Pseudo-Sprites auf dem VC 20

Der VC 20 kennt von Haus aus leider nicht die freibeweglichen Grafikobjekte des C 64, die sogenannten Sprites. Das bedeutet aber nicht, daß man auf die Vorteile der Sprites oder MOBs gänzlich verzichten muß.

Das Programm ist für den VC 20 mit 8 KByte Speichererweiterung konzipiert. Es läuft jedoch mit einigen Änderungen auch bei nur 3 KByte Speichererweiterung.

Vor dem Eintippen oder Laden muß man POKE 44,32:POKE 8192,0: NEW eingeben, womit der Basic-Anfang im Speicher auf die Adresse dezimal 8193 ($2001) erhöht wird. Somit ergibt sich folgende Speicheraufteilung:

4096 — 4607 Bildschirm
4608 — 8191 frei
8192 — 16383 (bei + 8 KByte) Basic-Programmspeicher
8192 — 24575 (bei + 16 KByte) Basic-Programmspeicher
8192 — 32767 (bei + 24 KByte) Basic-Programmspeicher

Der freie Bereich wird nun vollständig von dem Maschinenspracheprogramm gebraucht. Die Aufteilung des Speicherraums ist die folgende:

4608 — 5119	Sprite-Control-Block (SCB), wird später erklärt
5120 — 6143	freidefinierbarer Zeichensatz, die ersten 128 Zeichen, stehen zur freien Verfügung
6144 — 7167	freidefinierbarer Zeichensatz, die zweiten 128 Zeichen, werden vom Programm zur Erstellung der 9 Sprites gebraucht und stehen somit nicht zur freien Verfügung
7168 — 8191	Maschinenspracheprogramm, Beschreibung siehe Text

Die Pseudo-Sprites sollten eine Auflösung von 16 x 16 Punkten haben, das sind 256 Punkte oder 4 Zeichen im freidefinierbaren Zeichensatz (Bild 1). Damit aber ein 16 x 16 Punkte großes Zeichen jede Position auf dem Bildschirm einnehmen kann, braucht man eine 24 x 24 Punkte große Umdefinier-Matrix, in die das Zeichen hineinkopiert wird. Das Aussehen dieser Umdefinier-Matrix ist in Bild 2 zu sehen. Das Programm übernimmt nun die Aufgabe, das Zeichen in die Umdefinier-Matrix zu kopieren (Bild 3), in die richtige X-Position zu schieben (Bild 4), und dasselbe mit der Y-Position zu tun.

Außerdem werden die Zeichen, die später auf dem Bildschirm von den Sprites verdeckt werden, mit in die Umdefinier-Matrix hineinkopiert. So entsteht der Eindruck, daß die Sprites wirklich über die Zeichen wandern. Beim späteren Löschen werden die verdeckten Zeichen wieder hergestellt. Wie funktioniert das nun?

Im Speicher ab dezimal 4608 ist 9mal (für jedes Sprite einer) der sogenannte Sprite-Control-Block (SCB) eingerichtet. Er hat die Aufgabe, die momentane X- und Y-Position, die Farbe des Sprites, den Bildschirmmodus (gesetzt/gelöscht) des Sprites, die durch die Umdefinier-Matrix verdeckten 9 Zeichen und Farben zwischenzuspeichern:

Byte 0 — 8	Bildschirmcode der verdeckten Zeichen
Byte 9 — 17	Farbcode der verdeckten Zeichen
Byte 18	X-Position des Sprites
Byte 19	Y-Position des Sprites
Byte 20	Farbe des Sprites
Byte 21	Bildschirmmodus (gesetzt= $00/gelöscht=$FF)

Die Basisadresse des SCB errechnet sich somit aus der Formel Basisadresse = 4608 + Spritenummer x 22. Das Zwischenspeichern und die Auswertung der Parameter übernimmt natürlich das Maschinenprogramm. Über den SCB werden auch im nachfolgend beschriebenen Programm »Sprite-Definer« die Sprites initialisiert und deren Farbe festgelegt. Da nur die oberen 128 Zeichen des Zeichensatzes für die Sprites verwendet werden, hat man eine ausreichende Anzahl von noch frei definierbaren Zeichen, nämlich genau 128, zur Verfügung. Außerdem kommen jeweils 13 Zeichen, nämlich 4 für das Sprite und 9 für die Umdefinier-Matrix hinzu, wenn man auf ein Sprite verzichtet. Die Matrixen werden im Speicher so abgelegt:

128 — 131	Grundmatrix Sprite 0
132 — 140	Umdef.-Matrix Sprite 0
141 — 144	Grundmatrix Sprite 1
145 — 153	Umdef.-Matrix Sprite 1

Konkret wird das Programm (Listing 1) nun folgendermaßen bedient: Vor dem Laden oder Eingeben wird POKE 44,32:POKE 8192,0:NEW eingetippt. Ist nun das Maschinen-

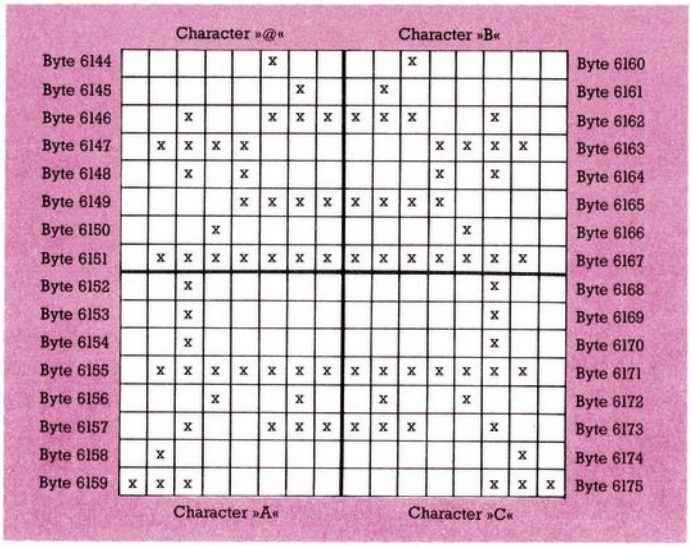

Bild 1. Die Speicheraufteilung in der Grundmatrix des Sprites Nummer 0. Das höchstwertige Bit eines Bytes steht links.

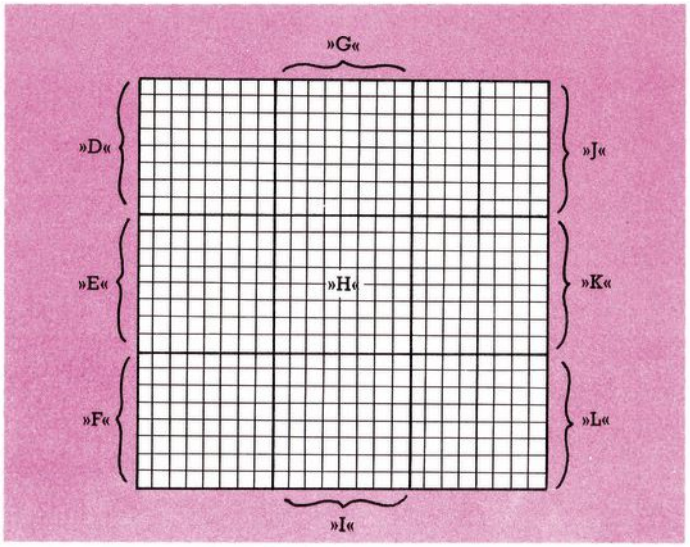

Bild 2. Die Undefinier-Matrix hat 24 x 24 Punkte und ist notwendig, um Sprites auch punktweise verschieben zu können

spracheprogramm im Speicher, kann es mit einem Monitorprogramm auch noch einmal abgespeichert werden. Später muß man es nur noch mit LOAD »name«, 1,1 laden.

Die Bedienung:
Sind die Sprites definiert, muß dem Maschinenspracheprogramm mitgeteilt werden, wo der Zeichensatz liegt, den es verwalten soll. Das geschieht mit den Befehlen POKE 677, Lowbyte:POKE 678, Highbyte, in unserem Fall also POKE 677,0:POKE 678,20, da der Zeichensatz auf der Adresse 5120 beginnt.

Soll nun ein Sprite auf den Bildschirm, muß zuerst einmal in Adresse 683 die Spritenummer gePOKEt werden (Achtung, keine Zahl über 8 angeben, da sich das Programm dann selbst zerstören könnte). Schließlich werden in Adresse 673 die X-Koordinate (maximal 159) und 674 die Y-Koordinate (maximal 167) gesetzt. Dann kann das Programm mit SYS 8021 sofort aufgerufen und auf dem Bildschirm das Sprite betrachtet werden, vorausgesetzt man hat vorher mit POKE 36869,205 auf den freidefinierbaren Zeichensatz geschaltet.

