Tips & Tricks für Einsteiger

Wir bringen Ihnen diesmal einen kurzen Beitrag über eins der wichtigsten Themen, die Computer-Neulinge beschäftigen: Wie man fehlerfrei Programme abtippt! Daneben viele kurze Programmiertips und nützliche Hinweise im Umgang mit dem C 64.

Eigentlich sollten wir an dieser Stelle keine Eigenwerbung betreiben, aber wir tun's trotzdem. Für Einsteiger auf dem C 64 haben wir nämlich einen ganz heißen Tip: Unser Sonderheft 5/86. Titel: »Grundlagen«. Von der Grafik- bis zur Musik-Programmierung, vom gekonnten Umgang mit der Datasette bis hin zum sinnvollen Einsatz des Diskettenlaufwerks bietet Ihnen dieses Sonderheft Antwort auf alle grundlegende Programmierfragen. Viele ausführliche und leicht verständliche Artikel helfen Ihnen, ein C 64-Profi zu werden. (tr)

Programme fehlerfrei abtippen

Jeder C 64-Fan, egal ob blutiger Anfänger oder ausgefuchster Profi, macht beim Abtippen von Programmen Fehler. Der Profi ist jedoch meist in der Lage, den Fehler schnell zu erkennen und zu beheben. Wir geben Ihnen ein paar Tips, wie auch Sie zum Tipp-Profi werden.

Der wichtigste Punkt, den man sich unbedingt merken sollte, ist folgender: Der C 64 »versteht« nur Eingaben, die nicht länger sind als 80 Zeichen (zwei Bildschirmzeilen)! Egal, ob man sich im Direktmodus befindet (also ohne Zeilennummer) oder gerade eine Basic-Zeile eingibt. Alles, was über die zweite Bildschirmzeile hinausgeht, wird vom Computer als neues Kommando verstanden, und der Rest »vergessen«.

Was tut man aber, wenn eine Zeile, die in der 64'er abgedruckt ist, doch über zwei Bildschirmzeilen hinausreicht? Zuerst einmal muß man sagen, daß bis auf drei Ausnahmen alle Leerzeichen vor und nach Basic-Befehlen ersatzlos weggelassen werden können. Für den C 64 ist es also völlig egal, ob Sie

```
100 FOR I=1 TO 100 : NEXT I
```

oder

```
100FORI=1TO100:NEXTI
```

eintippen! Unsere Listings enthalten diese Leerzeichen nur, um die Lesbarkeit des Programms zu erhöhen. Außerdem können Sie sich eine Menge Zeit sparen, wenn Sie beim Abtippen die Leerzeichen weglassen. Lediglich vor den Befehlen »AND«, »OR« und »NOT« sollte ein Zeichen Zwischenraum gelassen werden. Der C 64 könnte sonst einen »syntax error« melden, wo keiner ist. Ein weiterer Punkt, um beim Abtippen Platz und Zeit zu sparen, ist, alle Basic-Befehle abgekürzt einzugeben. Generell kann man sagen, daß die Abkürzung eines Befehls sein erster Buchstabe plus der zweite Buchstabe zusammen mit der Shift-Taste ist. Es gibt aber einige Ausnahmen. Dies alles können Sie im Anhang D der Bedienungsanleitung zu Ihrem C 64 entnehmen. Dieser Anhang ist überhaupt einer der nützlichsten des gesamten Handbuchs. Am besten, man kopiert sich diese Seite und hängt sie über seinen C 64.

