

PC/M datormuseum  
inventarienummer:



GRAF 806

manual

# INNEHÅLL

1	INLEDNIG.....	1
2	APPLIKATION AV EPROM.....	2
3	BASIC-PROGRAMMERING.....	3
3.1	Nya BASIC-instruktioner.....	3
3.1.1	BASIC-instruktioner i EPROM.....	3
3.1.2	BASIC-instruktioner i RAM.....	4
3.2	Programexempel i BASIC.....	5
3.2.1	FGCTL MIX.....	5
3.2.2	FGDRAW och FGRUB.....	5
3.2.3	FLASH och NO FLASH.....	6
3.2.4	FGSPEC.....	6
3.2.5	LDIR.....	6
3.2.6	LDDR.....	7
3.2.7	FLASH, MIXSTATUS och MIXLINE.....	7
3.3	Tillägg till FGDRAW.....	7
4	ASSEMBLER-PROGRAMMERING.....	8
4.1	Hopptabeller i EPROM.....	8
4.2	Programexempel i assembler.....	13
4.2.1	Rita figur.....	13
4.2.2	Grafikscroll.....	14
5	SYSTEMVARIABLER.....	16
6	GRAFIK.....	18
6.1	Bildminne.....	18
6.1.1	Minneskonfiguration.....	18
6.1.2	Pixel och färgnummer.....	19
6.2	Färgval.....	19
7	BLANDADE PROGRAMEXEMPEL.....	20
7.1	Grafik.....	20
7.1.1	Animering.....	20
7.1.2	Oberoende bilder.....	21
7.1.3	Blanda 256- och 512-mod med varandra.....	22
7.2	Hur man gör extended BASIC m h a PROM'et.....	22
7.2.1	Kompilering av inparametrar.....	22
7.2.2	Parameterläsning och programkörning.....	23
8	BILAGOR.....	24
8.1	Systemprogram.....	24
8.1.1	EXTBASIC.BAC.....	24
8.1.2	DRAWMODU.BAS.....	24
8.1.3	DEMO.BAC.....	24
8.2	Timeout på PR: och V24:.....	25
8.2.1	Förklaringar.....	25
8.2.2	Programexempel med timeout.....	25
8.3	Referenslitteratur.....	26

## 1. INLEDNING

- Snabbdata EPROM: \*
- \* Innehåller alla funktioner som fanns tidigare.
  - \* FGLINE är 25%-48% snabbare i utritningsloopen.
  - \* FGPAINT är <=70% snabbare (beror på ytans typ).
  - \* FGFILL är 44% snabbare (endast vid 256-mod och pixelmasken <> 00001111B).
  - \* Möjlighet att blanda färger.
  - \* 9 nya instruktioner i BASIC:
    - FGCTL MIX
    - FLASH
    - NO FLASH
    - FGDRAW
    - FGRUB
    - FGSPEC
    - FGCLR
    - PROCEDURE
    - DO
  - \* 40 assemblerentries i hopptabell.
  - \* Kompletterande systemprogram med bl a fler BASIC-instruktioner.
    - LDIR
    - LDDR
    - MIXSTATUS
    - MIXLINE
  - \* Har timeout på PR: och V24:

För att uppnå förståelse och enighet mellan användare, som refererar till denna skrifts programlistor och exempel, följer alla listningar och exempel en standard som redovisas nedan.

**INTEGER-mode**      Används konsekvent i alla exempel. Förekommer flyttalsvariabler noteras dessa med suffixet punkt (.).

Ex: 10 A.=Flyt.(8)+13.45

**EXTEND-mode**      Används konsekvent i alla exempel.

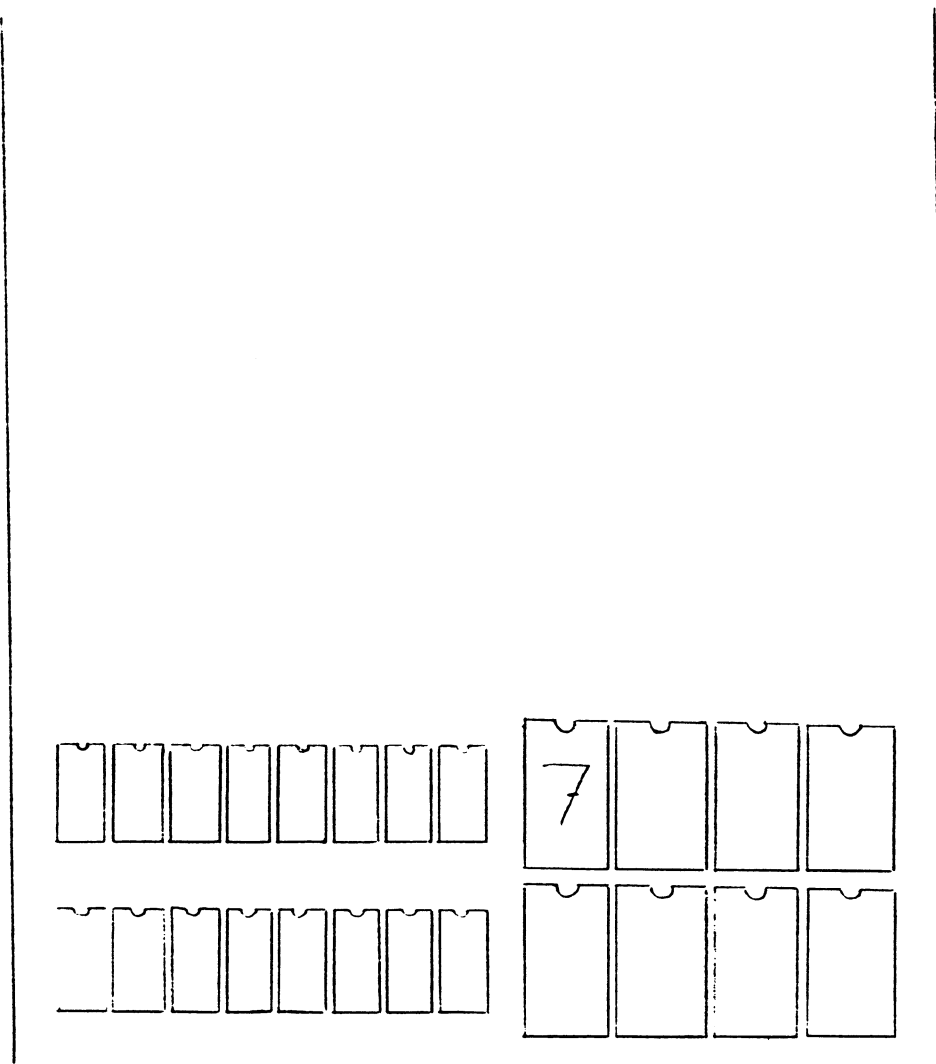
**OPTION BASE 0**      Används konsekvent i alla exempel.