Wird nun das Sprite auf eine andere Position gesetzt, so verschwindet es vollständig von der alten Position, und die Zeichen, die auf diesem Platz waren, erscheinen wieder mit ihrer alten Farbe. Will man aber das Sprite ganz vom Bildschirm löschen, POKEt man wieder in 683 die Spritenummer und ruft das Maschinenspracheprogramm diesmal mit SYS 8099 auf. PRINT »CLR/HOME« sollte man nicht verwenden, da im SCB noch die alten Bildschirmzeichen gespeichert sind und beim nächsten Setzen wieder auf ihren alten Plätzen auf dem Bildschirm erscheinen würden.

Der Sprite-Generator

Nun zum Programm »Sprite-Definer« (Listing 2).

Dieses Programm ist ein Sprite-Generator in Basic, der bei der Erstellung von Sprites recht hilfreich sein kann. Das Programm verdeutlicht auch, wie die Definition der Sprites und die Bedienung des Maschinenspracheprogramms erfolgt.

Obwohl sich das Programm fast von selbst erklärt, hier doch einige kurze Erläuterungen:

Startet man das Programm mit RUN, erscheint als erstes die Begrüßung und die Aufforderung »Bitte warten!«. Das Programm kopiert nämlich jetzt den Zeichensatz aus dem ROM ins RAM, was in Basic naturgemäß etwas dauert.

Jedesmal, wenn man in einem Menüteil eine Eingabe gemacht hat, wird man »Richtig?« gefragt. Tippt man hier für N (Nein), so kann die Eingabe wiederholt werden. Drückt man aber den Linkspfeil, so kommt man wieder ins Hauptmenü.

Die Tastenbelegung im Editiermodus:

Cursor-Tasten	Cursor-Bewegungen
Leertaste	Punkt setzen
Delete-Taste	Punkt löschen
CLR/HOME	Gitter löschen
RETURN	Modus beenden mit Änderung des Sprites, vorher aber Abfrage
Linkspfeil	Modus beenden, aber ohne Änderung des Sprites
I-Taste	Sprite invertieren

Bei der Funktion »Weiter« kommt man in ein zweites Menü, das weitere Funktionen zur Verfügung stellt. Aus diesem Menü gelangt man mit »zurück« wieder ins Hauptmenü. Beim Speichern werden die Sprites als reiner Speicherauszug auf Kassette gebracht, so daß das Laden im Prinzip auch mit LOAD »name«,1,1 möglich ist.

Sicherlich kann das Maschinenspracheprogramm noch weiter verbessert werden. So wäre zum Beispiel eine Spritesteuerung per Interrupt durchaus denkbar. Leider funktioniert das Maschinenprogramm nicht mit den üblichen Grafikmodulen, da diese den Bildschirminhalt auch mit dem freidefinierbaren Zeichensatz aufbauen. Sollen Sprites auch miteinander oder übereinander dargestellt werden, dann muß das Setzen und Löschen nach folgender Reihenfolge durchgeführt werden, da es sonst zu Schwierigkeiten mit dem SCB kommen kann:
Sprite 0 setzen, Sprite 1 setzen,..., Sprite n setzen. Hiernach die Berechnungen für die neuen Positionen durchführen.
Sprite n löschen,.Sprite n-1 löschen,..., Sprite 0 löschen. Danach Vorgang von oben wiederholen.

Noch eins zum »Sprite-Definer«: Die erste REM-Zeile muß auf jeden Fall mit 16 Sternchen eingegeben werden, da sich das Programm später mit POKEs selbst verändert und andernfalls, wäre die REM-Zeile kürzer, die folgende Zeile in Mitleidenschaft ziehen würde. Doch nun wünsche ich allen, die das Programm eintippen, viel Spaß und vielleicht ein bißchen C 64-Feeling.

