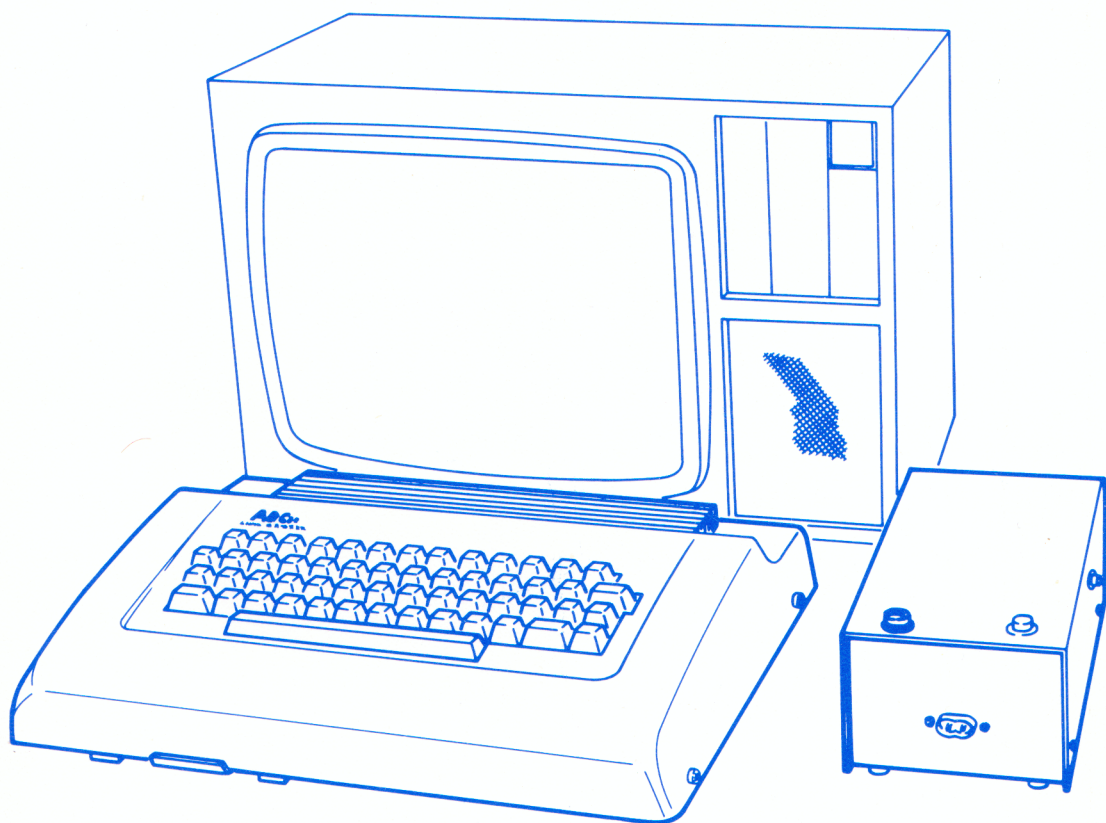


BRUKSANVISNING



ABC80

Förord

Denna bruksanvisning till den svenska datorn ABC 80 försöker att ge Dig som användare en kort presentation och beskrivning av ABC 80.

Utöver denna beskrivning innehåller denna bruksanvisning en uppställning av olika kommandon, funktioner, operatorer, typ av variabler etc som man kan använda i ABC 80. Den som vill lära sig programmering rekommenderas boken "ABC OM BASIC" som behandlar ABC 80 BASIC från grunden.

Den som vill lära sig hur ett mikrodatorsystem fungerar bör läsa "Mikrodatorns ABC", som är skriven med utgångspunkt från ABC 80-systemet.

Böckerna kan läsas med god behållning även av den som aldrig arbetat med en dator förr. Du som redan kan BASIC kan komma tillrätta med bilagan "Kommandon, satser, funktioner och operatorer" då Du börjar att programmera på ABC 80.

ABC 80 är en kraftfull och snabb universaldator som lätt kan byggas ut och anpassas för de mest skiftande applikationer.

Separata bruksanvisningar och läroböcker finns för flexskivesystem, skrivare (printer), m.m.

Produktblad finns med beskrivning på tillbehör och kablar.

I övrigt se Litteraturförteckningen.

Copyright Luxor-Scandia Metric

Innehållsförteckning

1. Det här är ABC-80

1.1	Grundutrustning	4
1.2	Vad Du kan göra med ABC-80	4
1.3	Köpta program och egna program	5
1.4	Kringutrustning	6
1.5	Dialogen mellan Dig och ABC-80	7
1.6	Lagringsmöjligheter	9
1.7	Ytterligare specialiteter	10
1.8	Start av ABC-80	11
1.9	Felorsaker, åtgärder	12
1.10	Ett programexempel	13
1.11	Programlagring	15
1.12	Visning av program på bildskärmen	15

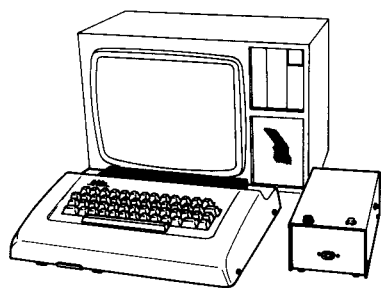
2. Introduktion till BASIC

2.1	Allmänt	17
2.2	Program	18
2.3	Kommandon	19
2.4	Utrymme	19
2.5	Aritmetiken i ABC-80	19
2.6	Enkla variabler	20
2.7	Indexerade variabler	20
2.8	Strängar och fält	21
2.9	Satser och uttryck	21
2.10	Jämförelseuttryck och jämförelseoperatorer	22
2.11	Stränguttryck	23
2.12	Läsning av strängar	24
2.13	Variabelbeteckningar	24
2.14	Filnamn	24
2.15	ASCII-aritmetiken	25
2.16	Användning av ABC-80 som kalkylator	25
2.17	Programmeringstips	26
2.18	Rättelser av fel i programmet	26
2.19	Praktiska råd	27
2.20	Grafisk mod	29
2.21	ABC-80 bussen	33
2.22	Intern parallell I/O-krets (PIO)	34
2.23	Ljudgeneratören	36

3. Användningsexempel

3.1	Två programexempel	38
3.2	CHAIN – för mycket stora program	39
3.3	Inläggning av maskinspråksprogram	41
3.4	ABC-80:s realtidsklocka	42
3.5	Återanvändning av data	43

4.	Kommandon, satser, funktioner, operatorer	
4.1	Kommandon	45
4.2	Instruktioner (satser)	47
4.3	Strängfunktioner	50
4.4	Matematiska funktioner	51
4.5	Specialfunktioner	52
4.6	Minnesåtkomst och IN/UT-portar	52
4.7	Specialtecken	53
4.8	Uttryck	53
4.9	Relationsoperatorer	54
4.10	Logiska operatorer	54
4.11	Kontrolltecken	54
5.	Referenskort	55
6.	Fel-meddelanden	56
7.	Övrigt	59
	Videografik	31
	ASCII-kod	32
	Minneskarta	59
	Errator	60
	Koder på tangentbord	61
	Appendix	62
8.	Litteraturförteckning	64
9.	Sakregister	66



1. Det här är ABC 80

1.1 Grundutrustningen

ABC 80 består av en i tangentbordet inbyggd dator, en HF/Videoenhet och en HF-omkopplarbox. Datorn är uppbyggd kring mikroprocessorn Z80A. Systemprogramvaran för programspråket BASIC, lagras i 16kByte ROM/PROM.

Förutom den utrustning som levereras med datorn behöver Du en bildenhet, en vanlig TV eller en monitor t ex 230 7280-11 från Luxor.

Tangentbordet är utfört enligt svensk skrivmaskinsstandard. Ett antal specialtangenten för BASIC har tillkommit. Tangentbordet använder Du för att:

- a) ge indata till ett program, dvs mata in siffror eller text, som ska bearbetas av ABC 80.
- b) ge fasta kommandon, s k systemkommandon, till ABC 80 – t ex ladda in ett program från ett kassetminne, starta ett program eller begära utskrift av ett program.
- c) producera program.

HF/Videoenheten innehåller HF- och videodel, kraftenhet för datorn samt ljuddel med en högtalare för återgivning av ljudsignaler.

HF-omkopplarboxen används vid anslutning av HF/Videodelen till antenningången på en vanlig TV-mottagare. Du kan med denna välja om du vill använda TV:n som bildenhet eller som TV.

Som bildenhet används en vanlig TV med HF- eller videoingång alternativt en monitor. För text som ABC 80 visar finns plats för 24 rader å 40 tecken på bildskärmen. Här sker kommunikationen mellan dig och datorn: Du matar in värden, datorn svarar, datorn frågar Dig, Du svarar. Bildskärmen visar text, siffror, figurer, diagram och kurvor.

På bildskärmen får Du även felmeddelanden då någonting i programmet inte fungerar eller när Du gjort något annat fel (se sid 12 och 56).

1.2 Vad Du kan göra med ABC 80. Några exempel

ABC 80 för arbete, utbildning och avkoppling

Med färdiga program (ABC-program) samt program som Du själv utvecklar, har ABC 80 alla möjligheter att behandla olika data: läsa, lagra, beräkna och jämföra data av olika slag. Resultatet visas sedan i form av siffror, text eller grafiska diagram på bildskärmen. Kan också skrivas ut med hjälp av extra printer resp ritas med plotter.

ABC 80:s kvantitativa möjligheter begränsas endast av tillgänglig minneskapacitet (som dock kan byggas ut) och sökhastigheten hos kassetminnet. Sökhastigheten kan ökas genom att ansluta det betydligt snabbare flexskivminnet. Detta ger Dig och Din ABC 80 stora möjligheter till användning inom många områden:

På kontoret: för registrering (av kund-, lager-, reservdels-, medlemslistor och olika adressregister), för bokföring, försäljningsstatistik, resultatredovisning, m m.

I hemmet: för nöje och avkoppling med olika spel (luffarschack, mänlandning, schack, osv), registrering av Dina samlingar (LP, fri-

märken, inventarier), som hjälp vid deklarationen och läxläsningen i Na-ämnena, m m, samt för utveckling av egna program och datateknik som hobby.

Inom industrin: som hjälp vid beräknings- och konstruktionsarbeten, vid test av program och programdelar, som komponent i mät- och reglersystem för produktionsövervakning och styrning, som kvalificerad intern terminal till egen stordator, för intern utbildning i datateknologi och programmering.

I skolan: som hjälpmedel vid beräkningar och laborationer i naturvetenskapliga ämnen, företagsekonomi och statistik, för utbildning i datateknik och BASIC-programmering, som hjälpmedel i skolans administration för schemaläggning, inventarie- och läromedelsregister, osv.

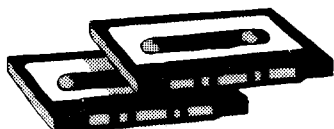
1.3 Köpta och egna program

Om Du kör köpta program (ABC-program)

Ett enkelt sätt att använda ABC 80 är med hjälp av färdiga program. Ett stort antal finns för bl a matematik, statistik, teknik och ekonomi. Inom ordbehandlingsområdet utvecklas kontinuerligt en programvara som förvandlar ABC 80 (med lämplig skrivare) till en ordbehandlingsmaskin. Bokföring, lagerredovisning, försäljningsstatistik är viktiga program som finns färdiga eller under framtagande.

Dessutom finns en rad olika spelprogram för t ex luffarschack, vanlig schack, månländning.

Gemensamt för dessa standardprogram är att de har mycket lågt pris eftersom de produceras i stora mängder.



Om Du gör egna program

För Dig som är förtrogen med BASIC och om standardprogrammen inte räcker till eller fyller Dina krav, är det lätt att göra egna program. Innan Du börjar programmera bör Du noga läsa igenom kap 4 "Kommandon, instruktioner och funktioner" i bruksanvisningen. Dessutom kan Du programmera i assembler och maskinkod, vilket ger ABC 80 stora möjligheter för kommunikation med yttre enheter, styrning/övervakning, avläsning av instrument, processer etc.

Tillhörande information

Standardprogrammen levereras på kassett eller flexskiva med utförliga program- och körbeskrivningar. Dokumenten beskriver dels allt vad programmet *kan* uträtta och dels *exakt hur* detta går till. Ett bra program har alltid "inbyggda körbeskrivningar" (dvs klartextanvisningar som visas direkt på bildskärmen). Därför kan Du lägga körbeskrivningarna åt sidan så fort Du kört programmet några gånger!



Litteratur

Kring ABC 80 utvecklas kontinuerligt litteratur och läromedel. "ABC om BASIC" tar upp BASIC från grunden och är en utmärkt "första-bok" för den som vill lära sig BASIC. Boken är skriven med utgångspunkt från ABC 80-datorn.

"Mikrodatorns ABC" beskriver pedagogiskt och ingående hur ett

mikrodatorsystem fungerar i praktiken. Även här använder författaren ABC 80 som exempel.

Ytterligare litteratur, kursmaterial och programpaket utvecklas kontinuerligt. Se litteraturlistan sid. 64.

1.4 Kringutrustningar

Ökade möjligheter med extra tillbehör

ABC 80 i standardutförande har en hel rad intressanta användningsmöjligheter. Det finns dessutom många tillbehör som ökar ut möjligheterna ytterligare.

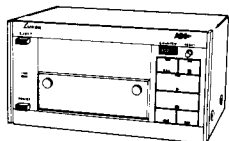
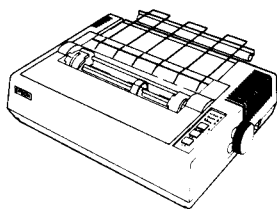
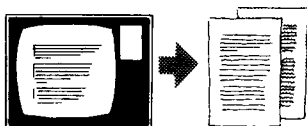
Tillbehörsprogrammet kommer successivt att ökas ut när nya användningsmöjligheter dyker upp och nya program utvecklas.

Skrivare. Vill Du bevara vad som kommer upp på bildskärmen, kan Du ansluta en skrivare till Din ABC 80. Här finns ett flertal modeller att välja mellan bl a:

Snabbskrivare med matrisskrift (5×7 punkter), 40–120 tkn per rad och s k traktormatning.

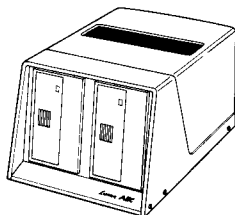
Skönskrivare av "typhjuls"-typ för utskrift av brev och dylikt där hög skrivkvalitet erfordras.

Utskrift av siffror och text



Kassetminne. Används för lagring av program och data. Kassetminnet står i förbindelse med datorn i tangentbordet och kan därifrån fjärrmanövreras för t ex automatisk bearbetning av data. Kassetminnet kan naturligtvis också användas manuellt. Räkneverket underlättar sökandet på bandet. På en 30 min kassett finns utrymme för ett 20-tal program eller flera tusen data-värden.

Snabbare sökning



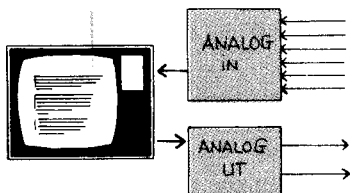
Flexskivminne. Sök- och inläsningshastigheten hos kassetminnet är låg (i synnerhet med fullskrivna 60 min. kassetter). Behöver Du i något sammanhang snabbare sökning av data, kan Du ansluta en dubbel flexskivenhet till ABC 80.

Större minneskapacitet



Flexskivor. Till flexskivminnena hör flexskivor. Dessa ger, utöver snabbare och säkrare sökning, även betydligt större minneskapacitet åt ABC 80.

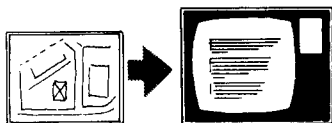
Anslutning av yttre enheter



Modulkort. Du kan lätt ansluta olika yttre enheter till Din ABC 80. För ändamålet finns f n ett stort antal modulkort.

Med hjälp av korten kan Du koppla till många yttre enheter, såsom mätinstrument och givare, regler- och styrutrustning, andra datorsystem, m m. Du kan också fördubbla ABC 80:s arbetsminne med extra minneskort.

Digitaliseringsbord

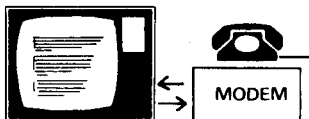


Digitaliseringsbord. Med hjälp av ett digitaliseringsbord (digitizer) kan Du överföra data från ritningar, kartor och diagram direkt till ABC 80 **utan** manuell inmatning! Exempel på användning: Beräkning av arealer, t ex tomtytter. Analys av diagram och kurvor, t ex bestämning av en kurvas matematiska beskrivning.

Grafisk presentation



Plotter. Vill Du ha en grafisk presentation av beräkningar och tabellvärden från Din ABC 80, kan Du ansluta en plotter. ABC 80-plottern ritar på vanligt papper med format upp till 189×250 mm. Upplösning ner till 0.125 mm.



Modem. För att överföra data på långt avstånd kan Du ansluta ett telefonmodem.

Med hjälp av telefonmodemet kan Du använda ABC 80 som Teledataterrminal, terminal till andra datorer eller som datainsamlingssystem där mätpunkterna ligger på stort avstånd från varandra.

Du kan välja mellan att köpa ett ABC 80-akustiskt modem eller att hyra modem från Televerket.

1.5 Dialogen mellan Dig och ABC 80

Via tangentbord och bildskärm kan Du kommunicera med Din ABC 80. De instruktioner Du ger och de data Du skriver in kommer samtidigt upp på skärmen som bekräftelse (och kontrollmöjligheter för Dig). Likaså kommer resultaten upp vartefter på bildskärmen.

ABC 80 kan också visa Dig vad Du skall göra under ett pågående program (instruktionerna är då inlagda i programmet). Det kan vara nödvändigt om Du har flera alternativa fortsättningar att välja mellan eller när Du har gjort något fel.

Felmeddelanden kommer alltid upp på skärmen om något är gale, exempelvis om internminnet är fullt, om radnummer saknas i en instruktion, om datorn inte hittar rätt fil eller inte förstår givna instruktioner, osv. Förklaring till felkoderna finns på sida 56.

Bildenheten

Bildskärmen har plats för 24 rader och maximalt 40 tecken per rad på skärmen. Texten rullas automatiskt uppåt när skärmen är full och nya rader tillkommer (s k scrolling).

Vad bildskärmen visar: Skärmen kan visa alla siffror, bokstäver och grafer (se sid 32). Med ABC 80 har Du förutom text också möjligheter att rita figurer, kurvor och grafiska bilder.

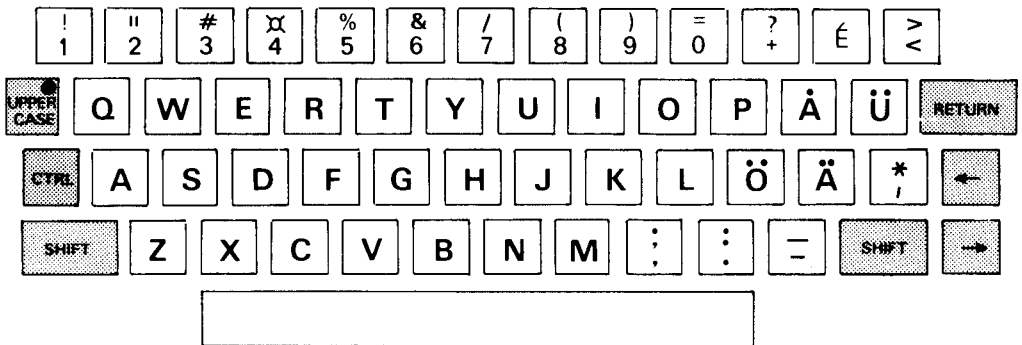
Hela bildskärmen delas då upp (med hjälp av en inbyggd grafisk mod) i 72 grafiska rader (numrerade från 0-71) med 78 grafiska positioner på varje rad (2-79).

