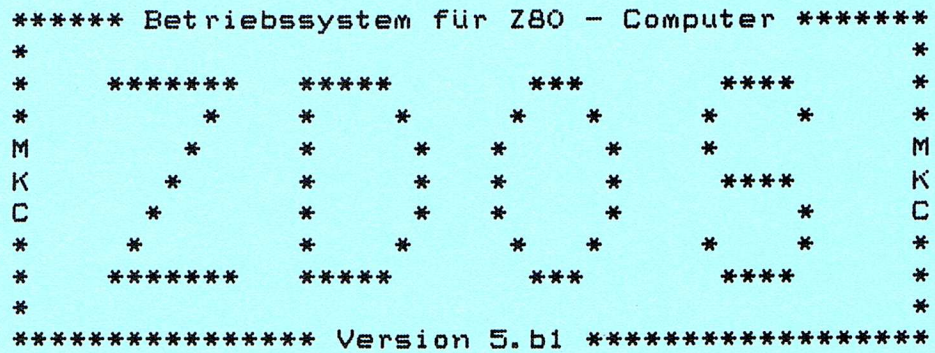

Inhaltsverzeichnis

Programmieren unter LEAS

Dieses Kapitel beschreibt die Programmumgebung für ein unter **ZDOS** laufendes Programm.
Programme werden durch Eingabe ihres Namens im **KI** gestartet. Ihnen steht die Eingabezeile des **KI** zur Verfügung (zur Parameterübergabe).
Ferner können alle Programme die **LEAS**-Routinen durch den Aufruf **CALL 0005H** mitbenutzen. Hierbei sind die **ZDOS**-Konventionen einzuhalten.
Programme werden als Unterprogramme durch den **KI** aufgerufen. Als Programmende sind möglich:
RET, **JP 0000H** oder der **LEAS**-Aufruf **SYSTEM RESET**.
Programme können das **LEAS**-Datei-System voll mitbenutzen. Hierzu ist für den Programmierer die Kenntniss dieses Datei-Systems erforderlich. (**FCB**)

Außerdem besteht noch die Möglichkeit, von Programmen aus die **HEAS**-Routinen aufzurufen. (vgl. **HEAS**-Benutzer-Handbuch)

Diese Möglichkeit sollte aber nur ausgenutzt werden, falls es keine andere Möglichkeit gibt, die gewünschte Funktion zu erreichen. **HEAS**-Aufrufe von Anwenderprogrammen aus können **LEAS** stören.

Um CP/M-kompatible Programme zu erstellen, sind folgende Einschränkungen zu beachten:

- Startadresse (= Ladeadresse) = 0100H
- Kein Aufruf der **LEAS**-Funktionen 25 bis 255
- vor Aufruf des Betriebssystems sind die Register AF, BC, DE und HL zu retten, falls sie später noch benutzt werden sollen
- Da CP/M für den 8080 definiert ist, sollte nur 8080-Code benutzt werden

Aufruf eines Anwenderprogrammes mittels KI

Wie bereits erwähnt, wird ein Programm durch Eingabe seines Dateinamens aufgerufen. Der Dateityp von ausführbaren Programmen muß **.COM** sein. Der Dateityp wird bei Aufruf nicht mit eingegeben.

Beispiel:

Aufruf des Programmes PROG.COM

```
0A)PROG DATEI1 DATEI2 A B C
```

Der Kommandointerpreter lädt das Programm PROG.COM an die im Eintrag auf der Diskette vereinbarte Stelle im Speicher, baut bis zu zwei File Control Blocks ab Adresse **005CH** und **006CH** auf, kopiert die eingegebene Kommandozeile in den Bereich ab Adresse **0080H**, setzt die **DMA**-Adresse auf 0080H, hat das Laufwerk vor dem Prompt als Betriebslaufwerk selektiert und ruft dann das geladene Programm als Unterprogramm auf. Jedes Anwenderprogramm muß mit einem ausführbaren Befehl beginnen. Der Stackpointer ist vom **KI** gesetzt, im Stack sind 10 Bytes frei. Falls mehr Platz benötigt wird, muß der Stackpointer vom Programm neu gesetzt werden. Aufrufe von **LEAS**-Funktionen benötigen keinen zusätzlichen Platz im Stack, da **LEAS** seinen internen Stack benutzt und keine Register zerstört. Ein Programm wird beendet durch den Befehl **RET** (es ist sicherzustellen, daß der Stackpointer wieder auf dem Wert steht, den er bei Aufruf des Programmes hatte), einen Sprung nach Adresse **0000H** oder durch den **LEAS**-Aufruf **SYSTEM RESET**. Die beiden letzten Möglichkeiten führen zu einem Systemwarmstart, d.h. das System wird neu von der Diskette geladen. Bei Beendigung durch **RET** darf der Kommandointerpreter vom Programm nicht überschrieben worden sein. Der Speicherbereich von Adresse **0000H** bis **007FH** ist für **LEAS** reserviert und darf vom Programm nicht verändert werden. Ferner sollten **LEAS** und **HEAS** keinesfalls überschrieben werden.

