

Neues vom SMON

Diese Erweiterung stellt elf weitere Befehle zur Verfügung. So läßt sich der Monitor zum Beispiel frei im Speicher verschieben und Sprites oder Zeichensätze können sehr einfach erstellt und geändert werden.

Um die Befehlserweiterung zu initialisieren, geht man folgendermaßen vor:

1. SMON absolut laden.
2. Den Basic-Lader (siehe Listing) eintippen und abspeichern.
3. Nach dem Start des Laders die Startadresse (dezimal) Ihrer SMON-Version eingeben:
zum Beispiel 49152 (= \$C000).
4. Den erweiterten SMON zum Beispiel mit "SMONEX" Startadresse Endadresse abspeichern.

Die neuen Routinen werden, genau wie die meisten bereits vorhandenen, durch einen Buchstaben, zum Teil gefolgt von Adressenangaben, aufgerufen. Bei den ersten drei Ausgabebefehlen kann der Speicherinhalt durch Überschreiben der Zeile geändert werden.

Z 4000 (4100) (Zeichendaten)

gibt den Speicherinhalt von \$4000 (bis \$40FF) folgendermaßen aus: Jeweils ein Byte pro Zeile wird in 8-Bit-Form dargestellt. Dabei ist ein »*« ein gesetztes, ein ».« dagegen ein nicht gesetztes Bit. Die beiden Zeichen sind willkürlich gewählt und können durch Überschreiben der Speicherzellen \$xE65, \$xE2D (Bit = 1) und \$xE69, \$xE30 (Bit = 0) in den Bildschirm-Code (!) der gewünschten Zeichen geändert werden.

Die Anwendung dieses Befehls liegt beispielsweise in der gezielten und anschaulichen Beeinflussung bestimmter Steuerbits in VIC, CIA, etc. Andererseits lassen sich — besonders in Verbindung mit dem Kommando »Q« — Zeichendaten leicht modifizieren.

H 4000 (4100)

entspricht dem Befehl »Z« mit dem Unterschied, daß jeweils drei Byte pro Zeile ausgegeben werden. Das entspricht dem Format für Spritedaten. Auf diese Weise steht mit dem erweiterten SMON ein kleiner »Sprite-Editor« zur Verfügung.

N 4000 (4100) (Normaldarstellung)

interpretiert den Speicherinhalt von \$4000 (bis \$40FF) als Bildschirm-Code und gibt 32 Zeichen pro Zeile aus.

U 4000 (4100) (Übersicht)

Wie »N«, jedoch werden in einer Zeile 40 Zeichen dargestellt. Änderungen sind nur mit »N« möglich. Dieser Befehl dient hauptsächlich dazu, im Speicher abgelegte Bildschirminformationen so auszugeben, wie sie tatsächlich im 40-Zeichen/Zeile-Format aussehen würden. Dieser Befehl ist recht nützlich, um professionelle Videospiele zu analysieren, da hier Spielszenen oft im Bildschirm-Code gespeichert sind.

E 4000 (4100) (Erase)

ist der bereits in 64'er Ausgabe 2/85 vorgeschlagene Erase-Befehl zum Füllen des Speicherbereiches von \$4000 bis \$40FF mit \$00.

Y 40

kopiert die vorhandene SMON-Version in nur drei Sekunden nach \$4000 bis \$4FFF und nimmt dabei alle notwendigen Anpassungen vor. Die ursprüngliche Speicherversion des Monitors bleibt unverändert. Mit »G 4000« kann man in den neuen SMON springen. Von dem Byte-Wert, der übergeben werden muß, wird nur das obere Nibble (\$4) gewertet, so daß sich theoretisch 16 SMON-Versionen im Speicher unterbringen lassen, wobei natürlich nicht alle Möglichkeiten sinnvoll sind.

Auf diese Weise läßt sich stets die erforderliche Speicher-version herstellen, ohne daß langwierige Änderungen notwendig sind.

Q 2000

kopiert den Zeichensatz aus dem ROM von \$D000 bis \$DFFF in das RAM nach \$2000. Dort kann er mit dem Befehl »Z« nach Belieben geändert werden. Möchte man zum Beispiel das Zeichen »A« in ein »Ä« umdefinieren, so ist der Zeichensatz mit »Q 2000« ins RAM zu kopieren. Anschließend kann mit »Z 2000 2015« der Bereich in binärer Form auf dem Bildschirm ausgegeben werden, in dem auch das Zeichen »A« steht. Dieses kann nun (siehe Bild) in ein »Ä« geändert werden, indem man mit dem Cursor an die zu ändernde Stelle fährt und für einen Punkt, der gesetzt werden soll, ein »*« und für einen Punkt der nicht gesetzt werden soll ein ».« setzt. So, jetzt ist der Zeichensatz umdefiniert, aber noch nicht aktiviert. Als nächstes muß dem Videocontroller die Startadresse des neuen Zeichensatzes mitgeteilt werden. Dazu ist die Adresse \$D018, in der eine hexadezimale 15 steht, durch eine hexadezimale 18 zu ersetzen.

