

```

4340 GETA$:A=VAL(A$):ONAGOTO 4350,4360:GOTO4340
4350 D=1:GOTO4370
4360 D=8
4370 PRINT"□"
4380 FORJ=1TO6:PRINT:NEXTJ
4390 IFD=1THENPRINTSPC(6)"RECORDER O.K. ?"
4400 IFD=8THENPRINTSPC(6)"FLOPPY O.K. ?"
4410 PRINT:PRINT:INPUT"  FILENAME ";N$
4420 OPEN1,D,1,N$
4430 PRINT#1,SW:PRINT#1,DI:PRINT#1,N1(I)
4440 FORK=1TON1(I)
4450 PRINT#1,X(I,K):PRINT#1,Y(I,K):PRINT#1,SW(I,K)
4460 NEXTK
4470 OPEN2,8,15:INPUT#2,F,B$
4480 IF F=0THEN4500
4490 PRINT"  FEHLER: "B$:CLOSE2:CLOSE1:GOTO4410
4500 CLOSE2:CLOSE1:D=0:GOTO890
4510 :
4520 :
4530 REM EINGABE DATENFILE
4540 :
4550 D=1:GOTO4570
4560 D=8
4570 PRINT"□"
4580 FORJ=1TO6:PRINT:NEXTJ
4590 IFD=1THENPRINTSPC(12)"RECORDER O.K. ?"
4600 IFD=8THENPRINTSPC(12)"FLOPPY O.K. ?"
4610 GETA$:IFA$=""THEN4610
4620 PRINT"  ":PRINT:INPUT"          ANZAHL DATENFILE
S";AK$
4630 IFAK$=""THEN150
4640 AK=VAL(AK$)
4650 DIMN$(AK):PRINT
4660 FORJ=1TOAK
4670 IFJ<0THENRUN
4680 PRINT:PRINT"          FILENAME";J:INPUT"  "
";N$(J)
4690 IFN$(J)=""THENJ=J-2:NEXTJ
4700 NEXTJ
4710 FORJ=1TOAK
4720 OPEN1,D,0,N$(J):OPEN2,8,15:INPUT#2,F,B$
4730 IFF=0THEN4770
4740 PRINT"  FEHLER: "B$:CLOSE2:CLOSE1:DIR"$
4750 GETA$:IFA$<>""THENPRINT"  ":GOTO4660
4760 GOTO4750
4770 INPUT#1,SW,DE,N1(J)
4780 IFN1(J)>N1THENN1=N1(J)
4790 CLOSE1:CLOSE2:NEXTJ
4800 N1=N1+1
4810 DIMX(AK,N1),Y(AK,N1),SW(AK,N1)
4820 DIMA(N1),B(N1),C(N1),D(N1),H(N1),M(N1,N1),V(N
1),O(N1),P(N1),Q(N1)
4830 FORJ=1TOAK
4840 OPEN1,D,0,N$(J)
4850 INPUT#1,SW,DE,N1(J)
4860 IFDE>DITHENDI=DE
4870 FORK=1TON1(J)
4880 INPUT#1,X(J,K),Y(J,K),SW(J,K)
4890 NEXTK:CLOSE1
4900 NEXTJ:IFZW>DITHENDI=ZW
4910 DIMZ(DI+3),T(DI+3):D=0:GOTO260
4920 :
4930 REM VERGLEICH FKT - SPL
4940 :
4950 PRINT"  ":PRINT:PRINT:INPUT"          FUNKTION F(X
)=";F$
4960 IFF$=""THENRUN
4970 PRINT"  FEHLER: "B$:CLOSE2:CLOSE1:DIR"$
4980 PRINT"GOTO5000"
4990 POKE631,19:POKE632,13:POKE633,13:POKE198,3:EN
D
5000 PRINT"  "          FUNKTIONSDATEN
"
5010 PRINT:PRINT:INPUT"  ANFANGS X-WERT";XA$
5020 IFA$=""THEN4950
5030 XA=VAL(XA$)
5040 PRINT:PRINT:INPUT"  END X-WERT";XE$
5050 IFXE$=""THEN5010
5060 XE=VAL(XE$)
5070 PRINT:PRINT:INPUT"  ANZAHL ZWISCHENWERTE";ZW$
5080 IFZW$=""THEN5040
5090 ZW=VAL(ZW$)
5100 SV=(XE-XA)/ZW:E5=1
5110 GOTO130
5120 G=0:E5=2
5130 FORX=XATOXE+.0001STEP5V
5140 DEFFNF(X)=X+2
5150 Y=FNFX(X):GOSUB3300
5160 NEXTX
5170 IFE6=1THEN1050
5180 IFE6=2THEN3440
5190 GOTO2340
READY.

```

Listing »Spline«
(Schluß)

Dem C 64 und Plus/4 Arbeit aufstapeln

Wollen Sie viele verschiedene Tätigkeiten direkt hintereinander ausführen und haben keine Lust, die ganze Zeit neben dem Computer sitzen zu müssen? Die Lösung heißt Batch- oder Stapel-Verarbeitung. Zusätzlich stellen wir Ihnen noch einen komfortablen Editor zur Verfügung.

Hier werden gleich zwei Programme auf einmal vorgestellt, die beide sehr nützlich sind: EDI.BAS – ein total in Basic geschriebener und trotzdem vollwertiger, schneller Screen-Editor

BATCH – ein Programm, das einen Hauch von Großrechner-Komfort zugänglich macht, indem ganze Folgen von Programmen und Eingaben automatisch ablaufen, ohne daß der Benutzer Hand anlegen muß.

Die Batch-Verarbeitung arbeitet rein theoretisch wie ein sehr großer Tastaturpuffer. Man kann dem Computer im voraus die Eingaben für die nächsten Minuten oder gar Stunden geben. In diesen Eingaben können selbstverständlich LOAD- und RUN-Befehle enthalten sein, so daß sich ganze Programmketten auf einen Befehl hin abarbeiten lassen. Da man sich das wie einen Stapel von Eingaben vorstellen kann, von dem der Computer sich immer eine neue Arbeit abholt, bis der Stapel abgetragen ist, nennt man diese Prozedur Stapel- oder englisch Batch-Verarbeitung.

