```
7355 IF I>9 THEN I=9:GOTO 7325                    <004>
7360 IF I<0 THEN I=0                              <101>
7365 GOTO 7325                                    <059>
7370 IF A=29 THEN POKE AD+I,PEEK(AD+I) AND
     127:GOTO 7350                                <023>
7375 IF A=157 THEN POKE AD+I,PEEK(AD+I) AN
     D 127:I=I-1:GOTO 7355                        <173>
7380 IF A<>13 THEN 7330                           <248>
7385 POKE AD+I,PEEK(AD+I) AND 127                 <004>
7390 SYS PR,14,13,F1$;"NAMEN EINGEBEN (RET
     URN)"                                        <126>
7395 RETURN                                       <141>
7400 REM---------------------------------         <006>
7402 REM PARAMETERSATZ NS HOLEN                   <143>
7404 SA=SO+NS*107:SYS GE,SA                       <163>
7406 FOR I=0 TO 2                                 <140>
7408 : AD=SA+7*I                                  <231>
7410 : BF(I)=USR(AD+73)                           <189>
7412 : X=BF(I)/17.0327/440                        <202>
7414 : Y=LOG(X)/LOG(2) :REM X=2↑Y                 <239>
7416 : Y=Y+4+9/12+49/1200                         <010>
7418 : O(I)=INT(Y):Y=(Y-O(I))*12                  <220>
7420 : T(I)=INT(Y)                                <081>
7422 : OF(I)=INT((Y-T(I))*100-48.5)               <053>
7424 : S(I)=(PEEK(AD+75)=1)                       <216>
7426 : A$=CHR$(49+I):GOSUB 3570                   <126>
7428 : C(I)=PEEK(AD+77) AND 254                   <228>
7430 : C1(I)=C(I) OR 1                            <219>
7432 : AD(I)=PEEK(AD+78)                          <225>
7434 : SR(I)=PEEK(AD+79)                          <004>
7436 NEXT I                                       <244>
7438 RF=PEEK(SA+94)                               <051>
7440 EG=(PEEK(SA+95)=1):GOSUB 3740                <243>
7442 SU=(PEEK(SA+96)=1):GOSUB 3840                <019>
7444 RETURN                                       <191>
7450 REM---------------------------------         <057>
7452 REM PARAMETERSATZ NS ABSPEICHERN             <112>
7454 SA=SO+NS*107:SYS PU,SA                       <239>
7456 FOR SN=0 TO 2                                <247>
7458 : AD=SA+7*SN                                 <114>
7460 : SYS DO,AD+73,BF(SN)                        <023>
7462 : POKE AD+75,-S(SN)                          <203>
7464 : POKE AD+77,C(SN)                           <020>
7466 : POKE AD+78,AD(SN)                          <089>
7468 : POKE AD+79,SR(SN)                          <124>
7470 NEXT SN                                      <110>
7472 POKE SA+94,RF                                <083>
7474 POKE SA+95,-EG                               <245>
7476 POKE SA+96,-SU                               <020>
7478 RETURN                                       <225>
7500 REM=================================         <091>
7505 REM UNTERMENUE DISKETTE                      <098>
7510 REM=================================         <101>
7515 M=7:PV=8:SYS CL:PRINT"{HOME}";F1$            <215>
7520 PRINT" DISK"                                 <133>
7525 SYS PR,5,4,F2$;"F1"                          <065>
7530 SYS PR,5,7,F1$;"SOUNDS LADEN"                <017>
7535 SYS PR,7,4,F2$;"F3"                          <079>
7540 SYS PR,7,7,F1$;"SOUNDS ABSPEICHERN"          <221>
7545 RETURN                                       <036>
7600 REM---------------------------------         <207>
7605 REM SOUNDS LADEN                             <161>
7610 SYS PR,5,7,F2$;"SOUNDS LADEN"                <098>
7615 SYS PR,10,4,F1$;"DATEINAME ";                <018>
7620 DN$="SOUNDS{14SPACE}"                        <105>
7625 SYS PR,10,16,DN$                             <040>
7630 SYS PR,10,14,;:INPUT DN$                     <037>
7635 OPEN 8,8,8,DN$+",P,R":CLOSE 8                <107>
7640 OPEN 1,8,15:INPUT#1,A,A$,X,Y:CLOSE 1         <188>
7645 SYS PR,12,4,"{29SPACE}"                      <153>
7650 IF A=0 THEN 7660                             <039>
7655 SYS PR,12,3,A;A$;X;Y:GOTO 7625               <253>
7660 A=0:LOAD DN$,8,1                             <112>
7665 SYS PR,5,7,F1$;"SOUNDS LADEN"                <152>
7670 NS=-1:A=211:GOTO 1550                        <145>
7700 REM---------------------------------         <052>
7705 REM SOUNDS ABSPEICHERN                       <198>
7710 SYS PR,7,7,F2$;"SOUNDS ABSPEICHERN"          <137>
7715 SYS PR,10,4,F1$;"DATEINAME ";                <119>
7720 DN$="SOUNDS{14SPACE}"                        <206>
7725 SYS PR,10,16,DN$                             <141>
7730 SYS PR,10,14,;:INPUT DN$                     <138>
7735 SYS 50198:REM MODULATOR AUS                  <099>
7740 REM BEREICH $8000-$8A00 AUF DISK             <000>
7745 OPEN 1,8,1,"@:"+DN$                          <058>
7750 POKE 252,0:POKE 253,144:REM $9000            <214>
7755 POKE 780,252:REM AKKU                        <089>