Die Parameterübergabe im Einzelnen:

Ab der Adresse **5CH** steht der erste File-Control-Block, d.h:

5CH	=	laufwerk	
5DH - 64H	=	dateiname	
65H - 67H	=	dateityp	
68H	=	ex	(immer = 0)
69H	=	ad	(immer = 0)
6AH	=	sy	(immer = 0)
6BH	=	rc	(immer = 0)

Falls kein **FCB** erstellt werden kann, steht:

5CH - 67H	=	blank
68H - 6BH	=	0

Ab der Adresse **6CH** steht der zweite File-Control-Block, d.h:

6CH	=	laufwerk	
6DH - 74H	=	dateiname	
75H - 77H	=	dateityp	
78H	=	ex	(immer = 0)
79H	=	ad	(immer = 0)
7AH	=	sy	(immer = 0)
7BH	=	rc	(immer = 0)

Falls kein **FCB** erstellt werden kann, steht:

6CH - 77H	=	blank
78H - 7BH	=	0

Im **DMA**-Puffer ab Adresse **0080H** steht eine Kopie der Eingabezeile an den **KI**, beginnend mit:

80H	=	Anzahl der gültigen Zeichen
81H - ..	=	Kopie der Eingabezeile, beginnend hinter dem Namen des aufgerufenen Programms.

Bedeutung der Adressen von 0000H bis 00FFH

Innerhalb der **Seite** 0 sind Werte abgelegt, die für den Betrieb von **LEAS** erforderlich sind. Die benötigten Bereiche werden im folgenden beschrieben:

Adresse	Inhalt

00H - 02H	Sprung in **HEAS**-Warmstart. Hieraus läßt sich auch die Adresse der **HEAS**-Sprungleiste mit folgendem Programm bestimmen:
	LD HL,(0001H)
	LD L,0
	Jetzt steht in Register HL die Anfangsadresse der **HEAS**-Sprungleiste.

03H	**IOBYTE** (siehe dort)
04H	Betriebslaufwerk für **KI** (0 für A bis 7 für H) + Benutzernummer im oberen Nibble.
05H - 07H	Sprung in **LEAS.** Hierbei muß in Register C die Funktionsnummer stehen. Ferner läßt sich der Inhalt der Adressen 06 und 07 auch als erste reservierte Speicherstelle interpretieren, d.h. Programme dürfen den Speicher bis zu dieser Adresse benutzen, sie überschreiben dann allerdings den beim Lauf nicht benötigten Kommandointerpreter.
08H - 0AH	reserviert für **ZDOS**-Erweiterung
0BH - 2FH	frei für RST-Befehle
30H - 3FH	reserviert für DEBUG-Programme **PSP** benutzt die Adressen 38H bis 3AH für **TRACE** und **BREAKPOINT.**
40H - 5BH	reserviert
5CH - 7CH	empfohlener File Control Block. (Der **KI** trägt hier bis zu 2 **FCB**'s ein).
7DH - 7FH	**RANDOM**-Erweiterung des **FCB** (falls erforderlich)
80H - FFH	Voreingestellter **DMA**-Puffer, gleichzeitig wird die Kommandozeile beim Aufruf von Programmen vom **KI** nach hier kopiert.

Für das Beispiel **PROG DATEI1 DATEI2 A B C** steht ab Adresse 005CH folgendes:

5CH...6BH	DEFB 0,'DATEI1_____',0,0,0,0
6CH...7BH	DEFB 0,'DATEI2_____',0,0,0,0
80H...92H	DEFB 12H,'_DATEI1 DATEI2 A B C'
93H...FFH	nicht definiert

Hierbei ist der _ jeweils durch ein Leerzeichen zu ersetzen.

Beschreibung der einzelnen LEAS-Funktionen

Funktion 0: SYSTEM RESET

Eingabe: C = 00H
Ausgabe: -

Die Funktion SYSTEM RESET übergibt die Kontrolle an ZDOS. Es meldet sich der Kommandointerpreter. ZDOS wird von Laufwerk A geladen, das Betriebslaufwerk wird wieder ausgewählt und die DMA-Adresse wird auf 80H gesetzt. Die Funktion SYSTEM RESET entspricht einem Sprung auf Adresse 0000H.

Funktion 1: CONSOLE INPUT

Eingabe: C = 01H
Ausgabe: A = ASCII-Zeichen

Die Funktion CONSOLE INPUT liest ein Zeichen von der logischen Konsole in Register A, d.h. sie wartet, bis ein Zeichen eingegeben worden ist. Ausdruckbare Zeichen, Backspace, Carriage return und line feed werden ausgegeben. Tabs werden durch Blanks expandiert, aber als Tab (09H) im Akku übergeben.

Funktion 2: CONSOLE OUTPUT

Eingabe: C = 02H
E = ASCII-Zeichen
Ausgabe: -

Die Funktion CONSOLE OUTPUT sendet das ASCII-Zeichen aus Register E an die logische Konsole. Auch hier werden Tabs durch Blanks expandiert.

Funktion 3: READER INPUT

Eingabe: C = 03H
Ausgabe: A = ASCII-Zeichen

Die Funktion READER INPUT liest ein Zeichen vom logischen Streifenleser in Register A, d.h. sie wartet auf das nächste Zeichen vom Leser.

Funktion 4: PUNCH OUTPUT

Eingabe: C = 04H
E = ASCII-Zeichen
Ausgabe: -

Die Funktion PUNCH OUTPUT sendet ein Zeichen aus Register E an den logischen Streifenstanzer.

