UNIVERSELL EPROM PROGRAMMERARE

> EP <

48, 64, 128
48H, 64H, 128H

ABC80/ABC800/DTC

Ver 3.4

1983-01-22

Utveckling och konstruktion - POLYTECH HB, Lund.
Anpassning till ABC800 och
Försäljning - MIKRODIDAKT HB, Lund.

INNEHÅLLSFÖRTECKNING

1. INLEDNING

EPROM-programmerarna EP är avsedda att anslutas till ABC 80, ABC 800 eller DTC. Anslutningen sker antingen till diskenhetens busskontakt eller till separat expansionslåda. Tillhörande programvara levereras på flexskiva eller kassett (ABC80). EPROM-programmerarna kan användas för programmering av följande:

EPROM-typer:	EP-version:
8748 Enchip mikrodator med 1 K EPROM	Samtliga
2508 5V 1 K EPROM	Samtliga
2758 5V 1 K EPROM	Samtliga
2516 5V 2 K EPROM	Samtliga
2716 5V 2 K EPROM	Samtliga
2532 5V 4 K EPROM	Samtliga
2732 5V 4 K EPROM	Samtliga
2732A 5V 4 K EPROM	Samtliga
2564 5V 8 K EPROM	EP64,EP64H,EP128,EP128H
2764 5V 8 K EPROM	EP64,EP64H,EP128,EP128H
27128 5V 16 K EPROM	EP128,EP128H
8749H Enchips mikrodator med 2 K EPROM	EP48H,EP64H,EP128H

27856 ? 32K

2. TEKNISK BESKRIVNING

På kortet finns tre socklar i vilka EPROMen placeras. Den största sockeln användes vid programmering av 8748 och 8749H. Mellansockeln användes till typerna 2564, 2764 och 27128. Till de övriga typerna användes den minsta sockeln. Vidare finns i kortets nedre vänstra hörn tre rader med byglingsstift. Med hjälp av dessa anpassas kortet till den aktuella EPROM-typen. Med den medföljande byglingskontakten skall stiften ihopkopplas parvis i antingen rad A,B eller C enligt tabell nedan:

EPROMTYP	BYGLING
8748	A
2508 / 2758	A
2516 / 2716	A
2764	A
27128	A
8749H	A
2532	B
2564	B
2732 / 2732A	C

Med den 6-poliga DIP-omkopplaren kan kortets adress ändras. Den övre switchen (#1) i omkopplaren är MSB, den undre switchen (#6) är LSB. Programmeraren samt programmet EP levereras med adress 32 (dec) inställd.

OBSERVERA ATT VID VARJE ÄNDRING AV DIP-OMKOPPLAREN MÅSTE ÄVEN VARIABELVÄRDET PÅ RAD 10 I EPV34.BAC ÄNDRAS TILL MOTSVARANDE VÄRDE.

En lysdiod markerar att kortet är "on-line" med datorn.

De för programmeringen nödvändiga spänningarna genereras internt på kortet.

3. HANDHAVANDE

Anslut kortet till ABC-bussen. Detta kan ske antingen till en av diskenhetens busskontakter eller till ett expansionschassi. på I/O sidan.

Starta programmet med kommandot 'RUN EP'. Installera byglingskontakten på den stiftrad som gäller för aktuellt EPROM.
Ange EPROM-typ.
Om byglingskontakten sitter fel anges detta på skärmen. EPROMet får sättas i och tas ur närsomhelst utom då *-markerade kommandon exekveras (se nedan).

EP använder en buffertarea i RAM för editering. Buffertarean logiska adress i kommandon nedan har inget samband med dess fysiska adress i datorn.
Vidare bestäms buffertareans storlek av tillgängligt RAM-utrymme i ABC 80. Programmet EP tar vid uppstart själv reda på aktuellt RAM-utrymme samt skriver ut fysisk start adress och storlek på tillgänglig buffertarea.

EP initieras i samband med uppstart av programmet vilket innebär att initieringen förloras om kortet av någon anledning blir spänningslöst. Omstart måste då ske.