7760 POKE 781,8  :REM X-REG = $08                 <254>
7765 POKE 782,154:REM Y-REG = $9A                 <121>
7770 SYS 65496    :REM KERNAL SAVE                <217>
7775 CLOSE 1                                      <078>
7780 OPEN 1,8,15:INPUT#1,A,A$,X,Y:CLOSE 1         <073>
7785 IF A=0 THEN 7795                             <184>
7790 SYS PR,12,4,A;A$;X;Y:GOTO 7725               <135>
7795 SYS PR,12,4," {30SPACE}"                     <048>
7800 SYS PR,7,7,F1$;"SOUNDS ABSPEICHERN"          <226>
7805 SYS 50185:RETURN:REM MODULATOR AN            <019>
8000 REM=================================         <081>
8010 REM VORBESETZUNG, INITIALISIERUNG            <066>
8020 REM=================================         <101>
8030 PRINT"{CLR,12DOWN,7SPACE}";                  <244>
8035 PRINT"BITTE 30 SEKUNDEN WARTEN"              <104>
8040 REM---------------------------------         <137>
8045 REM MS-PROGRAMME IN KASSETTENPUF.            <244>
8050 FOR I=0 TO 22   :REM KERNAL-GOSUB            <120>
8055 POKE 828+I,PEEK(43139+I):NEXT                <180>
8060 FOR I=0 TO 129:READ X:POKE 851+I,X:NE
     XT                                           <066>
8065 POKE 785,169:POKE 786,3:REM USR              <063>
8070 REM---------------------------------         <167>
8072 REM MS-PROGRAMME AB $C423                    <048>
8074 FOR I=50211 TO 50291                         <195>
8076 READ X:POKE I,X:NEXT                         <228>
8078 REM---------------------------------         <175>
8080 REM NAMEN DER TOENE                          <003>
8082 FOR I=0 TO 11:READ TN$(I):NEXT               <064>
8100 REM---------------------------------         <197>
8110 REM SPRUNGVEKTOREN                           <169>
8120 READ A$:L=LEN(A$)                            <116>
8130 ON L GOTO 8150,8220,8280,8310                <003>
8150 REM L=1  ASCII-ZEICHEN                       <218>
8160 A=ASC(A$)                                    <110>
8170 IF M>=0 THEN V%(M,A)=ZN:GOTO 8120            <137>
8180 FOR I=0 TO 7:V%(I,A)=ZN:NEXT                 <108>
8190 GOTO 8120                                    <113>
8200 IF LEFT$(A$,1)<>"M" THEN 8120                <213>
8220 REM L=2  MENUE-INDIKATOR                     <178>
8230 IF LEFT$(A$,1)<>"M" THEN 8120                <243>
8240 A$=RIGHT$(A$,1)                              <035>
8250 IF A$="A" THEN M=-1 :GOTO 8120               <113>
8260 M=VAL(A$)           :GOTO 8120               <076>
8280 REM L=3  ASCII-WERT                          <123>
8290 A=VAL(A$) :GOTO 8170                         <099>
8310 REM L=4  ZEILENNUMMER ODER ENDE              <163>
8320 IF A$="ENDE" THEN 8340                       <248>
8330 ZN=VAL(A$) :GOTO 8120                        <237>
8340 REM---------------------------------         <150>
8350 REM TONNUMMERN                               <064>
8360 A$="Q2W3ER5T6Y7UI9OOP@-*£↑{HOME};="          <120>
8370 FOR I=0 TO 24                                <104>
8380 : A=ASC(MID$(A$,I+1,1))                      <045>
8390 : TN%(A)=I                                   <084>
8400 : FOR J=0 TO 7:V%(J,A)=2000:NEXT             <127>
8410 NEXT I                                       <165>
8450 REM---------------------------------         <037>
8455 REM FREQUENZEN DER TONLEITER                 <075>
8460 FOR I=0 TO 24:TH(I)=2↑(I/12):NEXT            <075>
8465 REM---------------------------------         <052>
8470 REM KONSTANTEN                               <203>
8472 GS=828:GT=857:PR=869:DO=897:CL=959           <208>
8474 NA=50229:GE=50245:PU=50276                   <139>
8480 SI=54272:MO=49152:FA=55296                   <126>
8482 SO=36864                                     <162>
8484 F1$=CHR$(154):F1=14                          <012>
8490 F2$=CHR$(5)  :F2=1                           <123>
8495 KV$(0)="TRIAN ":KF$(0)="{4AAN}"              <166>
8500 KV$(1)="SAWUP ":KF$(1)="{4AL}"               <189>
8505 KV$(2)="SAWDWN":KF$(2)="{4AGN}"              <226>
8510 KV$(3)="SQUARE":KF$(3)="{4LOL}"              <014>
8515 MD$(0)="RESET" :MD$(1)="RESET"               <253>
8520 MD$(2)="HOLD " :MD$(3)="RUN{2SPACE}"         <028>
8525 BL$(0)="":BL$(1)=" ":BL$(2)="{2SPACE}
     "                                            <096>
8530 BL$(3)="{3SPACE}":BL$(4)="{4SPACE}"          <006>
8535 BL$(5)="{5SPACE}":BL$(6)="{6SPACE}"          <015>
8540 REM---------------------------------         <127>
8545 REM SOUNDPARAMETER VORBESETZUNGEN            <124>
8550 FOR SN=0 TO 2                                <097>
8555 : POKE SI+7*SN+5,10 :AD(SN)=10               <087>
8560 : POKE SI+7*SN+6,138:SR(SN)=138              <243>
8565 : O(SN)=3+SN:OF(SN)=0:T(SN)=0                <070>
8570 : BF=440*2↑(O(SN)-4+(T(SN)-9)/12+OF(S
     N)/1200)                                     <122>
8572 : BF(SN)=BF*17.0327                          <153>