**SINGLE**              Används konsekvent i alla exempel.

## 2. APPLIKATION AV PROM

OBS! Luxors ettåriga garanti på själva datorn påverkas ej av bytet (om man inte har sönder något).

1. Se till att strömmen är avslagen.
2. Tag loss sladdarna till datorn.
3. Skruva loss skruvarna till kåpan på datordelen.
4. Lyft av kåpan.
5. Byt PROM nr 7 mot det nya PROM'et, se figur 1 (var försiktig med benen på PROM'et).
6. Lyft tillbaka kåpan, skruva ihop och sätt tillbaka sladdarna.
7. Sätt på datorn, sätt in disketten med demoprogram, tryck på reset och kör DEMO.BAC.



Figur 1.

### 3. BASIC-PROGRAMMERING

#### 3.1 Nya BASIC-instruktioner

##### 3.1.1 BASIC-instruktioner i EPROM

###### FGCTL MIX

Format FGCTL MIX färgvalskommando

Där <färgvalskommando> är en sträng.

Funktion Blandar färger två och två.

Exempel **10 FGCTL MIX BLK+BLK+RED+YEL+BLK+BLU+GRN+CYA**

###### FGDRAW

Format FGDRAW xindex,yindex,data

Där <data> är en sträng uppbyggd på följande vis:

Byte 1: ffffkkkx (binärt)

ffff=färgnummer (0-15)

kkk=Kommando (0-7)

0=Point

1=Line

2=Fill

3=Paint

4-7=Optioner

x=Bit 8 i x-koordinaten

Byte 2: Resten av x-koordinaten

Byte 3: y-koordinaten

Funktion Placerar en sprite där xindex och yindex anger.

###### FGRUB

Format FGRUB xindex,yindex,data

Funktion Fungerar som FGDRAW men alla färger sätts automatiskt till 0.

###### FLASH

Format FLASH fv1,fv2 <,fv3 <,fv4,fv5,fv6>>

Där färgvalen fungerar som i FGCTL MIX.

Funktion Växlar mellan två, tre eller sex färgval, vilket får effekten att färger blinkar. Se 3.2.7 för animering.

###### NO FLASH

Format NO FLASH

Funktion Stänger av FLASH.

## FGSPEC

Format FGSPEC färgmask, pixelmask, xindex, yindex, data

Färgmask "ANDas" med systemvariabeln FÄRG. Pixelmask bestämmer vilka bit'ar i pixeln som skall påverkas. Bit 3 i pixelmasken sätts alltid till 1. Xindex, yindex och data fungerar som i FGDRAW.

Funktion Ger programmeraren möjlighet att:

- \* Rita en sprite med samma data i valbar färg genom att variera färgmasken. Detta förutsätter att alla instruktioners färger är satta till 15.
- \* Rita på en bild utan att förstöra den genom att använda en pixelmask av t ex typen 00001000B.

## FGCLR

Format FGCLR

Funktion Tömmer grafikminnet snabbt och behändigt.

## PROCEDURE

Format PROCEDURE argument

Där argumentet kan vara ett heltal eller en sträng. BASIC-tolken förstår även förkortningen PROC.

Funktion Fungerar som avancerad slaskvariabel.

Exempel

```
10 PROCEDURE FNDummy
20 PROCEDURE FNNils
30 PROCEDURE CALL(Stupid,Nothing)
```

## DO

Format DO argument

Funktion Samma som PROCEDURE.

### 3.1.2 BASIC-instruktioner i RAM

Instruktionerna som följer laddas in i RAM m h a EXTBASIC.BAC.

## LDIR

Format LDIR från,till,antal

Där motsvarigheten i assembler är: HL=från  
DE=till  
BC=antal

Funktion Flyttar data (oftast i samband med grafikbildminnet). Adresspekarna räknas upp.

Format LDDR från,till,antal

Funktion Fungerar som LDIR men adresspekarna räknas ned.

## MIXSTATUS

Format MIXSTATUS längd, färgnr1, färgnr2, färgnr3

<längd> anger längden på varje liten linje.  
<färgnr1> färg som ritas först.  
<färgnr2> " " " sedan.  
<färgnr3> " " " sist.

1171611

Funktion Initierar status för MIXLINE.

## MIXLINE

Format MIXLINE x,y

Funktion Drar en linje från senast angiven punkt till x,y.  
Linjen är ritad i tre färger som anges i MIXSTATUS.

## 3.2 Programexempel i BASIC

### 3.2.1 FGCTL MIX

```
100 FGCTL MIX BLK+BLK+GRN+GRN+YEL+GRN+YEL+YEL
110 !
120 FGCLR
130 CHRS(12);
140 !
150 FOR I=1 TO 3
160 FGPOINT I*20,0,I
170 FGFILL I*20+19,239
180 NEXT I
190 !
200 END
```

3  
0110  
01116

### 3.2.2 FGDRAW och FGRUB

```
100 FGCTL BLK+RED+GRN+YEL+BLU
110 !
120 FGCLR
130 CHRS(12);
140 !
150 ! Initierar en triangel med en röd,grön och gul sida
160 ! som är fylld med blått.
170 !
180 Figur$=CHRS(16,0,0,16+2,30,30,32+2,30,0,48+2,0,0,64+6,20,5)
190 !
200 FOR I=1 TO 5
210 FGDRAW I*40,40,Figur$ ! Ritar triangel
220 NEXT I
230 !
240 FOR I=1 TO 2000 : NEXT I ! Pause 2 sek
250 !
260 FOR I=1 TO 5
270 FGRUB I*40,40,Figur$ ! Suddar triangel
280 NEXT I
290 !
300 END
```

30

1 RED 1 GRN 1 YEL 1 BLU

### 3.2.3 FLASH och NO FLASH

```
100 FGCTL MIX BLK ! Initiera till 256*240 mod
110 !
120 FGCLR
130 ; CHR$(12);
140 !
150 FLASH BLK+BLK+MAG+MAG+YEL+YEL, BLK+BLK+MAG+WHT+YEL+YEL
160 !
170 FOR I=1 TO 5000
180   X=INT(RND*250.)
190   Y=INT(RND*230.)
200   F=INT(RND*2.)+1
210   FGPOINT X,Y,F
220   FGFILL X+4,Y+4
230 NEXT I
240 !
250 FOR I.=1. TO 2000. : NEXT I. ! Pause
260 !
270 NO FLASH ! Stänger av blinket
280 END
```

