

Schnelle Spielegrafik beim C16

Diese Einführung in die »Innereien« des C16 zeigt, wie schnelle, anspruchsvolle (Spiele-) Grafiken mit einem Zehntel des bisher nötigen Speicherbedarfs realisiert werden können.

Inzwischen hat es sich unter C16-Besitzern wohl schon herumgesprochen, daß es Schwierigkeiten bei der Kombination von Spielprogrammen mit der hochauflösenden Grafik gibt. Hier soll nun eine alternative Möglichkeit zur Grafikerzeugung aufgezeigt werden, die besonders für die Spieleprogrammierung geeignet ist. Die Rede ist von einem selbstdefinierten Zeichensatz als Ersatz für Hochauflösung und Shapes, der nur 1 KByte Speicher (statt 10 KByte bei Hochauflösung) belegt.

Auch der C16 hält, wie seine Vorgänger, einen Farb- und einen Videospeicher für die Bildschirminformationen bereit. Dieser liegt im Speicherbereich zwischen \$0800 (2048) und \$0FFF (4095), wobei das erste KByte dem Farbspeicher zugeordnet ist. In der Struktur der einzelnen Bits innerhalb einer Farbspeicherstelle hat sich jedoch – um wieder die anderen beiden Computer (C64 und VC20) zum Vergleich heranzuziehen – etwas geändert. Die untersten 4 Bit jeder Speicherstelle geben nach wie vor die jeweilige Farbe an der dazugehörigen Bildschirmposition wieder. Zusätzlich ist es jetzt beim C16 möglich, auch den Helligkeitswert jedes Zeichens individuell zu bestimmen. Dieser sogenannten Luminanz sind die Bits 4-6 (also 7 Helligkeitsstufen) zugeordnet. Bit 7 entscheidet schließlich noch darüber, ob das Zeichen normal oder blinkend dargestellt wird.

Reverse Zeichen werden im Gegensatz zu VC20 und C64 hardwaremäßig (durch Setzen des Bit 7 in der entsprechenden Videospeicherstelle) vom TED erzeugt. Somit benötigt der C16 nur noch 2 KByte Speicherplatz für die Zeicheninformationen im ROM.

Natürlich ist es auch denkbar, die Daten in den RAM-Speicher zu übertragen, um dort Änderungen am Zeichensatz vorzunehmen. Dies geschieht prinzipiell genauso wie beim VC20 oder C64. Man muß lediglich die neue Adresse des Zeichensatzes im Register 14 (\$FF13, 65299) vermerken und auf RAM-Zugriff schalten (Bit 2 in Register 13). Das ist in der Praxis alles viel einfacher, als es im ersten Moment den Anschein hat. Dazu möchte ich zunächst eine kurze Beschreibung des beim C16 verwendeten »Bankswitching« vorausschicken: Der C16 kann hardwaremäßig 128 KByte Speicher verwalten; 64 KByte ROM und noch einmal soviel RAM-Speicher. Da die Adreßleitungen nach wie vor aber nur für 64 KByte Gesamtkapazität ausgelegt sind, bedient man sich eines Tricks – dem Bankswitching. Das bedeutet, es wird je nach Bedarf zwischen 64 KByte ROM und RAM hin- und hergeschaltet. Dazu ist sowohl die CPU als auch der TED-Chip in der Lage. Ob er auf den Zeichensatz im ROM oder auf einen im RAM zugreifen soll, entscheidet ein Registerbit (\$FF13 Bit 2). Somit ist genau dieses bei der Umschaltung auf den RAM-Zeichensatz zu ändern.

Zum besseren Verständnis zunächst ein kleines Beispiel. Dazu verwenden wir den eingebauten Monitor des C16. Als erstes wird der Zeichensatz aus dem ROM – er liegt bei Adresse \$D000 – mittels des Transferbefehls (T D000 D800 1000) in den normalen Speicher kopiert. Danach ändern wir, damit man die beiden (im Moment noch identischen) Zeichensätze unterscheiden kann, das erste Zeichen (es ist der »Klammeraffe« @ in Adresse \$1000 bis \$1007. Dazu kann man ihn beispielsweise abwechselnd mit \$00 und \$FF füllen.

Als nächstes müssen die Registerinhalte so abgeändert werden, daß der TED nur noch auf unseren eigenen Zeichensatz zugreift. Dazu wird zunächst in Register 13 (\$FF12) Bit 2 ausgeschaltet, damit von nun an auf RAM zugegriffen wird. In unserem Beispiel ersetzt man derzeitigen Wert \$C4 durch \$C0. Danach ist noch die Zugriffsadresse für den Zeichengenerator von ursprünglich \$D000 auf \$1000 (der neuen Adresse) zu ändern; dazu wird das High-Byte der Zugriffsadresse in Bit 4-7 dieses Registers eingetragen (statt \$D1 jetzt also \$11). Nun kann man anhand des Klammeraffen-Zeichens kontrollieren, ob man den richtigen Zeichensatz vor sich hat.