Der Editor – EDI.BAS

Wie oft muß in Programmen eine größere Menge Text eingegeben werden oder es werden Datenfiles gebraucht, in denen bestimmte Werte drinstehen oder korrigiert werden sollen. Man findet dann oft mehr oder weniger aufwendig gestaltete Editor-Routinen, die aber lange nicht an den Komfort heranreichen, den ein echter Screen-Editor bietet.

Die Routine EDI.BAS, die im Listing 1 die Zeilen 1 bis 299 umfaßt, ist ein kleiner Auszug aus einem kompletten Textsystem. Sie ist völlig eigenständig und kann auch ohne den Rest dieses Listings verwendet werden.

Eine besondere Eigenschaft von EDI.BAS ist, daß es sich automatisch an 40- und 80-Zeichen-Schirme anpassen kann! Wen die Verfahrensweise interessiert, der sollte sich die Zeile 350 ansehen. Dort wird ein Zeichen an die 41. Bildschirmposition gesetzt und dann überprüft, ob es sich schon in der zweiten Zeile befindet.

Vor der ersten Benutzung als eigenständiges Programm ist EDI.BAS durch Aufruf der Zeile 200 zu initialisieren. Dies entfällt, wenn man das komplette Listing abgetippt hat.

EDI.BAS bearbeitet einen zeilenweise in A\$() abgespeicherten Text. Aufgerufen wird der Editor nach der Initialisierung einfach mit GOSUB 100.

EDI.BAS erklärt sich durch seine Help-Funktion fast von selbst. Diese Funktion kann man durch Eintippen von »H« und Return in der Kommandozeile erreichen.

Einschränkungen bestehen nur darin, daß nur die allernotwendigsten Funktionen vorhanden sind und daß die Zeilenlänge auf 77 Zeichen beschränkt werden mußte.

Erst dadurch, das EDI.BAS vollkommen in Basic programmiert wurde, ohne POKE- und PEEK-Befehle zu benutzen, konnte erreicht werden, daß die Routine auf fast allen Commodore-Rechnern, so dem C 64, C 128 oder dem 8032 anstandslos läuft.

Batch-Verarbeitung

Viele Aktionen sind nicht direkt programmierbar, so zum Beispiel die Aufrufe mehrerer Programme hintereinander und die Versorgung dieser Programme mit diversen Eingaben. Solche Befehlsfolgen können mit BATCH in einem Zug eingetippt und dann automatisch durchgeführt werden. Bedient wird BATCH folgendermaßen:

a) Start einfach mit RUN (ohne Parameter): Es wird ein Filename erfragt, unter dem die Kommandofolge auf der Floppy abgelegt werden soll beziehungsweise schon vorhanden ist. Wenn das File noch nicht existiert, kann es mit dem in BATCH 64 eingebauten Editor erstellt werden. Ist die Kommandofolge fertig, wird sie abgespeichert. Sie wird anschließend nochmals eingelesen und in einem geschützten RAM-Bereich abgelegt. Zum Schluß kann der Batch-Job mit einem unter dem Cursor bereitgestellten SYS-Befehl gestartet werden.

b) Start mit Angabe von Parametern: In einem Kommando-Mutterfile können manche Eingaben (zum Beispiel Filenamen) mit formalen Parametern %a, %b, ..., %z gekennzeichnet werden. Beim Aufruf von BATCH 64 werden diese dann mit aktuellen Inhalten versorgt und gelangen so zur Ausführung. Dazu muß hinter dem RUN-Befehl ein Doppelpunkt geschrieben werden, gefolgt von der Liste der aktuellen Parameter-Inhalte (Strings), jeweils durch Kommata getrennt. Der erste Parameter ist immer der Name des Kommando-Mutterfiles; wird hier ein Minus (-) eingegeben, wird der Parameter erfragt. Die folgenden entsprechen der alphabetischen Reihenfolge %a, ... Soll ein Parameterstring selbst ein Komma enthalten, muß er in Gänsefüßchen eingeschlossen werden. Gänsefüßchen können nicht übergeben werden! Beispiel: RUN: Mutterfile, erster Par, zweiter,, vierter, "ein komma: "

Bleibt ein Parameter leer (hier %c), erzeugt er im Kommando-file eine leere Stelle. Auf diese Weise wird aus dem Kommando-Mutterfile ein aktuelles Kommandofile (Batchfile) erzeugt, das auf Wunsch auch noch verändert werden kann. Vor Ausführung muß es auf jeden Fall noch einmal auf Floppy gespeichert werden. Der endgültige Aufruf erfolgt mit SYS wie oben bei a).

c) Aufruf mit Angabe des Kommandofilenamens: Ist schon ein fertig verwendbares Kommandofile vorhanden, kann es ohne weitere Nachfragen direkt gestartet werden. Dazu schreibt man dessen Filenamen (und nur diesen) durch Doppelpunkt getrennt hinter das RUN des Startkommandos für das Programm BATCH 64. Beispiel: RUN:kommando.bat (also ähnlich wie oben das Mutterfile).

Wirkungsweise:

Der Kommandotext wird zusammen mit einer zugehörigen Maschinen-Routine unterhalb des aktuell zugänglichen RAM-Endes abgelegt (relokiert). Dieser Speicherbereich wird durch Umsetzen des Top-of-RAM-Pointers vor Überschieben durch Basic geschützt. Die Maschinenroutine wird in die Interrupt-Routine des Computers eingeschleift. Stellt diese nach Aktivierung fest, daß der Tastaturpuffer leer ist,

```
0 REM SAVE"00:BATCH64",B:REM BATCH-VERARBE
  ITUNG AUF C 64 , VERWENDET: <100>
1 REM EDI.BAS ZEILENORIENTIERTER ASCII-T
  EXT-EDITOR FUER ALLE CBM-RECHNER <135>
2 REM VON P. KITTEL , FFM 1985 <075>
4 GOTO 200: REM COLD START MI
  T INITIALISIERUNG <063>
20 BJ=B:BI=CS+1:IF BI>LEN(A$)THEN BI=LEN(A
  $):IF BI=0 THEN BI=1 <028>
22 GOSUB 28:CS=BI-1:IF BI<>13 AND BI<=ML A
  ND((BI<>17)OR K)THEN 22 <114>
24 IF BI=13 THEN RETURN <106>
25 IF BI=17 AND K=0 THEN K=B:RETURN <064>
26 IF LEN(A$)>250 THEN A$=LEFT$(A$,LEN(A$)
  -1):IF BI>LEN(A$)THEN BI=BI-1:PRINT"LE
  FT": <176>
27 PRINT BL$;:GOTO 22 <123>
28 BI=0:BJ$=MID$(A$,BI,1):BI$=MID$(A$,BI+1
  ,1):IF BJ$=""THEN BJ$=" " <036>
29 IF (ASC(BJ$)AND 127)=18 THEN BJ$=CHR$(AS
  C("#")+ASC(BJ$)AND 128)) <158>
30 GET B$:IF B$<>""THEN PRINT"RVOFF"BJ$;
  :GOTO 34 <248>
31 BJ=(1+BJ)AND 15:PRINT CHR$(18+16*(BJ AN
  D 8));BJ$; <196>
32 IF BJ$=G$THEN PRINT G$(LEFT,RVOFF)":;I
  F BI<LEN(A$)AND BI$<>G$THEN PRINT BI$(
  LEFT)": <116>
33 PRINT"LEFT)":;RETURN <046>
34 IF BJ$=G$THEN PRINT G$(LEFT)":;IF BI<L
  EN(A$)AND BI$<>G$THEN PRINT BI$(LEFT)":
  ; <207>
35 PRINT"LEFT)":;B=ASC(B$):B1=B AND 127:I
  F B1>31 THEN 39 <163>
36 IF B1>12 THEN ON B1-12 GOTO 74,38,38,38
  ,38,87,76,60,38,38,38,38,38,38,38,46 <006>
37 REM 13:RETURN,17:VERT,18:REV,19:HOME,20
  :DEL,29:HORIZ <189>
38 RETURN <096>
39 IF BI>ML THEN PRINT BL$;:RETURN <109>
40 BI=BI+1:IF B=222 THEN B$=CHR$(255) <014>
41 PRINT B$;:IF B$=G$THEN PRINT G$(LEFT)":
  ;:IF BI<=LEN(A$)AND BI$<>G$THEN PRINT B
  I$(LEFT)": <104>
42 IF BI>2 THEN B$=LEFT$(A$,BI-2)+B$ <146>
43 B$=B$+MID$(A$,BI) <086>
44 A$=B$:RETURN <121>
46 IF B=29 THEN 54 <157>
48 IF BI>1 THEN PRINT B$;:BI=BI-1 <128>
50 IF BI<2 THEN PRINT BL$; <253>
52 RETURN <110>
54 IF BI>=ML THEN PRINT BL$;:RETURN <102>
56 PRINT B$;:BI=BI+1:IF LEN(A$)<BI THEN A$
  =A$+" " <100>
58 RETURN <116>
60 IF B=148 THEN 66 <225>
62 IF BI<2 THEN PRINT BL$;:RETURN <243>
63 PRINT B$;:B$=MID$(A$,BI):IF BI>2 THEN B
  $=LEFT$(A$,BI-2)+B$ <037>
64 A$=B$:BI=BI-1:RETURN <175>
66 L=LEN(A$):BJ$=RIGHT$(A$,1):IF L>=ML THE
  N PRINT BL$;:RETURN <106>
68 PRINT B$(SPACE,LEFT)":;B$=" "+MID$(A$,
  BI):IF BI>1 THEN B$=LEFT$(A$,BI-1)+B$ <087>
72 A$=B$:RETURN <149>
74 PRINT:IF B=13 THEN B=17:B1=B:BI=1:GOTO
  B1 <068>
75 B=141:RETURN <121>
76 IF B=19 THEN PRINT LEFT$(CB$,BI-1):;BI=
  1:RETURN <001>
78 IF BI<LEN(A$)THEN PRINT SPC(LEN(A$)-BI)
  ;:BI=LEN(A$) <121>
79 IF BI<ML AND RIGHT$(A$,1)<>" "THEN BI=B
  I+1:A$=A$+" ":PRINT"RIGHT)": <102>
80 RETURN <138>
81 IF M>=N THEN RETURN <078>
82 B$=A$(M+1):IF B$=""THEN RETURN <244>
83 IF MID$(B$,BI,1)=" "THEN IF BI<LEN(B$)T