(Markus Leberecht/ev)

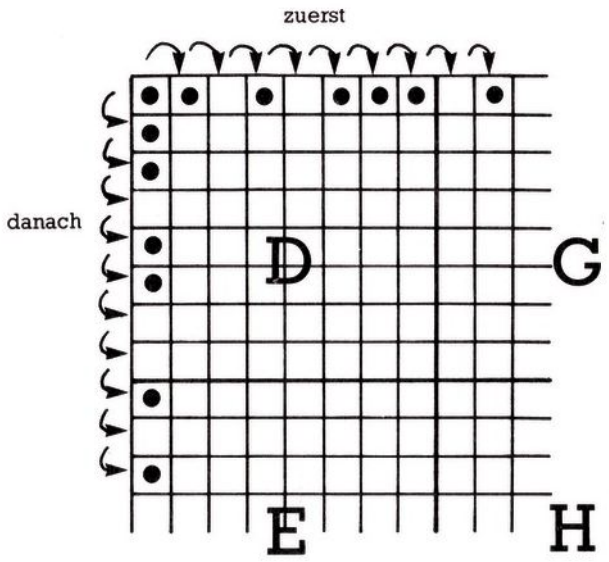

Bild 4. Zur punktgenauen Justierung wird das Sprite nach dem Kopieren in die Umdefinier-Matrix noch horizontal und vertikal verschoben

