Janich & Klass
Computersysteme

```
****** Betriebssystem für Z80 - Computer *******
*                                              *
*     *******    *****        ***        ****  *
*           *    *    *      *   *      *    * *
M         *      *     *    *     *     *      M
K       *        *     *    *     *      ****  K
C     *          *     *    *     *          * C
*   *            *    *      *   *      *    * *
*     *******    *****        ***        ****  *
*                                              *
****************** Version 5.b ******************
```



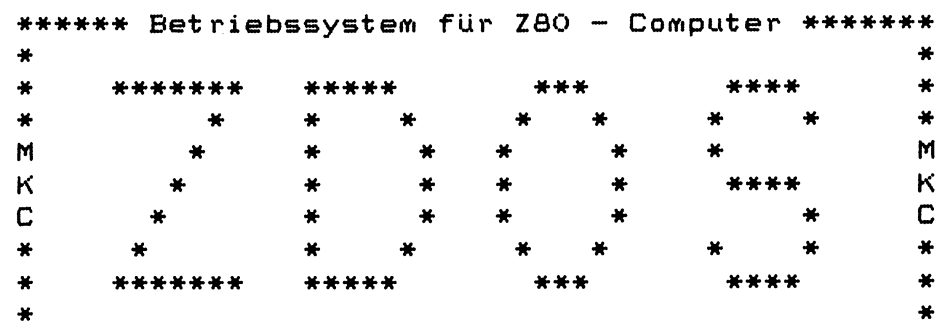
Neumarktstr. 46, D-5600 Wuppertal 1, Tel. 0202/442480

ZDOS Version 5b1 ist im wesentlichen kompatibel zu CP/M 2.2. Es laufen alle Programme, die unter CP/M laufen, auch unter ZDOS. Einige Einschränkung sind die Programme XSUB und SUBMIT. Diese sind im ZDOS-Betriebssystem durch das Programm DO ersetzt. Von diese Einschränkung sind allerdings auch die Programme betroffen, die die CP/M-Version der Prozeduren benutzen. (Beispiel: DBASE Quit to program).

Ferner erlaubt ZDOS nur 1K oder 2K Blockgrößen. Diese Einschränkung tritt nur bei gravierenden Änderungen im HEAS auf. HEAS ist nicht vollständig kompatibel zu BIOS, d.h. HEAS kann erst nach leichten Änderungen als BIOS für CP/M-Systeme benutzt werden. Ein BIOS muß erweitert werden, um die HEAS-Funktionen zu erfüllen.

CP/M-Programme laufen unter ZDOS. Dies heißt aber nicht, daß ZDOS-Programme unter CP/M laufen müssen. Dies ist nur möglich, wenn beim Erstellen eigener Programme die CP/M-Restriktionen eingehalten wurden.

Da bei ZDOS die Ladeadresse des Files im File-Controll-Block steht, dieses Byte aber bei CP/M immer 0 ist, kann es bei manchen Kopierprogrammen Änderungen in der Ladeadresse geben. PIP arbeitet auch unter ZDOS richtig; I(nterchange) setzt die Ladeadresse konstant auf 0.






## Inhaltsverzeichnis

Ein Computersystem besteht aus folgender Hardware (physikalisch vorhandenen Baugruppen):

1. der CPU, das ist die zentrale Rechen- und Verarbeitungseinheit
2. dem Arbeitsspeicher, in dem die Programme ausgeführt werden. Als Speicher werden heute meistens dynamische RAMs, d.h. Halbleiterspeicher verwendet. RAMs sind flüchtige Speicherbausteine, sie verlieren ihren Inhalt beim Ausschalten des Computers. Daher wird ein Massenspeicher (oder Hintergrundspeicher) zur dauerhaften Speicherung benötigt.
3. dem Hintergrundspeicher, meistens Disketten- oder Winchesterlaufwerken, auf denen alle Programme und Daten abgespeichert sind. Gespeichert wird auf Magnetplatten, diese sind dauerhafte wiederverwendbare Speichermedien. Disketten sind flexible Magnetplatten, die in Diskettenlaufwerken gelesen und beschrieben werden können (häufigster Hintergrundspeicher bei kleinen Computersystemen.)
4. einem Terminal oder Datensichtgerät zur Kommunikation zwischen Computer und Benutzer. Hier gibt der Benutzer seine Befehle an den Computer und hier erhält er auch die Antworten des Computers.
5. möglicherweise einem Drucker, um Ausgaben des Computers in schriftlicher Form zu erhalten.

Zum Betrieb benötigt ein Computersystem außer der Hardware noch Software. Software sind die Programme und Daten, die dem Computer angeben, was er zu tun hat. Sie werden in der Hardware des Computers (dem Speicher) gespeichert. Jedes Computersystem benötigt mindestens ein Betriebssystem. Das Betriebssystem übernimmt die Verwaltung der Computerkomponenten, es organisiert die Speicherung von Dateien (Programmen und Daten, z.B. Texte) auf der Diskette und es ermöglicht das Aufrufen von Programmen. Das Betriebssystem erlaubt also dem Benutzer den Umgang mit dem Computer. Es gibt unterschiedliche Betriebssysteme für Computer. Minimalforderung an ein Betriebssytem sind folgende Punkte:

- Verwaltung des Haupt- und Massenspeichers
- Verwaltung der Peripheriegeräte
- Kommunikation mit dem Benutzer
- Bereitstellen von Dienstprogrammen

Ferner sind die folgenden Forderungen wünschenswert:

- Portabilität (d.h. nicht nur für genau eine Hardware-Konfiguration geeignet)
- Benutzerführung (d.h. Hinweistexte auf der Diskette; der Idealfall wäre erreicht, falls kein Handbuch mehr erforderlich ist)
- breites Angebot von Programmiersprachen und Anwendungsprogrammen

Das hier beschriebene Betriebssystem **ZDOS** erfüllt im Wesentlichen alle oben genannten Anforderungen. Es ist kompatibel zu CP/M Version 2.2 von Digital Research, d.h. alle für CP/M angebotenen Programme laufen auch unter **ZDOS**. Somit stehen für **ZDOS** Compiler für fast alle Programmiersprachen (z.B. FORTRAN, BASIC, PASCAL, COBOL usw.) und mehr als hundert Anwenderprogramme zur Verfügung.
Zusätzlich zu den CP/M-Funktionen ermöglicht **ZDOS** es, den Drucker im Hintergrund zu bedienen, d.h. während des Druckens kann normal am System weitergearbeitet werden.
Ferner bietet **ZDOS** ab Version 5.b die Möglichkeit, Dateien, die auf dem Betriebslaufwerk nicht gefunden werden, von Laufwerk A zu holen. Zur Ausnutzung dieser Erweiterung sollte man B als Betriebslaufwerk wählen, die Systemprogramme werden dann, falls nicht auf B, von Laufwerk A geholt. Dies vereinfacht die Bedienung erheblich.
In **ZDOS** wird bei jedem Aufruf an **LEAS** kontrolliert, ob der Benutzer durch Eingabe von ^S das System anhalten will; dies ermöglicht ein Abbrechen eines beliebigen Programmes. Da diese Erweiterung bei Programmen, die die **HEAS**-Sprungleiste benutzen, eventuell zu Störungen führt, ist sie durch Aufruf der LEAS-Funktion 250 ausschaltbar. Sie wird automatisch nach jedem Warmstart wieder eingeschaltet.
**ZDOS**-Aufrufe ändern nur die zur Ausgabe benötigten Register (im Gegensatz zu CP/M, das fast alle Register zerstört).
ZDOS benötigt einen Z80-Computer, bestehend aus mindestens 32K RAM, einem Floppy-Laufwerk als Massenspeicher und einem Terminal. Es unterstützt maximal 64K Byte RAM, bis zu 8 Floppylaufwerke, ein logisches Terminal, einen logischen Drucker und je einen logischen Streifenleser und -stanzer oder ähnliche Peripheriegeräte. Jedes dieser logischen Geräte kann einem von vier physikalischen Geräten zugewiesen werden. Weiterhin unterstützt das zu **ZDOS** gehörende **HEAS** als Option auch RAM-Disketten, d.h. Halbleiterspeicher, der von **ZDOS** wie eine Diskette verwaltet wird.
Durch die Sektorgröße von 512 Byte und die für 4 MHz Systemtakt optimierte Verschränkung ist **HEAS** bei Diskettenzugriffen sehr schnell.
**ZDOS** wartet auf Peripheriegeräte, d.h. ein angesprochenes, aber nicht eingeschaltetes oder nicht angeschlossenes Gerät hält den Computer an (Fehlermeldung: Gerätename, Fragezeichen; z.B. **DISK?** oder **LPT?**). Wenn also der Computer nichts tut, sollte zuerst überprüft werden, ob alle Geräte eingeschaltet und angeschlossen sind und ob die richtigen Disketten eingelegt worden sind. Erst danach sollte mit ^C oder ^S und ^C versucht werden, das System neu zu starten. Nur als letzte Möglichkeit dient der RESET-Schalter. Hierbei ist zu beachten, daß es Programme gibt, die sehr lange Ausführungszeiten benötigen!!

**ZDOS** besteht aus den in sich abgeschlossenen Modulen:

- dem **Konsolinterpreter** (KI), der zur Kommunikation zwischen Benutzer und Computer dient. Der **KI** nimmt Befehlszeilen entgegen und interpretiert sie durch **LEAS**-Aufrufe.
- dem **logischen Ein/Ausgabe-System** (LEAS). Dieses enthält die zur Speicherverwaltung (RAM und Diskette) benötigten Routinen und zusätzlich die Ein-/Ausgabe-Routinen. **LEAS** ist im Prinzip eine Sammlung von Unterprogrammen, die alle auch von Anwenderprogrammen aufgerufen werden können. Die Diskettenverwaltung läßt die auf der Diskette vorhandenen Sektoren und Spuren (physikalisch) für den Anwender unsichtbar werden. Unter **LEAS** besteht eine Diskette aus Inhaltsverzeichnis, Daten und Programmen (Dateien). Eine Datei besteht unter **ZDOS** aus mehreren zusammengehörigen Sektoren, die unter einem Namen ansprechbar sind. So ist es für den Anwender nicht mehr erforderlich, zu wissen, wo auf der Diskette Programme und Daten abgespeichert sind. Sie sind mit dem im Inhaltsverzeichnis abgelegten Namen jederzeit wieder aufrufbar.
- dem **hardwareabhängigen Ein/Ausgabe-System** (HEAS). **HEAS** besteht aus den Treiberroutinen für die Peripheriegeräte. Durch Änderung von nur **HEAS** kann **ZDOS** an unterschiedliche Hardware angepasst werden.
- dem **Programmbereich** (PB). Hier werden Programme des Anwenders und Hilfsprogramme des Betriebssystems ausgeführt. Der Programmbereich beginnt bei Adresse 100H und endet am Beginn von **LEAS**. Der Konsolinterpreter darf von Programmen überschrieben werden. Er wird beim Programmende durch einen LEAS-Aufruf (Warmstart) neu geladen.

Unter **ZDOS** dürfen Disketten nicht beliebig gewechselt werden. Disketten dürfen keinesfalls während eines Zugriffs des Computers auf das entsprechende Laufwerk entnommen werden. Zu diesem Zeitpunkt darf auch der Computer keinesfalls unterbrochen werden.
Es wird davor gewarnt, billige und schlechte Disketten zu verwenden. Eine Diskette, die sich nicht fehlerfrei formatieren läßt, gehört in den Papierkorb, aber keinesfalls in den Computer. Eine Diskette, die permanente Schreib- oder Lesefehler verursacht, kann meist nach neuer Formatierung weiter verwendet werden. Die auf ihr stehenden Daten gehen beim Formatieren verloren. (**Kopien von allen wichtigen Disketten**).
Eine Diskette, bei der ein **BAD-SECTOR**-Fehler gemeldet wird, der nicht durch mehrfaches Versuchen übergangen werden kann, wird an genau dieser Stelle immer wieder denselben Fehler melden. In diesem Fall sollte sie soweit möglich kopiert werden, und ab dann mit der Kopie weitergearbeitet werden. Ein Weiterarbeiten mit der Originaldiskette ist riskant, da eventuell Daten nicht mehr abgespeichert werden können. **ZDOS** übergeht fehlerhafte Sektoren nicht.

Als Beispiel für den Umgang mit dem Computer (unter ZDOS) soll jetzt die Systemdiskette dupliziert werden. (Dies ist unbedingt notwendig, da auf sie keinesfalls geschrieben werden darf). Folgende Schritte sind abzuarbeiten:

1. Starten des Computers

   Einschalten aller Teile des Computersystems

   Einlegen der Systemdiskette in Laufwerk A, das ist das obere Laufwerk, so daß der Aufkleber zur Türe des Laufwerks zeigt. Der längliche Schlitz in der Diskettenhülle ist dabei hinten.
   Bei einer anderen Lage der Diskette kann der Computer (die Köpfe des Diskettenlaufwerks) beschädigt werden, sie ist also in jedem Fall zu vermeiden.

2. Bis jetzt ist weiter noch nichts geschehen. Der Urlader, ein Programm, das in einem PROM (dauerhafter Nur-Lese-Speicher) ist, wartet auf die Eingabe von Leerzeichen an dem Datensichtgerät. Erst wenn die Baudrate (Übertragungsgeschwindigkeit) des Datensichtgerätes bestimmt ist, erscheinen Meldungen des Urladers auf dem Bildschirm. Nach mehreren Diskettenzugriffen meldet sich dann anschließend das Betriebssystem (der Konsolinterpreter KI), das vom Urlader (im PROM) von der Diskette in den Arbeitsspeicher geladen worden ist, mit

   0A>

   Das heißt: Benutzer 0, Betriebslaufwerk A

   Sollte diese Meldung nicht erscheinen, liegt ein Fehler oder eine Fehlbedienung vor. Hierzu sind weitere Hilfen im "Urlader"-Kapitel angegeben.
   Ab jetzt befinden wir uns im Konsolinterpreter KI.

3. Wenn die Meldung 0A> erschienen ist, erwartet das Betriebssystem (der Konsolinterpreter) weitere Befehle vom Benutzer. Wir wollen jetzt eine Diskette duplizieren. Dazu ist eine neue Diskette nötig. Diese kommt in Laufwerk B, das untere Laufwerk.

   > Jede neue Diskette muß vor der Benutzung formatiert werden, d.h. mit speziellen Daten vorbeschrieben werden.

4. Wir formatieren also jetzt die neue Diskette. Dazu gibt es ein Hilfsprogramm FORMAT. Dieses wird aufgerufen mit der Eingabe:

   FORMAT<cr>

Nach jeder Eingabe ist das Wagenrücklaufzeichen (Carriage Return oder New-Line) notwendig (hier mit <cr> angedeutet), damit der Computer arbeitet. Falsch getippte Zeichen können durch Eingabe von CONTROL H, d.h. gleichzeitiges Betätigen der CONTROL und der H-Taste korrigiert werden. Hierdurch wird das letzte eingegebene Zeichen gelöscht. Ferner kann die gesamte bereits eingegebene Zeile durch Eingabe von CONTROL X oder RUBOUT gelöscht werden. Eingaben werden erst nach Carriage-Return beachtet.

Ab jetzt befinden wir uns im dem soeben aufgerufenen Program "FORMAT", also nicht mehr im Konsolinterpreter; d.h. ab jetzt gelten die KI-Befehle nicht mehr. Jetzt sind nur die Befehle für "FORMAT" zulässig (siehe bei FORMAT).

Wir wollten eine Diskette formatieren. Dazu muss die neue Diskette in Laufwerk B, das ist das untere Laufwerk, eingelegt werden.
Das Programm "FORMAT" meldet sich mit:

```
FORMAT V. 3.2 (26.10.82)
(B)  5,25", MFM, DS, 40 TRK
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) =
```

oder einer ähnlichen Meldung. Falls die Meldung NOT INSTALLED, PLEASE INSTALL erscheint, ist zuerst das für den jeweiligen Computer erforderliche Format durch Eingabe von A bis I zu wählen. Danach geht es für beide Fälle gleich weiter:
Es ist B einzugeben, um die Diskette in Laufwerk B zu formatieren. FORMAT schreibt die Nummer der momentan bearbeiteten Spur auf den Bildschirm:
FORMATTING TRACK 00 (als Beispiel für Spur 0)
Bei zweiseitigen Disketten wird zweimal gezählt, einmal für die Vorder- und nochmals für die Rückseite. Erst danach ist die Diskette fertig formatiert; jetzt kann die Aufforderung:
HIT CARRIAGE RETURN TO RETURN TO ZDOS
durch Eingabe von Carriage-Return befolgt werden.

5. Jetzt sind wir wieder im Konsolinterpreter (KI). Als nächstes wird das Programm SYSGEN zum direkten Kopieren von Disketten aufgerufen:

```
SYSGEN
```

Es meldet sich mit:

```
SYSTEM GENERATION AND COPY PROGRAMM
(TYPE X FOR EXIT)
(B)  5,25", MFM, DS, 40 TRK
SELECT: SYSGEN (S), COPY (C), COPY TRACK (T) OR INSTALL (I)
```

oder einer ähnlichen Meldung. Falls die Meldung NOT INSTALLED, PLEASE INSTALL erscheint, ist zuerst das für den jeweiligen Computer erforderliche Format durch Eingabe von A bis I zu wählen (wie bei FORMAT). Danach geht es für beide Fälle gleich weiter:
Es soll die Diskette in A auf die in B kopiert werden. Damit ist COPY durch Eingabe von C auszuwählen; die anschließende Frage:

SOURCE DISK FOR COPY =

ist mit A und die nächste:

DESTINATION DISK =

ist mit B zu beantworten. Jetzt wird kopiert; dabei wird die momentane Spur wie bei FORMAT angezeigt. Falls ein READ ERROR gemeldet wird, ist die gesamte Kopie nicht gelungen; es ist mit FORMAT neu zu beginnen.

Im Normalfall erscheint die Anforderung:

HIT CARRIAGE RETURN TO RETURN TO ZDOS

Nach Eingabe von Carriage-Return meldet sich wieder der Konsolinterpreter mit 0A). Damit ist die Kopie der Systemdiskette fertig.
Die soeben erstellte Kopie kann durch Aufruf von SYSGEN mit dem Original verglichen werden (VERIFY). Wenn kein VERIFY-ERROR gemeldet wird, stimmen die beiden Disketten überein.
Trotzdem sollte die soeben neu erstellte Diskette (die Kopie) jetzt vorsichtshalber in Laufwerk A ausprobiert werden.

Benutzt wurden in obigem Beispiel die Programme: FORMAT, SYSGEN und das Betriebssystem ZDOS, bestehend aus KI, LEAS und HEAS.

## Eingaben von der Konsole

Die Eingabe an das System erfolgt zeilenweise. Zeilen werden durch Carriage-return oder New-line abgeschlossen, d.h. Kommandos werden erst dann ausgeführt. Die Line-Feed-Taste dient zur logischen Trennung von Eingaben, die eigentlich in unterschiedlichen Zeilen stehen müßten. Innerhalb einer Zeile sind folgende Korrekturmöglichkeiten gegeben:

^H oder **Backspace** löscht das zuletzt eingegebene Zeichen

^X oder **Rubout** bzw. **Delete** löscht die gesamte Zeile

^I oder **Tab** springt auf die nächste Tabulatorposition (Tabulatoren sind alle 8 Zeichen gesetzt)

Folgende Zeichen haben Sonderbedeutungen:

^C als erstes Zeichen veranlaßt einen Warmstart.

^P protokolliert alle Ein- und Ausgaben auf dem Drucker, das zweite ^P schaltet den Drucker wieder aus.

**Line-Feed** ermöglicht die Eingabe von mehreren Kommandos in einer Zeile. Es wird durch das '**&**'-Zeichen dargestellt.

^S stoppt jedes Programm beim nächsten **LEAS**-Aufruf. Jetzt gibt es 2 Möglichkeiten:

^Q setzt die Programmausführung fort,

^C bricht das Programm ab und übergibt die Kontrolle wieder an das Betriebssystem.

Alle anderen Eingaben nach ^S werden ignoriert.

Im **KI** können Listings auf die Konsole nur durch Eingabe von Carriage-Return, Escape oder Blank abgebrochen werden. Bei **TYPE** mit der P-Angabe wird das Listing alle 20 Zeilen gestoppt; es kann durch Eingabe von Carriage-Return, Line-Feed oder Escape fortgesetzt werden; das zur Fortsetzung verwendete Zeichen kann nicht zum Abbrechen des Listings verwendet werden (dies verhindert ein ungewolltes Abbrechen des Listings bei Terminals, die Autorepeat haben).

^char heißt CONTROL char und bedeutet, daß die CONTROL-Taste und der entspechende Buchstabe (char) gleichzeitig betätigt werden.

Fehler können von allen Teilen des Betriebssystems gemeldet werden.

### Fehlermeldungen des Kommandointerpreters

| | |
|---|---|
| **NO FILE** | Die im Kommando angegebene Datei ist nicht vorhanden. |
| **BAD LOAD** | Die angegebene Datei kann nicht geladen werden; sie würde **KI** überschreiben. |
| **NO SPACE** | Für den zu rettenden Speicherbereich reicht der auf der Diskette vorhandene freie Platz nicht aus. |
| **FILE EXISTS** | Eine Datei mit dem neuen Namen existiert bereits. (bei **RENAME**) |
| **READ ERROR** | Die angegebene Datei kann nicht vollständig gelesen werden. (Meist wurde versucht, Random-Dateien zu lesen.) |
| xx? | Dieses Kommando ist unverständlich. Der nicht interpretierbare Teil steht vor dem Fragezeichen. |
| **ALL (Y/N)?** | Keine wirkliche Fehlermeldung; es wird nur vorsichtshalber nochmal gefragt, ehe alle Dateien gelöscht werden. **Y** löscht, **N** läßt bestehen. |

### LEAS Fehlermeldungen

Alle **LEAS**-Fehlermeldungen haben folgende einheitliche Form:

**ZDOS ERR ON** laufwerk: erklärung

Hierbei ist laufwerk der Name des betroffenen Laufwerks (A...H) und erklärung einer der folgenden Texte:

| | |
|---|---|
| **SELECT** | Das ausgewählte Laufwerk existiert nicht. Bei Betätigung einer beliebigen Taste führt das System einen Neustart durch und selektiert Laufwerk A. |
| **BAD SECTOR** | Fehler bei Lese- oder Schreibzugriffen auf die Diskette. **CARRIAGE RETURN** veranlasst einen weiteren Versuch, ^C bricht ab. |
| **R/O** | Das angegebene Laufwerk befindet sich im Nur-Lese-Zustand. Wahrscheinlich wurde die Diskette vorher ohne Meldung an **ZDOS** gewechselt. Die Betätigung einer beliebigen Taste führt zu einem Neustart. |

| | |
|---|---|
| FILE R/O | Die ausgewählte Datei ist eine Nur-Lese-Datei, d.h sie kann nicht überschrieben oder umbenannt werden. **CARRIAGE RETURN** ermöglicht das Überschreiben nur für USER 0, ^C bricht ab. |
| SPOOLING | Diese Fehlermeldung tritt auf, wenn während des Druckens im Hintergrund (**LIST**) versucht wird, den Drucker anzusprechen. In diesem Fall hat **LIST** immer Vorrang. **CARRIAGE RETURN** setzt das Programm, das zu drucken versucht hat, fort, übergeht aber alle Druckeraufrufe dieses Programms. ^C bricht das Programm ab. |
| ERASE SPOOLING | Es wurde versucht, eine Datei, die momentan gedruckt wird, zu löschen. Dies ist erst nach Eingabe von **STOP** oder Aufruf von **LEAS**-Funktion 254 möglich. |

## HEAS Fehlermeldungen

| | |
|---|---|
| DISK? | Die Diskette ist nicht richtig im Laufwerk eingelegt. Diskette richtig einlegen und dann Carriage-Return betätigen. Die Operation wird dann fortgesetzt. ^C bricht ab! |
| LPT? | Drucker nicht bereit, einschalten bzw. anschließen oder selektieren. Eingabe wie bei DISK? |
| CONA? | V24-Schnittstelle A hat Handshake on und das angeschlossene Gerät wird nicht ready. |
| CONB? | Wie bei CONA?, aber Schnittstelle B |
| *** WRITE PROTECTED *** | Die Diskette ist schreibgeschützt. |

Der letzte Fehler ist nur eine Informationen, die Kontrolle wird an **LEAS** übergeben, **LEAS** meldet
**ZDOS ERROR ON** laufwerk: **BAD SECTOR**
Jetzt kann, falls gewünscht, die Operation durch Eingabe von **CARRIAGE RETURN** nach Entfernen des Schreibschutzes fortgesetzt werden. Der Schreibschutz ist ein Streifen, der von der Diskette entfernt (8") oder auf die Diskette geklebt (5,25") werden muß.

**Weitere Fehler können von den Dienstprogrammen gemeldet werden. Sie sind dann dort beschrieben.**

In der folgenden Beschreibung bedeutet:

| | |
|---|---|
| laufwerk: | Ausgewähltes Laufwerk A:,B:,...,H: |
| dateiname | Name der gewünschten Datei (max. 8 Zeichen) |
| .dateityp | Typ der gewünschten Datei (max. 3 Zeichen) |
| länge | Länge des Bereichs in Einheiten von 256 Bytes (zwischen 0 und 255) |
| $adresse | Startadresse des Bereichs (hexadezimal) |

Angaben in Klammern können weggelassen werden.

Falls in einem Kommando das laufwerk: weggelassen wird, ist das Betriebslaufwerk oder, falls auf ihm die Datei nicht gefunden werden kann, Laufwerk A gemeint.

Folgende Befehle sind im **KI** implementiert:

**HELP** (kommando)
Ausdrucken eines Hinweistextes auf dem Schirm. Die ersten 4 Zeichen des Kommandos sind signifikant. Benötigt wird die Datei **SYSTEM.HLP**.
Bsp: 4C>**HELP DIR**

**DIR** ((laufwerk:)(dateiname.dateityp))
Ausdrucken des Inhaltsverzeichnisses
Bsp: 0A>**DIR B:**

**TYPE** (laufwerk:)dateiname.dateityp (P)
Ausgabe von Textdateien auf dem Terminal, das P bewirkt eine seitenweise Ausgabe, d.h. alle 20 Zeilen wird das Listing gestoppt, es kann mit CR, ESC oder Blank fortgesetzt werden.
Bsp: 0A>**TYPE A:TEXT.TXT**

**ERA** (laufwerk:)dateiname.dateityp
Löschen einer Datei
Bsp: 0A>**ERA TEXT.TXT**

**REN** (laufwerk:)neuername.neuertyp=altername.altertyp
Umbenennen einer Datei
Bsp: 4A>**REN A:TEXT1.ASC=TEXT.TXT**

**SAVE** länge (laufwerk:)dateiname.dateityp ($adresse)
Ablegen eines Speicherbereichs auf Diskette
Bsp: 15A>**SAVE 25 PROG.COM $4000**

**LOAD** (laufwerk:)dateiname.dateityp ($adresse)
Laden einer Datei in den Speicher
Bsp: 0A>**LOAD PROG.COM**

fett gedruckte Zeichen in Beispielen sind Eingaben am Terminal; normale sind Ausgaben. Vor dem > steht die Benutzernummer (0 bis 15) und das Betriebslaufwerk (A bis H). **Blanks sind signifikant.**

laufwerk:
Auswahl des Betriebslaufwerks
Bsp: OB)**A:**

**LIST** (laufwerk:)dateiname.dateityp
Ausgabe einer Textdatei auf den Drucker im Hintergrund (Tabs müssen vom Drucker expandiert werden)
Bsp: OB)**LIST A:HILFE**

**STOP**
Abbrechen des **LIST**-Kommandos (Dies ist die einzige Möglichkeit, ein **LIST**-Kommando abzubrechen!)

**NEW**
Meldung eines Diskettenwechsels an das System

(laufwerk:)**dateiname** (parameterliste)
Laden und Ausführen eines Maschinenprogramms mit dem dateityp COM. Die Parameterliste wird an das Programm übergeben.
Bsp: OA)**B:ASS PROGRAMM/L**

Die Angabe des Laufwerkes ist nur erforderlich, falls weder das Betriebslaufwerk noch Laufwerk A gemeint ist. (Auf Laufwerk A wird automatisch gesucht, falls die Datei sonst nicht gefunden werden kann). Ausnahmen sind: **ERA, REN, DIR** und **SAVE.**
dateiname.dateityp können auch mehrdeutig angegeben werden. Ein Fragezeichen ersetzt ein beliebiges Zeichen an seiner Position. Ein Stern ist gleichbedeutend mit Fragezeichen bis zum Ende von Dateiname oder -typ.
Bsp: TEST?.* kann stehen für TEST1.COM, TEST.TXT oder TESTA.AAA, aber nicht für TEST12.COM; TEST*.* könnte für TEST12.COM stehen. Bei mehrdeutigen Dateibezeichnungen werden alle hiervon erfaßten Dateien beeinflußt (d.h. bei ERA *.* werden alle Dateien der Betriebsdiskette gelöscht).