Ein Zeicheneditor

Mit Hilfe des Programms »Matrix-Editor« (Listing) kann man nun komfortabel am Zeichensatz »herumpfuschen«. Dazu kurz noch einige Erläuterungen. Das vorliegende Basic-Programm (Listing) wird einfach in den Speicher geladen und gestartet. Eine der integrierten Maschinenroutinen verschiebt das ganze Programm automatisch so im Speicher,

Multiregister 1

- Bit 0-2 : Bitweise vertikale Verschiebung des Bildschirms
- Bit 3 : 24 oder 25 Zeilen
- Bit 4 : Bildschirm abschalten

Multiregister 2

- Bit 0-2 : Bitweise horizontale Verschiebung des Bildschirms
- Bit 3 : 38 oder 40 Spalten
- Bit 4 : Multicolor-Flag
- Bit 6 : PAL- oder NTSC-Norm
- Bit 7 : Hardwaremäßige Invertierung ein/aus

Multiregister 3

- Bit 1 : Rasterinterrupt
- Bit 2 : Lichtgriffelinterrupt
- Bit 3 : Timer 1 interrupt
- Bit 4 : Timer 2 interrupt
- Bit 6 : Timer 3 interrupt
- Bit 7 : Interrupt-Flag

Multiregister 4

- Bit 1 : Rasterinterrupt verhindern
- Bit 2 : Lichtgriffelinterrupt verhindern
- Bit 3 : Timer 1 abschalten
- Bit 4 : Timer 2 abschalten
- Bit 6 : Timer 3 abschalten

Multiregister 5

- Bit 0-3 : Lautstärke
- Bit 4 : Tongenerator 1 einschalten
- Bit 5 : Tongenerator 2 einschalten
- Bit 6 : Tongenerator 2 Rauschen

Multiregister 6

- Bit 0-1 : Frequenz von Tongenerator 1 (Bit 8 und 9)
- Bit 2 : Zeichensatzzugriff (ROM oder RAM)
- Bit 3-7 : Adresse des Bitmapping-Speichers

Multiregister 7

- Bit 2-7 : Adresse des Zeichengenerators

Tabelle 1. Die Multiregister im TED

Reg. Nr.	Adresse	Funktion	Erläuterung
1	\$FF00	Timer 1	(Low-Byte)
2	\$FF01	Timer 1	(High-Byte)
3	\$FF02	Timer 2	(Low-Byte)
4	\$FF03	Timer 2	(High-Byte)
5	\$FF04	Timer 3	(Low-Byte)
6	\$FF05	Timer 3	(High-Byte)
7	\$FF06	Multiregister 1	
8	\$FF07	Multiregister 2	
9	\$FF08	Tastatur	
10	\$FF09	Multiregister 3	
11	\$FF0A	Multiregister 4	
12	\$FF0B	Rasterinterrupt	
13	\$FF0C	Hardware Cursor Position	(Bit 8 und 9)
14	\$FF0D	Hardware Cursor Position	(Bit 0 bis 7)
15	\$FF0E	Tongenerator 1	
16	\$FF0F	Tongenerator 2	(Bit 0-7)
17	\$FF10	Tongenerator 2	(Bit 8 und 9)
18	\$FF11	Multiregister 5	
19	\$FF12	Multiregister 6	
20	\$FF13	Multiregister 7	
21	\$FF14	Adresse des Farb- und Videospeichers	
22	\$FF15	Hintergrundfarbe	
23	\$FF16	Vordergrund	
24	\$FF17	Multicolor 1	
25	\$FF18	Multicolor 2	
26	\$FF19	Rahmenfarbe	
27	\$FF1A	Bitmapping-Register	
28	\$FF1B	Bitmapping-Register	
29	\$FF1C	Abgetastete Rasterzeile	(Bit 8)
30	\$FF1D	Abgetastete Rasterzeile	(Bit 0-7)
31	\$FF1E	Rasterspalte	
32	\$FF1F	Vertikal-Adresse	

Tabelle 2. Die Register des TED

daß der zu editierende Zeichensatz ab Adresse \$1000 (wo sich eben normalerweise das Programm befindet) geladen werden kann.

Mit dem Cursor sucht man sich dann ein zu bearbeitendes Zeichen aus, wobei man sich wahlweise die Originalzeichen oder den geänderten Zeichensatz zeigen lassen kann. Durch einen Druck auf die RETURN-Taste wählt man das entsprechende Zeichen an. Dieses wird nun 8-fach vergrößert, damit man es – ebenfalls mit dem Cursor – editieren kann. Dabei sind die Funktionstasten das wichtigste Hilfsmittel: Mit F1 kann ein Punkt gesetzt werden (der Cursor bewegt sich dabei eine Position nach rechts), F2 löscht einen Matrixpunkt an der Zeigerposition. Mit der dritten Funktionstaste wählt man die Copy-Routine an. Damit läßt sich ein zweites Zeichen (wahlweise aus den beiden Original-Zeichensätzen oder aus dem zu editierenden) selektieren, das in das gerade bearbeitete Zeichen kopiert wird.

