µAB
MIKROKONSULT AB

Bruksanvisning för CP/M® på ABC80

® CP/M är ett registrerat varumärke tillhörande Digital Research, Californien USA.

FÖRORD

Denna bruksanvisning beskriver CP/M[R] på ABC80. Den vänder sig dels till Dig som skall använda CP/M för att köra färdiga program, dels till Dig som tänker utveckla egna program under CP/M. Anvisningen beskriver i första hand de funktioner och kommandon som är speciella för ABC80. Den innehåller också en kortfattad beskrivning av alla standardkommandon i CP/M.

För ytterligare beskrivningar och exempel hänvisar vi till "The CP/M Handbook" som levereras med systemet.

Kapitel 1 är en allmän introduktion till CP/M

Kapitel 2 beskriver hur man startar upp CP/M och hur man skriver kommandon.

Kapitel 3 beskriver hur man använder flexskivor och skrivare.

Kapitel 4 beskriver några vanliga kommandon under CP/M.

Kapitel 5 beskriver de kommandon som inte ingår i standard CP/M utan är speciella för ABC80.

Kapitel 6 innehåller en kortfattad beskrivning av alla CP/M-kommandon.

Dessutom finns ett antal bilagor med beskrivning på terminalrutinen, anvisningar om hur man anpassar systemet efter sin hårdvara, litteraturförteckning mm.

Denna bruksanvisning gäller BIOS version 3.3.

R CP/M är ett registrerat varumärke tillhörande Digital Research, Californien USA.

1.1 Allmänt om CP/M

CP/M (Control Program for Microprocessors) är ett operativsystem för mikrodatorer. Det utvecklades 1976 av Gary Kildall och säljs numera av hans företag, Digital Research. Redan från början fick CP/M stor spridning och är idag det mest använda operativsystemet för mikrodatorer. Enligt uppgift har CP/M sålts i c:a 200.000 exemplar.

CP/M är uppbyggt så att man lätt kan anpassa det till ett nytt system genom att skriva om en liten del kallad BIOS. Genom att skriva om BIOS kan man också anpassa det till nya yttre enheter t ex skivminnen eller skrivare. CP/M har anpassats till de flesta maskiner som använder någon av mikroprocessorerna 8080, 8085, 8086, 8088 eller Z80.

Eftersom anropen till CP/M sker likadant på alla maskiner kan ett program som skrivits på en maskin användas på en annan utan några ändringar. Denna flyttbarhet är främsta orsaken till CP/M:s popularitet.

Förutom själva kontrollprogrammet innehåller CP/M ett antal hjälpprogram eller kommandon. Med dessa kan man skriva ut, kopiera och ta bort filer. Det finns också en editor för redigering av text och program, en assembler, och flera andra hjälpprogram.

1.2 CP/M på ABC80

För att ABC80-ägare ska kunna använda CP/M har MYAB utvecklat hårdvara och mjukvara för att köra CP/M på ABC80.

Hårdvaran, UNI80, består av ett kretskort med 64 kbyte RAM, ett PROM som kopplas in vid uppstart för att läsa in CP/M från skiva och logik som kopplar om mellan CP/M och ABC80.

Mjukvaran är ovan nämnda BIOS, skrivet för ABC80. Det innehåller drivrutiner för de flesta förekommande flexskiveenheter till ABC80. Det innehåller också rutiner för hantering av bildskärm och tangentbord samt för anpassning till serie- och parallellskrivare.

2 Att använda CP/M

2.1 Uppstart av CP/M, (kallstart)

CP/M på ABC80 startas upp på följande sätt:

1) Sätt i en flexskiva med system i enhet A.

2) Tryck in resetknappen på ABC80 c:a 3 sekunder.

Efter en stund kommer utskriften:

```
64k CP/M vers 2.2 - MYAB BIOS vers 3.3 XY

A>
```

CP/M är nu klart att använda. Bokstaven och >-tecknet kallas **prompter** och anger att systemet är klart att ta emot ett kommando. På raden innan anger 64 tillgängligt minne, 2.2 är versionsnummer på CP/M och 3.3 är versionsnummer på BIOS. Bokstäverna X resp Y anger vilken typ av skiva resp skrivare systemet är avsett för. Se vidare bilaga F.

2.2 Att använda CP/M

Du kan nu ge kommandon, dvs instruktioner till CP/M genom att skriva på tangentbordet. Allt du skriver kommer att skrivas ut, **ekas**, på bildskärmen.

Förutom de vanliga tecknen kan du skriva speciella tecken, **kontrolltecken**, genom att trycka ner tangenten CTRL samtidigt med en annan tangent. Kontrolltecknen ekas som ^X där X är den andra tangenten. Vissa av kontrolltecknen har speciella funktioner, se 2.8.

För att ge ett kommando skriver du kommandots namn och trycker sedan på tangenten RETURN. Denna tangent betecknas i fortsättningen med <ret>.

2.3 Omstart av systemet (varmstart)

Om du trycker ^C, dvs tangenterna CTRL och C samtidigt, läses systemet in igen från skiva. Dessutom görs vissa initieringar i systemet. Detta är det normala sättet av avbryta ett program under körning. Det måste också göras varje gång du bytt någon av flexskivorna.

För att du skall kunna göra varmstart måste det finnas en kopia av systemet på skivan i enhet A.

2.4 Aktuella skivan

I CP/M betraktas alltid en av skivorna som "aktuell skiva". CP/M antar att alla program och filer ligger på aktuella skivan om du inte anger något annat. Bokstaven i promptern

anger vilken skiva som är aktuell skiva.

Aktuella skivan kan ändras med kommandona A: resp B:, som sätter aktuell skiva till A resp B.

2.5 Filer och filnamn

På flexskivorna ligger informationen lagrad som namngivna filer. En fil kan innehålla ett program eller text eller data.

Ett filnamn består av 3 delar: enhetsnamn (t ex A: eller B:), själva namnet (max 8 tecken), och filtypen (max 3 tecken).

Ett typiskt filnamn är:

```
A:STAT.COM
```

som betecknar filen STAT av typ COM på skiva A. Om enheten och kolonet utelämnas används aktuell skiva.

Ett filnamn får innehålla alla tecken utom:

```
< > . , ; : = ? * Ä Å
```

Man bör dock undvika små bokstäver och kontrolltecken i filnamn.

I vissa fall behöver man referera till en grupp av filer, t ex alla filer av samma typ. Detta görs med ofullständiga filnamn (ofn).

Ett ofullständigt filnamn får även innehålla tecknen frågetecken (?) och asterisk (*). Ett frågetecken betecknar ett godtyckligt tecken och en asterisk betecknar ett godtyckligt antal godtyckliga tecken. För att förtydliga tar vi som exempel den medföljande systemskivan. Den innehåller följande filer:

```
ABCDISK.COM
ASM.COM
COPYDISK.COM
DDT.COM
DUMP.ASM
ED.COM
FORMAT.COM
LOAD.COM
MOVCPM.COM
PIP.COM
SD.COM
SET.ASM
SET.COM
STAT.COM
SUBMIT.COM
SYSGEN.COM
XSUB.COM
```

S*.COM betecknar då STAT.COM, SUBMIT.COM och SYSGEN.COM medan *.* är alla filer (alla namn stämmer). ?O??????.???

betecknar COPYDISK.COM, LOAD.COM och MOVCPM.COM. Prova med kommandot DIR ofn, som skriver ut alla filnamn som "stämmer" med ofn.

Själva CP/M-systemet är inte lagrat som en fil, utan ligger i en särskild area på skivan. Det syns därför inte när du gör DIR och kan bara kopieras med ett särskilt kommando (SYSGEN).

För ytterligare information om filer och deras egenskaper, se bilaga B.

2.6 Stora och små bokstäver

På tangentbordet på ABC80 kan man skriva både stora och små bokstäver. När du skriver ett kommando under CP/M kommer samtliga små bokstäver i kommandoraden att översättas till stora. Du kan därför valfritt skriva kommandon och filnamn med stora eller små bokstäver. CP/M kommer att tolka dem som stora.

Eftersom CP/M är ett amerikanskt system betraktas inte tecknen å, ä, ö, Å, Ä, och Ö som bokstäver. De kommer således inte att översättas.

Observera att de flesta program inte översätter till stora bokstäver. Du kan således mata in både stora och små bokstäver med t ex ED.

I vissa program, t ex CBASIC, kan man tom skapa filer med namn som innehåller små bokstäver. **Det finns ingen möjlighet att nå sådana filer med vanliga CP/M kommandon.** Det är därför lämpligt att alltid använda stora bokstäver i filnamn.

2.7 Kommandon

Ett kommando ges genom att skriva kommandots namn följt av <ret>. Ett exempel är kommandot DIR som skriver ut vilka filer som finns på en skiva.

Många kommandon kan ha en eller flera **parametrar** som anger vad kommandot skall göra. Parametrarna skrivs efter kommandonamnet.

Exempel:

```
ERA  *.TXT
```

som tar bort alla filer av typ TXT.

De flesta kommandon är helt enkelt speciella program, lagrade på skiva. Sådana kommandon kallas också **systemprogram.** Med CP/M-terminologi kallas de för **transienta kommandon.** Alla sådana kommandon ligger lagrade på filer av typ .COM. För att köra kommandot STAT måste således filen STAT.COM finnas på aktuella skivan. Du kan även köra kommandon på den andra skivan genom att skriva ut skivans namn, t ex B:ASM.

Vissa kommandon är dock inbyggda i CP/M. De är alltid till-

gängliga oberoende av vilka skivor som sitter i.

2.8 Specialtecken

När du skriver på tangentbordet har vissa kontrolltecken en speciell funktion. Följande specialtecken finns:

^C	Varmstart, dvs systemet läses in och initeras.
^E	Flyttar cursorn till ny rad utan att sända det som redan skrivits. Används för att skriva rader som inte går in på en rad på skärmen.
^H eller <-	Tar bort senast skrivna tecken och backar cursorn.
^I eller ->	Flyttar cursorn till nästa tab-läge. Tabbarna är placerade i var 8:e position.
^P	Startar samtidig utskrift på skrivaren av allt som skrivs på skärmen. Funktionen avbryts med ytterligare ett ^P. Observera att utskriften görs på LST:. Om du inte har satt någon enhet till LST: med STAT kommer alla tecken att dubbleras på skärmen. Sätt rätt enhet enligt 3.5 eller 3.7.
^S	Stoppar tillfälligt utskriften på skärmen. Utmatningen startas igen genom att trycka på valfri tangent (t ex ^S).
(^R)	Flyttar cursorn till ny rad och skriver om raden som är under inmatning.
(^U)	Förkastar raden som håller på att matas in och flyttar cursorn till ny rad.
^X	Backar till början på raden.
^Z	Filslutstecken vid inmatning från tangentbordet, används av PIP och ED.

