

MS™-DOS

GENIE 16 C

Benutzerhandbuch

TCSA
COMPUTER GMBH

***Handbuch
zum***

GENIE 16 C

Zur Beachtung:

TCS Computer GmbH behält sich das Recht vor, Änderungen und Verbesserungen der in diesem Handbuch beschriebenen Hard- und Software zu jeder Zeit und ohne Ankündigung vorzunehmen.

Copyright © 1985 by TCS Computer GmbH

Alle Rechte vorbehalten, insbesondere auch diejenigen aus der spezifischen Gestaltung, Anordnung und Einteilung des angebotenen Stoffes. Der auszugsweise oder teilweise Nachdruck sowie fotomechanische Wiedergabe oder Übertragung auf Datenträger zur Weiterverarbeitung ist untersagt und wird als Verstoß gegen das Urheberrechtsgesetz und als Verstoß gegen das Gesetz gegen den unlauteren Wettbewerb gerichtlich verfolgt. Für etwaige technische Fehler, sowie für die Richtigkeit aller in diesem Buch gemachten Angaben, übernehmen der Herausgeber und Autor keine Haftung.

Vorwort

Dieses Handbuch wurde zur Einweisung und Weiterführung in die diskettenunterstützte Programmierarbeit auf dem Rechner GENIE 16 C unter dem Betriebssystem MS-DOS geschrieben.

In den Kapiteln 1 bis 4 wird Ihnen grundsätzliches über die Benutzung einer Diskette als Speichermedium, den Hardwareaufbau einer GENIE 16 C Rechneranlage, sowie den Aufbau und die Funktion der angeschlossenen Tastatur Schritt für Schritt nahegebracht.

Zur perfekten Beherrschung des Betriebssystems und Disk-BASICs ist es jedoch unumgänglich, auch die weiteren Kapitel dieses Handbuchs durchzuarbeiten. Sie haben die genaue Beschreibung aller DOS- und Disk-BASIC-Befehle zum Inhalt.

Stellen Sie sich zu Beginn Ihrer Programmiersuche ein Duplikat Ihrer Systemdiskette her und legen Sie das Original an einen sicheren Ort (siehe Kapitel 5). Arbeiten Sie nur mit dem Duplikat. Nur so vermeiden Sie die Zerstörung Ihrer Original MS-DOS Diskette durch eine mögliche fehlerhafte Programmierung.

Im Anhang finden Sie für Ihre Programmierarbeit wertvolle Tabellen, Listen und die Erklärungen der zusätzlichen Programme, die sich auf Ihrer MS-DOS-Diskette befinden.

Anhand des Umfangs dieses Handbuchs mögen Sie ersehen, daß wir uns Mühe gegeben haben, alle Zusammenhänge möglichst ausführlich zu beschreiben. Wenn Sie trotzdem etwas nicht auf Anhieb verstehen oder der Rechner eingegebene Kommandos mit Fehlermeldungen quittiert, dann lesen Sie sich bitte den entsprechenden Abschnitt nochmals in Ruhe durch und geben alles nochmal neu ein. In der Regel funktioniert dann plötzlich alles.

Sollten Sie innerhalb der vorliegenden Beschreibung auf Punkte stoßen, die Ihnen unverständlich oder fehlerhaft dokumentiert scheinen, so sind wir für Verbesserungsvorschläge oder Anregungen selbstverständlich jederzeit dankbar.

Viel Erfolg bei der Arbeit mit Ihrem GENIE und dem MS-DOS wünscht Ihnen

Helmut Scholz

Inhaltsverzeichnis

	Seite
Vorwort	5
Inhaltsverzeichnis	7
1 Einführung	11
1.1 BROM, DIOS, Disk-BASIC und MS-DOS	12
1.2 Vorteile des Floppybetriebs	15
1.3 Die Floppy-Disk	16
1.4 Behandlung und Handhabung einer Disk	19
2 Hardware	22
2.1 Die Hauptplatine	22
2.2 Die Rückansicht des GENIE 16 C	23
2.3 Die Tastatur	25
2.4 Zusätzliche Peripherie	26
2.5 Aufbau der Anlage	27
2.6 GENIE 16 C und die Diskettenlauf- werke	28
2.7 Einlegen und Entfernen einer Diskette	29
3 Inbetriebnahme	30
3.1 Vorbereitungen zur Inbetriebnahme	30
3.2 Einschalten des Systems	32
4 Die Tastatur des GENIE 16 C	35
4.1 Beschreibung der Tastatur	35
4.2 Tastenkombinationen	43
4.3 Das DOS und die Tastatur	44
4.4 Eingaben	46

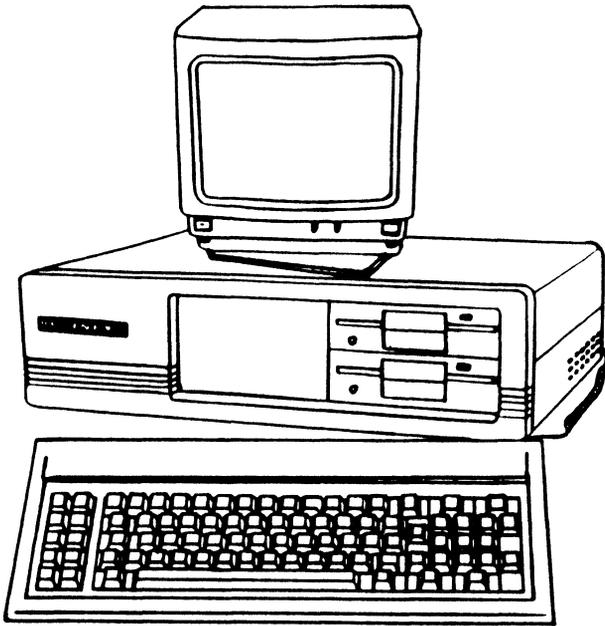
	Seite	
5	Das Diskettenbetriebssystem (MS-DOS)	47
5.1	Einführung ins MS-DOS	47
5.2	Das Booten des DOS	49
5.3	Eingabe des Datums	50
5.4	Eingabe der Zeit	52
5.5	Wechsel von Laufwerk A nach Laufwerk B	54
5.6	Formatieren einer Diskette	55
5.7	System- und Nicht-System-Disketten	57
5.8	Formatieren einer Systemdiskette	59
5.9	Kopieren von Disketten	60
5.10	Files	63
5.11	Wildcards	68
5.12	Anzeige des Inhaltsverzeichnisses	70
5.13	Einführung in die File-Directories	72
5.14	DOS-Befehle und Path-Beschreibungen	77
6	Einführung in die DOS-Befehle	83
6.1	Allgemeine Hinweise	83
6.2	Interne und externe Befehle	84
6.3	Batchverarbeitung	90
6.4	Input und Output	98
7	Die DOS-Befehle	102
8	Batchverarbeitungsbefehle	164
9	DOS-Editier- und Funktionstasten	173
10	Der Zeileneditor (EDLIN)	178
11	File-Vergleichsprogramm (FC)	214

	Seite
12 BASIC	223
12.1 Für den Anfänger in BASIC	223
12.2 GENIE 16 C BASIC	228
12.3 Laden von BASIC	229
12.4 BASIC-Prompt	230
12.5 Programme in BASIC	231
12.6 Typen und Namen von Variablen	232
12.7 Datenfelder	235
12.8 Arithmetische Funktionen	237
12.9 Der BASIC-Editor	240
12.10 Der freie Speicherplatz	244
12.11 Format von gespeicherten Programmen	245
12.12 Der Zeichensatz	247
12.13 Abfragen der Tastatur	248
12.14 Der Zufallszahlengenerator	251
12.15 'Softkeys' und Tastenfallen	252
12.16 Töne	256
12.17 Der Bildschirm des GENIE 16 C	260
12.18 Der Textschirm	263
12.19 Der unsichtbare Schirm	266
12.20 Der Graphik-Bildschirm	267
12.21 GENIE 16 C BASIC Kurzreferenzliste	279
12.22 BASIC-Fehlermeldungen	332
13 Erweiterungsmöglichkeiten	340
13.1 Drucker	341
13.2 Plotter	342
13.3 Monitor	343
13.4 Zusätzlicher Speicher	344
13.5 Hard Disk	345

	Seite
Anhang A: Diskettenfehler	346
Anhang B: Konfigurieren des Systems	348
Anhang C: ASCII-Codes des GENIE 16 C	351
Anhang D: Steckerbelegungen	352
Anhang E: Das GENIE 16 C BIOS	354
Anhang F: Zusätzliche Programme	364
ANSI.SYS	364
GERMAN.COM	371
LINK.COM	372
Anhang G: Definieren von Zeichen im Graphik-Modus	373
Anhang H: DEBUG	376
Anhang I: GENIE 16 C BASIC Fehlermel- dungen	402
Anhang J: Kurzeingabe von Schlüssel- wörter	404
Anhang K: GENIE 16 C BASIC Befehls- tabelle	405
Anhang L: Erweiterte Tastencodes	419

1. Einführung

Zu Beginn sollen einige grundlegende Begriffe erklärt werden, deren Verständnis für den Umgang mit dem GENIE 16 C hilfreich ist. Es folgt eine einführende Beschreibung der Hardware des GENIE 16 C, sowie Hinweise zur ersten Inbetriebnahme.



1.1 BROM, DIOS, MS-DOS und Disk-BASIC

* BROM

Das "Bootstrap-Read-Only-Memory" (BROM) ist die Software, die hardwaremäßig (in einem 8 kByte EPROM) in Ihr GENIE 16 C eingebaut ist. Diese Software übernimmt zwei wesentliche Funktionen. Zum einen werden beim Einschalten des GENIE 16 C eine Reihe von Testroutinen durchgeführt (Speichertests etc.), zum anderen wird durch das BROM das Betriebssystem MS-DOS von der Diskette geladen (gebootet).

Wie bereits erwähnt, hat BROM zwei Funktionen - Hardwaretest und Laden des Betriebssystems von Diskette. Dabei können die Testroutinen in 3 Abschnitte unterteilt werden. Zunächst wird der Hauptspeicher (RAM) getestet. Es folgt eine Überprüfung der wichtigsten integrierten Schaltkreise und der verschiedenen Ein/Ausgabeeinheiten (z.B. Tastatur und Diskettenstation).

Eine Auflistung der dabei möglichen Fehlermeldungen finden Sie im Anhang.

Nach fehlerfreiem Durchlauf der Testroutinen lädt das BROM das MS-DOS von der Disk in Laufwerk A. Sollte Ihnen die Handhabung von Disketten noch nicht geläufig sein, sei auf die folgenden Abschnitte dieser Einführung verwiesen.