8575 :  C(SN)=64          :C1(SN)=65              <232>
8580 :  S(SN)=(SN=0)                              <170>
8585 : SYS DO,MO+7*SN+2,2048 :REM PW              <081>
8590 : POKE  MO+7*SN+4,0     :REM PORTA           <211>
8595 NEXT SN                                      <215>
8600 POKE  SI+23,0:RF=0  :REM RES/FILT            <028>
8605 SYS DO,MO+21,0      :REM FILTERFR.           <117>
8610 POKE  MO+23,15      :REM MOD/LAUT            <165>
8615 FOR I=24 TO 72 :REM KSV,LFO,EG AUS           <034>
8620 : POKE MO+I,0:NEXT                           <069>
8625 EG=0           :REM EG ABKOPPELN             <049>
8630 SU=0           :REM SUSTAIN AUS              <058>
8635 REM---------------------------------         <222>
8640 REM SOUNDS VORBESETZEN                       <163>
8645 FOR NS=0 TO 23                               <243>
8650 : GOSUB 7450                                 <130>
8655 : AD=SO+107*NS+97                            <172>
8660 : FOR I=AD TO AD+9:POKE I,46:NEXT            <168>
8665 NEXT NS:NS=0                                 <218>
8700 REM---------------------------------         <031>
8870 REM FARBEN, BILD, MODULATOR                  <047>
8880 POKE 53280,14:POKE 53281,6                   <050>
8890 PRINT F1$;"{RVOFF}";:POKE 650,128            <143>
8900 GOSUB 4200 :REM BILD ZEILEN 15-23            <220>
8910 SYS MO+1033:REM MODULATOR AKTIV.             <200>
8920 A=2:RETURN                                   <231>
9000 REM=================================         <061>
9010 REM DATAS                                    <081>
9020 REM=================================         <081>
9030 REM MASCHINENPROGRAMME                       <056>
9060 DATA 032,089,003,076,174,167,032             <071>
9070 DATA 253,174,032,138,173,032,247             <080>
9080 DATA 183,076,163,168,032,253,174             <099>
9090 DATA 032,158,183,138,072,032,253             <099>
9100 DATA 174,032,158,183,138,168,104             <117>
9110 DATA 170,024,032,240,255,032,253             <104>
9120 DATA 174,076,160,170,032,253,174             <129>
9130 DATA 032,138,173,032,247,183,165             <142>
9140 DATA 020,133,158,165,021,133,159             <142>
9150 DATA 032,253,174,032,138,173,032             <152>
9160 DATA 247,183,160,000,120,165,020             <151>
9170 DATA 145,158,200,165,021,145,158             <177>
9180 DATA 088,096,032,247,183,160,000             <190>
9190 DATA 120,177,020,133,099,200,177             <194>
9200 DATA 020,088,133,098,162,144,056             <213>
9210 DATA 076,073,188,169,032,162,000             <222>
9220 DATA 157,000,004,157,000,005,202             <202>
9230 DATA 208,247,162,087,157,000,006             <239>
9240 DATA 202,016,250,096                         <164>
9250 REM---------------------------------         <072>
9260 REM MASCHINENPROGRAMME AB $C423              <162>
9270 DATA 032,253,174,032,138,173,032             <017>
9280 DATA 247,183,165,020,133,158,165             <039>
9290 DATA 021,133,159,096,032,035,196             <046>
9300 DATA 160,000,177,158,032,210,255             <043>
9310 DATA 200,192,010,208,246,096,032             <054>
9320 DATA 035,196,160,000,177,158,153             <076>
9330 DATA 000,192,200,192,073,208,246             <075>
9340 DATA 177,158,153,183,211,200,192             <095>
9350 DATA 094,208,246,177,158,141,023             <112>
9360 DATA 212,096,032,035,196,160,000             <103>
9370 DATA 185,000,192,145,158,200,192             <121>
9380 DATA 073,208,246,096                         <062>
9400 REM---------------------------------         <222>
9410 REM  NAMEN DER TOENE                         <058>
9420 DATA C,C#,D,D#,E,F,F#,G,G#,A,A#,H            <061>
9500 REM---------------------------------         <067>
9510 REM DATAS FUER SPRUNGVERTEILER               <010>
9520 DATA MA,4000,X,3500,V,3600,032               <006>
9530 DATA MA,3700,160,3800,S                      <159>
9540 DATA MA,2550,133,2560,134                    <242>
9550 DATA MA,2570,135,2580,136,4500,214           <188>
9555 DATA MA,5500,L,6000,H,6500,K                 <203>
9560 DATA MA,5000,F,7000,211,7500,D               <012>
9565 DATA MA 1,4654,145,4660,017                  <008>
9570 DATA MA1,4666,029,4680,157                   <024>
9580 DATA MA2,5160,157,5145,029                   <022>
9600 DATA MA3,5675,145,5690,017                   <053>
9610 DATA MA3,5720,157,5705,029                   <057>
9620 DATA MA4,6410,157,6440,029                   <062>
9630 DATA MA5,6830,145,6850,017                   <078>
9640 DATA MA5,6870,157,6890,029                   <102>
9650 DATA MA5,6935,133,134,6975,135,136           <248>
9700 DATA MA6,7110,A,B,C,7200,+,7300,N            <062>
9750 DATA MA7,7600,133,7700,134                   <189>
9900 DATA ENDE                                    <113>
```