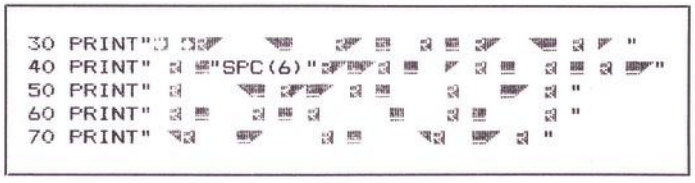
In unseren Listings sind des öfteren Zeichen enthalten, die Sie auf der Tastatur vergeblich suchen werden. Da wären zum Beispiel die geschweiften Klammern. Alles, was innerhalb der geschweiften Klammern steht, dürfen Sie nicht eintippen, sondern müssen die darin angegebene Taste drücken. Auf dem Bildschirm erscheint dann ein reverses Zeichen (also dunkles Zeichen auf hellem Grund). Dies hat folgenden Grund: Geben Sie auf Ihrem C 64 einmal ein Anführungszeichen ein und drücken dann zum Beispiel die »Bildschirm löschen«-Taste (also <SHIFT> und <CLR>). Was passiert? Der Bildschirm wird nicht, wie man es vermutet hätte, gelöscht, sondern es erscheint ein reverses Herzchen. Dies ist der Steuer-Code für »Bildschirm löschen«. Immer, wenn der C 64 auf ein solches Zeichen innerhalb von PRINT-Anweisungen im Programmablauf stößt, wird das Zeichen nicht auf dem Bildschirm ausgegeben, sondern die entsprechende Funktion (hier: Bildschirm löschen) ausgeführt. Dies gilt für alle Farbtasten, sämtliche Cursor-Steuer-Funktionen und die Funktions-Tasten.

Weiterhin werden Sie in unseren Listings unter- beziehungsweise überstrichene Buchstaben und Zeichen finden. Die Erklärung ist ganz einfach: Unterstrichen bedeutet: Buchstabe/Zeichen zusammen mit der <Shift>-Taste drücken. Überstrichen bedeutet: Buchstabe/Zeichen zusammen mit der »Commodore-Taste« (ganz links unten auf der Tastatur) drücken.

Warum haben wir unsere Listings mit diesen Zeichen versehen? Die Grafik-, beziehungsweise (Cursor-) Steuerzeichen sind auf dem Drucker, also auch in der 64'er, nur sehr schwer zu unterscheiden. Wenn Sie schon einmal ein Basic-Programm aus einer anderen Computerzeitschrift abgetippt haben, werden Sie die Vorteile unserer »übersetzten« Zeichen erkennen. Um Ihnen das Gesagte noch einmal zu verdeutlichen, zeigen wir Ihnen ein kleines Beispiel: Wenn Sie dies in der 64'er lesen,

```
30 PRINT"{CLR,SPACE,LIG.BLUE,RVSON}£{3SPAC
   E}₹{RVOFF,3SPACE,RVSON}£{SPACE,RVOFF,2S
   PACE,RVSON,SPACE,RVOFF,SPACE,RVSON}£{2S
   PACE}₹{RVOFF,SPACE,RVSON,SPACE}£ "          <074>
40 PRINT"{SPACE,RVSON,SPACE,RVOFF}"SPC(6)"
   {RVSON}£{RVOFF}£{RVSON,SPACE,RVOFF,2SPA
   CE}£{SPACE,RVSON,SPACE,RVOFF,2SPACE,RVS
   ON,SPACE,RVOFF,SPACE,RVSON,SPACE,RVOFF}
   £"                                          <218>
50 PRINT"{SPACE,RVSON,4SPACE}₹{RVOFF,SPACE
   ,RVSON}£{RVOFF}£{SPACE,RVSON,SPACE,RVOF
   F,4SPACE,RVSON,3SPACE,RVOFF}£{SPACE,RVS
   ON,SPACE}"                                  <175>
60 PRINT"{SPACE,RVSON,SPACE,RVOFF,3SPACE,R
   VSON,SPACE,RVOFF,SPACE,RVSON,5SPACE,RVO
   FF,3SPACE,RVSON,SPACE,RVOFF,4SPACE,RVSO
   N,SPACE}"                                   <107>
70 PRINT" ₹{RVSON,3SPACE,RVOFF}£{4SPACE,RV
   SON,SPACE,RVOFF,4SPACE}₹{RVSON,2SPACE,R
   VOFF}£{SPACE,RVSON,SPACE}"                  <173>
```

und Sie es korrekt eintippen, sollte auf Ihrem Bildschirm dies zu sehen sein:



Zum Schluß soll noch ein seltener Spezialfall erklärt werden. Wenn Sie in einem unserer Listings ein {DEL} sehen, so müssen Sie folgendermaßen vorgehen (gilt nur innerhalb von Anführungszeichen):

SHIFT-2, DEL, SHIFT-DEL, DEL, SHIFT-2, DEL

Auf Ihrem Bildschirm sollte nun ein reverses T erscheinen.

