 65535)";
330 INPUT D%
340 B$=" "
350 FOR I%=15% TO 0% STEP -1%
360   B$=B$+CHR$((D% AND 2%üI%)/2%üI%+48%)
370 NEXT I%
380 B$="TALET BLEV "+B$
390 RETURN
400 REM
410 REM SUBRUTINEN BIN-DEC
420 REM
430 ; CUR(15%,0%) SPACE$(80%)
```

```
440 ; CUR(15%,0%) "ANGE BINÄRTALET MAX 1111 1111 1111 1111 ";
450 INPUT B$
460 A%=LEN(B$) : D%=0%
470 FOR I%=1% TO A%
480   P%=INSTR(I%,B$,"1")
490   D%=D%+2ü(A%-P%)
500   I%=P%
510 NEXT I%
520 D$="TALET BLEV "+NUM$(D%)
530 RETURN
540 REM
550 REM FLER TAL
560 REM
570 ; CUR(15%,0%) SPACE$(80%)
580 ; CUR(15%,5%) "FLER TALOMVANDLINGAR (J/N) <Ja>";
590 INPUT S$
600 IF S$=" " GOTO 10
610 S$=CHR$(ASCII(S$) AND 95%)
620 IF S$="J" GOTO 10
630 GOTO 260
```

********** Kapitel 3 - Digital signalering **********

Avsnitt 3.1 Digital signalering med I/O-kort 4005 och 4006

Exempel 1: Styrning av utgångarna hos I/O-kort 4005 och 4006

```
10  REM STYRNING AV UTSIGNALERNA PÅ KORT 4005 OCH 4006
20  REM STYRNINGEN SKER INDIVIDUELLT
30  REM EN UTGÅNG PÅVERKAS PER GÅNG
40  REM
50  REM FUNKTIONERNA
60  REM
70  DEF FNN%(S%,P%)=S% AND (65535% AND 2%üP%)
80  DEF FNE%(S%,P%)=S% OR 2üP%
90  REM
100 Z%=INP(7%)
110 OUT 1%,2% : REM KORTVAL FÖR ATT VÄLJA 4006 BYT UT 2%
    MOT 3%
120 ; CHR$(12%)
130 ; "VILKEN GRUPP SKALL LÄSAS";
140 INPUT G%
150 IF G%<1% OR G%>4% GOTO 130
160 IF G%=1% G%=0%
170 ; "VIKEN UTGÅNG I GRUPPEN 0-7";
180 INPUT P%
190 IF P%<0% OR P%>7% GOTO 170
200 ; "ETTSTÄLLNING ELLER NOLLSTÄLLNING AV PORTEN";
210 INPUT L%
220 IF L%<0% OR L%>1% GOTO 200
230 IF L%=0% THEN S%=FNN%(S%,P%)
240 IF L%=1% THEN S%=FNE%(S%,P%)
250 OUT G%,S%
260 GOTO 120
270 END
```

Avsnitt 3.1 Digital signalering med I/O-kort 4005 och 4006

Exempel 2: Läsning av insignaler med I/O-kort 4005 och 4006

```
10  REM LÄSNING AV EN INGÅNG PÅ KORTEN 4005 OCH 4006
20  REM
30  REM
40  REM FUNKTIONEN
50  REM
60  DEF FNI%(P%)=(INP(0%) AND 2%üP%)/2%üP%
70  REM
80  Z%=INP(7%)
90  OUT 1%,2% : REM TÄNK PÅ ATT ADDERA 128 OM GRUPP 2
    PÅ 4006 SKALL ANVÄNDAS
100 REM
110 ; CHR$(12%)
120 ; "VILKEN INGÅNG SKALL LÄSAS (0-7)";
130 INPUT P%
140 IF P%<0% OR P%>7% GOTO 120
150 ; : ; : ; "INGÅNGEN ÄR ";
160 IF FNI%(P%)=0% THEN ; "NOLLSTÄLLD" : ; : ;
170 IF FNI%(P%)=1% THEN ; "ETTSTÄLLD" : ; : ;
180 GOTO 120
190 END
```

Avsnitt 3.2 Digital signalering med I/O-kort 4008/4011 och 4095 OPTO-kort

Exempel 1: Rinnande ljus med 4095

```
10  REM RINNANDE LJUS MED HJÄLP AV 4095
20  REM
30  Z%=INP(7%)
40  OUT 1%,7%
50  REM
60  REM EVIGHETS LOOP
70  REM
80  FOR L%=0% TO 7%
90  IF L%=0% THEN OUT 0%,2ü0%+2ü7%
100 IF L%>0% THEN OUT 0%,2%üL%+2%ü(L%-1%)
110 FOR V=1 TO 200 : NEXT V
120 NEXT L%
130 GOTO 80
140 END
```

Avsnitt 3.2 Digital signalering med I/O-kort 4008/4011 och 4095 OPTO-kort

Exempel 2: Inläsning av data från optoingångarna på 4008/4011

```
10  REM LÄSNING AV EN INGÅNG
20  REM
30  Z%=INP(7%)
40  REM
50  REM DEFINITION SOM MASKER UT EN BIT
60  REM
70  DEF FNM%(P%)=(INP(0%) AND 2ü(P%-1%))/2%ü(P%-1%)
80  REM
90  ; CHR$(12%)
100 FOR L%=1% TO 8%
110   OUT 1%,4%
```

```
120    B1$=B1$+NUM$(FNM%(L%))
130    OUT 1%,4%+128%
140    B2$=B2$+NUM$(FNM%(L%))
150 NEXT L%
160 ; : ; "GRUPP 1 :"; : ; B1$
170 ; : ; "GRUPP 2 :"; : ; B2$
180 ; : ; "TRYCK PÅ EN TANGENT FÖR NY MÄTNING";
190 GET G$
200 GOTO 100
210 END
```

Avsnitt 3.3 Digital signalering med I/O-kort 4085

Exempel 1: Styrning av utgångarna A och B på I/O-kort 4085