Funktion 5: LIST OUTPUT

Eingabe: C = 05H
E = ASCII-Zeichen

Die Funktion LIST OUTPUT sendet ein Zeichen aus Register E an den logischen Drucker.

Funktion 6: DIRECT CONSOLE INPUT/OUTPUT

Eingabe: C = 06H
E = FFH für INPUT bzw. ASCII-Zeichen für OUTPUT
Ausgabe: A = ASCII-Zeichen oder 0 (nur bei INPUT)

Die Funktion DIRECT CONSOLE INPUT/OUTPUT ist für besondere Anwendungen gedacht. INPUT wartet nicht auf eine Eingabe; falls kein Zeichen eingegeben wurde, wird im Akku eine 00H zurückgegeben. Echobetrieb wird umgangen. Falls in Register E beim Aufruf ein FFH steht, bedeutet dies INPUT; alles andere wird als auszugebendes Zeichen angesehen. Die Funktion 6 sollte im Normalfall nicht benutzt werden, da sie alle LEAS-Sonderfunktionen umgeht.

Funktion 7: GET I/O-BYTE

Eingabe: C = 07H
Ausgabe: A = IOBYTE

Die Funktion GET I/O-BYTE übergibt den momentanen Wert von I/O-BYTE im Register A. Das I/O-BYTE enthält die Zuordnungen zwischen logischen und physikalischen Peripheriegeräten.

Funktion 8: SET I/O-BYTE

Eingabe: C = 08H
E = I/O-BYTE
Ausgabe: -

Die Funktion SET I/O-BYTE setzt das I/O-BYTE auf den in Register E gegebenen Wert.

Funktion 9: PRINT STRING

Eingabe: C = 09H
DE = Adresse der Zeichenkette
Ausgabe: -

Die Funktion PRINT STRING sendet die Zeichenkette, deren Anfangsadresse in Register DE steht, an die logische Konsole. Die Zeichenkette endet mit einem Dollar-Zeichen. Das Dollar-Zeichen wird nicht ausgegeben. Tabs werden expandiert.

Funktion 10: READ CONSOLE BUFFER

Eingabe: C = 0AH
DE = Adresse des Zeilenpuffers
Ausgabe: -

Die Funktion READ CONSOLE BUFFER ermöglicht zeilenweise Eingaben von der logischen Konsole. Hierbei sind folgende Editiermöglichkeiten gegeben:

^H oder Backspace löscht das zuletzt eingegebene Zeichen im Speicher und auf der Konsole.
^X löscht die gesamte Zeile im Speicher und auf der Konsole.
RUBOUT oder DELETE hat dieselbe Wirkung wie ^X.
^I oder TAB springt auf die nächste Tabulatorposition (auf der Konsole), im Speicher steht 09H.
CARRIAGE RETURN oder NEW LINE beendet die Eingabe der Zeile. Es erscheint nicht im Speicher.
LINE FEED beendet wie CARRIAGE RETURN eine logische Eingabezeile, erlaubt aber anschließend innerhalb derselben physikalischen Zeile weitere Eingaben auf Vorrat.
^C, als erstes Zeichen in der Eingabezeile, startet das System neu. An allen anderen Positionen hat ^C keine Bedeutung.
^P startet die Protokollierung aller Ein- und Ausgaben auf dem logischen Drucker. Das zweite ^P schaltet die Protokollierung wieder aus.

Beschreibung des Zeilenpuffers:

```
    mx,i ,c1,c2,...,ci,...,cn
DE+ 00 01 02 03     i+1    n+1
```

Hierbei bedeutet:

mx	die Länge des Puffers, ist vor Aufruf zu setzen!
i	die Anzahl der gültigen Zeichen
c1...cn	die Zeichen, c1...ci sind gültig, ci+1...cn sind ungültig.

Funktion 11: GET CONSOLE STATUS

Eingabe: C = OBH
Ausgabe: A = Status der Konsole

Die Funktion GET CONSOLE STATUS überprüft, ob ein Zeichen von der Konsole eingegeben wurde. Eine OOH in Register A entspricht keiner Eingabe, ein FFH zeigt an, daß ein Zeichen eingegben wurde.

Funktion 12: RETURN VERSION NUMBER

Eingabe: C = OCH
Ausgabe:HL = Versionsnummer
A = L
B = H

ZDOS ist kompatibel zu CP/M 2.2 und übergibt deshalb 0022H im Register HL. Aus Kompatibilitätsgründen zu CP/M 1.4 wird der Inhalt des Registers H auch noch in Register B und der von L auch in A übergeben.

Funktion 13: RESET DISK SYSTEM

Eingabe: C = ODH
Ausgabe: -

Die Funktion RESET DISK SYSTEM setzt LEAS auf den Anfangszustand zurück, d.h. A als Betriebslaufwerk, alle anderen Laufwerke inaktiv und DMA-Adresse auf 0080H.

Funktion 14: SELECT DISK

Eingabe: C = 0EH
E = zu selektierendes Laufwerk (0...7)
Ausgabe: -

Die Funktion SELECT DISK macht das angegebene Laufwerk zum Betriebslaufwerk, das Laufwerk ist aktiv, d.h. das Inhaltsverzeichnis wird gelesen und verarbeitet. Die Diskette sollte nicht ausgetauscht werden, ohne NEW oder ^C einzugeben. Eine 00H in Register E heißt Laufwerk A, die 07H Laufwerk H.