Varje grafisk position på skärmen kan Du tända eller släcka med särskilda grafiska funktioner. Du anger då positionernas läge antingen med motsvarande koordinatpar eller om möjligt direkt med en matematisk funktion.

Tangenterna och hur de fungerar

För inskrivning av programinstruktioner och data har ABC 80 ett konventionellt tangentbord, uppställt enligt svensk skrivmaskinsstandard (se "Det här är ABC 80" på sida 4).

Inskrivningen sker som på en vanlig skrivmaskin, men här är det markören (en blinkande ruta) som markerar var nästa tecken skrivs in.



Utöver de vanligaste skrivtecknen finns ytterligare några tangenter, för olika programinstruktioner:

Valutatecknet används till att beteckna s k strängvariabler vid texthantering.

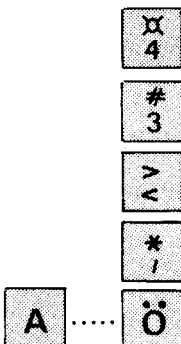
"Nummer"-tecknet används vid datafilhantering.

"Mindre än"-tecknet behövs i matematiska sammanhang liksom

"Större än"-tecknet.

"Asterisk"-tecknet används bl a vid multiplikation samt till markering av rubriker på skärmen.

Alla bokstäver, såväl stora som små finns på tangentbordet samt siffror.





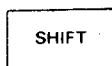
Kolon-tangenten används när Du vill skriva mer än en sats på varje rad.



%-tangenten används vid programmering för heltalsvariabler.



Semikolon-tangenten används bl a som förkortning av PRINT-instruktionen.



"SHIFT"-tangenten har samma funktion som på en vanlig skrivmaskin, dvs Du använder den när Du vill skriva STORA bokstäver eller övre tecknet på dubbeltangenterna. Skifftangenten hålls nedtryckt när Du skriver Dina tecken.

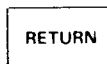


Stora bokstäver genomgående kan Du skriva in med UPPER Case-tangenten. Tryck en gång på UPPER CASE-tangenten. Alla bokstäver tolkas härfter som versaler (endast bokstavstangenterna A–Ö, samt É och Ü påverkas). Tangenten används t ex när Du vill skriva rubriker på skärmen.

(Den röda lampan inuti tangenten lyser efter första nedtryckningen; vill Du skriva som vanligt igen, trycker Du en gång till på UPPER CASE-tangenten).



"Kontroll"-tangenten används när Du måste skriva in speciella kontrolltecken. Tangenten hålls nedtryckt medan Du trycker på en teckentangent. (Koderna hittar Du under "tangentbordet" på sida 61).



"Vagnretur"-tangenten använder Du precis som på en vanlig skrivmaskin när Du skall börja med en ny rad (en rad kan vara upp till 120 tecken lång eller 3 rader på skärmen). Samtidigt får RETURN ABC 80 att ta emot den inskrivna raden. RETURN-tangenten använder DU också för att mata in data när Du gör ett program.



"Backstegs"-tangenten använder Du om Du har gjort ett felslag. Genom att trycka ned den backar Du tillbaka (ett steg för varje tryckning) och kan skriva över det som var fel.



Ytterligare tecken på raden skrivs ut om Du använder denna tangent i samband med ED-kommandot, (se sida 27).

1.6 Lagringsmöjligheter

Allt som kan visas på skärmen, kan också, om Du så vill, sparas, antingen lagrat på kassett, utskrivet på skrivare eller lagrat på flexskivor. Se avsnitt 1.4 "Kringutrustningar".

Kassettminnet

För att lagra program och data kan Du använda ett kassettminne. När ABC 80 läser programmet (eller texten) omvandlar den de inspelade signalerna till text och siffror. Program eller data lagras med c:a 5 sek. mellanrum. Om ABC 80 inte hittar information inom 10 sekunder ges felmeddelandet **ERR 21**.

Programmen till ABC 80. ABC-programmen är inspelade på kassetter av datakvalitet. Gör Du egna program, bör Du alltid välja kassetter av hög kvalité för att få säker inspelning.

Vid laddning av program från kassett har man medhörning i ABC 80:s högtalare.

1.7 Ytterligare "specialiteter"

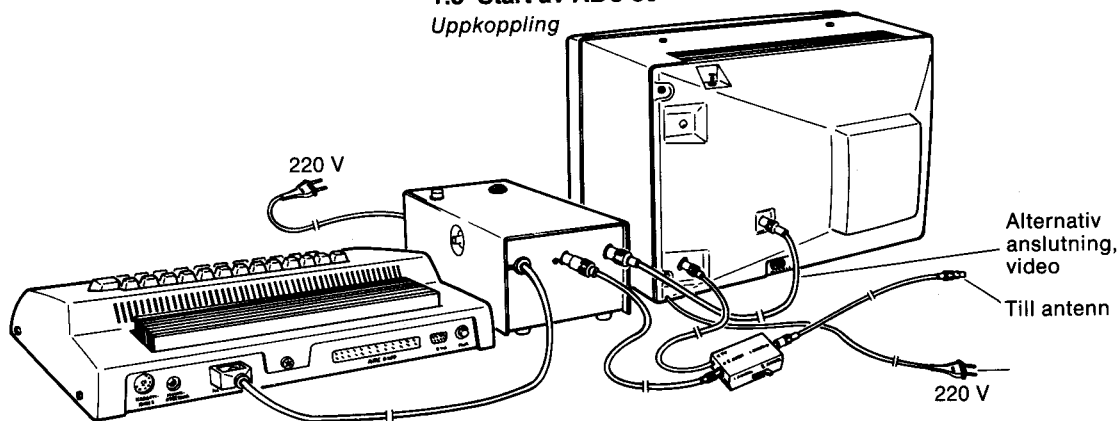
Realtidsklocka: I ABC 80 finns inbyggt en "kvartsklocka" som går när strömmen är påslagen. Du kan när som helst läsa av tiden från Ditt program, t ex för tidtagning eller start av anslutna yttre enheter. (Se kap. 3.4)

Ljudsignaler: Med hjälp av en inbyggd ljudgenerator, kan Du få fram ljudsignaler. Genom att blanda tre olika signalkällor kan Du bilda ett stort antal olika ljudeffekter. Dessa kan användas bl a som varnings-signal eller som kvittens på att ABC 80 har slutfört en uppgift.

Programmeringen av tongeneratoren beskrivs i kapitel 2.23.

1.8 Start av ABC 80

Uppkoppling



Anslut kabeln med den rektangulära kontakten från HF/Videoenheten till tangentbordet (kontakten går bara att ansluta på ett sätt).

Det finns två möjligheter att ansluta bildenheten (TV:n), dels till TV:ns antenningång (HF) och dels till TV:ns videoanslutning om sådan finns. OBS! Endast HF-kablar levereras med datorn. Figuren här intill visar båda alternativen.

- Anslut HF-omkopplarboxen till antenningången på TV:n med kabeln märkt "TV".
- HF-kabeln ansluts från HF/Videoenheten till HF-omkopplarboxens kontakt märkt "TV-spel".
- Om Du vill använda TV:n dels som bildenhet och dels som TV, kopplas TV:ns antennsladd till HF-omkopplarboxens andra uttag.
- På HF-omkopplarboxen finns en omkopplare som växlar mellan TV och dator. Ställ omkopplaren i datorläget.
- Ställ in TV:n på kanal 36 UHF.
- Anslut nätkontakterna från TV:n och HF/Videoenheten till vägguttaget. (OBS! 220V växelström).
- Slå till strömbrytarna på TV:n och HF/Videoenheten. När lamporna på HF/Videoenheten lyser, är ABC 80 påkopplad.
- När bildenheten är uppvärmd, visas texten ABC 80 på skärmen. Ställ in kontrast och ljusstyrka med hjälp av kontroller på TV:n.

Anslutningskontroll

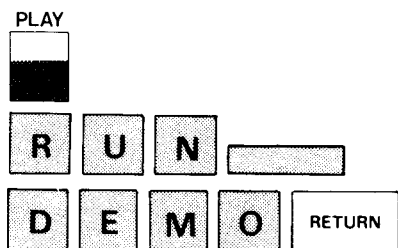
Har Du gjort något fel vid uppkopplingen indikeras felet och åtgärdas på följande sätt:

Lampan ("POWER ON") till höger på HF/Videoenheten lyser inte: kontrollera att det finns ström i vägguttaget samt att nätsladden till HF/Videoenheten sitter i ordentligt. Se till att strömbrytaren på HF/Videoenheten är tillslagen.

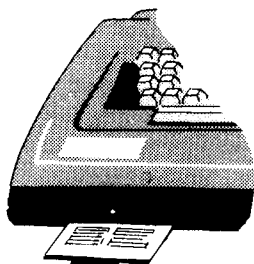
Lampan ("DATA ON") till vänster på HF/Videoenheten lyser inte: kontrollera att kabeln från HF/Videoenheten till tangentbordet är ordentligt ansluten.

Skärmen visar ingen bild: kontrollera att strömbrytaren för TV:n är tillslagen. Öka kontrasten. Räcker inte det, ökar Du ljusstyrkan med kontrollen på TV:n.

När apparaten är uppkopplad och kontrollerad är Din ABC 80 klar för programladdning.



ERR 21



```
10 FOR I%=0% TO 16383%
20 A% = A% + PEEK (I%)
30 NEXT I%
40 ;A%; I%
RUN
11273 16384
```

Se kontrollsiffran på undersidan av tangentbordet.

Programladdning från kassett

Prova först ett program från instruktionskassetten. Sätt in kassetten och återspola den.

Tryck ner "PLAY". Eftersom kassetminnets drivmotor styrs av ABC 80 händer ingenting förrän Du ger startkommandot från tangentbordet:

Skriv **RUN** och mellanslag samt programets namn (t ex **DEMO**) och tryck **RETURN**. Du kan även skriva **RUN CAS:**. Nu startar motorn och ABC 80 söker från kassetten upp programmet **DEMO** som nu laddas in i arbetsminnet och körs direkt. (Du hör i högtalaren om det finns ett program på bandet).

Erforderliga meddelanden och instruktioner får Du fortlöpande via bildskärmen. På instruktionskassetten finns ett flertal program.

ABC 80 hittar inte programmet: om ABC 80 inte hittar programmet i kassetten, kommer meddelandet "**ERR 21**" upp på bildskärmen. På referenskortet under tangentbordet hittar Du uttydningen av felkoden: "Hittar ej filen". Samtidigt med felmeddelandet hörs en ljudsignal från den inbyggda högtalaren i bildenheten.

Kolla så att Du satt in rätt kassett med rätt sida uppåt. Spola sedan tillbaka kassetten med återspolningsknappen. Drag sedan fram bandet manuellt tills det att Du ser det magnetiserade skiktet på bandet mitt framför bandöppningen på kassetten. När bandet stannat, skriv startkommandot **RUN DEMO** tryck **RETURN** och försök igen. Ett annat fel som kan uppträda vid programladdning är "**ERR 35**" (läs fel) se referenskortet sid 55. Samma åtgärd som ovan.

1.9 Felorsaker, åtgärder

ABC 80 uppför sig "konstigt": Om Du misstänker att ABC 80 ger felaktiga resultat eller bilden tillfälligt försvinner bör Du som första åtgärd göra följande:

Tryck **RESET** (återställning) på baksidan av tangentbordet.

Minnet för bildinnehållet nollställs och ABC 80 startas upp på nytt, bildskärmen töms och Du får upp **ABC 80** på skärmen.

Prova nu att köra programmet igen.

Är felet fortfarande kvar, kan Du skriva in följande enkla kontrollprogram (s k checksumma) i BASIC på tangentbordet: (Du behöver inte veta vad koderna i detalj betyder, därmed är det viktigt att Du trycker på **RETURN** efter varje rad!)

Kontrollprogrammet tar ca 15 sekunder att köra, varpå en kontrollsumma kommer upp på bildskärmen. Summan skall vara densamma som finns på etiketten på tangentbordets undersida, t ex 11273. Den andra summan på skärmen skall sedan alltid vara "16384". Om Du får fel kontrollsumma har Din ABC 80 troligen ett s k hårdvarufel dvs någon komponent (som t ex minne, processor eller logik) fungerar inte som den ska. Åtgärd: Kontakta service.

Är kontrollsiffrorna däremot riktiga kan det vara något s k mjukvarufel (fel i programmet). Åtgärd: Kontrollera programmet ytterligare. Hjälper inte detta kontakta service (ett hårdvarufel kan inte helt uteslutas även om Du fått riktiga kontrollsiffror).

Du kan även testa arbetsminnet med följande program:

```

10 REM MINNES-TEST PROGRAM * * * *
30 REM PROGRAMMET TESTAR ANVÄNDAR-MINNET I RAM.
60 REM E ÄR START-ADRESS PÅ TESTAREAN
70 REM S ÄR SLUT-ADRESS PÅ TESTAREAN
90 E=53248
100 S=61439
110 T%=85%
120 GOSUB 240
130 GOSUB 280
140 T%=170%
150 GOSUB 240
160 GOSUB 280
170 T%=255%
180 GOSUB 240
190 GOSUB 280
200 T%=0%
210 GOSUB 240
220 GOSUB 280
230 GOTO 360
240 FOR I=E TO S
250 POKE I, T%
260 NEXT I
270 RETURN
280 FOR I=E TO S
290 R%=PEEK(I)
300 IF R%<>T% THEN GOTO 330
310 NEXT I
320 RETURN
330 PRINT "FEL I BYTE PÅ ADRESS", I
340 PRINT R%, "ISTÄLLET FÖR", T%
350 GOTO 370
360 PRINT CHR$(7);"INGA FEL"
370 END

```

Testen tar ca 2,5 minuter.

Om ABC 80 klarar testen, skrivs "INGA FEL" på skärmen. Om inte så skrivs adressen till den felaktiga minnespositionen ut. Åtgärd: Kontakta service.

1.10 Ett programexempel

I avsnitt 1.8 beskrev vi hur Du kör ett program. Som Du märkte är det enkelt att hantera ABC 80 med färdiga programkassetter. Tack vare ett växande utbud av ABC 80-program har Du god chans att finna ett som passar just Dina behov. Har Du dessutom möjlighet att komplettera med egna program får Du troligen ännu större nytta och nöje av ABC 80.

I den här bruksanvisningen kommer vi inte att gå in på hur Du programmerar. Vi nöjer oss med att ge några exempel.

Om Du snabbt vill lära Dig BASIC-programmering på ABC 80, rekommenderar vi Dig i stället en lärobok för nybörjare "ABC om BASIC" som är skriven med utgångspunkt just från ABC 80.

Hur många sädeskorn?

Vi hämtar ett exempel från sagans värld:

Det fanns en gång en man i Orienten som betraktades som schackspelets uppfinnare. Hans kung som snart blev mycket förtjust i spelet utlovade därför en belöning till honom. Mannen begärde följande: 1 sädeskorn för första rutan, två för den andra, 4 för den tredje, åtta för den fjärde osv. "Det är ju bara 64 rutor", tänkte kungen och gick utan vidare med på förslaget. Som vi kanske anar fick kungen svårt att hålla sitt löfte.

I dag kan vi snabbt få ett svar med hjälp av ABC 80 hur många sädeskorn kungen egentligen hade lovat.

Program	Förklaringar
10 LET A=0	Totalt antal sädeskorn
20 LET B=1	Börja med ett sädeskorn
30 LET I=1	Första rutan
40 LET A=A+B	Lägg antalet sädeskorn i rutan till totala antalet korn
50 LET B=B *2	Fördubbla antalet sädeskorn
60 LET I=I+1	Nästa ruta
70 IF I<=64 THEN 40	Fortsätt lägga till på 64 rutor
80 PRINT A	Skriv ut totala antalet korn

Du kan utelämna instruktionen "LET".

Genom att använda ABC 80 får vi ögonblickligen reda på hur många sädeskorn det blev. Svaret blir 1.84465E+19. Vi får visserligen inte några exakta siffror men en god uppskattning av storleksordningen blir det ändå. Genom att använda oss av ASCII-aritmetiken i ABC 80 kan vi få upp till 29 siffrors noggrannhet. Vi ändrar en aning och programmet får då följande utseende: (Se ASC II-aritmetic 2.15).

```
10 LET A="0"
20 LET B="1"
30 LET I=1
40 LET A=ADD (A, B, 0)
50 LET B=MUL (B, "2", 0)
60 LET I=I+1
70 IF I<=64 THEN 40
80 PRINT A
```

Svaret blir 18 446 744 073 709 551 615 st ($=2^{64}-1$). Detta är det exakta antalet sädeskorn på alla rutorna tillsammans.

1.11 Programlagring

Sök upp det första lediga utrymmet på kassetten. Notera startnumret för varje program på kassetten. Det är lämpligt att Du börjar inspelningen direkt efter ett föregående program.

Tryck först på RECORD och PLAY. Bandspelaren startar dock inte förrän Du skriver **SAVE**, programnamnet och trycker på **RETURN**, t ex **SAVE JOHN** (där John är programnamnet) **RETURN**. När programmet är lagrat på kassetten stannar bandet automatiskt.

Om Du trycker ner tangenten STOP återgår tangenterna REC och PLAY i viloläge.

Som Du snart märker tar det inte lång tid att lagra ett program, vilket betyder att åtskilliga program kan rymmas på varje kassett. Men för att det skall gå snabbt att hitta de olika programmen bör Du inte lagra mer än 5–10 program per kassett beroende på hur långa de är. Viktiga program lagrar Du lämpligen på två kassetter, så att Du har en kopia om bandet skulle skadas.

När Du har lagrat ett program eller en mängd data, t ex en prislista, har Du skapat en fil som kan ha filnamn på högst åtta bokstäver. Filnamnet behövs för att Du ska hitta rätt bland programmen på kassetten.

OBS! Lagra programmen på endast den ena kassettsidan.

Start av program

Tryck först ner tangenten PLAY.

Skriver du **RUN DEMO** **RETURN** så letar ABC 80 på kassetten till den hittar en fil med namnet DEMO, läser in programmet och börjar köra det.

Om Du vill undersöka ett program före körning skriver Du istället **LOAD** och programmets namn, exempelvis **LOAD DEMO** **RETURN**.

ABC 80 läser då in programmet som tidigare, men utan att starta det. Du kan då t ex skriva **LIST** för att kontrollera programmet på bildskärmen och göra eventuella ändringar innan Du kör igång.