De viktigaste tecknen att känna till är ^C, ^H, ^I och ^S. De tecken som står inom parentes används bara när CP/M körs på en skrivande terminal.

3 Flexskivor och yttre enheter

UNI80 kan användas ihop med följande flexskiveenheter:FD2, FD2U, FD2UD, FD2D, FD4, Datadisk 80, Datadisk 82, Datadisk 84, Datadisk 86, Datadisk 88 och Luxor 830.

Förutom flexskivorna kan man ansluta en serieskrivare eller en terminal till V24-kontakten på ABC80. Det finns också möjlighet att ansluta en skrivare via ett anpassningskort på ABC-bussen.

3.1 Olika typer av flexskiveenheter.

CP/M till ABC80 levereras antingen på 5 1/4"-skiva eller också på 8"-skiva. I bägge fallen levereras CP/M på en enkelsidig skiva i enkel densitet.

UNI80 kan använda såväl enkel- som dubbelsidiga skivor i enkel eller dubbel densitet. Se bara till att omkopplarna på flexskiveenheten står i rätt läge. Systemet kontrollerar storleken på skivorna efter varje varmstart.

Om du har en flexskiva som medger dubbel densitet bör du kopiera över dina skivor till dubbel densitet så att du får mer plats. Bilaga D beskriver hur du gör. I vissa fall måste du tillverka ett nytt system för att kunna utnyttja hela kapaciteten hos din flexskiva, se bilaga F.

3.2 Formattering och initiering

Innan en flexskiva kan användas måste den formatteras. Detta görs med kommandot FORMAT (se 5.1).

3.3 Byte av flexskivor, varmstart

CP/M har en del information om skivorna lagrad internt i minnet. Denna information uppdateras varje gång du gör varmstart. Varje gång du bytt någon av skivorna måste du därför göra varmstart, dvs trycka ^C.

I vissa fall upptäcker systemet att du bytt en skiva. Skivan kommer då att bli skrivskyddad (se bil B) tills nästa varmstart.

Vissa program, t ex WORDSTAR, uppmanar dig att byta skiva under programmet. När du kör ett sådant program skall du **inte** trycka ^C när du bytt skiva eftersom programmet då avbryts. Programmet tar istället själv hand om initieringen av systemet.

3.4 Kopiering av flexskivor

Man har ofta behov av att kopiera flexskivor. Kommandot COPYDISK (se 5.2) gör en kopia av en hel skiva. COPYDISK kan emellertid bara kopiera mellan två skivor i samma format och densitet.

Om du vill kopiera till en skiva i annan densitet, eller om

du vill kopiera enstaka filer, kan du använda kommandot PIP (se 6.8). Själva CP/M-systemet kopieras med kommandot SYSGEN (se 6.14).

3.5 RAM - skivan

På UNI80 finns 64 kbyte RAM-minne som används under CP/M. Det ordinarie minnet i ABC80:n frigörs då och kan användas för andra ändamål. Du kan nå detta minne som enhet C:

Enhet C: används på exakt samma sätt som flexskivorna A: och B:. Eftersom det ligger i RAM blir det c:a 200 gånger snabbare än skivorna.

Enhet C: kan användas för att lagra ofta använda kommandon och filer. Kom bara ihåg att inte stänga av datorn utan att kopiera de filer som du vill ha kvar till en flexskiva.

Enhet C: rymmer 3 kbyte mindre än tillgängligt minne i ABC80, dvs 13 kbyte i en standard ABC80 och 29 kbyte i en 32 kbyte ABC80.

3.6 V24-kontakten, Serieskrivare

När du använder UNI80 kan du ansluta en serieskrivare till V24-kontakten på ABC80.

Med kommandot:

```
STAT LST: = UL1:
```

talar du om för systemet att serieutgången skall användas som utskriftsenhet.

Baudraten sätts med kommandot:

```
SET BAUD=xx
```

där xx är 110, 300, 600, 1200, 2400 eller 4800.

3.7 Körning från yttre terminal

Om du så önskar kan du köra CP/M från en terminal ansluten till V24-kontakten. Detta görs på följande sätt:

1) Anslut terminalen till V24-kontakten.

2) Sätt baudraten på det sätt som anges under 2.4.

3) Skriv följande kommandon:

```
STAT LST: = UL1:
STAT RDR: = PTR:
STAT CON: = BAT:
```

Du kan nu skriva kommandon från den anslutna terminalen

istället för från ABC80:n. För att komma tillbaka skriver du:

```
STAT CON: = TTY:
```

För en närmare förklaring, se STAT (6.11).

3.8 Skrivare

Om du har en skrivare mad anpassningskort till ABC-bussen kan du använda den under CP/M. Systemet kan anpassas till flera olika skrivare, se bilaga F.

Med kommandot:

```
STAT LST: = LPT:
```

talar du om för systemet att skrivaren skall användas som utskriftsenhet.

4 Några vanliga kommandon

4.1 DIR

DIR skriver ut namnen på de filer som finns på flexskivorna. Kommandot har formen:

```
DIR ofn
```

där ofn är ett ofullständigt filnamn (se 2.4). Om ofn utelämnas skrivs namnen på alla filer på aktuella skivan ut.

Man kan också skriva ut skivans innehåll med STAT ofn (se 6.12).

4.2 TYPE

TYPE skriver ut innehållet i en fil på skärmen. Kommandot har formen:

```
TYPE filnamn
```

Om du först trycker ^P (se 2.6) kommer filen även att skrivas ut på skrivaren.

4.3 REN

REN ändrar namnet på en fil. Kommandot har formen:

```
REN nfilnamn=gfilnamn
```

Filen gfilnamn får då namnet nfilnamn.

Exempel:

```
REN GAMMAL.TXT=AKT.TXT
```

döper om filen AKT.TXT till GAMMAL.TXT.

4.4 PIP, kopiering av filer

PIP kan användas för att kopiera en fil. Kommandot har då formen:

```
PIP tfil=ffil
```

Kommandot gör en kopia av filen ffil och ger den namnet tfil. Tfil behöver inte ligga på samma skiva som ffil.

Exempel:

```
PIP B:KOPIA.TXT=ORG.TXT
```

kopierar ORG.TXT på aktuella skivan till KOPIA.TXT på skiva B.

PIP kan även användas för att slå ihop filer, numrera rader mm (se 6.8).

Själva CP/M-systemet kopieras med SYSGEN (se 6.14).

5 Särskilda kommandon på ABC80

5.1 FORMAT, formattering av nya skivor

Innan en flexskiva kan användas under CP/M måste den formatteras. Detta görs med kommandot FORMAT. Kommandot används på följande sätt:

Sätt skivan som skall formatteras i enhet B.

```
A> FORMAT
Formatteringsprogram 1.0 för 5" skiva.
Enkel/dubbel densitet (e/d) ? e <ret>
Formattering av skiva B: - Ok (j/n) ? j <ret>
Helt säker (j/n) ? j <ret>
Fler skivor med samma format (j/n)? n <ret>
```

Om du vill ha en skiva i dubbel densitet svarar du "d" på första frågan. Du får då också frågan:

```
Enkel/dubbel-sidigt (e/d) ?
```

Om du vill formattera fler skivor i samma format sätter du i nästa skiva och svarar "j" istället för "n" på sista frågan.

Om du har 8"-skivor står det naturligtvis 8" istället för 5" på första raden.

5.2 COPYDISK

Copydisk kopierar en hel flexskiva från enhet A till enhet B. Det används t ex för arkivkopiering ("backup") av skivor. Skivan i enhet B måste vara formatterad i samma format som skivan i enhet A. COPYDISK kollar själv formatet på skivorna.

När du skrivit kommandot kommer texten:

```
Copydisk 1.0
Kopiering av skiva A till skiva B
Ok (J/N)?
```

Sätt nu skivan som skall kopieras i enhet A, och en tom skiva i enhet B, svara sedan "j <ret>". Innehållet på skiva A kopieras nu över till enhet B. För varje kopierat spår skriver COPYDISK en punkt på skärmen.

När kopieringen är klar kommer frågan:

```
Kopiera fler skivor (J/N)?
```

Du kan nu sätta i två nya skivor och fortsätta kopieringen.

När du kopierat färdigt sätter du tillbaka systemskivan i enhet A och svarar "n <ret>".

5.3 ABCDISK

ABCDISK används för att kopiera filer från en ABC80-skiva till en CP/M-skiva. När man kör ABCDISK skall ABC-skivan sitta i enhet B. Filerna kopieras till enhet A. Om du vill kopiera ett BASIC-program skall det sparas med LIST så att det blir en fil av typ BAS. Filer av typ BAC går inte att använda under CP/M.

När du kör kommandot skrivs följande ut på skärmen:

```
Detta program kopierar filer från
en ABC-skiva i B: till CP/M-filer
med samma namn på skiva A:.
Alla filnamn på ABC80-skivan kommer
att räknas upp. Skriv J efter de filer
som skall kopieras.
```

Programmet räknar nu upp filerna på ABC80-skivan en i taget. Skriv J efter de som skall kopieras och N efter övriga.

5.4 SET

SET sätter hastigheten på V24-porten. Kommandot har formen:

```
SET BAUD=xx
```

Baudrdaten xx skall vara 110, 300, 600, 1200, 2400 eller 4800 baud.

6 Samtliga CP/M-kommandon

Detta avsnitt innehåller en beskrivning av samtliga standard kommandon som finns under CP/M på ABC80. De kommandon som bara finns på ABC80 har redan beskrivits i avsnitt 4. De är markerade med (s) i nedanstående lista. Vissa kommandon är inbyggda i CP/M och är alltid tillgängliga oberoende av vilka skivor som sitter i. De är markerade med (i) i nedanstående lista. Övriga kommandon är lagrade på skiva som .COM-filer.

