veb mikroelektronik „wilhelm pieck" mühlhausen
im veb kombinat mikroelektronik

Redaktionsschluß der vorliegenden Ausgabe: Dezember 1988

0 Präambel

Das Erarbeiten von Anwenderprogrammen für ein KC-Floppy-System hat vor allem dann Bedeutung, wenn nicht auf Standardsoftware zurückgegriffen werden kann, die Standardsoftware zu modifizieren (installieren) ist oder Sonderprobleme zu lösen sind. Voraussetzung für den Programmierer sind entsprechende Hardwarekenntnisse sowie Kenntnisse in der Assemblerprogrammierung /1/.

In diesem Handbuch erfolgt die Beschreibung der Hardware des D004 FLOPPY DISK BASIS aus der Sicht des Programmierers. Die Erläuterung der Systemgrundlagen erfolgt getrennt in zwei Teilen für die beiden möglichen Betriebsarten.

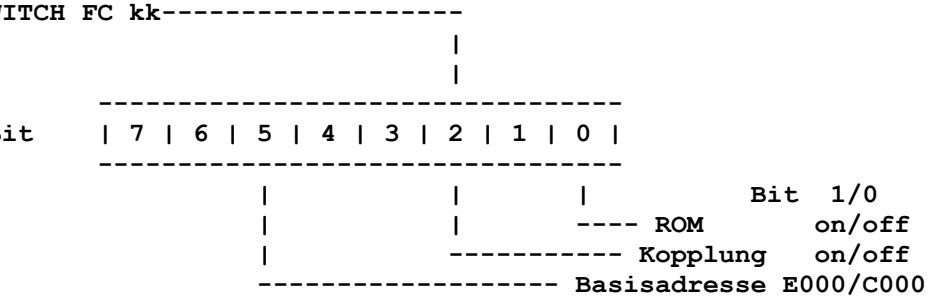
Im Teil A erfolgt die Beschreibung der Komponenten des MicroDOS zur Nutzung in Maschinenprogrammen /2/ bzw. in höheren Programmiersprachen /3/, /4/, /5/, /6/.

Im Teil B zur CAOS-Betriebsart werden Hinweise zu deren Nutzung in Anwenderprogrammen und zum Aufbau der Diskettendateien gegeben. Hierbei sei besonders auf den Modul M027 DEVELOPMENT /7/ verwiesen.

1. Erweiterungen im KC-System

1.1. Einführung

Das KC-Floppy-System enthält mehrere Hardware-Komponenten, die der Erweiterung des KC-Grundgerätes entsprechend dem Modulkonzept des KC-Systems dienen. Die folgende Beschreibung ist eine Ergänzung zum Abschnitt 2.1. des Manuals. Die Moduladresse der Erweiterung ist 0FCH. Beim SWITCH-Kommando gelten folgende Bitbedeutungen:



1.2. ROM-Erweiterung

Der interne ROM von 8 KByte enthält Routinen zum Systemstart und zum BIOS der PC-Betriebsart bzw. Start der CAOS-Betriebsart. Er wird nur zum Start der entsprechenden Betriebsart benötigt. Weiterhin sind zwei ergänzende Zusatzfunktionen enthalten (siehe Abschnitte 3 und 4). Über Bit 5 des Steuerbytes beim SWITCH-Kommando kann der ROM wahlweise auf die Basisadresse C000H oder E000H geschaltet werden.

1.3. Koppel-RAM

Der Zugriff auf den Koppel-RAM von 1 KByte zum Datenaustausch mit dem Prozessorsystem im FLOPPY DISK BASIS wird über Bit 2 des Steuerbytes gesteuert. Ist der Koppel-RAM online geschaltet, so leuchtet die Connection-LED des FLOPPY DISK BASIS und das KC-Grundgerät kann auf den Koppel-RAM zugreifen. Der Koppel-RAM liegt nicht im Speicher- sondern im I/O-Adreßbereich des Grundgerätes.

Der Koppel-RAM belegt vier I/O-Adressen im I/O-Adreßbereich des Prozessors. Jeder dieser vier Adressen ist ein Speicherblock von 256 Byte zugeordnet. Der Zugriff innerhalb des Datenblockes erfolgt über den höherwertigen Teil des Adreßbusses. Dazu dienen die E/A-Befehle

```
OUT (C),r
```

und

```
IN r,(C)
```

bzw. die Block-E/A-Befehle. Das Register C enthält hierbei den niederwertigen Teil der Adresse (Blocknummer 0-3) und das B-Register die Adresse im Block. Die I/O-Adressen sind: F0H bis F3H.

1.4. Koppel-Steuerung

Zur Steuerung des Prozessorsystems im FLOPPY DISK BASIS durch den Prozessor im KC-Grungerät enthält das FLOPPY DISK BASIS eine spezielle Koppel-Steuerung, deren Zugriff über das gleiche Bit beim SWITCH- bzw. JUMP-Kommando wie der Koppel-RAM gesteuert wird.

Die Koppel-Steuerung besteht KC-seitig aus einem 4-Bit-Latch, auf das über die I/O-Adresse F4H zugegriffen werden kann. Es kann vom KC-Grundgerät eine Steuerung der RESET- und der NMI-Leitung des Prozessors im FLOPPY DISK BASIS erfolgen. Die Bits des Koppelregisters haben folgende Bedeutungen:

```
BIT   7 6 5 4 3 2 1 0
              | | | |
              | | |  - Freigabe des Prozessors
              | |  --- Setzen des Prozessors in Dauer-RESET
              |  ----- Ausgabe RESET-Impuls
               ------- Ausgabe NMI-Impuls
```

Nach dem Einschalten des FLOPPY DISK BASIS befindet sich der Prozessor im RESET. Alle Bits des Koppel-Registers sind LOW. Alle Aktivitäten werden durch '1'-Impulse ausgelöst. Dazu wird in aufeinanderfolgenden Befehlen erst eine '1' und danach wieder eine '0' ausgegeben.

Durch Ausgabe eines '1'-Impulses über Bit 0 wird der RESET-Zustand des Prozessors aufgehoben. Eine Bussteuerung gibt auf den Datenbus 00 (= NOP-Befehl) aus. Gleichzeitig werden die Nullen in den Speicher geschrieben. Dies geschieht, bis der Prozessor die Adresse FC00H erreicht. Ab dieser Adresse erfolgt die normale Befehlsabarbeitung. Mit Aufhebung des RESET-Zustandes wird die SYSTEM-LED eingeschaltet. Sie zeigt dem Programmierer, daß das Prozessorsystem im FLOPPY DISK BASIS arbeitet.

Durch Ausgabe einer '1' über Bit 1 kann der Prozessor in den Dauer-RESET geschaltet werden. Dieser kann nur wie oben beschrieben aufgehoben werden und hat stets ein Löschen des Speichers von 0 bis FBFFH zur Folge.

Über Bit 2 ist das Auslösen eines RESET-Impulses ohne Löschen des Speichers möglich.

Über Bit 3 kann ein NMI-Impuls für den Prozessor ausgelöst werden.

2. Prozessorsystem im D004 FLOPPY DISK BASIS

2.1. Speicheraufteilung

Der Speicherbereich des Prozessorsystems im FLOPPY DISK BASIS ist ausschließlich aus Schreib-Lese-Speichern aufgebaut. Der Bereich von 0 bis FC00H ist ein dynamischer RAM. Der Bereich von FC00H bis FFFFH ist ein statischer RAM, auf welchen beide Prozessoren zugreifen können. Es gilt folgende Adreßzuordnung:

KC-I/O-Adresse HIGH	KC-I/O-Adresse LOW	D004-Adresse
00	F0	FC00
FF	F0	FCFF
00	F1	FD00
.....		
FF	F3	FFFF
(B)	(C)	- Register beim I/O-Zugriff

2.2. I/O-Adressen

Im Prozessorsystem des FLOPPY DISK BASIS sind 16 I/O-Adressen belegt. Sie dienen der Prozessorankopplung des Floppy Disk Controllers (FDC) /14/ und des Counter Timer Circuits (CTC) /15/. Es gilt folgende Adreßzuordnung:

I/O-Adresse	Bedeutung
F0H	CS-FDC (Chipselect)
F2H	DAK-FDC (DMA-Acknowledge)
F4H	Input-Gate
F6H	Select-Latch
F8H	TC-FDC (Terminalcount)
FAH	nicht verwendet
FCH - FFH	CTC (Kanal 0 - 3)

Das Input-Gate gestattet dem Prozessor das Lesen folgender Signale der FD-Steuerung über den Datenbus:

Datenbit	Signal
7	DRQ (DMA-Request)
6	INT (Interrupt)
5	RDY (Drive-Ready)
4	IDX (Index - Spuranfang)

Alle vier Kanäle des CTC sind hardwaremäßig kaskadiert. Die Zählereingangsfrequenz des Kanals 0 beträgt 500kHz. Der Kanal 3 ist vom Betriebssystem interruptmäßig zur Laufwerksteuerung belegt. Die Kanäle 0 und 1 werden von der Systemuhr verwendet. Kanal 1 arbeitet dabei interruptgesteuert.

Die Adressen der Interruptserviceroutinen sind folgender Tabelle zu entnehmen:

CTC-Kanal	Adresse
0	FBE0
1	FBE2
2	FBE4
3	FBE6

2.3. Floppy-Disk-Interface

Der externe Anschluß für Diskettenlaufwerke mit der Bezeichnung "FD - INTERFACE" hat die in folgender Tafel aufgeführte Belegung.

Anschluß	Signal	Bedeutung	
A1	/FLT	Fault - Fehler	n.c. (Eingang)
A2	SE3	Select Drive 3 - Auswahl	Ausgang
A3	/IX	Index - Spuranfang	Eingang
A4	SE0	Select Drive 0 - Auswahl	Ausgang
A5	SE2	Select Drive 2 - Auswahl	Ausgang
A6	GND		Masse
A7	/ST	Step - Schritt	Ausgang
A8	/WD	Write Data - Schreibdaten	Ausgang
A9	/WE	Write Enable - Schreibbefehl	Ausgang
A10	GND		Masse
A11	GND		Masse
A12	GND		Masse
A13	/RDY	Ready - Bereitschaftsmeldung	Eingang
Anschluß	Signal	Bedeutung	
B1	/FR	Fault Reset - Fehler rücksetzen	n.c. (Ausgang)
B2	GND		Masse
B3	GND		Masse
B4			n.c. (Ausgang)
B5	SE1	Select Drive 1 - Auswahl	Ausgang
B6	/TS	Two Side - zweiseitig	Eingang
B7	/SD	Step Direktion - Schrittrichtung	Ausgang
B8	GND		Masse
B9	GND		Masse
B10	/TR0	Track 0 - Spur 0	Eingang
B11	/WP	Write Protekted - Schreibschutz	Eingang
B12	/RD	Read Data	Eingang
B13	/SS	Side Select - Kopfauswahl	Ausgang

n.c. (not connected):
Das D004 FLOPPY DISK DRIVE benutzt diese Signale nicht.

Der Ausgang 'FD INTERFACE OUT' des D004 FLOPPY DISK DRIVE besitzt die gleiche Anschlußbelegung wie das Basisgerät mit einem Unterschied. Die vier Auswahlsignale SE0 bis SE3 werden im FLOPPY DISK DRIVE zyklisch getauscht, damit jedes weitere angeschlossene Laufwerk mit der gleichen internen Codierung auf SE1 betrieben werden kann.

Es erfolgt die Umcodierung in folgender Weise:

Eingang	Ausgang
SE0	SE3
SE1	SE2
SE2	SE1
SE3	SE0

Damit ist der Anschluß von maximal vier Minifolienspeicherlaufwerken für MFM-Aufzeichnung unter Beachtung der Anschlußbedingungen möglich.

3. Systemcheck

Der ROM im FLOPPY DISK BASIS enthält ein Systemcheckprogramm. Der Aufruf erfolgt mit 'JUMP FC FF'. Der Systemcheck liefert als erstes die Ausgabe der Versionsnummer des ROM sowie dessen Erstellungsdatum. Anschließend werden folgende Tests abgearbeitet:

- Bildung einer Prüfsumme über den ROM
- Schreib-, Lese- und Verkopplungstests im Koppel-RAM
- Programmabarbeitung im Koppel-RAM
- Programmabarbeitung im dRAM

Verlaufen die Tests erfolgreich, so wird 'O.K.' ausgegeben. Im Fehlerfall erfolgt eine Fehlermitteilung als Hinweis für den Service.

4. V24-Ladefunktion

Mit dem Aufruf 'JUMP FC nn' wird ein V24-Ladeprogramm aufgerufen. Aufgabe dieses Programmaufrufes soll es sein, Programme von einem anderen Computer in den KC zu laden, ohne auf die Kassette oder Diskette zugreifen zu müssen. Beim KC 85/4 ist eine vergleichbare Funktion bereits Bestandteil des Betriebssystems.

Die Laderoutine wird beim Aufruf auf den Bereich ab BC00H kopiert und der CAOS-ROM wird wieder zugeschaltet. Danach wird der erste V24-Modul M003 in der Modulkette aktiviert bzw., wenn dieser nicht vorhanden ist, mit Fehlermeldung abgebrochen. Die Datenübertragung erfolgt über den Kanal 2 (rechts). Die Sende- und die Empfangsroutine werden auf die USER-Kanäle 1 eingetragen.

Die Übertragungsrate wird mittels Parameter nn festgelegt. Der Parameter nn kann im Bereich zwischen 1 und FEH liegen (vgl. Systemcheck im Abschnitt 3 und Wiederstart im Handbuch für den Bediener /8/ Teil A Abschitt 1.2.5.). Der Parameter nn bildet die Zeitkonstante für den CTC (vgl. /9/ und /10/). Es gelten folgende Vorzugswerte für nn:

nn	Übertragungsrate	Bemerkung
01	54 KBaud	nur zwischen KC 85
17	2400 Baud	
2E	1200 Baud	

Die Übertragung erfolgt mit acht Bit, ohne Paritätsprüfung und mit zwei Stoppbits. Im Gegensatz zu den Duplextreiberroutinen der Kassette C 0171 /10/ arbeitet die Empfangsroutine ohne Protokollfunktion, d. h. die Daten werden mit maximaler Geschwindigkeit direkt in den Speicher übernommen. Die erwarteten Daten müssen folgendem Format entsprechen:

Byte	Bedeutung
1	Ladeadresse LOW
2	Ladeadresse HIGH
3	Länge LOW
4	Länge HIGH
5	Startadresse LOW
6	Startadresse HIGH
ab 7	Datenbytes entsprechend Länge

Die Empfangsroutine kann mit der <BRK>-Taste abgebrochen werden

SOFTWARE

TEIL A - PC-BETRIEBSART

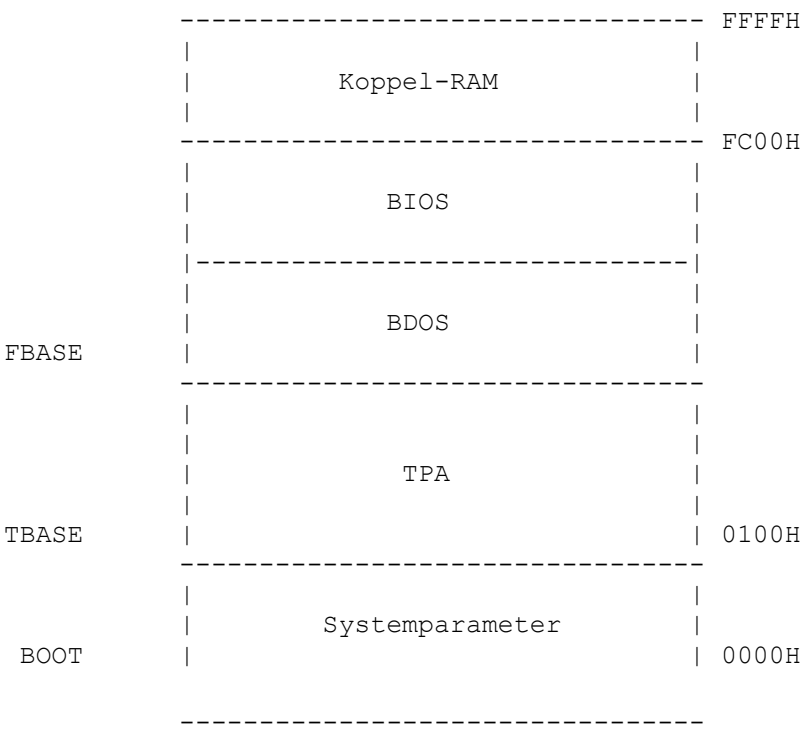
1. Einführung

Dieser Teil des Handbuchs beschreibt die Systemorganisation, die Speicherstruktur und die Eintrittspunkte des Betriebssystems MicroDOS. Es werden die notwendigen Informationen gegeben, um Programme zu schreiben, welche unter MicroDOS laufen.

MicroDOS ist logisch in drei Teile aufgeteilt, in das Basis-Ein/Ausgabe-System (BIOS), das Basis-Platten-Betriebssystem (BDOS), welches auch den Kommandoprozessor enthält, und das Feld für zeitweilig geladene Programmme (TPA).

Das BIOS ist ein hardwareabhängiger Modul, welcher die Anbindung von peripheren Geräten auf einem niedrigen Niveau definiert. Das BDOS gewährleistet einen benutzerorientierten Zugriff zum Massenspeicher und den anderen peripheren Geräten. Als Besonderheit ist hierbei zu beachten, daß zum BIOS gehörende Programmteile durch den Prozessor des KC-Grundgerätes abgearbeitet werden.

Der TPA ist der Speicherbereich, in dem verschiedene nichtresidente Teile des Betriebssystems und Anwenderprogramme ausgeführt werden. Die unteren 256 Byte des Speichers sind für Systemparameter reserviert. Der Speicher im Prozessorsystem des FLOPPY DISK BASIS ist wie folgt aufgeteilt:



Die exakte Adresse von FBASE ist abhängig von der konkreten Version des MicroDOS. Der Maschinencode auf der Speicherzelle BOOT führt einen System-Warmstart durch, wobei alle Programme und Variablen geladen bzw. initialisiert werden, welche zur Übergabe der Steuerung an den Kommandoprozessor notwendig sind.

Transiente Programme brauchen deshalb nur zur Position BOOT springen, um zum Kommandoniveau von MicroDOS zurückzukehren. Der prinzipielle Eintrittspunkt zum BDOS liegt bei BOOT + 0005H, wo sich ein Sprung nach FBASE befindet. Die Zelle BOOT + 0006H enthält den Wert von FBASE und kann zur Bestimmung des verfügbaren Speichers benutzt werden.

2. Systemstart

Als Systemstart beim MicroDOS wird das Laden des Betriebssystems in den oben dargestellten Speicher und die Übergabe der Steuerung an den Kommandointerpreter bezeichnet. Dabei laufen in beiden Prozessorsystemen mehrere Prozesse zum Teil parallel ab.

2.1. Vorgänge im KC-System

Beim 'JUMP FC' wird der CAOS-ROM abgeschaltet und das Programm im ROM des FLOPPY DISK BASIS gestartet. Dieses löst folgende Aktivitäten aus:

- Löschen des Koppel-RAM
- Kopieren des Urladeprogrammes in den Koppel-RAM
- Freigabe des Prozessors im FLOPPY DISK BASIS
- Warten auf Quittung im Koppel-RAM; bei negativer Quittung Fehlermeldung und Abbruch; bei positiver Quittung Programmfortsetzung und Entscheidung, ob PC- oder CAOS-Betriebsart vorliegt
- Erzeugen einer Hilfsadreßtabelle zur Bildschirmarbeit
- Feststellung, ob KC 85/2 bzw. /3 oder KC 85/4 vorliegt
- Aufbau der RAM-Floppy-Verwaltungstabelle
- Umkopieren der Gerätetreiberprogrammschleife
- Übergang in die Kommandoabarbeitung über den Koppel-RAM

2.2. Initialisierung des MicroDOS

Die Vorgänge im Prozessorsystem des FLOPPY DISK BASIS beginnen mit der Freigabe des RESET. Der Prozessor löscht als erstes den dRAM und beginnt mit der Abarbeitung des Urladeprogrammes ab Adresse FC00H. Der Urlader lädt die ersten 512 Byte einer Diskette im Format 5*1024 oder 16*256 vom Laufwerk 0 in den dRAM ab Adresse 9000H.

Danach werden die ersten vier Byte überprüft, ob sie das Systemkenzeichen 'SYSL' enthalten. Stimmen nur die ersten zwei Byte überein, so wird eine CAOS-Betriebsartdiskette angenommen. Stimmen alle vier Byte, wird dem KC die PC-Betriebsart gemeldet. In allen anderen Fällen wird ein Fehler übermittelt.

Diese 512 Byte stellen den BOOT-Lader für das MicroDOS dar. Dessen Aufgabe ist es, das Betriebssystem aus den zwei Systemspuren der Diskette in den Bereich ab FBASE zu laden und zu starten. Dazu ist vor dem BDOS noch ein Grundinitialisierungsprogramm enthalten, welches nach dem erfolgreichen Systemstart im TPA wieder überschrieben werden kann. Dieses überträgt unter anderem auch mit MSYSG installierte Drucker- und Kopplungstreiber in den KC.

3. Systemdatenstrukturen

Im folgenden werden einige grundlegende Datenstrukturen des Betriebssystems erläutert.

3.1. Systemparameter

Auf den ersten 256 Bytes des Speichers befinden sich folgende vom System benutzte Datenfelder, welche nachfolgend beschrieben werden.

Adresse	Bedeutung
0000H	Sprung zum Warmstart WBOOT
0003H	I/O-Byte
0004H	aktuelles Laufwerk
0005H	Sprung zum BDOS
0040H-0042H	Systemuhr
005CH	erster Dateisteuerblock
006CH	zweiter Dateisteuerblock
0080H	Standard-DMA-Puffer

Der Sprung auf Adresse 0000H führt direkt zum entsprechenden Ansprung in der BIOS-Sprungtabelle. Es wird der Warmstart durchgeführt und die Kontrolle an den Kommandoprozessor übergeben. Durch die Sprungadresse kann die Adresse des CP/M-kompatiblen BIOS-Sprungvektors ermittelt werden.

Das I/O-Byte dient der Aufspaltung der logischen Kanäle auf verschiedene physische Geräte und wird im System MicroDOS im KC verwaltet.

Die Nummer des aktuellen Laufwerks wird aus Kompatibilitätsgründen zu CP/M 2.2 auf der Adresse 0004H abgelegt.

Auf Adresse 0005H befindet sich der allgemeine Eintrittspunkt in das BDOS. Gleichzeitig gibt die Sprungadresse den ersten für das System reservierten Speicherplatz oberhalb des TPA an.

Ab Adresse 005CH werden vom CCP, entsprechend den im Kommando angegebenen Parametern, ein oder zwei Dateisteuerblöcke angelegt (siehe auch Abschnitt 3.2.). Für die Benutzung im Anwenderprogramm muß der zweite Dateisteuerblock auf einen anderen Speicherplatz verschoben werden, da dieser sonst vom BDOS überschrieben würde.

Auf Adresse 0080H beginnt der Standard-DMA-Puffer, welcher vom System für die Diskettenarbeit benutzt wird. Beim Aufruf eines Kommandos wird hier zusätzlich die Kommandozeile abgelegt.