  HEN BI=BI+1:GOTO 83 <170>
84 IF MID$(B$,BI,1)=" "THEN BI=1 <071>
85 RETURN <143>
87 BI=BI+1:IF BI>ML+1 THEN PRINT BL$; <181>
88 PRINT CHR$(ASC("#")+B AND 128)):;GOTO
  42 <085>
90 PRINT"LX (X = ZAHL) = _DESCHEN(2SPACE)X
  ZEILE OD. <003>
91 PRINT"EX (X = ZAHL) = _INFUEGEN X ZEILE
  N OD. <200>
```

```

92 PRINT" H(2SPACE) FUER HILFE ODER":PRINT"-
(2SPACE) FUER ABBRUCH. <121>
93 PRINT" (DOWN) IN DEN TEXTZEILEN: ":PRINT" G
URSORTASTEN UND INS/DEL NORMAL <042>
94 PRINT" SHIFT-RETURN(2SPACE)= ZURUECK ZU
BEFEHLSZEILE <246>
95 PRINT" RETURN(8SPACE)= NAECHSTER ZEILENA
NFANG <247>
96 PRINT" HOME(10SPACE)= AKTUELLER ZEILENAN
FANG <228>
97 PRINT" SHIFT-HOME(4SPACE)= ZEILENENDE. <042>
99 PRINT" (DOWN) TASTE DRUECKEN. ":FOR I=-1 T
O 0:GET A$:I=A$="":NEXT <002>
100 K=0:PRINT" (CLR)":N=N*(N)=0:G#=CHR$(3
4):BL#=CHR$(7) <003>
101 PRINT" (HOME)":TAB(69);M=(M<0)+(M-N)*(M
>N) <016>
102 PRINT" (HOME)AEND/ (RVSON)L (RVOFF)OESCH/
(RVSON)E (RVOFF)INFUEG/ (RVSON)H (RVOFF)I
LFE, ENDE MIT -, " <135>
103 INPUT" ZEILENNR. / BEFEHL (L./E./H) = ":A$
:B#=LEFT$(A$,1):IF B#="-" THEN RETURN <207>
104 V=M:IF M<0 THEN V=0 <141>
105 B=VAL(MID$(A$,2)):IF B#="L" THEN B=B+V-
1:GOSUB 185:M=V:GOTO 100 <141>
106 M=VAL(A$):IF B#="E" THEN 190 <151>
110 IF M<0 OR M>N THEN 100 <150>
111 PRINT" (HOME) CURSOR AKTIV, (2SPACE) RETUR
N: NAECHSTE ZEILE, (2SPACE) COMMANDOZ.: "
<134>
112 PRINT" SHIFT-RETURN, Z.: " <251>
113 ES$=","+CHR$(20)+LEFT$(CB$,50):FOR I=1
TO 39-40*CB$:PRINT"-":NEXT:PRINT <137>
114 IF B#="H" THEN PRINT" (DOWN) BEFEHLSZEILE
":PRINT" ZU AENDERNDE ZEILENNR. ODER":
GOTO 90 <082>
118 I=4-6*CB$:ZV=M-I:ZB=M+I+1:IF ZV<0 THEN
ZV=0:ZB=2*I+1 <100>
119 FOR J=ZV TO ZB:IF J>N THEN A$="":GOTO
122 <235>
120 A$=A$(J) <170>
122 L=LEN(A$):IF L=0 THEN PRINT LEFT$(CP$,
7B)ES$:GOTO 128 <223>
124 FOR BI=1 TO L:B#=MID$(A$,BI,1):IF B#=G
# THEN PRINT G#:CHR$(20); <143>
125 IF (ASC(B#) AND 127)=18 THEN B#=CHR$(ASC
("#")+(ASC(B#) AND 128)) <030>
126 PRINT B#:NEXT <110>
127 PRINT LEFT$(CP$,7B-L);ES$ <006>
128 NEXT <138>
130 PRINT" (HOME)":FOR I=ZV TO M:PRINT:NEXT
<017>
155 A$=A$(M):L=LEN(A$):IF CS>L-1 THEN CS=L
-1:IF CS<0 THEN CS=0 <052>
156 PRINT SPC(CS); <188>
160 GOSUB 20 <106>
161 IF LEN(A$)<2 THEN 169 <252>
162 IF RIGHT$(A$,1)="" THEN A$=LEFT$(A$,LE
N(A$)-1):GOTO 161 <234>
169 A$(M)=A$:IF K<0 THEN 180 <194>
170 M=M+1+2*(K>20):IF M<ZV OR M>ZB THEN 10
0 <248>
171 PRINT" (HOME)"TAB(69);M;" (LEFT,3SPACE)"
:K=0:IF M>N THEN N=M <131>
172 GOTO 130 <148>
180 IF A$(N)="" THEN N=N-1:IF N>-1 THEN 180
<012>
182 GOTO 100 <110>
185 IF V<0 OR V>B OR B>N THEN PRINT" (DOWN)
UNMOEGLICH!":FOR I=1 TO 200:NEXT:RETUR
N <123>
186 M=B-V+1:IF B=N THEN 188 <009>
187 FOR I=B+1 TO N:A$(I-M)=A$(I):NEXT <063>
188 N=N-M:IF N<MZ THEN FOR I=N+1 TO N+M:A$
(I)="" :NEXT:RETURN <143>
190 IF B>N>MZ OR B<1 THEN V=-1:GOSUB 185:G
OTO 100 <234>
191 FOR I=N TO V STEP -1:A$(I+B)=A$(I):NEXT
<134>
192 FOR I=V TO V+B-1:A$(I)="" :NEXT:IF CS=0
OR CS>LEN(A$(V+B)) THEN 199 <159>
193 A$(V)=LEFT$(A$(V+B),CS):A$(V+B)=MID$(A
$(V+B),CS+1) <167>
199 N=N+B:M=V:GOTO 100 <057>
