Das Disketten Operations-System II

DOS 2.0

Das Handbuch ist auf Grundlage des amerikanischen Originals der Firma Atari Inc., Sunnyville übersetzt worden. Die Herausgabe erfolgt im Namen und in Verantwortung der Firma IRATA GMBH 1 Berlin 44, Hermannstr.9

Die fehlenden Seitennummern sind drucktechnisch bedingt. Es sind keine fehlenden Seiten.

Vorwort

Das Handbuch für das Atari-Disketten-Operationssystem II (Disk-Operating-System II) ist für zwei Gruppen von Benutzern gedacht: Den neuen und den fortgeschrittenen.

Der Neuling wird feststellen, daß das Disketten-Operations-System (D O S) hier mit klar definierten Begriffen und konsequenten Arbeitsabläufen erklärt ist. Er sollte sich allerdings zuvor noch informieren in

a) dem Bedienungshandbuch für Atari 400 bzw. 800,

b) der speziellen Bedienungsanleitung für das Atari-810-Plattensystem bzw. für das 815-Doppelplatten-System,

c) einer der Atari-Programmanweisungen (vor allem BASIC).

Der Neubenutzer des D O S sollte sich dann mit den Abschnitten I - IV dieses Buches bekannt machen. Die Abschnitte V und VI richten sich an den schon besser trainierten Benutzer.

Ein solcher ist man dann, wenn man bereits auf dem Atari-D O S I früher gearbeitet hat oder schon Kenntnisse des ganzen ATARI-Computer-Systems, vom ATARI-BASIC und womöglich des Assemblers und anderer Programmiersprachen besitzt.

Für beide dieser Gruppen sind, wenn immer möglich, Grafiken eingefügt, sodaß alles klar vollziehbar wird.

In Anhang B finden sich Anmerkungen zu den angewandten Fachausdrücken, ein Nachschlageregister der Therme enthält Anhang J. Abschnitt IV befaßt sich ausdrücklich mit der Anwendung von BASIC-Befehlen.

Der erfahrene Benutzer sollte Anhang I lesen, wo die wichtigsten Unterschiede zwischen D O S I und D O S II erklärt sind, besonders die neu hinzugekommene MEN.SAV-Funktion.

INHALT-

DIE BENUTZUNG DIESES HANDBUCHES

Dies Handbuch wurde mit voller Rücksicht auf den Benutzer geschrieben. Jeder Abschnitt behandelt eine Phase der zweiten Version des ATARI-Plattensystems II (DOS II). Der DOS-Neuling kann leicht die zum Arbeitsbeginn mit DOS nötigen Angaben finden, ohne daß er mit Nebeninformationen belastet wird.

Andererseits sieht der fortgeschrittene Benutzer schnell, was er zusätzlich zur Verfügung hat, um komplexere Arbeiten zu gestalten.

FÜR DEN NEULING IM DOS

Abschnitt 1 erklärt die Benutzung dieses Handbuchs und die Handgriffe für die wichtigsten Operationen:

a) Definition vom DOS.
b) Vorbereitung des Systems.
c) Erklärung des DOS-Menus.

Die Abschnitte 2 und 3 diskutieren und erklären alles, was zur Diskette gehört, vom Formatieren bis zum Speichern.

Der Anfänger in DOS sollte die Abschnitte 1 und 2 durchlesen, bevor er die praktische Arbeit anfängt. Dadurch wird er von vornherein mit der Materie vertraut.

FÜR NEULINGE UND FORTGESCHRITTENE

Abschnitt 3 bringt den Leser unmittelbar mit dem DOS II in Kontakt, indem er erklärt, wie man die einzelnen Disketten mit Hilfe von Ablagekennzeichnungen und Zusatzkennzeichnungen voneinander unterscheidet. Dieser Abschnitt führt außerdem in das Laden und Abspeichern von Programmen ein.

In Abschnitt 4 überschneiden sich die Informationen für Anfänger und Fortgeschrittene. Hier findet man die detaillierte Beschreibung aller Möglichkeiten des DOS-Menues und Hinweise zu Ihrem Einsatz. Einige dieser Wahlmöglichkeiten kommen wohl nur für Kenner der Assembler/Editor-Kassetten und des hexadezimalen Systems in Frage.

FÜR FORTGESCHRITTENE

Abschnitt 5 wiederholt die bei DOS II angewandten BASIC-Befehle und gibt Beispielprogramme wieder, die den Gebrauch der Input/Output-Befehle im einzelnen zeigen. Jedes Befehlsformat wird durch ein Beispiel ergänzt, aus dem man sehen kann, welche Arten von Daten in den einzelnen Parametern angewandt werden.
Abschnitt 6 bietet zusätzliche Informationen über ATARI-Plattenlaufwerke und die Disketten (dies ist für beide Benutzergruppen interessant!). Dazu werden Besonderheiten der Abspeicherung und Auffindung von Daten behandelt. Die Zusammenfassung des Handbuchs weist ein Verzeichnis von Spezialausdrücken auf sowie Zusatzangaben für Fortgeschrittene, z. B. Speicherpläne, Fehlermeldungen oder Einsparung von RAM-Platz.

WIE SETZT MAN DOS II IN GANG

DOS ist eine Abkürzung für "Disketten-Operations-System". Ohne ein solches System könnte das ATARI-Privatcomputersystem nicht mit den zugehörigen Plattenlaufwerken in Verbindung treten. Das DOS besteht aus umfassenden Dienstprogrammen, die folgende Vorgänge ermöglichen.

a) Speicherung von Programmen auf Diskette,
b) Ablesen von Programmen von der Diskette,
c) Aufbau und Ergänzen von Datenablagen, die für Programme gebraucht werden,
d) Kopieren von Plattenspeicherungen,
e) Löschen alter Plattenspeicherungen,
f) Einlesen und Abspeichern von Binärablagen (für Fortgeschrittene),
g) Verschieben von Datenblocks in den Speicher und aus dem Speicher, auf den Monitor, auf die Platte oder an den Drucker.

Wer noch nie mit einem DOS umgegangen ist, wird das ATARI-DOS II leicht verständlich und handhabbar finden. Wer zuvor DOS I von ATARI benutzt hat, wird feststellen, daß sich bei den Menue-Möglichkeiten und deren Befehlsparametern einiges geändert hat. DOS II hat einige zusätzliche Vorteile, weil es dem Benutzer mehr freien Speicherraum läßt und größere Flexibilität bietet (siehe Anhang H).

VORBEREITUNG DES SYSTEMS

Zunächst prüft man, ob das vorhandene ATARI-Grundgerät wenigstens 16K RAM besitzt. Dann schließt man das 810-Plattengerät bzw. das 815-Doppellaufwerk an.

Die Detailanweisungen für den Anschluß finden sich im Bedienungsbuch für ATARAI 400 bzw. 800 und in den Anleitungen, die den Plattengeräten beigegeben sind.

ANSCHLUSS ZUSÄTZLICHER PLATTENGERÄTE

Wer zusätzliche ATARI-Plattenlaufwerke an die vorhandene ATARI anschließen will, kann das mit dem sogenannten "DAISY-CHAINING: Jedes Plattengerät hat an der Rückseite zwei Ausgänge mit der Bezeichnung I/O PERIPHERAL; man verbindet nun die Leitung des zweiten Plattengerätes mit dem ersten, und schließt dann die Leitung des ersten Laufwerks an die Grundmaschine an. Abbildung 1-1 zeigt einige der möglichen Zusammenstellungen von Plattenanlagen.

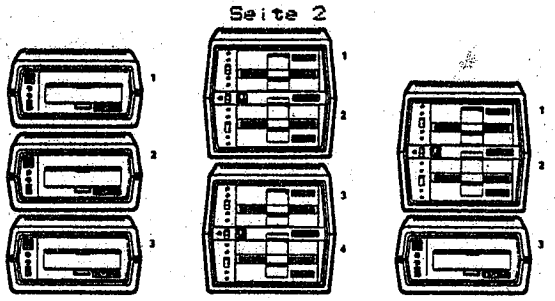
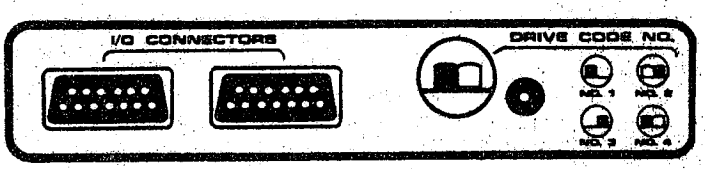


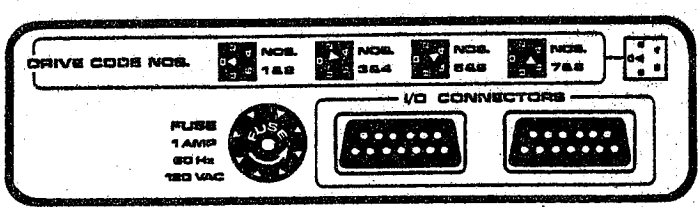
Figure 1-1 Disk Drive Configurations Available

DAS EINSTELLEN VON LAUFWERK CODE ZIFFERN

An der Rückseite jedes ATARI 810-Gerätes sieht man eine Vertiefung mit zwei Tasten. Die weiße und die schwarze Taste muß man an jedem Gerät so verschieben, wie es der Code-Definitionen dieses Gerätes entspricht.
Hierzu richtet man sich nach Abbildung 1-2 a.
An der Rückseite des Doppelgerätes 815 sitzt ein Wähler, den man auf die richtige Codierung einstellen muß (entsprechend Abb. 1-2 b). Wenn nur ein 815-Gerät vorhanden ist, muß das obere Laufwerk immer auf 1, das untere Laufwerk auf 2 eingestellt sein. Unbedingt ist darauf zu achten, daß jedes Teilgerät eine von anderen verschiedene Codeziffer hat.



(a) ATARI 810 Disk Drive



(b) ATARI 815 Dual Disk Drive

Figure 1-2 Drive Code Settings

KENNZEICHNUNG DER PLATTENGERÄTE

Sobald die Codeziffern für die Plattenwerke richtig eingestellt sind, sollte man jedes Einzelgerät mit einem entsprechenden Etikett versehen, sodaß immer klar ist, auf welches Gerät sich ein Befehl bezieht.
Auf jeden Fall muß die Haupt- (Master-) Diskette bzw. die Systemdiskette IMMER in das Laufwerk Nr. 1 gebracht werden.

EINSCHIEBEN EINER DISKETTE

Dieser Vorgang ist einfach, muß aber sehr sorgfältig erfolgen. Eine falsch sitzende Diskette kann Startfehler verursachen, sobald der Einlesevorgang beginnt. Außerdem kann die Diskette selbst beschädigt werden. Man schaltet das Laufwerk (die Laufwerke) ein, wartet, bis das Kontrollicht für BUSY ausgeht und schiebt dann die Diskette wie folgt ein:

1. Diskette aus der äußeren Tasche nehmen. Dabei Vorsicht: Die Diskette muß an der schwarzen versiegelten Schutzhülle gehalten werden. Keinesfalls die freie Oberfläche der Platte selbst berühren, weil hierdurch die Lese-Schreib-Schicht beschädigt werden könnte. Die Diskette auch nicht so halten, daß der Finger durch die Mittelöffnung geht. Keinesfalls die Schutzhülle entfernen!
2. Die Diskette ist mit der Etikettseite nach oben zu halten. Das Etikett muß zum Benutzer hinzeigen, der Pfeil darauf zum Einschub des Laufwerks (Abb. 1-3). Eine etwa vorhandene Schreibschutz-Nut muß links vom Benutzer liegen!
Es ist zu beachten, daß eine Haupt-(Master)-Diskette keine Schreibschutz-Nut hat, da sie vom Hersteller bereits mit automatischem Schreibschutz versehen ist.

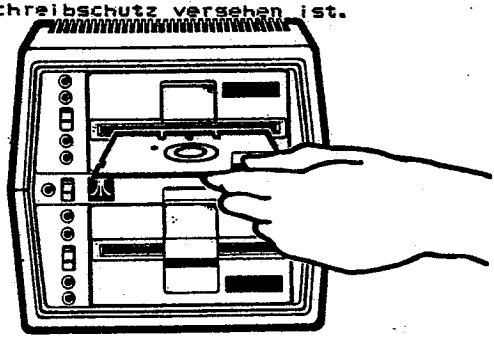


Figure 1-3 Inserting a Diskette Into a Disk Drive

3. Jetzt öffnet man den Schacht des Laufwerks (oberes Laufwerk), falls es sich um eine ATARI 815 handelt und schiebt vorsichtig aber kräftig die Diskette ein. Schließen des Einschubdeckels.

WAS IST EIN DOS MENUE

Das DOS-Menue wird durch die Hauptdiskette in den Computer gelesen. Diese steckt immer im Laufwerk 1. Das DOS-Menue entspricht, wie der Name sagt, im Prinzip einer Speisenkarte; es enthält eine Auswahl von Arbeitsmöglichkeiten, die über den Bildschirm projiziert werden. Der Benutzer sucht das gewünschte aus, in dem er den entsprechenden Codebuchstaben tastet und dann die Returntaste drückt. Damit steht die gewählte Teilarbeit zur Verfügung.

Die Menue-Listen für ATARI 810 (Einfachdichte, entsprechend 128 Bytes pro Sektor) und die für 815 (Doppeldichte) entsprechend 256 Bytes pro Sektor) sind gleich.

Der einzige Unterschied besteht in der Versions-Bezeichnung wie folgt:
a) Einfachdichte (auf ATARI 810): Hier scheint die Versions-Bezeichnung 2 OS oben rechts im Bild. Das S bedeutet einfache Dichte.
b) Doppeldichte (auf ATARI 815): Hier erscheint die Versions-Bezeichnung 2.OD oben rechts im Bild. D bedeutet doppelte Dichte.

WIE RUFT MAN DAS DOS MENUE AN

Nachdem das Bildgerät eingeschaltet, die Hauptdiskette eingeschoben und das Plattengerät ebenfalls eingeschaltet ist, wird das DOS II wie folgt geladen:

I MIT EINER ROM KASSETTE

1. das Kontrollicht BUSY leuchtet während des Ladevorgangs auf. Keinesfalls versuchen, die Diskette herauszunehmen, während das Licht an ist! Falls die BASIC-Kassette eingelegt ist, erscheint das Bereitschaftswort READY auf dem Bild, sobald DOS II vollständig geladen ist (bei Verwendung der Assembler-Editor-Kassette heißt das Bereitschaftswort EDIT). Damit ist der erste Teil des Ladevorgangs abgeschlossen.
2. Jetzt DOS-Tasten und RETURN drücken. Das DOS II-Menue erscheint nun im Bild (vgl. Abb. 1-4). Damit ist der zweite Teil des Ladevorgangs beendet.

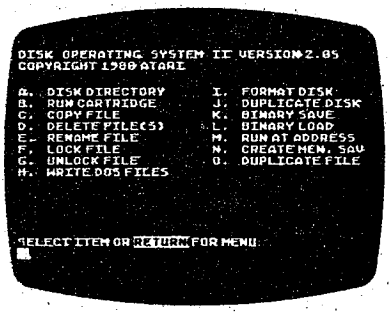


Figure 1-4 The DOS II Menu

Bei Kassettenbenutzung gelangen zunächst die Dienst-Programme FILE MANAGEMENT SUBSYSTEM und MINI-DOS, dann die Dienstprogramme DISK UTILITY PACKAGE und das eigentliche DOS-Menue in den Computer.

II OHNE VERWENDUNG EINER ROM KASSETTE

In diesem Fall wird DOS II vollständig geladen, sobald der Computer eingeschaltet ist. Dabei leuchtet die BUSY-Lampe am Plattenlaufwerk auf. Während dies Signal brennt, darf auf keinen Fall die Diskette rausgezogen werden. Sobald DOS II eingelesen ist, erscheint das DOS-Menue automatisch auf dem Bildschirm. Nachdem man eine Wahl per Codebuchstaben getroffen hat, gibt die Anlage Bereitschaftsmeldung auf dem Monitor und verlangt weitere Informationen. Insofern handelt es sich um ein Menue mit "Selbstbereitstellung". Gewöhnlich lautet die Bereitschaftsanzeige: SELECT ITEM OR RETURN FOR MENU (wähle Einzelprogramm oder kehre auf Menue zurück).

Das bedeutet, daß die Anlage vom Benutzer erwartet:
a) daß er einen Buchstaben eintastet und RETURN drückt, um die Wahl anzugeben, die er wünscht, oder
b) daß er nur RETURN drückt und dadurch zur Wiedergabe des ganzen Menues zurückkehrt.

DIE WAHLMÖGLICHKEIT DES DOS II

Weiter unten werden alle DOS II-Wahlverzweigungen genannt und kurz erklärt. Bitte nicht die Verzweigungen benutzen, bevor man sie wirklich verstanden hat! Details werden im Abschnitt 4 erklärt.

DAS PLATTENVERZEICHNIS (Disk Directory)

Es enthält eine Liste aller Datenablagen, die auf Diskette gespeichert sind. Wenn man Wahl A?????? und zweimal RETURN drückt, erhält man die Wiedergabe der Ablage-Kennzeichen, der Zusatzbezeichnungen (falls vorhanden), die Anzahl der jeweils belegten Sektoren sowie die Anzahl der noch freien Sektoren.

DER BEFEHL ROMKASSETTEN ABLAUF (RUN CARTRIDGE):

(nur benutzbar, wenn Kassette in der Computerkonsole vorhanden). Hierdurch wird die Rückkehr auf Kassettensteuerung möglich. (Bei der ATARI 800 handelt es sich um die im linken Fach sitzende Kassette).

DER BEFEHL KOPIERE ABLAGE (COPY FILE).

Man benutzt diese Wahlmöglichkeit, falls zwei oder mehr Plattenträger vorhanden sind und Ablagen von einer Platte auf die andere kopiert werden sollen. Man kann mit Hilfe dieses Programms auf zwei gleichlautende Plattenablagen auf ein- und dieselbe Diskette speichern, in dem man der ursprünglichen Ablage eine zweite Bezeichnung beigibt. Der Befehl: Lösche Ablage (DELETE FILE). Dieser Zweig des Menus erlaubt die Löschung einer Ablage von der Diskette, wodurch dann wieder Sektoren frei werden.

UMBENENNUNGEN RENAME FILE

Mit dieser Anwahl kann man den Namen einer Datenablage ändern.

Ablagesperre (LOCK FILE)

Diese Option verhindert, daß versehentlich gelöscht oder geändert wird. Das Ablesen der Daten ist bei LOCK FILE möglich, eine Aufzeichnung wird jedoch verhindert. - Bei der Wiedergabe des Gesamtverzeichnisses (DISK DIRECTORY) werden die gesperrten Ablagen jeweils mit einem Stern versehen.

Entsperrung der Ablage (UNLOCK FILE)

Dieses Programm nimmt den Stern vom Namen der gegebenenfalls angewählten gesperrten Ablage fort und hebt deren Sperrung auf.

Schreibe neues DOS

Man benutzt diese Option, wenn man von der Masterdiskette stammende DOS-Teile auf irgendeiner anderen Diskette ersetzen oder hinzufügen will (DOS.SYS DUP.SYS).

FORMATIERE DISKETTE (FORMAT DISKETTE)

Diesen Zweig wählt man, um eine leere Diskette zu formatieren. Das ist notwendig, bevor man mit irgendeiner Aufzeichnung auf dieser Platte beginnen kann. Man muß dabei sicher sein, daß nichts mehr auf der Diskette steht, was erhalten bleiben soll.

Das Format-Programm wird normalerweise für die formatierten Disketten der ATARI 810 angewandt (CX8111).

DUPLIZIERE DISKETTE (DUPLIKATE DISK)

Hierdurch erreicht man das exakte Kopieren eines Diskenteninhalts auf eine zweite Diskette (Details werden in Abschnitt II beschrieben).

DUPLIZIERE ABLAGE (DUPLICATE FILE)

Hiermit kann man auch dann eine Ablage von einer Diskette auf eine zweite kopieren, wenn man nur ein Laufwerk an der Anlage besitzt (vgl. den Abschnitt über Ablagen-Duplizierung).

RICHTE MEM.SAV EIN (CREATE MEM.SAV)

Diese Option erlaubt Einräumen freien Sektorraums auf der Diskette für das Aufzeichnen des im RAM befindlichen Programms, während die DUP.SYS-Ablage benutzt wird (vgl. wegen Details Abschn. 4, CREATE MEM.SAV). Wir raten, auf jeder neuen Diskette, die als Systemdiskette dienen soll, ein solches MEM.SAV-FILE einzurichten.

Zwar wird man bei größerer Erfahrung mit DOS feststellen, daß es Fälle gibt, in denen MEM.SAV-Ablagen keine nützliche Funktion besitzen, doch der Unbequemlichkeit des Wartens, bis die Daten in den Speicher gelesen sind, steht der Vorteil gegenüber, daß eine Löschung von der Diskette ausgeschlossen ist. Ein Anwendungsbeispiel wäre, daß man im ROM kein Programm hat, das erhalten werden soll, wenn das DOS eingegeben wird.

BINÄRSPEICHERUNG (BINARY SAVE)*

Diese Wahlmöglichkeit speichert die Inhalte bestimmter Speicherplätze auf eine Diskette (sie manipuliert Programme in Assemblersprache).

BINÄR EINLESUNG (BINARY LOAD)

Mit dieser Option kann man ein Objektprogramm von einer Diskette holen. Es handelt sich hier um die Umkehrung des Programms BYNARY SAVE (bei der Binäreinlesung werden ebenfalls Programme in Assemblersprache manipuliert).

LAUFE AB ADRESSE (RUN AT ADDRESS)*

Bei dieser Wahl kann man die hexadezimale Startadresse eines Objektprogrammes eingeben, nach dem es per BINARY LOAD in den Speicher gebracht worden ist (RUN AT ADDRESS führt Programme in Assemblersprache aus).

* Die Optionen BINARY SAVE, BINARY LOAD und RUN AT ADDRESS sind mehr für den fortgeschrittenen Benutzer von DOS II bestimmt. Sie werden im einzelnen im Abschnitt 4 behandelt.

DISKETTEN

DIE HAUPTDISKETTE (Masterdiskette)

Die sogenannte Masterdiskette enthält die Programme des Diskettenoperationssystems. Diese Programme schließen alle Dienstroutinen und System-Ordnungsroutinen ein, mit deren Hilfe das Plattensystem des ATARI läuft. Ohne ein Disketten-Operationssystem ist das Plattengerät nicht ansprechbar.

Jede Masterdiskette enthält:

a) eine DOS.SYS-Ablage. Das ist eine Programmserie, die die sogenannten Ordungs- Routinen (FILE MANAGEMENT SYSTEM) sowie nie im Rahmen untergebrachten Teile des DUP.SYS (das sogenannte MINI-DOS) enthält. Diese im RAM stehenden Teile von DUP.SYS bestehen aus den Unterprogrammen, die im FILE MANAGEMENT-SYSTEM gesteuert werden: Ablagenlöschung, Umbenennung von Ablagen, Sperrung und Entsperrung von Ablagen und Disketten-Formatierung.

b) eine DUP.SYS-Ablage:

Es handelt sich hier um ein Paket von Disketten-Dienstroutinen, welches das DOS MENU und die DOS-Unterprogramme enthält, die nicht vom FILE MANAGEMENT SYSTEM gesteuert werden.

Immer, wenn man das DOS-MENU sehen oder diese DOS-Funktion ausführen will (binäres Einlesen, binäres Abspeichern, Anlauf ab Adresse, Ablauf der Kassette, Ablagenkopierung, Ablagen-Duplizierung, Platten-Duplizierung), so muß man das DUP.SYS-Paket in den Speicher lesen, in dem man DOS tastet und dann RETURN drückt.

Anmerkung:

Wenn man das DUP.SYS-Paket einliest, werden normalerweise Daten im unteren Programmbereich des Speichers überschrieben, wo BASIC- oder Assemblersprachen-Programme stehen.

Jedoch transportiert das MINI-DOS-SYSTEM beim Aufbau einer MEM.SAV-Ablage auf der Kassette alle Daten, die sich im RAM befinden, auf die Diskette, bevor es das DUP.SYS einliest.

Nach Beendigung der Arbeit mit dem DUP.SYS ermöglicht MEM.SAV die automatische Rücklesung der zuvor gelöschten Programmelemente.

DIE AUTORUN.SYS-ABLAGE.

Dieses FILE wird eingesetzt, um die Periphär-Geräte (soweit vorhanden) zu überprüfen und um Maschinencode-Programme ablaufen zu lassen (vergl. hierzu Abschnitt 4 dieses Handbuchs, wo AUTO-RUN-SYS detailliert behandelt wird).

FORMATIERTE DISKETTEN II AUF ATARI 810

Zusätzlich zur Masterdiskette II (CX 8104) ist dem ATARI 810 - Plattengerät eine formatierte Diskette II (CX 8111) beigegeben. Obwohl diese Diskette keinerlei Ablagen oder Programme enthält, ist sie vom Hersteller vorformatiert worden. D.h., daß diese Diskette in Spuren und Sektoren eingeteilt wurde, bevor man sie in die Schutzhülle tat (vergl. Abbildung 2-1), sie hat also eine verbesserte Sektoren-Einteilung. Diese verbesserte Einteilung erlaubt dem Benutzer eine schnellere Speicherung, bzw. Rücklesung von Informationen, als dies mit den auf der 810 selbst formatierten Disketten möglich wäre. Diese sogenannte Leerdiskette ist als Reserve-Masterdiskette gedacht.

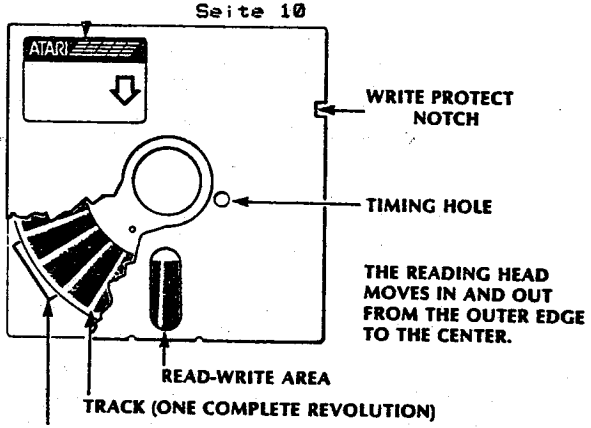


Figure 2-1

A Formatted

Diskette

an spricht hier auch von einer Systemdiskette. Es ist die Haupt- skette, mit der man normalerweise arbeitet, damit das Original icht in Gefahr gerät (vergl. Anweisung zum Herstellen einer 'stemdiskette).

ills man irgendwann die Ablage auf einer solchen vorformatierten iskette löschen will, muß man die Wahlmöglichkeit D.DELETEFILES enutzen. Dadurch erhält man die verbesserte Sektoreneinteilung, it deren Hilfe man einen schnelleren Ablauf erreicht.
)llte der Benutzer aus irgendeinem Grunde diese Diskette eformatieren wollen (etwa um sie auf beschädigte Sektoren zu esten), so geht die verbesserte Sektoreneinteilung verloren.
.B.: Auf dem ATARI-810-Plattengerät kann man auch solche eerdisketten benutzen, die nicht vorformatiert sind. In dem Fall uß man sie selber mit Hilfe der FORMAT DISK-Wahlmöglichkeit des)S-MENUS selbst formatieren, bevor man DOS-Ablagen oder Programme darauf speichern kann (vergl. hierzu die Anweisungen ber das Formatieren von Disketten). In solchen Fällen ist atürlich der Vorteil der höheren Arbeitsgeschwindigkeit von)rformatierten Disketten nicht gegeben.