```
10  REM STYRNING AV DE TVÅ UTSIGNALERNA PÅ 4085
20  REM
30  Z%=INP(7%)
40  OUT 1%,6%
50  REM
60  ; : ; "UTSIGNAL A="; : ; L1%
70  ; : ; "UTSIGNAL B="; : ; L2%
80  ; : ; : ; "SIGNAL A SKALL STÄLLAS TILL";
90  INPUT L1%
100 IF L1%<0% OR L2%>1% THEN 80
110 ; : ; "SIGNAL B SKALL STÄLLAS TILL";
120 INPUT L2%
130 IF L2%<0% OR L2%>1% THEN 110
140 OUT 0,4*L1%+8*L2%
150 GOTO 60
160 END
```

Avsnitt 3.3 Digital signalering med I/O-kort 4085

Exempel 2: Läsning och presentation av 32 ingångar

```
10  REM LÄSNING AV 32 INSIGNALER MED 4085
20  REM
30  Z%=INP(7%)
40  OUT 1%,6%
50  ; CHR$(12%)
60  ; "VILKEN GRUPP"; : INPUT G%
70  IF G%<0% OR G%>3% GOTO 60
80  OUT 0%,G%
90  ; : ; "DECIMALTALET AV INLÄSNINGEN" INP(0%)
100 ; "BINÄRTALET: ";
110 FOR I%=0% TO 7%
120 . .; (INP(0%) AND 2%üI%)/2%üI%;
130 NEXT I%
140 ; : ; : GOTO 70
150 END
```

Avsnitt 3.3 Digital signalering med I/O-kort 4085

Exempel 3: Avläsning av de fyra interruptingångarna

```
10  REM LÄSNING AV INTERRUPTSIGNALER
20  REM
30  Z%=INP(7%)
40  OUT 1%,6%
50  ; CHR$(12%)
60  ; : ; "DECIMALTALET AV INLÄSNINGEN" INP(1%)-240%
70  ; "BINÄRTALET: ";
80  FOR I%=3% TO 0% STEP -1%
90    ; (INP(1%) AND 2%üI%)/2%üI%
100 NEXT I%
110 ; : ; : ; : ; "TRYCK RETURN FÖR NÄSTA VÄRDE"; : GET G$
120 ; : ; : GOTO 60
```

Avsnitt 3.4 Digital signalering med reläkort 4103

Exempel 1: Rinnande ljus med reläkort 4103

```
10  REM RINNANDE LJUS MED HJÄLP AV 4103
20  REM
30  Z%=INP(7%)
40  OUT 1%,8%
50  REM
60  FOR I%=0% TO 7%
70    IF I%=0% THEN OUT 0%,2%ü0%+(2%ü7%)
80    IF I%>0% THEN OUT 0%,2%üI%+(2%üI%-1%)
90    FOR Z%=0% TO 200% : NEXT Z%
100 GOTO 80
110 END
```

********** Kapitel 4 - Analog signalering **********

Avsnitt 4.1 Analog signalering med DAC 4083 och 4084

Exempel 1: Strömstyrning med D/A-kortet 4083

```
10  REM STRÖMSTYRNING MED D/A KORTET 4083
20  REM
30  Z%=INP(7%)
40  OUT 1%,11%
50  ; CHR$(12%)
60  ; "VILKEN KANAL SKALL STÄLLAS"; : INPUT K%
70  IF K%<1% OR K%>2% GOTO 60
80  ; : ; "ANGE ÖNSKAD STÖMSTYRKA I mA (0-20)";
90  INPUT S%
100 IF S%<0% OR S%>20% GOTO 80
110 REM
120 U%=4095%/10%*S%/2%
130 REM
140 IF K%=1% OUT 0%,U%,2%,SWAP%(U%)
150 IF K%=2% OUT 3%,U%,4%,SWAP%(U%)
160 ; : ; : ; : GOTO 60
170 END
```

Avsnitt 4.1 Analog signalering med DAC 4083 och 4084

Exempel 2: Styrning av spänningsutgångarna på kort 4083

```
10  REM SPÄNNINGSSTYRNING MED D/A-KORTET 4083
20  REM
30  Z%=INP(7%)
40  OUT 1%,11%
50  ; CHR$(12%)
60  ; "VILKEN KANAL SKALL STÄLLAS"; : INPUT K%
70  IF K%<1% OR K%>2% GOTO 60
80  ; : ; "ANGE SPÄNNINGEN I V (0-10)"; : INPUT S
90  IF S<0 OR S>10 GOTO 80
100 REM
110 U%=4095%/10%*S
120 REM
130 IF K%=1% OUT 0%,U%,2%,SWAP%(U%)
140 IF K%=2% OUT 3%,U%,4%,SWAP%(U%)
150 ; : ; : ; : GOTO 60
160 END
```

Avsnitt 4.1 Analog signalering med DAC 4083 och 4084

Exempel 6: Generering av sinusvåg

```
10  REM GENERERING AV EN SINUSSIGNAL MED D/A-KORTET 4083
20  REM
30  S%=65408%
40  REM
50  REM **** ASSEMBLER RUTINEN 4.5 ****
60  REM
70  POKE 65408%,6%,50%,33%,255%,251%,35%,126%,211%,0%,35%
80  POKE 65418%,126%,211%,2%,16%,246%,24%,239%
90  REM
100 Z%=INP(7%)
110 OUT 1%,11%
120 REM
130 FOR I=0 TO 2*PI STEP 2*PI/50
140   REM
150   REM *** GENERERING AV SINUSSIGNALEN
160   REM
170   S%=(.5+SIN(I)/2)*4095%
180   REM
190   REM *** UPPLÄGGNING AV VÄRDENA
200   REM
210   POKE 64512+P%,S%,SWAP%(S%)
220   REM
230   P%=P%+2
240 NEXT I
250 Z%=CALL(S%)
260 END
```

Avsnitt 4.1 Analog signalering med DAC 4083 och 4084

Exempel 7: Styrning av strömutgångarna hos kort 4084

```
10  REM STRÖMSTYRNING MED D/A-KORTET 4084
20  REM
30  Z%=INP(7%)
40  OUT 1%,12%
50  ; CHR$(12%)
60  ; "VILKEN KANAL SKALL STÄLLAS"; : INPUT K%
70  IF K%<1% OR K%>4% GOTO 60
80  ; : ; "ANGE STRÖMMEN I mA (0-20)"; : INPUT S%
90  IF S%<0% OR S%>20% GOTO 80
100 REM
110 U%=255%/10%*S%/2%
120 REM
130 IF K%=1% OUT 0%,U%
140 IF K%=2% OUT 2%,U%
150 IF K%=3% OUT 3%,U%
160 IF K%=4% OUT 4%,U%
170 ; : ; : ; : GOTO 60
```

Avsnitt 4.1 Analog signalering med DAC 4083 och 4084

Exempel 8: Styrning av spänningsutgångarna hos kort 4084