Kommandot SD hör inte till standard CP/M utan kommer från "CP/M Users Group".

Det bör påpekas att det SYSGEN som följer med är en speciell version för ABC80, och inte det som följer med standard CP/M, samt att MOVCPM inte fungerar som det brukar på vanliga CP/M-system.

Följande kommandon finns:

(s)	ABCDISK	Kopierar från ABC80-skiva till CP/M-skiva.
	ASM	Assemblerar en fil.
(s)	COPYDISK	Kopierar en hel skiva från A: till B:.
	DDT	Används vid test (debugging) av assemblerprogram.
	DIR	Skriver ut filkatalogen.
	ED	Texteditorn, för inmatning och redigering av text.
(i)	ERA	Tar bort filer.
(s)	FORMAT	Formatterar en flexskiva.
	LOAD	Gör om en .HEX-fil till en .COM-fil.
	MOVCPM	Gör om CP/M till en annan storlek.
	PIP	Kopierar och modifierar filer.
(i)	REN	Byter namn på en fil.
(i)	SAVE	Sparar minnet på disk.
(u)	SD	Som DIR fast bättre
	STAT	1) Skriver ut information om skivorna. 2) Kopplar ihop logiska och fysiska enheter.
	SUBMIT	Kör ett antal kommandon upplagda i en fil.
	SYSGEN	Kopierar själva CP/M-systemet från en skiva till en annan.

(i) TYPE Skriver ut en fil på skärmen.

(i) USER Sätter användarnummer.

6.1 ASM

Detta kommando assemblerar en textfil innehållande källtext med standard 8080-mnemonics. Assemblern producerar två filer, en objektfil i intel-hex format (typ .HEX) och en listfil (typ .PRN). Objekt- och listfilerna får samma namn som källfilen.

Objektfilen kan sedan laddas med LOAD (se 6.6) så att du får ett körbart program (= ett nytt kommando). Listfilen innehåller källtexten med eventuella felmeddelanden samt den producerade maskinkoden i hexadecimal form. Dessutom skrivs alla felmeddelanden ut på skärmen.

Kommandot har formen:

```
ASM filnamn
```

eller

```
ASM filnamn.kol
```

(Obs! kol är **inte** filtypen).

Om man använder första formen på kommandot assembleras filen filnamn.ASM. Objektfilen hamnar i filnamn.HEX och listfilen i filnamn.PRN. Samtliga på aktuella skivan.

Om man använder den andra formen är k, o och l parametrar som anger enhet för käll-, objekt- och listfilerna. k anger på vilken enhet (A-P) som källfilen finns, o anger på vilken enhet objektfilen skall hamna och l anger var utskriften skall hamna. Om o=Z produceras ingen objektfil. Om l=X skrivs listan ut på skärmen och om l=Z blir det ingen listfil alls.

Exempel:

ASM TEST	Assemblerar TEST.ASM på aktuella skivan. Objektfilen TEST.HEX och listfilen TEST.PRN läggs på aktuella skivan.
ASM TEST.BAZ	Assemblerar B:TEST.ASM. Lägger objektfilen på A: och producerar ingen listfil.
ASM TEST.BZZ	Assemblerar B:TEST.ASM och producerar varken objektfil eller listfil. Bra för snabb testassemblering när man söker efter fel. Alla felmeddelanden skrivs ut på skärmen.

6.2 DDT

DDT är ett program som används för att testköra framför allt assemblerprogram. Man kan köra programmet en instruktion i taget, sätta brytpunkter, titta & ändra i minnet osv.

Kommandot har formen:

```
DDT
```

eller

```
DDT filnamn
```

Om du ger kommandot med filnamn läser DDT in filen, som ska vara av typ .COM eller .HEX, och skriver ut texten NEXT PC följt av en rad med 2 tal. NEXT är adressen i hex till nästa lediga sida och PC är startvärde för programräknaren.

När DDT är klart att ta emot ett kommando kommer DDT:s prompter, ett minustecken (-).

Alla kommandon i DDT består av en bokstav. Eventuella parametrar skrivs direkt efter bokstaven utan mellanslag. Parametrarna åtskiljs med kommatecken.

Alla tal skrivs hexadecimalt. Alla bokstäver kan skrivas som små eller stora.

Följande kommandon finns:

Aadr	(Assemble) Startar den inbyggda assemblern. Första instruktion läggs i adr. Skriv standard 8080-mnemonics med hexadecimala tal (inga h behövs). Tom rad avslutar.
Dadr1,adr2	(Dump) Skrivar ut innehållet i minnet från adr1 till adr2. Utskriften sker hexadecimalt med ASCII-översättning i kanten om det går. Om adr2 utelämnas skrivs 12 rader ut. Om adr1 utlämnas sker utskrift från senast utskrivna adress.
Fa1,a2,tal	(Fill) Fyller minnet från a1 till a2 med tal. Bra för att nollställa minnet eller för att fylla det med någon konstant.
Gs,b1,b2	(Go) Startar körning av program i adress s med brytpunkter i adress b1 och b2. Om programmet utför en instruktion som ligger i adress b1 eller b2 stoppas det. Observera att det inte finns någon möjlighet att få tillbaka kontrollenöver programmet om det aldrig kommer till någon av brytpunkterna. Utelämnas s tas nuvarande pc som startpunkt.
Ifilnamn	(Input FCB) Lägger in filnamn i standard FCB i adress 05ch. Antar standardskivan. För att läsa från annan skiva kan du gå in och ändra med s i 05ch.

Htall,tal2 (Hex) Beräknar summan av och skillnaden mellan tall och tal2 hexadecimalt och skriver ut resultatet.

Ladr1,adr2 (List) Skriver ut innehållet i minnet från adr1 till adr2 i assemblerform. Utelämnas adr2 skrivs 12 rader, utelämnas båda skrivs 12 rader från den sist utskrivna adressen.

Ma1,a2,a3 (Move) Flyttar minnesarean mellan a1 och a2 till a3.

R (Read) Läser in den fil, vars filnamn har lagts i FCB:t med I-kommandot, till minnet. Filen skall vara av typ .COM eller .HEX.

Sadr (Set) Sätter minnet till visst innehåll. adr är första adressen. Adressen och minnescellens innehåll skrivs ut. Om du nu ger ett . hexadecimalt tal följt av <ret>, skrivs det in i cellen. Om du bara trycker <ret> ändras inte innehållet i cellen. Sedan skrivs nästa adress ut. För att avbryta skriver du ett icke hexadecimalt tecken (tex .) följt av <ret>.

Tant (Trace) Används för att köra ett program en instruktion i taget. ant anger hur många instruktioner som ska köras. Om ant utelämnas körs 1 instruktion. För varje instruktion skrivs alla register + instruktionen ut. (Se X nedan.) När ant instruktioner är körda skrivs stoppadressen ut. Du kan också stoppa i förtid genom att trycka ner någon tangent.

Uant (Untraced run) Som T, men ingenting skrivs ut. Kan användas för att komma en bit framåt i programmet snabbare än med T men utan att riskera att förlora kontrollen som med G.

X (Examine) Skriver ut alla register och flaggor på samma sätt som T. Data skrivs på formen:

```
CfZfMfEfIf A=bb B=dddd D=dddd H=dddd S=dddd
P=dddd inst
```

där f är 0 eller 1, (motsvarande flaggas värde), bb är en byte och dddd ett ord. inst är den disassemblerade instruktionen på adressen i P, programräknaren.

Xr Genomatt skriva Xr där r är någon av bokstäverna C,Z,M,E,I, A,B,D,H,S eller P kan du ändra motsvarande flagga eller register på samma sätt som med S-kommandot.

6.3 DIR

DIR skriver ut filbiblioteket på flexskivorna, dvs namnen på

de filer som finns på skivorna. Kommandot har formen:

```
DIR ofn
```

där ofn är ett ofullständigt filnamn (se 2.4). Om ofn utelämnas skrivs hela biblioteket på aktuella skivan ut.

Man kan även skriva ut biblioteket med STAT ofn (se 6.12).

6.4 ED

6.4.1 Inledning

ED är ett program för inmatning och redigering av text, (en editor). ED används för uppläggning och ändring av källprogram, och för att skapa .SUB-filer som senare skall användas med SUBMIT. ED kan också användas för enklare textbehandling, t ex för att lägga upp brev och listor, som senare skall skrivas ut med PIP.

ED startas med kommandot:

```
ED filnamn.typ
```

Om filnamn.typ redan finns kan du göra ändringar och tillägg i den. Om filen inte finns kommer ny fil att skapas.

6.4.2 Textfiler

Innan vi beskriver ED skall vi titta lite på hur en textfil är uppbyggd under CP/M. En textfil är helt enkelt en fil som innehåller text, dvs skrivbara tecken. Texten är normalt uppdelad i rader.

Under CP/M lagras filen som en följd av tecken. Raderna avgränsas då med teckenkombinationen <CR><LF>. Slutet på filen markeras med tecknet ^Z.

Texten:

```
Rad1
Rad2
```

lagras således som:

```
Rad1<CR><LF>Rad2<CR><LF>^Z
```

Dessa detaljer behöver du normalt inte tänka på eftersom ED tar hand om dem. Det är dock bra att veta att det finns två tecken mellan raderna.

6.4.3 Filerna och arbetsarean

I ED finns en arbetsarea i minnet där texten lagras. Arbetsarean rymmer c:a 30.000 tecken dvs c:a 15 fullskrivna A4-sidor. ED kan bara arbeta med text som ligger i arbetsarean. Om du av någon anledning måste arbeta med en större textfil kan du ta in en del i taget i arbetsarean. Det är dock bättre att istället dela upp texten på flera mindre filer.

ED används normalt för att ändra i en existerande textfil. Arbetsgången blir då följande. Först läser du in hela den gamla filen till arbetsarean, sedan gör du alla ändringar och tillägg, och slutligen skriver du ut den ändrade texten. När du skapar en ny fil gör du på samma sätt, så när som på att det inte finns någon gammal fil att läsa in. Eftersom ED inte ändrar i den gamla filen utan skriver till en ny så

finns den gamla alltid kvar som reserv.

För att allt detta skall fungera så smidigt som möjligt gör ED på följande sätt. När du skriver kommandot:

```
ED filnamn.typ
```

skapar ED en tom fil med namnet filnamn.$$$. Denna fil är den nya filen som du skriver ut den ändrade texten till.