DAS FORMATIEREN VON DISKETTEN

ede Leerdiskette muß formatiert werden, damit sie benutzbar wird es sei denn, man benutzt eine schon formatierte Diskette II für TARI 810 (CX 8111)). Das Formatieren teilt die Diskette so ein, aß DOS II erkennen kann, wo die Information jeweils steht. Im egensatz zur Schallplatte, die eine spiralförmig laufende Rille at, sind auf der Computerplatte die Spuren als konzentrische reise und nur magnetisch aufgetragen und unsichtbar.
enn die Diskette formatiert ist, enthält sie 40 konzentrisch ngeordnete Spuren, die in 18 "Tortenstücke" (Sektoren) ufgeteilt sind. Die Speicherkapazität ergibt sich aus der ultiplikation der Spurenzahl (40) mit der Sektorenzahl (18) leich 720 Spurensektoren.

.3 dieser 720 Einzelsektoren werden allerdings von DOS II belegt
ind stehen daher nicht für Ablagen und andere Daten zur
/erfügung. Praktisch sieht die Einteilung so aus:
; Sektoren für Systemanwahl (Booten)
3 Sektoren für das Verzeichnis (Diskdirectory)
. Sektor für die Mengenübersicht (Volumtable)
[Sektor (720) ist nicht adressierbar.
.3 gehen also der allgemeinen Benutzung verloren.

)araus ergibt sich, daß pro Platte nur 707 Sektoren frei
/erfügbar sind. Die ATARI 810, ein Plattengerät mit einfacher
)ichte, kann 128 Bytes pro Sektor speichern. Da jedoch 3 Bytes
)ro Sektor für das Organisationssystem verloren gehen, bleibt
eine freie Kapazität von 88.375 Bytes auf einer solchen Platte
)brig. Um eine Diskette zu formatieren, muß man die Option
[.FORMATDISK anrufen. Das erübrigt sich natürlich, wenn man
formatierte Disketten II (CX 8111) benutzt. Falls man ein
.aufwerk 810 als Gerätenummer 1 einsetzt, muß man die Anweisungen
)nter A unten beachten. Bei Einsatz eines 815-Doppelgerätes als
Nummer 1 und Nummer 2 gelten die Anweisungen unter B unten.
A. Anweisungen für das ATARI-810-Plattengerät:
1. Schalte das Gerät ein. Warte bis Kontroll Lampe erlischt.
2. prüfe, ob Einstellung an der Rückseite des Gerätes auf Nr. 1
steht (vergl. Abbildung 1-2 betr. Gerätecodierung).
3. schiebe die 810-Masterdiskette II ein (CX8104) und schließe
den Einschubdeckel.
4. schalte Computer-Konsole an. Jetzt wird das DOS in den
Speicher gelesen.
5. Wenn die Bereitschaftsanzeige (READY) erscheint (bei Benutzung
der BASIC-Cassette), taste DOS und drücke RETURN-Taste. Nach
einigen Sekunden erscheint das DOS-MENU auf dem Bildschirm. Wenn
keine Cassette benutzt wird, stellt sich das DOS-MENU automatisch
ein.
6. Taste I, um die Option FORMAT anzuwählen, und drücke dann
RETURN.
7. Sobald die Bereitschaftsmeldung (WHICH DRIVE TO FORMAT)
erscheint, nimm die Masterdiskette aus dem Gerät und schiebe
Leerdiskette ein. Schließe den Laufwerkschacht, Taste 1 und
drücke die RETURN-Taste.
8. Sobald die Bereitschaftsanzeige "TYPE Y TO FORMAT DISK 1"
erscheint, taste Y und drücke RETURN. Das Kontrollicht BUSY
leuchtet auf, und das System formatiert jetzt automatisch die
Diskette.
9. Sobald die Bereitschaftsanzeige "SELECT ITEM OR RETURN FOR
MENU" erscheint, ist das Formatieren beendet, und man kann die
Diskette bespeichern.
Sind zwei oder mehr 810-Laufwerke angeschlossen, so kann man eine
Leerdiskette auf jedem der Laufwerke formatieren, nur muß man
wissen, welcher Gerätecode zu welchem Gerät gehört, weil man
sonst auf die Bereitschaftsanzeige "WHICH DRIVE TO FORMAT" nicht
richtig reagieren kann.
B. Anweisungen für das ATARI-815-Doppelplattengerät:
1. Schalte Laufwerk bzw. Laufwerke ein. Warte, bis BUSY-Lampe
ausgeht.

2. Prüfe, ob die Codeschaltung an der Geräterückseite richtig eingestellt ist (vergl. Abbildung 1-2).

3. Schiebe die Masterdiskette von 815 (CX 8201) in Laufwerk 1 ein und schließe die Klappe.

4. Schalte die Computerkonsole ein. DOS lädt jetzt in den Speicher.

5. Sobald die Bereitschaftsanzeige "READY" erscheint (bei Benutzung einer BASIC-Cassette) taste DOS und drücke RETURN. Nach einigen Sekunden erscheint das DOS Menu auf dem Bildschirm. Wenn keine Cassette verwendet wird, kommt das DOS-Menu automatisch.

6. Taste I ein, um FORMAT anzuwählen und drücke RETURN.

7. Sobald die Bereitschaftsanzeige "WHICH DRIVE TO FORMAT" erscheint, schiebe die Leerdiskette in Laufwerk 1, schließe die Öffnung, taste 2 und drücke RETURN.

8. Sobald die Bereitschaftsanzeige "TYPE Y TO FORMAT DISK 2" erscheint, taste Y und drücke RETURN. Die BUSY-Lampe geht an und das System formatiert nun die Diskette in Laufwerk 2.

9. Wenn die Bereitschaftsanzeige "SELECT ITEM OR RETURN FOR MENU" erscheint, ist die Formatierung beendet und die Diskette kann gespeichert werden.

Herstellen einer Systemdiskette aus der Masterdiskette: Die erste vorzunehmende Plattenoperation besteht im Duplizieren der Masterdiskette. Man macht das deswegen, damit die Original Hauptdiskette geschützt bleibt. Das herzustellende Duplikat bezeichnet man als Systemdiskette (Arbeitsdiskette). Sie wird bei der laufenden Arbeit eingesetzt, um das DOS in den Speicher zu laden. Es gibt zwei Arten zum Herstellen der Systemdiskette, je nach den zur Verfügung stehenden Laufwerken:

Herstellen einer Systemdiskette auf dem Laufwerk 810:

- Schalte Monitor und Laufwerk ein und warte, bis die BUSY-Lampe (obere rote Lampe) erlischt.
- Nimm die Masterdiskette aus der äußeren Hülle.
- Schiebe sie in das Laufwerk und schließe dessen Deckel,
- schalte die Computerkonsole ein.
- Falls eine BASIC-Cassette angewandt wird, erscheint jetzt die Bereitschaftsanzeige "READY". Taste DOS und drücke RETURN. Falls keine Cassette benutzt wird, erscheint das DOS MENU automatisch auf dem Monitor.

Nimm die Masterdiskette heraus und schiebe eine formatierte Diskette ein. Diese Leerdiskette kann eine der folgenden sein:

- eine ATARI 810 II, formatiert (CX 8111),
- eine zuvor vom Benutzer mit Hilfe von DOS II formatierte Diskette,
- eine mit DOS II neu formatierte Diskette.

7. Drücke H und RETURN-Taste zur Anwahl der Option "WRITE DOS FILES".
8. Wenn Bereitschaft ("DRIVE TO WRITE DOS FILES TO?" erscheint, taste 1 und RETURN.
9. Wenn Bereitschaft ("TYPE Y TO WRITE DOS TO DRIVE 1") erscheint, Y tasten und RETURN drücken.
10. Die Meldung WRITING NEW DOS FILES zeigt sich jetzt auf dem Monitor.
11. Sobald die Nachricht " SELECT ITEM OR RETURN FOR MENU" erscheint, ist die Masterdiskette dupliziert und eine Systemdiskette ist gefertigt.

An dieser Stelle empfehlen wir dringend, eine MEM.SAV- Ablage zu fertigen (vergl. unter Abschnitt über MEM.SAV weiter unten). MEM.SAV richtet eine bestimmte Anzahl von Sektoren auf der Systemdiskette (oder einer anderen Diskette) für das gerade im RAM-Programm ein, das während des Ablaufs der DOS-Funktionen abgestellt werden soll. Man spricht hierbei von der DUP.SYS-Ablage (vergl. Abschnitt über die DUP.SYS-Funktion).

12. Taste N und drücke RETURN, um eine MEM.SAV-Ablage auf der Systemdiskette zu erstellen.
13. Sobald die Bereitschaftsmeldung "TYPE Y TO CREATE MEM.SAV erscheint, Y eingeben und RETURN drücken.
14. Wenn die Bereitschaftsmeldung "SELECT ITEM OR RETURN TO MENU" erscheint, ist die MEM.SAV-Ablage auf der Systemdiskette gespeichert.

Sind zwei oder mehr 810-Diskettenlaufwerke vorhanden, kann man die formatierte Diskette in jedes beliebige Plattenfach einschieben, bevor man die Option H.WRITE DOS FILES wählt.

Man muß allerdings wissen, welches Laufwerk man benutzt, damit man auf das Signal "DRIVE TO WRITE DOS FILES TO?" richtig reagieren kann (vergl. Schritt 8 oben).

II: Das Erstellen einer Systemdiskette mit dem ATARI 815.

1. Doppelplattengerät einschalten. Prüfen, ob der Gerätecode an der Rückseite richtig eingestellt ist(vergl. Abbildung 1-2) das obere Laufwerk sollte mit 1, das untere mit 2 nummeriert sein.
2. Die Masterdiskette aus der äußeren Hülle nehmen.
3. Die Masterdiskette in Fach 1 einschieben und den Fachdeckel schließen.
4. Die Computerkonsole einschalten.
5. Bei Benutzung einer BASIC-Cassette erscheint jetzt die Meldung READY auf dem Monitor. DOS eintasten und RETURN-Taste drücken. Falls keine Cassette verwendet wird, zeigt sich das DOS MENU automatisch.
6. Sobald die Bereitschaftsmeldung "SELECT ITEM OR RETURN FOR MENU" erscheint, H tasten und RETURN-Taste drücken, um die Funktion "WRITE DOS FILES" anzuwählen.
7. Wenn die Meldung "DRIVE TO WRITE FILES TO?" kommt, eine formatierte Diskette in Laufwerk 2 einschieben und den Verschluß zudrücken. Dann 2 eintasten und RETURN-Taste drücken.

Bei Erscheinen der Meldung "TYPE Y TO WRITE DOS TO DRIVE 2", Y tasten und danach RETURN-Taste drücken.
Die Meldung "WRITING NEW DOS FILES" kommt jetzt ins Bild.
Wenn das DOS Menu und die Meldung "SELECT ITEM OR RETURN FOR MENU" erscheint, ist die Systemdiskette bespeichert.

erkung: Wir empfehlen auch hier dringend, eine MEM.SAV- Ablage erstellen (s. unten). MEM.SAV bereitet eine bestimmte Anzahl Sektoren für das Ablegen des gerade im RAM befindlichen gramms vor, wodurch mehr Platz für den Einsatz der DUP.SYS- age geschaffen wird (vergl. Abschnitt über DUP.SYS).
Jetzt die soeben erstellte Systemdiskette aus Gerät 2 herausnehmen und in Gerät 1 einlegen. Die Masterdiskette sorgfältig in die äußere Schutzhülle legen und aufbewahren (vergl. Abschnitt über Aufbewahrung von Disketten).
Jetzt N eintasten und RETURN-Taste drücken, um eine MEM.SAV-Ablage auf der neuen Systemdiskette zu erstellen.
Wenn die Bereitschaftsmeldung "TYPE Y TO CREATE MEM.SAV" erscheint, Y eingeben und RETURN-Taste bedienen.
Wenn die Bereitschaftsmeldung " SELECT ITEM OR RETURN TO MENU" erscheint, ist die MEM.SAV-Ablage auf der neuen Diskette gespeichert.
DOS II 16 K bytes vom Gesamtspeichervorrat verbraucht, ist es ht ratsam, DOS II auf jede Diskette zu bringen. Man sollte die kette in dieser Reihenfolge einlesen:
Anwahl über eine Systemdiskette,
Ersetzen der Systemdiskette durch die jeweils gewünschte Programmdiskette.
tung: Wird die Anlage aus irgendeinem Grunde ausgeschaltet, so man jedwede Programmdiskette, die kein DOS II enthält, ächst herausnehmen und die Systemdiskette einschieben, um die eitsfolge auf dem Computer wieder in Gang zu bringen.

SCHREIBSCHUTZ AUF DER DISKETTE

reibschutz bedeutet einfach, daß versehentliches Überschreiben htiger Daten auf der Diskette verhindert werden soll.

man sehen kann, hat die Masterdiskette von DOS II auf der ken Seite der Hülle keine Nut, so daß es unmöglich ist, agen auf sie aufzubringen. Insofern ist sie bereits mit reibschutz versehen. Leere und vorformatierte Disketten haben se Nut links auf der Hülle, sodaß sie beschrieben werden nen. Normalerweise besteht keine Notwendigkeit, eine temdiskette mit Schreibschutz zu versehen, da hierdurch der ck der MEM.SAV-Ablage durchkreuzt würde, das im RAM indliche Programm notfalls auf die Diskette zu übernehmen. Es n daher sein, daß man die aufzuhebenden Daten lieber auf eine ere Diskette übernehmen will, die man dann mit Schreibschutz sieht. Wie versieht man wertvolle Disketten mit Schreibschutz? Jeder Diskettenschachtel von ATARI befindet sich ein Blatt mit Ben Kennzeichnungs-Etiketten und ein zweites Blatt mit kleinen bstklebenden Schreibschutzkappen.

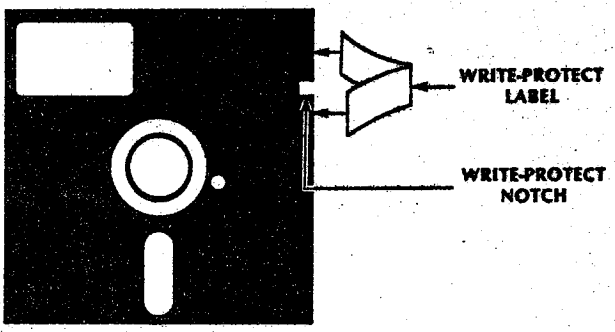


Figure 2-2 Write-Protecting a Diskette

Der Schreibschutz wird einfach dadurch erreicht, daß man eine der Klebekappen über die Nut in der Diskettenkante legt und dort festklebt (vergl. Abbildung 2-2).
Benutzt man das Doppelplattenserät ATARI 815, kann man auch den Schreibschutzschalter (WRIT PROT) am Laufwerk benutzen. Dieser Schalter leuchtet rot, wenn er eingeschaltet ist und verhindert das Hinzufügen, Ändern, Umbenennen oder Löschen von Ablagen. Ist der Schreibschutz nicht mehr gewünscht, so sollte man unbedingt daran denken, ihn abzuschalten. Beim Versuch, eine schreibgeschützte Diskette zu bespeichern, erscheint die Fehlermeldung 144 auf dem Monitor.

DAS ETIKETTIEREN VON DISKETTEN

Hierfür stehen die selbstklebenden Etiketten zur Verfügung.
Man sollte sie vor dem Aufkleben beschriften, damit die Diskette nicht beschädigt wird. Das Etikett wird üblicherweise rechts oben auf die Diskettentasche aufgebracht.
Speichern auf Disketten mit einfacher und mit doppelter Dichte.
Der prinzipielle Unterschied zwischen den Geräten 810 und 815 (doppelt) besteht in der Art, wie Informationen auf ihnen codiert werden:

a) ATARI 810:
Die Information wird hier in Blöcken zu 128 Bytes (mit einfacher Dichte) auf die Platte übertragen. Jeder Block belegt einen Sektor der Diskette.
b) ATARI 815:
Hier wird die Information in Blöcken zu 256 Bytes (mit doppelter Dichte) übertragen, und jeder Block belegt einen Sektor der Diskette.

DOS II GIBT ES IN ZWEI VERSIONEN

a) 2.OS zum Aufzeichnen von Daten mit der 810 mit Einfachdichte.
b) 2.OD zum Aufzeichnen von Daten mit der 815, dem Doppelserät, mit doppelter Dichte.

WELCHE DISKETTEN WÄHLT MAN

Um Disketten-Operationen erfolgreich durchführen zu können, muß man die richtige Version von DOS II für die Plattengeräte und die richtigen Leerdisketten für die Datenaufzeichnung verwenden. Die Tabelle in Abbildung 2-3 zeigt die für das jeweilige Plattengerät zu wählende Diskettenart:
Aus Abbildung 2-3 ersieht man, daß für das Plattengerät 810 nur die Masterdiskette 2 (CX8104) geeignet ist, für die Speicherung von Programmen und Daten kann man wählen zwischen der 810-Leerdiskette (CX 8100), der 810/815-Leerdiskette (CX8202) und der formatierten Diskette II von 810.

DISK DRIVE	CX8104 ATARI 810 MASTER DISKETTE	CX8111 ATARI 810 FORMATTED DISKETTE	CX8100 ATARI 810 BLANK DISKETTE	CX8202 ATARI 810/815 BLANK DISKETTE	CX8201 ATARI 815 MASTER DISKETTE
ATARI 810	X	X	X	X	
ATARI 815				X	X

Figure 2-3 Correct Diskettes for Your Disk Drive

enn man ein oder mehrere 815-Doppelgeräte hat, muß man die 815er asterdiskette (CX8201) mit doppelt dichter Version von DOS II enutzen, sowie die 810/815-Leerdiskette (CX8202), die den eräten beigegeben werden. Da die CX8202-Diskette völlig leer st, muß sie jeweils zuerst formatiert werden, wenn man sie zum uplizieren der Masterdiskette benutzen will (vergl. Abschnitt ber das Formatieren).

enn man sowohl das Gerät 810, sowie das Gerät 815 einsetzt, muß an die Masterdiskette für 815 (CX8201) wählen und sie in das ach 1 des Doppelgerätes 815 einschieben. Stets prüfen, ob jedes lattenfach seine eigene Codeziffer hat!

as Gerät 810 soll bei dieser Kombination mit 3 beziffert werden.

DAS AUFHEBEN VON DISKETTEN

a die Platten biegsam sind, können sie leicht verletzt werden. an beachte folgende Regeln:

- Disketten, die nicht benutzt werden, immer in die Papiertasche einlegen,
- die Disketten aufrecht abstellen, so wie man es mit Schallplatten tut, nicht flach aufeinander legen!

c) Disketten stets mindestens 20 Zentimeter entfernt vom TV-Gerät bzw. von irgendwelchen Quellen magnetischer Felder aufbewahren!

d) Nicht in der Nähe von Hitzequellen lagern!

Die Disketten sind ein wichtiger und wertvoller Teil der Gesamtrechenanlage. Wenn sie sorgfältig behandelt werden, arbeiten sie viele Stunden lang zuverlässig und problemlos.

DIE BENUTZUNG VON DOS II

In diesen und den folgenden Abschnitten soll erklärt werden, wie man Ablagen erstellt und mit ihnen arbeitet.

Zunächst: die Kennzeichnung von Ablagen:

Ablagen werden in zwei Klassen eingeteilt:
1. Programmablagen. Das sind Gruppen von Instruktionen, nach denen der Computer bestimmte Arbeiten ausführt.

2. Datenablagen. Sie enthalten gewöhnlich die für ein Programm benötigten Daten, jedoch nicht dessen Instruktionen. So wäre z.B. ein Namens- und Adressverzeichnis eine solche Datenablage, die jederzeit auf neuen Stand gebracht werden kann.

So wie man jemanden beim Namen nennt, muß man auch eine Informationsablage namentlich bezeichnen, um sie anrufen zu können. Der Ablage-Name auf der Diskette ist Teil der Ablagen Kennzeichnung (kurz auch FILESPECs genannt).

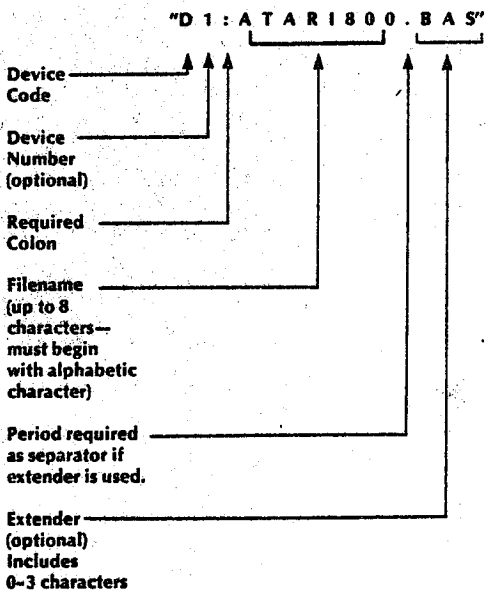
Solche FILESPECs (von "FILESPECIFICATION", vergl. Abbildung 3-1) haben 6 Schlüsselelemente.
Ruft man ein FILE mit falschem Namen an, so wird er ebensowenig reagieren, wie eine mit falschem Namen angesprochene Person. Statt dessen erscheint die Fehlercodeziffer 170 auf dem Monitor.

Für die Ablagen-Namen gelten folgende Regeln:

1. der Name darf höchstens 8 Zeichen lang sein,
2. es dürfen nur die Buchstaben A bis Z und die Ziffern 0 bis 9 benutzt werden,
3. das erste Zeichen in einem FILE-Namen muß ein Buchstabe sein,
4. die Zeichen * und ? dürfen auf keinen Fall als Namensbestandteil benutzt werden (vergl. den Abschnitt über wilde Karten).
5. Die Ablage-Namen DOS.SYS, DUP.SYS, AUTORUN.SYS und MEM.SAV sind für DOS II reserviert.

Namensergänzungen und deren Anwendung
Es dürfen den FILE-Namen bis zu drei Buchstaben große Ergänzungen angehängt werde, um die Datenart einer Ablage näher zu kennzeichnen. Dabei stehen alle zulässigen Kombinationen von Buchstaben und Ziffern zur Verfügung, z.B. :
SYS = Systemablage
BAS = BASIC-Programmablage
DAT = Datenfile
MUS = Musikprogramm-Ablage
ASM = Ablage in Assemblersprache
OBJ = Objektprogramm
SRC = Quellenprogramm
LST = Ablage, die über LIST-Befehl gespeichert wurde
SVE = Ablage, die über SAVE-Befehl gespeichert wurde.

Abbildung 3-1



Wenn man eine Ergänzung von mehr als 3 Zeichen benutzt, dann werden die zusätzlichen Zeichen vom DOS II ignoriert. Das Beispiel in Abbildung 3-2 illustriert zulässige und unzulässige Ablagebezeichnungen mit entsprechenden Erklärungen.

CASHFLOW	zulässiger Name
ATARI.BAS	zulässiger Name
3ATARI.DAT	unzulässiger Name, da erstes Zeichen nicht alphabetisch.
ATARI22.ASM	zulässiger Name
ATARI#	unzulässiger Name
A1234567.BA2	zulässiger Name
B ATARI.LST	unzulässiger Name, da er Leerschritte enthält
DOS.SYS	unzulässiger Name, da für DOS reserviert
DOSSYS	zulässiger Name
TEST1.123	zulässiger Name
ATARI.BASIC	zulässiger Name. Es ist zu beachten, daß DOS die letzten beiden Buchstaben der Ergänzung einfach weglässt.

Figur 3-2 Beispiele für zulässige und unzulässige Ablagenamen

WILDE KARTEN

ATARI DOS erkennt zwei "Wilde Karten" an, die man ersatzweise statt Zeichen in einem FILEnamen verwenden kann. Wilde Karten werden durch die beiden Sonderzeichen * und ? charakterisiert.

Das Fragezeichen ersetzt ein einzelnes Zeichen. Der Stern (*) kann für jede gültige Zeichenkombination stehen und ist daher erheblich flexibler. Die folgenden Beispiele illustrieren, wie man Sternchen und Fragezeichen einsetzen kann.

Beispiele:

*BAS	listet alle Programmdateien auf einer Diskette in Laufwerk 1, die auf BAS enden.
D2:*.*	listet alle Programmdateien auf der Diskette in Laufwerk 2.
PRO*.BAS	listet alle Programmdateien auf der Diskette in Laufwerk 1, die mit PRO beginnen und BAS als Ergänzung tragen.
TEST??	listet alle Programmdateien auf der Diskette in Laufwerk 1, die mit TEST beginnen und mit einer beliebigen Zwei-Zeichen-Kombination aufhören.

Abbildung 3-3 faßt die Wahlmöglichkeiten des DOS Menus zusammen und zeigt, ob sie den Einsatz von wilden Karten in ihren Parametern zulassen:

DOS MENU OPTION	WILD CARDS
A. Disk Directory	Yes
B. Run Cartridge	No
C. Copy File	Yes
D. Delete File	Yes
E. Rename File	Yes
F. Lock File	Yes
G. Unlock File	Yes
H. Write DOS File	No
I. Format Disk	No
J Duplicate Disk	No
K. Binary Save	No
L. Binary Load	No
M. Run at Address	No
N. Create MEM. SAV	No
O. Duplicate File	Yes

Figure 3-3 DOS Menu Options That Can and Cannot Use Wild Cards

ANWAHLFEHLER

Wenn man das System startet, können Anwahlfehler auftauchen, und zwar aus den folgenden Gründen (vergl. Abbildung 3-4):

1. weil die eingeschobene Diskette kein DOS enthält,
2. weil die Diskette falsch eingeschoben wurde,
3. weil die Diskette verkratzt, verbogen oder verschmiert ist. In diesen Fällen ist eine andere zu benutzen.
4. Weil die Diskette doppelte Dichte hat und das Laufwerk einfache Dichte erfordert, bzw. umgekehrt.

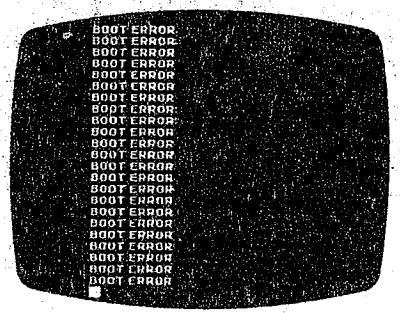


Figure 3-4 Boot Errors

Die folgenden Bedingungen ergeben ebenfalls einen Anwahlfehler, ohne daß jedoch eine Fehlermeldung auf dem Monitor erscheint:

1. wenn das Laufwerk nach der Computerkonsole eingeschaltet wurde,
2. wenn das Laufwerk nicht richtig mit der Konsole verbunden ist,
3. wenn der Hauptverbindungsstecker sich gelockert hat,
4. wenn der Adapterstecker in der PWR-Steckdose des Laufwerks sich gelockert hat,
5. wenn die Codeeinstellung des Laufwerks nicht stimmt.

Wenn nach genauer Prüfung sich keiner dieser Fehler zeigt, geht man wie folgt vor:

1. man schiebt die Masterdiskette oder Systemdiskette in Laufwerk 1 und wählt das System neu an.

2. man nimmt die Masterdiskette heraus und hebt sie sorgfältig auf,
3. man schiebt die problematische Diskette wieder ein und speichert alle erfaßbaren Ablagen auf einer anderen Diskette mit Hilfe des Kopierverfahrens ab (vergl. die Wahlmöglichkeit C.COPYFILE in Abschnitt 4).

4. Mit der problematischen Diskette in Laufwerk 1 löscht man dann alle Ablagen mit der Funktion DELETE FILE (S).
5. Jetzt versucht man, diese Diskette erneut zu benutzen. Gelingt das nicht, muß man sie neu formatieren.

Anmerkung: Wenn es sich um eine Diskette CXS111 für das Gerät 810 handelt, sollte dieser Schritt nur der allerletzte sein, da die verbesserte Formatierung möglichst nicht verlorengehen soll.

6. Wenn die Neuformatierung versagt, enthält die Diskette fehlerhafte Sektoren und muß ausgeschieden werden.

DAS ABSPEICHERN, EINLESEN UND ABARBEITEN VON PROGRAMMEN

Nach Einrichten der Systemdiskette, bzw. Formatieren einer Leerdiskette kann man die jeweiligen eigenen Programme aufzeichnen. Bei Abschaltung des Computers selbst verliert man alle im Zentralspeicher enthaltenen Programme, mit einer Plattenanlage hat man jedoch die Möglichkeit, Programme zu bewahren und zurückzuholen, ohne daß man sie neu eintasten muß. Die folgenden BASIC-Befehle benutzt man, um Programme abzustellen, bzw. wieder einzugeben. Wir haben zur Illustration ein Musterprogramm wiedergegeben, das der Benutzer eintasten und schrittweise auf Diskette abspeichern, bzw. wieder einlesen kann. Die erste Gruppe von Instruktionen gilt für ATARI 810-Laufwerk, die zweite für das 815er Doppelgerät.

