ON ERROR GOTO

Routinen, die bei einem auftretenden Programmfehler zu einer bestimmten Zeile verzweigen, sind inzwischen Standardausrüstung jeder guten Basic-Erweiterung. Das vorliegende Maschinenprogramm ist nur 88 Byte lang.

Um einen Fehler während des Programmlaufs abzufangen, kennen viele Computer den Befehl ON ERROR GOTO. Mit diesem kurzen Maschinenprogramm kann auch Ihr C 64 auf Fehler im Programm reagieren.

Nachdem das Programm (Listing 1) im Speicher steht, kann es mit SYS 50000, ZL aktiviert werden. Dabei bedeutet ZL die Zeilennummer, zu der bei einem Fehler verzweigt werden soll. Die Fehlernummer wird dann in Speicherstelle 2 gespeichert und kann mit PEEK(2) abgefragt werden. Tabelle 1 zeigt, welche Nummer welcher Fehlermeldung entspricht. Die Zeile, in der der Fehler auftrat, wird in Speicherstelle 249 und 250 gespeichert. Damit Sie zu der fehlerhaften Zeile zurückspringen können, wurde noch die Routine »GOTO X« eingebaut. Damit läßt sich zu einer berechneten Zeile springen. Aufgerufen wird diese Routine mit SYS 50076, ZL. Schauen wir uns einmal das Beispiel ab Zeile 200 im Basic-Lader an. In Zeile 210 wird mit SYS 50000, 1000 die Fehlerbehandlungsroutine auf Zeile 1000 gelegt. In Zeile 220 wird der Benutzer aufgefordert, eine Zahl einzugeben. Solange diese Zahl positiv ist, wird in Zeile 230 die Wurzel aus dieser Zahl errechnet und ausgegeben. Bei einer negativen Zahl wird zur Zeile 1000 verzweigt. Hier wird die Fehlernummer ausgegeben. In diesem Falle 14. In Zeile 1010 wird nun die Zeile errechnet, in der der Fehler auftrat und ausgegeben. Zeile 1020 schließlich ruft die »GOTO X«-Routine auf und springt in die Zeile ZL —10.

Listing 2 zeigt den dokumentierten Assemblerteil der Routine. (Kunz/tr)

```
100 REM *** ON ERROR UND GOTO X ***          <029>
101 DATA 32,253,174,32,107,169,165,20        <202>
102 DATA 164,21,133,247,132,248,169,116      <191>
103 DATA 160,195,141,0,3,140,1,3,96,169      <188>
104 DATA 139,160,227,141,0,3,140,1,3         <006>
105 DATA 96,138,48,30,165,58,201,255         <148>
106 DATA 240,27,134,2,165,57,164,58,133      <145>
107 DATA 249,132,250,165,247,164,248         <193>
108 DATA 133,20,132,21,32,163,168,76         <254>
109 DATA 174,167,76,116,164,138,76,58        <188>
110 DATA 164,32,253,174,32,138,173,32        <156>
111 DATA 247,183,76,143,195                  <154>
112 S=0:FOR I= 50000 TO 50087 :READ D        <150>
113 POKE I,D:S=S+D:NEXT                      <177>
114 PRINT"DIE DATAZEILEN SIND"               <206>
115 IF S<>10886 THEN PRINT"FEHLERHAFT":STO
    P                                        <163>
116 PRINT"IN ORDNUNG":END                    <113>
120 :                                        <096>
130 :                                        <106>
200 REM BEISPIEL FUER ON ERROR UND GOTO X    <028>
210 SYS 50000,1000                           <244>
220 INPUT"EINE ZAHL ";A                      <237>
230 WU=SQR(A):PRINT"DIE WURZEL AUS"A"="WU    <130>
240 END                                      <242>
1000 PRINT"FEHLER NUMMER"PEEK(2)             <214>
1010 ZL=PEEK(249)+PEEK(250)*256-10           <239>
1020 SYS 50076,ZL                            <039>
```

Listing 1. Der Basic-Lader zu »On-Error«. Bitte verwenden Sie zur Eingabe den neuen Checksummer (Seite 6.)

Fehlernummer	Fehlermeldung
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 FILENAME
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
22	TYPE MISMATCH
23	STRING TOO LONG
24	FILE DATA
25	FORMULA TOO COMPLEX
27	UNDEF'D FUNCTION
28	VERIFY
29	LOAD

Tabelle 1. Die Fehlermeldungen des Betriebssystems, jeweils mit dem Zusatz »ERROR«.

```
      .ba=50000
jsr   $aefd       ; prüft auf Komma
jsr   $a96b       ; holt Zeilennummer nach $14/$15
lda   $14         ;
ldy   $15         ; Zeilennummer in
sta   $f7         ;
sty   $f8         ; $f7/$f8 speichern
lda   #$74
ldy   #$c3        ; Vektor für Fehlermeldung
sta   $0300       ;
sty   $0301       ; auf neue Routine
rts
-----------------------------------------------
lda   #$8b        ;
ldy   #$e3        ; Vektor für Fehlermeldung
sta   $0300       ; zurückstellen
sty   $0301       ;
rts
-----------------------------------------------
txa
bmi   noerr       ; größer 127 dann kein Fehler
lda   $3a         ;
cmp   #$ff        ; Direktmodus?
beq   dirmod      ; ja
stx   $02         ; Fehlernummer speichern
lda   $39         ;
ldy   $3a         ; Momentane Zeilennummer
sta   $f9         ; nach $f9/$fa
sty   $fa         ;
lda   $f7         ;
ldy   $f8         ; Zeilennummer für on error
sta   $14         ; nach $14/$15
sty   $15         ;
go jsr $a8a3      ; goto-Befehl
jmp   $a7ae       ; Zur Interpreterschleife
noerr jmp $a474   ; ready ausgeben
dirmod txa
jmp   $a43a       ; Fehlermeldung ausgeben
routine goto x
jsr   $aefd       ; prüft auf Komma
jsr   $ad8a       ; numerischen Wert holen
jsr   $b7f7       ; nach $15/$14 wandeln
jmp   go
```

Listing 2. Der dokumentierte Assemblerteil.