Übrigens kann man sich bei eventueller Fehlbedienung (egal in welchem Programmteil) mit der Escape(ESC)-Taste von dieser Funktion wieder befreien. Auch in das eigentliche Auswahlménú (welches nach dem Programmstart erscheint) gelangt man durch Drücken der ESC-Taste.

Selbstverständlich ist auch das Laden und Speichern des Zeichensatzes möglich (diese Funktionen wählt man mit F6 und F7 an). Ferner ist ein Hilfsschirm, der über die wichtigsten Funktionen informiert, mittels der HELP-Taste abrufbar. Damit kann man sich noch über einige zusätzliche Funktionen des »Matrix-Editors« informieren, die daher hier nicht im einzelnen besprochen werden müssen.

Nun will man den gespeicherten Zeichensatz natürlich auch in seinen eigenen Programmen benutzen. Dabei ist zu beachten, daß man den Basic-Anfang (Adresse 44) vor dem Laden des Zeichensatzes verschieben muß, denn dort, wo normalerweise der Anfang des Basic-Speichers ist (bei

\$1000 beziehungsweise dezimal 4096), soll ja später der Zeichensatz abgelegt werden. Die Verschiebung wird mit »POKE 44,20 : POKE 20*256,0 : NEW« durchgeführt. Damit sind die Zeichendaten vor dem Zugriff des Basic-Interpreters geschützt und können von diesem nicht mehr überschrieben werden.

Wenn das Programm fertig erstellt ist, kann man Programm und Zeichensatz auch gleichzeitig abspeichern. Dazu muß zunächst ein kleiner Vorspann zu dem eigentlichen Programm hergestellt werden:

```
10 POKE 44,20
```

```
20 LOAD "TEIL 2" ,8,1 : REM " ,1,1" bei Datasette
```

Dieser wird nun zuerst gespeichert. Als nächstes wird der Basic-Anfang wie oben beschrieben verschoben. Dann wird der Zeichensatz mit »LOAD "Zeichensatz",8,1« von Disk oder Band (»,1,1«) nachgeladen. Schließlich und endlich lädt man dann das eigentliche Programm in den Basic-Speicher. Somit ist alles komplett. Diese Mischung von Zeichendaten und Programm wird nun hinter den zuvor generierten Vorspann gespeichert. Dazu gibt man ein:

```
»POKE 44,16 : SAVE "TEIL 2"« (»,8« für Diskette).
```

Damit kann man nun die mit dem »Matrix-Editor« erzeugten Zeichen in eigene Programme übernehmen.

Statt in hochauflösender Grafik mit Shapes zu arbeiten, kann man mit dieser Methode im normalen Textmodus bleiben und dennoch beliebige Grafiken auf den Bildschirm bringen, entweder mit PRINT-Anweisungen oder über POKES in den Bildschirmspeicher. Das bringt nicht nur einen Geschwindigkeitsvorteil, sondern schont vor allem den begrenzten C16-Speicher. Mit über 11000 freien Bytes sollte sich schon einiges anfangen lassen.

(Christoph Sauer/ev)

```
10 PRINT "{CLR}";
20 CHAR1,6,2,"{RVSON,FLASHOFF}BITTE WART
EN !!!!!!"
30 POKE56,62:POKE55,223:CLR
40 TRAP 2570
50 PRINTCHR$(8)
60 ZA=4096:E$=CHR$(27)
70 FORT=1TO8:KEYT,CHR$(T+34):NEXT
80 IFPEEK(44)=20THEN140
90 FORT=16096TO16381
100 READD$:D=DEC(D$):S=S+D
110 POKET,D:NEXT
120 IFS<>35182THENPRINT "{CLR,FLASHON}PRU
EFSUMMENFEHLER !! {FLASHOFF}":END
130 SYS 16278:RUN
140 W=0:X1=0
150 POKE16174,109:POKE16271,64:SYS16128
160 COLOR 0,2,6:COLOR 4,5,3:PRINT "{CLR,P
URPLE,RVSON}";
170 FORT=1TO40:PRINT "{SSPACE}";:NEXT
180 CHAR 1,5,2,"{WHITE,2SPACE}M A T R I
X{4SPACE}E D I T O R{2SPACE}"
190 CHAR 1,10,4,"{BLACK}(C) 1985 BY C. S
AUER {RVOFF}"
200 SYS 16096
210 CHAR1,0,14,"DARSTELLUNG: "
220 CHAR1,0,16,"{RVSON}F1 {RVOFF,3SPACE}:
STANDARD {3SPACE} ZEICHENSATZ "
230 PRINT:PRINT "{RVSON,FLASHON}F2 {FLASHO
FF,RVOFF,3SPACE}: UMDEFINIERTER ZEICHENS
ATZ "
240 PRINT "{RVSON}F3 {RVOFF,3SPACE}: ZEICH
EN ZUR BEARBEITUNG LADEN"
```

Listing »Matrix-Editor« für den C16.