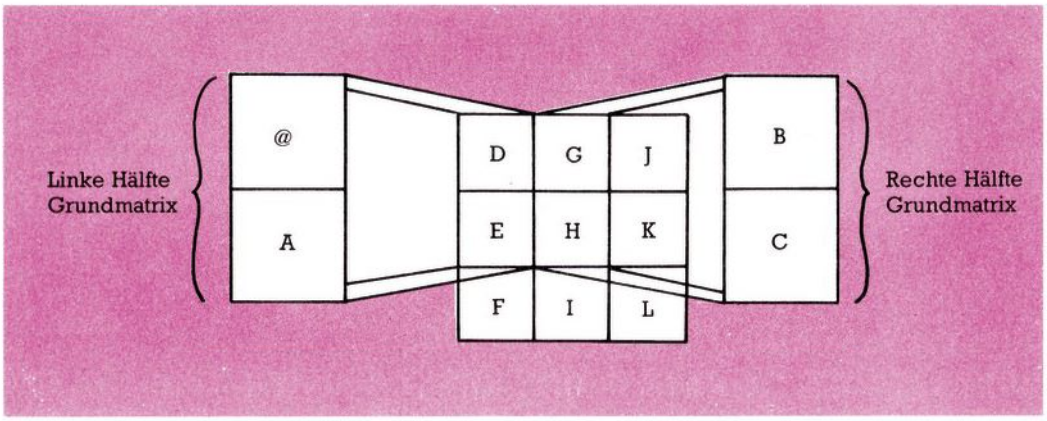

Bild 3. Hineinkopieren der Grundmatrix in die Umdefinier-Matrix

```
1 REM PSEUDOSPRITES
2 REM FUER VC 20
3 REM (DATA-LADER)
4 REM
5 REM SPRITES VON 7168 BIS 8192
6 REM
7 REM 'SAVE' VOR 'RUN' !
8 REM
1001 DATA024,032,121,029,173,165,002,133
1002 DATA025,173,166,002,133,026,142,169
1003 DATA002,173,163,002,042,046,169,002
1004 DATA024,042,046,169,002,024,042,046
1005 DATA169,002,024,101,025,133,025,165
1006 DATA026,109,169,002,133,026,024,165
1007 DATA025,105,032,133,029,165,026,105
1008 DATA000,133,030,173,161,002,041,007
1009 DATA141,167,002,173,161,002,024,074
1010 DATA074,074,141,161,002,024,173,162
1011 DATA002,041,007,141,168,002,173,162
1012 DATA002,074,074,074,141,162,002,032
1013 DATA121,029,162,072,145,029,200,202
1014 DATA208,250,024,160,000,024,165,029
1015 DATA109,168,002,133,031,165,030,105
1016 DATA000,133,032,177,025,145,031,200
1017 DATA192,016,208,247,024,165,031,105
1018 DATA024,133,031,165,032,105,000,133
1019 DATA032,024,165,025,105,016,133,027
1020 DATA165,026,105,000,133,028,160,000
1021 DATA024,177,027,145,031,200,192,016
1022 DATA208,247,173,167,002,201,000,240
1023 DATA060,234,169,024,141,170,002,165
1024 DATA029,141,118,029,165,030,141,119
1025 DATA029,032,121,029,024,032,117,029
1026 DATA162,024,032,117,029,162,048,032
1027 DATA117,029,024,173,118,029,105,001
1028 DATA141,118,029,173,119,029,105,000
1029 DATA141,119,029,206,170,002,208,217
1030 DATA206,167,002,208,197,032,148,029
1031 DATA024,032,121,029,173,163,002,105
1032 DATA004,141,163,002,032,126,029,032
1033 DATA137,029,238,163,002,238,162,002
1034 DATA032,126,029,032,137,029,238,163
1035 DATA002,238,162,002,032,126,029,032
1036 DATA137,029,056,173,162,002,233,002
1037 DATA141,162,002,238,161,002,238,163
1038 DATA002,032,126,029,032,137,029,238
1039 DATA163,002,238,162,002,032,126,029
1040 DATA032,137,029,238,163,002,238,162
1041 DATA002,032,126,029,032,137,029,056
1042 DATA173,162,002,233,002,141,162,002
1043 DATA238,161,002,238,163,002,032,126
1044 DATA029,032,137,029,238,163,002,238
1045 DATA162,002,032,126,029,032,137,029
1046 DATA238,163,002,238,162,002,032,126
1047 DATA029,032,137,029,096,126,120,007
1048 DATA096,169,000,170,168,096,024,174
1049 DATA162,002,172,161,002,032,240,255
1050 DATA096,024,173,163,002,174,164,002
1051 DATA032,161,234,096,024,032,121,029
1052 DATA133,251,173,136,002,133,252,174
1053 DATA162,002,224,000,240,016,024,165
1054 DATA251,105,022,133,251,165,252,105
1055 DATA000,133,252,202,208,240,024,165
1056 DATA251,109,161,002,133,251,165,252
1057 DATA105,000,133,252,024,032,121,029
1058 DATA169,000,133,253,169,148,133,254
1059 DATA174,162,002,224,000,240,016,024
1060 DATA165,253,105,022,133,253,165,254
1061 DATA105,000,133,254,202,208,240,024
1062 DATA165,253,109,161,002,133,253,165