När körningen senare avslutas kommer ED att döpa om filnamn.typ till filnamn.BAK och filnamn.$$$ till filnamn.typ.

När du är klar finns således den nya texten i filnamn.typ och den gamla i filnamn.BAK. Om det fanns en gammal filnamn.BAK kommer den att försvinna.

Om ED avbryts t ex med ^C eller med Q-kommandot kommer den gamla filen att finns kvar oförändrad.

6.4.4 Teckenpekaren

ED är en sk pekar-editor. Namnet kommer av att det i ED finns en **teckenpekare** (TP) som pekar någonstans i arbetsarean. De flesta kommandon i ED utgår från TPs läge.

TP kan stå var som helst i texten, med står oftast i början på en rad. Du kan ta reda på var TP står med kommandot T.

För att riktigt förstå hur TP fungerar bör du tänka på att TP alltid står **mellan** två tecken.

Alla kommandon som går framåt från TP gäller alltså fr o m tecknet efter TP, och alla som går bakåt gäller fr o m tecknet före TP. När TP står i början på en rad står den alltså efter tecknen <CR><LF>, som skiljer raderna åt, och före första tecknet på raden.

6.4.5 Kommandon i ED

När ED är klart att ta emot ett kommando, skriver det ut följande prompter:

```
nn:*
```

Där nn är numret på den rad i texten som TP befinner sig på.

Du kan nu ge ett ED-kommando följt av <ret>.

De flesta ED-kommandon består av en bokstav. Kommandona kan skrivas med antingen stor eller liten bokstav. **Obs!** Om någon av kommandona I, F, S eller N skrivs med stor bokstav kommer alla små bokstäver i den inmatade texten att översättas till stora. Skriv därför alltid kommandona med liten bokstav vid normalt bruk.

Samtliga kommandon i ED finns beskrivna i nästa avsnitt. Kommandona är för tydlighets skull skrivna med stor bokstav.

I beskrivningen står det (-) framför vissa av kommandona.

Detta betyder att kommandot kan föregås av ett minsutecken.

Till många av kommandona hör ett tal "n". som skall vara mellan 0 och 65535. On talet utelämnas används värdet 1. Tecknet "#" ger ett stort tal (65535), och används när du vill upprepa ett kommando så många gånger som möjligt. Observera att talet 0 har en speciell betydelse för vissa kommandon.

I vissa kommandon ingår en eller flera **strängar.** En sträng är en följd av tecken avslutat med tecknet ^Z. Om strängen innehåller specialtecknet ^L kommer detta att tolkas som följden <CR><LF>, dvs ny rad.

Om en sträng står sist på en rad och inte följs av något annat kommando kan det avslutande ^Z utelämnas.

6.4.6 Samtliga kommandon i ED

n: Flyttar TP till början på rad n.

:m Utför det följande kommandot tom rad m. Detta kommando kan användas ihop med det ovanstående för att utföra ett kommando på ett antal rader.

Exempel:

```
10::15T
```

skriver ut raderna 10-15.

(-)n Flyttar TP n rader framåt (n) eller bakåt (-n) och skriver ut raden TP hamnar på. TP hamnar i början på raden. Samma sak som (-)nLT.

Om utelämnas får du kommandot <ret>, som flyttar TP till början på nästa rad, och skriver ut den.

nA (Append) Läser in n rader till arbetsarean.

Man börjar normalt arbetet med #A, som läser in hela filen (eller så mycket som får plats i arbetsarean).

0A läser in tills arbetsarean är halvfull.

(-)B (Begin/Bottom) Flyttar TP till början (B) resp slutet (-B) på arbetsarean.

(-)nC (Character move) Flyttar TP n tecken framåt eller bakåt (-). Observera att det finns två tecken, <CR> och <LF>, mellan raderna.

(-)nD (Delete) Tar bort n tecken framåt eller bakåt (-) från TP. Observera att konstiga saker kan hända om du tar bort ett av tecknen <CR> eller <LF> mellan raderna. Om du däremot tar bort bägge slås raderna ihop till en rad.

E (End) Avslutar ED på normalt sätt, dvs skriver ut texten till den nya filen och döper om filerna.

nFsträng^Z (Find) Letar reda på den n:te förekomsten av "sträng" och sätter TP omedelbart efter. Om ED inte hittar "sträng" (n gånger) kommer TP inte att flyttas.

Om kommandot inte följs av ytterligare kommandon på samma rad kan ^Z utelämnas.

H (Head) Gör samma sak som E, följt av ED igen, dvs skriver ut texten till den nya filen, döper om filerna, och skapar en ny .$$$-fil. Kommandot kan användas för att se till att de ändringar du gjort skrivs ut på skivan, efter som innehållet i arbetsarean kan förstöras t ex vid strömavbrott.

I (Insert) Efter kommandot I är du i inmatningsmod. Du kan nu skriva in text som läggs i arbetsarean efter TP. Inmatningen avslutas med tecknet ^Z.

I inmatningsmod kommer allt du skriver att läggas in i arbetsarean tecken för tecken med följande undantag:

^L eller
<ret> Lägger in <CR><LF> i arbetsarean, dvs ger ny rad i texten.

^Z Avslutar inmatningen.

Dessutom har alla specialtecken enl 2.6 sin vanliga betydelse.

Observera att tabbar (-> eller ^I) läggs in som tab-tecken (^I) i filen. ED visar dock tabbarna expanderade till var 8:e position. Vid utskrift med PIP skall enhet PRN: eller parametern T8 användas om du vill ha tabbarna expanderade.

Isträng^Z Lägger in strängen "sträng" i arbetsarean efter TP utan att gå över i inmatningsmod. Om strängen innehåller tecknet ^L kommer detta att ge ny rad.

Om det avslutande ^Z utelämnas kommer strängen att följas av ny rad, dvs Istr<ret> är samma sak som Istr^L^Z<ret>.

nJsöksträng^Zinsträng^Zslutsträng^Z (Juxtapose) Detta något speciella kommando kombinerar F, I och D. Det letar först reda på strängen "söksträng" (på samma sätt som F). Därefter lägger det in "insträng" i arbetsarean direkt efter "söksträng". Slutligen tar det bort alla tecken fram till första förekomsten av "slutsträng". "Slutsträng" lämnas dock kvar oförändrad.

Alltihop upprepas n gånger.

Exempel:

```
J.^ZNy mening.^Z^L^Z
```

byter ut allt från nästa punkt (.) till slutet på raden (^L) mot texten "Ny mening.".

(-)nK (Kill) Tar bort n rader framåt eller bakåt (-).

Om TP inte står i början på en rad tar +K bort allt efter TP på raden medan -K tar bort allt före TP.

(-)L (Line move) Flyttar TP n rader framåt eller bakåt (-). TP sätts i början på raden.

0L flyttar TP till början på den rad den står på.

nMkommando^Z (Macro) Upprepar kommandot "kommando" n gånger. Om n utelämnas eller är 0 eller 1 upprepas kommandot så många gånger det går.

Som vanligt kan det avslutande ^Z utelämnas om kommandot

nNsträng^Z (Next) Hittar nästa förekomst av "sträng" i en stor fil (större än arbetsarean). Samma sak som F med den skillnaden att ED automatiskt utför A och W om strängen inte finns i arbetsarean.

^Z kan utelämnas om kommandot står sist på kommandoraden.

O (Original) Tar bort alla ändringar som gjorts sedan du började använda ED och börjar om med den gamla filen. Om du använt kommandot H kommer den gamla filen att vara den du hade när du senast gjorde H.

När du ger kommandot O frågar ED:

```
O-(Y/N)?
```

Svara Y (Yes) om du verkligen vill ta bort alla ändringar, annars N.

(-)nP (Page) Flyttar TP en sida (22 rader) framåt eller bakåt (-). och skriver ut en sida. Kommandot upprepas n gånger.

Q (Quit) Avbryter ED **utan** att genomföra de ändringar som gjorts i texten. Den gamla filen blir kvar oförändrad. Om du använt kommandot H kommer filen att innehålla de ändringar som gjordes innan H-kommandot. (Se även O-kommandot).

Om du ger kommandot Q ställer ED frågan:

```
Q-(Y/N)?
```

Svara Y om du vill avbryta utan ändringar, annars N.

Obs! Det normala sättet att avsluta ED är med kommandot E.

Rfilnamn^Z (Read) Läser in innehållet i filen filnamn.LIB och lägger det i arbetsarean efter TP.

R (Read) Läser in innehållet i filen X$$$$$$$.LIB och lägger det efter efter TP. Se även X-kommandot.

nSsöksträng^Zbytsträng^Z (Substitute) Letar reda på första förekomsten av strängen "söksträng" och byter ut den mot "bytsträng". TP hamnar efter den utbytta strängen. Alltihop upprepas n gånger.

(-)nT (Type) Skriver ut n rader före resp efter (-) TP. Om TP inte står i början på raden ger:

0T texten från början av raden fram till TP.
T texten från TP till slutet på raden.
0TT hela raden

T-kommandot flyttar inte TP.

(-)U (Upper Case) Efter kommandot U kommer alla små bokstäver att översättas till stora. Översättningen återställs med -U.

(-)V (Verify) Efter kommandot -V kommer ED inte längre att skriva ut radnummer. Promptern blir då bara "*". Bara V återställer radnumren.

0V Ger utskriften:

xxxx/yyyy

där xxxx är det återstående utrymmet i arbetsarean och yyyy är arbetsareans totala storlek.

nW (Write) Skriver ut rader från arbetsarean till den nya filen (filnamn.$$$), för att ge mer plats i arbetsarean. Kommandot används bara för mycket stora filer.

0W skriver till arbetsarean är högst halvfull.

nX (Xfer) Skriver ut n rader från TP till filen X$$$$$$$.LIB. Detta kommando kan användas tillsammans med R för att flytta ett avsnitt i texten.

Om X$$$$$$$.LIB redan innehåller text läggs den nya texten efter den gamla. 0X tömmer filen X$$$$$$$.LIB.

nZ (Sleep) Ger en paus på ungefär n sekunder. Kan t ex användas i ett M-kommando så att du hinner se vad som händer.

