Der DATA BECKER Führer

Schneider CPC

Alles auf einen Blick

Bernd Grohmann

Der DATA BECKER Führer

Schneider CPC

ISBN 3-89011-406-7

Wichtiger Hinweis:
Die in diesem Buch wiedergegebenen Schaltungen, Verfahren und Programme werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden.

Alle Schaltungen, technischen Angaben und Programme in diesem Buch wurden von dem Autoren mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. DATA BECKER sieht sich deshalb gezwungen, darauf hinzuweisen, daß weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernommen werden kann. Für die Mitteilung eventueller Fehler ist der Autor jederzeit dankbar.

Inhaltsverzeichnis

1. Der CPC-Führer

Dieses Buch soll Ihnen die praktische Arbeit mit Ihrem Schneider-Computer erleichtern. Sicher haben Sie schon festgestellt, daß die Informationen, die Sie immer wieder benötigen, auf zahlreiche Bücher verteilt sind. Es hat sich leider gezeigt, daß das Hauptproblem im Umgang mit dem Computer nicht das Verständnis "der Sache" an sich ist, sondern vielmehr darin liegt, sich die zum Arbeiten notwendigen Details zu merken. Nachdem man "die Sache" einmal gelernt hat, wird man in den meisten Fällen nur noch Probleme mit der Auswahl eines Befehls oder seiner Schreibweise haben. Dieses Beispiel läßt sich auch auf andere Gebiete (z.B. die Hardware) übertragen.

Zu diesem Führer

Wir versuchen auch, Mängel der Originalliteratur zu beheben. Ein Beispiel: Mit dem Schneider-Handbuch ist es leider schwer, einen BASIC-Befehl für eine bestimmte Aufgabe zu finden. Das Handbuch ordnet streng alphabetisch, während Sie hier eine Systematik finden.

Außer dem Inhaltsverzeichnis gibt es zum zum schnellen Suchen und Finden am Ende des Buchs alphabetische Register. Zunächst ein Themenverzeichnis, das auf spezielle Problemstellungen eingeht. Danach folgt ein umfangreiches Stichwortregister, mit dem Sie gezielt Begriffe und Befehle nachschlagen können.

2. ROM-Software
2.1 Locomotive-BASIC und Amstrad-DOS

Liste der verwandten Abkürzungen

<...>	"..." kann, muß aber nicht angegeben werden
adr	Adresse
array	Indizierte Variable, allgemein
ausdr	Ausdruck, allgemein
ausdr$	Stringausdruck
BinDatPar	Binär-Datei-Parameter-Liste = adr(Datei-Anfang), Datei-Länge, <adr(Programm-Start)>
Buchst	Buchstabe
BuchstBer	Buchstabenbereich =Buchst<-Buchst>
col	Farbe
const	Konstante, allgemein
dx	x-Versatz relativ zur aktuellen Position
dy	y-Versatz relativ zur aktuellen Position
extSgn	Erweiterungszeichen
Fb	Feldbreite
file	Datei
ForPar	Formalparameter
ForSch	Formatschablone
Fs	Farbstift
HGM	Hintergrundmodus (ab CPC 664)
hka	Hüllkurvenabschnitt = step, stepW, stepZ oder TonP, stepZ bei 'ENT' bzw. regWert, HKP bei 'ENV'
HKP	Hüllkurvenperiode
int	Integerausdruck
k	Tonkanal
ks	Kanalstatus
lhk	Lautstärkehüllkurve
li	linke Grenze
line	Zeilennummer
lineBer	Zeilenbereich = line<-line>
logAusdr	logischer Ausdruck
nVar	numerische Variable
ob	obere Grenze
opt	Option

par	Parameter
pemod	Farbstiftmodus = Reaktion mit vorhandenem Bildschirmspeicherinhalt: 0 überschreiben/normal 1 XOR 2 AND 3 OR
port	Schnittstellennummer
re	rechte Grenze
real	Real-Ausdruck
regWert	Registerwert
sep	Separator (Komma oder Semikolon)
step	Schrittanzahl
stepW	Schrittweite
stepZ	Schrittzeit = Vielfache von 10 ms
strm	Stream
thk	Tonhüllkurve
time	Zeitintervall = Vielfache von 20 ms
timer	Zeitgebernummer
TonP	Tonperiode
un	untere Grenze
var	Variable allgemein
var$	Stringvariable
x	x-Koordinate
y	y-Koordinate

AUTO <line> <,stepW>

Erzeugt ab line automatisch Zeilennummern zur Programmeingabe, die sich jeweils um stepW unterscheiden.

DELETE <lineBer>

Löscht die Programmzeilen im angegebenen Bereich bzw. das ganze Programm.

EDIT line

Listet Programmzeile line und übernimmt sie in den Eingabepuffer.

FRE(ausdr)

Ergibt den noch zur Verfügung stehenden Speicherplatz.

HIMEM

Nicht beschreibbare Variable, die die höchste vom BASIC beanspruchte Speicheradresse beinhaltet.

LIST <lineBer> <,strm>

Listet den lineBer in den strm.

MEMORY adr (ab 664 repariert)

Setzt die höchste vom BASIC benutzbare Speicheradresse fest.

NEW

Löscht den Programmspeicher incl. Variablen. Alles andere (z.B. RSX-Module) bleibt erhalten.

RENUM <lineneu> <,<linealt> <,stepW>>

Numeriert die Programmzeilen neu. Ändert auch GOTO und GOSUB.

TROFF

Schaltet Tracer ab.

TRON

Schaltet Tracer ein.

AMSDOS, Disketten-Manipulation

A:

Laufwerk A wird zum Standard-Laufwerk bestimmt.

B:

Laufwerk B wird zum Standard-Laufwerk bestimmt.

CAT

Listet Inhalt, USER-Nummer und freien Platz auf der Diskette im angemeldeten (Standard-) Laufwerk.

CHAIN file <,line>

Lädt Programmfile in den Speicher und startet selbsttätig in Zeile line bzw. am Prog.-Anfang.

CHAIN MERGE file <,line> <,DELETE lineBer>

Sortiert die Zeilen des Programms file in die im Speicher vorhandenen; evtl. belegte Zeilen werden überschrieben. Mit DELETE kann vor dem Laden der lineBer gelöscht werden. Sonst wie CHAIN.

CLOSEIN

Eingabedatei schließen.

CLOSEOUT

Ausgabedatei schließen.

CPM

Lädt Betriebssystem CP/M von Laufwerk A.

DIR <,file>

Listet Disketteninhalt im CP/M-Format.

DISC

DISC.IN und DISC.OUT zusammen.

DISC.IN

Bestimmt Floppy als Dateieingabeeinheit

DISC.OUT

Bestimmt Floppy als Dateiausgabeeinheit

DRIVE,"x"

Bestimmt "x" als Standardlaufwerk.

EOF

Nicht beschreibbare Variable, die den Zustand des Eingabefiles beschreibt. EOF=0 -> File nicht leer.

ERA,file

Löscht alle nicht schreibgeschützten Dateien namens file.

LOAD file<,adr>

Lädt BASIC-Programm file. Binärdateien können in adr geladen werden. Geschützte Programme sind nicht mit LOAD ladbar.

MERGE file

Wie CHAIN MERGE, nur ohne automatischen Start.

OPENIN file

Datei zum Lesen öffnen.

OPENOUT file

Datei zum Beschreiben öffnen.

REN,fileneu,filealt

Gibt einer Datei einen neuen Namen.

SAVE file <,filetyp <,BinDatPar>>

Schreibt das Programm auf das ausgewählte Speichermedium.

```
filetyp:   ohne = Basic-Programm,
             A = ASCII-Datei,
             B = Binär-Datei (z.B. Masch.-Programm)
```

SPEED WRITE int

Aufzeichnunggeschwindigkeit auf Cassette festlegen, 0 => 1 kbd, 1 => 2 kbd

TAPE

TAPE.IN und TAPE.OUT gleichzeitig.

TAPE.IN

Bestimmt Recorder zur Eingabeeinheit.

TAPE.OUT

Bestimmt Recorder zur Ausgabeeinheit.

USER,int

Bestimmt die Benutzernummer.

Bildschirm-Manipulation

BORDER col <,col>

Bildschirmrandfarbe wählen. Bei Angabe von 2 Farben blinken diese.

CLS <strm>

Löscht Bildschirmfenster strm bzw. den ganzen Bildschirm.

CURSOR <int1> <,int2> (ab CPC 664)

Setzt (int1/2 = 1) oder löscht (int1/2 = 0) den System- bzw. Benutzerschalter. Cursor ist sichtbar, wenn beide = 1.

FRAME

Stimmt Schreiben auf dem Bildschirm mit der Bildschirm-Synchronisation ab.

INK pen,col <,col>

Dem pen wird die Farbe col bzw. 2 blinkende Farben zugewiesen.

LOCATE <strm,> x,y

Textcursor setzen.

MODE int

Bildschirmmodus auswählen

0 = 20, 1 = 40, 2 = 80 Zeichen je Zeile

PAPER <strm,> pen

Wählt Hintergrundfarbstift.

PEN <strm,> <pen> <,HGM> (HGM ab CPC 664)

Wählt Schriftfarbstift und Hintergrundmodus.

POS(strm)

Ergibt die Textcursorspalte.

SPEED INK int1,int2

Legt die Dauer der Sichtbarkeit der blinkenden Farben in Vielfachen von 20ms fest.

VPOS (strm)

Ergibt die Textcursorpositionszeile.

WINDOW <strm,> li, re, ob, un

Legt Größe und Lage eines Bildschirmfensters fest.

WINDOW SWAP strm <,strm>

Tauscht zwei Fenster gegeneinander aus. Wird ein strm einmal weggelassen (nicht gestattet bei POS und VPOS), so wird vom BASIC außer bei CLS strm = 0 angenommen.

CLG <pen>

Löscht den Grafik-Bildschirm, indem er ihn mit dem hier oder zuletzt in einem CLG angegebenen pen einfärbt.

DRAW x, y, <, <pen> <,pemod>> (pemod ab CPC 664)

Zieht eine Linie vom aktuellen Grafik-Cursorort zum Punkt (x,y).

DRAWR dx,dy <, <pen> <,pemod>> (pemod ab CPC 664)

Zieht eine Linie vom aktuellen Grafik-Cursorort (x,y) zum Punkt (x+dx,y+dy).

FILL pen (ab CPC 664)

Füllt Bildschirmteilfläche mit dem Farbstift pen aus.

GRAPHICS PAPER pen

Grafik-Hintergrundfarbe wählen.

GRAPHICS PEN <pen> <,HGM>

Grafik-Malfarbe und/oder -Hintergrundmodus wählen.

MASK int1, int2 (ab CPC 664)

Definiert die setzbaren Bildpunkte in einer Linie.

MOVE x,y <, <pen> <,pemod>> (pemod ab CPC 664)

Setzt den Grafik-Cursor auf den Punkt mit den Koordinaten x,y.

MOVER dx,dy <, <pen> <pemod>> (pemod ab CPC 664)

Versetzt den Grafik-Cursor um dx Punkte horizontal und dy Punkte vertikal.

ORIGIN x,y <, li, re, ob, un>

Legt den Koordinaten-Ursprung und evtl. die Größe der Graphik fest.

PLOT x,y <, <pen> <,pemod>> (pemod ab CPC 664)

Setzt den Punkt mit den Koordinaten x,y; Farb-Stift und -Modus wählbar.

PLOTR dx,dy <,<pen> <,pemod>> (pemod ab CPC 664)

Setzt Punkt (x+dx,y+dy).

TAG <strm>

Hiernach werden Texte für strm zum Graphik-Cursorort umgeleitet.

TAGOFF <strm>

Schaltet TAG ab.

TEST(x,y)

Ergibt die Farbstift-Nummer des adressierten Pixels und setzt den Graphik-Cursor.

TESTR(dx,dy)

Ergibt die Farbstift-Nummer des um dx,dy vom derzeitigen Graphik-Cursorort verschobenen Pixels.

XPOS

Ergibt Spalte (X-Koordinate) des Graphic-Cursors.

YPOS

Ergibt Zeile (Y-Koordinate) des Graphic-Cursors.

Definieren, Dimensionieren, Vereinbarungen

DEFFN var<(ForPar)>=ausdr

Definiert eine Benutzerfunktion.

DEFINT BuchstBer <,BuchstBer...>

Definiert alle Variablen, die mit einem Buchstaben aus dem/den genannten Bereich/en anfangen, als Integer-Zahlen.

DEFREAL BuchstBer <,BuchstBer...>

Wie DEFINT, bloß für Real-Variablen.

DEFSTR BuchstBer <,BuchstBer...>

Wie DEFINT, bloß für String-Variablen.

DEG

Trigonometrie wird im Gradmaß ausgeführt.

DIM array <,array ...>

Dimensioniert Felder

KEY int,ausdr$

Dem Erweiterungszeichen Nr. int wird der Stringausdruck zugewiesen.

KEYDEF taste, dauerfunktion <,normalzeichen <,shiftzeichen <,controlzeichen>>>

Tastenfunktionen definieren. Zeichen können ASCII oder Erweiterungszeichen sein.

PI

System-Konstante. Wert: 3.14159265

RAD

Trigonometrie wird im Bogenmaß ausgeführt.

SPEED KEY startverzögerung,wiederholrate

Der Startwert und die Zeitabstände für die Tastenwiederholfunktion werden festgelegt.

SYMBOL int,int1,int2,...,int8

ASCII-Zeichen int wird aus den bis zu 8 folgenden int zeilenweise erstellt.

SYMBOL AFTER int

Legt das 1. vom Benutzer definierbare Zeichen fest.

Ein- und Ausgabe-Befehle

INKEY (int)

Funktion, mit der getestet werden kann, ob die Taste Nr. int gedrückt ist. Ergebnis:

```
-1 = Taste int nicht gedrückt
 0 = Taste int gedrückt
32 = Taste int und Shift gedrückt
128 = Taste int und Control gedrückt
160 = Taste int, Shift und Control gedrückt.
```

INKEY$

Tastaturabfrage, welche Taste gedrückt wurde. Ergebnis: ASCII-Zeichen.

INP (port)

Ergibt das von der Schnittstelle mit der I/O-Adresse port angebotene Byte.

INPUT <strm,> <;> <<text> <sep>> var <,var...>

Eingabe von Daten vom strm (9=Tape/Disc). <;> kann Zeilenvorschub nach der Eingabe verhindern. Wenn<sep> =";", dann "?" nach dem Text, wenn =",", dann nicht.

JOY(int)

Joystick Nr. int (0 oder 1) abfragen.

LINE INPUT <strm,> <;> <<text> <sep>> var$

Eingabe eines Textes beliebigen Inhaltes. Sonst wie INPUT.

OUT port,int

Ausgabe des int an den I/O-Port Nr. port.

PRINT <strm,> <USINGforSch> ausdr<sep> <<USINGforSch> ausdr...>

Ausgabe mit USING formatierbarer Daten an den angegebenen Stream.

SPC(int)

Fügt int Leerzeichen in die Ausgabe ein.

TAB(int)

Setzt die Ausgabe in der nächsten freien Spalte Nr. int fort.

WAIT port,int

Wartet darauf, daß Port Nr. port das Byte int anbietet.

WIDTH int

Legt die Druckerzeilenbreite fest.

WRITE <strm,> ausdr<sep> <<ausdr<sep>>...>

Wie PRINT, schreibt jedoch Strings mit Anführungszeichen.

ZONE int

Legt die Tabulierung für Komma-Tabulator fest.

USING forSch

Formatiert die in PRINT/WRITE-Befehlen folgenden Ausdrücke. "1 Ziffer" bedeutet, daß das Zeichen den Platz für eine Ziffer freihält. "n Stellen" bedeutet, daß die Zeichenfolge für n Stellen des Formates steht.

Numerik-Schablone:

#	1 Ziffer
.	Dezimalpunkt
,	1 Ziffer, vor Dez.-Punkt, trennt Tausendergruppen.
££	2 Stellen, 1 Ziffer stellt Pfundzeichen zwischen Vorzeichen und Zahl.
**	2 Stellen, ersetzt führende Nullen durch "*".
**£	3 Stellen, erst "*" für führende Nullen, dann Pfundzeichen, dann die Zahl.
$$	wie **, aber mit Dollarzeichen.
**$	wie **, aber mit Dollarzeichen.
+	vor oder nach der Zahl, zeigt auch "+".
-	oder + am Ende der Zahl setzt dorthin das "Vorzeichen".

String-Schablone:

!	nur das 1. Zeichen.
\leerzeichen\	Stringbreite=Anzahl Leerzeichen + 2.
&	String gedruckt, wie geschen.

Eingriffe in den Programmablauf

CALL adr <par<,par...>>

Aufruf einer Maschinenroutine.

CONT

Programmfortsetzung nach 2x ESC.

END

Programmende.

FOR nVar=real1 TO real2 STEP real3

Schleifeneröffnung

GOSUB line

BASIC-Unterprogramm-Aufruf.

GOTO line

Programmfortsetzung bei Zeile line.

IF log Ausdr THEN opt<ELSEopt>

Bedingte Programmausführung. Bei logAusdr <>0 (wahr), wird die THEN-Option ausgeführt, bei =0 (falsch), die ELSE-Option.

NEXT <nVar <,nVar...>>

Schließt die innerste Schleife bzw. die angegebenen innersten nacheinander.

ON BREAK CONT

Verhindert Programmunterbrechungen mit ESC.

ON BREAK GOSUB line

Springt zum Unterprogramm ab Zeile line, wenn 2x ESC gedrückt wurde.

ON BREAK STOP

Hebt die beiden "ONBREAK..."-Befehle auf.

ON int GOSUB line <,line...>

Springt in das in der Liste an Position int stehende Unterprogramm.

ON int GOTO line <,line...>

Springt zur in der Liste an Position int stehenden Zeile.

ON SQ(k) GOSUB line

Springt zum Unterprogramm line, wenn in der Warteliste des Tonkanals k Platz ist.

RETURN

Rücksprung aus dem innersten Unterprogramm.

RUN ausdr$

Lädt Programm ausdr$ und startet es.

RUN <line>

Startet das vorhandene Programm in der ersten bzw. angegebenen Zeile.

STOP

Unterbricht den Programmablauf.

WEND

Schließt die innerste WHILE-Schleife.

WHILE log Ausdr

Wiederholt einen Programmteil, solange der logische Ausdruck wahr ist.

Zeit-Befehle

AFTER time <,timer> GOSUB line

Einmaliger Unterprogrammaufruf nach time*20ms, die im Zähler Nr. timer gestoppt werden.

EVERY time <,timer> GOSUB line

Regelmäßiger Unterprogrammaufruf alle time*20ms.

REMAIN(timer)

Ergibt den Zählerstand und setzt den Zähler außer Betrieb.

TIME

300stel-Sekunden zählende System-Variable.

Error-Handling

DERR (ab CPC 664)

System-Variable, die den Code des letzten Diskettenfehlers beinhaltet.

ERL

System-Variable, die die Zeilennummer des letzten Fehlers beinhaltet.

ERR

System-Variable, die den Code des letzten Fehlers beinhaltet.

ERROR int

Erzeugt den Fehler Nr. int

ON ERROR GOTO line

Nach erkanntem Fehler soll das Programm in Zeile line eine Fehlerbehandlungsroutine anspringen. line = 0, dann normale Fehlermeldung.

RESUME <line>

Rückkehr aus einer Fehlerbehandlungsroutine zur Zeile line bzw. zu der mit dem Fehler.

RESUME NEXT

Dto., aber Sprung zu der der Fehlerzeile folgenden Zeile.

String-Manipulationen

COPY CHR$(strm) (ab CPC 664)

Ergibt das Zeichen, auf dem der Textcursor im angegebenen Fenster steht.

FN var$ <(forPar)>

Benutzerdefinierte Stringfunktion.

LEFT$(ausdr$,int)

Ergibt die int linken Zeichen des Strings, falls soviele vorhanden sind, sonst ist das Ergebnis mit dem String identisch.

LOWER$(ausdr$)

Ergibt den String, wobei alle Buchstaben klein geschrieben sind.

MID$(ausdr$,int1<,int2>)

Ergibt die int2 Zeichen ab Zeichen Nr. int1 des String-Ausdruckes. Ohne int2 wird der Rest des Strings genommen.

MID$(var$,int1<,int2>)=ausdr$

Überschreibt mit dem ausdr$ die Zeichen in var$ ab Nr. int1, wobei die Länge von var$ konstant bleibt, wenn sie nicht durch int2 neu festgelegt wird.

RIGHT$(ausdr$,int)

Ergibt die int rechten Zeichen des Stringausdruckes, wenn soviele vorhanden sind, anderenfalls ist das Ergebnis mit ausdr$ identisch.

SPACE$(int)

Ergibt int Leerzeichen.

STRING$(int1,ausdr2)

Ergibt int1-mal das 1. Zeichen des ausdr2$.

UPPER$(ausdr$)

Ergibt den String, wobei alle Buchstaben groß geschrieben sind.

Zahl(String)- und String(Zahl)-Funktionen

ASC(ausdr$)

Ergibt den ASCII-Code des 1. Zeichens des Strings.

BIN$(int1<,int2>)

Ergibt int1 in binärer Form mit mindestens int2 Stellen.

CHR$(int)

Ergibt das ASCII-Zeichen Nr. int.

DEC$(real,forSch) (ab CPC 664 repariert)

Ergibt das in die forSch gezwängte Ergebnis des numer. Ausdruckes.

HEX$(int1<,int2>)

Wie BIN$, aber hexadezimal.

INSTR$(<int1,>ausdr1$,ausdr2$)

Ergibt die 1. Stelle nach dem int1. Zeichen, bzw. 1. Stelle überhaupt, an der der ausdr2$ in ausdr1$ vorkommt.

LEN(ausdr$)

Ergibt die Länge des String-Ausdruckes.

STR$(real)

Ergibt den numerischen Ausdruck als String.

STRING$(int1,int2)

Ergibt int1-mal das Zeichen mit dem ASCII-Code int2.

VAL(ausdr$)

String in Zahl umwandeln. Zahl im String wird ohne Exponent gelesen. Zahl muß am Anfang des Strings stehen (sonst ist VAL = 0). Es darf Text folgen.

Zahlentypwandlung

CINT(real)

Ergibt real in eine Integer-Zahl gerundet.

CREAL(int)

Ergibt int in eine Real-Zahl gewandelt.

FIX(real)

Ergibt den ganzzahligen Anteil von real.

INT(real)

Ergibt den auf die nächst kleinere ganze Zahl abgerundeten Wert von real.

ROUND(real<,int>)

Ergibt den auf int Nachkommastellen gerundeten Wert von real. Ist int negativ wird auf -int Stellen vor dem Komma gerundet.

Arithmetik- und Logik-Operatoren

AND

Logische Und-Verknüpfung

MOD

Divisionsrest

OR

Logische Oder-Verknüpfung

XOR

Logische Entweder-Oder-Verknüpfung (Exclusiv-Oder)

+

Addition

-

Subtraktion

*****

Multiplikation

/

Division

^

Potenz

Arithmetik- und Logik-Funktionen

ABS(real)

Betrag

ATN(real)

Arcus Tangens

COS(real)

Cosinus

EXP(real)

e-Funktion = e ^ real, e = 2.7182818284

FN nVar <(forPar)>

Vom Benutzer DEFinierbare Funktionen

LOG(real)

natürlicher Logarithmus (zur Basis e)

LOG10(real)

Dekadischer Logarithmus (zur Basis 10)

NOT logAusdr

Negation (rechnet -(logAusdr+1))

SGN(real)

Vorzeichen (entspricht real/ABS(real))

SIN(real)

Sinus

SQR(real)

Quadratwurzel (real größer oder gleich 0)

TAN(real)

Tangens

UNT(int)

Wandelt Zahl zwischen 0 und 65535 in Zahl zwischen -32768 und +32767 (2cr-Komplement).

Sound-Befehle

ENT thk<,hka1 ... ,hka5>

Definiert Tonhüllkurve thk.

ENV lhk<,hka1 ... ,hka5>

Definiert Volumenhüllkurve lhk

ON SQ(k) GOSUB

siehe Seite 23ff.

RELEASE k

Hebt den Wartezustand des Tonkanals k auf.

SOUND ks, TonP <,time <,vol0 <,lhk <,thk <,rauschperiode>>>>>

Tonerzeugung, vol0 ist Anfangslautstärke.

SQ(k)

Ergibt den Kanalstatus.

Diverse Befehle

CLEAR

Alle Variablen werden gelöscht, alle offenen Dateien und Benutzer-Funktionen gehen verloren.

CLEAR INPUT (ab CPC 664)

Löscht den Eingabepuffer.

DATA const<,const...>

Die Konstanten werden mit READ in der aufgeführten Reihenfolge gelesen.

DI

Verhindert alle Programmunterbrechungen außer mit ESC.

EI

Hebt DI auf.

ERASE var<,var...>

Löscht die angegebenen Felder (Arrays)

LET var=ausdr

Wertzuweisung. "LET" kann i.a. entfallen.

MAX(real<,real...>)

Ergibt den größten Wert der in der Liste aufgeführten numerischen Ausdrücke.