Noch ein Wort zu den Zahlen innerhalb der eckigen Klammern, die in allen unseren Basic-Listings auftauchen: Um unseren Lesern eine maximale Sicherheit beim Abtippen zu gewähren, haben wir ein Programm namens »Checksummer« entwickelt. Der Checksummer ist auf jeder Programmser-

vice-Diskette enthalten. Die Anleitung wurde zuletzt in der 64'er, Ausgabe 3/85 auf Seite 55 veröffentlicht. Beim Eingeben von Listings bringt der Checksummer eine sogenannte Prüfsumme für jede Basic-Zeile auf den Bildschirm. Diese Prüfsumme muß mit der Zahl, die im Heft abgedruckt ist, übereinstimmen. Das bedeutet für Sie, daß Sie diese Zahlen nicht mit abtippen dürfen! Sie dienen lediglich der Kontrolle. Übrigens: Auch dem Checksummer ist es »egal«, ob Sie die Leerzeichen vor und nach den einzelnen Basic-Befehlen mit abtippen oder nicht. Die Prüfsumme bleibt in jedem Fall gleich. (tr)

»Pseudo-Scroll« unsichtbar?

Einige Leser werden wahrscheinlich das Programm Pseudo-Scroll von N. Bergerhoff aus der 64'er, Ausgabe 3/86, abgetippt haben und auf ein Problem gestoßen sein: Denn wenn Sie einen älteren C 64 besitzen, funktioniert das Programm einwandfrei, bis auf den Aufruf, der den ganzen Bildschirm mit Shift-X vollschreibt (zum Beispiel SYS 49232,1,0,0). Dies geschieht zwar auch, aber die Zeichen sind nicht zu sehen, da sie die Hintergrundfarbe annehmen. Bei neueren C 64 nehmen die Zeichen die aktuelle Zeichenfarbe an. Dies liegt an der Routine, die die Hintergrundfarbe setzt, die es in zwei Versionen gibt. Erläutert wurde dies schon in der 64'er, Ausgabe 1/86, Seite 76.

Für alle diejenigen, die nun ein älteres Modell des C 64 besitzen und die in den Genuß von »Pseudo-Scroll« kommen wollen, zeigt Listing 1 ein Programm, das diesen Fehler aufhebt.

Dabei lädt man »PSEUDO-SCROLL« gefolgt von dem Befehl »NEW«. Daraufhin wird Listing 1 wieder geladen und gestartet. Das Programm nimmt dann die notwendigen Änderungen vor. Nun gibt man den Filenamen an, unter dem das geänderte Programm gespeichert werden soll, und fertig ist die Version von Pseudo-Scroll für alle älteren C 64.

(A. Lazarevic/tr)

```
11 POKE 49315,76:POKE 49316,96:POKE 49317,
   193:POKE 49318,96                          <224>
12 FOR T=49504 TO 49534:READ DA:POKE T,DA:
   NEXT T                                     <098>
13 INPUT"{CLR}FILENAMEN ";N$                  <213>
14 OPEN 1,8,1,N$                              <217>
15 PRINT#1,CHR$(0);CHR$(192);                 <106>
16 FOR I=49152 TO 49534                       <226>
17 PRINT#1,CHR$(PEEK(I));                     <123>
18 NEXT I                                     <102>
19 CLOSE 1                                    <030>
20 DATA 157,232,6,232,208,250,160,0,162,21
   6,132,248,134,249,173,134,2,145            <124>
21 DATA 248,200,208,251,230,249,166,249,22
   4,220,208,243,96                           <173>
```

Listing 1. Die Erweiterung zu »Pseudo-Scroll« aus der 64'er, 3/86

Pfeifender C 64

Warum denn immer vor dem Bildschirm warten, bis der C 64 ein Programm von Datasette geladen hat? Mit einem kleinen Trick gibt der C 64 einen lauten Pfeifton von sich, wenn das Programm im Speicher steht.

1. Bildschirm löschen
2. Folgende POKEs eingeben:
   S=54272:POKES+24,15:POKES+6,240:POKES+1,50:POKES+4,33
   dann aber nicht <RETURN>, sondern
3. <SHIFT-RETURN> drücken!
4. LOAD "gewünschtes Programm" eingeben und <RETURN> drücken.
5. Dann zuerst die <HOME>- und dann die <RETURN>-Taste drücken. Auf dem Bildschirm hat das zunächst keine Wirkung.
6. Erst jetzt die <PLAY>-Taste an der Datasette drücken.