J

bringt den letzten Ausgabebefehl (K, D, M, Z, H, N, U) auf den Bildschirm zurück. Mit RETURN wird der letzte Befehl noch einmal ausgeführt.

Zum Schluß noch ein Tip:

DATA-Zeilen in Hex-Byte-Darstellung sind wegen ihrer konstanten Länge (immer zwei Ziffern pro Wert!) übersichtlicher als solche mit dezimalen Zahlen. Da für die Ausgabe von Hex-Werten bereits alle Routinen im SMON integriert sind, kann der »B«-Befehl (Basic-DATA-Zeilen erzeugen) durch Verändern eines einzigen Sprungbefehles dahingehend manipuliert werden, daß der Speicherinhalt künftig in Form von Hex-Byte ausgegeben wird:

Disassemblieren Sie dazu den Byte-Ausgabebefehl mit »D x99F« und ersetzen »JSR BDD1« durch »JSR x32A«. Für das »x« muß der 4-KByte-Block, in dem die zu ändernde SMON-Version steht, eingesetzt werden. Liegt Ihre SMON-Version bei \$C000, so ersetzen Sie das »x« durch ein »C«.

Die Gesamtlänge der DATA-Zeile kann außerdem durch Verändern der Speicherzelle \$x9AE variiert werden. Bei dem Wert \$1C werden zum Beispiel genau acht Hex-Byte pro Zeile ausgegeben.

(Mark Richters/ah)

```

100 REM *****
110 REM *
120 REM * SMON - ERWEITERUNG *
130 REM *
140 REM * VON MARK RICHTERS *
150 REM * ALLERSTR. 4 *
160 REM * 2806 OYTEN *
170 REM * TEL.: 04207/1870 *
180 REM *
190 REM *****
200 :
210 PRINT"BITTE DIE STARTADRESSE"
220 PRINT"IHRE SMON-VERSION EINGEBEN!"
230 INPUT H : H=H/4096
240 IF H<>INT(H) THEN 210

```

Erweiterung zum SMON. Bitte die Eingabe-Hinweise auf Seite 54 beachten.

```

250 :
260 :
270 DIM W(75)
280 FOR I=0 TO 9
290 : W(48+I)=I
300 : W(65+I)=I+10
310 NEXT I
320 :
330 FOR I=1 TO 4
340 : READ K,Z : K=K+H*4096
350 : FOR J=K TO K+Z-1
360 : READ A$
370 : A=W(ASC(A$))
380 : B=W(ASC(RIGHT$(A$,1)))
390 : S=S+A+B
400 : IF ASC(A$)=42 THEN A=H
410 : P=A*16+B : POKE J,P
420 : NEXT J
430 NEXT I
440 :
450 IF S=7031 THEN PRINT"OK!":GOTO 470
460 PRINT"FEHLER IN DATAS !"
470 STOP
480 :
490 DATA 97,7
500 DATA 27,3A,3B,2C,28,29,21
510 :
520 DATA 739,28
530 DATA A2,06,DD,61,*0,F0,08,CA
540 DATA 10,F8,A9,2E,20,D2,FF,20
550 DATA CA,*2,C9,2E,F0,F9,4C,85
560 DATA*F,4C,B2,*F
570 :
580 DATA 781,2
590 DATA F0,ED
600 :
610 DATA 3593,503
620 DATA A9,80,2C,A9,00,85,AB,20
630 DATA 64,*2,24,AB,10,03,A2,29
640 DATA 2C,A2,28,20,40,*3,20,23
650 DATA*3,A0,06,A2,00,A1,FB,0A
660 DATA 48,B0,03,A9,2E,2C,A9,2A
670 DATA 91,D1,AD,86,02,91,F3,68
680 DATA C8,E8,E0,08,D0,E9,20,67
690 DATA*3,24,AB,30,04,C0,1E,90
700 DATA DA,20,5D,*4,90,C4,60,A0
710 DATA 08,2C,A0,18,20,7E,*2,20
720 DATA B8,*2,A2,08,A9,00,85,AA
730 DATA 20,CA,*2,C9,2E,F0,07,C9
740 DATA 2A,F0,04,4C,D1,*2,18,26
750 DATA AA,8B,CA,D0,EB,A5,AA,81
760 DATA FB,C1,FB,D0,EE,20,67,*3
770 DATA C0,00,D0,D6,60,A9,80,2C
780 DATA A9,00,85,AB,20,64,*2,20
790 DATA 51,*3,24,AB,10,0B,A9,21
800 DATA 20,D2,FF,20,23,*3,A0,08
810 DATA 2C,A0,00,A2,00,A1,FB,20
820 DATA 4F,*4,D0,F9,20,5D,*4,90
830 DATA DE,60,20,7E,*2,A2,00,A0
840 DATA 08,B1,D1,81,FB,C1,FB,D0
850 DATA AA,20,56,*4,90,F3,60,20
860 DATA 8D,*2,29,F0,85,FF,20,3C
870 DATA*F,20,D6,*9,20,3C,*F,A9
880 DATA 14,85,FB,A9,02,05,FF,85
890 DATA FC,A9,D1,85,FD,A9,0F,05
900 DATA FF,85,FE,20,68,*A,A0,00
910 DATA A2,0D,BD,F2,*F,05,FF,85
920 DATA FC,CA,BD,F2,*F,85,FB,B1
930 DATA FB,29,0F,05,FF,91,FB,CA
940 DATA 10,E8,A9,2B,85,FB,A9,00
950 DATA 05,FF,85,FC,A0,35,B1,FB
960 DATA 29,0F,05,FF,91,FB,88,88
970 DATA 10,F4,A9,DE,85,FB,A9,0F
980 DATA 05,FF,85,FC,A0,13,B1,FB
990 DATA 29,0F,05,FF,91,FB,88,88
1000 DATA 10,F4,60,A5,FF,85,A9,20
1010 DATA 43,*F,68,68,29,F0,85,A5
1020 DATA 18,69,10,85,A7,A9,00,85
1030 DATA A4,85,A6,85,AB,60,20,7A
1040 DATA*2,A9,00,4C,C7,*9,20,7E
1050 DATA*2,A0,00,A9,D0,84,FD,85
1060 DATA FE,78,A9,03,85,01,A2,10
1070 DATA B1,FD,91,FB,C8,D0,F9,E6
1080 DATA FC,E6,FE,CA,D0,F2,A9,27
1090 DATA 85,01,58,60,48,C9,4A,D0
1100 DATA 10,A0,27,B9,00,02,91,D1