200 REM INITIALISIERUNG EDI.BAS: -----
<086>
210 REM EINFUEHRUNG DER BENUTZTEN VARIABLE
N: <248>
220 DIM A$,B$,BI,BJ,BJ$,G$,BI$,BL$,CB
$,CP$,CS,ES$,I,J,K,L,M,ML,MZ,N,V,CB
<037>
221 DIM ZB,ZV <051>
230 MZ=600: REM MAXIMALE ZEILENZAHLE <152>
240 ML= 77: REM MAX. ZEICHENZAHLE/ZEILE G
GF. PER INPUT (NUR KLEINER MOEGLICH) <021>
250 N = -1: REM AKTUELLER FUELLSTAND IN A
$() <147>
251 M = -1: REM ZULETZT EDIERTE ZEILENNR. <099>
262 CB$="(4LEFT)":FOR I=1 TO 5:CB$=CB$+CB$
:NEXT:CB$=CB$+LEFT$(CB$,127):REM CURS
BACK <195>
263 CP$="(4SPACE)":FOR I=1 TO 5:CP$=CP$+CP
$:NEXT:CP$=CP$+LEFT$(CP$,127):REM SPAC
E <152>
270 DIM A$(MZ): REM TEXT ZEILENWEISE <235>
280 : REM EDITOR MIT GOSUB100 AU
FRUFEN <213>
284 : REM TEST AUF 40- BZW. 80-Z
EICHEN-SCHIRM: <026>
285 REM PRINT" (2HOME,CLR,CTRL-N)"SPC(40)"T
(HOME,DOWN)":;OPEN 1,3:GET#1,A$:CLOSE 1:
CB=A$="":REM S.U. <162>
290 REM <098>
299 REM EDI.BAS FERTIG -----
<202>
300 PRINT" (UP,2RIGHT)":DIM P$(27),NZ(27):
NP=0:OPEN 1,3:REM PARAMETER HOLEN <164>
301 C=0:FOR I=2 TO 79:GET#1,A$:IF A$="":TH
EN C=I:I=80 <243>
302 NEXT:IF C=0 THEN CLOSE 1:MP=-1:GOTO 35
0 <136>
310 MP=0:FOR P=1 TO 27:G$="":FG=1:FC=0 <006>
311 FOR I=-1 TO 0:I=-1:GET#1,A$:IF A$=CHR$
(34) THEN FG=1-FG:GOTO 320 <059>
312 C=C+1:IF C>79 OR A$=CHR$(13) THEN I=0:P
$(FC*P)=G$:MP=MP+FC*(P-MP):P=99:GOTO 3
20 <157>
313 IF FG THEN IF A$="," THEN I=0:P$(FC*P)=
G$:MP=MP+FC*(P-MP):GOTO 320 <137>
315 IF FC OR (A$<>"") THEN FC=1:G$=G$+A$ <105>
320 NEXT:NEXT:CLOSE 1:IF MP=0 THEN 350 <156>
330 FOR P=1 TO MP:IF P$(P)="" THEN 335 <248>
331 P$=P$(P):L=LEN(P$) <056>
332 IF RIGHT$(P$,1)="" THEN L=L-1:P$=LEFT$
(P$,L):GOTO 332 <123>
333 P$(P)=P$ <120>
335 P$(P-1)=P$(P):NEXT:P$(MP)="" :MP=MP-1
<185>
340 NP=-1:FOR P=0 TO MP:IF P$(P)<>"" THEN N
P=P <121>
341 NEXT:MP=NP:F$=P$(0):IF F$="" THEN F$="-
" <020>
349 REM ZEILE 285 HIERHER VERLEGT: <035>
350 PRINT" (2HOME,CLR,CTRL-N)"SPC(40)"T (HOM
E,DOWN)":;OPEN 1,3:GET#1,A$:CLOSE 1:CB
=A$="":REM 40/80 ? <067>
351 PRINT" (CLR,DOWN,SPACE,RVSON,CTRL-N,SPA
CE) (SHIFT-SPACE) (SHIFT-SPACE) (SHIFT
-SPACE) (SHIFT-SPACE) (SHIFT-SPACE,RVO
FF,DOWN) <073>
352 FP=0:IF MP=0 AND F$<>"" AND F$<>"" THEN
FP=1:GOTO 511 <228>
353 IF MP<0 THEN 500 <170>
360 PRINT" (2DOWN) PARAMETER: ":PRINT" (DOWN) E
ILE: "F$"(DOWN) <117>
361 FOR P=1 TO MP:IF P$(P)<>"" THEN PRINT" (
3SPACE) "CHR$(P+64)": "P$(P) <018>
362 NEXT:GOTO 510 <233>
500 PRINT" (DOWN) NAMEN FUER SEQ-COMMANDO-EI
LE EINGEBEN, <202>
502 PRINT" WENN NOCH KEINES EXISTIERT: EIN
FACH(2SPACE)- <094>
505 INPUT" (DOWN) FILENAME = ":F$ <129>
510 IF F$="" THEN 520 <239>
511 PRINT" (DOWN) FILE WIRD EINGELESEN, BITT
E GEDULD. (3DOWN) <006>
512 N$=CHR$(0):CR$=CHR$(13):N=-1:OPEN 1,8,
3,F$ <090>
513 FOR Z=-1 TO 0:G$="":FOR I=-1 TO 0:GET#
1,A$:S=ST:A$=LEFT$(A$+N$,1) <219>
514 I=(S=0) AND (A$<>CR$):IF A$<>CR$ THEN G$=
G$+A$:IF A$="%" THEN IF MP>0 THEN GOSUB
1000 <197>
515 NEXT:N=N+1:A$(N)=G$:PRINT G$:Z=S=0:NEX
T:CLOSE 1:IF MP>0 THEN 520 <215>
516 IF FP THEN 650 <184>
517 PRINT" (3DOWN) AENDERUNGEN NOETIG (J/N)
? ": <231>