MANIPULATIONEN BEI LAUFWERK 810:

1. schalte das Laufwerk ein
2. schiebe die Systemdiskette in Fach 1,
3. schalte die Computerkonsole und den Monitor an,
4. sobald die BUSY-Lampe erlischt, die Systemdiskette herausnehmen und eine formatierte, neue Diskette einschieben.
5. Das Programm von Abbildung 3-5 eintasten,
6. SAVE"D:INTEREST.SAV" tasten und RETURN-Taste drücken,
7. sobald die BUSY-Lampe ausgeht und das Signal READY erscheint, ist das eingetastete Programm auf der Diskette abgestellt.
8. NEW eintasten und RETURN-Taste drücken, um das Programm im RAM zu löschen.

Zur Neueinlesung des Programms geht man so vor:

9. LOAD "D:INTEREST.SAV" tasten und RETURN-Taste drücken,
10. Wenn das Bereitschaftszeichen READY auf dem Monitor erscheint, kann das Programm durch Tasten von RUN und RETURN-Taste gestart werden.

11. Man kann das Programm auch durch Eingabe von RUN"D:INTEREST.SAV", sowie RETURN anlaufen lassen.

Anmerkung: lösche nicht das Musterprogramm von der Diskette, es wird in Abschnitt 5 wieder gebraucht!

```
100 REM *** INTEREST
110 PRINT "IF YOU TYPE THE AMOUNT OF P
RINCIPLE"
120 PRINT "AND THE INTEREST RATE PER Y
EAR, I WILL"
130 PRINT "SHOW YOU HOW YOUR MONEY GRO
WS, YEAR BY"
140 PRINT "YEAR, TO STOP ME, PRESS THE
BREAK KEY."
150 PRINT
160 PRINT "PRINCIPAL";
165 INPUT P
170 PRINT "INTEREST RATE";
175 INPUT R
180 LET N=1
190 PRINT
200 LET A=P*(1+R/100)^N
210 PRINT "YEAR = ";N
220 PRINT "AMOUNT =";A
230 LET N=N+1
240 GOTO 190

READY
```

Figure 3-5 Sample Interest Program

MANIPULATIONEN BEI LAUFWERK 815

1. Laufwerk einschalten
2. Systemdiskette in Fach 1 einschieben und eine formatierte Diskette in Fach 2.
3. Computerkonsole und Monitor einschalten.
4. das Programm von Abbildung 3-5 eintasten.
5. SAVE"D2:INTEREST.SAV" und RETURN tasten.
6. Sobald die BUSY-Lampe ausgeht und das Bereitschaftssignal READY auf dem Monitor erscheint, ist das eingetastete Programm richtig auf der Platte in Fach 2 gespeichert.
7. NEW und RETURN tasten, um das Programm im RAM zu löschen. Jetzt kann das abgespeicherte Programm wieder eingelesen werden:

8. LOAD"D2:INTEREST.SAV" und RETURN tasten.
9. Das Programm ist jetzt arbeitsbereit. RUN und RETURN tasten.

10. Man kann das Programm auch durch Eingabe von RUN"D:INTEREST.SAV" und RETURN-Taste laden und starten.

Anmerkung: Dieses Programm sollte nicht von der Diskette gelöscht werden, da es in Abschnitt 5 noch gebraucht wird.

WAHL EINER MOEGLICHKEIT DES DOS MENUS

Um eine Option des DOS Menus zu wählen, geht man so vor:

1. DOS eintasten und RETURN-Taste drücken.
2. auf dem Monitor erscheint das Menu mit den 15 Optionen.
   Vergl. hierzu Abbildung 1-4 'das DOS-II-Menu".

3. Den Wahlbuchstaben eintasten und RETURN drücken.
4. Danach erscheint eine Bereitschaftsmeldung mit der Aufforderung der Parameter, die zur Ausführung der gewählten Funktionen nötig sind. Ein Parameter ist eine zusätzliche Information (manchmal wahlweise), die spezifiziert, wie der Befehl ausgeführt werden soll.

5. Die Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU erscheint immer dann, wenn der Computer eine Aufgabe abschließt. Wenn man ein neues Unterprogramm wählen will, tastet man den entsprechenden Kennbuchstaben ein und drückt die RETURN-Taste. Die untere Hälfte des Monitorbildes schiebt sich daraufhin nach oben und macht Platz für die nächste Bereitschaftsmeldung. Wenn man jetzt RETURN drückt, wird der Monitor gelöscht und zeigt dann wieder das DOS Menu.

A. DAS DISKETTENVERZEICHNIS (DISK DIRECTORY)

Das Diskettenverzeichnis enthält eine Liste aller Ablagen auf einer Diskette. Auf entsprechenden Befehl gibt es die FILE-Bezeichnungen wieder und (falls vorhanden) deren Zusatzbezeichnungen, sowie die Anzahl der für jedes Programm belegten Sektoren.

Die Liste wird entweder vollständig oder nur teilweise gezeigt, je nach den eingegebenen Parametern. In diesen Parametern können auch s.g. wilde Karten verwandt werden.

Taste A ein und drücke RETURN-Taste, nachdem die Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU erschienen ist. Der Monitor zeigt sofort danach die Meldung für die Eingangsfunktion: DIRECTORY-SEARCH SPEC, LIST FILE.

Wenn man danach wieder RETURN drückt, wird eine Listung aller FILE-Namen auf der Diskette gezeigt, die Sektorenzahl pro Ablageprogramm, sowie die Zahl der freien Sektoren auf der Diskette. Das folgende Beispiel zeigt die Ablage, die das Verzeichnis der Systemdiskette für DOS II enthält:

```
SELECT ITEM OR RETURN FOR MENU
A RETURN
DIRECTORY-SEARCH SPEC, LIST FILE?
RETURN
RETURN

DOS SYS 039 (ATARI 810 Disk Drive)
```

DUP SYS 042 (ATARI 810 Disk Drive)
DUP SYS 021 (ATARI 815 Dual Disk Drive)

MEM SAV 045 (ATARI 810 Disk Drive)
MEM SAV 022 (ATARI 815 Dual Disk Drive)

581 FREE SECTORS (ATARI 810 Disk Drive)
626 FREE SECTORS (ATARI 815 Dual Disk Drive)

SELECT ITEM OR RETURN FOR MENU

FÜR DIE ANWAHL DES PLATTENVERZEICHNISSES

man aus der Eingangsmeldung für das Plattenverzeichnis ieht, hat diese Funktion zwei Parameter, nämlich je einen für IRCH SPEC und LIST FILE.

n man für sie keinen spezifischen FILE-Namen (FILESPEC) eitstellt, setzt das DOS automatisch für beide die Ersatzwerte *.*,E: ein. Der erste Ersatzparameter, D1:*.*, weist das DOS daß es alle Ablagenamen mit den zugehörigen Sektorenlängen gen soll, die auf der zur Zeit in Fach 1 liegenden Diskette hen.

diesem Punkt kann man wählen, ob man eine Einzelablage, rere Ablagen oder alle Ablagen auf der angewählten Diskette en will. Wird keine bestimmte Diskette angesprochen, nimmt DOS an, daß die in Fach 1 enthaltene Diskette gemeint ist (Fach 1 also das automatisch angewählte Fach).

zweite selbstgewählte Parameter, nämlich E:, zeigt dem DOS, die gesamte Information ins Bild kommen soll. Wenn man also keinen Parameter spezifiziert und nur RETURN drückt, dann tet das DOS auf dem Monitor alle FILE-Bezeichnungen und FILE-torenzahlen, die überhaupt auf der Diskette in Fach 1 halten sind.
ls man einen Drucker angeschlossen hat, kann man eine dauernde ierung des Verzeichnisses erhalten, indem man P: für den iten Parameter einsetzt. Im Beispiel unten werden die Daten eines FILEs gedruckt, DOS.SYS.
Taste A und drücke RETURN,
taste, nach der Bereitschaftsmeldung für das Verzeichnis, DOS.SYS,P: und drücke RETURN-Taste.
Wenn ein Drucker angeschlossen und eingeschaltet ist, wird ein Teilverzeichnis statt auf dem Monitor über den Drucker ausgegeben.
dem Monitor, bzw. Drucker erscheint:

.SYS 039 (bei einfacher Dichte),

kein Drucker angeschlossen, bzw. eingeschaltet, so zeigt der itor den Fehlercode 138.

esmal wenn die Funktion "Diskettenverzeichnis" des DOS II eine gabe erledigt hat, zeigt sich die Bereitschaftsmeldung SELECT M OR RETURN FOR MENU. Abbildung 4-1 illustriert verschiedene e zur Benutzung dieser Funktion (Option).

erkung: Wenn Ablagenamen projiziert werden, sind die entlichen Namen von den Zusätzen durch einen Leerschritt

SELECT ITEM OR RETURN FOR MENU
A RETURN
DIRECTORY—SEARCH SPEC, LIST FILE?
***.SYS** RETURN

Lists all files from Drive 1 diskette with .SYS extender on the screen.

SELECT ITEM OR RETURN FOR MENU

Example 2:

SELECT ITEM OR RETURN FOR MENU
A RETURN
DIRECTORY—SEARCH SPEC, LIST FILE?
D2:,P: RETURN

Lists all files on Drive 2 diskette on the line printer.

Figure 4-1

SELECT ITEM OR RETURN FOR MENU

Example 3:

SELECT ITEM OR RETURN FOR MENU
A RETURN
DIRECTORY—SEARCH SPEC, LIST FILE?
EO?.* RETURN

Lists all 3-letter filespecs from the Drive 1 diskette that begin with EO.

SELECT ITEM OR RETURN FOR MENU

B. LAUFE AUF CASSETTE (RUN CARTRIDGE)

Immer wenn man den Wählbuchstaben D tastet, schaltet die ATARI-Anlage auf die eingeschobene Cassette um. Ist die BASIC-Cassette eingelegt, so gibt der Monitor die Bereitschaftsmeldung READY.

Bei Einsatz der ASSEMBLER-EDITOR-Cassette erscheint die Meldung EDIT. Falls keine Cassette bereitgestellt wurde, meldet der Monitor NO CARTRIDGE.

Beispiel:

SELECT ITEM OR RETURN FOR MENU (Monitormeldung), taste B und drücke RETURN.
Wenn die Diskette in Laufwerk 1 eine MEM.SAV-Ablage enthält, wird das BASIC- , bzw. ASSEMBLER-Programm automatisch durch Eintasten von DOS und Drücken der RETURN-Taste auf die Diskette gespeichert und dann wieder in den Speicher geladen, wenn wieder B und RETURN-Taste angewählt wird (RUN CARTRIDGE).

Dies setzt natürlich voraus, daß die Diskette in Laufwerk 1 noch dieselbe ist, die vor dem Anruf von DOS darin war, und daß das MEM.SAV-Programm darauf nicht durch Kopier- oder Dupliziervorgänge zerstört worden ist (COPY FILE, DUPLICATE FILE oder DUPLICATE DISK). Eine besondere Meldung weist den Benutzer daraufhin, daß MEM.SAV durch solche Befehle unwirksam werden kann (vergl. Abschnitt über MEM.SAV).

Falls auf der benutzten Diskette keine MEM.SAV-Funktion vorgesehen war (Laufwerk 1), als das DOS eingelesen wurde, geht jedes etwa eingegebene BASIC-, bzw. ASSEMBLER-Programm durch die Einlesung von DOS verloren.
Das Programm kann nicht mehr zurückgewonnen werden, es sei denn, man hat es zuvor auf einem separaten Datenträger abgestellt.
Diese Programm-Überschreibungen durch DOS II ergeben sich dadurch, daß der Benutzerbereich des Speichers auch von den Dienstroutinen der Plattenanlage gebraucht wird, die im DUP.SYS-Paket enthalten sind.
Diese gemeinsame Speichernutzung mit dem DUP.SYS erhöht die verfügbare Speicherkapazität gegenüber DOS I.

C. KOPIERE ABLAGE (COPY FILE)

Diese Option (C) ist von Bedeutung, wenn man zwei Laufwerke anschließt und den Inhalt einer Platte auf eine zweite kopieren will.
Zwei Parameter gehören in diesem Fall zu dem Befehl COPY FILE: a FROM, b TO.
Der Parameter für FROM besteht gewöhnlich aus einem FILEnamen (FILSPEC), sei es mit "wilden Karten", oder ohne. Die Benutzung solcher wilden Karten im ersten Parameter liefert einen bequemen Weg, eine ganze Gruppe von FILEs von einer Platte auf die andere zu übertragen (vergl. Beispiel 6). Die Option A kann beim zweiten Parameter benutzt werden, um zwei sich ergänzende Ablagen (FILEs) aneinander zu hängen.
Der zweite Parameter ist gewöhnlich jedoch ein Filename (FILSPEC), - kann aber auch aus einer Gerätekennung bestehen, z.B. E: (Monitor), P: (Drucker) oder D: (Diskette) (vergl. Beispiele 3, 5 und 6 in Abbildung 4-2).
Die Option COPY FILE kann auch zum Erstellen einer Reservekopie von einer wichtigen Ablage verwandt werden, die auf derselben Diskette mit denselben FILEnamen, jedoch mit anderem Zusatz oder auch mit völlig verschiedenen FILEnamen gespeichert werden soll.
Wenn die Ablage, die man unter einem neuen Namen kopieren will, aus mehreren aneinandergehängten FILEs (s.g. kombinierten FILEs) besteht, so wird die Kopie dieses FILEs "komprimiert" gespeichert, d.h. daß sie weniger Sektoren beansprucht, als das Original.
Anmerkung: Beim Versuch, eine DOS.SYS-Ablage zu kopieren, wird immer ein Fehler gemeldet. DOS.SYS läßt sich ausschließlich mit der Option H.WRITE DOS.SYS aufzeichnen. Wenn man versucht, eine Ablage zu kopieren, wie die beschriebene, und sich auf der Systemdiskette ein MEM.SAV befindet, so erscheint auf dem Monitor eine neue Meldung. Sie zeigt sich, nachdem man die Nummern des Herkunftslaufwerks und die des Ziellaufwerks eingegeben hat.
Diese Meldung lautet: TYPE Y IF OK TO USE PROGRAM AREA CAUTION: A Y INVALIDATES MEM.SAV (deutsch: taste Y, wenn Programmbereich benutzt werden darf, Vorsicht, ein Y löscht das MEM.SAV!). Dadurch wird der Benutzer daran erinnert, daß DOS II, um den Kopierprozeß zu beschleunigen, womöglich den ganzen Platz beanspruchen kann, indem vielleicht gerade ein erstelltes Programm sich befindet.

Der Kennbuchstabe Y zeigt dem DOS II an, daß der Benutzer mit der Zerstörung des alten Programms im RAM, bzw. des MEM.SAV einverstanden ist.

Dagegen gibt ein N an, daß diese Bereiche nicht berührt werden sollen. DOS II benutzt dann eine viel kleinere Zone in der Anlage, um die Kopie durchzuführen. Das führt zwar zum selben Ergebnis, dauert aber wesentlich länger.
Es steht dann auch noch die Möglichkeit zur Verfügung, mit E: die Kopie nur auf den Monitor zu bringen, oder sie, mit P: , nur ausdrucken zu lassen.

Man achte auf zwei Dinge besonders:

1. Daß man keine einzeln gekennzeichneten BASIC FILEs aneinanderhängen darf, d.h. solche, die mit dem SAVE-Befehl gespeichert wurden. Denn jedes gekennzeichnete FILE besitzt eine eigene Symboltafel usw. und nur das Erste der FILEs würde in diesem Fall geschrieben. Dagegen kann man durchaus zwei BASIC-Ablagen verbinden, die per LIST-Befehl aufgezeichnet worden waren, oder auch zwei binäre FILEs, die mit dem ASSEMBLER -EDITOR-Band oder mit DOS II produziert worden sind. Gekennzeichnete und nichtgekennzeichnete FILEs werden in Teil 5 erklärt.

2. Bedenke man, daß bei Verknüpfungsoperationen mit FILEs eine gegenseitige Anpassungsstörung entstehen kann, wenn die Ablagen mit LIST produziert wurden und Zeilenangleichungs-Codes besitzen (MATCHLINE NUMBERS).

Example 1:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
D1:DOSEX. BAS, D2:DOSEX.BAS RETURN

SELECT ITEM OR RETURN FOR MENU
```

Copies DOSEX.BAS from D1 to D2.

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
D1:DOSEX.BAS,D1:DOSEX.BAK

SELECT ITEM OR RETURN FOR MENU
```

Creates backup copy of file on same diskette.

Example 3:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
D1:DOSEX.LST,E: RETURN

SELECT ITEM OR RETURN FOR MENU
```

Displays the program listing on screen.

Example 4:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
E:,D1:TEMP.DAT RETURN

PETER RETURN
BILL RETURN
RAY RETURN
STEVE RETURN
RETURN 3
SELECT ITEM OR RETURN FOR MENU
```

Copies any succeeding data into a file named TEMP.DAT. Type data on screen that you want to be stored in TEMP.DAT file

Terminates entry of data.

Example 5:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
D1:DISEX.LST,P: RETURN

SELECT ITEM OR RETURN FOR MENU
```

Lists the program listing DISEX.LST on the printer.

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
*.*,D2: RETURN
```

Copies all files from D1 to D2 except those having .SYS extender.

```
SELECT ITEM OR RETURN FOR MENU
```

Example 7:

```
SELECT ITEM OR RETURN FOR MENU
C RETURN
COPY—FROM, TO?
D1:PROG2,PROG1/A RETURN
```

Appends PROG2 file on D1 to the PROG1 file.

```
SELECT ITEM OR RETURN FOR MENU
```

D. LöSCHE ABLAGE (DELETE FILE)

Mit Hilfe dieser Option lassen sich Ablagen von einer Diskette und vom Plattenverzeichnis (DISK DIRECTORY) löschen. In den FILEnamen (FILESPECs) dürfen dabei "wilde Karten" benutzt werden. Anmerkung: Mit DOS II lassen sich keine FILEs auf solchen Disketten löschen, die mit DOS I formatiert worden sind. In solchen Fällen muß zur Löschung DOS I verwendet werden. Die Prüfungsmeldung gibt dem Benutzer nocheinmal Gelegenheit, den Entschluß zum Löschen zu überdenken. Durch Anfügen des Hinweises N (No Verification request) an den Eingangsteil des FILEnamens kann man diesen Programmteil überspringen. Man kann auch alle Ablagen auf einer Diskette löschen, ohne daß deren Formatierung verloren geht. Beispiel 4 illustriert die Schritte zum Löschen aller Ablagen von der Diskette in Fach 1. Es ist zu beachten, daß hierbei die /N-Option zwar benutzt wird, daß die Prüfungsfrage jedoch nicht für jede einzelne Ablage auf der Diskette beantwortet werden muß. Wenn man versucht, eine gesperrte Ablage zu löschen, erscheint Fehlermeldung 167 (FILE gesperrt) auf dem Monitor. Wer formatierte Disketten II für ATARI 810 besitzt, kann ohne Schwierigkeiten alle Informationen von der Diskette nehmen, ohne die verbesserte Formatierung darauf zu zerstören.

E. NEUBEZEICHNUNG DER ABLAGE (RENAME FILE)

Durch diese Option lassen sich Ablage-Bezeichnungen (FILE NAMES) ändern. Die Funktion enthält zwei Parameter, OLD NAME und NEW. Der Parameter OLD NAME besteht immer aus einem vollständigen FILEnamen (FILESPEC). Wenn keine Gerätekennzahl genannt wird, nimmt der Computer an, daß Diskette 1 gemeint ist (automatische Anwahl). Der Parameter für NEW bezieht sich einfach auf den neuen FILEnamen. Dabei gilt als Gerätekennzahl immer dieselbe, wie die beim alten Namen angegebene.

Achtung: Keinesfalls Ablagen auf einer DOS II-Diskette mit Hilfe von DOS I umbenennen! Ganz allgemein sollte niemals DOS I für DOS II-Disketten verwandt werden.

Wenn man versucht, einen FILEnamen auf einer schreibgeschützten Diskette zu ändern, erscheint Fehlermeldung 144 (Gerät blockiert) auf dem Monitor. Wenn man einen Namen ändern will, der garnicht auf der Diskette steht, wird Fehler Nummer 170 (FILE nicht gefunden) angezeigt.

Wenn sich Fehler Nummer 167 zeigt, so deutet das daraufhin, daß der Benutzer einen gesperrten FILE umbenennen wollte (vergl. F.LOCK FILE).

Example 1:

```
SELECT ITEM OR RETURN FOR MENU
D RETURN
DELETE FILESPEC
D2:REM*.BAS RETURN                    All files that begin with REM
                                      and that have a .BAS extender.
TYPE "Y" TO DELETE...                 Verification prompt.
REM1.BAS!                             Deletes REM1.BAS.
Y RETURN
REMBAA. BAS                           Deletes REMBAA.BAS.
Y RETURN
SELECT ITEM OR RETURN FOR MENU
```

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
D RETURN
DELETE FILESPEC
D: TEMP.DAT RETURN                    A single file.
TYPE "Y" TO DELETE...                 Verification prompt.
TEMP.DAT
N RETURN                              If Y is typed, file will be
                                      deleted.
SELECT ITEM OR RETURN FOR MENU
```

Example 3:

```
SELECT ITEM OR RETURN FOR MENU
D RETURN
DELETE FILESPEC
DOXEX.BAS/N RETURN                    File will be deleted without
                                      requesting verification.
SELECT ITEM OR RETURN FOR MENU
```

Example 4:

```
SELECT ITEM OR RETURN FOR MENU
D RETURN
DELETE FILESPEC
*.*/N RETURN                          Deletes all files from the
                                      Drive 1 diskette.
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-3 Using the Delete File Option

Example 1:

```
SELECT ITEM OR RETURN FOR MENU
E RETURN
RENAME, GIVE OLD NAME, NEW
D2: TEMP.DAT,NAMES.DAT RETURN

SELECT ITEM OR RETURN FOR MENU
```

Changes the file on Drive 2 from TEMP.DAT to NAMES.DAT.

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
E RETURN
RENAME, GIVE OLD NAME, NEW
*.8KB,*.BAS RETURN

SELECT ITEM OR RETURN FOR MENU
```

All files with extender 8KB have their extenders changed to .BAS

Figure 4-4 Using the Rename File Option

Example 1:

```
SELECT ITEM OR RETURN FOR MENU
F RETURN
WHAT FILE TO LOCK?
DOS.SYS RETURN
SELECT ITEM OR RETURN FOR MENU
```

Locks the DOS.SYS File.

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
F RETURN
WHAT FILE TO LOCK?
D1:*.BAS RETURN

SELECT ITEM OR RETURN FOR MENU
```

Locks all files on D1 with an extender of .BAS.

Example 3:

```
SELECT ITEM OR RETURN FOR MENU
F RETURN
WHAT FILE TO LOCK?
T*.* RETURN

SELECT ITEM OR RETURN FOR MENU
```

Locks all files on D1 that begin with T

Example 4:

```
SELECT ITEM OR RETURN FOR MENU
F RETURN
WHAT FILE TO LOCK?
*.* RETURN

SELECT ITEM OR RETURN FOR MENU
```

Locks all D1 files.

Figure 4-5 Using the Lock File Option

F. SPERRE ABLAGE (LOCK FILE)

Man setzt diese Option ein, um eine einzelne Ablage mit Schreibschutz zu versehen. Ein so gesperrtes FILE kann nicht bespeichert, angehängt, umbenannt oder gelöscht werden. Jeder Versuch, dies doch zu tun, löst die Fehlermeldung 167 aus. Mit Hilfe wilder Karten kann man mehrere FILEs gleichzeitig sperren. Eine gesperrte Ablage ist im Plattenverzeichnis (DISK DIRECTORY) mit einem Stern (*) versehen. Diesen Stern darf man nicht mit dem Stern der wilden Karten verwechseln! Durch Formatieren einer Diskette werden auch gesperrte FILEs auf dieser Diskette zerstört, denn die Formatierung ignoriert den LOCK FILE-Befehl.

G. ENTSPERRE ABLAGE (UNLOCK FILE)

Mit dieser Wahlmöglichkeit können zuvor mit F. gesperrte Ablagen wieder entsperrt werden.
Nach Durchführen der Entsperrung erscheint der zuvor als Sperrsignal beigefügte Stern nicht mehr vor den entsprechenden FILEnamen, sobald der nächste Befehl A (Plattenverzeichnislisten) gegeben wird. Wilde Karten können in den FILESPECs benutzt werden.

SELECT ITEM OR RETURN FOR MENU
G RETURN
WHAT FILE TO UNLOCK?
DOSEX.BAS RETURN

Unlocks DOSEX.BAS file on D1.

SELECT ITEM OR RETURN FOR MENU

SELECT ITEM OR RETURN FOR MENU
G RETURN
WHAT FILE TO UNLOCK?
PROB?.DAT RETURN

Unlocks all 5-letter files beginning with PROB and having a .DAT extender.

SELECT ITEM OR RETURN FOR MENU
G RETURN
WHAT FILE TO UNLOCK?
T* * RETURN

Unlocks files beginning with the letter T on Drive 1.

SELECT ITEM OR RETURN FOR MENU

H. SCHREIBE DOS-ABLAGE (WRITE DOS FILE)

Um DOS II (bestehend aus DOS.SYS und DUP.SYS) auf einer Diskette zu speichern, muß diese zuvor mit Hilfe von DOS II formatiert worden sein (vergl. Option I. FORMAT DISC), es sei denn, man benutzt eine vorformatierte Diskette II für ATARI 810 (CX8111). Die mit DOS II zu beschreibende Diskette kann in jedes beliebige Fach eingeschoben werden.