### 3.2.4 FGSPEC

```
100 FGCTL BLK+RED+GRN+BLK+BLK+BLK+BLK+BLK+WHT+WHT+GRN
110 !
120 FGCLR
130 ; CHR$(12);
140 !
150 ! Initierar en rektangel för FGDRAW
160 Figur$=CHR$(240,0,0,242,0,20,242,20,20,242,20,0,242,0,0)
170 Figur2$=CHR$(240,0,0,244,20,20)
180 !
190 ! Slumpar ut 100 röda rektanglar
200 FOR I=1 TO 100
210   FGSPEC 17,15,RND*200,RND*200,Figur$
220 NEXT I
230 !
240 ! Slumpar ut 100 gröna rektanglar
250 FOR I=1 TO 100
260   FGSPEC 34,15,RND*200,RND*200,Figur$
270 NEXT I
280 !
290 ! Flyttar en vit rektangel från vänster till höger utan
300 ! att skada de röda och gröna rektanglarna.
310 ! Den vita rektangeln rör sig framför de röda rektanglarna
320 ! men bakom de gröna.
330 !
340 FOR I=2 TO 200 STEP 2
350   FGSPEC 0,8,I-2,120,Figur2$ ! Sudda gammal rektangel
360   FGSPEC 255,8,I,120,Figur2$ ! Rita ny rektangel
370   FOR I.=1. TO 100. : NEXT I. ! Pause
380 NEXT I
390 !
400 END
```

### 3.2.5 LDIR

```
100 A$='Hej på dej'
110 B$='
120 LDIR VARPTR(A$),VARPTR(B$),LEN(A$)
130 ; B$
140 END
```



### 3.2.6 LDDR

```
100 ; CHR$(12)
110 !
120 A$=' --->
130 A=VARPTR(A$)+LEN(A$)-1 ! A=Adressen till sista tecknet
140 !
150 FOR I=1 TO LEN(A$)
160   LDDR A-1,A,LEN(A$)-1
170   ; CUR(10,10)A$
180 NEXT I
190 !
200 END
```

### 3.2.7 FLASH, MIXSTATUS och MIXLINE

```
100 FGCTL MIX BLK ! Initiera grafiken till 256*240 mod
110 !
120 FGCLR
130 ; CHR$(12);
140 !
150 F1$=BLK+BLK+BLU+BLU+GRN+GRN+GRN+GRN
160 F2$=BLK+BLK+GRN+GRN+BLU+BLU+GRN+GRN
170 F3$=BLK+BLK+GRN+GRN+GRN+GRN+BLU+BLU
180 FLASH F1$,F2$,F3$
190 !
200 MIXSTATUS 2,1,2,3 ! Längd 2
210 FGPOINT 40,40
220 MIXLINE 40,120
230 MIXLINE 120,120
240 MIXLINE 120,40
250 MIXLINE 40,40
260 !
270 FOR I.=1. TO 5000. : NEXT I. ! Pause 5 sek
280 !
290 FLASH F3$,F2$,F1$,F3$,F2$,F1$ ! Öka hastigheten
300 !
310 FOR I.=1. TO 5000. : NEXT I. ! Pause 5 sek
320 !
330 NO FLASH ! Stäng av FLASH
340 !
350 END
```

### 3.3 Tillägg till FGDRAW

När EXTBASIC.BAC är inladdat, så är antalet användbara kommandon i FGDRAW 6 st.

0=Point  
1=Line  
2=Fill  
3=Paint  
4=Mixstatus  
5=Mixline  
6-7=Optioner

Mixstatus består av 5 bytes: byte 1: kommando  
" 2: längd  
" 3: färgnr 1  
" 4: " 2  
" 5: " 3

Mixline fungerar precis som de andra, men färgen spelar ingen roll eftersom det är angivet med mixstatus.

#### 4. ASSEMBLER-PROGRAMMERING

##### 4.1 Hopptabeller i EPROM

Dec	Hex	Namn	
28672	7000	INIPROM	Initierar optionsprommet.
28675	7003	FGPOINT	Placerar en punkt på skärmen.  In: BC = x HL = 239-y  Förstör AF, BC och HL.
28678	7006	FGLINE	Drar en linje mellan två punkter.  In: BC = x1 HL = 239-y1 BC' = x2 HL' = 239-y2  Förstör alla register utom DE, IX och IY.
28681	7009	FGFILL	Fyller en rektangel mellan två punkter.  In: BC = x1 HL = 239-y1 BC' = x2 HL' = 239-y2  Förstör alla register utom DE, IX och IY.
28684	700C	FGPAINT	Fyller ett begränsat område.  In: BC = x HL = 239-y  Förstör alla register utom DE, IX och IY.
28687	700F	DRAWRUB	Ritar en sprite.  In: DE = Adress till dataarean HL = Längden på dataarean  FEE7 = FFH om DRAW FEE7 = 00H om RUB FED7 = x-index FED9 = y-index  Förstör alla register utom IY.
28690	7012	GETPOINT	Läser en punkt på skärmen.  In: BC = x HL = 239-y  Ut: A = Färgnummer  Förstör AF, BC, DE och HL.

Dec	Hex	Namn	
28693	7015	TÖMBILD	Laddar färg 0 i hela bildminnet. Förstör inga register.
28696	7018	BILDSYNC	Väntar på synk-signal från video. Förstör inga register.
28699	701B	NYFÄRG	Väljer färger (som FGCTL).  In: B = Första färgnummer (oftast 0) HL = Startadress för färgvalsdata (≤16 bytes)  Förstör AF, BC och HL.
28702	701E	COMPNRML	Kompilerar instruktioner av typ "FGPOINT".  In: DE = Svares-buffert HL = Source-buffert  Ut: DE = Framräknad svarsbuffertpekare HL = Framräknad buffertpekare  Förstör allt utom IY.
28705	7021	COMPDRAW	Kompilerar instruktioner av typ "FGDRAW".  (Se COMPNRML för in och utparametrar)
28708	7024	GETXY	Hämtar koordinater, färger och bit-masker till BASIC-instruktioner av typ "FGPOINT".  In: DE = BASIC-programpekare  Ut: BC = x1 DE = Framräknad programpekare HL = 239-y1 BC' = x2 HL' = 239-y2  FEF5 = Färgnummer FEF6 = Pixelmask FEFC = x FEFE = 239-y  Förstör alla register utom IY.
28711	7027	HRADDRESS	Beräknar adress i bildminnet.  In: BC = x HL = 239-y  Ut: A = Pixelmask E = Bildsort 0=240, 1=512 och 2=256 HL = Adress i bildminnet  Förstör AF, BC, E och HL.
28714	702A	DOPAINT3	Speciellt inhop i FGPAINT.