3.2. Der Dateisteuerblock

Der Dateisteuerblock ist eine Datenstruktur, die vom Dateisystem beim Zugriff auf die Dateien über das Verzeichnis verwendet wird. Alle Operationen mit Dateien benötigen als Ausgangsinformation diese Daten. Außerdem belegen die Funktionen des wahlfreien Zugriffs die drei Byte hinter dem Dateisteuerblock. Beim Aufruf der Dateifunktionen adressiert das Registerpaar DE den Dateisteuerblock für die interessierende Datei.
Der Dateisteuerblock (FCB) besteht aus einem Feld von 33 Byte im Fall des sequentiellen Zugriffs und einer Folge von 36 Byte bei wahlfreiem Zugriff.

Der Standard-FCB auf Adresse 005CH kann für wahlfreien Zugriff benutzt werden, wenn die drei Bytes ab 007DH für diesen Zweck zur Verfügung stehen. Im folgenden ist das Format des Dateisteuerblocks aufgezeigt:

dr	f1	f2	---	f8	t1	t2	t3	ex	s1	s2	rc	d0	---	dn	cr	r0	r1	r2
00	01	02		08	09	10	11	12	13	14	15	16	---	31	32	33	34	35

mit:

dr	Laufwerkcode 0-16 0 - momentan angewähltes Laufwerk 1 - Laufwerk A 2 - Laufwerk B usw.
f1...f8	Dateiname in ASCII-Großbuchstaben Bit 7=0
t1...t3	Dateityp in ASCII-Großbuchstaben Bit 7=0 t1: Bit 7=1 - schreibgeschützte Datei t2: Bit 7=1 - Systemdatei
ex	momentane Erweiterungsnummer, normalerweise = 0, bei Ein/Ausgabe im Bereich 0 - 31
s1	für interne Systembenutzung reserviert
s2	für interne Systembenutzung reserviert, = 0 bei Datei eröffnen, kreieren und suchen
rc	Datensatzanzahl (von 0 - 128)
d0...dn	wird vom Betriebssystem ausgefüllt
cr	Datensatzzähler für sequentiellen Zugriff, wird vom Benutzer auf 0 gesetzt
r0...r2	Nummer für wahlfreien Datensatz von 0 - 65535 mit Überlauf in r2, niederwertiges Byte in r0, höherwertiges Byte in r1

Jede Datei, die mit Hilfe von MicroDOS benutzt wird, muß einen dazugehörigen FCB haben, welcher den Dateinamen und eine Sektorbelegung für alle folgenden Operationen bereitstellt. Wenn auf eine Datei zugegriffen werden soll, ist es Sache des Programmierers, die niederen 16 Byte des FCB auszufüllen und das "cr"-Feld zu initialisieren. Normalerweise werden Byte 1-11 mit ASCII-Zeichen für Dateiname und Dateityp belegt, während alle anderen Bytes mit 0 gefüllt werden.

Die Dateisteuerblöcke werden im Inhaltsverzeichnis der Diskette gespeichert und vor der Durchführung einer Dateioperation in den Hauptspeicher gebracht (siehe BDOS-Funktionen Datei eröffnen bzw. erzeugen im Abschnitt 6.3.). Die Speicherkopie des Dateisteuerblocks wird während der Dateioperationen korrigiert und nach Abschluß derselben auf der Diskette abgespeichert (siehe Funktion Datei schließen).

3.3. Der Standardpuffer

Der Standardpuffer ab Adresse 80H wird vom System als Zwischenspeicher für Diskettenoperationen genutzt. Auch Anwenderprogramme können diesen Puffer verwenden. Zusätzlich wird vom Kommandointerpreter der Teil der Kommandozeile, welcher auf den Programmnamen folgt, im Standardpuffer abgelegt (siehe Abschnitt 5).

3.4. Der Systemsteuerblock

Der Systemsteuerblock ist eine Datenstruktur, die sich im Basisdiskettenbetriebssystem befindet. MicroDOS nutzt diesen Bereich für die Sicherung der Systemdaten und Verbindungen von BDOS und BIOS. Die Anwenderprogramme können auch einige Parameter, die sich in diesem Bereich befinden, nutzen. Programme, die den Systemsteuerblock nutzen, können nur mit MicroDOS oder SCP bzw. CP/M Version 3.0 arbeiten.

Im folgenden sind die Felder des Systemsteuerblocks aufgeführt, die den Nutzerprogrammen zugänglich sind. Die Position jedes Feldes wird durch den Versatz bezüglich des Systemsteuerblockanfangs angegeben.

00 - 04	reserviert
05	Versionsnummer
06 - 09	Anwenderflags
0A - 0F	reserviert
10 - 11	Rückkehrcode
12 - 19	reserviert
1A	Konsolenbreite
1B	aktuelle Spalte des Cursors
1C	Seitenlänge
1D - 21	reserviert
22 - 23	Zuordnungsvektor für Konsoleneingabe
24 - 25	Zuordnungsvektor für Konsolenausgabe
26 - 27	Zuordnungsvektor für Zusatzeingabe
28 - 29	Zuordnungsvektor für Zusatzausgabe
2A - 2B	Zuordnungsvektor für Druckerausgabe
2C	Seitenmodus
2D	reserviert
2E	Backspace (^H) aktiv (0:Zeichen wird im Puffer und auf Bildschirm gelöscht; 1: Zeichen wird nochmals auf Bildschirm ausgegeben)
2F	Rubout (DEL) aktiv (Bedeutung wie bei Backspace)
30 - 34	reserviert
35 - 36	Pufferadresse für Kalt- und Warmstart
37	Begrenzer für Zeichenkette (Standard: $ = 24H)
38	Druckerflag
39 - 3B	reserviert
3C - 3D	aktuelle DMA-Adresse
3E	aktuelles Laufwerk
3F - 43	reserviert
44	aktuelle Nutzernummer
45 - 49	reserviert
4A	Multisektorzähler (= 1)
4B	BDOS-Fehlermodus (vgl. BDOS-Funktion 45)
4C - 4F	Laufwerke für Dateisuche
50	Laufwerk für temporäre Dateien
51	Laufwerknummer für Fehleranzeige

54	Flag für Diskettenwechsel
52 - 56	reserviert
57	Flag für Länge der Fehlermeldung
58 - 59	Datum (nicht verwendet)
5A - 5C	Zeit (nicht verwendet)
5D - 5E	Adresse des gemeinsamen Speicherbereichs
5F - 61	Sprung zur Fehlermeldung
62 - 63	Adresse des TPA-Endes (wird auf 6, 7 kopiert)

4. Logische Geräte

4.1. Laufwerke

Die auf dem FLOPPY DISK BASIS implementierte Version des MicroDOS verwaltet maximal acht logische Laufwerke mit den Bezeichnungen von A...H.

Laufwerk A ist stets das RAM-Floppy, dessen Kapazität vom Grundgerät und den im Gesamtsystem gesteckten RAM-Modulen abhängt. Laufwerk B ist beim Start das Systemlaufwerk. Es liegt immer auf dem physischen Laufwerk 0 und hat das Format 5*1024*80*2 mit zwei Systemspuren. Die Laufwerke C...H können beliebig installiert werden.

4.2. Konsole und Drucker

Als Konsole werden die Bildschirmausgabe und die Tastatureingabe bezeichnet. Die Standardbildschirmausgabe (vgl. Abschnitt 7.3.) erfolgt wahlweise mit 24 Zeilen zu 80 Zeichen oder 32 Zeilen zu 40 Zeichen (CAOS-Bildschirmformat). Anlage 2 enthält die auch im SCP bzw. CP/M üblichen Steuerzeichen. Darüber hinaus enthält die Konsolenschnittstelle eine Vielzahl zusätzlicher Funktionen zur optimalen Nutzung der Möglichkeiten des KC-Systems.

Zu diesen Funktionen gehören unter anderem die Grafik, die Farbsteuerung, die Tonausgabe und die Modulverwaltung. Alle diese Funktionen werden über ESCape-Folgen gesteuert. Durch Ausgabe des ESCape-Codes und eines Unterscheidungscodes wird die Konsolenausgabe auf die Sonderfunktion umgeschaltet. Anschließend wird ggf. eine der Funktion entsprechende Anzahl von Parametern übertragen. Anlage 3 gibt einen Überblick über die ESCape-Funktionen. In den Programmbeispielen in Anlage 6 wird ebenfalls darauf zurückgegriffen.

Bei Benutzung der ESCape-Funktionen (außer der auch im SCP üblichen Cursorpositionierung) sollte über die BIOS-Schnittstelle (Abschnitt 7.4.) gearbeitet werden, da die BDOS-Schnittstelle (Abschnitt 6.2.) die Codierung 09H in einer Parameterliste als Tabulator interpretiert und in Leerzeichen (20H) umsetzt.

Als Beispiel sei das Setzen eines Punktes auf die Position X=100, Y=50 angegeben:

```
LD   A,1BH      ;ESCape
CALL BIOSOUT    ;Konsolenausgabe
LD   A,'A'      ;Funktionscode
CALL BIOSOUT
LD   A,100      ;X-Wert low
CALL BIOSOUT
LD   A,0        ;X-Wert high
CALL BIOSOUT
LD   A,50       ;Y-Wert
CALL BIOSOUT
```

Die Tastatur arbeitet mit einer gegenüber dem CAOS auf drei Codes je Taste erweiterten Tastaturcodetabelle (vgl. /11/ bzw. /12/). Eine Umbelegung der Tastencodes ist möglich. Das Programmmbeispiel 6.6. zeigt die Tastencodeveränderung für den Schreibmaschinenmodus. Aus dem Beispiel kann neben der Tastaturcodetabelle auch deren Übertragung in den Speicher des Grundgerätes entnommen werden.

Neben den üblichen Codierungen, welche zum MicroDOS übertragen werden, sind in der Tastaturcodetabelle einige Codes möglich, die direkt ausgeführt werden. Aus der folgenden Zusammenstellumg sind diese Codes zu entnehmen:

Codes	Bedeutung
82H	Cursor on (unter Umgehung von MicroDOS)
83H	Cursor off (unter Umgehung von MicroDOS)
84H	Autorepeat on/off
85H	PAGE-/SCROLLmodus
86H	Hintergrundfarbe (erwartet Hexziffer)
87H	Aufruf Hardcopy (Unterprogrammadresse auf B799H)
88H	Vordergrundfarbe (erwartet Hexziffer)
89H	Umschaltung 80/40-Zeichen mit Bildschirmlöschen
8AH	Tastenclick on/off
8BH	Umschaltung erste Tastaturebene groß/klein
8CH	Umschaltung Zeichensatz amerikanisch/deutsch
8DH	RESET im FLOPPY DISK BASIS (Taste zweimal hintereinander betätigt / wird im MicroDOS nicht verwendet!)
8EH	NMI-Auslösung im FLOPPY DISK BASIS (wird von einigen Debuggern unterstützt)
8FH	Hex-Input (erwartet zwei Hexziffern und wird wie eine Tastenbetätigung interpretiert)
90H	SHIFT LOCK
91H	Control
F1H - FCH	Funktionstasten

Die Verwaltung der Funktionstasten entspricht der im CAOS, wobei nur der Aufruf der Taste <F1> verändert ist (Control-Ebene). Eine Funktionstastenbelegung kann in einem Programm (Assembler oder höhere Programmiersprache) vereinbart und in den Speicherbereich ab B900H in das Grundgerät übertragen werden.

Die Standarddruckerausgabe des MicroDOS arbeitet über einen Druckertreiber im KC-Grundgerät. Dieser wird entweder beim Kaltstart als V24-Minimaltreiber aus dem ROM oder als installierter Treiber aus dem MicroDOS in den RAM des Grundgerätes kopiert und gestartet. Um die Flexibilität des Systems zu gewährleisten, ist es möglich, eigene Druckertreiber im System mittels MSYSG (/8/) zu installieren. Die Treiberroutine muß folgenden Bedingungen genügen:

- Dateityp LST
- Speicherbereich 200H bis 37FH (max. drei Sektoren)
- Einsprungadresse für Initialisierung 200H
- Einsprungadresse für Zeichenausgabe 202H
- Eintragung einer Hardcopy-Ausgabe (optional) auf B799H im CAOS-Arbeitsspeicher

Anlage 6.4. enthält einen CENTRONICS-Treiber für den Modul M001 DIGITAL IN/OUT (/13/) ohne Hardcopy-Funktion.

4.3. Zusatzkanäle

MicroDOS verwaltet außerdem je einen sequentiellen Kanal zum Datensenden und -empfangen. Im FLOPPY DISK BASIS werden beide Kanäle normalerweise einer V24-Duplexschnittstelle zugeordnet. Beim Systemstart wird, wenn vorhanden, der rechte Kanal eines M003 Moduls für eine Duplexübertragung mit DTR-Protokoll bei 1200 Baud initialisiert. Die Installation eines eigenen Koppeltreibers mittels MSYSG ist unter Beachtung folgender Bedingungen möglich:

- Dateityp KOP
- Speicherbereich 380H bis 3FFH (1 Sektor)
- Einsprungadresse für Initialisierung Koppeltreiber 380H
- Einsprungadresse für Zeichenausgabe 382H
- Einsprungadresse für Zeicheneingabe 384H

Anlage 6.5. enthält das Listing des Koppeltreibers V24H12.KOP.

5. Kommandoeingabe

Beim Programmaufruf über die Kommandoeingabe analysiert der Kommandoprozessor die eingegebene Zeile und entscheidet, ob ein residentes Kommando eingegeben wurde oder ein Programm von Diskette geladen werden muß.

Der Kommandoprozessor konstruiert die ersten sechzehn Bytes von zwei möglichen Dateisteuerblöcken für ein transientes Programm durch Prüfung des Teils der Kommandozeile, welcher nach dem Namen des transienten Programms folgt. Dabei werden nicht spezifizierte Felder mit Leerzeichen gefüllt. Der erste Dateisteuerblock wird ab Adresse 005CH aufgebaut und kann in dieser Form für nachfolgende Dateioperationen genutzt werden.

Der Anfang des zweiten Dateisteuerblocks liegt im "d0...dn"-Feld des ersten und muß vor der Benutzung in einen anderen Teil des Speichers umgeladen werden. Wenn z. B. der Benutzer folgende Zeile eingibt:

```
PROGNAME B:DATEI1.XXX DATEI2.YYY
```

so wird die Datei "PROGNAME" in den TPA geladen und der Standard-Dateisteuerblock ab Adresse 005CH mit dem Laufwerkcode 2, dem Dateinamen "DATEI1" und dem Dateityp "XXX" initialisiert.

Der zweite Laufwerkcode bekommt den Wert 0 zugewiesen, welcher ab Adresse 006CH geladen wird. Der Dateiname "DATEI2" wird ab Adresse 006DH abgelegt und der Dateityp "YYY" wird acht Byte dahinter (Adresse 0075H) angeordnet.

Alle restlichen Felder bis "cr" werden auf Null gesetzt. Es muß nochmals darauf hingewiesen werden, daß es dem Programmierer obliegt den zweiten Dateinamen und Dateityp in einen anderen Speicherbereich zu verschieben, bevor die Datei mit dem Steuerblock auf 005CH eröffnet werden kann, angesichts der Tatsache, daß die Eröffnungsfunktion den zweiten Dateinamen und -typ anderenfalls überschreibt.

Falls keine Dateinamen in dem Kommando angegeben wurden, so enthalten die Felder ab 005CH und 006CH Leerzeichen. In jedem Fall werden Kleinbuchstaben in Großbuchstaben umgewandelt, um den MicroDOS-Dateinamenvereinbarungen zu entsprechen. Für das oben angeführte Kommando würde der Puffer folgenden Inhalt aufweisen:

```
00  01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
18     B  :  D  A  T  E  I  1  .  X  X  X     D  A  T  E  I  2  .  Y  Y  Y
```

wobei das erste Byte die Anzahl der gültigen Zeichen angibt und dann folgen die Zeichen selbst, welche vom Kommandoprozessor gegebenenfalls in Großbuchstaben umgesetzt werden. Die Kommandoparameter müssen vom Programm ausgewertet werden, bevor der Puffer durch Diskettenoperationen überschrieben wird.

6. BDOS-Funktionen

6.1. Vorbemerkungen

Das Diskettengrundbetriebssystem (BDOS) stellt den Kern des MicroDOS-Betriebssystems dar. Es unterstützt die Ein- und Ausgabe auf den logischen Geräten (Konsole, Drucker, Zusatzkanäle) sowie die Dateiarbeit auf den Disketten. Über BDOS-Rufe können folgende Aufträge angefordert werden:

- Zeichenorientierte Ein- und Ausgabe
  - Zuordnung der logischen zu den physischen Geräten
  - Zeichen- und Zeichenkettenweise Ein- und Ausgabe über die Konsole
  - Zeichenweiser Datenaustausch über die Zusatzkanäle
  - Zeichenweise Ausgabe über den Druckerkanal

- Arbeit mit den Disketten
  - Dateiverwaltung und -manipulation allgemein
  - sequentieller und direkter Dateizugriff

- Systemverwaltung
  - Initialisierung System
  - Ermittlung des Systemzustandes
  - Zugriff auf Systemparameter
  - direkter Zugriff auf die physische Systemschnittstelle (BIOS)

6.2. Aufruf der BDOS-Funktionen

Im folgenden werden die einzelnen Systemfunktionen mit ihren Eingangs- und Ausgangsparametern beschrieben. Der Aufruf der Systemfunktionen erfolgt durch einen CALL auf die Adresse 0005H, wobei die Funktionsnummer im Register C und notwendige Eingangsparameter im Registerpaar DE übergeben werden. Bei der Rückkehr werden die Ausgangsparameter im Akkumulator und im Registerpaar HL zurückgegeben. Feldeintragungen werden in einem Feld übergeben, dessen Anfang beim Aufruf im Registerpaar DE adressiert wrd.

Es ist zu beachten, daß alle Register vom System verändert werden können. Deshalb ist es angebracht, wichtige Registerinhalte auf den Stack zu retten. Das Betriebssystem benutzt einen eigenen Stapel, so daß das aufrufende Programm nur mit einer Stapel-Ebene durch den BDOS-Aufruf belastet wird.

Bei der Benutzung der BDOS-Funktionen ist zu beachten, daß nur bei den BDOS-Rufen bis einschließlich Nummer 40 volle Kompatibilität zum SCPX gewährleistet ist. Das heißt, daß Programme, die nur diese BDOS-Rufe verwenden, auch unter SCPX und CP/M 2.2 lauffähig sind. Programme, die die anderen BDOS-Rufe nutzen, können u. U. leistungsfähiger sein. Bei ihnen ist Kompatibilität zu CP/M 3.0 und SCP 3.0 (PC 1715W) gegeben.

MicroDOS entspricht einer CP/M-Version 2.6 und liegt damit im Funktionsumfang zwischen SCPX (CP/M 2.2) und SCP 3.0 (CP/M 3.0).

6.3. Liste der BDOS-Funktionen

Die BDOS-Funktionen sind nach den Funktionsnummern geordnet.

Funktion 0: System rücksetzen

Eingangsparameter:
Register C: 00H

Diese Funktion gibt die Steuerung an MicroDOS auf dem Kommando-Niveau zurück (Warmstart). Sie hat den gleichen Effekt wie ein Sprung auf die Adresse BOOT. Vor dem Aufruf der Funktion 0 kann durch Funktion 108 ein Rückkehrcode gesetzt werden.

Funktion 1: Konsoleneingabe

Eingangsparameter:
Register C: 01H
Ausgangsparameter:
Register A: ASCII-Zeichen

Die Konsoleneingabe-Funktion liest das nächste Zeichen von der Konsole in Register A ein. Darstellbare Zeichen und die Steuerzeichen Wagenrücklauf, Zeilenvorschub und Rückwärtsschritt (Ctrl-H) werden zur Konsole zurückgegeben. Tabulatorzeichen (Ctrl-I) werden expandiert auf Spalten mit acht Zeichen. Es wird überprüft, ob Start/Stop der Bildschirmausgabe (Ctrl-S) oder Start/Stop der Druckerausgabe (Ctrl-P) auftreten. Das BDOS kehrt nicht eher zum aufrufenden Programm zurück, bevor nicht ein Zeichen auf der Konsole eingegeben wurde.

Funktion 2: Konsolenausgabe

Eingangsparameter:
Register C: 02H
Register E: ASCII-Zeichen

Das in Register E enthaltene Zeichen wird zur Konsole gesendet. Wie bei Funktion 1 wird die Überprüfung von Start/Stop der Bildschirm- und Druckerausgabe durchgeführt. Ist die parallele Druckerausgabe eingeschaltet, so werden alle Zeichen auch an den Drucker gesendet.

Funktion 3: Zusatzeingabe

Eingangsparameter:
Register C: 03H
Ausgangsparameter:
Register A: ASCII-Zeichen

Die Zusatzeingabe liest ein Zeichen von dem zusätzlichen logischen Kanal in das Register A. Die Steuerung kehrt nicht zurück, bevor nicht ein Zeichen gelesen wurde.

Funktion 4: Zusatzausgabe

Eingangsparameter:
Register C: 04H
Register E: ASCII-Zeichen

Die Zusatzausgabefunktion sendet das im Register E enthaltene Zeichen zum zusätzlichen logischen Kanal.

Funktion 5: Druckerausgabe

Eingangsparameter:
Register C: 05H
Register E: ASCII-Zeichen

Die Druckerausgabe-Funktion sendet das im Register E enthaltene Zeichen zum logischen Drucker.

Funktion 6: direkte Konsolen-Ein/Ausgabe

Eingangsparameter:
Register C: 06H
Register E: 0FFH (Eingabe) oder ASCII-Zeichen (Ausgabe)
Ausgangsparameter:
Register A: Zeichen oder Status

Die direkte Ein/Ausgabe wird von MicroDOS unterstützt für solche speziellen Anwendungen, wo einfache Ein/Ausgabeoperationen gefordert sind. Im allgemeinen sollte die Benutzung dieser Funktion vermieden werden, da die normalen Steuerzeichenfunktionen von MicroDOS (d. h. Crtl-S und Ctrl-P) umgangen werden.

Bei Eintritt in die Funktion 6 enthält Register E entweder den hexadezimalen Wert FFH, was eine Konsoleneingabe kennzeichnet, oder ein ASCII-Zeichen. Wenn der Eingangsparameter FFH ist, so kehrt die Funktion mit A = 0 zurück, falls kein Zeichen von der Konsole bereitsteht. Ansonsten enthält Register A das nächste Eingabezeichen von der Konsole.

Ist der Wert des Registers E nicht gleich FFH, so behandelt die Funktion 6 diesen Wert als gültiges ASCII-Zeichen und sendet dieses zur Konsole.

Funktion 7: Status Zusatzeingabe

Eingangsparameter:
Register C: 07H
Ausgangsparameter:
Register A: Status

Die Funktion 7 realisiert in der auf dem KC-Floppy-System vorliegenden Implementierung des MicroDOS die Abfrage des Zustandes der Druckerausgabe. Wenn freier Platz im Übergabepuffer zum Druckertreiber ist, wird im Akkumulator der Wert 0FFH übergeben, ansonsten der Wert 00H.

Funktion 8: Status Zusatzausgabe

Eingangsparameter:
Register C: 08H
Ausgangsparameter:
Register A: Status

Die Funktion 8 ist in der vorliegenden Implementierung des MicroDOS kurzgeschlossen und übergibt stets im Akkumulator den Wert 00H.

Funktion 9: Zeichenkette ausgeben

Eingangsparameter:
Register C: 09H
Registerpaar DE: Adresse der Zeichen

Die Zeichenkettenausgabefunktion sendet Zeichen aus dem Speicher, beginnend ab der in DE übergebenen Adresse, zur Konsole, bis in der Zeichenkette das im Systemsteuerblock gesetzte Endekennzeichen erkannt wird. Standardmäßig wird '$' als Endekennzeichen angenommen. Tabulatoren werden expandiert, wie in Funktion 2, und Start/Stop-Zeichen für Bildschirmausgabe und Druckerausgabe werden ausgewertet. Bei paralleler Druckerausgabe werden alle Zeichen auf dem Drucker ausgegeben.