Falls eine Datei mit dem Namen eines **KI**-Kommandos ausgeführt werden soll, muß zur Unterscheidung vor dem Namen eine Laufwerksbezeichnung angegeben werden. Beispiel:
Das Programm mit dem Namen LOAD.COM soll ausgeführt werden. Zur Unterscheidung zum **KI**-Befehl **LOAD** ist:
OA)**A:LOAD** einzugeben, falls das Programm LOAD auf der Diskette in Laufwerk A ist.

Ausgaben des **KI** können durch Eingabe von **^S** gestoppt, anschließend durch **^Q** fortgesetzt, und durch Eingabe von **Carriage-Return, Escape** oder **Blank** abgebrochen werden. **^S, ^C** veranlaßt einen Systemwarmstart. **TYPE** mit **P** kann durch Eingabe von Carriage-Return, Escape oder Blank nach einem Stop fortgesetzt werden. Das hierfür verwendete Zeichen ist als Abbruchzeichen gesperrt.

Die nun folgenden Befehlsbeschreibungen sind mit HELP befehl im System aufrufbar. Signifikant sind die ersten 4 Zeichen eines Befehls.

## KI-Kommando D I R

D I R zeigt das Inhaltsverzeichnis einer Diskette in folgender Form an:

```
A:  filename.ext $adr  filename.ext $adr  filename.ext $adr
```

### Eingabemoeglichkeiten:

| | |
|---|---|
| DIR | Inhaltsverzeichnis des Betriebslaufwerkes |
| DIR A: | Inhaltsverzeichnis von "A" (alle Files) |
| DIR A:*.* | Inhaltsverzeichnis von "A" (alle Files) |
| DIR *.ext | nur Files vom Typ "ext" |
| DIR X*.* | alle Files, deren Name mit "X" anfaengt |

D I R zeigt Files des aktuellen USERs und die Files von USER 0 an, ohne sie explicit zu unterscheiden. SYS-Files werden von D I R nicht erfasst.

## KI-Kommando T Y P E

T Y P E druckt ein File auf dem Terminal und bei ^P auch auf dem Drucker aus. Die Eingabe von P bewirkt ein seitenweises Drucken (jeweils 23 Zeilen); fortgesetzt wird das Listing nach Eingabe von ⟨CR⟩, ⟨ESC⟩ oder ⟨BLANK⟩. Listings werden durch Eingabe von ^S gestoppt, mit ^Q nach einem ^S fortgesetzt und durch Eingabe von ⟨CR⟩, ⟨ESC⟩ oder ⟨BLANK⟩ abgebrochen. Ein ^C nach einem ^S fuehrt einen Systemwarmstart durch.

### Eingabemoeglichkeiten:

| | |
|---|---|
| TYPE filename.ext | liste die Datei "filename.ext" |
| TYPE filename.ext P | liste die Datei "filename.ext", Stop nach jeweils 23 Zeilen |
| TYPE A:*.MAC | drucke das erste MAC-File der Diskette "A". |

Zur Verhinderung von Fehleingaben ist das Zeichen, das soeben zum Fortsetzen des Listings eingegeben wurde, nicht als Abbruchzeichen zulaessig. Es ist darauf zu achten, nur Files zu "typen", die nur druckbare ASCII-Zeichen enthalten.

## KI-Kommando L I S T

L I S T druckt ein File im Hintergrund auf dem Systemdrucker. L I S T kann nur durch Eingabe von S T O P abgebrochen werden. Waehrend des LISTENs eines Files sind die Programme FORMAT und SYSGEN nicht benutzbar. Die Diskette, die das LIST-File enthaelt, darf waehrend des "listens" nicht gewechselt werden.

### Eingabemoeglichkeiten:

| | |
|---|---|
| LIST filename.ext | Drucken von "filename.ext" im Hintergrund; Datei ist auf dem Betriebslaufwerk. |
| LIST B:TEXT.ABC | Drucken des Files "TEXT.ABC" der Diskette "B" |

L I S T arbeitet waehrend der Wartezeiten auf die Tastatur bei LEAS-Aufrufen; d.h. es arbeitet nicht bei HEAS-Aufrufen, z.B. in Verbindung mit dem WORDSTAR.

## KI-Kommando E R A

E R A loescht Dateien von der Diskette. Es sind sowohl einzelne Dateien wie auch ganze Dateigruppen erfassbar.

### Eingabemoeglichkeiten:

| | |
|---|---|
| ERA *.* | loesche alle Dateien des Betriebslaufwerks; hier wird vorsichtshalber noch mal nachgefragt und erst nach Eingabe von Y geloescht |
| ERA B:ABCDEF.* | loesche auf Laufwerk "B" die Files mit dem Namen "ABCDEF", ohne auf den Typ zu achten. |
| ERA A*.* | loesche alle Files auf dem Betriebslaufwerk, deren Name mit A beginnt. |

Die Files werden nicht wirklich geloescht, sie werden nur im Inhaltsverzeichnis als geloescht deklariert. Es koennen nur Dateien des jeweiligen Benutzers geloescht werden; die auch aufrufbaren Dateien des USER 0 (Systemprogramme) sind von anderen nicht loeschbar.

## KI-Kommando R E N

R E N benennt Dateien um. Es koennen Dateien und teilweise auch Dateigruppen erfasst werden. R E N kann nur Files des jeweiligen USERs umbenennen.

### Eingabemoeglichkeiten:

| | |
|---|---|
| REN A:ABC.123=DEF.456 | Das File "DEF.456" auf Laufwerk "A" wird in "ABC.123" umbenannt |
| REN *.MAC=*.SRC | Alle Dateien des Typs "SRC" erhalten den neuen Typ "MAC" |
| REN ?80.XXX=?80.COM | Alle Files, die als 2. und 3. Buchstaben "80" und als Typ "COM" haben, bekommen den Typ "XXX" |

Falls der neue Name bereits auf der Diskette gefunden wird, erscheint die Fehlermeldung FILE EXISTS, und REN wird nicht ausgefuehrt; es ist erst das File mit dem neuen Namen zu loeschen, dann kann das REN-Kommando wiederholt werden.

## KI-Kommando N E W

N E W teilt dem System einen Diskettenwechsel mit. N E W ist immer dann erforderlich, wenn auf eine nach dem letzten Systemwarmstart gewechselte Diskette geschrieben werden soll; die Diskette waere sonst R/O (read only). Nach einem ^C ist NEW nicht erforderlich.

### Eingabemoeglichkeit:

| | |
|---|---|
| NEW | Meldung Diskettenwechsel an ZDOS |

## KI-Kommando S A V E

S A V E rettet einen Speicherauszug auf die Diskette. Angegeben wird die Laenge in 256-Byte, der Name und die Anfangsadresse. Falls keine Adresse angegeben wird, wird 100H genommen.

### Eingabemoeglichkeiten:

| | |
|---|---|
| SAVE 1 TEST.COM $4500 | speichere 256 Byte ab Adresse 4500H unter dem Namen TEST.COM |
| SAVE 255 B:MEM | speichere 255*256 Byte ab Adresse 100H unter dem Namen MEM auf Diskette B |

## KI-Kommando L O A D

L O A D laedt ein File in den Speicher. Erforderlich sind: Filename und Startadresse, falls nicht die Originaladresse genommen werden soll.

### Eingabemoeglichkeiten:

| | |
|---|---|
| LOAD PROGRAM.COM | Lade das File PROGRAM.COM an seine Startadresse |
| LOAD PROGRAM.COM $100 | Lade PROGRAM.COM ab Adresse 100H unabhaengig von der Originaladresse |

Die Fehlermeldung **BAD LOAD** sagt aus, dass das File zu lang ist und daher nicht geladen werden kann, ohne Systemteile zu ueberschreiben. Das Laden wird abgebrochen, bevor das System ueberschrieben wird.

## KI-Kommando S T O P

S T O P ist ausser dem RESET-Schalter die einzige Moeglichkeit ein LIST-Kommando (Drucken im Hintergrund) abzubrechen.

### Eingabemoeglichkeit:

| | |
|---|---|
| STOP | "LISTen" eines Files beenden |

## andere Kommandos

Ferner sind alle Dateien mit dem Dateityp .COM genauso wie KI-Kommandos ausführbar; d.h. eine .COM-Datei mit dem Namen TEST.COM ist durch die Eingabe:
0A>TEST
genau wie ein KI-Kommando ausführbar; einziger Unterschied ist, daß sie umbenannt werden kann und nicht Teil des KI ist, sondern vor Ausführung von der Diskette in den Programmbereich PB geladen wird. Das erfordert, daß alle benötigten .COM-Dateien auf einer Diskette im Computer erreichbar sein müssen.

Dies ermöglicht es, beliebige neue Kommandos zu generieren. Im Prinzip ist jedes ausführbare Programm unter dem KI ein Kommando (oder einem Kommando vergleichbar).

Wie bereits beim KI gesagt, kann die Kommandoliste durch ausführbare Programme (Dateityp = .COM) erweitert werden. Dies wird bei den nun folgenden Dienstprogrammen (oder auch Kommandos?) ausgenutzt. Es gibt also noch die folgenden Befehle:

CONFIG (parameterliste)
Einstellen von Schnittstellenparametern; falls die parameterliste weggelassen wird, arbeitet CONFIG interaktiv.

COPY (laufw:)(kopiename.typ) (laufw:)dateiname.typ
Kopieren der Datei "dateiname.typ" in die Datei "kopiename.typ". Falls die beiden Namen gleich sind, kann "kopiename.typ" weggelassen werden, dann muß in jedem Fall laufwerk angegeben werden. Es ist möglich mit nur einem Laufwerk zu kopieren (Pause zum Diskettenwechsel vor dem Zurückschreiben)

DUMPLOD (laufwerk:)dateiname.(lod)
Ausdrucken einer Datei vom Typ ".LOD"

DO name parameterliste
Ausführen der Prozedur "name" mit den Werten aus der Parameterliste

FILCOM (laufw:)datei1.typ (laufw:)datei2.typ
Vergleichen der Dateien datei1 und datei2
Unterschiede werden als HEX-Werte ausgegeben

FILES (laufw:)dateiname.typ
Ausgabe eines ausführlichen Inhaltsverzeichnisses aller zu dateiname.typ passenden Dateien; ferner Angabe des belegten und des freien Platzes der Diskette
FILES DSK:
Ausgabe der (vereinbarten) Daten des Betriebslaufwerkes
FILES (laufw:)dateiname.typ =attr
Setzen von Attributen für Dateien. Es gibt die Attributpaare: SYS/DIR und R/O/R/W.

FORMAT
Formatieren einer Diskette. FORMAT ist ein menugesteuertes interaktives Programm

GENCOM (laufw:)dateiname $adresse
Generieren eines ausführbaren Programmes (Typ .COM) für die Startadresse adresse aus einer .LOD-Datei

IO
Ausgabe der Einstellung des IO-Bytes (Zuordnung zwischen logischen und physikalischen Peripheriegeräten)
IO ldev:=pdev:
Verändern des IO-Bytes. Das logische Gerät ldev wird dem physikalischen Gerät pdev zugewiesen

MTEST
ist ein Speichertestprogramm und wird nur benötigt, falls das Computersystem getestet werden muß

PSP (laufw:)dateiname.typ
Aufruf des Maschinensprachenentwicklungs- und Test-Paketes. Falls dateiname.typ angegeben ist, wird die entsprechende Datei gleichzeitig geladen. PSP ist ein interaktives Programm

SYSGEN
Systemerstellungs-, Systemkopier- und allg. Kopierproramm. SYSGEN ist interaktiv und menugesteuert

USER nummer (name)
Wechseln des Benutzerbereiches. Programme sind nur dem jeweiligen Benutzer zugänglich. nummer und name müssen zusammengehören, sonst wird der Wechsel nicht gestattet. (Ausnahme ist USER 0, er ist immer und für alle anderen Benutzer ohne besonderen Schutz erreichbar.)

Dies sind die mit dem ZDOS-System mitgelieferten Dienstprogramme, diese Liste kann durch Zukauf von weiteren Programmen fast beliebig verlängert werden.

**CONFIG** ist ein Dienstprogramm zur Einstellung der normalerweise vorhandenen Schnittstellen und Peripheriegeräte. **CONFIG** basiert auf Menüs, und ist daher weitgehend selbsterklärend. Beim Aufruf erscheint folgendes Menü:

```
H.K.M. CONFIGURATOR PROGRAM VERSION 2.6 (23.09.82)

SELECT I/O-PORT

(A)     FOR SERIAL I/O PORT A, NORMALLY READER/PUNCH

(B)     FOR SERIAL I/O-PORT B, NORMALLY CONSOLE

(P)      FOR PRINTER ( EPSON MX 80 )

(X)     FOR EXIT TO ZDOS
```

Jetzt kann durch Eingabe von **A**, **B** oder **P** die gewünschte Funktion ausgewählt werden. **X** beendet **CONFIG** und kehrt zum Betriebssystem zurück.

Bei Eingabe von **A** erscheint folgendes Menü:

```
SELECT BAUDRATE AND HANDSHAKE SIGNALS FOR PORT A

(A)  FOR 9600 BAUD          (J)  FOR HANDSHAKE ON
(B)  FOR 4800 BAUD          (K)  FOR HANDSHAKE OFF
(C)  FOR 2400 BAUD          (L)  FOR NO PARITY
(D)  FOR 1200 BAUD          (M)  FOR PARITY ODD
(E)  FOR  600 BAUD          (N)  FOR PARITY EVEN
(F)  FOR  300 BAUD          (O)  FOR 7 BITS
(G)  FOR  150 BAUD          (P)  FOR 8 BITS
(H)  FOR  110 BAUD          (Q)  FOR 1 STOP BIT
(I)  FOR   75 BAUD          (R)  FOR 2 STOP BITS
(X)  FOR EXECUTION
```

Hierbei wird bei **HANDSHAKE ON** nur auf Peripheriegeräte gewartet, d.h. **CTS** und **DCD** werden beachtet. **RTS** ist konstant aktiv, **DTR** ist konstant passiv.

Durch Eingabe von **P** im Hauptmenü können folgende Einstellungen an einen angeschlossenen EPSON MX80 übergeben werden:

Achtung: CONFIG kann nur richtig arbeiten, wenn vom BOOT beim Kaltstart die Baudrate des Terminal übegeben worden ist (CTC-Time-Constant)

MX-80 CONTROL CODES

(B) FOR HORIZONTAL TAB EVERY 8-TH COLUMN
(C) FOR 132 CHARACTERS/LINE
(D) FOR 80 CHARACTERS/LINE
(E) FOR PRINT EMPHASIZED
(F) FOR PRINT NOT EMPHASIZED
(G) FOR DOUBLE PRINTING
(H) FOR SINGLE PRINTING
(X) FOR EXIT

(vgl. EPSON MX80 Manual)

Die gewählten Einstellungen in den Untermenüs werden erst bei Verlassen des jeweiligen Menüs durch **X** ausgeführt. Im Untermenü **B** ist zu beachten, das bei Veränderungen der Baudrate auch die Baudrate des an Port B angeschlossenen Terminals umzustellen ist!

Zur richtigen Einstellung der Baudrate der seriellen Schnittstellen ist die Kenntnis des Systemtaktes erforderlich. Dieser wird indirekt vom Urlader **BOOT641** oder einer neueren Version an **HEAS** übergeben und in der **CONFIG**-Tabelle abgespeichert. Bei Modifikation des Bootstraploaders ist hierauf zu achten!

Eine einmal mit **CONFIG** gewählte Einstellung bleibt bis zum nächsten Kaltstart des Systems bestehen. Nach jedem Kaltstart gelten wieder die Standardeinstellungen. **CONFIG** muß also nach jedem Kaltstart neu aufgerufen werden, um andere als die Standardeinstellungen zu erhalten. (dies ist möglich in der Prozedur **INIT.DO**)

**CONFIG** verarbeitet die Eingabezeile, d.h. die sonst in den Menüs selektierten Einstellungen können auch hinter dem Aufruf stehen. Beispiel:

```
0A>CONFIG AAJMRXX
CONFIGURATION FOR PORT A IS:
9600 BAUD, HANDSHAKE: YES PARITY: ODD, 8 BIT, 2 STOPBIT
0A>
```

wählt das Menu für Schnittstelle **A**, stellt 9600 baud (A) ein, schaltet Handshake auf on (J), setzt Parity odd (M) und definiert 2 Stopbits (R); dann wird zuerst das Schnittstellenmenu (X) und dann **CONFIG** verlassen (X).

Die Eingabe der Schnittstellenparameter mittels der Eingabezeile ist die einzig sichere Version, die Parameter der Terminal-Schnittstelle zu verändern; alle interaktiven Versuche können zu Störungen des Computersystems führen.

Das Programm **COPY** ermöglicht es, einzelne Dateien zu kopieren. Die Dateilänge ist durch den zur Verfügung stehenden Speicher beschränkt. (ca 56KBytes bei 64K-Systemen.) **COPY** wird folgenderweise aufgerufen:

**COPY zieldatei quelldatei** bzw.
**COPY laufwerk: quelldatei.**

**COPY** erlaubt es, Dateien mit nur einem Laufwerk zu kopieren. Nach Einlesen der Quelldatei ist die Diskette zu wechseln und **RETURN** einzugeben; erst danach wird die Datei auf die jetzt neu eingelegte Diskette kopiert.
Die bei **COPY** verwendeten Dateinamen entsprechen der unter **ZDOS** üblichen Schreibweise:
**laufwerk:dateiname.dateityp**

Ferner besteht die Möglichkeit, mit **SYSGEN** ganze Disketten zu duplizieren (mit 2 Laufwerken); **COPY** kann nur einzelne Dateien kopieren. **COPY** kopiert bei der Eingabe:
0A)COPY A: B:*.*
nicht etwa alle Dateien von B nach A, sondern **nur die erste Datei** von B nach A.

## COPY-Fehlermeldungen

| | |
|---|---|
| NO SOURCE FILES | Die Quell-Datei kann nicht gefunden werden; falsche Eingabe |
| NO DIRECTORY SPACE | Inhaltsverzeichnis der Zieldiskette ist voll, Kopie kann nicht zurückgeschrieben werden |
| OUT OF DATA SPACE | Zieldiskette ist voll, Kopie kann nicht zurückgeschrieben werden |
| CAN'T CLOSE DESTINATION | Betriebssystemfehler, kann eigentlich nie vorkommen |
| SORRY FILE OVERLAPS SYSTEM -- NO COPY -- | Quelldatei zu lang, kann mit COPY nicht kopiert werden, da sie nicht vollständig in den verfügbaren Speicher passt |

## COPY-Meldungen und Anweisungen

| | |
|---|---|
| INSERT DESTINATION DISK AND TYPE <CR> | Quelldatei ist eingelesen, vor dem Schreiben ist die Zieldiskette einzulegen; nur falls Quell- und Zieldatei auf dem selben Laufwerk sind. |
| INSERT SYSTEM DISK AND TYPE <CR> | Betriebssystemdiskette einlegen, anschließend erfolgt Rückkehr ins Betriebssytem. |
| COPY COMPLETE | Kopie ist erstellt |

Beispiele:

1. Das Programm **COPY.COM** soll von der Diskette in Laufwerk **A** auf die in **B** befindliche Diskette kopiert werden, und dort wieder **COPY.COM** heißen.

```
0A>COPY B: A:COPY.COM

COPY COMPLETE
0A>
```

2. Wie bei 1., nur gleichzeitiges Umbennenen der neuen Datei in **TEST.**

```
0A>COPY B:TEST A:COPY.COM

COPY COMPLETE
0A>
```

3. Kopieren mit nur einem Laufwerk.
Die Datei TEST soll auf eine neue Diskette kopiert werden. Die neue Diskette ist bereits formatiert.

```
0A>COPY A: A:TEST

INSERT DESTINATION DISK AND TYPE <CR>
                        (neue Diskette einlegen und
                        RETURN drücken)
COPY COMPLETE
INSERT SYSTEM DISK AND TYPE <CR>
                        (Diskette mit ZDOS einlegen
                        und RETURN drücken)
0A>
```

### Achtung

Bei **COPY** muß die richtige Diskette angegeben werden; es sucht nicht auf Laufwerk A weiter.

**DUMPLOD** erstellt auf dem Bildschirm einen formatierten Ausdruck (Dump) einer **.LOD**-Datei. Aufruf:

```
0A)DUMPLOD (laufwerk:)dateiname
```

Beispiel:

```
0A)DUMPLOD PSP
DUMPLOD VERS 1.2

  0000 24 0000 C3 03 00 31 5A 0C 2A 06
  0009 12 0008 00 22 01 00 21 00 00 22
  0012 00 0010 06 00 21 38 00 36 C3 23
  001B 09 0018 36 50 23 36 00 21 6C 0C
  0024 08 0020 36 00 21 7C 0C 36 50 11
  002D 40 0028 DC 06 0E 09 CD 05 00 01
  0036 48 0030 72 0C CD 93 02 3A 80 00
  003F 10 0038 B7 CA C2 00 11 5C 00 32
  0048 40 0040 6C 0C 0E 0F CD 05 00 3C
  0051 24 0048 CA 25 05 C3 BB 04 18 72
  005A 04 0050 33 33 ED 73 6A 0C 31 6A
```

Eine Zeile besteht aus folgenden Feldern:

1. relative Adresse in der **.LOD**-Datei
2. 8 Bit Relokatierinformation
3. relative Adresse in der zugehörigen **.COM**-Datei
4. 8 Byte Maschinencode

Die Relokatierbits geben an, ob das zugehörige Byte Maschinencode absolut (Bit = 0) oder seitenabhängig (Bit = 1 ) ist. Das höchstwertige Bit der Relokatierinformation gehört zum ersten, das niedrigstwertige Bit zum achten der nachfolgenden Bytes.

In **.LOD**-Dateien sind nur die Felder 2 und 4 enthalten; die Adressen werden von **DUMPLOD** beim Ausdrucken berechnet. Die Aufzeichung erfolgt in 8-Bit Worten. Aufgezeichnet wird in der Reihenfolge:
1 Rel.-Byte, 8 Byte Maschinencode, 1 Rel.-Byte, 8 Byte Maschinencode, usw.
Innerhalb einer **.LOD**-Datei gilt eine Zeile, d.h. Rel-Byte und die folgenden 8 Byte Maschinencode, die vollständig aus FF besteht, als Merker für das Ende des Programms; sonst gilt auch das Ende der Datei.

## DUMPLOD-Fehlermeldungen

| | |
|---|---|
| NO LOD FILE | die zu druckende Datei kann nicht gefunden werden (auch nicht auf Laufwerk A) |
| MEMORY OVERFLOW | Datei ist zu lang (länger als ca. 56KByte) |

Das Programm **DO** ermöglicht die Verarbeitung von Prozeduren (vgl. auch Kap. 2.0). Eine Prozedur ist eine Kette von Befehlen, die unter einem Namen (Dateityp = **.DO**) auf der Diskette abgelegt werden, und so immer wieder durch Aufruf mit diesem Namen abgearbeitet werden können. Eine Prozedur wird durch folgende Eingabe gestartet:

0A>**DO prozedur par1 par2 par3 ...**

Die Parameter par1 bis maximal par9 ersetzen die formalen Platzhalter innerhalb der Befehlskette.

Beispiel: Die Befehlskette:

```
DIR $1:
DIR $2:
TYPE $3
```

wird unter dem Namen **INFO.DO** auf der Diskette abgelegt.

Mit dem Aufruf: **DO INFO A B A:SCRIPT.TXT** wird die Prozedur **INFO** ausgeführt. Die Platzhalter $1, $2 und $3 werden durch A, B und A:SCRIPT.TXT ersetzt. Somit werden jetzt die Befehle:

```
DIR A:
DIR B:
TYPE A:SCRIPT.TXT
```

im KI ausgeführt.

Für das Erstellen von Prozeduren gelten folgende Vereinbarungen:

^P und ^S sind nicht möglich.
**$1 ... $9** stehen für die Parameter 1 bis 9
**%1 ... %9** stehen für evtl. benötigte Parameter

> **Achtung: Die Parameter mit $ sollten kleinere Nummern haben als die mit %. So ist z.B. $1 $2 %3 möglich, %1 $2 hingegen nicht sinnvoll.**

**$$** ergibt das Dollarzeichen **$**
**^char** (d.h. (nur hier) zuerst ^ und dann **char**) ergibt das zugehörige Kontrollzeichen.
^Z (gleichzeitig) beendet die Befehlskette
**CARRIAGE RETURN** und ein evtl. folgendes **LINE FEED** beenden eine Prozedurzeile.
^M ergibt Carriage-Return
^C veranlasst ein Warmstart

**Eine Schachtelung von Prozeduren ist nicht erlaubt.**

## DO-Fehlermeldungen

| | |
|---|---|
| **NO PROCEDURE** | Die Prozedur existiert nicht |
| **BUFFER OVERFLOW** | Befehlskette zu lang |
| **PARAMETER ERROR** | Parameterfehler innerhalb der Befehlskette |
| **PARAMETER MISSING** | beim Aufruf wurde ein $-Parameter vergessen |

## Abbruchkriterien während der Laufzeit einer Prozedur

Alle Dienstprogramme brechen bei schwerwiegenden Fehlern jede Prozedur ab. Bei Eingabe von ^S, ^C am Terminal (STOP und Abbruch-Funktion) wird jede Prozedur abgebrochen. Die Antwort ^C bei einer LEAS oder HEAS-Fehlermeldung bricht Prozeduren ab.

> Eine Prozedur kann durch Eingabe von ^S, ^C am Terminal während ihrer Ausführung abgebrochen werden. Ferner können Programme bei Fehlern die Prozedur durch HEAS-Aufruf abbrechen.

Prozeduren können aneinandergehängt werden. So ist z.B. eine endlose Prozedur möglich, die sich selbst imme wieder aufruft:

```
SYSGEN
V                    Vergleich der Disketten in A und B
A
B
^M                   zurück zu ZDOS
DO TEST              Wiederaufruf der Prozedur TEST.DO
```

**FILCOM** dient dazu, 2 Dateien zu vergleichen. Mit der Eingabe: **FILCOM datei1 datei2** werden die Dateien **datei1** und **datei2** miteinander verglichen. Als Beispiel sollen die beiden Assembler Quelltexte **TEST.MAC** und **TEST.BAK** miteinander verglichen werden. (TEST.BAK ist die alte Version von TEST.MAC)

Listing von TEST.BAK:

```
            TITLE TEST
    ;
            LD      DE,MESS
            LD      C,9                 ; PRINT STRING
            CALL    8                   ; ZDOS SERVICE CALL
            RET
    ;
    MESS:   DEFB    0DH,0AH,'TEST LAEUFT$'
    ;
            END
```

Listing von TEST.BAK:

```
            TITLE TEST
    ;
            LD      DE,MESS
            LD      C,9                 ; PRINT STRING
            CALL    5                   ; ZDOS SERVICE CALL
            RET
    ;
    MESS:   DEFB    0DH,0AH,'TEST LAEUFT$'
    ;
            END
```

Vergleich der beiden Dateien:

```
    0A>FILCOM B:TEST.BAK B:TEST.MAC

    H.K.M. FILE COMPARE PROGRAM
    TYPE CR TO EXECUTE
    OR ^C TO ABORT                          (RETURN eingeben)
                                      (Anfang des Vergleichs)
    ** 003C: 38  35                    (Unterschied gefunden)
    ** FINISHED **                      (Ende des Vergleichs)
    TYPE CR TO RETURN TO ZDOS               (RETURN eingeben)
    0A>
```

### FILCOM-Fehlermeldungen

| | |
|---|---|
| **** NO FILE **** | keine Datei mit diesem Namen |
| **** FILELENGTH NOT EQUAL **** | unterschiedliche Längen |
| **** ABORTED **** | Vergleich abgebrochen |

und

| | |
|---|---|
| **** FINISHED **** | normal beendet |

**FILES** ist ein Programm mit folgenden Funktionen:

1. erweitertes Directory-Listing mit Angabe von Dateiname, Lade- bzw. Startadresse, Attributen, Dateilänge in Blocks und Anzahl der Sektoren und Directory-Einträge. Die Dateilänge bezieht sich auf den wirklich belegten Platz auf der Diskette, die Anzahl der Sektoren stimmt nur für sequentielle Dateien, bei RANDOM-Dateien ist **RECS** die Nummer des letzten Sektors. Beispiel:

```
0A>FILES B:*.*
   FILE          ADR     ACC  SYS/DIR  RECS  LENGTH  ENTRIES
DO       .COM    0100    R/W    DIR       5      2K        1
FILCOM   .COM    0100    R/W    DIR       7      2K        1
GENCOM   .COM    0100    R/W    DIR       4      2K        1
PSP      .LOD    0100    R/W    DIR      28      4K        1
BYTES ALLOCATED :  16K     BYTES FREE :  372K
```

2. Ausgabe der Diskparameter des Betriebslaufwerks: Angezeigt werden die Kapazität, die Anzahl der Einträge im Inhaltsverzeichnis, die Extentlänge, die Blocklänge, die Sektorgröße und die Anzahl der Systemspuren. Beispiel:

```
0A>FILES DSK:
CAPACITY:                  388 KBYTE
DIRECTORY ENTRIES:          64
EXTENT=                     32 KBYTE
BLOCK=                       2 KBYTE
RECORD=                    128  BYTE
RECORDS/TRACK:              40
RESERVED TRACKS:             2
```

3. Setzen der Datei-Attribute
   Dateiattribute sind **SYS** bzw. **DIR** und **R/W** bzw. **R/O**. **SYS** bedeutet, daß die Datei nicht im Inhaltsverzeichnis erscheint; **DIR** hebt **SYS** wieder auf. **R/O** bedeutet Nur-Lese-Datei; **R/W** heißt Schreib/Lese-Datei.