Dec	Hex	Namn	
28717	702D	SAVEXY	<p>Sparar x- och y-koordinaterna.</p> <p>In: BC = x HL = 239-y</p> <p>Ut: BC' = Gammalt x-värde HL' = Gammalt y-värde</p> <p>Förstör BC' och HL'</p>
28720	7030	PARFIX1	<p>Omvandlar ett tecken till rätt paritet.</p> <p>In: E = Tecknet A = 0=Space, 1=Mark, 2=Odd, 3=Even</p> <p>Ut: A = Omvandlat tecken</p> <p>Förstör E och flaggor.</p>
28723	7033	BYTEPNKT	<p>XOR (HL) AND C XOR (HL) LD (HL),A RET</p>
28726	7036	FORMERA	<p>Omvandlar en sträng med färgvals-tecken till hårdvarukompatibel kod (ej packad).</p> <p>In: C = Antal tecken i strängen DE = Destination HL = Source</p> <p>Förstör AF, BC och HL</p>
28729	7039	READ	<p>Läser en sträng och omvandlar till packad hårdvarukompatibel färgvalskod, samt att färgvalet utförs.</p> <p>In: DE = Programpekare i BASIC HL = Adressen till en 16 bytes lång buffert + 15.</p> <p>Ut: DE = Framräknad programpekare HL = HL-10H F = Z om ", " finns efter behandlad parameter.</p> <p>Förstör alla register utom IY.</p>
28732	703C	MAXTEST	<p>Genererar error 201 om hela bildminnet är RAM-disk. Förstör AF.</p>
28735	703F	SUB19	Terminalrutin
28738	7042	SUB20	Terminalrutin
28741	7045	SUB21	Terminalrutin
28744	7048	SUB22	Terminalrutin
28747	704B	SUB23	Terminalrutin

Dec	Hex	Namn	
28750	704E	COMPTWO	Kompilerar två heltalsvariabler. (Se COMPNRML för in- och utparametrar)
28753	7051	COMPONE	Kompilerar en heltalsvariabel.  In: F = Testvärden, carry=1 ger retur och no zero ger carry=1 och A=227. DE = Svarsbuffert HL = Buffert  Ut: A = Ev errormeddelande. F = Carry=1 om error och no zero om ", " saknas efter inparametern. DE = Framräknad svarsbuffertpekare. HL = Framräknad buffertpekare.  Förstör alla register utom IY.
28756	7054	COMPFLSH	Kompilerar två strängar. (Se COMPNRML för in- och utparametrar)
28759	7057	COMPSONE	Kompilerar en sträng. (Se COMPONE för in- och utparametrar)
28765	705D	PUNKTADR	Testar grafik-koordinater och sparar dem.  In: BC = x HL = y  Ut: BC = x HL = 239-y BC' = Gammalt x-värde. HL' = Gammalt y-värde (239-y).  Förstör alla register utom IY.
28768	7060	PARFIX2	Omvandlar ett tecken med paritet till normal 7 bitars ASCII-kod.  In: A = 0=Space, 1=Mark, 2=Odd och 3=Even E = Tecknet  Ut: A = Tecknet i 7 bitars ASCII-kod. Om A=7FH var tecknet felaktigt.  Förstör E och flaggorna.
32692	7FB4	HRSAVE	Sparar en ruta i arbetsminnet från bildminnet.  In: A = Höjd BC = Antal bytes i X-led DE = Destination i arbetsminnet HL = Adress till rutans övre vänstra hörn  Förstör AF, BC, DE och HL.

Dec	Hex	Namn	
32705	7FC1	HRLOAD	Laddar en ruta från arbetsminnet i bildminnet.  In: A = Höjd BC = Antal bytes i X-led DE = Adress till rutans övre vänstra hörn HL = Source i arbetsminnet  Förstör AF, BC, DE och HL.
32720	7FD0	XFILL	Gör Byte XOR Färg.  In: B = Antal bytes C = Färg
32727	7FD7	EXOR	XOR (HL) RET
32729	7FD9	GETMEM	LD A, (HL) RET
32731	7FDB	PUTMEM	LD (HL), A RET
32733	7FDD	LOADDR	LDDR RET
32765	7FFD	LOADIR	LDIR RET

## 4.2 Programexempel i assembler

### 4.2.1 Rita figur

Detta programexempel visar hur man använder sig av de vanliga grafikrutinerna i assembler.

```
EX1      ZPROG Rita figur
;
;          ORG      0FC00H          Programmet laddas in i DOSBUF7
;
FGPOINT  EQU      7003H
FGLINE   EQU      7006H
FGPAINT  EQU      700CH
TÖMBILD  EQU      7015H
NYFÄRG   EQU      701BH
PUNKTADR EQU      705DH
;
;          IY-09H=FEF7H  BILDSORT
;          IY-0AH=FEF6H  PIXMALL
;          IY-0BH=FEF5H  FÄRG
;
START    LD        (IY-09H),02H    Grafikmod 256*240
         LD        (IY-0AH),0FH    Sätt riktig pixelmask
         LD        HL,0000H
         LD        (0FEF8H),HL     Ingen korrigering av x
         LD        HL,00FFH
         LD        (0FEFAH),HL     Maximalt x-värde = 255
;
         XOR      A                A=00H
         OUT      (06H),A          Visa och skriv i bildminne 0
         CALL     TÖMBILD          Ladda färgnr 0 på hela skärmen
         CALL     FGCTL            Färgval
;
         LD        (IY-0BH),11H    Utritning sker med färgnr 1
         LD        BC,30           X=30
         LD        HL,30           Y=30
         CALL     PUNKTADR        Omvandla Y och spara X och Y
         CALL     FGPOINT        Sätt en röd punkt
;
         LD        BC,130          X=130
         LD        HL,30           Y=30
         CALL     PUNKTADR
         CALL     FGLINE          Dra en röd linje
;
         LD        BC,230          X=230
         LD        HL,230          Y=230
         CALL     PUNKTADR
         CALL     FGLINE          Dra en röd linje
;
         LD        BC,30           X=30
         LD        HL,30           Y=30
         CALL     PUNKTADR
         CALL     FGLINE          Dra en röd linje
;
         LD        (IY-0BH),22H    Utritning sker med färgnr 2
         LD        BC,60           X=60
         LD        HL,40           Y=40
         CALL     PUNKTADR
         CALL     FGPAINT        Fyll triangeln med gulgrönt
;
         RET                      Retur till BASIC
;
```

```

; -----
;
FGCTL      LD      HL, FÄRGVAL      HL=Adress till färgvalsdata
          LD      B, 00H           Första färgnummer är 0
          CALL   NYFÄRG           Utför färgvalet
          RET
;
FÄRGVAL   DEFB   00H              Färgnr 0 = svart
          DEFB   11H              "      1 = röd
          DEFB   23H              "      2 = gröngul
          DEFB   00H
          DEFB   00H, 00H, 00H, 00H
          DEFB   00H, 00H, 00H, 00H
          DEFB   00H, 00H, 00H, 00H
;
          END      START