Funktion 15: OPEN FILE

Eingabe: C = 0FH
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion OPEN FILE aktiviert eine bestehende Datei für den entsprechenden Benutzer. LEAS durchsucht das Inhaltsverzeichnis der gewählten Diskette nach einem dem FCB entsprechenden Eintrag. In Register DE steht die Adresse des FCB. Im FCB müssen die Bytes 0 bis 12 vom Anwenderprogramm gesetzt sein, die restlichen Bytes werden von LEAS eingetragen, falls OPEN FILE erfolgreich ist. Im Akku wird ein Suchcode zurückgegeben, 0 bis 3 bedeutet erfolgreiches Eröffnen, FFH keine passende Datei vorhanden.
cr muß vom Anwenderprogramm auf 00 gesetzt werden, falls die Datei sequentiell gelesen oder beschrieben werden soll.

Achtung

Es gibt eine zweite OPEN FILE Funktion (OPEN FILE II) mit der LEAS-Nummer 251. Diese Funktion sucht die gewünschte Datei auf Laufwerk A, falls sie auf dem angegebenen Laufwerk nicht gefunden wurde. (vgl. dort)

Funktion 16: CLOSE FILE

Eingabe: C = 10H
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion CLOSE FILE schließt eine Datei, d.h. der FCB wird wieder auf die Diskette zurückgeschrieben. Erst jetzt ist das File dauerhaft gespeichert. CLOSE FILE ist nur nach Beschreiben eines Files erforderlich. CLOSE FILE erfordert ein vorheriges OPEN oder MAKE FILE und WRITE. CLOSE FILE bleibt wirkungslos ohne vorhergehenden WRITE-Aufruf. Suchcode wie bei Funktion 15.
(CLOSE FILE darf unbesorgt benutzt werden, es kann nichts durch ein CLOSE zuviel passieren, ein CLOSE zuwenig ist tragisch, da die Daten nicht wiedergefunden werden können)

Funktion 17: SEARCH FIRST

Eingabe: C = 11H
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion SEARCH FIRST sucht den ersten zum FCB passenden Eintrag im Inhaltsverzeichnis der Diskette. Der Sektor des Inhaltsverzeichnisses, in dem der Eintrag enthalten ist, wird in den DMA-Bereich kopiert. Der Suchcode im Akku gibt die relative Nummer des Eintrags in diesem Sektor an (0...3). Die Startadresse des Eintrags ergibt sich aus:
32 * Suchcode + DMA-Adresse.
Der Suchcode OFFH bedeutet, daß keine passende Datei gefunden wurde.

Funktion 18: SEARCH NEXT

Eingabe: C = 12H
Ausgabe: A = Suchcode

Die Funktion SEARCH NEXT sucht den nächsten passenden Eintrag im Inhaltsverzeichnis. SEARCH NEXT darf erst nach SEARCH FIRST aufgerufen werden. Wie bei SEARCH FIRST wird der Sektor des Inhaltsverzeichnisses, in dem der Eintrag enthalten ist, in den DMA-Bereich kopiert. Der Suchcode im Akku gibt die relative Nummer des Eintrags in diesem Sektor an (0...3). Die Startadresse des Eintrags ergibt sich aus: 32 * Suchcode + DMA-Adresse. Der Suchcode OFFH bedeutet, daß keine passende Datei gefunden wurde.

Funktion 19: DELETE FILE

Eingabe: C = 13H
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion DELETE FILE löscht alle zum FCB passenden Dateien auf der gewählten Diskette. FFH im Akku heißt, daß keine passende Datei gefunden wurde.

Funktion 20: READ SEQUENTIAL

Eingabe: C = 14H
DE = Adresse des FCB
Ausgabe A = Fehlercode

Die Funktion READ SEQUENTIAL kopiert den nächsten Sektor der Datei in den DMA-Bereich, vorausgesetzt, daß sie vorher eröffnet wurde. READ SEQUENTIAL incrementiert das cr-Feld automatisch und eröffnet, falls erforderlich, den Folgeeintrag. 01H wird im Akku zurückgegeben, falls versucht wird, ungeschriebene Daten zu lesen (i.a. bei Dateiende). 00H im Akku steht für eine erfolgreiche Leseoperation.

Funktion 21: WRITE SEQUENTIAL

Eingabe: C = 15H
DE = Adresse des FCB
Ausgabe: A = Fehlercode

Die Funktion WRITE SEQUENTIAL schreibt den nächsten Sektor der Datei aus dem DMA-Bereich auf die Diskette. Um eine Datei dauerhaft zu speichern, muß nach dem letzten Schreibzugriff CLOSE FILE aufgerufen werden. Der Fehlercode 01H bedeutet, daß die Diskette oder ihr Inhaltsverzeichnis voll ist. 00H steht für eine erfolgreiche Schreiboperation.

Funktion 22: MAKE FILE

Eingabe: C = 16H
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion MAKE FILE bereitet eine neue Datei auf der Diskette vor. Sie entspricht OPEN FILE, trägt aber vorher einen neuen Eintrag ins Inhaltsverzeichnis der Diskette ein. Der Suchcode FFH bedeutet, daß das Inhaltsverzeichnis voll ist. Es ist sicherzustellen, daß noch keine Datei mit diesem Eintrag auf der gewählten Diskette existiert. Am einfachsten wird vorher mit dem gleichen FCB die Funktion DELETE FILE aufgerufen. 00...03 im Akku heißt, daß der Aufruf von MAKE FILE erfolgreich durchgeführt wurde. Die Lösung, versuchsweise erstmal OPEN FILE zu versuchen, und nur falls das nicht geht, MAKE FILE zu benutzen, kann nicht empfohlen werden (Probleme bei OPEN FILE II).