Startadr i O
end adr 16k -1FF ?

L ta sedan ut master

stoppa i nytt prom V

P

CO

4. KOMMANDON

(Adresser i nedanstående kommandon anges alltid hexadecimalt. Inledande nollor behöver ej anges).

Byglingskontaktens placering kontrolleras automatiskt före varje kommando-exekvering.

OBS Då *-markerade kommandon exekveras får EPROMet ej sättas i eller tas ur sockeln. **OBS**

O OFFSET BUFFER

Format: Oxxxx (default: x=0)

Medför att buffertarean börjar på adress xxxx.

Ex: Kommandot O1000 medför att buffertarean börjar på adress 1000. (Den fysiska placeringen av bufferten i datorns RAM-minne ändras naturligtvis inte.)

Buffertens logiska adress används vid filhantering i INTEL-HEX format eller ABS format då dessa filbeskrivningar innehåller information om adressen för koden. Se kommandon nedan.

D DISPLAY BUFFER

Format: Dxxxx yyyy
Dxxxx (default: y=bufferend)
D (default: x=offset
y=bufferend)

Listar innehållet mellan adresserna xxxx och yyyy. Listningen kan tillfälligt stoppas genom att trycka <space>. Listningen kan återstartas genom att trycka på valfri tangent. Genom att trycka E' avbrytes listningen.

Ex: Kommandot D120 170 listar innehållet i buffertarean 120-170 på skärmen.

D/X DISPLAY BUFFER (16 ords format)

Format: D/Xxxxx yyyy

Samma som D-kommandot ovan, med den skillnaden att data presenteras med 16 ord per rad.
Användbart vid utskrift på printer.

E ENTER DATA INTO BUFFER

Format: Exxxx
E (default: x=offset)

Innehållet på angiven adress skrivs ut samt nytt innehåll kan därefter skrivas in till höger om markören följt av <RETURN>. Observera att flera värden kan skrivas in på samma rad. Kommandot lämnas genom att skriva E'.

Ex: Kommandot E23 ger på skärmen:
0023 46>13 FE 45 56 ED <RETURN>

M MOVE BUFFER

Format: Mxxxx yyyy zzzz

Flyttar buffertarean xxxx-yyyy till buffertarea med början i zzzz.

Ex: Kommandot M0 100 200 flyttar buffert innehållet i 0-100 till 200-300. Innehållet i 0-100 påverkas ej.

F FILL BUFFER

Format: Fxxxx yyyy zz
Fxxxx yyyy (default: z=0)

Fyller angiven buffertarea (xxxx-yyyy) med angivet innehåll (zz).

Ex: Kommandot F0 120 99 fyller buffertarean 0-120 med 99.

CH CHECKSUM IN BUFFER

Format: CHxxxx yyyy
CHxxxx (default: y=bufferend)
CH (default: x=offset
y=bufferend)

Beräknar check-summan på innehållet i bufferten mellan adresserna xxxx och yyyy.

Ex: Kommandot CH0 500 beräknar checksumman i buffertarean 0-500.

* V VERIFY IF EPROM ERASED

Format: V

Kontrollerar om EPROMet är raderat och skriver ut check-summan.

* T TYPE EPROM

Format: Txxxx yyyy
Txxxx (default: y=EPROM-end)
T (default: x=0,y=EPROM-end)

Listar innehållet i EPROMet mellan adresserna xxxx och yyyy. Listningen kan tillfälligt stoppas genom att trycka <space>. Listningen kan återstartas genom att trycka på valfri tangent. Genom att trycka E' avbrytes listningen.

* T/X TYPE EPROM (16 ords format)

Format: T/Xxxxx yyyy
T/Xxxxx (default: y=EPROM-end)
T/X (default: x=0,y=EPROM-end)

Samma som T-kommandot ovan, med den skillnaden att data presenteras med 16 ord per rad. Användbart vid utskrift på printer.

* L LOAD BUFFER FROM EPROM

Format: L

Kommandot frågar efter start- och slutadress i EPROM samt startadress i buffert.