Bitte beachten Sie die Eingabehinweise auf Seite 76.

```

250 CHAR1,0,21,"{RVSON}CLEAR{RVOFF}":PRI
NT:PRINT "{RVSON}HOME {SPACE,RVOFF}: ZEICH
ENSATZ INITIALISIEREN"
260 X=SX:Y=SY
270 GOSUB2670
280 IFA$="% "THENFA=1:SYS16122:GOTO1590
290 IFA$="{CLR}"THEN320
300 IFA$=CHR$(13)THENC=W:POKE2368+X1,113
: SX=X:SY=Y:GOTO370
310 GOTO270
320 COLOR 0,7,1:COLOR 1,7,1
330 CHAR1,0,21,"{WHITE}CLEAR"
340 CHAR1,0,22,"HOME : {SPACE,RVSON,FLASH
ON,5SPACE}SIND SIE SICHER ?{SPACE,RVOFF
,FLASHOFF}"
350 GETKEYA$: IFA$="J"THENSYS16349
360 GOTO160
370 GOSUB2850:POKE65298,196
380 FORT=0T07
390 BE(T)=PEEK(ZA+T+C*8)
400 BN=BE(T):GOSUB950
410 NEXT
420 COLOR 0,2,6:COLOR 4,4,5:PRINT "{CLR}"
;
430 PRINT "{WHITE,4RIGHT}+++++++"
440 PRINT "{4RIGHT}+(GREEN)76543210(WHITE
)F"
450 FORT=1T08
460 PRINT "T" {LEFT}. F(BLACK);BO$(T-1);
" {WHITE}F ";BE(T-1)
470 NEXT
480 PRINT "{4RIGHT}+++++++"
490 CHAR 1,22,6,"{RVSON,BLUE}STANDARD ZE
ICHEN"
500 CHAR 1,21,11,"{RVSON}SELBSTDEF. ZEIC
HEN"
510 POKE16271,106:POKE16174,123:SYS 1612
8:REM RASTERINTERRUPT
520 POKE2398,60:POKE3422,C
530 POKE2598,60:POKE3622,C
540 CHAR 1,21,11,"{RVSON}SELBSTDEF. ZEIC
HEN"
550 PRINTE$"M"
560 CHAR 1,0,17,"{LIG.BLUE}U*****
*****I"
570 PRINT "{4SPACE}F U N K T I O N S T A
S T E N{5SPACE}"
580 PRINT "R*****
*****"
590 PRINT "{RVSON}F1{RVOFF}: PUNKT SETZE
N{3SPACE}{RVSON}F4{RVOFF}: DARSTELLUNG{
3SPACE}"
600 PRINT "{RVSON}F2{RVOFF}: PUNKT LOESC
HEN {RVSON}F5{RVOFF}: ENDE{10SPACE}"
610 PRINT "{RVSON}F3{RVOFF}: KOPIEREN{7S
PACE}{RVSON}F6{RVOFF}: Z. SATZ{2SPACE}LA
DEN "
620 PRINT "{RVSON}HE{RVOFF}: HILFSCHIRM{
5SPACE}{RVSON}F7{RVOFF}: ZS. SPEICHERN
"
630 PRINT "J*****
*****";
640 P=2:D=1
650 GETKEY A$
660 IFA$=CHR$(27)THEN150
670 IFA$="&"THEN190
680 IFA$=" '"THEN210
690 IFA$="#"THENP=1:GOTO820
700 IFA$="$"THENP=0:GOTO820
710 IFA$="%"THEN1110
720 IFA$="{CLR}"THENFORT=0T07:GOSUB2850:
POKEZA+C*8+T,0:NEXT:GOTO380
730 IFA$="{INST}"THENFORT=0T07:GOSUB2850

