Tips & Tricks für Einsteiger

Hier geben wir Ihnen eine Reihe von kurzen Listings und Hinweisen, die interessante Effekte hervorrufen und leicht in eigene Programme einzubauen sind.

Diese Rubrik hat in erster Linie die Zielsetzung, den Neulingen unter Ihnen den Einsteig in die Computerei zu erleichtern. In jeder Ausgabe bringen wir deshalb einen kurzen, leicht verständlichen Artikel über ein (scheinbar) schwieriges Thema. Bitte schreiben Sie mir, welche Frage Sie am meisten beschäftigt. Das Thema, das am meisten gewünscht wird, werde ich dann erklären. ABER: Bitte schicken Sie keinen Fragenkatalog, sondern nur Ihre wichtigste Frage! Stichwort: »Profis helfen Einsteigern«. (Thomas Röder)

»Compiler« und »Packer« — Was ist denn das?

Oft werden Sie in der 64'er in der Anleitung zu einem Basic-Programm folgendes gelesen haben: »Aus Geschwindigkeitsgründen empfiehlt es sich, das Programm zu compilieren«. Punkt. Nach weiteren Erklärungen muß der unerfahrene Leser vergeblich suchen. Was ist denn ein Compiler, mit dem sich Basic-Programme compilieren lassen? Nun, wie Sie vielleicht schon einmal gehört haben, »versteht« Ihr C 64 in Wirklichkeit kein Wort Basic, sondern nur reine Maschinensprache (stutzen Sie jetzt nicht, sondern lesen Sie bitte weiter).

Da Maschinensprache aber für Computer-Neulinge relativ kompliziert und schwer zu erlernen ist, wurde in den C 64 ein »Basic-Interpreter« eingebaut. Dies ist ein Maschinenprogramm, das, sobald Sie ein Basic-Programm eingeben und mit »RUN« starten, automatisch jedem Basic-Befehl entsprechend eine Maschinenroutine ausführt. Da aber jeder Befehl bei jedem Zeilendurchlauf neu übersetzt werden muß, sind Basic-Programme auch viel langsamer als reine Maschinenprogramme (ein schnelles Spiel in Basic zu schreiben wäre undenkbar).

Ein (Basic-)Compiler ist ein (in Maschinensprache geschriebenes) Programm, das Ihr Basic-Programm in Maschinensprache übersetzt (= compiliert). Das Ergebnis der Compilierung (das »Compilat«) ist zwar noch kein reines Maschinenprogramm, sondern lediglich eine Art »Zwischen-Code« (soll hier nicht näher erläutert werden). Trotzdem ergibt sich für das bearbeitete Programm meist ein enormer Geschwindigkeitszuwachs. Es steht hier »meist«, weil die Geschwindigkeit des Compilats stark von der Programmierweise des Basic-Programms und vom verwendeten Compiler abhängt.

Auf der Diskette (Compiler benötigen in den allermeisten Fällen eine Diskettenstation) erkennen Sie compilierte Programme meistens durch ein vorangestelltes »C/« im Programmnamen. Wenn Sie so ein Compilat nun laden und listen, erscheint nur noch eine einzige Basic-Zeile, die einen »SYS«-Befehl zum Starten des übersetzten Basic-Programms und oft eine Mitteilung über den verwendeten Compiler enthält. Gestartet wird das Ganze einfach mit »RUN«. Es sei an dieser Stelle erwähnt, daß sich compilierte Programme nicht mehr verändern lassen! Damit sollte der Begriff »Compiler« genügend erklärt sein.