överför angiven area av EPROMet till angiven area i bufferten.

* CO COMPARE BUFFER / EPROM

Format: CO

Kommandot frågar efter start- och slutadress i buffert samt startadress i EPROM.

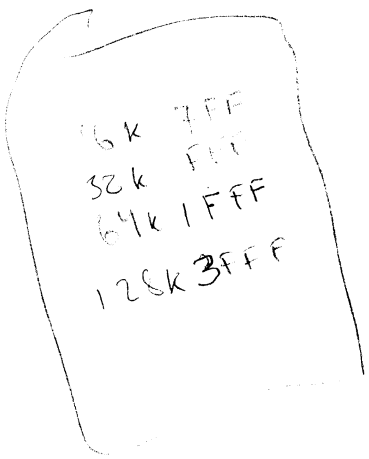
Jämför minnesinnehållen samt skriver ut en tabell med de adresser (och data) där innehållen skiljer sig åt.

* P PROGRAM

Format: P

Kommandot frågar efter start- och slutadress i buffert samt startadress i EPROM.

Programmerar EPROMet med innehållet i angiven buffertarea samt skriver ut checksumman.


16k 7FF
32k FFF
64k 1FFF
128k 3FFF

Genom att trycka <space> visas vilken adress som programmeras. Programmeringen avbrytes genom att trycka E'.

En kort ton hörs i högtalaren då programmeringen är klar.

Om EPROMet ej är raderat testas om det är möjligt att programmera i alla fall, om inte erhålles 'ILLEGAL BIT PATTERN', och frågas: "PROGRAM ANYWAY (Y/N) ?".
Genom att trycka 'Y' påbörjas programmeringen.

Om EPROMet ej är programmerbart avbrytes programmeringen.

S SAVE BUFFER ON FILE

Format: S

Kommandot frågar efter start- och slutadress i buffert samt filnamn.

Lagrar innehållet i angiven buffertarea på fil med angivet namn.

Filformat kompatibelt med **ASMEDIT**-programmen från Mikrodidakt.

S/I SAVE BUFFER ON FILE (Intel hex format)

Format: S/I

Samma som S-kommandot men lagring på fil i Intel Hex Code format.

S/A SAVE BUFFER ON FILE (ABS format)

Format: S/A

Samma som S-kommandot med lagring på fil i ABS format.

G GET FILE TO BUFFER

Format: G

Kommandot frågar efter startadress i buffert samt filnamn.

Laddar in angiven fil till bufferten med början på angiven adress.

Filens slutadress i bufferten skrivs ut.

G/I **GET FILE TO BUFFER (Intel hex format)**

Format: G/I

Laddar in angiven fil till bufferten enligt Intel Hex Code format.

Antalet bytes som lagrats i bufferten skrivs ut.

G/A **GET FILE TO BUFFER (ABS format)**

Format: G/A

Laddar in angiven fil till bufferten enligt ABS format.

Antalet bytes som lagrats i bufferten skrivs ut.

Q **QUIT TO SYSTEM OR CHANGE EPROM-TYPE**

Format: Q

Ger möjlighet att återvända antingen till operativsystemet eller till huvudmenyn där ny EPROM-typ kan väljas.

H **HELP**

Format: H

Skriver ut kommando-menyn för den valda EPROM-typen.

<ctrl-P> **PRINTER ON/OFF**

Kopierar på printer det som skrives på skärmen.

5. FELUTSKRIFTER

Felutskrift	Orsak
WHAT?	Ogiltigt inmatningsformat från tangentbord.
NOT A HEX DIGIT	Ej hexadecimalt värde.
INVALID NO. OF ARGUMENTS	Fel antal argument.
INVALID BUFFER ADDRESS	Angiven adress överstiger övre buffertgräns.
INVALID EPROM ADDRESS	Angiven adress överstiger övre EPROM-gräns.
INVALID BLOCK SIZE	Angiven adressarea för stor.
BUFFER END	Då man med E-kommandot når buffertgränsen.
BUFER FULL	Inläst fil får ej plats i buferten. (Gäller för filer i ASMEDIT-format)
NO SUCH FILE	Angiven fil finns ej.
FILE READ ERROR	Inläsningsfel från fil.
INVALID FILE FORMAT	Fel filformat.
CHECKSUM ERROR	Inläsningsfel. (Gäller Intel Hex- och ABS-filer)
ERROR xx	Fel vid exekvering av SAVE-kommandot. xx anger feltyp enligt dator manual.