Listing »Sound-Editor« (Schluß)

Memory Map mit Wandervorschlägen (Teil 8)

Neben der dynamischen Tastaturabfrage werden die Adressen 153 bis 158 behandelt. Deren Aufgabe ist die Ausgabe von Fehlermeldungen.

Die dynamische Tastaturabfrage ist das Kernthema mit dem wir uns heute beschäftigen wollen. Zuvor aber noch einige Speicherstellen, die wie beim letzten Mal das Betriebssystem als Zeiger für Ein/Ausgabeoperationen benutzt.

Adresse 153 ($99)

Nummer des Eingabe-Gerätes

Das Betriebssystem verwendet diese Speicherzelle, um festzuhalten, welches Gerät zur Eingabe verwendet werden soll.

Die Nummern sind wie folgt festgelegt:

0	= Tastatur
1	= Datasette
2	= RS232 (User-)Port
3	= Bildschirm
4, 5	= Drucker
8-11	= Floppy-Laufwerke

Nach dem Einschalten oder nach RESET des Computers steht in 153 eine 0 (Tastatur). Nach jedem Einsatz eines anderen Gerätes wird diese Speicherzelle wieder auf 0 gesetzt, so daß wir immer die Tastatur zur Verfügung haben.

Für Maschinenprogrammierer ist diese Adresse sicherlich wertvoll. Die Routine, welche die Eingabegeräte festlegt, sobald der Befehl INPUT# beziehungsweise GET# ausgeführt wird, heißt CHKIN und beginnt beim C 64 ab Adresse 61966 ($F20E), beim VC 20 ab 62151 ($F2C7).

Für Basic-Programmierer habe ich in der Literatur nur eine Anwendung gefunden, und die wurde bereits bei der Besprechung der Speicherzelle 19 (Teil 3 des Kurses in Ausgabe 1/85, Seite 127) angekündigt.