Nun zum »Packer« (oft auch als »Compressor« bezeichnet). Ein Beispiel: Auf der Leserservice-Diskette zur 64'er, Ausgabe 11/85, befinden sich aus Platzgründen drei »gepackte« (beziehungsweise »komprimierte«) Basic-Programme (»Lyric 3.0«, »Betriebs.« und »Sequenzer«). Wenn diese geladen werden, so erscheint nach dem Listen ebenfalls nur eine einzige Basic-Zeile mit einem »SYS«-Befehl. Dies hat sehr viele Leser zu der Annahme verleitet, dies seien compilierte Programme, die sich nicht mehr verändern lassen. Dem ist nicht so! Werden diese gepackten Programme gestartet, so »stellt« sich der Computer für ein paar Sekunden »tot«. Während dieser Zeit bringt eine Maschinenroutine das gepackte Programm wieder in seine ursprüngliche Form. Danach startet das entpackte Basic-Programm.

Wie kommt man nun an ein solches Basic-Programm heran? Ganz einfach: »RUN« eintippen (RETURN-Taste nicht vergessen) — kurz warten — RUN/STOP-Taste drücken und gedrückt lassen — warten bis sich der C 64 mit »BREAK IN...« meldet. Danach steht das Programm wieder als »echtes« Basic-Programm im Speicher Ihres Computers. Es kann jetzt verändert und wieder auf Diskette oder Kassette gespeichert werden.

Was macht nun so ein »Packer«? Wie der Name schon sagt, »packt« er ein Programm (egal, ob Basic oder Maschinensprache) auf ein Minimum seiner ursprünglichen Länge. Allen Packern ist gemeinsam, daß sie sich wiederholende Zeichenfolgen zusammenfassen. Es gibt verschiedene, mehr oder weniger effektive Methoden, dies zu erreichen. Einen der besten Packer, den wir kennen, nämlich den »flexiblen Code-Compactor«, finden Sie im 64'er Sonderheft 5/85 (Thema: Floppy und Datasette) auf Seite 80. (tr)

Die Schüttel-Schrift

Dieses kleine Programm (Listing 1) läßt den Bildschirm in einem vorgewählten Bereich hin- und herschütteln. Es eignet sich auch zum Einbau in eigene Programme. Als erstes müssen Sie Listing 1 abtippen. Danach wird über zwei POKEs festgelegt, welcher Bildschirmbereich »geschüttelt« werden soll:

```
POKE 49317, Wert 1
POKE 49318, Wert 2
```

Die Werte 1 und 2 errechnen sich wie folgt:

Wert 1 = Startzeile (1 bis 25) x 8 + 41

Wert 2 = Endzeile x 8 + 50

Wenn Sie Listing 1 mit »RUN 60000« gestartet und die beiden POKEs errechnet und eingegeben haben, können Sie die eigentliche »Schüttel-Routine« starten: SYS 49152. Wenn es Sie genug geschüttelt hat, beenden Sie den Spuk mit SYS 49299. (Erik Becker/tr)

```
59900 REM                                              <013>
59910 REM DATAS                                        <210>
59920 REM                                              <035>
60000 DATA 120,169,40,141,20,3,169,192,141,21          <099>
60001 DATA 3,173,165,192,141,18,208,173,17,208         <020>
60002 DATA 41,127,141,17,208,169,129,141,26,208        <036>
60003 DATA 169,1,133,251,169,8,133,252,88,96           <054>
60004 DATA 173,25,208,141,25,208,48,7,173,13           <215>
60005 DATA 220,88,76,49,234,173,18,208,205,166         <162>
60006 DATA 192,176,70,165,251,201,1,240,33,165         <215>
60007 DATA 252,141,22,208,201,7,240,11,230,252         <019>
60008 DATA 173,166,192,141,18,208,76,129,234,169       <093>
60009 DATA 1,133,251,173,166,192,141,18,208,76         <194>
60010 DATA 129,234,165,252,141,22,208,240,11,198       <076>
60011 DATA 252,173,166,192,141,18,208,76,129,234       <117>
60012 DATA 169,0,133,251,173,166,192,141,18,208        <142>
60013 DATA 76,129,234,169,8,141,22,208,173,165         <167>
60014 DATA 192,141,18,208,76,129,234,120,169,0         <126>
60015 DATA 141,26,208,169,49,141,20,3,169,234          <106>
60016 DATA 141,21,3,88,96,40,82                        <132>
60100 REM                                              <215>
60110 REM EINLESEN DER DATEN                           <187>
60120 REM                                              <235>
61000 FOR I=49152 TO 49318                             <057>
61100 READ A                                           <177>
61200 POKE I,A                                         <129>
61300 S=S+A                                            <015>
61400 NEXT                                             <193>
61500 IF S<>22085 THEN PRINT"FEHLER IN DATAS !":
      END                                              <006>
61600 CLR                                              <243>
```