Anmerkung: Wie gesagt, kann mit DOS II keine Diskette beschrieben werden, die mit Hilfe von DOS I formatiert wurde. Entsprechend sollte DOS I nur dann für die Option WRITE DOS FILE eingesetzt werden, wenn es um die Beschriftung einer mit DOS I formatierten Diskette geht. Obwohl wir davon abraten, DOS I und DOS II zu vermengen, soll hier darauf hingewiesen werden,

daß per DOS I immerhin das DOS I-Paket selbst auch auf eine mit DOS II formatierte Diskette aufgebracht werden kann. Achtung: Niemals sollte man mit der Option WRITE DOS FILE auf eine Diskette schreiben, die DOS II enthält! Bei der Arbeit mit Disketten, die teils mit DOS I, teils mit DOS II formatiert worden sind, sollte man DOS II benutzen, um sich vor Fehlern zu schützen, die wertvolle Disketten beschädigen könnten. Sobald die DOS-Ablagen auf die Diskette gebracht sind (vergl. Abbildung 4-7), wird der Monitor gelöscht und es erscheint anschließend sowohl das Menu, wie die Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU. Wenn man versucht, eine DOS-Ablage auf eine schreibgeschützte Diskette zu bringen, ergibt sich die Fehleranzeige 144. Ebenso ergeht eine Fehlermeldung, wenn versucht wird, eine neue DOS-Ablage auf eine Diskette in einem 815-Gerät mit eingeschaltetem WRIT PROT-Signal zu bringen.

```
SELECT ITEM OR RETURN FOR MENU
H
DRIVE TO WRITE DOS FILES TO?
1
TYPE "Y" TO WRITE DOS TO DRIVE 1.
Y
WRITING NEW DOS FILES

SELECT ITEM OR RETURN FOR MENU
```

Figure 4-7 Using the Write DOS File Option

I. FORMATIERE DISKETTE (FORMAT DISKETTE)

Diese Option dient, wie der Name sagt, zum Formatieren von Disketten. Eine Diskette kann leer sein, oder schon Daten tragen, die überschrieben werden sollen. Das Formatieren bringt ein digitales Muster auf die Disketten, mit dessen Hilfe Daten auf dieser gespeichert und wiedergefunden werden können. Das Formatieren einer Diskette nimmt etwa zwei Minuten auf dem Plattengerät 810 und zweieinhalb Minuten auf dem Doppelgerät 815 in Anspruch. Das Beispiel in Abbildung 4-8 zeigt Laufwerk 1 als dasjenige, in dem formatiert werden soll, jedoch kann auch jedes andere Laufwerk angesprochen werden. Eine Diskette mit schadhaften Sektoren läßt sich nicht formatieren. Ergeht eine Meldung über schadhafte Stellen an das DOS II, so versucht das System zwei Mal die Formatierung durchzuführen. Erst nach diesen Versuchen, die bis zu fünfzehn Minuten in Anspruch nehmen können, ergeht, falls sie erfolglos waren, eine Fehlermeldung 173. Eine neue Diskette mit schadhaften Stellen sollte man an den Hersteller zurückschicken. Disketten, die nicht von ATARI stammen, können unter Umständen den Anforderungen der ATARI-Laufwerke nicht genügen.

```
SELECT ITEM OR RETURN FOR MENU
!
WHICH DRIVE TO FORMAT?
1
TYPE "Y" TO FORMAT DISK 1
Y
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-8 Using the Format Disk Option

Achtung: Das Formatieren einer Diskette zerstört immer die zuvor aufgebrachten Ablagen, bzw. Formatierungen. Wenn man also eine vorformatierte Diskette für ATARI 810 II (CX 8111) formatiert, so verliert man die Vorteile der Vorformatierung.

Man sollte also in solchen Fällen lieber die Option D. wählen (lösche Ablage) (delete FILE (S)).

J. DUPLIZIERE PLATTE (Duplicate Disk)

Mit Hilfe dieses MENUS läßt sich ein genaues Duplikat von jeder beliebigen Diskette herstellen. Man kann dies Programm auch mit einem Einzellaufwerk nutzen, indem man die Quellen- und die Ziel-Diskette immer wieder manuell austauscht, bis das Duplizieren erledigt ist.

Ebenso können natürlich mehrere Laufwerke damit arbeiten, indem man Quellen- und Ziel-Diskette in zwei verschiedene Laufwerke legt und automatisch duplizieren läßt.

Das Duplizieren geht Sektor für Sektor vor sich. Die Quellen-informationen werden also nicht nur vollständig kopiert, sondern auf der Zieldiskette wieder in dieselben Sektorenpositionen gebracht wie vorher. Auch das Verzeichnis der Ausgangsdiskette wird mitkopiert. Logischerweise werden durch das Kopieren alle etwaigen, vorherigen Inhalte der Zieldiskette zerstört.

Die Quellendiskette muß mit DOS II formatiert sein, sonst kann man diese MENUwahl nicht verwenden. Der Versuch, eine DOS I-Diskette zu kopieren endet mit einer Fehlermeldung.

Als Zieldiskette kann jede beliebige, formatierte ATARI- Diskette dienen, d.h., man kann irgendeine formatierte ATARI- 810-Diskette II benutzen (CX 8111) oder irgendeine auf dem Plattensystem selbst mit DOS I oder DOS II formatierte Diskette. Nimmt man eine alte Diskette als Zieldiskette, so sollte man sicher sein, daß die dort enthaltenen Daten wertlos sind, da sie durch die Überschreibung zerstört werden.

Zu beachten ist, daß eine mit DOS I formatierte Diskette nicht duplizierbar ist und daß nicht vom Gerät 810 an Gerät 815 dupliziert werden kann. Versucht man das, so erscheint eine Fehlermeldung auf dem Monitor.

Eine annähernd genaue Kopie zwischen 815 und 810 bzw. von DOS I an DOS II ist allerdings möglich, mit dem Befehl COPYFILE, der zusätzlich die Unteroption *.* tragen muß (s.oben). In solchen Fällen sind die Daten auf beiden Disketten zwar gleich, jedoch gibt es Unterschiede in der Art, wie sie abgelegt sind.

Da es einen echten Duplikationsvorgang zwischen DOS I und DOSII bzw. zwischen den Laufwerken 810 und 815 nicht gibt, sollte man stets die BASIC- oder ASSEMBLER-Programme, die momentan im ROM

sind, abspeichern, bevor man eine Duplizierung dieser Art beginnt. Für die Funktion DUPLIKATEDISK gibt es keinen internen Puffer, wie das bei COPYFILE der Fall ist, dadurch wird MEM.SAV zerstört, wenn man dem DOS II freien Lauf gibt, d.h. wenn man es die Programmzone überschreiben läßt.

Die Option DUPLIKATEDISK benutzt nämlich stets die Programmzonen (wo gegebenenfalls BASICProgramm liegt) als Puffer für die Verschiebung der Datengruppen von der Quellendiskette auf die Zieldiskette, sofern nur ein Laufwerk benutzt wird.

DIE DUPLIKATION MIT HILFE EINES EINZIGEN LAUFWERKES

Ist nur ein Einzellaufwerk vorhanden, so ist Fach 1 sowohl Quellen-, wie Zieladresse (vergl. Abbildung 4-9).

Vorsichtshalber sollte man die Quellendiskette immer mit Schreibschutz versehen. Wenn dann die Quellendiskette versehentlich als Zieldiskette eingeschoben wird, erscheint eine Fehlermeldung 144 und die Diskette wird nicht verändert.

Wenn nach der Aufforderung TYPE "Y" IF OK TO USE PROGRAM AREA etwas anderes als Y und die RETURN-Taste gedrückt wird, so wird das Programm abgebrochen und die ursprüngliche Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU erscheint wieder auf dem Monitor.

Abbildung 4-9 zeigt, wie mit einem DISKDRIVE dupliziet wird:

```
SELECT ITEM OR RETURN FOR MENU
J RETURN
DUP DISK—SOURCE, DEST DRIVES?
1,1 RETURN
INSERT SOURCE DISK, TYPE RETURN
TYPE "Y" IF OK TO USE PROGRAM AREA?
  CAUTION: A "Y" INVALIDATES MEM. SAV
Y RETURN
INSERT DESTINATION DISK, TYPE RETURN
RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-9 Using the Duplicate Disk Option With a Single Disk Drive

Anmerkung: Wie häufig das DUP-Programm den Einschub der Quellen- bzw. der Zieldiskette anfordert, hängt davon ab, wieviele Einzelablagen ein gegebenes System enthält und wieviel RAM belegt wird. Zum Kopieren einer vollen Diskette mögen bei einem 48 K-System zwei Einschübe genügen, während ein 16 K-System fünf oder sechs Einschübe benötigen kann.

DAS DUPLIZIEREN MIT MEHRFACH DISKDRIVE

Wenn sowohl das Gerät 810 wie das 815 eingesetzt wird, so muß der Anwender die Platten sorgfältig nach Einfachdichte und Doppeldichte unterscheiden und entsprechend etikettieren. Dadurch wird der Einsatz im falschen Laufwerk vermieden.

Auch beim Mehrfachlaufwerk ist es nötig, ein etwa im ROM stehendes BASIC-Programm vorher abzustellen, da auch hier der Anwenderbereich überschrieben und das MEM.SAV ungültig gemacht wird.

Es ist zu beachten, daß die Quellendiskette stets in D1 und die Zieldiskette in D2 liegen muß (vergl. Abbildung 4-10). Welches von zwei gleichen Laufwerken man jeweils benutzt, ist gleichgültig.

Während der Duplizierung bleibt der Läufer auf dem Bildschirm. Der Vorgang kann bei fast voller Quellendiskette etliche Minuten dauern.

```
SELECT ITEM OR RETURN FOR MENU
J RETURN
DUP DISK—SOURCE, DEST DRIVES
1,2 RETURN
INSERT BOTH DISKS, TYPE RETURN
RETURN
TYPE "Y" IF OK TO USE PROGRAM AREA
  CAUTION: A "Y" INVALIDATES MEM. SAV
Y RETURN
SELECT ITEM OR RETURN FOR MENU
```

K. BINÄRES ABSPEICHERN (BINARY SAFE)

Anmerkung: Diese Instruktion wird wahrscheinlich nicht von einem ATARI-Anfänger benutzt werden. Wer die Bedeutung hexadezimaler Zahlen nicht kennt, bzw. einiges Wissen über ASSEMBLER-Sprache besitzt, kann die Informationen, die nach dem ersten Beispiel folgen, zunächst übergehen.
Die MENU-Wahl K wird eingesetzt, um Inhalte von Speicherstellen in der Form des Objektprogramms (binär) abzuspeichern. Programme, die mit ASSEMBLER-EDITOR-Cassette erstellt wurden, haben ebenfalls dieses binäre Format. Parameter für diese MENU-Wahl, nämlich START, END, INIT, RUN, sind hexadezimale Zahlen. Parameter START und END werden für jede binäre Ablage, bzw. jedes binäre Programm benötigt.
Die Adressen für INIT (Vorbereitung des Programms) und RUN sind wahlweise zu nennen und ermöglichen beliebige Ausführungsweisen beim Einlesen des Programms (vergl. Beispiele 2, 3 und 4).
Im unteren Beispiel wird eine Ablage mit der Bezeichnung BINFIL.OBJ und der Startadresse 3C00, sowie der Endadresse 5BFF auf einer Diskette in Fach 1 gespeichert.
Abbildung 4-11 zeigt, wie BINARYSAFE eingesetzt wird:

```
SELECT ITEM OR RETURN FOR MENU
K RETURN
SAVE—GIVE FILE, START, END, INIT, RUN
BINFIL.OBJ, 3C00, 5BFF RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-11 The Most Elementary Use of Binary Save

INFORMATIONEN ÜBER WAHLWEISE BENUTZBARE PARAMETER

Alle Binärablagen, die mit Hilfe einer Option BINARYSAFE oder mit der ASSEMBLER-EDITOR-Cassette erstellt werden, tragen am Beginn eine Kopfangabe von sechs Bytes Länge. Aus dieser Kopfangabe läßt sich unschwer die Start-, bzw. die Endadresse erkennen.

Header Byte #	Decimal Number	Hex Number	Description
#1	255	FF	Identification code for
#2	255	FF	binary load file
#3	0	00	Starting address (LSB)
#4	60	3C	(MSB)
#5	255	FF	Ending address (LSB)
#6	91	5B	(MSB)

File data segment contains
8191 (Dec) bytes of data.

Die beiden wahlweise zu benutzenden Parameter INIT und RUN bieten die Möglichkeit, ein Ablageprogramm in binärer ASSEMBLER-Sprache sofort nach dem Einlesen laufen zu lassen. Eine Ablage, die sowohl den INIT- wie den RUN-Parameter benutzt, wird "LOAD AND GO FILE" genannt. Enthält eine Ablage keine Angaben zu diesen Parametern, so spricht man lediglich von einem "LOADFILE", da sie nur eingelesen wird, jedoch erst zu arbeiten beginnt, wenn ein Befehl M. RUN AT ADRESS gegeben wird.

Im Allgemeinen definiert der Parameter der RUN-Adresse den Punkt in einem Programm, wo dessen Ausführung beginnen soll, sobald das ganze Programmpaket in den Speicher geladen ist, d.h. sobald "END OF FILE" erreicht ist. Daher kann es praktisch immer nur eine einzige Startadresse geben, auch bei einer kombinierten Programmablage. Wenn man z.B. mehrere kleine Programme mit je einer eigenen RUN- Adresse aneinander hängt, so wird nur die letzte zu ladende RUN- Adresse befolgt. ** Wenn eine INIT-Adresse angegeben ist, so wird der Programmpunkt, auf den sie hinweist, sofort ausgeführt, sobald diese Adresse selbst im RAM steht. Das gilt selbst dann, wenn sie aus mehreren LOAD AND GO FILES besteht, die verkettet sind. In solchen Fällen wird jedes Segment des FILES mit einer INIT-Adresse sofort nach dem Einlesen dieser INIT-Adressse ausgeführt. Somit würde jedes Segment dieser Art eingelesen und ausgeführt, bevor das folgende an die Reihe kommt.* Die Ausführung von Programmteilen, auf die eine INIT-Adresse hinweist, hat immer Vorrang vor solchen mit RUN-Adresse. FILES, die per ASSEMBLER-EDITOR-Cassette mit Hilfe der LOAD AND GO-Option erstellt worden sind, können in den gewünschten INIT- und RUN-Adressen des programmeigenen Codes abgespeichert werden, gefolgt von dem zu kontrollierenden Code. Die RUN-Adresse wird stets in den Registern 2E0 (LOW) und 2E1 (HIGH) (hexadezimal) gespeichert. Die INIT-Adresse steht immer in den Registern 2E2 (LOW) und 2E3 (HIGH) (hexadezimal). Man beachte, daß die INIT-Adresse sofort nach dem Einlesen ausgeführt wird, sodaß natürlich der Code auf den sie hinweist, zuvor schon im Speicher angekommen sein muß. Anmerkung: IOCB Nummernzeichen 1 ist während der Ausführung eines Codes, der per INIT angesprochen wird, geöffnet. Daher ist dies System in solchen Fällen nicht verfügbar und darf durch das Benutzerprogramm dann nicht beansprucht werden. ** Ein RTS (RETURN) am Ende eines Programms gibt die Kontrolle stets an das DOS II zurück. * Jedes Codesegment muß auf RTS(RETURN) enden, wenn das nächste Segment eingelesen werden soll, oder es muß ein Rücksprung auf DOS II erfolgen. Die binäre Abspeicherung mit wahlweise einsetzbaren Parametern. Das Beispiel in Abbildung 4-13 illustriert ein Programm in ASSEMBLER-Sprache, das eine Datenzone benutzt, die erst vorbereitet werden muß, bevor das Programm sie verwenden kann. Angenommen der Startcode reicht von 4.000 (hexadezimal) bis 41FF (hexadezimal) und das Hauptprogramm von 4.200 (hexadezimal) bis 4FFF (hexadezimal). Zum Zweck der Illustration soll angenommen werden, daß sowohl der Vorbereitungscode und das Hauptprogramm ausführbare Code enthalten und das der Vorbereitungscode auf RTS (RETURN) endet. Im folgenden Beispiel wird davon ausgegangen, daß das Programm (LAGPRG.OBJ) schon im Speicher steht.

```
SELECT ITEM OR RETURN FOR MENU
K RETURN
SAVE—GIVE FILE, START, END, INIT, RUN
LAGPRG.OBJ, 4000, 4FFF, 4000, 4200 RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-13

Folgendes passiert, wenn man dieses Programmpäckchen in den Speicher lädt: 1. der Speicher wird von 4000 bis 4FFF durch dieses Programm belegt, 2. die INIT-Adresse 4000 (hexadezimal) wird in den Registern 2E2 und 2E3 (hexadezimal) gespeichert.

3. Das Vorbereitungsprogramm, das in 4000 bis 4FFF steht, läuft ab.
4. Die RUN-Adresse 4200 (hexadezimal) wird in den Registern 2E0 und 2E1 (hexadezimal) festgehalten.
5. Das Hauptprogramm von 4200 bis 4FF beginnt zu laufen und läuft weiter bis zur Ausführung eines RETURN (RTS), bzw. bis SYSTEM-RESET oder BREAK getastet wird. Im Falle von verketteten Ablagen ist das Ergebnis komplizierter, je nach dem, wie die jetzt verbundenen FILES ursprünglich einzeln erstellt worden waren. Im nächsten Abschnitt werden einige Fälle von FILE-Kombinationen behandelt. Die Struktur einer kombinierten Ablage (COMPOUND FILE): Vor dem Betrachten des nächsten Beispiels muß man sich den Aufbau eines solchen Verbund-FILES einmal ansehen:
Er ist aus mehreren binären Ablagen zusammengesetzt. Dieser Vorgang kann auf zwei Arten erfolgt sein:
a) mit der Option C.COPYFILE plus der Anhänge-Option. Eine mit dieser Option erstellte Ablage paßt nicht an das ASSEMBLER-EDITOR-Einlese-Programm, obwohl es mit Hilfe der Option L. von DOS II (BINARYLOAD) eingegeben werden kann. Wenn eine Kompaktibilität mit der ASSEMBLER-EDITOR-Cassette gewünscht wird, kann man den zweiten Weg wählen, nämlich
b) die Option K.BINARYSAFE, die schon besprochen wurde.
Diese beiden FILE-Arten werden in Anhang 1 illustriert. Der einzige wirkliche Unterschied zwischen ihnen besteht darin, daß be Verwendung von COPYFILE (C.) der Hexadezimale Erkennungscode FFFF jedem Segment beigegeben wird.

Dagegen sind diese Codes für jedes Segment bei Anwendung von Option K. (BINARYSAFE) nicht in der endgültigen Form des FILES eingeschlossen. Diese Ablageform ist die einzige kombinierte Form, die mit der ASSEMBLER-EDITOR-Cassette zusammenpaßt.

Dagegen passen beide kombinierte FILE-Arten zur Option L. (BINARYLOAD) von DOS II.

Man muß sich nun klarmachen, was passiert, wenn eine kombinierte Ablage eingelesen wird, deren Einzelbestandteil vor der Kombination mit INIT- und RUN-Adressen versehen worden waren (d ese Adressen kann man sich am Besten als Bestandteile der Datengruppe jedes Segments vorstellen). Beispiel 3:
Es wird angenommen, daß drei Ablagen vorhanden sind, von denen jede eine RUN-ADRESSE besitzt, jedoch keine INIT-Adresse. Das Be spiel (4-14) zeigt eine Art, wie ein solches FILE erstellt werden kann:

```
SELECT ITEM OR RETURN FOR MENU
K RETURN
SAVE FILE—GIVE FILE, START, END, INIT, RUN
PART1.OBJ, 2000, 21FF,, 2000 RETURN
SELECT ITEM OR RETURN FOR MENU

SELECT ITEM
K RETURN
SAVE ITEM—OR RETURN—FOR MENU
PART2.OBJ/A, 2200, 23FF,, 2200 RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-14 Using Binary Save to Save Compound Files

Die anderen beiden Ablagen, PART2.OBJ und PART3.OBJ, die genauso erstellt werden wie PART1.OBJ, können zum Gesamtprogramm WHOLE.OBJ mit Hilfe von K.BINARYSAFE oder C.COPYFILE plus der Verbund-Option kombiniert werden. Was geschieht, wenn dieses neue FILE eingelesen wird?

1. PART1.OBJ läuft in den Speicher, wird aber nicht ausgeführt (kein INIT),
2. die RUN-Adresse für PART1.OBJ wird in 2E0 und 2E1 gespeichert.
3. PART2.OBJ läuft ein, wird aber nicht ausgeführt (kein INIT).
4. die RUN-Adresse für PART2.OBJ wird in 2E0 und 2E1 gespeichert, wodurch die RUN-Adresse von PART1.OBJ überschrieben wird.
5. PART3.OBJ wird eingelesen, aber nicht ausgeführt (kein INIT),
6. die RUN-Adresse für PART3.OBJ wird in 2E0 und 2E1 gespeichert, wodurch die RUN-Adresse von PART2.OBJ überschrieben wird.
7. Programm wird ab der RUN-Adresse von PART3.OBJ ausgeführt, da nun das Ende der Ablage (END OF FILE) erreicht ist.

Beispiel 4:
Ein anderes Beispiel für FILE-Kombination soll von einer Drei-Segment-Ablage namens BIGFILE.OBJ aussehen (Abbildng 4-15). Es wird angenommen, daß jedes Segment in einem verschiedenen Speicherbereich eingelesen wird, und das SEG1.OBJ eine INIT Adresse besitzt, aber keine RUN-Adresse, das SEG2.OBJ weder INIT- noch RUN-Adresse hat, das SEG3.OBJ eine INIT-Adresse und eine RUN-Adresse bei SEG2.OBJ hat und auf SEG1.OBJ folgend eingelesen wird.

Beim Einlesen von BIGFILE.OBJ geschieht folgendes:
SEG1.OBJ wird eingelesen.
SEG1.OBJ beginnt bei der INIT-Adresse zu arbeiten,
SEG2.OBJ wird eingelesen,
SEG3.OBJ wird eingespeichert, so daß es SEG1.OBJ überschreibt.
SEG3.OBJ beginnt bei seiner INIT-Adresse zu arbeiten.
SEG2.OBJ beginnt bei der in SEG3.OBJ angegebenen RUN-Adresse zu arbeiten.
Offensichtlich bietet also diese Option große Leistung und Flexibilität, wenn man große Ablagen erstellen will, die unmittelbar einzulesen und abzuarbeiten sind.
Beispiel 5:

Um eine existierende LOAD-Ablage in eine LOAD-AND-GO-Ablage zu verwandeln, kann man die Ablage in den Speicher laden und dann unter einem neuen Namen mit K.BINARYSAFE wieder abspeichern. Das wirft allerdings einige Probleme auf, da man gelegentlich die Endadresse, die von der Ablage belegt wird, vergißt, oder auch weil das FILE sich mit Segmenten vermischen kann, die nicht in derselben Folge auch im Speicher stehen. Daher könnte die neue Ablage mehr Platz auf der Diskette einnehmen als die alte oder ähnliches. Solche Probleme kann man durch Anwendung der im Folgenden gezeigten Prozedur vermeiden. Dieses nächste Beispiel illustriert ein LOAD-FILE auf einer RUN - Adresse bei 4.000 (hexadezimal), der in ein LOAD-AND-GO-FILE umgewandelt wird.

In Abbildung 4-15 wird ein auf FF00 (im O.S.RAM) stehendes FILE an das Ende von LOADFILL.OBJ angehängt. Da die RUN-Adresse dieses FILES dieselbe ist, an der das LOAD-FILE normalerweise auch zu arbeiten beginnt, startet dieses, sobald das gesamte angehängte FILE im RAM steht.

```
SELECT ITEM OR RETURN FOR MENU
K RETURN
SAVE FILE—GIVE FILE, START, END, INIT, RUN
LOADFIL.OBJ/A, FF00, FF00,, 4000
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-15 Converting an Existing Load-Only File to a Load-and-Go File

L. BINÄRE EINLESUNG (BINARYLOAD)

Anmerkung: Diese Instruktion wird von Anfängern gewöhnlich noch nicht angewandt. Es handelt sich um eine Wahlmöglichkeit, mit der man eine Ablage in ASSEMBLER-Sprache (binär) in das RAM lesen kann, die zuvor mit Hilfe der Option K. oder per ASSEMBLER-EDITOR-Cassette auf Platte gespeichert worden war.

Falls die RUN- oder INIT-Adresse in den Registerstellen 2E0 und 2E1 oder 2E2 und 2E3 der Ablage angehängt worden war, beginnt diese sofort nach Einlesung automatisch zu laufen. Im Fall eines LOAD-AND-GO-FILES, werden die INIT- bzw. RUN-Adressen ignoriert, falls man /N hinter dem Namen eintastet (vergl. Beispiel 1 in Abbildung 4-16). Ein solcher Fall kann mit Hilfe der RUN-AT-ADRESS-Option des MENUs abgearbeitet werde.

Ein Beispiel für den Einsatz dieser Option ohne die Unteroption /N wird im zweiten Teil von Abbildung 4-16 gegeben. Da an diesem FILE die Startadresse (in Stelle 2E0 und 2E1) angehängt worden war (vergl. Beispiel 1 für K.BINARYSAFE), beginnt dieses FILE zu arbeiten, sobald es geladen ist.

Example 1:

```
SELECT ITEM OR RETURN FOR MENU
L RETURN
LOAD FROM WHAT FILE?
MYFILE.OBJ/N RETURN
SELECT ITEM OR RETURN FOR MENU
```

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
L RETURN
LOAD FROM WHAT FILE?
BINFIL.OBJ RETURN
```

Example 3:

```
SELECT ITEM OR RETURN FOR MENU
L RETURN
LOAD FROM WHAT FILE?
MACHL.OBJ RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-16 Using the Binary Load Option

Beispiel 3 in Abbildung 4-16 zeigt eine Ablage mit dem Namen MACHL.OBJ, die weder RUN- noch INIT-Adresse besitzt. In diesem Fall wird die Bereitschaftsmeldung SELECTITIM OR RETURN FOR MENU gegeben, sobald das Einlesen des FILEs beendet ist. Um ein FILE auszuführen, das keine angehängte RUN-, bzw. INIT- Adresse besitzt, benutzt man die als nächste behandelte Option, nämlich M.RUN AT ADRESS. N. Laufe ab Adresse (RUN AT ADRESS).
Diese Instruktion wird noch nicht von Anfängern benutzt.
Man kann mit Hilfe dieser Programmwahl die hexadezimale Startadresse einer Objektprogramm-Ablage eingeben, nachdem man diese mit der Option BINARY-LOAD in den Speicher geladen hat. Die Option N. wird eingesetzt, falls die Startadresse dem Objekt-Programm-FILE nicht beigegeben ist.
In Abbildung 4-17 fängt das Programm ab der Stelle 3.000 (hexadezimal) an zu arbeiten. Man muß sehr vorsichtig sein, wenn man diese hexadezimalen Adressstellen eingibt. Wenn man eine Adresse ohne ausführbaren Code nennt, ergeben sich Schwierigkeiten. Es könnte z.B. eine Sperrung des Systems eintreten, die eine neue Anwahl des Programms nötig machen würde.

SELECT ITEM OR RETURN FOR MENU
M RETURN
RUN FROM WHAT ADDRESS?
3000 RETURN

Figure 4-17 Using the Run at Address Option

N. ERSTELLE MEM.SAV (CREATE MEM.SAV)

Mit Hilfe dieser Option kann man auf der Diskette eine Ablage erstellen, die MEM.SAV genannt wird. Diese nimmt bei jedem Anruf von DOS den Inhalt des unteren Teils der Anwenderzone des Speichers auf. Sobald man DOS eintastet und RETURN drückt, stellt der Computer das im RAM befindliche Benutzerprogramm im MEM.SAV-FILE ab, bevor er den Inhalt von DUP.SYS von der Diskette in den Speicher lädt. Nach Abschluß des DOS-Programms gibt man durch Eintasten von B. und Drücken der RETURN-Taste einfach die Kontrolle an die Programmcassette zurück, wodurch das ursprüngliche Programm automatisch wieder in den Speicher geladen wird. Ist keine Cassette eingesetzt, hat das Tasten von B. keinen Effekt. In diesem Fall muß auf die Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU geantwortet werden. Man darf keinesfalls zulassen, daß DOS den ganzen für den Anwender reservierten Speicherraum benutzt, wenn man zum Abspeichern die Funktionen COPYFILE, DUPLICATEFILE oder DUPLICATEDISK benutzt! DOS weiß nicht, ob das zu schützende Anwenderprogramm ganz oder nur teilweise im MEM.SAV aufbewahrt ist. Wenn DOS die gesamte Anwenderzone benutzt, macht es die Ablage im MEM.SAV unbrauchbar. Falls dies passiert, wird das zuvor abgestellte Programm bei der Rückkehr auf die Cassettensteuerung nicht mehr vorgefunden.

Wenn man versucht, mit Hilfe der Option N. ein MEM.SAV-FILE auf eine Diskette zu bringen, die bereits ein solches enthält, so erfolgt die Meldung MEM.SAV FILE ALLREADY EXISTS und danach die Bereitschaftsmeldung SELECT ITEM OR RETURN FOR MENU auf dem Monitor. Abbildung 4-18 illustriert die Schritte, die zum Erstellen eines MEM.SAV auf einer Diskette im Laufwerk 1 nötig sind. N.D.: MEM.SAV-FILEs können nur im Laufwerk 1 erstellt werden!

WOZU BRAUCHT MAN MEM.SAV

Diese spezielle Ablage erlaubt das vorübergehende Aufbewahren des im RAM befindlichen Programms auf einer Diskette. Um zu funktionieren, muß das MEM.SAV-Programm, das 45 Sektoren hat, auf einer Diskette im Laufwerk 1 sein. Diese darf nicht mit Schreibschutz versehen sein, wenn MEM.SAV funktionieren soll. Wenn MEM.-SAV auf der Diskette enthalten ist, dann wird die vom DUP.SYS später überschriebene Speicherzone zuvor im MEM.SAV aufbewahrt, wann immer DOS angewählt wird.