```

```

:POKEZA+C*8+T,255:NEXT:GOTO380
740 IFA$="("THEN1580
750 IFA$=")"THEN1720
760 IFA$="*"THEN1370
770 FA=0
780 GOSUB2800
790 IFFA=0THEN820
800 IF A$="{HOME}"THENX=0:Y=0:GOTO820
810 GOTO650
820 IFD=1THEND=0:X=0:Y=0:AX=0:AY=0:Z=PEE
K(3157)
830 IFX>7THENX=0:Y=Y+40
840 IFX<0THENX=7:Y=Y-40
850 IFY>280THENY=0
860 IFY<0THENY=280
870 POKE3157+AX+AY,Z
880 Z=PEEK(3157+X+Y)
890 IFZ=81THENPOKE3157+X+Y,209:GOTO910
900 POKE3157+X+Y,190
910 AX=X:AY=Y
920 IFP=0THENGOSUB1020:X=X-1:Z=32:GOTO82
0
930 IFP=1THENGOSUB1020:X=X+1:Z=81:GOTO82
0
940 GOTO650
950 BO$(T)=" "
960 FORB=1T08
970 BN=BN/2
980 IF BN=INT(BN)THENBO$(T)=" "+BO$(T):E
LSEBO$(T)="_"+BO$(T)
990 BN=INT(BN)
1000 NEXT
1010 RETURN
1020 T=Y/40
1030 ON P+1GOTO1040,1050
1040 BE(T)=BE(T)AND255-2↑(7-X):GOTO1060
1050 BE(T)=BE(T)OR2↑(7-X)
1060 BN=BE(T):GOSUB950
1070 CHAR 1,0,T+2,"":P=2
1080 PRINT "{WHITE}T+1" {LEFT}. {RIGHT}+(B
LACK)"BO$(T)" {WHITE}+(6SPACE,5LEFT)"BE(T
)"
1090 POKEZA+C*8+T,BE(T)
1100 RETURN
1110 POKE16174,109:POKE16271,64:SYS16128
1120 COLOR 0,7,5:COLOR 4,13,2:PRINT "{CLR
,PURPLE,RVSON}";
1130 PRINT "{WHITE,RVSON,4SPACE}Z E I C H
E N{5SPACE}K O P I E R E N{3SPACE,RVOFF
}"
1140 SYS16096
1150 CHAR1,0,14,"KOPIEREN VON:"
1160 CHAR1,0,16,"{RVSON}F1{RVOFF,3SPACE}
: STANDARD{3SPACE}ZEICHENSATZ"
1170 PRINT:PRINT "{RVSON,FLASHON}F2{FLASH
OFF,RVOFF,3SPACE}: UDEFINIERTER ZEICHEN
SATZ"
1180 PRINT "{RVSON}F3{RVOFF,3SPACE}:"
1190 FL=2:ZS=0
1200 GOSUB2670
1210 IFFL=1THENCHAR1,0,18,"{RVSON}F3{RVO
FF,3SPACE}: UMSCHALTUNG GROSS/KLEIN"
1220 IFFL=2THENCHAR1,0,18,"{RVSON}F3{RVO
FF}":FORT=1T035:PRINT " ";:NEXT
1230 IFA$=CHR$(13)THENPOKE2368+X1,113:GO
SUB2850:GOTO1280
1240 IFA$=CHR$(27)THEN420
1250 IFA$="%"ANDFL=1ANDZS=0THENPRINTCHR$
(14):ZS=4:GOTO1200:ELSE1200
1260 IFA$="%"ANDFL=1ANDZS=4THENPRINTCHR$
(142):ZS=0
1270 GOTO1200
1280 IFFL=1THENAD=208+ZS:GOTO1300

```

```

1290 AD=ZA/256
1300 W1=W*8:C1=C*8:W2=INT(W1/256):C2=INT(C1/256)
1310 W1=W1-W2*256:POKE16115,AD+W2
1320 C1=C1-C2*256:POKE16118,ZA/256+C2
1330 POKE16114,W1:POKE16117,C1
1340 SYS16111:PRINTCHR$(142)
1350 GOTO380
1360 PRINT" {CLR,BLUE,SPACE,4RIGHT}ZEICHE
NSATZ:";FF=6
1370 SYS16122
1380 PRINT" {CLR,RVSON,5SPACE}H{2SPACE}I{
2SPACE}L{2SPACE}F{2SPACE}S{2SPACE}S{2SPA
CE}C{2SPACE}H{2SPACE}I{2SPACE}R{2SPACE}M
{4SPACE}"
1390 PRINT" {2DOWN,3SPACE}NM{CTRL-@}R{RIG
HT}X{WHITE,LIG.GREEN}" THENFN
EWAIT
1410 PRINT" NMUIFM{21SPACE}U*****I{
CTRL-@}Y{GREEN,F8,WHITE,LIG.GREEN}" RCLR
TAB(MID$GOTAB(END ODER SWAIT DWAIT O
DER PUDEF "S$") PUDEF
1430 PRINT" {3SPACE}FG{23SPACE}J*****
X{CTRL-@}T{GREEN,SHFT-SPCE,WHITE,LIG.GR
EEN}" RCLREND XWAIT
1450 PRINT" {2DOWN,RVSON}ESC{RVOFF}: RUEC
KSPRUNG IN EINE HOEHERE EBENE"
1460 PRINT" {DOWN,RVSON}CLEAR{RVOFF,SPACE
,RVSON}SHIFT{RVOFF}: ZEICHEN(SATZ) LOESC
HEN"
1470 PRINT" {RVSON}HOME{SPACE,RVOFF,6SPAC
E}: CURSOR IN LINKE OBERE ECKE"
1480 PRINT" {DOWN,RVSON}INS{RVOFF}: ZEICH
EN FUELLEN"
1490 PRINT" {DOWN,RVSON}F3{RVOFF}: EIN GE
WAELTES ZEICHEN WIRD KOPIERT"
1500 PRINT" {RVSON}F4{RVOFF}: SCHREIBMODU
S (AUSPROBIEREN DES ZS.)"
1510 PRINT" {RVSON}F5{RVOFF}: BEENDEN DES
PROGRAMMS"
1520 PRINT" {RVSON}F6{RVOFF}: ZEICHENSATZ
VON BAND ODER DISK LADEN"
1530 PRINT" {RVSON}F7{RVOFF}: ZEICHENSATZ
ABSPEICHERN"
1540 PRINT" {DOWN,11SPACE,RVSON}ZURUECK M
IT{SPACE,FLASHON}ESC{FLASHOFF,RVOFF}";
1550 GETKEYA$
1560 IFA$<>CHR$(27) THEN1550
1570 GOTO420
1580 SYS16122
1590 PRINT" {CLR,RVSON,3SPACE}Z E I C H E
N S A T Z{4SPACE}L A D E N{3SPACE}"
1600 PRINT" {3DOWN,RVSON,FLASHON}B{FLASHO
FF}AND ODER{SPACE,FLASHON}F{FLASHOFF}LOP
PY ?{RVOFF,2SPACE}";
1610 GETKEYA$
1620 IF A$="B" THENGN=1:PRINT"BAND":GOTO1
650
1630 IF A$="F" THENGN=8:PRINT"FLOPPY":GOT
01650
1640 IF A$=CHR$(27) THEN420:ELSE1610
1650 GOSUB 1830
1660 PRINT" {2DOWN,RVSON,FLASHON}SIND SIE
SICHER ?{FLASHOFF,RVOFF}"
1670 GETKEY A$: IFA$<>"J" THEN420
1680 LOAD F$,GN,1
1690 GOSUB1930
1700 IFFA=1 THENFA=0:GOTO150
1710 GOTO420
1720 SYS16122
1730 PRINT" {CLR,RVSON}Z E I C H E N S A
T Z{2SPACE}S P E I C H E R N"
1740 PRINT" {3DOWN,RVSON,FLASHON}B{FLASHO