Funktion 10: Konsolenpuffer lesen

Eingangsparameter:
Register C: 0AH
Registerpaar DE: Pufferadresse
Ausgangsparameter:
Zeichen von der Konsole im Puffer

Diese Funktion liest eine Zeile von eingegebenen Zeichen in einen Puffer, der durch DE adressiert ist. Die Eingabe wird abgeschlossen, wenn der Eingabepuffer überläuft oder wenn ein RETURN eingegeben wird.

Der Puffer erhält folgende Form:

DE	+0	+1	+2	+3	+4	+5	+6	+7	.....	+n
	mx	nc	c1	c2	c3	c4	c5	c6	.....	??

wobei mx die maximale Anzahl von einzulesenden Zeichen (1 bis 255) und nc die tatsächliche Anzahl von eingelesenen Zeichen (vom BDOS bei Rückkehr gesetzt) ist. Es folgen die eingelesenen Zeichen.

Ist nc kleiner als mx, so folgen dem letzten Zeichen unbestimmte Informationen, welche im obigen Beispiel mit "??" gekennzeichnet sind. Eine Reihe von Steuerzeichen wird während der Eingabe erkannt:

DEL	löscht das letzte Zeichen
Ctrl-C	Neustart von MicroDOS (am Beginn der Zeile)
Ctrl-E	gibt das physikalische Ende der Zeile an
Ctrl-H	Rückwärtsschritt
Ctrl-J	(Zeilenvorschub) beendet die Eingabe
Ctrl-M	(ENTER) beendet die Eingabe
Ctrl-R	schreibt die aktuelle Zeile neu
Ctrl-U	löscht die aktuelle Zeile
Ctrl-X	geht zurück zum Anfang der Zeile
Ctrl-W	wiederholt den letzten Pufferinhalt
Ctrl-^	löscht das letzte Wort

Die Erneuerung des Pufferinhalts bei Eingabe von Ctrl-W erfolgt über den Pufferzähler. Wird dieser im Anwenderprogramm verändert, kann der regenerierte Inhalt falsch sein.

Funktion 11: Konsolenstatus holen

Eingangsparameter:
Register C: 0BH
Ausgangsparameter:
Register A: Konsolenstatus

Die Funktion Konsolenstatus überprüft, ob ein Zeichen auf der Konsole eingegeben wurde. Falls ein Zeichen eingegegben wurde, so wird im Register A ein Wert ungleich 0 übergeben, im allgemeinen der Wert 01H. Ist kein Eingabezeichen vorhanden, kehrt die Funktion mit A = 0 zurück.

Funktion 12: Versionsnummer holen

Eingangsparameter:
Register C: 0CH
Ausgangsparameter:
Registerpaar HL: Versionsnummer

Die Funktion 12 übergibt einen 2-Byte-Wert, wobei H = 00H anzeigt, daß es sich um ein CP/M 2.2-kompatibles System handelt (H = 01H bei MP/M). Im Register L übergibt MicroDOS einen hexadezimalen Wert entsprechend der Versionsnummer des Systems, also z. B. 26H bei der Version 2.6. Die Verwendung dieser Funktion erlaubt das Schreiben von versionsabhängigen Programmen.

Funktion 13: Diskettensystem rücksetzen

Eingangsparameter:
Register C: 0DH

Diese Funktion wird benutzt, um vom Programm aus das Dateisystem in den Anfangszustand zu versetzen. Dabei sind alle Laufwerke für Schreiben und Lesen zugelassen (siehe Funktionen 28 und 29), Laufwerk A ist angewählt und die DMA-Adresse ist auf den Standardwert 0080H festgelegt.

Diese Funktion kann von Programmen verwendet werden, welche während der Abarbeitung einen Diskettenwechsel erfordern. Siehe dazu auch Funktion 37.

Funktion 14: Laufwerk anwählen

Eingangsparameter:
- Register C: 0EH
- Register E: Laufwerkcode

Die Funktion kennzeichnet das in E bezeichnete Laufwerk als das aktuelle Laufwerk für die folgenden Dateioperationen, wobei E = 0 für Laufwerk A steht, 1 für Laufwerk B und so weiter bis 7 für Laufwerk H in einem voll ausgebauten System mit acht logischen Laufwerken.

Das Laufwerk wird in einen "on line"-Status versetzt, welcher insbesondere das entsprechende Inhaltsverzeichnis bis zum nächsten Kaltstart, Warmstart oder Systemrücksetzen aktiviert. Dateisteuerblöcke, welche den Laufwerkskode 0 aufweisen, beziehen sich automatisch auf das momentan aktuelle Laufwerk. Laufwerkcodes von 1 bis 16 ignorieren die Standard-Anwahl und beziehen sich direkt auf ein Laufwerk von A bis H.
War die Operation erfolgreich, wird im Akkumulator der Wert 00H zurückgegeben. Tritt jedoch ein Fehler auf, so wird bei der Fehlerbehandlung durch das System eine entsprechende Meldung ausgegeben und der Warmstart vollzogen. Bei Fehlerbearbeitung durch das Anwenderprogramm enthält Register A den Wert 0FFH und Register H einen der folgenden Fehlercodes:

01: Diskettenfehler
04: Auswahlfehler

Funktion 15: Datei eröffnen

Eingangsparameter:
- Register C: 0FH
- Registerpaar DE: FCB-Adresse

Ausgangsparameter:
- Register A: Verzeichniscode

Die Dateieröffnung wird benutzt, um eine Datei zu aktivieren, welche bereits auf der angewählten Diskette existiert. Das BDOS sucht im Inhaltsverzeichnis nach Übereinstimmung mit den Positionen 1 bis 14 des durch DE adressierten Dateisteuerblocks (Byte s1 ist automatisch auf Null gesetzt), wobei ein Fragezeichen (3FH) in jeder dieser Positionen als Übereinstimmung gewertet wird. Normalerweise ist jedoch kein Fragezeichen eingefügt und weiterhin sind die Bytes "ex" und "s2" gleich Null. Wenn ein Eintrag des Inhaltsverzeichnisses übereinstimmt, wird die relevante Information in die Bytes "d0" bis "dn" des Dateisteuerblocks kopiert, wodurch der Zugriff auf diese Datei bei nachfolgenden Lese- und Schreiboperationen ermöglicht wird. Auf eine Datei darf nicht zugegriffen

werden, bevor nicht eine entsprechende Eröffnung erfolgreich durchgeführt wurde. Die Funktion 15 übergibt in A einen "directory code" mit dem Wert von 0 bis 3, falls die Eröffnung erfolgreich durchgeführt wurde. Dieser Wert kennzeichnet, an welcher Stelle im DMA-Puffer der interessierende Eintrag liegt. Falls die Datei nicht gefunden wurde, wird im Register A der Wert 0FFH übergeben. Der momentane Datensatz "cr" muß vom Programm auf Null gesetzt werden, wenn die Datei sequentiell vom ersten Datensatz an gelesen werden soll.

Bei Fehlerbearbeitung im Anwenderprogramm wird einer der folgenden Fehlercodes im Register H übergeben:

00: Datei existiert nicht
01: Diskettenfehler
04: Auswahlfehler
09: mehrdeutiger Name

Funktion 16: Datei schließen

Eingangsparameter:
Register C: 10H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion führt die Umkehrung der Dateieröffnung durch. Wurde der durch DE adressierte Dateisteuerblock durch eine vorhergehende Eröffnung oder Erzeugung (siehe Funktionen 15 und 22) aktiviert, so speichert die Dateischließung den Dateisteuerblock in das Inhaltsverzeichnis der Diskette.

Der FCB-Suchvorgang ist derselbe wie bei der Eröffnungsfunktion. Der Ausgangsparameter im Akkumulator bei einem erfolgreichen Abschluß ist 0, 1, 2, oder 3, während 0FFH übergeben wird, falls der Dateiname im Inhaltsverzeichnis nicht gefunden wurde. Nach Leseoperationen braucht eine Datei nicht geschlossen werden. Nach Schreiboperationen ist in jedem Fall die Datei zu schließen, um den neuen Eintrag im Inhaltsverzeichnis abzuspeichern.

Die Fehlercodes für die Fehlerbearbeitung im Anwenderprogramm entsprechen denen der Funktion 15.

Funktion 17: ersten Eintrag suchen

Eingangsparameter:
Register C: 11H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion durchsucht das Inhaltsverzeichnis auf Übereinstimmung mit dem durch DE adressierten Dateisteuerblock. Der Wert 0FFH wird übergeben, wenn die Datei nicht gefunden wurde, anderenfalls die Werte 0, 1, 2, oder 3, welche anzeigen, daß die Datei vorhanden ist. Der aktuelle DMA-Bereich wird mit dem Datensatz des Inhaltsverzeichnis' gefüllt, welcher dem FCB entspricht. Das Register A gibt in diesem Fall die Nummer des Eintrages im Datensatzes an. Die relative Anfangsadresse kann folglich mit A*32 berechnet werden. Obwohl es normalerweise für Anwenderprogramme nicht

notwendig ist, kann der Inhaltsverzeichniseintrag von dieser Position im Puffer geholt werden.

Ein Fragezeichen (3FH) in irgendeiner Position von "f1" bis "ex" bedeutet Übereinstimmung mit dem entsprechenden Feld im Inhaltsverzeichnis des angewählten Laufwerks. Wenn das "dr"-Feld ein Fragezeichen enthält, ist die automatische Laufwerkauswahl unterbunden und das aktuelle Laufwerk wird auf Übereinstimmung geprüft, wobei die Suchfunktion jeden übereinstimmenden Eintrag übergibt, unabhängig von der Benutzernummer. Wenn das "dr"-Feld kein Fragezeichen enthält, wird das "s2"-Byte automatisch auf Null gesetzt.

Auch diese Funktion übergibt die oben genannten Fehlercodes an das aufrufende Programm, wenn die Fehlerbearbeitung durch das Anwenderprogramm eingestellt ist.

Funktion 18: nächsten Eintrag suchen

Eingangsparameter:
Register C: 12H
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion entspricht der Funktion 17 bis auf die Tatsache, daß die Suche im Inhaltsverzeichnis ausgehend vom letzten übereinstimmenden Eintrag fortgesetzt wird. Wie bei Funktion 17 übergibt Funktion 18 den Wert 0FFH, wenn kein Eintrag mehr gefunden wurde.

Für die richtige Ausführung der Funktion 18 dürfen zwischen den Operationen 17 und 18 und zwischen aufeinanderfolgenden Aufrufen der Funktion 18 keine anderen Diskettenoperationen des BDOS erfolgen.

Funktion 19: Datei löschen

Eingangsparameter:
Register C: 13H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion entfernt Dateien, welche mit dem durch DE adressierten Dateisteuerblock übereinstimmen. Dateiname und Dateityp können mehrdeutig sein (d. h. Fragezeichen in verschiedenen Positionen). Der Laufwerkcode darf nicht mehrdeutig sein, wie bei den Funktionen 17 und 18.

Die Funktion 19 übergibt bei erfolgreicher Abarbeitung im Akkumulator den Verzeichniscode von 0 bis 3 und beim Auftreten eines Fehlers den Wert 0FFH. Im Modus der Fehlerbearbeitung durch das System erfolgt eine Fehlermitteilung und anschließend der Warmstart. Bei Fehlerbearbeitung durch das Anwenderprogramm werden die folgenden Codes übergeben:

00: Datei existiert nicht
01: Diskettenfehler
03: Datei geschützt
(Schreibschutz oder
Systemdatei)
04: Auswahlfehler

Funktion 20: sequentiell lesen

Eingangsparameter:
Register C: 14H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Fehlercode

Wurde der durch DE adressierte Dateisteuerblock mittels der Funktion 15 aktiviert, so liest diese Funktion den nächsten 128 Byte langen Datensatz in den Puffer mit der momentanen DMA-Adresse. Der Datensatz wird von der Position "cr" der Erweiterung gelesen und das "cr"-Feld wird automatisch auf die nächste Position erhöht. Wenn das "cr"-Feld überläuft, wird die nächste logische Erweiterung automatisch eröffnet und das "cr"-Feld auf Null gesetzt für die nächste Leseoperation.

Der Wert 00H wird im Register A übergeben, wenn die Leseoperation erfolgreich war, während ein Wert ungleich 00H übergeben wird, falls bei der Operation ein Fehler auftrat. Dabei kennzeichnet der Wert 01H das Dateiende und der Wert 0FFH das Auftreten eines physischen Fehlers. Im letzteren Fall enthält das Register H einen der folgenden Fehlerkodes:

01: Diskettenfehler
04: Auswahlfehler

Funktion 21: sequentiell schreiben

Eingangsparameter:
Register C: 15H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Wenn der durch DE adressierte Dateisteuerblock durch eine der Funktionen 15 oder 22 aktiviert wurde, so schreibt diese Funktion den 128 Byte langen Datensatz auf der momentanen DMA-Adresse in die durch diesen FCB gekennzeichnete Datei. Der Datensatz wird an der durch "cr" vorgegebenen Position abgelegt, und das "cr"-Feld wird automatisch auf die nächste Position erhöht. Läuft das "cr"-Feld über, so wird automatisch die nächste logische Erweiterung eröffnet und das "cr"-Feld wird in Vorbereitung der nächsten Schreiboperation auf Null gesetzt.

Schreiboperationen können in existierenden Dateien ausgeführt werden, wobei neu geschriebene Datensätze die schon existierenden überschreiben.

Register A wird von einer erfolgreichen Schreiboperation mit dem Wert 00H übergeben, während ein Wert ungleich Null eine nicht ausgeführte Schreiboperation anzeigt. Dabei bedeuten im einzelnen die Werte:

01: kein Platz im Inhaltsverzeichnis
02: kein Platz auf der Diskette
FF: physischer Fehler (siehe Register H)

Beim Auftreten eines physischen Fehlers wird im Register H einer der folgenden Fehlercodes übergeben:

01: Diskettenfehler
03: Datei geschützt
04: Auswahlfehler

Funktion 22: Datei erzeugen

Eingangsparameter:
Register C: 16H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion gleicht der Eröffnungsfunktion mit dem Unterschied, daß der Dateisteuerblock eine Datei bezeichnen muß, welche noch nicht in dem ausgewählten Inhaltsverzeichnis existiert (d. h. in demjenigen Verzeichnis, welches explizit durch ein "dr"-Feld ungleich Null vorgegeben ist bzw. bei "dr" = 0 in dem aktuellen Verzeichnis). Das BDOS erzeugt einen Eintrag für eine leere Datei im Inhaltsverzeichnis.

Der Programmierer muß sicher sein, daß doppelte Dateinamen nicht auftreten, d. h. vorhergehende Löschoperationen sind erforderlich, wenn die Möglichkeit der Dublizität besteht. Bei Rückkehr enthält Register A die Werte 0, 1, 2 oder 3, wenn die Operation erfolgreich war oder ansonsten den Wert 0FFH. Im letzteren Fall wird im Register H die Fehlerursache übergeben:

01: Diskettenfehler
04: Auswahlfehler
08: Datei vorhanden
09: mehrdeutiger Name

Die Funktion aktiviert den FCB, so daß keine Eröffnung notwendig ist.

Funktion 23: Datei umbenennen

Eingangsparameter:
Register C: 17H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion benutzt den durch DE adressierten Dateisteuerblock, um alle Dateieintragungen mit dem in den ersten 16 Byte angegebenen Namen in den mit den zweiten 16 Byte angegebenen Namen umzubenennen. Der Laufwerkcode "dr" auf Position 0 wird benutzt, um das Laufwerk anzugeben, während der Code auf Position 16 mit Null angenommen wird. Bei Rückkehr ist das Register A auf einen Wert von 0 bis 3 gesetzt, wenn die Umbenennung erfolgreich war oder 0FFH, falls keine Datei mit dem ersten angegebenen Namen vorhanden ist.

Funktion 24: Abfrage Laufwerke im 'online'-Zustand

Eingangsparameter:
Register C: 18H
Ausgangsparameter:
Registerpaar HL: Anwahlvektor

Der Anwahlvektor, welcher vom MicroDOS-System übergeben wird, ist ein 16-Bit-Wert in HL, bei dem das niederwertigste Bit von Register L dem Laufwerk A entspricht und das höchstwertigste Bit des Registers H dem sechzehnten Laufwerk P (in der Implementierung auf KC-Floppy-System bis max. Laufwerk H) entspricht. Ein "0"-Bit zeigt an, daß das Laufwerk noch nicht angewählt wurde, während ein "1"-Bit ein durch eine explizite Anwahl bzw. durch eine Dateioperation mit einem "dr"-Feld ungleich Null angewähltes Laufwerk markiert. Kompatibilität mit älteren Versionen des Betriebssystems CP/M ist gesichert, da bei der Rückkehr Register A gleich Register L ist.

Funktion 25: aktuelles Laufwerk

Eingangsparameter:
Register C: 19H
Ausgangsparameter:
Register A: aktuelles Laufwerk

Funktion 25 holt die Nummer des momentan angewählten Standardlaufwerks in Register A. Die Laufwerknummer liegt im Bereich von 0 bis 7 entsprechend den Laufwerken A bis H.

Funktion 26: DMA-Adresse setzen

Eingangsparameter:
Register C: 1AH
Registerpaar DE: DMA-Adresse

Im MicroDOS wird die Adresse, auf welcher vor einer Schreiboperation bzw. nach einer Leseoperation der 128 Byte lange Datensatz liegt, mit DMA-Adresse bezeichnet.

Nach einem Kaltstart, einem Warmstart oder einem Rücksetzen des Diskettensystems ist die DMA-Adresse automatisch auf 0080H gesetzt. Die DMA-Adreßsetzfunktion kann benutzt werden, um diesen Standardwert zur Adressierung eines anderen Speicherfeldes, wo die Daten liegen, zu ändern.

Dadurch erhält die DMA-Adresse den in DE vorgegebenen Wert bis zu einer folgenden Adreßsetzfunktion oder einem Neustart bzw. RESET.

Einige Funktionen des BDOS nutzen den DMA-Puffer für die Übergabe von Parametern, so übergibt z. B. die Funktion 46 die Größe des freien Diskettenplatzes im DMA-Puffer.

Funktion 27: Belegungsvektor holen

Eingangsparameter:
Register C: 1BH
Ausgangsparameter:
Registerpaar HL: Belegungsvektor

Für jedes angewählte Laufwerk wird im Hauptspeicher eine Belegungstabelle angelegt. Verschiedene Programme nutzen die Information des Vektors, um die Größe des verbleibenden Speicherplatzes festzustellen. Funktion 27 übergibt die Basisadresse der Belegungstabelle für das gerade angewählte Laufwerk. Die Information kann jedoch ungültig sein, wenn die Diskette in diesem Laufwerk gewechselt wurde. Beim Auftreten eines Fehlers wird im Registerpaar HL der Wert 0FFFFH übergeben.

Funktion 28: Schreibschutz setzen

Eingangsparameter:
Register C: 1CH

Diese Funktion gewährleistet einen vorübergehenden Schreibschutz für das gerade angewählte Laufwerk. MicroDOS führt die Diskette beim Schreiben automatisch in den Schreib/Lesezustand zurück. Funktion 28 wird nur zwecks Kompatibilität mit SCPX bzw. CP/M 2.2 ausgeführt.

Funktion 29: Schreibschutzvektor holen

Eingangsparameter:
Register C: 1DH
Ausgangsparameter:
Registerpaar HL: Schreibschutzvektor

Funktion 29 übergibt im Registerpaar HL einen Bit-Vektor, der diejenigen Laufwerke angibt, bei denen ein vorübergehender Schreibschutz gesetzt ist. Wie bei Funktion 24 entspricht das niederwertigste Bit dem Laufwerk A, während das höchstwertigste Bit dem Laufwerk P zugeordnet ist. Das R/O-Bit wird durch die Funktion 28 gesetzt.

Funktion 30: Dateiattribute setzen

Eingangsparameter:
Register C: 1EH
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Verzeichniscode

Diese Funktion erlaubt die programmtechnische Manipulation von ständigen Indikatoren, welche den Dateien zugeordnet sind. Im einzelnen werden die Schreibschutz- und Systemattribute (t1 und t2) gesetzt oder rückgesetzt. Das Registerpaar DE adressiert einen eindeutigen Dateisteuerblock mit den gesetzten Attributen. Funktion 30 sucht im Inhaltsverzeichnis nach Übereinstimmung und ändert den entsprechenden Eintrag, damit er die gewählten Attribute enthält.

Die Indikatoren f1 bis f4 werden gegenwärtig nicht genutzt, können aber für Anwenderprogramme nützlich sein, da sie nicht vom Suchprozeß während einer Dateieröffnung oder einer Dateischließung berührt werden.

Die Funktion 30 gibt bei erfolgreichem Abschluß im Register A den Verzeichniscode von 0 bis 3 zurück. Im Fehlerfall enthält Register A den Wert 0FFH und das Register H einen der folgenden Fehlercodes:

00: Datei existiert nicht
01: Diskettenfehler
04: Auswahlfehler
09: mehrdeutiger Name

Funktion 31: Parameteradresse holen

Eingangsparameter:
Register C: 1FH
Ausgangsparameter:
Registerpaar HL: DPB-Adresse

Funktion 31: Parameteradresse holen

Eingangsparameter:
Register C: 1FH
Ausgangsparameter:
Registerpaar HL: DPB-Adresse

Die Adresse des BIOS-residenten Diskettenparameterblocks (DPB) für das gerade aktive Laufwerk wird als Resultat dieser Funktion im Registerpaar HL übergeben. Diese Adresse kann für zwei Zwecke benutzt werden. Erstens können die Diskettenparameter für Anzeigezwecke oder zur Berechnung von Speicherplatz geholt werden oder transiente Programme können die Parameter ändern, wenn die Diskettenumgebung wechselt.

Funktion 32: Benutzernummer holen/setzen

Eingangsparameter:
Register C: 20H
Register E: 0FFH oder Benutzernummer
Ausgangsparameter:
Register A: Benutzernummer oder kein Wert

Durch Aufruf der Funktion 32 kann ein Anwenderprogramm die aktuelle Benutzernummer ändern oder abfragen. Ist Register E = 0FFH, so wird der augenblickliche Wert der Benutzernummer im Register A übergeben, wobei der Wert im Bereich von 0 bis 15 liegt. Ist Register E ungleich 0FFH, so wird die Benutzernummer auf den Wert von E gesetzt (modulo 16).

Funktion 33: wahlfrei lesen

Eingangsparameter:
Register C: 21H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Fehlercode

Die Funktion "wahlfrei lesen" ähnelt der Funktion "sequentiell lesen" mit dem Unterschied, daß die Leseoperation mit einer bestimmten Datensatznummer durchgeführt wird, welche durch den 24 Bit-Wert aus den drei dem FCB folgenden Bytes gebildet wird (Bytepositionen r0 bei 33, r1 bei 34 und r2 bei 35). Die Folge der 24 Bit ist mit dem niederwertigsten Byte (r0) zuerst, dem mittleren Byte (r1) in der Mitte und dem höherwertigsten Byte (r2) zuletzt abgespeichert.

Das MicroDOS-System benutzt das r2-Byte nicht, ausgenommen bei der Berechnung der Dateigröße (Funktion 35). Byte r2 muß Null sein, da r2 ungleich Null einen Überlauf nach dem Dateiende anzeigt.