   FILES Fehlermeldungen

   | | |
   |---|---|
   | SYNTAX ERROR | falsches Attribut, es gibt nur SYS DIR R/W R/O |
   | NO SUCH FILE | die zu verändernde Datei existiert nicht |

   Beispiel:

```
0A>FILES B:ZDOS.TXT =R/O
FILES SET TO R/O
   FILE          ADR       ACC   SYS/DIR
ZDOS     .TXT    4D00      R/O     DIR
BYTES ALLOCATED :  246K  BYTES FREE :  186K
```

**FILES** sucht nicht automatisch auf Laufwerk A.

FORMAT ist ein Progamm zur Formatierung von soft-sektorierten Disketten; es ist abhängig von der verwendeten Hardware (Controller uPD 765)

Die in dem hier verwendeten System benutzten Disketten sind soft-sektoriert und müssen daher vor Benutzung formatiert werden. Als Format wird **MFM** mit der Sektorlänge von 512 Byte / Sektor verwendet. Auf 5,25" Disketten werden 10, auf 8" Disketten 15 Sektoren pro Spur angeordnet. Die zweite Seite der Diskette wird hinter der ersten angehängt. Die Daten eines jeden Sektors werden invertiert aufgezeichnet. Sektoren werden ab 1, Spuren ab 0 gezählt. Füllerbyte für invertierte Aufzeichnung ist 1A, für normale Aufzeichnung wird E5 verwendet (8" Single Density). Seiten sind 0 (auch für einseitige Disketten) und 1 (zweite Seite).

Ferner wird für 8" Disketten zusätzlich das "Standard CP/M Format" unterstützt, d.h. **FM**-Aufzeichnung (Single Density) mit 26 Sektoren / Spur, 128 Byte Sektorlänge einseitig auf 77 Spuren und **nicht**invertierter Aufzeichnung.

Da das **Format**-Programm für alle Versionen gültig sein soll, ist es für die jeweilige Hardware installierbar. Dies geschieht beim ersten Aufruf, ist aber jederzeit später noch zu verändern. Es gibt ein Menu für von **ZDOS** verwendete Diskettenformate und eine Möglichkeit, ein USER-Diskettenformat zu vereinbaren. Aus dem Menu kann ein Diskettenformat durch Angabe des Kennbuchstabens (A..K) gewählt werden; für das USER-Format werden die benötigten Parameter interaktiv abgefragt (hierbei muß der Benutzer genau über das gewünschte Format Bescheid wissen). Zur Wahl des richtigen Kennbuchstabens sollte man sich an die Meldung des **HEAS** erinnern. Mögliche Meldungen sind:

| | |
|---|---|
| HEAS VERSION 5.5 DMA-VERSION 5,25" | **A** oder **B** |
| (80 Track) | **C** oder **D** |
| HEAS VERSION 5.5 NON-DMA-VERSION 5,25" | **H** oder **I** |
| HEAS VERSION 5.5 DMA-VERSION 8" | **E** oder **F** |
| HEAS VERSION 5.5 EPC 5,25" | **J** |
| (80 Track) | **K** |

Der jeweils erste fett gedruckte Buchstabe gilt für Single-Sided-Laufwerke, der zweite für Double-Sided-Laufwerke. (Dies kann ausprobiert werden: Installiert wird die Double-Sided-Version, falls dann das **FORMAT**-Programm am Anfang der zweiten Seite hängen bleibt, war das Laufwerk Single-Sided. In jedem Fall muß die erste Seite einwandfrei formatierbar sein.)

Den Unterschied zwischen 80 und 40 Track-Laufwerken kann man notfalls folgendermaßen bestimmen: Es wird eine Seite mit 80 Spuren formatiert; anschließend wird mit SYSGEN (siehe dort) versucht, die Spuren 40 bis 80 zu lesen. Wenn dies fehlerfrei möglich ist, handelt es sich um ein 80 Spur Laufwerk, sonst wahrscheinlich um ein 40 Spur-Laufwerk. Das USER-Diskettenformat ist in den Grenzen des uPD 765-Floppy-Controller-ICs frei wählbar (siehe techn. Unterlagen); leichte Modifikationen können durch Auswählen eines Standardformates (A..K) und anschließendes Ändern der eingestellten Parameter erfolgen. (Ein gleichbleibender Wert muß hier neu eingegeben werden, Carriage-Return setzt eine 0 ein.) Das Installieren von **FORMAT** bezieht sich nur auf das FORMAT-Programm und nicht auf **SYSGEN** oder **HEAS.**

Einige Beispiele zur Benutzung von **FORMAT:**

Die Diskette in Laufwerk B soll formatiert werden. FORMAT ist bereits installiert; es soll das normale Format verwendet werden. Die Hardware besteht aus Double-Sided-40-Track-Laufwerken. Eingaben sind die fett gechriebenen Buchstaben.

```
0A>FORMAT
FORMAT V. 3.3 (01.02.83)
(Type X for Exit)
(B)   5,25", MFM, SS, 40 TRK   - FDC at 00
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = B
FORMATTING TRACK 00   bis 39 für Seite 0
FORMATTING TRACK 00   bis 39 für Seite 1
hit CARRIAGE RETURN to return to ZDOS
0A>
```

Formatieren von nur Spur 5 einer Diskette (FORMAT ist bereits richtig installiert) -- gezählt wirdd ab 0 --

```
0A>FORMAT
FORMAT V. 3.3 (01.02.83)
(Type X for Exit)
(B)   5,25", MFM, SS, 40 TRK   - FDC at 00
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = T
START FORMATTING AT SIDE (0..1): 0
STARTING TRACK (0..79): 5
STOP FORMATTING AT SIDE (0..1): 0
LAST TRACK (0..79): 5
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = B
FORMATTING TRACK 05
hit CARRIAGE RETURN to return to ZDOS
0A>
```

Mit dem FORMAT aus dem vorigen Beispiel soll eine einseitige Diskette formatiert werden. Damit ist FORMAT vor der Ausführung umzuinstallieren:

```
0A>FORMAT
FORMAT V. 3.3 (01.02.83)
(Type X for Exit)
(B)    5,25", MFM, SS, 40 TRK  - FDC at 00
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = I
TYPE A...K TO SELECT, U FOR YOUR OWN LAYOUT
OR X TO CHANGE FDC-BASE-ADDRESS
(A)    5,25", MFM, SS, 40 TRK
(B)    5,25", MFM, DS, 40 TRK
(C)    5,25", MFM, SS, 80 TRK
(D)    5,25", MFM, DS, 80 TRK
(E)       8", MFM, SS, 77 TRK
(F)       8", MFM, DS, 77 TRK
(G)       8" STANDARD CP/M
(H) NON-DMA, MFM, SS, 40 TRK
(I) NON-DMA, MFM, DS, 40 TRK
(J)     EPC, MFM, DS, 40 TRK
(K)     EPC, MFM, DS, 80 TRK
A
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = B
FORMATTING TRACK 00   bis 39 für Seite 0
hit CARRIAGE RETURN to return to ZDOS
```

Es soll ein FORMAT für zweiseitige 35-Track-Mini-Laufwerke erstellt werden. Es gibt 3 Laufwerke im Computersystem: Zuerst ist (B) zu wählen, danach kann die Anzahl der Spuren geändert werden.

```
0A>FORMAT
FORMAT V. 3.3 (01.02.83)
(Type X for Exit)
not installed, please install  - FDC at 00
TYPE A...K TO SELECT, U FOR YOUR OWN LAYOUT
OR X TO CHANGE FDC-BASE-ADDRESS
(A)    5,25", MFM, SS, 40 TRK
(B)    5,25", MFM, DS, 40 TRK
(C)    5,25", MFM, SS, 80 TRK
(D)    5,25", MFM, DS, 80 TRK
(E)       8", MFM, SS, 77 TRK
(F)       8", MFM, DS, 77 TRK
(G)       8" STANDARD CP/M
(H) NON-DMA, MFM, SS, 40 TRK
(I) NON-DMA, MFM, DS, 40 TRK
(J)     EPC, MFM, DS, 40 TRK
(K)     EPC, MFM, DS, 80 TRK
B
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = I
TYPE A...I TO SELECT, U FOR YOUR OWN LAYOUT
OR X TO CHANGE FDC-BASE-ADDRESS
```

```
(A)    5,25", MFM, SS, 40 TRK
(B)    5,25", MFM, DS, 40 TRK
(C)    5,25", MFM, SS, 80 TRK
(D)    5,25", MFM, DS, 80 TRK
(E)       8", MFM, SS, 77 TRK
(F)       8", MFM, DS, 77 TRK
(G)       8" STANDARD CP/M
(H) NON-DMA, MFM, SS, 40 TRK
(I) NON-DMA, MFM, DS, 40 TRK
(J)     EPC, MFM, DS, 40 TRK
(K)     EPC, MFM, DS, 80 TRK
U
(H) = HEX, (D) = DEC, (S) = STRING INPUT WANTED !
RETRY-COUNTER (D) = 3 = 3
FM (0) OR MFM (1) : 01 = 1                                  (MFM)
SINGLE (0) OR DOUBLE (1) SIDED : 01 = 1      (zweiseitig)
TRACKS / SIDE (D) = 40 = 35          (35 Spuren pro Seite)
FORMAT GAP LENGTH (H) = 1E = 1E            (1EH für 5,25")
FILLER BYTE (H) = 1A = 1A       (invertierte Aufzeichung)
N (SECTOR-SIZE) = 02 = 2           (512 Byte Sektorlänge)
          (0=128, 1=256, 2=512, 3=1024 Bytes Sektorlänge)
SECTORS / TRACK (D) = 10 = 10      (10 Sektoren pro Spur)
R/W-GAP-LENGTH (H) = 10 = 10               (10H für 5,25")
NUMBER OF DISKS = 2 = 3           (3 Laufwerke im System)
1 STEP (0) or 2 STEPS (1) = 0 = 0                  (normal)
                       (eine  1  hier  emuliert ein  40  Spur-
                          Laufwerk auf einem 80 Spur-Laufwerk)
NAME FOR THIS FORMAT = 5,25", DOUBLE SIDED, 35 TRACK
FORMAT DISK (A..D or I for INSTALL or T for SETTRACK) = B
FORMATTING TRACK 00   bis 34 für Seite 0
FORMATTING TRACK 00   bis 34 für Seite 1
hit CARRIAGE RETURN to return to ZDOS
to save this Version, type: SAVE 15 FORMAT.COM
0A>SAVE 15 FORMAT35.COM
```

ab jetzt ist die so installierte Version **FORMAT35** jederzeit wieder aufrufbar und ohne weitere Installation benutzbar.

Ein falsch installiertes FORMAT hängt das System auf
**GENCOM** ist ein Lader für Page-relokatierbare Dateien (Dateityp = .LOD). Da das Betriebssystem seitenweise (d.h. in Schritten zu 256 Byte) relokatierbar ist, muß vor dem Kopieren auf die Systemspuren aus der .LOD-Datei eine .COM-Datei erstellt werden. Hierzu wird **GENCOM** aufgerufen:

**GENCOM laufwerk:dateiname $adresse**

**GENCOM** erstellt eine Datei mit dem Namen **dateiname.COM** auf der durch **laufwerk** selektierten Diskette, bzw. der Diskette A, falls die .LOD-Datei auf laufwerk nicht gefunden werden konnte.

Folgende Fehler können gemeldet werden:

| | |
|---|---|
| ** NO MEMORY ** | oberhalb der Adresse OFFFFH |
| ** CAN'T SAVE COM-FILE ** | die Diskette ist voll |
| ** NO LOD-FILE ** | **dateiname.LOD** existiert nicht |
| ** NO ADDRESS ** | die Angabe **$adresse** fehlt |
| ** MEMORY OVERFLOW ** | die Datei ist zu lang |

Das in **LOD**-Dateien verwendete Format wird bei **DUMPLOD** (vgl. Kap. 2.3) näher beschrieben.

Prinzipiell kann Software im **LOD**-Format nicht direkt ausgeführt werden; sie muß in jedem Fall erst mittels **GENCOM** in ein ausführbares Format (**COM**-Datei) umgeformt werden.

Achtung: die Meldung ** NO MEMORY ** erscheint auch bei falschem .LOD-Code. Die GENCOM-Version von ZDOS 5.a kann LOD-Files der **ZDOS**-Version 5.b nicht richtig verarbeiten (Ende-Kennung ist ab jetzt neu in LOD-Dateien).

Die Ende-Kennung besteht aus einem Rel-Byte FF und 8 folgenden Maschinencode-Bytes FF. (vergleiche DUMPLOD).

Die höchste Anfangsadresse für **ZDOS**, d.h. KI, **LEAS** und **HEAS** ist OE000H, da sich alle Teile auf den KI-Anfang beziehen. **PSP** kann dann maximal an Adresse OD300H beginnen.

**IO** ist ein Hilfsprogramm zur Zuordnung der logischen zu den physikalischen Geräten. Folgende Eingaben sind möglich: (vgl. Die **IOBYTE**-Funktion)

| | |
|---|---|
| OA>**IO** | erstellt eine Liste der aktuellen Zuordnungen. |
| OA>**IO ldev:=pdev:** | ordnet dem logischen Gerät **ldev:** das physikalische Gerät **pdev:** zu. |

Beispiele:

```
OA>IO          (Zuordungsliste)

CON: IS CRT:  (TTY:,CRT:,BAT:,UC1:)
RDR: IS TTY:  (TTY:,PTR:,UR1:,UR2:)
PUN: IS TTY:  (TTY:,PTP:,UP1:,UP2:)
LST: IS LPT:  (TTY:,CRT:,LPT:,UL1:)


OA>IO LST:=TTY:        (Zuweisung)

CON: IS CRT:  (TTY:,CRT:,BAT:,UC1:)
RDR: IS TTY:  (TTY:,PTR:,UR1:,UR2:)
PUN: IS TTY:  (TTY:,PTP:,UP1:,UP2:)
LST: IS TTY:  (TTY:,CRT:,LPT:,UL1:)
```

Logische Geräte **ldev:** sind: **CON:**, **RDR:**, **PUN:**, **LST:**.

### IO-Fehlermeldungen

| | |
|---|---|
| END OF LINE : xx? | Zeile ist unvollständig |
| xx? | Gerätename ist nicht zulässig |

Im **HEAS** verwendete Zuordnung zwischen physikalischen Geräten **pdev:** und Schnittstellen auf der CPU-Karte:

| | | | |
|---|---|---|---|
| **TTY** | = | **SIO** Port A | |
| **CRT** | = | **SIO** Port B | |
| **PTR** | = | ..... | (frei) |
| **PTP** | = | ..... | (frei) |
| **LPT** | = | **PIO** Centronics | (8 bit) |
| **UC1** | = | **SIO** Port B | |

```
    UR1   =    SIO Port B
    UR2   =    SIO Port B      (8 bit)
    UP1   =    SIO Port B
    UP2   =    ..........      (frei)
    UL1   =    ..........      (frei)
```

**Speichertestprogramm**

**MTEST** ist ein Speichertestprogramm, das den Speicher ab Adresse 2000H bis zu Adresse FFFFH testet. Hierzu werden quasizufällige Muster in den Speicher geschrieben, und nach Beschreiben des gesamten Speichers wieder gelesen und verglichen. **MTEST** findet fast alle in Speichern vorkommenden Fehler. **MTEST** überschreibt beim Testen des Speichers das Betriebssystem und muß daher eigene Ein-Ausgabe-Routinen benutzen. **MTEST arbeitet nur mit dem** Terminal, das an Port **B** der **SIO** angeschlossen ist. Nach dem Aufrufen meldet sich das Programm mit:
**MEMORY TEST PROGRAM**
Anschließend wird ein statischer RAM-Test durchgeführt.
**STATIC TEST RUNNING**
Nach Beendigung des statischen Speichertest:
**STATIC TEST COMPLETE**
beginnt der dynamische Speichertest mit insgesamt 256 Durchläufen. Jeder erfolgte Durchlauf wird durch einen Punkt auf dem Terminal dokumentiert.
**DYNAMIC TEST RUNNING**
Nach Beendigung des letzten der 256 Durchläufe erfolgt die Meldung:
**DYNAMIC TEST COMPLETE**
Anschließend geht das Testprogramm in eine **Endlos-schleife.** Zum Weiterarbeiten mit dem System ist jetzt der **RESET-Taster** zu betätigen und damit das System neu zu starten.
Von **MTEST** gefundene Fehler im Speicher werden folgender-weise angezeigt:
***** ERROR *** adresse SHOULD BE data1 IS data2 OR data3**
D.h. an der angegebenen Adresse wurde statt data1 ein anderer Wert data2 oder data3 gelesen. Unterschiede zwischen data2 und data3 deuten auf fehlende bzw. nicht ansprechbare Speicherchips hin. Zum Deuten anderer Feh-ler sei hier der Algorithmus zur Erzeugung der Quasi-zufallsfolge angegeben:

```
;
;****** DYNAMIC TEST **********************************
;
DYNTES: LD      B,0         ; Laufzähler
STRT:   LD      HL,2000H    ; Startadresse
FILL:   LD      A,L         ; Berechnung Quasizufalls-
        XOR     H           ; folge L xor H xor B
        XOR     B           ; mit HL = Adresse und
        LD      (HL),A      ; B = Durchlaufzähler
        INC     HL
        LD      A,H
        CP      0           ; High(Endadresse+1)
        JR      NZ,FILL
```

## 0. PSP-Aufruf

Es gibt 2 Möglichkeiten, den PSP aufzurufen:

```
0A)PSP
0A)PSP (laufwerk:)dateiname.typ
```

Beide Aufrufe starten den PSP, der zweite Aufruf lädt vorher noch die Datei (laufwerk:)Dateiname.typ an die in ihrem FCB vereinbarte Adresse im Speicher. Diese Eingabe ersetzt die Befehle:

```
)I(laufwerk:)dateiname.typ
)R.
```

Es ist darauf zu achten den Dateityp .COM bei ausführbaren Programmen beim Aufruf mitanzugeben. Die Eingabezeile:

```
0A)PSP__
```

veranlasst die Meldung: FILE NOT FOUND, da nach einer Datei mit dem Namen blank gesucht wird, die es unter ZDOS nicht geben kann!

## 1. Einleitung

**PSP** stellt ein Hilfsmittel dar, um interaktiv Maschinenprogramme sowie den Zustand des Rechners zu testen und zu kontrollieren. Nach der Anmeldung (SIGN ON MESSAGE) verlangt **PSP** Eingaben mit ')', falls das zuletzt eingegebene Kommando erfolgreich ausgeführt wurde und mit '?', falls das zuletzt eingegebene Kommando wegen syntaktisch falscher Eingabe nicht ausgeführt werden konnte.
Jedes Kommando kann bis zu 80 Zeichen lang sein (das Zeilenende wird als 81-tes Zeichen automatisch hinzugefügt), wobei das erste Zeichen das Kommando darstellt. Es gibt folgende Kommandos:

Ybyte1 byte2 ...
    Byte-String im Speicher lokalisieren

Danfang ende
    Speicherausdruck (DUMP) in hex und ASCII

Fanfang ende wert
    Füllen des Speichers mit konstanten Werten

Gstart stop
    Ausführung (GO) eines Maschinenprogramms

Lanfang ende
    Disassemblierung (LIST) des Speicherinhaltes

Mvon bis nach
    Verschieben (MOVE) von Speicherinhalten

Xregister
    Register-Anzeige und Veränderung

Sadresse
    Ersetzen (SUBSTITUTE) von Speicherinhalten

Tvon bis mode
    Schrittweise Überwachung (TRACE) von Programmen

Vanfang1 anfang2
    Vergleichen (VERIFY) zweier Speicherbereiche

Itext
    Eingabe von Dateinamen

Radresse
    Datei einlesen

Wvon bis
    Datei abspeichern

Das Kommandozeichen benötigt, je nach Kommando, bis zu drei Hexadezimalzahlen, die mit einem beliebigen Trennzeichen (alle Zeichen außer 0..9 und A...F) abgeschlossen werden können.

## 2. Die PSP-Kommandos

Die einzelnen Kommandos werden nachfolgend beschrieben. Bei der Beschreibung der Kommandos werden die hexadezimalen Zahlen durch 'hhhh' mit Leerzeichen getrennt dargestellt. Sie dürfen von einem bis zu vier hexadezimale Zeichen enthalten (Längere Zahlen werden automatisch auf die vier letzten gekürzt).

### 2.1 Byte-String im Speicher lokalisieren

Mit dem **Y**-Kommando wird der gesamte Speicher nach dem Vorkommen eines Bytestrings durchsucht. Ausgegeben wird die Adresse des Stringanfangs.
Das Format ist:

**>Yhh hh hh hh .....**

### 2.2 Ausdrucken des Speicherinhaltes

Mit dem D-Kommando kann man Speicherinhalte in hexadezimaler und zeichenweiser (ASCII) Darstellung ausdrucken. Die möglichen Formate sind:

**>Dhhhh hhhh**
**>Dhhhh**

Im ersten Fall wird der Speicher zeilenweise von der Anfangs- bis zur Endadresse ausgegeben. Im zweiten Fall werden ab hhhh 16 Zeilen dargestellt.

```
>D1005 1009
      0 1  2 3  4 5  6 7  8 9  A B  C D  E F
1000  1234 1235 1236 1237 1238 1239 1241 1242 #.4.5.6.7.8.9.A.B
```

In ASCII nicht darstellbare Speicherinhalte werden als '.' dargestellt. Das höchstwertige Bit wird bei der ASCII-Darstellung nicht beachtet.

## 2.3 Das Laden des Speichers mit Konstanten

Das F-Kommando hat die Form:

**>Fhhhh hhhh hh**

wobei die erste Hexadezimalzahl die Anfangsadresse, die zweite die Endadresse und die dritte eine Bytekonstante, mit der der Speicherbereich beschrieben werden soll, ist. Man beachte:

- Falls die zweite Adresse kleiner als die erste ist, wird über FFFF hinaus bis zu der angegebenen Adresse der Speicher gelöscht.
- Das ZDOS-Betriebssystem wird zwar von Platte geladen, jedoch ist nach Löschen des Bereichs 0000-0100 bzw. E800-FFFF das System nur noch mit Hardware-RESET zu laden.

## 2.4 Das Starten von Benutzerprogrammen

Mit dem Kommando G wird ein Programm gestartet. Das Kommando kann folgende Gestalt annehmen:

**>Ghhhh**
**>Ghhhh hhhh**
**>G,hhhh**

Die erste Form startet ein Benutzerprogramm, wobei die Kontrolle dem Monitor entzogen wird. Die zweite Form startet ein Benutzerprogramm an der angegebenen Adresse und führt es bis zur zweiten aus, wonach die Kontrolle wieder an den Monitor gelangt. **Man stelle sicher, daß die zweite Adresse auch erreicht wird.** Die dritte Form dient dazu, fortlaufend von der zuletzt erreichten Adresse bis zur durch hhhh angegebenen auszuführen.

## 2.5 Die Disassemblierung von Programmen

Mit dem L-Kommando lassen sich Programme im Speicher disassemblieren. Die Form ist:

**>Lhhhh**
**>Lhhhh hhhh**

Die erste Zahl gibt die Anfangsadresse, die zweite (falls vorhanden) die Endadresse an. Jederzeit kann die Disassemblierung durch Eingabe eines beliebigen Zeichens abgebrochen werden. Bei fehlender Endadresse werden 20 Bytes disassembliert.

## 2.6 Das Verschieben von Speicherinhalten

Mit dem M-Kommando werden Speicherinhalte verschoben. Die Form ist:

```
>Mhhhh hhhh hhhh
```

Die drei angegebenen Adressen sind: von, bis, nach. Man beachte, daß im Gegensatz zu den meisten Implementierungen sich die Bereiche ohne Schaden überlappen dürfen.

## 2.7 Registerausgabe und Änderung

Mit dem Kommando X werden die Register angezeigt und bei Hinzufügung eines Registernamens verändert. Registernamen sind: S, A, B, D, H, X und Y. Als Register werden Doppelregister (16 bit) betrachtet.

Ein Beispiel zur Anzeige:

```
>X
 PC   (PC)  SP   AF   BC   DE   HL   IX   IY (BC DE HL IX IY)
1234  00   2345 3456 4567 5678 6789 7890 8901 00 00 00 00 00
>
```

Ein Beispiel zur Veränderung von Registern:

```
>XA
0000 1234          (d.h. A=12, F=34)
>
```

## 2.8 Das Verändern von Speicherinhalten

Mit dem S-Kommando werden Speicherinhalte einzeln angezeigt und möglicherweise geändert. Das Kommando hat die Form:

```
>S1FFC
1FFC 21=
1FFD 00=11
1FFE 11=00
1FFF C3=
2000 FF=00
?
```

Folgendes ist geschehen:

Der Inhalt von 1FFC wird als 21 angezeigt. Das eingegebene RETURN läßt ihn unverändert. Der Inhalt von 1FFD war 00 und wird zu 11 geändert. Der Inhalt von 1FFE war 11 und wird zu 00 geändert. Der Inhalt von 1FFF bleibt unverändert C3. Der Inhalt von 2000 wird als FF angegeben, jedoch war der Versuch ihn zu 00 zu ändern nicht erfolgreich, da der Monitor sich mit '?' meldet. Der Grund ist einfach: Der Speicherbereich ab 2000 war schreibgeschützt. Verlassen wird das S-Kommando durch Eingabe von Leerzeichen und dann Carriage Return.

## 2.9 Das schrittweise Überwachen von Programmen

Mit dem T-Kommando kann auf schnellste Weise die Ausführung eines Maschinenprogrammes verfolgt werden. TRACE überwacht nur Hauptprogramme; Unterprogramme werden immer in Echtzeit ohne Überwachug durchgeführt. Das Kommando hat die Gestalt:

**)Thhhh hhhh m**

Dabei bedeuten die ersten beiden Hexadezimalzahlen die Adressen von bis, das m den Modus. Es gibt drei verschiedene Betriebsarten (Modi):

m=2 Es wird ununterbrochen der angegebene Bereich ausgeführt und überwacht

m=1 Nach jeder Seite (12 Zeilen) wird die überwachte Ausführung unterbrochen. Sie kann mit RETURN fortgeführt werden oder mit 'M' beendet werden.

m=0 (oder andere Eingabe)
Es wird ähnlich wie bei der seitenweisen Überwachung nach jedem Befehl mit der Anforderung einer Eingabe unterbrochen.

```
)T1000 1011 0
1000  LD      HL,2121     FF80 0000 0000 0000 2121 0000 0000
1003  JP      NZ,100A     FF80 0000 0000 0000 2121 0000 0000
M
)
```

Erläuterungen zum Ausdruck:

Die erste Zahl stellt die Adresse des auszuführenden Befehls dar; darauf folgt die mnemotechnische Darstellung des Befehls. Die nachfolgende Kolonne gibt die Registerinhalte in der Reihenfolge: SP, AF, BC, DE, HL, IX, IY **nach der Ausführung** des Befehls wieder.

Man beachte:

- Da die Befehle nicht simuliert sind, sondern von der Z80-CPU durchgeführt werden und mit Stolperstein die Analyse anschliessend durchgeführt wird, sind einige Punkte zu beachten.

- Unterprogrammaufrufe werden nicht verfolgt, sie sind also gegebenenfalls getrennt zu prüfen.

- Die Gruppe der **RESTARTS** kann nicht überwacht werden und führt zu einem Abbruch mit Rückkehr in den Monitor.

- Die Gruppe der **RETURNS** wird nicht ausgeführt, im bedingten Fall führt eine zutreffende Bedingung wie bei einem unbedingten **RETURN** zum Abbruch.

- Die Bereiche des **PSP** können nicht überwacht werden.

- Programme im ROM oder EPROM können nicht überwacht werden.

- Befehle, die den Stackpointer umladen, sind nicht zulässig. (z.B. EX IY,SP)

## 2.10 Der Vergleich von Speicherinhalten

Mit dem V-Kommado werden zwei angegebene Speicherbereiche miteinander verglichen. Die Form ist:

**)V8000 9000**

Die angegebenen Bereiche werden byteweise miteinander verglichen. Dies geschieht solange, bis sich die Inhalte unterscheiden. Darauf werden die Inhalte ausgegeben und auf eine Eingabe gewartet. Mit RETURN wird der Vergleich fortgesetzt, mit jeder anderen Eingabe (vornehmlich Zwischenraum) gelangt man wieder in den Monitor. **Falls die beiden Adressen gleich sind entspricht dies einer unendlichen Schleife!**

## 2.11 Das I-Kommando

Das I-Kommando erfüllt zwei Aufgaben.