```

```

FF}AND ODER{SPACE,FLASHON}F{FLASHOFF}LOP
PY ?{RVOFF,2SPACE}";
1750 GETKEYA$
1760 IF A$="B" THENGN=1:PRINT"BAND":GOTO1
790
1770 IF A$="F" THENGN=8:PRINT"FLOPPY":GOT
01790
1780 IF A$=CHR$(27) THEN420ELSE1750
1790 GOSUB1830
1800 POKE157,0:POKE158,20:POKE178,0:POKE
179,16:SYS61857
1810 GOSUB1930
1820 IFFF=1 THENRETURN:ELSEGOTO420
1830 IFGN=1 THENINPUT" {2DOWN,RVSON}FILENA
ME{RVOFF}";F$:GOTO1860
1840 INPUT" {2DOWN,RVSON}FILENAME{RVOFF }
(?=DIR) ";F$
1850 IFF$="?" THENPRINT" {CLR}":DIRECTORY:
GOTO1830
1860 IFLEN(A$)>16 THENF$=LEFT$(F$,16)
1870 POKE174,GN:POKE171,LEN(F$)
1880 FORT=1 TOLEN(F$)
1890 POKE1630+T,ASC(MID$(F$,T,1))
1900 NEXT
1910 POKE175,95:POKE176,6
1920 RETURN
1930 IF GN<>8 THENRETURN
1940 IF DS=0 THENRETURN
1950 PRINT" {2DOWN,FLASHON}FLOPPYFEHLER: {
FLASHOFF }"DS$
1960 PRINT" {DOWN,RVSON}WEITER MIT TASTE {
RVOFF}"
1970 GETKEY A$
1980 RETURN
1990 PRINT" {CLR,RVSON,3SPACE}D A R S T E
L L U N G S M O D U S{4SPACE}"
2000 FOR T=4 TO35:CHAR1,T,5,"$":NEXT
2010 FOR T=6 TO19:CHAR1,4,T,"_">{30SPACE}_"
:NEXT
2020 FOR T=4 TO35:CHAR1,T,20,"$":NEXT
2030 CHAR1,4,5,"_":CHAR1,35,5,"I"
2040 CHAR1,4,20,"J":CHAR1,35,20,"X"
2050 CHAR1,10,23,"ENDE MIT {SPACE,RVSON}R
ETURN{RVOFF}"
2060 POKE2021,19:POKE2022,6:POKE2023,5:P
OKE2024,34
2070 SYS16122:POKE65298,PEEK(65298) AND25
1
2080 OPEN1,0:PRINT" {HOME}";
2090 INPUT#1,A$
2100 PRINT" {2HOME}":CLOSE1:GOTO420
2110 PRINT" {CLR,RVSON,12SPACE}E{3SPACE}N
{3SPACE}D{3SPACE}E{15SPACE}";FF=0
2120 PRINT" {3DOWN}WOLLEN SIE DEN ZEICHEN
S. ABSAVEN ? ";
2130 GETKEYA$
2140 IF A$=CHR$(27) THEN420
2150 IF A$="J" THENPRINT"JA":FF=1:GOSUB17
20
2160 IF A$<>"N" THEN2130
2170 POKE65298,PEEK(65298) OR4:SYS16122
2180 COLOR 0,7,5:COLOR 4,13,2
2190 PRINTE$L{CLR,BLACK}"
2200 END
2210 DATA A2,7F,8A,9D,40,0D,A9,0A
2220 DATA 9D,40,09,CA,10,F4,60,A2
2230 DATA 07,BD,34,12,9D,00,10,CA
2240 DATA 10,F7,78,20,CE,F2,58,60
2250 DATA 78,A9,0D,BD,14,03,A9,3F
2260 DATA 8D,15,03,58,60,AD,09,FF
2270 DATA 29,02,F0,3D,AD,1C,FF,29

```

Listing «Matrix-Editor» für den C16 (Fortsetzung)

Grafikbeispiel für den C 16

Unsere »Blockgrafik« ist ein Beispiel dafür, daß Sie auch mit dem vorhandenen Zeichensatz fantastische Bilder erzeugen können.