Das "r0, r1"-Bytepaar wird als Doppelbytewert behandelt, welcher den zu lesenden Datensatz angibt. Dieser Wert reicht von 0 bis 65535 und gewährleistet den Zugriff zu jedem einzelnen Datensatz einer 8 MByte-Datei. Um eine Datei im wahlfreien Zugriff zu verarbeiten, muß zuerst die Basiserweiterung 0 mit Funktion 15 oder 22 eröffnet werden. Unabhängig davon, ob die gewünschten Daten hierin enthalten sind oder nicht, wird dadurch gesichert, daß die Datei im Inhaltsverzeichnis eingetragen ist und entsprechenden Operationen zugänglich ist.

Die ausgewählte Datensatznummer wird dann in dem Feld r0, r1 abgelegt und das BDOS wird aufgerufen, um den Datensatz zu lesen. Bei der Rückkehr enthält Register A einen Fehlercode, wie unten aufgeführt, oder den Wert 00H, wenn die Operation erfolgreich war. In diesem Fall enthält der momentane DMA-Bereich den gelesenen Datensatz.

Im Gegensatz zum sequentiellen Lesen wird die Datensatznummer nicht erhöht. Deshalb würden nachfolgende Leseoperationen denselben Datensatz lesen.

Bei jeder wahlfreien Leseoperation werden die Werte für die logische Erweiterung und den momentanen Datensatz automatisch gesetzt.

Deshalb kann die Datei sequentiell, ausgehend von der durch wahlfreien Zugriff bestimmten Position, gelesen oder geschrieben werden. In diesem Fall wird der letzte Datensatz noch einmal gelesen beim Übergang vom wahlfreien zum sequentiellen Lesen bzw. der letzte Datensatz wird überschrieben beim Übergang vom wahlfreien zum sequentiellen Schreiben.

Selbstverständlich kann die Datensatznummer nach jedem wahlfreien Zugriff erhöht werden, um den Effekt des sequentiellen Zugriffs zu erreichen.

Es folgen die Fehlercodes, welche im Register A übergeben werden:

01	Lesen von nicht existenten Daten
02	(wird nicht von wahlfreien Operationen übergeben)
03	die momentane Erweiterung kann nicht geschlossen werden
04	Suche nach nicht geschriebener Erweiterung
05	(wird nicht von Leseoperationen übergeben)
06	Suche über das physikalische Ende der Diskette hinaus
FF	physischer Fehler (siehe Register H)

Die Fehlercodes 1 und 4 treten auf, wenn eine wahlfreie Leseoperation auf einen Datenblock zugreift, welcher vorher nicht geschrieben wurde bzw. auf eine Erweiterung, die nicht erzeugt wurde. Fehlercode 3 tritt normalerweise bei richtiger Funktion des Systems nicht auf, kann jedoch durch einfaches Neulesen oder durch Neueröffnung der Erweiterung 0 gelöscht werden, solange die Diskette nicht physikalisch schreibgeschützt ist.

Der Fehlercode 6 tritt auf, wenn das Byte r2 nicht Null ist. Normalerweise können auftretende Fehlercodes als Datenverlust angesehen werden. Der Wert 0FFH zeigt einen physischen Fehler an, dessen Ursache im Register H genauer spezifiziert wird. Der Rückkehrcode 00H zeigt vollständige Operationen an.

Funktion 34: wahlfrei schreiben

Eingangsparameter:
- Register C: 22H
- Registerpaar DE: FCB-Adresse

Ausgangsparameter:
- Register A: Fehlercode

Die Funktion "wahlfrei schreiben" wird auf die gleiche Weise eingeleitet wie die Funktion "wahlfrei lesen", nur werden hierbei Daten vom momentanen DMA-Bereich auf die Diskette geschrieben. Wenn die Erweiterung oder der Datenblock, auf welchen geschrieben werden soll, noch nicht belegt wurde, wird dies vor der Schreiboperation ausgeführt. Wie bei der Leseoperation wird im Ergebnis der Funktion die Datensatznummer nicht verändert.

Die Nummer der Erweiterung und die Datensatznummer werden im Dateisteuerblock gesetzt in Übereinstimmung mit dem Datensatz, welcher geschrieben wurde.

Wiederum können sequentielle Operationen folgen, wobei auch hier der momentan adressierte Datensatz noch einmal gelesen oder geschrieben wird, wenn die sequentielle Operation beginnt. Ebenso kann die Nummer des Datensatzes nach jeder Operation erhöht werden, um den Effekt des sequentiellen Schreibens zu erhalten. Allerdings erfolgt nach dem letzten Datensatz einer Erweiterung nicht ein automatisches Umschalten auf die nächste, wie beim sequentiellen Schreiben.

Die übergebenen Fehlercodes sind die gleichen wie beim wahlfreien Lesen, wobei zusätzlich der Fehlercode 5 auftritt, welcher anzeigt, daß eine neue Erweiterung auf Grund eines Überlaufs des Inhaltsverzeichnisses nicht erzeugt werden kann.

Funktion 35: Dateigröße berechnen

Eingangsparameter:

Register C: 23H

Registerpaar DE: FCB-Adresse

Ausgangsparameter:

Dateigröße in r0, r1, r2

Bei der Ermittlung der Dateigröße adressiert das Registerpaar DE einen Dateisteuerblock im Format für wahlfreien Zugriff, d. h. die Bytes r0 bis r2 sind vorhanden. Der Dateisteuerblock enthält einen eindeutigen Dateinamen, welcher im Inhaltsverzeichnis gesucht wird. Bei Rückkehr enthalten die Bytes für wahlfreien Zugriff die "virtuelle" Dateigröße, welche effektiv die Nummer des Datensatzes darstellt, der dem Dateiende folgt.

Wenn nach Aufruf der Funktion 35 das höchste Datensatzbyte r2 gleich 1 ist, dann enthält die Datei die maximale Anzahl von Datensätzen (65535). Anderenfalls bilden die Bytes r0 und r1 einen 16 Bit-Wert, welcher die Dateigröße darstellt (wie vorher ist r0 das niederwertigste Byte).

Daten können an das Ende einer existierenden Datei angefügt werden durch Aufruf der Funktion 35, um die Positionen für wahlfreien Zugriff mit dem Ende der Datei zu belegen, und durch anschließendes wahlfreies Schreiben von Datensätzen, beginnend an der vorbelegten Datensatzadresse.

Die "virtuelle" Größe der Datei stimmt mit der realen Größe überein, wenn die Datei sequentiell geschrieben wurde. Ist die Datei jedoch im wahlfreien Modus geschrieben worden, dann existieren Lücken in der Plazierung der Datensätze und die Datei enthält weniger Datensätze, als durch die Größe angezeigt wird. Wenn z. B. nur der letzte Datensatz einer 8 MByte-Datei im wahlfreien Zugriff geschrieben wird (d. h. Datensatznummer 65535), wird die "virtuelle" Größe der Datei mit 65535 Datensätzen angegeben, obwohl nur ein Datenblock vorhanden ist.

Funktion 36: Feld für wahlfreien Zugriff setzen

Eingangsparameter:
Register C: 24H
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Feld für wahlfreien Zugriff gesetzt

Diese Funktion setzt automatisch das Feld für wahlfreien Zugriff einer Datei, die bis zu einem bestimmten Punkt sequentiell gelesen oder geschrieben wurde. Die Funktion kann auf zwei Arten nützlich sein.

Erstens ist es oft notwendig, eine Datei zuerst sequentiell zu lesen und nach verschiedenen "Schlüssel"-Informationen zu durchsuchen. Ist der Schlüssel gefunden, so wird Funktion 36 aufgerufen, um die Position des zugehörigen Datensatzes zu bestimmen. Die so gewonnene Datensatzposition kann in einer Tabelle für späteren Zugriff abgelegt werden. Nachdem die Datei durchsucht wurde und alle Schlüsseldatensätze in einer Tabelle abgelegt wurden, kann man auf einen bestimmten Datensatz über einen wahlfreien Zugriff, unter Benutzung der vorher abgespeicherten Datensatznummer, zurückgreifen.

Dieses Schema kann leicht auf Dateneinheiten mit variabler Länge ausgedehnt werden, da dann zusätzlich zum Schlüssel und zur Datensatznummer nur die relative Position im Datensatz abgespeichert werden muß, um die exakte Startposition der Schlüsselinformation später zu finden.

Die zweite Verwendungsmöglichkeit der Funktion 36 tritt beim Umschalten vom sequentiellen Lesen oder Schreiben auf den wahlfreien Zugriff auf. Wenn eine Datei bis zu einem bestimmten Punkt sequentiell bearbeitet wurde, kann durch Aufruf der Funktion 36, welche die Datensatznummer setzt, die wahlfreie Bearbeitung ab dem bis dahin erreichten Punkt der Datei erfolgen.

Funktion 37: Diskette rücksetzen

Eingangsparameter:
Register C: 25H
Registerpaar DE: Diskettenvektor
Ausgangsparameter:
Register A: 00H

Funktion 37 realisiert programmgesteuert das Rücksetzen der Disketten, die durch den im Registerpaar DE übermittelten Vektor angegeben werden. Beim Rücksetzen wird die Diskette logisch ausgeschaltet und in das Regime Lesen/Schreiben versetzt. Das niedrigste Bit des Diskettenvektors entspricht dem Laufwerk A, das höchste dem Laufwerk P. Der Wert 1 des Bits bedeutet, daß die entsprechende Diskette zurückgesetzt werden muß.

Funktion 40: Wahlfreies Schreiben mit Auffüllen durch Nullen

Eingangsparameter:
Register C: 1EH
Registerpaar DE: FCB-Adresse
Ausgangsparameter:
Register A: Rückkehrcode
Register H: Fehlercode

Diese Funktion arbeitet analog der Funktion des wahlfreien Schreibens 34 mit der Ausnahme, daß der zugewiesene Block vor dem Schreiben mit Nullen aufgefüllt wird. Wenn beim Anlegen der Datei diese Funktion genutzt wurde, enthalten die freien Sätze in den Blöcken Nullen. Wenn Operation 34 verwendet wurde, enthalten die freien Sätze nicht initialisierte Daten.

Funktion 45: Festlegen des Regimes der Fehlerbearbeitung

Eingangsparameter:
- Register C : 2DH
- Register E : Regime der Bearbeitung

Funktion 45 legt das Regime der Fehlerbearbeitung des BDOS fest. Wenn im Register E der Wert 0FEH (254 dezimal) angegeben wird, dann kehrt das BDOS mit einem Fehlercode im Akkumulator zum aufrufenden Programm zurück. In allen anderen Fällen wird bei Auftreten eines Fehlers dieser auf der Konsole angezeigt, und der Benutzer hat die Möglichkeit, den Fehler zu ignorieren oder einen Warmstart auszuführen.

Funktion 46: Abfragen des freien Platzes auf der Diskette

Eingangsparameter:
- Register C: 2EH
- Register E: Diskette

Ausgangsparameter:
- Register A: Fehlerkennzeichen
- Register H: Fehlercode

Funktion 46 bestimmt die Anzahl der freien Sektoren (128 Byte lange Sätze) auf der Diskette, die durch Register E angegeben wird. Der Wert 0 entspricht der Diskette A, 1 der Diskette B usw. bis 15 für die Diskette P.

Der Wert der Anzahl der freien Sektoren wird im Dual-Code in den ersten drei Bytes des aktuellen Puffers des Direktzugriffs (DMA) in folgendem Format zurückgegeben:

Byte 0 - niedrigstes Byte
Byte 1 - mittleres Byte
Byte 2 - höchstes Byte

Funktion 46 gibt bei erfolgreichem Abschluß im Register A den Wert 00H zurück. Beim Auftreten eines physischen Fehlers werden bei Fehlerbearbeitung durch das System die Fehlermitteilung auf der Konsole ausgegeben und der "Warmstart" vollzogen.

Im Modus der Fehlerbearbeitung durch das Anwenderprogramm wird im Register A der Wert 0FFH zurückgegeben, dabei enthält Register H einen der folgenden Fehlercodes:

01: Diskettenfehler
04: Auswahlfehler

Funktion 47: Wechsel des Programms

Eingangsparameter:
Register C: 2FH

Funktion 47 ermöglicht den Aufruf eines anderen Programms aus dem laufenden Programm ohne Zusammenwirken mit dem Bediener.

Das aufrufende Programm muß in den Puffer des Direktzugriffs bei Abbruch (ab Adresse 0080H) einen Kommandosatz einspeichern.

Die Funktion des Wechsels des Programms gibt dem aufrufenden Programm nicht die Steuerung zurück. Die festgestellten Fehler werden vom Kommandointerpreter bearbeitet.

Funktion 49: Abfragen/Stellen der Parameter des Systemsteuerblocks

---

Eingangsparameter:
Register C: 31H
Registerpaar DE: Adresse der Parameter
Ausgangsparameter:
Register A: Parameterbyte
Registerpaar HL: Parameterwort

Funktion 49 ermöglicht den Zugriff auf den Systemsteuerblock. Der Systemsteuerblock ist ein 100 Byte großer Datenbereich (Abschnitt 3.4.), der Marken und Daten beinhaltet, die vom System genutzt werden. Für die Anwendung dieser Funktion speichert das aufrufende Programm im Registerpaar DE die Adresse des Parameterblocks, der die Funktion bestimmt. Die Struktur des Parameterblocks kann folgendermaßen beschrieben werden:

```
SCBPB:  DEFB   OFFSET  ;Position im Systemsteuerblock
        DEFB   SET     ;0FFH-Setzen des Bytes
                       ;0FEH-Setzen des Wortes
                       ;00 - Abfrage des Parameters
        DEFW   VALUE   ;Parameter zum Setzen
                       ;(Byte oder Wort)
```

OFFSET bestimmt die Position des Parameters, der gestellt oder abgefragt werden muß, innerhalb des Systemsteuerblocks. SET bestimmt, ob der Parameter gelesen oder geschrieben wird und ob es sich um einen Byte- oder Wortparameter handelt. VALUE enthält das Wort oder das Byte für das Setzen.

Die Funktion 49 muß mit Vorsicht angewendet werden, da der Systemblock Systemvariablen enthält, deren Veränderung durch die Anwendungsprogramme zu Fehlern in der Arbeit des Systems führen kann.

Funktion 49 ist zum Stellen der Parameter nur dann zu verwenden, wenn kein analoges Resultat durch eine andere Funktion erreicht werden kann.

Funktion 50: Aufruf der BIOS-Funktionen

Eingangsparameter:
Register C: 32H
Registerpaar DE: Adresse der Parameter
Ausgangsparameter:
BIOS-Parameter

Die Funktion 50 realisiert den Aufruf der BIOS-Funktionen. Das aufrufende Programm übergibt in das Registerpaar DE die Adresse des Parameterblocks, der die Nummer der BIOS-Funktion und die zu ihrer Ausführung notwendigen Parameter bestimmt. Dieser Parameterblock hat folgende Struktur:

```
BIOSBP:  DEFB  FUNC   ; Nummer der Funktion
         DEFW  REGA   ; Inhalt des Registers A
         DEFW  REGB   ; Inhalt des Registerpaars BC
         DEFW  REGD   ; Inhalt des Registerpaars DE
         DEFW  REGH   ; Inhalt des Registerpaars HL
```

Die Nummer der Funktion im Feld FUNC für den Aufruf der BIOS-Funktion muß sich in den Grenzen von 00H...10H (0...16) bewegen. Es folgt eine Aufstellung der Funktionsnummern, der notwendigen Eingabedaten, der zurückgegebenen Werte und der Wirkungen dieser Funktionen.

Funktion	Bedeutung
0	Kaltstart
1	Warmstart
2	Konsolenstatus Wenn ein Zeichen bereit ist zur Eingabe, wird in Register A der Wert 0FFH zurückgegeben, sonst 00H.
3	Konsoleneingabe Das eingegebene Zeichen wird in Register A zurückgegeben.
4	Konsolenausgabe Ausgabe des Zeichens in Register C auf die Konsole.
5	Druckerausgabe Ausgabe eines Zeichens aus dem Register C auf den Drucker.
6	Zusatzausgabe Ausgabe eines Zeichens in Register C auf den Zusatzkanal.
7	Zusatzeingabe Eingabe eines Zeichens vom Zusatzkanal in das Register A.
8	Setzen des Diskettenkopfes auf Spur 0.
9	Wahl des Laufwerks Register C enthält 0 für Diskette A, 1 für

B usw. bis 0FH für Diskette P. Die Funktion gibt im Registerpaar HL die Adresse des Parameterkopfes des Laufwerks zurück. Wird ein nicht existierendes Laufwerk angegeben, wird im Registerpaar HL der Wert 0000H zurückgegeben.

10 Setzen auf die durch das Registerpaar BC angegebene Spur.

11 Setzen auf den durch das Registerpaar BC angegebenen Sektor.

12 Stellen der Adresse des DMA-Puffers auf den durch Registerpaar BC angegebenen Wert.

13 Lesen der Diskette mit den Parametern, die durch die Funktionen 9, 10, 11 und 12 festgelegt wurden. Bei erfolgreicher Ausführung der Funktion wird im Register A der Wert 00H zurückgegeben, sonst ein Wert ungleich 0.

14 Schreiben auf die Diskette mit den Parametern, die durch die Funktionen 9, 10, 11 und 12 festgelegt wurden. Bei erfolgreicher Ausführung der Funktion wird im Register A 00H zurückgegeben, sonst ein Wert ungleich 0.

15 Abfrage des Zustandes des logischen Gerätes LIST:
Ist das Zeichen bereit zur Eingabe, wird in Register A der Wert 0FFH zurückgegeben, sonst 00H.

16 Wandelt die logische Nummer des Sektors, die im Registerpaar BC übergeben wurde, in die physische Nummer um. Die Adresse der Umwandlungstabelle wird im Registerpaar DE übergeben.

Funktion 108: Abfrage/Stellen des Rückkehrcodes

Eingangsparameter:
Register C: 6CH
Registerpaar DE: Rückkehrcode oder FFFFH
Ausgangsparameter:
Registerpaar HL: Rückkehrcode

Funktion 108 ermöglicht dem Programm vor dem Abschluß einen Rückkehrcode zu stellen.

Der Rückkehrcode wird vor der bedingten Ausführung des Kommandos (Kennzeichen ':' im Kommandosatz) überprüft, und er kann auch von einem Programm, das mit Hilfe der Funktion 47 von einem anderen Programm aufgerufen wurde, überprüft werden. Das gewährleistet das Abarbeiten der

Programme nur im Falle des erfolreichen Abschlusses des vorhergehenden Programms.

Für das Abfragen des Rückkehrcodes übergibt das aufrufende Programm im Registerpaar DE den Wert 0FFFFH und für das Stellen den Rückkehrcode selbst. Die Werte des Rückkehrcodes werden unten angeführt.

Code	Bedeutung
0000 - FEFF	erfolgreicher Abschluß
FF00 - FFFE	kein erfolgreicher Abschluß
FF80 - FFFC	Reservekodes
FFFD	Abschluß wegen eines Fehlers von BDOS
FFFE	Abschluß bei Eingabe von CTL-C

Funktion 110: Abfragen/Stellen des Begrenzungszeichens einer Folge

Eingangsparameter:
Register C: 6EH
Registerpaar DE: Begrenzer oder FFFFH
Ausgangsparameter:
Register A: Begrenzer

Funktion 110 gewährleistet das Abfragen oder das Stellen des Begrenzers einer Zeichenkette für die Funktion 9. Wenn im Registerpaar DE der Wert 0FFFFH übergeben wird, dann wird im Register A der aktuelle Wert des Begrenzers zurückgegeben, in allen anderen Fällen wird der Begrenzer durch das Zeichen im Register E ersetzt. Beim Warmstart wird das Zeichen "$" als Begrenzer festgelegt.

Funktion 111: Ausgabe eines Puffers auf die Konsole

Eingangsparameter:
Register C: 6FH
Registerpaar DE: CCB-Adresse

Funktion 111 gibt auf das logische Gerät CONOUT: den Inhalt eines Puffers aus, der durch den im Registerpaar DE angegebenen Zeichensteuerblock bestimmt wird. Nachstehend wird das Format des Zeichensteuerblocks beschrieben:

Byte 0/1: Anfangsadresse des Puffers
Byte 2/3: Länge des Puffers

Funktion 112: Ausgabe eines Puffers auf dem Drucker

Eingangsparameter:
Register C: 70H
Registerpaar DE: CCB-Adresse

Funktion 112 gibt auf dem logischen Drucker den Inhalt eines Puffers aus, der durch den im Registerpaar DE angegebenen Zeichensteuerblock bestimmt wird. Nachstehend wird das Format des Zeichensteuerblocks beschrieben:

Byte 0/1: Anfangsadresse des Puffers
Byte 2/3: Länge des Puffers

Funktion 152: Vorbereiten des Dateisteuerblocks

Eingangsparameter:
Register C: 98H
Registerpaar DE: Adresse des Blocks PFCB
Ausgangsparameter:
Register A: Rückkehrcode

Funktion 152 gewährleistet die Vorbereitung eines Dateisteuerblocks aus dem Namen der Datei. Das aufrufende Programm übergibt im Registerpaar DE die Adresse eines Parameterblocks, der folgendes Format besitzt:

```
PFCB: DEFW STRING  ; Adresse des Dateinamens
      DEFW AFCB    ; Adresse des vorzubereitenden Dateisteuerblocks
```

Die Dateibezeichnung muß folgendermaßen angegeben werden:

[D:]Dateiname[.Dateityp]

wobei die Felder in den eckigen Klammern nicht obligatorisch sind. Die Länge der Folge, die den Namen enthält, darf 128 Byte nicht überschreiten. Funktion 152 untersucht den angegebenen Dateinamen und bereitet den Dateisteuerblock vor. Leer- und Tabulatorzeichen vor dem Namen werden übergangen. Als Begrenzer für den Dateinamen dient eines der folgenden Zeichen:

Zeichen	Hexadezimalcode
Null	00H
CR	0DH
Tabulator	09H
Leerzeichen	20H
:	3AH
;	3BH
=	3DH
^	5FH
.	2EH
[	5BH
<	3CH
>	3EH
,	2CH

Wenn im angegebenen Namen Steuerzeichen mit dem Code von 0 bis 20H auftreten, die nicht in der Tabelle angegeben sind, gibt die Funktion 152 im Registerpaar HL den Code 0FFFFH zurück. Bei erfolgreicher Vorbereitung des Namens überprüft die Funktion das nächste Zeichen. Ist das nächste Zeichen 0 oder <Wagenrücklauf> (0DH), wird im Registerpaar HL der Wert 0 zurückgegeben. Wenn das nächste Zeichen eines der Begrenzer ist, wird im Registerpaar HL die Adresse des Begrenzers zurückgegeben, sonst die Adresse des ersten folgenden Leer- oder Tabulatorzeichens.

7. BIOS-Funktionen

7.1. Beschreibung der BIOS-Schnittstelle

Das BIOS ist der von der Hardware abhängige Modul des Betriebssystems MicroDOS. Es beinhaltet die für die spezielle Hardware notwendigen Ein-/Ausgaberoutinen. Damit bildet es die Schnittstelle zwischen der Hardware und dem hardwareunabhängigen Teil des Betriebssystems bzw. dem Anwenderprogramm.