```
10  REM SPÄNNINGSSTYRNING MED D/A-KORTET 4084
20  REM
30  Z%=INP(7%)
40  OUT 1%,12%
50  ; CHR$(12%)
60  ; "VILKEN KANAL SKALL STÄLLAS"; : INPUT K%
70  IF K%<1% OR K%>4% GOTO 60
80  ; : ; "ANGE SPÄNNINGEN I V (0-10)"; : INPUT S
90  IF S<0 OR S>20 GOTO 80
100 REM
110 U%=255%/10%*S
120 REM
130 IF K%=1% OUT 0%,U%
140 IF K%=2% OUT 2%,U%
150 IF K%=3% OUT 3%,U%
160 IF K%=4% OUT 4%,U%
170 ; : ; : ; : GOTO 60
```

Avsnitt 4.2 Analog signalering med ADC-kort 4115

Exempel 1: Läs ingång 1 och 3 från ADC-kort 4115

```
10  REM LÄSNING AV INGÅNG 1 OCH 3 PÅ 4115
20  REM
30  Z%=INP(7%)
40  OUT 1%,13%
50  REM
60  REM VÄLJER KANAL 1 MED SPÄNNINGSOMRÅDE 0 V TILL +10 V
    OCH 12 BITARS UPPLÖSNING
70  REM
80  OUT 2%,1%,3%,0%
90  IF INP(1%)>=128% GOTO 90
```

```
100 ; : ; : ; 'SPÄNNINGEN PÅ KANAL 1 ÄR '
    (SWAP%(INP(1%))+INP(0%))*10/4095 ' V'
110 REM
120 REM VÄLJER KANAL 3 MED SPÄNNINGSOMRÅDE -5 V TILL +5 V
    OCH 8 BITARS UPPLÖSNING
130 REM
140 OUT 2%,3%+32%,4%,0%
150 IF INP(1%)>=128% GOTO 150
160 ; : ; 'SPÄNNINGEN PÅ KANAL 3 ÄR ' (INP(0%)-255/2)*10/255 ' V'
170 GET Z$ : GOTO 80
180 END
```

Avsnitt 4.2 Analog signalering med ADC-kort 4115

Exempel 2: Avläsning av 16 enkla och 8 differentiella utgångar

```
10  REM LÄSNING AV 16 ENKLA OCH 8 DIFF. INGÅNGAR PÅ 4115
20  REM
30  Z%=INP(7%)
40  OUT 1%,13%
50  ; CHR$(12%)
60  ; 'ANGE TVÅ KANALER SOM SKALL LÄSAS';
70  INPUT K1%,K2%
80  IF K1%>=32% M1%=5% ELSE M1%=10%
90  IF K2%>=32% M2%=5% ELSE M2%=10%
100 REM
110 ; 'ANGE ÖNSKAD UPPLÖSNING (8/12 BITAR)'; : INPUT U%
120 IF U%=8% U%=4% ELSE U%=3%
130 REM
140 REM VÄLJER KANAL 1 OCH STARTAR OMVANDLINGEN
150 REM
160 OUT 2%,K1%,U%,0%
170 REM
180 GOSUB 310 : R1%=M1%*R%
190 REM
200 REM VÄLJER KANAL 2 OCH STARTAR OMVANDLINGEN
210 REM
220 OUT 2%,K2%,U%,0%
230 REM
240 GOSUB 310 : R2%=M2%*R%
250 REM
260 ; : ; : ; 'KANAL ' K1% ' = ' R1% ' V'
270 ; 'KANAL ' K2% ' = ' R2% ' V'
280 ; : ; : ; : GOTO 70
290 END
300 REM
310 REM UTRÄKNING AV RESULTATET BEROENDE PÅ UPPLÖSNINGEN
320 REM
330 IF U%=3% R%=(SWAP%(INP(1%))+INP(0%))/4095
340 IF U%=4% R%=INP(0%)/255
350 RETURN
```

Avsnitt 4.3 Analog signalering med kort 4021 och 4022

Exempel 1: 4021 med en PT100 givare

```
10  REM TEMPERATURMÄTNING MED 4021
20  REM
30  OUT 1%,9%
40  Z%=INP(7%)
50  GOSUB 180
60  REM
70  REM LÄSNING AV GIVAREN
80  REM
90  I%=(INP(1%) AND 15%)*256%+INP(0%)
100 REM
110 R=.534*I%+.533 : REM RÄKNAR UT RESISTANSEN
    PÅ PT-100 GIVAREN
120 REM
130 T=.997861*(-245.93+((6195.2*R)/(2619.1-R)))
140 REM
150 ; CUR(12%,43%) LEFT$(NUM$(T),4%)
160 GOTO 70
170 REM
180 REM UTSKRIFT AV TEMPERATUREN
190 REM
200 ; CHR$(12%)
210 ; CUR(12%,10%) "TEMPERATUREN ÄR FÖR
    NÄRVARANDE :        ' CELSIUS"
220 RETURN
```

Avsnitt 4.3 Analog signalering med kort 4021 och 4022

Exempel 2: 4021 och 4022 sammankopplade

```
10  REM MÄTNING AV TEMPERATUR MED 4021 OCH 4022
20  REM
30  Z%=INP(7%)
40  DIM R(14%),T(14%),I%(14%)
50  ; CHR$(12%)
60  ; CUR(3%,5%) "TEMPERATURMÄTNING MED 4021 & 4022"
70  REM
80  REM SÄTTER VÄRDENA PÅ KALIBRERINGS-RESISTORERNA
90  REM
100 R(12%)=148.93
110 T(12%)=98.78
120 REM
130 REM LÄSER AV KALIBRERINGS-RESISTORERNA FÖR BERÄKNING
    AV DEN RÄTA LINJEN
140 REM
150 FOR K%=12% TO 13%
160   OUT 1%,10% : REM ADRESSERAR MULTIPLEXERN
170   OUT 0%,0%
180   OUT 0%,16%+K% : REM LÄGGER UT EN KANALS VÄRDE TILL 4021
190   REM
200   OUT 1%,9% : REM ADRESSERAR KONVERTERAREN
210   FOR V=0 TO 2000 : NEXT V
220   I%(K%)=SWAP%(INP(1%) AND 15%)+INP(0%)
230 NEXT K%
240 REM
250 REM RÄKNAR UT EKVATIONEN FÖR DEN RÄTA LINJEN
260 REM
```