Se även Filnamn 2.14

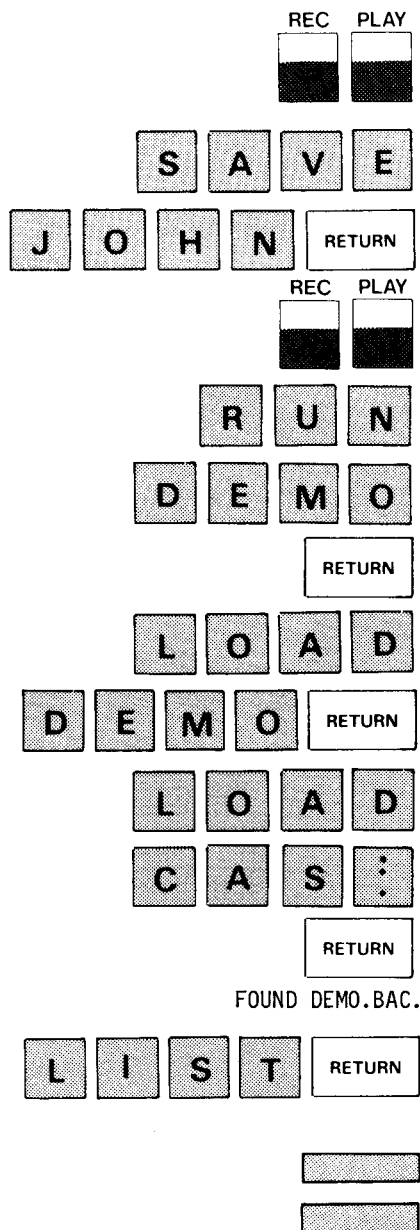
Om Du bara vill komma åt nästa program på bandet, utan att veta namnet, skriver Du **LOAD CAS:** eller **RUN CAS:** **RETURN** (**CAS** står för CASsette dvs kassett).

På skärmen visas då namnet på det program som ABC 80 plockat fram, t ex "**FOUND DEMO. BAC**" (**BAC** står för BASIC-program). Det kunde också ha stått t ex ".**TEXT**." efter programmet vilket hade visat att det var en **TEXT**-fil, oftast en datafil som ett program hade skrivit in på bandet.

1.12 Visning av program på bildskärmen

För att Du skall få upp ett program på skärmen använder Du **LIST**-kommandot.

Skriv **LIST** och tryck på **RETURN** varpå ABC 80 skriver ut upp till 23 radnummer av programmet (skärmen är då full) och stannar. Datorn arbetar med 120 tecken i raden. När Du har granskat färdigt de första 23 radnumren, "rullar" Du fram raderna som följer genom att trycka på mellanslagstangenten. Varje tryckning ger en rad till. Håller Du



RETURN			
L	I	S	T
	=	=	=
1	0	0	-
	"	=	=
	2	0	0
%	=	=	=
5	0	0	-
-)	=	
-	9	0	
)	%	
	9	5	
RETURN			

tangenten nedtryckt, "rullas" texten fram en rad i taget tills Du släpper tangenten. För att avsluta listningen trycker Du på **RETURN**.

Du kan också lista bara en del av programmet genom att efter **LIST** ange de rader Du vill ha listade.

LIST 100-200 ger raderna 100 till 200.

LIST 500- ger raderna från 500 och framåt.

LIST -90 ger raderna fram till och med rad 90.

LIST 95 ger enbart rad 95.

Tecknet "-" (minustecknet) kan också bytas ut mot ett komma ",", som ger samma resultat.

2. Introduktion till BASIC

2.1 Allmänt

Genom den snabbt växande användningen av datorer i vårt samhälle kommer allt fler människor i beröring med datorer och deras verksamhetsområden. Detta skapar ett starkt behov av utbildning inom datorområdet.

För att tillgodose detta utbildningsbehov skapades programmeringsspråket (**B**eginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode) med målet att vara enkelt att både lära och använda.

BASIC är:

- ett programmeringsspråk med goda möjligheter till kommunikation mellan maskin och människa. Det är ett dialogspråk, som ger användaren den så nödvändiga och värdefulla direktkontakten med datorn.
- är enkelt att lära därför att det innehåller relativt få satser och kommandon med logiska och entydiga användningsregler.
- är ett generellt programmeringsspråk som är oberoende av vilken dator man använder. I praktiken skiljer sig de olika BASIC-versionerna från varandra i fråga om kvalitéer och omfattning.

ABC 80 BASIC

Den omfattar både den europeiska standarden ECMA 55 och den amerikanska ANSI-standard, vilket ger en garanti för att många program från andra BASIC-versioner direkt kan köras på ABC 80. Du kan hämta program från tidskrifter och böcker, både för hobby och professionellt bruk. Genom de användbara heltalsvariablerna ökas snabbheten väsentligt.

En viktig egenskap hos ABC 80 BASIC är att den är lätt och bekväm att använda. Den kraftfulla editeringsfunktionen ED hjälper Dig till sekundschnabba rättningar av en programrad. I tidigare BASIC-versioner måste hela programraden skrivas om även om bara ett enda tecken skall ändras!

För den som inte tidigare stiftat bekantskap med BASIC rekommenderas programmeringshandboken "ABC om BASIC" som behandlar BASIC från grunden med utgångspunkt just från ABC 80. (Se under "Litteraturförteckning" på sida 64).

ABC 80 BASIC ligger lagrad i en BASIC-tolk bestående av 4 st ROM-kretsar (läsminnen) och står till förfogande omedelbart efter inkopplingen av apparaten.

Denna BASIC-tolk innehåller flera egenskaper vilka underlättar programmeringen t ex

ED	För korrigerig av rader i ett program
TRACE	Ger möjlighet att följa programexekveringen rad för rad
MERGE	Möjliggör sammanlänkning av flera program
ASC II-aritmetik	för beräkningar med upp till 29 siffror

Även mycket kraftfulla stränghanteringsfunktioner ingår t ex **LEFT** □, **MID** □, **RIGHT** □ och **INSTR** vilket möjliggör sökning i och hantering av textsträngar.

2.2 Program

Vad är ett BASIC-program?

Ett datorprogram är en serie instruktioner som talar om för en maskin hur inmatade data behandlas.

Ett BASIC-program innehåller en lista av exakta instruktioner, som skall utföras i bestämd ordningsföljd, av datorn. Listan består här av ett antal rader med instruktioner eller satser där varje sats börjar med ett programradnummer. (I ABC 80 BASIC får flera instruktioner förekomma på samma rad om de skiljs åt med ett kolon (:)) OBS! Gäller ej **DATA**-satser.)

BASIC-språket har entydiga och klara regler för formulering av dessa satser. Detta är nödvändigt, eftersom datorn saknar intuition och utför exakt vad den är instruerad att göra.

Så här används BASIC-satserna

Datorns arbete kan sammanfattas i några moment: den kan läsa data, beräkna data, lagra data, skriva data, jämföra data och hoppa i programmets instruktionsföljd.

LET	– satsen används för att ge en variabel ett värde.
READ	– satserna används för att ge en eller flera variabler ett värde, från en DATA-sats.
DATA	
ON-RESTORE	
RESTORE	
INPUT	är instruktioner för överföring av data; till och från datorn, till skrivaren, osv.
PRINT	
INP	
OUT	
PREPARE	
OPEN	
CLOSE	
GOTO	Ovillkorligt hopp
IF-THEN	villkorligt hopp
ON	– satser; villkorligt hopp
FOR ...	Program-loopar (benämning på ett programavsnitt, som genomlöps ett visst antal gånger)
NEXT	
GOSUB	För hopp till subrutiner och återgång.
RETURN	
ON ERROR GOTO	För att hantera olika typer av fel (t ex fel indata) och ge egna felmeddelanden finns dessa möjligheter.
ERRCODE	
PEEK	För att kombinera BASIC-program med maskinkod
POKE	
CALL	
STOP	För att söka fel i ett program.
TRACE	
NOTRACE	
REM	För att skriva kommentarer i programmet.

2.3 Kommandon

Du finner i ABC 80 BASIC s k kommandon, som efter inmatning från tangentbordet och tryck på **RETURN** utförs direkt, t ex

RUN	exekverar ("kör") programmet som finns i minnet. Alla variabler nollställs.
LOAD	söker ett program (fil) på kassetten, varefter inläsning sker, t ex LOAD CAS: KORN .
SAVE	lagrar programmet som finns i minnet på kassett eller flexskiva, t ex
LIST	SAVE CAS: FINANS eller LIST CAS: FINANS .
SCR	raderar programmet i minnet.
NEW	Det är viktigt att detta kommando utförs innan ett nytt program skrivs in i minnet, eftersom det nya programmet i annat fall blandas med det gamla.
REN	omnumrering av rader i programmet.
LIST	genererar en utskrift av programmet som finns i minnet.
MERGE	hämtar program från kassett utan att radera det program som finns i minnet med andra radnummer. Alla till ABC 80 hörande (instruktioner), kommandon, funktioner, operatorer och specialtecken beskrivs i detalj i kap. 4.

2.4 Utrymme

I grundutförande har ABC 80 16 kbyte RAM. Det betyder att 16 000 tecken, siffror eller bokstäver, ryms i ABC 80:s internminne (således ej på kassetten). Det betyder inte att Du själv har tillgång till 16 000 tecken, utan ABC 80 använder en del internt.

Hur mycket ledigt minne finns kvar?
skriv (på en rad)

PRINT PEEK(65064) * 256 + PEEK(65063)
— PEEK(65057) * 256 — PEEK(65056) RETURN

ABC 80 svarar med antal bytes som ej upptas av program eller data. Överskrider Du utrymmet i ABC 80 får Du automatiskt ett felmeddelande om detta. T ex **ERR 3** betyder då att minnet är fullt och **ERR 7** talar om att Du har använt för stort heltal. Räcker inte utrymmet till för Ditt program, kan Du bygga ut minnet genom att ansluta minneskort, eller använda Dig av s k **CHAIN**-instruktioner.

2.5 Aritmetiken i ABC 80

0.1E-127 till 0.999999E127	Vid numerisk behandling kan ABC 80 hantera talvärden från 0.1×10^{-127} till 0.999999×10^{127} . Resultaten av alla beräkningar visas avrundade till 6 siffror.
—32768 till +32767	För de s k heltalvariablerna är kapaciteten från —32768 till +32767. Dessa heltalsvariabler används när Du vill ha mycket snabba program (s k iterativa beräkningar med många loopar). Numeriska data kan Du mata in i ABC 80 på tre olika sätt: 147 som heltal, 153.79 som decimaltal (Obs: decimalpunkt), 181E-2 eller i 10-exponentform (flyttal). 181E-2 är detsamma som 181×10^{-2} . För noggrannare beräkningar upp till 29 siffror kan Du använda ASCII-aritmetiken.

2.6 Enkla variabler

ABC 80 BASIC är avsedd för alfanumerisk databehandling vilket innebär att data som skall behandlas och skrivas ut kan vara både siffror och bokstäver.

I ABC 80 kan både alfabetiska data (vanliga ord) och numeriska data (siffror och tal) uppträda som konstanter eller variabler.

Vilken räknemaskin som helst kan behandla konstanta tal, med ABC 80 kan Du också instruera och arbeta med variabler. Konstanten behåller sitt värde genom hela programmet, medan variablen kan ändra värde många gånger under programmets gång.

Flyttalsvariabler. Flyttalsvariabler använder Du när Du arbetar med tal med decimalpunkt (dvs icke-heltal) eller med mycket stora tal.

A
Z1 **Ö9** Flyttalsvariabler betecknas i ABC 80 BASIC med **en** bokstav (från A till Ö) eller **en** bokstav och **en** siffra (mellan 0 och 9).

En flyttalsvariabel kan ha värden från $\pm 0.1 \text{ E} - 127$ till $\pm 0.999999 \text{ E} + 127$. (E betecknar 10-exponenten, dvs $0.1 \text{ E} - 6$ svarar mot 0.1×10^{-6}).

Resultaten av alla beräkningar avrundas automatiskt till 6 siffror.

I%
Y1%
Ö9% *Heltalsvariabler.* En heltalsvariabel betecknas som en flyttalsvariabel åtföljt av %. Heltalsvariablerna kan ha heltalsvärden mellan -32768 och $+32767$. Heltalsvariabler går avsevärt snabbare att räkna med än flyttalsvariabler och tar dessutom mindre plats i minnet. Du kan därför använda dem som loopräknare och för indexering m m, eller överhuvud taget när Du inte behöver räkna med tal med decimalpunkt.

2.7 Indexerade variabler

För att bearbeta större datamängder, listor, matriser eller datamängder, som är ordnade på något sätt, använder Du indexerade variabler eller fält.

B(0), B(1), B(2),
B(3), B(4) En indexerad variabel betecknas som en heltalsvariabel eller flyttalsvariabel, men med en eller två heltal inom parentes. T ex en lista med tal betecknas som **B(I)**, där I antar värden 0, 1, 2, 3, 4

B(0) B(1) B(2)
B(3) B(4) Om Du inte specificerar det andra av de två möjliga heltalen inom parentes uppfattar ABC 80 den indexerade variabeln **B(I)** som en 1-dimensionell matris.

Variabeln **B(0)** avser första variabeln i fältet B och **B(1)** det andra, osv. Denna metod att referera till elementen med variabelns nummer inom fältet kallas för *indexering* och numren kallas **index**. Index utgörs av ett tal, en variabel eller ett aritmetiskt uttryck och måste vara positivt. Elementen kan vara flyttal, heltal eller strängar. Flyttalen tar 5 byte per element, heltalen 2 byte.

M(R, K) En 2-dimensionell matris **M(R,K)** är ett fält som kan illustreras grafiskt som en rektangel. Första index anger höjden och andra index anger bredden:

M(0,0)	M(0,1)	M(0,2)M(0,K)
M(1,0)	M(1,1)	M(1,2)M(1,K)
M(2,0)	M(2,1)	M(2,2)M(2,K)
M(R,0)	M(R,1)	M(R,2)M(R,K)

80 DIM Z(7), X(3,4), B(11,2)
90 DIM Y(50)

10 B1 α = "NUMMER OCH NAMN"

A α

K α

P2 α

B α (5)

X α (N)

Z α (M, N)

B och B α

DIM B α (R)

DIM B α (R) = M

DIM B α (R, K)

DIM B α (R, K) = M

Index kan här endast anta heltalsvärden (om index ej är heltal avkortas de till heltal).

DIM-satsen. En **DIM**-sats använder Du till att definiera max antal värden i en matris eller sträng.

DIM-satsen har formen

DIM variabel (n) , variabel (n,m)

Om en indexerad variabel används **utan DIM**-sats kan man använda index från 0 till 10.

I kapitel 2.17 omtalas hur man använder algebraiska uttryck i **DIM**-satser.

2.8 Strängar och fält

En numerisk konstant beskriver ett bestämt tal. När det gäller godtyckliga karaktärer (bokstäver och andra tecken) dvs text, används begreppet sträng.

En sträng representerar en viss följd av tecken, t ex en bokstav eller ett eller flera ord. En strängkonstant skall alltid omges av citations-tecken (") eller apostrof (').

B1 α har alltså värdet **NUMMER OCH NAMN**.

Analogt med det numeriska fallet finns även variabler för bokstäver och tecken, s k alfanumeriska variabler eller strängvariabler.

Strängvariabler. En strängvariabel betecknas med en bokstav (valfritt från A till Ö) eller en bokstav och en siffra (0 till 9) åtföljt av valutatecknet α (sol; svarar mot \$ i amerikansk datorlitteratur).

Liksom i det numeriska fallet finns det 1- och 2-dimensionella matriser med element bestående av strängar.

Du kan använda samma namn till en numerisk variabel som till en strängvariabel. (Se även sida 24).

får således förekomma i samma program.

DIM-satsen definierar storlek och antal för strängar och fält.

Definierar en strängvektor med R+1 strängar:

B α (0) – B α (R). Varje sträng får ha max 80 tecken.

Definierar en strängvektor (1-dimensionell matris) med R+1 strängar. Varje sträng får ha max M tecken.

Definierar en 2-dimensionell matris med (R+1) * (K+1) strängar. Varje sträng får ha max 80 tecken.

Definierar en 2-dimensionell matris med (R+1) * (K+1) strängar. Varje sträng får ha max M tecken.

2.9 Satser och uttryck

Ett aritmetiskt uttryck bestående av operatorer, tal och numeriska variabler är dels order till datorn att beräkna uttryckets värde, dels en regel för hur detta går till.

Aritmetiska uttryck

Aritmetiska uttryck erhåller t ex i aritmetiska satser formen $V = a$, där V är ett variabelnamn och a är ett aritmetiskt uttryck. Några exempel på aritmetiska uttryck:

2.71828, 3E+6, 12%	konstanter
T3, A(9), R1	variabelnamn (variabler)
B+C, 6.28 * F, S/T	uttryck med aritmetisk operator
SIN (T), EXP(B-12.3)	funktionsuttryck
(X+Z), (TAN(A+B))	aritmetiska klammeruttryck
9.1 * (R-(5+2)/A)	allmänt aritmetiskt uttryck

Aritmetiska operationer

För de vanliga räknesätten finns i ABC 80 följande operatörer:

Prioritet	Operator	Innebörd
1	Ü eller * *	exponentiering
2	*	multiplikation
2	/	division
3	+	addition
3	-	subtraktion

Operatorernas prioritet avgör i vilken ordning en beräkning utförs; m a o i ett aritmetiskt uttryck utförs först exponentieringsoperationer, sedan multiplikationer/divisioner och sist additioner/subtraktioner. Vid lika prioritet utförs operationerna från vänster till höger. Vid uttryck med parenteser utförs operationen inom parentesen först, vid flera parentesnivåer börjar man med den innersta nivån. Exponentiering, Ü, kan även utföras med operatören ** på ABC 80.

När båda operanderna i en division utgörs av heltalsvariabler (%-tecken) erhålls en heltalsdivision. Det innebär att decimalerna bortfaller så att $5\%/3\%=1\%$. Vid exponentiering av negativa tal måste exponenten vara ett heltal.

ex: $(-1)\text{Ü}2\%$

2.10 Jämförelseuttryck och jämförelseoperatorer

Jämförelseuttryck (logiska uttryck) erhåller Du genom att kombinera aritmetiska uttryck, variabler eller tal med jämförelseoperatorer. Värdet av ett jämförelseuttryck kan vara antingen "SANT" eller "FALSKT".

Jämförelseuttryck använder Du i ett program för att utföra villkorliga hopp.

Jämförelseoperatorer:

Prioritet	Operator	Innebörd
4	>	större än
4	>=	större än eller lika med
4	=	lika med
4	<=	mindre än eller lika med
4	<	mindre än
4	<>	skilt från

Ett jämförelseuttryck kan också utgöra en del av ett aritmetiskt uttryck, t ex

$$S = 10 * T + 4 * (Z > 100)$$

Ett **SANT** jämförelseuttryck har det decimala värdet -1 motsvarande binärt 11111111 11111111 och ett **FALSKT** motsvarande decimalt 0, binärt 00000000 00000000.

Boolska uttryck och logiska operatörer
 Jämförelseuttryck (Boolska uttryck) kan Du med hjälp av logiska operatörer sammansätta till större jämförelseuttryck. Värdet är antingen **SANT** eller **FALSKT**.

NOT
AND
OR
XOR
IMP
EQV

Prioritetsnivån vid beräkning av ett uttryck är 5. Vid sammansatta jämförelseuttryck skall parenteser användas.

Exempel
100 IF NOT A > B THEN 220
110 IF A > B AND C = D THEN 200
120 IF X < Y OR B > D THEN 210
130 IF M = N X OR S > T THEN 30

Logiska operationer på "bit-nivå"
 Logiska operationer fungerar på ABC 80 binärt, m a o kan Du behandla och analysera data på bit-nivå. Avsnitt 2.22 beskriver en bits operationer på V24-anpassningen (interfacet).