Beim Laden des MS-DOS sucht das BROM nach einer Datei namen DIOS.COM, deren Funktion im folgenden Abschnitt beschrieben ist. Hat BROM die DIOS.COM-Datei gefunden, wird diese in den RAM geladen und die weitere Steuerung des Gerätes an DIOS übergeben.

* DIOS

DIOS (Disk Input Output System) handhabt die Disketten-Ein/Ausgabe, sowie weitere grundlegende Systemfunktionen. DIOS befindet sich auf der mitgelieferten MS-DOS-Diskette unter dem Namen DIOS.COM und wird vom BROM geladen.

DIOS benötigt 8 kByte RAM, wobei jeweils die obersten 8 kByte des verfügbaren RAM-Bereichs belegt werden.

Wichtiger Hinweis:

Um eine Diskette booten zu können, muß DIOS entweder auf ihr vorhanden sein oder zuvor einmal von Ihrer MS-DOS-Diskette in den RAM geladen worden sein. Mit anderen Worten, es ist nicht möglich, eine Diskette, auf der sich die Datei DIOS.COM nicht befindet, direkt nach dem Einschalten zu laden. Vielmehr muß vorab immer ein MS-DOS mit DIOS.COM gebooten worden sein.

* MS-DOS

Es gibt bereits eine große Anzahl von verschiedenen Betriebssystemen, die den Floppy-Disk Betrieb mit Personal Computern ermöglichen. Sie unterscheiden sich zwar in vielen Einzelheiten, haben aber auf der anderen Seite auch viele Gemeinsamkeiten.

MS-DOS bietet dem Programmierer eine Vielzahl von Programmiermöglichkeiten. Ein genaues Studium des Handbuchs und einige Zeit praktischer Erfahrung sind jedoch unumgänglich, um das System perfekt zu beherrschen.

* Disk-BASIC

Auf der mitgelieferten MS-DOS-Diskette befindet sich auch das sogenannte Disk-BASIC. Es handelt sich dabei um eine leistungsstarke Version der Programmiersprache BASIC. In BASIC können Sie selbst Programme erstellen oder fertige Programme benutzen.

1.2 Vorteile des Floppy-Betriebs

Jeder Computer braucht einen externen Massenspeicher, um Programme und Daten dauerhaft und stromausfallsicher abzuspeichern. Beim GENIE 16 C sind hierfür 2 Diskettenlaufwerke vorgesehen, mit denen eine schnelle, bequeme und sicherere Daten- und Programmspeicherung möglich ist.

Sie brauchen sich beim Floppy-Disk-Betrieb nicht darum zu kümmern, auf welcher Stelle der Diskette die Speicherung erfolgt. Jede abzulegende Information erhält vor der Speicherung vom Programmierer einen Namen und kann dann durch Aufruf dieses Namens wieder von der Floppy in den Arbeitsspeicher des Computers geladen werden. Man hat also sofortigen, wahlfreien Zugriff auf alle gespeicherten Informationen.

1.3 Die Floppy-Disk

Eine Floppy-Disk - auch Diskette oder nur Floppy genannt - ist eine dünne flexible Kunststoffscheibe, die beidseitig mit einer magnetisierbaren Schicht versehen ist. Auf dieser Schicht lassen sich, ähnlich wie bei einem Tonband, mit Hilfe eines beweglichen Magnetkopfes Aufzeichnungen machen, die dann später wieder gelesen oder gelöscht werden können.

Typisch für alle Disketten ist, daß sie in einer fest verklebten Hülle stecken und auch mit dieser Hülle in das Laufwerk geschoben werden. In der Mitte der Hülle befindet sich ein größeres Loch, in das die Antriebseinrichtung eingreift. Ein länglicher Schlitz ermöglicht den Kontakt des Magnetkopfes mit der Diskettenoberfläche. Das neben der Antriebsöffnung liegende, sogenannte Indexloch dient dem Floppy-Controller zu Synchronisationszwecken.

Mittels einer Auskerbung am Rand der Disketten erkennt das Laufwerk, ob Schreib- und Löschezugriffe auf die Disk erlaubt sind oder nicht. Wird dieser Schlitz mit einem undurchsichtigen Klebestreifen (Lichtschranke!) verschlossen, so ist die Diskette "schreibgeschützt" und ihr Inhalt kann auch selbst versehentlich nicht mehr geändert werden. Das Auslesen von Informationen ist immer möglich.

Beim Betriebssystem MS-DOS werden nur 5 1/4" Disketten, auch Mini-Floppys genannt, verwendet.

Grundsätzlich erfolgt die magnetische Aufzeichnung bei einer Diskette nicht wie bei einer Schallplatte in einer langen Spirale, sondern in konzentrischen Ringen, die Spuren genannt werden. Jede Spur wiederum ist in einzelne Sektoren unterteilt.

Damit der Computer die verschiedenen Spuren und Sektoren einer Diskette erkennen kann, muß diese "formatiert" werden. Man versteht darunter das Einschreiben von bestimmten Informationen, aus denen der Rechner die gerade unter dem Magnetkopf vorbeilaufende Spur, sowie den Anfang bzw. die Nummer eines Sektors erkennen kann.

Unter MS-DOS werden ausschließlich "softsektorierte" Disketten verwendet, die nur ein Indexloch zur Erkennung des ersten Sektors auf einer Spur besitzen.

Der zu verwendende Diskettentyp hängt vom Laufwerk ab.

Da im GENIE 16 C serienmäßig doppelseitige 40-Spur-Laufwerke eingesetzt werden und der Floppycontroller die Datenaufzeichnung in doppelter Schreibdichte erfolgt, ist es ratsam, Disketten höherer Qualität zu verwenden.

In diesem Zusammenhang werden Disketten mit folgenden Spezifikationen angeboten:

einfache / doppelte Schreibdichte

ein- / doppelseitige Aufzeichnung

40 / 80 Spuren

das heißt, der Hersteller garantiert für deren Einhaltung.

Auf Grund der hohen Qualitätsanforderungen der Hersteller an das Material der Disketten ist es aber möglich, daß z.B. auch eine nur für einseitige Aufzeichnung zertifizierte Diskette bei doppelseitiger Verwendung fehlerfrei arbeitet.

Kennbuchstaben für Disketten

SD = SINGLE DENSITY = einfache Schreibdichte

DD = DOUBLE DENSITY = doppelte Schreibdichte

SS = SINGLE SIDED = einseitige Aufzeichnung

DS = DOUBLE SIDED = doppelseitige Aufzeichnung

Benutzen Sie für die diskettenunterstützte Programmierung mit Ihrem GENIE 16 C nur Disketten mit dem Zertifikat 40 Spur DS/DD.

1.4 Behandlung und Handhabung einer Disk

Obwohl Floppys verhältnismäßig unempfindlich sind, sollten Sie doch einige Regeln beim Umgang mit ihnen beachten, um keine Pannen beim Betrieb zu erleben. Sie müssen wissen, daß die Aufzeichnungen sehr gedrängt erfolgen. Auf einer Spurlänge von ca. 10 mm werden mehrere Tausend Einzelinformationen (Bits) gespeichert. Wenn aber auch nur ein Bit wegen Verschmutzung oder Beschädigung der Disk-Oberfläche falsch gelesen wird, kann ein ganzes Programm unbrauchbar werden.

W I C H T I G:
=====

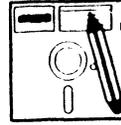
Die Floppy muß immer mit der Schreibe-kerbe nach oben und den länglichen Lese-schlitz nach hinten in die Öffnung des Laufwerks eingeschoben werden. Am besten wird sie dort angefaßt, wo sich das Etikett auf der Hülle befindet. Nach dem Einschieben der Diskette muß die Klappe am Laufwerk wieder geschlossen werden.

Folgende Hinweise sollten Sie unbedingt befolgen:

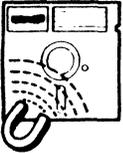
- * Niemals die Diskette am Leseschlitz berühren.
- * Diskette nach Gebrauch sofort mit dem Etikett nach oben in die Schutzhülle einschieben.
- * Diskette niemals falten, biegen oder knicken.
- * Niemals mit Kugelschreiber oder hartem Bleistift auf das Etikett schreiben, da Gefahr des Durchdrückens besteht.
- * Diskette niemals in der Nähe magnetischer Felder z.B. auf einem Netzgerät, Monitor, Telefonapparat, Lautsprecher u. ä. lagern, da dadurch Informationen auf ihr gelöscht werden können.
- * Diskette niemals einer Temperatur über ca. 50 Grad Celsius aussetzen. Vorsicht bei direkter Sonnenstrahlung, (z.B. im Auto)! Die schwarze Hülle kann sich dabei sehr stark aufheizen und verformen.



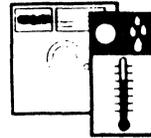
Magnetfläche
nicht berühren!



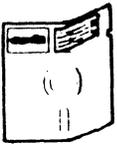
Nicht hart
beschriften!



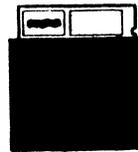
Kein
Magnetismus!



Nicht feucht,
warm oder
kalt lagern!



Nicht falten
knicken oder biegen!



Nach Gebrauch
zurück in die
Schutzhülle!

2. Hardware

2.1 Die Hauptplatine

Der GENIE 16 C besteht aus einer Zentraleinheit mit Tastatur. Erweiterungen sind über Steckkarten möglich, die in die auf der Platine befindlichen freien Slots gesteckt werden können.

Auf der Hauptplatine befindet sich der 8088 Mikroprozessor, das RAM (der Arbeitsspeicher mit maximal 640 kByte Inhalt), das ROM (der Festspeicher mit dem 8 kByte BROM zum Inhalt) und die Elektronik, die es dem System ermöglicht, mit Ihnen Verbindung aufzunehmen.

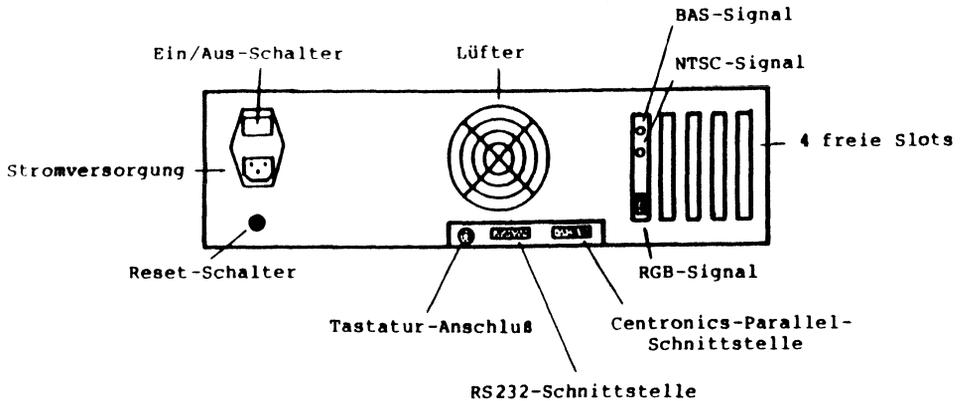
Der Rechner bezieht seine Informationen über die Tastatur. Weiterhin sind Anschlüsse für Drucker (Centronics parallel) und eine RS232-Schnittstelle) an der Rückseite des Geräts vorhanden.