1063 DATA254,105,000,133,254,024,032,121
1064 DATA029,165,029,133,031,165,030,133
1065 DATA032,024,032,121,029,162,000,173
1066 DATA165,002,133,027,173,166,002,133
1067 DATA028,189,098,030,048,075,168,177
1068 DATA251,157,060,003,072,169,000,141
1069 DATA169,002,104,010,046,169,002,010
1070 DATA046,169,002,010,046,169,002,101
1071 DATA027,133,027,173,169,002,101,028
1072 DATA133,028,160,000,177,027,017,031
1073 DATA145,031,200,192,008,208,245,024
1074 DATA165,031,105,008,133,031,165,032
1075 DATA105,000,133,032,189,098,030,168
1076 DATA177,253,157,069,003,232,076,007
1077 DATA030,096,000,022,044,001,023,045
1078 DATA002,024,046,255,024,032,121,029
1079 DATA173,136,002,133,252,152,133,251
1080 DATA169,000,133,253,169,148,133,254
1081 DATA173,162,002,240,017,168,024,165
1082 DATA251,105,022,133,251,165,252,105
1083 DATA000,133,252,136,208,240,024,165
1084 DATA251,109,161,002,133,251,165,252
1085 DATA105,000,133,252,173,162,002,240
1086 DATA017,168,024,165,253,105,022,133
1087 DATA253,165,254,105,000,133,254,136
1088 DATA208,240,024,165,253,109,161,002
1089 DATA133,253,165,254,105,000,133,254
1090 DATA024,032,121,029,189,098,030,048
1091 DATA015,168,189,060,003,145,251,189
1092 DATA069,003,145,253,232,076,204,030
1093 DATA096,024,032,121,029,173,161,002
1094 DATA074,074,074,141,161,002,173,162
1095 DATA002,074,074,074,141,162,002,076
1096 DATA108,030,024,032,121,029,177,251
1097 DATA153,060,003,200,192,022,208,246
1098 DATA096,024,032,121,029,185,060,003
1099 DATA145,251,200,192,022,208,246,096
1100 DATA024,032,121,029,133,251,169,018
1101 DATA133,252,169,128,141,163,002,173
1102 DATA171,002,201,000,240,038,024,165
1103 DATA251,105,022,133,251,165,252,105
1104 DATA000,133,252,232,236,171,002,208
1105 DATA237,024,032,121,029,024,173,163
1106 DATA002,105,013,141,163,002,232,236
1107 DATA171,002,208,242,096,024,032,024
1108 DATA031,024,032,121,029,032,250,030
1109 DATA173,161,002,072,173,162,002,072
1110 DATA173,081,003,048,015,173,078,003
1111 DATA141,161,002,173,079,003,141,162
1112 DATA002,032,225,030,024,032,121,029
1113 DATA104,141,162,002,141,079,003,104
1114 DATA141,161,002,141,078,003,173,080
1115 DATA003,141,164,002,032,000,028,169
1116 DATA000,141,081,003,032,024,031,032
1117 DATA009,031,096,032,024,031,024,032
1118 DATA121,029,032,250,030,173,081,003
1119 DATA048,030,173,078,003,141,161,002
1120 DATA173,079,003,141,162,002,032,225
1121 DATA030,024,032,121,029,169,255,141
1122 DATA081,003,032,024,031,032,009,031
1123 DATA096,234,234,234,234,234,234,234
1124 DATA234,234,234,234,234,234,234,234
1125 DATA234,234,234,234,234,234,234,234
1126 DATA234,234,234,234,234,234,234,234
1127 DATA234,234,234,234,234,234,234,234
1128 DATA234,234,234,234,234,234,234,234
1130 S=0:FORI=7168TO8191:READD:S=S+D
1131 POKEI,D:NEXT
1132 IFS<>111729THENPRINT"■FEHLER!■"
READY.
```