Die im BIOS enthaltenen Ein-/Ausgaberoutinen können in drei Gruppen zusammenfaßt werden:

1. Systeminitialisierung
2. Zeichenein- und -ausgabe
3. Diskettenein- und -ausgabeoperationen

Die Routinen erreicht man über einen sogenannten "Sprungvektor". Im MicroDOS ist der Sprungvektor selbst Bestandteil des BDOS. Der Sprungvektor stellt eine zusammenhängende Folge von Sprungbefehlen dar. Nachfolgend sind die Routinen der oben genannten Gruppen angegeben, zu denen je ein Sprungbefehl im Sprungvektor enthalten ist:

Systeminitialisierung:	Kaltstartroutine
	Warmstartroutine
Zeichenein- / -ausgabeoperationen:	Status CONSOLE-Gerät
	Eingabe von CONSOLE-Gerät
	Ausgabe auf CONSOLE-Gerät
	Ausgabe auf LIST-Gerät
	Status LIST-Gerät
	Ausgabe auf PUNCH-Gerät
	Eingabe von READER-Gerät
Diskettenein- / -ausgabeoperationen:	Positionieren Spur Null
	Laufwerk auswählen
	Spur auswählen
	Transformation Sektornummer
	Sektor auswählen
	Datenpufferadresse setzen
	Selektierten Sektor lesen
	Schreiben selektierten Sektor

7.2. Initialisierung

Es gibt im BIOS zwei Routinen zur Initialisierung des Systems, den Kaltstart und den Warmstart.

Die Kaltstartroutine BOOT wird nur nach dem Urladen oder der Neuinstallation von MicroDOS im Speicher aktiviert. Sie führt eine grundlegende Systeminitialisierung sowohl des Betriebssystems als auch der Hardware durch und gibt einen System-Kaltstarttext auf den Bildschirm aus.

Wenn eine INITIAL.SUB Datei auf dem Systemlaufwerk vorhanden ist, so wird diese gestartet. Die Initialisierung schließt mit der Übergabe der Steuerung an den CCP ab.

Die Warmstartroutine WBOOT wird immer dann aktiviert, wenn ein Nutzerprogramm zur Adresse 0000H verzweigt. Nach der Initialisierung der Systemparameter wird der CCP aufgerufen.

7.3. Logische Ein-/Ausgabekanäle

MicroDOS unterstützt vier logische Kanäle:

- einen Ein/Ausgabekanal CON:
- einen Eingabekanal RDR:
- zwei Ausgabekanäle LST: und PUN:

Über die im Punkt 6.3. beschriebenen BDOS-Funktionen sind diese Kanäle ansprechbar.

Die Unterprogramme, höhere Programmiersprachen und verschiedene Kommandos (z. B. PIP, STAT, ..., siehe dazu auch /8/) greifen über die BDOS-Funktionen auf die E/A-Kanäle zu.

Die Schlüsselbegriffe CON:, LST:, PUN: und RDR: stellen die Kanalbezeichnungen dar, wie sie u. a. in den Programmen PIP und STAT Verwendung finden. Die vier logischen E/A-Kanäle haben folgende Eigenschaften:

CON:	(CONSOLE) Dieser logische Kanal kann sowohl zur Ein- als auch zur Ausgabe von Zeichen genutzt werden. Eingaberoutinen: CONIN für zeichenweise Eingabe CONST zur Abfrage, ob ein Zeichen verfügbar ist Ausgaberoutine: CONOUT für zeichenweise Ausgabe
LST:	(LIST) Der logische Kanal dient zur Ausgabe einzelner Zeichen. Das LIST-Gerät ist normalerweise ein Drucker. Eingaberoutine: LISTST für Statusabfrage des LIST-Gerätes Ausgaberoutine: LIST für zeichenweise Ausgabe
PUN:	(PUNCH) Der logische Kanal dient zur zeichenweise Ausgabe. Ausgaberoutine: PUNCH für zeichenweise Ausgabe
RDR:	(READER) Der logische Kanal dient zur zeichenweisen Eingabe. Eingaberoutine: READER für zeichenweise Eingabe

Jedem logischen E/A-Kanal kann genau einer von vier möglichen logischen Subkanälen zugeordnet werden. Diese Zuordnung wird durch den Inhalt des I/O-Bytes auf Adresse 0003H bestimmt. Die Belegung des I/O-Bytes kann z. B. mittels des transienten Programmes STAT geändert werden.

Da alle physischen Gerätetreiber vom Programm im KC-Grundgerät verwaltet werden, wird eine Kopie des I/O-Bytes beim Warmstart im Koppel-RAM übergeben (siehe Anlage 6.4.:). Es gilt folgende Zuordnung zwischen I/O-Byte, Subkanälen und physischen Gerätetreibern:

log. E/A-Kanal	Subkanal	Bits im I/O-Byte 7 6 5 4 3 2 1 0	physischer Treiber
CON:	TTY:	0 0	Standardbildschirmausgabe
	CRT:	0 1	Standardbildschirmausgabe
	BAT:	1 0	Standardbildschirmausgabe
	UC1:	1 1	Standardbildschirmausgabe
RDR:	TTY:	0 0	Standardzusatzeingabe
	PTR:	0 1	(RET)
	UR1:	1 0	USER IN 1
	UR2:	1 1	USER IN 2
PUN:	TTY:	0 0	Standardzusatzausgabe
	PTP:	0 1	Standarddruckerausgabe
	UP1:	1 0	USER OUT 1
	UP2:	1 1	USER OUT 2
LST:	TTY:	0 0	Standarddruckerausgabe
	CRT:	0 1	Standardbildschirmausgabe
	LPT:	1 0	Standardzusatzausgabe
	UL1:	1 1	USER OUT 1

Die physischen Gerätetreiber haben dabei folgende Bedeutung:

Standardbildschirmausgabe:	Ausgabe im wahlweisen Format 80/40 Zeichen mit ESCape-Steuerung
Standarddruckerausgabe:	Ausgabe über den Druckertreiber ab 200H im KC
Standardzusatzein- / -ausgabe:	Arbeit über Treiber ab 380H im KC
USER OUT 1/2:	Ausgabe über vom CAOS-Unterprogramm 2/3 verwalteten Treiber (Hier können bereits unter CAOS verwendete Treiber ohne Veränderung genutzt werden.)
USER IN 1/2:	Eingabe über vom CAOS-Unterprogramm 6/7 verwalteten Treiber (Hier können bereits unter CAOS verwendete Treiber ohne Veränderung genutzt werden.)
(RET):	Eingabe kehrt sofort ohne Zeichen zurück.

7.4. Liste der BIOS-Funktionen

Im Betriebssystem MicroDOS hat der BIOS-Sprungvektor nachfolgend beschriebenen Aufbau. Die symbolischen Sprungadressen dienen nur zum besseren Verständnis der nachfolgenden Beschreibungen.

Sprungnummer	Befehl	Funktion
0	JMP BOOT	; Kaltstartroutine
1	JMP WBOOT	; Warmstartroutine
2	JMP CONST	; CONSOLE-Status abfragen
3	JMP CONIN	; CONSOLE-Eingabe
4	JMP CONOUT	; CONSOLE-Ausgabe
5	JMP LIST	; LIST-Ausgabe
6	JMP PUNCH	; PUNCH-Ausgabe
7	JMP READER	; READER-Eingabe
8	JMP HOME	; Spur Null einstellen
9	JMP SELDSK	; Laufwerk auswählen
10	JMP SETTRK	; Spur auswählen
11	JMP SETSEC	; Sektor auswählen
12	JMP SETDMA	; Datenpufferadresse setzen
13	JMP READ	; Selektierten Sektor lesen
14	JMP WRITE	; Selektierten Sektor schreiben
15	JMP LISTST	; LIST-Status abfragen
16	JMP SECTRAN	; Umrechnen Sektornummer

Die angegebenen Routinen werden als Unterprogramme aufgerufen, enden also mit einem Rücksprung (mit Ausnahme der Warm- und Kaltstartroutine, für die eigene Regeln gelten). Dabei werden eventuell benötigte Parameter in folgenden Prozessorregistern übergeben:

an das BIOS:	8-Bit-Werte in Register C, 16-Bit-Werte im Registerpaar BC, (zweiter 16-Bit-Wert im Registerpaar DE)
vom BIOS:	8-Bit-Werte in Register A, 16-Bit-Werte im Registerpaar HL

Ein Programm kann neben dem Aufruf über die BDOS-Funktion 50, die BIOS-Routinen auch unmittelbar nutzen. Der Ausgangspunkt dazu ist der Sprung auf Adresse 0. Hier befindet sich ein Sprung zur Warmstartroutine (zweite Eintragung im Sprungvektor). Aus der Zieladresse dieses Sprungbefehls und der Nummer der benötigten BIOS-Routine läßt sich leicht die Adresse berechnen, die das Programm gegebenenfalls aufrufen muß:

(<Sprungnummer> -1) * 3 +<Zieladresse des Sprungs auf Adresse 0>

Wenn in einem Programm beispielsweise der Zustand des CONSOLE-Gerätes (Sprungnummer = 2) gebraucht wird, dann kann das mit dem folgenden Unterprogramm geschehen:

```
LD  HL,0     ; HL löschen
LD  DE,1     ; (Nummer der BIOS-Routine) - 1
ADD HL,DE    ; Abstand des Sprungs vom Anfang
ADD HL,DE    ; berechnen
ADD HL,DE
```

```
EX  DE,HL
LD  HL,(1)    ; Adresse Warmstart
ADD HL,DE     ; Adresse der BIOS-Routine
JP  (HL)      ; Sprung zur BIOS-Routine
```

Beachte:
Dieses Unterprogramm kann nicht zur Berechnung der Adresse der Kaltstartroutine verwendet werden!

7.5. Verwaltung der Diskettenlaufwerke

Auf Grund der Vielfalt von Diskettenlaufwerken und Diskettenformaten schließt das BIOS die Möglichkeit der Anpassung an verschiedene Laufwerke und Diskettenformate ein. Deshalb enthält BIOS Tabellen, die dem Nutzer die Disketten- und Laufwerkseigenschaften mitteilen.

Diskettenparameterkopf DPH

Jedem Laufwerk ist ein 16 Byte großer Diskettenparameterkopf (DPH - Disk Parameter Header) zugeordnet, der Informationen über das Diskettenlaufwerk enthält und Arbeitsbereiche für bestimmte BDOS-Operationen einschließt.

Durch die BIOS-Routine SELDSK wird das Laufwerk ausgewählt und außerdem die Adresse des zugehörigen DPH im Registerpaar HL zurückgegeben.

Ein DPH hat folgenden Aufbau:

Byte	Name	Bedeutung
0, 1	XLT	Adresse der Übersetzungstabelle für die Sektornummer. Ist die Adresse gleich 0, dann stimmen logische und physische Sektornummer überein.
2 - 7		Arbeitsbereich für BDOS reserviert
8, 9	DIRBUF	Adresse eines 128-Byte-Verzeichnispuffers. Alle DPH enthalten die gleiche Adresse.
10, 11	DPB	Adresse des Diskettenparameterblockes (DPB). Jedes Laufwerk hat einen eigenen DPB.
12, 13	CSV	Adresse eines Puffers, der für das Speichern eines Prüfsummenvektors zur Prüfung auf Diskettenwechsel erforderlich ist. Jedes Laufwerk hat einen eigenen Puffer.
14, 15	ALV	Adresse eines Vektors, der die Diskettenbelegung wiederspiegelt. Bit n des Vektors gleich 1 bedeutet, daß der Block n der Diskette von einer Datei belegt ist. Bit n gleich 0 bedeutet, daß der Block unbelegt ist. Die ersten Blöcke, und damit die ersten Bits, sind durch das Verzeichnis belegt. Jedes Laufwerk hat einen eigenen Vektor.

Die für die verschiedenen Laufwerke zuständigen DPH stehen lückenlos hintereinander.

Die im DPH erfaßten Daten und Speicherbereiche werden für jedes Laufwerk getrennt bereitgestellt. Eine Ausnahme ist der 128-Byte-Puffer für die Verzeichnisauswertung. Er kann nur einmal im System vorhanden sein, da das

BDOS immer nur ein Laufwerk zur Zeit erfassen kann und bei jeder Laufwerkumschaltung das Verzeichnis neu abfragt.

Diskettenparameterblock DPB

Der Diskettenparameterblock (DPB) für jedes Laufwerk ist wesentlich umfangreicher. In diesem Block sind alle Informationen zusammengefaßt, die zur Verwaltung der betreffenden Diskette notwendig sind.

Dies umfaßt:

- **Informationen zur Speicherkapazität und**
- **Informationen zur Speicherorganisation.**

Der DPB enthält unter anderem:

- **Angaben zur Anzahl von Sektoren pro Spur,**
- **Angaben zur Anzahl von Sektoren pro Block,**
- **Angaben zur Größe und Lage des Verzeichnisses sowie dazu, ob die Verzeichniseinträge bei jedem Zugriff auf Diskettenwechsel überprüft werden sollten und schließlich**
- **eine Angabe zur Anzahl der auf der betreffenden Diskette für das Betriebssystem reservierten Spuren.**

Diese Informationen sind im DPB wie folgt festgehalten:

Byte	Name	Bedeutung
0, 1	SPT	Sektoren pro Spur
2	BSH	Blockverschiebungsfaktor. Darin ist die Blockgröße verschlüsselt als: Log_2 (<Blockgröße>/128) Dieser Wert stellt ein Maß für die Anzahl der Sektoren pro Block dar.
3	BLM	Blockmaske, wiederspiegelt ebenfalls die Blockgröße. Für die Blockmaske gilt: $2^{BSH} - 1$ Zwischen Blockgröße, Blockverschiebungsfaktor und Blockmaske bestehen folgende feste Beziehungen

Blockgröße	BSH	BLM
1024	3	7
2048	4	15
4096	5	31
8192	6	63
16384	7	127

4	EXM	Extentmaske, ist definiert durch die Blockgröße und die Anzahl der Blöcke pro Diskette. Ihre Größe hängt von der Organisation des Verzeichniseintrages ab. Dieser enthält als wesentlichsten Teil für die Speicherverwaltung die Nummern der jeweils belegten Blöcke: 16 Einträge zu je 1 Byte bei weniger als 256 Blöcken pro Diskette oder 8 Einträge zu je 2 Byte bei mehr als 255 Blöcken pro Diskette. Im einzelnen bestehen die Beziehungen:

Blockgröße	Extentmaske für mehr als 255 Blöcke	Extentmaske für weniger als 256 Blöcke
1024	-	0
2048	0	1
4096	1	3
8192	3	7
16384	7	15

5, 6	DSM	Anzahl der Blöcke pro Diskette minus 1 (einschließlich des Verzeichnisses, aber ohne Systemspuren)
7, 8	DRM	Anzahl der Verzeichniseintragungen minus 1. Die Größe einer Verzeichniseintragung beträgt 32 Byte.
9, 10	AL0, AL1	16-Bit-Vektor, in dem die vom Verzeichnis belegten Blöcke vermerkt sind. Dieser Vektor wird beim ersten Laufwerkzugriff an den Anfang der Belegungstabelle kopiert und dient so zur Reservierung der Verzeichnisblöcke. Er ist aus diesem Grund umgekehrt als sonst üblich organisiert: Die Zählung beginnt mit dem höchstwertigen Bit, so hat der Vektor beispielsweise bei vier Verzeichnisblöcken den Wert F000H.
11, 12	CKS	Größe des Verzeichnis-Prüfvektors (Anzahl zu prüfender Verzeichniseintragungen dividiert durch 4)

13, 14	OFF	Anzahl der Systemspuren
14	PSH	physischer Sektorverschiebungsfaktor, darin ist die physische Sektorgröße verschlüsselt als: Log_2 (<Sektorgröße>/128) (außer Laufwerk A)
15	PHM	physische Sektormaske, widerspiegelt ebenfalls die physische Sektorgröße $2^{PSH} - 1$ (außer Laufwerk A)

Unmittelbar an den DPH schließt sich an:

Diskettendefinitionsblock DDB

Die beiden folgenden Tabellen sind nur für die Laufwerke B bis H vorhanden. Der DDB beschreibt physische Kennwerte der Diskette:

Byte	Name	Bedeutung
1	EOT	Nr. des letzten Sektors der Spur
2	GAP	GAP3 Lücke
3	NTR	Anzahl der Spuren

Laufwerkparameterblock DRPB

Byte	Name	Bedeutung
1	PUN	physische Gerätenummer (0...4)
2	DTYP	Drivetyp
3	FTP	erste Spur mit Präkompensation
4	TSS	Schrittzeit
5	HLT	Kopfladezeit
6	CUR	erste Spur mit Schreibstrombegrenzung

Laufwerksteuerung

Die Laufwerksteuerung umfaßt drei Schritte, die zum Adressieren eines Sektors auf der Diskette notwendig sind. Mit Sektor wird im weiteren ein 128 Byte großer Aufzeichnungsabschnitt auf der Diskette bezeichnet.

1. Auswahl des gewünschten Laufwerkes
   (mittels der Routine SELDSK)
2. Schreib-Lese-Kopf auf die Spur setzen, in der sich die Information befindet.
   (mittels der Routine SETTRK)
3. Das Einstellen der Sektornummer erfolgt in zwei Schritten.

Im ersten Schritt erfolgt über die Routine SECTRAN die Umwandlung der logischen Sektornummer in die physische Sektornummer.

Diese Umwandlung ermöglicht eine Diskettenorganisation, die einen zeitoptimalen Zugriff auf die gewünschte Information gewährleistet.

Im zweiten Schritt wird über die Routine SETSEC die Adressierung des physischen Sektors vorgenommen

Datenverkehr

Der Datenverkehr umfaßt neben dem Lesen und Schreiben von Informationen weiterhin die Festlegung der Adresse des Datenpuffers, jenes 128-Byte-Bereiches im Speicher, der die Daten von der Diskette übernimmt bzw. von dem sie kommen.

Liegt der Ort der Aufzeichnung auf der Diskette fest, und ist der Ort des Datenpuffers im Speicher bestimmt, dann können die Daten gelesen oder auf die Diskette geschrieben werden. Dazu bietet das BIOS die Routinen:

- **SETDMA**
  Festlegen des Datenpuffers als Ziel oder Herkunft der Daten.
- **READ**
  Lesen eines 128-Byte-Sektors von der Diskette und Übertragen in den Datenpuffer.
- **WRITE**
  Schreiben der im Datenpuffer vorliegenden Information in den adressierten 128-Byte-Sektor.

RAM-Floppy

Der Zugriff auf das RAM-Floppy geschieht nutzerseitig wie auf jedes andere Laufwerk. Es besitzt eine Spurgröße von 16 KByte, wobei die Anzahl der Spuren vom Grundgerät und der Anzahl der gesteckten Module abhängig ist.

Das RAM-Floppy wird im KC-Grundgerät verwaltet und die Steuerung erfolgt über den Koppel-RAM.

Zur Verwaltung des RAM-Floppys wird im Speicher des KC eine Tabelle angelegt. Diese wird ab Adresse 3CFFH mit fallenden Adressen aufgebaut. Zu jeder Spur (ein 16 KByte-Block) gehören in der Tabelle zwei Byte.

Das höherwertige Byte enthält die Moduladresse des Speicherblockes und das niederwertige das Steuerbyte, um den Block auf der Adresse 8000H 'online' zu schalten.

Beim KC 85/4 bilden die internen Speicher die Spuren 0 und 1. Die Speichermodule werden nach steigenden Moduladressen einbezogen. Da das RAM-Floppy ohne Systemspuren arbeitet, beträgt der Offset stets 0 und das Directory liegt folglich ab Adresse 8000H des entsprechend zugewiesenen Speicherblockes.

Die Directory- und Blockgröße richten sich nach der Gesamtgröße des RAM-Floppys. Beim Kaltstart wird nur das jeweils erste Byte einer Directory-Eintragung im RAM-Floppy auf E5H gesetzt. Die Dateien selbst sowie deren Namen und dessen Blockzuordnungen bleiben unbeeinflußt. Somit ist es möglich, Dateien im RAM-Floppy mit einem geeigneteten universellen Dienstprogramm (/16/) nach einem Kaltstart zu regenerieren.

Das Directory befindet sich auf Spur 0, ab Sektor 1. Pro Eintrag stehen 32 Byte zur Verfügung. Ein Sektor kann vier Directory-Einträge enthalten. Die

Einträge beginnen auf den Adressen 00H, 20H, 40H... Die folgende Tabelle enthält die Bedeutung der einzelnen Bytes in einem Directory-Eintrag.

Byte	Bedeutung
0	0E5H -> Datei gelöscht 00H bis 0FH User-Bereich (von 0 bis 15)
1 - 8	Dateiname (ASCII-Zeichen), zusätzlich kann Bit 7 gesetzt sein. Bedeutung des Bits 7 bei einigen Dienstprogrammen: für Byte 1: Kennzeichnung der Datei als Quelle beim Kopieren (optional). Beim Auflisten des Directorys erscheint zwischen Name und Typ anstelle des Punktes ein >-Zeichen. für Byte 2: Kennzeichnung der Datei als Ziel beim Kopieren (optional). Beim Auflisten des Directorys erscheint zwischen Name und Typ anstelle des Punktes ein <-Zeichen (auch, wenn Bit 1 gesetzt ist). Byte 3 - 8: ohne Bedeutung, können beliebig genutzt werden
9 - 0BH	Dateityp (ASCII-Zeichen), zusätzlich kann Bit 7 gesetzt sein. Bedeutung des Bits 7 für Byte 9: Schreibschutz für Byte 0AH: Systemfile für Byte 0BH: ohne Bedeutung, kann beliebig genutzt werden
0CH	Extentnummer; Belegt eine Datei mehr als acht Blöcke, sind mehrere Directory-Einträge nötig. Diese werden bei 0 beginnend durch die Extentnummer festgelegt.
0DH - 0EH	00H
0FH	Anzahl der Sektoren, die durch diesen Directory-Eintrag belegt werden (maximal 80H, je 10H pro Block) Der letzte Block muß nicht voll belegt sein.
10H - 1FH	Blockadressen; In der angegebenen Reihenfolge ist die Datei auf dem RAM-Floppy verteilt. Je nach Kapazität werden ein oder zwei Byte je Block belegt.

Der Directory-Aufbau ist auf den anderen Laufwerken prinzipiell geich, wobei sich jedoch das Directory installationsabhängig meist auf Spur 2 befindet. Die Anzahl der Directory-Einträge und deren maximale Anzahl sind vom Diskettenformat abhängig (vgl. DPB).

8. Systemuhr

Die Systemuhr von MicroDOS arbeitet mit den CTC-Kanälen 0 und 1 (Abschnitt 2.2.). Kanal 0 ist als Zähler mit der Konstante 125 bei einem Eingangstakt von 500 kHz programmiert. Der Kanal 1 zählt die Ausgangsimpulse des Kanals 0 mit der Zählkonstante 125 und liefert Sekundeninterrupts. Mit diesen werden die Uhrzellen auf der Seite 0 aktuallisiert. Die Zeit ist in gepackten BCD-Zahlen codiert. Die Adresse 40H enthält die Stunden, 41H die Minuten und 42H die Sekunden.

TEIL B - CAOS-BETRIEBSART

1. Systemstart

Der Systemstart der CAOS-Betriebsart erfolgt, wie der Start der PC-Betriebsart mittels 'JUMP FC'. *) Beim 'JUMP FC' wird der CAOS-ROM abgeschaltet und das Programm im ROM des FLOPPY DISK BASIS gestartet. Dieses löst folgende Aktivitäten aus:

*) Werden in der CAOS-Betriebsart Speichermodule verwendet, die den Speicherbereich von C000H bis FFFFH belegen (z. B. TEXOR oder 64 KByte-RAM), dürfen diese erst nach dem Systemstart zugeschaltet werden. Vor einem Wiederstart mit 'JUMP FC 0' müssen diese Module ebenfalls 'off line' geschaltet werden.