Intern finden Sie auf der CRTC-Karte (Video-Karte einen Light-Pen-Anschluß.

Datenspeicherung ist Hilfe der zwei eingebauten doppelseitigen 40 Spur 5 1/4" Diskettenlaufwerken über die ebenfalls eingebaute Floppycontroller-Karte möglich.

Die Aufzeichnung der Daten auf Diskette erfolgt in doppelter Schreibdichte.

2.2 Die Rückansicht des GENIE 16 C



Die Anschlüsse des GENIE 16 C:

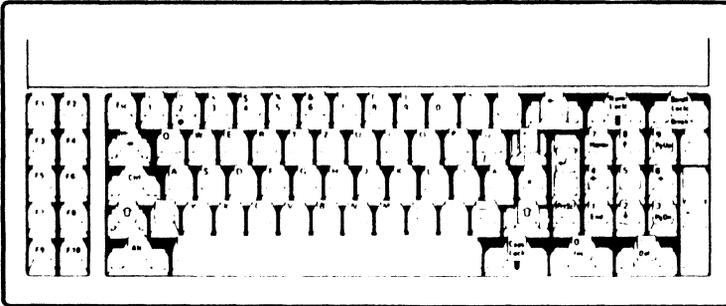
- Stromversorgung** - Kaltgerätesteckeranschluß zur Spannungsversorgung (220 V)
- Ein/Aus Schalter** - Netzschalter zum Ein- und Ausschalten des GENIE 16 C
- RESET-Schalter** - führt einen "Kaltstart" durch, mit einer Wirkung wie bei Einschalten des Gerätes (DIOS wird neu geladen)
- Tastatur-Anschluß** - DIN-Buchse zum Anschluß der Tastatur

- RS232-Schnittstelle - DB25-Buchse zum Anschluß von seriellen Druckern, Akustikkopplern u. ä.
- Centronics-Parallel-Schnittstelle - dient zum Anschluß eines Druckers mit Standard-Centronics-Parallel-Schnittstelle. (DB 25)
- NTSC-Signal - dient zum Anschluß eines Farbmonitors mit NTSC-Eingang.
- BAS-Signal - dient zum Anschluß eines monochromen Monitors mit BAS-Eingang (S/W)
- RGB-Signal - dient zum Anschluß eines Farbmonitors mit RGB-Eingang.
- Freie Slots - 4 intere IBM-kompatible frei Slots

2.3 Die Tastatur

Die Tastatur dient dazu, dem Computer Befehle und Informationen zu geben.

Prinzipiell gleicht die Tastatur des GENIE 16 C einer Schreibmaschinentastatur. Genau wie bei einer Schreibmaschine sind manche Tasten mit mehreren Funktionen belegt, diese können aufgerufen werden, indem man gleichzeitig eine andere, besondere Taste drückt. Diese Funktionen werden wir später besprechen.



2.4 Zusätzliche Peripherie

Zusätzlich zum Computer benötigen Sie einen Bildschirm. Zu diesem Zweck können verschiedene Typen von Datensichtgeräten angeschlossen werden.

Es lassen sich wahlweise Schwarzweiß- oder Farbgeräte anschließen, wobei für Farbausgabe je ein Anschluß für RGB-Monitore und Farbmonitore mit F-BAS-Eingangssignal zur Verfügung steht.

Jeder Drucker mit einer Standard Centronics Schnittstelle kann mit dem GENIE 16 C verbunden werden. Beachten Sie bitte beim Anschluß die Angaben im Druckerhandbuch.

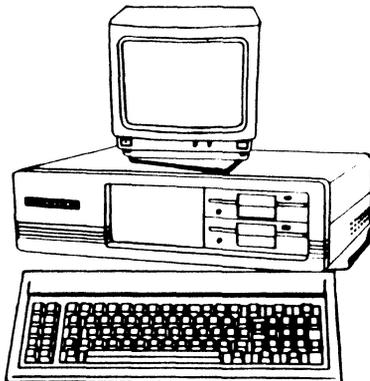
Über die eingebaute RS232-Schnittstelle haben Sie die Möglichkeit Kontakt zu anderen Rechnern aufzunehmen. Zu diesem Zweck benötigen Sie entweder ein akustisch gekoppeltes Modem, mit dem Sie das GENIE 16 C als Terminal an einem entfernt aufgestellten Großrechner oder an einem öffentlichen Datendienst betreiben können. Oder Sie haben einen zweiten Rechner zur Verfügung, den Sie direkt mit dem GENIE 16 C über die RS232-Schnittstelle zum Datenaustausch verbinden können. Natürlich lassen sich über die RS232-Schnittstelle z.B. auch Drucker ansteuern.

2.5 Aufbau der Anlage

Stellen Sie die Anlage vor sich auf den Tisch, so daß Sie einen guten Zugang zur Vorder- und Rückseite des Gerätes haben, um alle notwendigen Anschlüsse herzustellen.

Die Tastatur wird mit dem GENIE 16 C über ein Spiralkabel verbunden. Stecken Sie den Stecker in die DIN-Buchse an der Rückseite des Gerätes. Unter der Tastatur finden Sie einen Bügel, der nach hochklappen ein leichtes Kippen der Tastatur bewirkt. Während der Arbeit kann die Tastatur frei bewegt werden. Sie können sie auf Ihre Knie legen, auf den Tisch oder an andere Plätze stellen, an denen Sie sie benötigen.

Schließen Sie ein Datensichtgerät an den entsprechenden Ausgang der Zentraleinheit.



2.6 GENIE 16 C und die Diskettenlaufwerke

Entfernen Sie den Transportschutz aus den Diskettenlaufwerken. Jedes der Diskettenlaufwerke hat am Einschubschacht einen Verschuß. Je nach Laufwerkstyp handelt es sich dabei um eine Klappe oder einen Hebel.

Befindet sich die Klappe oder der Hebel in hochgestellter Position, so können Disketten eingelegt bzw. entfernt werden, ansonsten ist das Laufwerk betriebsbereit.

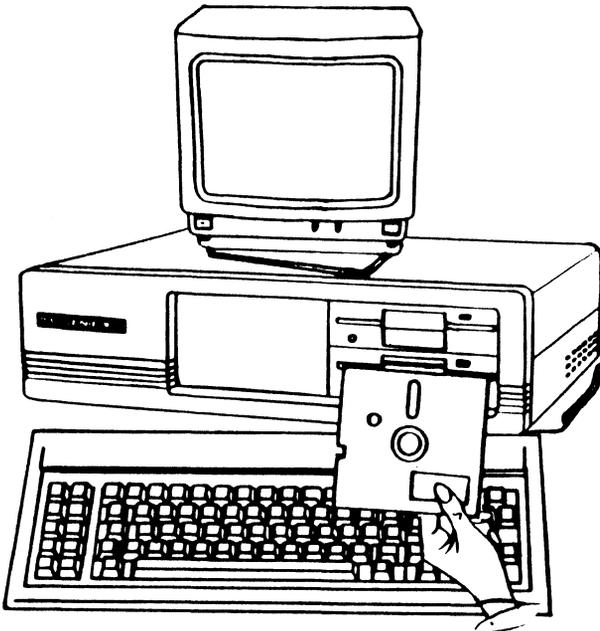
Bei geschlossenem Laufwerk senkt sich der Schreib/Lesekopf auf das Diskettenmaterial. An jedem Laufwerk befindet sich eine kleine Leuchtdiode. Leuchtet diese auf, ist das Laufwerk aktiv, und es dürfen auf keinen Fall Disketten eingelegt oder entfernt werden.

Wollen Sie die Disketten über Nacht in den Laufwerken lassen, empfiehlt es sich, die Laufwerksklappe zu öffnen.

Schalten Sie niemals das GENIE 16 C ein, wenn sich Disketten im Laufwerk befinden. Der Disketteninhalt könnte durch einen magnetischen Impuls zerstört werden.

2.7 Einlegen und Entfernen einer Diskette

Nehmen Sie eine leere Diskette und üben Sie damit das Einlegen von Disketten. Stecken Sie die Diskette waagrecht mit dem Aufkleber nach oben und zu Ihnen hin in den Einschubschacht. Schieben Sie die Diskette vorsichtig bis zum Anschlag in das Laufwerk. Die Diskette sollte vollständig in das Laufwerk passen und am Ende einrasten, wenden Sie bitte keine Gewalt an. Anschließend schließen Sie das Laufwerk. Dieses sollte vollkommen mühelos geschehen. Das Herausnehmen der Diskette erfolgt in umgekehrter Reihenfolge, wobei die Diskette nach Öffnung der Klappe bereits etwas aus dem Laufwerkschacht herausgeschoben wird, damit Sie sie besser entnehmen können.



3. Inbetriebnahme

3.1 Vorbereitungen zur Inbetriebnahme

An dieser Stelle muß auf zwei wichtige Regeln hingewiesen werden.

Es dauert eine gewisse Zeit, bevor alle Baugruppen eines elektronischen Gerätes nach Einschaltung mit der Betriebsspannung versorgt werden und es ordnungsgemäß arbeiten kann. Bei einem Radio ist dieser Vorgang z.B. für den meist im Lautsprecher zu hörenden "Einschaltknacks" verantwortlich. Was beim Radio der Lautsprecher, ist beim Laufwerk der Magnetkopf:

Es kann also passieren, daß ein unerwünschter Impuls auf eine im Laufwerk befindliche Diskette aufgezeichnet wird, der die dort gespeicherten Daten zerstört. Je nachdem, an welcher Stelle dies passiert, kann ein Sektor für die Datenaufzeichnung unbrauchbar werden, was nicht unbedingt sofort auffallen muß, ein Programm zerstört werden, womit es verloren ist oder schlimmstenfalls das Inhaltsverzeichnis der Disk in Mitleidenschaft gezogen werden, wodurch diese teilweise oder vollständig unlesbar wird, da der Rechner sich auf ihr nicht mehr zurechtfindet.

Regel 1: Floppys sollten nur ins Laufwerk gesteckt werden, wenn dieses eingeschaltet ist, bzw. aus ihm entnommen werden, bevor es ausgeschaltet wird.