MIN(real<,real...>)

Ergibt den kleinsten Wert der in der Liste aufgeführten numerischen Ausdrücke.

PEEK(adr)

Ergibt den Inhalt der Speicheradresse.

POKE adr,int

Schreibt den Wert int in die Speicherstelle mit der Adresse adr.

RANDOMIZE real

Initialisiert den Zufallsgenerator.

READ var<,var...>)

Ordnet den Variablen var die DATA-Konstanten zu.

REM

REM oder einfaches Anführungszeichen erklären den Rest der Programmzeile zu einer Kommentarzeile.

RESTORE <line>

Setzt den DATA-Zeiger auf das DATA-Kommando in der Zeile line bzw. an den Programmanfang.

RND <(real)>

Ergibt eine neue Zufallszahl, wenn das Argument fehlt oder positiv ist. Ist es 0, dann wird die letzte Zahl wiederholt. Ist es negativ, wird eine neue Zufallszahlenfolge begonnen.

CONTROL-Steuerzeichen

Diese Steuerzeichen werden bei der Stringausgabe erkannt und ausgeführt. Der Parameter ist der ASCII-Code des nächsten Zeichens.

Nr	ASC	Parameter	Funktion
0			keine
1	A	0-255	ASCII-Zeichen Nr. Par an Text-Cursor
2	B		Text-Cursor-Benutzerschalter aus
3	C		Text-Cursor-Benutzerschalter ein
4	D	0-2	Text-Modus bestimmen
5	E	0-255	ASCII-Zeichen Nr. Par an Grafik-Cursor
6	F		Text-Bildschirm einschalten
7	G		Piep-Ton, Tonwarteschlange löschen
8	H		Text-Cursor 1 Position zurücksetzen
9	I		Text-Cursor 1 Position vorrücken
10	J		Text-Cursor 1 Zeile tiefer rücken
11	K		Text-Cursor 1 Zeile hochrücken
12	L		Text-Fenster löschen, -Cursor in Pos. 1,1
13	M		Text-Cursor an Zeilenanfang setzen
14	N	0-15	Hintergrund-Farbstift wählen
15	O	0-15	Schrift-Farbstift wählen
16	P		Zeichen unter dem Text-Cursor löschen
17	Q		Zeilenteil links vom Cursor löschen
18	R		Zeilenteil rechts vom Cursor löschen
19	S		Fenster bis zum Cursor löschen
20	T		Fenster ab Cursor löschen
21	U		Text-Bildschirm ausschalten
22	V	0-1	Transparentmodus ein bzw. aus
23	W	0-3	Grafik-Hintergrund-Modus
24	X		Farbstifte von Pen und Paper tauschen
25	Y	9 Param.	Benutzer-Zeichen Nr. Par1 definieren. Par2 bis Par9 definieren die Zeichen-Matrix
26	Z	4 Param.	Fenster def., Param. wie im BASIC-Befehl
27			ESCAPE, im BASIC keine Funktion
28		3 Param.	Farben für Farbstift, wie Befehl INK
29		2 Param.	Farben für Rand, wie Befehl BORDER
30			Text-Cursor in Pos. 1,1 bringen
31		2 Param.	Text-Cursor setzen, wie Befehl LOCATE

2.2 Betriebssystem

Restart-Befehle

RST 00

Kaltstart

RST 08

Sprung in eine Routine in Speicherblock 0. Bit 14 und 15, die für die Adressenangabe nicht benötigt werden, entscheiden, ob RAM oder ROM adressiert wird. Die Adresse wird in den beiden dem RST08 folgenden Bytes übergeben.

Speicherkonfigurationswahl mit Bits 14 und 15:

Bit	Block	Adressen	Bit=1	Bit=0
14	0	0000 - 3FFF	RAM	Betriebssystem-ROM
15	3	C000 - FFFF	RAM	BASIC-ROM

RST 10

Sprung in eine Routine eines Expansion-ROMs. Die Sprungadresse, die in den beiden dem RST10 folgenden Bytes stehen muß, kann Werte zwischen 0000 und FFFF annehmen. Es wird ein Sprung relativ zum derzeit eingeschalteten ROM ausgeführt. Bits 0 bis 13 geben die Adresse im jeweiligen ROM an, Bits 14 und 15 ergeben den Offset zum derzeitigen ROM.

RST 18

Sprung in eine Routine in beliebigem RAM- oder ROM-Speicherblock. Dem RST18 muß die Adresse des Parameter-Blockes folgen, der aus 3 Bytes besteht.

byte(s)	Inhalt
00,01	Sprungadresse im RAM bzw. ROM
02	Blockselektor

Der Blockselektor kann die ROMs 00 bis FB auswählen. Die Codes FC - FF (Bits 2 bis 7 gesetzt) dienen der RAM/ROM-Wahl, wobei die Bits 0 und 1 wie die Bits 14 und 15 beim RST08 wirken.

RST 20

RAM-Inhalt unabhängig von der Speicherkonfiguration lesen. Im HL-Register steht die RAM-Adresse, in A wird der Inhalt zurückgegeben.

RST 28

Sprung in eine Betriebssystem-Routine, deren Adresse dem RST28 folgt. Vor dem Sprung wird das Betriebssystem-ROM ein-, danach wieder ausgeschaltet.

RST 30

frei für Benutzer

RST 38

Hardware-Interrupt-Einsprung. Einen externen Interrupt erkennt das Programm an dem in AF' gesetzten Carry-Flag. In diesem Fall wird ein Unterprogramm ab Adresse 003B angesprungen.

Betriebssystem-Einsprungpunkte

464	664	6128	Funktion	Übergabe-Register:	ein;aus
B900	B900	B900	KL Block 3-ROM einschalten		
B903	B903	B903	KL Block 3-ROM ausschalten		
B906	B906	B906	KL Block 0-ROM einschalten		
B909	B909	B909	KL Block 0-ROM ausschalten		
B90C	B90C	B90C	KL alte ROM-Konfig. rekonstruieren		A;
B90F	B90F	B90F	KL Expansion-ROM einschalten		C;
B912	B912	B912	KL aktuelle Exp.-ROM-Nr. erfragen		;A
B915	B915	B915	KL Ermittle Expansion-ROM-ID		C;HL,A
B918	B918	B918	KL altes Exp.-ROM wieder einschalten		C;
BACB	BAC6	BAC6	LD A,(HL) unbedingt aus dem RAM		
BADC	BAD7	BAD7	LD A,(IX+00) unbedingt aus dem RAM		
BB06	BB06	BB06	KM Auf Zeichen von Tastatur warten		;A
BB09	BB09	BB09	KM Zeichen von Tastatur lesen		;A
BB5D	BB5D	BB5D	TXT Zeichen auf Bildschirm schreiben		A;
BB60	BB60	BB60	TXT Zeichen von Bildschirm lesen		;A
BBC0	BBC0	BBC0	GRA Graphic-Cursor setzen		DE=x, HL=y
BBEA	BBEA	BBEA	GRA Punkt zeichnen		DE=x, HL=y
BBF6	BBF6	BBF6	GRA Linie zeichnen		DE=x, HL=y

Adressen für relative GRA-Koord.-Angaben um 3 höher

Adressen der Betriebssystem-Variablen

Allgemeines

464	664	6128	Inhalt
B187	B8B4	B8B4	Timer, niederwertiges Wort (LSW)
B189	B8B6	B8B6	Timer, höherwertiges Wort (MSW)
B196	B8C3	B8C3	auszuführender Befehl
B1A8	B8D5	B8D6	aktuelles Expansion-ROM
B1A9	B8D6	B8D7	Einsprungadresse in Expansion-ROM
B1AB	B8D8	B8D9	aktuelle RAM/ROM-Konfiguration

Bildschirm und Text

464	664/6128	Inhalt
B1C8	B7C3	Bildschirm-Modus
B1CB	B7C6	Bildschirm-Adresse (MSB)
B20C	B6B5	aktuelles Bildschirm-Fenster
B20D	B6B6	Fenster-Parameter
B285	B726	akt. Cursor-Zeile
B287	B728	akt. Cursor-Spalte
B288	B729	akt. Fensterbegrenzung oben
B289	B72A	akt. Fensterbegrenzung links
B28A	B72B	akt. Fensterbegrenzung unten
B28B	B72C	akt. Fensterbegrenzung rechts
B28C	B72D	akt. Roll Count
B28D	B72E	akt. Cursor-Flag
B28F	B72F	akt. Pen
B290	B730	akt. Paper
B291	B731	akt. Hintergrund-Modus
B294	B734	1. Benutzer-definierte Zeichen
B296	B736	Adresse der Benutzer-Zeichen-Matrix
B2C3	B763	Steuerzeichen-Ausführungs-Sprungtabelle

Grafik

464	664/6128	Inhalt
B328	B693	Urspungsspalte, -zeile
B32C	B697	akt. X-, Y-Koordinaten
B330	B69B	Gr.-Fenster-Begrenz.-X-Koord. links
B332	B69D	Gr.-Fenster-Begrenz.-X-Koord. rechts
B334	B69F	Gr.-Fenster-Begrenz.-Y-Koord. oben
B336	B6A1	Gr.-Fenster-Begrenz.-Y-Koord. unten
B338	B6A3	Grafik-Pen
B339	B6A4	Grafik-Paper
B342	B6A5	X-, Y-Koordinaten-Rechenpuffer

Tastatur

464	664/6128	Inhalt
B4DE	B628	Zeiger auf Erweiterungszeichen (EWZ)
B4E1	B62B	Basisadresse des EWZ-Speichers
B4E3	B62D	Endadresse des EWZ-Speichers
B4E5	B62F	Anfangsadresse des freien EWZ-Speichers
B4E7	B631	SHIFT-Lock-Zustand
B4E8	B632	CAPS-Lock-Zustand
B4E9	B633	Autorepeat-Verzögerung
B547	B691	Adresse der Autorepeat-Tabelle

Tastatur - ASCII-Code-Tabellen

464	664/6128	Inhalt
B541	B68B	Taste allein
B543	B68D	Taste mit SHIFT
B545	B68F	Taste mit CONTROL

Sound

464	664/6128	Inhalt
B55C	B1F8	Kanal-A-Parameter
B59B	B237	Kanal-B-Parameter
B5DA	B276	Kanal-C-Parameter
B60A	B2A6	Hüllkurven der Lautstärke
B6FA	B396	Hüllkurven des Tons

Cassetten-Rekorder-Interface

464	664/6128	Inhalt
B800	B118	Cass.-Message-Flag
B802	B11A	Eingabe-Puffer-Zustand
B803	B11B	Eingabe-Puffer-Basisadresse
B805	B11D	Eingabe-Puffer-Zeiger
B807	B11F	Eingabe-File-Header
B847	B15F	Ausgabe-Puffer-Zustand
B848	B160	Ausgabe-Puffer-Basisadresse
B84A	B162	Ausgabe-Puffer-Zeiger
B84C	B164	Ausgabe-File-Header
B8D1	B1E9	Aufzeichnungs-Geschwindigkeit

Aufbau des Cassetten-File-Headers

Byte(s)	Inhalt
00 - 0F	Datei-Name
10	Block-Nummer
11	letzter Block, wenn Inhalt <> 00.
12	Datei-Typ, (geschützte Datei, wenn Bit 0 gesetzt)
14,13	Anzahl der Datei-Informations-Bytes
16,15	Lade-Adresse
17	erster Block, wenn Inhalt <> 00.
19,18	Länge des Files
1B,1A	Start-Adresse nach Laden (vom BASIC nicht genutzt)

2.3 Resident System Extensions (RSX)

Einbinden des RSX in das Betriebssystem

Jedes RSX muß dem Betriebssystem mitteilen, wie die von ihm zur Verfügung gestellten Befehle heißen und wo sie ausgeführt werden. Dazu müssen in dem RSX zwei Tabellen vorhanden sein:

```
Tabelle 1 (Adressen):
Bytes     Inhalt
00 - 01   Adresse der Tabelle 2
02 - 04   JP Befehl1
03 - 05   JP Befehl2
:         :
```

In der Tabelle 2 werden die Namen der Befehle in Großbuchstaben abgelegt, wobei im Byte des letzten Buchstaben jedes Befehles das Bit 7 gesetzt sein muß. Das Ende der Tabelle wird durch ein Null-Byte angezeigt.

Außerdem benötigt das Betriebssystem ein 4 Bytes langes Feld für die RSX-Verwaltung, das im Bereich 4000 - BFFF liegen muß.

Das Einbinden des RSX übernimmt die Betriebssystem-Routine bei Adresse BCD1. Sie interpretiert den Wert in BC als Adresse der Tabelle 1 und den in HL als Adresse des Verwaltungsfeldes.

Parameter-Übergabe

Den RSX-Befehlen wird in Register A die Anzahl der übergebenen Parameter und in IX die Adresse des Parameterblockes mitgeteilt. Es sind bis zu 32 Parameter möglich. Die Parameter sind stets 16-Bit-Integer-Zahlen. Der letzte Parameter steht in (IX+00) und (IX+01).

3. CPC-Software: Bedienungsübersichten

3.1 Bankman für den CPC 6128

Das Programm Bankman ist ein BASIC-RSX. Es muß vor dem Aufruf eines BASIC-Programmes, das mit dem Bankman arbeitet, im Speicher stehen. Ein Laden des Bankman vom aufgerufenen BASIC-Programm aus ist nicht möglich.

Die unten verwendeten Abkürzungen entsprechen den im BASIC benutzten.

Zusätzliche Bildschirm-Speicher

SCREEN SWAP ,<int1,> ,int2 ,int3

Tauscht die Inhalte der Bildschirme int2 und int3 aus. Soll nur ein Teil der Bildschirminhalte getauscht werden, kann mit intl die Anzahl der zu tauschenden 256-Byte-Blöcke festgelegt werden.

SCREENCOPY ,<int1>,int2,int3

Überschreibt den Inhalt des Bildschirms int2 mit dem des Bildschirms int2. Soll nur ein Teil überschrieben werden, kann mit intl die Anzahl der zu überschreibenden 256-Byte-Blöcke festgelegt werden.

BANK OPEN,int

Legt die Länge der Datensätze der RAM-Datei fest. Satz 0 wird aktueller Satz, Speicher wird nicht gelöscht.

BANKWRITE,ivar,ausdr$<,int>

Schreibt den Textausdruck in den aktuellen bzw. durch int bestimmten Datensatz. Danach enthält die Integer-Variable ivar 0, wenn die Schreiboperation erfolgreich war, -1, wenn der geforderte Satz außerhalb des 64k-Bereiches läge, und der nächste Satz wird aktueller Satz.

BANKREAD,ivar,var$<,int>

Liest den Inhalt des aktuellen bzw. durch int bestimmten Datensatz in die Stringvariable var$. Danach enthält die Integer-Variable ivar 0, wenn die Leseoperation erfolgreich war, -1, wenn der geforderte Satz außerhalb des 64k-Bereiches läge, und der nächste Satz wird aktueller Satz.

BANKFIND,ivar,ausdr$<,int1<,int2>>

Sucht in der RAM-Datei nach dem ausdr$ ab dem aktuellen bzw. durch int1 bestimmten Datensatz. Zu int1 kann zusätzlich mit int2 die Nummer des letzten zu testenden Datensatzes festgelegt werden. Nach beendeter Suche enthält die Integer-Variable ivar die Nummer des gesuchten Datensatzes oder -1, wenn int2 kleiner als int1 oder der erste Satz außerhalb des 64k-Bereiches läge, -3, wenn kein passender Datensatz vorhanden war.

3.2 Textomat

Programmstart ..... RUN "TEXTOMAT"

Kommandomodus einleiten ..... CONTROL-Taste

Menümodus einschalten. im Kommandomodus mit RETURN

Auswahl der Menüpunkte .....durch Leer- oder Cursortasten

Ausführen des gewählten Menüpunktes .....RETURN-Taste

Menüebene verlassen .....ESC

DEL .....Zeichen links vom Cursor löschen

CLR ..... Zeichen unter Cursor löschen

SHIFT-DEL ..... Einfügemodus ein/aus

SHIFT-Cursor-rechts .....Cursor ein Wort rechts

SHIFT-Cursor-links ..... Cursor ein Wort links

SHIFT-Cursor-hoch .....Cursor an Textanfang

SHIFT-Cursor-ab .....Cursor an Textende

Besondere Kommandos
(zuerst CONTROL drücken, dann:)

Cursor-hoch ..... eine Seite zurückblättern

Cursor-ab .....eine Seite vorblättern

Cursor-rechts ..... Cursor in letzte Bildschirmspalte mit Text

Cursor-links ..... Cursor in erste Bildschirmspalte

DEL .....Zeile löschen

SHIFT-DEL ..... Zeile einfügen

t ..... Trennvorschlag für das Formatieren

s ..... Steuerzeichen

ENTER/RETURN ..... Menümodus

Steuerzeichen im Text

(Erst CONTROL drücken, dann "s" (erscheint invers), dann Steuerzeichen eingeben)

d ..... Datum einfügen

f ..... Fettschrift ein/aus

U ..... Unterstreichen ein/aus

c ..... Zentrieren ein/aus

o ..... Exponent-Modus

u ..... Index-Modus

0...9 ..... frei definierbare Steuerzeichen (Druckertabelle)

bl ..... Blocksatz ein/aus

abx ..... Zeilenabstand (x=1,2,3)

axx ..... Adresse aus Feld xx einsetzen

lnxxxx ..... Leerzeilen drucken

lrxx ..... linken Rand auf xx setzen

erxx ..... diese Zeile xx Zeichen nach links schieben

swxx ..... Seitenwechsel, wenn nicht noch xx Zeilen frei

dixx ..... Zeichendichte (xx =10,12,15)

rrxxx ..... rechten Rand auf xxx setzen

ndx name ..... nächsten Text name von Laufwerk x einlesen

Rechnerfunktionen
(zuerst Cursor auf Zahl stellen, CONTROL drücken, dann)

c..... Rechenspeicher löschen

+ ..... Addition/erste Zahl laden

- ..... Subtraktion

*** ..... Multiplikation

/ ..... Division

% ..... Prozent

0...8 ..... Ausgabegenauigkeit

3.3 Datamat

Programmstart ..... RUN"DATAMAT"

Kommandomodus einleiten ..... CONTROL-Taste

Menümodus einschalten. im Kommandomodus mit RETURN

Auswahl der Menüpunkte .....durch Leer- oder Cursortasten

Ausführen des gewählten Menüpunkts .....RETURN-Taste

Menüebene verlassen ..... ESC

ENTER .....Cursor in nächstes Eingabefeld

DEL .....Zeichen links vom Cursor löschen

CLR ..... Zeichen unter Cursor löschen

SHIFT-Cursor-rechts ..... Leerzeichen einschieben

SHIFT-Cursor-links ..... wie DEL

SHIFT-CLR .....Löscht alle Eingabefelder

SHIFT-ENTER ..... Abschluß der Eingabemaske

Abweichende Funktionen beim Erstellen der Eingabemaske

ENTER ..... Cursor in nächste Zeile

SHIFT-Cursor-hoch .....Löscht Zeile

SHIFT-Cursor-ab .....Fügt Zeile ein

SHIFT-CLR ..... Löscht ganze Eingabemaske

SHIFT-ENTER ..... Auswahl des Indexfeldes

^ ..... Anfang und Ende einer Eingabemaske

4. CP/M-Anwenderebene

4.1 Datei-Namen

Formaler Aufbau eines Dateibezeichners:

l:filename.typ

l bezeichnet das Laufwerk, in dem die Datei steht bzw. in das die Datei geschrieben werden soll.

filename Der Dateiname kann 0 bis 8 Zeichen lang sein und sollte nach Möglichkeit keines der folgenden Zeichen enthalten oder 0 Zeichen lang sein:

Leerzeichen . : = _ , ; < > * ? [ ]

.typ Der Dateityp kann 0 bis 3 Zeichen lang sein und sollte keines der obigen Zeichen enthalten. Die folgende Tabelle zeigt einige häufig vorkommende Dateitypen:

COM	Befehlsdatei, von CP/M ausführbar.
ASM	Assembler-Quellprogramm
PRN	vom Assembler erzeugtes Programm-Listing.
HEX	Programmcode im Hex-Format
SUB	Submit-Datei
$$$	Zwischendatei während des Programmlaufes
BAK	Sicherheitsdatei, wird meist autom. erzeugt.
OVR	Overlay

Quellprogramme höherer Programmiersprachen:

```
BAS  Basic
BIN  Binärdatei, Maschinenprogramme zu Basic
COB  Cobol
FOR  Fortran
PAS  Pascal
INC  Include-Datei zu Pascal
```

4.2 CP/M2.2-Befehle und -Programme

CP/M-Befehle, die in diesem Abschnitt nicht behandelt werden, sind für CP/M2.2 und CP/M3.0 identisch und in Abschnitt 4.4 beschrieben.

ASM filename<.shp> — **8080-Assembler**

Das Programm übersetzt ein Assembler-Quellprogramm mit der Datei-Kennung ASM in eine HEX-Datei und eine Protokoll-Datei mit der Kennung PRN.

s (A..P) Laufwerk in dem die Quelldatei steht

h (A..P,Z) Laufwerk in das die Hex-Datei soll

p (A..P,X,Z) Laufwerk in das das Protokoll soll

- **X** Daten an Konsole
- **Z** Datei nicht anlegen

Der Assembler kennt neben den 8080-Mnemonics noch einige Pseudobefehle:

EQU wert einem symbolischen Namen einen festen Wert zuweisen.

SET wert einem symbolischen Namen einen Wert zuweisen, der mit anderen SET-Befehlen wieder geändert werden kann.

DS x x Bytes ab aktueller Adresse freihalten.

DB list die Byte-Werte der Liste ab der aktuellen Adresse im Speicher ablegen.

DW list die Wort-Werte der Liste ab der aktuellen Adresse im Speicher ablegen.

ORG adr Adresse des nächsten Befehls/Datums festlegen.

IF log arg Der Quelltext bis zum nächsten ENDIF wird nur übersetzt, wenn das logische Argument wahr ist.

ENDIF Ende des bedingten Programmteiles.

END<adr> Programmende markieren und evtl. Programmstartadresse angeben (diese wird durch DDT erkannt, nicht aber durch CCP).

CLOAD/CSAVE file

Dateien von Diskette auf Cassette kopieren und umgekehrt.

DDT <file>

Die Befehle des symbolischen Debuggers:

Ifile Datei für nächsten R-Befehl wählen.

R<rel> Datei an die Adresse laden, die um rel gegenüber der Originaladresse verschoben ist.

Gadr1,adr2,adr3
Programm ab adr1 ausführen, bis adr2 oder adr3 erreicht wird.

Tstep Befehle einzeln abarbeiten, nach jedem Schritt Prozessorzustand ausgeben.

Ustep dto., jedoch Zustand einmalig am Ende anzeigen.

X Prozessorzustand ausgeben (Flags, Register)

Xreg Inhalt des Registers reg ausgeben, evtl. ändern

Dadr1,adr2 Inhalt des Speichers von adr1 bis adr2 in Hex- und ASCII-Code ausgeben.

Ladr1,adr2 Inhalt des Speichers in Mnemonics ausgeben

Aadr Ab adr mnemonisch eingegebene Befehle in Hex-Code übersetzt ablegen. (Mini-Assembler).

Sadr Ab adr im Hex-Code eingegebene Daten ablegen

Fadr1,adr2,wert
Speicher von adr1 bis adr2 mit wert füllen.

Madr1,adr2,adr3
Speicherbereich von adr1 bis adr2 nach adr3 kopieren.

Hwert1,wert2
Summe und Differenz der Werte ausgeben.

^C Programm beenden, Warmstart

DIR <l:><file>

Das DIR-Programm gibt Auskunft über Disketteninhalte.

alle Dateien des angemeldeten Laufwerkes listen.

l: Laufwerk auswählen.

file Datei auswählen, Jokerzeichen * und ? erlaubt.

FILECOPY file

Dateien mit einem Laufwerk auf andere Diskette kopieren.

LOAD file

Aus einer HEX-Datei eine COM-Datei erzeugen.

MOVCPM <mem><*>

CP/M nach Systemspeicher-Änderung neu konfigurieren. mem kann eine Zahl zwischen 64 und 179 sein und gibt an, um wie viele Seiten (= 256 Bytes) das CP/M verschoben werden soll.

SAVE p file

Der residente SAVE-Befehl schreibt den Inhalt von p Seiten zu je 256 Bytes des TPA unter dem Namen file auf Diskette.

SETUP

Interaktive System-Installation.