6. TEKNISK SPECIFIKATION

Kortdimension:	300 x 100 mm
Bussanslutning:	Till ABC-bussen eller 4680-bussen
Kortkontakt:	64-pol Europadon DIN 41612
Strömförbrukning:	+5V 250 mA max. +12V 150 mA max.
Intern spänning:	DC/DC-omvandlare 12/28 V
Kortadress:	6-pol DIP-omkopplare ger 64 möjliga adresser (0-63). Dessutom måste ändring ske på rad 10 i EPV34.BAC för korrekt adressering.

7. FILFORMAT

ABC's ABS-format

Lagras i block där varje block inledes med en blockbeskrivninge enligt:

DATA block

Byte #	Funktion
1	00 = blockstart - FF = läs nästa sektor.
2	antal byte - om 00 = ENDblock se nedan.
3	00 i normalt block.
4	mest signifikanta adressdel där data skall lÄggas.
5	4 inverterat.
6	minst signifikant adressdel.
7	6 inverterat.
8 - ?	data. obs ett block sträcker sig inte över sektorn. Max antal byte 0F3H.
-	checksumma beräknat på data.

Här följer nästa blockbeskrivning eller eventuellt FF enligt ovan varvid nästa sektor läses in.

ENDblock

Byte #	Funktion
1	Blockstart.
2	00 - antal byte.
3	00
4	Programstartadress mest signifikanta delen.
5	4 inverterat.
6	Programstartadress minst signifikanta delen.
7	6 inverterat.

INTEL-HEX format

Intels hexadecimala format är ett format för kodning av data i ASCII-tecken. Två typer av block är tillåtna, dvs DATA RECORD och END RECORD.

Data record:

Record start	ASCII kolon (:)
Record längd	Två hexadecimala siffror anger längden.
Laddningsadress	Fyra hexadecimala siffror anger startadress för blocket.
Record typ	Två hexadecimala siffror anger typ. Data record = 00
Data	2 till 64 hexadecimala siffror (32 data bytes).
Checksumma	Två hexadecimala siffror anger

	checksumman på blocket beräknat fr o m Record längd t o m Data. Checksumman utgör tvåkomplementet modulo 256. Dvs summan inklusive Checksumman skall vara noll.

End Record:

Record start	ASCII kolon (:).
Record längd	00
Program start	Fyra hexadecimala siffror anger eventuell programstart.
Record typ	01, anger End record.
Checksumma	Beräknad f o m Record längd.

APPENDIX A gäller endast ABC80

PROMNING AV BASIC-PROGRAM

BASIC-program för ABC 80 kan lagras i EPROM för direkt åtkomst efter uppstart. Koden kan antingen lagras i ASCII-format (.BAS-fil) eller i ABC 80:s interna kod (.BAC-fil). Båda metoderna kommer att beskrivas efter en inledande jämförelse.

ASCII-format

Programmet ligger i EPROM, men måste läsas in till RAM-minnet med hjälp av en speciell I/O-rutin för att kunna köras. Detta innebär att programmet belägger dubbel minnesarea.

Internkod-format

Programmet exekveras direkt i EPROM och upptar ingen del av RAM-minnet förutom variabelvärdena. Innan lagring i EPROM göres, måste programmet vara kompilerat på den adress där det senare skall exekveras, vilket innebär att det är nödvändigt att ha RAM-minne där det senare skall finnas EPROM-minne. En enkel lösning på detta är att man utvecklar programmet i en ABC 80 med 32 K RAM för att senare använda en ABC 80 med 16 K RAM.