Funktion 23: RENAME FILE

Eingabe: C = 17H
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion RENAME FILE benennt Dateien um. Der hierzu erforderliche neue Dateiname steht im zweiten Teil des FCB ab DE + 16. Der dr-Code des zweiten Dateinamens wird ignoriert. RENAME FILE benennt alle passenden Dateien um. Für Fragezeichen im neuen Dateinamen werden im Inhaltsverzeichnis der Diskette die entsprechenden Zeichen des alten Dateinamens eingesetzt.

Funktion 24: RETURN LOGIN VECTOR

Eingabe: C = 18H
Ausgabe: HL = LOGIN-VECTOR
A = L
B = H

Die Funktion RETURN LOGIN VECTOR übergibt in Register HL einen Vektor, dessen Bits den Laufwerken entsprechen. Das niedrigste Bit im Register L entspricht Laufwerk A, das höchste dem Laufwerk H, Register H ist immer 00H. Ein gesetztes Bit (=1) bedeutet, daß das zugehörige Laufwerk aktiv ist.

Funktion 25: RETURN CURRENT DISK:

Eingabe: C = 19H
Ausgabe: A = Betriebslaufwerk

Die Funktion RETURN CURRENT DISK übergibt im Akku die Nummer des Betriebslaufwerks (0 für A bis 7 für H).

Funktion 26: SET DMA ADDRESS

Eingabe: C = 1AH
DE = DMA-Adresse
Ausgabe: -

Die Funktion SET DMA ADDRESS setzt die Adresse für nachfolgende Lese- und Schreiboperationen fest. Ab Adresse DMA erwartet LEAS einen Puffer für 128 Byte. Voreinstellung für die DMA-Adresse ist 0080H.

Funktion 27: GET ADDRESS OF ALLOCATION VECTOR

Eingabe: C = 1BH
Ausgabe:HL = Adresse des ALLOCATION-Vektors

Die Funktion GET ADDRESS OF ALLOCATION VECTOR übergibt in Register HL die Adresse des Belegungsvektors der Betriebsdiskette. Ein gesetztes Bit in diesem Vektor entspricht einem belegten Block auf der Diskette. Das höchste Bit im ALLOCATION-Vektor entspricht dem Block 0. Die Länge des ALLOCATION-Vektors in Bits ist gleich der Kapazität der Diskette in Blocks. (vgl. Funktion 31)
Der ALLOCATION-Vektor ist nur gültig, falls nach dem letzten RESET DISK SYSTEM diese Diskette nicht gewechselt wurde.

Funktion 28: WRITE PROTECT DISK

Eingabe: C = 1CH
Ausgabe: -

Die Funktion WRITE PROTECT DISK versetzt das Betriebslaufwerk in einen Nur-Lese-Zustand, d.h. beim Versuch, auf eine Diskette in diesem Laufwerk zu schreiben, gibt LEAS eine Fehlermeldung aus und schreibt nicht.

Funktion 29: GET READ ONLY VECTOR

Eingabe: C = 1DH
Ausgabe:HL = READ ONLY VECTOR
A = L
B = H

Die Funktion GET READ ONLY VECTOR übergibt im Register HL den READ ONLY VECTOR. Ein gesetztes Bit ist gleichbedeutend mit einer softwaremäßig schreibgeschützten Diskette (vgl. Funktion 24).

Funktion 30: SET FILE ATTRIBUTES

Eingabe: C = 1EH
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion SET FILE ATTRIBUTES setzt bei allen zum FCB passenden Eintragungen im Inhaltsverzeichnis der gewählten Diskette die Attribute, die auch im FCB gesetzt sind. Attribute sind die höchsten Bits von f1 bis t3. Im File Control Block nicht gesetzte Attribute werden auf der Diskette gelöscht.
(vgl. File Control Block)

Funktion 31: GET ADDRESS OF DISK PARAMETERS

Eingabe: C = 1FH
Ausgabe:HL = Adresse von DPB
A = L
B = H

Die Funktion GET ADDRESS OF DISK PARAMETERS übergibt im Register HL die Adresse des Disk-Parameter-Blocks. Dieser enthält Informationen über die verwendeten Laufwerke. Für nähere Informationen siehe Beschreibung von HEAS.