Es ist dies eine MERGE-Routine. Leider funktioniert dieses Verfahren nicht bei dem 1541 Floppy-Laufwerk. Erfunden wurde die Routine von Brad Templeton und ist von Jim Butterfield unter dem Namen »Magic

Merge« für den VC 20/C 64 adaptiert worden. Ich gebe zu, in der Zwischenzeit sind noch andere, vielleicht auch kürzere MERGE-Routinen veröffentlicht worden. Aber diese hier verwendet gleich drei interessante Ingredienzen, nämlich die Speicherzellen 19 und 153, außerdem die sogenannte »Dynamische Tastenabfrage«. Wer die letztere nicht kennt, sollte sich zum Verständnis den Texteinschub 1 gleichen Namens ansehen.

Ein MERGE (deutsch: zusammenführen, verschmelzen) besteht darin, ein auf Band gespeichertes Programm zu einem im Computer stehenden anderen Programm so dazuzuladen, daß dieses nicht überschrieben, sondern ergänzt wird. Wichtig ist dabei, daß das Programm vom Band höhere Zeilennummern hat als das Programm im Computer. Außerdem muß das Programm auf dem Band als Datei gespeichert sein. Das wird so erreicht:

1. **Programm eintippen**
2. **Direkt eingeben:**
   OPEN 1,1,1,"Name":CMD1:LIST
3. **Erst wenn READY kommt, direkt eingeben**
   PRINT # 1:CLOSE1

Damit ist das Programm auf dem Band gespeichert. Nun kommt der eigentliche MERGE-Vorgang.

4. **Es steht ein Programm im Computer**
5. **Band mit dem Programm »Name« einlegen**
6. **Direkt eingeben:**
   POKE 19,1:OPEN 1
7. **Sobald READY erscheint, Bildschirm löschen (SHIFT-CLR).**
8. **Dreimal Cursor-Down**
9. **Direkt eingeben:**
   PRINT CHR$(19):POKE 198,1:
   POKE 631,13:POKE 153,1
10. **Das Band beendet den Ladevorgang mit einer Fehlermeldung, die wir ignorieren.**
11. **Nach CLOSE 1 sind beide Programme zusammengefügt.**

Wie gesagt, Schritt 6 verwendet Zelle 19 (bitte dort nachlesen), Schritte 8 und 9 sind die »Dynamische Tastenabfrage«, und Schritt 9 verwendet zusätzlich die hier zur Diskussion stehende Speicherzelle 153, um die Datasette als Eingabegerät zu definieren.

Adresse 154 ($9A)

Nummer des Ausgabe-Gerätes

Diese Speicherzelle entspricht der Zelle 153, nur steht hier die Nummer des Gerätes, über das die Ausgabe läuft.

Nach dem Einschalten und nach Ausgabeoperationen wird der Wert immer auf 3 gesetzt. Das ist entsprechend der oben genannten Zuordnung der Bildschirm.

Für Maschinenprogrammierer sei erwähnt, daß Basic bei den Befehlen PRINT# oder CMD die Routine CHKOUT einsetzt, welche die Adresse 154 belegt. Sie steht im C 64 ab Adresse 62032 ($F250), im VC 20 ab 62217 ($F309).

Adresse 155 ($9B)

Fehlerkontrolle bei Bandoperationen

Die Commodore-Datasette ist deswegen so zuverlässig, weil sie mehrere Methoden zur Fehlererkennung beziehungsweise Korrektur von Lese- und Schreibfehlern verwendet.

Eine der Methoden ist die sogenannte Parity-Prüfung. Sie ist nichts anderes als eine Quersummenbildung der einzelnen Stellen jedes Bytes, deren Resultat überprüft wird.

Die Speicherzelle 155 wird bei dieser Parity-Prüfung eingesetzt.

Adresse 156 ($9C)

Flagge für korrektes Byte vom Band

In dieser Speicherzelle wird zwischengespeichert, ob das vom Band gelesene Byte die Prüfungen bestanden hat, also richtig ist oder nicht.

Adresse 157 ($9D)

Flagge für Meldungen

Man muß zwischen zwei Arten von Meldungen unterscheiden:
— Meldungen des Betriebssystems
— Meldungen des Basic-Übersetzers

Die Meldungen des Betriebssystems kennen wir als Angaben zum Ablauf, wie SEARCHING FOR, FOUND, PRESS PLAY ON TAPE und so weiter. Normalerweise nicht bekannt ist die Meldung I/O ERROR #, wobei nach dem Zeichen # Zahlen von 0 bis 29 stehen können. Diese Zahlen beziehen sich auf Meldungen des Übersetzers (Interpreter), die ausschließlich Fehlermeldungen sind. Das mag verwirrend klingen, klärt sich aber sofort. Die Flagge in 157 kann vier Werte annehmen, 0, 64, 128 und 192.