```

Listing 1. Programm EDI.BAS/BATCH. Bitte mit dem Checksummer Seite 6 eingeben.

```

518 GET A$:IF A$<>"J"AND A$<>"N"THEN 518 <040>
519 PRINT A$:IF A$="N"THEN 650 <118>
520 PRINT "DOWN"FILE MUSS MIT EINGEBAUTEM <168>
EDITOR <168>
521 PRINT "BE/ER-ARBEITET WERDEN. <015>
522 PRINT "GEBEN SIE IN DER SPAETEREN KOMM <222>
AND- <222>
523 PRINT "ZEILE (SPACE,RVSON)H(RVOFF,SPACE <081>
)FUER HILFE EIN. <081>
524 PRINT "DOWN"HINWEIS: MIT DIESEM EDITOR <174>
KOENNEN NUR <174>
525 PRINT " (9SPACE)STANDARDZEICHEN, & UND <151>
&VS <151>
526 PRINT " (9SPACE)EINGEGEBEN WERDEN, (SPAC <049>
E,RVSON)KEINE (RVOFF) <049>
527 PRINT " (9SPACE)ANDEREN STEUERZEICHEN! <236>
528 PRINT "DOWN"ALTERNATIVE: FILE MIT ANDE <040>
REM REINEN <203>
529 PRINT " (9SPACE)ASCII-EDITOR ERZEUGEN. <002>
530 PRINT "DOWN"DRIVE FUER FILE-SPEICHERU <211>
NG (0/1) ? " ; <208>
531 GET D$:IF D$<>"0"AND D$<>"1"THEN 531 <232>
532 PRINT D$:IF MP>0 THEN MP=0:GOTO 550 <037>
540 GOSUB 100 <063>
550 FA$=F$:PRINT " (2HOME,CLR,DOWN)FILE-INHA <215>
LT OK (J/N) ? " ; <010>
551 GET A$:IF A$<>"J"AND A$<>"N"THEN 551 <054>
552 PRINT A$:IF A$="N"THEN 540 <033>
560 INPUT "DOWN"FILE-NAME FUER SPEICHERUNG <247>
= " ;F$:IF F$="-"THEN 560 <157>
570 IF NP>0 THEN IF F$=FA$THEN PRINT "DOWN <005>
)NICHT PARAMETER-FILE UEBERSCHREIBEN!" <107>
:GOTO 560 <063>
600 PRINT "DOWN"FILE WIRD GESPEICHERT, BIT <169>
TE GEDULD. <071>
610 OPEN 1,8,15,"S"+D$+"":+F$:CLOSE 1:OPEN <168>
1,8,3,(D$+"":+F$+"",S,W") <168>
620 FOR I=0 TO N:PRINT#1,A$(I):NEXT:CLOSE <114>
1 <087>
630 IF LEN(F$)<5 OR RIGHT$(F$,4)<>".MOT"TH <100>
EN 650 <205>
640 PRINT "DOWN"BITTE MIT PARAMETERANGABE <047>
NEU STARTEN." :END <092>
650 M=0:FOR I=0 TO N:M=M+LEN(A$(I))+1:NEXT <061>
700 TR=55:REM PLUS/4: 55, 8XXX/40XX: 52 <052>
701 T=PEEK(TR)+256*PEEK(TR+1)-1:REM PLUS/4 <052>
: IFT>32767THENT=32767 <071>
702 TH=INT(T/256):TL=T-256*TH <156>
703 LH=INT(M/256):LL=M-256*LH <147>
704 B=T-M:BH=INT(B/256):BL=B-256*BH <000>
710 PA=BH:IF BL<122 THEN PA=PA-1 <003>
720 P=256*PA:POKE P,TL:POKE P+1,TH:POKE P+ <229>
2,BL:POKE P+3,BH:POKE P+4,FP <136>
721 L=LEN(F$):IF L>16 THEN L=16 <154>
722 FOR I=1 TO L:N$(I)=ASC(MID$(F$,I,1)):N <224>
EXT <239>
730 POKE P+5,L:FOR I=1 TO L:POKE P+5+I,N$( <065>
I):NEXT <217>
740 POKE 55,0:POKE 56,PA <173>
750 CLR <173>
800 A$="":I=0:N$=CHR$(0) <047>
805 TR=55:REM PLUS/4: 55, 8XXX/40XX: 52 <092>
810 M=PEEK(TR+1):P=256*M <061>
815 TL=PEEK(P):TH=PEEK(P+1):BL=PEEK(P+2):B <052>
H=PEEK(P+3):FP=PEEK(P+4) <071>
820 L=PEEK(P+5):F$="":FOR I=1 TO L:F$=F$+C <061>
HR$(PEEK(P+5+I)):NEXT <052>
830 T=TL+256*TH <071>
835 PRINT "DOWN"FILE "F$" WIRD NOCH EINMAL <168>
836 PRINT "EINGELEN, BITTE GEDULD. DOWN" <168>
840 OPEN 1,8,3,F$:FOR I=-1 TO 0:GET#1,A$:I <183>
=ST=0:POKE T,ASC(A$+N$):T=T-1:NEXT:CLO <156>
SE 1 <147>
850 RESTORE:READ N <000>
860 FOR I=0 TO N:READ A$:IF A<0 THEN A=M <003>
861 POKE P+I,A:NEXT <229>
870 POKE P+ 71,TL:POKE P+ 72,TH <136>
871 POKE P+118,BL:POKE P+119,BH <154>
900 PRINT "DOWN"BATC-JOB MIT (4SPACE)SYS" <224>
P" (4SPACE)STARTEN: <239>
910 IF FP THEN PRINT:SYS P:END <065>
920 PRINT "DOWN"SYS"P" (3UP)" <217>
999 END <173>
1000 GET#1,A$:S=ST:A$=LEFT$(A$+N$,1):A=ASC <047>
(A$) <092>
1010 FOR P=1 TO MP:IF A=P+64 THEN G$=LEFT$ <061>
(G$,LEN(G$)-1)+P$(P):P=MP:A=-1 <052>
1011 NEXT:IF A>-1 THEN G$=G$+A$ <071>

```

```

1020 I=(S=0)AND(A$<>CR$):RETURN <090>
6000 A=7*4096:FOR Z=0 TO 13:PRINT A;"DATA" <121>
; <121>
6010 FOR I=0 TO 7:PE=PEEK(A):A=A+1:IF I>0 <022>
THEN PRINT","; <022>
6020 PRINT " ";RIGHT$("00"+MID$(STR$(PE),2) <062>
,3);:NEXT:PRINT:NEXT <190>
6030 END <096>
7000 RESTORE:READ N:A=28672 <022>
7010 FOR I=0 TO N:READ P:IF ABS(P)=112 THE <133>
N PRINT CHR$(18-(P<0)*128);A+(I AND 2 <234>
48); <060>
7020 NEXT:END <126>
8000 REM " (2SPACE)GENUTZTE MASCHINENADRESSE <123>
N: <060>
8010 REM " (2SPACE)55-56 (4SPACE)POINTER TOP <126>
OF RAM <123>
8020 REM " (2SPACE)$0314 (4SPACE)POINTER I& <250>
ROUTINE <050>
8030 REM " (2SPACE)$0277 (4SPACE)ERSTES BYTE <169>
I&STATURPUFFER <155>
8040 REM " (2SPACE)$C6 (6SPACE)ANZAHL BYTES I <160>
M I&STATURPUFFER <060>
28600 REM SOURCE-FILE: IRQBAT.SRC <173>
28666 DATA 119 <039>
28672 DATA 120, 173, 020, 003, 141, 116,-1 <039>
12, 173 <189>
28680 DATA 021, 003, 141, 117,-112, 169, 0 <250>
51, 141 <231>
28688 DATA 020, 003, 169,-112, 141, 021, 0 <124>
03, 088 <081>
28696 DATA 096, 189, 005, 001, 041, 252, 2 <081>
01, 044 <081>
28704 DATA 208, 017, 189, 006, 001, 201, 2 <081>
49, 208 <081>
28712 DATA 010, 169, 052, 157, 005, 001, 1 <081>
69, 249 <081>
28720 DATA 157, 006, 001, 165, 198, 240, 0 <081>
13, 201 <081>
28728 DATA 002, 176, 044, 173, 119, 002, 2 <081>
01, 003 <081>
28736 DATA 240, 037, 208, 047, 230, 198, 1 <081>
73, 255 <081>
28744 DATA 255, 141, 119, 002, 173, 071,-1 <081>
12, 208 <081>
28752 DATA 003, 206, 072,-112, 206, 071,-1 <081>
12, 173 <081>
28760 DATA 071,-112, 205, 118,-112, 208, 0 <081>
20, 173 <081>
28768 DATA 072,-112, 205, 119,-112, 208, 0 <081>
12, 173 <081>
28776 DATA 116,-112, 141, 020, 003, 173, 1 <081>
17,-112 <081>
28784 DATA 141, 021, 003, 076, 255, 255, 0 <081>
00, 000 <081>