- Löschen des Koppel-RAM
- Kopieren des Urladeprogrammes in den Koppel-RAM
- Freigabe des Prozessors im FLOPPY DISK BASIS
- Warten auf Quittung im Koppel-RAM; bei negativer Quittung Fehlermeldung und Abbruch (Fehlermelderoutine wird auf Adresse 0 im KC abgearbeitet) bei positiver Quittung und Meldung, daß CAOS-Diskette gestartet wurde, Programmfortsetzung (bei PC-Betriebsart siehe dort)
- Kopieren FLOAD ab Adresse 0
- Zuschalten ROM im Grundgerät
- CAOS-Warmstart durch Sprung auf E000H

Die Vorgänge im Prozessorsystem des FLOPPY DISK BASIS beginnen mit der Freigabe des RESET. Der Prozessor löscht als erstes den dRAM und beginnt mit der Abarbeitung des Urladeprogrammes ab Adresse FC00H. Der Urlader lädt die ersten 512 Byte der CAOS-Diskette vom Laufwerk 0 in den dRAM ab Adresse 9000H. Anschließend erfolgt die Quittierung im Koppel-RAM. Diese 512 Byte stellen den BOOT-Lader für das Diskettenbetriebssystem der CAOS-Betriebsart dar. Die Aufgabe des BOOT-Laders ist es, das Betriebssystem aus den zwei Systemspuren der Diskette zu laden und zu starten.

Nach der Aktivierung des Betriebssystems wird die Datei INITIAL.SUB geladen. Diese löst das Laden und Starten des Diskettenerweiterungsprogrammes (DEP) aus. Alle weiteren Zugriffe auf die Diskette erfolgen über das Kommando FLOAD bzw. nachladbare Programme.

2. Schnittstellenbeschreibung DEP

Die Programmschnittstelle der Diskettenerweiterung im CAOS ist für Maschinenproramme nutzbar und gestattet somit, die Diskettenarbeit in Anwenderprogramme einzubinden. Die Schnittstelle liegt im Koppel-RAM.

2.1. Belegung Koppel-RAM

Durch Manipulation und Auswertung bestimmter Bytes im Koppel-RAM kann über das Programm DEP, welches im FLOPPY DISK BASIS im Hintergrund läuft, auf die Diskette zugegriffen werden. Im Koppel-RAM werden folgende Bytes verwendet:

I/O-Adresse HIGH	LOW	Bedeutung
80	F3H	Steuerbyte
81	F3H	Fehlercode
82	F3H	Dateibezeichnungs- und Directory-Puffer max. 39 Byte
80	F2H	Sektorpuffer 128 Byte
B	C	- Register

Der Dateibezeichnungspuffer beträgt maximal 12 Byte. Davon werden max. acht Byte für den Namen, ein Byte für den Punkt und drei Byte für den Dateityp verwendet. Ist der Name kürzer als acht Zeichen, so werden Leerzeichen aufgefüllt. Gleiches gilt, wenn der Typ weniger als drei Zeichen enthält. Ist der Name genau acht Zeichen lang, so kann der Punkt entfallen.

Fehlender Dateityp beim Laden und Retten wird durch KCC ersetzt. Der Dateityp COM wird beim Retten in KCC übertragen.

2.2. Funktionen der Bits im Steuerbyte

Alle Aktivitäten werden durch Setzen bestimmter Bits im Steuerbyte ausgelöst. Dabei ist stets Bit 0 zu setzen. Das rückgesetzte Bit 0 signalisiert die abgeschlossene Operation. Die Bits haben folgende Bedeutungen:

Bit	Bedeutung
0	= 1 Anforderung = 0 bedeutet Operation ausgeführt
1	= 1 Schreiben auf Diskette = 0 Lesen
2,4,5	Funktionsauswahl lt. Tabelle
3	Open - Eröffnen eines Dateizugriffs
6	Close - Abschluß eines Dateizugriffs
7	= 1 bei der Rückmeldung bedeutet Fehler = 0 kein Fehler

Funktionsauswahl:

Bit 5	4	2	Bedeutung
0	0	0	- Sektor lesen/schreiben (sequentiell)
0	0	1	- DIR-Anforderung
0	1	0	- ERA
0	1	1	- STAT
1	0	0	- REN
1	0	1	- SET R/O Schreibschutz setzen
1	1	0	- SET R/W Schreibschutz aufheben
1	1	1	- Byte lesen/schreiben für BASIC

Sektor schreiben:

Open: 0BH
Close: 43H
normal: 03H

Sektor lesen:

Open: 09H
Close: 41H
normal: 01H

Directory-Anforderung:

Open: 0DH -> 39 Zeichen ab 82F3H
nächste Einträge: 05H -> 39 Zeichen ab 82F3H
Rückmeldung 04H im Steuerbyte = Ende

Datei löschen:

Name eintragen
Steuerbyte: 11H

Diskettenstatusanzeige:

Steuerbyte:15H
Text ab 82F3H 20 Zeichen

Datei umbenennen:

Alter Name im Puffer
Open: 29H
Neuer Name im Puffer
Steuerbyte: 21H

Schreibschutz setzen:

Name im Puffer
Steuerbyte: 25H

Schreibschutz aufheben:

Name im Puffer
Steuerbyte: 31H

BASIC-Schnittstelle:

Steuerbytes:

35H Byte-Lesen
37H Byte-Schreiben
3BH OPEN-Lesen
3FH OPEN-Schreiben
75H CLOSE-Lesen

77H CLOSE-Schreiben
Ist beim Close-Schreiben Bit 7 bei der Anforderung gesetzt, wird als Dateiendekennzeichen der RUN-SWITCH eingetragen.

2.3. Fehlermeldungen

Ist eine Operation nicht erfolgreich verlaufen, so wird das Bit 7 im Steuerbyte bei rückgesetztem Bit 0 gesetzt. In der darauffolgenden Speicherzelle ist die Codierung des Fehlers enthalten. Es gelten folgende Fehlercodes:

Code	Bedeutung
00	Datei nicht vorhanden
01	Diskettenfehler allgemein: Laufwerk nicht bereit Lesen bzw. Schreiben nicht möglich Diskettenverzeichnis voll Diskette schreibgeschützt
02	Diskette voll
03	Datei schreibgeschützt
04	falsches Diskettenformat
08	Dateibezeichnung bereits vorhanden
09	Dateibezeichnung nicht eindeutig (? enthalten)
43	Prüfsummenfehler (CRC) - Sektor fehlerhaft
49	Indexfehler - kein Spuranfang auffindbar
52	Laufwerk nicht bereit
53	Diskette schreibgeschützt
56	ID-Feld-Fehler - Sektor nicht auffindbar
77	unzulässige Dateibezeichnung (z. B. Steuerzeichen oder Codes > 80H)
81	Dateiende überschritten (z. B. stimmen Dateilänge und Eintragung im ersten Sektor nicht überein)
82	Diskette voll

Bei fehlerhaften Dateien sollte die Diskette in der PC-Betriebsart mit einem universellen Dienstprogramm (z. B. DIENST) untersucht werden.

3. Dateiaufbau

In der CAOS-Betriebsart besitzen die Dateien prinzipiell den gleichen Aufbau wie in der PC-Betriebsart, um einen Dateiaustausch zu realisieren. Die Directory-Einträge entsprechen denen des MicroDOS. In den verschiedenen Anwenderbereichen bestehen jedoch einige Besonderheiten gegenüber Dateien in der PC-Betriebsart, die nachfolgend beschrieben werden.

3.1. Maschinenprogramme

Programme, die mit FSAVE oder der Option 'S' des EDAS (Modul M027) erzeugt wurden, besitzen vor den eigentlichen Daten einen Vorsektor (128 Byte), der analog zur Magnetbandaufzeichnung unter CAOS die folgenden Informationen enthält:

Byte	Inhalt
16	Anzahl der Parameter (2 oder3)
17	Anfangsadresse LOW
18	Anfangsadresse HIGH
19	Endeadresse +1 LOW
20	Endeadresse +1 HIGH
21	Startadresse (optional) LOW
22	Startadresse (optional) HIGH

Bei von EDAS erzeugten Programmen sind zusätzlich in den ersten 11 Byte der Name und der Typ eingetragen.

3.2. BASIC-Dateien

BASIC-Dateien (Programme und Datenfelder) enthalten keinen Vorsektor. Mit der CSAVE-Anweisung erzeugte Dateien sind in der internen Darstellung abgespeichert und somit Editoren nicht zugänglich.

Dateien, die mit LIST#1 bzw. PRINT#1 abgespeichert wurden, sind ASCII-Dateien und somit auch in der PC-Betriebsart, z. B. mit dem Textprozessor, bearbeitbar. Das Endekennzeichen ist 03H und wird somit nicht von allen Programmen (z. B. TP) als solches erkannt.

3.3. TEXOR-Dateien

TEXOR-Dateien entsprechen in ihrem Aufbau den im Punkt 3.1. beschriebenen Maschinenprogrammen, es ist also ein Vorsektor enthalten. Die Texte selbst sind in der TEXOR-eigenen Darstellung abgespeichert und somit dem Textprozessor nicht direkt zugänglich.

Textzeilen im TEXOR besitzen keine Endekennzeichen. Absätze sind durch ein Byte 00 getrennt. Umlaute entsprechen nicht dem ASCII.

3.4. EDAS-Quellprogramme

EDAS-Quellprogramme enthalten einen Vorblock, der in den ersten 11 Byte den Dateinamen und den Dateityp 'ASM' beinhaltet. Das Dateiendekennzeichen ist 03H.

3.5. FORTH-Quellprogramme

FORTH-Dateien besitzen keinen Vorsektor. Sie enthalten den Quelltext, wobei ein Screen vier Sektoren belegt. Der Quelltext enthält keine Steuerzeichen.

4. Einbindung der CAOS-Betriebsart in Anwenderprogramme

4.1. Nutzung von SERVICE im BASIC

Das Programm SERVICE enthält einen Programmteil zur Nutzung der Komponenten von SERVICE von der BASIC-Interpreterebene aus. Die Programme werden mit der CALL-Anweisung aufgerufen. Es gelten folgende Adressen zum Aufruf mit den angegebenen Parameterübergaben:

CALL*D8 FLOAD

Dateiname und -typ müssen ab Adresse 0 'gepoket' werden. Programmbeispiel:

```
10 N$="TEST.KCP    ":FORZ=0TO11
20 POKEZ,ASC(MID$(N$,Z+1,1)):NEXT
30 CALL*D8:!LADEN BILDINHALT
```

CALL*DB FSAVE

Name und Typ müssen ab Adresse 0 'gepoket' werden. Die Argumente müssen ab Adresse 0B781H - 14209 'gevpoket' werden.

ARGN - Anzahl der Argumente 2 oder 3
ARG1 - Ladeadresse
ARG2 - Endeadresse+1
ARG3 - Startadresse (bei Bedarf)

Das Programmbeispiel rettet den Bildschirminhalt einschließlich Farbattributspeicher des KC 85/3:

```
10 AN=14209:!ARGN
20 VPOKE AN,2:VPOKE AN+1,0:VPOKE AN+2,128
30 VPOKE AN+3,0:VPOKE AN+4,178
40 N$="TEST.KCP    ":FOR Z=0 TO 11
50 POKE Z,ASC(MID$(N$,Z+1,1)):NEXT
60 CALL*DB:!RETTEN BILDINHALT
```

CALL*DE DIR

Auflisten des Disketteninhaltes auf dem Bildschirm

CALL*E1 STAT

Anzeige des freien Speicherplatzes auf der Diskette

CALL*E4 REN

Umbenennen einer Datei, wobei erst der alte und danach der neue Name angefordert werden

CALL*E7 SETRO

Setzen des Schreibschutzes für Datei, deren Name angefordert wird

CALL*EA SETWR

Aufheben des Schreibschutzes für Datei, deren Name angefordert wird

CALL*ED ERA

Löschen einer Datei, deren Name angefordert wird

Das Programm SERVICE kopiert beim Programmstart einen Programmteil in den Speicherbereich D8H - FFH. Darf dieser Speicherbereich z. B. für Maschinenprogramme nicht überschrieben werden, kann das Programm SERVICE ohne Selbststart abgespeichert werden:

FSAVE BE00 C000
Name:SERVICE1

Beim Laden von FLOAD auf die Adresse 0 wird der Programmteil auf Adresse D8H überschrieben!

4.2. Nutzung von SERVICE in Maschinenprogrammen

Das Programm SERVICE besitzt eine Schnittstelle, die den Aufruf der Programmkomponenten unterstützt. Sind der Bildwiederholspeicher abgeschaltet und ein Anwenderstack definiert, so kann die oben für BASIC beschriebene Schnittstelle auch in Maschinenprogrammen genutzt werden.

Anderenfalls steht als Einsprungadresse die BE00H zur Verfügung. Das Unterprogramm wird über die Programmnummer im Register A ausgwählt:

UP-Nr.

0 FLOAD
Name muß ab Adresse 0 eingetragen werden.

1 FSAVE
Name muß ab Adresse 0 eingetragen werden. Die Argumente müssen ab Adresse 0B781H eingetragen werden.
ARGN - Anzahl der Argumente 2 oder 3
ARG1 - Ladeadresse
ARG2 - Endeadresse+1
ARG3 - Startadresse (bei Bedarf)

2 DIR
Auflisten des Disketteninhaltes auf dem Bildschirm

3 STAT
Anzeige es freien Speicherplatzes auf der Diskette

4 REN
Umbenennen einer Datei, wobei erst der alte und danach der neue Name angefordert werden

5 SETRO
Setzen des Schreibschutzes für Datei, deren Name angefordert wird

6 SETWR
Aufheben des Schreibschutzes für Datei, deren Name angefordert wird

7 ERA
Löschen einer Datei, deren Name angefordert wird

Alle Register können nach dem Aufruf verändert sein!

4.3. Nutzung der Programme FLOAD und FSAVE in Maschinenprogrammen

Die auf der CAOS-Diskette enthaltenen Programme FLOAD.KCC und FSAVE.KCC können von Maschinenprogrammen aufgerufen oder in diese eingebunden werden. Sie sind speicherverschieblich.

Die Routine FLOAD kann folgendermaßen aufgerufen werden:

```
LD       HL,NAME     ;Zeiger auf Namen
CALL     FLOAD+36H   ;Ladeadresse (Offset) +36H
```

Die Routine FSAVE kann folgendermaßen aufgerufen werden:

```
LD       HL,ARGX      ;aktuelle Parameter
LD       DE,ARGN      ;Systemarbeitszellen
LD       BC,7         ;Anzahl der Bytes
LDIR                  ;Umladen
LD       HL,NAME      ;Zeiger auf Name
CALL     FSAVE+22H    ;Ladeadresse (Offset) +22H

ARGX:   DEFB     3            ;3 Argumente, d.h. Selbststart
        DEFW     ANFANG       ;Anfangsadresse
        DEFW     ENDE         ;Endeadresse +1
        DEFW     START        ;Startadresse

NAME:   DEFM     'TEST01.KCA  ' ;Dateiname
```

Anlage 1: Reservierte Speicherplätze

Speicherplätze von	bis	Inhalt
0000H	0002H	Sprung zum BIOS-Eintrittspunkt WBOOT. Damit ist ein einfacher programmierter Neustart (Sprung zu Adresse 0) möglich.
0003H		Enthält das IOBYTE
0004H		Nummer des aktuellen Laufwerkes und Benutzernummer
0005H	0007H	Enthält eine Sprunganweisung zum BDOS. Die Sprunganweisung liefert einmal den Haupteintrittspunkt in das BDOS und zum anderen stellt die Sprungadresse die niedrigste vom Betriebssystem verwendete Speicheradresse dar. Debugger verändern die Sprungadresse, um den durch sie reduzierten Speicher zu kennzeichnen.
0008H	0037H	RST 1 bis RST 6, von MicroDOS nicht verwendet
0038H	003AH	RST 7, Enthält während Debug-Mode eine Sprunganweisung in den Debugger für programmierte Haltepunkte, wird vom Betriebssystem aber nicht benutzt
003BH	005BH	reserviert
0040H	0042H	Systemuhr
005CH	007FH	durch CCP erzeugter Standard-FCB
0080H	00FFH	Standard-Datenpuffer

Anlage 2: Bildschirmsteuerzeichen

Hex. code	Dez. code	Wirkung des Kommandos
01	1	Setzen Cursor auf linke obere Ecke (HOME-Position)
07	7	BEEP
08	8	Setzen Cursor um eine Position nach links
0A	10	Setzen Cursor um ein Zeile nach unten
0C	12	Bildschirm löschen und Cursor auf linke obere Ecke setzen
0D	13	Setzen Cursor auf Zeilenanfang
14	20	Löschen ab Cursorposition bis Bildschirmende
15	21	Setzen Cursor um eine Position nach rechts
16	22	Löschen ab Cursorposition bis Zeilenende
18	24	Löschen Cursorzeile und setzen Cursor an den Zeilenanfang
1A	26	Setzen Cursor um eine Zeile nach oben
1B	27	Einleiten der direkten Cursorposition ESC, Zeile+80H, Spalte+80H
7F	127	Löschen Zeichen auf Cursorposition und Cursor um eine Position nach links
82	130	Cursor einschalten
83	131	Cursor ausschalten

Anlage 3: ESCape-Funktionen

Dez.	Hex.	dez.	Bedeutung	Name	Anz.	Parameter
A	41H	65	Punktsetzen	PSET	3	XX,Y
B	42H	66	Punktlöschen	PRES	3	XX,Y
C	43H	67	Grafikfarbe	GFARB	1	F
D	44H	68	Linie	LINE	6	XX1,Y1,XX2,Y2
E	45H	69	Kreis	CIRCL	4	XX,Y,R
F	46H	70	Fensteraufruf	WINDOW	1	N
G	47H	71	Tonausgabe	SOUND	6	T1,V1,T2,V2,L,Z
H	48H	72	Textfarbe	COLORZ	2	F,H
I	49H	73				
J	4AH	74				
K	4BH	75				
L	4CH	76	Fenster initialisieren	WININI	5	N,X1,Y1,X2,Y2
M	4DH	77	Grafikschirm löschen	CLSG	0	-
N	4EH	78	Vordergrund wechseln	INK	1	F
O	4FH	79	Hintergrund wechseln	BACK	1	H
P	50H	80	Bildschirmmode wechseln	BSMODE	0	-
Q	51H	81	1 Byte lesen	READ1	2	AA
R	52H	82	N*256 Bytes lesen	READN	3	AA,N
S	53H	83	1 Byte schreiben	WRITE1	3	AA,B
T	54H	84	NN Bytes schreiben	WRITEN	4+	AA,NN,(NN*B)
U	55H	85	Unterprogramm aufrufen	GOSUB	2	AA
V	56H	86	Direktausgabe Zeichencode	CAOSBS	1	B
W	57H	87	Ausgabe USERPORT	USOUT2	1	B
X	58H	88	Ausgabe USERPORT	USOUT3	1	B
Y	59H	89				
Z	5AH	90				
Ä	5BH	91	CAOS-Sprungverteiler	CAOSP	1	N
Ö	5CH	92	Rückkehr ins CAOS	EXIT	0	-
Ü	5DH	93	Wechsel am./dt.	ZSATZ	0	-
^	5EH	94	PAGE/ SCROLL-Mode	SCRMOD	0	-
_	5FH	95	KEYBORD kl/gr	KBDMOD	0	-
`	60H	96	Wechsel IRM-Ebene	IRMEB	1	S
a	61H	97	IRM-Auflösung hoch	IRMHI	0	-
b	62H	98	IRM-Auflösung niedrig	IRMLO	0	-
c	63H	99	SWITCH	SWITCH	2	M,S
d	64H	100	Punkttesten	PTEST	2	XX
e	65H	101	Funktionstaste belegen	FKEY	1	N
	80H - D0H	128 - 208	Locate Zeile, Spalte			

Bedeutung der Parameter:

1 Zeichen = 1 Byte
Doppelzeichen = 2 Byte (low/high)

AA	**Adresse**	**T**	**Tonhöhe**
R	**Radius**	**L**	**Lautstärke**
Z	**Tondauer**	**N**	**Nummer**
S	**Steuerbyte**	**B**	**Byte**
H	**Hintergrund**	**M**	**Moduladresse**
XX	**X-Koordinate**	**F**	**Farbcodes**
Y	**Y-Koordinate**	**V**	**Vorteiler**

Anlage 4: Koppel-RAM-Belegung

```
Seite 3 - FF00H = I/O-Adresse F3H im KC

fünf  32 Byte-Puffer                          Adresse im D004
CIBUFF:  defs 32   ;Konsoleneingabe                FF00H
COBUFF:  defs 32   ;Konsolenausgabe                FF20H
LOBUFF:  defs 32   ;Druckerausgabe                 FF40H
AIBUFF:  defs 32   ;Zusatzeingabe (normal V.24 )   FF60H
AOBUFF:  defs 32   ;Zusatzausgabe (normal V.24 )   FF80H

fünf Ringpufferzeiger
INPTR:  defb  LOW(CIBUFF)    ;KC                   FFA0H
        defb  LOW(CIBUFF)    ;D004
        defb  0
        defb  0
        defb  LOW(AIBUFF)    ;KC
        defb  LOW(AIBUFF)    ;D004
OUTPTR: defb  LOW(COBUFF)                          FFA6H
        defb  LOW(COBUFF)
        defb  LOW(LOBUFF)
        defb  LOW(LOBUFF)
        defb  LOW(AOBUFF)
        defb  LOW(AOBUFF)

IOBYTE als Kopie des System-I/O-Bytes, wird bei jedem Warm-
start aktualisiert
IOBYTE: defb 0                                     FFACH
AIANF:  defb 0               ;Anforderung Zusatzeingabekanal
                             ; <>0 schaltet z.B. DTR
                             ; Eingabe -> 0        FFADH
MEMANF:   defb 0             ;Anforderung zur Speicherinhalts-
                             ;übertragung
                             ; <>0 fordert Daten an
                             ; (über ESCape's)
                             ; Daten im Puffer -> 0
                             ; Fehler -> FFH       FFAEH

RAM-Floppy-Zellen
TRACK:    defb 0    ;Spur max. 256    a 16K Byte   FFAFH
SECTOR:   defb 0    ;Sektor max. 127  a 128 Byte   FFB0H
CONTR:    defb 0    ;Steuerbyte - Lesen      04H
                                - Schreiben  06H
                                - Quittiert  00H   FFB1H
SIZE:     defb 0    ;verfügbare Spuranzahl         FFB2H
UROK:     defb 0    ; 0 - Anfangswert
                    ; 1 - o.k.
                    ; 2 - floppy not ready
                    ; 3 - can't read
                    ; 4 - no system
                    ; 5 - CAOS-Betriebsart
KTABAD:   defw 0    ; Anfangsadresse Tastaturcode  FFB3H
                    ; dient der Änderung der Umcodierungstabelle
BSSTAT:   defb 0    ; STATUS - Bildschirmausgabe
                    ; BIT 0 - 80/40Zeichen
                    ;     1 - US/DT.
                    ;     2 - groß/klein
                    ;     3 - SCROLL/PAGE
ESCTAB:   defw 0    ; Adresse der ESCape-Tabelle

Seite 2  FE00H - FEFFH
```