```

Här följer BASIC-programmet för ovanstående exempel:

```

100 POKE 64512,253,54,247,2,253,54,246,15,33,0
110 POKE 64522,0,34,248,254,33,255,0,34,250,254
120 POKE 64532,175,211,6,205,21,112,205,98,252,253
130 POKE 64542,54,245,17,1,30,0,33,30,0,205
140 POKE 64552,93,112,205,3,112,1,130,0,33,30
150 POKE 64562,0,205,93,112,205,6,112,1,230,0
160 POKE 64572,33,230,0,205,93,112,205,6,112,1
170 POKE 64582,30,0,33,30,0,205,93,112,205,6
180 POKE 64592,112,253,54,245,34,1,60,0,33,40
190 POKE 64602,0,205,93,112,205,12,112,201,33,107
200 POKE 64612,252,6,0,205,27,112,201,0,17,35
210 POKE 64622,0,0,0,0,0,0,0,0,0,0
220 POKE 64632,0,0,0
230 Start=64512
240 !
250 ; CHR$(12);
260 PROCEDURE CALL(Start) ! Exekvera assemblerprogrammet
270 !
280 END

```

#### 4.2.2 Grafikscroll

Detta exempel visar hur man kan använda blockförflyttningsrutinerna.

Programmet förflyttar hela bilden en pixel nedåt och fyller raden högst upp med färgnr 0.

```

EX2      ZPROG Grafikscroll
;
          ORG      0FC00H           Programmet laddas in i DOSBUF7
;
PUTMEM   EQU      7FDBH
LOADDR   EQU      7FDDH
;
START    LD      HL, 777FH         HL=Source
          LD      DE, 77FFH         DE=Destination
          LD      BC, 128*239       BC=Antal bytes
          CALL   LOADDR           Flytta allt en rad nedåt

```



```

;
; Fyller raden högst upp med färgnr 0.
;
      XOR    A                A=00H
      CALL  PUTMEM           Ladda färgnr 0 i minnescell
      LD    H,D
      LD    L,E
      DEC   DE
      LD    BC,127          BC=Antal bytes
      CALL  LOADDR          Ladda färg 0 på översta raden
      RET
;
      END    START

```

Här nedan följer ett BASIC-program som använder ovanstående assemblerprogram.

```

100 POKE 64512,33,127,119,17,255,119,1,128,119,205
110 POKE 64522,221,127,175,205,219,127,98,107,27,1
120 POKE 64532,127,0,205,221,127,201
130 Start=64512
140 !
150 FGCTL BLK+RED+GRN+YEL+BLU
160 FGCLR
170 ; CHR$(12);
180 !
190 FOR I=0 TO 200 STEP 20
200   FGPOINT I,I,INT(RND*4.)+1 ! Slumpa rektanglar
210   FGFILL I+19,I+19
220 NEXT I
230 !
240 FOR I=0 TO 239
250   PROCEDURE CALL(Start) ! Flytta ned bilden en pixelrad
260 NEXT I
270 !
280 END

```

## 5. SYSTEMVARIABLER

Dec	Hex	Typ	Namn	
65138	FE72	B	TIMEOUT	Sekunder i timeout 1-128, 0=ingen timeout.
65139	FE73	B		Slask i timeout
65140	FE74	B		Slask i timeout
65141	FE75	S16	FLASH6	Färgval 6 i FLASH (kompilerad form).
65157	FE85	S16	FLASH5	Färgval 5 i FLASH
65173	FE95	S16	FLASH4	Färgval 4 i FLASH
65189	FEA5	S16	FLASH3	Färgval 3 i FLASH
65205	FEB5	S16	FLASH2	Färgval 2 i FLASH
65221	FEC5	S16	FLASH1	Färgval 1 i FLASH (kompilerad form).
65222	FEC6			
65223	FEC7			
65224	FEC8			
65225	FEC9			
65226	FECA			
65227	FECB			
65228	FECC			
65229	FECD			
65230	FECE			
65231	FECF			
65232	FEDO			
65233	FED1			
65234	FED2			
65235	FED3			
65236	FED4			
65237	FED5	W	CLOCK	Adress till klock-rutinen.
65238	FED6			
65239	FED7	W	XKORRIG	X-index i FGDRAW.
65240	FED8			
65241	FED9	W	YKORRIG	Y-index i FGDRAW.
65242	FEDA			
65243	FEDB	W	OPTION1	Adress till option 1 i FGDRAW.
65244	FEDC			
65245	FEDD	W	OPTION2	Adress till option 2 i FGDRAW.
65246	FEDE			
65247	FEDF	W	OPTION3	Adress till option 3 i FGDRAW.
65248	FEE0			
65249	FEE1	W	OPTION4	Adress till option 4 i FGDRAW.
65250	FEE2			
65251	FEE3	W	SP1SAVE	Slask i FGPAINT.
65252	FEE4			
65253	FEE5	W	SP2SAVE	Slask i FGPAINT och FGLINE.
65254	FEE6			
65255	FEE7	B	DRAWSYS	FF=DRAW OO=RUB
65256	FEE8	B	FLASHSYS	Innehåller antal färgval i FLASH.