6.4.7 Sammanslagning av kommandon

Det är tillåtet att skriva flera kommandon på samma rad. Kommandona skrivs då direkt efter varandra på raden. För kommandon som innehåller strängar måste det avslutande ^Z vara med för att skilja av strängen från nästa kommando.

Exempel:

```
#a-b-5t
```

Läser in hela filen till arbetsarean, flyttar TP till sista raden och skriver ut de 5 sista raderna.

Kommandona E, H, O och Q får inte skrivas ihop utan måste stå ensamna på en rad.

Möjligheten att skriva ihop kommandon är särskillt användbar ihop med M-kommandot.

Exempel:

```
mfnorsk^Z-5iny^Z0tt <ret>
```

Detta kommando letar reda på strängen "norsk", backar 5 tecken till början på ordet, lägger in strängen "ny" och skriver ut hela den ändrade raden. Kommandot upprepas till det når slutet på arbetsarean.

Tecknet ^L i en sträng tolkas som tidigare nämnts som <CR><LF> dvs ny rad.

Kommandot:

```
s^L^L ^Z
```

slår ihop två rader genom att byta <CR><LF>, som skiljer raderna, mot ett blanktecken.

Som exempel på vad man kan göra med ED ger vi slutligen ett kommando som sätter ihop alla avstavade ord i en text.

```
ms-^L^Z^Zs ^Z^L^Z
```

Kommandot letar reda på tecknet "-" följt av ny rad och tar bort båda två. Därefter byter det nästa blanktecken i texten mot ny rad så att raden inte skall bli för lång.

6.4.8 Felmeddelanden

ED har följande felmeddelanden:

? Felaktigt kommando, förstår ej.

> Arbetsarean full eller för lång sträng i ett kommando.

0 LIB-filen i R-kommandot finns inte.

Kommandot kan inte utföras önskat antal gånger.

Detta meddelande behöver inte tyda på något fel. Det är det normala meddelandet från ett kommando som upprepas

tills det når slutet på arbetsarean, t ex #S eller M utan antal.

6.5 ERA

Kommandot ERA tar bort en eller flera filer. Kommandot har formen:

    ERA ofn

där ofn är ett ofullständigt filnamn.

Exempel:

ERA FIL.COM	Tar bort filen FIL.COM från aktuella skivan.
ERA B:*.TMP	Tar bort alla filer med typ .TMP från skiva B:.

Ofullständiga filnamn bör användas med försiktighet eftersom det är lätt gjort att ta bort mer än man tänkt sig. Det säkraste är att bara ta bort en fil i taget.

6.6 LOAD

Kommandot LOAD används för att tillverka ett körbart program av en Intel-hex fil. Kommandot har formen:

    LOAD filnamn

LOAD läser filen filnamn.HEX och gör om den till den körbara filen filnamn.COM. I fortsättningen kan programmet köras genom att skriva:

    filnamn <ret>

Filen blir med andra ord ett nytt kommando.

Filen filnamn.HEX måste innehålla korrekta Intel-hex rader, t ex från ASM. Programmet måste börja i 100H och raderna i filen skall vara ordnade efter stigande adress. Eventuella luckor i filen fylls ut med noll. LOAD kan med andra ord bara användas för att tillverka en "vanlig" .COM-fil som skall köras som ett kommando.

Program som skall ligga i en annan del av minnet kan tillverkas med DDT (se 6.2).

6.7 MOVCPM

Kommandot MOVCPM lägger ett exemplar av systemet i minnet. MOVCPM används oftast när du vill tillverka ett nytt system efter att ha ändrat i BIOS. MOVCPM kan också relokera CP/M för an annan minnesstorlek. Eftersom UNI80 innehåller 64 kbytes RAM behöver denna möjlighet bara användas om du ökat storleken på BIOS.

Obs! Enligt "The CP/M Handbook" kan det relokerade systemet köras direkt. Detta går inte på ABC80, bl a därför att BIOS

är uppdelat i två delar (se bilaga F).

Kommandot har formen:

    MOVCPM n *

där n är den minnesstorlek i kbytes (16-64) som systemet skall relokeras till. Om n sätts till '*' kollar MOVCPM hur mycket minne som finns i maskinen. På ABC80 ger således '*' och '64' samma resultat.

Det relokerade systemet läggs i minnet på adress 800H. Du kan nu lägga till BIOS och BOOT med DDT och sedan spara det nya systemet med SYSGEN (se bilaga F).

6.8 PIP

Kommandot PIP används för kopiering och konvertering av filer. Det kan också användas för att kopiera till och från yttre enheter, t ex skrivaren. Vid konverteringen kan PIP även utföra andra funktioner, t ex numrering av rader och indelning i sidor.

6.8.1 Direkt och interaktiv körning

Kommandot kan användas antingen direkt, eller också interaktivt. Vid direkt användning har kommandot formen:

    PIP pip-kommando

Pip-kommandot utförs då, och du får CP/M-prompter på vanligt sätt. Vid interaktiv användning skriver du bara:

    PIP

och får en ny promter: "*" (asterisk). Du kan nu skriva ett pip-kommando följt av <ret>. När kommandot utförts får du åter promptern "*" och kan skriva ett nytt kommando. Du kommer tillbaka till CP/M genom att skicka en tom rad, dvs genom att bara tryck på <ret>.

Vid interaktiv användning kan du ta ut systemskivan när du fått den nya promptern. Du kan sedan sätta i en nya skiva i enhet A och kopiera från den till enhet B. Detta är användbart om du vill kopiera från en skiva som inte innehåller filen PIP.COM.

Observera att du inte kan byta skivan du kopierar till eftersom skivor som byts medan du kör ett program blir skrivskyddade (se bilaga B).

6.8.2 Kopiering och sammanslagning av filer

I det enklaste fallet används PIP för kopiering av filer. Kommandot har då formen:

    PIP nfil=gfil

eller

    *nfil=gfil   (vid interaktiv användning)

Detta kommando skapar en kopia av filen gfil och ger den namnet nfil. Om det redan finns en fil med namnet nfil kommer den att avlägsnas.

Kopieringen kan när som helst avbrytas genom att trycka på valfri tangent. PIP skriver då texten "ABORTED" på skärmen för att visa att överföringen inte avslutats.

PIP kan även användas för att slå ihop filer. Kommandot har då formen:

    nfil = gfil1,gfil2,...

(I fortsättningen skriver vi bara ut själva pip-kommandot.)

Den nya filen nfil kommer nu att innehålla hela gfill följt av hela gfil2 osv. Filerna som skall slås samman skall vara textfiler, dvs de skall avslutas med filslutstecknet ^Z (se bilaga B). Om du vill slå ihop icke-textfiler kan du använda parametern O (se nedan).

Vid kopiering till en fil med namnet nfil.typ skapas först en tillfällig fil med namnet nfil.$$$. När kopieringen är klar döps denna sedan om till nfil.typ. Om det redan finns en fil med namnet nfil.typ tas denna inte bort förrän nfil.$$$ skall döpas om. Detta gör att den gamla nfil.typ kan förekomma en eller flera gånger till höger om likhetstecknet.

O, se bilaga Bm filen nfil.typ är skrivskyddad (R/O) skrivs frågan:

```
DESTINATION FILE IS R/O. DELETE(Y/N)?
```

ut. Om du svarar med tecknet "Y" tas den gamla filen bort och kopieringen slutförs, annars kommer texten:

```
** NOT DELETED **
```

Exempel:

`X.ASM=Y.TXT`	Gör en kopia av filen Y.TXT och ge den namnet X.ASM.
`A:BOK.TXT= B:KAP1.TXT,B:KAP2.TXT`	Slår ihop textfilerna KAP1.TXT och KAP2.TXT på skiva B: till BOK.TXT på skiva A:.
`A:BOK.TXT=A:BOK.TXT,B:KAP3.TXT`	Slår ihop filerna A:BOK.TXT och B:KAP3.TXT. Låt resultatet ersätta filen A:BOK.TXT

6.8.3 Kopiering mellan skivor

Man har ofta behov av att kopiera en eller flera filer mellan två skivor utan att ändra namn på filerna. I dessa fall kan du skriva på följande sätt:

`e:=ofn`	Kopierar alla filer som beskrivs av ofn från aktuella skivan till enhet e:.
`e:=f:ofn`	Kopierar alla filer som beskrivs av ofn från enhet f: till enhet e:.
`filnamn=e:`	Kopierar filen filnamn från enhet e: till aktuella skivan.

Exempel:

B:=*.COM Kopierar alla kommandofiler (typ .COM) från aktuella skivan till skiva b:

C:=B:*.TXT Kopierar alla filer av typ .TXT från enhet B: till enhet C:

6.8.4 Kopiering till och från yttre enheter

PIP kan också användas för kopiering till och från yttre enheter. Du använder då fysiska eller logiska enhetsnamn istället för filnamn i pip-kommandona. Logiska och fysiska enheter finns närmare beskrivna i bilaga A. PIP accepterar följande logiska enhetsnamn:

CON:, RDR:, PUN: och LST:

och följande fysiska enhetsnamn:

TTY:, CON:, UC1:
PTR:, UR1:, UR2:
PTP:, UP1:, UP2:
LPT:, UL1:

(Observera att BAT: inte finns med eftersom den bara är en kombination av RDR: och LST:)

PRN: Skriver på LST: med expanderar tab-tecknen till var 8:e kolumn, numrerar raderna och ger ny sida efter var 60:e rad. (Samma sak som LST: med parametrar ÄT8P60NÅ)

INP:
OUT: Speciella in- och utrutiner som kan läggas in direkt i PIP med DDT. Detta är programmering i högre skolan och nämns här endast för fullständighetens skull.

Förutom dessa enheter finns det två alternativ som ser ut som enhetsnamn men som bara lägger in ett antal tecken vid överföringen:

NUL: Ger 40 st ASCII-null tecken. Används för att få en bit tom remsa vid stansning av hålremsa.

EOF: Lägger in CP/M:s filslutstecken ^Z. (Detta görs automatiskt vid kopiering av textfiler.)

Filnamn och enhetsnamn kan blandas fritt i pip-kommandon. Vid läsning från yttre enheter läser PIP tills ett filslutstecken (^Z) påträffas. Se dock även parametrarna O och Q nedan.

Vid kopiering från RDR: kan du om du så önskar ge filslutstecknet från tangentbordet istället. PIP kollar hela tiden tangentbordet och om du skriver ett ^Z kommer detta att

hanteras som ett normalt filslutstecken från RDR:.