1. Vorbereitung für Lesen oder Schreiben.

   ```
   )ITEST.COM
   )R
   ```

   bewirkt, daß die Datei TEST.COM geladen wird.

2. Setzen der Speicherbereiche 005C ... 007F (voreingestellter FCB) und 0080 ... 00FF (Eingabezeile). In dieser Form ersetzt das I-Kommando die Eingabe im KI. Dies ermöglicht es, Programme zu testen, die die KI-Eingabezeile benutzen.

   ```
   )ITEST.COM
   )R
   )IDATEI1 DATEI2 A B C
   )Gstartadresse
   ```

   ersetzt die KI-Eingabe:

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

## 2.12 Einlesen einer Datei

Mit dem R-Kommando wird eine mittels I-Kommando angegebene Datei geladen. Der **PSP** überwacht hierbei den verfügbaren Speicher. Das R-Kommando hat die Form:

```
)R
)Rhhhh
```

hhhh ist die Adresse, ab der die Datei geladen werden soll. Achtung: **Die beiden letzten Stellen von hhhh werden als 00 angenommen.** R allein lädt die Datei an die im FCB vereinbarte Adresse. ( siehe auch unten).

## 2.13 Abspeichern einer Datei

Das W-Kommando lädt einen Speicherbereich auf die Diskette. Es hat die Form:

```
)WhhOO hhFF
```

Unter dem mittels I-Kommando vereinbarten Dateinamen wird der Bereich zwischen Start- und Endadresse auf Diskette abgelegt.

Das Programm **SYSGEN** erfüllt 3 Funktionen:

- Betriebssystem kopieren (Spur 0 und 1)
- ganze Disketten (oder einzelne Spuren) kopieren
- Vergleich von Disketten (nach dem Kopieren)

Da die Systemspuren unter dem **ZDOS**-Dateisystem nicht erreichbar sind, ist zum Kopieren des Betriebssystems ein besonderes Kopierprogramm erforderlich. Diese Aufgabe wird von **SYSGEN** übernommen. **SYSGEN** ist ein Kopierprogramm, das sowohl das Betriebssystem, wie auch ganze Disketten oder nur bestimmte Spuren kopieren oder vergleichen kann. Es ist genauso wie **FORMAT** (vgl. dort) installierbar. Folgende Beispiele verdeutlichen die Benutzung von **SYSGEN**:

Installierung **I** für einseitige Mini-Disketten **A**; nur Betriebssystem kopieren **S** von **A** nach **B**; das anschließende **RETURN** übergibt die Kontrolle wieder an **ZDOS**.

```
0A)SYSGEN
SYSGEN VERS. 3.2 (26.10.82)
(TYPE X FOR EXIT)
(B)  5,25", MFM, DS, 40 TRK
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                 SETTRACK (T) OR INSTALL (I) I
TYPE A...I TO SELECT, U FOR YOUR OWN LAYOUT
OR X TO CHANGE FDC-ADDRESS

(A)  5,25", MFM, SS, 40 TRK
(B)  5,25", MFM, DS, 40 TRK
(C)  5,25", MFM, SS, 80 TRK
(D)  5,25", MFM, DS, 80 TRK
(E)     8", MFM, SS, 77 TRK
(F)     8", MFM, DS, 77 TRK
(G)     8" STANDARD CP/M
(H)  5,25", NON-DMA, SS
(I)  5,25", NON-DMA, DS
A
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                 SETTRACK (T) OR INSTALL (I) S
SOURCE DISK (OR CR TO COPY FROM MEMORY, STARTING AT 1000H) = A
DESTINATION DISK (OR X TO SKIP WRITING) = B
HIT CARRIAGE RETURN TO RETURN TO ZDOS
TO SAVE SELECTED VERSION TYPE: SAVE 15 SYSGEN.COM
```

Kopieren von nur Spur 2; SYSGEN ist richtig installiert:
-- gezählt wird bekanntlich ab Spur 0 --

```
0A>SYSGEN
SYSGEN VERS. 3.2 (26.10.82)
(TYPE X FOR EXIT)
(B)  5,25", MFM, DS, 40 TRK
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                SETTRACK (T) OR INSTALL (I) T
START COPYING AT SIDE : 0
STARTING TRACK : 2
STOP COPYING AT SIDE: 0
LAST TRACK : 2
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                SETTRACK (T) OR INSTALL (I) C
SOURCE DISK = A
DESTINATION DISK = B
COPYING TRACK 02
HIT CARRIAAGE RETURN TO RETURN TO ZDOS
TO SAVE SELECTED VERSION TYPE: SAVE 15 SYSGEN.COM
```

Ein gültiges Betriebssystem ist bereits im Speicher ab Adresse 1000H geladen und soll auf die Diskette **B** geschrieben S werden.

```
0A>SYSGEN
SYSGEN VERS. 3.2 (26.10.82)
(TYPE X FOR EXIT)
(B)  5,25", MFM, DS, 40 TRK
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                SETTRACK (T) OR INSTALL (I) S
SOURCE DISK (OR CR TO COPY FROM MEMORY, STARTING AT 1000H) =
DESTINATION DISK (OR X TO SKIP WRITING) = B
HIT CARRIAGE RETURN TO RETURN TO ZDOS
TO SAVE SELECTED VERSION TYPE: SAVE 15 SYSGEN.COM
0A>
```

Das Betriebssystem kann auf mehrere Methoden in den Speicher gelangen.

- mit der Prozedur GEN
- mit SYSGEN von einer Diskette
- Mit dem PSP mit dem R1000-Befehl
- Im KI durch LOAD ZDOS.COM $1000

Eine einseitige Mini-Diskette soll kopiert werden und anschließend geprüft werden. Also: Kopieren auswählen **C**, von **A** nach **B** und Rückkehr ins Betriebssytem **RETURN**.
Danach wieder **SYSGEN** aufrufen, Verify wählen V von A nach **B**.

```
0A)SYSGEN                                          (kopieren)
SYSGEN VERS. 3.2 (26.10.82)
(TYPE X FOR EXIT)
(A)  5,25", MFM, SS, 40 TRK
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                 SETTRACK (T) OR INSTALL (I) C
SOURCE DISK FOR COPY = A
DESTINATION DISK = B
COPYING TRACK 00     (zählt bis 39 hoch)
HIT ANY KEY TO RETURN TO ZDOS
TO SAVE SELECTED VERSION TYPE: SAVE 15 SYSGEN.COM
0A)SYSGEN                                       (vergleichen)
SYSGEN VERS. 3.2 (26.10.82)
(TYPE X FOR EXIT)
(A)  5,25", MFM, SS, 40 TRK
SELECT:  SYSGEN (S), COPY (C), VERIFY (V),
                 SETTRACK (T) OR INSTALL (I) V
SOURCE DISK = A
DESTINATION DISK = B
VERIFY TRACK 00     (zählt bis 39 hoch)
HIT ANY KEY TO RETURN TO ZDOS
TO SAVE SELECTED VERSION TYPE: SAVE 15 SYSGEN.COM
0A)
```

Unter ZDOS sind wie unter CP/M 16 Benutzer (USER) zugelassen. Das heißt nicht, daß ZDOS oder CP/M ein Multi-User-Betriebssystem ist. Die unterschiedlichen Benutzer können nie gleichzeitig arbeiten. Um einen gewissen Schutz zwischen ihnen zu gewährleisten, existieren die 16 User-Areas. In ZDOS sind Systemprogramme als Dateien des Benutzers 0 gespeichert. Diese sind allgemein benutzbar, können aber nur vom Benutzer 0 gelöscht oder beschrieben bzw. verändert werden. Der normale Anwender arbeitet als Benutzer 1 bis 15. Seine Dateien sind nur für ihn zugänglich. Als Schutz muß ein Password eingegeben werden. Hierzu dient das Programm USER:

```
0A)USER nummer password
```

Das Setzen der Benutzernummer "nummer" wird nur dann ausgeführt, wenn das richtige "password" miteingegeben wurde.

Beim Ausliefern des Systems existieren keine "passwords". Sie können beim Wechsel der Benutzernummer vereinbart werden. (Dazu muß die Systemdiskette beschreibbar sein; WRITE-PROTECT ist also vorher zu entfernen). Ein beim Wechsel in einen Benutzerbereich benutztes "password" gilt als gesetzt, und muß ab jetzt immer zum Wechsel benutzt werden.

Ein Beispiel:

Beginnend mit der Originaldiskette (keine "passwords") wird

```
0A)USER 2
```

eingebeben. Das System meldet sich jetzt mit:

```
2A)USER 4 otto
```

Ab jetzt ist als "password" für USER 4 "otto" vereinbart; d.h. wenn jetzt nach einem Kaltstart der Befehl:

```
0A)USER 4
```

eingegeben wird, erscheint folgende Fehlermeldung:

```
** WRONG PASSWORD **
0A)
```

Der Benutzerbereich wird nicht geändert. Es bleibt der Benutzerbereich 0 erhalten.

Der Befehl USER 0 ist immer möglich. Er wird nicht geschützt.

-- Achtung --

Ein einmal gesetztes password kann nicht wieder gelöscht oder geändert werden. Die einzige Möglichkeit besteht darin, das unveränderte Programm USER wieder neu von der ZDOS-Systemdiskette zu holen (ohne passwords)

Eine Prozedur ist eine Kette von Befehlen, die unter einem Namen auf der Diskette abgelegt ist. Sie kann durch **DO** name jederzeit aufgerufen werden. **DO** ist da Programm, das sie von der Diskette liest und für die Verarbeitung durch den KI vorbereitet. (vgl. Kap. 1.7)
Mit **ZDOS** werden 2 Prozeduren mitgeliefert:

Die Systemgenerierungsprozedur GEN.DO

Die Prozedur **GEN.DO** dient zur Systemgenerierung. Sie erstellt ein Betriebssystem und schreibt es auf die Diskette in Laufwerk **B.**

Aufruf:

```
0A)DO GEN version $adresse side
```

| | | | |
|---|---|---|---|
| mit: | version | = | ZDOSxx mit xx wie unten beschrieben |
| | adresse | = | Anfangsadresse des Kommandointerpreters (E000H maximal). |
| und: | side | = | S für single-sided-Laufwerke bei double-sided entfällt die Angabe S |

Erläuterung von xx:

| xx | | Diskettenversion | |
|---|---|---|---|
| 5N4 | 2 * 5" | NON-DMA | DD,DS 40 TRACK |
| 5N8 | 2 * 5" | NON-DMA | DD,DS 80 TRACK |
| 5N84 | 2 * 5" | NON-DMA | DD,DS 80/40 TRACK |
| 5D4 | 2 * 5" | DMA | DD,DS 40 TRACK |
| 5D8 | 2 * 5" | DMA | DD,DS 80 TRACK |
| 5D84 | 2 * 5" | DMA | DD,DS 80/40 TRACK |
| 5E4 | 2 * 5" | EPC | DD,DS 40 TRACK |
| 5E8 | 2 * 5" | EPC | DD,DS 80 TRACK |
| 5E84 | 2 * 5" | EPC | DD,DS 80/40 TRACK |
| 8 | 2 * 8" | DD,DS | |
| 8S | 1 * 8" | DD,DS + 1 * 8" | SD,SS |

Die normale Adresse fuer ZDOS ist $E000 (bei Optionen verschiebt sie sich nach vorne)! Wenn **ZDOS** ab Adresse 1000H in den Speicher geladen wird, gelten folgende Adressen:

| | |
|---|---|
| KI | 1000 bis 17FF |
| LEAS | 1800 bis 23FF |
| HEAS | 2400 bis 2DFF |

Achtung! Der HEAS-Datenbereich und die Interrupt-Tabelle werden nicht gespeichert. Geladen werden von **ZDOS** nur 7,5 KByte.

Es gibt keine reine Single-Density Version!

## Die Initialisierungsprozedur INIT.DO

**INIT.DO** ist die Prozedur, die bei jedem Kaltstart des Systems automatisch aufgerufen wird. Hierin können beliebige Programme aufgerufen werden; z.B. **CONFIG, IO** und auch andere Benutzerprogramme.

**INIT.DO** wird als leere Prozedur geliefert. Sie ist vom Benutzer an die jeweiligen Gegebenheiten anzupassen.

Einige Beispiele:

1. Serieller Drucker mit Handshake bei 9600 Baud:

```
)) START INIT ((
IO LST:=TTY:
CONFIG AAJXX
)) STOP INIT ((
```

2. Maschinensprachenentwicklungssystem, direkter Aufruf von **PSP:**

```
PSP                     Monitor Aufruf
XS                      SP auf 3000H setzen
3000
X                       Register anzeigen
```

Zur Ausführung der Prozedur **INIT.DO** sind auf der Diskette in Laufwerk **A** folgende Dateien erforderlich:

DO.COM
INIT.DO
und die in **INIT.DO** aufgerufenen Programme

Im **HEAS** ist die wahlweise mögliche **IOBYTE**-Funktion implementiert. Das **IOBYTE** gibt die Zuordnungen zwischen logischen und physikalischen Geräten an. Es betrifft die Konsole (CON), den Drucker (LST) und den Lochstreifenleser (RDR) und -stanzer (PUN). Als **IOBYTE** wird der Inhalt der Adresse **0003** interpretiert. Das **IOBYTE** besteht aus 4 Feldern zu je 2 Bit in folgender Reihenfolge:

| | LIST | | PUNCH | | READER | | CONSOLE | |
|---|---|---|---|---|---|---|---|---|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MSB | | | | | | | LSB |

Es gelten folgende Zuordnungen:

| | | |
|---|---|---|
| **CON:** | 00 | **TTY:** als Konsole |
| | 01 | **CRT:** als Konsole |
| | 10 | **BAT:** Batch Betriebsart, d.h. **RDR:** als Eingabegerät und **LST:** als Ausgabegerät |
| | 11 | **UC1:** als Konsole |
| **RDR:** | 00 | **TTY:** als Streifenleser |
| | 01 | **PTR:** als Streifenleser |
| | 10 | **UR1:** als Streifenleser |
| | 11 | **UR2:** als Streifenleser |
| **PUN:** | 00 | **TTY:** als Streifenstanzer |
| | 01 | **PTP:** als Streifenstanzer |
| | 10 | **UP1:** als Streifenstanzer |
| | 11 | **UP2:** als Streifenstanzer |
| **LST:** | 00 | **TTY:** als Drucker |
| | 01 | **CRT:** als Drucker |
| | 10 | **LPT:** als Drucker |
| | 11 | **UL1:** als Drucker |

hierbei ist
**TTY** eine Teletype
**CRT** ein Datensichtgerät
**PTR** ein schneller Streifenleser
**PTP** ein schneller Streifenstanzer
**LPT** ein Zeilendrucker

dagegen sind **UC1**, **UR1**, **UR2**, **UP1**, **UP2** und **UL1** vom Anwender jeweils zu implementierende Geräte, deren Namen festliegen.

Ein Beispiel:
Beim Kaltstart des Systems wird das **IOBYTE** = 81H gesetzt, d.h.

**LST:** = **LPT:** (10)
**PUN:** = **TTY:** (00)
**RDR:** = **TTY:** (00)
**CON:** = **CRT:** (01)

Die einfachste Möglichkeit, eine ZDOS-Diskette zu erstellen, ist eine vorhandene ZDOS-Systemdiskette der gleichen Version zu duplizieren. Damit werden aber teilweise zu viele Dateien übertragen. Deshalb sollten die überflüssigen (d.h. die in einem speziellen Anwendungsfall nicht unbedingt benötigten) Dateien anschließend gelöscht werden.

Folgende Dateien werden fast immer benötigt:

| | |
|---|---|
| SYSTEM.HLP | Hilfstexte des Systems |
| USER.COM | Benutzerbereich wechseln |
| DO.COM | Prozeduren starten |
| + die benötigten Prozeduren | ????????.DO |

empfohlen wird dann noch:

| | |
|---|---|
| FILES.COM | |
| CONFIG.COM | je nach System |
| IO.COM | je nach System |

Die restlichen Dateien hängen vom Anwendungsfall ab.

Ein Beispiel:

neue Diskette formatieren (FORMAT)
System von vorhandener Systemdiskette kopieren (SYSGEN S)
Dateien kopieren (entweder alle mit SYSGEN C oder nur bestimmte mit COPY b: a:dateiname)
anschließend neu erstellte Diskette in Laufwerk A prüfen (Kaltstart!)

Folgende Schritte sind zur Inbetriebnahme von **ZDOS** erforderlich:

1. Es ist sicherzustellen, daß die verwendete Hardware, also der Computer funktioniert.

   Benötigt wird zusätzlich zum Rechner ein Terminal mit V24-Schnittstelle, das an das Port **B** der SIO bzw. des DART angeschlossen werden muß. Seine Baudrate kann 1200, 2400, 4800 oder 9600 Baud betragen. Diese 4 Baudrateen werden vom Bootstraploader erkannt. Im BOOT werden die Handshake-Leitungen des Ports nicht beachtet.

2. Das mit **ZDOS** zusammen gelieferte EPROM vom Typ 2716 mit dem Aufkleber **BOOT641** oder einer höheren Versionsnummer ist auf Adresse 0 in den Computer einzusetzen.

3. Der Computer ist einzuschalten. Jetzt meldet sich nach Eingeben von mehreren Leerzeichen am Terminal der Urlader **BOOT641**; die Baudrate des Terminals wird angezeigt, nach Durchführung eines einfachen Speichertest wird die vorhandene Speichergröße in 16k Schritten angegeben und anschließend der Mode des Floppycontrollers bestimmt (DMA oder NON-DMA). Anhand dieser Angaben ist zuerst zu prüfen, ob die ausgelieferte Version von **ZDOS** auf diesem System lauffähig ist, d.h. die Adresse von **ZDOS** muß mindestens 2000H kleiner sein als die Speichergröße und bei der **NON-DMA**-Meldung darf kein **ZDOSxD** geliefert sein.
   Ist dieses sichergestellt, kann **ZDOS** prinzipiell auf diesem Computer laufen.

4. Jetzt wartet der Computer noch immer darauf, daß eine Diskette in Laufwerk **A** (das ist das Laufwerk mit der Laufwerksadresse 0) eingelegt wird. Unter der Voraussetzung, daß bereits sichergestellt wurde, daß keinesfalls auf eine schreibgeschützte Diskette geschrieben werden kann, und der Prüfung, daß die **ZDOS**-Diskette wirklich schreibgeschützt ist (bei 5,25" Disketten schützt der Aufkleber über dem Write-Protect-Loch, bei 8" Disketten darf kein Aufkleber über dem Write-Protect-Loch vorhanden sein), kann jetzt die **ZDOS**-Diskette in Laufwerk **A** eingelegt werden.

5. Nach Schließen der Laufwerkstür greift der Urlader auf die Diskette zu und lädt das Betriebssystem. Anschließend meldet sich **HEAS** mit seiner Versionsnummer. Nach weiteren Diskettenzugriffen meldet sich der **KI** mit **0A>**.

   Jetzt muß die evtl. vorhandene Boot-LED aus sein; d.h. das EPROM mit dem Urlader wird ab jetzt nicht mehr selektiert.

6. Jetzt ist prinzipiell sichergestellt, daß **ZDOS** auf diesem Computer lauffähig ist. Als erstes ist jetzt von der Original-**ZDOS**-Diskette eine Kopie zu machen. (Programm **FORMAT** und **SYSGEN**).

7. **Keinesfalls darf jemals auf die Original-ZDOS-Diskette geschrieben werden.** Zum Arbeiten ist immer eine Kopie der Originaldiskette zu verwenden.

8. Die jetzt erstellte Kopie von **ZDOS** sollte nun auf das jeweilige System angepaßt werden. Hierzu ist erforderlich:

   Generierung eines passenden **ZDOS** mittels der Prozedur **GEN.**

   Erstellung eines zu diesem **ZDOS** passenden **PSP.** Der **PSP** kann als höchste Adresse bei einem **$E000-ZDOS** maximal für die Adresse **D300** erstellt werden. Eingaben hierzu:
   **GENCOM PSP $D300**
   bzw. eine andere Adresse bei anderem **ZDOS**-Anfang.

   Installierung von **FORMAT** und **SYSGEN** und anschließendes **SAVE** der generierten Versionen auf der Diskette.

   Nun sollte das so erhaltene Betriebssystem ausgiebig getestet werden. Erst nach diesen Tests sollten Änderungen an **HEAS** erfolgen (falls erforderlich).

9. Beim Betrieb von **ZDOS** auftretende Fehler sollten so genau wie möglich spezifiziert werden und zusammen mit einen einfachen Beispiel auf dem mitgelieferten Formular an den Lieferanten eingeschickt werden. Bei wirklichen Fehlern wird Ihnen gegen den Preis einer Diskette nach Beseitigung des Fehlers eine neue Version zugeschickt. Dieses bezieht sich selbstverständlich nur auf Fehler, die durch ein standardmäßiges **ZDOS**-Betriebssystem verursacht werden. **Bitte bei Reklamationen immer die Originaldiskette mit Aufkleber einsenden.**

HEAS ist über eine definierte Sprungleiste aufrufbar. Diese Sprungleiste beginnt immer mit einer Seitengrenze. Es gibt zwei Möglichkeiten, ihre Lage im Speicher zu lokalisieren:

1. Die Speicherzellen 0001 und 0002 enthalten den Adressteil des Befehls: JP WBOOT. Sie zeigen also auf Sprungleiste + 3. Damit ergibt sich folgende Aufrufmöglichkeit:

```
        LD   HL,(1)
        LD   L,functionsnummer * 3
        LD   DE,RETADR
        PUSH DE
        JP   (HL)
RETADR:
```

2. Die Sprungleiste enthält 17 mal den Befehl JP adresse. Damit kann nach 17 mal C3 in Abständen von 3 Byte gesucht werden. Diese Methode ist sicherer, aber auch wesentlich aufwendiger. Die erste Lösung versagt nur bei speziellen Debuggern oder ähnlichen Programmen, die einen Warmstart resident überdauern.

HEAS-Sprungleiste

| | | | |
|---|---|---|---|
| 00 | JP | BOOT | Einsprung nach Kaltstart |
| 03 | JP | WBOT | Warmstart |
| 06 | JP | CST | Konsolzeichen eingegeben ? |
| 09 | JP | CIN | Konsoleingabe |
| 0C | JP | COUT | Konsolausgabe |
| 0F | JP | LIST | Druckerausgabe |
| 12 | JP | PNCH | Lochstreifenausgabe |
| 15 | JP | RDR | Lochstreifeneingabe |
| 18 | JP | HOME | Kopf auf Spur 0 |
| 1B | JP | SELDSK | Laufwerk selektieren |
| 1E | JP | SETTRK | Spur einstellen |
| 21 | JP | SETSEC | Sektor auswählen |
| 24 | JP | SETDMA | DMA-Puffer festlegen |
| 27 | JP | READ | Sektor lesen |
| 2A | JP | WRITE | Sektor schreiben |
| 2D | JP | LST | Drucker fertig ? |
| 30 | JP | SECTRA | Sektorverschränkung berechnen |
| 33 | JP | CBUF | Pufferadresse übergeben |
| 36 | JP | CONFIG | Konfiguration übergeben |
| 39 | 00 oder FF | | schnelle oder langsame Laufwerke |

HEAS-Aufrufe verändern im Wesentlichen alle Register. HEAS verfügt nicht wie LEAS über seinen eigenen Stack.

## Systemkaltstart

Beim Einschalten des Systems und bei jedem RESET führt das System einen Kaltstart aus. Hierbei kopiert der Bootstraploader das Betriebssystem (**KI**, **LEAS** und **HEAS**) von den Systemspuren der Diskette in Laufwerk A in den Arbeitsspeicher. Gleichzeitig werden alle implementierten Peripherieschnittstellen initialisiert. Der **KI** meldet sich auf der Konsole und erwartet die Eingabe eines Kommandos. Benutzt wird immer nach einem Kaltstart der Bereich des Benutzers 0. Ein anderer muß durch USER selektiert werden. Nur nach einem Kaltstart läuft die INIT-Prozedur ab. Sie ist durch Eingabe von ^S, ^C abbrechbar, falls dies erforderlich ist (z.B. der Speicherinhalt soll nicht verändert werden, da noch "gesaved" werden soll).

## Systemwarmstart

Ein Systemwarmstart wird entweder durch Eingabe von ^C oder als Ende eines Anwenderprogramms bei dem **LEAS**-Aufruf **SYSTEM RESET** oder bei Sprung auf Adresse 0000H ausgeführt. Hierbei werden nur **KI** und **LEAS** neu von der Diskette geladen. **KI** meldet sich wie bei einem Kaltstart, ändert aber weder das Betriebslaufwerk noch den Benutzer. Die CONSOLE-CHECK-Option wird nach jedem Warmstart wieder aktiviert. Ein Warmstart setzt alle Disketten zurück, d.h. anschließend sind alle Disketten wieder im R/W-Zustand, d.h. beschreibbar (unter ZDOS).

****** Ab Version 5.b1 löscht HEAS den jetzt ausge-
****** lagerten LEAS-Datenbereich bei jedem Warmstart.

LEAS-Aufrufe sind Befehle von Anwendungsprogrammen an das Betriebssystem. Damit kann dem Anwendungsprogrammierer die Arbeit stark vereinfacht werden. Er kann alle Funktionen des Disketten-Betriebssystems benutzen. LEAS-CALLS benutzen folgende formalisierte Übergabe:

Jeder Aufruf geht auf Adresse 0005. Dort wird von LEAS ein Sprung in LEAS eingetragen. Damit ist eine Unabhängigkeit von sowohl der LEAS-Version wie auch der Lage im Speicher erreicht. Die gewünschte Funktion wird LEAS vom Anwendungsprogramm in Register C mitgeteilt. Es gibt die Funktionen 0 bis 40 und 250 bis 255. LEAS stellt Unterprogramme für folgende Tätigkeiten zur Verfügung:

1. **Zeichen-Ein- und Ausgabe an Peripheriegeräte**

   - Konsoleingabe
   - Konsolausgabe
   - Lochstreifen einlesen
   - Lochstreifen stanzen
   - Ausgabe an den Drucker
   - direkte Konsol-E/A
   - Setzen von E/A-Zuordnungen
   - zeilenweise Ein- und Ausgabe
   - Abfrage des Konsolstatus

2. **Dateimanipulationen (Disketten-E/A)**

   - Rücksetzen des Disksystems
   - Laufwerksauswahl
   - Datei erstellen
   - Datei eröffnen
   - Datei schließen
   - Datei umbenennen
   - Inhaltsverzeichnis durchsuchen
   - sequentielles oder wahlfreies Lesen und Schreiben

3. **Druckaufrufe**

   - Datei auf den Drucker ausgeben
   - Druckausgabe abbrechen

4. **Systemwarmstart**

Benötigte Werte werden an LEAS in Register E oder DE übergeben. Antworten von LEAS erscheinen in A oder HL. Auf der nächsten Seite ist eine Funktionsübersicht aufgeführt. Im einzelnen sind die Funktionen im LEAS-Benutzer-Handbuch erklärt.

| 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 0FFH |
| 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 = Loginvect |
| 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<br>set: E = user | get: A = User<br>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

Achtung: Aus Kompatibilitätsgründen ist BA gleich HL, falls in HL Werte übergeben werden.