2.11 Stränguttryck

På ABC 80 kan Du använda både numeriska uttryck och stränguttryck.

Andra exempel på stränguttryck:

"NAMN", "LUXOR"	strängkonstant
A □, N4 □, P □ (5 %)	strängvariabel
V □ + N □	stränguttryck med
P □ + "–PROGRAM"	länkningoperator
LEFT □ (H □, 2%)	strängfunktion

En specialitet i ABC 80 är "ASCII"-aritmetiken.

I avsnitt 2:15 beskrivs hur Du utför numeriska beräkningar med strängvariabler och konstanter.

LEFT □	funktioner för sträng-
MID □	hantering. Med en ansluten
RIGHT □	skrivare lämpar sig ABC 80
LEN	för textbearbetning.
INSTR	
SPACE □	
STRING □	

Stränguttryck och jämförelseoperatorer

Jämförelseoperatorer kan också användas mellan två stränguttryck. Därvid jämförs ASCII-koden för strängarna tecken för tecken från vänster till höger. Jämförelsen avbryts då två olika tecken påträffas. Om det ena uttrycket "tar slut" före det andra betraktas det som det mindre av de båda.

Ex. **IF B** □ = "JA"
THEN 140

IF ...THEN använder Du bl a
 för alfanumerisk sortering.
 Vid jämförelser erhåller A
 lägsta värdet, B de näst lägsta
 etc. (Se ASCII-tabellen sid 32.)
 Ex. "A" < "B" SANT
 "A" = "B" FALSKT
 ("B" har större ASCII-värde än "A")

2.12 Läsning av strängar

Strängkonstanter kan uppträda som aritmetiska konstanter i **DATA**-satser och läsas med hjälp av **READ**-satser.

- 100 READ A** □, **B** □ Strängkonstanterna behöver inga citationstecken om Du inte har blanktecken eller (,) i strängarna.
- 210 INPUT P** □, **S** □ (1%) För inmatning av strängkonstanter i ett program använder Du en **INPUT**-sats.
- 220 INPUTLINE L** □ (5) Vid inmatning av strängar är det ofta praktiskt att använda **INPUTLINE**. Vid inmatning med **INPUTLINE** sparas mellanslag och tecken för vagnretur och radframmatning i slutet på strängen.

Utskrift av strängar

- 70 PRINT "NAME", A** □ **PRINT**-satsen används som i det numeriska fallet. Glöm inte citationstecknet vid alfabetiska konstanter.

2.13 Variabelbeteckningar

Variabler betecknar Du i ABC 80 antingen med en bokstav eller med både en bokstav och en siffra.

Samtliga bokstäver utom É och Ü får användas. (Bokstaven O bör undvikas p g a likheten med siffran 0 (noll).)

Siffror från 0 till 9 kan användas. Heltalsvariabler erhåller dessutom ett %-tecken och strängvariabler ett □-tecken.

Ex. på variabler:

A, B1, Ö9	Flyttalsvariabler
T%, D5%	Heltalsvariabler
A □, F1 □	Strängvariabler (avsnitt 2.8)
R(2), A(T%), S(3,1)	Indexerade flyttalsvariabler (avsnitt 2.7)
S%(1), A%(N%), F3%(I)	Indexerade heltalsvariabler (avsnitt 2.7)
B □ (5), V1 □ (M%)	Indexerade strängvariabler (avsnitt 2.8)
FNA, FN%	Funktioner

Variablerna **A**, **A%**, **FNA**, **FNA%**, **A(I)** och **A** □ kan Du således använda samtidigt.

2.14 Filnamn

Ett filnamn består av tre delar: enhet, namn och filtyp. Enheten anger var filen finns, den kan vara **CAS**: (kassett), **DRO**: eller **DR1**: eller **PR**: (skrivare). Om enheten utelämnas antas vanligen **CAS**:. Om flexskiva finns ansluten antas i stället först **DRO**: därefter **DR1**..

Namnet består av högst 8 bokstäver eller siffror. Första tecknet måste vara en bokstav.

Efter **PR**: behövs varken namn eller filtyp. Om namnet utelämnas efter **CAS**: används den första påträffade filen på kassetten.

Filtypen består av en punkt följt av tre bokstäver. Om filtypen utelämnas antas följande:

SAVE -sats	.BAC
LIST -sats	.BAS
LOAD -sats	.BAC eller .BAS
UNSAVE -sats	.BAC eller .BAS

Exempel: **LUFFAR.BAC**
DRO :BREV.BAS

2.15 ASCII-aritmetiken

ASCII-aritmetiken använder Du när Du har många siffror i en beräkning, tex i ekonomiska kalkyler med stora belopp. Du kan också använda den för att få önskat antal decimaler vid utskrift av resultaten.

ADD ☒
SUB ☒
MUL ☒
DIV ☒
COMP %

Funktionerna består av de fyra räknesätten: addition, subtraktion, multiplikation och division samt numerisk jämförelse av två talsträngar.

Talsträngarna får innehålla högst 29 siffror plus eventuellt tecken (+) eller (–) och decimalpunkt, men ej exponent. Speciellt vid division kan det vara nödvändigt att minska antalet decimaler i resultatet för att undvika spill (overflow). Nämnaren expanderas då med antalet angivna decimaler + 1.

PRINT DIV ☒ ("2","3",26)

Skriv ut 2/3 med största möjliga antal decimaler: först täljaren, därefter nämnaren, åtföljt av antalet decimaler i resultatet. Divisionen ger

0.666666666666666666666666666667 där den sista (26:e) siffran är avrundad.

PRINT MUL ☒ ("1234567789",
"123456789012345677890",0)

Vid multiplikation med största möjliga antal siffror skriver Du: vilket ger resultatet **152415787517147887501905210**.

2.16 Användning av ABC 80 som kalkylator

Du kan också använda ABC 80 som avancerad kalkylator. Flera BASIC-satser kan nämligen användas direkt om Du utelämnar radnumret.

P R I N T
1 7 5 * 4 7
RETURN

Beräkna tex 175×47 . Du skriver **PRINT 175 *** (gångar) **47** och trycker **RETURN** och svaret "**8225**" kommer omedelbart upp på skärmen. (I stället för **PRINT**-satsen kan Du också använda förkortningen ; (semikolon).

Ett annat exempel: Beräkna $\sin \frac{\pi}{4}$:

Du skriver **PRINT SIN (PI/4)** **RETURN** och svaret blir **0.707111**.

Genom att använda BASIC-satserna direkt och utan radnummer har Du också fått ett utmärkt hjälpmedel för programutveckling, testning och rättning av program och programdelar. Är Du osäker hur en viss sats fungerar i Ditt program, kan Du ju enkelt pröva den och få direktbesked utan att starta programmet.

2.17 Programmeringstips

Flera satser på en rad. I ABC 80 kan man skriva flera BASIC-satser på en programmerad rad. Detta spar plats och ger snabbare körning. Programraden får bli 120 tecken lång inkl mellanslag. Satserna åtskiljs med kolon (:)

Gör 3 radframmatningar

30 PRINT : PRINT : PRINT

Fråga och svar

40 ; "Vad heter Du" : INPUT A ⌘ (Semikolon är förkortning för PRINT).

Du kan skriva kommentarer direkt efter en sats

40 GOSUB 200 : REM Inläsnings-rutinen

Utskrift av citationstecken. Man kan som strängavgränsare använda både 'enkla' och "dubbla" citationstecken. Detta gör det möjligt att på ett enkelt sätt **skriva ut** även dessa specialtecken.

Utskrift av enkelt citationstecken:

PRINT "Han hette O'Rourke"

ger:

Han hette O'Rourke

Utskrift av dubbelt citationstecken:

PRINT "'Aahh", sa han, "en ABC 80!"

ger

"Aahh", sa han, "en ABC 80!"

Man kan också skriva citationstecknet två gånger:

PRINT "12" "TV-SKÄRM"

ger

12" TV-SKÄRM

DIM A(N). Det är möjligt att ha uttryck i en DIM-sats. Detta kan vara värdefullt när man testar program.

Genom att ändra rad 10, anpassas **hela programmet** nedan till en viss storlek på fältet A.

Ex. 10 N = 10 : REM Nuvarande storlek

20 DIM A (N)

30 DIM B (2 * N, 2 * N)

40 FOR I = 1 TO N

. .
. .
. .

I stället för att ändra på alla rader som beror på A:s dimension behöver man bara ändra en enda rad!

Se även boken "ABC om programmering och dokumentation".

2.18 Rättelser av fel i programmet

I den händelse att ett nytt program innehåller fel så måste felen rättas till. Det finns två typer av fel:

Formella fel. Du har använt BASIC-språket på ett icke tillåtet sätt. ABC 80 upptäcker sådana fel under själva programmeringen. (När Du trycker **RETURN** ger ABC 80 då ett felmeddelande (se "Felmeddelanden" på sida 56).

Logiska fel. Logiska fel i programmet är i allmänhet svårare att rätta

```

10 REED
10 RE
A D
10 READ

CTRL X

1 0 RETURN

E D 1 0 RETURN
10 For I = 1 TO 10000
■
10 FOR I ■
10 FOR I% ■

10 FOR I% = 1 TO 10000■

10 FOR I% = 1 TO 100■
RETURN

```

till än formella fel. Programmet går oftast att köra men ger felaktiga resultat eller inga resultat alls.

Rättelse av felaktiga tecken. För att omedelbart kunna rätta ett felinslag utan att skriva om hela raden, använder Du backtangenter för att gå tillbaka till felet och skriver sedan över det felaktiga tecknet, t ex **10 READ A**.

Rättelse av ej avslutad rad. För att ta bort en felaktig, oavslutad rad använder Du kontrolltangenter **CTRL** och skriver X. (Jämför även back-tangenten i **nedtryckt** läge).

Rättelse av programsats. En felaktig programsats kan Du ta bort så här:

skriv in radnummer och tryck **RETURN** . T ex om hela raden **10 LET A = 1** skall bort skriv **10 RETURN** .

Vill du däremot ändra i en färdigskriven programrad, kan Du enkelt göra det med hjälp av **ED**-kommandot.

Säg att raden **"10 FOR I = 1 TO 10000"** skall ändras till **"10 FOR I% = 1 TO 100"**.

Skriv först **ED 10** och tryck på **RETURN** och raden kommer upp på skärmen.

Flytta sedan markören med hjälp av pil-tangenten (→) till tecknet som skall ändras/kompletteras (här "I").

Skriv sedan % (shift-tangenten nedtryckt samtidigt)

Nu är den första ändringen gjord.

Flytta nu markören till positionen där nästa ändring skall göras, dvs efter 10000.

Backa sedan tillbaka två gånger och de båda sista nollorna raderas ut. Tryck **RETURN** och hela raden är rättad.

2.19 Praktiska råd

Eftersträva att:

- Förenkla handhavandet
- Öka snabbheten
- Minimera minnesutrymmet

Dessa metoder är generella för all programmering. Vilket sätt som är det bästa beror på användarens speciella behov. Man kan inte förbättra ett program maximalt på alla tre punkterna. Det går t ex inte att minimera minnesutrymmet för programmet samtidigt som man förenklar handhavandet. För att göra ett program extra lättanvänt måste man ta minnesutrymme i programmet för förklarande text, uppmaningar och frågor till användaren.

Förenkla handhavandet av ett program. För att ett program ska bli lättanvänt måste datorn – via bildskärmen – ge användaren instruktion steg för steg om vad som skall matas in. Programmet skall alltså förse med "inbyggda körinstruktioner" som i klartext talar om för användaren vad som skall göras.

När datorn ställer en fråga på bildskärmen, bör de möjliga svarsalternativen också anges, även om det bara är ja/nej.

Finns det många alternativ kan det vara lämpligt att låta datorn skriva en s k meny på skärmen. Detta betyder att användaren får upp de olika fallen på skärmen med en siffra eller en bokstav till varje alter-

nativ. Användaren väljer ett alternativ genom att ange motsvarande bokstav eller siffra.

Ex. Vad vill Du göra?

Ange motsvarande siffra:

- 1 – Budget
- 2 – Statistik
- 3 – BASIC-kurs
- 4 – Spel
- ?

Vill Du välja "statistik", slå in 2 och tryck på **RETURN**

Öka snabbheten i ett program: Ibland kan det vara önskvärt att använda ett program, som går så fort som möjligt. Text om programmet används ofta och är mycket stort. För att uppnå största snabbhet i ett program måste en del goda programmeringsregler brytas, som gör att programmet kan bli svårare att förstå sig på, att söka och rätta till fel i och överhuvud taget modifiera.

Trots att ABC 80 som sådan är en snabb dator behöver man ibland göra programmen snabbare.

Här är några tips, som kan användas för att snabba upp ett program:

- Använd i första hand heltalsvariabler (se sida 20) om det bara är hela tal, som skall behandlas. Sålunda bör programmering av loopar göras med heltalsvariabler. Snabbheten kan i gynnsamma fall ökas 2 el 3 gånger.
- Om en subrutin används endast en eller två gånger bör den skrivas ut som del i huvudprogrammet (**GOSUB** och **RETURN** tar tid).
- Använd **FOR**-loopar vid programmering av loopar.
- Försök få in så många satser som möjligt på samma programrad (1 programrad = max 120 tecken). Satserna skiljs åt med kolon (:).

Försök i första hand använda ovanstående tips i programdelar som upprepas många gånger i ett program, text i loopar! Om en loop genomlöps 1000 gånger och man kan spara 1 sekund vid varje varv sparar man ju totalt 1000 sek eller nästan 17 minuter!

Minimera minnesutrymmet för ett program. Vad ska man göra om programmet inte får plats i ABC 80:s minnesutrymme? Om det inte är aktuellt att öka minneskapaciteten finns inget enkelt svar på frågan. Naturligtvis är tipsen i "Öka snabbheten i ett program" användbara. Här är speciellt heltalsvariablerna värdefulla, om man vill spara på minnet! Genom att använda heltalsvariabler i stället för flyttalsvariabler kan i gynnsamma fall flera tusen extra datavärden få plats i ABC 80:s minne! Det är också viktigt att **DIM**-satsen används på rätt sätt så att ej onödigt minnesutrymme reserveras för variablerna i programmet. Alltså:

- Använd heltalsvariabler i möjligaste mån.
- Använd subrutiner överallt där så är möjligt.
- Skriv flera satser på samma rad.

Här följer ytterligare några tips:

- Försök om möjligt använda **ON**-satser, text **ON... GOTO** eller **ON...GOSUB** vid flerval.
- Använd **IF-THEN-ELSE** med flera satser på varje villkor.

- Initiera ej variabler genom att använda DATA-satser. Läs i stället in motsvarande värden från datafil på kassett eller flexskiva. Då sparar man datasatsernas programutrymme.
- Dela upp programmet med **CHAIN** (Kapitel 3.2).

Det är ofta en ganska tidsödande process att försöka minska minnesutrymmet för ett program. Om programmet får plats i minnet och det fungerar felfritt finns sällan anledning att minimera programutrymmet.

2.20 Grafisk mod

Med ABC 80 har Du också möjlighet att rita figurer, grafiska bilder m m på skärmen. Varje teckenposition ("bokstavsposition") kan då tolkas som 6 grafiska punkter.

Det finns två specialtecken som styr den grafiska moden:

CHR □ (151) som betyder "START GRAFIK" och

CHR □ (135) som betyder "SLUT GRAFIK"

Dessa tecken verkar på en rad, dvs efter en **CHR □ (151)** tolkas alla tecken grafiskt på den raden, tills en eventuell **CHR □ (135)** gör att man återgår till vanlig text igen.

Exempel

PRINT CHR □ (151); "abc"; "ABC"; CHR □ (135); "abc", "ABC"
ger på skärmen:

 **ABC abc ABC**

Observera att stora bokstäver ej förändras i grafisk mod. Du kan alltså fritt blanda grafiska tecken och stora bokstäver.

När Du skriver text motsvarande grafiska symboler har skärmen 24 rader à 40 tecken.

Du kan skriva i vilken teckenposition som helst på skärmen med funktionen **CUR(R, K)**, där R är radnummer och räknas från 0 till 23 uppifrån och ner och K är kolumnnummer (pos på raden) och räknas 0 till 39 från vänster till höger.

10 PRINT CUR (1, 3); "=n"

Om man sätter hela skärmen i grafisk mod (se programexempel) får man 72 grafiska rader, numrerade 0–71, med 78 grafiska positioner på varje rad, numrerade 2–79. (De två första positionerna upptas av START GRAFIK-tecknet).

SETDOT R, K	tänder en grafisk position
CLRDOT R, K	släcker en grafisk position
DOT(R, K)	anger om punkten R, K är tänd eller släckt.

Exempel

10 SETDOT 0,2 tänder en punkt upp i vänster hörn på skärmen.

20 IF DOT (R,K) THEN CLRDOT R,K ELSE SETDOT R,K

Denna programrad släcker punkten R,K om den är tänd och tänder den om den är släckt.

Programexempel med grafisk mod

```
10 FOR I = 1 TO 24 : : : CHR$(151); : NEXT I
20 FOR J = 5 TO 57 STEP 4
30 R = 41 - J/2
40 K = 36 - J/2
50 FOR I = 1 TO J
60 SETDOT R + I, K
70 SETDOT R + J, K + I
80 SETDOT R + J - I, K + J
90 SETDOT R, K + J - I
100 NEXT I
110 NEXT J
```

Programmet börjar med att sätta hela skärmen i grafisk mod. Detta görs genom att skriva ut **CHR\$(151)** först på alla 24 raderna på skärmen. Detta tecken är osynligt på skärmen men finns där och får ej skrivas över med något annat tecken.

































































Använd alltså bara de grafiska positionerna 2 till 79 på varje rad.

Här har Du också ett utmärkt tillfälle att se nyttan med %-variablerna (heltal). När Du har kört programmet några gånger kan Du använda ED-kommandot för att lägga in % efter alla variabelnamn i programmet, I%, R%, J%, K%. %-variablerna ökar snabbheten avsevärt! ED-kommandot kan användas såsom visas i avsnitt 2.18 "Rättelser av fel i programmet".

På de två följande sidorna finner Du en VIDEOGRAFIK-karta och en tabell med ASCII-tecknens grafiska motsvarigheter. På ett separat kartblad ritas Du lämpligen upp din bild. Du kan sedan slå upp de olika tecknen i tabellen och skriva ut en rad itaget. Videografik-kartan använder Du då för att få rätt proportioner på bilden.