1. Der Wert 0 unterdrückt alle Meldungen des Betriebssystems. Dieser Modus tritt nach RUN beim Ablauf eines Programms ein.
2. Der Wert 64 läßt nur Fehlermeldungen des Betriebssystems zu. Dieser Modus ist normalerweise nicht vorgesehen, kann aber künstlich erzeugt werden.
3. Der Wert 128 unterdrückt die Fehlermeldung des Betriebssystems. Dieser Modus entspricht dem Normalfall.
4. Der Wert 192 läßt alle Meldungen zu. Auch dieser Modus ist nur künstlich herzustellen.

Das folgende Beispiel macht das deutlich. Geben Sie direkt ein:

```
POKE 157,0:LOAD "$",9
```

Wir versuchen, vom Gerät mit der Nummer 9, das ist eine zweite Floppy, die Directory zu laden. Wir erhalten entsprechend Punkt 1 nur die Meldung des Übersetzers »? DEVICE NOT PRESENT«. Verändern wir den POKE-Befehl für Punkt 2:

```
POKE 157,64:LOAD"$",9
```

Wir erhalten jetzt »I/O ERROR #5 ? DEVICE NOT PRESENT«.

```
POKE 157,128:LOAD"$",9
```

ergibt die Meldung »SEARCHING FOR $
? DEVICE NOT PRESENT«.

Schließlich nehmen wir noch den letzten Fall:

```
POKE 157,192:LOAD"$",9
```

Jetzt erhalten wir alles:

```
»SEARCHING FOR $
I/O ERROR #5
? DEVICE NOT PRESENT«
```

Da die Fehlermeldung des Betriebssystems und die zugehörigen Nummern in keinem Handbuch erwähnt sind, habe ich sie interessehalber in der folgenden Tabelle zusammengefaßt.

#	MELDUNG (ERROR)
1	TOO MANY FILES
2	FILE OPEN
3	FILE NOT OPEN
4	FILE NOT FOUND
5	DEVICE NOT PRESENT
6	NOT INPUT FILE
7	NOT OUTPUT FILE
8	MISSING FILE NAME
9	ILLEGAL DEVICE NUMBER
10	NEXT WITHOUT FOR
11	SYNTAX
12	RETURN WITHOUT GOSUB
13	OUT OF DATA
14	ILLEGAL QUANTITY
15	OVERFLOW
16	OUT OF MEMORY
17	UNDEF'D STATEMENT
18	BAD SUBSCRIPT
19	REDIM'D ARRAY
20	DIVISION BY ZERO
21	ILLEGAL DIRECT
22	TYPE MISMATCH
23	STRING TOO LONG
24	FILE DATA
25	FORMULA TOO COMPLEX
26	CAN'T CONTINUE
27	UNDEF'D FUNCTION
28	VERIFY
29	LOAD

Das nächste Mal fahren wir ab Speicherzelle 158 fort.

(Dr. Helmuth Hauck/ah)

Texteinschub 1. Dynamische Tastenabfrage

Jedesmal wenn Sie eine Taste drücken, wird der ASCII-Codewert des Zeichens oder der Funktion dieser Taste ermittelt und im »Tastaturpuffer« gespeichert. Dieser Pufferspeicher liegt in den Speicherzellen 631 bis 640.

Die eigentliche Abfrage und Umcodierung eines Tastendrucks habe ich im Kurs »Alle Tasten-, Zeichen- und Steuercodes« in den Ausgaben 4/84 bis 7/84 beschrieben. Die ASCII-Codetabelle finden Sie da auch. Sie ist natürlich in allen Handbüchern enthalten, leider aber nicht immer ganz vollständig. Zuerst will ich Ihnen die Arbeitsweise des Tastaturpuffers zeigen.

Der Computer holt sich den ASCII-Codewert aus dem Tastaturpuffer und wenn gerade kein Programm läuft, druckt er das Zeichen auf den Bildschirm oder führt die Funktion der Taste aus. Das ist der oft zitierte »Direkt-Modus«.

Wenn aber ein Programm läuft, dann bleiben die Codezahlen im Puffer so lange stehen, bis der Computer fertig ist. Dann erst werden sie herausgeholt und verarbeitet. Das will ich Ihnen beweisen.

Tippen Sie im Direkt-Modus ein:

```
FOR K=1 TO 15000:NEXT K (RETURN)
```

Während diese an sich sinnlose Zeitschleife 15000mal im Kreise rumrennt, haben Sie genügend Zeit, mehrere Tasten zu drücken, zum Beispiel die erste Buchstabenreihe (QWERTYUIOP@*↑). Natürlich sehen Sie am Bildschirm gar nichts, denn das Programm der Schleife läuft ja noch. Sobald aber die Schleife zu Ende ist, erscheinen 10 der getippten Buchstaben. Quod erat demonstrandum!