Regel 2: Der Zugriff des Rechners auf die Diskette darf in keinem Fall unterbrochen werden, weder durch Ausschalten, noch durch Herausnehmen der Diskette aus dem Laufwerk.

Dies würde mit großer Wahrscheinlichkeit zu einem Datenverlust führen, da der Computer nicht mehr in der Lage ist, eine evtl. laufende Aufzeichnung ordnungsgemäß zu beenden, bzw. das Vorhandensein neuer Daten im Inhaltsverzeichnis zu vermerken. Ein Zugriff des Rechners auf die Diskette ist am Aufleuchten der Leuchtdiode am Laufwerk zu erkennen.

Bevor Sie nun das Gerät einschalten, sollten Sie nochmals folgende Punkte überprüfen:

1. Ist das Netzkabel korrekt angeschlossen?
2. Ist ein Monitor an die entsprechende Buchse des GENIE 16 C angeschlossen?
3. Befinden sich keine Disketten in den Laufwerken?
4. Ist die Tastatur ordnungsgemäß angeschlossen?

3.2 Einschalten des Systems

Schalten Sie den Monitor und anschließend den Rechner ein, indem Sie den Schalter an der Rückseite des GENIE 16 C betätigen.

Das GENIE 16 C wird jetzt nach einem akustischen Signal einen mehrere Sekunden dauernden Selbsttest durchführen und Sie fragen, ob ein RAM-Test durchgeführt werden soll. Wenn an dieser Stelle innerhalb einiger Sekunden keine Antwort (Y/N) gegeben wird, erfolgt lediglich eine Kontrolle der Speichergröße.

Anschließend beginnt Laufwerk A (unteres Laufwerk) zu surren, wobei die rote Leuchtdiode darauf aufleuchtet. DIOS.COM wird von der im Laufwerk eingelegten MS-DOS-Diskette geladen. Nach einer kurzen Pause ertönt ein erneutes akustisches Signal und MS-DOS wird gebootet.

Sollte wider erwarten nichts auf dem Bildschirm erscheinen, drehen Sie am Helligkeitsregler des Datensichtgerätes, bis Sie ein deutliches Bild haben.

Sollte auf Ihrem Bildschirm eine Fehlermeldung erscheinen, so ist Ihr Rechner nicht in Ordnung.

(Weitere Informationen über auftretende Fehler finden Sie im Anhang).

Am Ende des Bootvorganges sollte auf dem Bildschirm folgende Meldung zu sehen sein.

DIOS VERSION 2.21

Microsoft MS-DOS version 2.11
Copyright 1981,82,83 Microsoft Corp.

Command v. 2.11
Current date is Tue 1-01-1980
Enter new date:

Sie können nun dem Rechner das aktuelle Datum mitteilen oder nur die <ENTER>-Taste betätigen, wobei das ausgegebene Datum übernommen wird. Es folgt nun die Ausgabe

Current time is 0:08:30:16
Enter new time:

Sie können hier wie bei der Frage nach dem aktuellen Datum verfahren.

Die Ausgabe von

A>

signalisiert Ihnen, daß Sie sich in der DOS-Befehlsebene befinden.

Um ins BASIC zu gelangen, geben Sie an dieser Stelle

BASIC

ein.

Die Ausgabe eines

OK

mit einem darunter blinkenden Strich zeigt Ihnen an, daß Sie nun in BASIC arbeiten können.

Der blinkenden Strich "-" auf dem Bildschirm unter dem "OK" repräsentiert den sogenannten CURSOR. Er zeigt Ihnen an, wo der nächste Buchstabe, den Sie eintippen werden, auf dem Bildschirm erscheint.

Sobald Sie einen Buchstaben auf der Tastatur getippt haben, bewegt sich der Cursor um eine Stelle nach rechts. Auf der alten Cursorposition erscheint das soeben getippte Zeichen. Auf der Zahlentastatur finden Sie vier Tasten mit Pfeilen in unterschiedlichen Richtungen.

Bei Betätigung dieser 'Pfeiltasten' bewegt sich der Cursor in die angezeigte Richtung. Wird eine Taste länger festgehalten, wird die entsprechende Funktion in schneller Folge wiederholt.

4. Die Tastatur des GENIE 16 C

=====

4.1 Beschreibung der Tastatur

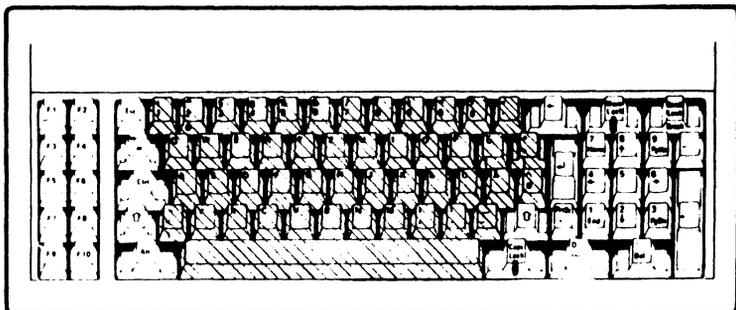
Man kommuniziert mit dem Genie 16 C über die Tastatur. Auf den ersten Blick erscheint diese sehr kompliziert - sie hat immerhin 83 verschiedene Tasten. Trotzdem werden Sie sich nach dieser Anleitung schnell auf ihr zurechtfinden.

Werden im nachfolgenden Text besondere Tasten erwähnt, so sind diese mit '<' und '>' geklammert. Es handelt sich dann um eine einzige Taste (z.B. <Ctrl>).

Die folgende Tabelle erklärt die Abkürzungen der einzelnen Tastenbezeichnungen:

<Ctrl>	==	Control
<Esc>	==	Escape
<Caps Lock>	==	Capitals lock
<Alt>	==	Alternate
<Num Lock>	==	Number Lock
	==	Delete
<Pg Up>	==	Page Up
<Pg Dn>	==	Page Down
<Ins>	==	Insert
<Prt Sc>	==	Print Screen
< ↑ >	==	Hochpfeil (Taste 8 auf der Zehnertastatur)
< ↓ >	==	Pfeil abwärts (Taste 2 auf der Zehnertastatur)
< ← >	==	Linkspfeil (Taste 4 auf der Zehnertastatur)
< → >	==	Rechtspfeil (Taste 6 auf der Zehnertastatur)

Im folgenden wird die Bedeutung der einzelnen Tasten im BASIC erklärt. Einige Tasten haben im DOS eine andere Funktion. Diese wird im Kapitel über das DOS gesondert erklärt.



Die im obigen Bild schraffierten Tasten entsprechen denen auf einer Schreibmaschine.

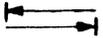
Auf der obersten Tastenreihe befinden sich die Ziffern 0-9 und eine Reihe von Zusatzzeichen, die über die <Shift>-Taste zu erreichen sind. <↑>

Die breite Taste ganz unten ist die Leertaste (Spacebar)

Wenn Sie noch nie eine Computertastatur bedient haben, sind Ihnen viele der Tasten sicherlich noch unbekannt. Diese Tasten dienen dazu, Ihnen die Arbeit mit dem Computer zu erleichtern, Befehle zu geben, Programme zu schreiben und zu verbessern.

<Esc> (Escape)

Diese Taste entfernt die Zeile, auf der der Cursor sich befindet, so daß sie neu eingegeben werden kann.

<

Dies ist der Tabulator. Er arbeitet genauso, wie der Tabulator auf einer Schreibmaschine. Bei den meisten Programmen ist der Tabulator auf acht Zeichen eingestellt.

<Caps Lock>

Diese Taste schaltet den Großschriftmodus um. Nach dem Einschalten des GENIE 16 erscheinen bei Betätigen einer Buchstabentaste Kleinbuchstaben, mit <Shift> erscheinen Großbuchstaben. <Caps Lock> schaltet dies um.

<Ctrl>

Wird diese Taste mit einer anderen zusammen gedrückt, ergibt sich eine Funktion. Drücken Sie beispielsweise <Ctrl> und <Home> zusammen, wird der Bildschirm gelöscht und der Cursor in die obere, linke Ecke des Bildschirms positioniert.

<  >

Dies ist die Shift-Taste. Das Drücken von <Shift> mit einer Buchstabentaste zusammen ergibt einen Großbuchstaben (siehe <Caps Lock>).

Auf der Tastatur befinden sich zwei Shift-Tasten (rechts und links).

<Alt>

Diese Taste erzeugt in Verbindung mit den Buchstabentasten die wichtigsten BASIC-Befehle, so daß Sie diese nicht immer neu eingeben müssen.

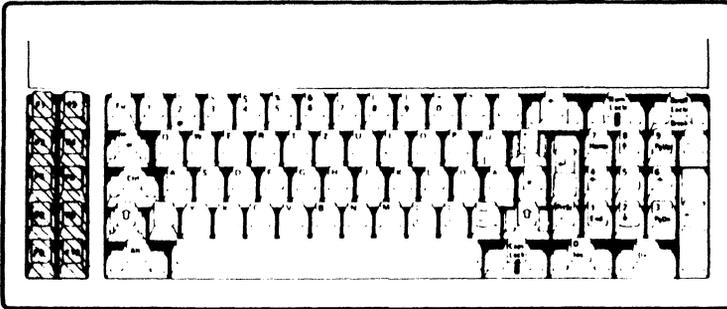
<  >

Bewegt den Cursor um eine Stelle nach links und löscht bei jeder Betätigung einen Buchstaben.

<  >

Es handelt sich um die sogenannte <ENTER>-Taste, mit der Sie Ihre Eingaben zur Ausführung an den Rechner abschicken.

Die Funktionstasten:



Diese Tasten erfüllen verschiedene Funktionen, je nach dem, welches Programm Sie benutzen. So erlauben Sie in BASIC die Eingabe von häufig gebrauchten Befehlswörtern.

Im 40 Zeichenformat Sie sehen am Ende des Schirms folgende Zeile:

```
1LIST 2RUN 3LOAD" 4SAVE" 5EDIT
```

Im 80 Zeichenformat erscheint zusätzlich

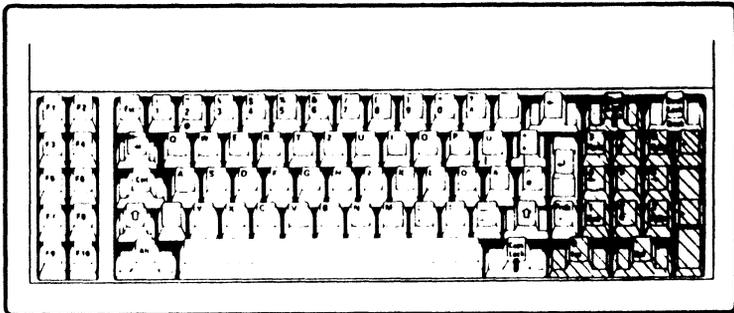
```
6CHDIR 7RMDIR 8FILES 9SHELL 0SYSTEM
```

Die Nummer vor jedem Befehl entspricht der Zahl auf den Funktionstasten.