TKN	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= plats för CHR α (151) = "START GRAFIK"

ASCIIKod	T	G	ASCIIKod	T	G	ASCIIKod	T	G	ASCIIKod	T	G
32	Blank		56	8		80	P	P	104	h	
33	!		57	9		81	Q	Q	105	i	
34	"		58	:		82	R	R	106	j	
35	#		59	;		83	S	S	107	k	
36	¤		60	<		84	T	T	108	l	
37	%		61	=		85	U	U	109	m	
38	&		62	>		86	V	V	110	n	
39	'		63	?		87	W	W	111	o	
40	(	64	È	É	88	X	X	112	p	
41)		65	A	A	89	Y	Y	113	q	
42	*		66	B	B	90	Z	Z	114	r	
43	+		67	C	C	91	Ä	Ä	115	s	
44	,		68	D	D	92	Ö	Ö	116	t	
45	-		69	E	E	93	Å	Å	117	u	
46	.		70	F	F	94	Ü	Ü	118	v	
47	/		71	G	G	95	-	-	119	w	
48	0		72	H	H	96	é		120	x	
49	1		73	I	I	97	a		121	y	
50	2		74	J	J	98	b		122	z	
51	3		75	K	K	99	c		123	ä	
52	4		76	L	L	100	d		124	ö	
53	5		77	M	M	101	e		125	å	
54	6		78	N	N	102	f		126	ü	
55	7		79	O	O	103	g		127		

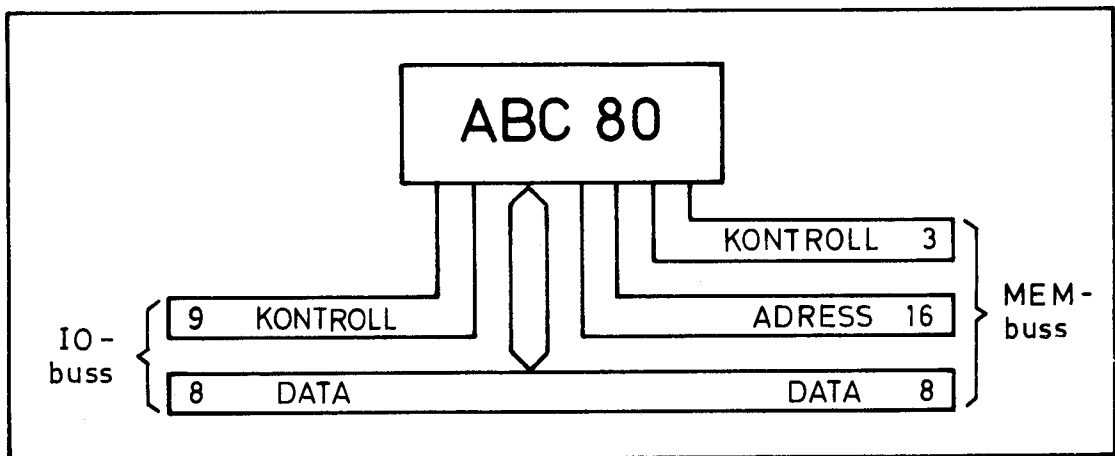
Koder tolkade i teckenmod (T) och grafmod (G)

2.21 ABC 80-bussen

ABC 80 har flera anslutningsmöjligheter för yttre enheter. Den viktigaste är den 64-poliga kontakten på tangentbordets baksida, den så kallade ABC 80-bussen. Här kan Du ansluta minnesexpansionskort eller specialinterface av flera olika slag. Du har möjlighet att välja mellan ett 50-tal olika kort. ABC 80 har på detta sätt stora utbyggnadsmöjligheter.

En utförlig beskrivning av ABC 80-bussen finner Du i boken "Mikrodatorns ABC" av G Markesjö.

ABC-bussen består av 16 adress-, 8 data- och 3 kontrollerledningar för kommunikation med yttre minneskort. För I/O-kort står förutom de 8 dataledningarna även 9 kontrollerledningar till förfogande.



ABC-bussens princip

ABC-bus Pin nr	Portadress	Funktion och beteckning	Basic-kommando
A22	Utg 0	$\overline{\text{OUT}}$ Data out	OUT 0%, X%
A23	Utg 1	$\overline{\text{CS}}$ Card select	OUT 1%, X%
A21	Utg 2	$\overline{\text{C1}}$ Command 1	OUT 2%, X%
A20	Utg 3	$\overline{\text{C2}}$ Command 2	OUT 3%, X%
A19	Utg 4	$\overline{\text{C3}}$ Command 3	OUT 4%, X%
A18	Utg 5	$\overline{\text{C4}}$ Command 4	OUT 5%, X%
A17	Ing 0	$\overline{\text{INP}}$ Data in	INP (0%)
A16	Ing 1	$\overline{\text{STAT}}$ Status in	INP (1%)
A15	Ing 7	RST Reset för alla I/O	INP (7%)

Kontrollerledningar för ABC-bussens I/O-del

Se även appendix.

2.22 Intern parallell I/O-krets (PIO)

För anslutning av det inbyggda tangentbordet, kassett- och V24- (modem) interfacen finns i ABC 80 en inbyggd PIO-krets. Följande portadresser gäller:

Portadress	Port	Funktion
56	A	Tangentbord dataport
57	A	Tangentbord kontrollport
58	B	Modem- och kassettinterface dataport
59	B	Modem- och kassettinterface kontrollport

Funktionerna hos de enskilda bitarna i B-porter är:

Bit	I/O	Benämning	Funktion	Vikt (signifikans)
0	IB0	RXD	Received Data	1
1	IB1	CTS	Clear to Send	2
2	IB2	DCD	Data Character Detect	4
3	OB3	TXD	Transmitted Data	8
4	OB4	RTS	Request To Send	16
5	OB5		Motorrelä	32
6	OB6		Data ut (kassett)	64
7	IB7		Data in (kassett)	128

Modemkontaktens utseende:

(IB0 betyder INBIT0)

(OB3 betyder OUTBIT 3)

(DSR betyder Data Set Ready)

Modem-anslutningakontakt (V24-interface)

Vill Du ansluta ABC 80 till det kommande Datavision-(Teledata)-systemet eller utnyttja den som en terminal använder Du modem-interfacet.

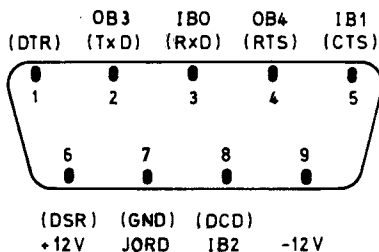
Du kan även via denna kontakt ansluta lysdioder, tryckknappar, reläer m m. Kontakten sitter på baksidan av tangentbordet, bredvid reset-knappen.

– Insignalnivåer: $\pm 2,5V$ till $\pm 30V$, där minusspänningen ger "1"-bit till PIO-ingången.

– Utsignalnivåer: $\pm 7V$, max 10mA.

Som ovan nämnts innehåller modemkontakten tre ingångar och två utgångar, vilka kan styras från port B i den inbyggda PIO:n. Ingångsbitarna 0, 1, 2 har vikterna 1, 2, 4, medan utgångsbitarna 3 och 4 har vikterna 8 resp 16.

V24 motsvarar det amerikanska RS 232-snittet.



Reläutgång

ABC 80 styr kassettbandspelarens drivmotor automatiskt vid text programladdning. Detta görs via en reläutgång som Du också kan använda till andra ändamål. Allt Du behöver är en extra sladd med $\varnothing 2,5$ teleplugg.

Reläet klarar att bryta upp till 100 mA.

Exempel

10 OUT 58,32 :REM Slå till reläet (PIO bit 5)

.

.

.

40 OUT 58,0 :REM Slå av reläet

Bithantering

Inläsningskommandot **INP (58%)** läser in en byte (Bit 0 ... Bit 7) från port B på PIO-kretsen i ABC 80. I denna byte ingår nu inte bara inbitarna utan också utbitarna.

Ex: **A% = INP (58%)** ger för A% värdet 12, om bitarna 2 och 3 i port B är "1". (Vikt 4+8)

Utmatningskommandot **OUT 58%, 24%** matar ut värdet 24 till PIO-kretsens port B.

Ex: **100 OUT 58%, 8%** sätter bit 3, övriga bitar nollställs.

Önskar Du sätta eller nollställa en utbit utan att påverka övriga utbitar läser Du först in tillståndet hos utbitarna med **INP (58%)** och använder sedan i **OUT**-kommandot **OR**- eller **AND**-operatorn för att maska bort övriga bitar.

Ex: **110 OUT 58%, 8% OR INP (58%)** sätter bit 3, port B utan att påverka övriga bitar.

120 OUT 58%, 247% AND INP (58%) nollställer bit 3, port B utan att påverka övriga bitar.

(Anm.: $247 = 128 + 64 + 32 + 16 + 0 + 4 + 2 + 1$)

150 IF NOT (1% AND INP (58%)) THEN 150

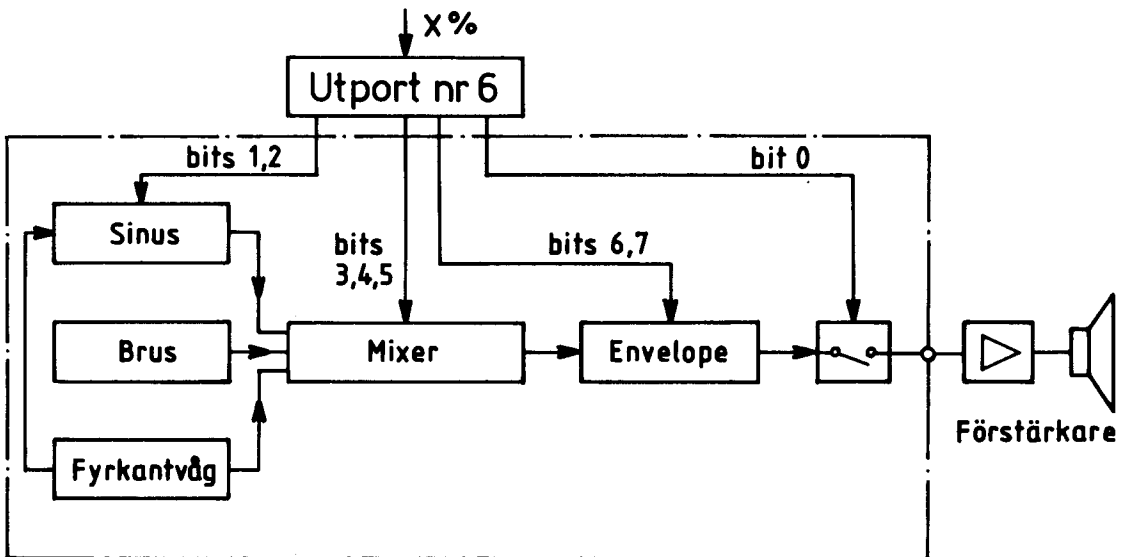
Programmet stannar på rad 150 tills PIO:s inbit 0 ändrad tillstånd från "0" till "1".

2.23 Ljudgeneratorn

ABC 80 är försedd med en speciell ljudgeneratorkrets, som kan styras från Ditt BASIC-program. BASIC-kommandot **OUT 6,X**, där X är ett heltal mellan 0 och 255, lagrar talet X i binär form i ett 8 bitars minne.

Minnet styr ljudgeneratorkretsen, vilken genererar tre oberoende signalspänningar som med hjälp av en blandare och en vågforms-(ENVELOPE)-generator sedan kombineras till olika ljud-effekter. Ljudet kommer ut i datorns högtalare.

Ljudgenerators princip



För att erhålla den önskade ljudeffekten bestämmer Du med hjälp av tabellen nedan vilka bitar som skall sättas "1". Lägg sedan ihop deras "vikter" till ett decimalt tal. Detta tal motsvarar det ovan nämnda talet X.

Vikt	128	64	32	16	8	4	2	1
Bit	7	6	5	4	3	2	1	0
	Vågform		Blandare			Sinuston		Till
00	ingen		000	Sinus		00	hög ton	0 Från
	inverkan		001	Brus				
01	ingen		010	Puls		01	låg ton	
	inverkan		011	Brus+Sinus				
10	snabb av-		100	Puls+Brus		10	pulston-	1 Till
	klingning		101	Puls +Sinus			styrt	
11	sinusöver-		110	Puls+Brus+Sinus		11	pulston-	
	lagring		111	(Tyst)			styrt	

Tabell för beräkning av det decimala talvärdet X

Ex: Du vill ha

VÄGFORM: opåverkad

BLANDARE: ren ton + brus

REN TON: låg frekvens

Motsvarande bitmönster blir: 00 011 011 = $16 + 8 + 2 + 1$

BASIC-kommandot lyder: **10 OUT 6, 27**

När Du gett brusgeneratorn ett kommando, fortsätter denna att ljuda tills Du skickat ett kommando med bit 0 = "0", dvs nollställd.

Ex: 5 sekunders alarmsirén

10 OUT 6,7 : REM Slå på

20 FOR I=1 TO 5000: NEXT I : REM Dröj 5 sek

30 OUT 6, 0 : REM Slå av

Ex: Ett "skott"

10 OUT 6,137 : REM Brus med kort ljudpuls som envelope

3. Användarexempel

3.1 Två programexempel

Användarens kommunikation med datorsystemet är huvudsakligen enkelriktad och styrd av programmet. Din uppgift är bland annat att möjliggöra en så fullständig dubbelriktad kommunikation som möjligt mellan ABC 80 och användaren.

Sätt dig nu i användarens situation och mata in följande lilla "användarprogram" i din ABC 80 och kör sedan programmet. Försök samtidigt observera vilka frågor och reflektioner som dyker upp inom dig under programexekveringen!

```
10 REM " * * * * * "
20 REM " * PROGRAM-EXEMPEL 1. * "
30 REM " * * * * * "
40 REM
50 PRINT CHR$(12)
60 PRINT : PRINT
70 INPUT A : INPUT G
80 R=(A+(G * G))/(2 * G) : ; R : R3=R2 : R2=R1 : R1=R
90 IF R=R3 GOTO 60
100 G=R : GOTO 80
```

Förmodligen hade du blivit positivt inställd till programmet om det istället hade haft följande utseende:

```
10 PRINT CHR$(12)
20 PRINT " * * * * * "
30 PRINT " * PROGRAM-EXEMPEL 2. * "
40 PRINT " * * * * * "
50 PRINT
60 PRINT "PROGRAMMET BERÄKNAR KVADRAT-"
70 PRINT "ROTEN UR ETT TAL MED HJÄLP AV"
80 PRINT "SUCCESSIVA APPROXIMATIONER."
90 PRINT : PRINT
100 PRINT " _____ "
110 ONERROR GOTO 340
120 PRINT
130 PRINT "UR VILKET TAL SKALL"
140 PRINT "KVADRATROTEN BERÄKNAS .....";
150 INPUT A : PRINT
160 IF A=0 OR A<1 GOTO 340
170 PRINT "GÖR EN GROV GISSNING"
180 PRINT "AV ROTENS VÄRDE .....";
190 INPUT G : PRINT
200 IF G=0 THEN G=1
210 IF G<0 THEN G=G * (-1)
220 PRINT : PRINT "BERÄKNINGSRESULTAT:"
230 PRINT
240 R=(A+(G * G))/(2 * G) : ; R
250 R3=R2 : R2=R1 : R1=R
260 IF R=R3 THEN PRINT : GOTO 280
270 G=R : GOTO 240
```

```

280 PRINT "
290 PRINT : PRINT "NY BERÄKNING?";
300 GET S
310 IF S="J" OR S="j" THEN 340
320 IF S=CHR(13) THEN 340
330 PRINT "NEJ!": GOTO 350
340 PRINT CHR(12): GOTO 100
350 PRINT : PRINT "Körningen klar!"
360 PRINT "    STOP!": PRINT
370 END

```

Med detta exempel har vi illustrerat att "självklarheter" för programmeraren/systemeraren inte är lika självklara för en användare eller för den som vill göra ändringar i givet program.

3.2 CHAIN – för mycket stora program

För att kunna köra mycket stora program, större än vad som får plats i minnet, finns det i ABC 80 möjlighet att dela upp programmet i mindre delar.

Varje del laddar då själv med hjälp av CHAIN in nästa programdel och fortsätter körningen.

Första programdelen med alla dess variabler, försvinner helt.

P O K E Om man behöver ha kvar vissa variabelvärden, kan dessa läggas undan högst upp i minnet i en speciell POKE-area. (Se minneskartan). Detta måste göras efter stängning av alla filer (CLOSE). Gäller speciellt printern.

P E E K Nästa programdel kan då hämta tillbaka dessa med PEEK-funktionen. Detta måste göras innan ny fil öppnas (OPEN). Gäller speciellt printern.

Använd t ex följande färdiga subrutiner:

Sätt A% = startadress på lagringsutrymmet

t ex A% = 65408%

```

100 REM * Spara sträng *
110 FOR J%=0% TO LEN(I)-1%
120 POKE A%+J%,ASC(MID I(J%,J%+1%,1%))
130 NEXT J%
140 POKE A%+J%,255%
150 A%=A%+J%+1% : RETURN
160 REM
200 REM * Återställ sträng *
210 I=""
220 IF PEEK(A%)=255% THEN A%=A%+1% : RETURN
230 I=I+CHR(PEEK(A%))
240 A%=A%+1% : GOTO 220
300 REM * Lagra flyttal *
310 I=NUM(I) : GOSUB 100 : RETURN
320 REM
400 REM * Återställ flyttal *
410 GOSUB 200 : I=VAL(I) : RETURN
420 REM
500 REM * Lagra heltal *

```



```

510 POKE A %,SWAP %(I), I%
520 A%=A%+2% : RETURN
530 REM
600 REM * Återställ heltal *
610 I%=SWAP%(PEEK(A%))+PEEK(A%+1%)
620 A%=A%+2% : RETURN

```

Exempel på användning

Du har skrivit ett gigantiskt program BUDGET, som är för stort för att få plats i ABC 80:s interna programminne på en gång.

Du delar då upp programmet i två delar BUDGET1 och BUDGET2, som kan köras efter varandra.

BUDGET2 behöver vissa data från BUDGET1, närmare bestämt variablerna A1,B1,Ö% och fältet S(10).

Vi skriver då:

```

10 REM * BUDGET1 *
.
.
.
9870 REM Slut BUDGET1, Spara undan A1, B1, Ö%, S(10)
10000 A% = 65408%: REM Början av POKE-area
10010 I = A1 :GOSUB 30000 :REM Spara I (=A1)
10020 I = B1 :GOSUB 30000 :REM Spara I (=B1)
10030 I% = Ö% :GOSUB 50000 :REM Spara I% (=Ö%)
10040 FOR J = 1 TO 10
10050 I = (S(J) : GOSUB 30000 :REM Spara ett element i taget
10060 NEXT J
10070 CHAIN "BUDGET2" :REM Nu laddas och körs BUDGET2
.
.
.
30000 REM * Lagra flyttal *
50000 REM * Lagra heltal *

10 REM * BUDGET2 *
20 REM Hämta sparade variabler
30 A% = 65408% :REM Adress till första variabeln
40 GOSUB 40000 :A1=I :REM Hämta A1
45 GOSUB 40000 :B1=I : REM Hämta B1
50 GOSUB 60000 :Ö% = I% :REM Hämta Ö%
60 FOR J = 1 TO 10
70 GOSUB 40000 :S(J) = I : REM Hämta ett element
80 NEX J
90 REM Nu börjar del 2 av BUDGET
.
.
.
40000 REM * Återställ flyttal *
.
.
.
60000 REM * Återställ heltal *

```

Kom ihåg att ladda A% med rätt startadress i POKE-arean, och att vid återställandet ta variablerna i samma ordning som vid undansparandet!

3.3 Inläggning av maskinspråksprogram

I vissa sammanhang har man glädje av maskinspråksprogram i ABC 80.

P O K E Med hjälp av POKE-kommandot har Du möjlighet att lägga in sådana maskinspråksprogram direkt i minnet.

C A L L Dessa kan sedan anropas med CALL. OBS att CALL endast används i tilldelningssatser (se sida 18).

Exempel

A% = CALL(65408)

Värdet av CALL-anropet är ett heltal som hämtas från Z80-processorns HL-reg vid återhoppet. Maskinspråksrutinen kommer tillbaka till BASIC med ett enkelt RET (maskinspråksreturn med kod C9 hexadecimalt eller 201 decimalt).