STAT parameter

Laufwerks- und Dateizustände ausgeben oder ändern. Mit Parameter werden die verschiedenen Zustände ausgewählt:

<l:>	Anzahl der freien Speicherbytes ausgeben
VAL:	Liste der STAT-Befehle ausgeben
DEV:	IO-Byte ausgeben
l:DSK:	physikalische Laufwerksparameter ausgeben
l:USR:	User-Bereiche und Anz. Dateien darin ausgeben
l:=R/O	Laufwerk gegen Beschreiben schützen
file<opt>	Dateilängen, -attribute<, und mehr> listen
log:=phys:	der logischen die physikalische Geräteeinheit zuordnen. Mehrere Zuordnungen können im Befehl aufgelistet werden.

```
log. Einheit                mögliche phys. Geräte
CON: Konsole                CRT: TTY: BAT: UC1:
RDR: Lochstreifenleser      PTR: TTY: UR1: UR2:
PUN: Lochstreifenstanzer    PTP: TTY: UP1: UP2:
LST: Lister                 CRT: TTY: LPT: UL1:
```

physikalische Geräte:

```
CRT: Bildschirm-Terminal
TTY: Fernschreiber
PTR: Lochstreifenleser
PTP: Lochstreifenstanzer
LPT: (Zeilen)-Drucker
UC1: Benutzer-definiertes Ein-/Ausgabe-Gerät
URn: Benutzer-definierte Eingabe-Geräte
UPn: Benutzer-definierte Ausgabe-Geräte
UL1: Benutzer-definiertes Ausliste-Gerät
```

Optionen zu STAT file:

```
$S      Virtuelle Dateilänge mit ausgeben
$R/O    Datei gegen Beschreiben schützen
$R/W    Schreibschutz aufheben
$SYS    Systemattribut ein
$DIR    Systemattribut aus
```

SYSGEN

Betriebssystem im Speicher auf System-Spuren kopieren.

4.3 CP/M3.0-Befehle und -Programme

Befehle, die in diesem Abschnitt nicht behandelt werden, sind für CP/M3.0 und CP/M2.2 identisch und im folgenden Abschnitt 4.4 beschrieben.

System-Verwaltung

DATE<SET><mm/dd/yy hh:mm:ss><c>

Das Programm stellt die systemeigene Uhr interaktiv (SET) bzw. direkt, oder gibt die Uhrzeit aus (ohne Parameter einmalig, mit c bis zum nächsten Tastendruck).

DEVICE <NAMES> <VALUES> <log:=phys <opt1>> <CON: <opt2>>

NAMES zeigt die implementierten Namen und vereinbarten Übertragungsgeschwindigkeiten der Peripherie-Geräte an.

VALUES zeigt die Zuordnung der physikalischen Geräte zu den logischen Kanälen, die jederzeit beliebig geändert werden kann. Es können auch mehrere Geräte an einem Kanal und ein Gerät an mehreren Kanälen anliegen.

log. Kanäle		phys.	Standardeinstellung
CONIN:	Konsoleneingabe	CRT	Tastatur
CONOUT:	Konsolenausgabe	CRT	Bildschirm
AUXIN:	Hilfseingang,	SIO	nicht belegt
AUXOUT:	Hilfsausgang,	SIO	nicht belegt
LST:	Lister-Ausgabe	LPT	Centronics-Port

CON:<PAGES>
zeigt das Bildschirmformat

CON:<COLUMNS=x><LINES=y>

legt normalerweise ein neues Bildschirmformat fest. (Konnte bei CPCs bisher nicht beobachtet werden)

SIO<bd> Übertragungsgeschwindigkeit der seriellen Schnittstelle wählen.

SIO<XON><NOXON>

X-Protokoll ein-/ausschalten

DIR <l:> <file> <opt>

DIR gibt die durch die Parameter und Optionen spezifizierten Directory-Einträge aus. Ohne Optionen ist DIR ein eingebauter (residenter) Befehl.

alle Dateien des angemeldeten Laufwerkes

l: bestimmtes Laufwerk auswählen

file Datei auswählen, Jokerzeichen * und ? erlaubt

Optionen sind in eckigen Klammern anzugeben. Die folgenden Optionen sind möglich, wenn die Datei DIR auf Diskette vorliegt:

ATT	Dateiattribute f1 bis f4 mit ausgeben
DATE	Zeiteinträge der Dateien mit ausgeben
DIR	nur Dateien ohne System-Attribut
DRIVE=llist	alle aufgelisteten Laufwerke untersuchen
EXCLUDE	alle Dateien außer der angegebenen
FF	Formfeed, wenn Druckerecho ein
FULL	alle Directory-Informationen
LENGTH=n	(n=5..65535) Zeilenzahl pro Seite
MESSAGE	akt. Laufwerksnr. und User-Bereich
NOPAGE	nicht seitenweise ausgeben
NOSORT	nicht alphabetisch sortieren
RO	nur Dateien mit Schreibschutz
RW	nur Dateien ohne Schreibschutz
SIZE	nur Dateiname und -länge
USER=ulist	Dateien der aufgelisteten User-Bereiche

Die Listen werden wie folgt angelegt, wobei l für Laufwerk und u für User-Bereich steht:

```
ALL             Alle l bzw. u
l bzw. u        nur l bzw. u
(l1,l2<,..>)    Alle in der Liste aufgeführten lx/ux
```

HELP <stichwort><EXTRACT><CREATE>

HELP bietet die Möglichkeit, sich über die CP/M-Befehle und Programme zu informieren. Leider sind alle Texte englisch. Bei der Befehlseingabe kann ein Stichwort mit angegeben werden.

Mit der Option EXTRACT kann der Text in der HELP.HLP-Datei geändert (oder gar übersetzt?) werden. Die Änderung wird mit der Option CREATE auf Diskette übernommen. Die Optionen werden in eckigen Klammer angegeben.

INITDIR l:

bereitet das Directory auf die erweiterten Directory-Einträge (Datum,Uhrzeit und Paßwörter) vor.

LANGUAGE n

Zeichensatz an Sprachraum anpassen:

```
n  Zeichensatz
0  amerikanisch (Standard-Einstellung)
1  französisch
2  deutsch
3  englisch (außer amerik. einziger mit [])
4  dänisch
5  schwedisch
6  italienisch
7  spanisch
```

PALETTE inkliste

Mit diesem Befehl kann man den Farbstiften neue Farben zuweisen. Möglich sind Werte von 0 bis 63. Es können bis zu 16 Parameter angegeben werden.

Im 80-Zeichen/Zeile-Modus wirken jedoch nur die ersten beiden. Der erste Parameter bestimmt dann die Hintergrundfarbe und der zweite die Schriftfarbe. Die Farben werden so aus den drei Grundfarben zusammengesetzt, daß blau durch die Bits 0 und 1, rot durch 2 und 3 und grün durch die Bits 4 und 5 gesteuert wird. Je Grundfarbe sind 3 Helligkeitsstufen möglich: kein Bit, eines von beiden Bits oder beide Bits gesetzt.

PROFILE.SUB

Diese Submit-Datei beinhaltet CP/M-Befehle, die vom CCP unmittelbar nach dem Kaltstart automatisch ausgeführt werden.

SET <l:><file><option>

Disketten- und Betriebsparameter festlegen.

Optionen beim Setzen von Disketten-Attributen (SET <l:>):

NAME=discname.typ	Name zuordnen, Regel wie für Dateien
PROTECT=<ON><OFF>	Paßwortschutz ein- bzw. ausschalten

Optionen beim Setzen von Datei-Attributen (SET file):

CREATE=ON	Zeiteintrag bei Datei-Erstellung
ACCESS=ON	Zeiteintrag bei jedem Zugriff
UPDATE=ON	Zeiteintrag bei jedem Schreibzugriff
PROTECT=READ	Paßwort-Schutz gegen jeden Datei-Zugriff

PROTECT=WRITE	PW-S gegen Beschreiben,Löschen,Rename
PROTECT=DELETE	PW-S gegen Löschen und Umbenennen
PROTECT=NONE	Paßwort-Schutz aufheben
ARCHIVE=<ON><OFF>	Archiv-Attribut ein- bzw. ausschalten
<SYS><DIR>	System-Attribut ein- bzw. ausschalten
Fx=<ON><OFF>	(x=1..4) Attr. Fx ein- bzw. ausschalten

Optionen beim Setzen von Disketten- und Datei-Attributen:

RO	Schreibschutz einschalten
RW	Schreibschutz ausschalten

SHOW <l:><option>

Laufwerks- und Dateizustandsinformationen ausgeben. SHOW allein gibt den freien Speicherplatz aller angeschlossenen Laufwerke aus. Die Angabe eines Laufwerkes reduziert die Ausgabe auf die Daten dieses Laufwerkes. Zur Abfrage spezieller Informationen stehen folgende Optionen zu Verfügung:

DIR	Anzahl der freien Directory-Plätze ausgeben
SPACE	Zugriffs-Modus und freien Speicherplatz ausgeben
USERS	User-Bereiche und Anzahl der Dateien darin ausgeben
LABEL	Diskettenname ausgeben
DRIVE	physikalische Laufwerksparameter

SETDEF <llist><opt>

Mit der Angabe einer Laufwerks-Liste wird CP/M angewiesen eine Datei in der in der Liste angegebenen Reihenfolge in den Laufwerken zu suchen. Der * steht für das angemeldete Laufwerk (Beispiel für llist: *,a,b)

Optionen:

DISPLAY=ON/OFF	Der Name der z.Z. bearbeiteten Datei wirdauf dem Bildschirm angezeigt.
TEMP=l:	Temporäres Laufwerk wählen
<NO>PAGE	Bildschirm-weise Ausgabe ein-/ausschalten

SET24X80 <ON><OFF>

Bildschirm auf 24x80 Zeichen stellen bzw. 24x80-Modus ausschalten. Im 24x80-Modus wird die 25. Zeile als Statuszeile benutzt.

SETKEYS file

Die in der Datei file stehenden Tastaturbelegungen werden eingeführt.

Aufbau einer Zeile des Tastaturbelegungsfiles:

```
x <N> <S> <C> "Zeichen" <Kommentar>
oder
E nr "Text" <Kommentar>
x = Tasten-Nummer
N = Taste allein
S = Taste mit SHIFT
C = Taste mit CONTROL
E = Erweiterungszeichen Nummer nr definieren
```

Ein Zeichen kann aus dem ASCII-Zeichen selber oder dem in einfachen Anführungszeichen geschriebenen ASCII-Code bestehen. Ein Text ist eine Folge von Zeichen. Hexadezimal-Zahlen für nr oder ASCII-Code werden durch ein vorangestelltes gekennzeichnet. Control-Steuerzeichen und Erweiterungszeichen werden durch ein vorangestelltes ^ gekennzeichnet.

Einige Beispiele:

46 C "^C" Warmstart durch drücken der Taste 46 und CONTROL

6 N S C "^'8E'"
Erweiterungszeichen 8E auf Taste 6

E 8E "DIR *.BAK ^M"
Mit Taste 6 alle BAK-Dateien listen

SETLST file

Das file wird zum Drucker gesandt. So kann dieser bequem initialisiert werden, da in der Datei alle Arten von Steuerzeichen (^'ASCII-Code',^ASCII-Zeichen oder ^'Steuercode-Name') stehen können.

SETSIO

Ohne Parameter werden die aktuellen Daten der seriellen Schnittstelle (RS 232, nicht serienmäßig eingebaut) angezeigt (vgl. DEVICE, AUX-Kanal).

Parameter:

XON/XOFF	X-Protokoll ein/aus
HANDSHAKE ON/OFF	Überwachung der Datentransfers ein/aus
BITS b	b ist Anzahl der Bits pro Block
PARITY <NONE>	Parity-Bit wird übertragen, bzw. nicht

RX baud	Empfangsgeschwindigkeit festlegen
TX baud	Sendegeschwindigkeit festlegen
STOP s	Anzahl der Stop-Bits

Maschinenprogramm-Bearbeitung

HEXCOM file

Übersetzt das mit MAC erzeugte file.HEX in das ausführbare Programm file.COM.

LINK<neu<opt>=>alt1<opt><,altx<opt>..>

Verbindet speicherrelative Programme.

MAC file <$opt>

8080-Macro-Assembler. Aus den Dateien file.ASM und file.LIB werden die Datei file.HEX, die Symbol-Tabelle file.SYM und das Protokoll file.PRN erzeugt.

RMAC file <$opt>

8080-Macro-Assembler, der verschiebbaren Objektcode (in Datei file.REL) erzeugt. Er kennt folgende Optionen:

Pl	(l=A..P,X,Z) Laufwerk für PRN-Datei
Rl	(l=A..P,Z) Laufwerk für REL-Datei
Sl	(l=A..P,X,Z) Laufwerk für SYM-Datei
X	Ausgabe an Konsole
Z	Datei unterdrücken

SAVE

Das SAVE-Programm wird aufgerufen, bevor das zu sichernde Programm in den TPA gelangt. Mit dem nächsten Warmstart wird SAVE aktiviert. Die benötigten Daten werden erfragt.

SID

Symbolischer 8080-Debugger (Befehle wie DDT). Zu SID gehören die UTL-Programme TRACE (Programmverlauf vor einem Haltepunkt ausgeben) und HIST (Graphische Darstellung der Benutzungshäufigkeit von Programmteilen)

XREF file <$P>

erstellt eine Crossreferenzliste, d.h. XREF gibt zu den mit MAC oder RMAC erzeugten Symboldateien unter Zuhilfenahme der gleichnamigen PRN-Datei die Programmzeilen der Symbole aus.

Graphik-Unterstützung

GENGRAF file

Mit diesem Programm wird der GSX-Lader in das Programm file eingefügt.

GSX.SYS

Enthält den peripherieunabhängigen Teil der Graphics-System-Extension.

ASSIGN.SYS

Diese Datei enthält die Codes und Namen der Peripherie-Treiberprogramme DD-DMP1, DDHP7470, DDSHINWA, DDFXLR7, DDMODE0, DDMODE1, DDMODE2. Die Treiber werden je nach Anforderung vom GSX geladen.

DRIVERS.GSX

Ist eine Textdatei, in der man nachlesen kann, welche PRL-Datei für welche Anwendung gedacht ist.

4.4 Gemeinsame Befehle und Programme

DUMP file

Dateiinhalt in Hex-Code und in CP/M3.0 auch in ASCII-Code ausgeben. Der Datenfluß läßt sich mit ^S unterbrechen, ^Q setzt die Ausgabe fort, ^C bricht DUMP ab.

ED file

Zeilen-Editor

nA	n Zeilen aus file an Puffer anhängen
nW	n Zeilen aus file in Zwischendatei schreiben
nX	n Zeilen aus Puffer in Hilfsdatei schreiben
R	Text in der Hilfsdatei in Puffer-Text einfügen
Rdatei	Text in datei.LIB in Puffer-Text einfügen
<+><->nK	n Zeilen ab (+)/ bis (-) Textzeiger löschen

Zeichen-Editor

Itext	Text mit Zeilenende einfügen
Itext^Z	Text einfügen
I	Text-Einfüge-Modus ein (mit ^Z beenden)
<+><->nD	n Zeichen löschen
nSalt^Zneu^Z	n-ten Text alt durch Text neu ersetzen
Jtext^Zneu^Zweg^Z	Text suchen, dahinter neu einfügen, dann Text bis weg ausschließlich löschen.

Textzeiger bewegen

<+><->B	Textzeiger an Pufferanfang bzw. -ende
<+><->nC	Textzeiger n Zeichen versetzen
<+><->nL	Textzeiger n Zeilen versetzen
nFtext	Textzeiger auf n-tes Auftreten des text setzen
nNtext	dto., ggf. Text aus file nachladen

Textausgaben

<+><->n	wie nL, aber neue Zeile ausgeben
<+><->nT	n Zeilen ab bzw. bis zum Textzeiger ausgeben
<+><->nP	dto., aber Textzeiger versetzen

diverse Befehle

<+><->V	Zeilennummern-Ausgabe ein bzw. aus
OV	freien/gesamten Pufferplatz in Bytes ausgeben
nMliste	Die in der liste aufgeführten Befehle n-mal ausführen.
nZ	n Zeiteinheiten warten
<+><->U	"alle Eingaben in Großbuchstaben" ein bzw. aus. Ist -U gewählt, entscheidet der jeweils eingegebene Befehl, wie der Text zu behandeln ist. Wird der Befehl als kleiner Buchstabe eingegeben, wird der Text wie vorge funden bearbeitet; ist der Befehlsbuchstabe groß ge schrieben, erfolgt die Umwandlung des zu bearbeitenden Textausschnittes in Großbuchstaben.

Editieren beenden

E	Puffer- und file-Restinhalt in Zwischendatei, Dateien schließen und umbenennen, ED beenden.
H	wie E, aber ED neu beginnen.
O	Zwischendatei löschen und ED neu beginnen.
Q	Zwischendatei löschen und ED beenden.

ERA file

ERA ist in beiden CP/M-Versionen ein residenter (im CCP enthaltener) Befehl zum Löschen von Dateien auf einer Diskette. Im Dateinamen dürfen auch Jokerzeichen vorkommen.

In CP/M3.0 läßt ERA die Option CONFIRM oder C zu, wenn die Datei ERASE auf Diskette vorliegt. Sie bewirkt, daß vor jeder eventuellen Löschung das Einverständnis des Bedieners erfragt wird.

PIP ziel=quelle1<opt><,quelle2<opt>>

Datenübertragung zwischen Peripheriegeräten. PIP ohne Argument lädt und startet das PIP-Programm.

ziel	Datei-Einfachname oder Ausgabe-Einheit
quellen	Datei-Einfachnamen oder Eingabe-Einheiten
l:=quelle	Das Ziel hat den gleichen Namen, wie die Quelle, jedoch soll es in einem anderen Laufwerk sein.
l:	Ziellaufwerk, das vom Quelllaufwerk verschieden sein muß.
quelle	Datei-Ein- oder -mehrfachname
Optionen:	
L	alle Buchstaben in Kleinbuchstaben umwandeln
U	alle Buchstaben in Großbuchstaben umwandeln
Z	alle Bits 7 in der Datei löschen
F	alle ^L (Form Feed) aus der Datei entfernen
Dn	alle Zeichen löschen, die über Spalte n hinausragen
Tn	Tabulationsspaltenbreite auf n setzen
N	Zeilen mit fortlaufenden Nummern versehen
N2	dto., führende Nullen nicht ausgeblendet
Pn	nach allen n Zeilen ^L (Form Feed) einfügen
Stext^Z	Datei ab text kopieren
Qtext^Z	Datei bis text kopieren

Die S- und die Q-Option können je Einzelquelle nur einmal angegeben werden.

E	eingelesenen Text an Konsole ausgeben
V	Zieldatei überprüfen, wenn auf Diskette
O	ganze Datei kopieren, ^Z als HEX-Wert 1A ansehen
W	Schreibschutz ignorieren
R	auch Systemdateien kopieren
Gn	Datei aus Benutzerbereich n holen
B	blockweise bis ^S einlesen, Ende mit ^Z
H	prüfen, ob Intel-Hex-Format
I	dto. und Kommentar nach Hex-Datei ignorieren

REN neu=alt

Datei umbenennen. neu und alt müssen Einfachnamen sein. REN ist ein residenter Befehl. Das CP/M3.0-Programm REN<AME> kann auch Mehrfachnamen verarbeiten.

SUBMIT file <para1<,para2..>>

Befehle, die in der Datei file stehen, werden ausgeführt. Dabei wird $1 im Dateitext durch Parameter 1 ersetzt, $2 durch para2, etc.

TYPE file

Residenter Befehl zur Ausgabe von ASCII-Dateien. Der Ausgabefluß kann mit ^S unterbrochen und mit ^Q fortgesetzt werden.

Das CP/M3.0-Programm TYPE stellt die Optionen PAGE (Ausgabe alle 24 Zeilen unterbrechen) und NOPAGE zur Verfügung.

USER n

Benutzerbereich umschalten. USER ist ein residenter Befehl. Wird bei CP/M3.0 der Benutzerbereich n weggelassen, wird er vom CCP erfragt.

5. CP/M-SOFTWARE

5.1 WordStar

Die Zeichen +, - und * geben Turbo-Pascal-Benutzern Auskunft über die Verfügbarkeit der Befehle:

*	bedeutet Funktion identisch
+	Funktion ähnlich
-	Funktion wesentlich anders

Die Befehle in runden Klammern gibt es nur in Turbo-Pascal.

Cursor-Steuerung

^S	+	1 Zeichen nach links
^D	+	1 Zeichen nach rechts
^E	+	1 Zeile nach oben
^X	+	1 Zeile nach unten
^A	*	1 Wort nach links
^F	*	1 Wort nach rechts
^I	*	zur nächsten Tabulatorposition
^QS	*	zum linken Rand
^QD	*	zum rechten Rand
^QE	*	zum oberen Rand
^QX	*	zum unteren Rand
^QB	*	zum Blockanfang
^QK	*	zum Blockende
^QR	*	zum Dateianfang
^QC	*	zum Dateiende
^QP	*	zur letzten Position
^QV		zur Position vor dem letzten Such- o. Blockbefehl
^Qx		zum Merker x (x=0...9)

Editorbefehle

^V	*	Einfüge-Modus ein/aus
^B		neu formatieren
(^QI	*	Automatische Tabulierung ein/aus)
DEL	*	Zeichen links vom Cursor löschen
^G	*	Zeichen unter dem Cursor löschen
^T	*	Ab Cursor bis Wortende nach rechts löschen
^QDEL		Zeilenteil links vom Cursor löschen
^QY	*	Zeilenteil rechts vom Cursor löschen
^Y	*	ganze Zeile löschen
^N	*	Zeile einfügen
^U	*	Befehlsausführung abbrechen
^JH		Hilfsstufe setzen
^QQ		Nächsten Befehl wiederholen
(^QL	*	Änderungen in der aktuellen Zeile widerrufen)

Such- und Ersetzbefehle

^L	*	Ersetzen/Suchen fortsetzen
^QA	*	Text ersetzen
^QF	*	Text suchen

Optionen zu den Such- und Ersetzbefehlen

B	*	Rückwärts suchen
G	*	Im ganzen Text ersetzen
N	*	Ersetzen ohne Benutzerzustimmung
U	*	Unabhängig von Groß- und Kleinschreibung suchen
W	*	nur nach dem Suchbegriff als ganzes Wort suchen
int	*	Vorgang wird int-mal ausgeführt

Block- und Datei-Bearbeitung

(Alle Befehle dieser Gruppe werden mit ^K eingeleitet.)

x		Merker x setzen/löschen (x=0...9)
B	*	Blockanfang markieren
K	*	Blockende markieren
(T	*	Wort als Block markieren)
H	*	Markierter Block sichtbar/unsichtbar
N		Spaltenblockmodus ein/aus
C	*	Block kopieren
V	*	Block verschieben

Y	*	Block löschen
W	*	Block in Datei speichern
R	*	Textbaustein/Datei einlesen
O		Datei kopieren
P		Datei drucken
E		Datei umbenennen
J		Datei löschen
F		Inhaltsverzeichnis ein/aus
L		Anderes Standard-Laufwerk anmelden
Q		Bearbeitung ohne Sichern abbrechen
D		Text in Datei speichern, zurück ins StartMenü
X		Text in Datei speichern, zurück ins Betriebssystem
S		Text in Datei sichern, Cursor zum Textanfang

Formatierung

(Alle Befehle dieser Gruppe werden mit ^O eingeleitet)

G	Absatz um einen Tabulator einrücken
C	Zeile zentrieren
X	Rand freigeben
N	Tabulator löschen
I	Tabulator setzen
F	Zeile wird Formatzeile
L	linken Rand setzen
R	rechten Rand setzen
S	Zeilenabstand setzen
V	Umschaltung zwischen festem und variablem Tabulator
D	Druckbefehle anzeigen/nicht anzeigen
H	Trennhilfe ein/aus
P	Seitenanzeige ein/aus
T	Formatzeile ein/aus
E	Weicher Trennstrich ein/aus
J	Blocksatz ein/aus
W	Wortumbruch ein/aus

Die Formatzeile legt linken, rechten Rand und die Tabulatoren für den folgenden Text fest. Erstes und letztes Zeichen der Formatzeile legen die Ränder fest. Für die Tabs gilt:

- löscht Tabulatoren an dieser Stelle
I setzt Standard-Tabulator an dieser Stelle
  setzt den Dezimal-Tabulator

Druckersteuerung

(Alle Befehle dieser Gruppe werden mit ^P eingeleitet)

B	Fettdruck ein/aus
D	Doppelanschlag ein/aus
S	Unterstreichen ein/aus
T	Exponentmodus (hochstellen) ein/aus
V	Indexmodus (tiefstellen) ein/aus
X	Durchstreichen ein/aus
Y	Farbband-Umschaltung
H	letztes Zeichen überdrucken
M	oder RETURN: Zeile überdrucken
N	Standard-Schriftdichte ein
A	zweite Schriftdichte ein
O	festes (einzelnes) Leerzeichen
C	Druckpause
Q	Anwender-Steuerzeichen 1
W	Anwender-Steuerzeichen 2
E	Anwender-Steuerzeichen 3
R	Anwender-Steuerzeichen 4
F	Phantom-Leerzeichen
G	Phantom-RUBOUT

Punktbefehle

(Alle Befehle dieser Gruppe werden mit einem Punkt eingeleitet, der in der ersten Spalte einer neuen Zeile stehen muß.)