Warum nur 10 Buchstaben? Nun, der Tastaturpuffer hat nur 10 Plätze, logisch?

Jetzt müssen wir noch eine weitere Speicherzelle ins Spiel bringen. In die Zelle 198 kann man nämlich eine Zahl hineinPOKEn, welche die Anzahl der Zeichen im Tastaturpuffer begrenzt.

Wiederholen Sie bitte das Experiment von oben, nur soll die direkt eingegebene Zeile erweitert werden:

```
FOR K=1 TO 15000:NEXT K: POKE 198,6 (RETURN)
```

Und siehe da, jetzt werden nur die 6 Buchstaben Q bis Y ausgedruckt. Diese Anwendung des Tastaturpuffers nutzen wir für das Kochrezept »**Dynamische Tastenabfrage**« aus. Allerdings müssen wir dazu prinzipiell die ASCII-Codewerte verwenden, so wie im nächsten Beispiel.

Löschen Sie bitte den Bildschirm und geben Sie ein (identisch für VC 20 und C 64):

```
10 PRINT CHR$(65)
20 PRINT CHR$(156)
30 PRINT CHR$(66)CHR$(13)CHR$(67)
```

65 ist der Code für A, 156 für die Farbe »purple«, 66 für B, 13 für »RETURN« und 67 für C. Das heißt also, daß wir zuerst ein A drucken, dann auf »purple« umschalten, darunter dann das B schreiben, einmal RETURN geben und dann noch das C folgen lassen.

Bild 1 zeigt den Ausdruck auf dem Bildschirm, wenn Sie diese Zeilen LISTen und starten.

Zur Erklärung: Die Leerzeile zwischen A und B ist bedingt durch die PRINT-Anweisung in Zeile 20, welche nur die Farbe umschaltet. Obwohl die Codes für B und C zusammen in einer Zeile stehen, werden sie doch durch das »RETURN« (13) untereinandergesetzt.

Anstelle der 13 können Sie alle mögliche anderen Steuerfunktionen setzen. Bild 2 zeigt das Resultat von 17, nämlich »Cursor down«. Wenn Sie die 8 nehmen, können Sie den Zeichensatz nicht mehr ändern. Der Einsatz der gleichzeitig gedrückten SHIFT- und Commodore-Taste funktioniert erst nach CHR$(9) wieder.

Als nächstes wollen wir die ASCII-Codewerte und den Tastaturpuffer im »Programm-Modus« einsetzen.

Das Resultat von Bild 2 wollen wir jetzt durch POKEn der ASCII-Werte in den Tastaturpuffer wiederholen.

```
 5  POKE 198,5
10  POKE 631,65
20  POKE 632,156
30  POKE 633,66: POKE 634,17: POKE 635,67
99  END
```

Prinzipiell macht dieses Programm das gleiche wie das Programm in Bild 2. Trotzdem erhalten wir nach LIST und RUN ein anders Ergebnis, nämlich das von Bild 3.

Ist das ein Fehler? Natürlich nicht. Nach RUN laufen zuerst mal alle POKE-Befehle ab. Zeile 5 gibt an, wieviele Zeichen im Puffer stehen sollen. In Zeile 99 findet das Programm das ENDe und meldet sich mit READY. Jetzt erst wird im Tastaturpuffer nachgeschaut. Dort findet der Computer zuerst das A, dann »purpur«, dann das B welches sofort neben das A gesetzt wird. Das ist auch logisch, denn es fehlt ja jede Angabe, eine Zeile tiefer zu gehen. Um das zu erreichen, müssen wir in der Zeile 10 den Codewert für RETURN einschieben:

10 POKE 631,65: POKE 632,13

Vorsicht! Sie müssen in Zeile 20 und 30 alle POKE-Adressen um 1 erhöhen und auch die Zahl in Zeile 5. Nehmen sie 10, dann haben Sie Platz für Erweiterungen. So, jetzt LIST und RUN eingeben und es erscheint Bild 4 — und wir haben schon wieder ein Problem! Aber alles im Computer ist logisch! Nach dem A findet er den Wert für »RETURN« also führt er den Befehl aus, auf dem der Cursor gerade steht. Der steht auf dem A. Da das kein gültiger Basic-Befehl ist, druckt der Computer die Fehler-Meldung und zeigt READY an. Danach allerdings macht er weiter wie oben.

```
LIST

10  PRINT CHR$(65)
20  PRINT CHR$(13)
30  PRINT CHR$(66) CHR$
    (13) CHR$(67)

READY.

RUN
A

B
C
READY.
```

Bild 1. Testprogramm

```
LIST

10  PRINT CHR$(65)
20  PRINT CHR$(17)
30  PRINT CHR$(66) CHR$
    (17) CHR$(67)

READY.

RUN
A
B
  C
READY.
```

Bild 2. CR ersetzt durch »Cursor down«

```
LIST
5  POKE 198,5
10  POKE 631,65
20  POKE 632,156
30  POKE 633,66:POKE 63
4,17:POKE 635,67
99  END
READY.

RUN
READY.
AB
  C
```

Bild 3. Programm (Bild 2) in den Tastaturpuffer gePOKEt

Und jetzt kommt die ASCII-Zahl 141 (SHIFT-RETURN) voll zur Geltung. Diese Kombination nämlich setzt den Cursor an den Anfang der nächsten Zeile, ohne den Befehl »RETURN« auszuführen.