5EDIT heißt also, wenn Sie F5 drücken, wird EDIT auf dem Schirm ausgegeben.

Den Funktionstasten können mit dem KEY-Befehl andere Funktionen zugeordnet werden.

<Num Lock>



Die Taste rechts oben heißt <Num Lock>, was für 'Number Lock' steht. Einmaliges Drücken aktiviert die Zehnertastatur, nochmaliges Drücken deaktiviert diese wieder.

Wenn <Num Lock> aktiv ist, was Sie auch am Aufleuchten der in der Taste eingebauten Leuchtdiode erkennen können, kommt den Tasten des Zehnerblocks folgende Bedeutung zu:

1. Die Zehnertastatur (0-9) ist aktiv
2. ist der Dezimalpunkt
3. <-> ist die Minustaste
4. <+> ist die Plustaste

Wenn <Num Lock> deaktiviert ist kommt den Tasten des Zehnerblocks folgende Bedeutung zu:

1. <Home> bewegt den Cursor in die obere, linke Ecke.
2. < ↑ > bewegt den Cursor eine Zeile hoch.
3. <Pg Up> Wird ein Textverarbeitungssystem benutzt, wird mehrere Zeilen zurückgeblättert.
4. < ← > Bewegt den Cursor eine Stelle nach links.
5. < → > Bewegt den Cursor eine Stelle nach rechts.
6. <End> Bewegt den Cursor ans Ende der momentanen Zeile.
7. < ↓ > Bewegt den Cursor eine Zeile nach unten.
8. <Pg Dn> Bewegt den Cursor mehrere Zeilen nach unten (nur bei Textverarbeitung).
9. <Ins> Wird <Ins> einmal gedrückt, können Buchstaben eingefügt werden (INSERT-Modus). Nochmaliges Drücken bringt den Rechner wieder in den normalen Modus.
10. Löscht den Buchstaben, auf dem sich der Cursor befindet und rückt den Rest der Zeile nach.

Die Bedeutung dieser Tasten kann sich je nach Art des verwendeten Softwaresystems verändern. Bitte schlagen Sie dann im zugehörigen Handbuch nach.

Es gibt noch zwei weitere Tasten auf der Zehnertastatur:

<Prt Sc> (Print Screen = Drucke den Bildschirm)

Diese Taste erzeugt ein * auf dem Schirm oder gibt, wenn Sie sich im BASIC befinden und gleichzeitig <Shift> gedrückt wird, den Inhalt des Bildschirms auf einem angeschlossenen Drucker aus.

<Scroll Lock>/<Break>

Diese Taste wird nur mit <Ctrl> zusammen benutzt.

4.2 Tastenkombinationen

Manchmal müssen Sie mehrere Tasten zugleich drücken, wie z.B. die Shift-Taste und eine Buchstabentaste, um Großbuchstaben zu bekommen.

Die gebräuchlichsten Tastenkombinationen sind:

1. <Ctrl> <Scroll Lock>/<Break>

Bricht das Programm ab (BREAK-Funktion).

2. <Ctrl> <Num Lock>

Stoppt das Programm. Die Ausführung mit kann jeder beliebigen Taste fortgeführt werden.

3. <Ctrl> < ← > bzw. <Ctrl> < → >

Setzt den Cursor auf den Anfang des nächsten Wortes in der angegebenen Richtung.

4. <Ctrl> <Home>

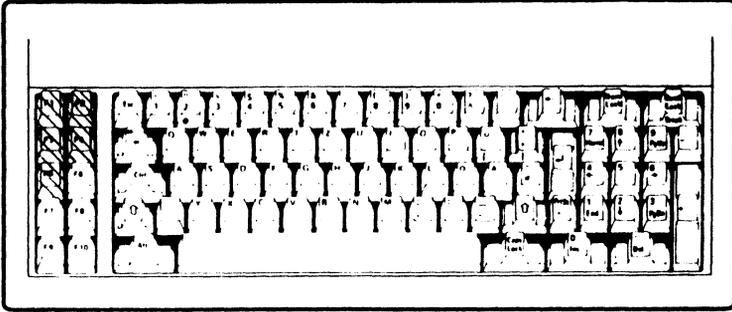
Löscht den Bildschirm und bewegt den Cursor in die obere, linke Ecke des Bildschirms.

5. <Ctrl> <Alt>

Diese Kombination erzeugt einen RESET, d.h. der Rechner wird, wie nach dem Einschalten, neu initialisiert.

Achtung: Der Speicherinhalt geht verloren!!

4.3 Das DOS und die Tastatur



In der DOS-Befehlsebene haben die Funktionstasten eine andere Bedeutung als im BASIC. Ausführlich werden ihre Funktionen im nächsten Kapitel erläutert.

<F1> Zeigt erneut, Buchstabe für Buchstabe, die letzte eingegebene Kommandozeile an.

<F2> Die Eingabe von F2 und einem Zeichen bewirkt die Ausgabe der zuvor eingegebenen Zeile bis zum ersten Auftreten des angegebenen Zeichens.

Beispiel: Letzte Kommandozeile sei
'TYPE TEST.TXT' gewesen.
<F2> & '.' erzeugt dann die
Ausgabe von 'TYPE TEST'

<F3> Die letzte eingegebene Kommandozeile erscheint.

<F4> Eingabe von F4 und einem Zeichen bewirkt das Überspringen aller Zeichen im Eingabepuffer bis zum Erscheinen des eingegebenen Zeichens. F4 ist also das Gegenteil von F2.

Beispiel: Es steht wieder
'TYPE TEST.TXT' im Eingabepuffer. <F4> & '.' überspringt alle Zeichen bis zum Punkt, so daß <F3> danach nur noch '.TXT' ausgibt.

Die Ctrl-Taste hat auch im DOS eine besondere Funktion.

1. <Ctrl> <Num Lock>
Stoppt den Rechner. Das Drücken einer beliebigen Taste läßt das System fortfahren.
2. <Ctrl> <Shift> <Prt Sc>
Alles, was auf dem Bildschirm ausgegeben wird, erscheint auch auf dem Drucker. Zum Abschalten muß dieselbe Tastenkombination erneut gedrückt werden.
3. <Ctrl> <Scroll Lock/Break>
Stoppt die Ausführung eines laufenden Programms.

4.4 Eingaben

Eingaben auf der GENIE 16 C Tastatur sind sehr einfach, trotzdem lassen sich Tippfehler nicht immer vermeiden. Um einen Fehler auszubessern, muß die <←> Taste gedrückt werden. Bei jeder Betätigung wird das Zeichen links vom Cursor gelöscht und der Cursor kommt auf dessen Position. Nun kann das richtige Zeichen eingegeben werden.

Wenn die Kommandozeile vollständig eingegeben wurde, muß zur Bestätigung die ENTER-Taste gedrückt werden. Der Computer wird den Befehl nicht ausführen, solange <ENTER> nicht gedrückt wurde.

5. Das Diskettenbetriebssystem (MS-DOS)

5.1 Einführung ins MS-DOS

Für die Arbeit mit Floppy-Laufwerken ist das Betriebsprogramm, das auch DOS (Disk Operating System) genannt wird, notwendig. Dieses Programm, oder zumindest Teile davon, müssen im RAM des Rechners stehen.

Das DOS ist Ihr 'stiller Partner', wenn Sie mit dem GENIE 16 C arbeiten. Es ist ein Programm, das zwischen der Hardware und Ihnen bzw. einem anderen Programm Verbindungen schafft.

Durch Benutzung von MS-DOS ist eine die Voraussetzung der Kompatibilität mit dem IBM PC gewährleistet.

Dadurch erhält der GENIE 16 C-Benutzer Zugriff auf eine große Auswahl an Software, die für den IBM bzw. IBM-kompatible geschrieben wurde.

MS-DOS erlaubt Ihnen, Programme zu laden und zu starten, Disketten zu kopieren und zu formatieren, Files zu verändern und beliebige Hardware zu steuern, die an den Rechner angeschlossen ist.

Auf der mit MS-DOS beschrifteten beiliegenden Diskette befinden sich eine Reihe von Programmen, die zusammen das MS-DOS Betriebssystem ergeben.

Um das DOS benutzen zu können, müssen Sie wissen, wie man es lädt, startet und bedient.

Die komplette Auflistung der MS-DOS-Befehle und ihre Funktionen sind in einem der folgenden Kapitel dieses Handbuchs zu finden.

Sie müssen jedoch nur einen kleinen Teil der MS-DOS Kommandos zu kennen, um damit arbeiten zu können.

Stellen Sie sich unter dem DOS einen Werkzeugkasten vor, worin sich als Werkzeuge die einzelnen Programme befinden. Wie in jedem Werkzeugkasten sind einige Werkzeuge derart spezialisiert, daß sie nur sehr selten eingesetzt werden, andere dagegen werden Sie so gut wie jedes Mal einsetzen, wenn Sie den Rechner benutzen. Mit letzteren sollen Sie nun bekanntgemacht werden.

Eine Bemerkung zur Schreibweise:

Das DOS akzeptiert zwar seine Eingaben in Groß- und Kleinschrift. In diesem Handbuch sind jedoch alle Eingaben in Großschrift ausgeführt.

5.2 Das Booten des DOS

Wenn Sie mit Ihrem MS-DOS arbeiten wollen, muß die Diskette gebootet werden. Mit dem englischen Wort "booten" bezeichnet man das erste Laden des Betriebssystems von der System-Diskette in den Arbeitsspeicher.

1. Schalten Sie den Monitor bzw. Fernseher, den evtl. vorhandenen Drucker und zuletzt den Computer ein. Sie hören einen kurzen Kontrollton, gefolgt vom Selbsttest. Nun beginnt das Laufwerk zu surren und die Leuchtdiode auf Laufwerk A leuchtet auf.
2. Schieben Sie nun die DOS-Diskette in Laufwerk A (unten) und schließen Sie es.

Das DOS wird jetzt von der Diskette in den Hauptspeicher geladen.

Sobald der Ladevorgang beendet ist, erlischt die Leuchtdiode, und auf dem Bildschirm erscheinen die Fragen nach Datum und Zeit.

Sollte der Computer bereits eingeschaltet sein, stecken Sie die DOS-Diskette in Laufwerk A und drücken <Ctrl>, <Alt> und gleichzeitig, was einen System-Reset bewirkt.

Der Ladevorgang des MS-DOS ist mit dem oben beschriebenen identisch.