Funktion 32: SET OR GET USER CODE

Eingabe: C = 20H
E = FFH für GET oder USER-Code für SET
Ausgabe: A = USER-Code nur bei GET

Die Funktion SET OR GET USER CODE übergibt den User-Code in Register A, falls in Register E FFH steht. Sonst wird der Inhalt von Register E als neuer USER-Code übernommen.
Der USER-Code ist eine Zahl zwischen 0 und 15. Es sind immer nur Dateien des momentanen Benutzers und in **ZDOS** auch die des Benutzers 0 ansprechbar. Alle übrigen Dateien sind geschützt, sie scheinen nicht zu existieren.
USER 0 Dateien sind für alle erreichbar

Funktion 33: READ RANDOM

Eingabe: C = 21H
DE = Adresse des FCB
Ausgabe: A = Fehlercode

Die Funktion READ RANDOM liest Sektoren wahlfrei von der Diskette. Die Nummer des zu lesenden Sektors steht in r0 bis r2. (r0 ist niedrigstes und r2 ist höchstwertiges Byte). r2 muß in dieser Version von ZDOS immer 0 sein. Um eine Datei wahlfrei zu lesen, muß sie zuerst wie bei READ SEQUENTIAL mit OPEN FILE eröffnet werden. Die zum sequentiellen Zugriff erforderlichen Werte werden bei jedem READ RANDOM gesetzt; es ist daher möglich, eine lückenloses Datei nach einem RANDOM-Zugriff sequentiell weiterzulesen. Hierbei wird als erstes wieder der im RANDOM-Mode gelesene Sektor gelesen. Bei Rückkehr wird im Akku einer der folgenden Fehlercodes übergeben:

00H	kein Fehler
01H	Versuch, unbeschriebene Daten zu lesen
03H	CLOSE für den momentanen Eintrag nicht möglich (nur bei WRITE RANDOM)
04H	nicht existierender Folgeeintrag
05H	Inhaltsverzeichnis voll (nur bei WRITE)
06H	Sektornummer zu groß

Die Fehlercodes 1 und 4 treten auf, wenn ein Programm versucht, nicht existente Daten zu lesen. Der Code 0 besagt nicht, daß der gelesene Sektor gültige Daten enthält; er bedeutet nur, daß in diesem Block ein gültiger Sektor enthalten ist. Sicherzustellen, daß nur vorher beschriebene Sektoren gelesen werden, ist Sache des Anwenderprogrammes. LEAS beschreibt neue Blocks entweder mit 1AH (WRITE RANDOM) oder mit 00H (WRITE RANDOM-II).

Funktion 34: WRITE RANDOM

Eingabe: C = 22H
DE = Adresse des FCB
Ausgabe: A = Fehlercode

Die Funktion WRITE RANDOM entspricht READ RANDOM, nur daß Daten nicht gelesen, sondern geschrieben werden. Nach Beendigung des Schreibens muß die Datei durch CLOSE geschlossen werden.
WRITE RANDOM schreibt möglicherweise lückenhafte Dateien. Da im Inhaltsverzeichnis ein Block (8, 16.. Sektoren) die kleinste Einheit ist, existieren Blocks, in denen nicht alle Sektoren vom Anwenderprogramm beschrieben worden sind. Um hier definierte Werte zu erhalten, füllt ZDOS wie CP/M diese Sektoren mit 1AH. (oder mit OOH bei WRITE RANDOM-II).

Funktion 35: COMPUTE FILE SIZE

Eingabe: C = 23H
DE = Adresse des FCB
Ausgabe: -

Die Funktion COMPUTE FILE SIZE berechnet die Länge einer Datei. Die hier berechnete Länge entspricht bei lückenlosen (sequentiellen) Dateien ihrer physikalischen Länge. Bei RANDOM-Dateien wird als Länge die Nummer des letzten Sektors plus 1 genommen. Die Länge steht in r0 bis r2. Diese Funktion ermöglicht es, ab dem Ende einer Datei weiterzuschreiben (in RANDOM-Mode).

Funktion 36: SET RANDOM RECORD

Eingabe: C = 24H
DE = Adresse des FCB
Ausgabe: -

Die Funktion SET RANDOM RECORD ermöglicht den Übergang von sequentiellem auf wahlfreien Zugriff. Die Sektornummer für RANDOM zeigt auf den nächsten Sektor.

Funktion 37: LOGOUT DISKS

Eingabe: C = 25H
DE = Laufwerksvektor
Ausgabe: -

Die Funktion LOGOUT DISKS ermöglicht es, gezielt Disketten wieder auszutragen. Hierzu wird in Registerpaar DE ein Vektor, in dem Bits für die einzelnen Laufwerke sind, übergeben. Eine 1 bedeutet austragen, eine 0 keine Änderung.
Für Laufwerk A steht das niedrigste Bit (01H), für Laufwerk H das höchste (80H). Anwendung: Gezielte Erlaubnis, einzelnen Disketten zu wechseln; nach dem Wechsel ist die getauschte Diskette erst auszutragen. Das dann folgende SELECT setzt den READ/ONLY-Status zurück.

Funktion 38 MP/M

entfällt bei ZDOS wie CP/M

Funktion 39 MP/M

entfällt bei ZDOS wie CP/M

Funktion 40: WRITE RANDOM II

Eingabe: C = 28H
DE = Adresse des FCB
Ausgabe: A = Fehlercode

Die Funktion WRITE RANDOM II entspricht READ RANDOM, nur daß Daten nicht gelesen, sondern geschrieben werden. Nach Beendigung des Schreibens muß die Datei durch CLOSE geschlossen werden.
WRITE RANDOM II schreibt wie WRITE RANDOM möglicherweise lückenhafte Dateien. Da im Inhaltsverzeichnis ein Block (8, 16.. Sektoren) die kleinste Einheit ist, existieren Blocks, in denen nicht alle Sektoren vom Anwenderprogramm beschrieben worden sind. Um hier definierte Werte zu erhalten, füllt ZDOS wie CP/M diese Sektoren mit 00H. (oder mit 1AH bei WRITE RANDOM).