```
270 K=(R(12%)-R(13%))/(I%(12%)-I%(13%))
280 L=R(13%)*I%(12%)-R(12%)*I%(13%)
290 L=L/(I%(12%)-I%(13%))
300 REM
310 REM LÄSER ALLA KANALER
320 REM
330 FOR L%=0% TO 11%
340   ; CUR(L%+5%,8%) "=>"
350   OUT 1%,10% : REM ADRESSERING AV MULTIPLEXERKORTET
360   OUT 0%,0%
370   OUT 0%,16%+L% : REM LÄGGER UT EN KANAL TILL
      KONVERTERINGSKORTET
380   REM
390   OUT 1%,9% : REM VAL AV KONVERTERINGS KORTET
400   OUT 0%,I%(L%),2%,SWAP%(I%(L%)) : REM TIDIGARE VÄRDE
      TILL RÄKNAREN
410   FOR V=0 TO 2000 : NEXT V
420   I%(L%)=SWAP%(INP(1%) AND 15%)+INP(0%)
430   REM
440   R(L%)=K*I%(L%)+L
450   T(L%)=.997861*(-245.93+(6195.2*R(L%)/(2619.1-R(L%))))
460   ; CUR(5%+L%,8%) "   KANAL " L%,T(L%) '" CELSIUS"
470 NEXT L%
480 GOTO 310 : REM LÄSER ALLA KANALER EN GÅNG TILL
490 END
```


Exempel 1: Användning av Philips multimeter PM 2528

```
10  REM MÄTNING MED EN AUTOMATISK MULTIMETER
20  REM
40  OPEN 'IEC:' ASFILE 49%
50  ; CHR$(12%)
60  ; ' MÄTNING MED PHILLIPS PM 2528 '
70  ; CUR(5%,10%)' 1. MÄTNING AV LIKSPÄNNING'
80  ; CUR(6%,10%)' 2. MÄTNING AV VÄXELSPÄNNING'
90  ; CUR(7%,10%)' 3. MÄTNING AV VÄXELSPÄNNING'
100 ; CUR(8%,10%)" 4. MÄTNING AV MOTSTÅND (2W)"
110 ; CUR(9%,10%)" 5. MÄTNING AV MOTSTÅND (4W)"
120 ; CUR(10%,10%)' 6. MÄTNING AV LIKSTRÖM'
130 ; CUR(11%,10%)' 7. MÄTNING AV VÄXELSTRÖM'
140 ; CUR(12%,10%)' 8. MÄTNING AV TEMPERATUR'
150 ; CUR(13%,10%)' 9. MÄTNING AV HÖGFREKVESSPÄNNING'
160 ; CUR(14%,10%)'10. MÄTNING AV SPÄNNINGSTOPPAR'
170 ; CUR(15%,10%)'11. MÄTNING AV SPÄNNINGSBOTTNAR'
180 ; CUR(16%,10%)'12. MÄTNING AV TOPP-TOPP SPÄNNING'
190 REM
200 ; CUR(18%,5%)"VILKET ALTERNATIV ÖNSKAS"; : INPUT A%
210 REM
220 REM INLÄSNING AV KODEN FÖR OMRÅDET
230 REM
240 RESTORE 280
250 FOR L%=1% TO A%
260 READ K$,E$
270 NEXT L%
280 DATA F00,VOLT,F01,VOLT,F02,VOLT,F03,OHM,F04,OHM,F05,AMPERE
290 DATA F06,AMPERE,F07,GRADER C,F08,VOLT,F09,VOLT,F10,VOLT,F11,VOLT
300 REM
310 REM MÄTNING AV DET VALDA OMRÅDET
```

```
320 REM
330 REM STÄLLER MULTIMETERN I VALT LÄGE + AUTOMATISK
    SKALNING, EXTERN START
340 REM INGEN SRQ,SNABB MÄTNING OCH HÖG UPPLÖSNING PÅ MÄTVÄRDET
350 REM
360 CMD "?U6",Kod$+"R0T1D0S1H1"
370 REM
380 CMD "?U6"+CHR$(8) ! STARTAR EN MÄTNING
390 REM
400 CMD "?V5" ! DEFINIERAR MULTIMETER SOM TALKER OCH DATORN
    SOM LISTENER
410 REM
420 Svar$=IEC$(12) ! LÄSER VÄRDET FRÅN BUSSEN
430 REM
440 REM UTSKRIFT AV SVARET
450 REM
460 ; CUR(20%,5%)'SVARET AV MÄTNINGEN BLEV 'S$' 'E$
470 ; CUR(22%,5%)"NY MÄTNING ELLER ÅTERGÅNG (Å/M)<M>"; : INPUT F$
480 REM
490 REM MASKAR SMÅ BOKSTÄVER TILL STORA
500 REM
510 IF CHR$(ASC(F$) AND 95%)='Å' GOTO 10
520 GOTO 380
530 END
```

Exempel 2: Användning av Philips räknare PM 6671

```
10  REM MÄTNING MED PHILIPS PM 6671 RÄKNARE
20  REM
30  OPEN 'IEC:' ASFILE 49%
40  ; CHR$(12%)
50  ; ' MÄTNING MED PHILLIPS PM 6671 '
60  ; CUR(5%,10%)' 1. COUNT A (MANUEL)'
70  ; CUR(6%,10%)' 2. COUNT A START/STOP BY B'
80  ; CUR(7%,10%)' 3. COUNT A GATED BY B'
90  ; CUR(8%,10%)" 4. FREQUENCY A"
100 ; CUR(9%,10%)" 5. PERIOD A"
110 ; CUR(10%,10%)' 6. TIME INTERVAL A-B SINGLE'
120 ; CUR(11%,10%)' 7. TIME INTERVAL A-B AVERAGE'
130 ; CUR(12%,10%)' 8. PULSE DURATION A'
140 ; CUR(13%,10%)' 9. RATIO A/B'
150 ; CUR(14%,10%)'10. PHASE A-B'
160 ; CUR(15%,10%)'11. RPM'
170 ; CUR(16%,10%)'12. ÄNDRING AV MÄTTID. NU : 'T$' Sek'
180 REM
190 ; CUR(18%,5%)'VILKET ALTERNATIV ÖNSKAS'; : INPUT A%
200 IF A%=12% ; CUR(20%,10%)'VILKEN MÄTTID ÖNSKAS (0.01 - 99 S.)';
    : INPUT T$ : GOTO 40
210 REM
220 REM INLÄSNING AV KODEN FÖR OMRÅDET
230 REM
240 RESTORE
250 FOR L%=1% TO A%
260 READ K$,E$
270 NEXT L%
280 DATA F9, ,F4, ,F2, ,F15,Hz,F13,SEK,F6,SEK
290 DATA F11,SEK,F12,SEK,F7,Hz,F3,GRADER,F14,Hz
300 REM
310 REM MÄTNING AV DET VALDA OMRÅDET
320 REM
```