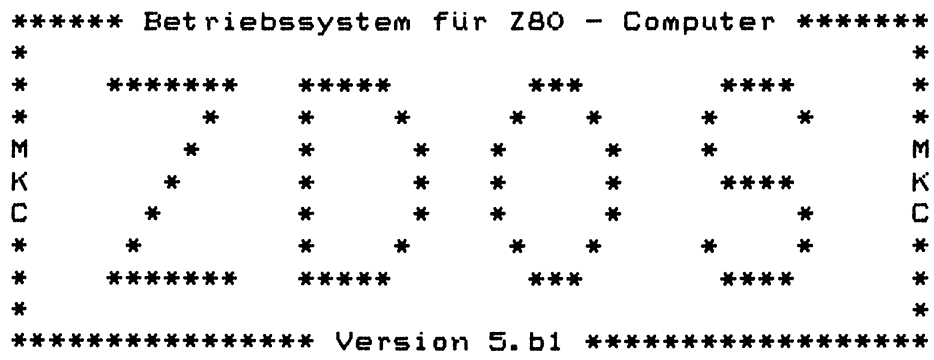
## PSP-Kommandos

| | |
|---|---|
| Ybyte1 byte2 ... | Byte-String im Speicher lokalisieren |
| Danfang ende | Speicherausdruck (DUMP) in hex und ASCII |
| Fanfang ende wert | Füllen des Speichers mit konstanten Werten |
| Gstart stop | Ausführung (GO) eines Maschinenprogramms |
| Lanfang ende | Disassemblierung (LIST) des Speicherinhaltes |
| Mvon bis nach | Verschieben (MOVE) von Speicherinhalten |
| Xregister | Register-Anzeige und Veränderung |
| Sadresse | Ersetzen (SUBSTITUTE) von Speicherinhalten |
| Tvon bis mode | Schrittweise Überwachung (TRACE) von Programmen |
| Vanfang1 anfang2 | Vergleichen (VERIFY) zweier Speicherbereiche |
| Itext | Eingabe von Dateinamen |
| Radresse | Datei einlesen |
| Wvon bis | Datei abspeichern |

## ASCII-Tabelle (deutsch)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | msd |
|---|---|---|---|---|---|---|---|---|---|
| 0 | NUL | DLE | SP | 0 | § | P | ` | p | |
| 1 | SOH | DC1 | ! | 1 | A | Q | a | q | |
| 2 | STX | DC2 | " | 2 | B | R | b | r | |
| 3 | ETX | DC3 | # | 3 | C | S | c | s | |
| 4 | EOT | DC4 | $ | 4 | D | T | d | t | |
| 5 | ENQ | NAK | % | 5 | E | U | e | u | |
| 6 | ACK | SYN | & | 6 | F | V | f | v | |
| 7 | BEL | ETB | ' | 7 | G | W | g | w | |
| 8 | BS | CAN | ( | 8 | H | X | h | x | |
| 9 | HT | EM | ) | 9 | I | Y | i | y | |
| A | LF | SUB | * | : | J | Z | j | z | |
| B | VT | ESC | + | ; | K | Ä | k | ä | |
| C | FF | FS | , | < | L | Ö | l | ö | |
| D | CR | GS | - | = | M | Ü | m | ü | |
| E | SO | RS | . | > | N | ^ | n | ß | |
| F | SI | US | / | ? | O | _ | o | DEL | |

lsd






## 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

OA>**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:<br>LD HL,(0001H)<br>LD L,0<br>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<br>**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 = 0BH
Ausgabe: A = Status der Konsole

Die Funktion GET CONSOLE STATUS überprüft, ob ein Zeichen von der Konsole eingegeben wurde. Eine 00H in Register A entspricht keiner Eingabe, ein FFH zeigt an, daß ein Zeichen eingegben wurde.

## Funktion 12: RETURN VERSION NUMBER

Eingabe: C = 0CH
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 = 0DH
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 00H 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 OOH. (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)

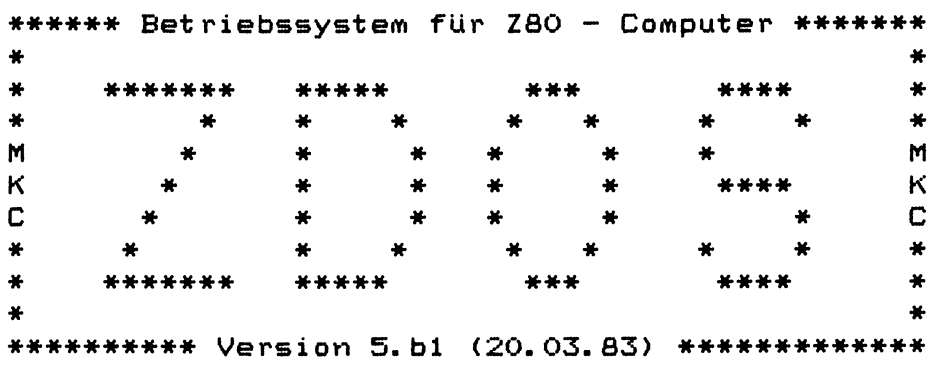
cr laufender Record, ist vor Lesen bzw. Schreiben vom Anwenderprogramm zu setzen (am Anfang auf 0), wird bei sequentiellem Lesen bzw. Schreiben automatisch weitergezählt.

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<br>set: E = user | get: A = User<br>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






## Inhaltsverzeichnis

**HEAS** ist der einzige Teil in **ZDOS**, der hardwareabhängig ist. In **HEAS** stehen alle Treiberroutinen für die Peripheriegeräte und die von **LEAS** benötigten Tabellen. Zusätzlich übernimmt **HEAS** die Zuordnung zwischen logischen und physikalischen Geräten.

**HEAS** besteht aus einzelnen Unterprogrammen, die alle über eine gemeinsame Sprungleiste erreicht werden können. Die **HEAS**-Routinen können von Programmen auf folgende Methode aufgerufen werden:

```
          LD    HL,(1)          ; Warmstart aus HEAS-Leiste
          LD    L,funktion      ; Nummer der Funktion * 3
          LD    DE,RETADR       ; diese Befehle ersetzen
          PUSH  DE              ; CALL  (HL)
          JP    (HL)
RETADR:
```

Hierbei wird aus der Nummer der gewünschten Funktion durch Multiplikation mit 3 die relative Adresse innerhalb der **HEAS**-Sprungleiste berechnet. Die Nummer erhält man durch Zählen der Einträge in der Sprungleiste. Als Beispiel hat die Funktion **CONIN** die Nummer 3 (vierter Eintrag, beginnend ab 0 zu zählen ergibt 3). Damit muß in L 3*3 = 9 geladen werden, um **CONIN** aufzurufen.

**Achtung!**

Nach **HEAS**-Aufrufen ist **nicht sichergestellt**, daß alle **LEAS**-Funktionen noch **einwandfrei** arbeiten. (Die **LEAS**-Funktionen, die auf dieselben **HEAS**-Routinen wie ein Anwenderprogramm zugreifen, können gestört werden; z.B. funktioniert die Tab-Auflösung der **LEAS**-Funktionen nach Benutzung von **HEAS-CONOUT** nicht mehr immer einwandfrei).

```
**********************************************************
**                                                      **
**    Die oben angeführte Methode ist die einzig        **
**    erlaubte Zugriffsversion auf HEAS-Routinen.       **
**    Jeder andere Zugriff verhindert die               **
**    Portabilität eines Programmes.                    **
**                                                      **
**********************************************************
```

Jetzt folgt eine Beschreibung der **HEAS**-Routinen

**HEAS** besteht aus der **Sprungleiste**, den in der Sprungleiste aufgerufenen **Routinen**, einem **Datenbereich** und den **Utility-Routinen.**

**HEAS** beginnt mit folgender Sprungleiste:

| | | |
|---|---|---|
| JP | **BOOT** | Einsprung nach Kaltstart |
| JP | **WBOT** | Warmstart |
| JP | **CST** | Konsolzeichen eingegeben ? |
| JP | **CIN** | Konsoleingabe |
| JP | **COUT** | Konsolausgabe |
| JP | **LIST** | Druckerausgabe |
| JP | **PNCH** | Lochstreifenausgabe |
| JP | **RDR** | Lochstreifeneingabe |
| JP | **HOME** | Kopf auf Spur 0 |
| JP | **SELDSK** | Laufwerk selektieren |
| JP | **SETTRK** | Spur einstellen |
| JP | **SETSEC** | Sektor auswählen |
| JP | **SETDMA** | DMA-Puffer festlegen |
| JP | **READ** | Sektor lesen |
| JP | **WRITE** | Sektor schreiben |
| JP | **LST** | Drucker fertig ? |
| JP | **SECTRA** | Sektorverschränkung berechnen |
| JP | **CBUF** | Pufferadresse übergeben |
| JP | **CONFIG** | Konfiguration übergeben |

**Achtung**

In **HEAS**-Routinen dürfen im wesentlichen nur die Register AF, BC, DE und HL benutzt werden. die Register IX und IY dürfen keinesfalls geändert werden, da sie in **LEAS** benötigt werden. Der zweite Registersatz wird von **ZDOS** generell nicht benutzt und steht daher dem Anwender uneingeschränkt zur Verfügung. Einzige Ausnahme sind einige Z80-CP/M-Programme, die wissen, daß CP/M nur die 8080-Register benutzt, und deshalb den zweiten Registersatz vor einem Systemaufruf nicht retten. Diese Programme sind aber sehr selten. Im normalen **HEAS** werden weder IX, IY noch der zweite Registersatz benutzt.

Nun folgt eine Beschreibung der in der **HEAS**-Sprungleiste aufgerufenen Rouinen in der Reihenfolge aus der Sprungleiste. **HEAS** hat keinen eigenen Stack; es benutzt den Stack des aufrufenden Programmes (also normalerweise den LEAS-Stack) mit. **HEAS** verändert im wesentlichen die Register **AF**, **BC**, **DE** und **HL**. Nicht alle HEAS-Routinen dürfen alle Register verändern. Dies ist bei den einzelnen Routinen angegeben.

**BOOT** Kaltstarteinsprung

Sprungleiste + 00H
Eingabe: A = Baudratencode
Ausgabe: C = 0 (Benutzer 0 und Betriebslaufwerk 0)
darf
ändern: AF, BC, DE, HL

**BOOT** ist die Routine, die vom Urlader **BOOT63** aufgerufen wird. **BOOT** initialisiert die Seite 0, setzt das Interrupt-System (falls erforderlich) und übergibt in Register C eine 0 für Laufwerk A. **BOOT** springt dann in den **KI**. (Adresse = Ladeadresse des **ZDOS**-Betriebssystems). Ferner hat **BOOT** die Aufgabe, alle I/O-Schnittstellen zu initialisieren; hierzu wird vom Urlader **BOOT63** in Register A der Wert, der in den CTC zu Baudratenerzeugung geschrieben wird, übergeben. **BOOT** trägt den vom Urlader erhaltenen Baudratencode in die Tabelle bei CINIT+10 und einen abgeleiteten Code bei CINIT+15 ein (für **CONFIG**).

Der Baudratencode ergibt sich aus folgender Aufstellung:

| Baudraten-code | CINIT+15 | Kommentar |
|---|---|---|
| 13 | 80 | 9600 Baud, 4 MHz |
| 26 | 81 | 4800 Baud, 4 MHz |
| 52 | 82 | 2400 Baud, 4 MHz |
| 104 | 83 | 1200 Baud, 4 MHz |
| 8 | 00 | 9600 Baud, 2.5 MHz |
| 16 | 01 | 4800 Baud, 2.5 MHz |
| 32 | 02 | 2400 Baud, 2.5 Mhz |
| 64 | 03 | 1200 Baud, 2.5 MHz |
| ? | 09 | eine andere Baudrate, angenommen wird 2.5 MHz |

WBOT Warmstartroutine

Sprungleiste + 03H

Eingabe: -
Ausgabe: C = Benutzernummer und Betriebslaufwerk (aus der Adresse 0004)
darf ändern: AF, BC, DE, HL

**WBOT** lädt das Betriebssystem (nur **KI** und **LEAS**) neu von der Diskette in Laufwerk A, setzt die Parameter der Seite 0 neu und übergibt beim Sprung in den **KI** in Register C die vor dem Warmstart als Betriebslaufwerk eingestellte Diskette (0 für A, 1 für B usw.)
Auf der Seite 0 sind zu setzen:

| | |
|---|---|
| 00H - 02H | JP WBOT (**KI**-Ladeadresse + 1403H) |
| 03H | Grundeinstellung für das **IOBYTE** |
| 05H - 07H | JP LEAS (**KI**-Ladeadresse + 806H) |

Ab der Version 5b1 (HEAS551) muß WBOT den LEAS-Datenbereich DATA löschen.

CST Konsolstatus

Sprungleiste + 06H

Eingabe: -
Ausgabe: A = FFH, falls Zeichen vorhanden
= 00H sonst
darf ändern: AF, BC

**CST** übergibt im Akku ein FFH, falls ein Zeichen von der Konsole eingegeben wurde, und ein 00H, falls kein Zeichen vorhanden ist.

CIN Konsoleingabe

Sprungleiste + 09H

Eingabe: -
Ausgabe: A = Zeichen
darf ändern: AF, BC

**CIN** liest das nächste Zeichen von der Konsole in den Akku, das höchste Bit muß gelöscht werden. **CIN** wartet, bis ein Zeichen eingegeben wird.

COUT Konsolausgabe

Sprungleiste + 0CH
Eingabe: C = Zeichen
Ausgabe: -
darf
ändern: AF, BC

COUT sendet das Zeichen aus Register C an die Konsole. Eventuell benötigte Warteschleifen nach CARRIAGE RETURN oder Schirm löschen sind hier zu implementieren.

LIST Druckerausgabe

Sprungleiste + 0FH
Eingabe: C = Zeichen
Ausgabe: -
darf
ändern: AF, BC

LIST sendet das Zeichen aus Register C an den logischen Drucker.

PNCH Lochstreifen stanzen

Sprungleiste + 12H
Eingabe: C = Zeichen
Ausgabe: -
darf
ändern: AF, BC

PNCH sendet das Zeichen aus Register C an den logischen Lochstreifenstanzer.

Diese Routine ist am Anfang nicht unbedingt erforderlich. Sie sollte erst eingefügt werden, wenn der Rest vollständig getestet ist.

RDR Lochstreifen lesen

Sprungleiste + 15H

Eingabe: -
Ausgabe: C = Zeichen
darf
ändern: AF, BC

**RDR** liest ein Zeichen vom Lochstreifenleser in den Akku, das höchste Bit wird gelöscht.

Diese Routine ist am Anfang nicht unbedingt erforderlich. Sie sollte erst eingefügt werden, wenn der Rest vollständig getestet ist.

HOME Laufwerk rücksetzen (Spur 0 suchen)

Sprungleiste + 18H

Eingabe: -
Ausgabe: -
darf
ändern: AF, BC, DE, HL

**HOME** positioniert den Kopf des selektierten Laufwerks auf Spur 00. Dies gilt solange, bis durch SETTRK eine andere Spur festgelegt wird.

SELDSK Laufwerk selektieren

Sprungleiste + 1BH

Eingabe: C = Laufwerksnummer
Ausgabe: HL = Adresse Disk-Header
= 0, falls Laufwerk nicht existent
darf
ändern: AF, BC, DE, HL

**SELDSK** selektiert ein Laufwerk. In Register C steht die Nummer des gewünschten Laufwerks (0 für A bis 7 für H). Bei Rückkehr wird im Register HL die Adresse des zugehörigen Disk-Headers bzw. eine 0, falls die Disk nicht existiert, übergeben. Dieses Laufwerk bleibt, bis durch einen erneuten Aufruf von SELDSK ein anderes Laufwerk bestimmt wird.

SETTRK — Spur für folgende Schreib- oder Leseoperationen festsetzen

Sprungleiste + 1EH
Eingabe: C = Spur (ab 0 gezählt)
Ausgabe: -
darf ändern: AF, BC, DE, HL

**SETTRK** positioniert den Kopf des selektierten Laufwerks auf die in Register C stehende Spur (gezählt wird ab Spur 0; die zweite Seite liegt hinter den Spuren der ersten Seite). **SETTRK** muß das SEEK-Kommando nicht unbedingt ausführen. Es reicht, wenn die Spurnummer gespeichert wird. Jede Ausführung kann der READ oder WRITE-Routine überlassen werden. Die hier festgelegte Spur gilt, bis durch SETTRK oder HOME eine andere Spur für ein Laufwerk bestimmt wird.

SETSEC — Sektor für folgende Schreib- oder Leseroutinen festsetzen

Sprungleiste + 21H
Eingabe: BC = Sektor
Ausgabe: -
darf ändern: AF, BC, DE, HL

**SETSEC** wählt den Sektor für nachfolgende Schreib- oder Leseoperationen aus. Register C enthält die Nummer des Sektors (gezählt wird ab Sektor 1). Die Sektornummer ist solange gültig, bis durch SETSEC eine neue festgelegt wird.

SETDMA — Pufferadresse für Schreib- oder Leseoperationen festlegen

Sprungleiste + 24H
Eingabe: BC = Schreib-/Lese-Adresse
Ausgabe: -
darf ändern: AF, BC, DE, HL

**SETDMA** legt den Speicherbereich für Schreib- oder Leseoperationen fest. Im Register BC wird die Anfangsadresse des DMA-Puffers übergeben. Seine Länge beträgt 128 Byte. Die Pufferadresse bleibt gleich, d.h. sie zählt nicht nach einer Operation um 128 Byte weiter. Die Pufferadresse kann nur durch einen SETDMA-Aufruf verändert werden.

READ Sektor lesen

Sprungleiste + 27H

Eingabe: -
Ausgabe: A = 0 falls erfolgreich
= 1 bei Fehler (ergibt: BAD SECTOR)
darf
ändern: AF, BC, DE, HL

**READ** liest den mit **SETSEC** gegebenen Sektor des selektierten Laufwerks von der eingestellten Spur in den DMA-Puffer. **READ** gibt im Akku eine 00H zurück, falls kein Fehler aufgetreten ist. **READ** sollte 10 Versuche zu lesen unternehmen, und erst wenn immer Fehler aufgetreten sind, eine 01H im Akku als Fehlermeldung übergeben.

**WRITE** Sektor schreiben

Sprungleiste + 2AH

Eingabe: C = Writetype (0, 1 oder 2)
Ausgabe: A = 0 falls erfolgreich
= 1 bei Fehler (ergibt: BAD SECTOR)
darf
ändern: AF, BC, DE, HL

**WRITE** beschreibt den vorher definierten Sektor mit den Daten aus dem DMA-Puffer. Auch hier sollten 10 Versuche durchgeführt werden, ehe im Akku eine 01H als Fehlermeldung zurückgegeben wird. Eine 00H im Akku bedeutet fehlerfreie Ausführung des **WRITE**-Befehls. Um mit anderen Sektorgrößen als 128 Byte effektiv arbeiten zu können, wird von LEAS beim Aufruf der **WRITE**-Routine der Writetyp mit übergeben. Es gilt folgender Code:

0 = normales Schreiben
1 = Schreiben ins Inhaltsverzeichnis
2 = Schreiben des ersten Sektors eines neuen Blocks.

Diese zusätzliche Information ermöglicht einen Zeitgewinn, da nicht mehr vor jedem Schreibzugriff in einen großen Sektor dieser gelesen werden muß (Typ 2 = neuer Block). Ferner kann das Inhaltsverzeichnis dadurch abgesichert sein, daß ein Schreibzugriff sofort durchgeführt wird, während normales Schreiben erst dann ausgeführt wird, wenn der Buffer benötigt wird. In einem HEAS mit einer von 128 Byte abweichenden Sektorgröße muß der Schreibtyp unter allen Umständen mit übergeben werden. Im Zweifelsfalle sollte ein Programm den Typ 1 übergeben. Dabei wird der Sektor vorher gelesen, dann geändert und sofort zurückgeschrieben. Dies ist zwar langsam, aber dafür sicher.

LST Druckerstatus

Sprungleiste + 2DH

Eingabe: -
Ausgabe: A = 0 falls Drucker nicht bereit
= FF falls Drucker bereit
darf
ändern: AF, BC

**LST** übergibt im Akku ein FFH, falls der Drucker bereit ist, ein Zeichen zu übernehmen; falls nicht, eine 00H.

SECTRA Sektorverschränkung berechnen

Sprungleiste + 30H

Eingabe: BC = logischer Sector (ab 0)
DE = Übersetzungstabelle für Sektorverschränkung
Ausgabe: HL = physikalischer (transformierter) Sektor (ab 1)
darf
ändern: AF, BC, DE, HL

**SECTRA** transformiert den logischen Sektor aus Register BC in den zugehörigen physikalischen Sektor und übergibt diesen im Register HL. Diese Sektorverschränkung erhöht die Geschwindigkeit bei aufeinanderfolgenden Diskzugriffen. Die Adresse der zu dem jeweiligen Laufwerk gehörenden Sektorverschränkungstabelle wird in Register DE von **LEAS** mit übergeben. (Achtung: bei nicht 128 Byte Sektorgröße ist **SECTRA** an dieser Stelle nicht empfehleswert.)
Ein Beispiel für die zur Transformation benutzte Tabelle, deren Adresse bei Aufruf von **SECTRA** in Register DE übergeben wird

```
XLT:    DEFB    1,7,13,19,25
        DEFB    5,11,17,23
        DEFB    3,9,15,21
        DEFB    2,8,14,20,26
        DEFB    6,12,18,24
        DEFB    4,10,16,22
```

Als Sektor wird an **SECTRA** 0 bis 25 übergeben. Durch Addition der Sektornummer zur Adresse XLT wird ein Zeiger auf die Sektornummer (jetzt physikalisch) gesetzt, der an das aufrufende Programm zurückgegeben werden soll. Durch Laden von Register L mit der Speicherzelle, auf die der soeben berechnete Zeiger hinweist, erhält man nach Löschen von Register H den physikalischen Sektor. Als Beispiel soll der Sektor 7 transformiert werden. Durch Addition von 7 zu XLT und anschließendes Laden des

Inhalts dieser soeben berechneten Position in der XLT-Tabelle erhält man eine 17. Dies ist der zu dem logischen Sektor 7 gehörende physikalische Sektor. **Logische Sektoren sind hier im Bereich von 0 bis 25; physikalische zwischen 1 und 26.**

CBUF Zeichen aus Buffer holen

Sprungleiste + 33H

Eingabe: -
Ausgabe: HL = Adresse des Zeichenpuffers
A = nächstes Zeichen aus dem Buffer
darf ändern: AF, HL

**CBUF** übergibt in Register A das nächste Zeichen aus dem Prozedurbuffer und in HL die Adresse des Buffers.
Die Pufferlänge des Prozedurpuffers beträgt 128 Bytes. Ein Programm, das den Puffer lesen will, muß selbst dafür sorgen, daß der Inhalt des Puffers nach jedem Lesen um 1 Byte nach vorne geschoben wird. Eine 0 im Puffer entspricht dem Ende einer logischen Zeile (Eingabe am Terminal: LINE FEED); zwei Nullen signalisieren das Ende der Eingaben. Der Rest des Puffers ist nicht definiert. Der Puffer wird unter **ZDOS** bei Aufruf der Funktion **READ CONSOLE BUFFER** und Ausnutzung der **Multi-Line**-Fähigkeit von **ZDOS** gefüllt.

CONFIG Adresse des Initialisierungstabelle übergeben

Sprungleiste + 36H

Eingabe: -
Ausgabe: HL = Adresse der Initialisierungstabelle
darf ändern: HL

**CONFIG** übergibt in HL die Adresse der Initialisierungstabelle.
In der Initialisierungtabelle stehen die Werte, die bei der Initialisierung in SIO und CTC der CPU-Karte geschrieben worden sind. (Die Write-Register der SIO sind nicht lesbar!)

In **HEAS** sind alle Tabellen zur Beschreibung der verwendeten Hardware enthalten. **LEAS** ist voll hardwareunabhängig (Ausnahme: **Z80**-CPU).

Die erste Gruppe der Beschreibungstabellen bezieht sich auf die Disketten-Laufwerke. Hierin werden die Parameter des Disk-Layouts angeben.

## 1. Disk-Parameter-Basis-Tabellen

Diese Tabelle muß einmal für jedes Laufwerk vorhanden sein. Ferner müssen alle Disk-Parameter-Basis-Tabellen in aufsteigender Reihenfolge der Laufwerke fest aneinanderhängend angeordnet sein. In der Disk-Parameter-Basis-Tabelle sind folgende Einträge enthalten:

| | |
|---|---|
| DPBT+ 0 | Adresse des Vektors, der die Sektorverschränkung enthält. |
| + 2 | ohne Bedeutung |
| + 4 | ohne Bedeutung |
| + 6 | ohne Bedeutung |
| + 8 | Adresse eines 128 Byte großen Puffers, der von **LEAS** für das Directory benötigt wird. Dieser Wert ist in allen Disk-Parameter-Basis-Tabellen gleich. |
| + A | Adresse des Disk-Parameter-Blocks. Im Disk-Parameter-Block stehen die Daten der jeweiligen Diskette. Gleiche Laufwerke benutzen denselben Disk-Parameter-Block |
| + C | Adresse eines Datenbereichs für **LEAS**, in dem der Prüfvektor für das Directory von **LEAS** abgelegt wird. Jede DPBT muß einen eigenen Vektor zur Verfügung stellen. |
| + E | Adresse des Belegungsvektors der jeweiligen Diskette. (Jede DPBT muß einen eigenen Belegungsvektor haben.) |
| +10 | Beginn der nächsten Disk-Parameter-Basis-Tabelle, bzw. Ende der letzten DPBT. |

Alle Werte sind 2 Byte lang; gespeichert sind sie in der Reihenfolge: lower und dann higher Byte.

## 2. Disk-Parameter-Block

Im Disk-Parameter-Block sind folgende Parameter des Laufwerks vorhanden:

DPB + 0 Sektoren / Spur (2 Byte lang)
Achtung: gemeint sind Sektoren zu 128 Byte!

+ 2 Block-Schiebe-Faktor (1 Byte Lang)
= 3 für 1K Blockgröße
= 4 für 2K Blockgröße
= 5 für 4K Blockgröße
= 6 für 8K Blockgröße
= 7 für 16K Blockgröße

+ 3 Block-Schiebe-Maske (1 Byte lang)
= 07H für 1K Blockgröße
= 0FH für 2K Blockgröße
= 1FH für 4K Blockgröße
= 3FH für 8K Blockgröße
= 7FH für 16K Blockgröße

+ 4 Extent-Maske (1 Byte lang)
= 0 für 1 Extent / Dir.-Eintrag
= 1 für 2 Extents / Dir.-Eintrag
= 3 für 4 Extents / Dir.-Eintrag
= 7 für 8 Extents / Dir.-Eintrag
= F für 16 Extents / Dir.-Eintrag

+ 5 Kapazität des Laufwerks in Blocks, gezählt wird ab 0! (Länge 2 Bytes)
**Achtung, es gibt nur ganze Blocks!**

+ 7 Anzahl der Einträge im Directory (2 Bytes lang)

+ 9 Bit-Maske für reservierte Directory-Blocks (2 Bytes lang)
Für jeden reservierten Block ist von vorne beginnend ein Bit auf 1 zu setzen. Beispiel:
2 reservierte Blocks ergeben folgendes:
(DPB+9) = 11000000B = 0C0H
(DPB+A) = 00000000B = 000H

+ B Länge des Prüfvektors; (2 Bytes lang) benötigt 1 Byte pro Sektor des Directories

+ D Anzahl der Systemspuren (2 Bytes lang); normal sind 2 Spuren

+ F Adresse des Sektor-Puffers für Spooling (für alle DPBs immer gleich). Die Länge des Sektorpuffers beträgt 128 Bytes. (2 Bytes lang)

+11 Adresse des File-Control-Blocks für Spooling (für alle DPBs immer gleich). Platzbedarf für den FCB: 33 Bytes.

Beispiel: 8" Diskette, 26 Sektoren, 1 K Blocks, 242 Blocks Kapazität, 64 Directory-Einträge, dafür zwei Blocks reserviert, damit 16 Sektoren für das Directory und 2 Systemspuren.

```
; DISK PARAMETER BLOCK ( FOR ALL DISKS )
DPB:        DEFW    26          ; SECTORS / TRACK
            DEFB    3,7,0
            DEFW    242,63
            DEFB    0C0H,0
            DEFW    10H,2
            DEFW    SBUF        ; SECTOR BUFFER
            DEFW    FCB         ; FILE CONTROL BLOCK
```

Im Folgenden werden die benötigten Puffer beschrieben:

1. Der Prüfvektor CSV für die Directories hat 1 Byte für jeden Sektor des Directories als Länge. Damit ergibt sich in diesem Bespiel eine Länge von 16 Bytes.

```
CSV0:       DEFW    0,0,0,0,0,0,0,0
CSV1:       DEFW    0,0,0,0,0,0,0,0
```

2. Der **Belegungsvektor ALV** muß beim Kaltstart des Systems gelöscht sein. Seine Länge berechnet sich folgendermaßen:
Für jeden Block Kapazität der Diskette muß ein Bit zur Verfügung gestellt werden. Das so erhaltene Ergebnis ist auf ganze Bytes aufzurunden. Für das Beispiel folgt:
242 Blocks ergeben 242 Bits Länge
Das ergibt 242/8 Byte = 30,25 Bytes
Durch Aufrunden ergeben sich 31 Bytes pro Diskette

```
ALV0:       DEFW  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
ALV1:       DEFW  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

3. Der Prozedurbuffer hat die Länge 128 Byte. Direkt davor müssen zwei Bytes stehen (**READ CONSOLE BUFFER**). Das erste dieser Bytes hat den Wert 07EH, der des zweiten ist ohne Belang. Der Prozedurbuffer bietet die Möglichkeit, beim Kaltstart des Systems automatisch ein Kommando ausführen zu lassen. Im folgenden Beispiel ist das der Aufruf der Prozedur **INIT**.

```
            DEFB    07EH,7
BUF:        DEFB    'DO INIT',0,0,0,0,0,0,0,0,0
            DEFS    70H
```

4. Für SPOOLING sind ein **FCB** und ein 128 Byte langer Sektorpuffer SBUF erforderlich.