5.3 Eingabe des Datums

Zur Eingabe des Datums benutzen Sie die Zahlentasten in der obersten Reihe der Tastatur.

1. Geben Sie eine bzw. zwei Ziffern für den Monat ein.
Das Datum muß in amerikanischer Form eingegeben werden. (Der Monat kommt also vor dem Tag.)
2. Geben Sie einen Schrägstrich '/' ein.
(Ein Bindestrich '-' ist ebenfalls zulässig.)
3. Nun folgen eine oder zwei Ziffern zur Angabe des Tages.
4. Geben Sie wieder einen Schrägstrich '/' oder Bindestrich '-' ein.
5. Anschließend können Sie die Jahreszahl zwei- oder vierstellig angeben.
'1985' ist also ebenso zulässig wie '85'.
6. Drücken Sie <ENTER>

Das Datum wird nun im Speicher abgelegt und ist immer aktuell sofern der Rechner nicht abgeschaltet wird.

Sollten Sie einen Fehler gemacht haben, gibt Ihnen das DOS eine Fehlermeldung in der Art:

```
Invalid date  
Enter new date:
```

Überprüfen Sie dann folgendes:

1. Monat, Tag und Jahr müssen mit einem Schrägstrich oder Bindestrich voneinander getrennt sein.
2. Es dürfen nur Zahlen und keine Buchstaben an dieser Stelle eingegeben werden.
3. Das DOS nimmt keine unmöglichen Daten an. Haben Sie z.B. 34 als Tag eingegeben, meldet das DOS einen Fehler. Zulässige Jahreszahlen sind 1980 bis 2079.

Das DOS besteht nicht darauf, daß Sie das Datum eingeben. Wollen Sie die Eingabe umgehen, drücken Sie einfach <ENTER>.

5.4 Eingabe der Zeit

Nach der Eingabe des Datums erscheint z.B.:

```
Current time is 9:30:42.21  
Enter new time:
```

Angezeigt werden Stunden, Minuten, Sekunden und Hundertstelsekunden.

Um die Zeit einzugeben, benutzen Sie wieder die Zahlentasten auf der obersten Reihe der Tastatur.

1. Geben Sie die Stunden zwischen 0 und 23 an.
2. Tippen Sie einen Doppelpunkt ':'.
3. Geben Sie die Minuten ein.
4. Tippen Sie wieder einen Doppelpunkt ':'.
5. Geben Sie die Sekunden ein.
6. Tippen Sie nun einen Punkt '.'.
7. Geben Sie die Hundertstelsekunden ein
8. Drücken Sie <ENTER>.

Wenn Sie eine Eingabe weglassen, so werden alle folgenden Werte auf Null gesetzt (Eingabe 9:30 entspricht 9:30:00.00).

Wenn das DOS einen Fehler bei der Eingabe der Zeit entdeckt, meldet es:

Invalid Time
Enter new Time:

Überprüfen Sie dann folgendes:

1. Haben Sie Doppelpunkte zwischen Stunden, Minuten und Sekunden gesetzt?
2. Falls Sie Hundertstelsekunden eingegeben haben, haben Sie einen Punkt als Trennzeichen verwendet?

Wie beim Datum ist es auch nicht unbedingt notwendig die Zeit einzugeben. Drücken Sie dann einfach <ENTER> wenn die Eingabe der Zeit vom Rechner verlangt wird.

5.5 Wechseln von Laufwerk A zu Laufwerk B

Nachdem Sie Datum und Zeit eingegeben haben, erscheint das sogenannte Prompt:

A>

und Sie befinden sich in der DOS-Befehls-ebene. Unterschiedliche Programme haben auch unterschiedliche Prompts, wenn Sie aber 'A>' sehen, erwartet das DOS einen Befehl.

Gegebenenfalls wollen Sie Programme auf dem zweiten Laufwerk benutzen. Um auf das zweite Laufwerk umzuschalten, tippen Sie:

B: <ENTER>

Das Prompt ist nun:

B>

Schalten Sie nun zurück.

A: <ENTER>

Das Prompt ist wieder :

A>

5.6 Formatieren einer Diskette

Jede neue Diskette muß vor der Benutzung formatiert werden. Dabei werden auf der Diskette Daten von evtl. vorhergehenden Aufzeichnungen gelöscht. Das Kommando zum Formatieren einer Diskette lautet:

FORMAT

Bei der Formatierung beschreibt der Rechner die Diskette mit einem Muster, prüft sie auf Fehler und baut ein Inhaltsverzeichnis auf, in dem die Filenamen gespeichert werden.

Um eine Diskette zu formatieren, ist folgendes durchzuführen:

1. Booten Sie MS-DOS in Laufwerk A
2. Wenn das Prompt erscheint, geben Sie ein:

```
FORMAT B: <ENTER>
```

3. Es erscheint die Meldung:

```
Insert new diskette for drive B:  
and strike any key when ready.
```

4. Legen Sie Ihre neue Diskette in Laufwerk B und drücken Sie eine beliebige Taste.

5. Es erscheint nun die Meldung:

```
Formatting ....
```

6. Ist das Formatieren beendet, sehen Sie folgende Meldung:

Formatting...Format complete

362496 bytes total disk space

362496 bytes available on Disk

Format another (Y/N) ?

Hinweis: Wenn bei der Formatierung einer Diskette eine Fehlermeldung ausgegeben wird, empfehlen wir Ihnen, die verwendete Diskette zu entfernen und gegen eine andere auszutauschen.

7. Drücken Sie "Y" für "Ja" (wenn Sie weitere Disketten formatieren wollen) oder "N" für "Nein".

5.7 System- und nicht-System Disketten

MS-DOS besteht aus drei verschiedenen Files, die beim Booten alle von der Systemdiskette geladen werden:

COMMAND.COM bearbeitet die eingegebenen Befehle und startet die entsprechenden Programme

ERSOIO.BIN kontrolliert die Hardware der Anlage.

MSDOS.SYS verwaltet Ihre Disketten.

ERSOIO.BIN und MSDOS.SYS sind normalerweise 'versteckt', d.h. sie sind nicht im Inhaltsverzeichnis der Diskette eingetragen.

Um eine Diskette, die diese drei Files enthält, in einem GENIE 16 C booten zu können, muß jedoch auch der File DIOS.COM auf ihr enthalten sein, da DIOS.COM für die Ein/Ausgabe von Diskette verantwortlich ist und vom BROM (Boot ROM) aufgerufen wird. Disketten, auf denen diese vier Programme stehen, nennt man Systemdisketten. Sie können diese vier Programme von der DOS-Diskette auf eine vorher formatierte Diskette übertragen. Diese Diskette ist dann eine Systemdiskette. Sie kann anstelle der DOS-Diskette benutzt werden, wann immer das System gebootet werden soll.

Wird versucht, das DOS von einer Diskette ohne System zu laden, erfolgt eine Fehlermeldung:

Non-DOS disk error reading drive A
Abort, Retry, Ignore?

Nehmen Sie die Diskette aus dem Laufwerk, legen Sie eine Systemdiskette ein und drücken Sie eine Taste. Das DOS wird geladen.

5.8 Formatieren einer Systemdiskette

Der Ablauf ist ähnlich wie bei einer normalen Formatierung, nur daß der Befehl lautet:

```
FORMAT B: /S <ENTER>
```

Das 'S' steht für System. Wenn das Formatieren beendet ist, erscheint folgende Meldung:

```
Formatting ....Format complete  
System transferred
```

```
362496 bytes total disk space  
40960 bytes used by system  
321536 bytes available on disk
```

```
Format another (Y/N)?
```

Die Meldung teilt Ihnen mit, wieviel Platz die Systemfiles einnehmen und wieviel Platz auf der Diskette noch verfügbar ist. Die Diskette enthält nun die Files

```
COMMAND.COM  
ERSOIO.BIN  
MSDOS.SYS
```

Damit im GENIE 16 C Sie die so erstellte Diskette bootet können, müssen Sie noch den File DIOS.COM mit Hilfe des Befehls

```
COPY A:DIOS.COM B:
```

von der MS-DOS-Diskette in Laufwerk A auf die neu formatierte Systemdiskette in Laufwerk B kopieren.

5.9 Kopieren von Disketten

Wenn Sie Ihre MS-DOS-Systemdiskette ordnungsgemäß erworben haben, dürfen Sie für den eigenen Gebrauch so viele Kopien herstellen, wie Sie benötigen.

Von diesem Recht sollten Sie so bald wie möglich nach dem Kauf des Systems Gebrauch machen, da nie ausgeschlossen werden kann, daß eine System-Diskette beschädigt wird. Heben Sie also das Original gut auf und arbeiten Sie nur mit Duplikaten.

Es muß allerdings auch erwähnt werden, daß gewisse kommerzielle Programme nicht kopierbar sind. In diesem Fall verpflichtet sich der Lieferant jedoch im allgemeinen zur Ersatzlieferung, falls das Original beschädigt wird.

Das Kopieren einer ganzen Diskette wird durch ein DOS-Kommando gesteuert.

Der DOS-Befehl zum Kopieren einer gesamten Diskette lautet:

DISKCOPY

Zur Sicherheit empfehlen wir Ihnen, zunächst das DOS selbst zu duplizieren und nur mit dem erstellten Duplikat alle weiteren Programmierversuche durchzuführen.

Lesen Sie sich bitte, bevor Sie mit dem Kopiervorgang beginnen, die folgende Anleitung sorgfältig durch.

Booten Sie das DOS ganz normal auf Laufwerk A. Versichern Sie sich, daß das Prompt 'A>' ist.

Legen Sie jetzt eine formatierte (!) Diskette, auf die Sie das DOS kopieren wollen, in Laufwerk B.

Geben Sie ein:

DISKCOPY A: B:

und drücken Sie <ENTER>.

Sollte das DOS einen Fehler melden, überprüfen Sie folgendes:

1. Zwischen A: und B: müssen Leerzeichen stehen
2. Doppelpunkte dürfen nur nach A und nach B stehen.

Wenn Ihre Eingabe ordnungsgemäß vollzogen wurde befindet sich jetzt folgende Meldung auf dem Bildschirm:

Insert source diskette in Drive A:
Insert formatted target diskette in drive B:

Strike any key when ready.

Die 'Source Disk' ist in diesem Falle Ihre original MS-DOS-Diskette und diese befindet sich schon in Laufwerk A und die formatierte Zieldiskette sollte sich auch schon schon in Laufwerk B befinden.

Drücken Sie nun eine beliebige Taste, um DISKCOPY mitzuteilen, daß die Kopieraktion beginnen kann.