Die folgenden Aufrufe existieren nur in ZDOS. Sie sind nicht in CP/M enthalten.

Funktion 250: DISABLE CONSOLE CHECK

Eingabe: C = FAH
Ausgabe: -

Wie bereits erwähnt, prüft LEAS bei jedem Aufruf, ob der Benutzer am Terminal das laufende Programm unterbrechen will. (dies geschieht durch Eingabe von ^S, ^C.) Dazu muß ZDOS ständig Zeichen einlesen, (falls vorhanden). Diese Zeichen fehlen aber dann dem Anwendungsprogramm, das nicht über ZDOS geht, um Zeichen zu lesen. (Bsp: WORDSTAR). Um dieses Problem zu umgehen, gibt es die Funktion DISABLE CONSOLE CHECK; sie schaltet das Überprüfen der Konsole durch ZDOS aus; eingeschaltet wird automatisch nach jedem Warmstart.

Funktion 251: OPEN FILE II

Eingabe: C = FBH
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion OPEN FILE II aktiviert eine bestehende Datei für den entsprechenden Benutzer. LEAS durchsucht das Inhaltsverzeichnis der gewählten Diskette nach einem dem FCB entsprechenden Eintrag. In Register DE steht die Adresse des FCB. Im FCB müssen die Bytes 0 bis 12 vom Anwenderprogramm gesetzt sein, die restlichen Bytes werden von LEAS eingetragen, falls OPEN FILE erfolgreich ist. Im Akku wird ein Suchcode zurückgegeben, 0 bis 3 bedeutet erfolgreiches Eröffnen, FFH keine passende Datei vorhanden.
cr muß vom Anwenderprogramm auf 00 gesetzt werden, falls die Datei sequentiell gelesen oder beschrieben werden soll. OPEN FILE II sucht auf Laufwerk A weiter, falls die Datei auf dem angegebenen Laufwerk nicht gefunden wurde. Falls sie auf A gefunden wird, wird Laufwerk A im FCB eingetragen (01 als drive-code).

Achtung

Es gibt eine zweite OPEN FILE Funktion (OPEN FILE) mit der LEAS-Nummer 15. Diese Funktion sucht nicht; falls die Datei auf dem angegebenen Laufwerk nicht gefunden wurde, wird sofort ein Fehler (nicht gefunden) gemeldet. Die Funktion OPEN FILE ist CP/M-kompatibel, OPEN FILE II existiert in CP/M nicht.

Funktion 252: RETURN NUMBER OF DISKS

Eingabe: C = FCH
Ausgabe: A = Zahl der Laufwerke

Die Funktion RETURN NUMBER OF DISKS übergibt bei Rückkehr im Akku die Anzahl der implementierten Laufwerke (1 bis 8). Dies schafft eine Möglichkeit, die Fehlermeldung SELECT ERROR zu umgehen.

Funktion 253: GET SPOOL STATUS

Eingabe: C = FDH
Ausgabe: A = Status

Die Funktion GET SPOOL STATUS dient dazu, den Zustand des Spoolers festzustellen. Sie übergibt im Akku eine 00H, wenn momentan keine Datei im Hintergrund gedruckt wird. Jeder andere Wert im Akku zeigt an, daß die Ausgabe einer Datei an den Drucker noch nicht beendet ist.

Funktion 254: STOP SPOOLING

Eingabe: C = FEH
Ausgabe: -

Die Funktion STOP SPOOLING bricht das Drucken im Hintergrund mit sofortiger Wirkung ab, dies ist die einzige Möglichkeit, die Funktion SPOOL FILE abzubrechen.

Funktion 255: SPOOL FILE

Eingabe: C = FFH
DE = Adresse des FCB
Ausgabe: A = Suchcode

Die Funktion SPOOL FILE veranlaßt LEAS, die im FCB angegebene Datei im Hintergrund auszudrucken. Sie blockiert den Drucker für alle anderen Zugriffe. Der Suchcode FFH gibt an, daß die angegebene Datei entweder nicht existiert oder daß noch eine andere Datei gedruckt wird. Es wird empfohlen, vor Aufruf von SPOOL FILE mittels GET SPOOL STATUS festzustellen, ob der Spooler frei ist.

File Control Block

File Control Blocks werden im Inhaltsverzeichnis der jeweiligen Diskette abgespeichert und verweisen auf Dateien. Dateien bestehen aus Records (hier zu 128 Byte). Mehrere Records werden zu einem Block zusammengefaßt. Die kleinste belegbare Einheit auf einer Diskette ist ein Block. Dateien sind durch ihren File Control Block erreichbar.

