Programmierter Direktmodus

Programmierter Direktmodus hört sich wie ein Widerspruch in sich an. Entweder man befindet sich im Direktmodus oder es läuft ein Programm, beides gleichzeitig scheint kaum vereinbar. Dennoch gibt es eine Verbindung, die bisher ungeahnte Möglichkeiten eröffnet.

Das Geheimnis dieser Verbindung liegt im Tastaturpuffer (Tabelle 1) und dem Umstand, daß der Computer, nachdem er im Programm auf eine END-Anweisung trifft, so viele Zeichen aus dem Tastaturpuffer holt, wie der Anzahlspeicher angibt und sofort ausführt. Dies kann man sich zunutze machen, indem man mittels PRINT eine Anweisung auf den Bildschirm bringt, den Cursor veranlaßt, in diese Zeile zu springen und den Computer durch ein im Puffer abgelegtes RETURN mit der Abarbeitung der eingegebenen Bildschirmzeile fortfahren läßt. Dabei werden alle Zeichen über ihren ASCII-Code in den Puffer gebracht. Ein kleines Beispiel soll dieses Vorgehen verdeutlichen:

Geben Sie Beispiel 1 ein.
Was passiert? — Der Computer geht nach der END-Anweisung in den Direktmodus über und führt die beiden Steuerzeichen CURSOR/HOME und RETURN (CHR$(19 beziehungsweise 13)) aus. Der Cursor springt also in die erste Bildschirmzeile, wo zu lesen ist:

I=I+1 : GOTO 20
Diese Zeile führt der Computer nun aus und springt, nachdem er die Variable I um 1 hochgezählt hat, zurück in Programmzeile 20. Jetzt hilft nur noch RUN/STOP.

Beispiel 2
Dieses Programm bewirkt folgendes:

— Zeile 100 wird in das Programm eingefügt
— Zeile 350 wird aus dem Programm gelöscht
— der in Zeile 70 stehende Spruch wird durch einen — meiner Meinung nach treffenderen — ersetzt
— das geänderte Programm wird gelistet.

Anstelle des LIST-Befehles könnte auch ein GOTO beziehungsweise GOSUB-Befehl wieder in das Programm zurückspringen. Allerdings ist auch hier zu beachten, daß, sobald man neue Basic-Zeilen einfügt (beziehungsweise löscht), die Variablenwerte verloren gehen.

Mittels dieser Methode kann man zum Beispiel ein Programm entwickeln, das die DATA-Zeilen eines Sprites berechnet, ins laufende Programm übernimmt und dann die restlichen Programmzeilen herauslöscht, so daß nur ein Sprite-Ladeprogramm übrigbleibt, welches sofort abgespeichert werden kann. Eine weitere sinnvolle Anwendungsmöglichkeit des »Programmierten Direktmodus« können Sie den folgenden Ausführungen entnehmen.

System Lademenü

Das System soll die folgenden Aufgaben erfüllen:
(1) Mit verschiedenen Programmen (eventuell auf verschiedenen Disketten) arbeiten, ohne daß ständig LOAD und RUN gegeben werden muß.

(2) Den ärgerlichen »file not found error« verhindern, der bereits auftritt, wenn man ein Leerzeichen zuviel oder zuwenig eingibt.

(3) Programmiertes Aufrufen von Programmen (eventuell mit Parameterübergabe). Arbeiten mit dem Programmsystem ohne genaue Kenntnis der verwendeten Filenamen.

Die Idee

Mit Hilfe des Commodore-Programmes DOS 5.1 und der Methode des Programmierten Direktmodus kann man das oben genannte Ziel erreichen. Man geht dabei folgendermaßen vor:

Auf jede Diskette, die nach diesem System arbeiten soll, bringt man das Programm DOS 5.1 sowie dessen Lader unter einem möglichst kurzen einprägsamen Filenamen (hier »£«-Listing 1). Weiterhin kopiert man das Programm »Lademenü« auf jede der Disketten und trägt in dieses die Filenamen ein (Listing 2). Dabei ist darauf zu achten, daß die Filenamen »£« und »Lademenü« überall exakt gleich sind.

Der Arbeitsablauf gestaltet sich dann in folgender Weise: Nach dem Einschalten von Computer und Floppy legt man die gewünschte Diskette ein und lädt »£«. Dieses Programm initialisiert nach dem Starten die bekannten DOS 5.1-Befehle. Das

Beispiel 1

```
10 I=1
20 PRINT"⟨shift-clr/home⟩⟨ 6xCursor down⟩"I". Lauf"
30 FOR J=0 TO 1500 : NEXT
40 PRINT"⟨shift-clr/home⟩ I=I+1 : GOTO 20"
50 FOR J=0 TO 1500 : NEXT
60 POKE 631,19 : POKE 632,13 : POKE 198,2 : END
```

Beispiel 2

```
10 PRINT"⟨shift-clr/home⟩";
20 PRINT 100 "REM == Zeile 100 wird eingefügt =="
30 PRINT 350
40 PRINT"70 REM Morgenstund hat Blei im Hintern"
50 PRINT"LIST"
60 POKE 631,19 : FOR I=1 TO 5 : POKE 632 + I,13 :
NEXT
65 POKE 198,6 : END
70 REM Morgenstund hat Gold im Mund
350 REM == Diese Zeile wird gelöscht!==
```

Tabelle 1

	Speicherstelle (n) dezimal	hex
Tastaturpuffer	631 — 640	0277 — 0280
erweitert*	631 — 645	0277 — 0285
Anzahlspeicher	198	00C6
max. Größe des Tastaturpuffers	649	0289