Exempel:

LST:=PROG.ASM	Skriver ut filen PROG.ASM på enhet LST: (skrivaren).
TEST.TXT=CON:	Läser in text från tangentbordet och lagra dem i TEST.TXT. Texten skall avslutas med tecknet ^Z från tangentbordet.

6.8.5 Parametrar

PIP kan även ha en eller flera parametrar som anger hur data skall modifieras vid överföringen. Parametrarna skall stå mellan tecknen "Ä" och "Å" omedelbart efter namnet på den fil eller enhet de gäller. (Orsaken till att tecknen "Ä" och "Å" valts är att de svarar mot vänster och höger hakparentes på amerikanska maskiner.)

Följande parametrar finns:

B	Blockvis överföring. PIP läser in data och lagrar den i arbetsminnet ända till tecknet ^S (ASCII X-off) kommer. Då skrivs data ut på skivan. Detta kommando används om om du skall läsa in från en kassettbandspelare eller V24-porten. Du kommer annars att förlora de tecken som kommer in medan PIP är upptaget med att skriva på skivan.
Dn	tar bort alla tecken fr o m det n:te på varje rad, dvs tag bara med de (n-1) första tecknen på varje rad.
E	Skriver ut alla överföringar på bildskärmen samtidigt som de utförs.
F	Tar bort alla Form-Feed tecken (^L) ur filen. P-parametern kan användas samtidigt för att lägga in nya.
Gn	Hämtar filen från användararea n (se 6.16).
H	Överföring av Intel-hex fil. Alla överförda rader kommer att kontrolleras så att de verkligen är korrekta Intel-hex rader. (Detta görs även automatiskt vid kopiering från en yttre enhet till en fil av typ .HEX).
I	Som ovan dvs överföring av Intel-hex fil. Om I-parametern används kopieras dock inte rader som bara innehåller :00.
L	Översätter alla stora bokstäver till små bokstäver (gäller inte Å, Ä, Ö).
N	Numrerar de överförda raderna. Radnumret skrivs utan inledande nollor och följt av kolon.

N2 Som ovan men inledande nollor skrivs ut och numret följs av ett tab-tecken (^I). Tab-tecknet expanderas om T-parametern också används.

O Behandlar filen som en icke-textfil, dvs överför hela filen utan att ta hänsyn till eventuella filslutstecken (^Z). (Sätts automatiskt vid kopiering av .COM-filer.)

Pn Lägger in tecknet form-feed (^L) efter var n:te rad. Detta tecken ger ny sida på de flesta skrivare. Om n utelämnas blir n=60.

Qsss^Z Överför bara t o m första förekomsten av strängen "sss". Strängen kan innehålla ett godtyckligt antal tecken, och ^Z markerar slutet på strängen.

R Kopierar även filer med attributet $SYS. Normalt kopieras bara "vanliga" filer, dvs $DIR filer. (se bilaga B).

Ssss^Z Kopiera fr o m första förekomsten av strängen "sss".

Anm. Strängarna "sss" kommer att översättas till stora bokstäver om PIP används direkt. Om PIP däremot används interaktivt kan både stora och små bokstäver användas.

Tn Expanderar alla tab-tecken (^I) till var n:te kolumn. För normala CP/M-filer skall n vara 8.

U Översätter små bokstäver till stora (utom å, ä, ö).

V Kontrollerar (verifiera) att data blir riktiga genom att kontrolläsa efter varje skrivning. Fungerar bara vid överföring till en fil på skiva.

W Skriver över skrivskyddade filer (R/O-filer) utan att först fråga på skärmen.

Z Nollställer paritetsbiten på alla överförda tecken.

Exempel:

PROG.ASM=B:ÄVÅ Kopierar PROG.ASM från enhet B: till aktuella skivan, och kontrollerar att resultatet blir riktigt.

LPT:=DEL1.TXTÄNT8Å Kopierar DEL1.TXT till LPT:, Numrerar raderna och expanderar tabbarna till var 8:e kolumn.

A.HEX=B.HEXÄIÅ,C.HEXÄHÅ Kopierar B.HEX till A.HEX och tar bort den avslutande :00-raden. Fortsätter sedan med samtliga rader i C:HEX. Kontrollerar hela tiden att de överförda raderna är riktiga Intel-hex rader.

X.COM=RDR:ÄOQSLUT^ZÅ Kopierar från RDR: till X.COM.

Kopierar även eventuella filslutstecken (^Z), men avbryter kopieringen när strängen "SLUT" påträffas.

PRT:=TEST.TXTÄFP50Å Skriver ut filen TEST.TXT på skrivaren. Tar bort alla gamla form-feed tecken och lägger in nya efter var 50:e rad.

6.8.6 Kopiering av Intel-hex filer

Om du ger parametern H (eller I) eller om du kopierar från en yttre enhet till en fil av typ .HEX så kommer PIP att göra vissa kontroller. För varje överförd rad kommer PIP att kontrollera att det är en riktig Intel-hex rad, dvs att kontrollsumman är riktig. Om en rad är felaktig kommer PIP att skriva ett felmeddelande på skärmen och stanna. Tryck på <ret> för att fortsätta kopieringen. Den felaktiga raden kan sedan rättas med ED.

6.9 REN

REN ändrar namnet på en fil. Kommandot har formen:

```
REN nfilnamn=ofilnamn
```

Filen gfilnamn får då namnet nfilnamn.

Exempel:

```
REN GAMMAL.TXT=AKT.TXT
```

döper om filen AKT.TXT till GAMMAL.TXT.

6.10 SAVE

SAVE används för att spara innehållet i minnet på en fil. Det används i allmänhet bara om man ändrat direkt i ett program i minnet med DDT, och vill spara det ändrade programmet.

Kommandot har formen:

```
SAVE n filnamn
```

SAVE sparar innehållet i TPA, dvs innehållet i minnet med början i adress 100H, på filen filnamn. n är antalet 256-bytes sidor som skall sparas.

6.11 SD

SD (Super DIR) fungerar på samma sätt som DIR. Enda skillnaden är att filerna skrivs ut i bokstavsordning och att storleken på varje fil skrivs ut.

SD kommer från en samling med program som givits ut av "CP/M Users Group", och får därför kopieras fritt.

6.12 STAT

Kommandot STAT har flera helt skilda funktioner. Det kan ge information om filer och skivor, sätta filattribut, koppla ihop logiska och fysiska enheter mm. De olika funktionerna beskrivs separat nedan.

6.12.1 Utskrift av filbiblioteket på en skiva

Den vanligaste användningen av STAT är att skriva ut vilka filer som finns på en skiva. Kommandot har då formen:

```
STAT ofn
```

Kommandot ger följande utskrift:

```
Recs    Bytes    Ext Acc
48       6k       1 R/O A:ED.COM
58       8k       1 R/O (A:PIP.COM)
48       6k       1 R/W A:TAB.DAT

Bytes remaining: 23k
```

Här är Bytes filens storlek i bytes, Recs är antalet records och EXT är antalet extents. Av dessa siffror är normalt bara storleken intressant. Acc anger om filen är skrivskyddad (R/O) eller ej (R/W). (se 6.12.4).

Om ett filnamn står inom parentes, som A:PIP.COM ovan, är filen en systemfil, dvs den har attributet $SYS (se 6.12.4).

Siffran efter "Bytes remaining" anger hur mycket utrymme som finns kvar på skivan.

6.12.2 Kvarvarande utrymme på skivorna

Kommandot:

```
STAT
```

ger följande utskrift, med en rad för varje enhet som används sedan senaste omstart:

```
A: R/W, SPACE: xxK B: R/W, SPACE: xxK
```

Här anger xx hur mycket utrymme som finns kvar på respektive enhet. R/O anger att enheten är skrivskyddad och R/W anger att det går att både läsa och skriva på enheten. En enhet blir skrivskyddad om den sätts till R/O (se 6.12.4) eller om du bytt skivan utan att göra varmstart.

Om du bara vill veta hur mycket utrymme som finns kvar på en viss enhet skriver du istället:

```
STAT e:
```

där e är enhetens namn.

6.12.3 Att sätta filattribut

STAT kan även användas för att sätta vissa egenskaper (attribut) hos en fil. En fil kan ha två attribut, dels kan den vara skrivskyddad (R/O), dels kan den vara en system-fil ($SYS). En skrivskyddad fil kan varken ändras eller tas bort utan att först ta bort skrivskyddet. Kommandot:

```
STAT ofn $R/O
```

skrivskyddar en eller flera filer som beskrivs av det ofullständiga filnamnet ofn. Kommandot:

```
STAT ofn $R/W
```

tar bort skrivskyddet igen.

En fil med attributet $SYS kommer inte att synas när man gör DIR och går inte att nå med vanliga program. Däremot kan den köras som ett kommando (om det är en .COM-fil).

Kommandot:

```
STAT ofn $SYS
```

ger en eller flera filer attributet $SYS medan:

```
STAT ofn $DIR
```

tar bort det igen.

Filattributen finns utförligare beskrivna i bilaga B

6.12.4 Att skrivskydda en skiva

En skiva kan skrivskyddas med kommandot:

```
STAT e:=R/O
```

där e är skivans namn. Skivan förblir skrivskyddad tills nästa varmstart.

6.12.5 Hopkoppling av logiska och fysiska enheter

Kommandot STAT kan även användas för att koppla ihop logiska och fysiska enheter. Det har då formen:

```
STAT lenhet:=fenhet:
```

Följande kombinationer av logiska och fysiska enheter är tillåtna:

```
CON: = TTY:, CRT:, BAT: eller UC1:
RDR: = TTY:, PTR:, UR1: eller UR2:
PUN: = TTY:, PTP:, UP1: eller UP2:
LST: = TTY:, CRT:, LPT: eller UL1:
```

Kommandot:

```
STAT DEV:
```

visar hur de logiska enheterna är kopplade till de fysiska.