Im Prinzip wird durch dieses Verfahren der Speicherinhalt hin und her geschwenkt, was zu einer besseren Speicherausnutzung für Anwenderzwecke führt. Die Ausstauschoperation dauert etwa 21 Sekunden.

Wenn man dann auf Cassette zurückschaltet, wird das DUP.SYS-Programm seinerseits von dem aus dem MEM.SAV zurückkehrenden ursprünglichen Programm überschrieben. Das dauert ca. 7 Sekunden.

Wenn man gerade mit einem BASIC-Programm arbeitet und aus irgendeinem Grunde auf DOS zurückkehren muß, so kann man das mit MEM.SAV tun, ohne daß man das Programm mit separater Manipulation auf Diskette retten und dann wieder eingeben muß.

Wenn die Arbeit mit DOS beendet ist und das System auf die Cassette zurückgeschaltet wird, so läuft das MEM.SAV-Programm automatisch in den reservierten Speicherteil und das BASIC- Programm ebenfalls in die zugehörige Programmzone des Speichers zurück.
Ein Beispiel des Gebrauchs von MEM.SAV gibt Abbildung 4-19:

1. Taste LOAD"D:NYPROG.BAS" und drücke RETURN.
2. Lasse das Programm ausschreiben, dann taste RUN und RETURN.
3. Es funktioniert und soll durch RENAME auf eine Reserveplatte gebracht werden.
4. Taste DOS und drücke RETURN,
5. Treffe die MENU-Wahl (E. für RENAME FILE) und gebe dem zu bewahrenden Programm den Namen NYPROG.OLD.
6. Taste B und drücke RETURN, um auf BASIC zurückzuspringen. Mit Hilfe von MEM.SAV wird die modifizierte Version von NYPROG.BAS automatisch wieder in das RAM eingeladen.
7. Taste SAFE"D:NYPROG.BAS" und drücke RETURN, um das modifizierte Programm unter dem Originalnamen aufzuzeichnen.

DAS SCHREIBEN VON ASSEMBLER PROGRAMMEN MIT HILFE VON MEM.SAV

Dieses Dienstprogramm erlaubt auch das Erstellen von Programmen in ASSEMBLER-Sprache (bzw. das Einlesen binärer Daten), die sich den Anwenderbereich im RAM mit dem DUP.SYS teilen.

Das bedeutet, daß man ohne Bedenken Programme oder Daten in den Bereich von LOWMEM (das sich je nach der Anzahl der Laufwerke bzw. der FILEs, die gleichzeitig offen sein können, verschiebt) bis HIMEM (das sich entsprechend den benutzten Graphikgängen verschiebt) einlesen kann (vergl. Anhang C, Speicherplan).

Beispiel:

Angenommen, ein binäres Ablageprogramm soll eingelesen und sofort danach abgearbeitet werden: Es handelt sich um ein LOAD-AND-GO-FILE. Die RUN-Adresse ist in einem solchen Programm schon enthalten, so daß die Option RUN AT ADRESS nicht angewählt werden muß.

In einem solchen Fall ist ein MEM.SAV-FILE unnötig. Da das FILE LOAD-AND-Go ist, läuft es einfach in den Speicher und wird sofort ausgeführt. Der sicherste Weg, zu DOS zurückzukehren, ist dann die völlige Neuanwahl. Falls man das DUP.SYS-Programm während der Ausführung des Binär-FILEs nicht überschrieben hat, kann es mit einem RETURN (RTS) am Ende des Programms zurückgeholt werden. Falls das Binär-FILE während der Einlesezeit das DUP.SYS-Programm überschreibt, wird dies vom DOS festgestellt und DOS lädt nach dem RETURN am Endes des Eigenprogramms das DUP.SYS automatisch wieder ein und führt es aus.

Achtung: wenn bei Ausführung des LOAD-AND-GO-FILEs Bereiche unterhalb LO berührt werden, in denen DOS.SYS, DUP.SYS oder Teile des Operationssystems stehen, so kann es sein, daß der Computer nach Erreichen des RETURN-Befehls keine Laufanweisung mehr bekommt und das System stecken bleibt.

In einem solchen Fall muß die Netzverbindung aus- und wieder angeschaltet werden, damit die Anlage weiterarbeiten kann.

DAS EINLESEN VON BINÄRABLAGEN MIT HILFE VON MEM.SAV

Dieser Abschnitt handelt von der Einlesung eines binären Ablageprogramms, das nicht sogleich bei der Einlesung ausgeführt wird, bzw. von der Einlesung einer Ablage, die Daten für ein anderes Programm enthält.

Solange das mit LOAD einzulesende Programm keinerlei Teile des DUP.SYS-Bereichs überlagert, ist kein MEM.SAV-Programm nötig.

Wenn aber eine solche Überlagerung eintritt, so braucht man ein MEM.SAV-FILE auf Diskette 1, weil die Einlesung sonst nicht funktioniert. Ist ein MEM.SAV vorhanden, so geschieht nach Ausführung der Option L.LOAD BINARY FILE folgendes:

1. Man benutzt das LOAD BINARY FILE-Programm, um das eigene Programm einzulesen.

2. Das ursprüngliche MEM.SAV wird von der Diskette in den Speicher gelesen, wobei es das DUP.SYS überschreibt und zerstört.
3. Das Eigenprogramm wird über das ursprüngliche MEM.SAV geschrieben, und modifiziert dieses ganz oder teilweise.

4. Das neue MEM.SAV-Programm im RAM wird im MEM.SAV-Bereich der Diskette abgestellt.

5. DUP.SYS wird von der Diskette wieder in den Speicher transportiert.

6. Der Benutzer bleibt im DOS, solange er nicht eine der folgenden anderen Teilprogramme anwählt:

RUN CARTRIDGE: Hierbei wird das eigene Programm in den Speicher geladen (vom MEM.SAV) und es erfolgt ein Übergang zur Cassette mit dem BASIC-ASSEMBLER-Sprachen-Programm.

RUN AT ADRESS: Hierbei wird ebenfalls das eigene Programm vom MEM.SAV in den Speicher geladen und man beginnt mit dem an der jeweils angegebenen Adresse stehenden Programmcode zu arbeiten.

LOAD BINARYFILE: Diese Funktion dient dem Einlesen eines LOAD-AND-GO-FILEs. Dabei wird sowohl MEM.SAV als auch das neue Programm am Ende des Leseprogramms im Speicher stehen, falls das neu hinzukommende Programm zwar auch einen Teil des DUP.SYS, jedoch nicht das Originalprogramm überschreibt. Falls das neue Programm DUP.SYS überhaupt nicht berührt, endet der Ladevorgang damit, daß nur das neue Programm im Speicher steht.

Da dieses ein LOAD-AND-GO-Programm ist und unabhängig von der Berührung mit DUP.SYS eingelesen wird, bleibt man unter der Kontrolle dieses Programms, so lange bis ein RETURN (RTS) ausgeführt wird.

Anmerkung: Wenn man zwei FILEs gleichzeitig im RAM haben will, von denen eines ganz oder teilweise die DUP.SYS-Zone belegt, während das andere außerhalb dieser Zone bleibt, geht man am Besten so vor, daß man die beiden Programme verbindet und als ein Gesamtprogramm einliest.

O. DUPLIZIERTE ABLAGE (DUPLICATE FILE)

Diese Option benutzt man, wenn nur ein Plattenlaufwerk zur Verfügung steht und man eine Ablage von einer Diskette auf die andere kopieren will. Man beachte, daß ein Einzellaufwerk immer die Codeziffer 1 tragen muß.

Da nur eine Ansteuerung möglich ist, muß die Quelldiskette manuell gegen die Zieldiskette ausgetauscht werden. Falls ein FILE sehr lang ist, muß dieser Austausch mehrfach während des Duplizierprozesses erfolgen. Die Erlaubnis an DOS, Anwenderbereiche im Speicher mitzubenutzen, vermindert die Anzahl der Diskettenwechsel bei langen FILEs. In diesem Fall wird bekanntlich jedoch das MEM.SAV-FILE zerstört.

Auch "wilde Karten" sind bei dieser Option erlaubt. In Beispiel 2 wird gezeigt, daß selbst dann, wenn dem DOS Anwenderberreich im RAM zur Verfügung steht (bei einem FILE-Namen mit wilden Karten), die Programme nur eines ums andere kopiert werden. Man muß also mindestens einmal pro FILE-Namen die Disketten auswechseln.

Das 2. Beispiel zeigt den Gebrauch einer "wilden Karte" zum Kopieren von Ablagen mit fünf Buchstaben, beginnend mit TEST, von einer Diskette auf die andere. Dieses Beispiel geht davon aus, daß die Quelldiskette nur zwei FILEs enthält, die der Prüfung auf TEST? genügen.

In Beispiel 3 wurden sowohl die FILE-Namen als auch die Zusätze durch wilde Karten ersetzt. DOS kopiert daher hier alle FILEs, außer denen, die den Zusatz .SYS tragen.

Dieses Beispiel geht davon aus, daß nur drei Ablagen kopiert werde sollen: MEM.SAV, TEST1 und TEST2.

Beispiel 1:

```
SELECT ITEM OR RETURN FOR MENU
O RETURN
NAME OF FILE TO MOVE?
DOSEX.BAS RETURN
TYPE "Y" IF OK TO USE PROGRAM AREA
CAUTION: A "Y" INVALIDATES MEM.SAV
Y RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN
INSERT DESTINATION DISK, TYPE RETURN
RETURN
SELECT ITEM OR RETURN FOR MENU
```

Example 2:

```
SELECT ITEM OR RETURN FOR MENU
O RETURN
NAME OF FILE TO MOVE?
TEST? RETURN
TYPE "Y" IF OK TO USE PROGRAM AREA
CAUTION: A "Y" INVALIDATES MEM.SAV
Y RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN

  COPYING--D1: TEST1
INSERT DESTINATION DISK, TYPE RETURN
RETURN

INSERT SOURCE DISK, TYPE RETURN
RETURN

  COPYING--D1: TEST2
INSERT DESTINATION DISK, TYPE RETURN
RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN
SELECT ITEM OR RETURN FOR MENU
```

Example 3:

```
SELECT ITEM OR RETURN FOR MENU
O RETURN
NAME OF FILE TO MOVE?
*.* RETURN
TYPE "Y" IF OK TO USE PROGRAM AREA
CAUTION: A "Y" INVALIDATES MEM.SAV
Y RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN

  COPYING--D1: MEM.SAV
INSERT DESTINATION DISK, TYPE RETURN
RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN
  COPYING--D1: TEST1
INSERT DESTINATION DISK, TYPE RETURN
RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN
  COPYING--D1: TEST2
INSERT DESTINATION DISK, TYPE RETURN
RETURN
INSERT SOURCE DISK, TYPE RETURN
RETURN
SELECT ITEM OR RETURN FOR MENU
```

Figure 4-20 Using the Duplicate File Option

WEITERE INFORMATIONEN FÜR DEN ANWENDER

BASIC-BEFEHLE, DIE IM VERBINDUNG MIT DOS BENUTZT WERDEN

Bevor man die BASIC-Befehle besprechen, die in Verbindung mit DOS II benutzt werden, muß etwas darüber gesagt werden, wie diese Befehle in gespeicherten, bzw. zurückgeholten Programmen arbeiten. Die folgenden Absätze erklären die beiden Arten von Ablagen, die BASIC-Programme enthalten.

NICHT GEKENNZEICHNETE UND GEKENNZEICHNETE PROGRAMME

Die erste Art von Ablagen, "ungekennzeichnete", enthält ATASCII-Zeichen, sodaß ein Blick entsteht, wie beim Ausdruck eines BASIC-Programms. Die ungekennzeichneten Programme bewahren nicht ihre Symboltafeln bei jeder Ein- und Ausgabe. Die Symboltafel verbindet den Variablennamen mit der Speicheradresse, an der die Werte für diese Variable abgelegt sind.
Um ein FILE in ungekennzeichnete Form abzuspeichern, bzw. wieder einzulesen, benutzt man die Befehle LIST, bzw. ENTER.
Die zweite Art, "gekennzeichnete" Programme, stellt die konzentrierte Version eines BASIC-Programms dar. Hierzu gehörige Programme enthalten EIN-BYTE-"Kennzeichen", statt der ATASCII-Zeichen, um die BASIC-Befehle zu symbolisieren.
Gekennzeichnete Programme werden zwischen dem Plattenlaufwerk und der Computerkonsole mit Hilfe der Befehle SAVE und LOAD hin- und hergeschoben. Gekennzeichnete Versionen von Programmen sind gewöhnlich kürzer als ungekennzeichnete. Aus diesem Grunde ziehen viele Programmierer es vor, ihre endgültigen Programme in gekennzeichneter Form abzuspeichern, weil sie dann weniger Platz brauchen und schneller einlesbar sind. Eine gekennzeichnete Version behält ihre Symboltafel von Einlesung zu Einlesung bei.

LOAD (LO.)
LOAD filespec
LOAD"D1:DOSEX.BAS" RETURN

Dieser Befehl dient dazu, ein FILE von einer bestimmten Diskette in einem Diskettenlaufwerk in den Anwenderbereich des RAM zu laden. Die Anwendung dieses Befehls zum Einlesen einer Ablage setzt voraus, daß diese zuvor mit Hilfe des BASIC-Befehls SAVE aufgezeichnet wurde. Dieser Befehl liest nur gekennzeichnete Versionen ein.
Er kann auch zum "Verketten" von Programmen verwendet werden (Abbildung 5-1). Wenn ein Programm so groß ist, daß es nicht in das zur Verfügung stehende RAM paßt, setzt man den LOAD-Befehl auf die letzte Zeile des ersten Programmteils (Abb. 5-1).
Dann liest das Programm bei Erreichen dieser Stelle automatisch den nächsten Teil von der Diskette ein. Allerdings muß der zweite Teil insoweit selbständig sein, daß er keinerlei Daten oder Variablen des ersten Teils benötigt. Das nun eingelesene Programm arbeitet erst, wenn der Benutzer RUN eingegeben und RETURN drückt. In diesem Moment ist das vorige Programm mitsamt seinen Variablen gelöscht (vergl. weiteres Beispiel unter RUN).

```
100 REM Chain programm
110 LOAD"D1:CHAIN.BAS"
```

SAVE (S.)

```
SAVE filespec
SAVE "D1: EXAMP2.BAS"
RETURN
```

Dieser Befehl veranlaßt die Anlage, ein Programm auf Diskette abzuspeichern, wobei der filespec (Name) im Befehl genannt wird. SAVE ist das Pendant zu LOAD und speichert gekennzeichnete Programme ab.

LIST (L.)

```
LIST "D: DATFIL.LST"
LIST "P:"
LIST "P:", 10, 100
```

Eine der Anwendungen des LIST-Befehls in BASIC ähnelt stark dem SAVE-Befehl, da hierbei ein Programm aus dem Anwenderbereich des RAM genommen und in ein bestimmtes Diskettenfach transportiert werden kann, und zwar mit jedem beliebigen Namen, der gewünscht wird (vergl. erstes Beispiel).

Das Programm wird allerdings im Standard-ATASCII-Text, und nicht in gekennzeichneter Form gespeichert. Unterschiede in der Datenformatierung machen LIST ebenfalls viel flexibler als SAVE. Wie in den obigen Formatbeispielen gezeigt, kann ein einzelnes Peripheriegerät (z.B. P:,E:,C:,D:,D2: usw.) angegeben werden, oder es lassen sich bestimmte Zeilennummern auf ein bestimmtes Gerät listen (z.B. "P:",100,200).

Wenn man hinter dem LIST-Befehl kein Gerät spezifiziert, werden alle Zeilennummern, die man eingibt, auf dem Monitor angezeigt. Dieser (E:) tritt stets automatisch in Aktion, wenn nichts anderes angegeben wird.

Alles in allem liegt der Hauptunterschied zwischen LIST und SAVE darin, daß LIST Standardtext in ATASCII an verschiedene Peripheriegeräte transportiert, wogegen SAVE lediglich gekennzeichnete BASIC-Programme auf eine Diskette bringen kann.

ENTER (E.)

```
ENTER filespec
ENTER "D: LIST2.LST"
```

Dieser Befehl veranlaßt den Computer, ein FILE von der Diskette mit dem angegebenen filespec in den Speicher zu übertragen. Das Programm wird ungekennzeichnet eingelesen und wird der Reihe nach bei Empfang interpretiert. Im Gegensatz zu LOAD zerstört ENTER kein im RAM gespeichertes Programm, vermischt jedoch das zu ladende Diskettenprogramm mit dem im RAM befindlichen.

Falls in den beiden Programmen gleiche Zeilennummern vorkommen, ersetzt die eingelesene Zeile jeweils die RAMzeile mit der gleichen Nummer.

RUN

RUN filespec
RUN "D2: MYFILE.BAS"

Dieser Befehl veranlaßt den Computer, das angegebene FILE (spec) zu lesen und abzuarbeiten (LOAD und RUN), insofern handelt es sich hier um die Kombination zweier Befehle. Der RUN-Befehl kann jedoch nur mit gekennzeichneten FILEs arbeiten. So ist z.B. ein Befehl RUN "D2: LIST.LST" nicht ausführbar. Zur Verkettung von Programmen und um ein zweites Segment eines FILEs automat sch zum Einlesen und Arbeiten zu bringen, kann man einen RUN"D:f lespec" als letzte Zeile des ersten Segments benutzen. Allerdings muß auch hier das zweite Teilprogramm unabhängig von Variab en und anderen Daten des ersten sein. Vor dem Ablauf des ersten Segments sollte man sicherstellen, daß sein Inhalt auf der Diskette bewahrt wird, da das RUN-STATEMENT den im RAM befindlichen ersten Programmteil auslöscht, sobald der zweite geladen wird.

INPUT/OUTPUT-Kontrollblocks (IOCBs)

Eine INPUT/OUTPUT-Operation wird durch einen I/O-Kontrollblock (IOCB) gesteuert. Ein solcher IOCB ist ein Datenblock, bestehend aus Angaben über die Art der I/O-Operation, der Pufferlänge, der Pufferadresse und zwei weiteren Kontrollvariablen, von denen die zweite gewöhnlich auf Null steht. ATARI BASIC stellt acht IOCBs auf und verwendet drei davon wie folgt:

1. IOCB#0 wird von BASIC benutzt für I/O von, bzw. an E:,
2. IOCB#6 wird von BASIC benutzt für I/O von, bzw. an S:,
3. IOCB# 7 wird von BASIC benutzt für LPRINT-, CLOAD- und SAVE-Befehle.

Die IOCBs Nummer 1 bis Nummer 5 können frei benutzt werden, die festgelegten drei sollte man möglichst nicht anwählen, es sei denn, ein bestimmtes Programm braucht sie nicht für interne Zwecke.

IOCB #0 kann nie durch ein BASIC-Programm geöffnet oder geschlossen werden.

BENUTZUNG DER OPEN/CLOSE-BEFEHLE

OPEN (O.)

OPEN/CLOSE
INPUT/PRINT
PUT/GET
STATUS
XIO

Das OPEN-Statement verbindet einen bestimmten IOCB mit der richtigen Gerätesteuerung, bereitet jeweils nötige Kontrollvariablen vor, die zum zentralen INPUT und OUTPUT gehören (vergl. Wortverzeichnis im Anhang) und gibt gerätespezifische Wahlelemente an die Gerätesteuerung weiter. Die Parameter für dieses Statement werden in Abbildung 5-2 gezeigt.

#	Kennzeichnung, die vom Anwender eingegeben wird.
iocb	Eine Zahl zwischen 1 und 7 (einschl.), die sich auf ein Gerät, bzw. ein FILE bezieht.
aexp1	Eine Zahl, die angibt, welche Art von Operation erfolgen soll.

Code 4 = INPUT-Operation, setzt den FILEzeiger auf den FILEbeginn.
6 = INPUT-Operation für Plattenverzeichnis.
8 = OUTPUT-Operation, setzt den FILEzeiger auf den FILEbeginn.
9 = Abschluß-Operation end-of-file, setzt den FILEzeiger auf das FILEende.
12= INPUT- und OUTPUT-Operation, setzt den FILEzeiger auf den FILEbeginn.

aexp2 Geräteabhängiger Hilfscode. Eine 83 (ASCII S) an dieser Stelle veranlaßt den ATARI-Drucker 820, seitlich zu drucken, sonst steht diese Stelle immer auf 0 (Null).

filespec Spezifischer FILEhinweis (vergl. Abschnitt 1, Definition des filespec).

Abbildung 5-2: Erläuterung der Parameter von OPEN-Statements

Im Beispiel OPEN#2,8,0,"D1:ATARI800.BAS", wird das IOCB#2 geöffnet, um den OUTPUT auf ein File in Plattenfach 1 mit der Kennung ATARI800.BAS zu ermöglichen. Falls in Fach 1 eine Ablage dieses Namens vorhanden ist, zerstört das OPEN-Statement das vorhandene File und baut ein neues auf. Wurde das IOCB schon zuvor geöffnet, so erscheint ERROR-129 auf dem Monitor (File schon geöffnet).

CLOSE (CL.)

CLOSE #iocb
300 CLOSE #2

Der CLOSE-Befehl löst das zuvor für die Lese/Schreib-Operation geöffnete IOCB aus. Die Zahl hinter den Kennzeichen muß dieselbe sein, die als IOCB-Bezugszahl im OPEN-Statement benutzt wurde (vergl. Beispiel unten).
Wenn der betreffende IOCB schon für ein bestimmtes Gerät geöffnet wurde und nun versucht wird, ihn noch einmal für ein zweites Gerät zu verwenden, ohne ihn vorher zu schließen, so erscheint ebenfalls ERROR-129 auf dem Monitor. Derselbe IOCB darf nämlich nicht gleichzeitig für mehrere Geräte eingesetzt werden. Dagegen wird kein Fehler gemeldet, wenn man ein File schließt, das schon vorher geschlossen worden war.

```
10 OPEN#1,8,0,"D:FIL.BAS"
20 CLOSE#1
```

Abbildung 5-3: Beispiel für öffnen und Schließen eines Files. Der END-Befehl schließt alle offenen Files (außer IOCB#0).

DER GEBRAUCH DER INPUT/PRINT-BEFEHLE

INPUT (I.)

```
100 INPUT #2; X, Y
100 INPUT #2; N$
```

Dieser Befehl dient zum Anfordern von Daten (numerisch oder in Strinsform) von einem bestimmten Gerät. INPUT ist das Pendant zu PRINT. Wenn man diese Instruktion ohne ein #IOCB benutzt, nimmt die Maschine an, daß das Gerät (E:) gemeint ist (automatische Anwahl des Monitors). INPUT benutzt den s.g. RECORD-INPUT/OUTOPUT (Erklärung im Abschnitt PRINT unten).

```
 5 REM **          ERSTELLE DATENABLAGE          **
 7 REM **          OEFFNE MIT 8          **
10 OPEN #1,8,0,"D:WRITE.DAT"
20 DIM WRT$(60)
30 ? "GEBE SATZ MIT HOECHSTENS 80 ZEICHEN EIN "
35 INPUT WRT$
38 REM **          SCHREIBE DATEN AN DISKETTE    **
40 ? #1,WRT$
45 REM **          SCHREIBE DATENABLAGE          **
50 CLOSE #1
55 REM ** OEFFNE DATENABLAGE ZUM LESEN **
58 REM ** OEFFNE MIT4,ZUM LESEN **
60 OPEN #1,4,0,"D:WRITE.DAT"
65 REM ** LESE DATEN VON DISKETTE **
70 INPUT #1,WRT$
75 REM ** DRUCKE DATEN **
80 ? WRT$
85 REM ** SCHLIESSE DATENFILE **
90 CLOSE #1
```

Abbildung 5-4: Muster eines INPUT/PRINT-Programms

In Abb. 5-4 gestattet die Zeile 35 dem Benutzer, Daten über die Tastatur einzugeben (Ersatzgerät, wenn nichts anderes programmiert). Auf Zeile 70 liest das INPUT-Statement den Strinsinhalt aus dem geöffneten File.

PRINT (PR.or?)

```
100 PRINT #2; X, Y
100 PRINT #2; A$
100 ? C$
100 PRINT "X = ",X
```

Dieser Befehl schreibt einen Ausdruck auf das geöffnete Peripheriegerät (String oder arithmetische Form) mit der gleichlautenden IOCB-Kennzahl.

Wenn keine IOCB-Nummer spezifiziert wurde, schreibt die Anlage den Ausdruck über Monitor aus, der hierbei automatisch angewählt wird.
Falls die Information an ein Gerät gerichtet wird, das nicht offen ist, zeigt der Monitor ERROR-133.
PRINT führt den s.g. RECORD-I/O aus. Solche Records sind Gruppen von Bytes, die durch Zeilen-Ende-Symbole getrennt sind (9B Hexadezimal). Die Größe eines Records kann willkürlich gewählt werden. Sie läßt sich durch die Länge eines Strings auf einem Diskettenfile oder durch das Format einer arithmetischen Variablen festlegen. Sie kann auch der Länge eines Zeichenstrings entsprechen, der von der Tastatur kommt und durch RETURN-Taste beendet wird.
Das INPUT-Statement kann im Allgemeinen keine Records lesen, die länger als 110 Zeichen sind. Es ist daher zu empfehlen, beim PRINT eines Records auf eine Diskette nicht mehr als 110 Zeichen zu wählen, für den Fall, daß dieser Record später wieder mit INPUT eingegeben werden soll.

DIREKTZUGRIFF MIT DEN BEFEHLEN NOTE/POINT

NOTE (NO.)

NOTE #iocb, avar, avar
NOTE #2, A, B

Ablagen werden sequenziell erstellt und vom Anfang zum Ende hin abgegriffen. Will man jedoch die Records einer Ablage in nichtsequenzieller Weise erfassen (direkt), kann man entweder die Ablage sequenziell lesen und an dem gewünschten Record anhalten, oder man muß eine spezielle Methode des direkten Zugriffs auf den Record haben.

```
1 REM ** DIESES PROGRAMM LIEST **
2 REM ** DATENZEILEN VON TASTATUR **
3 REM ** UND SPEICHERT SIE AUF **
4 REM ** DISKETTE **
20 DIM A$(40)
25 OPEN #1,8,0,"D:DATFIL.DAT"
27 OPEN #2,8,0,"D:POINTS.DAT"
30 REM ** LESE DATENZEILE VON TASTATUR **
40 INPUT A$
41 LPRINT A$
42 REM ** NUR WENN RETURN DANN STOP **
45 IF LEN(A$)=0 THEN 100
50 NOTE #1,X,Y
55 REM ** SPEICHERE DATENZEILE **
60 PRINT #1;A$
61 REM **-SPEICHERE ZEIGER AM ANFANG DER DATENZEILE **
65 PRINT #2;X;",";Y
70 LPRINT "SECTOR # =";X,"BYTE # = ";Y
90 GOTO 40
95 REM ** ZEIGE ENDE DES FILES AN **
100 PRINT #2;0;",";0
110 END
```

Das vorige Beispiel ist sehr zeitaufwendig für große Ablagen. Daher enthält DOS II die Befehle NOTE und POINT, die es ermöglichen, ein Einzelfile direkt zu erfassen. Um einen Record zu erreichen, ohne durch alle übrigen davorliegenden Records gehen zu müssen, muß man dem Computer angeben, welchen Record man sucht. Dies erfordert eine "Notiz" (NOTE) im betreffenden Ablagesektor. Daher gibt man einen NOTE-Befehl vor jeder Aufzeichnung und hebt den zurückgemeldeten Wert in einer Tafel auf. Dieser NOTE-Befehl nimmt den Wert des laufenden Ablagezeigers (file-pointers) für die Ablage auf, die gerade der spezifizierte IOCB benutzt. Dieser Zeiger legt die genaue Position innerhalb des Files fest, wo das nächste Byte gelesen oder geschrieben werden soll. Der Befehl hält die absolute Plattensektor-Nummer in der ersten arithmetischen Variablen fest und die laufende Byte-Nummer in der zweiten. Sektornummern reichen von 1 bis 719, Bytenummern von 0 bis 124. Das folgende Programmbeispiel zeigt einen Weg, wie man Tastatureingaben mit Hilfe von NOTE in eine spezifische Ablagenstelle bringen kann. Der Ausdruck in Abb. 5-5 stellt einen Musterabschnitt des NOTE-Programms dar. Im Probelauf werden Zahlen benutzt, jedoch kann man jeden beliebigen String für A$ von bis zu 40 Zeichenlänge eintasten. Dieses Musterprogramm wurde mit einer Diskette abgearbeitet, die die Ablagen DOS.SYS, DUP.SYS und MEM.SAV enthielt. Andere Anwender mögen andere Sektoren- und Bytenummern haben. In unserem Fall wurden die Eingänge 45,55,75, 80,90,100 und 110 gewählt.

```
45
SECTOR # = 145      BYTE # = 9
55
SECTOR # = 145      BYTE # = 12
75
SECTOR # = 145      BYTE # = 15
80
SECTOR # = 145      BYTE # = 18
90
SECTOR # = 145      BYTE # = 21
100
SECTOR # = 145      BYTE # = 24
110
SECTOR # = 145      BYTE # = 28
```

Figure 5-6 Sample Run of NOTE Program

POINT (P.)