Listing 1. Die »Schüttelschrift«

Tips zum Directory

Dieser Tip zeigt, wie man das Inhaltsverzeichnis einer Diskette teilweise laden kann.

Mit LOAD''$:NAME'',8 wird nur der Eintrag »NAME« geladen. Es werden Diskettenname, ID, Programmtyp, -länge und die freien Blocks ausgegeben.

Mit LOAD''$:A*'',8 werden alle Einträge, die mit »A« anfangen, geladen.

Mit LOAD''$:???'',8 werden alle Einträge, die drei Zeichen lang sind, geladen. (Sascha Sadewasser/tr)

Hilfe bei DATA-Wüsten

Kennen Sie das nicht auch? Man tippt eine meterlange DATA-Wüste in seinen Computer ein, startet sie und erfreut sich Meldungen wie zum Beispiel: out of data error, illegal quantity error oder andere. Folge: heftiges Vergleichen der DATA-Ziffern.

Um dies zu verhindern, habe ich mir folgenden Trick einfallen lassen: Man nehme das zu bearbeitende Programm und gebe folgenden Einzeiler ein.

```
FOR I=1 TO 20000 : READA: PRINT A : PRINT : POKE 198,0 : WAIT
198,1 : NEXT I
```

Nach jedem Tastendruck steht jeweils ein DATA-Wert auf dem Bildschirm, den Sie überprüfen sollten. Sollten Sie einen Fehler entdecken, sofort im Heft markieren und weiter überprüfen. Erst nachdem Sie einen OUT OF DATA ERROR erhalten, können Sie mit der Korrektur beginnen. Normalerweise sind die DATAs jetzt in Ordnung. Wenn nicht, haben Sie einen Fehler gemacht und müssen die Überprüfung wiederholen...
(Jens Ingo Jakubczyk/tr)

Der Piepser

```
10 g=40:H=20
20 INPUT"{ CLEAR} EINGABE";EG$
30 INPUT"{ DOWN} LÄNGE DES TONS";G
40 INPUT"{ DOWN} PAUSE ZWISCHEN TÖNEN";H
50 GOSUB 10000
9997 :
9998 END
9999 REM ***** --- TONERZEUGUNG ---
10000 A=54272:B=54273:C=54276:D=54277:
      E=54278:F=54296
10005 POKE F,15:POKE D,16+9:POKE E,4*16+4
10010 POKE A,29:POKE B,69
10015 FOR I=1TOLEN(EG$)
10020 PRINTMID$(EG$,I,1);
10025 POKEC,33:FORT=1TOG:NEXTT
10030 POKEC,32:FORT=1TOH:NEXTT:NEXTI
10035 PRINT:RETURN
```

Dieses Listing schreibt jede eingegebene Buchstabenfolge langsam auf den Bildschirm, das heißt, so wird Buchstabe für Buchstabe auf den Bildschirm gedruckt. Man kann sowohl die Länge eines Tons (auf jeden Buchstaben folgt ein Ton) als auch den Abstand (Pause) zwischen Buchstabe und Ton verändern. (Richard Eisenmenger/tr)

Sprites suchen

```
1  V=53248:POKEV+1,128:POKEV,128:POKEV+23,255:
   POKEV+29,255:POKEV+21,255
2  PRINT"{ CLR} ";A
3  GETA$:IFA$=""THEN3
4  IFA$="{ CRSR-Rechts} " THENA=A+1:IFA=256THENA=255
5  IFA$="{ CRSR-nach unten} "THENA=A-1:IFA=-1 THENA=0
6  IFA$="M"THENPOKEV+28,255
7  IFA$="N"THENPOKEV+28,0
8  POKE2040,A:GOTO2
```