För att kunna läsa in programmet på hexadecimal form kan Du använda t ex följande BASIC-sekvens:

```
1 REM * HEXPOKE *
10 S$ = "0123456789ABCDEF"
20 FOR I% = 65408% TO 65535%
30 INPUT H$: REM TVÅ HEXSIFFROR IN
40 D% = INSTR (1%, S$, LEFT $(H$, 1%))
50 IF D%=0% THEN PRINT "Slutadress:"; I% + 1% : STOP
60 POKE I%, (D% - 1%) * 16% + INSTR (1%, S$, RIGHT $(H$, 2%)) - 1%
70 NEXT I%
```

Programmet begär 2 hex-siffror i taget och avslutas med icke hex-siffra, varvid maskinspråksprogrammets slutadress skrivs ut.

Lämplig plats att lägga maskinspråksprogram i är POKE-arean högst upp i minnet. Detta gäller dock ej om Du har skrivare, då POKE-arean används som utskriftsbuffer.

Om Du inte använder filhantering på kassett, kan Du dessutom använda CASBUF 1 och 2 (adress 64256 till 64768 decimalt).

Mera utrymme för maskinspråksprogrammen. För att få in ännu större maskinspråksprogram kan man låna minnesutrymme från BASIC:en. I ett system utan flexskiveminne läggs Dina program in från 49152 decimalt till 64256 decimalt.

Dessa två adresser ("golvet" som utpekats av BOFA-pekaren på adress 65053 och "taket" som utpekats av EOFA-pekaren på adress 65064) kan ändras med hjälp av POKE-kommandot.

POKE 65053, 192 + 16 : NEW

höjer "golvet" 4 kilobytes = 16 st 256 byte-block

Maskinspråksprogram kan då läggas från $192 * 256 = 49152$ till $(192 + 16) * 256 - 1 = 53247$

POKE 65064, 251 - 8

sänker "taket" 2 kilobytes = 8 st 256 byte-block

Detta ger utrymme från $(251 - 8) * 256 + 1 = 62209$ till $251 * 256 = 64256$

OBSERVERA att POKE i båda fallen måste göras som **kommando**.

Testexempel:

Om Du vill göra ett enkelt test, kan Du lägga in och köra följande enkla maskinspråksrutin som bara returnerar värdet 7.

Decimal maskinkod	Program
33,7,0	LD HL,7
201	RET

10 POKE 65408, 33, 7, 0, 201 :REM Lägg ut maskinkoden

20 PRINT 2 * (CALL(65408) - 1%)

Ger utskrift 12

Rad 20 vill visa att CALL motsvarar ett funktionsanrop och alltså kan användas i uttryck.

Man kan också skicka med ett heltalsvärde till maskinspråksrutinen vid anrop:

30 B% = CALL(65408, D%)

Variabeln D% i exemplet ovan läggs då i Z80-processorns DE-register vid anropet.

För att lägga in större program bör Du alltid använda programmet HEXPOKE ovan.

3.4 ABC 80:s Realtidsklocka

13.45

I ABC 80 finns en inbyggd "kvartsklocka" som går hela tiden när strömmen är påslagen. Du kan när som helst läsa av tiden från Ditt program, t ex för tidtagning eller start av yttre enheter.

Du använder nedanstående subprogram när Du vill plocka fram klockan på skärmen.

```
5 REM *** Realtidsklocka
10 T1%=65008%
100 PRINT "Vill du ställa klockan (j/n)"; : GET A % : A %
110 IF A %="J" OR A %="j" THEN GOSUB 800
120 PRINT CHR % (12)
200 GOSUB 2000
210 PRINT CUR(10,12);"Klockan är nu:"
220 PRINT CUR (11,15) RIGHT % (NUM % (100+H%),3);";";RIGHT
    % (NUM % (100+M%),3);
230 PRINT ":",RIGHT % (NUM % (100+S%),3)
240 GOTO 200
800 PRINT "hh,mm,ss:"; : INPUT H%,M%,S%
999 REM

1000 REM * * * DENNA RUTIN SÄTTER TIDEN
1010 REM H%(tim), M%(min), S%(sek)
1020 Z=H% * 3600 + M% * 60 + S%
1030 Z1%=Z * 50/256
1040 Z%=NOT(50 * (Z-Z1%/50 * 256))
1050 Z1%=NOT Z1%
1060 POKE 65008%,Z%,Z1%,SWAP%(Z1%)
1070 RETURN
1999 REM
```

```

2000 REM * * * DENNA ROUTIN LÄSER TIDEN
2010 REM H%(tim), M%(min), S%(sek)
2020 D%=0
2030 IF (PEEK(T1%) AND 4%)=0 THEN 2020
2040 FOR I%=0% TO 2%
2050 Z%(I%)=255% XOR PEEK(T1%+I%)
2060 NEXT I%
2070 Z=((Z%(2) * 256)+Z%(1)) * 5.12+Z%(0)/50
2080 IF Z>86400 THEN Z=Z-86400 : D%=D%+1: GOTO 2080
2090 H%=Z/3600 : Z=Z-3600 * H%
2100 M%=Z/60 : S%=Z-60% * M%
2110 IF D%<> 0 THEN GOSUB 1000
2120 RETURN

```

Tiden lagras som ett 24-bitars binärt heltal på adress 65008%, 65009% och 65010% med nedräkning från en kristallstyrd oscillator var 20:e millisekund.

3.5 Återanvändning av data

Ibland behöver man spara vissa data för att senare återanvända dem. Några tänkbara situationer som kan uppkomma. Antag att Du sysslar med statistik och har några hundra uppgifter ("rådata") som Du vill behandla med flera sinsemellan olika metoder. Tanken med att använda kassetten för datalagring är att Du bara behöver mata in Dina data en enda gång, trots att Du vill använda dem många gånger.

Andra användningsområden är t ex enklare bokföring, där Du lagrar dagboken på kassett-band för att vid ett senare tillfälle "Läsa av" och bearbeta bandet, t ex saldera kontona och skriva ut rapporter etc.

Även inom ordbehandling förekommer lagring och läsning av data. Här kan Du på en kassett lagra t ex brev med standardtext, kundregister etc överhuvudtaget textmaterial som skall återanvändas. Med en skrivare till ABC 80 kan Du skriva ut en standardoffert om och om igen ...

För professionell datalagring eller där stora datamängder används räcker inte kassettnminnet till. Här bör istället det betydligt snabbare flexskivminnet användas (se tillbehör sida 6). Det gäller speciellt bokföring och ordbehandling.

I följande program använder Du två kassetter, en för data och en för inmatnings- och avläsningsrutinerna. Inmatningsrutinen sköter om att de data Du slår in på ABC 80:s tangentbord verkligen överförs till ABC 80:s kassettninne.

När Du vill bearbeta Dina data igen använder du avläsningsrutinen till att överföra data från kassetten till ABC 80 för lämplig behandling.

Arbetsgången blir följande:

- Ladda in "inmatningsrutinen" från programkassetten
- Byt till datakassett
- Starta och slå in Dina data som skall lagras (men bearbetas vid senare tillfälle).

Här är ett exempel på en inmatningsrutin:

```
10 FOR I = 1 TO 10 :REM TA EMOT 10 VAROR MED NUMMER
20 PRINT "ANGE VARA OCH NUMMER";
30 INPUT V$(I), N(I) :REM LÄS IN VARA OCH NUMMER
35 IF N(I) = 999999 THEN 50 :REM AVSLUTA INMATNINGEN
40 NEXT I :REM NÄSTA VARA OCH NUMMER
50 PREPARE "LAGER.TXT" AS FILE 1: REM BANDSPELAREN OCH
   SKRIVER FILNAMN
60 FOR I = 1 TO 10
70 PRINT #1, V$(I);";";N(I);";" :REM SKRIV DATA PÅ KASSETTEN
80 NEXT I
90 CLOSE 1 : REM SKRIV "FILSLUT" PÅ KASSETTEN OCH STOP-
   PA BANDSPELAREN,
```

Om Du inte på förhand vet hur många datapar Du tänker mata in, är det lämpligt att avsluta datainmatningen med något ologiskt värde, t ex varunummer 999999. Detta kan då vid den senare avläsningen indikera att data är slut.

När Du vill använda Dina lagrade data, gör då så här:

- SÄTT I PROGRAMKASSETTEN
- LADDA IN AVLÄSNINGSRUTINEN
- BYT TILL DATAKASSETTEN
- STARTA AVLÄSNINGSRUTINEN OCH DINA DATA PÅ KASSETTEN
LÄSES OCH BEARBETAS AUTOMATISKT AV ABC 80.

Avläsningsrutinen ser ut så här:

```
10 OPEN "LAGER.TXT" AS FILE 1: REM STARTA BANDSPELAREN
   OCH SÖK EFTER FILEN
20 FOR I = 1 TO 10 :REM FÖR ATT LÄSA 10 VÄRDEN
30 INPUT #1, V$(I), N(I): REM LÄS DATA FRÅN KASSETTEN TILL
   INTERNMINNET
35 IF N(I) = 999999 THEN 50 : REM AVSLUTA AVLÄSNINGEN
40 NEXT I
50 CLOSE 1: REM STANNA BANDSPELAREN.
```

I stället för att ha ett "ologiskt värde", t ex 999999 som slutmarkering, kan Du utnyttja ABC 80:s felhantering.

Om Du försöker läsa förbi slutet på filen får Du ERROR 34 "SLUT PÅ FILEN", som normalt avbryter programkörningen helt.

Med hjälp av satsen

```
15 ON ERROR GOTO 50
```

tar Du istället hand om felet själv och hoppar till rad 50 som stannar kassetminnet.

Funktionen ERRCODE innehåller felnumret.

4. Kommandon, satser, funktioner, operatorer

4.1 Kommandon

Kommando	Kort beskrivning	Exempel
CLEAR	Nollställer alla variabler och stänger alla filer.	CLEAR
CTRL C	Avbryter programkörning	
ED ...	Ger möjlighet till ändring i en programrad utan att den behöver skrivas om.	ED 40
KILL "..."	Raderar angiven fil från flexskivan. Filtyp måste anges	KILL "PROGRAM.BAC" KILL "DR0: PROGRAM.BAC"
LIST ...	Ger utskrift på bildskärmen av det program som finns i minnet. Följande varianter finns: Samtliga rader. Endast en rad. Allt mellan två radnummer. Allt t o m viss rad. Allt fr o m viss rad.	LIST LIST 100 LIST 100 -200 LIST -200 LIST 100 -
LIST ...	Överför program som finns i ABC 80 till en fil på kassett eller flexskiva. Utlämnad filtyp sätts till .BAS	LIST PROGRAM LIST DR1: PROGRAM LIST CAS: PROGRAM
LIST PR:	Ger utskrift på skrivare av det program som finns i ABC 80. PR: motsvaras i viss programvara av V24:	
LOAD ...	Överför program med angivet namn från kassett eller flexskiva till ABC 80. Raderar befintligt program i ABC 80. LOAD CAS: (Överför det första program som påträffas på kassetten till ABC 80)	LOAD PROGRAM LOAD DR1: PROGRAM LOAD CAS: PROGRAM
MERGE ...	Hämtar angivet program från kassett eller flexskiva utan att radera det program som finns i minnet. (Radering i det befintliga programmet sker endast om samma radnummer används i det överförda programmet).	MERGE PROGRAM3 MERGE DR0: PROGRAM
MERGE CAS:	Hämtar närmaste program från kassett utan att radera det program som finns i minnet. Om samma radnummer både finns på kassett och i minnet kommer raden från kassetten att användas.	MERGE CAS:

Kommando NAME "... "AS" ..."	Kort beskrivning Byter namn på en fil. Filtyp måste anges.	Exempel NAME "DR1:GAMMAL BAC" AS "NY BAC"
NEW	Raderar programmet i minnet och stänger alla filer.	NEW
NOTRACE	Upphäver verkan av TRACE. Jfr TRACE.	NOTRACE
POKE ..., ...,	Överför data i decimal form, byte för byte, till angiven minnesadress och därefter följande adresser:	POKE 65408, 201 POKE 65408, 61, 3, 75, 200
PRINT ...	Skriver på bildskärmen. PRINT X,Y,Z (jämför instruktionen STOP)	PRINT 2/3
REN ...	Numrerar om alla raderna i programmet och ändrar satser med GOTO, GOSUB o s v så att de syftar på rätt rad. Följande varianter finns: Numrerar raderna med början på 10 och med intervall 10. Numrerar raderna med början på ett visst radnummer, t ex 25, och med intervall 25. Numrerar raderna med början på visst radnummer, t ex 500, och med ett önskat intervall, t ex 20.	REN REN 25 REN 500, 20
RUN	Kör det program, som finns i minnet efter att ha nollställt alla variabler.	RUN
RUN ...	Motsvarar kommandot LOAD ... följt av kommandot RUN. Jämför LOAD. ...	RUN PROG1 RUNCAS: PROG1 RUN DR 1: PROG 1
RUN CAS:	Hämtar närmaste program från kassett och kör programmet efter att ha raderat ev gammalt program i ABC 80.	RUN CAS:
SAVE ...	Överför program som finns i ABC 80 till en fil på kassett eller flexskiva i kompilerad form. Utelämnad filtyp till .BAC	SAVE PROG1 SAVE DR0: PROG1 SAVE CAS: PROG1
SCR	Raderar programmet i minnet och stänger alla filer. Har samma effekt som NEW.	SCR
TRACE	Radnummer för utförda programrader skrivs ut under körning.	TRACE
UNSAVE ...	Raderar angiven fil på flexskiva. Utelämnad filtyp ersätts med .BAC alternativt .BAS	UNSAVE PROGRAM UNSAVE DR1: PROGRAM. TXT

4.2 Instruktioner

Instruktion

CHAIN "..."

Kort beskrivning

Hämtar angivet program från kassett eller flexskiva. Har samma effekt som RUN ...

Exempel

```
100 CHAIN "PROG2"
110 CHAIN "DR1: PROG3"
120 CHAIN A □
CHAIN "LIB"
```

CLRDOT R,K

Släcker en grafisk punkt på skärmen. Jfr SETDOT R,K.

40 CLRDOT 30,40

CLOSE .

Stänger angiven fil, dvs avslutar läsning eller skrivning mot filen. Om filen öppnats med PREPARE skrivs ett filslutmärke. Är angiven fil lagrad på kassett stoppas bandspelarmotorn.

10 CLOSE 2
CLOSE 1

DATA

Lagrar variabelvärden som ska läsas av en READ-sats.

20 DATA 5.31, 4, HEJ

Om strängar innehåller mellanslag eller kommatecken måste de omges av citatstecken.

30 DATA "HEJ DÄR"

DEF FN ...(...)= ...

Definierar egen funktion.

10 DEF FNA(X,Y) = X + X * Y

DIM ...

Reserverar utrymme för fältvariabler

Exempel.

Reserverar utrymmet för 26 flyttalsvariabler A(0) t o m A(25).

10 DIM A(25)

Reserverar utrymme för en sträng A med max 200 tecken.

20 DIM A □ = 200

Reserverar utrymme för 31 strängvariabler A □ (0) t o m A □ (30) om vardera maximalt 20 tecken.

30 DIM A □ (30)=20

Reserverar utrymme för en matris med heltalsvariabler innehållande 11 rader och 21 kolumner.

40 DIM A% (10,20)

Reserverar utrymme för en strängmatris med 4 rader och 6 kolumner och där varje element omfattar maximalt 30 tecken.

50 DIM A □ (3,5)=30

END

Avslutar programkörning, stänger alla filer och nollställer alla variabler.

200 END

FOR ... TO ... STEP ...

Utför ett programavsnitt det antal gånger som bestäms av loppvariabelns start- och slutvärde samt stegvärdet.

10 FOR X = 10 TO 25 STEP 5

Instruktion	Kort beskrivning	Exempel
GET	Väntar på tangentnedtryckning. Lagrar mottaget tecken i en strängvariabel, utan att skriva på TV-skärmen. Alla tecken accepteras.	10 GET A ☐
GOSUB	Medför hopp till subrutin på angiven rad. När ett RETURN påträffas sker återhopp till satsen efter motsvarande GOSUB.	30 GOSUB 90
GOTO	Medför hopp till angiven rad.	20 GOTO 10
IF ... THEN ... ELSE ...	Om uttrycket efter IF är sant så utförs det som står mellan THEN och ELSE annars utförs det som står efter ELSE. Om det bara står ett radnummer efter THEN eller efter ELSE utförs hopp till detta radnummer. (ELSE ... kan utelämnas).	50 IF A = B THEN PRINT A ELSE GO TO 300 60 IF A = B THEN 90 ELSE 200
INPUT	Skriver ett frågetecken på bildskärmen och väntar därefter på inmatning av data via tangentbordet. För data inmatade via tangentbordet gäller samma regler som för data i DATA-satser.	20 INPUT A 30 INPUT A % 40 INPUT A ☐
INPUT # ...	Läser från en fil med angivet nummer. I övrigt se INPUT.	10 INPUT # 1, A
INPUTLINE	Läser in en rad från tangentbordet, precis som den står, komplett med mellanslag, kommatecken och citationstecken. Även tecknen CR (vagnretur) och LF (radframmatning) ingår i slutet på raden.	20 INPUTLINE A ☐
INPUTLINE # ...	Läser från en fil med angivet nummer. I övrigt se INPUTLINE.	30 INPUTLINE # 1, A ☐
KILL ""	Raderar angiven fil från flexskivan. Filtyp måste anges.	20 KILL "DR0: EX.BAC" 60 KILL C ☐+, "BAS"
LET	Tilldelar en variabel ett värde. Uttrycket kan även skrivas direkt utan LET.	30 LET A = 5 30 A = 5 40 LET A ☐ = B ☐ 40 A ☐ = B ☐
NAME " "AS""	Byter namn på en fil. Filtyp måste anges.	20 NAME "DR1: GAMMAL.BAC AS "NY.BAC"
NEXT	Avslutar ett FOR-avsnitt, d v s begränsar en loop.	50 NEXT X
NOTRACE	Upphäver verkan av TRACE. Jfr TRACE.	60 NOTRACE