Dec	Hex	Typ	Namn	
65257	FEE9	B	FLAGS	b0=FLASH-interrupt on, b1=DOS urkopplat
65258	FEEA			Ej använd.
65259	FEEB			Ej använd.
65260	FEEC	S3	SCROLL	Hopp till nuvarande scrollrutin.
65261	FEED			
65262	FEEE			
65263	FEEF	S3	TERMINAL	Hopp till ev terminalrutin.
65264	FEF0			
65265	FEF1			
65266	FEF2	B	ATTRIB	Senast angivna attribut.
65267	FEF3	B	RAMDISK	Antal bilder.
65268	FEF4	B	BILDMAX	Antal bilder som används.
65269	FEF5	B	FÄRG	Senaste färgnummer (formaterat).
65270	FEF6	B	PIXMALL	Pixelmask. 256-mod: b0-b3 512-mod: b0-b1
65271	FEF7	B	BILDSORT	0=240 1=512 2=256
65272	FEF8	W	XKORR	Korrigerar x-koordinaten (=8 i 240-mod).
65273	FEF9			
65274	FEFA	W	XMAX	Högsta tillåtna x-koordinat.
65275	FEFB			
65276	FEFC	W	XOLD	Senaste x-koordinat.
65277	FEFD			
65278	FEFE	W	YOLD	Senaste y-koordinat.
65279	FEFF			

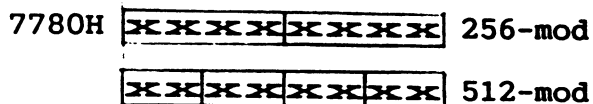
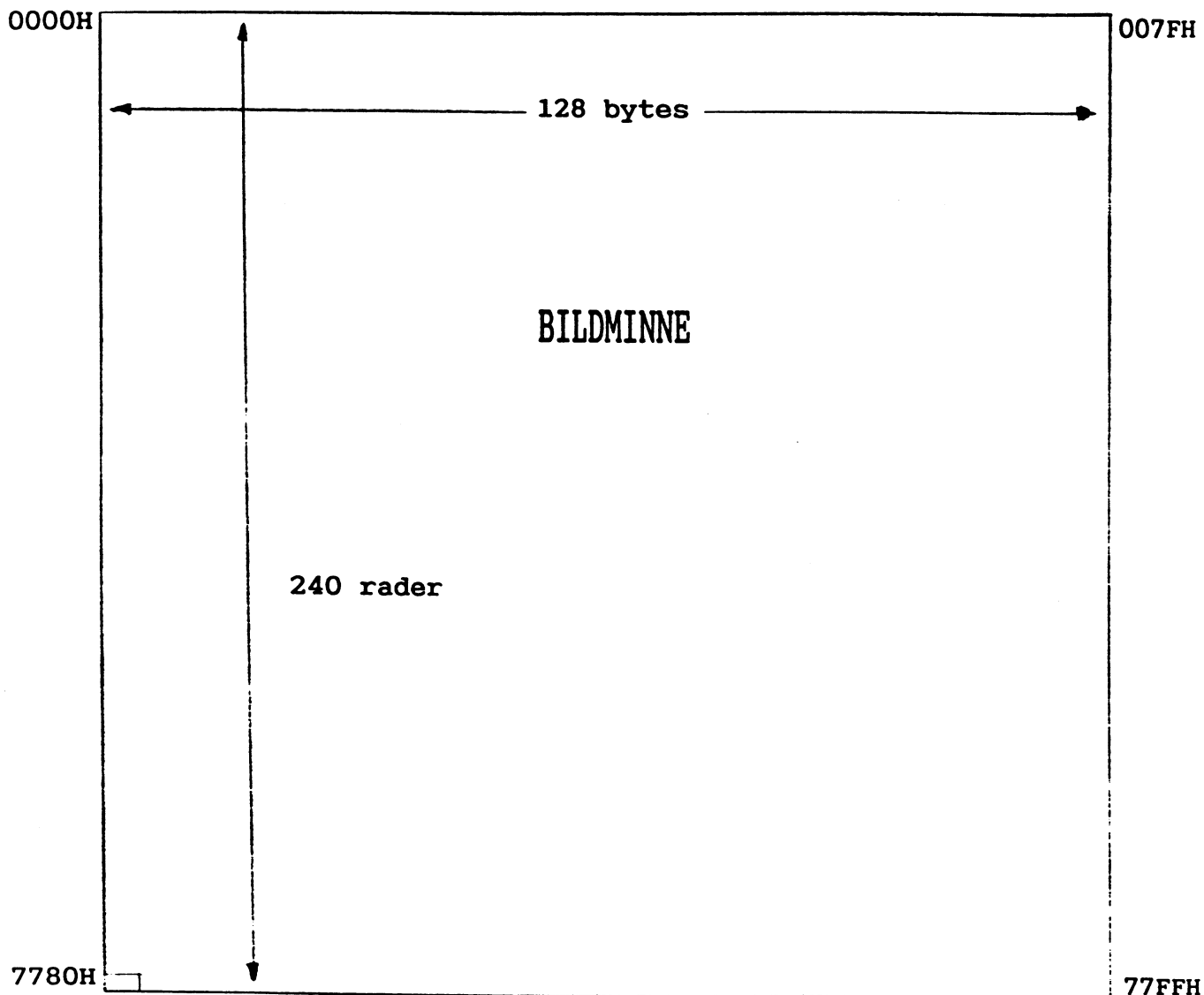
## 6. GRAFIK

### 6.1 Bildminne

#### 6.1.1 Minneskonfiguration

ABC806 har 4 (ev 16) bildminnen som vardera består av 32K byte. Av dessa 32K byte används 30K som reellt bildminne. I figur 2 kan vi se hur det är uppbyggt med adresser o s v.

Via port 6 bestämmer man vilken bild som skall visas och vilken bild man kan skriva och läsa i. Bit 4-7 anger vilket bildminne som man skriver i och bit 0-3 anger vilket bildminne som visas.



Figur 2.

## 6.1.2 Pixel och färgnummer

Varje byte i bildminnet innehåller 2 resp 4 pixels i bredd (beroende på grafikmod). En sådan pixel består då av 4 resp 2 bits som bestämmer den färg som syns på skärmen. När man ritar ut en pixel på skärmen använder man sig av två systemvariabler. Den ena bestämmer färg och den andra bestämmer vilka bitar som skall påverkas i pixeln.

Ex. I 256x240 pixelmod är pixelmasken normalt 00001111B (15), eftersom man skall kunna påverka alla bit'arna i en pixel (16 färger).

Man kan med hjälp av POKE-satser ändra grafik-mod och slipper därmed gå via FGCTL för att byta mod.