PROMNING AV BASIC-PROGRAM I ASCII-FORMAT

1. Spara programmet på fil med LIST-kommandot.

2. Kör programmet ASCPROM.
   ASCPROM omformatterar BASIC-programmet till ett format som är avpassat till EP64.

3. Kör programmet IOPROM.
   Besvara frågan om BASIC-format med 'A'.
   IOPROM genererar en fil innehållande en I/O-rutin som behövs för att läsa in EPROM-innehållet till RAM. Då IOPROM köres bestämmer man EPROM-areans startadress.

4. Kör EP.
   Läs in filen som genererats av IOPROM och därefter filen som genererats av ASCPROM. Den senare skall placeras omedelbart efter den tidigare.

5. Programmera EPROM.

6. För att köra programmet ges först BASIC-kommandot:
   ;CALL(X)
   där X=PROM-areans startadress. Detta anrop gör att 'EP:' definieras som en enhet, och måste göras efter varje <reset> av ABC 80.
   Programmet körs sedan med BASIC-kommandot 'RUN EP:'.

PROMNING AV BASIC-PROGRAM I INTERNKODSFORMAT

1. Bestäm adresser för EPROM-area och variabelarea i RAM. Vi kallar dessa PROMADR resp. VARADR (t.ex. PROMADR=32768 och VARADR=49152)

2. Placera RAM-minne på adressen PROMADR.

3. Kör programmet IOPROM.
   Besvara frågan om BASIC-format med 'I' och ange aktuella PROMADR- och VARADRvärden.
   IOPROM genererar en fil innehållande den I/O-rutin som utnyttjas för att starta upp BASIC-programmet från EPROM.
   Innan IOPROM avslutas sätts BOFA automatiskt till lämplig adress. Denna adress skrivs ut på skärmen. Anteckna denna adress!

4. Ladda in BASIC-programmet.
   För att få reda på storleken på programmet skall EOFA avläsas genom kommandot:
   ;PEEK(65054)+256*(PEEK(65055))
   Anteckna!

5. Nu skall variabelarean läggas på adress VARADR (se punkt 1).
   Gör:
   POKE 65054,VARADR,SWAP%(VARADR),VARADR,SWAP%(VARADR)

6. Programmet skall därefter kompileras:
   RUN
   och bryt med <ctrl-C> då programmet startat.

7. Gör POKE 65052,49152,SWAP%(49152)
   och därefter RUN EPFILER.

8. Ange de värden på BOFA och EOFA som skrivits ut under punkt 3 och 4, då programmet frågar efter start- resp. slutadress.

9. Tryck <reset>.
   Kör EP.
   Läs in filen som genererats av IOPROM och därefter filen som genererats av EPFILER. Den senare skall placeras omedelbart efter den tidigare.

10. Programmera EPROM.

11. Sätt EPROM på adressen PROMADR. (OBS. Inget RAM-minne får finnas på denna adress).

12. För att köra programmet ges först BASIC-kommandot
    ;CALL(PROMADR).
    Detta anrop gör att enheten 'EP:' infogas i enhetslistan. Anropet måste göras efter varje <reset> av ABC 80.
    Programmet startas med 'RUN EP:'.

APPENDIX B gäller endast ABC80

AUTOSTART

Efter <reset> testar BASIC-tolken om DOS-PROM finns i systemet. DOS-PROM anses finnas om adress 604BH innehåller en JP-instruktion. Finns DOS-PROM sker anrop av adress 604BH. Detta kan utnyttjas för autostart i en ABC 80 utan flexskiva, genom att man infogar följande maskinspråksrutin på adress 6000H.

```
1         ;**********************
2         ;
3         ;       autostart
4         ;
5         ;**********************
6
7
8         ;COMMAND=BASICTOLKENS KOMMANDOAVKODARE
9         COMMAND:     EQU        00F4H
10
11                     ORG   6000H
12
13        START:       CALL  INITEP   ;LÄNKA IN DEVICE
14                     LD    (IY+0EH),01H
15                     XOR   A
16                     LD    (0FE07H),A
17                     LD    HL,(0FE27H)
18                     LD    SP,HL
19                     EI
20                     LD    HL,CMDLINE
21                     JP    COMMAND
22
23        CMDLINE:     DEFM  'RUN EP:'
24                     DEFB  0DH
25
26                     ORG   0604BH
27                     JP    START
29
```