```
FCB:        DEFB    0           ; DISK / EMPTY - FLAG
            DEFW    0,0,0,0,0,0,0,0
            DEFW    0,0,0,0,0,0,0,0

SBUF        DEFS    80H         ; SPOOL BUFFER
```

5. Zur richtigen Funktion des Programmes CONFIG ist die Initialisierungtabelle CINIT im HEAS enthalten. In ihr stehen:
1, CTCA-Mode, CTCA-Time-Constant, 1, SIOA-Write-Register 3, SIOA-Write-Register 4, SIOA-Write-Register 5, der Code für die Baudrate des Ports A,
2, CTCB-Mode, CTCB-Time-Constant, 2, SIOB-Write-Register 3, SIOB-Write-Register 4, SIOB-Write-Register 5 und der Code für die Baudrate des Ports B. Beispiel:

```
CINIT:      DEFB    01,5,8,01,0C1H,44H,6AH,03
            DEFB    02,4DH,64,02,0C1H,44H,6AH,09
```

Der Baudratencode ist:

0 für 9600 Baud
1 für 4800 Baud
2 für 2400 Baud
3 für 1200 Baud
4 für 600 Baud
5 für 300 Baud
6 für 150 Baud
7 für 110 Baud
8 für 75 Baud
9 für eine unbekannte Baudrate

Das höchste Bit des Baudratencodes unterscheidet zwischen 2,5 und 4 MHz-Systemen. Bit 7 = 1 heißt 2,5 und Bit 7 = 0 heißt 4 MHz.

**Achtung**

Bei Umgehung der Baudratensuchroutine des **BOOT641** wird in **HEAS** 2,5 MHz als Systemtakt angenommen; dies ist bei 4 MHz-Systemen zu ändern. Im Normalfall wird vom **BOOT641** eine Information über die Terminalbaudrate und den Systemtakt an **HEAS** übergeben. (der Wert CTCB-Time-Constant).

6. Ferner enthält HEAS noch einen 128 Byte großen Directorybuffer DIRB, der von LEAS bei Directory-Zugriffen benutzt wird.

```
DIRB:       DEFS    80H     ; DIRECTORY-BUFFER
```

7. ab Version 5b1 ist der LAES-Datenbereich nach HEAS ausgelagert. Benötigt werden ca. 180 Byte direkt anschließend an die Sprungleiste, d.h. ab Adresse HEAS+03AH. Es ist darauf zu achten, daß die Routine CONFIG noch in der selben Page beginnt. (Falls nicht wird WRONG-SYSTEM gemeldet!)

Beschreibung von HEAS55

HEAS55 ist der hardwareabhängige Teil von ZDOS 5.b. HEAS55 ist über Steuervariable einfach an unterschiedliche Hardware adaptierbar. HEAS55 benutzt folgende Diskettenformate:

5,25" MFM = Double Density A oder H
512 Byte / Sektor
10 Sektoren / Spur
40 Spuren / Seite
einseitige Aufzeichnung mit invertieren Daten

MFM = Double Density B, I, J
512 Byte / Sektor
10 Sektoren / Spur
40 Spuren / Seite
beidseitige Aufzeichnung mit invertieren Daten

MFM = Double Density C
512 Byte / Sektor
10 Sektoren / Spur
80 Spuren / Seite
einseitige Aufzeichnung mit invertieren Daten

MFM = Double Density D oder K
512 Byte / Sektor
10 Sektoren / Spur
80 Spuren / Seite
beidseitige Aufzeichnung mit invertieren Daten

8" MFM = Double Density E
512 Byte / Sektor
15 Sektoren / Spur
77 Spuren / Seite
einseitige Aufzeichnung mit invertieren Daten

MFM = Double Density F
512 Byte / Sektor
15 Sektoren / Spur
77 Spuren / Seite
beidseitige Aufzeichnung mit invertieren Daten

FM = Single Density G
128 Byte / Sektor
26 Sektoren / Spur
77 Spuren / Seite
einseitige Aufzeichnung mit invertieren Daten
Standard CP/M - Format zum Austausch

Die fett geschriebenen Kennbuchstaben sind die bei FORMAT und SYSGEN zu wählenden Formate. In HEAS ist dies so nicht möglich. Hier muß das gewünschte Format über die Steuervariablen gewählt werden.

Erläuterung der Steuervariablen:

Es kann immer nur eine der beiden Zeilen aktiv sein, die andere ist durch das vorgestellte Semikolon auszublenden. Zum Ändern ist also das Semikolon zu versetzen.

EPC wählt zwiscchen der EPC-Version und der Version für die Floppycontrollerkarten FDC5 bzw. FDC8/5 aus:

```
;=================================================
;       FDC OR EPC WANTED                         I
;=================================================
EPC     EQU     0               ; EPC
;EPC    EQU     1               ; FDC5 OR FC8/5
;=================================================
```

DMAMOD wählt zwischen DMA und NON-DMA-Betrieb des Floppy-Controllers; 8" Disketten müssen immer im DMA-Betrieb vereinbart werden, der EPC kann nur im NON-DMA-Betrieb arbeiten:

```
;=================================================
;       DEFINE MODE (DMA OR NON-DMA)              I
;=================================================
DMAMOD  EQU     0               ; DMA MODE WANTED
;DMAMOD EQU     1               ; NON-DMA-MODE WANTED
;=================================================
```

Es folgt jetzt die Entscheidung zwischen 5,25" und 8" Disketten:

```
;=================================================
;       5,25" OR 8" DISKS WANTED ?                I
;=================================================
MINI    EQU     0       ; 5,25" DISKS
;MINI   EQU     1       ; 8" DISKS
;=================================================
```

Und die Anzahl der normalen Disketten:

```
;=================================================
;       DEFINE NUMBER OF DD-DISKS                 I
;=================================================
NDISKS  DEFL    2       ; NUMBER OF DISKS (MAX: 4)
;=================================================
```

-- Achtung --

Hier zählen nur die normalen Disketten, d.h. z.B. die RAM-Disks zählen hier genauso nicht wie die Disk B im MIXED-System, die 40 Spur Disketten liest.

bei 5,25" Disketten die Entscheidung, ob 80 oder 40 Spur Laufwerke verwendet werden:

```
;=================================================
;       DEFINE 5,25" DD-DISKS                     I
;=================================================
;       40 OR 80 TRACKS / SIDE                    I
;=================================================
;MAXT   EQU     40      ; 40 TRACKS / SIDE
MAXT    EQU     80      ; 80 TRACKS / SIDE
;=================================================
```

bei 80 Spur Laufwerken (5,25") die Abfrage, ob Laufwerk B 40 Spur Disketten lesen und schreiben soll:

```
;=================================================
;       MIXED SYSTEM (A reads 80 track, B 40 track  I
;=================================================
MIX     EQU     0       ; MIXED WANTED
;MIX    EQU     1       ; NORMAL SYSTEM
;=================================================
```

Bei 8" Laufwerken kann mit SD das zweite 8" Laufwerk für Standard-CP/M-Disketten vereinbart werden. Dann liest und schreibt Laufwerk "A" nach wie vor ZDOS-Disketten, während Laufwerk "B" Standard-CP/M-Disketten lesen und schreiben kann (zum Austausch von Daten und Programmen)

-- S/D benötigt das File SD.MAC auf Disk "B" --

```
;=================================================
;       DEFINE 8" DD-DISKS                        I
;=================================================
;       SINGLE DENSITY DISK WANTED ?              I
;       INCLUDES FILE:  B:SD.MAC                  I
;=================================================
;SD     EQU     0               ; SINGLE DENSITY
SD      EQU     1               ; DOUBLE DENSITY
;=================================================
```

Die als Option lieferbare MBYTE-Version wird mit MBYTE eingeschaltet. Änderungen für die RAM-Floppy erfolgen in dem getrennten File MBYTE.MAC.

-- MBYTE benötigt das File MBYTE.MAC auf Disk "B" --

```
;=================================================
;       OPTIONAL MBYTE  ( RAM - FLOPPY )          I
;       INCLUDES FILE:  B:MBYTE.MAC               I
;=================================================
;MBYTE  EQU     0               ; MBYTE-OPTION
MBYTE   EQU     1               ; NORMAL
;=================================================
```

Das als Option lieferbare gemischte 5,25"/8" System wird mit dem WINCH-Schalter eingeschaltet. Dies ist gedacht für eine spätere Erweiterung durch ein Winchster-Laufwerk, wird aber vorläufig nur für das 58-System benutzt.

-- WINCH benötigt die Datei WINCH.MAC auf Disk "B" --

```
;=====================================================
;       OPTIONAL WINCHESTER - DISK (UNIT A)          I
;       INCLUDES FILE:  B:WINCH.MAC                  I
;=====================================================
;WINCH  EQU     0               ; WINCHESTER OPTION
WINCH   EQU     1               ; NORMAL
;=====================================================
```

Ferner interessant sind die Portadressen der I/O-Chips. Sie sind, soweit von **ZDOS** benutzt, angegeben:

```
;=====================================================
;        ***** I/O-PORTS *****
;=====================================================
CONB    EQU     0E3H    ; CONSOLE B STATUS
CONA    EQU     0E1H    ; CONSOLE A STATUS
                        IFF     EPC
FDC     EQU     0F2H    ; EPC ONLY
TIMER   EQU     0E9H    ; MOTO-TIMER FOR EPC
RBOOT   EQU     0ECH    ; SWITCH OFF EPROM
                        ELSE
FDC     EQU     0       ; FLOPPY CONTROLL
                        ENDIF
FDD     EQU     FDC+1   ; FLOPPY DATA PORT
FTC     EQU     FDC+2   ; FLOPPY TERMINATE
DMA     EQU     FDC+3   ; FLOPPY-DMA-PORT
CTC     EQU     0E8H    ; CLOCK FOR CONA
PIOA    EQU     0E4H    ; PRINTER CONTROL
PIOB    EQU     0E6H    ; PRINTER DATA
```

Jetzt folgen die Wartezeiten für die Peripheriegeräte, nach denen von **HEAS55** eine Fehlermeldung ausgegeben wird.

```
PTIME   EQU     16      ; TIME-OUT (CENTRONICS)
STIME   EQU     16      ; TIME-OUT (V24)
```

An externen Adressen ist die Systemadresse = Anfangsadresse des KI = Ladeadresse anzugeben. Dies ist die Adresse, die auch bei GENCOM für alle Systemteile anzugeben ist.

```
;
;        ***** EXTERNALS *****
;
KI      EQU     0E000H
```

In HEAS55 ist die IOBYTE-Funktion implementiert. Hier sind die "Geräte" UC1, PTR, PTP, UP2 und UL1 noch frei. Zum Einfügen eigener Routinen sind die Zeilen der ASSIGN - TABLE:

```
;
;       **** ASSIGN - TABLES
;
TCIN:   DEFW    CAIN,CBIN,RDR,CBIN       ; TTY,CRT,BAT,UC1
TCOUT:  DEFW    CAOUT,CBOUT,LIST,CBOUT   ; TTY,CRT,BAT,UC1
TCST:   DEFW    CASTI,CBSTI,BATST,CBSTI  ; TTY,CRT,BAT,UC1
TRDR:   DEFW    CAIN,DUMY,CBIN,CBIN8     ; TTY,PTR,UR1,UR2
TPNCH:  DEFW    CAOUT,NO,CBOUT,NO        ; TTY,PTP,UP1,UP2
TLIST:  DEFW    CAOUT,CBOUT,PIO8,NO      ; TTY,CRT,LPT,UL1
TLST:   DEFW    CASTO,CBSTO,PIOST,BATST  ; TTY,CRT,LPT,UL1
```

zu ändern und die entsprechenden eigenen Routinen einzufügen.

| | | |
|---|---|---|
| Hierbei bedeutet: | TCIN | CONSOLE INPUT |
| | TCOUT | CONSOLE OUTPUT |
| | TCST | CONSOLE STATUS |
| | TRDR | READER INPUT |
| | TPNCH | PUNCH OUTPUT |
| | TLIST | LIST OUTPUT |
| | TLST | LIST STATUS |
| und | CAIN | SIO A IN (7 Bit) |
| | CASTI | SIO A Receiver Status |
| | CAOUT | SIO A OUT (8 Bit) |
| | CASTO | SIO A Transmitter Status |
| | CBIN8 | SIO B IN (8 Bit) |
| | PIO8 | Centonics Out (8 bit) |
| | PIOST | Centronics Status |

## Beschreibung des Urladers

Der Urlader **BOOT65** wird in einem EPROM 2716 geliefert. Er gehört zum Lieferumfang von **ZDOS**. Das EPROM ist z.B. als IC 3 auf der CPU-Karte HKM-Z-105x einzusetzen. In einem Standardsystem ist der Urlader direkt lauffähig. Für abweichende Hardware sind im **BOOT641** Patch-Areas vorgesehen, um den Urlader adaptieren zu können.

Für den Einplatinencomputer und die MKC CPU II existieren spezielle Urlader. (Die Beschreibung stimmt überein; einige Adressen und Routinen sind verändert.)

**BOOT65** bestimmt zuerst die Baudrate des an SIO Port B angeschlossenen Terminals. (kein Handshake!) Der Reihe nach wird auf 9600, 1200, 4800 und 2400 Baud verglichen. Hierzu ist am Terminal maximal 8 mal die Leertaste zu betätigen. Anschließend erscheint diese oder eine ähnliche Meldung:

**BOOTSTRAPLOADER VERS. 6.5**
**COPYRIGHT (C) 1981, 82 BY H.K.M.**
**???? BAUD,**

mit ???? = gefundene Baudrate des Terminals.
Jetzt folgt ein einfacher nicht zerstörender Speichertest. Ausgegeben wird der verfügbare RAM-Speicher in 16K-Seiten. Falls ein Speicherfehler gefunden wurde, wird die Seite, in der der Fehler auftrat, ausgegeben. Geprüft wird der Speicher ab Adresse 2000H oder 8000H (EPC-I). 48K MEMORY FAILED bedeutet, daß zwischen 32 und 48K ein Fehler gefunden wurde.
Jetzt wird die Betriebsart des Floppy-Controllers festgestellt. (Vorrang hat der DMA-Mode)
Meldung: DMA-MODE oder NON-DMA-MODE
Anschließend wird **ZDOS** von den ersten Spuren der Diskette in Laufwerk A an das obere Ende des lückenlosen Speichers geladen und anhand der **HEAS**-Sprungleiste eine evtl. notwendige Verschiebung berechnet und ausgeführt.
**BOOT65** kann nur ein gültiges **ZDOS** laden, bei anderem Inhalt erfolgt die Meldung **NO SYSTEM. BOOT641** wartet auf die Diskette, d.h. es wird **kein** DISK NOT READY gemeldet.

## Beschreibung der Patch-Area im **BOOT65**
(vgl. Listing im Anhang)

Prinzipiell sind am Anfang des **BOOT65** ab Adresse 0020H 9 mal 3 Bytes leer (Inhalt OFFH). Hier können Sprünge zu eigenen Routinen des Users einprogrammiert werden. Für diese Routinen steht kein Stack zus Verfügung. Die Rückkehradresse wird deshalb in Register IX übergeben.

| | | |
|---|---|---|
| 0020H | USER0 | wird vor Ausführung von BOOT65 angesprungen. z.B. Speicher einschalten |
| 0023H | USER1 | Initialisierung der User-Konsole, umgeht die Baudratenbestimmung |
| 0026H | USER2 | User-Send-Message-Routine |
| 0029H | USER3 | Festlegung der höchsten verfügbaren Adresse im Speicher, umgeht den Speichertest |
| 002CH | USER4 | DMA oder NON-DMA-Mode festlegen |
| 002FH | USER5 | Disk-Timing festlegen (nur für NON-DMA-Mode) |
| 0032H | USER6 | Disk-Timing festlegen (nur für DMA-Mode) |
| 0035H | USER7 | wird am Ende von **BOOT65** vor dem Sprung in **HEAS** ausgeführt |
| 0038H, | USER8 | User-Konsoleingaberoutine |

Die Adresse, ab der Benutzer-Routinen in das **BOOT63**-EPROM einprogrammiert werden dürfen, ist dem Listing im Anhang zu entnehmen. (Sie liegt ca. bei 6FEH)

## Achtung

Bei Umgehung der Baudratenbestimmung fehlt dem System die Kenntnis des Systemtaktes. Dies ist in **HEAS** zu beachten. Das normale **HEAS** nimmt jetzt 2,5 MHz Systemtakt an. Bei 4 MHz Takt ist dies in **HEAS** entsprechend zu ändern. Eine andere Möglichkeit besteht darin, den Urlader die Zeitkonstante, die in den CTC geschrieben werden müsste, übergeben zu lassen (z.B. 0DH für 9600 Baud bei 4 MHz Systemtakt).

## Adressen im System (Hardware)

********** I/O-Map **********

| Adresse | Baustein | Funktion | |
|---|---|---|---|
| 00 | uPD 765 | Floppy-Status-Port | |
| 01 | uPD 765 | Floppy-Daten-Port | |
| 02 | uPD 765 | Floppy-Terminate | |
| 03 | Z80 DMA | Floppy-DMA-Port | |
| E0 | Z80 SIO | Data SIO A | TTY |
| E1 | Z80 SIO | Command SIO A | |
| E2 | Z80 SIO | Data SIO B | CRT |
| E3 | Z80 SIO | Command SIO B | |
| E4 | Z80 PIO | PIO Data A | LPT |
| E5 | Z80 PIO | PIO Command A | |
| E6 | Z80 PIO | PIO Data B | |
| E7 | Z80 PIO | PIO Command B | |
| E8 | Z80 CTC | Kanal 0 | Baudrate für SIO A |
| E9 | Z80 CTC | Kanal 1 | |
| EA | Z80 CTC | Kanal 2 | Baudrate für SIO B |
| EB | Z80 CTC | Kanal 3 | |

Die MKC CPU-II und die MKC EPC-I Rechnerplatinen haben folgende abweichende Adressbelegung:

MKC EPC-I:

| | | |
|---|---|---|
| F0 | uPD 765 | Floppy-Status-Port |
| F2 | uPD 765 | Floppy-Status-Port |
| F1 | uPD 765 | Floppy-Daten-Port |
| F3 | uPD 765 | Floppy-Daten-Port |
| F4..F7 | uPD 765 | Floppy-Terminate |
| EC..EF | | Boot-Reset |

MKC CPU-II:

| | |
|---|---|
| EC..EF | Paging-Registerfile |

Alle anderen I/O-Adressen sind nicht belegt.

## I/O-Initialisierung

### SIO-Initialisierung

Beide Kanäle der SIO sind initialisiert für:
asynchrone Übrtragung
8 Bit, 1 Stop-Bit, kein Parity, kein Handshake,
* 16 Clock-Mode, DTR aus, RTS aktiv
Damit folgt:
WR1 = 0
WR2 = 0
WR3 = 0C1H
WR4 = 044H
WR5 = 06AH
WR6 nicht geladen
WR7 nicht geladen

### CTC-Initialisierung

Die Kanäle 0 und 2 des CTC werden im Counter-Mode betrieben. (Mode-Control-Byte = 04DH), die Zeitkonstanten sind für Kanal 0 (9600 Baud) entweder 8 oder 13 bei 2,5 oder 4 MHz; und für Kanal 2 entsprechend der Terminal-Baudrate eingestellt.

### PIO-Initialisierung

Die PIO ist initialisiert als CENTRONICS-Druckerschnittstelle. Kanal A ist das Status/Control-Port, Kanal B das Datenausgabeport. Folgende Belegung der PIO ist auf allen H.K.M. und MKC Karten einheitlich vorgesehen:

| | | |
|---|---|---|
| PIO Port A (Steuer-Port) | Bit 0 | = Selected (Input) |
| | Bit 1 | = Busy (Input) |
| | Bit 2 | = Paper end (Input) |
| | Bit 3 | = Error not (Input) |
| | Bit 4 | = Strobe not (Output) |
| | Bit 5 | = Select not (Output) |
| | Bit 6 | = Autolf not (Output) |
| | Bit 7 | = Init not (Output) |
| PIO Port B | Bit 0 bis 7 | = Daten (Output) |

Initialisierung:

| | | |
|---|---|---|
| Port B | OFH | Output Mode 0 |
| Port A | CFH | Control Mode 3 mit |
| | OFH | Bit 4..7 = out |
| | | Bit 0..3 = in |

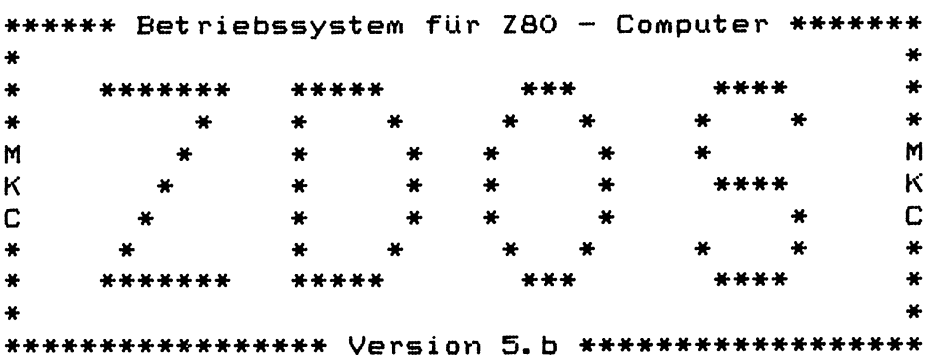
********** Memory-Map **********

| | |
|---|---|
| 0000 ... 0002 | JP Warmstart |
| 0003 | **IOBYTE** |
| 0004 | Betriebslaufwerk des **KI** |
| 0005 ... 0008 | JP **LEAS** (**LEAS**-Service-Call) |
| 005C ... 007F | File-Control-Block |
| 0080 ... 00FF | **DMA**-Puffer |
| 0100 ... E7FF | Programmbereich |
| | |
| D300 ... DFFF | **PSP**, falls geladen |
| E000 ... E7FF | **KI** |
| E800 ... F3FF | **LEAS** |
| E806 | **LEAS**-Einsprungstelle |
| F400 ... FEFF | **HEAS** |
| F400 | **HEAS**-Sprungleiste |
| FF00 ... FFFF | Vector-Interrupt-Page |
| FF00 ... FF7F | reserviert für **HEAS** |
| FF80 ... FFFF | frei für Anwenderprogramme |

Die Adressen, die hier ab D300 angegeben sind, sind abhängig von der Ladeadresse des Betriebssystems. Sie verschieben sich dementsprechend, wenn das Betriebssystem für eine tiefere Adresse generiert wird.

## Achtung

Die Lage der Vector-Interrupt-Page ist nicht absolut festgelegt. Sie befindet sich immer am oberen Ende des verfügbaren Schreib/Lesespeichers. Damit kann ihre Lage nur durch Auslesen des I-registers der Z80-CPU bestimmt werden. Die Vector-Interrupt-Page beginnt immer an einer Seitengrenze. Eine **ZDOS**-DMA-Version verlangt einen Vector-Interrupt, d.h. kein Programm darf den Interrupt "disablen". Ferner darf das I-Register nicht verändert werden. Die **ZDOS**-NON-DMA-Versionen schalten den Interrupt bei Floppyzugriffen aus, versetzen ihn aber anschließend wieder in den Ursprungszustand.






I n h a l t s v e r z e i c h n i s

## 1. -- Allgemeine Beschreibung --

MBYTE ist eine Option zu ZDOS. MBYTE emuliert eine Diskette im Hauptspeicher. Hierzu ist eine Erweiterung des Adressraumes erforderlich. Die MBYTE-Option basiert auf der TIJ 192K ADDRESS EXTENSION CARD oder der M.K.C. CPU II. Sie ist für 192 oder 256 K Byte RAM übersetzt; andere Speichergrößen sind durch Ändern von RAM in MBYTE.MAC und anschließendes neu assemblieren und linken leicht möglich. Die 192 oder 256 K Byte RAM ergeben eine Disk "C" mit 124 oder 190 K Byte Kapazität. Maximal möglich sind zwei RAM-Disketten mit je bis zu 480 K Byte RAM ("C" und "D"). Hierzu ist der maximale Speicherausbau des Systems auf 1 M Byte Speicher erforderlich. Diese Disketten können natürlich nicht formatiert werden; sie erzeugen allerdings auch keine Fehler. Der Inhalt einer RAM-Diskette geht beim Ausschalten des Computers verloren, si muß daher nach dem Einschalten vor dem ersten DIR - Zugriff mittels ERASE gelöscht werden. Wird dies vergessen, kann das System irgendwann später abstürzen. Die gewünschte der beiden Versionen kann durch die Steuervariable CPUII in MBYTE.MAC ausgewählt werden.

## 2. -- Hinweise zur Installation --

Geliefert werden die Dateien MBYTE.MAC, HEAS?X.LOD und HEAS?X.COM. HEAS.MAC ist bereits als Teil des Systems mit ZDOS geliefert worden. Falls bei MBYTE ein HEAS dabei ist, handelt es sich um eine neuere Version, die evtl. für MBYTE erforderlich ist. Wie in der ZDOS-Dokumentation beschrieben, ist HEAS durch Änderung von Steuervariablen an unterschiedliche (vorgesehene) Hardware leicht adaptierbar. Eine MBYTE-Version kann durch folgende Modifikation in HEAS erzeugt werden:
HEAS ist zu editieren; dabei muß das Semikolon vor der Zeile:

```
MBYTE   EQU     0
```

entfernt werden und dafür die Zeile:

```
MBYTE   EQU     1
```

durch Einfügen eines Semikolon ausgeschaltet werden. Anschließend ist HEAS neu zu assemblieren und neu zu linken. MBYTE.MAC wird hierbei durch den M80 automatisch von Laufwerk "B" dazu eingelesen. Benötigt werden hierzu:

- ein Editor
- der MACRO80 als Assembler (MICROSOFT)
- der LINK80 als Bindelader (MICROSOFT)

Das so erhaltene HEAS?X.COM ist jetzt über das Betriebssystem ZDOS zu patchen. Hierfür wird ZDOS.COM ab 1000H in den Speicher geladen und danach HEAS?X.COM ab 2400H darüber in den Speicher geladen. Anschließend kann mit SYSGEN das so erstellte System auf die Systemspuren einer **Testdiskette** kopiert werden.

-- Achtung --

HEAS muß für die richtige Adresse assembliert werden. In HEAS ist hierfür die Zeile

```
KI        EQU       OE000H
```

an die entsprechende Systemadresse anzupassen. Die KI-adresse ist gleichzeitig die Adresse, die bei **GENCOM** angegeben werden muß.

Einfacher ist die Verwendung des mitgelieferten **HEAS?X.LOD**-Files. Dieses ist über das entsprechende **ZDOS.LOD**-File zu kopieren. Dann kann durch **DO GEN ...** wie bisher ein Betriebssystem erstellt werden. Die anfangs erwähnte Methode muß nur bei Änderungen des **HEAS** oder **MBYTE** benutzt werden.

## 3. -- Überkopieren von HEASxx.LOD --

Folgende Schritte sind erforderlich:

1. Laden von ZDOSxx.LOD

   ```
   0A>LOAD ZDOSxx.LOD $100
   ```

2. Laden von HEASxx.LOD

   ```
   0A>LOAD HEASxx.LOD $1780
   ```

3. Speichern des so modifizierten ZDOS.LOD

   ```
   0A>SAVE 34 ZDOSxx.LOD $100
   ```

Jetzt sollte das so neu erstellte ZDOS getestet werden:

4. System generieren:

   ```
   0A>DO GEN ZDOSxx $adresse
   ```

5. Versuchen die Diskette aus Laufwerk "B" in Laufwerk "A" zu booten. (Kaltstart)

6. RAM-Diskette löschen

   ```
   0A>ERASE C
   ```

7. Inhaltsverzeichnis von "C" (sollte leer sein!)

   ```
   0A>DIR C:
   ```

8. File auf "C" kopieren, wieder Inhaltsverzeichnis aufrufen (jetzt muß ein File erscheinen) und dann kopiertes File (Programm) von "C" ausführen

## 4. -- Hardwarevoraussetzungen --

MBYTE setzt eine der beiden folgenden Rechnerkonfigurationen voraus:

1. basierend auf der RAM-Extension-Card

   - H.K.M. CPU - Karte
   - H.K.M. FDC 5 oder FDC 8/5 - Floppycontroller
   - T I J 192K RAM/ADDRESS EXTENSION CARD

2. basierend auf der M.K.C. CPU II

   - M.K.C. CPU II - Karte
   - H.K.M. FDC 8/5 - Floppycontroller
   - und mindestens eine weitere RAM-Karte (256 KByte)

optionell sind für beide Konfigurationen weitere Speicherkarten verwendbar:

- T I J 64/256K dyn. RAM-Karte
- H.K.M. CMOS RAM - Karte

Der Speicher ist durchgehend ab 00000H zu bestücken, RAM gibt die Obergrenze (= dem ersten Loch) des Speichers an. Die CPU arbeitet hierbei immer in Page 0, d.h. 00000H bis 0FFFFH, die Pages 1 bis x sind dann als RAM-Diskette verwendbar (10000H bis x0000H). Eine eventuelle memory-mapped-Video-Karte ist am oberen Ende des Speicherraumes unterzubringen (z.B. F0000H bis FFFFFH).

-- Achtung --

alle Speicherkarten müssen 20 Bit Adressen vollständig dekodieren. Eine unvollständige Dekodierung führt zu einer Katastrophe, da dann eine Speicherzelle in mehreren Dateien benutzt wird.

## 5. -- Systemanforderungen --

Die T I J 192K - Karte setzt ein DMA-fähiges System voraus. Sie ist in der DMA-Daisy-Chain hinter die FDC 8/5 - Karte zu stecken; d.h. der (bzw.) die Floppycontroller hat (haben) höhere Priorität. Die M.K.C. CPU II arbeitet nicht im DMA, sie muß zum Arbeiten in der RAM-Diskette den Interrupt disablen (selbstverständlich wird er nach Beendigung eines Zugriffs wieder in den vorherigen Zustand (EI oder DI) zurückgesetzt).

## 6. -- benötigte Software --

MBYTE setzt an Software zum Betrieb lediglich **ZDOS** voraus. Zum Erstellen bzw. Verändern sind zusätzlich ein Editor, der Assembler **MACRO80** und der Linker **LINK** erforderlich.

Aufruf:

```
0A>M80 =B:HEASxx
0A>L80 B:HEASxx/P:100,B:HEASxx/N/E
```

Hierzu muß sowohl **HEASxx.MAC** als auch **MBYTE.MAC** auf deß Diskette in Laufwerk "B" sein.

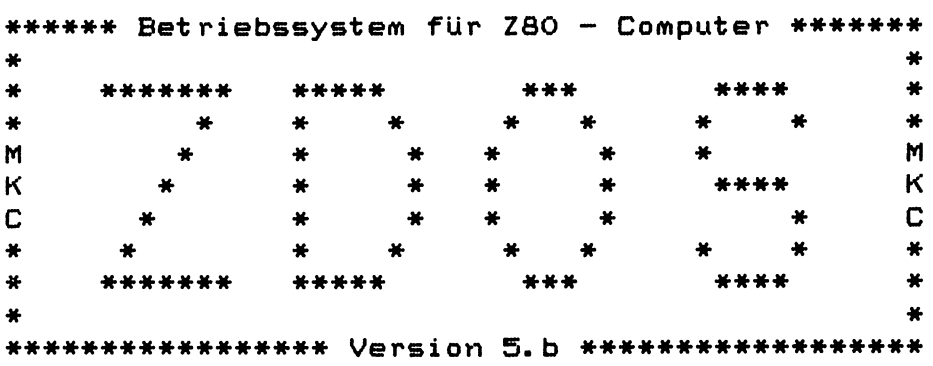
## 7. -- Dienstprogramm ERASE --

Das ZDOS-Dienstprogramm **ERASE** ist erforderlich, um RAM-Disketten vor der Benutzung in einen brauchbaren Zustand zu bringen. **ZDOS** setzt ein von ihm erzeugtes Directory voraus (oder eine neu formatierte Diskette). Alles andere führt zu evtl. schwerwiegenden Systemfehlern. **ERASE disk** "formatiert" das Directory der Diskette "disk"; d.h. die Sektoren des Inhaltsverzeichnisses werden mit 0E5H gefüllt. **ERASE** löscht nicht nur RAM-Disketten, sondern auch normale Disketten; es ist also darauf zu achten, das Laufwerk anzugeben. ERASE "formatiert" das Inhaltsverzeichnis einer Diskette.

Aufruf: `0A>ERASE C`

### Fehlermeldungen

| | |
|---|---|
| SELECT ERROR | Die Laufwerksangabe fehlte oder war falsch |
| WRITE ERROR | Das Laufwerk kann nicht beschrieben (gelöscht) werden. -> Hardware-fehler oder Schreibschutz |






I n h a l t s v e r z e i c h n i s

## 1. -- allgemeine Beschreibung --

5/8 ist eine Option zu ZDOS, die es ermöglicht, sowohl 5,25" als auch 8" Floppys zu verarbeiten. Benötigt werden hierzu 2 Floppycontroller H.K.M. FDC 8/5; einer für 5,25" und einer für 8" Laufwerke. Gebootet wird von 5,25" Disketten.

Im Normalfall ergibt sich also folgende Aufteilung der Laufwerke:

| | | |
|---|---|---|
| A | 5,25" | Double-Density-Laufwerk |
| B | 5,25" | Double-Density-Laufwerk |
| C | 8" | Double-Density-Laufwerk |
| D | 8" | Single-Density-Laufwerk |

Dieses Mix-System ist 200H Byte länger als ein normales ZDOS; es beginnt daher nicht auf Adresse E000H, sondern schon auf Adresse DE00H. Dies ist z.B. bei Generierung eines PSP mittels GENCOM zu beachten. Anstelle des normalen:
0A)GENCOM PSP $D300 muß hier: 0A)GENCOM PSP $D100
angegeben werden.

Dieses Mix-System ist hauptsächlich zum Kopieren von Software zwischen unterschiedlichen Disketten gedacht. Es ermöglicht es, Software von Standard-CP/M-Disketten (Format-Code A1) auf ZDOS-Disketten zu übertragen. Mit dem Mix-System kann natürlich auch wie mit einem normalen Systen gearbeitet werden.

## 2. -- Hardware - Voraussetzungen --

Die 5/8-Option setzt folgende Rechnerkonfiguration voraus:

- H.K.M. CPU oder M.K.C. CPU II
- 2 Floppycontroller H.K.M. FDC 8/5
- 64 KByte RAM bei der H.K.M. CPU

Der erste Floppycontroller ist eine ganz normale FDC 8/5 Karte. Er bedient die beiden 5,25" Laufwerke "A" und "B", die als Laufwerke "0" und "1" angeschlossen werden. Er hat das Standardadressprom. Der zweite Floppycontroller ist eine FDC 8/5 Karte mit einem Sonderadressprom. Er belegt anstelle der Adressen von 00H bis 03H die Adressen von 04H bis 07H. Er bedient die beiden 8" Laufwerke "C" und "D". "C" ist ein Double-Density-Double-Sided-Laufwerk, das als Laufwerk "0" gejumpert wird; "D" ist das Single-Density-Single-Side Laufwerk und wird als "1" gejumpert. Die beiden Floppycontroller arbeiten im DMA-MODE und müssen daher in einem DMA-fähigen System fest nebeneinander stecken. Das Mix-System läuft nur, wenn beide Controller vorhanden sind.

## 3. -- Besonderheiten in der Bedienung --

Die Bedienung des Mix-Systems entspricht im Wesentlichen der eines normalen ZDOS-Systems. Eine Ausnahme stellen nur FORMAT und SYSGEN dar. Es sollten 3 FORMAT und 3 SYSGEN generiert werden.

- FORMAT5 und SYSGEN5 für 5,25" Laufwerke
  (Formatcode A oder B)

  Bedienung wie üblich

- FORMAT8D und SYSGEN8D für das 8" Double-Density-Laufwerk
  (Formatcode E oder F und FDC-Base-Address = 04)

  Das 8" Double-Density-Laufwerk ist bei FORMAT und SYSGEN als Laufwerk A anzusprechen, während es unter ZDOS nach wie vor Laufwerk C bleibt.

- FORMAT8S und SYSGEN8S für das 8" Single-Density-Laufwerk
  (Formatcode G und FDC-Base-Address = 04)

  Das 8" Single-Density-Laufwerk ist bei FORMAT und SYSGEN als Laufwerk B anzusprechen, während es unter ZDOS nach wie vor Laufwerk D bleibt.

Ferner sind die Adressen der Floppycontroller und die gejumperten Adressen der Laufwerke bei Benutzung der FDC85-Testroutinen zu beachten.

Das Mix-System kann als ganz normales 5,25" System unter ZDOS5D oder ZDOS5N betrieben werden. Der Betrieb als 8" System unter ZDOS8 oder ZDOS8S ist nicht möglich (falsche Adresse des 8"-Controllers).

Das relativ langsame Arbeiten der Diskette D (Single Density) ist unter ZDOS normal.

## 4. -- 1:1-Kopieren von Disketten --

Zu beachten ist, das ein Kopieren mit **SYSGEN** nur zwischen gleichartigen Laufwerken möglich ist. Da **SYSGEN** unabhängig von **ZDOS** ist, können alle Formate kopiert werden. (Die Disketten müssen hierfür richtig formatiert worden sein.)

- 5,25" mit **SYSGEN5** von A nach B

- 8" DD mit **SYSGEN8D** von A nach B
  -- Achtung: 8"A und 8"B --
  unter **ZDOS** können diese Disketten nur in Laufwerk **C** gelesen werden.

- 8" SD mit **SYSGEN8S** von A nach B
  -- Achtung: 8"A und 8"B --
  unter **ZDOS** können diese Disketten nur in Laufwerk **D** gelesen werden.

## 5. -- Die HEAS und WINCH-Files --

Im WINCH.MAC-File werden die 5,25" Laufwerke vereinbart. Zu setzen sind die DISK-VALUES für die 5,25" Disks. Es folgt der Vereinbarungsteil des Files WINCH.MAC:

```
;-------------------------------------------------
;       PSEUDO-WINCHESTER-FILE                    I
;       2 5,25" FLOPPYS ALS WINCHESTER            I
;                                                 I
;       VERSION 1.1  (10.10.82) ZU HEAS5.5        I
;-------------------------------------------------
;
;       Voraussetzungen zum Betrieb der 5/8" Option
;       ist:
;       1 FDC 8/5 auf Adresse 00H..03H (5" Laufwerke)
;       1 FDC 8/5 auf Adresse 04H..07H (8" Laufwerke)
;       beide 5" Laufwerke Double Density als 0 und 1 gejumpert
;       1 8" Laufwerk Double Density als 0 gejumpert
;       1 8" Laufwerk Single Density als 1 gejumpert
;
;       in HEAS55 ist FDC equ 4 zu setzen (statt 0)
;                 ist WINCH equ 0 zu setzen (statt 1)
;                 ist SD equ 0 zu setzen (statt 1)
;                 ist DMA equ 0 zu setzen (statt 1)
;
;-------------------------------------------------
;       I/O-ADDRESSES                             I
;-------------------------------------------------
WFDC    EQU     0               ; 5,25" STATUS
WFDD    EQU     WFDC+1          ;       DATA
WFTC    EQU     WFDD+1          ;       FTC
WDMA    EQU     WFTC+1          ;       DMA
;-------------------------------------------------
;       DISK-VALUES                               I
;-------------------------------------------------
WDISKS  EQU     2               ; 5,25"
WMAXT   EQU     40              ; 40 tracks
WSPT    EQU     40
WCAPAC  EQU     400             ; double sided
WHLT    EQU     28
WSRT    EQU     8
```

In **HEAS55** sind also die beiden 8" Laufwerk zu definieren; d.h. es ist ein **HEAS55** für ein Double-Density-Double-Sided- und ein Standard-CP/M-Laufwerk zu erstellen (HEAS8S), in dem durch die Steuervariable **WINCH** die Mix-Option eingefügt wird und in dem der Floppycontroller die Basis-Adresse 04H hat. Erforderlich dazu sind **HEAS55.MAC** und **WINCH.MAC** auf Laufwerk "B" und **MACRO80** und **LINK80**. Für das Mix-System ist KI gleich DE00H zu setzen. (Anfangsadresse 200H Bytes früher!)

## 6. -- Hinweise zur Inbetriebnahme --

Wie in der ZDOS-Dokumentation beschrieben, ist HEAS durch Änderung von Steuervariablen an unterschiedliche (vorgesehene) Hardware leicht adaptierbar. Eine 5/8-Version kann durch folgende Modifikation in HEAS erzeugt werden:
HEAS ist zu editieren; dabei muß das Semikolon vor der Zeile:

```
WINCH     EQU       0
```

entfernt werden und dafür die Zeile:

```
WINCH     EQU       1
```

durch Einfügen eines Semikolon ausgeschaltet werden. Anschließend ist HEAS neu zu assemblieren und neu zu linken. WINCH.MAC wird hierbei durch den M80 automatisch von Laufwerk "B" dazu eingelesen. Benötigt werden hierzu:

- ein Editor
- der MACRO80 als Assembler (MICROSOFT)
- der LINK80 als Bindelader (MICROSOFT)

Das so erhaltene HEAS?X.COM ist jetzt über das Betriebssystem ZDOS zu patchen. -- Achtung: ZDOS ist für die Adresse DE00 zu generieren --. Hierfür wird ZDOS.COM ab 1000H in den Speicher geladen und danach HEAS?X.COM ab 2400H darüber in den Speicher geladen. Anschließend kann mit SYSGEN das so erstellte System auf die Systemspuren einer Testdiskette kopiert werden.

-- Achtung --

HEAS muß für die richtige Adresse assembliert werden. In HEAS ist hierfür die Zeile

```
KI        EQU       0DE00H
```

an die entsprechende Systemadresse anzupassen. Die KI-adresse ist gleichzeitig die Adresse, die bei GENCOM angegeben werden muß.

Einfacher ist die Verwendung des mitgelieferten HEAS?X.LOD-Files. Dieses ist über das entsprechende ZDOS.LOD-File zu kopieren. Dann kann durch DO GEN ... wie bisher ein Betriebssystem erstellt werden. Die anfangs erwähnte Methode muß nur bei Änderungen des HEAS oder WINCH benutzt werden.

## 7. -- Überkopieren von HEASxx.LOD --

Folgende Schritte sind erforderlich:

1. Laden von ZDOSxx.LOD

   ```
   0A>LOAD ZDOSxx.LOD $100
   ```

2. Laden von HEASxx.LOD

   ```
   0A>LOAD HEASxx.LOD $1780
   ```

3. Speichern des so modifizierten ZDOS.LOD

   ```
   0A>SAVE 34 ZDOSxx.LOD $100
   ```

Jetzt sollte das so neu erstellte ZDOS getestet werden:

4. System generieren:

   ```
   0A>DO GEN ZDOSxx $adresse  (maximal: DE00)
   ```

5. Versuchen die Diskette aus Laufwerk "B" in Laufwerk "A" zu booten. (Kaltstart)

6. Ausgiebiges Testen des so erstellten Systems

## 8. -- Inhalt des FDC 8/5 - Sonderadressproms --

Das Sonderadressprom mit dem unten angegebenen Inhalt wird als IC3 in die Floppycontrollerkarte FDC 8/5 eingesetzt. Sie ist für 8" Laufwerke einzustellen und bedient dann im 5/8-System die Laufwerke C und D.

| Adresse | Daten | Auswahl |
|---|---|---|
| 00..03 | F | nicht selektiert |
| 04 | 5 | FDC Status |
| 05 | 5 | FDC Daten |
| 06 | C | FDC TC |
| 07 | 9 | DMA |
| 08..FF | F | nicht selektiert |

```
                                TITLE   BOOTSTRAPLOADER VERS 6.5 (03.02.83)
                              ;
                                .Z80
                                .PHASE  0H
                              ;================================================
                              ;EPC    EQU     0               ; EPC I WANTED
0001                          EPC     EQU     1               ; FDC 5 or 8/5
                              ;================================================
                              ;
                              ;       **** I/O-PORTS ****
                              ;
                                IFF     EPC
                                .PRINTX / EPC I selected /
                              FDC     EQU     0F2H            ; FLOPPY STATUS
                              MSTART  EQU     8000H
                                ELSE
                                .PRINTX / FDC 8-5 selected /
0000                          FDC     EQU     0               ; FLOPPY STATUS
2000                          MSTART  EQU     2000H
00EC                          PAGE0   EQU     0ECH            ; REGISTER FOR 0000..3FFF
00ED                          PAGE4   EQU     PAGE0+1         ; REGISTER FOR 4000..7FFF
00EE                          PAGE8   EQU     PAGE4+1         ; REGISTER FOR 8000..BFFF
00EF                          PAGEC   EQU     PAGE8+1         ; REGISTER FOR C000..FFFF
                                ENDIF
0001                          FDD     EQU     FDC+1           ; FLOPPY DATA
0002                          FTC     EQU     FDC+2           ; FLOPPY TERMINATE
0003                          DMA     EQU     FDC+3           ; DMA - PORT
                              ;
00EA                          CTC     EQU     0EAH            ; BAUDRATE
00E3                          CONB    EQU     0E3H            ; SIO B STATUS
                              ;
                              ;       **** ENTRYPOINT ****
                              ;
0000  31 FFD0                 BOOT::  LD      SP,0FFD0H       ; STACK IS NOT USED
0003  C3 0334                         JP      GOBOOT
                              ;
                              ;       **** TABLE FOR Z-80 INTERRUPT MODE 2 ****
                              ;
0006  0638                    VECTOR: DEFW    TERMI
FFFF                          CL      EQU     0FFFFH
0008  FFFF FFFF                      DEFW    CL,CL,CL,CL,CL,CL,CL,CL,CL,CL,CL,CL
000C  FFFF FFFF
0010  FFFF FFFF
0014  FFFF FFFF
0018  FFFF FFFF
001C  FFFF FFFF
```

```
                        ;
                        ;       **** PATCH AREA FOR JUMPS INTO ****
                        ;       **** USER-ROUTINES, LOCATED AT ****
                        ;       **** END OF PROGRAM (LOC: LAST) ***
                        ;
                        ;       **** USER ROUTINES MUST NOT USE STACK ****
                        ;
                        ;       **** USE ONLY THE ALLOWED REGISTERS ****
                        ;
                        ;       **** USER FUNCTION 0 ****
                        ;       ***** SWITCH ON RAM *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL,IY
                        ;       RETURN          JP (IX)
                        ;
0020                    USER0:  EMPTY                   ; INITIALIZE I/O
                        ;
                        ;       **** USER FUNCTION 1 ****
                        ;       *** INIT USER-CONSOLE ***
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL,IY
                        ;       RETURN:         JP (IX)
                        ;
0023                    USER1:  EMPTY
                        ;
                        ;       **** USER FUNCTION 2 ****
                        ;       ***** SEND MESSAGE ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE
                        ;       RETURN:         JP (IX)
                        ;       HL POINTS TO MESSAGE
                        ;       MESSAGE ENDS ON '$'
                        ;
0026                    USER2:  EMPTY
                        ;
                        ;       **** USER FUNCTION 3 ****
                        ;       ***** CHECK MEMORY ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,IY
                        ;       RETURN:         JP (IX)
                        ;
                        ;       SET     IY = HIGHEST AVAILIBLE RAM
                        ;
0029                    USER3:  EMPTY
                        ;
                        ;       **** USER FUNCTION 4 ****
                        ;       ****** DEFINE MODE ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;       SET     ZERO-FLAG IF DMA-MODE
                        ;       SET     NON-ZERO-FLAG IF NON-DMA
                        ;
002C                    USER4:  EMPTY
```

```
                        ;
                        ;       **** USER FUNCTION 5 ****
                        ;       ** SPECIFY DISK TIMING **
                        ;       *** NON-DMA-MODE ONLY ***
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
002F                    USER5:  EMPTY
                        ;
                        ;       **** USER FUNCTION 6 ****
                        ;       ** SPECIFY DISK TIMING **
                        ;       ***** DMA-MODE-ONLY *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
0032                    USER6:  EMPTY
                        ;
                        ;       **** USER FUNCTION 7 ****
                        ;       **** EXECUTED BEFORE ****
                        ;       **** JP TO HEAS-BOOT ****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE
                        ;       RETURN:         JP (IX)
                        ;
0035                    USER7:  EMPTY
                        ;
                        ;       **** USER FUNCTION 8 ****
                        ;       ***** CONSOLE INPUT *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
0038                    USER8:  EMPTY
                                OFF
                        ;
                        ;       **** DISK ROUTINES ****
                        ;
                        ;       **** EQUATES ****
                        ;
000A                    RETRY   EQU     10              ; RETRY COUNTER
                        ;
00F0                    HLT     EQU     240             ; MS HEADLOAD TIME (8MHZ)
000E                    SRT     EQU     14              ; MS STEP RATE TIME (8 MHZ)
00F0                    HUT     EQU     240             ; HEADUNLOAD TIME
                        ;
00EE                    HLTS    EQU     ((HLT-1)/2)*2   ; COMPUTED VALUES FOR UPD 765
000F                    HUTS    EQU     (HUT+15)/16
0002                    SRTS    EQU     16-SRT
                                OFF
```

```
                        ;
                        ;       **** SPECIFY DISK TIMING FOR NON-DMA-MODE ****
                        ;
0234    21 023E         SPEC:   LD      HL,SPCT         ; .SPEC TABLE
0237    06 03                   LD      B,3             ; 3 BYTES COMMAND
0239    CD 02E3                 CALL    CMFD
                                RETURN
023E    03 2F EF        SPCT:   DEFB    03H,SRTS*16+HUTS,HLTS+1
                        ;
                        ;       **** SPECIFY FOR DMA-MODE ****
                        ;
0241    21 024B         SPECD:  LD      HL,SPCDT
0244    06 03                   LD      B,3
0246    CD 02D0                 CALL    MOTO
                                RETURN
024B    03 2F EE        SPCDT:  DEFB    3,SRTS*16+HUTS,HLTS
                                OFF
                        ;
                        ;       **** MAIN PROGRAM ****
                        ;
0334                    GOBOOT: IFT     EPC             ; CPU - II only (not EPC I)
0334    3E 00                   LD      A,0             ; SET MEMORY-PAGE-REGISTERS
0336    D3 EC                   OUT     (PAGE0),A       ; TO NORMAL CONFIGURATION
0338    3E 04                   LD      A,4
033A    D3 ED                   OUT     (PAGE4),A
033C    3E 08                   LD      A,8
033E    D3 EE                   OUT     (PAGE8),A
0340    3E 0C                   LD      A,0CH
0342    D3 EF                   OUT     (PAGEC),A
                                ENDIF
                                EXE     USER0           ; INITIALIZE
034F    21 006E                 LD      HL,NOBAUD
0352    D9                      EXX                     ; SAVE MESSAGE
                                EXE     USER1,INIT      ; HL' = .BAUDRATE
0361    21 00CA                 LD      HL,SGON         ; "BOOTSTRAPLOADER..."
                                EXE     USER2,SENDM     ; SEND MESSAGE
                                RUN     CBR             ; SEND BAUDRATE
                                EXE     USER2,SENDM
0387    21 0130                 LD      HL,BDMES        ; "BAUD"
                                EXE     USER2,SENDM
0398    21 0148         BDR9:   LD      HL,MEMORY       ; "MEMORY OK"
                                EXE     USER3,CHECK     ; CHECK MEMORY
03A9    7E                      LD      A,(HL)          ; ERROR ?
03AA    23                      INC     HL
03AB    08                      EX      AF,AF'
                                EXE     USER2,SENDM     ;"MEMORY OK/FAILED"
03BA    FD                      DEFB    0FDH            ; LD A,HIGH(IY)
03BB    7C                      LD      A,H
03BC    1F                      RRA                     ; COMPUTE SIZE
03BD    1F                      RRA
03BE    1F                      RRA
03BF    1F                      RRA
03C0    E6 0C                   AND     0CH
03C2    21 0138                 LD      HL,MSIZE
03C5    06 00                   LD      B,0
```

```
03C7  4F                      LD      C,A
03C8  09                      ADD     HL,BC
                              EXE     USER2,SENDM     ; 64K/48K/32K/16K
03D7  08                      EX      AF,AF'
03D8  B7                      OR      A
03D9  20 FE             STOP: JR      NZ,STOP         ; IF ERROR STOP
03DB  AF                      XOR     A
03DC  08                      EX      AF,AF'          ; A' = 0
03DD  FD                      DEFB    0FDH            ; SET STACK POINTER
03DE  7C                      LD      A,H             ; LD A,HIGH(IY)
03DF  67                      LD      H,A
03E0  FD                      DEFB    0FDH
03E1  7D                      LD      A,L             ; LD A,LOW(IY)
03E2  6F                      LD      L,A
03E3  11 002F                 LD      DE,2FH
03E6  B7                      OR      A
03E7  ED 52                   SBC     HL,DE
03E9  F9                      LD      SP,HL           ; STACK=HIGHEST_MEM-2FH
03EA  EB                      EX      DE,HL
03EB  21 06CD                 LD      HL,CMDTAB       ; COPY COMMAND-TABLES
03EE  01 002E                 LD      BC,CMDLEN
03F1  ED B0                   LDIR
03F3  F3                      DI                      ; SET INTERRUPT MODE 2
03F4  3E 00                   LD      A,HIGH(VECTOR)
03F6  ED 47                   LD      I,A
03F8  ED 5E                   IM      2
03FA  FB                      EI
                              EXE     USER4,MODE      ; DMA OR NON DMA MODE ?
0409  CA 04A8                 JP      Z,DMAMOD
040C  D9                      EXX
040D  16 01                   LD      D,1             ; NON DMA FLAG
040F  D9                      EXX
0410  21 017B                 LD      HL,NONMES       ; NON DMA MODE
                              EXE     USER2,SENDM
                        ;     5,25",DD,512 BYTE,10 SEC'S,INVERTED DATA
0421                    NONAGA:
0421  CD 0227           RDNDMA: CALL  INIFDC          ; SOFT RESET
                              EXE     USER5,SPEC      ; SPEC DISK TIMING
0432  CD 024E                 CALL    RECAL
0435  CD 024E                 CALL    RECAL
0438  06 0A                   LD      B,RETRY         ; NORM. 10 TIMES
043A  C5                RDLP0: PUSH    BC              ; SAVE
043B  FD E5                   PUSH    IY              ; HIGHEST RAM
043D  E1                      POP     HL
043E  11 1FFF                 LD      DE,1FFFH
0441  B7                      OR      A
0442  ED 52                   SBC     HL,DE           ; START OF ZDOS
0444  E5                      PUSH    HL
0445  21 0496                 LD      HL,RD0          ; READ TRACK 0 NONDMA
0448  06 09                   LD      B,9
044A  CD 02D0                 CALL    MOTO            ; TRANSFER COMMAND
044D  E1                      POP     HL              ; ADDRESS
044E  16 0A                   LD      D,10            ; 10 SECTORS
0450  CD 0287           RDL0:  CALL   RDDATA
0453  15                      DEC     D
```

```
0454    20 FA                   JR      NZ,RDL0         ; 10 DONE ?
0456    D3 02                   OUT     (FTC),A         ; TERMINATE
0458    CD 029D                 CALL    RESULT          ; ERROR ?
045B    C1                      POP     BC
045C    28 07                   JR      Z,RDT1          ; NO: READ TRACK 1
045E    10 DA                   DJNZ    RDLP0           ; YES: TRY AGAIN
0460    CD 02F4                 CALL    RDERR
0463    18 BC                   JR      RDNDMA
0465    CD 0261         RDT1:   CALL    SEEK1           ; TRACK 1
0468    06 0A                   LD      B,RETRY
046A    C5              RDLP1:  PUSH    BC
046B    FD E5                   PUSH    IY              ; SET UP PARAMETERS
046D    E1                      POP     HL
046E    11 0BFF                 LD      DE,0BFFH
0471    B7                      OR      A
0472    ED 52                   SBC     HL,DE
0474    E5                      PUSH    HL
0475    21 049F                 LD      HL,RD1
0478    06 09                   LD      B,9
047A    CD 02D0                 CALL    MOTO            ; TRANSFER COMMAND
047D    E1                      POP     HL
047E    16 05                   LD      D,5
0480    CD 0287         RDL1:   CALL    RDDATA          ; READ TRACK
0483    15                      DEC     D
0484    20 FA                   JR      NZ,RDL1
0486    D3 02                   OUT     (FTC),A
0488    CD 029D                 CALL    RESULT          ; ERROR ?
048B    C1                      POP     BC
048C    CA 0643                 JP      Z,LOADED        ; NO: IS LOADED
048F    10 D9                   DJNZ    RDLP1           ; TRY AGAIN
0491    CD 02F4                 CALL    RDERR
0494    18 BB                   JR      RDNDMA
0496    46 00 00 00     RD0:    DEFB    46H,0,0,0,1,2,10,10H,2
049A    01 02 0A 10
049E    02
049F    46 00 01 00     RD1:    DEFB    46H,0,1,0,1,2,10,10H,2
04A3    01 02 0A 10
04A7    02

                        ;
04A8                    DMAMOD:
04A8    D9                      EXX
04A9    16 00                   LD      D,0             ; DMA FLAG
04AB    D9                      EXX
04AC    21 018A                 LD      HL,DMAMES       ; DMA MODE
                                EXE     USER2,SENDM
04BD                    DMAAGA:
04BD    CD 0227                 CALL    INIFDC
                                EXE     USER6,SPECD     ; SPECIFY TIMING
04CE    CD 024E                 CALL    RECAL           ; SEEK TRACK 0
04D1    CD 024E                 CALL    RECAL
                        ;
                        ;       **** SINGLE OR DOUBLE DENSITY ****
                        ;
04D4    06 0A           TRYDD:  LD      B,RETRY
04D6    C5              TRYLP:  PUSH    BC
```