Sobald der Ladevorgang beendet ist, werden die unter 5. genannten Tasten ausgeführt und dadurch der Pfeifton (Punkt 2.) gestartet. Abschalten mit <RUN/STOP-RESTORE> oder POKES+4,32. (Markus Beinlich/tr)

Zahlen eingeben mit dem Joystick

Mit diesem Programm (Listing 2) ist es möglich, über den Joystick (Port 1) eine beliebige 6stellige Zahl einzugeben und auszuwerten. Auf dem Bildschirm erscheinen sechs Nullen, nachdem der Joystick nach oben bewegt wurde. Unter einer Ziffer erscheint ein Pfeil. Durch die Bewegung des Joysticks nach oben, wird die durch den Pfeil angezeigte Ziffer um eins erhöht. Bei Bewegung des Joysticks nach rechts oder links erscheint der Pfeil unter der nächsten Ziffer. Es ist immer die Ziffer zu verstellen, auf die der Pfeil zeigt! Auf Knopfdruck wird der Bildschirm gelöscht und die Zahl wird dargestellt. An diese Stelle (Zeile 60100) können Sie nun in Ihrem eigenen Programm die eingegebene Zahl weiter verarbeiten. Sie steht in der Variablen AZ. (Kai Huebers/tr)

```
60000 PRINT"{CLR,BLACK}":Z1=48:Z2=48:Z3=48
      :Z4=48:Z5=48:Z6=48:AZ=0                 <002>
60005 IF PEEK(56321)<>254 THEN 60005          <102>
60010 BA=1520:ZC=48:POKE BA,ZC:POKE BA+1,Z
      C:POKE BA+2,ZC:POKE BA+3,ZC:POKE BA+
      4,ZC                                    <143>
60011 POKE BA+5,ZC:FOR S=1 TO 100:NEXT        <195>
60020 IF PEEK(56321)<>255 THEN 60025:         <163>
60021 GOTO 60020                              <206>
60025 POKE BA+40,32:IF PEEK(56321)=254 THE
      N POKE BA,Z1:Z1=Z1+1:IF Z1>57 THEN Z
      1=48                                    <130>
60026 GOTO 60032                              <020>
60032 IF PEEK(56321)=247 THEN 60037           <095>
60033 GOTO 60039                              <158>
60037 IF PEEK(56321)=254 THEN POKE BA+1,Z2
      :Z2=Z2+1:IF Z2>57 THEN Z2=48            <238>
60038 BA=BA+1:IF BA>1525 THEN BA=1520         <201>
60039 IF PEEK(56321)=251 THEN 60041           <173>
60040 GOTO 60045                              <228>
60041 IF PEEK(56321)=254 THEN POKE BA-1,Z2
      :Z2=Z2+1:IF Z2>57 THEN Z2=48:           <065>
60042 BA=BA-1:IF BA<1520 THEN BA=1525         <141>
60045 IF PEEK(56321)=247 THEN 60050           <012>
60047 GOTO 60055                              <043>
60050 IF PEEK(56321)=254 THEN POKE BA+2,Z3
      :Z3=Z3+1:IF Z3>57 THEN Z3=48            <185>
60051 BA=BA+1:IF BA>1525 THEN BA=1520         <214>
60055 IF PEEK(56321)=247 THEN 60060           <030>
60057 GOTO 60065                              <117>
60060 IF PEEK(56321)=254 THEN POKE BA+3,Z4
      :Z4=Z4+1:IF Z4>57 THEN Z4=48            <130>
60061 BA=BA+3:IF BA>1525 THEN BA=1520         <226>
60065 IF PEEK(56321)=247 THEN 60070           <048>
60067 GOTO 60075                              <191>
60070 IF PEEK(56321)=254 THEN POKE BA+4,Z5
      :Z5=Z5+1:IF Z5>57 THEN Z4=48            <072>
60071 BA=BA+4:IF BA>1525 THEN BA=1520         <237>
60075 IF PEEK(56321)=247 THEN 60080           <066>
60077 GOTO 60085                              <010>
60080 IF PEEK(56321)=254 THEN POKE BA+5,Z6
      :Z6=Z6+1                                <147>
60081 BA=BA+5:IF BA>1525 THEN BA=1520         <248>
60085 POKE BA+40,30                           <218>
60089 IF PEEK(56321)=239 THEN 60091           <201>
60090 GOTO 60020                              <019>
60091 IF PEEK(56321)<>239 THEN GOTO 60020     <040>
60092 FOR S=1 TO 45:NEXT:IF PEEK(56321)<>2
      39 THEN 60020                           <024>
60097 Z1=PEEK(1520)-48:Z2=PEEK(1521)-48:Z3
      =PEEK(1522)-48:Z4=PEEK(1523)-48         <201>
60098 Z5=PEEK(1524)-48:Z6=PEEK(1525)-48       <248>
60099 AZ=(Z1*100000)+(Z2*10000)+(Z3*1000)+
      (Z4*100)+(Z5*10)+Z6                     <033>
60100 PRINT"{CLR,LIG.BLUE,2DOWN,4RIGHT}DIE
       ZAHL HEISST: ";AZ                      <204>
60110 GET A$:IF A$=""THEN 60110               <213>
60120 IF PEEK(56321)<>255 THEN 60110          <209>
```