Ex. Man befinner sig i 256-mod och vill använda sig av 512-mod ett tag (se 5 och 7.1.3).

```
POKE 65270,3,1,0,0,255,1 ! Ändrar till 512-mod
POKE 65270,15,2,0,0,255,0 ! Ändrar till 256-mod
```

När man går in i en grafikrutin via assembler måste man försäkra sig om att systemvariablerna FÄRG, PIXMALL, BILDSORT, XKORR och XMAX är riktiga.

FÄRG innehåller senaste färgnummer fast formerat. Med formerat menas att färgnumret finns i varje pixel i FÄRG. Färgnummer 1 är då 00010001B i 256-mod och 01010101B i 512-mod.

## 6.2 Färgval

Färgval görs via port 7. Exemplet som följer är samma rutin som finns i PROM'et (se NYFÄRG i 4.1 för in- och ut-parametrar).

NYFÄRG	CALL	BILDSYNC	Vänta på synksignal från video
	LD	C, 07H	C-Färgvalsport
OUTLOOP	LD	A, (HL)	A-Färgdata för pixeln
	OUT	(C), A	Ändra färg för färgnummer B
	INC	HL	Öka data-pekar
	INC	B	Öka färgnumret
	BIT	4, B	
	JR	Z, OUTLOOP	Hoppa om färgnummer <16
	RET		

Färgdata består av vvvvhhhhB, vvvv=färg i vänster del av pixeln  
hhhh=färg i höger del av pixeln

Färgerna är:	0000 BLK	1000 GBLK
	0001 RED	1001 GRED
	0010 GRN	1010 GGRN
	0011 YEL	1011 GYEL
	0100 BLU	1100 GBLU
	0101 MAG	1101 GMAG
	0110 CYA	1110 GCYA
	0111 WHT	1111 GWHT

## 7. BLANDADE PROGRAMEXEMPEL

### 7.1 Grafik

#### 7.1.1 Animering

Med animering menas att man får saker att röra sig på skärmen. Detta kan ske på flera olika sätt t ex:

1. Sudda, rita, sudda, rita, ..... i samma bildminne, men detta ger en förflyttning med mycket flimmer.
2. Sudda, rita, byt bild, sudda, rita, byt bild, ..... vilket ger en stadig bild utan flimmer.
3. Färgvalsbyten, t ex den animering som sker när man använder sig av MIXLINE och FLASH.
4. Kombination av bildbyten och färgvalsbyten, t ex den snurrande färgcirkeln i demoprogrammet.

Här nedan följer ett programexempel som snurrar en linje kring en punkt. Metod 2 används.

```
100 FGCTL MIX BLK+BLK+WHT+WHT
110 !
120 FOR I=0 TO 1
130   FGPICTURE 1,1,2 ! Anger vilken bild som skall tömmas
140   FGCLR
150 NEXT I
160 !
170 ; CHR(12); ! Töm textbildminnet
180 !
190 WHILE Vinkel.<100.
200   FOR I=0 TO 1
210   !
220   FGPOINT FNXpos(Vinkel.-PI),FNYpos(Vinkel.+PI),1
230   EGLINE FNXpos(Vinkel.),FNYpos(Vinkel.)
240   !
250   FGPICTURE 1,1-1,2 ! Visa bilden
260   !
270   FGPOINT FNXpos(Vinkel.-PI-.3),FNYpos(Vinkel.+PI-.3),10
280   EGLINE FNXpos(Vinkel.-.3),FNYpos(Vinkel.-3)
290   !
300   Vinkel.=Vinkel.+3
310   !
320   NEXT I
330 WEND
340 !
350 END
360 !
370 ! -----
380 !
390 DEF FNXpos(V.)=128+COS(V.)*30
400 !
410 DEF FNYpos(V.)=120+SIN(V.)*30
```

## 7.1.2 Oberoende bilder

Med oberoende bilder menas att man i ett bildminne har flera olika bilder, som ej kan påverka varandra. Detta gäller nästan enbart vid 256-mod.

Det hela kan verka underligt, men har en mycket simpel förklaring. Eftersom varje pixel består av 4 bit'ar, så kan man reservera en eller flera bit'ar i pixeln för en viss bild.

Ex. Man vill ha två oberoende bilder en med 8 olika färger och en med en enda färg. Bilden med 8 färger ligger fast som bakgrundsbild medan bilden med endast en färg skall förändras hela tiden. Bit 0,1 och 2 i pixeln reserveras då för den fasta bilden och bit 3 reserveras för den bild som skall förändras. Vid utritning av den fasta bilden används pixelmask 00001111B och vid utritning av den föränderliga bilden används 00001000B som pixelmask (se 6.1.2). Sedan bestämmer färgvalet vilken bild som skall vara överst (se 3.2.4 eller DEMO11.BAC).

Här nedan följer ett programexempel på "window" mellan två oberoende bilder. I exemplet är en bild definierad i bit 0 och en annan bild i bit 2. Bit 1 är definierad för att bestämma vilken bild som skall visas.

```
100 FGCTL MIX BLK+BLK+WHT+WHT+BLU+BLK+BLU+BLK+BLK+BLK+WHT+WHT+
    YEL+YEL+YEL
110 !
120 FGCLR
130 ; CHR(12);
140 !
150 Rektangel=CHR(240,0,0,242,0,15,242,15,15,242,15,0,242,0,0,240,
    2,2,244,13,13)
160 !
170 Triangel=CHR(240,0,0,242,15,30,242,30,0,242,0,0)
180 !
190 Sudda=CHR(240,0,0,244,255,239)
200 !
210 Window0=CHR(240,0,0,244,39,39)
220 Window1=CHR(240,0,39,244,4,0,244,39,4)
230 Window2=CHR(240,0,39,244,39,35,244,35,0)
240 !
250 FOR I=1 TO 50
260   FGSPEC 17,15,RND*240,RND*224,Rektangel
270 NEXT I
280 !
290 FGSPEC 34,2,0,0,Sudda ! Sätt bit 1 på hela skärmen
300 !
310 FOR I=1 TO 50
320   FGSPEC 68,4,RND*200,RND*200,Triangel ! Rita trianglar
330 NEXT I
340 !
350 FGSPEC 0,2,0,0,Window0 ! Öppna ett fönster till bild 1
360 !
370 FOR J=1 TO 100
380   FOR I=0 TO 195 STEP 5
390     FOR I.=1. TO 50. : NEXT I. ! En kort paus i förflyttningen
400     FGSPEC 0,2,I+5,I+5,Window2
410     FGSPEC 34,2,I,I,Window1
420   NEXT I
```

```

430 !
440 FOR I=200 TO 5 STEP -5
450 FGSPEC 0,2,I-5,I-5,Window1
460 FGSPEC 34,2,I,I,Window2
470 NEXT I
480 NEXT J
490 !
500 END