APPENDIX C

FILöVERSIKT - ABC80

Filer för promning av BASIC:

- IOPROM	Skapar device-rutiner för EPROM (kallat EP:)
- ASCPROM	Användes vid promning av BASIC-kod i LIST-format. Omvandlar BASIC listfil till ASMEDIT-format.
- EPFILER	Användes vid promning av BASIC-kod i internkod-format. Sparar minnesarean i ASMEDIT-format.
- IOPR1.SR	Device-rutiner för EPROM med LIST-kod.
- IOPR2.SR	Device-rutiner för EPROM med internkod.

övriga filer:

- EP.BAC
- EPREAD.BAC
- EPV34.OBJ
- EPV34.REL
- RELLOAD.OBJ
- EPV34.BAC

FILöVERSIKT - ABC800

- EP.BAC
- EPV34.BAC
- EPV34.COD

IOPR1.SR

```
                    0000           ORG    0000H
                    0001           MEM    0FF80H
                    0002 ;        IOPR1
                    0003 ;        =====
                    0004 ;
                    0005 ;I/O-RUTINER FöR BASICPROM MED
                    0006 ;ASCII-KOD.
                    0007 ;COPYRIGHT MIKRODIDAKT HB
                    0008 ;
                    0009 ;
                    000A ;CF 810929
                    000B ;
                    000C DEVLIST:EQU    0FE0AH       ;START DEVICE LISTA
                    000D
                    000E ;INITEP
                    000F ;======
                    0010 ;LÄNKAR IN ENHETEN 'EP' I ABC80:S
                    0011 ;ENHETSLISTA.
                    0012 ;BESKRIVNINGEN LÄGGES PÅ LEDIG
                    0013 ;PLATS I BILDMINNET.
0000  2A0AFE        0014 INITEP: LD     HL,(DEVLIST)
0003  22F87F        0015         LD     (7FF8H),HL
0006  3E45          0016         LD     A,45H        ;'E'
0008  32FA7F        0017         LD     (7FFAH),A
000B  3E50          0018         LD     A,50H        ;'P'
000D  32FB7F        0019         LD     (7FFBH),A
0010  3E20          001A         LD     A,20H        ;' '
0012  32FC7F        001B         LD     (7FFCH),A
0015  212200        001C         LD     HL,JPTABLE
0018  22FD7F        001D         LD     (7FFDH),HL
001B  21F87F        001E         LD     HL,7FF8H
001E  220AFE        001F         LD     (DEVLIST),HL
0021  C9            0020         RET
                    0021
                    0022
                    0023 ;I/O-RUTINER FöR ENHET 'EP'
0022  C32E00        0024 JPTABLE:JP     OPEN
0025  C32E00        0025         JP     OPEN
0028  C35600        0026         JP     CLOSE
002B  C33900        0027         JP     INPUT
002E  215800        0028 OPEN:   LD     HL,BASBAS    ;PEKAR BASIC I ASCII-KOD
0031  DD750A        0029         LD     (IX+0AH),L   ;AKTUELL BUFFERT-ADR
0034  DD740B        002A         LD     (IX+0BH),H
0037  A7            002B         AND    A            ;RESET CARRY
0038  C9            002C         RET
                    002D
0039  DD5E0A        002E INPUT:  LD     E,(IX+0AH)   ;DE PEKAR I BUFFERT
003C  DD560B        002F         LD     D,(IX+0BH)
003F  1A            0030 INPLOOP:LD     A,(DE)       ;HÄMTA BYTE FRÅN PROM
0040  FE03          0031         CP     03H          ;END OF FILE?
0042  280F          0032         JR     Z,EOF
0044  77            0033         LD     (HL),A       ;TILL BASICS BUFFERT
0045  23            0034         INC    HL
0046  13            0035         INC    DE
0047  FE0D          0036         CP     0DH          ;VAR DET CR?
0049  20F4          0037         JR     NZ,INPLOOP
004B  DD730A        0038         LD     (IX+0AH),E   ;SPARA NY BUFFERTPEKARE
```

```
004E  DD720B    0039           LD     (IX+0BH),D
0051  A7        003A           AND    A
0052  C9        003B           RET
0053  AF        003C EOF:      XOR    A              ;END OF FILE: A=0
0054  37        003D           SCF                   ;OCH CARRY=1
0055  C9        003E           RET
                003F
0056  A7        0041 CLOSE:    AND    A
0057  C9        0042           RET
                0043 BASBAS:   ;HÄR BöRJAR BASIC-KOD
```