Listing 2. Zahlen eingeben mit dem Joystick!

Zahlen rechtsbündig

```
10 f$="{6 space}":rem fuellstring
20 inputx:gosub1000:printx$:end
1000 x=int(x*100+.5)/100
```

```
1010 x$=f$+str$(x)
1020 ifx=int(x)thenx$=x$+".00"
1030 ifmid$(x$,len(x$)-1,1)="."thenx$=x$+"0"
1040 x$=right$(x$,9)
1050 return
```

Zu Beginn des Programms wird ein Füllstring definiert. Zur Aufbereitung der Variablen X wird ins Unterprogramm verzweigt. Zeile 1000 rundet X auf die Anzahl der Nachkommastellen (hier 2). Zeile 1010 wandelt X in eine Stringvariable und stellt ihr den Füllstring voran. In den Zeilen 1020 und 1030 werden bei Bedarf die Nachkommastellen auf zwei aufgefüllt. Zeile 1040 schneidet den String der passenden Länge heraus, hier sechs Vor- und zwei Nachkommastellen plus Dezimalpunkt.

(H. G. Sander/tr)

GET-Befehl sinnvoll angewendet

Hier ist ein sehr praktisches Unterprogramm, wenn in einem Programm eine Taste an verschiedenen Stellen gedrückt werden soll (in einem Menü wählen — eine Frage mit 'J/N' beantworten — und so weiter).

Bevor man die Routine durch ein »GOSUB« aufruft, werden die zugelassenen Tasten in der Variable ZT$ definiert (zum Beispiel: ZT$="1234", oder ZT$="JN" oder ZT$=CHR$(...)).

In der Zeile 10000 wird zuerst der Zähler »OK« auf Null gestellt; in Zeile 10010 folgt der übliche »GET«-Befehl mit der Stringvariablen OK$.

In der Zeile 10020 wird überprüft, ob die gedrückte Taste einer der in ZT$ definierten entspricht, bis eventuell der ganze String überprüft worden ist (Zeile 10030). Ist nichts gefunden worden, wird auf einen neuen Tastendruck gewartet (Zeile 10040).

Ist der Test in Zeile 10020 positiv, wird ins Hauptprogramm zurückgesprungen — mit folgenden Informationen:
— OK$ enthält die gedrückte Taste
— OK enthält den Rang von OK$ im String ZT$ (für ein »ON OK GOTO/GOSUB«)
— AW enthält den ASCII-Wert von OK$ (kann natürlich weggelassen werden).

```
10000 OK=0
10010 GET OK$:IF OK$="" "THEN 10010
10020 OK=OK+1:IF MID$(ZT$,OK,1)=OK$ THEN AW=ASC(OK$):RETURN
10030 IF OK<LEN(ZT$) THEN 10020
10040 GOTO 10000
```

(G. Gartner/tr)

Tips & Tricks für Profis

Wie nützt man am besten den $C000-Bereich für Basic-Programme? Wir zeigen Ihnen eine wirklich geniale Lösung und viele weitere interessante und trickreiche C 64-Leckerbissen.