PA	Seitenvorschub
CP	Bedingter Seitenumbruch
LH	Zeilenhöhe in 1/48-Zoll
CW	Zeichenbreite in 1/120-Zoll
SR	Abstand des Exponenten oder Index in 1/48-Zoll
PL	Papierlänge
MT	Oberer Rand
MB	Unterer Rand
PN	Seitennummer
PC	Spalte für Seitennummer
OP	Seitennummer unterdrücken
PO	Druckspalte auf Papier für erste Bildschirmspalte

HM Abstand der Kopfzeile
HE Kopfzeile definieren
FM Abstand der Fußnoten
FO Fußzeile definieren
IG oder "..": Zeile ist Kommentarzeile
BP Druckweg-Optimierung (1=ein, 0=aus)
UJ Mikro-Justifikation (1=ein, 0=aus)

Wichtige Werte zeigen die folgenden Tabellen:

Zeilendichte	Punktkommando	Zeichen/Zoll	Punktkommando
3.0-zeilig	.LH24	6	.CW20
2.5-zeilig	20	8	15
2.0-zeilig	16	10	12
1.5-zeilig	12	12	10
1.0-zeilig	8	15	8

MailMerge-Punktbefehle

(Alle Befehle dieser Gruppe werden mit einem Punkt eingeleitet, der in der ersten Spalte einer neuen Zeile stehen muß.)

FI Datei einfügen
LM linken Rand setzen
RM rechten Rand setzen
LS Zeilenabstand setzen
PF Zeilenformatierung beim Drucken
IJ Blocksatz bei der Eingabe
OJ Blocksatz beim Druck
DM Text auf Bildschirm beim Druck anzeigen
CS Bildschirm löschen und evtl. Meldung anzeigen
DF Datendatei definieren
RV Variablenwerte aus Datendatei lesen (Name, Name, ...)
SV Variable setzen (Name, Wert)
AV Variablenwert abfragen
RP wiederholen, bis alle Daten verarbeitet sind

5.2 Turbo-Pascal

Editieren

Die Editierkommandos entsprechen weitgehend den Befehlen des WordStar. In den Tabellen der Abschnitte über WordStar sind die Befehle, die auch in Turbo-Pascal implementiert sind, gekennzeichnet. So können diese Tabellen für beide Programme benutzt werden. Ein wesentlicher Unterschiede besteht bei dem Befehl ^KD.

^KD bricht die Edit-Funktion ab und führt ins Hauptmenü zurück. Das editierte Programm geht dabei nicht verloren, es wird aber auch nicht auf Diskette sichergestellt.

Standard-Typen mit einer Komponente

REAL

Reelle Zahlen mit einer 11 stelligen Dezimal-Mantisse und einem Exponenten zwischen -38 und +38.

INTEGER

Ganze Zahlen im Bereich -32767..32767 dezimal bzw. $0000..$FFFF hexadezimal.

BYTE

Ganze Zahlen im Bereich 0..255 dezimal bzw. $0..$FF hexadezimal.

BOOLEAN

Dieser Typ kann den Zustand TRUE oder FALSE haben.

CHAR

Die Zeichen des ASCII.

c1..c2

Teilbereichs-Typ. c1 und c2 sind Konstanten des übergeordneten Typs, wobei die Ordnungszahl von c1 kleiner als die von c2 sein muss.

Typen mit mehreren Komponenten

ARRAY scaltyp1<,scaltyp2<,...>>OFtyp3

Eine Komponente wird durch arraynameindex1 <,index2<,...>> ausgewählt.

RECORD feldbez1:feldtyp1;<feldbez2: feldtyp2; <...>>END;

Eine Komponente wird durch recname.feldbez ausgewählt.

SETOF scaltype

Ein Element wird durch seinen Wert ausgewählt. Mengen-Konstanten werden von eckigen Klammern begrenzt, in denen die Elemente oder Bereiche des scaltypes der Menge durch Kommata getrennt aufgezählt werden.

FILEOF type

Alle Komponenten sind gleichen Typs, der beliebig (außer FILE) sein darf. Eine Komponente wird über den File-Pointer ausgewählt, der nach jedem Zugriff auf das File auf die nächste Komponente zeigt. Der Pointer kann mit der SEEK-Prozedur auf eine bestimmte Komponente gestellt werden.

FILEOF TEXT

Eine Komponente entspricht einer Textzeile variabler Länge. Text-Files können nur sequentiell bearbeitet werden.

FILE

Nichttypisierte Datei. Alle Komponenten bestehen aus den 128 Bytes eines Disketten-Records. Statt READ- und WRITE-Prozeduren werden die BLOCKREAD/-WRITE-Prozeduren benutzt.

^type

Zeigertyp. Eine Variable vom Typ Zeiger kann auf eine dynamische Variable (im Heap) zeigen. Der Zeiger-Wert NIL zeigt ausdrücklich auf keine dyn. Variable und ist zu allen Zeigertypen kompatibel. Eine dyn. Variable wird durch dynvar^ ausgewählt, wobei dynvar^ vom Typ ^type ist.

Anweisungen

BEGIN<anw1;<anw2;<...>>>END;

Verbund-Anweisung

var:= ausdr;

Wertzuweisung

procname<(ausdr1<,ausdr2<...>>)>;

Ausführung der Prozedur procname. Die Anzahl der Parameter wird von der Prozedur-Deklaration festgelegt. Ist ein Parameter VAR-spezifiziert, darf der Ausdruck nur aus einer Variablen bestehen.

IF boolausdr THEN anw1 <ELSE anw2>;

Bedingte Anweisung

CASE scalausdr OF const1: anw1;<const2: anw2;<...>>
<ELSEanw3;<anw4;<...>>>END;

Fallunterscheidung

FOR scalvar:=ausdr1 <DOWN>TO ausdr2 DO anweisung;

Zählschleife. Nach Ausführung der anweisung wird der scalvar der Nachfolger/Vorgänger des derzeitigen Wertes zugewiesen. Ist der neue Wert größer/kleiner als der Wert des ausdr2, wird die Schleife beendet.

WHILE boolausdr DO <anweisung>;

Ist der boolausdr=TRUE, wird die Anweisung solange wiederholt, bis der boolausdr=FALSE ist.

REPEAT<anw1;<anw2;<...>>>UNTILboolausdr;

Die Anweisungen zwischen REPEAT und UNTIL werden mindestens einmal ausgeführt, bis der boolausdr=TRUE ist.

Operatoren

Die Operatoren sind in Gruppen gleicher Priorität eingeteilt. Die Gruppen sind nach fallender Priorität geordnet.

-	monadisches Minus (neg. Vorzeichen)	REAL,INTEGER
NOT	bitweise Negation	BOOLEAN,INTEGER
*	Multiplikation,Durchschnitt	SET,REAL,INTEGER
/	Division	REAL,INTEGER
DIV	Division	INTEGER
MOD	Modulus	INTEGER
AND	Arithmetisches, logisches 'und'	BOOLEAN,INTEGER
SHL	Linksschieben	INTEGER
SHR	Rechtsschieben	INTEGER
+	Addition	SET,STRING,REAL,INTEGER

-	Subtraktion	SET,REAL,INTEGER
OR	Arithmetisches, logisches 'oder'	BOOLEAN,INTEGER
XOR	Arith., logisches'exclusiv-oder'	BOOLEAN,INTEGER

Die Vergleichsoperatoren haben niedrigste Priorität.

=	ist gleich	skalare Typen,REAL
<>	ist ungleich	skalare Typen,REAL
<	ist kleiner als	skalare Typen,REAL
>	ist größer als	skalare Typen,REAL
<=	ist kleiner oder gleich	skalare Typen,REAL
>=	ist größer oder gleich	skalare Typen,REAL
IN	ist enthalten in	skalare Typen,REAL

String-Prozeduren und -Funktionen

DELETE(var,int1,int2)

Ab Position int1 in String var int2 Zeichen löschen. Die folgenden Zeichen int2 Positionen nach links verschieben.

INSERT(str,var,int)

Stringausdruck in die Variable ab Position int einfügen.

STR(real,var)

Numerischen Ausdruck in einen String umwandeln, der in der Stringvariable abgelegt wird. Diese Prozedur darf unter keinen Umständen (auch nicht mittelbar) in einer Funktion verwendet werden, die von einer WRITE-Prozedur aufgerufen wird.

VAL(str,var,err)

String in Zahl umwandeln. Tritt ein Syntaxfehler im Ausdruck auf, wird die Nr. des fehlerhaften Zeichens in err abgelegt, der Wert der Variable var ist undefiniert. Bei fehlerloser Umwandlung ist err=0. Diese Prozedur darf unter keinen Umständen (auch nicht mittelbar) in einer Funktion verwendet werden, die von einer WRITE-Prozedur aufgerufen wird.

COPY(str,int1,int2)

Ergibt die int2 Zeichen ab Position int1.

CONCAT(str1,str2<..,strx>)

Verbindet die Strings im Argument zu einem einzigen (entspricht der Stringaddition mit '+').

LENGTH(str)

Ergibt die Stringlänge, entspricht der Anzahl der Zeichen im String.

POS(str1,str2)

Ergibt die Position in str2, an der str1 das erste Mal auftritt. Tritt er nicht auf, ist POS=0.

Datei-Prozeduren und Funktionen

ASSIGN(fina,flle)

Der Datei file wird die Pascal-Dateivariable fina zugeordnet. file ist dabei ein String, der den Namen der Disketten-Datei beinhaltet.

REWRITE(fina)

Disketten-Datei fina anlegen. Eine schon vorhandene Datei fina wird überschrieben.

RESET(fina)

Datei öffnen, bzw. Zeiger auf Komponente 0 stellen.

FLUSH(fina)

Datei-Pufferspeicher-Inhalt auf Diskette speichern, wenn nach letztem Update in den Puffer geschrieben wurde. Danach Datei-Pufferspeicher löschen.

CLOSE(fina)

Datei schließen.

RENAME(fina,str)

Datei fina in str umbenennen.

ERASE(fina)

Datei löschen.

SEEK(fina,int)

Zeiger auf die Komponente Nr. int setzen.

READ(fina,varlist)

Die n Variablen der varlist mit den n nächsten Elementen der Datei fina laden.

READLN(fina,varlist)

Dto., danach wird der Zeiger hinter das nächstes CR/LF gestellt.

WRITE(fina,varlist)

Die Inhalte der Variablen der varlist werden in die Datei fina geschrieben.

WRITELN(fina,varlist)

Dto., CR/LF anfügen.

BLOCKREAD(fina,var,recs<,err>)

recs128-Byte-Records aus der Datei fina in die Variable var einlesen. In Variable err steht, wieviele Records tatsächlich übertragen wurden.

BLOCKWRITE(fina,var,recs<,err>)

recs128-Byte-Records aus der Variablen var in die Datei fina schreiben. In Variable err steht, wieviele Records tatsächlich übertragen wurden.

CHAIN(fina)

Aufruf des CHN-Programmes, das zuvor mit ASSIGN fina zugeordnet wurde. Ein CHN-Programm ist ein Programm, das keine eigene Turbo-Library besitzt und deshalb nicht vom CCP aus gestartet werden kann.

EXECUTE(fina)

Aufruf des COM-Programmes, das zuvor mit ASSIGN fina zugeordnet wurde.

EOF(fina)

Ergibt TRUE, wenn der Zeiger hinter der letzten Komponente der Datei fina steht.

SEEKEOF(fina)

dto., ergibt aber auch TRUE, wenn nur noch Leerzeichen, Tabs und CR/LF bis zum EOF folgen.

EOLN(fina)

ergibt TRUE, wenn der Zeiger auf einem CR oder EOF steht.

SEEKEOLN(fina)

Dto., ergibt aber auch TRUE, wenn nur Leerz. und Tabs bis zum nächsten CR oder EOF folgen.

FILEPOS(fina)

Ergibt die Nummer der aktuellen Komponente.

FILESIZE(fina)

Ergibt die Anzahl Komponenten.

Heap-Prozeduren und -Funktionen

NEW(pvar)

Pointer-Variable pvar erzeugen.

GETMEM(pvar,int)

Pointervariable pvar erzeugen, wobei für pvar im Heap int Bytes reserviert werden.

FREEMEM(pvar,int)

Mit GETMEM erzeugte pvar löschen, wobei int mit int in GETMEM identisch sein sollte. Innerhalb eines Programmes darf der Heap entweder nur mit DISPOSE oder nur mit MARK und RELEASE verwaltet werden.

DISPOSE(pvar)

pvar löschen, Heap-Platz für andere dyn. Variable frei.

MARK(pvar)

pvar markieren.

RELEASE(pvar)

Alle dyn. Variablen ab pvar löschen, wobei pvar vorher mit MARK markiert worden sein muß.

MEMAVAIL

Ergibt die Anzahl der freien Bytes im Heap.

MAXAVAIL

Ergibt die Größe des größten zusammenhängenden Heap-Bereiches in Bytes.

ORD(pvar)

Ergibt die Adresse auf die pvar zeigt.

PTR(int)

lädt einen Pointer mit der Adresse int.

Bildschirm-Prozeduren

CRTEXIT

Terminal-Reset-String zum Bildschirm senden.

CRTINIT

Terminal-Initialization-String zum Bildschirm senden.

CLREOL

Zeile rechts vom Cursor löschen.

CLRSCR

Bildschirm löschen, Cursor in Position 1,1 links oben bringen.

DELLINE

Zeile, in der der Cursor steht, löschen.

INSLINE

Zeile vor der, in der der Cursor steht, einfügen.

GOTOXY(x,y)

Cursor in Position x,y bringen.

READ(varlist)

Eingabe von Werten in Variablen von der Tastatur.

READLN(varlist)

wie READ, danach wird jedoch der nicht gelesene Rest im Tastaturpuffer gelöscht.

WRITE(varlist)

Ausgabe der Variablenwerte (nur CHAR, STRING, BOOLEAN, INTEGER, REAL).

WRITELN(varlist)

Wie WRITE, anschließend wird noch CR/LF an den Bildschirm gesandt.

Arithmetische Funktionen

ABS(real)	Betrag
ARCTAN(real)	Arcus Tangens
COS(real)	Cosinus
EXP(real)	Exponential-Funktion
FRAC(real)	nichtganzzahliger Anteil
INT(real)	rundet zur nächsten kleineren ganzen Zahl ab.
LN(real)	Logarithmus Naturalis

SIN(real)	Sinus
SQR(real)	Quadrat
SQRT(real)	Quadratwurzel

Skalare Funktionen

ODD(int)	TRUE, wenn int eine ungerade Zahl ist.
PRED(scal)	ergibt den Vorgänger von scal, wenn vorhanden.
SUCC(scal)	ergibt den Nachfolger von scal, wenn vorhanden.

Type-Wandlungs-Funktionen

CHR(int)	ergibt Char Nr. int
ORD(scal)	ergibt die Ordnungszahl von scal
ROUND(real)	rundet real zu einer Integerzahl
TRUNC(real)	ergibt den ganzzahligen Anteil
scalartype(scal1)	ergibt den Wert von Typ scalartype, der die gleiche Ordnungszahl hat, wie scal1, das von einem beliebigen anderen skalaren Typ sein darf.

Sonstige Prozeduren und Funktionen

BDOS(func<,para>)

BDOS-Funktion func aufrufen.

BIOS(func<,para>)

BIOS-Funktion func aufrufen.

DELAY(int)

Verzögerungsschleife, die etwa int Millisekunden dauert.

FILLCHAR(var,len,val)

ab Speicher-Adresse var len-mal val abspeichern.

HALT

Programmausführung beenden, zurück zum CCP.

MOVE(var1,var2,len)

Ab var1 len Bytes nach var2 übertragen, wobei var1,var2 Variablen beliebigen Typs sind.

RANDOMIZE

Zufallsgenerator initialisieren.

ADDR(name)

ergibt Adresse des ersten Bytes der Variable, Prozedur, Funktion, Type o.ä. name.

BDOS(func<,para>)

ergibt Rückgabewert der BDOS-Funktion func.

BDOSHL(func<,para>)

wie Funktion BDOS, jedoch Ergebnis in HL-Registerpaar.

BIOS(func<,para>)

ergibt Rückgabewert der BIOS-Funktion func.

BIOSHL(func<,para>)

wie Funktion BIOS, jedoch Ergebnis in HL-Registerpaar.

HI(int)

ergibt das höherwertige Byte von int.

LO(int)

ergibt das niederwertige Byte von int.

KEYPRESSED

ergibt true, wenn beliebige Taste gedrückt ist.

IORESULT

ergibt bei ausgeschalteter I-Option des Compilers Fehlernummer eines I/O-Fehler.

PARAMCOUNT

ergibt die Anzahl der vom aufrufenden Programm übergebenen Parameter im Befehlszeilenpuffer.

PARAMSTR(int)

ergibt den Parameterstring Nr. int

RANDOM(Int)

ergibt eine Zufallszahl (INTEGER) zwischen Null und int.

RANDOM

ergibt Zufallszahl (REAL) zwischen Null und Eins.

SIZEOF(name)

ergibt Anzahl der durch Variable oder Typ name belegten Bytes.

SWAP(int)

tauscht nieder- und höherwertiges Byte.

UPCASE(char)

ergibt char als Großbuchstaben.

Compiler-Befehle

Compilerbefehle werden durch (*$befehl*) aufgerufen. Im folgenden wird nur der Befehl aufgeführt. (Def) gibt die Voreinstellung (Default) an.

B+	(Def) CON: ist Std.-Dateien I. und O. zugewiesen.
B-	TRM: ist Standard-Dateien INPUT und OUTPUT zugewiesen.
C+	(Def) ^C und ^S sind wirksam.
C-	^C (Programmabbruch, wenn in READ<LN> eingegeben) und ^S (Bildschirmausgabe ein/aus) sind unwirksam.
I+	(Def) Ein-/Ausgabe-Fehler melden.
I-	Ein-/Ausgabe-Fehler müssen vom Programm selber mit der Standard-Funkton IORESULT überprüft werden.
Ifile	Die Datei file wird in die Compilierung aufgenommen. (Include-Datei)
R+	Indizierung wird geprüft.
R-	(Def) Indizierung von Feldern und Zuweisungen zu skalaren und Teilbereichs-Variablen werden nicht daraufhin überprüft, ob sie in den definierten Grenzen liegen.

V+ (Def) Prüft, ob VAR-Parameter-String korrekt.

V- Prüfung, ob aktueller String gleiche Länge hat, wie der formale String bei VAR-Parameter-Übergabe, wird nicht durchgeführt.

U+ Programm-Abbruch mit ^C jederzeit möglich.

U- (Def) Programm-Abbruch mit ^C nicht möglich. Programm-Ausführung wesentlich schneller als mit Abbruch-Option

A+ (Def) Erzeugt absoluten Code.

A- Erzeugt Code, der rekursive Aufrufe ermöglicht. Benötigt mehr Speicher, in der Ausführung langsamer.

Wn (Def:W2) Schachtelungs-Tiefe von WITH-Anweisungen. Erlaubt sind Tiefen von 1 bis 9.

X+ (Def) Array-Zugriff auf max. Geschwind. optimiert

X- Array-Zugriff auf minimale Codegröße optimiert.

Grafik-Prozeduren und -Funktionen (in GRAFIK.INC)

Die Prozeduren und Funktionen sind speziell für die CPC's entwickelt worden und somit nicht kompatibel zu anderen CP/M-Rechnern. Die meisten dieser Prozeduren und Funktionen entsprechen denen des gleichnamigen BASIC-Befehles.

INK(int,int1<,int2>)	Farbwahl für Stift int
PAPER(int)	Texthintergrundfarbstift wählen
PEN(int)	Textfarbstift wählen
GRAFPAPER(int)	Zeichenhintergrundfarbstift wählen
GRAFPEN(int)	Zeichenfarbstift wählen
CLG(pen)	Bildschirm mit pen überschreiben
CURSORON	Textcursor ein
CURSOROFF	Textcursor aus
MODE(int)	Bildschirmmodus festlegen
GRAFMODE(int)	Grafikmodus festlegen
ORIGIN(x,y)	Ursprung für absolute Bewegungen (z.B. DRAW)
GRAFMOVE(x,y)	Grafikcursor nach x,y (keine Linie)
GRAFMOVER(dx,dy)	Grafikcusor um dx,dy versetzen
PLOT(x,y)	Punkt an Position x,y
PLOTR(dx,dy)	Punkt an Position x+dx,y+dy
DRAW(x,y)	Linie zum Punkt x,y
DRAWR(dx,dy)	Linie von x,y nach x+dx,y+dy (relativ)
CIRCLE(x,y,r)	Kreis mit Mittelpunkt (x,y) und Radius r
FILLCIRCLE(x,y,r)	gefüllter Kreis
GRAFOUT(char)	Zeichenausgabe an Grafikcursorposition
TAG	lenkt Textausgabe an Grafikcursorposition
TAGOFF	schaltet TAG ab
GRAFDELAY(int)	Verlangsamung der Grafikausgabe

SPEEDKEY(int1,int2)	Wiederholrate der Tastatur
INITCALL	außer beim CPC6128 unbedingt vor eigener Firmwareroutine aufrufen!
GETMODE	ergibt Bildschirmmodus
TEST(x,y)	testet, ob Punkt x,y gesetzt
TESTR(dx,dy)	testet, ob Punkt x+dx,y+dy gesetzt
WHEREX	ergibt Spaltenposition des Textcursors
WHEREY	ergibt Zeilenposition des Textcursors
XPOS	ergibt x-Koordinate des Grafikcursors
YPOS	ergibt y-Koordinate des Grafikcursors
XVECTOR(grad,lang)	rel. x-Komponente des polaren Vektors
YVECTOR(grad,lang)	rel. y-Komponente des polaren Vektors
XCVECTOR(grad,lang)	wie XVECTOR, Rundungsfehler zu XERROR
YCVECTOR(grad,lang)	wie YVECTOR, Rundungsfehler zu YERROR

In neueren GRAFIK.INC-Versionen gibt es auch Sound-Befehle, die den gleichnamigen BASIC-Befehlen entsprechen.

Laufzeit-Fehlermeldungen

01	Gleitkommaüberlauf
02	Division durch Null versucht
03	SQRT-Argumentfehler (negatives Argument)
04	LN-Argumentfehler (Argument negativ oder Null)
10	Stringlängenfehler
11	ungültiger Stringindex bei COPY, DELETE oder INSERT

90	Index außerhalb des zulässigen Bereichs
91	Skalar oder Teilbereich außerhalb des zul. Bereichs
92	außerhalb des INTEGER-Bereichs
FF	Heap-Stack-Kollision

I/O-Fehlermeldungen bzw. IORESULT

00	kein Fehler
01	Datei nicht vorhanden
02	Lesen der Datei nicht möglich
03	Ausgabe in die Datei nicht möglich
04	Datei nicht offen
10	Fehler im numerischen Format
20	Operation auf logischem Gerät verboten
21	Operation im Direktmodus verboten
22	Zuordnung als Standarddatei verboten
90	Unpassende Recordlänge
91	EOF fehlt
99	unerwartetes EOF
F0	Diskettenschreibfehler (z.B. Diskette voll)
F1	Directory voll
F2	Dateigröße überschritten
FF	Datei verschwunden (Diskettenwechsel ?)

5.3 dBASE II

Datei-Struktur-Befehle

CREATE<nfile><FROMfile>

Richtet neue Datenbank ein.

Datentypen:

c	String (Zeichenkette)
n	numerisch (Fixpoint)
l	Boolean (T=true,wahr/ F=false,falsch)

COPY <bereich> TO nfile <STRUCTURE<EXTENDED>> <FIELD liste> <FORausdr> <SDF> <DELIMITED <WITH begrenz.-zeichen>>

Kopiert Daten aus der akt. Datenbank in die Datei nfile

DISPLAY STRUCTURE
LIST STRUCTURE

Anzeige der Struktur der aktuellen Datei.

MODIFY STRUCTURE

Veränderungen an der Dateistruktur vornehmen, löscht die Daten in dieser Datenbank.

Datei bearbeiten

COPY TO file

Kopiert die eröffnete Datenbank in die Datei file.

CLEAR

Alle Dateien schließen und alle temp. Variablen löschen.

DELETE FILE file

Datei file auf Diskette löschen.

DISPLAY FILES<ONl> <LIKE filemask>

Listet Dateien, die sich auf der Diskette in dem angemeldeten bzw. angegebenen Laufwerk befinden und der filemask genügen. Ohne LIKE werden alle Datenbanken gezeigt.

RESET

Zurücksetzen des Dateisystems in CP/M2.2 nach Diskettenwechsel.

QUIT <TOkommando1<,..,kommandox>>

Dateien schließen, dBase verlassen und Kommandos 1 bis x auf CCP-Ebene ausführen. Die Kommandos müssen in Anführungszeichen gesetzt werden.

RENAME file TO nfile

Datei file zu nfile umbenennen.

SELECT <PRIMARY><SECONDARY>

Arbeitsbereich umschalten.

USE <file <INDEX sfile liste>>

Alle Dateien schließen, Datei file und Index-Dateien der sfileliste öffnen.

Datenbanken verknüpfen, verlängern

APPEND <FROM file <FOR ausdr> <SDF> <DELIMITED WITH begrenz.-zeichen>> <BLANK>

Hängt Datensätze an die akt. Datei an.