```
330 CMD '?U*',Kod$+'G1AT0M'+Tid$+'S0'
340 REM
350 CMD "?J5" ! DEFINIERAR RÄKNAREN SOM TALKER OCH DATORN
    SOM LISTENER
360 REM
370 Svar$=IEC$(18) ! LÄSER VÄRDET FRÅN BUSSEN
380 GOSUB 500 : S$=R$ : REM BEHANDLAR SVARET
390 REM
400 REM UTSKRIFT AV SVARET
410 REM
420 ; CUR(20%,5%)'SVARET AV MÄTNINGEN BLEV 'S$' 'E$
430 ; CUR(22%,5%)'NY MÄTNING ELLER ÅTERGÅNG (M/Å)<M>';
    : INPUT F$
440 REM
450 REM MASKAR SMÅ BOKSTÄVER TILL STORA
460 REM
470 IF CHR$(ASC(F$) AND 95%)='Å' GOTO 10
480 GOTO 330
490 REM
500 REM TAR BORT ALLA OÖNSKADE TECKEN FRÅN SVARET
510 REM
520 T1%=INSTR(1%,S$,' ')
530 IF T1%=0% ; CUR(23%,10%)'      FÖR STORT VÄRDE      ' : GOTO 50
540 R$=MID$(S$,T1%+1%,LEN(S$)-5%)
550 RETURN
560 END
```

Exempel 3: Användning av Philips PM 5190 Synthesizer

```
10  REM MUSIK FÖR PHILIPS PM 5190
20  REM
30  REM FJÄRILNS VINGAR SYNS...
40  REM
60  ; CHR$(12%,26%)
70  REM
80  DIM F(12%,7%)
90  G=55% : A$="?U1"
100 REM
110 OPEN "IEC:" ASFILE 49%
120 CMD Adress$,"A.001D00"+CHR$(3)
130 REM
140 REM LOOP FÖR ATT LÄGGA UPP TONERNA
150 REM
160 FOR Q%=1% TO 7%
170 FOR T%=1% TO 12%
180 C%=(12%*(Q%-1%)+T%)-1%
190 F(T%,Q%)=(INT(G*(2ü(C%/12%))*100%+.5))/1E+5
200 NEXT T%
210 NEXT Q%
220 REM
230 FOR V=1 TO 3 : REM 1=SINUSVÅG :  2=FYRKANTSVÅG
    :  3=TREKANTSVÅG
240 REM
250 CMD Adress$,"W"+NUM$(Vågform)+"A.001"+CHR$(3)
260 READ T%,Q%,L%,T1%,A$
270 IF Q%=0% THEN 340
280 CMD Adress$,"F"+NUM$(Frekvens(Ton,Oktav+2))+"A"+A$+"D00"+CHR$(3)
290 T2%=(15%/L%+.5) : GOSUB 380
300 IF T1%=0 THEN 260
310 CMD Adress$,"A.001D00"+CHR$(3)
```

```
320 T2%=(15%/T1%+.5) : GOSUB 380
330 GOTO 260
340 CMD Adress$,"A.001D00"+CHR$(3)
350 FOR V1=1 TO 3000 : NEXT V1
360 RESTORE : NEXT V
370 GOTO 230
380 FOR V1=1 TO T1%*75 : NEXT V1 : RETURN
390 REM
400 REM DATA FÖR MELODIN
410 REM
420 DATA 8,2,8,100,"19.9",5,2,8,100,"19.9"
430 DATA 3,2,4,100,"19.9",3,2,4,100,"19.9",3,2,4,100,
        "19.9",3,2,4,100,"19.9"
440 DATA 8,2,4,100,"19.9",8,2,4,4,"19.9",12,2,8,100,
        "19.9",3,3,8,100,"19.9"
450 DATA 1,3,4,100,"19.1",1,3,4,100,"19.9",1,3,4,100,
        "19.9",3,3,8,100,"19.9",1,3,8,100,"19.9"
460 DATA 12,2,2,4,"19.9",8,2,8,100,"19.9",5,2,8,100,"19.9"
470 DATA 3,2,4,100,"19.9",3,2,4,100,"19.9",3,2,4,100,
        "19.9",3,2,4,100,"19.9"
480 DATA 8,2,4,100,"19.9",8,2,4,4,"19.9",3,3,8,100,
        "19.9",12,2,8,100,"19.9"
490 DATA 10,2,4,100,"19.9",10,2,4,100,"19.9",10,2,8,100,"19.9",
        8,2,8,100,"19.9",10,2,8,100,"19.9",12,2,8,100,"19.9"
500 DATA 8,2,2,4,"19.9",3,3,8,100,"19.9",1,3,8,100,"19.9"
510 DATA 12,2,4,100,"19.9",12,2,4,100,"19.9",12,2,4,100,
        "19.9",5,3,4,100,"19.9"
520 DATA 3,3,4,100,"19.9",1,3,4,4,"19.9",1,3,8,100,"19.9",
        12,2,8,100,"19.9"
530 DATA 10,2,4,100,"19.9",10,2,4,100,"19.9",10,2,4,100,
        "19.9",3,3,4,100,"19.9"
540 DATA 1,3,4,100,"19.9",12,2,4,4,"19.9",8,2,8,100,
        "19.9",5,2,8,100,"19.9"
550 DATA 3,2,4,100,"19.9",3,2,4,100,"19.9",3,2,4,100,
        "19.9",3,2,4,100,"19.9"
560 DATA 8,2,4,100,"19.9",8,2,4,4,"19.9",3,3,8,100,
        "19.9",12,2,8,100,"19.9"
570 DATA 10,2,4,100,"19.9",10,2,4,100,"19.9",10,2,8,100,
        "19.9",8,2,8,100,"19.9",10,2,8,100,"19.9",12,2,8,100,"19.9"
580 DATA 8,2,2,4,"19.9"
590 DATA 0,0,0,0,"0.00"
600 DATA 0,0,0,0,"0.00"
```

Exempel 5: Avläsning av Philips PM 3310 Oscilloskop