Es muß nicht immer HiRes sein. Gerade beim C 16 ist die Verwendung der hochauflösenden Grafik (HiRes) kritisch, weil ein erheblicher Speicherbereich für die Programmierung verloren geht. Diese Blockgrafik (Bild) zeigt, wie es auch anders geht. Das untenstehende Listing ist das dazugehörige Basic-Programm. (Jörg Gerjets/kn)

```

2280 DATA 01,D0,2C,AD,1D,FF,C9,91
2290 DATA B0,19,A9,C0,8D,12,FF,8D
2300 DATA 19,FF,AD,1D,FF,C9,93,90
2310 DATA F9,A9,C4,8D,12,FF,8D,19
2320 DATA FF,D0,0C,C9,B1,90,08,A9
2330 DATA C0,8D,12,FF,8D,19,FF,AD
2340 DATA 09,FF,29,02,F0,03,20,60
2350 DATA CE,2C,D8,07,10,0E,AD,01
2360 DATA FD,8D,D4,07,10,06,20,95
2370 DATA EA,20,5B,EA,20,E4,E3,AD
2380 DATA 09,FF,29,02,F0,25,8D,09
2390 DATA FF,2C,0B,FF,A9,C3,50,18
2400 DATA 20,BF,CF,20,CD,CE,A5,FB
2410 DATA 48,A9,00,85,FB,08,58,20
2420 DATA 11,DB,28,68,85,FA,A9,62
2430 DATA 8D,0B,FF,4C,BE,FC,A6,2D
2440 DATA 86,5A,A4,2E,84,5B,20,D0
2450 DATA 3F,86,58,86,2D,84,59,84
2460 DATA 2E,A6,2B,86,5F,A4,2C,84
2470 DATA 60,20,D0,3F,86,2B,86,A6
2480 DATA 84,2C,84,A7,20,C0,8B,20
2490 DATA 18,8B,A5,A6,D0,02,C6,A7
2500 DATA C6,A6,A9,00,AB,91,A6,60
2510 DATA 84,AB,18,8A,69,00,AA,A5
2520 DATA AB,69,04,AB,60,A9,00,85
2530 DATA A6,85,AB,A9,D0,85,A7,A9
2540 DATA 10,85,A9,A2,04,A0,00,B1
2550 DATA A6,91,AB,C8,D0,F9,E6,A7
2560 DATA E6,A9,CA,D0,F0,60
2570 IF ER=4 THEN RESUME NEXT
2580 IF ER=35 THEN 2110
2590 IF ER=30 THEN 2110
2600 IF ER=5 THEN PRINT "2DOWN,FLASHON" FLO
PPY EINSCHALTEN !!!":FORT=1 TO 1000:NEXT:R
ESUME 1700
2610 PRINTERR$(ER) IN "EL
2620 GOTO 2110
2630 X=SX:Y=SY
2640 POKE2368+X,1,10
2650 POKE2368+W,241:X1=W
2660 RETURN
2670 POKE2368+X1,10
2680 POKE2368+W,241:X1=W
2690 GETKEYA$
2700 GOSUB 2800
2710 IF X>40 THEN X=0:Y=Y+40
2720 IF X<0 THEN X=40:Y=Y-40
2730 IF Y>120 THEN Y=120
2740 IF Y<0 THEN Y=0
2750 W=Y+X
2760 IF W>127 THEN W=127
2770 IFA$="*" THEN SYS16122:CHAR1,0,16,"{R
VSON,FLASHON}F1{FLASHOFF,RVOFF}":PRINT:P
RINT "{RVSON}F2{RVOFF}":FL=1
2780 IFA$="*" THEN SYS16128:CHAR1,0,16,"{R
VSON}F1{RVOFF}":PRINT:PRINT "{RVSON,FLASH
ON}F2{FLASHOFF,RVOFF}":FL=2
2790 RETURN
2800 IFA$="{DOWN}" OR A$="X" THEN Y=Y+40:RET
URN
2810 IFA$="{RIGHT}" OR A$="D" THEN X=X+1:RET
URN
2820 IFA$="{UP}" OR A$="E" THEN Y=Y-40:RETU
RN
2830 IFA$="{LEFT}" OR A$="S" THEN X=X-1:RETU
RN
2840 FA=0:RETURN
2850 CHAR1,32,4,"{RVSON,FLASHON}WORKING{
RVOFF,FLASHOFF}"
2860 RETURN