6.12.6 Utskrift av alla möjligheter i STAT

Kommandot:

```
STAT VAL:
```

skriver ut alla möjliga parametrar till STAT dvs:

```
Temp R/O Disk: d:=R/O
Set Indikator: d:filename.typ $R/O $R/W $SYS $DIR
Disk Status  :    DSK: d:DSK
User Status  :    USR:
Iobyte Assign:
CON: = TTY:, CRT:, BAT: or UC1:
RDR: = TTY:, PTR:, UR1: or UR2:
PUN: = TTY:, PTP:, UP1: or UP2:
LST: = TTY:, CRT:, LPT: or UL1:
```

6.12.7 Diverse specialfunktioner

STAT har ytterligare några speciella funktioner:

```
STAT e:DSK
```

skriver ut diverse uppgifter om enhet e. Om e: utelämnas skrivs uppgifterna ut för alla enheter.

```
STAT USR:
```

Skriver ut vilken användare som är aktiv och vilka användare som har filer lagrade.

```
STAT ofn $S
```

Skriver ut uppgifter om filerna inklusive hela storleken av en direktåtkomstfil.

För närmare upplysningar om dessa funktioner, se "The CP/M Handbook".

6.13 SUBMIT

6.13.1 Allmänt

SUBMIT används för att köra flera kommandon i en följd. Kommandona som skall utföras skall ligga i en textfil. Kommandot har formen:

```
SUBMIT kfil
```

där kfil är namnet på filen som innehåller de kommandon som skall utföras. Filen måste ha filtypen .SUB, och filtypen skall utelämnas när man skriver kommandot. Kommandofilen läggs lämpligen upp men ED (se 6.4).

SUBMIT skriver ut kommandona på skärmen allt eftersom de utförs. De kommandon som utförs skriver på bildskärmen och läser från tangentbordet precis som vanligt.

Obs! SUBMIT fungerar bara om aktuella skivan är A: (se även 6.12.4).

Exempel:

Antag att du har en serieskrivare som går i 300 baud ansluten till V24-kontakten, och att du vill använda den som utskriftsenhet. Du kan då skapa filen SERIE.SUB med följande innehåll:

```
SET BAUD=300
STAT LST:=UL1:
```

När du nu i fortsättningen vill sätta serieskrivaren som utskriftsenhet skriver du bara:

```
SUBMIT SERIE
```

Man kan även "koppla ihop" kommandofiler genom att ha ett nytt SUBMIT-kommando sist i kommandofilen.

Körningen av kommandofilen kan när som helst avbrytas genom att trycka på ^H.

6.13.2 Parametrar

Submit kan även användas med parametrar till kommandofilen. Kommandot har då formen:

```
SUBMIT kfil par1 par2 ....
```

där par1, par2 ... är parametrar.

Om kommandofilen innehåller någon av teckensekvenserna:

```
$1, $2,...
```

så byts de ut mot motsvarande parametrar så att $1 byts mot par1, $2 mot par2 osv.

Om du vill ha ett $-tecken i kommandofilen måste du istället skriva $$.

Exempel:

Filen MKCMD.SUB innehåller följande:

```
ASM $1.AAZ
LOAD $1
ERA $1.HEX
```

Om du nu skriver:

```
SUBMIT MKCMD TEST
```

kommer följande kommandon att utföras:

```
ASM TEST.AAZ
LOAD TEST
ERA TEST.HEX
```

dvs A:TEST.TXT kommer att assembleras, och den producerade intel-hex filen kommer att laddas till ett kommando. Slutligen kommer intel-hex filen att tas bort.

6.13.3 XSUB

När SUBMIT används kommer alla kommandon att läsa från tangentbordet precis som vanligt. Om du lägger kommandot XSUB i kommandofilen kommer även dessa att läsa från filen.

Kommandot XSUB kan bara användas inne i en SUBMIT-fil. Det måste dessutom stå först i filen.

Exempel:

Filen MODIG.SUB innehåller följande:

```
XSUB
FORMAT
E
J
J
```

Du kan nu formattera skivan i enhet B i enkel densitet genom att skriva:

```
SUBMIT MODIG
```

6.14 SYSGEN

6.14.1 Allmänt

När du startar upp CP/M (kallstart) eller gör varmstart (se 2.2) läses själva systemet in från enhet A:. Det måste därför alltid finnas ett system på alla skivor som skall kunna sitta i enhet A: när man startar upp.

SYSGEN används för att lägga ett exemplar av själva systemet, eller rättare sagt kontrollprogrammet, på en skiva. Eftersom systemet ligger på en reserverad plats på skivan kan det även läggas på en skiva som redan innehåller filer, utan att filerna på skivan skadas.

6.14.2 Normal kopiering av systemet

Om du vill göra en normal kopia av systemet gör du på följande sätt (dina svar i fetstil):

1. Sätt en systemskiva i enhet A: och skivan som systemet skall kopieras till i enhet B:

2. A> **SYSGEN <ret>**
   SYSGEN VERSION m.n
   SOURCE DRIVE NAME (OR RETURN TO SKIP) **A**
   SOURCE ON A THEN TYPE RETURN **<ret>**
   FUNCTION COMPLETE
   DESTINATION DRIVE NAME (OR RETURN TO REBOOT) **B**
   DESTINATION ON B THEN TYPE RETURN **<ret>**
   FUNCTION COMPLETE
   DESTINATION DRIVE NAME (OR RETURN TO REBOOT) **<ret>**
   A>

Skivan i B: innehåller nu ett exemplar av systemet.

Observera att SYSGEN bara kopierar själva systemet. På den nya skivan är därför bara de inbyggda kommandona tillgängliga. De övriga kommandona kan kopieras med:

```
PIP B:=A:*.COM
```

6.14.3 Andra möjligheter hos SYSGEN

Som framgår av exemplet ovan arbetar SYSGEN i två steg. Först läses systemet in från en skiva, och sedan skrivs det ut på en annan. Man kan även använda SYSGEN för att bara läsa in systemet i minnet eller för att skriva ut ett system som redan finns i minnet. Systemet kan också skrivas ut flera gånger.

Vid inläsning hamnar systemet i adress 900H i minnet. Det kan sedan modifieras med t ex DDT innan det skrivs ut igen. För närmare upplysningar om hur man ändrar i systemet och hur man genererar nya system, se bilaga E.

Vi skall nu titta närmare på exemplet ovan:

A>SYSGEN <ret>

Starta SYSGEN.

SYSGEN VERSION m.n

SOURCE DRIVE NAME (OR RETURN TO SKIP)

Svara med namnet på den enhet som systemet skall läsas in från. Om systemet redan finns i minnet, svara bara **<ret>**. Om du svarar **x <ret>** kommer frågan:

SOURCE ON x THEN RETURN

Sätt en skiva som innehåller ett system i enhet x (om där inte redan finns en). Skriv **<ret>** när du är klar.

FUNCTION COMPLETE

Denna utskrift kommer när systemet lästs in till minnet.

DESTINATION DRIVE NAME (OR RETURN TO REBOOT)

Om systemet skall läggas på en skiva svarar du med namnet på enheten. Om du däremot vill avbryta, t ex för att ändra i systemet, svarar du bara **<ret>**. Om du svarar **x <ret>** svarar SYSGEN med:

DESTINATION ON x THEN TYPE RETURN

Sätt skivan som systemet skall läggas på i enhet x, skriv sedan **<ret>**.

FUNCTION COMPLETE

Denna utskrift kommer när systemet ligger på skivan.

Frågan "DESTINATION ..." kommer nu att upprepas. Du kan då antingen avbryta genom att svara **<ret>** eller också fortsätta och lägga systemet på fler skivor.

Observera att det SYSGEN som följer med systemet **inte** är det som följer med standard CP/M utan en specialversion för ABC80.

6.15 TYPE

TYPE skriver ut innehållet i en fil på skärmen. Kommandot har formen:

```
TYPE filnamn
```

TYPE expanderar eventuella tabbar till var 8:e position.

Om man först trycker ^P (se 2.6) kommer filen även att skrivas ut på skrivaren.

6.16 USER

Beskrivningen till USER är ej färdigskriven, se tills vidare "The CP/M Handbook".

Bilaga A: Logiska och fysiska enheter

CP/M är avsett att kunna användas ihop med olika hårdvara. På ABC80 kan du t ex ansluta skrivare både till V24-kontakten och via ett anpassninskort på ABC-bussen. Ett program som körs under CP/M skall inte vara beroende av hårdvaran. För att lösa detta detta har man i CP/M infört logiska och fysiska enheter. Ett program som använder skrivaren skriver på den **logiska** enheten LST:, som är utskriftsenheten. LST: är sedan kopplad till en av de **fysiska** enheterna LPT: (parallellskrivaren) eller UL1: (serieskrivaren). Den kan tom kopplas ihop med TTY: (bildskärmen) om du vill ha utskriften på bildskärmen istället för på skrivaren.

Det finns fyra logiska enheter:

CON:	Systemets huvudterminal. Det är på denna enhet som CP/M skriver ut promtern och läser in kommandon. Den används också som in/ut-enhet av de flesta program.
RDR:	En extra inenhet. (Används sällan.)
PUN:	En extra utenhet. (Används sällan.)
LST:	Utskriftsenheten. Används av de program som skriver på skrivaren. Används också av CP/M när du tryckt ^P.

Det finns 11 fysiska enheter. Deras betydelse är olika på olika datorsystem. På ABC80 gäller följande:

TTY:	Bildskärmen
CRT:	Bildskärmen
UC1:	Ej definierad
PTR:	Läser från V24-porten med ställbar baudrate
UR1:	Ej definierad
UR2:	Ej definierad
PTP:	Ej definierad
UP1:	Ej definierad
UP2:	Ej definierad
LPT:	Skrivare ansluten via ABC-bussen.
UL1:	Skrivare ansluten till V24-porten. Ställbar baudrate.

Dessutom finns "enheten" BAT:. BAT: är ingen egen enhet utan en kombination av LST: och RDR:. Om du sätter CON:=BAT: så kommer CP/M att läsa från det enhet som är definierad som RDR: och skriva på den enhet som är definierad som LST:. Detta kan användas om du vill köra från en yttre terminal ansluten till V24-porten (se 3.5).

De logiska enheterna kopplas ihop med de fysiska med kommandot STAT lenhet=fenhet (se 6.12.5). När du startar upp CP/M är samtliga logiska enheter kopplade till TTY:

Bilaga B: Flexskivor och filer.