```
04D7    06 02                   LD      B,2
04D9    21 04FC                 LD      HL,RDDDT        ; READ ID (DOUBLE DENSITY)
04DC    CD 02D0                 CALL    MOTO
04DF    CD 029D                 CALL    RESULT
04E2    C1                      POP     BC
04E3    28 1B                   JR      Z,RDDD          ; OK: DOUBLE DENSITY
04E5    C5                      PUSH    BC
04E6    06 02                   LD      B,2             ; ERROR: TRY SINGLE DENSITY
04E8    21 04FE                 LD      HL,RDSDT
04EB    CD 02D0                 CALL    MOTO
04EE    CD 029D                 CALL    RESULT
04F1    C1                      POP     BC
04F2    CA 05A7                 JP      Z,RDSD          ; OK: SINGLE DENSITY
04F5    10 DF                   DJNZ    TRYLP           ; TRY AGAIN
04F7    CD 02F4                 CALL    RDERR
04FA    18 D8                   JR      TRYDD
04FC    4A 00           RDDDT:  DEFB    4AH,0
04FE    0A 00           RDSDT:  DEFB    0AH,0
                        ;
                        ;       **** READ IN DOUBLE DENSITY ****
                        ;
0500    06 0A           RDDD:   LD      B,RETRY         ; NOM. 10 TIMES
0502    C5              RDDLP:  PUSH    BC              ; SAVE
                                OFF                     ; SET UP PARAMETERS
0524    E5                      PUSH    HL
0525    06 09                   LD      B,9
0527    21 0595                 LD      HL,RDD0
052A    CD 02D0                 CALL    MOTO            ; READ TRACK 0
052D    01 1203                 LD      BC,RDLEN*256+DMA
0530    E1                      POP     HL
0531    ED B3                   OTIR                    ; START DMA
0533    CD 02AA                 CALL    RESDMA          ; ERRORS ?
0536    C1                      POP     BC
0537    CA 0643                 JP      Z,LOADED        ; NO: IS LOADED
053A    FE 40                   CP      40H             ; ONLY 10 SECTORS READ ?
053C    20 0A                   JR      NZ,NOT5
053E    7A                      LD      A,D
053F    FE 04                   CP      4
0541    20 05                   JR      NZ,NOT5
0543    7B                      LD      A,E
0544    FE 0B                   CP      11
0546    28 08                   JR      Z,RDDT1         ; YES: READ TRACK 1 TOO
0548    10 4B           NOT5:   DJNZ    RDD0            ; ERROR, TRY AGAIN
054A    CD 02F4                 CALL    RDERR
054D    C3 04BD                 JP      DMAAGA
0550    CD 0261         RDDT1:  CALL    SEEK1           ; READ TRACK 1
0553    06 0A                   LD      B,RETRY
0555    C5              RDDLP1: PUSH    BC
                                OFF                     ; SET UP PARAMETERS
0577    E5                      PUSH    HL
0578    06 09                   LD      B,9
057A    21 059E                 LD      HL,RDD1T
057D    CD 02D0                 CALL    MOTO            ; TRANSFER COMMAND
0580    01 1203                 LD      BC,RDLEN*256+DMA
0583    E1                      POP     HL
```

```
0584    ED B3                   OTIR                ; START DMA
0586    CD 029D                 CALL    RESULT
0589    C1                      POP     BC
058A    CA 0643                 JP      Z,LOADED    ; NO ERROR: IS LOADED
058D    10 C6                   DJNZ    RDDLP1      ; ERROR: TRY AGAIN
058F    CD 02F4                 CALL    RDERR
0592    C3 04BD                 JP      DMAAGA
0595    46 00 00 00     RDD0:   DEFB    46H,0,0,0,1,2,15,10H,2
0599    01 02 0F 10
059D    02
059E    46 00 01 00     RDD1T:  DEFB    46H,0,1,0,1,2,10,10H,2
05A2    01 02 0A 10
05A6    02

                        ;
                        ;       **** READ IN SINGLE DENSITY ****
                        ;       ****** 8" FLOPPY'S ONLY ! ******
                        ;
05A7                    RDSD:
05A7    06 0A                   LD      B,RETRY
05A9    C5              RDSLP0: PUSH    BC
                                OFF                 ; SET UP PARAMETERS
05CB    E5                      PUSH    HL
05CC    06 09                   LD      B,9
05CE    21 0626                 LD      HL,RDS0T
05D1    CD 02D0                 CALL    MOTO        ; READ TRACK 0 (SD)
05D4    01 1203                 LD      BC,RDLEN*256+DMA
05D7    E1                      POP     HL
05D8    ED B3                   OTIR                ; START DMA
05DA    CD 029D                 CALL    RESULT      ; ERROR ?
05DD    C1                      POP     BC
05DE    28 08                   JR      Z,RDST1     ; NO: READ TRACK 1
05E0    10 C7                   DJNZ    RDSLP0      ; ERROR: TRY AGAIN
05E2    CD 02F4                 CALL    RDERR
05E5    C3 04BD                 JP      DMAAGA
05E8    CD 0261         RDST1:  CALL    SEEK1       ; READ TRACK 1
05EB    06 0A                   LD      B,RETRY
05ED    C5              RDSLP1: PUSH    BC
                                OFF                 ; SET UP PARAMETERS
0608    E5                      PUSH    HL
0609    06 09                   LD      B,9
060B    21 062F                 LD      HL,RDS1T
060E    CD 02D0                 CALL    MOTO        ; TRANFER COMMAND
0611    01 1203                 LD      BC,RDLEN*256+DMA
0614    E1                      POP     HL
0615    ED B3                   OTIR                ; START DMA
0617    CD 029D                 CALL    RESULT      ; ERROR ?
061A    C1                      POP     BC
061B    CA 0643                 JP      Z,LOADED    ; NO: IS LOADED
061E    10 CD                   DJNZ    RDSLP1      ; YES: TRY AGAIN
0620    CD 02F4                 CALL    RDERR
0623    C3 04BD                 JP      DMAAGA
0626    06 00 00 00     RDS0T:  DEFB    6,0,0,0,1,0,26,7,128
062A    01 00 1A 07
062E    80
062F    06 00 01 00     RDS1T:  DEFB    6,0,1,0,1,0,26,7,128
```

```
0633    01 00 1A 07
0637    80
                        ;
                        ;       **** TERMINATE READ-COMMAND ****
                        ;
0638    D3 02           TERMI:  OUT     (FTC),A         ; TERMINATE UPD 765
063A    F5                      PUSH    AF
063B    3E C3                   LD      A,0C3H          ; RESET DMA
063D    D3 03                   OUT     (DMA),A
063F    F1                      POP     AF
0640    FB                      EI
0641    ED 4D                   RETI
                        ;
                        ;       **** SYSTEM LOADED, CHECK IT ! ****
                        ;
0643                    LOADED:
                                OFF                     ; CHECK ZDOS !
067C    11 1400         GODOS:  LD      DE,1400H        ; HL = START OF SYSTEM
067F    19                      ADD     HL,DE           ; HL = .HEAS-COLDSTART
0680    D9                      EXX
0681    7E                      LD      A,(HL)          ; BAUDRATE VALUE
0682    D9                      EXX
0683    F5                      PUSH    AF              ; SAVE
                                EXE     USER7
068F    F1                      POP     AF
0690    E9                      JP      (HL)            ; GO TO COLDSTART
                        ;
                                OFF
071B    FFFF            LAST:   DEFW    CL              ; ROOM FOR USER
                                END
```

```
                                TITLE   BOOTSTRAP_LOADER VERS 6.5 (03.02.83)
                        ;
                                .Z80
                                .PHASE  0H
                        ;================================================
0000                    EPC     EQU     0               ; EPC I WANTED
                        ;EPC    EQU     1               ; FDC 5 or 8/5
                        ;================================================
                        ;
                        ;       **** I/O-PORTS ****
                        ;
                                IFF     EPC
                                .PRINTX / EPC I selected /
00F2                    FDC     EQU     0F2H            ; FLOPPY STATUS
8000                    MSTART  EQU     8000H
                                ELSE
                                .PRINTX / FDC 8-5 selected /
                        FDC     EQU     0               ; FLOPPY STATUS
                        MSTART  EQU     2000H
                        PAGE0   EQU     0ECH            ; REGISTER FOR 0000..3FFF
                        PAGE4   EQU     PAGE0+1         ; REGISTER FOR 4000..7FFF
                        PAGE8   EQU     PAGE4+1         ; REGISTER FOR 8000..BFFF
                        PAGEC   EQU     PAGE8+1         ; REGISTER FOR C000..FFFF
                                ENDIF
00F3                    FDD     EQU     FDC+1           ; FLOPPY DATA
00F4                    FTC     EQU     FDC+2           ; FLOPPY TERMINATE
00F5                    DMA     EQU     FDC+3           ; DMA - PORT
                        ;
00EA                    CTC     EQU     0EAH            ; BAUDRATE
00E3                    CONB    EQU     0E3H            ; SIO B STATUS
                        ;
                        ;       **** ENTRYPOINT ****
                        ;
0000    31 FFD0         BOOT::  LD      SP,0FFD0H       ; STACK IS NOT USED
0003    C3 0340                 JP      GOBOOT
                        ;
                        ;       **** TABLE FOR Z-80 INTERRUPT MODE 2 ****
                        ;
0006    0634            VECTOR: DEFW    TERMI
FFFF                    CL      EQU     0FFFFH
0008    FFFF FFFF               DEFW    CL,CL,CL,CL,CL,CL,CL,CL,CL,CL,CL,CL
000C    FFFF FFFF
0010    FFFF FFFF
0014    FFFF FFFF
0018    FFFF FFFF
001C    FFFF FFFF
```

```
                        ;
                        ;       **** PATCH AREA FOR JUMPS INTO ****
                        ;       **** USER-ROUTINES, LOCATED AT ****
                        ;       **** END OF PROGRAM (LOC: LAST) ***
                        ;
                        ;       **** USER ROUTINES MUST NOT USE STACK ****
                        ;
                        ;       **** USE ONLY THE ALLOWED REGISTERS ****
                        ;
                        ;
                        ;       **** USER FUNCTION 0 ****
                        ;       ***** SWITCH ON RAM *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL,IY
                        ;       RETURN          JP (IX)
                        ;
0020                    USER0:  EMPTY                   ; INITIALIZE I/O
                        ;
                        ;       **** USER FUNCTION 1 ****
                        ;       *** INIT USER-CONSOLE ***
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL,IY
                        ;       RETURN:         JP (IX)
                        ;
0023                    USER1:  EMPTY
                        ;
                        ;       **** USER FUNCTION 2 ****
                        ;       ***** SEND MESSAGE ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE
                        ;       RETURN:         JP (IX)
                        ;       HL POINTS TO MESSAGE
                        ;       MESSAGE ENDS ON '$'
                        ;
0026                    USER2:  EMPTY
                        ;
                        ;       **** USER FUNCTION 3 ****
                        ;       ***** CHECK MEMORY ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,IY
                        ;       RETURN:         JP (IX)
                        ;
                        ;       SET     IY = HIGHEST AVAILIBLE RAM
                        ;
0029                    USER3:  EMPTY
```

```
                        ;
                        ;       **** USER FUNCTION 4 ****
                        ;       ****** DEFINE MODE ******
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;       SET     ZERO-FLAG IF DMA-MODE
                        ;       SET     NON-ZERO-FLAG IF NON-DMA
                        ;
002C                    USER4:  EMPTY
                        ;
                        ;       **** USER FUNCTION 5 ****
                        ;       ** SPECIFY DISK TIMING **
                        ;       *** NON-DMA-MODE ONLY ***
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
002F                    USER5:  EMPTY
                        ;
                        ;       **** USER FUNCTION 6 ****
                        ;       ** SPECIFY DISK TIMING **
                        ;       ***** DMA-MODE-ONLY *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
0032                    USER6:  EMPTY
                        ;
                        ;       **** USER FUNCTION 7 ****
                        ;       **** EXECUTED BEFORE ****
                        ;       **** JP TO HEAS-BOOT ****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE
                        ;       RETURN:         JP (IX)
                        ;
0035                    USER7:  EMPTY
                        ;
                        ;       **** USER FUNCTION 8 ****
                        ;       ***** CONSOLE INPUT *****
                        ;
                        ;       FREE REGISTERS: AF,BC,DE,HL
                        ;       RETURN:         JP (IX)
                        ;
0038                    USER8:  EMPTY
```

```
                                OFF
                        ;
                        ;       **** DISK ROUTINES ****
                        ;
                        ;       **** EQUATES ****
                        ;
000A                    RETRY   EQU     10              ; RETRY COUNTER
                        ;
00F0                    HLT     EQU     240             ; MS HEADLOAD TIME (8MHZ)
000E                    SRT     EQU     14              ; MS STEP RATE TIME (8 MHZ)
00F0                    HUT     EQU     240             ; HEADUNLOAD TIME
                        ;
00EE                    HLTS    EQU     ((HLT-1)/2)*2   ; COMPUTED VALUES FOR UPD 765
000F                    HUTS    EQU     (HUT+15)/16
0002                    SRTS    EQU     16-SRT
                                OFF
                        ;
                        ;       **** SPECIFY DISK TIMING FOR NON-DMA-MODE ****
                        ;
0240    21 024A         SPEC:   LD      HL,SPCT         ; .SPEC TABLE
0243    06 03                   LD      B,3             ; 3 BYTES COMMAND
0245    CD 02EF                 CALL    CMFD
                                RETURN
024A    03 2F EF        SPCT:   DEFB    03H,SRTS*16+HUTS,HLTS+1
                        ;
                        ;       **** SPECIFY FOR DMA-MODE ****
                        ;
024D    21 0257         SPECD:  LD      HL,SPCDT
0250    06 03                   LD      B,3
0252    CD 02DC                 CALL    MOTO
                                RETURN
0257    03 2F EE        SPCDT:  DEFB    3,SRTS*16+HUTS,HLTS
                                OFF
                        ;
                        ;       **** MAIN PROGRAM ****
                        ;
0340                    GOBOOT: IFT     EPC             ; CPU - II only (not EPC I)
                                LD      A,0             ; SET MEMORY-PAGE-REGISTERS
                                OUT     (PAGE0),A       ; TO NORMAL CONFIGURATION
                                LD      A,4
                                OUT     (PAGE4),A
                                LD      A,8
                                OUT     (PAGE8),A
                                LD      A,0CH
                                OUT     (PAGEC),A
                                ENDIF
                                EXE     USER0           ; INITIALIZE
034B    21 008E                 LD      HL,NOBAUD
034E    D9                      EXX                     ; SAVE MESSAGE
                                EXE     USER1,INIT      ; HL' = .BAUDRATE
035D    21 00CA                 LD      HL,SGON         ; "BOOTSTRAPLOADER..."
                                EXE     USER2,SENDM     ; SEND MESSAGE
                                RUN     CBR             ; SEND BAUDRATE
                                EXE     USER2,SENDM
0383    21 013C                 LD      HL,BDMES        ; "BAUD"
```

```
                                EXE     USER2,SENDM
0394    21 0154         BDR9:   LD      HL,MEMORY       ; "MEMORY OK"
                                EXE     USER3,CHECK     ; CHECK MEMORY
03A5    7E                      LD      A,(HL)          ; ERROR ?
03A6    23                      INC     HL
03A7    08                      EX      AF,AF'
                                EXE     USER2,SENDM     ;"MEMORY OK/FAILED"
03B6    FD                      DEFB    0FDH            ; LD A,HIGH(IY)
03B7    7C                      LD      A,H
03B8    1F                      RRA                     ; COMPUTE SIZE
03B9    1F                      RRA
03BA    1F                      RRA
03BB    1F                      RRA
03BC    E6 0C                   AND     0CH
03BE    21 0144                 LD      HL,MSIZE
03C1    06 00                   LD      B,0
03C3    4F                      LD      C,A
03C4    09                      ADD     HL,BC
                                EXE     USER2,SENDM     ; 64K/48K/32K/16K
03D3    08                      EX      AF,AF'
03D4    B7                      OR      A
03D5    20 FE           STOP:   JR      NZ,STOP         ; IF ERROR STOP
03D7    AF                      XOR     A
03D8    08                      EX      AF,AF'          ; A' = 0
03D9    FD                      DEFB    0FDH            ; SET STACK POINTER
03DA    7C                      LD      A,H             ; LD A,HIGH(IY)
03DB    67                      LD      H,A
03DC    FD                      DEFB    0FDH
03DD    7D                      LD      A,L             ; LD A,LOW(IY)
03DE    6F                      LD      L,A
03DF    11 002F                 LD      DE,2FH
03E2    B7                      OR      A
03E3    ED 52                   SBC     HL,DE
03E5    F9                      LD      SP,HL           ; STACK=HIGHEST_MEM-2FH
03E6    EB                      EX      DE,HL
03E7    21 06C9                 LD      HL,CMDTAB       ; COPY COMMAND-TABLES
03EA    01 002E                 LD      BC,CMDLEN
03ED    ED B0                   LDIR
03EF    F3                      DI                      ; SET INTERRUPT MODE 2
03F0    3E 00                   LD      A,HIGH(VECTOR)
03F2    ED 47                   LD      I,A
03F4    ED 5E                   IM      2
03F6    FB                      EI
                                EXE     USER4,MODE      ; DMA OR NON DMA MODE ?
0405    CA 04A4                 JP      Z,DMAMOD
0408    D9                      EXX
0409    16 01                   LD      D,1             ; NON DMA FLAG
040B    D9                      EXX
040C    21 0187                 LD      HL,NONMES       ; NON DMA MODE
                                EXE     USER2,SENDM
                        ;       5,25",DD,512 BYTE,10 SEC'S,INVERTED DATA
041D                    NONAGA:
041D    CD 0233         RDNDMA: CALL    INIFDC          ; SOFT RESET
                                EXE     USER5,SPEC      ; SPEC DISK TIMING
042E    CD 025A                 CALL    RECAL
```

```
0431  CD 025A                 CALL   RECAL
0434  06 0A                   LD     B,RETRY          ; NORM. 10 TIMES
0436  C5              RDLP0:  PUSH   BC               ; SAVE
0437  FD E5                   PUSH   IY               ; HIGHEST RAM
0439  E1                      POP    HL
043A  11 1FFF                 LD     DE,1FFFH
043D  B7                      OR     A
043E  ED 52                   SBC    HL,DE            ; START OF ZDOS
0440  E5                      PUSH   HL
0441  21 0492                 LD     HL,RD0           ; READ TRACK 0 NONDMA
0444  06 09                   LD     B,9
0446  CD 02DC                 CALL   MOTO             ; TRANSFER COMMAND
0449  E1                      POP    HL               ; ADDRESS
044A  16 0A                   LD     D,10             ; 10 SECTORS
044C  CD 0293           RDL0:  CALL   RDDATA
044F  15                      DEC    D
0450  20 FA                   JR     NZ,RDL0          ; 10 DONE ?
0452  D3 F4                   OUT    (FTC),A          ; TERMINATE
0454  CD 02A9                 CALL   RESULT           ; ERROR ?
0457  C1                      POP    BC
0458  28 07                   JR     Z,RDT1           ; NO: READ TRACK 1
045A  10 DA                   DJNZ   RDLP0            ; YES: TRY AGAIN
045C  CD 0300                 CALL   RDERR
045F  18 BC                   JR     RDNDMA
0461  CD 026D           RDT1:  CALL   SEEK1            ; TRACK 1
0464  06 0A                   LD     B,RETRY
0466  C5                RDLP1:  PUSH   BC
0467  FD E5                   PUSH   IY               ; SET UP PARAMETERS
0469  E1                      POP    HL
046A  11 0BFF                 LD     DE,0BFFH
046D  B7                      OR     A
046E  ED 52                   SBC    HL,DE
0470  E5                      PUSH   HL
0471  21 049B                 LD     HL,RD1
0474  06 09                   LD     B,9
0476  CD 02DC                 CALL   MOTO             ; TRANSFER COMMAND
0479  E1                      POP    HL
047A  16 05                   LD     D,5
047C  CD 0293           RDL1:  CALL   RDDATA           ; READ TRACK
047F  15                      DEC    D
0480  20 FA                   JR     NZ,RDL1
0482  D3 F4                   OUT    (FTC),A
0484  CD 02A9                 CALL   RESULT           ; ERROR ?
0487  C1                      POP    BC
0488  CA 063F                 JP     Z,LOADED         ; NO: IS LOADED
048B  10 D9                   DJNZ   RDLP1            ; TRY AGAIN
048D  CD 0300                 CALL   RDERR
0490  18 8B                   JR     RDNDMA
0492  46 00 00 00       RD0:   DEFB   46H,0,0,0,1,2,10,10H,2
0496  01 02 0A 10
049A  02
049B  46 00 01 00       RD1:   DEFB   46H,0,1,0,1,2,10,10H,2
049F  01 02 0A 10
04A3  02

                        ;
```

```
04A4                    DMAMOD:
04A4    D9                      EXX
04A5    16 00                   LD      D,0             ; DMA FLAG
04A7    D9                      EXX
04A8    21 0196                 LD      HL,DMAMES       ; DMA MODE
                                EXE     USER2,SENDM
04B9                    DMAAGA:
04B9    CD 0233                 CALL    INIFDC
                                EXE     USER6,SPECD     ; SPECIFY TIMING
04CA    CD 025A                 CALL    RECAL           ; SEEK TRACK 0
04CD    CD 025A                 CALL    RECAL
                        ;
                        ;       **** SINGLE OR DOUBLE DENSITY ****
                        ;
04D0    06 0A           TRYDD:  LD      B,RETRY
04D2    C5              TRYLP:  PUSH    BC
04D3    06 02                   LD      B,2
04D5    21 04F8                 LD      HL,RDDDT        ; READ ID (DOUBLE DENSITY)
04D8    CD 02DC                 CALL    MOTO
04DB    CD 02A9                 CALL    RESULT
04DE    C1                      POP     BC
04DF    28 1B                   JR      Z,RDDD          ; OK: DOUBLE DENSITY
04E1    C5                      PUSH    BC
04E2    06 02                   LD      B,2             ; ERROR: TRY SINGLE DENSITY
04E4    21 04FA                 LD      HL,RDSDT
04E7    CD 02DC                 CALL    MOTO
04EA    CD 02A9                 CALL    RESULT
04ED    C1                      POP     BC
04EE    CA 05A3                 JP      Z,RDSD          ; OK: SINGLE DENSITY
04F1    10 DF                   DJNZ    TRYLP           ; TRY AGAIN
04F3    CD 0300                 CALL    RDERR
04F6    18 D8                   JR      TRYDD
04F8    4A 00           RDDDT:  DEFB    4AH,0
04FA    0A 00           RDSDT:  DEFB    0AH,0
                        ;
                        ;       **** READ IN DOUBLE DENSITY ****
                        ;
04FC    06 0A           RDDD:   LD      B,RETRY         ; NOM. 10 TIMES
04FE    C5              RDDLP:  PUSH    BC              ; SAVE
                                OFF                     ; SET UP PARAMETERS
0520    E5                      PUSH    HL
0521    06 09                   LD      B,9
0523    21 0591                 LD      HL,RDDO
0526    CD 02DC                 CALL    MOTO            ; READ TRACK 0
0529    01 12F5                 LD      BC,RDLEN*256+DMA
052C    E1                      POP     HL
052D    ED B3                   OTIR                    ; START DMA
052F    CD 02B6                 CALL    RESDMA          ; ERRORS ?
0532    C1                      POP     BC
0533    CA 063F                 JP      Z,LOADED        ; NO: IS LOADED
0536    FE 40                   CP      40H             ; ONLY 10 SECTORS READ ?
0538    20 0A                   JR      NZ,NOT5
053A    7A                      LD      A,D
053B    FE 04                   CP      4
053D    20 05                   JR      NZ,NOT5
```

```
053F   7B                     LD      A,E
0540   FE 0B                  CP      11
0542   28 08                  JR      Z,RDDT1         ; YES: READ TRACK 1 TOO
0544   10 4B           NOT5:  DJNZ    RDDO            ; ERROR, TRY AGAIN
0546   CD 0300                CALL    RDERR
0549   C3 04B9                JP      DMAAGA
054C   CD 026D         RDDT1: CALL    SEEK1           ; READ TRACK 1
054F   06 0A                  LD      B,RETRY
0551   C5               RDDLP1: PUSH   BC
                              OFF                     ; SET UP PARAMETERS
0573   E5                     PUSH    HL
0574   06 09                  LD      B,9
0576   21 059A                LD      HL,RDD1T
0579   CD 02DC                CALL    MOTO            ; TRANSFER COMMAND
057C   01 12F5                LD      BC,RDLEN*256+DMA
057F   E1                     POP     HL
0580   ED B3                  OTIR                    ; START DMA
0582   CD 02A9                CALL    RESULT
0585   C1                     POP     BC
0586   CA 063F                JP      Z,LOADED        ; NO ERROR: IS LOADED
0589   10 C6                  DJNZ    RDDLP1          ; ERROR: TRY AGAIN
058B   CD 0300                CALL    RDERR
058E   C3 04B9                JP      DMAAGA
0591   46 00 00 00     RDDO:  DEFB    46H,0,0,0,1,2,15,10H,2
0595   01 02 0F 10
0599   02
059A   46 00 01 00     RDD1T: DEFB    46H,0,1,0,1,2,10,10H,2
059E   01 02 0A 10
05A2   02
                       ;
                       ;       **** READ IN SINGLE DENSITY ****
                       ;       ****** 8" FLOPPY'S ONLY ! ******
                       ;
05A3                   RDSD:
05A3   06 0A                  LD      B,RETRY
05A5   C5               RDSLP0: PUSH   BC
                              OFF                     ; SET UP PARAMETERS
05C7   E5                     PUSH    HL
05C8   06 09                  LD      B,9
05CA   21 0622                LD      HL,RDS0T
05CD   CD 02DC                CALL    MOTO            ; READ TRACK 0 (SD)
05D0   01 12F5                LD      BC,RDLEN*256+DMA
05D3   E1                     POP     HL
05D4   ED B3                  OTIR                    ; START DMA
05D6   CD 02A9                CALL    RESULT          ; ERROR ?
05D9   C1                     POP     BC
05DA   28 08                  JR      Z,RDST1         ; NO: READ TRACK 1
05DC   10 C7                  DJNZ    RDSLP0          ; ERROR: TRY AGAIN
05DE   CD 0300                CALL    RDERR
05E1   C3 04B9                JP      DMAAGA
05E4   CD 026D         RDST1: CALL    SEEK1           ; READ TRACK 1
05E7   06 0A                  LD      B,RETRY
05E9   C5               RDSLP1: PUSH   BC
                              OFF                     ; SET UP PARAMETERS
0604   E5                     PUSH    HL
```

```
0605    06 09                   LD      B,9
0607    21 062B                 LD      HL,RDSIT
060A    CD 02DC                 CALL    MOTO            ; TRANFER COMMAND
060D    01 12F5                 LD      BC,RDLEN*256+DMA
0610    E1                      POP     HL
0611    ED B3                   OTIR                    ; START DMA
0613    CD 02A9                 CALL    RESULT          ; ERROR ?
0616    C1                      POP     BC
0617    CA 063F                 JP      Z,LOADED        ; NO: IS LOADED
061A    10 CD                   DJNZ    RDSLP1          ; YES: TRY AGAIN
061C    CD 0300                 CALL    RDERR
061F    C3 04B9                 JP      DMAAGA
0622    06 00 00 00     RDSOT:  DEFB    6,0,0,0,1,0,26,7,128
0626    01 00 1A 07
062A    80
062B    06 00 01 00     RDSIT:  DEFB    6,0,1,0,1,0,26,7,128
062F    01 00 1A 07
0633    80

                        ;
                        ;       **** TERMINATE READ-COMMAND ****
                        ;
0634    D3 F4           TERMI:  OUT     (FTC),A         ; TERMINATE UPD 765
0636    F5                      PUSH    AF
0637    3E C3                   LD      A,0C3H          ; RESET DMA
0639    D3 F5                   OUT     (DMA),A
063B    F1                      POP     AF
063C    FB                      EI
063D    ED 4D                   RETI
                        ;
                        ;       **** SYSTEM LOADED, CHECK IT ! ****
                        ;
063F                    LOADED:
                                OFF                     ; CHECK ZDOS !
0678    11 1400         GODOS:  LD      DE,1400H        ; HL = START OF SYSTEM
067B    19                      ADD     HL,DE           ; HL = .HEAS-COLDSTART
067C    D9                      EXX
067D    7E                      LD      A,(HL)          ; BAUDRATE VALUE
067E    D9                      EXX
067F    F5                      PUSH    AF              ; SAVE
                                EXE     USER7
068B    F1                      POP     AF
068C    E9                      JP      (HL)            ; GO TO COLDSTART
                        ;
                                OFF
0717    FFFF            LAST:   DEFW    CL              ; ROOM FOR USER
                                END
```

Fehlermeldeformular

Einsenden an Ihren Händler, bei dem Sie **ZDOS** gekauft haben.

ZDOS-Version:..... DMA ... NON-DMA ...

ZDOS-Nummer:..... 8" ... 5,25" ...

In welchem Programm ist der Fehler? .....

CPU-Karte: ..... modifiziert? ja ... nein ...

RAM-Karte: ..... modifiziert? ja ... nein ...

FDC-Karte: ..... modifiziert? ja ... nein ...

Laufwerke: ..... modifiziert? ja ... nein ...

Modifikationen an ZDOS? ja ... nein ...

Falls bei Modifikation ja angekreuzt ist, bitte hier die Änderung genau beschreiben:

Beschreibung des aufgetretenen Fehlers:

**Bitte in jedem Fall ein Beispiel beilegen, im dem der angegebene Fehler sich bemerkbar macht.** Falls möglich auf Diskette!

Anschrift: Name: .....

Straße: .....

Postleitzahl: .....

Ort: .....

**Original-ZDOS-Diskette miteinsenden**