128 Bytekoppelpuffer z.B. RAM-Floppy oder

256 Bytekoppelpuffer für RAM-Lesen

Seite 1 FD00H - FDFFH

frei verfügbar

Seite 0 FC00H - FCFFH

von MicroDOS verwendet

Anlage 5: BDOS-Übersicht

Nr.	Name	Eingangsparameter	Ausgangspararameter
0	System rücksetzen	-	-
1	Konsoleneingabe	-	A = Zeichen
2	Konsolenausgabe	E = Zeichen	-
3	Zusatzeingabe	-	A = Zeichen
4	Zusatzausgabe	E = Zeichen	-
5	Druckerausgabe	E = Zeichen	-
6	Direkte Konsolen-E/A	siehe Definition	
7	Status Zusatzeingabe	-	A = 00/FF
8	Status Zusatzausgabe	-	A = 00/FF
9	String ausgeben	DE = Puffer	-
10	String einlesen	DE = Puffer	siehe Definition
11	Konsolenstatus	-	A = 00/FF
12	Versionsnummer	-	HL = Versionsnr.
13	Disk rücksetzen	-	siehe Definition
14	Laufwerk wählen	E = Laufwerk	siehe Definition
15	Datei eröffnen	DE = FCB	A = Verzeichniscode
16	Datei schließen	DE = FCB	A = Verzeichniscode
17	erste Datei suchen	DE = FCB	A = Verzeichniscode
18	nächste Datei suchen	-	A = Verzeichniscode
19	Datei löschen	DE = FCB	A = Verzeichniscode
20	sequentiell lesen	DE = FCB	A = Fehlercode
21	sequentiell schreiben	DE = FCB	A = Fehlercode
22	Datei erzeugen	DE = FCB	A = Verzeichniscode
23	Datei umbenennen	DE = FCB	A = Verzeichniscode
24	Anwahlvektor holen	-	HL = Vektor
25	aktuelles Laufwerk	-	A = aktuelles Laufwerk
26	DMA-Adresse setzen	DE = DMA	-
27	Belegungsvektor holen	-	HL = Vektoradresse
28	Schreibschutz setzen	-	siehe Definition
29	Schreibschutzvektor holen	-	HL = Schreibschutzvektor
30	Dateiattribute	DE = FCB	siehe Definition
31	Parameteradresse holen	-	HL = DPB-Adresse
32	Benutzercode	siehe Definition	siehe Definition
33	wahlfrei lesen	DE = FCB	A = Fehlercode
34	wahlfrei schreiben	DE = FCB	A = Fehlercode
35	Dateigröße berechnen	DE = FCB	r0, r1, r2
36	Datensatz setzen	DE = FCB	r0, r1, r2
37	Disketten rücksetzen	DE = Diskettenvektor	A = 0
40	wahlfrei schreiben mit Auffüllen Nullen	DE = FCB	A = Fehlercode
45	Fehlermodus setzen	DE = Modus	-
46	Diskettenplatz bestimmen	E = Laufwerk	Sektorenanzahl
47	Programm wechseln	Kommando	-
49	Systemparameter	DE = Parameteradresse	HL = Parameter
50	BIOS-Funktionen	DE = Parameteradr	BIOS - spezifisch
108	Rückkehrcode	DE = Rückkehrcode	Rückkehrcode
110	Begrenzer	DE =	Begrenzer

		FFFH/Begrenzer	
111	Puffer anzeigen	DE = Parameteradresse	-
112	Puffer drucken	DE = Parameteradresse	-
152	FCB erzeugen	DE = Paramteradresse	FCB

Anlage 6: Programmbeispiele

Im folgenden sollen einige Beispiele die Nutzung der Systemfunktionen verdeutlichen. Die Quelldateien werden mit Hilfe des Textprozessors erstellt und dann mit dem Assembler (ASM in /2/) in REL-Dateien übersetzt. Mit dem Linker (LINK) können sie dann in ablauffähige COM-Programme umgesetzt werden, welche direkt unter MicroDOS arbeiten. Besonderen Wert wurde in den Beispielen auf die Nutzung der ESCape-Funktionen der Bildschirmausgabe und die Erstellung eigener Gerätetreiberprogramme gelegt.

6.1: DUMP-Ausgabe

Das erste Beispielprogramm ist allgemeiner Natur und läuft auch unter allen SCP bzw. CP/M-Systemen. Dieses Ausgabeprogramm liest eine Eingabedatei, welche im CCP-Kommando angegeben wird, und gibt diese auf der Konsole in hexadezimaler Form aus. Das Programm rettet den CCP-Stackpointer beim Eintritt, setzt den Stackpointer auf einen lokalen Stack und stellt den CCP-Stackpointer vor der direkten Rückkehr zum CCP wieder her. Das heißt, es wird kein Warmstart am Ende des Programms durchgeführt.

```
                        ;
                        ; Dump-Ausgabeprogramm
                        ;
                                .z80
0005                    bdos    equ   0005h     ; BDOS-Eintritt
0001                    cons    equ   1         ; Konsole lesen
0002                    typef   equ   2         ; Konsole schreiben
0009                    printf  equ   9         ; Puffer ausgeben
000B                    brkf    equ   11        ; Konsolenstatus
000F                    openf   equ   15        ; Datei eröffnen
0014                    readf   equ   20        ; Daten lesen
                        ;
005C                    fcb     equ   005ch     ; FCB-Adresse
0080                    buff    equ   0080h     ; Eingabepuffer
                        ;
                        ; Steuerzeichen
000D                    cr      equ   0dh       ; Wagenrücklauf
000A                    lf      equ   0ah       ; Zeilenvorschub
                        ;
                        ; FCB-Definitionen
005C                    fcbdn   equ   fcb+0     ; Diskettenname
005D                    fcbfn   equ   fcb+1     ; Dateiname
0065                    fcbft   equ   fcb+9     ; Dateityp
0068                    fcbr1   equ   fcb+12    ; Erweiterung
006B                    fcbrc   equ   fcb+15    ; Datensatzanzahl
007C                    fcbcr   equ   fcb+32    ; Datensatzzähler
007D                    fcbln   equ   fcb+33    ; FCB-Länge
                        ;
                        ; Stack setzen
0000'   21 0000                 ld    hl,0
0003'   39                      add   hl,sp
                        ; Stackpointer des CCP in HL
0004'   22 00F5'                ld    (oldsp),hl
                        ; SP auf lokalen wert setzen
0007'   31 0137'                ld    sp,stktop
                        ; Puffer einlesen und ausgeben
000A'   CD 00C1'                call  setup     ; Datei eröffnen
000D'   FE FF                   cp    255       ; Datei vorhanden?
000F'   C2 001B'                jp    nz,openok ; Springen wenn ok
                        ;
                        ; Datei nicht vorhanden, Fehlermeldung
```

```
0012'   11 00DD'                ld    de,opnmsg
0015'   CD 009C'                call  err
0018'   C3 0051'                jp    finis     ; Rückkehr
                        ;
                        ; eröffnung ok,pufferzeiger auf ende setzen
001B'   3E 80           openok:  ld    a,80h
001D'   32 00F3'                ld    (ibp),a   ; zeiger = 80h

                        ; HL auf nächste Adresse setzen
0020'   21 0000                 ld    hl,0      ; bei Null beginnen
                        ;
0023'   E5              gloop:   push  hl        ; Zeile retten
0024'   CD 00A2'                call  gnb
0027'   E1                      pop   hl        ; Zeile wiederholen
0028'   DA 0051'                jp    c,finis   ; Carry wird von gnb
                        ;                         bei Dateiende gesetzt
002B'   47                      ld    b,a
                        ; Hexzeichen ausgeben, Zeile testen
002C'   7D                      ld    a,l
002D'   E6 0F                   and   0fh       ; 4 Bit testen
002F'   C2 0044'                jp    nz,nonum
                        ; Zeilennummer ausgeben
0032'   CD 0072'                call  crlf
                        ; Konsole auf Abbruch testen
0035'   CD 0059'                call  break
                        ; Akkumulator-LSB=1, falls Zeichen bereit
0038'   0F                      rrca            ; Carry-Bit setzen
0039'   DA 0051'                jp    c,finis   ; Abbruch
                        ;
003C'   7C                      ld    a,h
003D'   CD 008F'                call  phex
0040'   7D                      ld    a,l
0041'   CD 008F'                call  phex
                        ;
0044'   23              nonum:   inc   hl
0045'   3E 20                   ld    a,' '
0047'   CD 0065'                call  pchar
004A'   78                      ld    a,b
004B'   CD 008F'                call  phex
004E'   C3 0023'                jp    gloop
                        ;
                        ; Ende der Ausgabe, zurück zum CCP
                        ;
0051'   CD 0072'        finis:   call  crlf
0054'   2A 00F5'                ld    hl,(oldsp)
0057'   F9                      ld    sp,hl
                        ; Stackpointer enthält CCP-Stackadresse
0058'   C9                      ret             ; zurücK zum CCP
                        ;
                        ; Unterprogramme
                        ;
                        ; Konsole auf Abbruch testen (Zeichen beliebig)
0059'   E5              break:   push  hl
005A'   D5                      push  de
005B'   C5                      push  bc   ; Register retten
005C'   0E 0B                   ld    c,brkf
005E'   CD 0005                 call  bdos
0061'   C1                      pop   bc
0062'   D1                      pop   de
0063'   E1                      pop   hl
0064'   C9                      ret
                        ;
                        ; Zeichen ausgeben
```

```
0065'   E5              pchar:   push  hl
0066'   D5                       push  de
0067'   C5                       push  bc  ; Register retten
0068'   0E 02                    ld    c,typef
006A'   5F                       ld    e,a
006B'   CD 0005                  call  bdos
006E'   C1                       pop   bc
006F'   D1                       pop   de
0070'   E1                       pop   hl  ; Register zurück
0071'   C9                       ret
                         ;
                         ; neue Zeile
0072'   3E 0D           crlf:    ld    a,cr
0074'   CD 0065'                call  pchar
0077'   3E 0A                    ld    a,lf
0079'   CD 0065'                call  pchar
007C'   C9                       ret
                         ;
                         ; Hex-Zeichen in A ausgeben
007D'   E6 0F           pnib:    and   0fh       ; untere 4 Bits
007F'   FE 0A                    cp    10
0081'   D2 0089'                 jp    nc,p10
                         ; kleiner gleich 9
0084'   C6 30                    add   a,'0'
0086'   C3 008B'                 jp    prn
                         ; größer oder gleich 10
0089'   C6 57           p10:     add   a,'a'-10
008B'   CD 0065'         prn:     call  pchar
008E'   C9                       ret
                         ;
                         ; Register A in Hex-Format ausgeben
008F'   F5               phex:    push  af
0090'   0F                       rrca
0091'   0F                       rrca
0092'   0F                       rrca
0093'   0F                       rrca
0094'   CD 007D'                 call  pnib
0097'   F1                       pop   af
0098'   CD 007D'                 call  pnib
009B'   C9                       ret
                         ;
                         ; Fehlermeldung ausgeben, DE adressiert die
                         ; Meldung, welche mit "$" endet
009C'   0E 09            err:     ld    c,printf  ; Puffer ausgeben
009E'   CD 0005                  call  bdos
00A1'   C9                       ret
                         ;
                         ; nächstes Byte holen
00A2'   3A 00F3'         gnb:     ld    a,(ibp)
00A5'   FE 80                    cp    80h
00A7'   C2 00B3'                 jp    nz,g0
                         ;
                         ; neuen datensatz lesen
00AA'   CD 00CE'                 call  diskr
00AD'   B7                       or    a         ; 0 wenn lesen ok
00AE'   CA 00B3'                 jp    z,g0      ; nächstes Byte
                         ; Ende der Daten, Carry setzen für EOF
00B1'   37                       scf
00B2'   C9                       ret
                         ;
                         ; Byte auf Adresse buff+<a> lesen
00B3'   5F               g0:      ld    e,a       ; puffer index
00B4'   16 00                    ld    d,0       ; 16 bit-wert
```

```
00B6'   3C                      inc   a         ; index erhöhen
00B7'   32 00F3'                ld    (ibp),a   ; und abspeichern
                        ; Zeiger ist erhöht und gerettet
00BA'   21 0080                 ld    hl,buff
00BD'   19                      add   hl,de

                        ; HL enthält absolute Adresse des nächsten
                        ; Zeichens
00BE'   7E                      ld    a,(hl)
                        ; Byte ist im Akkumulator
00BF'   B7                      or    a         ; carry rücksetzen
00C0'   C9                      ret
                        ;
                        ; Eingabedatei eröffnen
00C1'   AF              setup:  xor   a         ; Akku löschen
00C2'   32 007C                 ld    (fcbcr),a ; "cr" löschen
                        ;
00C5'   11 005C                 ld    de,fcb
00C8'   0E 0F                   ld    c,openf
00CA'   CD 0005                 call  bdos
                        ; Akkumulator enthält 255 bei Fehler
00CD'   C9                      ret
                        ;
                        ; Datensatz lesen
00CE'   E5              diskr:  push  hl
00CF'   D5                      push  de
00D0'   C5                      push  bc
00D1'   11 005C                 ld    de,fcb
00D4'   0E 14                   ld    c,readf
00D6'   CD 0005                 call  bdos
00D9'   C1                      pop   bc
00DA'   D1                      pop   de
00DB'   E1                      pop   hl
00DC'   C9                      ret
                        ; Feld für konstante Zeichenketten
00DD'   6E 6F 20 69     opnmsg: defm  'no input file present$'
00E1'   6E 70 75 74
00E5'   20 66 69 6C
00E9'   65 20 70 72
00ED'   65 73 65 6E
00F1'   74 24
                        ;
                        ; Variablenfeld
00F3'                   ibp:    defs  2 ; eingabe puffer zeiger
00F5'                   oldsp:  defs  2 ; ccp stack pointer
                        ;
                        ; lokaler Stack
00F7'                           defs  64; 32 niveaus reserviert
0137'                   stktop:
                        ;
```

6.2: Byteanzeige des Grundgerätes

Das zweite Beispielprogramm erwartet beim Aufruf nach dem Namen eine vierstellige Hexzahl als Adresse. Werden weniger als vier Ziffern oder keine Hexziffern eingegeben, so erfolgt eine Fehlermeldung. Bei korrekter Eingabe wird die eingegebene Adresse, gefolgt von deren Speicherinhalt, des KC-Grundgerätes ausgegeben. Abgeschlossen ist das Programm mit dem Warmstart.

```
                        ; LESEN EINES BYTES AUS DEM SPEICHER
                        ; DES KC UND ANZEIGE
                              .Z80
0005                    BDOS   EQU    5
0080                    DMA    EQU    80H
001B                    ESC    EQU    1BH
FFAE                    MEMANF EQU    0FFAEH  ;SPEICHERANFORDERUNG
FE00                    PUFFER EQU    0FE00H  ;ÜBERGABEPUFFER
                        ;
0000'   31 00D3'        START:LD      SP,STACK ;LOKALER STACK
0003'   2A 0001               LD      HL,(1)  ;WARMSTART
0006'   11 0009               LD      DE,9    ;CONOUT
0009'   19                    ADD     HL,DE
000A'   22 009A'              LD      (CALAD+1),HL
000D'   21 0082               LD      HL,DMA+2;HEXZAHL
0010'   11 0000               LD      DE,0
0013'   06 04                LD      B,4     ;4 DIGIT
0015'   7E             ST1:  LD      A,M     ;UMWANDLUNG ASCII->HEX
0016'   23                    INC     HL      ;NÄCHSTES BYTE
0017'   D6 30                SUB     30H     ;KEINE HEXZAHL
0019'   38 5B                JR      C,STERR
001B'   FE 0A                CP      0AH     ;DEZIMALZIFFER ?
001D'   38 06                JR      C,ST3
001F'   D6 07                SUB     7       ;A-F
0021'   FE 10                CP      16      ;KEINE HEXZIFFER ?
0023'   30 51                JR      NC,STERR
0025'   CB 23             ST3:  SLA     E
0027'   CB 12                  RL      D
0029'   CB 23                  SLA     E
002B'   CB 12                  RL      D
002D'   CB 23                  SLA     E
002F'   CB 12                  RL      D
0031'   CB 23                  SLA     E       ;4*RECHTSSCHIEBEN(DE)
0033'   CB 12                  RL      D
0035'   83                    ADD     A,E     ;ZIFFER NACH BIT 0-3
0036'   5F                    LD      E,A
0037'   10 DC                 DJNZ    ST1
0039'   3E 01             ST4:  LD      A,1     ;SETZEN FLAG
003B'   32 FFAE               LD      (MEMANF),A
003E'   3E 1B                 LD      A,ESC   ;EINLEITEN ESCAPE
0040'   CD 0095'              CALL    OUT
0043'   3E 51                 LD      A,'Q'   ;1 BYTE LESEN
0045'   CD 0095'              CALL    OUT
0048'   7B                    LD      A,E     ;ADRESSE LOW
0049'   CD 0095'              CALL    OUT
004C'   7A                    LD      A,D     ;ADRESSE HIGH
004D'   CD 0095'              CALL    OUT
0050'   3A FFAE         ST5:  LD      A,(MEMANF)
0053'   A7                    AND     A       ;BYTE BEREIT ?
0054'   20 FA                 JR      NZ,ST5
                        ;AUSGABE ADRESSE
0056'   7A                    LD      A,D     ;ADRESSE HIGH
0057'   CD 0080'              CALL    HEXB
```

```
005A'   7B                      LD      A,E     ;ADRESSE LOW
005B'   CD 0080'                CALL    HEXB
005E'   3E 20                   LD      A,' '   ;SPACE
0060'   CD 0095'                CALL    OUT
                        ;ABHOLEN BYTE
0063'   3A FE00                LD      A,(PUFFER)
                        ;AUSGABE DES HEX-ZEICHENS
0066'   CD 0080'                CALL    HEXB
0069'   3E 0A                   LD      A,0AH
006B'   CD 0095'                CALL    OUT
006E'   3E 0D                   LD      A,0DH
0070'   CD 0095'                CALL    OUT
0073'   C3 0000            WARM: JP      0       ;WARMSTART
                        ;
                        ;FEHLERAUSGABE
0076'   11 00A0'          STERR:LD      DE,TXT  ;FEHLERMELDUNG
0079'   0E 09                   LD      C,9     ;STRINGAUSGABE
007B'   CD 0005                 CALL    BDOS
007E'   18 F3                   JR      WARM
                        ;
0080'   F5                HEXB: PUSH    AF
0081'   0F                      RRCA
0082'   0F                      RRCA
0083'   0F                      RRCA
0084'   0F                      RRCA
0085'   CD 0089'                CALL    HEX
0088'   F1                      POP     AF
0089'   E6 0F                HEX:  AND     0FH
008B'   FE 0A                   CP      10
008D'   30 04                   JR      NC,HEX1
008F'   C6 30                   ADD     A,'0'
0091'   18 02                   JR      OUT
                        ;
0093'   C6 37              HEX1: ADD     A,'A'-10
                        ;BIOS-AUFRUF
0095'   E5                OUT:  PUSH    HL
0096'   D5                      PUSH    DE
0097'   C5                      PUSH    BC
0098'   4F                      LD      C,A
0099'   CD 0000           CALAD:CALL    0       ;BIOS-RUF
009C'   C1                      POP     BC
009D'   D1                      POP     DE
009E'   E1                      POP     HL
009F'   C9                      RET
                        ;
00A0'   20 4B 45 49       TXT:  DEFM    ' KEINE ADRESSE !'
00A4'   4E 45 20 41
00A8'   44 52 45 53
00AC'   53 45 20 21
00B0'   0A0D                    DEFW    0A0DH
00B2'   24                      DEFM    '$'
                        ;
00B3'                           DEFS    32      ;LOKALER STACK
00D3'                   STACK EQU     $
                                END
```

6.3: Datei in Speicher des Grundgerätes kopieren

Dieses Programmbeispiel zeigt, wie eine Datei von der Diskette in den Speicher des KC-Grundgerätes kopiert werden kann. Der Aufruf erfolgt über den Programmnamen (z. B. MOVE) mit nachfolgender Dateibezeichnung. An diese anschließend kann eine hexadezimale Zieladresse (vierstellig) eingegeben werden. Fehlt diese, wird als Vorzugsadresse 4000H angenommen.

Eine Datei darf somit nicht länger als 16 KByte sein. Wird in andere Speicherbereiche übertragen, so hat der Prorammierer selbst zu überwachen, daß keine Proramme überschrieben werden. Mit diesem Programm ist es einfach möglich, in der PC-Betriebsart für die CAOS-Betriebsart erstellte Maschinenprogramme im Speicher des Grundgerätes verfügbar zu machen.

```
                        ;
                        ; UEBERTRAGEN DATEI IN KC-SPEICHER
                        ; AB EINGEGEBENER ADRESSE ODER AB 4000H
                              .Z80
0005                    BDOS  EQU     5
005C                    FCB   EQU     5CH     ;NAME
006C                    FCB1  EQU     6CH     ;ADRESSE HEX
0080                    DMA   EQU     80H
001B                    ESC   EQU     1BH
4000                    ANFAD EQU     4000H   ;ANFANGSADRESSE FUER
                                              ;UEBERTRAGUNG
                        ;
0000'  31 011D'         START:        LD      SP,STACK ;LOKALER STACK
0003'  2A 0001                LD      HL,(1)  ;WARMSTART
0006'  11 0009                LD      DE,9
0009'  19                     ADD     HL,DE
000A'  22 00B1'               LD      (CALAD+1),HL
000D'  21 006D                LD      HL,FCB1+1
0010'  11 0000                LD      DE,0    ;ERFASSEN ADRESSE
0013'  06 04                  LD      B,4
0015'  7E              ST1:   LD      A,M
0016'  23                     INC     HL
0017'  D6 30                  SUB     30H
0019'  38 1E                  JR      C,ST2
001B'  FE 0A                  CP      0AH
001D'  38 06                  JR      C,ST3
001F'  D6 07                  SUB     7
0021'  FE 10                  CP      16
0023'  30 14                  JR      NC,ST2
0025'  CB 23           ST3:   SLA     E
0027'  CB 12                  RL      D
0029'  CB 23                  SLA     E
002B'  CB 12                  RL      D
002D'  CB 23                  SLA     E
002F'  CB 12                  RL      D
0031'  CB 23                  SLA     E
0033'  CB 12                  RL      D
0035'  83                     ADD     A,E
0036'  5F                     LD      E,A
0037'  10 DC                  DJNZ    ST1
0039'  78              ST2:   LD      A,B
003A'  FE 04                  CP      4
003C'  20 03                  JR      NZ,ST4
003E'  11 4000                LD      DE,ANFAD        ;STANDARDADRESSE
0041'  ED 53 00DB'     ST4:   LD      (MERAD),DE      ;MERKEN !
0045'  0E 0F                  LD      C,15    ;OPEN
0047'  11 005C                LD      DE,FCB
004A'  CD 0005                CALL    BDOS
```

```
004D'   3C                      INC     A
004E'   CA 00B7'                JP      Z,S7
0051'   AF                      XOR     A       ;cr=0
0052'   32 007C                 LD      (FCB+32),A
0055'   21 011D'                LD      HL,PUFFER
0058'   0E 14            S1:   LD      C,20    ;SEQU. LESEN
005A'   11 005C                LD      DE,FCB
005D'   E5                      PUSH    HL
005E'   CD 0005                CALL    BDOS
0061'   E1                      POP     HL
0062'   A7                      AND     A       ;DATEIENDE ?
0063'   20 0C                   JR      NZ,S20
0065'   11 0080                 LD      DE,DMA
0068'   EB                      EX      DE,HL
0069'   01 0080                 LD      BC,128
006C'   ED B0                   LDIR            ;UMLADEN
006E'   EB                      EX      DE,HL
006F'   18 E7                   JR      S1
0071'                   ;
0071'   3E 1B            S20:  LD      A,ESC   ;HL= ENDE
0073'   CD 00AC'                CALL    OUT
0076'   3E 54                   LD      A,'T'   ;UEBERTRAGUNG in KC
0078'   CD 00AC'                CALL    OUT
007B'   3A 00DB'                LD      A,(MERAD)       ;ADRESSE LOW
007E'   CD 00AC'                CALL    OUT
0081'   3A 00DC'                LD      A,(MERAD+1)     ;         HIGH
0084'   CD 00AC'                CALL    OUT
0087'   11 011D'                LD      DE,PUFFER
008A'   A7                      AND     A
008B'   ED 52                   SBC     HL,DE   ;BERECHNUNG DER LAENGE
008D'   7D                      LD      A,L
008E'   CD 00AC'                CALL    OUT     ;LAENGE LOW
0091'   7C                      LD      A,H
0092'   CD 00AC'                CALL    OUT     ;       HIGH
0095'   1A                SL:   LD      A,(DE)
0096'   CD 00AC'                CALL    OUT
0099'   2B                      DEC     HL
009A'   13                      INC     DE
009B'   7D                      LD      A,L
009C'   B4                      OR      H
009D'   20 F6                   JR      NZ,SL
009F'   3E 0A                   LD      A,0AH   ;LF
00A1'   CD 00AC'                CALL    OUT
00A4'   3E 0D                   LD      A,0DH   ;CR
00A6'   CD 00AC'                CALL    OUT
00A9'   C3 0000           WARM: JP      0       ;WARMSTART
                        ;
00AC'   E5                OUT:  PUSH    HL
00AD'   D5                      PUSH    DE
00AE'   C5                      PUSH    BC
00AF'   4F                      LD      C,A
00B0'   CD 0000           CALAD:CALL    0       ;BIOS
00B3'   C1                      POP     BC
00B4'   D1                      POP     DE
00B5'   E1                      POP     HL
00B6'   C9                      RET
00B7'   11 00C1'          S7:   LD      DE,TX2
00BA'   0E 09                   LD      C,9
00BC'   CD 0005                 CALL    5
00BF'   18 E8                   JR      WARM
                        ;
                        ;
00C1'   20 44 61 74       TX2:  DEFM    ' Datei nicht gefunden !'
```

```
00C5'   65 69 20 6E
00C9'   69 63 68 74
00CD'   20 67 65 66
00D1'   75 6E 64 65
00D5'   6E 20 21
00D8'   0A0D                    DEFW    0A0DH
00DA'   24                      DEFM    '$'
00DB'   4000            MERAD:  DEFW    ANFAD   ;MERKZELLE
00DD'                           DEFS    64
011D'                   STACK   EQU     $
011D'                   PUFFER  EQU     $       ;LADEPUFFER
                                END
```

6.4: Druckertreiber

Ein Beispiel für einen vom Installationsprogramm MSYSG verwaltbaren Druckertreiber, ist eine CENTRONICS-Routine für den Modul M001 (/13/). Dieser Treiber ist für die Modulschachtadresse 8 ausgelegt, kann aber durch Änderung der Speicherzelle MODSCH angepaßt werden. Die darauffolgenden acht Byte wurden aus Kompatibilitätsgründen zu den V24-Treibern freigehalten.