```

64'er

Listing »Matrix-Editor« für den C 16 (Schluß)

```

10 REM *****
15 REM **
20 REM ** 'BLOCKGRAPHIK'
25 REM **
30 REM *****
35 REM **
40 REM ** FUER C 16/116 & C 64
45 REM **
50 REM *****
55 REM ** JOERG GERJETS / 2980 NORDEN
60 REM **
65 REM ** (C) BY DPS MCMLXXXV
70 REM *****
75 REM ** BEI C 16/116 DURCH DRUECKEN
80 REM ** DER TASTEN 'e', '+', '-'
85 REM ** & '*' BEWEGUNG
90 REM ** [ESC] = ENDE
95 REM *****
100 PRINT "2HOME":PRINTCHR$(142):COLOR4,1:COLOR0,
2
110 PRINT "{CLR}";
120 PRINT "{BLACK,RVSON,4SPACE,BLUE}F 2 3 4{BLACK,2
SPACE}#####{4SPACE}↑ ↑ ↑{SPACE,BLUE,RVSON }2
4{SPACE,BLACK,2SPACE}"
130 PRINT "{RVSON,4SPACE,BLUE}U {2SPACE,BROWN,3SPACE
,BLUE,SPACE,RVOFF,CYAN}UCCCI{BLACK}UCCCCI{LIG,BLU
E}Z{BLACK}Z{RVSON}UURRRRRR{BLUE,RVSON}0{BROWN }B{S
PACE,BLUE}6{BLACK,2SPACE}"
140 PRINT "{RVSON,4SPACE,BLUE}E 1{BROWN }U{SPACE,BL
UE}5{RVOFF,CYAN}B{RED}B{SPACE,GREEN}B{CYAN}B{BLACK
}B{RVSON,ORANGE }M-3{SPACE,RVOFF,BLACK}B{BROWN}U{
BLACK,RVSON}B+++++B{BLUE,RVSON,SPACE,BROWN }B{SPA
CE,BLUE,SPACE,BLACK,2SPACE}"
150 PRINT "{RVSON,4SPACE,BLUE}L {2SPACE,BROWN,2SPACE
}B{BLUE,SPACE,RVOFF,CYAN}B{SPACE,GREEN}B{YELLOW}B{
CYAN}B{BLACK}BCCCCC{BROWN }B{BLACK,RVSON}B+++++B
{BLUE,RVSON}HT{BROWN,2SPACE,BLUE}B{BLACK,2SPACE}"
160 PRINT "{RVSON,4SPACE,BLUE,2SPACE}0 7 6{RVOFF,CY
AN}BCCCC{RVSON,RED}B30.37{RVOFF}B{BROWN}B{BLACK,R
VSON}BEEEEEE{BLUE,RVSON,2SPACE}10{SPACE,BLACK,2SP
ACE}"
170 PRINT "B{ORANGE}B{RVSON}YYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYY{RVOFF}B{BLACK}B"
180 PRINT "{ORANGE,SPACE,RVSON,SPACE,RVOFF,17SPACE,
RVSON,2SPACE,RVOFF,4SPACE,BLACK,11SPACE}. {SPACE,OR
ANGE,RVSON }"
190 PRINT "B{RVSON,SPACE,RVOFF,5SPACE,BLACK,RVSON}B
{RVOFF,ORANGE,11SPACE,RVSON,2SPACE,RVOFF,2SPACE,BL
ACK,5SPACE},{9SPACE,ORANGE,RVSON,SPACE,RVOFF}B"
200 PRINT "T{RVSON,SPACE,RVOFF,4SPACE,BLACK,SPACE,R
VSON,2SPACE}YYY B{RVOFF,ORANGE,5SPACE,RVSON,2SPACE
,RVOFF,13SPACE,ORANGE}B{3SPACE,ORANGE,RVSON,SPACE,
RVOFF}B"
210 PRINT "{RVSON}B{SPACE,RVOFF,6SPACE,BLACK}B{RVSD
N}B{RVOFF}B{RVSON}B{RVOFF}B{ORANGE,5SPACE,RVSON,2
SPACE,RVOFF,13SPACE,RED,RVSON}B{RVOFF,3SPACE,ORANG
E,RVSON,SPACE,RVOFF}B"
220 PRINT "{RVSON}B{SPACE,RVOFF,17SPACE,RVSON,2SPAC
E,RVOFF,4SPACE,BLACK}B{8SPACE,RED,RVSON}B{RVOFF,3S
PACE,ORANGE,RVSON,SPACE,RVOFF}B"
230 PRINT "T{RVSON,SPACE,RVOFF,13SPACE,BLACK}B{ORAN
GE,3SPACE,RVSON,2SPACE,RVOFF,3SPACE,BLACK,RVSON}B{
SPACE,RVOFF,8SPACE,RED,RVSON}B{RVOFF,3SPACE,ORANGE
,RVSON,SPACE,RVOFF}B"

```

Listing. Basic-Programm zur Blockgrafik.

Bitte beachten Sie die Eingabehinweise auf Seite 76.