Under CP/M lagras data som filer på flexskiva. Skivorna är uppdelade i sektorer om 128 eller 256 bytes. En fil består av en eller flera sektorer. Information om vilka sektorer som ingår i en fil finns i en särskild area på skivan, filkatalogen (eng directory). I filkatalogen står också filens namn och eventuella filattribut.

B.1 Skrivskydd av flexskivor

Normalt kan man både läsa och skriva på flexskivorna. Med CP/M terminologi säger vi att skivorna är R/W (efter engelskans Read/Write). Alla enheter blir R/W efter varje varmstart. En skiva kan även vara skrivskyddad, R/O efter engelskans Read Only. På en R/O-skiva kan man läsa men inte skriva.

Om du försöker skriva på en skrivskyddad enhet kommer felmeddelandet:

```
BDOS ERR ON d: READ ONLY
```

CP/M väntar sedan på att användaren skall trycka på <ret>. Då görs automatiskt varmstart och alla enheter blir R/W.

Om du byter en flexskiva utan att göra varmstart kommer skivan att bli skrivskyddad för att minska risken för att oavsiktligt skriva på skivan. När du byter någon av skivorna bör man därför alltid göra varmstart genom att trycka ^C.

Man kan även skrivskydda en enhet med kommandot STAT.

Allt som ovan sagts om flexskivor gäller även enhet C: "RAM-skivan".

B.2 Filattribut

Även enskilda filer kan skrivskyddas. En skrivskyddad fil kan inte ändras eller avlägsnas utan att ta bort skrivskyddet.

En fil kan ha ytterligare ett attribut (egenskap), nämligen $SYS. En fil med attributet $SYS kan inte nås av vanliga kommandon och program, och syns inte när man gör DIR. Däremot kan den fortfarande köras som ett kommando. Avsikten är att man skall kunna sätta kommandona till $SYS så att de blir dolda vid vanligt arbete. Observera att man måste använda parametern R om man vill kopiera sådana filer med PIP.

Skrivskyddet och attributet $SYS sätts och avlägsnas med kommandot STAT (se 6.12).

B.3 Filslutstecken

En fil består alltid av ett helt antal sektorer. För filer som innehåller körbara program eller data gör detta ingen-

ting. För filer som innehåller text, t ex källprogram, vill man däremot veta exakt var texten slutar. Detta löses genom att CP/M lägger ett särskilt tecken, ^Z, efter sista tecknet i en textfil. ^Z kallas då för filslutstecken.

^Z används även som filslutstecken av PIP och ED, vid läsning från yttre enheter och vid inmatning av text direkt från tangentbordet.

Bilaga C: Terminalrutinen

I BIOS för ABC80 finns en drivrutin som hanterar bildskärmen. När man skriver på TTY: eller CON: ser rutinen till att bildskärmen uppför sig som en vanlig terminal, men den har också en hel del andra möjligheter. Genom att skriva speciella tecken och teckensekvenser kan man flytta cursorn, lägga till och ta bort tecken och rader, skriva med blinkande (eller inverterad) text, tända och släcka cursorn mm. Bildskärmen uppför sig med andra ord som en intelligent terminal.

Kommandona till bildskärmen är valda så att de i tillämpliga delar överensstämmer med dem på terminalerna ADM3A och Televideo 91C. Många färdiga CP/M-program (t ex Wordstar) kan därför direkt utnyttja de möjligheter som terminalrutinen ger.

I tabellen nedan finns alla kontrollsekvenser och kontrolltecken. Deras värden är angivna både decimalt och hexadecimalt.

Tecken	Hex	Dec	Funktion
RETURN	0D	13	Flyttar cursorn till 1:a positionen på nästa rad.
LF	0A	10	Flyttar cursorn en rad ner.
BS	08	8	Backar cursorn en position.
VT	0B	11	Flyttar cursorn en rad uppåt.
SUB	1A	26	Tömmer skärmen.
RS	1E	30	Cursor Home, dvs flyttar cursorn till övre vänstra hörnet på skärmen.
FF	0C	12	Flyttar cursorn ett steg framåt.
BELL	97	7	Ger pip i högtalaren.

Kontrollsekvenser (ESC = 1B hex = 27 dec):

ESC, '*'	Tömmer skärmen (samma som SUB).
ESC, 'Y'	EOS, raderar resten av skärmen fr o m cursorns nuvarande läge.
ESC, 'T'	EOL, raderar resten av raden.
ESC, ')'	Startar blinkande text (eller inverterad text, se bilaga x).
ESC, '('	Avslutar på blinkande text.
ESC, 'E'	Insert line, lägger in en tom rad där cursorn står och flyttar resten en rad neråt.
ESC, 'Q'	Insert character, lägger in ett blanktecken där cursorn står och flyttar resten av raden ett steg åt höger.
ESC, 'R'	Delete line, tar bort raden som cursorn står på. Resten av skärmen flyttas upp en rad.
ESC, 'W'	Delete char, tar bort tecknet där cursorn står.

ESC, '=', r, k	Flyttar cursorn till rad r, kolumn k där r och k ges enligt nedan.
ESC, '.', r, k	Tänder en grafisk punkt i position r, k. (Motsvarar SETDOT i ABC-BASIC).
ESC, ' ', r, k	Släcker en grafisk punkt i position r, k. (Motsvarar CLRDOT).

I de tre sista alternativen enligt ovan skall rad- och kolumnadresserna ges med offset 32 (20 hexadecimalt). Detta betyder att ett blanktecken (' ')svarar mot första raden resp kolumnen, '!' mot andra osv.

Bilaga D: Kopiering till större skivor.

Om du har en flexskiveenhet med dubbel densitet och/eller en dubbelsidig enhet bör du kopiera över dina skivor. Detta görs på följande sätt:

1) Formattera en (8") eller två (5 1/4") skivor i önskat format med FORMAT (se 5.1).

2) Sätt enhet A i enkel densitet, enkelsidigt och enhet B i önskat format.

3) Sätt systemskivan i enhet A och en av de tomma skivorna i enhet B.

4) Kör följade kommandon:

```
A> SYSGEN <ret>
SYSGEN VERSION m.n
SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A THEN TYPE RETURN <ret>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B THEN TYPE RETURN <ret>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <ret>
A> PIP B:=A:*.* <ret>
A>
```

5) Skivan i enhet B är nu din nya systemskiva.

Om du har två orginalskivor skall du dessutom göra följande:

6) A> PIP <ret>
   *

   Ta nu ur systemskivan ur enhet A och sätt i källtextskivan.

```
* B:=A:*.* <ret>
```

   När kopieringen är klar, sätt i systemskivan i enhet A igen.

```
* <ret>
A>
```

Du kan nu sätta enhet A i det nya formatet, sätta i den nya systemskivan och köra.

Bilaga E: MOVCPM och ändringar av systemet.

Om man vill göra ett speciellt CP/M-system kan man ibland behöva flytta CP/M till andra adresser. För att möjliggöra detta finns ett program som heter MOVCPM med på systemskivan. När man kör MOVCPM kan man tala om önskad minnesstorlek. Eftersom BIOS är lite speciellt genom att det är uppdelat på 2 olika delar som ligger i olika minnesbankar fungerar MOVCPM inte exakt som på andra CP/M-system. Direkt körning av det genererade systemet går inte. Däremot kan MOVCPM användas för att relokera om CCP och BDOS. Sedan använder man den normala proceduren och lägger till sitt eget BIOS på samma sätt som görs i submit-filen SYSTEM. Obs att man är tvungen att ha nya offset-värden när man läser in de olika delarna med DDT.

Om man vill göra någon mindre patch av systemet kan man göra en kopia av systemet på följande sätt:

```
A> SYSGEN
SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A, THEN TYPE RETURN <ret>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <ret>
A> SAVE 40 CPM64.COM
A>
```

SYSGEN läser bootspåren från skivan och lägger i minnet med början i adress 0900h. CCP startar i 0980h och BIOS i 1F80h. Tänk på att koden ska köras i andra adresser om du tittar på hopp och dylikt.

Nu kan man använda DDT för att patcha systemet på önskat sätt. Gör så här:

```
A> DDT CPM64.COM
DDT VERS 2.2
NEXT  PC
2900 0000
-
(div ddt-kommandon) -G0
```

Nu finns i minnet en patchad version av systemet. Kör nu SYSGEN utan att köra något emellan (som skulle förstöra systemet i minnet).

```
A> SYSGEN
SOURCE DRIVE NAME (OR RETURN TO SKIP) <ret>
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B, THEN TYPE RETURN <ret>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <ret>
A>
```

Nu finns det nya systemet på skiva B:. Byt skivor och tryck reset för att läsa in det nya systemet. Efter att du har kontrollerat att det fungerar, kan du kopiera det nya systemet med SYSGEN på normalt sätt.

Bilaga F: Anpassning av systemet för olika skivor och skrivare

CP/M på ABC80 kan användas ihop med ett flertal olika skivor. Det kan dessutom användas ihop med ett antal olika skrivare.

Du kan se vilken typ av skivor resp skrivare som ditt system är anpassat för genom att titta på uppstartsmeddelandet. Efter versionsnumret på BIOS står där en eller två bokstäver, där den första bokstaven talar om vilken skivtyp systemet är avsett för och den andra talar om vilken skrivar-rutin som är installerad.

Observera att det förutom den angivna skrivarrutinen dessutom finns en drivrutin för V24-porten med i systemet.

Som standard levereras CP/M på enkelsidiga skivor i enkel densitet utan drivrutin för skrivare.

För att göra det lättare för dig att göra om systemet finns det en färdig kommandofil, SYS80, som gör större delen av arbetet.

Flexskiveenheten betecknas med följande bokstäver:

A 5 1/4" skivor med 40 spår.

B 5 1/4", 77-spår Micropolis från Hobby Data.

C 5 1/4", 80-spår, Luxor 832.

D 8" enkelsidigt, Datadisk 86.

D 8" dubbelsidig, Datadisk 88.

Anpassningskorten till skrivaren betecknas med följande bokstäver:

A Luxor's standard parallell-kort.

B Metric's gamla parallellkort.

C Metric's nya parallelkort.

D Sattco's UART-kort 4017.

E KL-elektronik's parallellkort.

Om du har en flexskiva eller ett anpassningskort som inte finns med i ovanstående lista kan du kontakta MYAB.

För att tillverka ett nytt CP/M-system gör du på följande sätt:

1) Kopiera över orginalskivorna till två nya skivor, helst