abgespeichert werden soll (Zeile 60120).
Zeile 60130 — siehe oben.

So dann wird der Floppykanal geöffnet (Zeile 60140), die dezimalen Eingaben auf »Diskettenformat« gebracht (Zeile 60150), damit die ersten beiden Bytes auf der Diskette als Ladeadresse geschrieben werden können (Zeile 60160). In den Zeilen 60170 bis 60190 werden schließlich die Daten ausgelesen und direkt auf die Diskette geschrieben. Die übrigen Zeilen dienen der Abfrage des Fehlerkanals der Floppy beziehungsweise schließlich der »Fertig«-Meldung.

Weitere Hinweise:
1) Das Programm ist in dieser Form sowohl für den VC 20 als auch für den C 64 verwendbar.
2) Bei entsprechender Abänderung des OPEN-Befehls sollte auch die Abspeicherung auf Kassette möglich sein (die Zeilen 60200 bis 60240 entfallen dann).
3) Vor dem Abspeichern sollten die DATAs natürlich korrekt sein, da nach dem Abspeichern eine Überprüfung noch schwieriger ist.
Bei Basicprogrammen sollten deshalb der Basicteil und die DATAs zunächst unabhängig voneinander eingegeben, zum Probelauf mit »MERGE« zusammengefügt und bei Fehlerlosigkeit der DATA-Teil dann entsprechend abgespeichert werden.
4) Die »eigenständigen« Maschinenprogamme werden dann mit LOAD'xy',8,1 geladen und mit dem SYS-Befehl gestartet.
Bei Basicprogrammen sollte dann (sofern nicht ein Autostartprogramm zum Laden aller Teile verwendet wird), die erste Programmzeile lauten:
IFA=0THENA=1: LOAD »Name des Maschinenspracheteils«,8,1
— das klingt zwar paradox, aber es funktioniert: nach dem RUN wird dann geladen und gestartet.
Daß in einem solchen Basicprogramm alle READ-Befehle etc. ausgebaut werden müssen, versteht sich wohl von selbst.

(Uwe Christian Parpart/gk)

Listing »Datawandler«

```
60000 REM *** DATAWANDLER ***
60010 REM *** (C) UWE CHR. PARPART ***
60020 PRINT"■■    *** D A T A W A N D L
E R ***"
60030 INPUT"■ JETZIGE ANFANGSADRESSE";AA
60040 INPUT"■     JETZIGE ENDADRESSE";EA
60050 PRINT"■ IST DIE ANFANGSADRESSE IDE
NTISCH"
60060 PRINT"■ MIT SPAETERER LADEADRESSE
(J/N)?"
60070 GETA$:IFA$=""THEN60070
60080 IFA$="J"THENLA=AA:GOTO60120
60090 IFA$="N"THEN60110
60100 GOTO 60070
60110 INPUT"■   SPAETERE LADEADRESSE";LA
60120 INPUT"■ NAME DES PROGRAMMS";L$
60130 OPEN1,8,1,L$+",P,W"
60140 HB=INT(LA/256):LB=LA-HB*256
60150 PRINT#1,CHR$(LB);CHR$(HB);
60160 FORI=AATOEA
60170 PRINT#1,CHR$(PEEK(I));
60180 NEXTI:CLOSE1
60190 REM *** ABFRAGE FEHLERKANAL ***
60200 OPEN1,8,15
60210 INPUT#1,A,B$,C,D
60220 PRINTA;B$;C;D
60230 CLOSE1
60240 PRINT"■■■■F E R T I G !■"
60250 END
READY.
```

Simons Basic: Befehle, die nicht im Handbuch stehen

Als Ergänzung zu den Artikeln über Simons Basic in den Ausgaben 4/84 und 5/84 wollen wir in dieser Ausgabe noch einige Befehle und Besonderheiten aufführen, die nicht in jedem Handbuch stehen.