Instruktion	Kort beskrivning	Exempel
ON ERROR GOTO	Ger hopp till visst radnummer om fel skulle uppstå under programkörningen. Endast de felkoder som i kap. 6 markeras med + ger uthopp. I exemplet medför ett felmeddelande hopp till rad 500.	10 ON ERROR GOTO 500
ON ... GOSUB ...	Hoppar till olika subrutiner, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 30. N = 2 hopp till rad 120 o s v.	10 ON N GOSUB 30, 120, 400
ON ... GOTO ...	Hoppar till olika radnummer i programmet, beroende på värdet av en variabel. I exemplet ger N = 1 hopp till rad 200, N = 2 hopp till rad 260 o s v.	10 ON N GOTO 200, 260, 310
ON ... RESTORE ...	Gör RESTORE till olika DATA-satser, beroende på värdet av en variabel. I exemplet hämtas för N = 1 data från och med rad 60, för N = 2 hämtas data från och med rad 70 o s v.	10 ON N RESTORE 60, 70, 80
OPEN ... ASFILE ...	Öppnar angiven fil för läsning och tilldelar filen filnummer (1-255)	20 OPEN "BREV" ASFILE 2
OUT 6, N	Aktiverar den inbyggda ljudgeneratoren.	30 OUT 6, N (1 ≤ N ≤ 255, udda tal) OUT 6, N
	Stänger ljudgeneratoren.	40 OUT 6, N (0 ≤ N ≤ 255, jämna tal) OUT 6, N
PREPARE ... ASFILE ...	Skapar och öppnar en angiven fil för skrivning och tilldelar filen ett filnummer (1-255).	10 PREPARE "BREV" ASFILE 2
PRINT	Skriver på TV-skärmen.	10 PRINT "HEJ", A 20 PRINT A, A%, G (utskrift i kolumner) 50 PRINT "TEXT"; A □ (utskrift utan mellanrum)
	Kan även innehålla beräkningsuttryck eller specialfunktionerna TAB och CUR. Förkortning för instruktionen PRINT.	30 PRINT TAB (5) (A - B) * 2 40 PRINT CUR (10,15) "TEXT" 10 ;"HEJ", A □
PRINT # ...	Skriver på en fil med angivet nummer.	10 PRINT # 1, "HEJ"
RANDOMIZE	Ger RND-funktionen ett slumpmässigt startvärde.	10 RANDOMIZE
READ	Läser variabelvärden från DATA-satser och tilldelar dem variabler.	10 READ A, A%, A □

Instruktion	Kort beskrivning	Exempel
REM	Kommentar kan skrivas i programmet.	30 REM * * START * *
RESTORE	Får nästa READ-sats att läsa data från och med en viss DATA-sats. Följande varianter finns: Läser data från och med den första DATA-satsen. Läser data från och med rad 200.	10 RESTORE 10 RESTORE 200
RETURN	Avslutar subrutin och medför återhopp till huvudprogrammet till instruktionen efter motsvarande GOSUB.	120 RETURN
SETDOT R,K	Tänder en grafisk punkt på skärmen. (Raden måste vara satt i grafisk mod.) $0 \leq R \leq 72$ $2 \leq K \leq 79$ Jfr CLRDOT R,K.	30 SET DOT 30, 40
STOP	Avslutar programkörningen, ger akustisk signal och skriver ut STOP-satsens radnummer. Variabler nollställs ej.	50 STOP
TRACE	Radnummer för utförda programrader skrivs ut under körning.	10 TRACE

4.3 Strängfunktioner

Strängfunktioner	Kort beskrivning	Exempel
ASC (A □)	Ger ASCII-värdet på första tecknet i A □	50 V = ASC (A □)
CHR □ (I)	Ger ett tecken med ASCII-värdet I. Kan innehålla upp till och med fyra ASCII-tecken.	10 PRINT CHR □ (7) 20 PRINT CHR □ (65, 66, 67)
INSTR (I,A □,B □)	Söker efter strängen B □ i strängen A □ med början i position I. Funktionen ger läget för första förekomsten av B □ i A □. Om B □ ej finns i A □ erhålls värdet 0.	60 P = INSTR (I, A □, B □)
LEFT □ (A □,J)	Ger de J första tecknen i strängen A □	10 C □ = LEFT □ (A □,J)
LEN (A □)	Ger aktuella längden av strängen A □	40 L = LEN (A □)
MID □ (A □,I,J)	Ger J tecken från och med teckenposition I i strängen A □	20 D □ = MID □ (A □,I,J)
NUM □ (A)	Ger en sträng med de tecken man skulle fått om man skrivit ut A med en PRINT-sats.	70 A □ = NUM □ (A)

Stränkfunktioner		Kort beskrivning	Exempel
RIGHT □ (A □,I)		Ger alla tecken i A □ från och med tecken I.	30 D □ = RIGHT □ (A □,I)
SPACE □ (I)		Ger en sträng av I st mellanslag (spacer).	50 PRINT SPACE □ (I)
STRING □ (I,C)		Ger en sträng av I st (CHR □ (C)).	60 PRINT STRING □ (I,C)
VAL (A □)		Ger värdet av A □ tolkat som ett tal. Sammanfogar två strängar.	80 A = VAL (A □) 10 A □ = B □ + C □
ASCII-funktion		Kort Beskrivning	Exempel
ADD □ (A □,B □,N)		Adderar strängarna A □ och B □ tolkade som tal. Resultatet avrundas till N decimaler. Strängarna får innehålla max 29 siffror, samt tecknen + och – samt decimalpunkt men inte exponent.	10 C □ = ADD □ (A □, B □, N.)
SUB □ (A □,B □,N)		Dito subtraktion. (A □ subtraherat med B □ och resultatet med N decimaler.)	10 C □ = SUB □ (A □,B □,N)
MUL □ (A □, B □, N)		Dito multiplikation. (A □ multiplicerat med B □ och resultat med N decimaler.)	10 C □ = MUL □ (A □,B □,N)
DIV □ (A □,B □,N)		Dito division. (A □ dividerat med B □ och resultatet med N decimaler.)	10 C □ = DIV □ (A □,B □,N)
COMP% (A □,B □)		Jämför A □ och B □ tolkade som tal. Ger följande resultat: –1 om A □ < B □ 0 om A □ = B □ 1 om A □ > B □	10 A = COMP% (A □,B □)

4.4 Matematisk funktion

Funktion	Kort beskrivning	Exempel
SIN (X)	sin x, x i radianer	10 Y = SIN (X)
COS (X)	cos x, x i radianer	20 Y = COS (5 * X)
TAN (X)	tan x, x i radianer	30 Y = TAN (X+2)
ATN (X)	arctan x	40 Y = ATN (.5)
LOG (X)	e-logaritmen av x	50 Y = LOG (756)
LOG 10 (X)	10-logaritmen av x	60 Y = LOG 10 (1000%)
EXP (X)	e ^x	70 Y = EXP (2)

Funktion	Kort beskrivning	Exempel
SQR (X)	Kvadratroten ur x, \sqrt{x}	80 Y = SQR (A * * 2 + B * * 2)
INT (X)	Största heltalet mindre än eller lika med x	90 Y = INT (Z)
FIX (X)	Heltalsdelen av x, [x]	100 Y = FIX (Z)
ABS (X)	Absolutbeloppet av x, [x]	110 Y = ABX (-6)
SGN (X)	-1 om $x < 0$ 0 om $x = 0$ 1 om $x > 0$	120 Y = SGN (X)
PI	π , d v s talet 3,14159	130 X = PI * R * * 2
RND	Ger slumpstal mellan 0 och 0,999999 inklusive dessa.	10 LET A = RND 20 LET B = 10 * RND

4.5 Specialfunktioner

Specialfunktioner

DOT (R,K)	Kort beskrivning Ger värdet SANT (-1) om den grafiska punkten R,K är tänd annars värdet FALSKT (0).	Exempel 30 IF DOT (R,K) THEN CLRDOT R,K ELSE SETDOT R,K
ERRCODE	Ger som resultat senaste felkoden.	10 A = ERRCODE
TAB (K)	Flyttar markören till position K på raden. Förekommer endast i PRINT-satser. $0 \leq K \leq 39$	10 PRINT TAB (10); " *"
CUR (R,K)	Flyttar markören till rad R och kolumn K på TV-skärmen. Förekommer endast i PRINT-satser. $0 \leq R \leq 23$ (RADER) $0 \leq K \leq 39$ (KOLUMNER)	10 PRINT CUR (11,12); "MITT PÅ SKÄRMEN"

4.6 Minnesåtkomst och IN/UT-portar

Specialtecken

CALL (A)	Kort beskrivning Anropar subrutin i maskinspråk på angiven adress (A). Efter utförd subrutin är CALL(A) lika med talet i HL-registret i mikroprocessorn Z80A.	Exempel 10 A=CALL (65408)
CALL (A,U)	Anropar subrutin i maskinspråk på angiven (A). Före subrutinanropet laddas mikroprocessorns DE-register med angivet uttryck (U). Efter utförd subrutin är CALL (A,U) lika med talet i HL-registret	20 8=CALL (65408, U%)

Specialtecken	Kort beskrivning	Exempel
INP (P)	Hämtar en byte från angiven port (P).	30 C% = INP (58)
OUT P1, D1, P2, D2 ...	Överför data D1 till port P1, osv.	40 OUT 58, 32
PEEK (A)	Hämtar en byte från angiven minnes-adress (A).	50 PRINT PEEK (65011)
POKE A, D1, D2 ...	Överför data D1, D2 ... till angivna minnesceller från och med angiven minnes-adress (A).	60 POKE 65008, 10, 5, 2
SWAP% (D)	Ger värdet av D tolkat som heltal (2 bytes), men med första och andra byten omkastade.	70 B% = SWAP % (D%)

4.7 Specialtecken

Specialtecken	Kort beskrivning	Exempel
CHR □ (7)	Ger pip i den inbyggda högtalaren.	10 PRINT CHR □ (7)
CHR □ (10)	Radframmatning	10 PRINT CHR □ (10)
CHR □ (12)	Tömmer bildskärmen och flyttar markören till övre vänstra hörnet.	10 PRINT CHR □ (12)
CHR □ (13)	Motsvaras tangentbordets retur tangent	10 PRINT CHR □ (13)
CHR □ (151)	"START GRAFIK", startar grafisk mod på en rad.	10 PRINT CHR □ (151); "1234"
CHR □ (135)	"SLUT GRAFIK" avslutar grafisk mod på raden.	10 PRINT CHR □ (151); "1234"; CHR □ (135); "1"

4.8 Uttryck

Ett uttryck består av ett antal tal och funktioner åtskilda av operatorer. Följande operatorer finns:

Prioritet	Operator	Betydelse	Exempel
1	** alt. Ü	exponentiering	B ** C%
2	*	multiplikation	A * B
2	/	division	A / C
3	+	addition	A + B
3	-	subtraktion	A - B

Prioriteten, som anger i vilken ordning operationerna utförs, är den gängse. Dvs först utförs exponentiering, därefter multiplikation och division och slutligen addition och subtraktion. Vid lika prioritet utförs operationerna från vänster till höger. Ordningsföljden ändras med parenteser.

Vid exponentiering av negativa tal måste exponenten vara av heltalstyp.

4.9 Relationsoperator

Operatorer

>	Kort beskrivning Större än. Minnesregel: stor öppning mot stort tal.	Exempel 10 IF A>B THEN GOTO 50
<	Mindre än.	20 IF A<B THEN 50
=	Lika med.	30 IF A = B THEN PRINT A
>=	Större än eller lika med.	40 IF A>=B THEN 70
<=	Mindre än eller lika med.	50 IF A<=B THEN 10
<>	Ej lika med. (Skilt från.)	60 IF A<>B THEN GOTO 10

Relationsoperatorer med prioritet 4 ger värdet SANT(–1) om uttrycket är sant, annars värdet FALSKT (0)

Relationsoperatorerna kan också användas mellan två stränguttryck. Då jämförs tecknens ASCII-representation teckenvis från vänster. Jämförelsen avslutas då två olika tecken påträffas. Strängen med den tidigare bokstaven (i alfabetet) anses då vara minst. Om någon sträng "tar slut" anses den vara den mindre av de två.

4.10 Logiska operatorer

NOT	ICKE. Är sant om operanden är falsk.	10 IF NOT A<B THEN 20
AND	OCH. Är sant om båda operanderna är sanna.	20 IF A>B and C=D THEN 30
OR	ELLER. Är sant om minst en av operanderna är sann.	30 IF A>B OR C=10 THEN 40
XOR	EXKLUSIVT ELLER. Är sant om endera av operanderna är sann men inte båda.	40 IF A=B XOR C=D THEN 50
IMP	IMPLICERAR. A IMP B är falskt endast om A är sann och B är falsk.	50 IF A IMP B THEN 60
EQV	EKVIVALENS. Är sant om båda operanderna är sanna eller om båda är falska.	60 IF A=B EQV C=D THEN 70
	De logiska operatorerna kan också användas på godtyckliga heltal. Operatorerna verkar då bit för bit på motsvarande binära tal.	70 A%=B% AND 15%

4.11 Kontrolltecken

CTRL **C**

Avbryter pågående programkörning eller listning

CTRL **X**

Backstegar och raderar hel rad vid inmatning

CTRL **H**

Backstegar och raderar ett tecken vid inmatning

CTRL **⌘**

Ger tecknen ■ (fylld ruta)

5. Referenskort

AB INPLASTOR MOTALA

80	LOAD ED	NEW MERGE	SCR SAVE RUN	SATSER: SE MANUAL REN
KOMMANDON: CLEAR		LIST		

ERR	FÖRKLARING	ERR	FÖRKLARING
0	Ej tillåtet öka "DIM"	•34	Slut på filen
1	Fel antal index	•35	Checksummafel vid läsning
2	Otillåtet som kommando	•36 *	Checksummafel vid skrivning
3	Minnet fullt	•37	Felaktigt recordformat
•4	För stort flyttal	•38 *	Recordnummer utanför filen
5	För stort index	•39 *	Filen skrivskyddad
6	Hittar ej detta radnummer	•40 *	Filen raderingsskyddad
•7	För stort heltal	•41 *	Skivan full
8	Finns ej i detta system	•42 *	Skivan ej klar
9	Index utanför strängen	•43 *	Skivan skrivskyddad
10	Texten får ej plats i strängen	44 *	Logisk fil ej öppen
11	Forstår ej	45 *	Fel logiskt filnummer
•12	Felaktigt tal	46 *	Fel enhetsnummer
13	Fel antal eller typ av argument	47 *	Fel trapnummer
14	Otillåtet tecken efter satsen	48 *	Fel i biblioteket
15	"=" saknas eller på fel plats	49 *	Felaktigt fysiskt filnummer
16	Radnummer saknas	50	Kvadrattrot ur negativt tal
17	Otill blandn. av tal och strängar	51	Enheten upptagen
18	"j" saknas eller på fel plats	52	Ej till denna enhet
•19	Kan ej öppna fler filer	53	Felaktig rad
20	För lång rad (>120 tkn)	54	• IEC både sändare och mott.
•21	Hittar ej filen	55	• IEC mottagare ej aktiv
22	Otillåten sats	56	• IEC sändare ej aktiv
23	"TO" saknas	57	Funktionen ej definierad
24	"NEXT" saknas	58	Ogiltigt tecken inläst
25	Felaktig sats efter "ON"	59	Fel programformat
26	Fel i "ON"-uttryck	60	Bit adress >16 bitar
27	"NEXT" utan "FOR"	61	Komma saknas
28	Fel variabel efter "NEXT"	62	DOT-adress utanför skärmen
29	"RETURN" utan "GOSUB"	63	"AS" saknas
•30	Data slut	64	Felaktig RENAME
31	Fel data till kommando	65	Spill i ASCII-aritmetik
32	Filen ej öppnad	66	Sträng ej numerisk
33	"AS FILE" saknas		

* ON ERROR GOTO

• avser fel på skiva

o avser option

Körning av program på kassett:

- | | |
|------------------------------------|--------------------------------|
| 1 Spola till önskat program | 3 Skriv RUN CAS: RETURN |
| 2 Tryck ned "PLAY" på bandspelaren | 4 Datorn laddas och kör direkt |

6. Felmeddelanden

- ERROR 0 EJ TILLÅTET ÖKA "DIM". Ett fält får inte ökas utöver sin ursprungliga längd.
- 1 FEL ANTAL INDEX. Antalet index överensstämmer ej med DIM.
- 2 OTILLÅTET SOM KOMMANDO. En instruktion ges direkt, som endast kan utföras som del av ett program.
- 3 MINNET FULLT. Program och data får ej plats.
- + 4 FÖR STORT FLYTTAL.
- 5 FÖR STORT INDEX. Försök att använda index större än motsvarande DIM.
- 6 HITTAR EJ DETTA RADNUMMER. Referens till ett radnummer som inte finns i programmet.
- + 7 FÖR STORT HELTAL. Försök att använda ett tal som ligger utanför datorns kapacitetsområde för heltal.
- 8 FINNS EJ I DETTA SYSTEM. Förekommer endast vid programutveckling.
- 9 INDEX UTANFÖR STRÄNGEN. Index för stort eller negativt.
- 10 TEXTEN FÅR EJ PLATS I STRÄNGEN. För liten Dimension på den mottagande strängen.
- 11 FÖRSTÅR EJ. Formellt BASIC-fel. Datorn kan inte känna igen ett kommando eller en instruktion.
- + 12 FELAKTIGT TAL. Talet innehåller tecken som inte är siffror.
- 13 FEL ANTAL ELLER TYP AV ARGUMENT.
- 14 OTILLÅTET TECKEN EFTER SATSEN. Formellt BASIC-fel. Datorn förväntade RETURN eller kolon (:).
- 15 "=" SAKNAS ELLER PÅ FEL PLATS. Formellt BASIC-fel vid LET och FOR-satser.
- 16 RADNUMMER SAKNAS. En rad utan radnummer påträffades.
- 17 OTILLÅTEN BLANDNING AV TAL OCH STRÄNGAR. Strängar och numeriska operationer får inte hopblandas.
- 18 ")" SAKNAS ELLER FEL PLATS. Formellt BASIC-fel.
- + 19 KAN EJ ÖPPNA FLER FILER. För många filer redan öppnade.
- 20 FÖR LÅNG RAD (>120 tkn). Längsta tillåtna program-eller datarad är 120 tecken.
- + 21 HITTAR EJ FILEN. Datorn kan inte hitta filen. Filen finns inte tillgänglig eller har sökts under fel namn.
- 22 OTILLÅTEN SATS.
- 23 "TO" SAKNAS. Förekommer i FOR-satser.
- 24 "NEXT" SAKNAS. En FOR-sats finns i programmet utan motsvarande NEXT-sats som ska avsluta looperna.
- 25 FELAKTIG SATS EFTER "ON". Formellt BASIC-fel.

- ERROR 26 FEL I ON-UTTRYCK.
- 27 "NEXT" UTAN "FOR". En NEXT-sats har skrivits in i programmet utan motsvarande FOR-sats.
- 28 FEL VARIABEL EFTER "NEXT". Formellt BASIC-fel.
- 29 "RETURN" UTAN "GOSUB". En RETURN-sats påträffad i programmet utan att en föregående GOSUB-sats har blivit utförd.
- + 30 DATA SLUT. Datalistan har blivit tömd och en READ-sats efterfrågade fler data.
- 31 FEL DATA TILL KOMMANDO. Felaktigt argument till kommandot t ex UNSAVE utan filnamn.
- 32 FILEN EJ ÖPPNAD. INPUT- eller PRINT-sats till en stängd kanal.
- 33 "AS FILE" SAKNAS. Förekommer i OPEN- och PREPARE-satser.
- + 34 SLUT PÅ FILEN. Försök att läsa efter filens slut.
- + 35 CHECKSUMMAFEL VID LÄSNING. Skivan eller kassetten skadad.
- + 36 ● CHECKSUMMAFEL VID SKRIVNING. Skivan skadad.
- + 37 FELAKTIGT RECORDFORMAT. Icke kompatibelt inspelningsformat. Kan också betyda fel på skiva eller kassett.
- + 38 RECORDNUMMER UTANFÖR FILEN. Försök att läsa längre än filen medger.
- + 39 ● FILEN SKRIVSKYDDAD. Filen är skyddad för utskrift.
- + 40 ● FILEN RADERINGSSKYDDAD. Filen är skyddad mot utplåning.
- + 41 ● SKIVAN FULL. Filen får ej plats på flexskivan.
- + 42 ● SKIVAN EJ KLAR: Ingen flexskiva inmatad i disken eller spaltluckan inte stängd.
- + 43 ● SKIVAN SKRIVSKYDDAD.
- 44 ● LOGISK FIL EJ ÖPPNAD.
- 45 ● FEL LOGISKT FILNUMMER.
- 46 ● FEL ENHETSNUMMER.
- 47 ● FEL ENHETSNUMMER.
- 48 ● FEL I BIBLIOTEKET.
- 49 ● FELAKTIGT FYSISKT FILNUMMER.
- 50 KVADRATROT UR NEGATIVT TAL. Logiskt fel i programmet.
- 51 ENHETEN UPPTAGEN.
- 52 EJ TILL DENNA ENHET. Exempelvis med INPUT från PR:
- 53 FELAKTIG RAD.
- 54 ○ IEC BÅDE SÄNDARE OCH MOTTAGARE. IEC-option.