* Anm.: Obwohl der Tastaturpuffer normalerweise nur 10 Zeichen faßt, können insgesamt 15 Zeichen dort abgelegt werden.

```
1 REM ^ = PFEIL NACH OBEN
2 REM _ = PFEIL NACH LINKS
10 POKE53281, 0:POKE53280, 0:PRINTCHR$(1
58):PRINTCHR$(142)
20 A=0:PRINT"■":PRINT"
"
22 PRINT"                  ■LADEMENUE"
25 PRINT"                  ":PRIN
T:PRINT
30 PRINT"            1 = LISTE DER DOS BE
FEHLE
31 PRINT"            2 =BEISPIELPROGRAMM"
32 PRINT"            3 =*HIER MUESSEN
33 PRINT"            4 =*SIE IHRE
34 PRINT"            5 =*PROGRAMME EIN-
36 PRINT"            6 =*TRAGEN
37 PRINT"            7 =*
38 PRINT"            8 =*
39 PRINT"            9 =*
40 PRINT"           10 = ENDE":PRINT
100 INPUT"IHRE WAHL";A:A=INT(A):IFA<10RA
>10THENPRINT"■":GOTO20
105 PRINT"■";
110 ONAGOTO300,111,112,113,114,115,116,1
17,118,119,
111 PRINT"^BEISPIELPRG"          :GOTO200
112 PRINT"^PRG2"                 :GOTO200
113 PRINT"^PRG3"                 :GOTO200
114 PRINT"^PRG4"                 :GOTO200
115 PRINT"^PRG4"                 :GOTO200
116 PRINT"^PRG5"                 :GOTO200
117 PRINT"^PRG6"                 :GOTO200
118 PRINT"^ UND SO WEITER      ":GOTO200
119 POKE198,0:PRINT"CIAO":END
120 POKE198,0:END
200 POKE631,19:POKE632,13:POKE198,2:END
300 PRINT:PRINT"DER DOS MANAGER BIETET F
OLGENDE BEFEHLE:":PRINT
310 PRINT"=============================
=========":PRINT
320 PRINT"     _ = SAVE
330 PRINT"     / = LOAD
340 PRINT"     ^ = LOAD MIT AUTOSTART
350 PRINT"     @ = ANZEIGEN DISKSTATUS
360 PRINT"    @$ = ANZEIGEN DIREKTORY
370 PRINT"=============================
=========":PRINT
380 PRINT"DIESE BEFEHLE KOENNEN SIE JETZ
T NUTZEN!"
390 PRINT"=============================
=========":POKE198,0
395 PRINT:PRINT:PRINT"                ■*T
ASTE*":WAIT198,1:GETA$:GOTO20
READY.
```

Listing 2. »Lademenü«

```
10 REM ^ = PFEIL NACH OBEN
20 POKE53280,0:POKE53281,0:PRINTCHR$(158
)
30 IFA=0THENA=1:LOAD"DOS 5.1",8,1
40 IFA=1THENSYS12*4096+12*256
50 PRINT"■";"^LADEMENUE"
60 POKE631,19:POKE632,13:POKE198,2:END
READY.
```

Listing 1. »£«

```
1 REM ^ = PFEIL NACH OBEN
100 REM BEISPIELPROGRAMM
110 :
120 BERNHARD LAUER
130 :
140 PRINT"■"
150 PRINT"AN JEDES LISTING MUESSEN SIE"
160 PRINT"DIE ZEILE":PRINT
170 PRINT"PRINT CHR$(147);"CHR$(34)"^LAD
EMENUE"CHR$(34)":POKE 631,19:";
180 PRINT"POKE 632,13:POKE198,2:END":PRI
NT
190 PRINT"ANSTELLE DES END ANFUEGEN!"
200 PRINT:PRINT:PRINT:PRINT
210 PRINT"■ ** TASTE **"
220 POKE 198,0:WAIT 198,1
230 PRINT"■";"^LADEMENUE":POKE631,19:POK
E632,13:POKE198,2:END
READY.
```

Listing 3. »Beispielprogramm«

RUN für »£« soll auch gleichzeitig das letzte sein, da nun der DOS-Befehl »↑« zur Verfügung steht. Mit dessen Hilfe wird das Lademenü dieser Disk geladen und gestartet, welches die Programme anbietet und automatisch richtig lädt und startet. Zusätzlich bietet das Lademenü auch einen Überblick über die wichtigsten DOS-Befehle, die ja nun zur Verfügung stehen.

Um den Kreis zu schließen wird an jedes Programm auf der Diskette anstelle des üblichen END die folgende Zeile eingefügt:

PRINT CHR$(147); »Lademenü« :POKE 631,19:POKE 632,13:POKE 198,2:END

Dadurch wird nach jedem regulären Programmabbruch wieder das Lademenü dieser, oder wenn die Diskette vorher gewechselt wurde, einer beliebigen anderen Disk geladen. Nun wird durch das Lademenü das nächste ausgewählte Programm geladen und gestartet. Will man Parameter an andere Programme übergeben, so erreicht man dies über sequentielle Dateien.

Die Lösung

- ☐ Listing »£«
- ☐ Listing »Lademenü«
- ☐ Listing »Beispielprogramm«

Ausblick

Schreibt man noch einen Autostart für »£« (eventuell mittels des Beispiels in Ausgabe 6/84) so kann man das »System Lademenü« zu einem System »Nie wieder RUN« ausbauen.

(Bernhard Lauer/rg)