```
10  REM PROGRAMMMERING OCH LÄSNING AV PHILIPS PM 3310
20  REM
30  ; CHR$(12)
40  REM
50  DIM D$=600,I$=900,Ö$=100,M(24),D(256),L$(13)=6
60  F$="MTCCBBD03798100C381E003310"
70  R$="RK00207B02100F1250000120"
80  REM
90  REM ---------- LÄS KOMMANDO
100 REM
110 ; CUR(0,8)"PROGRAMMERING AV PHILIPS PM 3310"
120 GOSUB 380
130 ; CUR(21,15)"KOMMANDO"; : INPUT K$ : IF LEN(K$)<2 GOTO 130
140 ; CUR(21,0)SPACE$(80)
```

```
150 K=(INSTR(1,"LF,DF,ÄF,SF,LR,SR,DR,LD,SD,OP,DD,TD,
    BD,CL",LEFT$(K$,2))+5)/3
160 ON K GOSUB 190,560,690,880,1050,1150,2090,1440,2380,
    2630,250,2270,2730,3410,300
170 GOTO 90
180 REM
190 ; CUR(21,15)SPACE$(65)
200 ; CUR(21,30)"?????"
210 RETURN
220 REM
230 REM --------- ÖPPNAR IEC-BUSSEN
240 REM
250 OPEN "IEC:" ASFILE 49
260 CMD "?U2",CHR$(27)+"0DD0"+CHR$(3) :  E4=1 :  RETURN
270 REM
280 REM --------- STÄNGER IEC-BUSSEN
290 REM
300 CLOSE 49 : E4=0 : RETURN
310 REM
320 REM --------- IEC-BUSSEN HAR EJ ÖPPNATS
330 REM
340 ; CUR(21,15)SPACE$(65)
350 ; CUR(21,30)"*** IEC:  EJ ÖPPNAT!!REM! ***"CHR$(7);
360 RETURN
370 REM
380 REM ---------- UTSKRIFT AV MENY'N
390 REM
400 ; CUR(2,10)"LF = LÄS      FRONT"
410 ; CUR(3,10)"DF = DISPLAY FRONT"
420 ; CUR(4,10)"ÄF = ÄNDRA   FRONT"
430 ; CUR(5,10)"SF = SKRIV   FRONT"
440 ; CUR(7,10)'LR = LÄS   REGISTER FRÅN PM 3310'
450 ; CUR(8,10)'SR = SKRIV   -"-    TILL  -"-'
460 ; CUR(9,10)'DR = DISPLAY  -"-  (INST&DATA)'
470 ; CUR(11,10)'LD = LÄS   DATA FRÅN DISK'
480 ; CUR(12,10)'SD = SKRIV -"-  TILL -"-'
490 ; CUR(14,10)'DD = DISPLAY DATA PÅ SKÄRM'
500 ; CUR(15,10)'TD = TILLVERKA DATA'
510 ; CUR(16,10)'BD = BEARBETA   -"-'
520 ; CUR(18,10)'OP = OPEN  IEC:'
530 ; CUR(19,10)'CL = CLOSE -"-'
540 RETURN
550 REM
560 REM ---------- LÄS FRONT
570 REM
580 IF E4=0 GOTO 340
590 ; CUR(21,15)SPACE$(65)
600 CMD "?U(",CHR$(27)+"0MT00"+CHR$(3)
610 CMD "?H5" :  F$=""
620 FOR I=1 TO 13
630 I1$=LEFT$(IEC$(3),2)
640 F$=F$+I1$
650 NEXT I
660 ; CUR(21,30)F$
670 RETURN
680 REM
690 REM ------- DISPLAY FRONT
700 REM
710 ; CHR$(12)
720 RESTORE 3920
730 FOR I=1 TO 13
```

```
740 READ L$(I)
750 NEXT I
760 RESTORE 3940
770 FOR I=1 TO LEN(F$) STEP 2
780 READ L$
790 ; L$(I/2+1)TAB(8)L$TAB(15)MID$(F$,I,2)"     "
800 NEXT I
810 IF K<>3 RETURN
820 ; CUR(22,10)'TRYCK PÅ EN TANGENT FÖR RETURN
    TILL MENYN'; : GET G$
830 ; CHR$(12)
840 RETURN
850 REM
860 REM PROGRAMMERING AV FRONTPANELEN
870 REM
880 GOSUB 690
890 ; CUR(1,20);
900 C=0
910 FOR I=1 TO LEN(F$)
920 GET Ö$
930 IF Ö$="<" GOTO 1010
940 IF Ö$=CHR$(13) AND (I AND 1)=1 ; CUR(I/2+1,20)MID$(F$,I,2)
    : Q9=1
950 IF Q9=1 Q9=0 : I=I+1 : GOTO 990
960 IF Ö$=CHR$(13) AND (I AND 1)=0 GOTO 920
970 F$=LEFT$(F$,I-1)+Ö$+RIGHT$(F$,I+1) : C=1
980 ; Ö$
990 IF (I AND 1)=0 ; CUR(I/2+1,20);
1000 NEXT I
1010 ; : IF C GOTO 880
1020 ; CHR$(12)
1030 RETURN
1040 REM
1050 REM ---------- SET FRONT
1060 REM
1070 IF E4=0 GOTO 340
1080 CMD "?U(",CHR$(27)+"0"+F$
1090 ; CUR(21,15)SPACE$(65)
1100 ; CUR(21,30)"ÖVERFÖRT "
1110 RETURN
1120 REM
1130 REM ---------- LÄS DATA FRÅN PM3310  REGISTER Ö
1140 REM
1150 GOSUB 1340
1160 IF E4=0 GOTO 340
1170 CMD "?U(",CHR$(27)+"0R"+CHR$(75+Ö)+"00"
1180 CMD "?H5"
1190 ; CHR$(12)
1200 R$=""
1210 FOR I=1 TO 12
1220 I$=IEC$(3)
1230 R$=R$+LEFT$(I$,2)
1240 NEXT I
1250 D$=""
1260 FOR I=1 TO 256
1270 I$=IEC$(3)
1280 D$=D$+LEFT$(I$,2)
1290 NEXT I
1300 ; CUR(21,15)SPACE$(65)
1310 ; CUR(21,30)"LÄST REGISTER "ASC(MID$(R$,2,1))-75
1320 RETURN
```