UPDATE FROM file ON feldname <ADD feldfolge> <REPLACEfeldfolge>

Ersetzt Daten in der akt. Datei oder fügt Daten hinzu.

JOIN TO file FOR ausdr <FIELD feldfolge>

Verknüpft die Dateien der beiden Arbeitsfeldern zur neuen Datei file.

TOTAL ON indexfeld TO file<FIELD feldfolge> <FOR ausdr>

Summiert für gleichen Inhalt von indexfeld die Inhalte der übrigen numerischen Felder, Ergebnisse in der Datenbank file.

Editieren, Ändern

BROWSE

Anzeige der kompletten Datenbank, wobei alle Inhalte geändert werden können.

EDIT<n>

Ändern des Inhaltes des Datensatzes n.

DELETE <scope> <FORausdr>

Markiert zu löschende Datensätze.

RECALL <scope> <FORausdr>

Löscht die DELETE-Marke.

PACK

Löscht die Datensätze mit DELETE-Marke.

INSERT <BEFORE> <BLANK>

Datensatz einfügen.

REPLACE <scope> feldx WITH ausdrx <FORausdr>

Ersetzt Daten unter den angegebenen Voraussetzungen.

CHANGE <scope> FIELD feldfolge <FORausdr>

Ändern einzelner Datenfelder.

MODIFY COMMAND file

Befehlsdatei file bearbeiten.

Arbeiten mit Datenbanken

GO<TO> <ausdr> <TOP> <BOTTOM>

Datensatz auswählen.

SKIP <<+><->ausdr>

Datensätze überspringen.

FIND feld

Nach dem feld im Hauptschlüssel suchen. Wird feld nicht gefunden, ist die Nummer des aktuellen Datensatzes #=0.

LOCATE <scope> <FORausdr>

Datensatz, auf den der ausdr zutrifft, suchen. Gibt es keinen solchen, ist EOF=TRUE.

CONTINUE

Befehl LOCATE wiederholen.

COUNT <scope> <FORausdr> <TO var>

Zählt die Sätze, auf die ausdr zutrifft.

SUM ausdrlist TO varlist<scope> <FORausdr>

Summiert alle Elemente der ausdrlist über einen Bereich.

Sortieren

SORT ON feld TO nfile <DESCENDING>

Sortiert die akt. Datei nach dem feld in die neue Datei nfile. Bei DESCENDING wird der größte Wert an den Anfang der Datei nfile sortiert.

INDEX ON ausdr TO nfile

Erzeugt eine Indexdatei.

Eingabe- und Ausgabebefehle

INPUT <text> TO var

Eingabe. Strings müssen in Anführungszeichen geschrieben werden.

ACCEPT<text>TOvar

String-Eingabe. Strings werden ohne Anführungszeichen akzeptiert.

WAIT <TOvar>

Programmausführung bis zum nächsten Tastendruck unterbrechen, evtl. das gedrückte Zeichen in var speichern.

READ

Neue Werte für die vorher in GET-Klauseln vermerkten Variablen einlesen. Dabei werden nur Zeichen akzeptiert, die der jeweiligen Maske entsprechen.

CLEARGETS

Deaktiviert alle GET-Klauseln.

§x,y <SAY ausdr <USING mask1>> <GET var <PICTURE mask2>>

Cursor in Position x,y bringen, dort den ausdr und Inhalt der Variablen var unter Verwendung der jeweiligen Maske ausgeben, und var für das nächste READ vormerken.

?ausdr,ausdr1,...

Die Ausdrücke ausgeben.

DISPLAY <scope> <FORausdr> <ausdrfolge> <OFF>

Inhalte der spezifizierte Datensätze anzeigen, nach je 15 Zeilen unterbrechen. OFF unterdrückt die Ausgabe der Datensatznummern.

LIST

Syntax und Funktion wie DISPLAY, aber ohne Unterbrechung.

REPORT<FORMfile><scope><FORausdr><TOPRINT><PLAIN>

Datenbanken auswerten.

EJECT

Seitenvorschub ausgeben.

Variablen

DISPLAY MEMORY

Name, Inhalt und Typ aller Variablen ausgeben.

LIST MEMORY

Dto. (vgl. DISPLAY, LIST auf Seite 95).

RELEASE var

Variable löschen.

STORE ausdr TO var

Speichert den Wert des Ausdruckes in der Variablen.

RESTORE FROM file

Variablen aus dem file in den Speicher laden.

SAVE TO file

Variablen im Speicher im file speichern.

Programmbefehle

MODIFY COMMAND file

Befehlsdatei schreiben oder ändern.

DO file

Programm in der Befehlsdatei file ausführen.

RETURN

Rücksprung in die Ebene, die mit DO aufrief.

IF ausdr

```
IF ausdr
optionen1
<ELSEoptionen2>
ENDIF
```

Führt die optionen1 aus, wenn der ausdr wahr ist, bzw. die optionen2, wenn der ausdr falsch ist. ENDIF gibt das Ende des IF-Befehles an.

DOCASE

```
DOCASE
     CASE ausdr1
       optionen1
     ..
     ..
     CASE ausdrx
       optionenx
OTHERWISE
 optionen
ENDCASE
```

Fallunterscheidung. Die logischen Ausdrücke sind voneinander völlig unabhängig. Es wird allerdings nur eine Optionen-Gruppe ausgeführt; auch wenn nachfolgende Ausdrücke ebenfalls wahr sein würden.

DO WHILE ausdr

```
DO WHILE ausdr
 optionen
ENDDO
```

Optionen (Befehle zwischen DOWHILE und ENDDO) solange wiederholen, bis der ausdr falsch ist.

CANCEL

Abbruch des Programmes.

LOOP

Sprung in Do-While-Schleifen zum ENDDO.

PEEK(adr)

wie in Basic.

POKE

wie in Basic, jedoch Argumentliste zulässig.

CALL string

Maschinenprogrammroutine mit der mit SET gewählten Adresse anspringen, Pointer auf string in HL-Register.

SET-Befehle

SET parameter <ON><OFF>

in Klammern "Default":

ECHO(OFF)	ausgeführte Befehle ausgeben
STEP(OFF)	Einzelschrittmodus
TALK(ON)	Ergebnis von Befehlen anzeigen
PRINT(OFF)	Druckerprotokoll
CONSOLE(ON)	Anzeige auf Bildschirm
ALTERNATE(OFF)	Protokoll in Datei
SCREEN(ON)	Seitenmodus für div. Befehle
LINKAGE(OFF)	Ausführung gleichartig in prim/sek
COLON(ON)	Feldgrenzen bei GET-Eingabe markiert
BELL(ON)	Tonsignal bei Fehler auf Console
ESCAPE(ON)	ESC-Taste unterbricht Programm
EXACT(OFF)	Vergleichsmodus bei Strings
INTENSITY(ON)	unterschiedliche Helligkeit
DEBUG(OFF)	Ausgabe auf Drucker trennen
CARRY(OFF)	Daten vom vorherigen Satz übertragen
CONFIRM(OFF)	zum nächsten Feld nur auf Befehl

```
EJECT(ON)       REPORT-Text beginnt auf nächster Seite
RAW(OFF)        Einfügen von "" bei DISPLAY/LIST
```

SET HEADING TO text

Kopfzeile für REPORT definieren.

SET FORMAT TO

SCREEN	mit § erzeugte Ausgaben auf Bildschirm
PRINT	mit § erzeugte Ausgaben auf Drucker
file	file wird Maskendatei für READ

SET DEFAULT TO l

l wird Standardlaufwerk

SET ALTERNATE TO file

Protokoll-Datei

SET DATE TO mm/dd/yy

Datum setzen

SET INDEX TO filelist

Schlüsseldateien aktivieren

SET MARGIN TO n

Linker Rand in Spalte n bei REPORT-Ausgabe

SET CALL TO adr

Adresse für nächsten CALL

DISPLAY STATUS

zeigt den Zustand der SET-Parameter

5.4 SuperCalc

Allgemeine Kommandos

"	Text-Eintrags-Anfang
'	Text-Eintrags-Anfang (wiederholender Text)
/	Slash-Kommando einleiten
!	Werte aktualisieren (Recalculate)
=	Tabellen-Cursor zur angegebenen Zelle (GoTo)
;	Fenster wechseln (bei geteiltem Blatt)
^E	Tabellen-Cursor hoch
^S	Tabellen-Cursor links
^D	Tabellen-Cursor rechts
^X	Tabellen-Cursor ab
?	Hilfs-Funktion

Bereichs-Angaben

B3	Zelle 3 in Spalte B
3	ganze Zeile 3
B	ganze Spalte B
B3:F3	Teilzeile
B3:B7	Teilspalte
B3:F7	Block
ALL	gesamtes Blatt
BK254	unterste, ganz rechte Zelle

Editier-Funktionen bei Dateneinträgen

^S	Cursor links
^D	Cursor rechts
^E	Leerzeichen einfügen
^X	Zeichen löschen
^C od. ^Z	Eintrags-Zeile löschen
ESC	Tabellencursorbewegung ein/aus

Slash-Kommandos

Zeichen- und Kleinbuchstaben-Erklärung:

<.>	Angaben in eckigen Klammern sind optional
r	Row(s), Reihe(n)
c	Column(s), Spalte(n)
f	Filename, Dateiname
z	Zelle
b	Bereich
p.	Partial ..., Teil-...
s.	Source-..., Quell-...
d.	Destination-..., Zuweisungs-...
_	ENTER oder RETURN

Da jedes Kommando mit einem anderen Buchstaben des Alphabetes beginnt, braucht man nur den 1. Buchstaben einzugeben. Ein Beispiel für den Befehlsaufbau: /Cb,z_ bedeutet, daß der Bereich b in den Bereich kopiert werden soll, in dem z die linke obere Ecke darstellt. Die Eingabe eines Kommandos wird mit dem Slash "/" eingeleitet.

BLANK<b>_ ..... aktuelle Zelle oder Bereich löschen.

COPY ..... Zellen kopieren (Bereich b an Position z)

b,z_	adjust, automatische Anpassung
,A_	Anpassung erfragen
,N_	nicht an neue Zelle anpassen
,V_	Values only, nur Werte der Formeln übernehmen

DELETE ..... Löschen

Cc_	...Column, Spalte
Rr_	...Row, Reihe
Ff_	...File, Datei
F_	Disketten-Inhaltsverzeichnis untersuchen

EDIT<z>_ ..... Zelle in akt. Zelle kopieren und editieren

FORMAT ..... Formatieren

G,	Global, im ganzen Programm
Cc,	Column, nur Spalten
Rr,	Row, nur Reihen
Eb,	Entry, nur den Bereich
D_	Default, Werte ändern
E_	nur Exponentialdarstellung
G_	General, jeweils optimale Zahldarstellung
I_	Integer, statt Dezimalzahlen
L_	Zahlen linksbündig schreiben
R	Zahlen rechtsbündig schreiben
$_	Finanz-Format, mit 2 Nachkommastellen
TL_	Text linksbündig schreiben
TR_	Text rechtsbündig schreiben
*_	Balkendiagramm
x_	Spaltenbreite auf x Zeichen/Spalte setzen

GLOBAL ..... **allgemeine Festlegungen**

A_	Automatically recalculate, " nach jeder Änderung
B_	Border display on/off, Spalten- und Zeilen leiste
C_	Column calculation, Spaltenweise rechnen
F_	Formula display on/off, Formeln oder Ergeb nis
M_	Manual recalculate, Neuberechnung auf Befehl
N_	Next move on/off, Cursor autom. in nächste Zelle
R_	Row calculation, Reihenweise rechnen
T_	Tab, Cursor nur in ungeschützte, beschriebene Zellen

INSERT ..... **Einfügen**

Cc_	Column, Spalten...
Rr_	Row, Reihen...

LOAD .....**Laden**

_	Disketten-Inhaltsverzeichnis untersuchen
f,	filename, Dateiname
A_	All, alles
P	Part b,z; nur Bereich b nach z
_	adjust, automatische Anpassung
,A_	Anpassung erfragen
,N_	nicht anpassen an neue Zelle
,V_	Values only, nur Werte der Formeln übernehmen

MOVE ..... **Verschieben**

Rsr,dr_	...Row, ...Reihe sr in die Reihe dr
Csc,dc_	...Column, ...Spalte sc in die Spalte dc

OUTPUT ..... **Ausgabe**

D	Display, ...was auf dem Bildschirm steht
C	Content, ...Zellenweise mit Statusinformationen
b,	Bereich, der ausgegeben werden soll
D	Disc
_	Disketteninhaltsverzeichnis untersuchen
f,	filename, Dateiname
B_	Backup, Abspeichern und Sicherheitskopie anlegen
C_	Change name, Name ändern
O_	Overwrite, Abspeichern ohne Sicherheitskopie
C_	Console, Ausgabe an Bildschirm
P_	Printer, Ausgabe an Drucker
S_	Setup, Ausgabeparameter ändern
L_	Length, Zeilenzahl pro Seite
W_	Width, Zeichenzahl pro Zeile
P_	Print, Drucken, Setup beenden
S_	Setup, Druckerspezifische Funktionen ein/aus

PROTECT<b>_ ..... **Bereich bzw. aktive Zelle schützen**

QUIT ..... **Supercalc beenden?**

N_	No, SuperCalc fortsetzen
Y_	Yes, Rückkehr ins Betriebssystem

REPLICATE ..... **Vervielfachen**

sz,	Zelle sz in...
dz	...Zelle dz kopieren
pr	...alle Zellen der Teilreihe pr kopieren
pc	...alle Zellen der Teilspalte pc kopieren
pr,	Teilreihe pr in ...
pc,	Teilspalte pc in ...
pc	... die Teilspalten pc kopieren
pr	... die Teilreihen pr kopieren
_	adjust, automatische Anpassung
,A_	Anpassung erfragen
,N_	nicht anpassen an neue Zelle
,V_	Values only, nur Werte der Formeln über nehmen

SAVE.....auf Diskette speichern

_	Disketten-Inhaltsverzeichnis untersuchen
f,	filename, Dateiname
B	Backup, Abspeichern und Sicherheitskopie an legen
C	Change name, Name ändern
O	Overwrite, Abspeichern ohne Sicherheitskopie
A_	All, Zelleninhalt,-Wert,Anz.-Format,etc.
V_	Values only, nur Ergebnisse und Anzeige format
P	Part, nur einen Teil abspeichern
A	All, wie oben
V	Values, wie oben
b_	Bereich, der abgespeichert werden soll

TITLE ..... Überschriften

B_	Both
C_	Clear
H_	Horizontal lock,
V_	Vertikal lock,

UNPROTECT<b>_ .Schutz der Zelle/des Bereich aufheben

WINDOW .....Bildschirm-Fenster

C_	Clear, 2. Fenster löschen
H_	Horizontal split, 2 Fenster untereinander
S_	Synchronize, Scroll in beiden Fenstern zugleich
U_	Unsynchronize, Fenster scrollen einzeln
V_	Vertikal split, 2 Fenster nebeneinander

XECUTE.....Startet ein .XQT-Programmes

f_	Startet das .XQT-Programm f
_	Disketteninhaltsverzeichnis untersuchen

ZAP..... alle Zelleninhalte (auch geschützte) löschen

Y_	Yes, Bestätigung
N_	No, Löschung wird doch nicht vorgenommen

Optionen zu "Diskettenverzeichnis untersuchen" .....

C	Choose drive, Laufwerk auswählen
D	Disc directory, entspricht CP/M-Kommando DIR
S	SuperCalc-Files, wie D, aber nur SuperCalc-Files

Formel-Einträge

Arithmetische und vergleichende Operatoren:

`+ - * / ^ oder ** = <> < > <= >=`

Logische Funktionen

IF(ausdr1,ausdr2,ausdr3)

Wenn ausdr1 wahr, dann ausdr2, ansonsten ausdr3 berechnen

OR(ausdr1,ausdr2)

Logisches ODER

AND(ausdr1,ausdr2)

Logisches UND, ergibt bei wahr 1, sonst 0.

NOT(ausdr)

Negation

Arithmetische Funktionen

z Zelle oder Ausdruck
b Zellenbereich oder -liste

Funktion	Wirkung
ABS(z)	Betrag von z
ACOS(z)	arccos von z, Ergebnis in Bogenmaß
ASIN(z)	arcsin von z, Ergebnis in Bogenmaß
ATAN(z)	arctan von z, Ergebnis in Bogenmaß
AVERAGE(b)	arithmetischer Mittelwert
COS(z)	cos von z in Bogenmaß
COUNT(b)	Anzahl der numerischen Elemente in b
EXP(z)	Potenz zur Basis e
INT(z)	Integer-Funktion
LN(z)	natürlicher Logarithmus (zur Basis e)
LOG10(z)	Dezimallogarithmus (zur Basis 10)
MAX(b)	Maximalwert in b
MIN(b)	Minimalwert in b
PI	Konstante 3.141592653...
SIN(z)	sin von z in Bogenmaß
SQRT(z)	Quadratwurzel von z
SUM(b)	Summe aller numerischen Zellenwerte in b
TAN(z)	tan von z in Bogenmaß

5.5 Multiplan (deutsch)

Zellenadressierung

x und y können relativ zur adressierenden Zelle angegeben werden, indem sie in rechteckige Klammern gesetzt werden.

Einzelne Zelle

ZxSy Zelle in der x. Zeile und y. Spalte

Zx S Zelle in der x. Zeile der gleichen Spalte, in der die Zelle adressiert wird.

Z Sy Zelle in der y. Spalte der gleichen Zeile, in der die Zelle adressiert wird.

Mehrere Zellen

Zx1:x2Sy1:y2 Alle Zellen in den Zeilen x1 bis x2, die die Spaltennummern y1 bis y2 besitzen.

Zx1:x2 S Alle Zellen der Zeilen x1 bis x2 der gleichen Spalte, in der der Bereich adressiert wird.

Z Sy1:y2 Alle Zellen der Spalten y1 bis y2 der gleichen Zeile, in der der Bereich adressiert wird.

AUSSCHNITT	Unterteilung des Bildschirmes zur Betrachtung mehrerer Ausschnitte aus dem Arbeitsblatt. (max. 8 Fenster)
.TEILEN	Bildschirm unterteilen.
.WAAGERECHT	Ausschnitte untereinander.
.SENKRECHT	Ausschnitte nebeneinander.
..BEZEICHNUNG	Ausschnitt so, daß die Überschriften links und oben sichtbar bleiben können, wenn der Cursor im Ausschnitt bewegt wird.
.UMRAHMEN	ein Fenster umrahmen bzw. Rahmen löschen.
LÖSCHEN	ein Fenster schließen.
VERBINDEN	Fenster miteinander synchronisieren.
BEWEGEN	... verschieben.
.SPALTEN	Spalten ...
..VON SPALTE	erste zu verschiebende Spalte
...BIS VOR SPALTE	Ziel der ersten Spalte
....SPALTENANZAHL	Anzahl der zu verschiebenden Spalten
.ZEILEN	Zeilen ...
..VON ZEILE	erste zu verschiebende Zeile
...BIS VOR ZEILE	Ziel der ersten Zeile
....ZEILENANZAHL	Anzahl der zu verschiebenden Zeilen

DRUCK	Ausgabe an ...
.DRUCKER	... Drucker
.PLATTE/DISKETTE	... Disk-Laufwerk
.RANDBEGRENZUNG	Druckformat festlegen
..LINKS	Anzahl Zeichen des linken Randes
..OBEN	Anzahl Zeilen des oberen Randes
..DRUCKBREITE	Anzahl Schriftzeichen je Zeile
..DRUCKLÄNGE	Anzahl Schriftzeilen je Seite
..SEITENLÄNGE	Papierlänge in Anzahl Zeilen je Seite
.OPTIONEN	
..BEREICH	zu druckender Zellenbereich
..STEUERZEICHEN	Druckersteuerzeichen
..FORMULN	Formeln ausdrucken
..Z/S-NUMMERN	auch Zeilen- und Spaltennummern drucken
EINFÜGEN	Spalten oder Zeilen einfügen.
FORMAT	... formatieren.
.FELDER	Zellenbereich ...
..AUSRICHTUNG	Positionierung der Feldinhalte
...STND	lt. FORMAT DEFAULT CELLS-Festlegung
...MITTE	zentriert in Feldmitte
...NORM	Text links-, Zahlen rechtsbündig
...LINKS	Text und Zahlen linksbündig
...RECHTS	Text und Zahlen rechtsbündig
..FORMATCODE	Text- und Zahlenformat
...STND	lt. FORMAT DEFAULT CELLS-Festlegung
...ZUSAMM	In aufeinanderfolgenden CONT-formatierten Feldern können Texte die Feldgrenzen überschreiten.
...E_FORM	Zahlen exponentiell darstellen
...FEST	Zahlen mit Festkomma darstellen
....DEZ-STELLEN	Anzahl Nachkommastellen
...NORM	Zahlen in angemessenem Format darstellen.

...GANZ	Zahlen ohne Nachkommastellen darstellen.
...DM	Zahlen mit 2 Nachkommastellen und nachfolgendem "DM" darstellen. Negative Zahlen stehen in runden Klammern.
...%	Hundertfaches der Zahl mit nachfolgendem Prozentzeichen darstellen.
*	Zahlen als Balkengraphik darstellen.
-	Altes Format beibehalten.
STANDARD	Standards festlegen
..FELDER	Alle Formate (außer DEF) wie unter FORMAT FELDER können gewählt werden.
..BREITE DER SPALTEN	Zellenbreite bestimmen (3 ... 32)
OPTIONEN	Darstellungsart wählen
..TAUSENDERPUNKTE	Zahlen mit Kommas in Tausender-Gruppen unterteilen.
..FORMELN	Formeln oder Ergebnisse anzeigen.
.BREITE DER SPALTEN	Zellenbreite einer Anzahl von Spalten bestimmen.
GEHEZU	Cursor zu einer bestimmten Zelle bewegen
.ZEILE SPALTE	Zelle über ihre Koordinaten auswählen
.NAME	springt die erste Zelle dieses Namens an.
.AUSSCHNITT	Fenster wechseln
..ZEILE SPALTE	dortige Zeile und Spalte wählen

HILFE	Erklärungen zu Multiplan. Statt HILFE aufzurufen, kann man, um sich bei einem bestimmten Befehl helfen zu lassen, auch den Cursor auf den betreffenden Befehl stellen und ein Fragezeichen eingeben.
KOPIEREN	... kopieren.
.NACH_UNTEN	in die Nachbarzellen nach unten ...
.RECHTS	in die Nachbarzellen nach rechts ...
.VON	beliebigen Bereich an bel. Stelle ...
LOESCHEN	Zeilen oder Spalten löschen.
NAME	Zellenbereich mit Name versehen.
ORDNEN	sortieren
.DER SPALTE	Spalte nach deren Inhalten sortiert werden soll.
.VON ZEILE	erste zu sortierende Zeile
.BIS ZEILE	letzte zu sortierende Zeile
....ORDER	auf- oder absteigend sortieren. Aufsteigend wird wie folgt sortiert : 1. Zahlen 2. Texte 3. Logische Werte, Fehlermeldungen 4. leere Zellen
QUIT	Multiplan-Lauf beenden. Es findet keine automatische Datensicherung statt.
RADIEREN	Anzugebenden Bereich oder aktuelle Zelle löschen.
SCHUTZ	Überschreib-Schutz
.FELDER	Zellenbereich schützen
.RECHENFORMELN	Alle Formel- und Textzellen schützen.
TEXT	Texteingabe in die aktuelle Zelle.

UEBERTRAGEN	Arbeitsblatt ...
.LADEN	... von Diskette laden.
.SPEICHERN	... auf Diskette speichern.
.BILDSCHIRMLÖSCHEN	... im Arbeitsspeicher löschen.
.DATEILÖSCHEN	... auf Diskette löschen.
.OPTIONEN	Dateiformat an andere DV-Programme anpassen
.UMBENENNEN	... mit neuem Namen auf Diskette speichern. Die Umbenennung wird auch in den mit diesem Arbeitsblatt verknüpften Blättern durchgeführt.
VERÄNDERN	Inhalt der aktuellen Zelle ändern.
WERT	Formeln und Zahlen in die aktuelle Zelle eingeben. Zahlen können auch direkt aus dem HauptMenü heraus eingegeben werden.
XTERN	Arbeitsblätter verknüpfen.
.KOPIE	Definieren, welche Texte oder Werte von einem anderen Arbeitsblatt übernommen werden sollen.
.LISTE	Listen aller Arbeitsblätter aus denen das aktuelle Daten bezieht oder die auf Daten des aktuellen Arbeitsblattes zurückgreifen.
.UMBENENNEN	Arbeitsblatt in einem System von verknüpften Arbeitsblättern durch ein anderes ersetzen.
ZUSÄTZE	Verarbeitungsmodi wählen.
.SOFORT RECHNEN	Nach jeder Eingabe Tabelle nachrechnen.
.ALARM AUS	Akustische Fehlermeldung.
.ITERATION	Iterative Tabellenberechnung
.ENDKRITERIUM	Zelle mit der Abbruchbedingung

Operatoren

+	Addition. Ein in einer Formel stehender Zellenbereich wird als Summe interpretiert.
-	Subtraktion, negatives Vorzeichen.
*	Multiplikation.
%	Multiplikation mit 100.
/	Division.
^	Potenzieren.
&	Textaddition.