Ersetzen Sie also die 13 in Zeile 10 durch 141, dann läuft's (Bild 5).

Es gibt übrigens noch eine interessante ASCII-Codezahl, die in keiner Tabelle steht, nämlich 131. Sie bedeutet dasselbe wie die geSHIFTete STOP-Taste, also die Funktion »LOAD + RUN«.

Wenn Sie diesen Code mit PRINT CHR$(131) ausprobieren, funktioniert er allerdings nicht. Deshalb steht er wohl auch nicht in den Tabellen.

In den Tastaturpuffer gePOKEt bringt er aber seine Wirkung.

Setzen Sie bitte in Zeile 30 an die Stelle von 67 die Zahl 131 und anstelle des C erscheint
LOAD und PRESS PLAY ON TAPE. Gut, nicht wahr?

So, jetzt haben wir alle Zutaten für unser Kochrezept zusammen. Löschen Sie bitte alles bisherige und tippen Sie ein:

```
10  PRINT CHR$(147)
20  FOR I=1 TO 5: PRINT I
30  FOR T=1 TO 500: NEXT T
40  NEXT I
```

Nach Löschen des Bildschirms (Zeile 10) drucken wir zum Ausschmücken die Zahlen 1 bis 5 untereinander (Zeile 20 und 40) und damit es nicht zu schnell geht, bremsen wir mit der Zeile 30.

```
50  PRINT "LIST" (Das ist natürlich sehr einfach, aber jetzt kommt's!)
60  POKE 198,5
70  POKE 631,145: POKE 632, 145: POKE 633,145
80  POKE 634,13
90  END
```

Nach RUN erscheinen erst die fünf Zahlen, dann wird in einer neuen Zeile das Wort LIST geschrieben. Nach END wird erst (wie immer) eine Zeile ausgelassen, dann READY gedruckt und schließlich springt der Cursor an den Anfang der Zeile darunter. Während der Cursor anfangen will, drei Zeilen unter dem Wort LIST zu blinken, findet der Computer im Tastaturpuffer 3 den ASCII-Code für »Cursor up«. Also geht dieser auch drei Zeilen hoch und will jetzt auf dem Wort LIST blinken.

Damit wird es aber wieder nichts, denn im Puffer steht ja noch der Code für RETURN (13). Das wird ausgeführt und zwar für das LIST. Es hat dieselbe Wirkung, als ob Sie direkt LIST tippen und danach auf die RETURN-Taste drücken, nämlich das Programm wird ausgeLISTet.

Alle sinnvollen Basic-Befehle, die Sie in der Zeile 50 PRINTen, werden durch diese dynamische Manipulation des Cursors ausgeführt.

Versuchen Sie es mit

```
50  PRINT "PRINT 16 * 35"
50  PRINT "LOAD"
50  PRINT "GOTO 10"
50  PRINT "RUN"
50  PRINT "RUN 50"
```

und falls Sie dieses kleine Programm geSAVEt haben

```
50  PRINT "SYS 64824"
50  PRINT "SYS 64763"
```

Die Kunst ist also, mit entsprechenden Codezahlen den Cursor an diejenige Stelle des Bildschirms zu bringen, wo innerhalb eines Programms eine geeignete Anweisung gedruckt worden ist. Man kann damit getrennte Programmteile nachladen, mit SYS-Befehlen Maschinenprogramme aufrufen, oder gar Programme durch sich selbst ändern lassen.

```
5 POKE 198,10
10 POKE 631,65:POKE 63
2,1 41
6 POKE 633,156
30 POKE 634,66:POKE 63
5,17:POKE 636,67
99 END
READY.
RUN
READY.
A
?SYNTAX
ERROR
READY.
  C
```

Bild 4. Nach »A« wird CR eingefügt

```
LIST
5 POKE 198,10
10 POKE 631,65:POKE 63
2,141
6 POKE 633,156
30 POKE 634,66:POKE 63
5,17:POKE 636,67
99 END
READY.

RUN
READY.
A
B
  C
```

Bild 5. SHIFT-CR statt CR