```
1330 REM
1340 REM --------- VAL AV REGISTER
1350 REM
1360 ONERRORGOTO 1360
1370 ; CUR(21,15)SPACE$(65)
1380 ; CUR(21,30)"VILKET REGISTER SKALL BEHANDLAS (0-3)";
1390 INPUT Ö$
1400 Ö=VAL(Ö$)
1410 IF Ö<0 OR Ö>3 GOTO 1360
1420 RETURN
1430 REM
1440 REM ------------ DISPLAY REGISTER DATA
1450 REM
1460 I1=0
1470 FOR I=3 TO 15 STEP 6
1480 GOSUB 2000
1490 I1=I1+1
1500 G(I1)=G
1510 NEXT I
1520 G(4)=FNH1(MID$(R$,21,1))*1000+FNH1(MID$(R$,22,1))*100
1530 G(4)=G(4)+FNH1(MID$(R$,23,1))*10+FNH1(MID$(R$,24,1))
1540 IF MID$(R$,19,1)="F" G(4)=-G(4)
1550 FOR I=7 TO 20
1560 M(I)=FNH1(MID$(R$,I,1))
1570 NEXT I
1580 ; CHR$(12)
1590 GOSUB 1850
1600 ; "****** REGISTER PARAMETERS *******"
1610 ; : ; "KANAL ";
1620 IF M(7) AND 2 ; "A ELLER B  (1 SVEP)"; ELSE
     ; "A OCH B  (2 SVEP)";
1630 ; "       "; : IF M(7) AND 1 ; "JÄMN" ELSE ; "UDDA"
1640 IF (M(8) AND 8)=0 ; "ADD   ";
1650 IF (M(14) AND 8)=0 ; "    B INVERTERAD";
1660 ;
1670 IF (M(8) AND 4)=0 ; "----- SETTING KANAL A -----"
1680 I1=7
1690 GOSUB 1920
1700 IF (M(14) AND 4)=0 ; "----- SETTING KANAL B -----"
1710 I1=13
1720 GOSUB 1920
1730 ; "=========== TIDBAS ========"
1740 ; "TIDBAS: "G(3)"   SEK/RUTA"
1750 ; "MODE:   "; : IF M(13) AND 8 ; "ROLL";
1760 IF M(13) AND 4 ; "    SINGLE";
1770 IF M(13) AND 2 ; "    MULTIPLE;"
1780 ;
1790 ; "DELAY  "G(4)
1800 ; CUR(22,10)'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
1810 GET G$
1820 ; CHR$(12)
1830 RETURN
1840 REM
1850 REM -------- DISP.PARAM.BYTE
1860 REM
1870 FOR I=1 TO LEN(R$) STEP 2
1880 ; MID$(R$,I,2)" ";
1890 NEXT I
1900 ; : RETURN
1910 REM
1920 REM ----- DISP.KANAL SETTING
```

```
1930 REM
1940 ; "FÖRSTÄRKNING  "G(I1/6)"  VOLT / RUTA"
1950 IF M(I1) AND 1 ; "EJ KALIBRERAD" ELSE ; "KALIBRERAD"
1960 IF M(I1+1) AND 1 ; "DC KOPPLAD" ELSE ; "AC KOPPLAD"
1970 IF M(I1+1) AND 2 ; "ZERO"
1980 RETURN
1990 REM
2000 REM ------- OMVANDLING AV 4 BYTE ASC TILL FLOAT G(3)
2010 REM
2020 G=VAL(MID$(R$,I+1,1))
2030 IF MID$(R$,I,1)="1" G=-G
2040 G=VAL(MID$(R$,I+2,2))*.01*10üG
2050 RETURN
2060 REM
2070 REM --------- SKRIV DATA TILL PM 3310
2080 REM
2090 GOSUB 1340
2100 IF E4=0 GOTO 340
2110 CMD "?U("
2120 R$="R"+CHR$(75+Ö)+RIGHT$(R$,3)
2130 FOR I=1 TO 24 STEP 2
2140 CMD ,MID$(R$,I,2)+CHR$(3)
2150 NEXT I
2160 FOR I=1 TO 512 STEP 2
2170 CMD ,MID$(D$,I,2)+CHR$(3)
2180 NEXT I
2190 ; CHR$(12)
2200 ; R$ : ; D$
2210 ; "SKRIVIT TILL REGISTER "ASC(MID$(R$,2,1))-75
2220 ; : ; 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
2230 GET G$
2240 ; CHR$(12)
2250 RETURN
2260 REM
2270 REM ---------- DISPLAY DATA
2280 REM
2290 ; CHR$(26,12)
2300 FOR I=1 TO 512 STEP 2
2310 ; CUR(10-(FNH(MID$(D$,I,2))+5)/10,3+I/16)"*";
2320 NEXT I
2330 ; CUR(22,0)'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
2340 GET G$
2350 ; CHR$(12)
2360 RETURN
2370 REM
2380 REM --------- LÄS FRÅN DISK
2390 REM
2400 GOSUB 2500 : IF K=0 RETURN
2410 INPUT §1,R$
2420 D$=""
2430 FOR I=0 TO 5
2440 INPUT §1,Ö$
2450 D$=D$+Ö$
2460 NEXT I
2470 CLOSE 1
2480 RETURN
2490 REM
2500 REM --------- VAL AV FIL
2510 REM
2520 ; CUR(21,15)SPACE$(65)
2530 ; CUR(21,30)"FILNAMN";
```