```

### 7.1.3 Blanda 256- och 512-mod med varandra

Om man läser 6.2 ordentligt inser man snart att det är möjligt att blanda de olika upplösningarna på en och samma bild. Därför kommer här nedan ett enkelt program-exempel på detta. I exemplet är 4 färger reserverade för högre upplösning i 256-mod.

```

100 FGCTL MIX BLK+BLK+BLK+WHT+BLU+MAG+YEL+YEL+WHT+BLK+WHT+WHT+GRN
110 !
120 FGCLR
130 ; CHR(12);
140 !
150 FOR I=0 TO 239 STEP 4
160 FGPOINT 0,0,2,I Dra en blållila linje
170 FGLINE 127,I
180 NEXT I
190 !
200 FGPOINT 127,239 ! Dela skärmen i 2 delar
210 FGLINE 127,0
220 !
230 FGPAINT 0,100,3 ! Måla lite gult i den tomma ytan till väster
240 !
250 POKE 65270,3,1,0,0,255,1 ! Byt till 512-mod
260 !
270 FOR I=0 TO 239 STEP 4
280 FGPOINT 256,0,1 ! Dra vita högupplösande linjer
290 FGLINE 511,I
300 NEXT I
310 !
320 POKE 65270,15,2,0,0,255,0 ! Byt tillbaka till 256-mod
330 !
340 FGPAINT 129,100,6 ! Måla lite grönt
350 !
360 END

```

## 7.2 Hur man gör extended BASIC m h a PROM'et

### 7.2.1 Kompilering av inparametrar

Exemplet nedan är hämtat från EXTBASIC.BAC.

```

;
COMPTAB DEFW COMPMOVE Kompilera LDIR
DEFW COMPMOVE Kompilera LDDR
DEFW COMPSTAT Kompilera MIXSTATUS
DEFW COMPTWO Kompilera MIXLINE
;
-----
;
COMPMOVE CALL COMPTWO Kompilera till och från
JP COMPONE Kompilera antal
;
COMPSTAT CALL COMPMOVE Kompilera längd,fnr1 och fnr2
JP COMPONE Kompilera fnr3

```



## 7.2.2 Parameterläsning och programkörning

Som instruerande exempel används instruktionen LDIR.

```
;
LOADPARA  RST  20H           Hämta source
           PUSH HL          Spara source
           ;
           INC  DE          Skippa ","
           RST  20H         Hämta destination
           PUSH HL          Spara destination
           ;
           INC  DE          Skippa ","
           RST  20H         Hämta antal
           PUSH HL          Spara antal
           ;
           EXX              Spara instruktionspekaren
           POP  BC          BC=Antal
           POP  DE          DE=Destination
           POP  HL          HL=Source
           RET
;
BLDIR     CALL  LOADPARA    Hämta parametrar
           CALL  LOADDIR    Exekvera LDIR
           EXX              Hämta instruktionspekaren
           RET
```

## 8. BILAGOR

### 8.1 Systemprogram

#### 8.1.1 EXTBASIC.BAC

Programmet EXTBASIC.BAC laddar in de instruktioner som nämns i 3.1.2. Programmet är relokerbart, dvs det letar upp en ledig plats i minnet och lägger sig där.

#### 8.1.2 DRAWMODU.BAS

Innehåller färdiga funktioner som skapar en data-sträng för FGDRAW.

- FNPoint(X,Y,Färg)
- FNLine(X,Y,Färg)
- FNFill(X,Y,Färg)
- FNPaint(X,Y,Färg)
- FNMixstatus(Längd,Färg1,Färg2,Färg3)
- FNMixline(X,Y)

Dessa adderas till ens egna program via "MERGE DRAWMODU.BAS"  
Svar fås i F som byggs på efter varv.

#### 8.1.3 DEMO.BAC

Demonstrerar lite vad man kan göra.

1. Inledning
2. Interferensmönster
3. Snabbare FGPAINT och sliskiga färger
4. Färgdemo
5. Färgcirkel
6. 500 st sprites
7. Luxors stjärna
8. Hatten (3dim-framställning av en funktion)
9. VVS-schema m h a MIXLINE
10. Staplar där en markeras med FLASH
11. Mysko mönster med 512-mode
12. Cirkclar
13. Window (7.1.2)
14. Preview på PFILL (pattern)
15. Avslutning

## 8.2 Timeout på PR: och V24:

### 8.2.1 Förklaringar

På GRAF 806 fr o m verion 4.0 finns det timeout på PR: och V24:. Detta har gjorts efter många önskemål från olika programmerare.

Timeout aktiveras genom att ladda antalet sekunder på adress 65138 där 0 (default) betyder ingen timeout och 1-128 är antal sekunder som får gå innan timeout genererar error 42. På det här sättet kan man behandla fel i kommunikationerna med ON ERROR GOTO.

OBS! Undvik en sekund eftersom 1 betyder 0-1 sekunds timeout.

Versionsnummer på PROM kan avläsas på adress 28764. Verifikationsnummer finns på adress 28762. Detta nummer läses av med PEEK2(28762) och skall vara 21761.

### 8.3 Programexempel med timeout

```
100 IF PEEK(28764)<40 : "Fel versionsnummer på PROM" : STOP
110 :
120 ON ERROR GOTO 300
130 :
140 OPEN "V24:VSA30E72.55B" AS FILE 1
150 :
160 POKE 65138,5 : 5 sekunders timeout
170 :
180 INPUT LINE #1,A
190 : #1,"Mottaget!"
200 : A
210 END
220 :
230 IF ERRCODE<>42 RESUME
240 : "Kommunikationsobjektet är dött"
250 STOP
```

### 8.3 Referenslitteratur

1. BIT FÖR BIT MED ABC800      Luxor datorer AB  
Tryck-center AB
2. Z80-Assembly Language      Zilog  
  Programming Manual
3. MANUAL BASIC II            Luxor datorer AB  
  ABC806
4. SERVICE MANUAL ABC806      Luxor datorer AB
5. BASIC II BOKEN            Jan Lundgren  
                                  Sören Thornell  
  
                                  ISBN 91-86064-04-5  
  
                                  Liber