Folgende Meldung erscheint:

Copying

Während nun das DOS von Laufwerk A nach Laufwerk B kopiert wird, werden die beiden Laufwerke abwechseln laufen, was auch durch das Aufleuchten der roten Leuchtdioden angezeigt wird.

Der erfolgreiche Abschluß der Kopieraktion wird Ihnen durch die Meldung

Copy complete
Copy another (Y/N) ?

übermittelt.

Tippen Sie jetzt 'N'.

Damit haben Sie dem Rechner mitgeteilt, daß Sie nicht keine weiteren Disketten kopieren wollen.

Es erscheint wieder das Prompt:

A>

Nehmen Sie die neue MS-DOS-Diskette aus Laufwerk B und beschriften Sie sie mit einem Filzstift. Arbeiten Sie von nun an nur noch mit diesem Duplikat und legen Sie die Originaldiskette an einen sicheren Ort.

5.10 Files

Eine Reihe zusammengehörender Informationen auf der Diskette nennt man 'File'. Sie können z.B. dem Computer Informationen über einen Lagerbestand geben und diese Informationen würden in einem File aufbewahrt. Ein anderes File könnte z.B. Informationen über Personal oder Geldbestände enthalten.

Filenamen:

Wollen Sie auf die Informationen eines Files zugreifen, müssen Sie dem DOS seinen Namen mitteilen. Dieser 'Filename' hat zwei Teile.

1. Filename:

Dieser kann bis zu acht Buchstaben lang sein und darf aus folgende Zeichen bestehen:

Buchstaben z.B. TCS

Zahlen 0 bis 9 z.B. TCS1985

und folgende Sonderzeichen:

\$ # & @ ! % () _ < > ' - / ^

2. Die Erweiterung:

Eine Erweiterung muß nicht angegeben werden, ist aber sinnvoll, um zwischen verschiedenen Filesorten zu unterscheiden.

So können Sie z.B. die Erweiterung

.TXT

benutzen, um einen Textfile zu kennzeichnen. Manche Programme ordnen ihren Files spezielle Erweiterungen zu, BASIC erwartet z.B.

.BAS

als Erweiterung.

Eine Erweiterung - beginnt immer mit einem Punkt '.'
- besteht aus einem, 2 oder 3 Buchstaben
- folgt unmittelbar dem Filenamem.

Beispiel : TROMME85.TXT

Der Filename heißt in diesem Beispiel 'TROMME85' und durch die Erweiterung '.TXT' ist kenntlich gemacht, daß es sich um einen Textfile handelt.

Merke: Benennen Sie ein File mit Filenamen und Erweiterung, müssen Sie diese Erweiterung bei jedem Zugriff auf dieses File angeben. Geben Sie einen Filenamen ohne Erweiterung an, so werden bestimmte Defaulterweiterungen benutzt. Unter MS-DOS sind dies die Erweiterungen

.COM bzw. .EXE

und unter BASIC

.BAS

Es ist sinnvoll, im Filenamen den Inhalt des Files zu erwähnen. Der Filename AAAAAAAAA.BBB ist sicherlich annehmbar, jedoch wenig informativ.

Wenn Ihr angegebener Filename vom DOS nicht akzeptiert wird, erhalten Sie eine Fehlermeldung.

Überprüfen Sie in diesem Fall folgende Fehlermöglichkeiten:

1. Sind Leerzeichen zwischen den Buchstaben?
2. Besteht der Filename aus mehr als 8 Zeichen?
Besteht die Erweiterung aus mehr als 3 Zeichen?
3. Sind im Namen oder in der Erweiterung Kommata enthalten?
4. Haben Sie den Punkt zwischen Filename und Erweiterung vergessen?

Filespecs:

Zusätzlich zum Filenamem können Sie dem DOS noch angeben, auf welchem Laufwerk sich der angegebene File befindet. Um das entsprechende Laufwerk anzusprechen, fügen Sie vor dem Filenamem noch den entsprechenden Buchstaben und einen Doppelpunkt ein. Diesen Komplex aus Laufwerksangabe, Filenamem und Erweiterung nennt man Filespec. 'Filespec' kommt aus dem Englischen, wird gebildet aus 'file' und 'specification' und kennzeichnet ein File genau. Ein gültiger Filespec wäre z.B:

A:TROMME.TCS

Es sind keine Leerzeichen zwischen den einzelnen Teilen des Filespecs erlaubt.

Wenn sich der gesuchte File auf dem aktiven Laufwerk befindet, kann die Angabe des Laufwerks entfallen.

Wird der File auf dem angegebenen Laufwerk nicht gefunden, erhalten Sie die Fehlermeldung:

Bad command or file name

In BASIC erscheint in diesem Falle die Ausgabe der Meldung:

File not found

Es gibt einige Namen, die nicht benutzt werden dürfen, weil sie reserviert sind.

Es sind z.B. Namen, die sich auf Peripheriegeräte beziehen.

Solche 'illegalen' Namen sind:

CON: Die Konsole. Normalerweise die Tastatur und der Bildschirm, es sei denn, der CTTY-Befehl wurde angewendet.

AUX:

oder

COM1: Serielle Schnittstelle #1

COM2: Serielle Schnittstelle #2

PRN:

oder

LPT1: Paralleler Drucker #1

LPT2: Paralleler Drucker #2

LPT3: Paralleler Drucker #3

NUL: Ein Dummy-Gerätename

Wichtiges über die Gerätenamen:

- Bevor Sie einen Gerätenamen benutzen, stellen Sie sicher, daß das Gerät betriebsbereit und angeschlossen ist.
- Anstelle von Quell- oder Zielfilenames kann jederzeit ein passendes Gerät angegeben werden.
- MS-DOS wird jede Laufwerksangabe bzw. Erweiterung, die mit Gerätenamen zusammen angegeben wurde, ignorieren.
- Die Eingabe eines Doppelpunktes nach dem Gerätenamen ist nicht unbedingt nötig.

5.11 Wildcards

Der Begriff 'Wildcard' bezeichnet zwei besondere Zeichen bei der Eingabe von Filenamen. Mit Hilfe dieser Zeichen können Sie mehrere Files, die ähnliche Namen haben, mit einem einzigen Filenamen bezeichnen.

Das '?' Wildcard:

Ein Fragezeichen (?) an einer Stelle im Filenamen bedeutet, daß dieser Platz von jedem beliebigen Zeichen ausgefüllt werden kann. So wählt das DOS sämtliche Filenamen aus, die in allen bis auf die durch '?' besetzten Zeichen übereinstimmen.

Ein Beispiel:

Nehmen wir an, folgende Files befänden sich auf der Diskette im aktiven Laufwerk:

TROMME84.TXT
TROMME83.TXT
TROMME85.TXT
PERSON84.TXT
PERSON83.TXT

Wenn Sie eingeben

DIR TROMME8?.TXT

werden alle TROMME-Files auf dem Bildschirm erscheinen (DIR ist ein DOS-Befehl zum Anzeigen des Inhaltsverzeichnisses einer Diskette. Mehr Information über DIR finden Sie in den nächsten Kapiteln).

Es können mehr als ein '?' verwendet werden, und das '?' kann jede beliebige Position im Filenamem bzw. in der Erweiterung einnehmen. DIR ??????84.TXT gibt also die Files

TROMME84.TXT und
PERSON84.TXT

aus.

Das '*' Wildcard:

Ein '*' in einem Filenamem bzw. einer Erweiterung entspricht einer Eingabe eines Filenamens bzw. einer Erweiterung, der bis zum Ende mit '?' aufgefüllt ist. TROMME*.C* entspricht also einer Eingabe von TROMME??.*??.

Ein Beispiel:

Nehmen wir an, folgende Files befänden sich auf der Diskette im aktiven Laufwerk:

LIST1.EXE
LIST2.EXE
LIST3.EXE
LIST4.EXE
LISTINGS

Geben Sie ein:

DIR LIST*.*

so werden alle fünf Filenamem angezeigt. Es besteht die Möglichkeit, '?' und '*' gleichzeitig anzuwenden. Mit *.* werden alle Files auf einer Diskette angesprochen.

5.12 Anzeige des Inhaltsverzeichnis

Um alle Files einer MS-DOS-Diskette anzuzeigen, geben Sie ein:

DIR <ENTER>

Auf dem Bildschirm sollte eine Tabelle etwa wie folgt erscheinen:

```
Volume in drive A is ms dos 16 c
Directory of A:\

COMMAND  COM      15957  11-10-83  12:03p
DIOS     COM      8192  12-03-84  12:08a
ANSI     SYS      2524  10-17-83   2:16p
ASSIGN   COM       811   9-24-84  12:03a
BASIC    EXE     69008  1-01-84  10:30p
CHKDSK   COM     6468  10-19-83  7:51p
COLOUR   BAS     1206   1-01-80  12:13a
DEBUG    COM    12223  10-19-83  7:52p
DISKCOMP COM     1983  11-02-84  5:53p
DISKCOPY COM    2444   3-08-83  12:00p
EDLIN    COM     8080  10-19-83  7:51p
EXE2BIN  EXE     1649  10-19-83  7:51p
EXEFIX   EXE    11776  10-19-83  7:52p
FC        EXE     2585  10-19-83  7:51p
FDISK    COM     6177   3-08-83  12:00p
FIND     EXE     6331  10-19-83  7:51p
FORMAT   EXE     4341   1-26-84  11:17a
GERMAN   COM     1573   3-08-83  12:00p
GFTPRN   COM      250   1-01-80  12:34a
GRAPHICS BAS     2304   5-02-85   1:01p
GRAPHPAT COM    1073   1-01-80  12:02a
LINK     EXE    42330  10-19-83  7:51p
MODE     COM     2281   1-01-80  12:06a
MORE     COM     4364  10-19-83  7:51p
PRINT    COM     4506   1-01-80  12:30a
RECOVER  COM     2295  10-19-83  7:51p
REVERSI  BAS     5984   5-02-85  12:15p
SET40    COM      16   1-01-80  12:02a
SET80    COM      16   1-01-80  12:03a
SHIPZON1 EXE     1282   8-01-84  12:10a
SHIPZON2 EXE     1282   8-01-84  12:11a
SMODE    COM     2108   1-15-85   2:36p
SORT     EXE     1632   1-26-84  11:54a
SYS      COM      948   1-01-80  12:38a
CONFIG   SYS      21    5-15-85  12:15p
AUTOEXEC      18    5-15-85  11:18a
AUTOEXEC BAT    84    5-15-85  11:19a
DOSCOPY  BAT     49    5-15-85  12:31p

38 File(s)          77824 bytes free
```

Inhaltsverzeichnis einer anderen Diskette:

1. Laden Sie das DOS in Laufwerk A
2. Legen Sie die zu untersuchende Diskette in Laufwerk B ein.
3. Geben Sie ein:

DIR B:

4. Drücken Sie <ENTER>

Sie müssen dem DOS sagen, wo sich die zu bearbeitende Diskette befindet. Geben Sie keine Laufwerksbezeichnung an, sucht das DOS auf dem aktiven Laufwerk.