```

Listing 1. Programm EDI.BAS/BATCH (Schluß)

holt sie aus dem Kommandotext das nächste Byte und legt es in den Tastaturpuffer. Die eigene Interrupt-Routine schaltet sich automatisch selbst ab, wenn:

- der ganze Kommandotext abgearbeitet ist,
- die Stoptaste gedrückt wird oder
- im Tastaturpuffer mehr als ein Tastendruck, also Eingaben des Benutzers, angetroffen werden.

Der Top-of-RAM-Pointer wird nicht zurückgesetzt!

Einschränkungen:

Es sind einerseits die RAM-Belegung und andererseits das Interrupt-Verhalten aufrufender Software zu beachten. Durch geeignetes Setzen des Top-of-RAM-Pointers vor Aufruf von BATCH 64 kann meist eine Kollision mit anderen Programmen in diesem Speicherbereich vermieden werden.

Da eine eigene Interrupt-Routine eingeschleift wird, kann es bei Programmen, die das auch tun, zu Kollisionen kommen. Bei wiederholten Aufruf von BATCH64 wird der Top-of-RAM-Pointer immer nur noch tiefer gesetzt, so daß er in vernünftigen Abständen wieder initialisiert werden sollte, zum Beispiel durch POKE56,160:clr

Als Detail sei noch auf die Parameterübergabe im RUN-Befehl hingewiesen, die durch die GET-Funktion aus dem Bildschirm bewerkstelligt wird. Dadurch sind Schreibweisen möglich, wie man sie sonst nur aus Betriebssystemkomman-

```

line
00002 0000 ;put"0:irqbat.src"
00003 0000 ;*
00004 0000 ;*****
00005 0000 ;*
00006 0000 ;* irq - bat
00007 0000 ;*
00008 0000 ;* batch-job-abwicklung ueber irq-routine
00009 0000 ;* version fuer 8xxx/40xx mit hinweisen fuer c 64, plus/4
00010 0000 ;*
00011 0000 ;* p. kittel, frankfurt 1985
00012 0000 ;*
00013 0000 ;*****
00014 0000 ;
00015 0000 irqptr = $fe90 ;pointer irq-routine (8xxx: 40090)
00016 0000 ; ; high-byte $fe=254 in datas korrigieren!
00017 0000 ; ; c-64: $0314, plus/4: $0212
00018 0000 ;
00019 0000 tasan = $9e ;anzahl in tastaturpuffer (8xxx)
00020 0000 ; ; c-64: $c6, plus/4: $ef
00021 0000 ;
00022 0000 taspuf = $026f ;tastaturpuffer (8xxx)
00023 0000 ; ; c-64: $0277, plus/4: $0527
00024 0000 ;
00025 0000 ;
00026 0000 ; z = $7000
00027 0000 ;
00028 7000 78 ; aktiv sei ; batch-routine aktivieren
00029 7001 ad 90 fe lda irqptr ;alten pointer retten
00030 7004 8d 74 70 sta irqend+1
00031 7007 ad 91 fe lda irqptr+1
00032 700a 8d 75 70 sta irqend+2
00033 700d ; ; statt 'start' bei c-64 und plus/4 start1 :
00034 700d a9 19 lda #start ;neuen setzen, bzw. start1
00035 7004 8d 90 fe sta irqptr
00036 7012 a9 70 lda #start ;bzw. start1 (-s.o.)
00037 7014 8d 91 fe sta irqptr+1
00038 7017 58 cli
00039 7018 60 rts ;fertig, gleich gehts los
00040 7019 ;
00041 7019 ;programmteil nur fuer cba 8/4xxx benoetigt:
00042 7019 bd 05 01 start lda $0105,x ;i/o-ende-schleife?
00043 701c 29 fc and #$fc ;ruecksprungadr pruefen
00044 701e c9 2c cmp #$2c
00045 7020 d0 11 bne start1 ;ne: nein
00046 7022 bd 06 01 lda $0106,x
00047 7025 c9 f9 cmp #$f9
00048 7027 d0 0a bne start1 ;ne: nein
00049 7029 a9 34 lda #$34 ;ruecksprungadr
00050 702b 9d 05 01 sta $0105,x ;manipulieren
00051 702e a9 f9 lda #$f9
00052 7030 9d 06 01 sta $0106,x
00053 7033 ;
00054 7033 a5 9e start1 lda tasan ;tastepuffer leer?
00055 7035 f0 0d beq holtas ;eq: ja, was dagegen tun
00056 7037 c9 02 cmp #2 ;mehr als eine taste?
00057 7039 b0 2c bcs reset ;cs: ja, abbruch wegen operator-eingriff
00058 703b ;
00059 703b ad 6f 02 lda taspuf ;zuletzt gedruckte taste
00060 703e c9 03 cmp #3 ;stop-taste?
00061 7040 f0 25 beq reset ;eq: ja, abbrechen
00062 7042 d0 2f bne irqend ;ne: einfach weiter

irqbat.....page 0003

line# loc code line
00063 7044 ;
00064 7044 e6 9e holtas inc tasan ;eine taste 'drucken'
00065 7046 ad ff ff holptr lda $ffff ;dummy-adresse wird modifiz.
00066 7049 8d 6f 02 sta taspuf ;in tastaturpuffer
00067 704c ad 47 70 lda holptr+1 ;low byte
00068 704f d0 03 bne nocar ;ne: kein uebertrag
00069 7051 ce 48 70 dec holptr+2 ;high byte
00070 7054 ce 47 70 nocar dec holptr+1 ;naechstes byte
00071 7057 ;
00072 7057 ad 47 70 lda holptr+1 ;abfrage auf ende
00073 705a cd 76 70 cmp endptr
00074 705d d0 14 bne irqend ;ne: nein
00075 705f ad 48 70 lda holptr+2
00076 7062 cd 77 70 cmp endptr+1
00077 7065 d0 0c bne irqend ;ne: nein, weiter
00078 7067 ;
00079 7067 ad 74 70 reset lda irqend+1 ;fertig, irq restaurieren
00080 706a 8d 90 fe sta irqptr
00081 706d ad 75 70 lda irqend+2
00082 7070 8d 91 fe sta irqptr+1
00083 7073 ;
00084 7073 4c ff ff irqend jmp $ffff ;dummy-adresse wird modifiziert
00085 7076 ;
00086 7076 00 endptr .byte 0,0 ;hier pointer auf (letztes byte -1)
00087 7077 00 ;
00088 7078 ;
00089 7078 .end