```
2540 INPUT S$
2550 IF LEN(S$)>8 S$=LEFT$(S$,8)
2560 S$=S$+".DSP"
2570 ONERRORGOTO 2600
2580 OPEN S$ ASFILE 1
2590 RETURN
2600 IF K=9 K=0 : ; CUR(21,30)
     "FILEN  "S$"    FINNS EJ !!       "ERRCODE : RETURN
2610 PREPARE S$ ASFILE 1 : RETURN
2620 REM
2630 REM --------- SKRIV TILL DISK
2640 REM
2650 GOSUB 2500
2660 ; §1,R$
2670 FOR I=0 TO 4
2680 ; §1,MID$(D$,1+I*100,100)
2690 NEXT I
2700 ; §1RIGHT$(D$,501)
2710 CLOSE 1 : RETURN
2720 REM
2730 REM -------- TILLVERKA DATA
2740 REM
2750 ; CUR(21,15)SPACE$(65)
2760 ; CUR(21,30)"S=SINUS F=FYRKANT T=TRIANGEL E=E-FUNKT.";
2770 INPUT Ö$
2780 ON INSTR(1,"SFTE",Ö$)+1 GOTO 2770,2800,2930,3080,3260
2790 REM
2800 REM -------- SINUS
2810 REM
2820 U=100
2830 P=2
2840 ; CUR(21,15)SPACE$(65)
2850 ; CUR(21,30)"SINUS     U-TOPP="U"    PERIODER="P
2860 K=P*2*PI/256
2870 D$=""
2880 FOR T=0 TO 255
2890 Ö=(U*(SIN(T*K))+.5) : GOSUB 3830 : D$=D$+Ö9$
2900 NEXT T
2910 RETURN
2920 REM
2930 REM -------- FYRKANT
2940 REM
2950 U=100
2960 P=3
2970 ; CUR(21,15)SPACE$(65)
2980 ; CUR(21,30)"FYRKANT     U-TOPP="U"    PERIODER="P
2990 P0=256/(P*2)
3000 D$=""
3010 FOR T=0 TO 255
3020 T1=T1+1
3030 IF T1>P0 T1=0 : U=U*(-1)
3040 Ö=U : GOSUB 3830 : D$=D$+Ö$
3050 NEXT T
3060 RETURN
3070 REM
3080 REM --------- TRIANGEL
3090 REM
3100 U=100
3110 P=3
3120 ; CUR(21,15)SPACE$(65)
3130 ; CUR(21,30)"TRIANGEL     U-TOPP="U"    PERIODER="P
```

```
3140 P0=256/(P*2)
3150 U1=(2*U)/P0
3160 V=-U-U1
3170 D$=""
3180 FOR T=0 TO 255
3190 T1=T1+1
3200 IF T1>P0 T1=0 : U1=U1*(-1)
3210 V=V+U1
3220 Ö=V : GOSUB 3830 : D$=D$+Ö9$
3230 NEXT T
3240 RETURN
3250 REM
3260 REM --------- E-FUNKTION
3270 REM
3280 U=100
3290 C=.5
3300 T1=.2
3310 ; CUR(21,15)SPACE$(65)
3320 ; CUR(21,30)"E-FUNKTION     U-TOPP="U"    TIDSKONSTANT="C
3330 C1=-T1*10/(256*C)
3340 D$="" : E=2.71828
3350 FOR T=0 TO 255
3360 V=U*Eü(T*C1)
3370 Ö=V : GOSUB 3830 : D$=D$+Ö9$
3380 NEXT T
3390 RETURN
3400 REM
3410 REM -------- BEARBETA DATA BERÄKNA MAX O MIN
3420 REM
3430 FOR I=1 TO 256
3440 D(I)=FNH(MID$(D$,I*2-1,2))
3450 NEXT I
3460 M0=500 : M2=-500
3470 FOR I=1 TO 256
3480 IF D(I)<M0 M0=D(I) : M1=I
3490 IF D(I)>M2 M2=D(I) : M3=I
3500 NEXT I
3510 ; CHR$(12)
3520 ; : ; "BERÄKNADE VÄRDEN:"
3530 ; "MAXVÄRDE ="M2" ENHETER  "M2*G(1)/127" VOLT     EFTER ";
3540 ; M3" ENHETER  "M3*G(3)/25.6" SEK."
3550 ; "MINVÄRDE ="M0" ENHETER  "M0*G(1)/127" VOLT     EFTER ";
3560 ; M1" ENHETER  "M1*G(3)/25.6" SEK."
3570 ; : ; 'TRYCK PÅ EN TANGENT FÖR RETURN TILL MENYN';
3580 GET G$
3590 ; CHR$(12)
3600 RETURN
3610 REM
3620 REM ========== OMVANDL AV HEX I ASC TILL INTEGER
3630 REM
3640 REM --------- ETT TECKEN
3650 REM
3660 DEFFNH1(Ö$)=INSTR(1,"0123456789ABCDEF",Ö$)-1
3670 REM
3680 REM --------- TVÅ TECKEN
3690 REM
3700 DEFFNH2(Ö$)=FNH1(LEFT$(Ö$,1))*16+FNH1(RIGHT$(Ö$,2))
3710 REM
3720 REM --------- TVÅ TECKEN TILL HELTAL
3730 REM
3740 DEFFNH(Ö$)=SWAP%(FNH2(Ö$))/256
```

```
3750 REM
3760 REM ----- 4 BIT TILL ASC
3770 REM
3790 Ö$=CHR$((Ö AND 15)+48)
3800 IF Ö$>"9" Ö8$=CHR$(ASC(Ö$)+7) ELSE Ö8$=Ö$
3810 RETURN
3820 REM
3830 REM ----- EN BYTE TILL TVÅ ASC
3840 REM
3850 GOSUB 3780
3860 Ö1$=Ö8$((Ö AND 255)/16)
3870 Ö9$=Ö1$+Ö8$
3880 RETURN
3890 REM
3900 REM ---------- DATA FÖR FRONTPANELEN
3910 REM
3920 DATA BYTE,  3&4,   5&6,   7&8,   9&10,  11&12, 13&14
3930 DATA 15&16, 17&18, 19&20, 21&22, 23&24, 25&26
3940 DATA COD,   AT.A,  MO.A,  AT.B,  MO.B,  BA.T
3950 DATA MO.T,  RE ,   MO.D,  SE.R,  TR+,   DEL,   DEL,
```

APPENDIX D

Referenser

ABC om mätdatorsystem	Eriksson, Magnusson, Rasmusson Metric/Liber
BASIC II boken	Lundgren, Thornell Liber
ABC om programmering och dokumentation	Lundgren, Thornell EMM-Data/Liber
Börja med BASIC	Björk, Nilsson Liber
Styr och mät med ABC-80	Westh Studentlitteratur
Avancerad programmering på ABC-80	Isaksson, Kärrsgård Metric/Studentlitteratur
DataBoard Systemmanualer	Sattco AB
ABC-Lab	Berggren Liber

ISBN 91-40-20486-3