Funktionen

ausdr	Ausdruck allgemein
dec	Anzahl Nachkommastellen (Dezimalen)
val	numerischer Ausdruck
tex	Textausdruck
log	logischer Ausdruck
zeb	Zellenbereichsangabe
zel	Zellenangabe
-list	Liste durch Komma getrennter Ausdruecke oder Angaben.

ABS(val)

Betrag.

ANZAHL(zlist)

Ergibt die Anzahl der Felder der zlist, die numerischen Inhalt haben.

ARCTAN(val)

Arcus Tangens.

BARWERT(val;vlist)

Gegenwartswert erwarteter Rückflüsse vlist bei einer Verzinsung von val.

COS(val)

Cosinus.

DELTA()

Endbedingung für Iteration. Weicht das Ergebnis einer Berechnung um weniger als DELTA() von seinem Vorgänger ab, wird die Iteration beendet.

DMARK(val;dec)

Das Ergebnis des val wird mit zwei Dezimalen (wenn durch dec nicht anders verlangt) und einem nachfolgenden "DM" angezeigt. Negative Zahlen werden in Klammern gesetzt.

EXP(val)

e-Funktion.

FALSCH()

Logischer Wert FALSCH.

FEST(val;dec)

val in Text mit dec Dezimalen umwandeln.

GANZZAHL(val)

Ergibt den ganzzahligen Anteil von val.

INDEX(zeb;x;y)

Ergibt den Wert der x. Zeile in der y. Spalte des Zellenbereiches zeb.

ISTFEHL(zel)

WAHR, wenn in Feld zel eine Fehlermeldung steht.

ISTNV(zel)

WAHR, wenn in FEld zel "NV" steht.

LÄNGE(tex)

Ergibt die Zeichenanzahl des tex.

LN(val)

logarithmus naturalis. Log. zu Basis e.

LOG10(val)

Logarithmus zur Basis 10.

MAX(vlist)

Ergibt den größten in der Liste vorkommenden Wert.

MIN(vlist)

Ergibt den kleinsten in der Liste vorkommenden Wert.

MITTELW(vlist)

arithmetischer Mittelwert.

NICHT(log)

WAHR, wenn log=FALSCH ist.

NV()

Ungültigkeitsfunktion. Ergibt "NV".

ODER(llist)

WAHR, wenn mindestens ein Wert der Liste WAHR ist.

PI()

Naturkonstante = 3.141592653

REST(val;val1)

Ergibt den Rest der Division val/val1.

RUNDEN(val;dec)

Rundet auf dec Dezimalen. Ist dec negativ, wird schon -dec Stellen vor dem Komma gerundet.

SIN(val)

Sinus.

SPALTE()

Spaltennummer in der die Funktion steht.

STABW(vlist)

Standardabweichung.

SUCHEN(val;zeb)

Sucht in der ersten Spalte des zeb nach dem Wert val (bzw. dem letzten, der kleiner als val ist, wenn val selbst nicht in der Spalte steht) und ergibt dann den Inhalt der letzten Spalte der gleichen Zeile.

SUMME(vlist)

Ergibt die Summe der Werte in der Liste.

TAN(val)

Tangens.

TEIL(tex;val;val1)

Ergibt val1 Zeichen aus dem String tex ab dem val. Zeichen.

UND(llist)

WAHR, wenn alle Werte der Liste WAHR sind.

VORZEICHEN(val)

Ergibt +1,-1 oder 0, wenn val grÖßer, kleiner oder gleich Null ist.

WAHR()

Logischer Wert WAHR.

WENN(log,ausdr,ausdr1)

Verzweigungsfunktion. Ist log=WAHR, wird der ausdr berechnet, ist log=FALSCH, wird der ausdr1 berechnet.

WERT(tex)

tex in numerischen Wert umwandeln.

WIEDERHOLEN(tex,val)

tex val mal wiederholen.

WURZEL(val)

Quadratwurzel.

ZÄHLER()

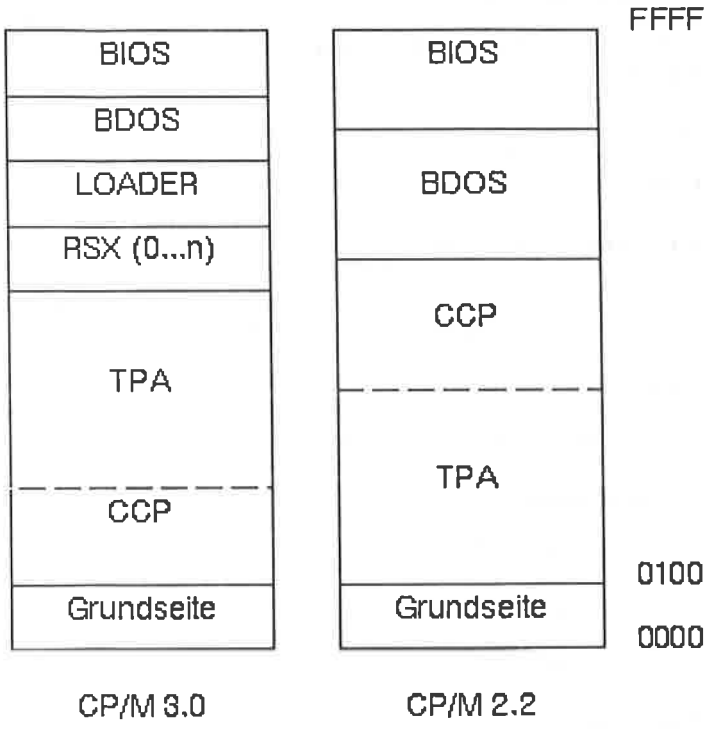
Iterationsschleifenzähler.

ZEILE()

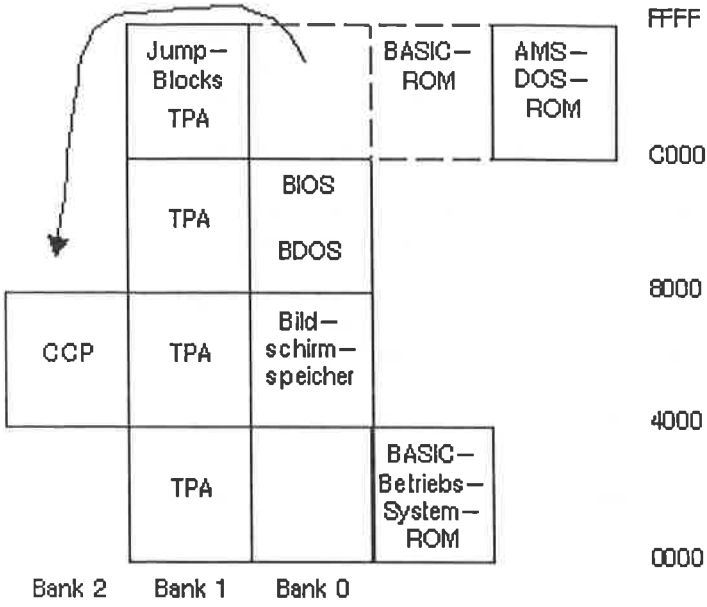
Zeilennummer in der die Funktion steht.

6. CP/M-Systemebene

6.1 Speicheraufbau



Logische Speicherverteilung unter CP/M



CP/M3.0 Speicherverteilung im CPC6128

6.2 Zeropage

Adresse(n)		Inhalt
0000 - 0002		JP WBOOT, Sprung zum BIOS-Warmstart
0003	*	IOBYTE
0004	*	Bezugslaufwerk und Benutzernummer
0005 - 0007		JP BDOS, Sprung ins BDOS oder letzten vorgelagerten RSX. Die Adressen 0006 und 0007 enthalten die erste dem TPA nicht mehr zur Verfügung stehende Speicher adresse.
0008 - 003A		ReSTart-Adressen für RST 1...RST 7
003B - 004F		reserviert
0050	+	Laufwerks-Nummer-Angabe bei Aufruf eines Programmes von Diskette
0051 - 0052	+	Adresse des Paßwortes der ersten im Befehl aufgeführten Datei im DMA-Puffer ab 0080 (adr=0, wenn kein Paßwort)
0053	+	Länge des Paßworts (len=0, wenn kein Paßwort)
0054 - 0056	+	dto. für 2. Datei
0057 - 005B		reserviert
005C - 007F		FCB der 1. Datei
006C - 007F		FCB der 2.Datei (,falls vorhanden. Beachte: Benutzt die zweiten 16 Bytes des 1. FCB.)
0080 - 00FF		Standart-DMA-Puffer und CCP-Befehls-Argumentspeicher
	*	nur CP/M2.2
	+	nur CP/M3.0

6.3 Kontroll- und Parameter-Blöcke

File-Control-Block FCB (Dateibeschreiber)

Die Disketten-Directories enthalten nur die Bytes 0 bis 31. In Disketten-Directories enthält das Byte 0 die User-Nummer.

Byte(s)	Name(n)	Inhalt
00	dr	Laufwerks-Nr., 0=angemeldetes Laufwerk
01 - 08	f1 - f8	Dateiname, ASCII-Code, Großbuchstaben
09 - 0B	t1 - t3	Typ-Name, ASCII-Code, Großbuchstaben
0C	ex	Extend-Nr., Eintrags-Nr.
0D - 0E	s1 - s2	reserviert für das System
0F	rc	Anzahl der Records im aktuellen Eintrag
10 - 1F	d0 - df	reserviert für das System
20	cr	relative Nummer des nächsten Blockes
21 - 23	r0 - r2	absolute Block-Nr. für direkten Zugriff

Extended File-Control-Block XFCB (CP/M3.0)

Der XFCB ist ein spezieller FCB, der Passwort und Datum verwaltet. Der XFCB wird nur auszugsweise auf Diskette geschrieben. Ist eine Diskette mit INITDIR.COM auf Datum- und Paßwort-Einträge vorbereitet worden, werden in jedem Directory-Record 3 FCBs verwaltet. In den verbleibenden 32 Bytes (dem SFCB) werden die Datums-Einträge der FCBs abgelegt.

Byte(s)	Name(n)	XFCB-Inhalt
00	dr	Laufwerks-Nr., 0=angemeldetes Laufwerk
01 - 08	f1 - f8	Dateiname, ASCII-Code, Großbuchstaben
09 - 0B	t1 - t3	Typ-Name, ASCII-Code, Großbuchstaben
0C	pm	password-mode, Schutz durch Kennwort bit 7 Schutz vor Lesen bit 6 Schutz vor Beschreiben bit 5 Schutz vor Löschen
0D - 0F	s1,s2,rc	reserviert für das System
10 - 17	password	Schutz-Kennwort
18 - 1F		reserviert für das System

Byte(s)	SFCB-Inhalt
00	immer 21H
01 - 0A	Erweiterung des 1. FCB
0B - 14	Erweiterung des 2. FCB
15 - 1E	Erweiterung des 3. FCB
1F	reserviert

Byte(s)	Erweiterung eines FCB
00 - 03	Erzeugungs- oder Zugriffs-Datum
04 - 07	Update-Datum
08	Paßwort-Modus
09	reserviert

Directory-Label (CP/M3.0)

Spezieller FCB, der durch das Programm INITDIR.COM ins Directory geschrieben wird und Paßwort- und Datum-Modi kontrolliert. Mit SET.COM können Einträge im Directory-Label vorgenommen bzw. geändert werden.

Byte(s)	Name(n)	Directory-Label-Inhalt
00	dr	Laufwerks-Nr., 0=angemeldetes Laufwerk
01 - 08	f1 - f8	Diskettenname, ASCII-Code, Großbuchstab.
09 - 0B	t1 - t3	Diskettentyp, ASCII-Code, Großbuchstaben
0C	dl	Directory-Label-Daten-Byte bit 7 Paßwortschutz für geschützte Dateien eingeschaltet bit 6 Zugriffsdatum im XFCB speichern bit 5 Updatedatum im XFCB speichern bit 4 Erzeugungsdatum im XFCB speichern bit 0 Directory-Label existiert
0D - 0F	s1,s2,rc	reserviert für das System
10 - 17	password	Schutz-Kennwort der Diskette
18 - 1B	ts1	Erzeugungsdatum
1C - 1F		Updatedatum

Disketten-Parameter-Block DPB

Byte(s)	Name	Inhalt
00 - 01	spt	Anzahl Records (=128 Bytes) je Spur
02	bsh	Blockverschiebungsfaktor
03	blm	Blockmaske
04	exm	Eintragsmaske
05 - 06	dsm	Anzahl Blöcke - 1 ohne Systemspurenblöcke
07 - 08	drm	Anzahl der mgl. Directory-Einträge - 1
09 - 0A	al0/1	Directory-Block-Belegungs-Maske, z.B. sind normalerweise Block 0 und 1 vom Directory belegt: al0/1 enthalten dann C000
0B - 0C	cks	Anzahl der auf Disk.-Wechsel zu prüfenden Directory-Records
0D - 0E	off	Anzahl der Systemspuren
0F	psh	Sektorverschiebungsfaktor (CP/M3.0)
10	phm	Sektormaske (CP/M3.0)

Blockgröße	bsh	blm	exm1	exm2	Sektorgröße	psh	phm
1k	3	7	n.d.	0	128	0	0
2k	4	15	0	1	256	1	1
4k	5	31	1	3	512	2	3
8k	6	63	3	7	1024	3	7
16k	7	127	7	15	2048	4	15
(exm1-Spalte, wenn dsm < 256)					4096	5	31

Disk-Parameter-Header DPH

Bytes	Inhalt
00,01	Adresse d. Übersetzungstabelle f. d. Record-Nummern
02,03	Spur-Nummer (BDOS)
04,05	Sektor-Nummer (BDOS)
06,07	Record-Nummer im Verzeichnis (BDOS)
08,09	Adresse des 128 Bytes langen Verzeichnispuffers
0A,0B	Adresse des Diskettenbeschreibers
0C,0D	Adresse der Prüfsummentabelle
0E,0F	Adresse der Belegungstabelle

System-Control-Block SCB (CP/M3.0)

Byte(s)	Inhalt
00 - 04	reserviert
05	BDOS-Version-Nummer
06 - 09	für Benutzer frei
0A - 0F	reserviert
10 - 11	Fehler-Code
12 - 19	reserviert
1A - 1C	Konsole-Spalten, aktuelle -Spalte, -Zeilen/Seite
1D - 21	reserviert
22 - 23	CONIN Redirection flag, Bit 7=0, dann keine
24 - 25	CONOUT dto.
26 - 27	AUXIN dto.
28 - 29	AUXOUT dto.

2A - 2B	LSTOUT dto.
2C	Seiten-Modus
2D	reserviert
2E - 2F	^H aktiv (CLR), Rubout aktiv (DEL)
30 - 32	reserviert
33 - 34	Konsole-Modus
35 - 36	reserviert
37	Output-Delimiter, Ausgabe-Ende-Zeichen
38	List-Output-Flag
39 - 3B	reserviert
3C - 3D	aktuelle DMA-Adresse
3E	Nummer des angemeldeten Laufwerkes
3F - 43	reserviert
44	aktuelle User-Nummer
45 - 49	reserviert
4A	Multi-Sektor-Zahl
4B	Fehler-Modus
4C - 4F	Laufwerke-Such-Kette
50 - 51	aktuelle Laufwerks-Nummer, Fehler-Laufwerk
52 - 56	reserviert
57	BDOS-Flags
58 - 5C	Datum
5D - 5E	Basis-Adresse des gemeinsamen Speicherbereiches
5F - 63	reserviert

6.4 BDOS-Funktionen

Die BDOS-Funktionsnummer wird in Register C übergeben. Eingaben müssen im Register DE (Byte-Daten nur in E) stehen. Byte-Ausgaben stehen im Akkumulator, Wort-Ausgaben in HL. Jede Funktionsbeschreibung besteht aus 1 bis 3 Teilen:

1. Funktionsnummer, Funktionsname, Funktionsbeschreibung
2. Eingaben
3. Ausgaben

Fehlermeldungen in Register A

Code	Directory-Code (Dir-Code)
00	erfolgreich
01	erfolgreich (nur BDOS-Funktionen 17 und 18)
02	erfolgreich (nur BDOS-Funktionen 17 und 18)
03	erfolgreich (nur BDOS-Funktionen 17 und 18)
FF	Fehler gefunden

Code	Fehlermeldung (Fehler)
00	Funktion erfolgreich durchgeführt
01	Directory voll od. gesuchte Daten nicht vorhanden
02	kein Block frei
03	Extent kann nicht geschlossen werden
04	gesuchter Extent nicht vorhanden
05	Directory voll
06	Fehler in der Record-Nummer für Direktzugriff
09	ungültiger FCB
0A	Diskettenwechsel bei geöffneter Datei
FF	physikalischer Fehler, siehe Register H (CP/M3.0)

Physikalische Fehler (CP/M3.0)

Mit der Funktion 45 können 3 verschiedene Möglichkeiten auf einen physikalischen Fehler zu reagieren ausgewählt werden:

1. Ausgabe einer Fehlermeldung auf dem Bildschirm und BIOS-Warmstart.

2. Rückgabe eines Fehlercodes in Register H, wobei Register A den Wert FF besitzt.

3. Ausgabe einer Fehlermeldung auf dem Bildschirm und Rückgabe eines Fehlercodes in Register H, wobei in Register A der Wert FF steht.

Code	Fehler-Meldung in Register H
00	kein Fehler od. zumindest Register H nicht zuständig
01	Disketten-Ein-/Ausgabe-Fehler
02	Diskette ist schreibgeschützt
03	Datei schreibgeschützt, oder durch Paßwort schreibgeschützt (Paßwort richtig) oder von anderem User geöffnet.
04	Laufwerksangabe falsch
07	Paßwort fehlt oder falsch
08	Datei existiert
09	Joker im Dateiname

Übersicht über alle CP/M-BDOS-Funktionen

Auf der nächsten Seite finden Sie eine Liste aller BDOS-Funktionen in numerischer Reihenfolge, die von CP/M2.2 und CP/M3.0 unterstützt werden. In der Tabelle ist die Verfügbarkeit der Funktionen unter den verschiedenen Systemen gekennzeichnet. In den folgenden Abschnitten, die die Funktionen näher erläutern, erfolgt dies nicht mehr.

0		System-Reset
1		Konsoleingabe
2		Konsolausgabe
3	*	Lochstreifenleser
	+	Hilfseingang
4	*	Lochstreifenstanzer
	+	Hilfsausgang
5		Druckerausgang
6		direkte Konsol-I/O
7	*	IOBYTE lesen
	+	Status Hilfseingang
8	*	IOBYTE beschreiben
	+	Status Hilfsausgang
9		Stringausgabe
10		Stringeingabe
11		Konsolstatus
12		Versionsnummer
13		Disk-System-Reset
14		Laufwerk auswählen
15		Datei öffnen
16		Datei schließen
17		ersten FCB suchen
18		weitersuchen
19		Datei löschen
20		sequentiell lesen
21		sequentiell schreiben
22		Datei erzeugen
23		Datei umbenennen
24		Login-Vektor lesen
25		Bez.-Laufw. ermitteln

26		DMA-Adresse festlegen
27		Adr. der Diskbelegung
28		Disk schreibschützen
29		R/O-Vektor holen
30		Dateiattribute setzen
31		Adr. des DPB holen
32		Benutzernummer
33		direkt lesen
34		direkt schreiben
35		Dateigröße berechnen
36		nächste Recordnummer
37		Laufwerks-Reset
40		Rec. reset, schreiben
44	+	Multisektorpreset
45	+	BDOS-Fehler-Modus
46	+	freie Disk-Records
47	+	Programmverkettung
48	+	DMA-Puffer ausleeren
49	+	SCB lesen/beschreiben
50	+	BIOS-Funktionsaufruf
59	+	Overlay laden
60	+	RSX-Aufruf
98	+	Blöcke freisetzen
99	+	Datei verkürzen
100	+	Disk-Label schreiben
101	+	Disk-Label lesen
102	+	XFCB lesen
103	+	XFCB schreiben
104	+	Systemuhr stellen
105	+	Systemuhr lesen
106	+	Standard-Paßwort
107	+	Seriennummer lesen
108	+	Prog.-Antwort-Code
109	+	Konsol-Modus
110	+	Stringbegrenzer
111	+	String an CONOUT:
112	+	String an LST:
152	+	FCB vorbereiten

0 — System-Reset

Unter CP/M3.0 wird das Disk-System nicht mit zurückgesetzt.

45 — BDOS-Fehler-Modus wählen.

Reg.E	Fehler-Modus
00-FD	Fehlermeldung und BIOS-Warmstart
FE	Fehlermeldung und Fehlercode in Register H
FF	Fehlercode in Register H

Eingabe: E=Fehler-Modus

49 — System-Control-Block-Daten lesen/beschreiben

Byte(s)	Inhalt des SCB-Parameter-Blockes (SCB-PB)
00	Offset im SCB
01	00=lesen, FE=Wort bzw. FF=Byte schreiben
<02<-03>>	zu schreibender Wert

Eingabe: DE=adr(SCB PB)

104 — Datum und Zeit schreiben

Byte(s)	Datumblock
00 - 01	Datum = Tage seit dem 1.1.1978
02	Stunden = 2 BCD-Ziffern
03	Minuten = 2 BCD-Ziffern

Eingabe: DE=adr(Datumblock)

105 — Datum und Zeit lesen.

Aufbau des Datumblockes wie bei Funktion 104.

Eingabe: DE=adr(Datumblock)
Ausgabe: A=Sekunden,Datum und Uhrzeit im Datumblock

106 Standard-Paßwort schreiben.

Zugriff auf eine paßwortgeschützte Datei ist nur möglich, wenn das Standard-Paßwort oder das beim Dateizugriff angegebene Paßwort mit dem der Datei übereinstimmt.

Eingabe: DE=adr(Paßwort)

32 Benutzernummer wählen/lesen

Eingabe: E=User-No
oder
Eingabe: E=FF, Ausgabe : A=User-No

12 Versionsnummer

Ausgabe: HL=Versionsnummer

107 Seriennummer lesen.

Die Seriennummer ist 6 bytes lang.

Eingabe: DE=adr(Seriennummernfeld)
Ausgabe: Seriennummer im Seriennummernfeld

Programm-Aufrufe

50 BIOS-Funktion aufrufen

Byte(s)	BIOS-Parameter-Block
00	BIOS-Funktions-Nummer
01	Inhalt des Registers A
03,02	Inhalt des Registerpaares BC
05,04	Inhalt des Registerpaares DE
07,06	Inhalt des Registerpaares HL

Eingabe: DE=adr(BIOS PB)
Ausgabe: von BIOS-Funktion abhängig

RSX-Parameter sind immer 16 Bit lang.

Byte	RSX-Parameter-Block
00	RSX-Funktions-Nummer
01	Anzahl der Parameter
02...	Parameter

Eingabe: DE=adr(RSX PB)
Ausgabe: A=Fehler, H=phys. Fehler

47 nächstes Programm aufrufen.

Ein Programm kann unter Umgehung des CCP mit dieser Funktion das nächste Programm zur Ausführung bringen. Mit Funktion 108 kann es ihm außerdem Zustands-Informationen mitliefern.

Eingabe: E=Chaincode,
Programm-Name im DMA-Puffer

108 Programm-Antwort-Code lesen/schreiben.

Der CCP übergibt immer 0000. (siehe auch Funktion 47.)

Code(s)	Bedeutung
0000-FEFF	Programm erfolgreich durchgeführt
FF80-FFFC	reserviert
FFFD	Programmabbruch wegen BDOS-Fehler
FFFE	Programmabbruch weil ^C eingegeben wurde
Eingabe:	DE=Antwort-Code
oder	
Eingabe:	DE=FFFF, Ausgabe: HL=Antwort-Code

IOBYTE

7 IOBYTE lesen.

Ausgabe: A=IOBYTE

8 IOBYTE beschreiben.

Eingabe: E=IOBYTE

Konsol-Ein-/Ausgaben

1 Konsoleingabe.

Eingabe eines Zeichens von CONIN: und Ausgabe desselben an CONOUT: (Echo).

Ausgabe: A=Zeichen

2 Konsolausgabe.

Ausgabe eines Zeichens an CONOUT:
Eingabe: E=Zeichen

6 direkte Konsolein-/ausgabe

Status: 0 kein Zeichen, FF Zeichen im Puffer.

Eingabe: E=Zeichen

Eingabe: E=FF
Ausgabe: A=Zeichen
Ist A=0, war kein Zeichen im Puffer.
Eingabe: E=FE
Ausgabe: A=Status (nur CP/M3.0)

Eingabe: E=FD
Ausgabe: A=Zeichen (nur CP/M3.0)
Funktion wartet auf ein Zeichen.

9 Stringausgabe

Der Stringbegrenzer ist defaultmäßig das "$". Es kann mit Funktion 110 geändert werden.

Eingabe: DE=adr(String)

Der String kann editiert werden.