```

Listing 2. Der Sourcecode für BATCH 64

dos anderer, größerer Computer kennt.

Zur Anpassung an andere Computer sind die DATA-Zeilen mit der Maschinenroutine (siehe unten) und die Zeilen 700 und 805 mit der Adresse des Top-of-Memory-Pointers zu verändern. Letztere ist auch zu beachten, wenn man diesen Pointer ab und zu wieder restauriert:

```
POKE56,160:CLR:REM C 64
```

```
POKE56,128:CLR:REM PLUS/4
```

```
POKE53,128:CLR:REM 8xxx/40xx
```

Daß man beim PLUS/4 nur die ersten 32 KByte des Speichers ausnutzen kann (siehe auch Zeile 701, Listing 1!), liegt daran, daß man bei höherer Plazierung der eigenen Interrupt-Routine noch Bank-Switching zwischen paralleliegendem RAM und ROM ausführen müßte. Diesen Ausbau der Routine will ich gern interessierten Lesern überlassen.

Noch etwas zum Verfahren: Der Kommandotext muß ins RAM gelegt werden, weil zum Zeitpunkt der Abarbeitung durchaus schon alle Floppy-Kanäle (beziehungsweise Puffer) belegt sein können, so daß kein direkter Zugriff auf das Kommandofile mehr möglich ist. Der Top-of-RAM-Pointer darf auch nicht automatisch beim Ausklinken der Interrupt-Routine zurückgestellt werden, denn anschließend müßte ja ein CLR-Befehl gegeben werden, der Variablen eines womöglich nicht in vollem Lauf befindlichen Basic-Programms einfach zerstören würde!

Theoretischerweise kann man in BATSCH 64 auch die Routine EDI-BAS weglassen und durch eine andere ersetzen, oder man benutzt einen externen Texteditor zur Erstellung der Kommandofiles. Auf jeden Fall muß das File anschließend reinen ASCII-Text enthalten, also nicht Bildschirmcode oder Vorspanne mit Tab- und Formatier-Informationen.

Listing 2 ist das Assembler-Listing der von BATCH verwendeten Maschinen-Routine, und zwar in der 8xxx/40xx-Version. Die Version für den C 64 können Sie den Hinweisen und den DATAs aus Listing 1 entnehmen. Hinweise gibt es auch zur Anpassung an den Plus/4.

Der Quelltext ist mit Gewalt (Adresse \$fe90 statt \$90 für 8xxx, das Byte \$fe=254 muß nachträglich zu Fuß in den DATAs auf 000 korrigiert werden!) so gestaltet, daß er für alle Computer die gleiche Länge hat. Die Anpassung an andere kann also direkt in den DATA-Zeilen von Listing 2 erfolgen, wenn man im Assembler-Listing die Änderungen und Vorkommen der fraglichen Werte herausucht.

Nur für 8xxx/40xx ist der Programmteil aktiv, der sich um die I/O-Ende-Routine kümmert. Dahinter verbirgt sich eine ganz böse Falle für jeden Interrupt-Routinen-Programmierer. Auf besagtem Rechner wird nämlich nach jeder Ein-/Ausgabe eine Routine aufgerufen, die einfach so lange wartet, bis der Interrupt-Pointer wieder seinen Normalwert hat! Das kommt von den Kassettenroutinen, deren Timing über eigene Interrupt-Routinen gemanagt wird. Wenn man den Pointer nun aber selbst verbogen hat und zum Beispiel einen LOAD-Befehl gibt, bleibt der Computer erst mal hängen, läßt sich aber immerhin mit der Stopptaste zurückholen. Zur Abhilfe schaut nun in unserer Routine ein Programmteil auf dem Stack nach, ob die Rücksprungadresse womöglich irgendwo in diese Warteschleife zeigt. Wenn ja, wird die Rücksprungadresse brutal auf den nächsten RTS-Befehl umgesetzt.

Testlauf

Erstellen Sie am besten ein Kommandofile namens TEST.BAT mit folgenden zwei Zeilen Inhalt (Kleinschreibung wegen Groß/Klein-Modus des Editors):

```
list-5
```

```
run:test-bat
```

Wenn Sie BATCH 64 mit RUN:TEST.BAT aufrufen, müßte sich ein immer wieder selbst anstartender Ablauf ergeben, der erst durch OUT OF MEMORY ERROR gebremst wird, wenn Top-of-Memory nach wenigen Stunden zu klein geworden ist.

(Dr. P. Kittels/bs)