POINT #iocb, avar, avar
100 POINT #2, A, B

POINT ist das Pendant zu NOTE. Dieser Befehl setzt den Filezeiger auf einen beliebig wählbaren Wert, der durch die arithmetischen Variablen bestimmt wird. POINT wird zum Lesen spezifizierter File-Positionen (Sektor und Byte) eingesetzt. Die erste arithmetische Variable spezifiziert die Sektornummer, die zweite spezifiziert die nächstfolgende Bytenummer, welche das nächste zu lesende Byte enthält, bzw. wohin das nächste Byte abgestellt werden soll. Wie beim NOTE-Befehl, reichen die Sektorennummern von 1 bis 719 und der Bytebereich geht von 0 bis 124.

Wenn man über ein geöffnetes File hinausweist, ergeht eine Fehlermeldung wegen unpassender Filenummer.

Die Programmlistung (Abb. 5-7) und der Musterablauf (Abb.5-8) enthalten ein Beispiel für den POINT-Befehl, wie er Daten einliest, die in dem Beispielprogramm für den NOTE-Befehl erstellt worden waren. Beim Ablauf druckt dieses Programm die Tastatureingaben in umgekehrter Folge von Sektoren und Bytes wie die Abspeicherung auf Diskette: (folgt Abb. 5-7 aus Original)

Nach dem Eintasten der Musterprogramme NOTE und POINT, Taste RUN und drücke RETURN!

```
1 REM ** DIESES PROGRAMM LIEST DIE **
2 REM ** ABLAGE,DIE NOTEST ERSTELLT **
3 REM ** HAT UND DRUCKT SIE IN **
4 REM ** UMGEKEHRTER FOLGE **
10 DIM B(20,1)
20 DIM A$(40)
25 REM ** OEFFNE DATENABLAGE **
30 OPEN #1,4,0,"D:DATFIL.DAT"
35 REM ** OEFFNE ZEIGERABLAGE **
40 OPEN #2,4,0,"D:POINTS.DAT"
45 REM ** LESE ZEIGER IN EIN ARRAY EIN **
50 FOR I=0 TO 20
60 INPUT #2;X,Y
70 B(I,0)=X:B(I,1)=Y
80 IF X=0 AND Y=0 THEN LAST=I:I=20
90 NEXT I
95 REM ** DRUCKE FILE IN UMGEKEHRTER FOLGE **
100 FOR I=LAST-1 TO 0 STEP -1
110 X=B(I,0):Y=B(I,1)
120 POINT #1,X,Y
130 LPRINT "SECTOR # = ";X,"BYTE # = ";Y
140 INPUT #1;A$
150 LPRINT A$
160 NEXT I
```

```
SECTOR # = 145      BYTE # = 28
110
SECTOR # = 145      BYTE # = 24
100
SECTOR # = 145      BYTE # = 21
90
SECTOR # = 145      BYTE # = 18
80
```

```
SECTOR # = 145      BYTE # = 15
75
SECTOR # = 145      BYTE # = 12
55
SECTOR # = 145      BYTE # = 9
45
```

Figure 5-8 Sample RUN of POINT Program

DER GEBRAUCH DER PUT/GET-Befehle

PUT (PU.)

PUT #iocb, aexp
100 PUT #6, ASC ("A")

Der PUT-Befehl schreibt ein einzelnes Byte (im Wert von 0 bis 255) auf das durch die IOCB-Kennzahl spezifizierte Periphärgerät. In Abbildung 5-9 wird der PUT-Befehl eingesetzt, um jede Zahl, die eingetastet wird, in ein als A(50) dimensioniertes Array zu schreiben. Man kann bis zu 50 Zahlen eintasten, jede von ihnen sollte allerdings kleiner als 256 sein.

Der PUT-Befehl wird zum Erstellen von Datenfiles eingesetzt, oder zum Anhängen von Daten an schon bestehende Files:

```
10 GRAPHICS 0:REM ** PUT/GET DEMO **
20 DIM A(50),A$(10)
30 GRAPHICS 0:? " PUT UND GET FUER DISK-PROG. BEISPIEL **"
40 ? "SOLL DIES EIN READ ODER EIN WRITE SEIN?":INPUT A$
50 IF A$="READ" THEN 170:REM ** WENN READ,DANN 170 **
60 IF A$<>"WRITE" THEN PRINT "?":GOTO 40:REM ** WENN WRITE,DANN WEITER
70 REM ** SCHREIBE ROUTINE **
80 OPEN #1,8,0,"D1:EXAMPL1.DAT"
90 ? "GEBE ZAHL KLEINER ALS 256 EIN!":INPUT X
95 REM ** SCHREIBE ZAHL AUF FILE **
100 PUT #1,X
110 IF X=0 THEN CLOSE #1:GOTO 130
120 GOTO 90
```

GET (GE.)

GET #iocb, avar
100 GET #2, X

Dieser Befehl liest ein einzelnes Byte von dem in der IOCB-Kennzahl spezifizierten Gerät in die spezifizierte Variable.

Der zweite Teil des Programmbeispiels (Abb. 5-10) unten illustriert den GET-Befehl. Er erlaubt das Auffinden jedes durch einen PUT-Befehl abgespeicherten Bytes.

Es ist zu beachten, daß INPUT/PRINT und GET/PUT unvereinba
Arten von INPUT/OUTPUT darstellen.
PRINT setzt Zeilen-End-Zeichen (end-of-line) zwischen die einzelnen Records (Speicherungs-Abschnitte) und INPUT benutzt diese Zeichen, um einen Record zu bestimmen.
GET und PUT dagegen schreiben lediglich einzelne Bytes auf ein File, ohne diese durch Zeilenende zu trennen. Eine Ablage, die per PUT-Statement erstellt wurde, sieht wie eine einzige große Datengruppe aus, es sei denn, man hat ein End-Of-Line-Zeichen (EOL)(9B Hexadezimal) in die Ablage miteingespeichert.

```
130 GRAPHICS 0:? :? " JETZT DIE DATEN IN DIE ABLAGE EINLESEN?":INPUT A$:?
140 IF A$="NEIN" THEN END :REM ** WENN NEIN DANN ENDE **
150 IF A$<>"JA" THEN 130:REM ** WENN JA,DANN WEITER **
160 REM ** SPEICHERE ROUTINE AB **
170 OPEN #2,4,0,"D1:EXAMPL1.DAT"
180 FOR E=1 TO 50
185 REM ** LESE ZAHLEN VOM FILE **
190 GET #2,G:A(E)=G
200 IF G=0 THEN 230
210 PRINT "BYTE # ";E;"=";G
220 NEXT E
230 CLOSE #2
```

Nach dem Eintasten des Musterprogramms PUT/GET, Taste RUN und drücke RETURN! Beim Ablauf des Programms in Abb. 5-10 druckt dieses die von der Tastatur eingegebenen Zahlen zusammen mit den Bytenummern, an denen sie jeweils gespeichert sind.
Nach dem Eintasten des Programms Taste RUN und drücke RETURN, wobei die Zahleneingänge 2,5,67,54,68 zu benutzen sind.
Abb. 5-11 ist der Musterablauf des PUT/GET-Programms:

BENUTZUNG DES STATUS-BEFEHLS

STATUS (ST.)

STATUS #iocb, avar
100 STATUS #5, ERROR

Der STATUS-Befehl wird zur Bestimmung der Bedingung (STATUS) einer Ablage eingesetzt. Es handelt sich um einen CIO-Befehl, der verschiedene Arten der Fehlerentstehung überprüft.
Die erste Reihe von Fehlern, auf die hin der STATUS prüft, ist die folgende:

Sektorpuffer verfügbar?	Wenn nicht, ERROR-161
Zulässige Gerätenummer?	Wenn nicht, ERROR-20
Zulässiger Ablagenname?	Wenn nicht, ERROR-170
Ablage auf Diskette?	Wenn nicht, ERROR-170
Ablage gesperrt?	Wenn ja, ERROR-167

Auch alle I/O-Kanalfehler (serial bus errors) werden mit Hilfe des STATUS-Befehls erkannt. Es handelt sich um folgende:

Gerät aus dem Takt	ERROR-138
Gerät nicht rückgemeldet	ERROR-139
Serielle Übertragung fehlerhaft	ERROR-140
Datenrahmen im seriellen Kanal überschritten	ERROR-141
Prüfsummenfehler im seriellen Kanal	ERROR-142
Gerät blockiert	ERROR-144

Zum Einsetzen dieses Befehls muß man die Ablage als nur -INPUT-Ablage (INPUT only file) eröffnen und sie dann schließen.

Nur dann kann der STATUS-Befehl eingesetzt werden. Es ist ratsam, die Form des XIO-Befehls für diesen Befehl zu wählen, da diese verlässlicher ist und es ermöglicht, einen spezifischen Filenamen mit dem gesuchten Fehler in Verbindung zu bringen.

Abb. 5-12 ermöglicht, den Zustand (STATUS) des Plattenfachs mit Hilfe eines TRAP-Statements zu prüfen. Man muß vor dem Ablauf des Programms das Plattengerät abschalten.

```
10 GRAPHICS 0:REM TRAP/STATUS DEMO
20 DIM A(50),A$(10),D$(1)
30 GRAPHICS 0:? "PUT UND GET BEISPIEL AUF PLATTE":?
40 ? "SOLL DIES EIN READ ODER WRITE SEIN":INPUT A$:?
50 IF A$="READ" THEN 160
60 IF A$<>"WRITE" THEN ? "?":GOTO 40
70 REM WRITE ROUTINE
80 TRAP 400:OPEN #1,8,0,"D1:EXAMPL1.DAT"
90 ? "GEBE EINE ZAHL UNTER 256 EIN":INPUT X
100 PUT #1,X
110 IF X=0 THEN CLOSE #1:GOTO 130
111 REM WENN X=0 SCHLIESE #1
120 GOTO 90
130 GRAPHICS 0:? :? " DATEN ABLEGEN JA ODER NEIN":?
140 IF A$="NEIN" THEN END
150 IF A$="JA" THEN 130
160 REM LESE ROUTINE AB
170 TRAP 400:OPEN #1,4,0,"D1:EXAMPL1.DAT"
180 FOR E=1 TO 50
190 GET #1,G:A(E)=G
200 IF G=0 THEN GOTO 230
210 ? "BYTE# ";E;"=";G
220 NEXT E
230 CLOSE #1
240 END
```

```
0 TRAP 40000:STATUS #1,ST:IF ST<>138 AND ST<>139 THEN ? "HILFE":? ST:GOTO
0 ? "IST DIE FLOPPY ANGESCHALTET"
0 ? "TIPPE JA WENN ERLEDIGT";:INPUT D$
0 CLOSE #1:GOTO 40
```

ERSETZEN DES XIO-BEFEHLS DURCH OPTIONEN DES DOS-MENUS

XIO (X.)

XIO cmdno, #iocb, aexp1, aexp2, filespec.
100 XIO 3, #6, 4, 0, "D: TEST.BAS"

Der Befehl XIO ist ein allgemeines INPUT/OUTPUT-Statement, das besondere Operationen ausführt. Man wendet ihn an, wenn man gewisse Funktionen verwenden will, die sonst über die Wahlmöglichkeiten des DOS-Menus erzeugt werden. Diese XIO-Befehle öffnen Files, schließen Files, speichern einen Status ab, stellen den Bezug auf Speicherstellen zum Schreiben, Lesen, Umbenennen, Löschen, Sperren oder Entsperren eines Files her. Zu beachten ist, daß XIO-Anrufe Filenamen (filespecs) benötigen. CMDNO (Befehlsnummer) wird angewandt, um festzulegen, welche dieser Operationen im Einzelnen durchgeführt werden sollen.

ÜBERSICHT

CMDNO	OPERATION	EXAMPLE
3	OPEN	XIO 3, #1, 4, 0, "D: TEST.BAS"
5	GET Record	XIO 5, #1, 0, 0, "D: TEST.BAS"
7	GET Characters	XIO 7, #1, 0, 0, "D: TEST.BAS"
9	PUT Record	XIO 9, #1, 0, 0, "D: TEST.BAS"
11	PUT Characters	XIO 11, #1, 0, 0, "D: TEST.BAS"
12	CLOSE	XIO 12, #1, 0, 0, "D: TEST.BAS"
13	STATUS Request	XIO 13, #1, 0, 0, "D: TEST.BAS"
32	RENAME	XIO 32, #1, 0, 0, "D: OLD, NEW"
33	DELETE	XIO 33, #1, 0, 0, "D: TEMP.BAS"
35	LOCK FILE	XIO 35, #1, 0, 0, "D: ATARI.BAS"
36	UNLOCK FILE	XIO 36, #1, 0, 0, "D: DOSEX.BAS"

Anmerkung: Den Gerätenamen beim Umbenennen einer Ablage nicht zweimal benutzen, also nicht formulieren: "D:OLD,D:NEW."!

PROGRAMMBEISPIEL FÜR EINSATZWEISEN DES XIO-BEFEHLS

Abb.5-13 unten ermöglicht die Erstellung einer Ablage für jeden Monat eines Jahres, in dem man jeweils Namen und Geburtstage eintragen kann. Das Programm verwendet XIO-Befehle zum Erstellen eines Files pro Monat, zum Sperren und Entsperren jeder Ablage je nach Notwendigkeit, sowie zum Schließen der Ablage, nachdem sie durchlaufen ist. Zeile 20 definiert das Plattenfile D:Birthday als FILE$ (Filebezeichnung). Dann, auf Zeile 170, wird das jeweilige FILE$ durch ein XIO-Statement für die Eingabe geöffnet. Das XIO-Statement auf Zeile 390 entsperrt den richtigen Einzelfall. Das Statement auf 400 erstellt das File und ermöglicht, darauf zu schreiben. Das nächste XIO, auf Zeile 430, schließt das File wieder und das dann folgende XIO sperrt es, um es gegen versehentliches Überschreiben oder Löschen zu schützen.

```
5 GRAPHICS 0
10 DIM A$(5),D$(15),FILE$(20),DATE$(20),
MON$(20),ERR$(20),NAME$(40)
20 FILE$="D:BIRTHDAY."
30 ERR$="ERROR IN MONTH #"
100 GRAPHICS 0:? "PLEASE TYPE MONTH NUMB
ER (1-12)"
110 TRAP 100:INPUT MONTH
120 TSTEND=0
130 MONTH=INT(MONTH)
140 IF MONTH<1 OR MONTH>12 THEN ? ERR$:GOTO 100
145 GOSUB 1000+MONTH
150 FILE$(12)=STR$(MONTH)
160 EOF=0
170 TRAP 700:XIO 3,#2,4,0,FILE$
180 TRAP 600:FOR I=0 TO 1 STEP 0
190 INPUT #2;NAME$
200 INPUT #2;DATE$
210 EOF=EOF+1
220 IF EOF=1 THEN ? "BITTE GEBURTSTAGE ";MON$;" SIND:":?
224 TEMP=LEN(NAME$)
225 NAME$(TEMP+1)=" "
226 NAME$(30)=" "
227 NAME$(TEMP+2,30)=NAME$(TEMP+1)
230 ? NAME$,DATE$
240 NEXT I
299 REM ** MACHE NEUE EINGABEN IN GEBURTSTAGEN **
300 ? " SOLLEN NEUE GEBURTSTAGE EINGETRAGEN   WERDEN?":INPUT
310 IF A$<>"JA" THEN 20
320 ? " BITTE DEN PERSONENNAMEN EINGEBEN"
330 INPUT NAME$
340 ? " BITTE DEN GEBURTSTAG DER PERSON (MM-DD-YY)"
350 INPUT DATE$
360 MONTH=INT(VAL(DATE$))
370 IF MONTH<1 OR MONTH>12 THEN ? ERR$,DATE$:GOTO 300
380 FILE$(12)=STR$(MONTH)
390 TRAP 400:XIO 36,#3,0,0,FILE$:OPEN #2,9,0,FILE$:GOTO 410
400 CLOSE #2:XIO 3,#2,8,0,FILE$
410 PRINT #2;NAME$
420 PRINT #2;DATE$
430 XIO 12,#2,0,0,FILE$
440 XIO 35,#2,0,0,FILE$
450 GOTO 300
600 CLOSE #2:IF EOF=0 THEN ? " KEIN GEBURTSTAG IM MONAT ";MON
```

```
610 MONTH=MONTH+1
620 IF MONTH>12 THEN MONTH=1
630 TSTEND=TSTEND+1
640 IF TSTEND=1 THEN 145
650 GOTO 300
700 EOF=0:GOTO 600
1001 MON$="JANUAR":RETURN
1002 MON$="FEBRUAR":RETURN
1003 MON$="MAERZ":RETURN
1004 MON$="APRIL":RETURN
1005 MON$="MAI":RETURN
1006 MON$="JUNI":RETURN
1007 MON$="JULI":RETURN
1008 MON$="AUGUST":RETURN
1009 MON$="SEPTEMBER":RETURN
1010 MON$="OKTOBER":RETURN
1011 MON$="NOVEMBER":RETURN
1012 MON$="DEZEMBER":RETURN
3000 FILE$(12,12+LEN(STR$(MONTH)))=STR$(MONTH):? FILE$
```

Wenn man dieses Programm abspielt, muß man eine Zahl von 1 bis 12 eintasten. Das Programm prüft dann, ob in der betreffenden Teilablage irgendwelche Eintragungen sind. Falls nicht, erscheint die Meldung NO BIRTHDAYS IN auf dem Monitor, gefolgt von dem Namen des Monats, der gerade gewählt wurde. Sind dort Eintragungen vorhanden, gibt der Bildschirm die Namen und Geburtstage für den betreffenden Monat wieder. In beiden Fällen gibt das Bild auch die Namen und Geburtstage des nächstfolgenden Monats wieder, damit bei dieser Gelegenheit sogleich auf einen etwa wichtigen Geburtstag der nächsten Zeit hingewiesen wird. Falls weitere Files nicht gezeigt, bzw. keine weiteren Eintragungen gemacht werden sollen, taste NO bei jeder Bereitschaftsmeldung, das Programm endet dann.

DAS SPEICHERN UND EINLESEN VON PROGRAMMEN UND DATEN MIT BASIC

Ein gekennzeichnetes Programm kann man nicht modifizieren. Deshalb muß man ein solches Programm vor dem Ändern in eine ungekennzeichnete Version umformen. Dadurch wird verhindert, daß die interne Symboltafel (vergl. Abschnitt über gekennzeichnete und ungekennzeichnet Ablagen) zu stark aufgebläht wird. In Teil 3 wurden die Befehle SAVE und LOAD benutzt, um gekennzeichnete Programme zu laden, bzw. abzurufen. Im selben Abschnitt wurde der SAVE-Befehl angewandt, um ein Programm, wie z.B. D.:INTEREST.SAV auf Diskette abzustellen. In dem nun folgenden Beispiel soll INTEREST.SAV modifiziert werden, und zwar, indem zuerst das gekennzeichnete Programm eingelesen wird, und indem es dann als ungekennzeichnete Version zurück auf die Diskette gebracht wird. Diese ungekennzeichnete Version kann dann in den Computer geladen und modifiziert werden. Sobald das Programm INTEREST.SAV von der Diskette gelöscht wurde, gehe man von Abb. 5-14 aus und Taste es wieder in die Maschine ein.

```
1100 REM ** ZINSBERECHNUNG **
110 ? " DIESES PROGRAMM ZEIGT,WIE SICH"
120 ? " IHR GELD AUF DER BANK JAEHRLICH"
130 ? " DURCH DIE ZINSEN ERHOEHT."
140 ? " ZUM BEENDEN BREAK-TASTE DRUECKEN!"
150 ?
160 ? " GUTHABEN:";
170 INPUT P
180 ? " ZINSRATE:";
190 INPUT R
200 N=1
210 ?
220 A=P*(1+R/100)^N
230 ? " JAHR = ";N
240 ? " GUTHABEN = ";A
250 N=N+1
260 GOTO 210
```

LIST und ENTER

Unten sind die Schritte zum Speichern und Rückholen dieses Programmes auf, bzw. von der Diskette mit Hilfe des LIST-, bzw. ENTER-Befehls angegeben. Wenn eine ATARI810-Plattenanlage vorhanden ist, gilt die erste Serie von Instruktionen, für die Anlage 815 gilt die zweite Serie.

Für Plattengerät 810:

1. Benutze die Systemdiskette, um DOS II einzulesen.
2. Nimm die Systemdiskette heraus und schiebe die Datendiskette ein.
3. Taste LOAD"D:INTEREST.SAV", dann drücke RETURN.
4. Taste LIST und drücke RETURN, dann kommt D:INTEREST.SAV auf den Monitor.
5. Taste jetzt LIST"D:INTEREST.LIS" und drücke RETURN, um das unsekennzeichnete Programm auf die Diskette zu bringen (dies wird nicht auf dem Monitor abgebildet).
6. Taste NEW und RETURN-Taste (hierdurch wird das Programm INTEREST.SAV im Speicher gelöscht).
7. Taste ENTER"D:INTEREST.LIS" und drücke RETURN. Hierdurch wird das Programm in unsekennzeichneter Form zurückgeladen.
8. Taste LIST und drücke RETURN. Mit Hilfe der entsprechenden Tasten wird jetzt die Programmzeile 160 von "PRINZIPAL" in "PRINZIPALAMOUNT" (Hauptbetrag) geändert.
9. Taste LIST"D:INTEREST.LIS" und drücke RETURN, um die Änderung in das schon auf der Diskette befindliche Programm zu bringen.
10. Taste ENTER"D:INTEREST.LIS" und drücke RETURN.

FÜR PLATTENGERÄT 815

1. Benutze die Systemdiskette in Fach 1, um das DOS II einzulesen und schiebe die Datendiskette in Fach 2.
2. Taste LOAD"D2:INTEREST.SAV" und drücke RETURN.
3. Taste LIST und drücke RETURN, dann erscheint D2:INTEREST.SAV auf dem Bild.
4. Taste nun LIST"D2:INTEREST.LIS" und drücke RETURN, um das ungekennzeichnete Programm auf Diskette zu bringen (das Programm erscheint nicht auf dem Monitor).
5. Taste NEW und drücke RETURN.

Um das Programm von der Diskette ablaufen zu lassen:

Taste RUN und drücke RETURN. Die Anlage holt das Programm und führt es aus (vergl. Abb. 5-15). Die Zahleneintragungen, die für das INPUT-Statement verwandt wurden, waren 1.200$ als Hauptbetrag und 12 % als Zinsrate.

```
RUN
IF YOU TYPE THE AMOUNT OF PRINCIPAL
AND THE INTEREST RATE PER YEAR, I WILL

SHOW YOU HOW YOUR MONEY GROWS, YEAR BY

YEAR, TO STOP ME, PRESS THE BREAK KEY.

PRINCIPAL?1200
INTEREST RATE?12

YEAR = 1
AMOUNT =1343.999988

YEAR = 2
AMOUNT =1505.28

YEAR = 3
AMOUNT =1685.913576

STOPPED AT LINE 200
```

Figure 5-15 Sample Run of Interest Program

Wenn man das Originalprogramm UND die modifizierte Version aufheben will, muß man dem modifizierten Programm einen neuen Namen geben.

Um ein Programm auszugeben oder abzuändern, unternimmt man folgende Schritte:

a) zum Abspeichern einer gelisteten Version auf Diskette: Taste LIST"D:INTEREST.LIS" und drücke RETURN, daraufhin werden die spezifischen Zeilen in das bestehende Programm eingefügt.

b) zum Rückholen der gelisteten Version auf Diskette:

1. Taste NEW und drücke RETURN (hierdurch wird die gekennzeichnete Version und deren Symboltafel im Speicher gelöscht).
2. Taste ENTER"D:INTEREST" und drücke RETURN, hierdurch wird die unsekennzeichnete Version in den Speicher übertragen.
3. Taste LIST und drücke RETURN.

Anmerkung: Wenn man sowohl das Original, als die revidierte Form des Programms aufbewahren will, muß man den Namen des modifizierten Programms ändern.

Man kann die Befehle LIST und ENTER auch verwenden, um Datenablagen, die man ausgeben will, zu speichern und zurückzuholen. Eine Datenablage enthält keinerlei Befehle oder Instruktionen. Sie enthält Namen für ein Adressenverzeichnis, Zahlen von Scheckbeträgen usw. . Dennoch muß man eine Möglichkeit haben, diese Datenablagen zu adressieren. Hierfür braucht man ein Programm. Bevor man also fortfährt, sollte man das Muster eines DATA-Programms (Abb. 5-16) eintasten. Will man dieses Programm stoppen, tastet man als nächste Schecknummer einfach eine Null ein.

```
1 REM ** DIESE PROGRAMM SCHREIBT EIN FILE **
2 REM ** PRUEFZAHLEN UND IHREN WERTEN **
5 DIM CHECKNAME$(40)
7 REM ** OEFFNEN MIT 8 ERZEUGT EIN DATA-FILE **
10 OPEN #1,8,0,"D:CHECKS"
20 CHECKAMT=0:CHECKNAME$=" "
25 ? " PRUEFZAHL";
30 INPUT CHECKNUM
35 IF CHECKNUM=0 THEN 80
38 REM ** SCHREIBE DATA ZUR DISKETTE **
40 ? " PRUEFWERT";
45 REM ** SCHLIESSE DATA-FILE **
50 INPUT CHECKAMT
55 REM ** OEFFNE DATA-FILE ZUM LESEN **
58 REM ** OEFFNEN MIT 4 IST NUR ZUM LESEN **
60 ? " WER WIRD GEPRUEFT MIT";
65 REM ** LESE DATA VON DISKETTE **
70 INPUT CHECKNAME$
75 REM ** DRUCKE DATA **
80 PRINT #1:CHECKNUM:IF CHECKNUM>0 THEN ? :GOTO 20
90 REM ** SCHLIESSE DATA-FILE **
100 CLOSE #1
```

OPEN und CLOSE

Um die Datenablage anzusprechen, z.B. den File D:CHECKS, benutzt man den BASIC-Befehl OPEN. Falls keine Ablage mit diesem Namen auf der Diskette ist, wird sie automatisch erstellt. Um Daten auf die Diskette zu bringen, bzw. von ihr zu holen, geht man so vor:

1. Schalte Plattengerät an.
2. Lege eine Systemdiskette ein.
3. Schalte den Computer ein und wähle das System an.
4. Taste OPEN#1,8,0,"D:DATA" und drücke RETURN. Das gestattet dem Computer, das File DATA in Plattenfach 1 zu erstellen.
5. Taste PRINT#1;X;",";Y;",";Z und drücke RETURN. X,Y und Z sind Zahlen.
6. Taste CLOSE#1 und drücke RETURN; das sagt dem Computer, das die Ablage beendet ist.