Byte(s)	Puffer
00	maximale Länge (maxlen = 1-255)
01	tatsächliche Länge des String (aktlen)
02...	String

Eingabe: DE=adr(Puffer), maxlen
Ausgabe: Puffer=maxlen,aktlen,String

11 Konsolstatus lesen

Ausgabe: A (0=BDOS-Puffer leer, 1=Puffer voll)

109 Konsolmodus lesen/wählen

Eingabe: DE=FFFF
Ausgabe: HL=Konsolmodus

Eingabe: DE=Konsolmodus

110 Ausgabebegrenzer lesen/wählen

Eingabe: DE=FFFF
Ausgabe: A=Begrenzer

Eingabe: E=Begrenzer

111 Block zur Konsolenausgabe

Bytes	Character-Control-Block (CCB)
01,00	Adresse des String-Block
03,02	Länge des String-Block

Eingabe: DE=adr(CCB)

3 — Hilfseingang, Lochstreifenleser

Ausgabe: A=Zeichen

4 — Hilfsausgang, Lochstreifenstanzer

Eingabe: E=Zeichen

5 — Druckerausgang

Eingabe: E=Zeichen

7 — Status Hilfseingang

Ausgabe: A=AUXIN:-Status

8 — Status Hilfsausgang

Ausgabe: A=AUXOUT:-Status

Status	Information
00	Hilfskanal ist nicht bereit
FF	Hilfsausgang ist bereit, ein Zeichen zu senden, bzw. am Hilfseingang liegt ein gültiges Zeichen an.

112 — Block zum Lister

Der Aufbau des CCB ist in Funktion 111 im vorigen Abschnitt dargestellt.

Eingabe: DE=adr(CCB)

13 Disk-System-Reset

Ausgabe: A=00h

14 Bezugslaufwerk wählen

Eingabe: E=Floppy-Nr
Ausgabe: A=Fehler, H=phys. Fehler

25 Bezugslaufwerk ermitteln

Ausgabe: A=Lauf.-Nr

28 Disk-Schreibschutz einschalten

Ausgabe: A=00h

29 Schreibschutz-Vektor lesen.

Der Protect-Vektor ist ein Wort, in dem für jedes Laufwerk ein Bit reserviert ist. Bit 0 ist dabei Laufwerk A zugeordnet. Eine 1 signalisiert ein schreibgeschütztes Laufwerk.

Ausgabe: HL=Protect-Vektor, A=LSB(Protect-Vektor)

24 Login-Vektor lesen

Der Login-Vektor ist ein Wort, in dem für jedes Laufwerk ein Bit reserviert ist. Bit 0 ist dabei Laufwerk A zugeordnet. Eine 1 signalisiert ein eingeloggtes Laufwerk. Funktion 14 und Datei-Funktionen beeinflussen den Login-Vektor.

Ausgabe: HL=Login-Vektor, A=LSB(Login-Vektor)

37 Login-/Protect-Vektor rücksetzen

Eingabe: DE=Reset-Vektor

31 Adresse des Disk-Parameter-Blocks lesen

Ausgabe: HL=adr(DPB)

Ausgabe: HL=FFFF bei Fehler

27 Adresse der Disk-Belegungstabelle lesen

Vorsicht bei CP/M3.0: Die Disk-Belegungstabellen liegen oftmals in Block 0 der Bank 0 und sind für transiente Programme über diese Funktion nicht zu erreichen. Zur Berechnung des freien Disk-Platzes wird Funktion 46 empfohlen.

Ausgabe: HL=Adresse

46 freien Disk-Platz berechnen

Eingabe: E=Floppy-Nr
Ausgabe: A=Fehler,H=phys.Fehler, Anzahl der freien Records in den ersten 3 Bytes des DMA-Puffers. Das LSB steht im ersten Byte.

98 Disk-Blöcke freigeben

Ausgabe: A=Fehler,H=phys.Fehler

100 Directory-Label beschreiben

Eingabe: DE=adr(FCB)
Ausgabe: A=Dir-Code,H=phys.Fehler

101 Directory-Label-Daten lesen

Eingabe: E=Floppy-Nr

Ausgabe: A=Label-Daten
oder A=00, wenn kein Directory-Label existiert
oder A=FF,H=phys.Fehler, wenn Fehler

152 FCB vorbereiten

Die Funktion wandelt den Dateinamen in dem String in einen FCB um. Der Dateiname darf die folgende Gestalt aufweisen: <d:>filename<.typ><;password>. Wird nur ein Laufwerksname angegeben, werden die Namens- und Typfelder des FCB mit Blanks gefüllt.

Bytes	PFCB-Inhalt
01,00	Adresse des Strings mit dem Dateinamen
03,02	Adresse des aufzubauenden FCB
Eingabe:	DE=adr(PFCB)

22 Datei erzeugen

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

15 Datei öffnen

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

16 Datei schließen

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

23 Datei umbenennen

Der neue Name muß in den zweiten 16 Bytes des FCB stehen.

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

99 Datei kürzen

Die Funktion gibt alle Records frei, deren Nummer größer als die in r0...r2 angegebene ist.

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

19 Datei löschen

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

30 Datei-Attribute setzen

Die Datei-Attribute f1'-f8' belegen die 8. Bits des Dateinamen, t1'-t3' die des Datei-Types. Ein gesetztes Bit entspricht einem gesetzten Attribut.

Attr.	Bedeutung
f1'-f4'	frei für Benutzer
f5'-f8'	reserviert als Interface-Attribute
t1'	Read-Only (R/O)
t2'	System. System-Dateien werden mit DIR nicht angezeigt. Sind sie unter USER0 abgelegt, können sie von anderen USER-Bereichen aus als R/O-Dateien mitbenutzt werden.
t3'	Archiv

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

17 erste spezif. Datei suchen

Eingabe: DE=adr(FCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

18 weitersuchen

Ausgabe: A=Dir-Code, H=phys. Fehler

102 Datum-Eintrag und Paßwort-Modus lesen

Eingabe: DE=adr(FCB),

Ausgabe: A=Dir-Code, H=phys. Fehler

103 XFCB der Datei schreiben

Eingabe: DE=adr(XFCB),
Ausgabe: A=Dir-Code, H=phys. Fehler

59 Overlay laden

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=Phys. Fehler

Records

26 DMA-Puffer-Adresse setzen

Alle Schreib- und Lese-Funktionen benutzen den DMA-Puffer als Quelle bei Schreib- bzw. Ziel bei Lese-Operationen.

Eingabe: DE=adr(Puffer)

36 nächsten Record-Nr. holen

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler,R0..2 im FCB oder H=phys. Fehler

35 letzte Record-Nr.+1 holen

Diese Nummer entspricht der Anzahl der von der Datei belegten Records, wenn sie sequentiell geschrieben wurde.

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler,R0..2 im FCB oder H=phys. Fehler

20 nächsten Record lesen

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=phys. Fehler

21 nächsten Record beschreiben

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=phys. Fehler

33 Record direkt lesen

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=phys. Fehler

34 Record direkt beschreiben

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=phys. Fehler

40 Record direkt beschreiben

Ein neuer Record wird vorher mit Nullen beschreiben.

Eingabe: DE=adr(FCB)
Ausgabe: A=Fehler, H=phys. Fehler

48 Puffer entleeren

Schreibt alle Daten in den internen De-/Blocking-Puffern in die entsprechenden Dateien, falls dies noch nicht geschehen ist. Ist das Lösch-Flag=FF, werden auch die Datenpuffer geleert.

Eingabe: E=Lösch-Flag
Ausgabe: A=Fehler, H=phys. Fehler

44 Multisektorpreset

Legt die Anzahl der bei einer Schreib- bzw. Leseoperation zu berücksichtigenden Records fest. Der CCP setzt Die Anzahl auf 1.

Eingabe: E=Anzahl Sektoren (erlaubt : 1 ... 128)
Ausgabe: A=Fehler, H=phys. Fehler

6.5 BIOS-Einsprungtabelle des CP/M2.2

Die Einsprung-Adresse einer Funktion ergibt sich aus der Basisadresse der BIOS-Einsprungtabelle und dem Offset (Off.). Die ersten 3 Bytes der Zeropage enthalten einen Sprung in die BIOS-Einsprungtabelle zum Warmstart.

Off.	Name	Funktion
00	BOOT	Kaltstart
03	WBOOT	Warmstart
06	CONST	Konsole-Status abfragen
09	CONIN	Konsole-Eingabe
0C	CONOUT	Konsole-Ausgabe
0F	LIST	Druckerausgabe
12	PUNCH	Lochstreifen-Stanzer-Ausgabe
15	READER	Lochstreifen-Leser-Eingabe
18	HOME	Laufwerks-Kopf auf Spur 0 stellen
1B	SELDSK	Laufwerk auswählen
1E	SETTRK	Spur auswählen
21	SETSEC	Sektor auswählen
24	SETDMA	Daten-Puffer-Adresse festlegen
27	READ	Sektor lesen
2A	WRITE	Sektor beschreiben
2D	LISTST	Drucker-Status abfragen
30	SECTRAN	Sektor-Nummer übersetzen

6.6 BIOS-Funktions-Nummern des CP/M3.0

Die BIOS-Funktionen werden mit BDOS-Funktion 50 angesprungen.

Nr.	Name	Funktion
0	BOOT	Kaltstart
1	WBOOT	Warmstart
2	CONST	Konsole-Eingabe-Status abfragen
3	CONIN	Konsole-Eingabe
4	CONOUT	Konsole-Ausgabe
5	LIST	Druckerausgabe
6	AUXOUT	Zusatzkanal-Ausgabe
7	AUXINR	Zusatzkanal-Eingabe
8	HOME	Laufwerks-Kopf auf Spur 0 stellen
9	SELDSK	Laufwerk auswählen
10	SETTRK	Spur auswählen
11	SETSEC	Sektor auswählen
12	SETDMA	Daten-Puffer-Adresse festlegen
13	READ	Sektor lesen
14	WRITE	Sektor beschreiben
15	LISTST	Drucker-Status abfragen
16	SECTRAN	Sektor-Nummer übersetzen
17	CONOST	Konsole-Ausgabe-Status abfragen
18	AUXIST	Zusatzkanal-Eingabe-Status abfragen
19	AUXOST	Zusatzkanal-Ausgabe-Status abfragen
20	DEVTBL	Adresse der Zeichen-Ein-/Ausgabe-Tab. laden
21	DEVINI	Zeichen-Ein-/Ausgabe-Treiber initialisieren
22	DRVTBL	Adresse der Laufwerks-Tabelle laden
23	MULTIO	Anzahl Sektoren setzen
24	FLUSH	Deblocking-Puffer leeren
25	MOVE	Speicherbereich kopieren
26	TIME	Uhrzeit schreiben/lesen
27	SELMEM	Speicherbank umschalten
28	SETBNK	Bank für folgende Disk-R/W-Operation wählen

29	XMOVE	Bänke für Bank-to-bank-MOVE
30	USERF	Systemspez. Funktion. In den CPCs wird sie zum Einsprung in CP/M-fremde Routinen benutzt. (z.B. laufen die GRAFIK.INC-Prozeduren des Turbo-Pascal meist über diese Funktion).
31	RESERV1	reserviert
32	RESERV2	reserviert

7. Hardware

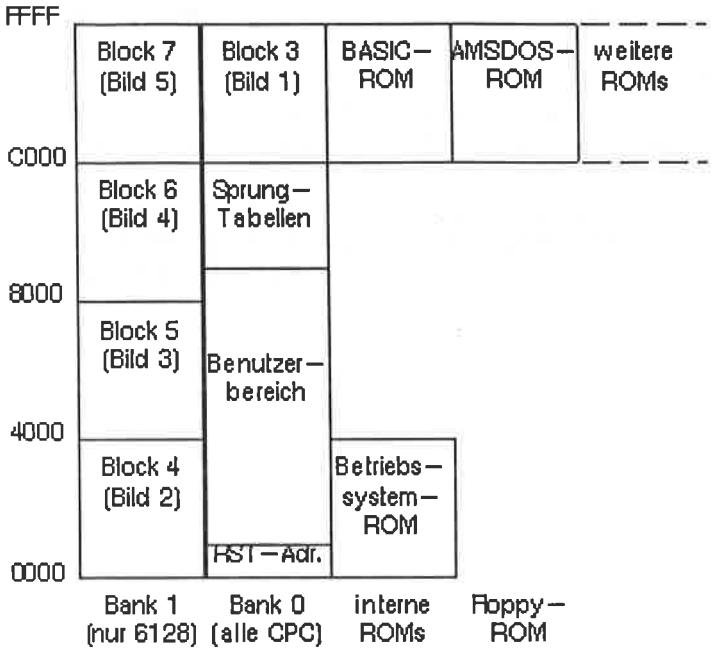
7.1 Speicher-Verwaltung

Bei Schreiboperationen wird grundsätzlich ins RAM geschrieben, egal welche Speicher-Konfiguration gewählt ist.

Gelesen wird in den Blöcken 1 und 2 immer aus dem RAM, da hier keine parallelliegenden ROMs vorgesehen sind.

In Block 0 kann statt des RAMs das Betriebssystem-ROM als Datenquelle ausgewählt werden. Weiter ROMs sind nicht vorgesehen.

In Block 3 können bis zu 252 Expansion-ROMs parallel zum BASIC-ROM verwaltet werden, wobei das AMSDOS-ROM im CPC 664 und 6128 einen dieser Plätze belegt.



Physikalischer Speicheraufbau

7.2 Assemblerbefehle für die Ein-/Ausgabe

Die einzigen Befehle, mit denen der Prozessor mit Peripherie-Bausteinen kommunizieren kann, sind der IN r,(C) und der OUT (C),r. Diese Befehle benutzen das BC-Register, um einen von 65536 möglichen Ein-/Ausgabekanälen zu adressieren. Die CPCs benutzen sogar nur das B-Register zur Bausteinauswahl.

7.3 Peripherie-Bausteine

Die Portadresse der Bausteine muß im B-Register stehen.

A11	1	40	A10	
A12	2	39	A9	
A13	3	38	A8	
A14	4	37	A7	
A15	5	36	A6	
Takt	6	35	A5	
D4	7	34	A4	
D3	8	33	A3	
D5	9	32	A2	
D6	10	31	A1	
(+5V) Vcc	11	30	A0	
D2	12	29	GND	
D7	13	28	nRFSH	
D0	14	27	nM1	
D1	15	26	nRESET	
nINT	16	25	nBUSRQ	
nNMI	17	24	nWAIT	
nHALT	18	23	nBUSAK	
nMREQ	19	22	nWR	
nMREQ	20	21	nRD	

Der Mik oprozessor Z80

nCPU_ADDR	1	40	MA0/CCL	
READY	2	39	Takt	
nCAS	3	38	Vcc1	
n244EN	4	37	nRESET	
nMWE	5	36	R	
nCAS_ADDR	6	35	GND	
nRAS	7	34	G	
XTAL	8	33	Vcc2	
Vcc2	9	32	B	
nINTERRUPT	10	31	D7	
nSYNC	11	30	D6	
nROMEN	12	29	D5	
nRAMRD	13	28	D4	
HSYNC	14	27	D3	
VSYNC	15	26	D2	
nIORQ	16	25	D1	
nM1	17	24	D0	
nMRQ	18	23	DISPEN	
nRD	19	22	Vcc1	
A15	20	21	A14	

Des Gate Array 40007

Vss	1	40	VSYNC	
nRES	2	39	HSYNC	
LPSTB	3	38	PA0	
MA0	4	37	PA1	
MA1	5	36	PA2	
MA2	6	35	PA3	
MA3	7	34	PA4	
MA4	8	33	D0	
MA5	9	32	D1	
MA6	10	31	D2	
MA7	11	30	D3	
MA8	12	29	D4	
MA9	13	28	D5	
MA10	14	27	D6	
MA11	15	26	D7	
MA12	16	25	nCS	
MA13	17	24	RS	
DISPTMG	18	23	E	
CUDISP	19	22	R/nW	
Vcc	20	21	CCLK	

Der CRTC HD 6845

Gate-Array GA

Portadresse: 7F

Die 4 Register des Gate-Array werden mit Bit 6 und 7 des eingeschrieben Datenbytes adressiert. Ein Lesen aus dem GA ist nicht möglich.

Registerauswahl durch Datenbyte

B7	B6	Register
0	0	Farbstift
0	1	Farbe
1	0	SYSTEM
1	1	PAL

System-Register

Bit	Funktion
5	?
4	V-Sync-Zähler löschen
3	RAM selektieren in Block 3
2	RAM selektieren in Block 0
1,0	Bildschirmmodus

Bildschirmmodus

B1	B0	Modus	Spalten	Farben
0	0	0	20	16
1	1	40	4	
1	0	2	80	2
1	1	0	(ohne blinken)	

Farbstift-Register

B4	B3-B0	Funktion
0	wert	Farbstiftwert
1	xxxx	Hintergrund

In das Farbstift-Register wird die Nummer des Farbstiftes eingegeben, dem danach eine Farbe durch das Farb-Register zugewiesen werden soll.

Die Intensitäten der Grundfarben werden im Farb-Register wie folgt codiert:

Farbe	grün	rot	blau
Bits	5,4	3,2	1,0

Da die beiden Bits einer Farbe gleichwertig sind, gibt es für jede Farbe nur 3 Helligkeitsstufen. Insgesamt also 27 Farbkombinationen, wie sie vom BASIC angeboten werden.

Das PAL-Register wird vom PAL-Baustein ausgewertet, hat also in den 464- und 664-Rechnern keine Funktion.

Programmable-Array-Logik im CPC 6128

Der PAL-Baustein belegt keine eigene Portadresse, sondern wird vom Gate-Array versorgt. Zum GA geschickte Daten mit gesetzten Bits 6 und 7 sind für den PAL. Der PAL-Baustein verwaltet die RAM-Bänke. Normalerweise (wie im CPC 464/664) ist Bank 0 eingeschaltet. Es sind 8 verschiedene Speicherkonfigurationen mit den Bits 0 bis 2 wählbar, wobei Nr. 1, 2 und 3 nur für CP/M verwendet werden. Die folgende Tabelle gibt an, welcher Block aus welcher Bank nach der Konfigurierung als welcher Bus-Block adressiert wird.

B2-B0	Bus-Block	Block	Bank	
0	0-3	0-3	0	Standardbetriebsart
1	(Modus für CP/M3.0)			
2	(Modus für CP/M3.0)			
3	(Modus für CP/M3.0)			
4	1	0	1	
5	1	1	1	die übrigen Blöcke sind
6	1	2	1	wie in der Standardbe-
7	1	3	1	triebsart belegt.

Parallel-Ein-/Ausgabe-Baustein PIO 8255

Portadressen:

Port A:	F4
Port B:	F5
Port C (lesen):	F6
Steuerregister:	F7

Steuerregister:

Bit(s)	Betriebsart
0: 1/0	PC0 - PC3 sind Ein-/Ausgabe-Port
1: 1/0	Port B ist Ein-/Ausgabe-Port
2: 1/0	Betriebsart für Port B und PC0 - PC3
3: 1/0	PC4 - PC7 sind Ein-/Ausgabe-Port
4: 1/0	Port A ist Ein-/Ausgabe-Port
5: 1/0	Betriebsart 1/0 für Port A und PC4-PC7,wenn B6=0
6: 1	Betriebsart 2 für Port A und PC3-PC7

Das Beschreiben des Steuerregister muß mit gesetztem Bit7 erfolgen.

Betriebsart 0 ermöglicht eine einfache Ein-/Ausgabe von Daten. Ausgaben werden gespeichert, Eingaben nicht.

Betriebsart 1 benutzt Leitungen des Port C für die Kontrolle der Datenflüsse in Port A bzw. B. Ein- und Ausgaben werden gespeichert. Die Steuerleitungen der Port C-Leitungen (PC 0..7) zeigt die folgende Tabelle.

Mit Betriebsart 2 (nur Port A) wird ein bidirektionaler Port realisiert. Die Steuerleitungen werden wie bei Betriebsart 1 Eingabe und Ausgabe benutzt. Die L-H-Flanke des nACKa-Signals bringt den Port in den hochohmigen Zustand.

Steuerleitungen (Port C)

bit	bei Eingabekanal	bei Ausgabekanal	bei Betr.-Art 2
0	INTRb-Ausgang	INTRb-Ausgang	nicht benutzt
1	IBFb-Ausgang	nOBFb-Ausgang	nicht benutzt
2	nSTBb-Eingang	nACKb-Eingang	nicht benutzt
3	INTRa-Ausgang	INTRa-Ausgang	INTRa-Ausgang
4	nSTBa-Eingang	nicht benutzt	nSTBa-Eingang
5	IBFa-Ausgang	nicht benutzt	IBFa-Ausgang
6	nicht benutzt	nACKa-Eingang	nACKa-Eingang
7	nicht benutzt	nOBFa-Ausgang	nOBFa-Ausgang

Funktion der Steuerleitungen:

INTR kann zu Interrupt-Erzeugung benutzt werden. Der High-Zustand zeigt an, daß Daten von der Peripherie gelesen worden sind bzw. gesendet werden. Interrupts können durch Löschen/Setzen des Bits PC 2 (für Port B), PC 4 (für Port A, Eingabe) und PC 6 (für Port A, Ausgabe) unterbunden/erlaubt werden.

IBF High-Zustand besagt, daß der Eingabepuffer voll ist

nSTB Low-Pegel führt zur Übernahme der anliegenden Daten und zum Setzen des IBF.

nOBF Low-Pegel zeigt an, daß gültige Daten am Port anliegen.

nACK Low-Pegel informiert die PIO darüber, daß die anliegenden Daten gelesen werden und löscht nOBF.

Port C beschreiben

Es werden einzelne Bits gelöscht/gesetzt, indem in das Steuerregister ein Datum mit gelöschtem Bit7 geschrieben wird:

bit(s)	Funktion
0	neuer Bitwert
3,2,1	Bit wählen
6,5,4	keine Funktion
7	Bit-Manipulations-Flag, muß auf 0 gesetzt sein

Ein Schreiben in das Steuer-Register mit gesetztem Bit 7 wird als Betriebsart-Wahl interpretiert.

Port C lesen:

Port C kann über Portadresse F6 jederzeit gelesen werden. Dabei haben die einzelnen Bits je nach Betriebsart verschiedene Bedeutungen (s.o.). Von den Betriebsarten nicht benutzte Leitungen können als einfache Ein-/Ausgabe-Leitungen verwendet werden.

In den CPCs wird die PIO nur in Betriebsart 0 benutzt, dabei haben die 24 IO-Leitungen folgende Funktionen:

An Port A ist der Sound-Generator (und damit die Tastatur-Abfrage) angeschlossen. Je nach Bedarf wird Port A als Eingang oder Ausgang programmiert.

Port B ist fest als Eingang programmiert.

Bit(s)	Funktion
0	Vertikal-Synchronisation
1-3	Firmenname in der Kaltstart-Meldung
4	SECAM-Brücke für CRTC-Programmierung
5	EXP-Leitung des Expansion-Anschlusses
6	Printer Busy
7	Daten vom Cassettenrecorder

Port C ist fest als Ausgang programmiert.

Bit(s)	Funktion
0 - 3	Tastatur-Matrix
4=1/0	Cassettenrecorder-Motor ein/aus
5	Daten zum Cassettenrecorder
6	BC1-Anschluß des Sound-Generators
7	BDIR-Anschluß des Sound-Generators

Sound-Generator PSG AY-3-8912

Funktionsauswahl (über Port C der PIO 8255):

BDIR	BC1	Funktion
0	0	PSG-Datenbus hochohmig
0	1	Daten aus PSG lesen
1	0	Daten in PSG schreiben
1	1	Register für nächsten Zugriff wählen

Funktion der Register:

Reg	Breite	Funktion
1,0	12	Periodendauer des Tones in Kanal A
3,2	12	Periodendauer des Tones in Kanal B
5,4	12	Periodendauer des Tones in Kanal C
6	5	Rausch-Ton-Höhe
7	8	Control-Register
8	4	Lautstärke des Kanal A
9	4	Lautstärke des Kanal B
10	4	Lautstärke des Kanal C
12,11	16	Periodendauer der Hüllkurve
13	4	Hüllkurvenform (HK-Form)
14	8	Port A

Bits	Control-Register-Inhalt	Bit	HK-Form
		0	Hold
0/1/2	=0/1 Kanal A/B/C: Ton ein/aus	1	Alternate
3/4/5	=0/1 Kanal A/B/C: Rauschen ein/aus	2	Attack
6/7	=0/1 Port A/B : Ein-/Ausgabe	3	Continue

Der PSG-Datenbus ist an Port A der PIO angeschlossen. Vor der Kommunikation mit dem PSG muß deshalb PortA der PIO auf Ein- bzw. Ausgabe-Betrieb programmiert werden.