Übrigens: Vor kurzem wollten wir in der Redaktion einen »too many files«-Error hervorrufen. Wir verwendeten dazu folgenden Einzeiler:

```
10 FOR I=1 TO 20 : OPEN I,2 : NEXT
```

Völlig ungläubig starrten wir auf die nach dem »RUN« ausgegebene Fehlermeldung unseres C 64. Sie lautete nämlich... Halt! Probieren Sie es selbst einmal aus. Kennen Sie die Erklärung? Auflösung folgt in der nächsten Ausgabe.

Auffrisierter SYS-Befehl

Wer selbst Basic-Erweiterungen programmiert, wird bei zeitkritischen Befehlen (zum Beispiel einer Plot-Routine) schnell an die Grenzen des SYS-Befehls mit Parameterübergabe stoßen. Er ist aufgrund der Adreßumrechnung (Sprungadresse aus dem Basic-Text holen, umrechnen, auf Fehler testen und einen JMP ausführen) manchmal zu langsam. Eine besonders geniale Methode bietet sich als Alternative an: die USR-Funktion. Sie ist um einen wesentlichen Faktor schneller als der SYS-Befehl und eine Parameterübergabe ist ebenso möglich (USR(x),y,z). Ein weiterer Vorteil: Manche Compiler verweigern den SYS-Befehl mit nachfolgenden Parametern. Ein Compiler, der die USR-Funktion nicht verarbeitet, ist uns hingegen nicht bekannt. Bei Befehlserweiterungen mit mehreren Befehlen kann dem Maschinenprogramm über den Übergabeparameter x mitgeteilt werden, welcher Befehl gewünscht wird. Übrigens: Der USR-Vektor steht in 785/786 dezimal. Das Maschinenprogramm müßte dann so aussehen:

```
JSR $B7F7   USR-Argument nach $0014/$0015 holen...
LDA $14     und schon steht der Parameter x zur
            Weiterverarbeitung bereit!
```

Das High-Byte ($0015) der Übergabevariablen x wird bei dieser Methode natürlich nicht berücksichtigt, aber wer hat schon eine Befehlserweiterung mit mehr als 256 Befehlen...

(Hartmut Kroos/tr)

LOAD-Schutz einmal anders

Kannten Sie den schon? Verwenden Sie zur Speicherung eines Programms auf Floppy folgenden Dateinamen:

```
SAVE CHR$(34),8
```

Im Directory erscheint dieses File dann mit dem Dateinamen "". Versuchen Sie nun einmal, dieses Programm wieder zu laden. Richtig! Dies geht nur mehr mit

```
LOAD CHR$(34),8
```

(Stefan Bültena/tr)

Hilfe beim Programme-Abtippen

Wenn man ein Programm abtippen möchte, sieht man sich öfters einer DATA-Wüste gegenüberstehen, beziehungsweise -sitzen. Das dabei am meisten vorkommende Zeichen ist zweifellos das Komma. Dieses liegt aber nun auf einer kleinen Taste auf der Tastatur, die, wenn man nicht dauernd auf die Tastatur schauen will, recht schwer zu erreichen ist. Die Folge ist: Man vertippt sich häufiger. Um das zu ändern, tue man folgendes:

1) ROM ins RAM kopieren

Zum Beispiel mit dem Einzeiler:

```
FOR I = 0 TO 8191 : POKE 40960 + I, PEEK (40960 + I) :
POKE 57344 + I,PEEK (57344 + I) : NEXT
```

Genialer geht es natürlich mit einem Maschinensprache-Monitor und einem Transfer-Befehl. Es müssen die Bereiche $A000 bis $BFFF nach $A000 und $E000 bis $FFFF nach $E000 verschoben werden.

2) Umschalten mit `POKE 1,53`

3) `POKE 60349,44`

Wenn man nun auf die SPACE-Taste drückt, entdeckt man, daß diese Kommanta produziert. Wer Lust hat, kann die Komma-Taste mit POKE 60336,32 mit der SPACE-Funktion belegen. Diese Umbelegung erleichtert das Abtippen um einiges.

(Peter Gorgs/tr)