Das obige Programm »Sprite-View« muß abgetippt und gespeichert werden. Nach dem Start erscheint ein weißes Sprite auf dem Bildschirm und in der linken oberen Ecke steht eine Zahl. Dies ist die Blockzahl des Sprites. Multipliziert man die Blockzahl mit 64, so erhält man die Startadresse des Sprites. Mit der Taste <Cursor-rechts> erhöht man die Blockzahl um 1, mit <Cursor-hinunter> erniedrigt man sie um 1. Mit <M> schaltet man den Multicolormodus des Sprites ein, mit <N> wieder aus. So kann man zwischen 256 Sprites blättern.

Programm-Erläuterung:

Zeile 1:	Initialisierung Sprite 0
Zeile 2:	druckt Blockzahl auf Bildschirm
Zeile 3:	warte auf Taste
Zeile 4:	wenn Taste = Cursor rechts, dann Blockzahl +1, wenn Blockzahl = 256, dann Blockzahl = 255
Zeile 5:	wenn Taste = Cursor runter, dann Blockzahl —1, wenn Blockzahl = —1, dann Blockzahl 0
Zeile 6:	wenn Taste = M, dann Multicolor ein
Zeile 7:	Wenn Taste = N, dann Multicolor aus
Zeile 8:	Blockzahl poken, Sprung nach 2

(Christoph Brochhaus/tr)

Reset-Schutz

Auf die Frage von Christoph von Treek im 64'er 2/86, wie ein Reset-Schutz von Basic aus realisierbar ist, habe ich mir folgendes überlegt:

Wie schon des öfteren im 64'er erwähnt, muß zu diesem Zweck ein Modul simuliert werden. Das geschieht durch POKEn der Zeichenfolge CBM80 ab Speicherzelle $8004(=32772):

```
A=32772:POKEA,195:POKEA+1,194;POKEA+2,205:
POKEA+3,56:POKEA+4,48
```

Das Betriebssystem »denkt« nun, ein Programm-Modul wäre in den Erweiterungsport des C 64 eingesteckt und versucht dieses an der Adresse, die sich aus dem Zeiger $8000/8001 ergibt, zu starten. Der Restore-Vektor steht in den Speicherstellen $8002/8003 im Lo-Hi-Byte-Format.

Das Problem ist jetzt die Auswahl der Startadresse, auf die die Vektoren zeigen sollen.

Man kann den Computer in einer eigenen Reset-Routine aufhalten, so daß man gezwungen wird, einen Master-Reset (Aus- und Einschalten des C 64) durchzuführen. Für Maschinenprogrammierer dürfte es kein Problem sein, wirksame Routinen zu schreiben. Als einfachste und aus Platzgründen kürzeste Lösung empfehle ich, zu den oben genannten POKEs noch folgende hinzuzufügen:

```
POKE32768,9:POKE32769,128:POKE32770,9:POKE32771,128:
POKE32777,238:POKE32778,32:POKE32779,208:
POKE32780,76:POKE32781,9:POKE32782,128
```

(Alexander Hilsbos/tr)

Explodierender Bildschirm

Dieser Basic-Einzeiler bewirkt, daß der Bildschirm wie bei professionellen Spielen vibriert (zum Beispiel bei Explosionen):

```
0 FOR A=0 TO 15:POKE 53270,A:NEXT:GOTO 0
```

(Jan Melichar/tr)

Zufallsgrafiken

Ich habe einen Einzeiler geschrieben, der einen ganz interessanten Effekt bewirkt. Das Listing lautet:

```
10 B=INT(RND(1)*127):POKE53272,B:GOTO 10
```

Das Programm basiert auf den Bildschirm-Codes, mit deren Hilfe sich immer wieder neue Zufallsgrafiken aufbauen.
(Michael Biehl/tr)