Zunächst die zusätzlichen Befehle in alphabetischer Reihenfolge:

BCKGNDS
Syntax: BCKGNDS f1,f2,f3,f4
— f1: normale Hintergrundfarbe
— f2: Hintergrundfarbe der Zeichen mit SHIFT-Taste
— f3: Hintergrundfarbe der REVERS-Zeichen und des Cursors (nicht der Schriftfarbe)
— f4: Hintergrundfarbe für Zeichen mit SHIFT-Taste im REVERS-Mode
Semantik: BCKGNDS legt die Hintergrundfarben fest und schaltet auf ECM (Extended-Color-Mode), dabei werden von jedem Zeichen zwei Bit vom ASCII-Code abgezweigt: es steht somit nicht mehr der gesamte Zeichensatz zur Verfügung.
NRM macht BCKGNDS rückgängig

COLOUR
Syntax: COLOUR, rf, hf
— rf: Rahmenfarbe
— hf: Hintergrundfarbe
Semantik: COLOUR setzt Rahmen- und Hintergrundfarbe und erspart somit das lästige POKE 53280,rf : POKE 53281,hf.

DISABLE
Syntax: DISABLE
Semantik: Setzt ON KEY-Anweisung außer Kraft

GRAPHICS:
Syntax: GRAPHICS
Semantik: Liefert Konstante $D000 = 53248; Adresse VIC

NRM
Syntax: NRM
Semantik: NRM macht MEM und BCKGNDS rückgängig.

ON KEY
Syntax: ON KEY Stringausdruck, diverse Anweisungen
Semantik: Wird eine Taste gedrückt, die im Stringausdruck des ON KEY-Befehls enthalten ist, so wird in den Anweisungsteil verzweigt. Die Tastatur wird dabei vor jedem Befehl abgefragt. Ein unbedingter Sprung erfolgt, wenn im Stringausdruck eine »eckige Klammer zu« ($5D) enthalten ist.

RESUME
Syntax: RESUME
Semantik: RESUME funktioniert nur nach ON KEY. Bei RESUME wird das Programm beim ursprünglichen Befehl fortgesetzt. RESUME entspricht somit dem RETURN bei GOSUB.

SOUND
Syntax: SOUND
Semantik: Liefert Konstante $D400 = 53972; Adresse SID

Punkte, die besonders zu beachten sind

AT
ist auch als Zuweisung möglich. Beispiel A$ = AT (Spalte, Zeile) B$. Die Cursorpositionierung erfolgt schon während der Zuweisung.
DUMP
Matritzen werden nicht angezeigt.
NO ERROR, OUT
NO ERROR schaltet nur ON ERROR ab, OUT gibt die Standardfehlermeldung aus.
OLD
Die Variablenwerte gehen verloren.
REPEAT, LOOP, EXEC
Für jede dieser Anweisungen existiert ein eigener Stack, der bis zu fünf Werte aufnehmen kann.
SCRSV, SCRLD, COPY, HRDCPY,
schließen Datei 1.
TRACE
Der TRACE-Befehl funktioniert nicht nach MEM.
Mehr über Simons Basic in: Das Commodore 64-Buch, Band 5.

(Hans Lorenz Schneider/aa)

Die Suche nach den Synthtischen

Das Programm ermöglicht die systematische Suche nach allen vom Betriebssystem unterstützten Steuerzeichen. Man ist nun nicht mehr angewiesen auf zum Teil lückenhafte Tabellen synthetischer Steuerzeichen, sondern kann sich stattdessen selbst auf die Suche begeben.

Von den so sagenumwobenen »synthetischen Steuerzeichen« war in früheren Ausgaben dieser Zeitschrift schon die Rede. Jedoch erhielt der Leser bislang noch kein einigermaßen handfestes »Kochrezept« zur erfolgreichen Suche nach ihnen. Mit dem folgenden Programm soll diese Lücke geschlossen werden.

Es gibt nach Eingabe des gewünschten ASCII-Wertes alle Tastenkombinationen aus, deren Betätigung die Tastaturdecodierroutine dazu veranlaßt, den entsprechenden ASCII-Wert in den Tastaturpuffer zu schreiben.