Abb. 5-17 stellt den Ablauf des Musterprogramms in Abb. 5-16 dar. Wir benutzen die Zahlen 100, 101 und 102 als CHECKNUMBER (Checknummer), die Zahlen 12.50, 24.35 und 102.67 als CHECKAMOUNT (Scheckbetrag), die Namen John Smith, George Brown und Heavy als WHO WAS THE CHECK TO? (Scheckempfänger).

```
READY
RUN
CHECK NUMBER?100
CHECK AMOUNT?12.50
WHO WAS CHECK TO?JOHN SMITH

CHECK NUMBER?101
CHECK AMOUNT?24.35
WHO WAS CHECK TO?GEORGE BROWN

CHECK NUMBER?102
CHECK AMOUNT?102.67
WHO WAS CHECK TO?HEAVY

CHECK NUMBER?0
```

Figure 5-17 Run of Sample Data Program

Abb. 5-18 zeigt, wie die Information von Abb. 5-17 auf der Diskette gespeichert wird.

```
100, 12.51, JOHN SMITH
101, 24.35, GEORGE BROWN
102, 102.67, HEAVY
0, 0
```

Figure 5-18 The Information Stored on Diskette

DAS ANSPRECHEN BESCHÄDIGTER ABLAGEN

Es gibt zwei Arten von Beschädigungen an Ablagen:
1. Beschädigungen innerhalb des Eingangs zum Plattenverzeichnis, das die Ablagennamen, die Verzeichniszeiger (die jeweils auf den ersten Sektor eines Fils zeigen) und die Sektorennummern enthält.
2. Beschädigungen an der Ablage selbst.

Sollte der Eingang des DISCDIRECTORY (Plattenverzeichnisses) beschädigt sein, kann man die Ablage nicht ansprechen. Falls das DISCDIRECTORY versehentlich gelöscht worden sein sollte, wird die Meldung ERROR-170 (File nicht gefunden) auf dem Monitor gezeigt. Wenn die Anzahl der auf dem DISCDIRECTORY ausgewiesenen Ablage nicht mit der tatsächlichen Anzahl von Sektoren in der Ablage übereinstimmt (kürzer ist), dann wird ERROR-164 (unpassende Ablagenanzahl) gemeldet. In diesem Fall kann man den durch das Verzeichnis abgedeckten Teil der Ablage ansprechen, indem man das GET-BYTE-Programm (Abb. 5-19) in Gang setzt.

```
10 OPEN #1,4,0,"D:FILE.1"
20 OPEN #2,8,0,"D:FILE.2"
30 TRAP 50
40 GET #1,A
50 GET #2,B
60 GOTO 40
70 CLOSE #1
80 CLOSE #2
```

Hier wird angenommen, daß Ablage 1 die Beschädigte ist, und Ablage 2 die noch Ansprechbare.

Anmerkung:
Man kann nur die Sektoren lesen, die VOR den beschädigten Sektoren liegen. Alle, die dahinter liegen, sind nicht erreichbar. Infolge dessen wäre es das Beste, die erhaltene Ablage von der schadhaften Diskette auf eine gute Diskette zu KOPIEREN, um zusätzliche Probleme zu vermeiden.
Wenn die Ablage selbst beschädigt ist, kann man auch das GET-BYTE-Programm anwenden, das dann alle guten Sektoren der schadhaften Ablage in eine Ersatzablage überträgt.

DIE AUTORUN.SYS-Ablage

Wenn ein AUTORUN.SYS-File auf der Diskette in Fach 1 existiert, wird dieses automatisch in das RAM geladen und ausgeführt (falls vorgesehen), jedesmal dann, wenn das System angewählt wird. Dieser gesamte Prozeß wird zuerst ausgeführt, bevor der Anwender die Kontrolle zurückerhält. Die AUTORUN.SYS-Ablage kann Daten, aber auch Objektcode enthalten. Letzterer wird dann zwar eingelesen aber nicht ausgeführt, oder er wird erst ausgeführt, wenn die Einlesung vollständig abgeschlossen ist.
Abb. 5-20 zeigt den Gebrauch des AUTORUN.SYS zum direkten Eingang in DOS, selbst dann, wenn eine Cassette angeschlossen ist. Nach Ablauf führt AUTORUN.SYS normalerweise zur Vorbereitungsroutine des DOS zurück. Wenn es bei einer Anwendung nicht zum Anfang zurückkehren sollte, oder wenn man den Gebrauch der Taste SYSTEMRESET zuläßt, BEVOR die Rückkehr einsetzt, so muß die Systemvorbereitung beendet werden, bevor AUTORUN.SYS abläuft.

Das geschieht durch Modifizieren von zwei Speicherstellen des Operationssystems: COLDST bei Adresse 244 (hexadezimal) und BOOT bei Adresse 9 (hexadezimal).
COLDST muß auf 00 gesetzt werden und BOOT auf 01.
Das unten gelistete Programm setzt diese beiden Adressen auf den richtigen Wert und springt dann indirekt auf den Vector für den DOS-Start.
Ist keine ASSEMBLER EDITOR-Cassette angeschlossen, so kann man eine entsprechende Ablage mit Hilfe der BASIC-POKE-Statements erstellen und dann das Binärfile im DOS aufbewahren. Die Liste der Dezimalzahlen, die man eingeben muß, sieht so aus:

Wenn diese Codes in BASIC eingegeben sind, taste DOS und drücke RETURN, um das File mit Hilfe der Wahlmöglichkeit K. BINARY SAVE des DOS-Menues abzuspeichern.

Es sei angemerkt, daß keine Zahl für den INIT-Parameter eingegeben wird. Wenn man den Computer aus- und dann wieder einschaltet, sollte man direkt das DOS anwählen. Um BASIC einzugeben, tastet man einfach B und drückt RETURN, oder man drückt nur SYSTEM RESET.

```
; Autorun Program
;
; Run DOS without going to cartridge.
;
COLDST = $244
BOOT = $09
DOSVEC = $0A
       * = 3A98
                                (HEX CODE)
DOSGO   LDX #0                  A2 00
        STX COLDST              8E 44 02
        INX                     E8
        STX BOOT                86 09
        JMP (DOSVEC)            6C 0A 00
        *= $2E0                 run address at 2E0
         WORD DOSGO             98 3A
         END
```

```
SELECT ITEM OR RETURN FOR MENU
K RETURN
SAVE-GIVE FILE, START, END-[,INIT,RUN]
AUTORUN.SYS, 3A98, 3AA2,, 3A98
SELECT ITEM OR RETURN FOR MENU
```

Figure 5-20 An AUTORUN.SYS Example for the Advanced User

ZUSÄTZLICHE INFORMATIONEN ÜBER DAS PLATTENSYSTEM ATARIDISKETTEN

ATARI-Disketten sind dünne, biegsame Scheiben mit einer Oxydschicht an der Oberfläche, ähnlich der auf Magnetbändern benutzten Schicht. Jede Diskette hat einen Durchmesser von 5 1/4 Zöll und ist in eine schwarze Spezialhülle eingeschlossen, die zum Schutz gegen Knicken, Verkratzen oder Verschmutzen dient.
Jedes Plattenlaufwerk erfordert eine speziell geeignete Diskettenart. Das Laufwerk ATARI 810 ist ein Gerät mit einfacher Aufzeichnungsdichte, das ATARI 815 ein solches mit doppelter Aufzeichnungsdichte.
Der wesentliche Unterschied zwischen beiden besteht darin, daß die Technik der doppelten Aufzeichnungsdichte eine Oberfläche von höherer Qualität erfordert, sodaß doppelt soviele Daten auf derselben Fläche untergebracht werden können.
Zwar werden beide Arten von Disketten auf gleiche Weise hergestellt, jedoch durchlaufen die Disketten mit Doppeldichte eine zusätzliche Qualitätskontrolle, die einwandfreies Arbeiten mit doppelter Dichte garantieren soll.
Eine leere, doppeltdichte Diskette kann ohne Weiteres auf einem Laufwerk mit einfacher Dichte verwendet werden, jedoch nicht umgekehrt.
Sobald eine Diskette für den Gebrauch in einem einfachdichten Laufwerk formatiert ist, kann sie nicht mehr für doppeltdichte Laufwerke genommen werden, es sei denn, man formatiert sie auf dem Gerät mit Doppeldichte neu. Umgekehrt gilt dasselbe.
Wenn in einem System beide Arten von Laufwerken und Disketten benutzt werden, muß man streng darauf achten, daß alle Disketten in klarer Weise etikettiert werden, sodaß man sofort die Art der Formatierung erkennen kann.

DAS LAUFWERK 810

Das Laufwerk 810 ist für eine einzige Platte, und zwar mit einfacher Aufzeichnungsdichte bestimmt. Es erfordert flexible, normale 5 1/4 Zoll-Disketten, nämlich die Masterdiskette II (CX8104), 810-Leerdisketten (CX8100), 810/815-Disketten (CX8202), sowie formatierte Disketten II (CX8111), von denen jede 88 K Bytes speichern kann.
Das ATARI 810-Laufwerk enthält einen eigenen Mikroprozessor, wodurch ihm eine automatisch wirkende Hilfskapazität zur Verfügung steht. Das bedeutet, daß der Laufwerkmotor nicht ständig läuft, sondern auf die besondere Anweisung hierzu wartet.

Das Laufwerk 815

Das Doppelgerät 815 enthält zwei Laufwerke und verwendet doppeltdichte Aufzeichnungstechnik. Ein einzelner Sektor auf einer mit diesem Gerät bearbeiteten Diskette kann 256 Datenbytes speichern, also zweimal soviel wie die Sektorkapazität auf dem Gerät 810 beträgt.
Jedes der beiden 815-Laufwerke hat seine eigene Gerätekennzahl.

DER BETRIEB DER PLATTENGERÄTE

Wenn man eine Diskette in das Laufwerk einschiebt, so wird das Achsloch in deren Mitte automatisch auf den Antriebszapfen gesetzt, und die Diskette ist plaziert. Die Diskette rotiert innerhalb ihrer Schutzhülle. Wenn man sie ansteuert, wird der Magnetkopf des Laufwerks über die Lese/Schreib-Oberfläche geschoben.

Beim Abspeichern von Daten auf einer Diskette verwandelt das Laufwerk die von der Computerkonsole kommenden Daten in codierte, elektrische Impulse. Diese magnetisieren winzige Bereiche der Oxydschicht auf jeder Diskette, während diese sich dreht.

Zum Ablesen von Daten stellt das Laufwerk den Magnetkopf so ein, daß der Bereich der Diskette, auf dem die zu lesenden Daten stehen, darunter liegt. Der Mikroprozessor des Laufwerks steuert Einstellung und Zeittakt der Diskette, bzw. des Magnetkopfes.

ANHANG A

ALPHABETISCHES VERZEICHNIS VON AUSDRÜCKEN, DIE FÜR BASIC RESERVIERT SIND UND FÜR PLATTENOPERATIONEN EINGESETZT WERDEN

Anmerkung: Der Punkt hinter allen abgekürzten Schlüsselworten ist unabdingbar.

Reserviertes Wort	Abkürzung	Kurzbeschreibung des BASIC-Statements
CLOSE	CL.	I/O-Statement, das benutzt wird, um ein Plattenfile nach Ende der INPUT/OUTPUT Operationen zu schließen.
DOS	DO.	Dieser Befehl veranlaßt, daß das Menue projiziert wird. Das Menue enthält alle DOS- Anwahlen. Kontrolle geht von Cassette auf DOS-Routinen über.
END		Stoppt die Programmausführung, schließt Files und dreht Geräusche ab. Programm kann neu gestartet werden mit Hilfe von CONT. (Anmerkung: END kann mehr als einmal in einem Programm verwendet werden).
ENTER	E.	I/O-Befehl, der zum Auffinden eines gelisteten Programms in ungekennzeichneter (textlicher) Form verwendet wird. Wird ein Programm oder Programmzeilen eingelesen, solange ein anderes noch im RAM steht, dann vermischt ENTER diese beiden Programme. Will man dieses nicht, muß man NEW tasten, bevor ENTER zum Einlesen eines Programmes in das RAM verwendet wird.

GET	GE.	Wird bei Plattenoperation benützt, um ein einzelnes Byte in eine bestimmte Variable von einem bestimmten Gerät her einzugeben.
INPUT	I.	Dieser Befehl verlangt Daten von einem bestimmten Gerät. Das ersatzweise angesteuerte Gerät ist E: (Monitor).
LIST	L.	Dieser Befehl gibt die ungekennzeichnete Version eines Programms an ein bestimmtes Gerät weiter.
LOAD	LO.	I/O-Befehl, der zum Auffinden eines abgespeicherten Programms in gekennzeichneter Form auf einem spezifizierten Außengerät verwandt wird.
NOTE	NO.	Dieser Befehl speichert die absolute Sektornummer und die laufende Bytenummer des File- Zeigers in seine zwei arithmetischen Variablen.
OPEN	O.	öffnet das angesprochene File zum Eingeben oder Ausgeben. Legt die Art von Operationen fest, die für ein bestimmtes File erlaubt sind.

POINT	P.	Dieser Befehl wird eingesetzt, zum Setzen des File-Zeigers auf eine bestimmte Stelle (Sektor und Byte) auf der Diskette.
PRINT	PR.or?	I/O-Befehl, veranlaßt die Ausgabe vom Computer an eine spezifizierte Ausgabeadresse in Record-Format.
PUT	PU.	Veranlaßt die Ausgabe eines einzelnen Bytes, d.h. eines Zeichens, vom Computer an ein spezifiziertes Gerät.
RUN	R.	Lädt und startet Ausführung eines bestimmten Filespec.
SAVE	S.	INPUT/OUTPUT-Statement, das die gekennzeichnete Version eines Programms in eine bestimmte Ablage auf einem bestimmten Gerät schreibt.
STATUS	ST.	Ruft STATUS-Routine für bestimmtes Gerät.
TRAP	T.	Veranlaßt Weiterführung des Programms auf einer bestimmten Zeilennummer, falls ein Programmfehler erscheint, so daß der Anwender das Programm kontrollieren und Fehler beheben kann.
XIO	X.	Allgemeines INPUT/OUTPUT-Statement, das in einem Programm benutzt wird, um DOS- Menu-Wahlen anzubieten und bestimmte INPUT-/OUTPUT-Befehle auszuführen.

ANHANG B

BEGRIFFE UND TERMINOLOGIE IM ZUSAMMENHANG MIT DOS II

SYSTEMRESET

Ücke die SYSTEMRESET-Taste auf der Tastatur

RETURN

Ücke die RETURN-Taste auf der Tastatur

()

ammern. Sie schließen ein, was wahlweise ist.

...

slassung. Wenn eine Auslassung hinter einem Element in Klam-rn steht, so bedeutet dieses, daß man das wahlweise anzugebende ement so oft nennen kann, wie man will, jedoch dazu nicht zwungen ist.

()

schweifte Klammern.Element, die in solchen Klammern ereinander stehen, sind gegeneinander austauschbar. Bei nfügung in einen Befehl oder in ein Statement nur eines davon rwenden!

GROSSBUCHSTABEN

t Großbuchstaben werden Befehle, Statements und andere nktionen geschrieben, so wie alles, was genau nach Vorlage ngetastet werden muß.

, . / : ; "

terpunktionszeichen. Sie müssen genau nach Vorschrift in das rmat eines Befehls oder Statements eingefügt werden. Klammern nd in solchen Fällen verboten.

cmdno

fehlsnummer (Comand-Number). Wird in XIO-Befehlen verwandt.

exp

sdruck (Expression). In diesem Handbuch gibt es davon drei ten: arithmetische, logische und stringförmige.

aexp

ithmetischer Ausdruck, im Allgemeinen bestehend aus einer riablen, Funktion, Konstanten oder zwei arithmetischen sdrücken, die durch einen arithmetischen Operator (aob) trennt sind.

aexp1

ithmetischer Ausdruck 1. Dieser stellt das erste lfskontrollbyte bei INPUT/OUTPUT dar, wenn er in Befehlen wie EN vorkommt.

aexp2

Arithmetischer Ausdruck 2. Dieser stellt das zweite Hilfskontrollbyte bei INPUT/OUTPUT dar, wenn er in Befehlen wie OPEN vorkommt. Gewöhnlich steht er auf 0. , wenn der ATARI-820-Drucker zusätzlich drucken soll, muß dieser Ausdruck auf 83 gesetzt werden.

filespec

File-Spezifikation. Gewöhnlich ein Stringausdruck, der sich auf ein File und dessen zugehöriges Gerät bezieht, z.B. "D1:MYPROG.BAS" für ein File in Laufwerk 1.

IOCB

INPUT/OUTPUT-Kontrollblock. Ein arithmetischer Ausdruck, der von 1 bis 7 bewertet sein kann. Das IOCB dient zum Bezug auf ein Gerät oder File. IOCB 0 ist für BASIC-Monitor-Programme reserviert und sollte nur benutzt werden, wenn der Monitor nicht in Aktion ist.

lineno

Line-Number. Eine Konstante, die eine bestimmte Programmzeile in einem BASIC-Programm in zurückgestelltem Gang kennzeichnet. Eine Zeilennummer kann jede ganze Zahl von 0 bis 32.767 sein. Die Zeilennummerierung legt die Reihenfolge der Programmausführung fest.

var

Variable. In unserem Zusammenhang gibt es arithmetische Variablen (avar), Matrix-Variablen (mvar) und String-Variablen (svar).

avar

Arithmetische Variable. Eine Stelle, an der ein Zahlenwert gespeichert ist. Variablennamen können 1 bis 120 alphanumerische Zeichen lang sein, müssen jedoch mit einem Großbuchstaben beginnen.

mvar

Matrix-Variable. Wird auch indexierte Variable genannt. Element eines Arrays oder einer Matrix.

svar

String-Variable. Eine Stelle, wo ein Zeichenstring gespeichert werden kann.

ANHANG C

FEHLERMELDUNGEN UND DIE DABEI ERFORDERLICHEN OPERATIONEN

erkung: Die Fehler 2 bis 21 können nur beim Ablauf von BASIC-Programmen vorkommen.

Speicherplatz ungenügend

ist nicht genügend Platz zum Speichern des Statements, bzw. Dimensionierung der neuen String-Variablen vorhanden. Lösche e unnötigen Variablennamen oder vergrößere den Speicherraum rgl.
l 11 des BASIC-Handbuches betr. Speicherraum).

Wertfehler (Value ERROR)

weder war die erwartete positive ganze Zahl negativ, oder der lenwert lag nicht im erwarteten Bereich.

Zu viele Variablen

maximale Zahl (128) von Variablennamen wurde überschritten. o müssen alle nicht mehr zu verwendenden Variablen gelöscht den (vergl. Teil 11 des BASIC-Handbuchs).

Stringlängen-Fehler

wurde versucht, von einer, bzw. in eine Speicherstelle zu en, die jenseits der dimensionierten Stringlänge liegt, oder hat 0 als Bezugsindex benutzt. Vergrößere das DIM! Nimm nicht ls Index!

Fehlende Daten (out of Data)

sind nicht soviele Daten in den DATA-Statements, wie die READ-tements erfordern.

Zeilennummer größer als 32767

fe die Zeilennummern-Angaben in Statements wie GOTO und TORE.

Fehler im Eingabe-Statement

wurde versucht, einen nichtnumerischen Wert in eine numerische iable einzugeben. Die Arten der Variable und die Eingabedaten d zu prüfen.

9 ARRAY- oder STRING-DIM- Fehler

Die DIM überschreitet 5460 numerische Arrays, bzw. 32767 für Strings; ein Array oder String kann neu dimensioniert worden sein, oder es kann ein Bezug auf ein undimensioniertes Array, bzw. einen undimensionierten String vorliegen.

11 Gleitkomma Über-, bzw. Unterschreitung

Es wurde versucht, durch 0 zu dividieren oder mit einer Zahl zu arbeiten, deren absoluter Wert kleiner als 1E-99 oder gleich 1E-98 ist.

12 Zeile nicht gefunden

Ein GOSUB, GOTO oder THEN bezog sich auf eine nicht existierende Zeilennummer.

13 FOR paßt nicht

Das Programm ist auf ein NEXT-Statement ohne passendes FOR gestoßen.

14 Zeile zu lang

Die Pufferlänge für BASIC-Zeilen wurde überschritten.

15 GOSUB- oder FOR-Zeile gelöscht

Das Programm ist auf ein NEXT- oder RETURN-Statement gestossen und das zugehörige FOR oder GOSUB war seit der letzten Benutzung des Programms bereits gelöscht.

16 RETURN-Fehler

Das Programm muß auf ein fehlendes GOSUB-Statement hin überprüft werden.

17 Syntax-Fehler

Der Computer hat eine Zeile mit fehlerhafter (unzulässiger) Formulierung angetroffen. Die Zeile ist herauszusuchen und zu korrigieren.

18 Fehler in der VAL-Funktion

Das String in einem VAL-Statement ist nicht numerisch.

Ladeprogramm (LOAD) zu lang

ist nicht genügend Speicherplatz für das einzulesende Programm handen.

Gerätenummer falsch

wurde eine Gerätenummer eingegeben, die nicht zwischen 1 und 7 .

LOAD-File-Fehler

wurde versucht, eine lesegesperrte Ablage zu lesen, statt er gekennzeichneten BASIC-Ablage. Gekennzeichnete Ablagen den mit dem SAVE-Befehl erstellt.

erkung: Folgende Fälle sind INPUT/OUTPUT-Fehler, die beim satz von Plattengeräten, Druckern und anderen Zusatzgeräten kommen können. Detailinformationen kann man den Anleitungen zu Geräten selbst entnehmen.

BREAK-Fehler

Benutzer hat während einer INPUT/OUTPUT-Operation die BREAK- te gedrückt. Dadurch wird der Ablauf gestoppt.

Das IOCB* ist schon offen

OPEN-Statement innerhalb einer Programmschleife oder das IOCB schon für eine andere Ablage, bzw. ein anderes Gerät belegt.

Nicht existierendes Gerät

wurde versucht, ein Gerät anzusprechen, das nicht in der ensteuerungstafel verzeichnet ist, d.h. ein undefiniertes ät. Dieser Fehler kann vorkommen, wenn versucht wird, das RI 850-Interfacemodul ohne Benutzung der RS-232-C AUTORUN.SYS- age zu benutzen. Häufig tritt dieser Fehler auch dadurch auf, man einen Ablagennamen spezifiziert, ohne ein Gerät ugeben, z.B. indem man formuliert "MYFILE" statt "D:MYFILE".

muß dann den INPUT/OUTPUT-Befehl auf Geräteangabe hin rprüfen und schließlich die richtige Gerätesteuerung einlesen l starten.

*IOCB = INPUT/OUTPUT-CONTROL BLOCK

IOCB nur schreibend

wurde versucht, eine nur schreibende Ablage zu lesen. Diese age muß zuerst für Lesung, bzw. Änderung (update) geöffnet den.

132 Unzulässiger Befehl

für Gerätesteuerung Es handelt sich um einen Code für CIO-Fehler. Der normale Code, der an die Gerätesteuerung gegeben wurde, ist nicht zulässig. Der Befehlscode ist entweder kleiner oder gleich 2 oder ist ein Spezialcode für eine Steuerung, für die keine Spezialcodes vorgesehen sind.

Die Codierung des XIO- oder IOCB-Befehls muß auf unzulässige Befehlselemente hin überprüft werden.

133 Gerät, bzw. Ablage nicht offen.

Die angesprochene Ablage, bzw. das angesprochene Gerät ist nicht geöffnet worden. Das OPEN- oder I/O-Statement muß auf falsche Spezifikation hin überprüft werden.

134 Falsche IOCB-Nummer

Es wurde versucht, einen unzulässigen IOCB-Index anzuwenden. Der zulässige Bereich für BASIC ist 1 bis 7, da BASIC den Einsatz von IOCB 0 nicht erlaubt.
Die Assembler-Editor-Cassette erfordert einen IOCB-Index, der ein Vielfaches von 16 und kleiner als 128 ist.

135 IOCB nur lesend

Es wurde versucht, ein Gerät, bzw. eine Ablage anzuschreiben, die nur für Lesen geöffnet war. Der betreffende File muß also zum Schreiben, bzw. Abändern geöffnet werden (read/write).

136 End of File (Ablagenende)

Das Eingabefile ist zu Ende, es enthält keine Daten mehr.

137 Abgeschnittene Datenserie

Typischerweise entsteht dieser Fehler dann, wenn die Datenserie (Record), die gelesen wird, die maximale im CIO-Anruf spezifizierte Serienlänge überschreitet (bei BASIC beträgt diese 113 Bytes).

Wenn man versucht einen INPUT-Befehl (der serienorientiert ist) für eine Ablage einzusetzen, die mit PUT-Befehlen (die byteorientiert sind) erstellt wurde, so kommt diese Art Fehler zustande.

138 Gerät nicht zeitgleich (Timeout)

Man hat einen Befehl über den seriellen Kanal geschickt und das angesprochene Gerät hat nicht innerhalb der vom O.S. vorgesehenen Zeitspanne geantwortet.

ıtweder ist die Gerätenummer falsch, oder der Benutzer hat ırsehentlich ein anderes Gerät spezifiziert. Das Gerät ist ıßerstande, innerhalb der vorgeschriebenen Zeit zu antworten. ımöglich ist es nicht richtig angeschlossen.

ənn es sich um eine Cassette handelt, kann es sein, daß deren ıudrate falsch gemessen wurde oder daß das Band nicht richtig ingelegt ist.

ın muß in diesem Fall alle Verbindungen überprüfen und ıchsehen, ob das Plattengerät eingeschaltet und auf die richtige ınnzahl eingestellt ist.

·üfe den Befehl in Hinsicht auf die Laufwerkkennzahl! Lasse ihn ɔch einmal ablaufen! Wiederholt sich der Fehler, müssen die attenlaufwerke technisch überprüft werden.

39 NAK im Gerät.

ıs Gerät kann nicht antworten, weil falsche Parameter, z.B. ein ıcht adressierbarer Sektor, vorliegen. Vielleicht hat das Gerät ɔch einen verstümmelten oder unzulässigen Befehl, bzw. unklare ıten vom Computer erhalten.

ın muß dann den I/O-Befehl auf unzulässige Parameter überprüfen ıd den Befehl noch einmal ablaufen lassen. Außerdem sind die ərbindungskabel zu prüfen. Es handelt sich hier um einen ərätefehler, so daß man auch auf die Unterlagen für das ətreffende Gerät zurückgreifen muß.

40 Fehler im seriellen Verbindungssystem

it 7 von SKSTAT im POKEY-Chip ist gesetzt. Das bedeutet, daß die ərbindung zwischen Außengerät und Computer gestört ist.

41 Läufer außer Bereich

ər Läufer ist außerhalb des für den gewählten Graphikgang ɔrgesehenen Bereichs geraten. Die Pixel-Parameter sind zu ıdern.

42 Serieller Kanal überlaufen

it 5 vom SKSTAT im POKEY ist gesetzt. Der Computer antwortete ıcht schnell genug auf eine serielle Eingabe oder das POKEY rhielt ein zweites 8-Bit-Wort über den seriellen Kanal, bevor ər Computer das vorige Wort abarbeiten konnte.

·ieser Fehler kommt selten vor. Tritt er mehr als einmal auf, so uß der Computer repariert werden.

143 Prüfsummen-Fehler

Die Komunikation über den seriellen Kanal ist verstümmelt. Die vom Gerät ausgesandte Prüfsumme ist nicht die selbe, wie die für den betreffenden Datenrahmen vom Computer errechnete. Für diesen Fall gibt es keine Standardabhilfe, da sowohl ein Hardware-, als auch ein Software-Fehler vorliegen kann.

144 Geräteblockade (Devise Done)

Das Gerät kann einen gültigen Befehl nicht ausführen. Entweder wurde versucht, auf ein schreibgeschütztes Gerät zu schreiben, oder das Gerät (z.B. Diskette) ist nicht in der Lage, auf den angesprochenen Sektor zu schreiben, bzw. von dort zu lesen. Die Schreibschutzkappe muß entfernt, bzw. der Schreibschutzschalter umgestellt werden. Vergl. hierzu die Spezialanweisungen zu den Geräten.