IOPR2.TXT

```
                    0000           ORG    0000H
                    0001           MEM    0FF80H
                    0002 ;        IOPR2
                    0003 ;        =====
                    0004 ;
                    0005 ;I/O-RUTINER FöR BASICPROM MED
                    0006 ;ABC80:S INTERNKOD
                    0007 ;COPYRIGHT MIKRODIDAKT HB
                    0008 ;
                    0009 ;
                    000A ;CF 810929
                    000B ;
                    000C DEVLIST:EQU    0FE0AH       ;START DEVICE LISTA
                    000D BOFA:   EQU    0FE1CH
                    000E EOFA:   EQU    0FE1EH
                    0010 VARADR: EQU    0000H        ;SÄTTES AV IOPROM.BAC
                    0011
                    0012 ;INITEP
                    0013 ;======
                    0014 ;LÄNKAR IN ENHETEN 'EP' I ABC80:S
                    0015 ;ENHETSLISTA.
                    0016 ;BESKRIVNINGEN LÄGGES PÅ LEDIG
                    0017 ;PLATS I BILDMINNET.
0000  2A0AFE        0018 INITEP: LD     HL,(DEVLIST)
0003  22F87F        0019         LD     (7FF8H),HL
0006  3E45          001A         LD     A,45H        ;'E'
0008  32FA7F        001B         LD     (7FFAH),A
000B  3E50          001C         LD     A,50H        ;'P'
000D  32FB7F        001D         LD     (7FFBH),A
0010  3E20          001E         LD     A,20H        ;' '
0012  32FC7F        001F         LD     (7FFCH),A
0015  212200        0020         LD     HL,JPTABLE
0018  22FD7F        0021         LD     (7FFDH),HL
001B  21F87F        0022         LD     HL,7FF8H
001E  220AFE        0023         LD     (DEVLIST),HL
0021  C9            0024         RET
                    0025
                    0026
                    0027 ;I/O-RUTINER FöR ENHET 'EP'
0022  C32E00        0028 JPTABLE:JP     OPEN
0025  C32E00        0029         JP     OPEN
0028  C34200        002A         JP     CLOSE
002B  C33600        002B         JP     INPUT
002E  214400        002C OPEN:   LD     HL,BASBAC    ;HL=ADRESS TILL BASIC-KOD
0031  221CFE        002D         LD     (BOFA),HL
0034  A7            002E         AND    A
0035  C9            002F         RET
                    0030
0036  210000        0031 INPUT:  LD     HL,VARADR    ;HL=RAM:S STARTADR
0039  221EFE        0032         LD     (EOFA),HL
003C  2220FE        0033         LD     (HEAP),HL
003F  AF            0034         XOR    A
0040  37            0035         SCF
0041  C9            0036         RET                 ;RET MED END OF FILE
                    0037
0042  A7            0038 CLOSE:  AND    A
0043  C9            0039         RET
                    003A BASBAC: ;HÄR BöRJAR BASIC-KOD
```

APPENDIX D

PROMNING AV MASKINKOD

Filer producerade av ASMEDIT kan direkt läsas in till bufferten. Filer med samma format kan även skapas med hjälp av programmet EPFILER.
Detta kan användas för att komma åt information belägen på absoluta adresser i ABC 80.