Es werden allerdings auch solche Kombinationen aufgeführt, deren »ASCII-Werte« zwar in den Decodiertabellen (die ab $EB81 im Betriebssystem beginnen) an entsprechender Stelle aufgeführt sind, jedoch NICHT im Tastaturpuffer erscheinen (zum Beispiel die »2« bei Betätigung der SHIFT-Taste). Der Leser wird diese jedoch schnell von der ersten Gruppe unterscheiden können.

Zum Schluß noch zwei Bemerkungen:

1) Bei gleichzeitigem Erscheinen von zwei oder mehreren Tastenkombinationen für einen bestimmten ASCII-Code kann oft — jedoch nicht immer — eine Kombination gegen eine andere ausgetauscht werden, ohne das Endresultat zu verändern (Beispiel für eine Ausnahme: Sowohl die RUN/STOP-Taste wie auch die CTRL C-Kombination belegen gemäß der Decodiertabellen den ASCII-Wert »3«, jedoch kann mit »CTRL C« kein Programm abgestoppt werden, da für die Abfrage der RUN/STOP-Taste eine gesonderte Routine zuständig ist, die nur diese Taste erkennt).

2) Alle »synthetischen Steuerzeichen«, für deren Erzeugung der Basic-Interpreter zuständig ist, kann dieses Programm nicht erkennen, da es lediglich auf die Tastaturdecodiertabellen im Kernal Bezug nimmt.

Da das Programm lediglich die Tastaturdecodiertabellen des Betriebssystems benötigt (und die dort aufgeführten ASCII-Werte in ein ARRAY einliest) kann es leicht durch entsprechende Abänderung der in Zeile 20 enthaltenen Anfangsadressen in eine auch auf dem VC 20 laufende Version umgeschrieben werden.

(Engin Gülen/gk)

Listing zur Steuerzeichensuche

```
1 REM          *  STEUERZEICHEN   *
2 REM           *  FUER DEN CBM-64 *
3 REM            *  ENGIN  GUELEN   *
4 REM             *  POSTWEG     2   *
5 REM              *  4192 KALKAR 1   *
9 POKE53280,0:POKE53281,0
10 DIMAS(3,63)
18 REM ANFANGSADRESSEEN DER
19 REM TASTATURDEKODIERTABELLEN
20 DATA60289,60354,60419,60536
30 FORI=0TO3:READA(I):NEXT
40 FORI=0TO3:FORJ=0TO63
50 AS(I,J)=PEEK(A(I)+J)
60 NEXTJ,I
159 REM TASTATURMATRIX
160 DATADEL,RETURN,CRSR(RIGHT),F7,F1,F3,F5,CRSR(DOWN)
161 DATA3,W,A,4,Z,S,E,SHIFT(L)
162 DATA5,R,D,6,C,F,T,X
163 DATA7,Y,G,8,B,H,U,V
164 DATA9,I,J,0,M,K,O,N
165 DATA+,P,L,-,.,DOPPELPUNKT,@,KOMMA
166 DATA£,*,;,HOME,SHIFT(R),=,↑,/
167 DATA1,←,CTRL,2,SPACE,C=,Q,RUN/STOP
170 DIMTA$(64):FORI=0TO63:READTA$(I):NEXTI
180 DATA-----,SHIFT,C=    ,CTRL  ,
190 FORI=0TO3:READA$(I):NEXT
200 INPUT"[■■■■■■■■■■■] ASCII-KODE ";AS%
210 IFAS%<00RAS%>255THEN200
220 PRINT:PRINT:FORI=0TO3:FORJ=0TO63
230 IFAS%=AS(I,J)THENPRINT"■"A$(I),:GOTO250
240 GOTO270
250 IFLEFT$(TA$(J),2)=LEFT$(A$(I),2)THENPRINT:GOTO270
260 PRINT"■"TA$(J)
270 NEXTJ,I
280 PRINTSPC(193)"■ TASTE "
290 GETA$:IFA$=""THEN290
300 GOTO200

READY.
```