145 Unzulässiger Bildgang

Es wurde versucht, die Bildausgabe mit einem unzulässigen Graphikmode zu eröffnen. Der Anruf für den Graphikgang ist zu prüfen, ebenso das Byte AUX2 im IOCB.

146 Funktion nicht eingeführt

Die Gerätesteuerung enthält die betreffende Funktion nicht; z.B. wurde versucht, PUT an die Tastatur zu richten oder spezielle Befehle an die Tastatur zu geben. Der I/O-Befehl ist daraufhin zu überprüfen.

147 Ungenügendes RAM

Es ist nicht genügend RAM für den gewählten Graphikgang vorhanden. Zusätzlicher Speicherraum ist frei zu machen oder es ist ein Graphikgang zu wählen, der weniger Platz beansprucht.

160 Plattenfach falsch

Es wurde ein Plattenfach außerhalb des Bereichs 1 bis 8 codiert, oder es wurde kein Puffer für das Laufwerk bereitgestellt, oder das Plattengerät war im Moment des Ansprechens nicht eingeschaltet. Vergl. Teil 1 dieses Handbuchs. Prüfe den Filenamen, bzw. Byte 1902 in Hinsicht auf die Gerätenummer, bzw. die eingerichteten Puffer.

Zuviele Ablagen OPEN

sind keine freien Sektorenpuffer für die Benutzung weiterer
es frei. Prüfe Stelle 1801 auf die Anzahl bereitgestellter
fer! Prüfe außerdem, ob nicht Files geöffnet sind, die nicht
en sein dürften!

Platte voll

der betreffenden Diskette sind keine Sektoren frei. Eine
ere Diskette mit freien Sektoren muß eingeschoben werden.

Unbehebbarer Fehler I/O-System

se Meldung bedeutet, daß die Filesteuerung defekt ist. Das DOS
r die Diskette können schadhaft sein. Ein anderes DOS
utzen!

Filenummer paßt nicht

Struktur der Ablage ist beschädigt, oder die POINT-Werte sind
sch. Eine der Fileverbindungen zeigt auf einen Sektor, der
em anderen File zugewiesen ist. Schalte das System ab und
rte das Programm neu! Mißlingt dies, ist die betreffende
age verloren. Versuche, die übrigen Files von der Diskette zu
ten und diese dann neu zu formatieren!

Filename falsch

Filespezifikation enthält unzulässige Zeichen. Dies ist zu
rprüfen und die entsprechenden Elemente sind zu entfernen.

Falsche Datenlänge in POINT

Bytezählung im POINT-Anruf war größer als 125 (bei
fachdichte), bzw. 253 (bei Doppeldichte). Überprüfe die
ameter im POINT-Statement!

Fehlercode 167 File gesperrt

Es wurde versucht, einen gesperrten File nicht zum Lesen, sondern zu anderen Zwecken anzusprechen. Benutze die DOS-Anwahl G zum Entsperren und versuche die Anwendung des Befehls erneut!

Fehler Nr. 168 Gerätebefehl ungültig.

Es wurde ein unzulässiger Befehl an der Software-Interface dieses Gerätes gegeben. Überprüfe die Unterlagen, die sich auf das Gerät beziehen, und versuche den Einsatz des Befehls erneut!

Fehler 169: Verzeichnis voll

Der gesamte Raum für das Plattenverzeichnis ist aufgebraucht

Fehler 170: File nicht gefunden.

Ein nicht im Plattenverzeichnis vorhandener File wurde angesprochen. Benutze die DOS-Anwahl A, um zu prüfen, ob der Filename korrekt geschrieben ist, und überprüfe , ob dieser Name auch auf der angesprochenen Diskette steht!

Fehler 171: POINT ungültig!

Es wurde versucht, den POINT-Befehl an einen File zu richten, der nicht für die UPDATE - Operation geöffnet ist. Überprüfe die Parameter des OPEN-Statements oder das Hilfsbyte 1 (aux1) des IOCB, das zum öffnen des Files dient!

Fehler 172: Unzulässige Ergänzung.

Es wurde versucht, mit Hilfe von II eine Ergänzung an eine Ablage anzuhängen, die mit DOS I erstellt wurde. DOS II kann jedoch keine Ergänzungen an DOS I - Files hängen. Kopiere die DOS I -Ablage auf eine DOS II -Diskette mit Hilfe von DOS II.

Fehler 173: Schadhafte Sektoren während Formatierung.

Das Plattengerät hat beim Formatieren defekte Sektoren gefunden. Es ist eine andere Diskette zu benutzen, da eine Diskette mit schadhaften Sektoren nicht formatiert werden kann.

Wenn dieser Fehler bei mehreren Disketten auftritt, muss das Laufwerk repariert werden.

NG D

II - Speicherplan für das 32K-System

ADDRESS Decimal	ADDRESS Hexadecimal	CONTENTS	
65535	FFFF	OPERATING SYSTEM	
49152	C000		
49151	BFFF	CARTRIDGES	
32768	8000		
32767	7FFF	SCREEN DISPLAY AREA	
varies			
varies		USER PROGRAM AREA	HIMEM**
varies			LOMEM*
2S: 3305	2D: 3A38	SB7 SB6 SB5 SB4 SB3	DISK UTILITY PROGRAMS
2S: 1D7C	2D: 247C		
6781	1A7D	Sector Buffer 2 Sector Buffer 1 Drive 4 Buffer Drive 3 Buffer Drive 2 Buffer Drive 1 Buffer	BUFFER AREA RESERVED FOR DOS II
6780	1A7C	MINI-DOS (RAM RESIDENT PORTION OF DUP)	
5440	1540		
5439	153F	FILE MANAGEMENT SUBSYSTEM	
1792	0700		
1791	06FF	USER RAM	
	0600		
	05FF		
0	0000	OPERATING SYSTEM RAM	

Anmerkung 1:

Für eine gegebene Bereitstellung von Laufwerk- bzw. Sektor-Pufferbereichen kann LOMEN durch die PEEK-Register 2E7 (LOW) und 2E8 (HIGH) hexadezimal oder 743 (LOW) und 744 (HIGH) dezimal bestimmt werden.

Anmerkung 2:

Um die Grösse des für Anwenderprogramme oder für HIGH MEM verfügbaren Platzes zu bestimmen, kann man entweder die Instruktion BASIC FRE (0) oder PEEK-Register 2E5 (LOW) und 2E6 (HIGH) hexadezimal bzw. 741 (LOW) und 742 (HIGH) dezimal einsetzen.

ANHANG F:

Die Beschleunigung von Datenübertragungen an die Plattengeräte

Die neue Version 2.0S von DOS II ist fähig, mit Leseprüfung aufzuzeichnen. Dies ist eine Sicherheitstechnik, die immer dann genutzt werden sollte, wenn besondere Zuverlässigkeit wichtiger ist als schnelle Übertragung. In dieser Ausstattung wird die DOS II - Masterdiskette ausgeliefert. Zur Zeiteinsparung kann die Information jedoch ohne die Leseprüfung auf Diskette geschrieben werden.

Die Speicherstelle 1913 (dezimal) enthält die Daten, die bestimmen, ob die Filesteuerung mit Leseprüfung aufgezeichnet (50 hexadezimal, 80 dezimal) oder ohne diese (57 hexadezimal, 87 dezimal).

Das Aufzeichnen ohne Leseprüfung läuft natürlich schneller, ist jedoch nicht so zuverlässig.

Um die vorhandene DOS II-Version über BASIC anzupassen, setzt man POKE 1913,80 für schnelle Aufzeichnung (ohne Leseprüfung). Will man lieber mit Leseprüfung aufzeichnen, so setzt man POKE 1913,87.

Um die auf der Diskette stehende DOS-Version zu ändern, so daß sie als Gebrauchsversion immer passt, tastet man einfach DOS und gibt dann den Befehl "H" (WRITE DOS -FILES) an das DOS MENUE, um die neue DOS-Versin aus dem RAM auf die Diskette zu speichern.

G:

Die Erweiterung des Anwenderbereichs im RAM

Abschnitt erklärt, wie man die Anzahl der Speicherregister , die der Anzahl von angeschlossenen Plattenlaufwerken icht. Man muss den korrekten LAufwerkcode in Register ezimal) eingeben (durch POKE). Die folgende Tafel liefert rrekten Zahlen entsprechend der Anzahl von 810-Geräten. t zu beachten, dass nur bis zu 4 810- Geräte bei DOS II zbar sind, da die Schalter an jedem Gerät nur von 1-4 n.

ist zu beachten, dass im binären Laufwerkcode eine 1 für Laufwerk des Systems vorhanden ist.

an früherer Stelle empfohlen wurde, die 815-Laufwerke mit 2 zu bezeichnen und das Laufwerk 810 mit 3, können auch Möglichkeiten zutreffen, wie man unten ersieht.

Codes für jeweils angeschlossene 810-Laufwerke

Drives Allocated	Decimal Drive Code	Binary Drive Code
Drive 1	01	00000001
Drive 2	02	00000010
Drive 3	04	00000100
Drive 4	08	00001000
Drives 1+2	03	00000011 (default)
Drives 1+3	05	00000101
Drives 1+2+3+4	15	00001111

das vorhandene System sowohl 810ner wie 815 - Geräte t, müssen die 810ner Geräte in Position 1-4 eingesetzt . Die Doppellaufwerke können per Schalter auf folgende aare eingestellt werden: 1+2, 3+4,5+6 oder 7+8. Die de Tafel zeigt die korrekten Eintragungen für die jeweilige von Plattenlaufwerken. Diese Daten sind gleichzeitig im gister 1802 (dezimal) gespeichert.

Drives 1+2	03	00000011
Drives 3+4	12	00001100
Drives 5+6	48	00110000
Drives 7+8	192	11000000

el 1:

n 810-Laufwrk und ein 815 Doppellaufwerk angeschlossen, so as DOS wie folgt aufzubauen:

aufwerk 1 benutzt werden muss, hat man entweder die 815 auf 2 einzustellen und die 810 auf 3, oder man muss die 810 auf die 815 auf 2 und 3 einstellen. Im ersten Fall lautet der de Anwahlcode 3 (s.Tafel oben) für die beiden 815 - erke und 4 für das Position 3 gesetzte 810-Laufwerk. Der code, der im Register 1802 einzugeben ist, lautet dann also 04 = 07 (dezimal) bzw. 0000 111 (binär).

Beispiel 2:

Angenommen, man hat 3 810- Laufwerke und 2 815-Doppelgeräte:
Da die 810 nur auf Position 1-4 einstellbar ist, so müsste man für diese 01,02 und 03 verwenden. Bei entsprechender Plazierung der Doppelgeräte in Position 5+6 bzw. 7+8 kämen die Codes 48 und 192 heraus(s.Tafel).

Den Gesamtwert der Codierung erhält man wiederum durch Addition:
01+02+03+48+192=246. Dieser Wert (246)muss dann in das Register 1802 eingegeben werden. Eine dezimale 246 entspricht der Binärzahl 11110110.

IG H

unterschiede zwischen DOS I (9/24/79) und DOS II

	DOS II
[	
ıl zeit betrug 11 Sekunden	Anwahlzeit beträst 7 Sekunden
:eit für die Platten-Dienst- nen war = 0, da während der ıl geladen wurde	Ladezeit für DUP beträst 9 Sekunden
ı wilden Karten für Kopie- /erfügbar	Wilde Karten verfügbar, mit deren Hilfe man jedes beliebige File von einer Diskette auf die andere bringen kann
ı MEM.SAV-Funktion	MEM.SAV-Funktion verfügbar, die dem Anwender mehr Speicherraum zur Verfügung hält. Dadurch leistungsfähigeres DOS.
rdert Anwahl von einem)r Länge	Erfordert Anwahl von 3 Sektoren Länge, mit deren Hilfe DOS II doppelt dichte Laufwerke steuern kann. DOS I und DOS II sind daher nicht ohne weiteres austauschbar.
AUTORUN.SYS	Besitzt AUTORUN.SYS-Ablage, die es ermöglicht, einen File sofort nach Einlesung arbeiten zu lassen
S werden im kleineren Puffer- ich kopiert bzw. dupliziert	Ganze FILES werden in beliebig grossen Pufferbereich kopiert bzw. dupliziert. Der Bereich kann die Grösse des Anwenderbereichs haben.
te keine Files koppeln	Die Option "/A" der Funktion SAVE Binery File erlaubt die Koppelung zweier Files
keine Lese-Lauf-Funktion D and GO)	Kann Lese-Lauf-version von Files erstellen, wodurch es möglich ist, einen File einzulesen und automatisch arbeiten zu lassen, ohne dass man

	eine RUN-Adresse eingibt
Schadhafte Sektoren auf Diskette wurden bei Formatierung nicht angezeigt	Disketten mit schadhaften Sektoren können nicht formatiert werden
Ränder wurden nicht automatisch ausgeglichen	Ränder werden automatisch auf Originalposition gebracht, sobald DOS II eingegeben wird
Nur eine Version von DOS:System mit einfacher Dichte	Zwei Versionen von DOS II, nähmlich einfache Dichte und doppelte Dichte der Plattenaufzeichnung
Muste Menue neu zeigen, bevor neuer Befehl angesteuert werden konnte	Wahl zwischen neuer Projezierung des Menues und Eingabe des nächsten Befehls
Konnte DOS-File nur auf Platte 1 bringen	Kann DOS-Files auf beliebige Platte bringen
Konnte nur 3 Files gleichzeitig öffnen	Kann bis zu 8 Files gleichzeitig öffnen
NOTE/POINT nicht beliebig adressierbar	NOTE/POINT beliebig adressierbar

ANHANG I:

Der Aufbau einer kombinierten Ablage

Kombinierter Ablagen-Aufbau unter Benutzung des C.COPY-Files mit Ergänzung (APPEND)

COMPOUND FILE STRUCTURE USING C. COPY FILE WITH APPEND

Byte No.	Decimal No.	Hex No.	Description
1	255	FF	Identification Code (PART 1)
2	255	FF	
3	0	00	Starting Address (PART 1)
4	80	50	
5	31	1F	Ending Address (PART 1)
6	80	50	
.	.	.	DATA (PART 1)
.	.	.	32 Bytes
.	.	.	
38	255	FF	Identification Code (PART 2)
39	255	FF	
40	32	20	Starting Address (PART 2)
41	80	50	
42	143	8F	Ending Address (PART 2)
43	80	50	
		.	DATA (PART 2)
.	.	.	112 Bytes
.	.	.	

COMPOUND FILE STRUCTURE USING K. BINARY SAVE WITH APPEND

Byte No.	Decimal No.	Hex No.	Decscription
1	255	FF	Identifier Code
2	255	FF	
3	00	00	Starting address (PART 1)
4	80	50	
5	31	1F	Ending address (PART 1)
6	80	50	
.			Data (PART 1)
.	.	.	32 Bytes
.	.	.	
38	32	20	Starting address (PART 2)
39	80	50	
40	143	8F	Ending address (PART 2)
41	80	50	
			Data (PART 2)
			112 Bytes

ANHANG J:

Verzeichnis der Fachausdrücke:

ADRESSE:
Eine Speicherstelle, die gewöhnlich durch eine 2 Bytes grosse Zahl in hexadezimaler oder dezimaler Form wiedergegeben wird (maximaler Bereich 0-FFF hexadezimal).

ALPHANUMERISCH :
Die Grossbuchstaben A-Z und die Ziffern 0-9 bzw. Kombinationen aus beiden Elementen. Ausgeschlossen sind bei diesem Begriff Graphiksymbole, Interpunktionszeichen und andere Sonderzeichen.

ANWAHL- (BOOT) PROGRAMM:
Das Startprogramm, dass dm Computer "aufbereitet", nachdem er eingeschaltet wurde. Am Schluss des Anwahlprogramms (nach dem Aufbereiten) ist der Computer bereit, höhere Programme einzulesen und auszuführen.

ARRAY :
Ein eindimensionaler Satz von Datenelementen. Es können einzelne ARRAYS und auch ganze Listen von ARRAYS angesprochen werden, indem man den variablen Namen + einem Zusatz eingibt. Z.B. das ARRAY B Element Nr. 10 würde als B(10) adressiert. Man beachte, dass String-ARRAYS von BASIC nicht berücksichtigt werden, dass man jedoch jedes einzelne Stringelement einzeln herauslesen kann: z.B. A$(10). Alle ARRAYS müssen vor dem Gebrauch dimensioniert werden. Eine MATRIX (s. dort) ist ein 2-dimensionales ARRAY.

ATASCII :
Die zum Abspeichern von Text verwandte Methode. Innerhalb des ATASCII-Systems ist jedem Zeichen und Graphiksymbol sowie den meisten Steuertasten eine Kennzahl zugewiesen. Diese Zahl ist ein Ein-Byte-Code(dezimal 0-2559). (ATASCII ist eine Modifikation von ASCII, des AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE) Vergl. die entsprechende Tafel im Handbuch für ATARI BASIC.

AUTORUN.SYS:
Ein Programm-Name, der für das Disketten-Operationssystem reserviert ist.

BAUD RATE :
Signalgeschwindigkeit oder Geschwindigkeit der Datenübertragung in Bits pro Sekunde.

Begrenzer (DELIMITER) :
Ein Zeichen, dass Beginn oder Ende eines Datenbegriffs markiert, jedoch nicht Teil dieses Begriffs ist. So sind z.B. die Anführungsstriche(") Begrenzer für Strings in den meisten BASIC - Systemen.

BINÄR-ABSPEICHERUNG :
Das Abspeichern eines Objektprogramms in (binärer) Maschinensprache auf eine Platte oder eine Programmkassette.

BINÄR - LESUNG :
Das Einlesen eines Objektprogramms in (binärer) Maschinensprache in den Computerspeicher.

BIT :
Abkürzung für "BINARY DIGIT"(binäre Ziffer). Die kleinste Informationseinheit. Sie kann 0 oder 1 enthalten.

BREAKE (Abbruch) :
Zum Zweck der Programmunterbrechung. Das Drücken der Taste "BREAKE" veranlasst eine solche Unterbrechung.

BYTE:
Ursprünglich Abkürzung für "BINARY TACHTALT", also "BINÄRE VIERERGRUPPE". Heute eine Gruppe von 8 Bits. Ein Byte kann ein Zeichen enthalten. Es hat einen Dezimal-Zählbereich von 0-255.

CIO :
Abkürzung für CENTRAL INPUT/OUTPUT-SYSTEM (Zentrales Eingabe/Ausgabe-System). Der Teil des Operationssystems , der die Eingabe und Ausgabe steuert.

CLOSE (Schliessung) :
Beendet den Zugriff zu einer Plattenablage. Bevor man die Ablage wieder erreichen kann, muss sie neu geöffnet werden (Vergl. OPEN).

DATEN :
Informationen jeder Art, gewöhnlich eine Serie von Bytes.

DEZIMAL :
Ein Zahlensystem, dass die Ziffern 0-9 benutzt. Dezimalzahlen stehen in binärcodierter Form im Computer (Vergl. BIT, Hexadezimal und Octal).

DICHTE:
Die Enge der Platzverteilung auf einem Speichermedium, dass heisst hier: Die Anzahl von Bytes pro Sektor. Platten mit "einfacher Dichte" speichern 128 pro Sektor, solche mit "doppelter Dichte" speichern 256 Bytes pro Sektor.

DISKETTE :
Kleine Platte. Ein Medium für Aufzeichnung und Wiedergabe ähnlich wie das Band, jedoch in Form einer flachen Scheibe, die in einer versiegelten Schutzhülle steckt. Die Zugriffszeit ist bei der Diskette erheblich kürzer als beim Band.

DOS :
Abkürzung für DISK - OPERATIONS-SYSTEM . Bezeichnung für die Software, die das Hantieren mit den Plattenlaufwerken ermöglicht bzw. erleichtert.

DOS.SYS :
Ein für das DOS reservierter Programmname.

EINGANGSPUNKT (ENTRY POINT):
Die Adresse, ander die Ausführung eines Programms in Maschinensprache bzw. einer Routine beginnen soll. Wird auch Transfer-Adresse genannt.

ERSATZFUNKTION (DEFOULT):
Eine Bedienung oder ein Zahlenwert, der automatisch in Funktion tritt oder in Funktion gebracht wird, solange der Computer nicht ausdrücklich andere Anweisungen erhält. Z.B. stellt sich die ATARI auf Graphikgang 0 ein, solange kein anderer Gang angegeben wird.

FILE (ABLAGE) :
Eine geordnete Sammlung zusammenhängender Daten. Die Ablage (FILE) stellt die grösste Gruppierung von Informationen dar, die mit Hilfe eines einzigen Namens adressiert werden kann. So ist z.B. ein BASIC-Programm als einzelner FILE auf einer Diskette "abgelegt" und kann über die Statements SAVE oder LOAD (u.a.) adressiert werden.

FILE - ENDE (END OF FILE):
Eine Markierung, die dem Computer angibt, dass das Ende einer bestimmten Programmablage erreicht ist.

FILE-NAME (Ablagenname):
Die alphanumerischen Zeichen, die zur Kennzeichnung eines FILES benutzt werden. Man kann hierfür bis zu 8 Ziffern und / oder Buchstaben verwenden. Das erste Zeichen muss ein Buchstabe sein.

FILENAMENS-ERWEITERUNG (EXTENDER):
0-3 zusätzliche Zeichen, die hinter dem (in diesem Fall vorgeschriebenen) Punkt am Ende des eigentlichen FILENAMENS stehen. So gelten die Buchstaben "BAS" in FILENAMEN PHONLIST und BAS als Erweiterung.

FILESPEZIFIKATION (FILESBACK):
Eine Folge von Zeichen, die ein bestimmtes Gerät und eine bestimmte Ablage (FILE) kennzeichnet. Ein solcher Namen darf NIEMALS für zwei bestehende FILES gleichzeitig verwandt werden. Geschieht dies doch und beide Ablage stehen auf der selben Diskette, so kann die zweite Ablage nicht adressiert werden. Dagegen wirken die DOS - Funktionen DELETE FILE (LöSCHE FILE) , RENAME FILE (BENENNE FILE NEU) und COPY FILE (KOPIERE FILE) auf beide Ablagen, da die Namen gleich sind.

FILE -ZEIGER (FILE POINTER) :
Ein Zeiger weist auf eine bestimmte Stelle innerhalb eines FILES hin. Jede Teilablage hat einen eigenen Zeiger.

FORMATIEREN :
Das Einteilen einer neuen oder völlig gelöschten Diskette in Spuren und Sektoren. Nach der Formatierung enthält eine Diskette 40 konzentrische Spuren mit je 18 Sektoren. Jeder Sektor kann bis zu 128 Bytes (bei einfacher Dichte) bzw. 256 Bytes (bei doppelter Dichte) aufnehmen.

HEXADEZIMAL (HEXA) :
Zahlensystem, das 16 alphanumerische Zeichen enthält: 0-9 und A, B, C, D, E, F.

HERKUNFTSADRESSE :
Das Gerät oder die Adresse, wo die an eine Zieladresse zu sendenden Daten enthalten sind. Vergl. Zieladresse.

INDEXIERTE ADRESSIERUNG : S. RANDEM EXESS

INPUT :
Ein BASIC-Befehl, der zum Anfordern von numerischen oder Stringdaten von einem bestimmten gerät dient.

INPUT (EINGABE) :
Das Übertragen von Daten von ausserhalb des Computers (z.B. von Diskette) in den Zentralspeicher. Das Gegenteil ist OUTPUT. Beide Worte werden oft zusammen benutzt , um Übertragungsoperationen im allgemeinen zu bezeichnen (Abkürzung I/O). Der Bezugspunkt ist immer der Computer: OUTPUT bedeutet also vom Computer weg, INPUT zum Computer hin.

IOCB (INPUT/OUTPUT-CONTROL BLOCK) :
Eine Zone im RAM, die für das adressieren eines Ein-oder Ausgabegerätes und das Verarbeiten von dorther kommender Daten bzw. für das adressieren und Übertragen von Daten zu einem OUTPUTGERÄT reserviert ist.

iocb :
Ein arytmetischer Ausdruck, der die Werte vom 1 - 7 annehmen kann. Diese Zahl wird für den Bezug auf ein Pereferiegerät oder eine Ablage verwandt.

KENNZEICHNEN :
Das Übersetzen von TExt-Förmigen BASIC-Quellencode und dessen Umwandlung in das interne Format, dAS DER BASIC-Übersetzer benutzt.

KILOBYTE oder K :
1024 Bytes . Eine Kapazität von 16 K RAM bedeutet also freie Speicherkapazität von 16 X 1024, also 16384 Bytes.

KORRIGIEREN (DEBUG) :
Hier das Auffinden und Beseitigen von Programmfehlern.

LAUFWERK - NUMMER :
Eine ganze Zahl von 1-8, die das jeweils zu benutzende Plattenlaufwerk benennt.

LAUFWERKSPEZIFIKATION (DRIVES BACK) :
Teil des Ablagenamens, der dem Computer angibt , welches Plattenlaufwerk angesprochen werden soll. Wird kein Laufwerk genannt, so spricht der Computer automatisch das Laufwerk 1 an.

MASCHINENSPRACHE :
Der Instruktionssatz für einen bestimmten Microprozessor-Chip, im Fall von ATARI, für den 6502.

NIEDRIGSTES BYTE :
Das Byte in der äussersten rechten Position innerhalb einer Zahl oder eines Wortes.

0-STRING :
Ein String, dass aus garkeinem Zeichen besteht, z.B. A$="" speichert das 0-STRING als A$.

OBJEKTCODE :
Maschinensprache, die von sogenannten "Quellencode", aus entwickelt ist, typischerweise aus der ASSEMBLERSPRACHE.

OCTAL :
Ein Zahlensystem, dass nur die Ziffern 0-7 benutzt. Adressen und Byte werden gelegentlich in octaler Form wiedergegeben.

OBERSTES BYTE :
Das Byte an der äussersten linken Position innerhalb einer Zahl oder eines Wortes (Gegenteil niedrigstes Byte).

OPEN :
Das Vorbereiten eines Files für den Zugriff durch Spezifizierung , ob eine Eingabe-oder eine Ausgabe-Operation erfolgen soll, und durch Angabe des Namens (FILESPEC).

PARAMETER:
Variablen in einem Befehl oder einer Funktion.

PERIPHERIE-GERÄT:
Ein Eingabe- oder Ausgabe-Gerät.

PLATTENVERZEICHNIS (DIRECTORY):
Eine Auflistung von Ablagen, die auf einer Platte enthalten sind, in Form der Listung von Namen und Größe der einzelnen Ablagen.

POKEY :
Ein normales I/O-Chip, das die Daten Übertragung über den seriellen Kanal steuert.

PUFFER :
Ein RAM-Bereich zur vorübergehenden Speicherung von Daten, die bald wieder verarbeitet, oder ein-, bzw. ausgegeben werden sollen usw..

QUELLENCODE:
Eine Serie von Instruktionen, die nicht in Maschinensprache geschrieben sind und daher zur Ausführung erst übersetzt werden müssen.

RANDOM ACCESS (Freier Zugriff):
Hier: die Methode, Daten von einer Diskette direkt aus dem Byte, bzw. dem Sektor abzulesen, wo diese gespeichert sind, ohne daß man die gesamte Plattenablage hintereinander lesen muß.

RECORD :
Ein Datenblock, der durch EOL-Zeichen (End of Line = 9B HEX) begrenzt ist.

SECTOR :
Ein Sektor ist der kleinste Datenblock, den man auf eine Diskette schreiben, oder von ihr lesen kann. Jeder Sektor mit einfacher Dichte kann 128 Bytes, jeder mit doppelter Dichte 256 Bytes aufnehmen.

SEQUENTIELLES ADRESSIEREN:
Die Methode, Byte nach Byte von der Diskette zu lesen, beginnend beim ersten Byte jedes Sektors.

STRING :
Eine Folge von Buchstaben, bzw. Zeichen, die in einer String-Variablen gespeichert sind. Der Name einer solchen Variablen muß auf $ enden.