```

```

1110 DATA 88,10,F8,68,C6,D6,4C,D6
1120 DATA*2,A0,06,D9,D7,*F,D0,0A
1130 DATA A0,27,B1,D1,99,00,02,88
1140 DATA 10,F8,88,10,EE,68,4C,FF
1150 DATA*2,A2,0A,DD,D0,*F,F0,06
1160 DATA CA,D0,F8,4C,D1,*2,20,C5
1170 DATA*F,4C,D6,*2,8A,0A,AA,BD
1180 DATA DD,*F,48,BD,DC,*F,48,60
1190 DATA 28,29,21,45,59,51,48,5A
1200 DATA 4E,55,44,4B,4D,52,*E,4F
1210 DATA*E,82,*E,56,*F,C7,*E,5E
1220 DATA*F,0B,*E,08,*E,85,*E,88
1230 DATA*E,06,00,87,03,2D,0C,5C
1240 DATA 0C,F5,0C,A2,0D,04,0E

```

64'er

Erweiterung zum SMON. (Schluß)

) 4000	..*****.	) 4000	..*****.
) 4001	..**..***.	) 4001	..**..***.
) 4002	..**..***.	) 4002	..**..***.
) 4003	..**..***.	) 4003	..**..***.
) 4004	..**..***.	) 4004	..**..***.
) 4005	..**..***.	) 4005	..**..***.
) 4006	..**..***.	) 4006	..**..***.
) 4007	..**..***.	) 4007	..**..***.
) 4008	..**..***.	) 4008	..**..***.
) 4009	..**..***.	) 4009	..**..***.
) 400A	..**..***.	) 400A	..**..***.
) 400B	..**..***.	) 400B	..**..***.
) 400C	..**..***.	) 400C	..**..***.
) 400D	..**..***.	) 400D	..**..***.
) 400E	..**..***.	) 400E	..**..***.
) 400F	..**..***.	) 400F	..**..***.
) 4010	..**..***.	) 4010	..**..***.
) 4011	..**..***.	) 4011	..**..***.
) 4012	..**..***.	) 4012	..**..***.
) 4013	..**..***.	) 4013	..**..***.
) 4014	..**..***.	) 4014	..**..***.

Bild. Originaler und abgeänderter Zeichensatz.
Aus dem A wird ein Ä.

Programme nachladen

Wenn man von Basic aus ein Programm nachladen will, so ist im allgemeinen die Tatsache störend, daß der Interpreter anschließend das Basic-Programm neu startet. Mit folgender Befehlsfolge wird dies umgangen:

```
SYS 57812"name",8,1:POKE 780,0:SYS 65493
```

Das gewünschte Maschinenprogramm wird geladen, und das Basic-Programm mit dem nächsten Befehl fortgesetzt.

Wenn man statt ...,8,1 die Sekundäradresse ...,8,0 setzt, und im X- und Y-Register des 6510 (Speicherstellen 781 und 782) Low-Byte und High-Byte einer bestimmten Adresse angibt, so wird das Programm an diese Adresse geladen. Eine Anwendung wäre zum Beispiel eine Merge-Routine:

```

1000 AD=PEEK(45)+PEEK(46)*256-2
1010 AH=INT(AD/256)
1020 AL=AD-256*AH
1030 POKE 781,AL
1040 POKE 782,AH
1050 SYS 57812"name",8,0
1060 POKE 780,0
1070 SYS 65493

```

(Martin Barth/tr)