Aufbau eines File Control Blocks:

```
      dr, f1,...,f8, t1, t2, t3, ex, ad, sy, rc, d0,...,d15, cr, r0, r1, r2
FCB+   0   1      8   9  10  11  12  13  14  15  16       31  32  33  34  35
```

Dabei bedeutet:

dr Drive Code (0...8) laufwerksnummer bzw. 0 für Betriebsdiskette

0 = Datei auf der Betriebsdiskette
1 = Datei auf Laufwerk A
:
8 = Datei auf Laufwerk H

f1...f8 Dateiname in ASCII-Format, höchstes Bit = 0, keine Kleinbuchstaben

t1...t3 Dateityp in ASCII-Format, keine Kleinbuchstaben

Die höchsten Bits von f1...f8, t1...t3 sind Dateiattribute. t1' sei das höchste Bit von t1. Benutzt werden momentan:

t1' = 1 heißt Datei nur lesbar
t2' = 1 heißt Systemdatei, erscheint nicht im Inhaltsverzeichnis (DIR)

ex Extentnummer, ist vor **OPEN** bzw. **MAKE** zu setzen, wird anschließend von **LEAS** weitergezählt.

ad höheres Byte der Ladeadresse, das niedrige Byte wird als 00H angenommen; ad = 0 heißt nicht Startadresse 0000H, sondern 0100H, um CP/M-kompatibel zu bleiben. (Erweiterung gegenüber CP/M). ad ist vor **MAKE** zu setzen!

sy reserviert für ZDOS, 80H bei OPEN, CLOSE nur falls = 0

rc Recordzähler (0 bis 80H)

d0...d15 Blocknummern, verweisen auf Blocks auf der Diskette, 00 bzw. 0000 bedeutet nicht belegt.
Falls auf der Diskette weniger als 256 Blocks gespeichert werden können, sind d0 bis d15 16 Blocknummern; bei mehr als 255 Blocks pro Diskette sind jeweils 2 Byte zusammengefaßt als eine Blocknummer zu betrachten. (low Byte, high Byte)

cr laufender Record, ist vor Lesen bzw. Schreiben vom Anwenderprogramm zu setzen (am Anfang auf 0), wird bei sequentiellem Lesen bzw. Schreiben automatisch weitergezählt.

r0...r2 RANDOMnummer (nur bei wahlfreiem Lesen bzw. Schreiben benötigt). r0 ist das niedrigste, r2 das höchste Byte.

Bei Zugriff auf Dateien sind dr, f1...f8, t1...t3 und ex zu setzen. Bei Aufruf von OPEN füllt das LEAS die restlichen Werte auf. Der FCB darf während der Benutzung der zugehörigen Datei selbstverständlich vom Anwenderprogramm nicht verändert werden.

Nummer		Funktion	Eingabe	Ausgabe
00H	0	SYSTEM RESET	-	-
01H	1	CONSOLE INPUT	-	A = Zeichen
02H	2	CONSOLE OUTPUT	E = Zeichen	-
03H	3	READER INPUT	-	A = Zeichen
04H	4	PUNCH OUTPUT	E = Zeichen	-
05H	5	LIST OUTPUT	E = Zeichen	-
06H	6	DIRECT CONSOLE INPUT/OUTPUT	in: E = OFFH oder Zeichen	in: A = Zeichen oder 0
07H	7	GET I/O-BYTE	-	A = IOBYTE
08H	8	SET I/O-BYTE	E = IOBYTE	-
09H	9	PRINT STRING	DE = .Text	-
0AH	10	READ CONSOLE BUFFER	DE = .Puffer	-
0BH	11	GET CONSOLE STATUS	-	A = 0 oder OFFH
0CH	12	RETURN VERSION NO	-	HL = Version
0DH	13	RESET DISK SYSTEM	-	-
0EH	14	SELECT DISK	E = Diskno.	-
0FH	15	OPEN FILE	DE = .FCB	A = Suchcode
10H	16	CLOSE FILE	DE = .FCB	A = Suchcode
11H	17	SEARCH FIRST	DE = .FCB	A = Suchcode
12H	18	SEARCH NEXT	-	A = Suchcode
13H	19	DELETE FILE	DE = .FCB	A = Suchcode
14H	20	READ SEQUENTIAL	DE = .FCB	A = Fehlercode
15H	21	WRITE SEQUENTIAL	DE = .FCB	A = Fehlercode
16H	22	MAKE FILE	DE = .FCB	A = Suchcode
17H	23	RENAME FILE	DE = .FCB	A = Suchcode
18H	24	RETURN LOGIN VECTOR	-	HL = Login-vector
19H	25	RETURN CURRENT DISK	-	A = Disknummer
1AH	26	SET DMA ADDRESS	DE = .DMA	-
1BH	27	GET ADDR(ALLOC)	-	HL = .Alloc
1CH	28	WRITE PROTECT DISK	-	-
1DH	29	GET R/O-VECTOR	-	HL = R/O-Vector
1EH	30	SET ATTRIBUTES	DE = .FCB	A = Suchcode
1FH	31	GET ADDR(DSK PARMS)	-	HL = .DPB
20H	32	SET/GET USER CODE	get: E = FFH set: E = user	get: A = User set: -
21H	33	READ RANDOM	DE = .FCB	A = Fehlercode
22H	34	WRITE RANDOM	DE = .FCB	A = Fehlercode
23H	35	COMPUTE FILE SIZE	DE = .FCB	-
24H	36	SET RANDOM RECORD	DE = .FCB	-
25H	37	LOGOUT DISKS	DE = Vector	-
28H	40	WRITE RANDOM II	DE = .FCB	A = Fehlercode
FAH	250	DISABLE CONSOLE CHECK	-	-
FBH	251	OPEN FILE II	DE = .FCB	A = Suchcode
FCH	252	RETURN NDISKS	-	A = Zahl der Disks
FDH	253	GET SPOOL STATUS	-	A = Status
FEH	254	STOP SPOOLING	-	-
FFH	255	SPOOL FILE	DE = .FCB	A = Suchcode

hierbei bedeutet: .xxxx Adresse von xxxx