Zu beachten ist, daß Programme für das Grundgerät am günstigsten mit der .PHASE-Anweisung auf die Zieladresse übersetzt werden, da MicroDOS-Programme im Allgemeinen ab Adresse 100H lauffähig sind. Beim Linken muß das Programm auf Adresse 100H gebunden werden. Für das unten stehende Beispiel kann folgendermaßen verfahren werden:

```
>LINK
*CENTR.LST/N/P:100
*CENTR/E
```

```
                                .Z80
                               .PHASE  200H
                            ;**************************************
                            ; CENTRONIC- TREIBER
                            ;**************************************
                            ;VEREINBARUNGEN:
F003                        CAOS    EQU     0F003H
0026                        MODU    EQU     26H     ;UP-NR. MODUL
0000                        CRT     EQU     0       ;UP-NR. CRT
                            ;SYSTEMZELLEN CAOS
                            ;
00EF                        MKENN   EQU     0EFH    ;MODULKENNBYTE DIO
0004                        PIOA    EQU     04H     ;PIO KANAL A DAT
0006                        PIOAC   EQU     06H     ;    STEUERWORT
0005                        PIOB    EQU     05H     ;PIO B DATEN
0007                        PIOBC   EQU     07H     ;    STEUERWORT
                            ;
                            ;**************************************
                            ;PARAMETERUEBERGABE BEI AUFRUF:
                            ;CENTRON MODULSCHACHT USER-OUT
                            ;         8 u.s.w.       2/3
                            ;**************************************
0200    18 0B                       JR      START   ;SELBSTSTARTADR.
0202    18 31                       JR      SDD
                            ;--------------------------------------
0204    08                  MODSCH:DEFB   8       ;MODULSCHACHT
0205                                DEFS    8       ;FREI
020D    3A 0204             START:LD      A,(MODSCH)
0210    47                          LD      B,A
0211    0E 80                       LD      C,80H
0213    ED 78                       IN      A,(C)
0215    FE EF                       CP      MKENN
0217    C0                          RET     NZ
                            ;
0218    68                  ST1:  LD      L,B
0219    3E 02                       LD      A,2
021B    16 01                       LD      D,1
021D    5A                          LD      E,D
021E    CD F003                     CALL    CAOS
0221    26                          DEFB    26H       ;ZUWEISUNG M001
0222    3E FF                       LD      A,0FFH
0224    D3 06                       OUT     (PIOAC),A ;BIT-MODE
0226    AF                          XOR     A
0227    D3 06                       OUT     (PIOAC),A ;AUSGABE
```

```
0229    3D                      DEC     A
022A    D3 07                   OUT     (PIOBC),A ;BIT-MODE
022C    3E 04                   LD      A,4
022E    D3 07                   OUT     (PIOBC),A ;BIT 2 EINGABE
0230    3E 01                   LD      A,1
0232    D3 05                   OUT     (PIOB),A  ;STROBE PASSIV
0234    C9                      RET
                        ;**************************************
                        ; ZEICHENAUSGABE ZEICHEN IN A
0235                    SDD:
                        ;DIREKTE BYTEAUSGABE
0235    C5              SD1:    PUSH    BC      ;AUSGABE DATEN
0236    F5                      PUSH    AF
0237    DB 05           SDA:    IN      A,(PIOB)
0239    CB 57                   BIT     2,A
023B    28 08                   JR      Z,SDB   ;BUSY?
023D    3E 05                   LD      A,5
023F    CD F003                 CALL    CAOS
0242    14                      DEFB    14H     ;WARTEN
0243    18 F2                   JR      SDA
                        ;
0245    F1              SDB:    POP     AF
0246    F5                      PUSH    AF
0247    D3 04                   OUT     (PIOA),A  ;DATEN
0249    AF                      XOR     A
024A    D3 05                   OUT     (PIOB),A  ;STROBE-
024C    3C                      INC     A
024D    D3 05                   OUT     (PIOB),A  ;IMPULS
024F    F1                      POP     AF
0250    C1                      POP     BC
0251    C9                      RET
                                .DEPHASE
                                END
```

6.5: Kopplungstreiber

Das Beispielprogramm für den Koppeltreiber entspricht dem Programm V24H12.KOP der Systemdiskette. Zum Assemblieren und Linken gilt das zum Druckertreiber Gesagte.

```
                        .Z80
                    ; V 24 TREIBER FUER KOPPLUNG
                    ; UEBER HARDWAREPROTOKOLL
                    ;*************************************
                    ;VEREINBARUNGEN:
F003                CAOS  EQU     0F003H          ;CAOS-SPRUNGVERTEILER
0026                MODU  EQU     26H             ;UP NR.CAOS- SWITCH
                    ;
00EE                MKENN EQU     0EEH            ;STRUKTURBYTE M003
0008                SIO   EQU     08H             ;ADRESSE SIO-DATEN
0009                SIOB  EQU     SIO+1           ;          -STEUER
000D                CTC1  EQU     0DH             ;CTC-KANAL1
                    ;
                          .PHASE  380H
0380    18 0D       INITK:        JR      INITD           ;INITIALISIERUNG
0382    18 34       KOPOUT:       JR      SDDK            ;BYTE-AUSGABE
0384    18 41       KOPIN:        JR      SDE             ;BYTE-EINGABE
0386    47          INTB2:        DEFB    47H             ;BETRIEBSART CTC
0387    2E                  DEFB    2EH             ;ZEITKONST.CTC
                                                    ; 1200 BAUD !!
0388    18          INTBS2:       DEFB    18H             ;->WR4
0389    04                  DEFB    4
038A    44                  DEFB    44H
038B    03                  DEFB    3               ;->WR3
038C    E1                  DEFB    0E1H
038D    05                  DEFB    5               ;->WR5
038E    6A          INTBE2:       DEFB    6AH             ;8BIT
038F    01 0880     INITD:        LD      BC,880H
0392    ED 78       NEXT: IN      A,(C)
0394    FE EE               CP      0EEH            ;V24 ?
0396    28 04               JR      Z,FOUND         ;GEFUNDEN
0398    04                  INC     B               ;WEITER SUCHEN
0399    20 F7               JR      NZ,NEXT
039B    C9                  RET
                    ;
039C    3E 02       FOUND:        LD      A,2
039E    16 01               LD      D,1
03A0    5A                  LD      E,D
03A1    68                  LD      L,B
03A2    CD F003             CALL    CAOS            ;ZUWEISEN M003
03A5    26                  DEFB    MODU
03A6    0E 0D               LD      C,CTC1
03A8    06 02               LD      B,INTBS2-INTB2  ;COUNTER
03AA    21 0386             LD      HL,INTB2
03AD    F3                  DI
03AE    ED B3               OTIR
03B0    0E 0B               LD      C,SIOB+2        ;INIT SIO
03B2    06 07               LD      B,INTBE2-INTBS2+1
03B4    ED B3               OTIR
03B6    FB                  EI
03B7    C9                  RET
                    ;************************************
                    ;DIREKTE BYTEAUSGABE
03B8    F5          SDDK: PUSH    AF
03B9    DB 0B               IN      A,(SIOB+2)
03BB    CB 57               BIT     2,A             ;PUFFER LEER ?
```

```
03BD    28 05          JR      Z,SD6K          ;->NEIN
03BF    F1             POP     AF
03C0    D3 09          OUT     (SIOB),A        ;BYTE AUSGEBEN
03C2    A7             AND     A               ;CY=0 READY
03C3    C9             RET
                ;
03C4    F1      SD6K:  POP     AF              ;CY=1 NOT READY
03C5    37             SCF
03C6    C9             RET
                ;************************************
                ; ZEICHENEINGABE ZEICHEN IN A
                ;RR0 BIT 0 - ZEICHEN IM EMPFAENGERPUFFER
                ;WR5 BIT 0 - EMPFAENGERFREIGABE
03C7    DB 0B   SDE:   IN      A,(SIOB+2)
03C9    CB 47          BIT     0,A             ;BYTE IM PUFFER ?
03CB    20 11          JR      NZ,SD7          ;->JA
03CD    3E 05          LD      A,5     ;WR5
03CF    D3 0B          OUT     (SIOB+2),A
03D1    3E EA          LD      A,0EAH
03D3    D3 0B          OUT     (SIOB+2),A      ;DTR-ON
03D5    DB 0B   SDE9:  IN      A,(SIOB+2)      ;RR0
03D7    CB 47          BIT     0,A             ;ZEICHEN IM PUFFER ?
03D9    20 07          JR      NZ,SDE6         ;->NEIN
03DB    CD 03EA SDE5:  CALL    OFF
03DE    DB 09   SD7:   IN      A,(SIOB)        ;BYTE EINLESEN
03E0    A7             AND     A
03E1    C9             RET
                ;
03E2    DD 7E 00 SDE6:         LD      A,(IX+13)
03E5    FE 03          CP      3               ;BRK ?
03E7    20 EC          JR      NZ,SDE9         ;->NEIN
03E9    37             SCF                     ;CY=1 BREAK
03EA    3E 05   OFF:   LD      A,5
03EC    D3 0B          OUT     (SIOB+2),A
03EE    3E 6A          LD      A,06AH
03F0    D3 0B          OUT     (SIOB+2),A      ;DTR OFF
03F2    C9             RET
                ;************************************
                       .DEPHASE
                       END
```

6.6: Tastaturbelegung

Das Programm zur Tastaturbelegung entspricht dem Programm TYPEMOD.COM der Systemdiskette. Das Beispiel soll folgendes verdeutlichen:

- Aufbau der Tastencodetabelle
- Bestimmen der Anfangsadresse der Tastencodetabelle
- Übertragung der Tastencodetabelle in den entsprechenden Speicherbereich des Grundgerätes

```
                                .Z80
                                TITLE   KEYBELEGUNG
                                SUBTTL  TEXTMODUS
                          ;DISKETTENPROGRAMM ZUR BELEGUNG DER TASTATUR
                          ;   SCHREIBMASCHINENMODUS
                          ;  AUFRUF :  TYPEMOD
  001B                    ESC   EQU     1BH     ;ESCAPE
  0005                    BDOS  EQU     5       ;SYSTEMRUF
  0000                    WARM  EQU     0       ;WARMSTART
  FFB4                    KTABAD        EQU     0FFB4H  ;ADRESSE DER
TASTATUR-
                                                ;TABELLE IM KC
  0000'   2A 0001         KEYNEW: LD    HL,(1)  ;WARMSTART
  0003'   11 0009                 LD      DE,9
  0006'   19                      ADD     HL,DE
  0007'   22 004E'                LD      (CALAD+1),HL    ;BIOS-AUSGABE
  000A'   11 0054'                LD      DE,STEXT
  000D'   0E 09                   LD      C,9
  000F'   CD 0005                 CALL    BDOS            ;STRINGAUSGABE
  0012'   ED 5B FFB4              LD      DE,(KTABAD)     ;KOPPEL-RAM
  0016'   21 00BD'                LD      HL,KTAB         ;NEUE TABELLE
  0019'   06 C0                   LD      B,KTABE-KTAB    ;LÄNGE
  001B'   3E 1B           KEYBEL:         LD      A,ESC
  001D'   CD 0049'                CALL    OUT
  0020'   3E 53                   LD      A,'S'           ;BYTEAUSGABE
  0022'   CD 0049'                CALL    OUT             ;SPEICHERUEBERGABE
  0025'   7B                      LD      A,E             ;ADRESSE LOW
  0026'   CD 0049'                CALL    OUT
  0029'   7A                      LD      A,D             ;ADRESSE HIGH
  002A'   CD 0049'                CALL    OUT
  002D'   13                      INC     DE
  002E'   7E                      LD      A,(HL)
  002F'   23                      INC     HL
  0030'   CD 0049'                CALL    OUT             ;AUSGABE BYTE
  0033'   10 E6                   DJNZ    KEYBEL
  0035'   3A FFB6                 LD      A,(0FFB6H)      ;US/DT ?
  0038'   CB 4F                   BIT     1,A
  003A'   20 0A                   JR      NZ,WST1
  003C'   3E 1B                   LD      A,1BH
  003E'   CD 0049'                CALL    OUT
  0041'     3E      5D                            LD      A,5DH
;UMSCHALTUNG AUF DT.
  0043'   CD 0049'                CALL    OUT
  0046'   C3 0000         WST1:  JP      WARM            ;RÜCKSPRUNG
                          ;
  0049'   E5              OUT:   PUSH    HL              ;BIOS-AUFRUF
  004A'   D5                     PUSH    DE              ;AKTUELLE ADRESSE
  004B'   C5                     PUSH    BC              ;WIRD BEIM START
  004C'   4F                     LD      C,A             ;EINGETRAGEN
  004D'   CD 0000         CALAD:         CALL    0
  0050'   C1                     POP     BC
```

```
  0051'   D1                    POP     DE
  0052'   E1                    POP     HL
  0053'   C9                    RET
                        ;
  0054'    54 61 73 74     STEXT:       DEFM    'Tastatur auf
Schreibmaschinen-
                                                 belegung umgeschaltet'
  0058'   61 74 75 72
  005C'   20 61 75 66
  0060'   20 53 63 68
  0064'   72 65 69 62
  0068'   6D 61 73 63
  006C'   68 69 6E 65
  0070'   6E 62 65 6C
  0074'   65 67 75 6E
  0078'   67 20 75 6D
  007C'   67 65 73 63
  0080'   68 61 6C 74
  0084'   65 74
  0086'   0A0D                  defw    0a0dh
  0088'   46 6F 6C 69           defm    'Folie 2 auflegen !
                                         (Version 12/10/88)'
  008C'   65 20 32 20
  0090'   61 75 66 6C
  0094'   65 67 65 6E
  0098'   20 21 20 20
  009C'   20 20 20 20
  00A0'   20 20 20 20
  00A4'   20 20 20 20
  00A8'   28 56 65 72
  00AC'   73 69 6F 6E
  00B0'   20 31 32 2F
  00B4'   31 30 2F 38
  00B8'   38 29
  00BA'   0A0D                  defw    0a0dh
  00BC'   24                    DEFM    '$'
                        ;
                        ;CODEWANDLUNGSTABELLE FUER TASTATUR
  00BD'   77 57 17      KTAB:   DEFB    'wW',17h
  00C0'   61 41 01              DEFB    'aA',1
  00C3'   32 22 3C              DEFB    '2"<'
  00C6'   08 01 08              DEFB    08H,1,08H       ;CUL: ^H,^A,^H
  00C9'   84 85 86              DEFB    84h,85h,86h     ;Autorep/scr/back
  00CC'   2D 3D 20              DEFB    '-=',20h
  00CF'   F2 F7 F1              DEFB    0F2h,0f7h,0f1h  ;F2
  00D2'   79 59 19              DEFB    'yY',19h
  00D5'   65 45 05              DEFB    'eE',5
  00D8'   73 53 13              DEFB    'sS',13h
  00DB'   33 23 3E              DEFB    '3#>'
  00DE'   7D 5D 20              DEFB    7dh,5dh,20h
  00E1'   10 87 88              DEFB    10h,87h,88h     ;^P/hardcopy/ink
  00E4'   7E 2A 20              DEFB    'ß*',32
  00E7'   F3 F8 F7              DEFB    0F3h,0f8h,0f7h  ;F3
  00EA'   78 58 18              DEFB    'xX',18h
  00ED'   74 54 14              DEFB    'tT',14h
  00F0'   66 46 06              DEFB    'fF',6
  00F3'   35 25 5F              DEFB    '5%_'
  00F6'   70 50 10              DEFB    'pP',10h
  00F9'   7F 19 89              DEFB    7fh,19h,89h     ;DEL/^Y/80-40
  00FC'   30 2B 20              DEFB    '0+ '
  00FF'   F5 FA 20              DEFB    0f5h,0fAh,020h  ;F5
  0102'   76 56 16              DEFB    'vV',16h
  0105'   75 55 15              DEFB    'uU',15h
```

```
0108'   68 48 08            DEFB    'hH',8
010B'   37 27 20            DEFB    '7',27h,32
010E'   6F 4F 0F            DEFB    'oO',0fh
0111'   16 8A 20            DEFB    16h,8ah,20H     ;INS/click/-
0114'   39 29 20            DEFB    '9) '           ;
0117'   03 03 20            DEFB    3,3h,20h        ;BRK/-/-
011A'   6E 4E 0E            DEFB    'nN',0eh
011D'   69 49 09            DEFB    'iI',9
0120'   6A 4A 0A            DEFB    'jJ',0ah
0123'   38 28 20            DEFB    '8(',20h
0126'   20 09 20            DEFB    ' ',9,' '       ;SPACE/TAB
0129'   6B 4B 0B            DEFB    'kK',0bh
012C'   2C 3B 20            DEFB    ',;',32
012F'   13 11 8F            DEFB    13h,11h,8fh     ;hex-input
0132'   6D 4D 0D            DEFB    'mM',0dh
0135'   7A 5A 1A            DEFB    'zZ',1ah
0138'   67 47 07            DEFB    'gG',7
013B'   36 26 20            DEFB    '6&',20h
013E'   20 20 20            DEFB    '   '           ;FEHLERCODE
0141'   6C 4C 0C            DEFB    'lL',0ch
0144'   2E 3A 20            DEFB    '.:',32
0147'   F6 FB 20            DEFB    0F6h,0fBh,020h  ;F6
014A'   62 42 02            DEFB    'bB',2
014D'   72 52 12            DEFB    'rR',12h
0150'   64 44 04            DEFB    'dD',4
0153'   34 24 5E            DEFB    '4$^'
0156'   7B 5B 20            DEFB    'äÄ',32
0159'   7C 5C 20            DEFB    'öÖ',32
015C'   2F 3F 20            DEFB    '/?',20h
015F'   F4 F9 20            DEFB    0F4h,0f9h,020h  ;F4
0162'   63 43 03            DEFB    'cC',3
0165'   71 51 11            DEFB    'qQ',11h
0168'   90 90 90            DEFB    90h,90h,90h     ;SHIFT LOCK
016B'   31 21 40            DEFB    '1!'
016E'   18 03 18            DEFB    18h,3,18h       ;CUD ^x/^c/^x
0171'   05 12 05            DEFB    5,12h,5         ;CUU ^E/^R/^E
0174'   04 06 04            DEFB    4,6,4           ;CUR ^D/^F/^D
0177'   91 1B 91            DEFB    91h,1bh,91h     ;F1= ^/ESC/^
017A'   0D 0D 0D            DEFB    0dh,0dh,0dh     ;CR
017D'                KTABE  EQU     $
                            END
```

Literaturverzeichnis

/1/ Kieser, Meder: "Mikroprozessortechnik". VEB Verlag Technik Berlin

/2/ "Systemhandbuch SCP - Anleitung für den Programmierer", Teil 2 - Assemblerprogrammierung. VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda

/3/ "Bedienungsanleitung und Sprachbeschreibung BASI". VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda

/4/ "Programmtechnische Beschreibung REDABAS". VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda

/5/ "Bedienungsanleitung und Sprachbeschreibung PASCAL 880/S". VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda

/6/ "Bedienungsanleitung BASC". VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda

/7/ Beschreibung zum Modul M027 DEVELOPMENT. VEB Mikroelektronik "W. Pieck" Mühlhausen

/8/ D004 Handbuch für den Bediener. VEB Mikroelektronik "W. Pieck" Mühlhausen

/9/ Beschreibung zum Modul M003 V24. VEB Mikroelektronik "W. Pieck" Mühlhausen

/10/ Beschreibung zur Programmkassette C0171/1. VEB Mikroelektronik "W. Pieck" Mühlhausen

/11/ Systemhandbuch KC 85/3. VEB Mikroelektronik "W. Pieck" Mühlhausen

/12/ Systemhandbuch KC 85/4. VEB Mikroelektronik "W. Pieck" Mühlhausen

/13/ Beschreibung zum Modul M001 DIGITAL IN/OUT. VEB Mikroelektronik "W. Pieck" Mühlhausen

/14/ Böhl, E.: Integrierte Floppy-Disk-Controller-Schaltungen U 8272 D08 und U8272 D04. rfe, Berlin 36 (1987) 11, S.703

/15/ Mikroprozessoren der II. Leistungsklasse, Hefte CPU und CTC. VEB Mikroelektronik "K. Marx" Erfurt

/16/ "Anwendungsbeschreibung und Bedienungsanleitung DIENST". VEB Robotron Büromaschinenwerk "E. Thälmann" Sömmerda