Listing 1. Das Maschinenprogramm zur Sprite-Kontrolle als Basic-Lader

```
1 REM SPRITE-GENERATOR
2 REM
10 POKE36879,25:PRINT"■";:POKE157,128
15 POKE677,0:POKE678,20
20 PRINT"■■    SPRITE-DEFINER     "
30 PRINT"■■■■   FUER VC=20    "
50 PRINT"■■   MARKUS LEBERECHT■"
60 PRINT"■■■■■■■BITTE WARTEN!"
70 AZ=32768:NZ=5120:FORA=0TO2047:POKENZ+
A,PEEK(AZ+A):NEXT
80 PRINTCHR$(8);"■";
90 PRINT"■■          MENUE          ■"
100 PRINT"■■■F1■ SPRITE ERSTELLEN
110 PRINT"■■■F2■ SPRITE EDITIEREN
120 PRINT"■■■F3■ SPRITE LOESCHEN
130 PRINT"■■■F4■ SP.-FARBE AENDERN
150 PRINT"■■■F5■ SPRITE(S) SAVEN
160 PRINT"■■■F6■ SPRITE(S) LADEN
170 PRINT"■■■F7■ SPRITE KOPIEREN
180 PRINT"■■■F8■ ■WEITER
190 GOSUB61000
200 F$="■■■■■■■■":FORA=1TO8:IFMID$(F$,A,
1)=A$THEN220
210 NEXTA:GOTO190
220 ONAGOTO290,320,450,520,570,690,750
230 PRINT"■■■          MENUE          ■"
240 PRINT"■■■■F1■ VERIFIZIEREN
250 PRINT"■■■■F2■ INITIALISIEREN
251 PRINT"■■■■F3■ SPRITE-DEMO
260 PRINT"■■■■F7■ ■ZURUECK■
270 PRINT"■■■■F8■ BEENDEN
280 GOSUB61000
285 F$="■■■■■■■■":FORA=1TO8:IFMID$(F$,A,
1)=A$THEN287
286 NEXTA:GOTO280
287 ONAGOTO810,840,880,280,280,280,80,94
0
288 GOTO280
290 PRINT"■■SPRITENUMMER:";
300 INPUTS%:PRINT"■RICHTIG?":GOSUB61000:
IFA$="N"THENPRINT"■■";:GOTO290
302 IFA$="←"THENPOKE36879,25:GOTO80
305 PRINT"■";:GOSUB50000:IFA$="←"THEN80
310 GOSUB40000:GOTO80
320 POKE36879,24:PRINT"■■SPRITENUMMER";:
INPUTS%
330 PRINT"■RICHTIG?":GOSUB61000:IFA$="N"
THEN320
335 IFA$="←"THENPOKE36879,25:GOTO80
340 BA%=NZ+1024+S%*104
350 PRINT"■■■■■■■■";:FORA=0TO15:FORB=0TO7
360 IFPEEK(BA%+A)AND(2↑(7-B))THENPRINT"*
";:GOTO380
370 PRINT" ";
380 NEXTB:PRINT:PRINT"■■■";:NEXTA
390 PRINT"■■■■■■■■■■■■■■■■";:FORA=0TO15:F
ORB=0TO7
400 IFPEEK(BA%+16+A)AND(2↑(7-B))THENPRIN
T"*";:GOTO420
410 PRINT" ";
420 NEXTB:PRINT:PRINT"■■■■■■■■■■■■■■■■";:NEXT
A
430 GOSUB50000:IFA$="←"THENPOKE36879,25:
GOTO80
440 GOSUB40000:POKE36879,25:GOTO80
450 POKE36879,26:PRINT"■■SPRITENUMMER:":
PRINT"■(9=KEINS)
460 INPUTS%:PRINT"■RICHTIG?":GOSUB61000
470 IFA$="N"OR(S%>9ORS%<0)THEN450
480 IFS%=9THENPOKE36879,25:GOTO80
490 PRINT"■■!! LOESCHE !!"
500 BA%=NZ+1024+S%*104
510 FORA=0TO31:POKEBA%+A,0:NEXT:POKE3687
9,25:GOTO80
520 POKE36879,27:PRINT"■■SPRITEFARBEN:"
530 FORA=0TO8:PRINT"■SPRITE NR."A"---"PE
EK(4608+A*22+20)AND15:NEXTA
540 INPUT"■WELCHES (9=EXIT)";S%:IFS%<0OR
S%>8THENPOKE36879,25:GOTO80
550 INPUT"FARBE";F%:IFF%<0ORF%>15THENPOK
E36879,25:GOTO80
560 POKE4608+22*S%+20,F%:GOTO520
570 POKE36879,28:INPUT"■■ANF.-SPRITE:";S
A%:IFSA%<0ORSA%>8THEN570
580 INPUT"■ENDSPRITE:    ■■■";SE%:IFSE%<0
ORSE%>8THENPRINT"■■";:GOTO580
585 PRINT"■RICHTIG?":GOSUB61000:IFA$="N"
THEN570
587 IFA$="←"THENPOKE36879,25:GOTO80
590 IFSA%>SE%THEN570
600 ST%=NZ+1024+SA%*104:EN%=NZ+1024+(SE%
+1)*104
610 POKE185,3:INPUT"■FILENAME:";FI$:FI$=
LEFT$(FI$,16):IFFI$=""THENPRINT"■■";:GOT
0610
620 FORA=1TOLEN(FI$):POKE8192+5+A,ASC(MI
D$(FI$,A,1)):NEXT:POKE183,LEN(FI$)
640 POKE186,1
650 POKE187,6:POKE188,32
660 POKE193,ST%-256*INT(ST%/256):POKE194
,ST%/256
670 POKE174,EN%-256*INT(EN%/256):POKE175
,EN%/256
680 SYS63106:POKE36879,25:GOTO80
690 POKE36879,29:INPUT"■■FILENAME:";FI$
700 PRINT"■RICHTIG?":GOSUB61000:IFA$="N"
THEN690
705 IFA$="←"THENPOKE36879,25:GOTO80
710 IFLEN(FI$)=0THEN730
720 FORA=1TOLEN(FI$):POKE8192+5+A,ASC(MI
D$(FI$,A,1)):NEXT
730 POKE183,LEN(FI$):POKE186,1:POKE187,6
:POKE188,32
740 SYS62786:POKE36879,25:GOTO80
750 POKE36879,30:INPUT"■■QUELLE=";S1%:IF
S1%<0ORS1%>8THENPOKE36879,25:GOTO80
760 INPUT"■ZIEL=";S2%:IFS2%<0ORS2%>8THEN
POKE36879,25:GOTO80
770 PRINT"■RICHTIG?":GOSUB61000:IFA$="N"
THEN750
775 IFA$="←"THENPOKE36879,25:GOTO80
780 PRINT"■■!! KOPIERE !!"
790 S1%=NZ+1024+104*S1%:S2%=NZ+1024+104*
S2%
800 FORA=0TO31:POKES2%+A,PEEK(S1%+A):NEX
T:POKE36879,25:GOTO80
810 POKE36879,31:PRINT"■■BESTAETIGEN":GO
SUB61000
815 IFA$="←"THENPOKE36879,25:GOTO80
820 PRINT:VERIFY"",1,1:PRINT"■■■TASTE":G
OSUB61000
830 POKE36879,25:GOTO80
840 POKE36879,24:PRINT"■■■■BESTAETIGEN":
GOSUB61000:IFA$="←"THENPOKE36879,25:GOTO
80
845 PRINT"■■!! INITIALISIERE !!"
850 FORA=0TO9:PRINT"■SPRITE"A:FORB=0TO31
:POKENZ+1024+A*104+B,0:NEXTB
860 POKE4608+A*22+20,0
870 FORB=0TO17:POKE4608+A*22+B,32:NEXTB,
A:POKE36879,25:GOTO80
880 INPUT"■■SPRITENUMMER:";S%:PRINT"■RIC
HTIG?":GOSUB61000
890 IFA$="N"THEN880
```