Video-Controller CRTC 6845

Portadresse des Adreßregisters: BC
Portadresse des Datenregisters: BD beim Schreiben
BF beim Lesen

Register (R/W-Spalte sagt, ob Lesen oder Schreiben mgl.):

Reg	r/w	Funktion
00	w	Zeichen je Zeile total mit Rand u. Strahlrücklauf
01	w	dargestellte Zeichen je Zeile
02	w	Horizontal-Synchronisations-Impuls-Position
03	w	H- und V-Synchronisations-Impuls-Breiten (4 Bits)
04	w	Anzahl Rasterzeilen total je Bild (7 Bits)
05	w	Feinabgleich der Bildwiederhol-Frequenz (6 Bits)
06	w	dargestellte Rasterzeilen je Bild (7 Bits)
07	w	Vertikal-Synchronisations-Impuls-Position (7 Bit)
08	w	Zeilensprung-Modus (Interlace, 2 Bits)
09	w	Rasterzeilen je Zeichenzeile (5 Bits)
0A	w	obere Cursor-Begrenzung (Hardware-Cursor, 7 Bits)
0B	w	untere Cursor-Begrenzung (Hardware-Cursor, 5 Bit)
0C	r/w	Bildschirm-Speicher-Adresse (MSB, 6 Bits)
0D	r/w	Bildschirm-Speicher-Adresse (LSB, 8 Bits)
0E	r/w	Cursor-Adresse (MSB, 6 Bits)
0F	r/w	Cursor-Adresse (LSB, 8 Bits)
10	r	Lightpen-Adresse (MSB, 6 Bits)
11	r	Lightpen-Adresse (LSB, 8 Bits)

Der Hardware-Cursor wird in den CPCs nicht verwendet.

Bits	obere Cursor-Begrenzung
4 - 0	obere Cursor-Rasterzeile
6,5	Cursormodus
B6 B5	Cursormodus
0 0	nicht blinkend
0 1	kein Cursor
1 0	schnell blinkend
1 1	langsam blinkend

Floppy-Disk-Controller FDC 765

Portadr. des Hauptstatusregisters: FB7E in Registerpaar BC
Portadr.des Kommunikations-Ports: FB7F in Registerpaar BC
Portadr. des Floppymotorschalters: FA7E in Registerpaar BC

Der Motorschalter schaltet mit gesetztem Bit0 die Motoren aller Laufwerke ein, egal, welches Laufwerk ausgewählt worden ist.

Hauptstatusregister

Bit	Aussage
0	Laufwerk A beschäftigt, löschen durch Status 0 lesen
1	dto., aber Laufwerk B
2	dto., Laufwerk C (in CPCs nicht vorgesehen)
3	dto., Laufwerk D (auch nicht vorgesehen)
4	FDC beschäftigt
5	Befehls-Ausführung läuft
6	=1/0: Daten für Prozessor da/erwarte Daten von Prozessor
7	FDC bereit zu Daten-Transfer

Die Bits 0 bis 3 können nur durch den Befehl STATUS 0 LESEN gelöscht werden.

Statusbyte 0

Bit(s)	Aussage
0	aktives Laufwerk
1,2	immer 0000
3	Laufwerk war nicht fertig
4	Laufwerksfehler od. Track 0 nicht erreicht
5	Suchbefehl beendet
7,6	Fehler-Code (Aufschlüsselung siehe unten)

Code	Aussage
0	Befehl erfolgreich ausgeführt (außer nach Schreib-/Lese-Befehlen)
1	Nach Schreib-/Lese-Befehlen immer gesetzt, sonst zeigt es den Abbruch einer begonnen Operation ab (z.B. Lesefehler).
2	ungültigen Befehl erkannt
3	Abbruch wegen Änderung des READY-Signals des bezogenen Laufwerkes bei der Befehlsausführung

Status-Byte 1

Bit	Aussage, wenn Bit gesetzt
0	Adreßmarke fehlt
1	Diskette ist schreibgeschützt
2	Sektor nicht gefunden od. ID-Feld-Fehler
3	immer 00
4	Prozessor hat Datum zu langsam abgeholt/gesendet
5	Checksum-Error
6	immer 00
7	Zugriff auf Sektor nach Spurende versucht

Status-Byte 2

Bit	Aussage, wenn Bit gesetzt
0	Adreßmarke im Datenfeld fehlt
1	Spur-Nummer in ID und Befehl verschieden oder Spur FF
2	beim Prüfen den spezifizierten Sektor nicht gefunden
3	beim Prüfen Prüfbedingung erfüllt
4	Spur-Nummer in ID und Befehl verschieden
5	Checksum-Error im Datenfeld
6	Beim Lesen oder Prüfen "gelöschten" Sektor gefunden
7	immer 00

Status-Byte 3

Bit(s)	Aussage, wenn Bit gesetzt
0	aktives Laufwerk
1,2,3	000000
4	Schreib-/Lese-Kopf auf Spur 0
5	READY-Signal des Laufwerkes
6	Diskette schreibgeschützt
7	FAULT-Signal des Laufwerkes

dr = Laufwerk (A=0, B=1)

Die Befehlsbearbeitung des FDC besteht aus bis zu 3 Teilen, der Befehls-Eingabe, -Ausführung und der Ergebnis-Ausgabe. Der 2. oder 3. Teil fehlt bei einigen der insgesamt 15 Befehle. Im folgenden beschreibt die 2.Spalte der Tabelle jeweils, wie der Befehl einzugeben ist und die 3., was der FDC als Ergebnis ausgibt. Der Prozessor muß das Ergebnis vollständig gelesen haben, bevor er den nächsten Befehl eingeben kann. Die Befehle, Daten und Ergebnisse werden über den Kommunikationsport ein- bzw. ausgegeben.

Befehle für den Daten-Transfer von/auf Diskette

Byte	Befehlseingabe	Ergebnisausgabe
0	Befehls-Code	Status-Byte 0
1	Laufwerksnummer	Status-Byte 1
2	Spurnummer	Status-Byte 2
3	Kopfadresse	Spurnummer
4	Sektoradresse	Kopfadresse
5	Sektorgröße	Sektoradresse
6	letzte Sektornummer der Spur	Sektorgröße
7	Lücke zwischen ID und Daten	
8	Sektorlänge, wenn Sektorgröße=0	

Code	Befehl
45	Sektor(en) beschreiben
46	Sektor(en) lesen
51	Sektor(en) und Speicher auf Identität überprüfen
59	dto., prüfe ob kleiner oder gleich
5D	dto., prüfe ob größer oder gleich
49	Daten schreiben und als gelöscht markieren
4C	als gelöscht markierte Daten lesen
42	Spur lesen
4D	Spur formatieren (Befehlsbytes 2 bis 4 entfallen)
4A	ID lesen (Befehlsbytes 2 bis 8 entfallen)

Spur 0 suchen

Byte	Befehlseingabe
0	07
1	Laufwerksnummer

Es werden bis zu 77 Step-Impulse in Richtung Spur 0 ausgeführt. Der Track0-Impuls oder der 77. Step beenden die Ausführung. Das Ergebnis muß explizit angefordert werden (STATUS 0 ABFRAGEN), bevor der nächste Schreib-/Lesebefehl gegeben werden kann.

beliebige Spur suchen

Byte	Befehlseingabe
0	0F
1	Laufwerksnummer
2	Spurnummer

Das Ergebnis muß explizit angefordert werden (Status 0 abfragen), bevor der nächste Schreib-/Lesebefehl gegeben werden kann.

Status 0 abfragen

Byte	Befehlseingabe	Ergebnisausgabe
0	00	Status-Byte
0	1	Spurnummer nach Suchbefehl

Status 3 abfragen

Byte	Befehlseingabe	Ergebnisausgabe
0	04	Status 3
1	00	

Laufwerks-Parameter eingeben

Byte	Befehlseingabe		
0	03		
1	Bits	Wert	Inhalt
	7..4s	Step-Periode:	(32-2s)ms
	3..0a	Kopf abheben:	a*32ms
2	Bits	Wert	Inhalt
	7..1	f. Kopf aufsetzen:	(f+1)*4ms
	0	DMA-Modus (muß 0 sein)	

PA3	1	40	PA4	
PA2	2	39	PA5	
PA1	3	38	PA6	
PA0	4	37	PA7	
nRD	5	36	nWR	
nCS	6	35	RESET	
GND	7	34	D0	
A1	8	33	D1	
A0	9	32	D2	
PC7	10	31	D3	
PC6	11	30	D4	
PC5	12	29	D5	
PC4	13	28	D6	
PC0	14	27	D7	
PC1	15	26	Vcc	
PC2	16	25	PB7	
PC3	17	24	PB6	
PB0	18	23	PB5	
PB1	19	22	PB4	
PB2	20	21	PB3	

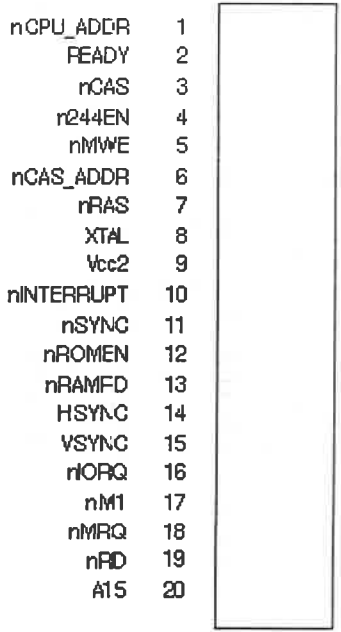
Der Parallelportbaustein 8255

CHANNEL_C	1	28	DA0	
TEST1	2	27	DA1	
Vcc	3	26	DA2	
CHANNEL_B	4	25	DA3	
CHANNEL_A	5	24	DA4	
Vss	6	23	DA5	
IOA7	7	22	DA6	
IOA6	8	21	DA7	
IOA5	9	20	BC1	
IOA4	10	19	BC2	
IOA3	11	18	BDIR	
IOA2	12	17	A8	
IOA1	13	16	nRESET	
IOA0	14	15	CLOCK	

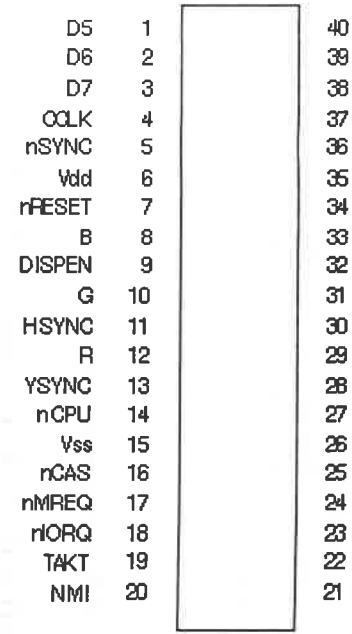
Der Soundchip AY—3—8912

RESET	1	40	Vcc (+5V)	
nRD	2	39	nRW/SEEK	
nWR	3	38	LDC/DIR	
nCS	4	37	FLTR/STEP	
A0	5	36	HOLD	
DB0	6	35	READY	
DB1	7	34	WPRT/2SIDE	
DB2	8	33	FLT/TRK0	
DB3	9	32	PS0	
DB4	10	31	PS1	
DB5	11	30	WDATA	
DB6	12	29	US0	
DB7	13	28	US1	
DRQ	14	27	SIDE	
nDACK	15	26	MFM	
TC	16	25	WE	
INDEX	17	24	SYNC	
INT	18	23	RDATA	
TAKT	19	22	WINDOW	
GND	20	21	BCLK	

Der Floppycontroller uPD 765 A



Das Gate Array 40008
(Pinbelegung identisch mit 40007)



Das Gate Array 40010

In den CPCs muß das Register C' den Inhalt des SYSTEM-Registers enthalten, während in B' die Portadresse des Gate-Arrays steht. Die übrigen Register des zweiten Registersatzes werden zur Berechnung der Adressen in den ROM's benutzt.

8.1 Adressierungsarten

Zugriff auf ein Byte-Datum (8bit)

Register direct (z.B.:INCB): Datum ist der Inhalt des Registers.

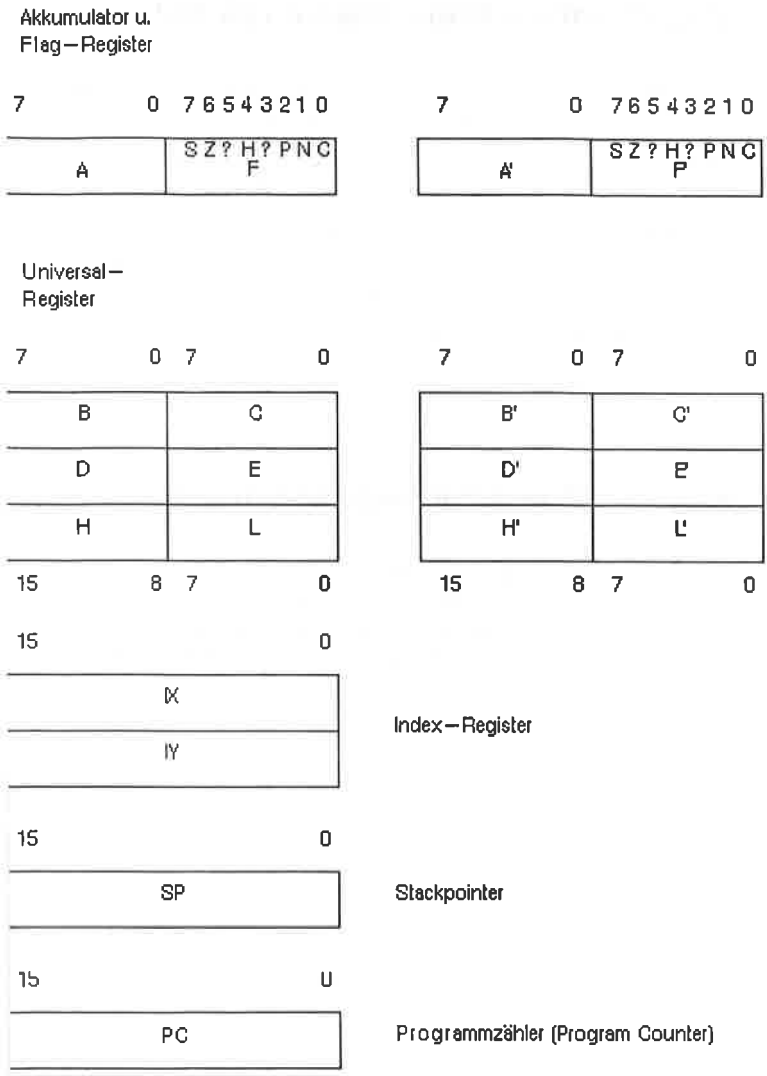
Immediate (z.B.:AND8F): Datum ist eine Konstante, die im Befehl angegeben wird

Extended (z.B.:LDA,(nn)): Datum steht im Speicher mit der Adresse nn. nn steht für einen Wert zwischen 0 und 65535 (=&FFFF)

Register indirect (z.B.:XOR(HL)): Hier wird der Inhalt eines Registerpaares als Speicheradresse interpretiert.

Indexed (z.B.:SLA(IX+d)): Datum steht in der Adresse, die durch die Summe von IX (oder IY) und der Distanz d gegeben ist. d hat einen Wert zwischen -128 und 127 und steht im Befehl als 2er-Komplement-Konstante.

Implied (z.B.:CPL): Befehl wirkt sich immer auf dasselbe bzw dieselben Register aus.



Das Programmiermodell des Z80

Zugriff auf ein Wort-Datum (16 bit)

Register direct (z.B.DECHL): Datum ist der Inhalt eines Registerpaares.

Immediate (z.B.LDBC,54AC): Datum ist eine Konstante, die im Befehl steht.

Extended (z.B.LDDE,(4801)): Das niederwertige Byte steht in der Adresse nn, das höherwertige in nn+1.

Stack addressing (PUSHAF): Datum wird in den Stack geschoben oder aus dem Stack zurückgeholt.

Zugriff auf beliebig lange Daten (Strings)

nur Implied Register indirekt
(z.B.:LDIR): Vor den Stringverarbeitungsbefehlen müssen die Register mit Parametern initialisiert werden.

8.2 Der Befehlssatz der Z80-CPU

Die Bedeutung der Kleinbuchstaben:

```
r       kann B,C,D,E,H,L,(HL) oder A sein
n       8-bit-Konstante zwischen 0 und 255
d       8-bit-Konstante zwischen -128 und 127, Distanz
e       8-bit-Konstante zwischen -128 und 127, Extension
qq      kann BC,DE,HL oder AF sein
ss,dd   können BC,DE,HL,SP sein
ii      kann BC,DE,IZ,SP sein, wobei IZ für IX und IY steht.
nn      16-bit-Konstante, Adresse
cc,c    stehen für eine der folgenden Bedingungen:
            NZ  no zero       Z   zero          Zero-Flag
            NC  no carry      C   carry         Carry-Flag
cc      kann zusätzlich stehen für:
            PO  parity odd    PE  parity even   Parity-Flag
            P   plus          M   minus         Sign-Flag
b       3-bit-Konstante zwischen 0 und 7, Bit-Nummer.
p       HEX-Konstante mit einem der folgenden Werte:
        00,08,10,18,20,28,30 oder 38, RST-Parameter.
```

Ladebefehle

Alle Z80-Ladebefehle beginnen mit "LD ". Deshalb haben wir in die Z80-Spalte nur noch die Parameter geschrieben. Alle Ladebefehle, bei denen eine Aussage über die Flags fehlt, beeinflussen diese nicht.

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
r1,r2	MOV r1,r2		
r,n	MVI r,n		
r,(IZ+d)	%		
(IZ+d),r	%		
(IZ+d),n	%		
A,(ss)	LDAX rp		dd,ss nur BC oder DE
(dd),A	STAX rp		rp nur B oder D
A,(nn)	LDA		
(nn),A	STA		
A,I	%	xx0x0-	I=Interruptvektor-
I,A	%	------	Register

A,R	%	xx0x0-	R=Refresh-
R,A	%	------	Register

16-Bit-Ladebefehle

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
dd,nn	LXI rp		
IZ,nn	%		
HL,(nn)	LHLD		
(nn),HL	SHLD		
dd,(nn)	%		
(nn),ss	%		
IZ,(nn)	%		
(nn),IZ	%		
SP,HL	SPHL		
SP,IZ	%		

Registerpaar-Inhalte in Stack retten und zurückholen

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
PUSH qq	PUSH rp	------	rp ohne SP
PUSH AF	PUSH PSW	------	
PUSH IZ	%	------	
POP qq	POP rp	------	qq ohne AF,rp ohne SP
POP AF	POP PSW	xxxxxx	
POP IZ	%	------	

Tausch-Befehle

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
EX AF,AF'	%	xxxxxx	
EX DE,HL	XCHG	------	
EX (SP),HL	XTHL	------	
EX (SP),IZ	%	------	
EXX	%	------	

Block-Transfer-Befehle

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
LDD	%	--0x0-	Registerbelegung:
LDDR	%	--000-	HL=adr(Quellblock)
LDI	%	--0x0-	DE=adr(Zielblock)
LDIR	%	--000-	BC=Blocklänge

8-bit-Arithmetik- und Logik-Befehle

(Statt r kann beim Z80 (nicht 8080) auch (IZ+d) stehen.)

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
ADD A,r	ADD r	xxxx0x	
ADD A,n	ADI	xxxx0x	
ADC A,r	ADC r	xxxx0x	
ADC A,n	ACI	xxxx0x	
SUB r	SUB r	xxxx1x	
SUB n	SUI	xxxx1x	
SBC A,r	SBB r	xxxx1x	
SBC A,n	SBI	xxxx1x	
AND r	ANA r	xx1x00	
AND n	ANI	xx1x00	
XOR r	XRA r	xx0x00	
XOR n	XRI	xx0x00	
OR r	ORA r	xx1x10	
OR n	ORI	xx1x10	
CP r	CMP r	xxxx1x	
CP n	CPI	xxxx1x	
DEC r	DCR r	xxxx1-	
INC r	INR r	xxxx0-	
CPL	CMA	--1-1-	Akku invertieren
DAA	DAA	xxxx-x	BCD-Korrektur in A
NEG	%	xxxx1x	Akku negieren

Rotations-Befehle

(Statt r kann auch (IZ+d) stehen.)

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
RLC	ARL C	--0-0x	
RRC	ARR C	--0-0x	
RLA	RAL	--0-0x	
RRA	RAR	--0-0x	
RLC r	%	xx0x0x	
RRC r	%	xx0x0x	
RL r	%	xx0x0x	
RR r	%	xx0x0x	
SLA r	%	xx0x0x	
SRA r	%	xx0x0x	
SRL r	%	xx0x0x	
RLD	%	xx0x0-	4-Bit Rotation
RRD	%	xx0x0-	4-Bit Rotation

Bit-Manipulations-Befehle

(Für r kann auch hier (IZ+d) eingesetzt werden.)

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
CCF	CMC	--x-0x	Carry-Flag invertieren
SCF	STC	--0-01	Carry-Flag setzen
BIT b,r	%	xx1x0-	Bit testen
RES b,r	%	------	Bit löschen
SET b,r	%	------	Bit setzen

Block-Such-Befehle

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
CPD	%	xxxx1-	Registerbelegung:
CPD R	%	xxxx1-	A=Suchwert
CPI	%	xxxx1-	HL=adr(Suchblock)
CPI R	%	xxxx1-	BC=Blocklänge

16-bit-Arithmetik

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
ADC HL,ss	%	xxxx0x	
ADD HL,ss	DAD rp	--x-0x	
ADD IZ,ss	%	--x-0x	
SBC HL,ss	%	xxxx1x	
DEC ss	DCX rp	------	
DEC IZ	%	------	
INC ss	INX	------	
INC IZ	%	------	

Ein- und Ausgabe-Befehle

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
IN A,(n)	IN	------	
IN r,(C)	%	xx0x0-	
OUT (n),A	OUT	------	
OUT (C),r	%	------	

Blockbefehle für die Ein- und Ausgabe

(Wegen der ungewöhnlichen I/O-Bausteinadressierung können diese Befehle in den CPCs nicht verwendet werden.)

Z80-Mnemonic	8080-Mnemonic	SZHPNC	Bemerkung
IND	%	xxxx1-	Registerbelegung:
IND R	%	x1xx1-	HL=adr(Zielblock)
INI	%	xxxx1-	C=Port-Nr.
INI R	%	x1xx1-	B=Blocklänge
OUT D	%	xxxx1-	Registerbelegung:
OTD R	%	x1xx1-	HL=adr(Quellblock)
OUT I	%	xxxx1-	C=Port-Nr.
OTI R	%	x1xx1-	B=Blocklänge

Sprung- und Unterprogramm-Befehle

(Diese Befehle beeinflußt nicht die Flags.)

Z80-Mnemonic	8080-Mnemonic	Bemerkung
CALL nn	CALL	Unterprogrammaufruf
CALL cc,nn	C cc	dto., bedingt
DJNZ e	%	= DEC B ! JRNZ e
JP (HL)	PCHL	Berechenbarer Sprung
JP (IX)	%	Berechenbarer Sprung
JP (IY)	%	Berechenbarer Sprung
JP nn	JMP	Sprung zur Adresse nn
JP cc,nn	J cc	dto., bedingt
JR e	%	Sprung zu (PC)+e
JR c,e	%	dto., bedingt
RET	RET	Unterprogramm-Rücksprung
RET cc	R cc	dto., bedingt
RETI	%	dto. aus INTerrupt-Subroutine
RETN	%	dto. aus NMI-Subroutine
RST p	RST	Unterprogrammaufruf

Diverse Steuerbefehle

Z80-Mnemonic	8080-Mnemonic	Bemerkung
DI	DI	Disable Interrupt
EI	EI	Enable Interrupt
HALT	HLT	hält die CPU an
IM 0	%	Interrupt-Modus 0
IM 1	%	Interrupt-Modus 1
IM 2	%	Interrupt-Modus 2
NOP	NOP	no operation

9. Anhang

9.1 ASCII-Code

Dez.	Hex.	ASCII	CTRL
0	00	NUL	@
1	01	SOH	A
2	02	STX	B
3	03	ETX	C
4	04	EOT	D
5	05	ENQ	E
6	06	ACK	F
7	07	BEL	G
8	08	BS	H
9	09	HT	I
10	0A	LF	J
11	0B	VT	K
12	0C	FF	L
13	0D	CR	M
14	0E	SO	N
15	0F	SI	O
16	10	DLE	P
17	11	DC1	Q
18	12	DC2	R
19	13	DC3	S
20	14	DC4	T
21	15	NAK	U
22	16	SYN	V
23	17	ETB	W
24	18	CAN	X
25	19	EM	Y
26	1A	SUB	Z
27	1B	ESC	[
28	1C	FS	\
29	1D	GS	]
30	1E	RS	^
31	1F	US	_

Steuerzeichen

Dez.	Hex.	ASCII
32	20	Space
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Zeichen u. Ziffern

Die Steuerzeichen werden auch oft durch einen sogenannten "Control-Code" angegeben. Zum Beispiel läßt sich das Steuerzeichen "BEL" durch den Code CTRL-G ("Control-Taste" gleichzeitig mit der Taste G) ausgeben.

Dez.	Hex.	ASCII
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Großbuchstaben

Dez.	Hex.	ASCII
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	DEL

Kleinbuchstaben

9.2 Themenverzeichnis

9.3 Stichwortverzeichnis

ISBN 3-89011-406-7

DM 19,80
SFr 19
ÖS 154