- ERROR
- 55 o IEC-MOTTAGARE EJ AKTIV. IEC-option.
 - 56 o IEC-SÄNDARE EJ AKTIV. IEC-option.
 - 57 FUNKTIONEN EJ DEFINIERAD. En sökt funktion för vilken det inte finns någon DEF-sats.
 - 58 OGILTIGT TECKEN INLÄST.
 - 59 FEL PROGRAMFORMAT. Programmet är sparat under en icke-kompatibel BASIC-version.
 - 60 BIT ADRESS 16 BITAR.
 - 61 KOMMA SAKNAS.
 - 62 DOT-ADRESS UTANFÖR SKÄRMEN.
 - 63 "AS" SAKNAS. Fel i NAME AS.
 - 64 FELAKTIG "RENAME". Det nya namnet är ej ett korrekt filnamn.
 - 65 SPILL I ASCII-ARITMETIK.
 - 66 STRÄNG EJ NUMERISK.

ANM: ● avser fel på flexskiva, o avser option och + avser fel som kan hanteras med ON ERROR GOTO.

7. Övrigt

ABC 80:s minneskarta
Följande är en beskrivning av hur ABC 80 använder tillgänglig minnesarea. Här kan Du hitta utrymme att lägga in maskinspråks rutiner och adresseringar av extra minneskort.

DECIMAL ADDRESS		HEX ADDRESS	OKTAL ADDRESS
65408	128 BYTES LEDIGT FÖR POKE	FF80H	377:200
65046	ENKLA VARIABLER	FE16H	376:026
	SYSTEMVARIABLER		
64768		FD00H	375:000
64512	CASBUF 1	FC00H	374:000
64256	CASBUF 2	FB00H	373:000
64000		FA00H	372:000
63744		F900H	371:000
63488		F800H	370:000
63232		F700H	367:000
62976		F600H	366:000
62720		F500H	365:000
	STACK		
	16 KB RAM ARBETSMINNE ^{x)}		
49152	16 KB RAM EXTERNT MINNE	C000H	300:000
32768	1 KB RAM BILDMINNE ^{x)}	8000H	200:000
31744	1 KB ROM (PRINTER-OPTION)	7C00H	174:000
30720	1 KB (LEDIGT)	7800H	170:000
29696	1 KB ROM (IEC-OPTION)	7400H	164:000
28672	4 KB ROM (FLEXSKIV-OPTION)	7000H	160:000
24576	8 KB ROM (LEDIGT)	6000H	140:000
16384	16 KB ROM ^{x)} BASIC	4000H	100:000
0		0	0

x) ABC 80 i grundutförande
Stackpekarens startadress 65063
"Slut på program"-pekaren 65054
"Början på program"-pekaren 65052

ABC 80 BASIC – ERRATA

I ABC 80 är BASIC-tolken lagrad i ett ROM-minne som innehåller drygt 16 000 bytes. Detta program har mycket noga uttestats för att vara felfritt, och kan anses som mycket tillförlitligt.

Några skönhetsfel och mindre felaktigheter finns dock fortfarande kvar. Dessa redovisas nedan:

1. Kommandot UNSAVE ger ej felutskrift om flexskivoption saknas.
2. Kontrollen av argumentet vid dimensionering av sträng bristfällig. Exempel: DIM A $\bar{\text{X}}$ utan argument gör att ABC 80 går vilse.
3. Kontrollen av typer i uttryck missar fallet Sträng + Heltal som alltså gör att ABC 80 går vilse. Exempel $S\% = S\bar{\text{X}} + S\%$.
4. I satserna REM och Data skjuts mellanslag in på båda sidor om kolon vid listning och ED-kommando. Dock ligger satsen korrekt lagrad och READ ger alltså korrekt resultat.
5. Utskrift av vissa tal i intervallet 0.00001 – 0.1 blir fel i samband med avrundning. Exempel: 0.99 E-5 skrivs ut som 0.0000001 i stället för 0.00001.
6. Kvadratroten ur noll, $\sqrt{0}$, kan ge upphov till felaktigheter. Flyttallets 10-exponent blir ej korrekt nollställd. Exempel: PRINT 1/SQR (0) gör att ABC 80 går vilse.

Om ABC 80 får ett avbrott (går vilse, låser sig) som beror på något av ovanstående, gör följande: Nollställ ABC 80 med RESET-tangenten (sitter på baksidan av tangentbordet) eller stäng av helt med on/off-knappen. Starta sedan upp på nytt.

ANMÄRKNING

Vid utveckling och testning av ABC 80 (såväl hårdvara som mjukvara) har stora ansträngningar gjorts för att eliminera felaktigheter.

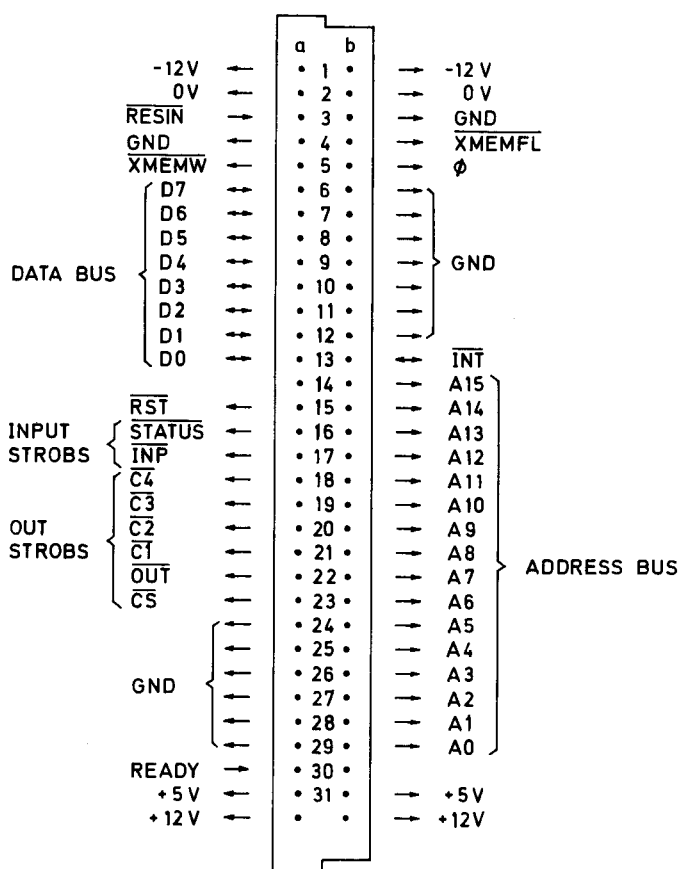
Om Ni mot förmodan upptäcker eventuella fel är vi tacksamma för ett omgående meddelande härom.

Koder på tangentbordet

ASCII-Kod	Ctrl	Shift	Tangent	ASCII-namn	Term.-funktion
0	X		E	NUL	Tidsutfyllnadstecken
1	X		A	SOH	—
2	X		B	STX	—
3	X		C	ETX	—
4	X		D	EOT	—
5	X		E	ENQ	—
6	X		F	ACK	—
7	X		G	BEL	"Pip" i högtalaren
8	X		H	BS	*) "←" tangenten
9	X		I	HT	*) "→" tangenten
10	X		J	LF	Radframmatning
11	X		K	VT	—
12	X		L	FF	Raderar skärmen
13	X		M	CR	*) "RETURN" tangenten
14	X		N	SO	—
15	X		O	SI	—
16	X		P	DLE	—
17	X		Q	DC1	—
18	X		R	DC2	—
19	X		S	DC3	—
20	X		T	DC4	—
21	X		U	NAK	—
22	X		V	SYN	—
23	X		W	ETB	—
24	X		X	CAN	*) Tar bort skriven rad
25	X		Y	EM	—
26	X		Z	SUB	—
27	X		Ä	ESC	—
28	X		Ö	FS	—
29	X		Å	GS	—
30	X		ü	RS	—
31	X	X	O	US	—
127	X		<	DEL	—

*) Dessa tecken påverkar skärmen direkt.

Appendix



ABC-bussbox

Max belastning på bussen

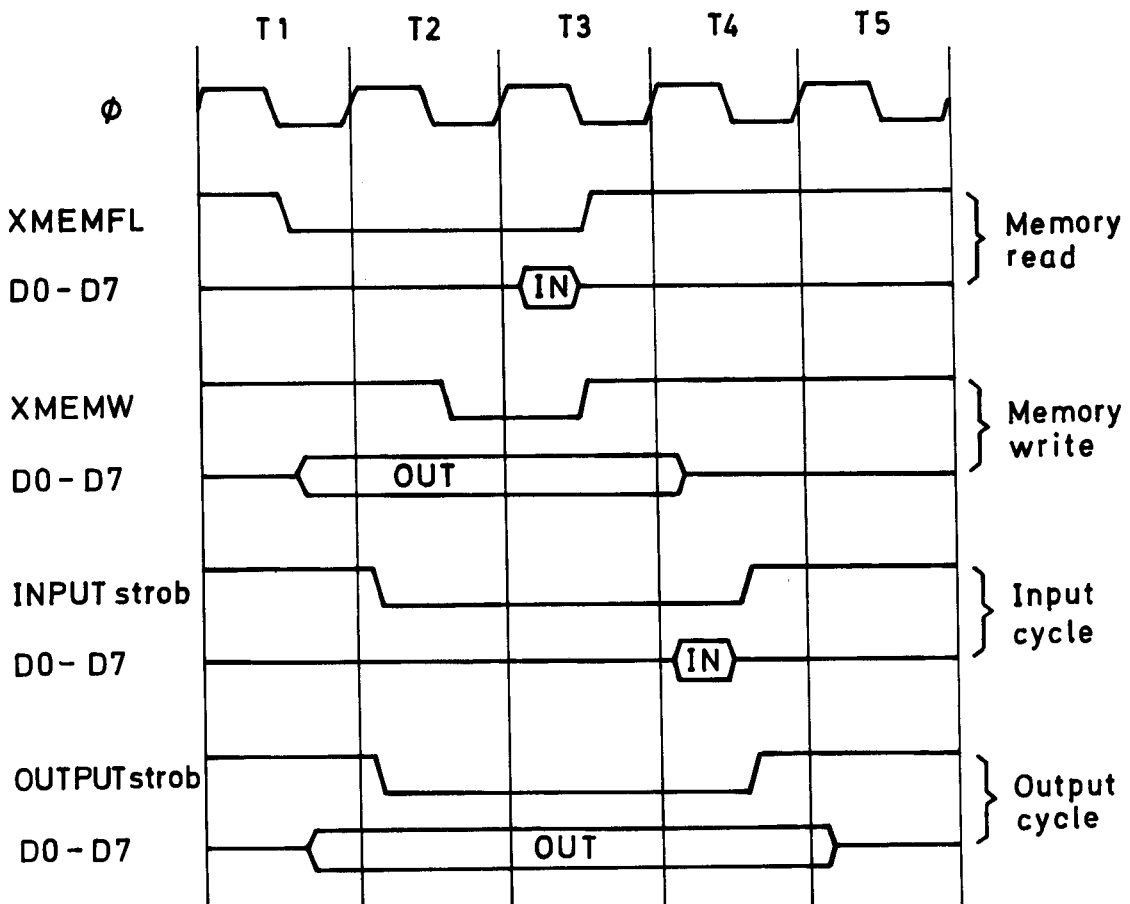
+ 5V	250 mA
+12V	125 mA
-12V	75 mA

Obs! $\pm 12V$ går även till V24-kontakten och det totala kraftuttaget får ej överskrida värdena i tabellen ovan.

Alla IN/UT-signaler är TTL-kompatibla. Följande drivkretsar användes:

adressbuss	74LS241	Texas Instrument
databuss	742S245	Texas Instrument
strobar	8205/3205	Intel
XMEMW/XMEMFL	74LS32	Texas Instrument
klocka	7406	Texas Instrument

Bussens tidsdiagram



8. Litteraturförteckning

- 1 **ABC om BASIC/Lär Dig BASIC från grunden med många intressanta användningsexempel**
Författare: Andersson, Kullbjer, Lundgren, Thomell
Förlag: DIDACT
ISBN-nr: 91-7260-254-6
Utgivn.år: 1979
- 2 **AVANCERAD PROGRAMMERING PÅ ABC 80/Lär dig professionell programmering på ABC 80. Behandlar BASIC och Assembler**
Författare: Kärrsgård, Isaksson
Förlag: Studentlitteratur
Utgivn.år: 1980
- 3 **MIKRODATORNS ABC/Beskriver hur ABC 80 är uppbyggd samt inre funktion**
Författare: Markesjö, G
Förlag: Esselte Studium
ISBN-nr: 91-24-29008-4
Utgivn.år: 1978
- 4 **STYR OCH MÄT MED ABC 80/Beskriver hur ABC 80 kan användas för styrnings- och mätningssapplikationer**
Författare: Westh, Åke
Förlag: Studentlitteratur
Utgivn.år: Hösten 1980
- 5 **ABC OM MÄTDATORSYSTEM/Beskriver grunderna för användning av ABC 80 i styr- och mätapplikationer**
Författare: Eriksson, P, Magnusson, H, Rasmusson, M
Förlag: Liber
Utgivn.år: 1980
- 6 **ABC OM DATORER**
Förlag: DIDACT
Utgivn.år: Hösten 1980
- 7 **BYGG UT ABC 80 MED DATABOARD 4680/Beskriver hur ABC 80 kan byggas ut för särskilt högt ställda krav på minneskapacitet och anpassningsmöjligheter**
Författare: SATTCO AB
Förlag: SATTCO AB
Utgivn.år: 1980
- 8 **ABC OM ANVÄNDARDOKUMENTATION/Beskriver hur användardokumentationen bör utformas**
Författare: TELUB/Luxor
Förlag: Luxor AB
ISBN-nr: Best.nr 66 79588-24
Utgivn.år: 1980
- 9 **ABC OM PROGRAMMERING OCH DOKUMENTATION**
Författare: Lundgren, Lundin
Förlag: EMMDATA
Utgivn.år: 1980
- 10 **THE BASIC HANDBOOK/Beskriver hur olika BASIC-dialekter anpassas till olika system**
Författare: Lien, David
Förlag: Compusoft Publishing, USA
ISBN-nr: 0-932760-00-7
Utgivn.år: 1979

- 11 **Z80-Assembly Language Programming Manual/Programming i Z80-Assembler**
Författare: ZILOG
Förlag: ZILOG, USA
Utgivn.år: 1978
- 12 **DATALOGI – EN INLEDANDE ÖVERSIKT/Ger en god inblick i datamaskinens sätt att arbeta**
Författare: Lunell, Hans
Förlag: Studentlitteratur
ISBN-nr: 91-44-15191-8
Utgivn.år: 1979
- 13 **DIGITALFELSÖKNING**
Författare: Brander, G
Förlag: Ingenjörsförl
ISBN-nr: 91-7284-083-8
Utgivn.år: 1979
- 14 **DATORANVÄNDNING MED IEC-BUS**
Författare: Windisch, S
Förlag: Liber läromedel
ISBN-nr: 91-40-10876-7
Utgivn.år: 1978
- 15 **DATAORDBOKEN/Begreppsförklaring – översättning till och från Sv, Eng, Ty, Fr**
Författare: SIS
Förlag: SIS
ISBN-nr: 91-7162-052-4
Utgivn.år: 1977
- 16 **DATALAGEN I PRAKTIKEN/Beskriver datalagens regler ur användarens synvinkel**
Författare: Vinge, P G
Förlag: Industriförl.
- 17 **EXEMPELSAMLING I ANVÄNDAROMRÅDEN**
Förlag: Studentlitteratur
Utgivn.år: 1980
- 18 **LÄROBOK I ASSEMBLER/Behandlar grunderna till Assembler under "Disc Operating System"**
Författare: Ärliehag, U-G
Förlag: Studentlitteratur
ISBN-nr: 91-44-07551-0
Utgivn.år: 1975

9. Sakregister

ABS 52
ADD 51
aritmetik 25
ASC 50
ASCII-aritmetik 25
ASCII-kod 32
ASCII-tabell 32
ATN 51

BASIC 17

CALL 52
CHAIN 39, 47
CHR 29, 50
CLEAR 45
CLOSE 47
CLRDOT 47
COMP 51
COS 51
CUR 52

DATA 47
datafil 43
datalogring 15
DEF FN ... 47
DIM 47
DIV 51
DOT 52

ED 45
END 47
ERR 9, 12, 55, 56
ERRCODE 52
EXP 51

felmeddelande 56
FIX 52
flexskiva 6
filnamn 24
flyttalsvariabel 20
FOR ...TO ... 47
fält 21

GET 48
GOSUB 48
GOTO 48
grafik 29
grafiska tecken 32

heltalsvariabel 20

IF ...THEN ... 14
IF ...THEN ...ELSE 48
indexerad variabel 20
IMP 54
INP 53
INPUT 48
INPUT ...48
INPUTLINE 48
INPUTLINE ... 48
INSTR 50
INT 52

kasettminne 6
KILL 45, 48
kolontecken 9
kontrolltangent 9, 61

Ledigt minne 19
LEFT 50
LEN 50
LET 48
LIST 45
ljudgenerator 36
LOAD ... 45
LOAD CAS: 45
LOG 51
Logisk operator 22, 53

MERGE ... 45
MERGE CAS: 45
MID 50
MUL 51

NAME ...AS ...46, 48
NEW 46
NEXT 48
NOT 54
NOTRACE 48
NUM 50
numerisk sträng 25, 51

ON ERROR GOTO ...49
ON ...GOSUB ...49
ON ...GOTO ...49
ON ...RESTORE ...49
OPEN ...ASFILE ...49
OR 54
OUT 49, 53

PEAK 53

PI 52
plotter 7
POKE 46, 53
PREPARE ...ASFILE ...49
PRINT 49
PRINT ... 49
procenttecken 9
programfel 12, 56
programlagring 15

RANDOMIZE 49
READ 49
realtidsklocka 42
REM 49
REN 45
RESTORE 50
RETURN 50
returtangent 9
RIGHT 51
RND 52
RUN 46
RUN ... 46
RUN CAS: 46

SAVE 46
SCR 46
semikolon 9
SETDOT 50

SGN 52
SIN 51
skifftangent 9
sol 21
SPACE 51
SQR 52
STOP 50
STRING 51
sträng 21
strängelement 21
strängvariabel 21
SUB 51
SWAP 53

TAB 52
TAN 51
tangentbord 8
TRACE 46, 50

UNSAVE 46
upper case 9

VAL 51
valutatecken 8
variabel 20
variabeltyper 20
videografikkarta 31

XOR 54