Listing 2.
Der Sprite-Generator

```
891 IFA$="←"THEN80
900 PRINT"[CLR][DOWN]...SPRITE-DEMO!"
910 POKE36869,205
920 FORA=-127TO127:POKE673,ABS(A):POKE674,ABS(A):POKE683,S%:SYS8021
930 NEXTA:POKE683,S%:SYS8099:POKE36869,192:GOTO80
940 PRINT"[CLR]WIRKLICH?":GOSUB61000
950 IFA$="J"THENNEW
960 GOTO80
999 END
```

Listing 2. Der Sprite-Generator (Schluß)

```
40000 PRINT"[HOME]!! WERTE AUS !!";
40005 FORA=0TO15:BB%=0:FORB=0TO7
40010 BB%=BB%-((PEEK(B%+A*22+B)AND127)=81)*2↑(7-B):NEXTB
40020 POKENZ+128*8+S%*104+A,BB%:NEXTA
40030 B%=B%+8
40040 FORA=0TO15:BB%=0:FORB=0TO7
40050 BB%=BB%-((PEEK(B%+A*22+B)AND127)=81)*2↑(7-B):NEXTB
40060 POKENZ+128*8+S%*104++16+A,BB%:NEXTA
40070 RETURN
50000 PRINT"[CLR]SPRITENR."S%
50010 PRINT"[DOWN]┌┬┬┬┬┬┬┬┬┬┬┬┬┬┬┬┐"
50020 FORA=0TO15:PRINT"[RIGHT]├                ┤":NEXT:PRINT"[RIGHT]└┴┴┴┴┴┴┴┴┴┴┴┴┴┴┴┘"
50030 B%=4096+22*3+3:X%=0:Y%=0
50035 D%=B%+Y%*22+X%
50040 IF(PEEK(D%)AND128)=0THENPOKED%,PEEK(D%)OR128:POKED%+33792,6:GOTO50060
50050 IF(PEEK(D%)AND128)THENPOKED%,PEEK(D%)AND127:POKED%+33792,6
50060 AX%=X%:AY%=Y%:GOSUB61000
50070 IFA$="[DOWN]"ANDY%<15THENY%=Y%+1:GOTO50200
50080 IFA$="[UP]"ANDY%>0THENY%=Y%-1:GOTO50200
50090 IFA$="[RIGHT]"ANDX%<15THENX%=X%+1:GOTO50200
50100 IFA$="[LEFT]"ANDX%>0THENX%=X%-1:GOTO50200
50110 IFA$=" "THENPOKED%,209:POKED%+33792,6:GOTO50060
50120 IFA$=CHR$(20)THENPOKED%,160:POKED%+33792,6:GOTO50060
50130 IFA$="[UP]"THENPRINT"[HOME][DOWN]...";:FORA=0TO15:PRINT"[RIGHT]                ":NEXT:GOTO50030
50140 IFA$="I"THENGOSUB60000:GOTO50035
50150 IFA$=CHR$(13)THEN50240
50160 IFA$="←"THENRETURN
50200 D%=B%+AY%*22+AX%
50210 IF(PEEK(D%)AND128)=0THENPOKED%,PEEK(D%)OR128:GOTO50230
50220 IF(PEEK(D%)AND128)THENPOKED%,PEEK(D%)AND127
50230 GOTO50035
50240 PRINT"[HOME][DOWN]...WIRKLICH(J/N)?";
50250 GOSUB61000:IFA$="J"THENRETURN
50260 PRINT"[LEFT]...[HOME]";:GOTO50000
60000 FORX=0TO15:FORY=0TO15:DD%=B%+22*Y+X
60010 IF(PEEK(DD%)AND127)=81THENPOKEDD%,32:POKEDD%+33792,6:NEXT:NEXT:RETURN
60020 IF(PEEK(DD%)AND127)=32THENPOKEDD%,81:POKEDD%+33792,6:NEXT:NEXT:RETURN
61000 POKE198,0:WAIT198,1:GETA$:RETURN
READY.
```