Anzeige eines einzelnen Filenamens:

1. Laden Sie das DOS in Laufwerk A
2. Legen Sie die zu bearbeitende Diskette in Laufwerk B ein
3. Geben Sie ein:

DIR B:<filename>

4. Drücken Sie <ENTER>

Der vollständige Filename, seine Größe, Datum und Zeit der letzten Änderung werden angezeigt.

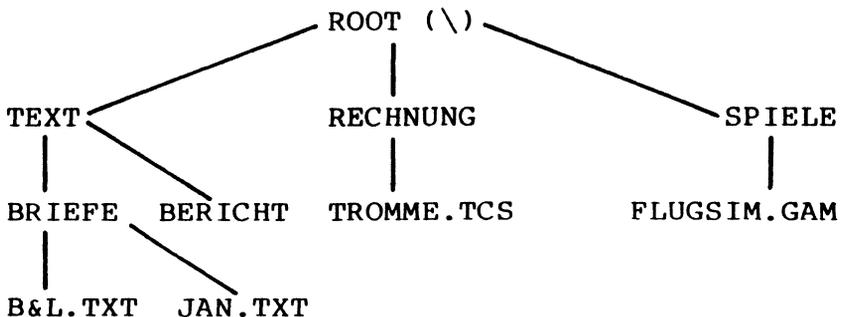
5.13 Einführung in die File-Directories (Inhaltsverzeichnis)

Wie schon erwähnt, werden alle Filenamen in einem Inhaltsverzeichnis auf jeder Diskette abgelegt. Dieses Inhaltsverzeichnis nennt man 'Directory'.

Das ursprüngliche Directory auf einer Diskette nennt man Root- oder Systemdirectory. Es wird jedesmal neu angelegt, wenn eine Diskette formatiert wird.

Wenn Sie eine große Zahl von Files auf einer Diskette haben, werden Sie es vorziehen, diese Files in einzelne Gruppen zu unterteilen. Um diese Unterteilung vorzunehmen, werden sog. Unterdirectories (auch Subdirectories genannt) gebildet. Es können z.B. alle Textfiles eine Gruppe bilden, alle Spielefiles oder alle Rechnungen usw. Diese Unterdirectories können wiederum die Namen von anderen Unterdirectories enthalten. Man nennt dies 'hierarchische Fileorganisation'.

Beispiel:



Die Namen ROOT, TEXT, BRIEFE, BERICHT, RECHNUNG und SPIELE sind Directories, B&L.TXT, JAN.TXT, TROMME.TCS und FLUGSIM.GAM sind Files.

Der umgekehrte Schrägstrich (\) neben ROOT ist ein besonderes Symbol für das Systemdirectory.

Wenn Sie den deutschen Tastaturtreiber (GERMAN.COM) geladen haben, erreichen Sie den umgekehrten Schrägstrich (\) durch gleichzeitiges Drücken der Tasten <Ctrl><Alt>< ? >.

In der ersten Ebene, das Rootdirectory, ist nur ein Element, eben das Rootdirectory. In der zweiten Ebene gibt es drei Elemente, TEXT, RECHNUNG und SPIELE. Alle Elemente sind Subdirectories. Das Subdirectory TEXT ist in sich noch unterteilt in die Subdirectories BRIEFE und BERICHT. Das Subdirectory BRIEFE enthält schließlich 2 Files, B&L.TXT und JAN.TXT

Benennung eines Subdirectories:

Die Regeln zum Benennen eines Subdirectories sind dieselben wie die Regeln zum Benennen eines Files.

Es ist möglich, denselben Filenamen zweimal zu benutzen, wenn sich beide Files in verschiedenen Unterdirectories befinden. Z.B. könnte ein File JAN.TXT auch im Subdirectory SPIELE stehen, ohne mit JAN.TXT aus BRIEFE in Konflikt zu kommen.

Das aktive Directory:

Das aktive Directory ist, genau wie das aktive Laufwerk, dasjenige Directory, welches das DOS benutzt, solange es nicht gezwungen wird, ein anderes zu benutzen.

Um zu überprüfen, in welchem Directory Sie sich befinden bzw. um die Directories zu wechseln, benutzen Sie den Befehl:

CHDIR

CHDIR heißt soviel wie 'CHange DIRectory'. Nach dem Start des DOS ist immer das Systemdirectory aktiv. Es bleibt solange aktiv, bis ein CHDIR ausgeführt wird.

(Weitere Details über CHDIR in den nächsten Kapiteln).

Paths (Wege) zu Unterdirectories und Files:

Wenn Sie hierarchisch aufgebaute Directories verwenden, muß das DOS wissen, über welchen Path man auf ein Unterdirectory bzw. den File zugreifen kann. Dieses geschieht, indem man dem DOS den sog. Pathnamen angibt. Es handelt sich dabei um eine Sequenz von Unterdirectory-Namen, die mit einem Filenamen enden. Die Namen müssen durch umgekehrte Schrägstriche (\) getrennt werden.

Der Path, den Sie angeben, kann vom Systemdirectory oder aktiven Directory ausgehen. Ein umgekehrter Schrägstrich am Anfang des Pathnamen läßt das DOS vom Systemdirectory ausgehen.

Beispiel:

```
CHDIR \TEXT\BRIEFE
```

wäre ein Path zum Unterdirectory BRIEFE:

Wird der erste Schrägstrich weggelassen, beginnt der Path im aktiven Directory. Befinden Sie sich z.B. im Directory TEXT und wollen ins Directory BERICHT, geben Sie ein:

```
CHDIR BERICHT
```

Wenn sich der gesuchte File im aktiven Directory befindet, muß natürlich kein Path angegeben werden.

Es gibt zwei besondere Kurzbefehle bei der Pathsuche, die MS-DOS bearbeiten kann:

- . Repräsentiert den Namen des aktiven Directories. MS-DOS legt diesen Namen an, wenn ein Subdirectory angelegt wird.
- .. Repräsentiert den Namen des Vorgängerdirectories vom aktuellen Directory. TEXT ist also das Vorgängerdirectory von BRIEFE und BERICHT etc.

Dies ist sinnvoll, da so ein schneller Wechsel von einem Directory in ein anderes ermöglicht wird. Ist das aktive Directory z.B. BRIEFE und Sie wollen in BERICHT, so wird eingegeben:

```
CHDIR ..\BERICHT
```

Die Kurzformen können auch mit DOS-Befehlen wie z.B. DIR verwendet werden.

DIR ..

Listet alle Files im Vorgängerdirectory.

DIR ..\..\

Listet alle Files im 'Vorvorgängerdirectory'

Eine Pathbeschreibung darf nicht länger als 63 Zeichen sein, vor jeder Pathbeschreibung darf ein Laufwerkskennbuchstabe stehen. Ein gültiger Pathname ist also:

B:\SPIELE\FLUGSIM.GAM

5.14 DOS-Befehle und Pathbeschreibungen

Es gibt zwei Arten von DOS-Befehlen:

1. Externe Befehle

Externe Befehle sind auf der Diskette als Programmfile enthalten. Sie müssen also von der Diskette geladen werden, bevor sie ausgeführt werden können. Wenn Sie dem DOS einen externen Befehl geben, sucht es im aktiven Directory nach dem entsprechenden File. Wenn die externen Befehle sich in einem anderen Directory befinden, muß dem DOS mitgeteilt werden, in welchem Directory es nach ihnen suchen soll. Dieses geschieht mit dem Befehl

PATH

Wenn z.B. BERICHT das aktive Directory ist, und die externen Befehle in TEXT stehen, weist der Befehl

PATH \TEXT <ENTER>

das DOS an, im aktiven Directory (BERICHT) und in TEXT zu suchen.

2. Interne Befehle

Interne Befehle sind ins DOS eingebaut und werden sofort ausgeführt. Oft können Paths als Befehlsargumente benutzt werden, was die Handhabung von Subdirectories sehr einfach und effektiv macht.

Einige Beispiele:

```
DIR \TEXT\BRIEFE <ENTER>
```

listet alle Files im Directory BRIEFE

```
DEL \TEXT\BRIEFE <ENTER>
```

löscht alle Files in Directory BRIEFE.

Wollen Sie alle Files in einem Unterdirectory löschen, fragt Sie der Computer:

```
'Are you sure (Y/N)?'
```

Das DOS verlangt also eine Bestätigung, um alle Files löschen zu können. Drücken Sie Y, wenn gelöscht werden soll und N, wenn nicht gelöscht werden soll.

Der Befehl

```
TYPE
```

zeigt den Inhalt eines Files auf dem Bildschirm an.

```
TYPE \TEXT\BRIEFE\JAN.TXT <ENTER>
```

Hier muß also am Pathende ein Filename eingegeben werden.

Anzeigen des aktiven Directories:

Um den Namen des aktiven Directories herauszufinden, geben Sie ein:

```
CHDIR <ENTER>
```

Ist die aktive Directory z.B. BERICHT, gibt Ihnen DOS die Meldung:

```
A: TEXT BERICHT
```

MS-DOS teilt Ihnen mit, in welchem Directory Sie sich befinden und mit welchem Laufwerk Sie gerade arbeiten.

Um den Inhalt des aktiven Directories anzuzeigen, geben Sie ein

```
DIR <ENTER>
```

Der Rechner wird Ihnen ein ähnliches Bild wie das folgende anzeigen:

```
Volume in Drive A has no label
Directory of A: TEXT BERICHT
.                <DIR>          1-01-80 12:00a
..              <DIR>          1-01-80 12:00a
ADDRESS         17984         1-01-80 12:59a
COLOUR.BAS     1408          1-01-80 12:08a
               4 File(s) 296960 bytes free
```

Aus den Angaben geht hervor, daß der Diskette bei ihrer Formatierung kein Name gegeben wurde.

BERICHT enthält hier 2 Files, ADDRESS und COLOUR.BAS

Wie Sie bemerkt haben werden, werden Files und Directories gemeinsam gelistet. Es kann also nicht der gleiche Name für ein File und eine Directory verwendet werden.

- . Repräsentiert das aktive Directory (hier BERICHT).
- .. Repräsentiert das Vorgängerdirectory des aktiven Directories (hier TEXT).

Erzeugung eines Subdirectories:

Um im aktiven Directory ein Subdirectory zu erzeugen, benutzen Sie den Befehl:

MKDIR (= MaKe DIRectory)
oder
MD

Um z.B. ein neues Subdirectory mit dem Namen TROMME zu erzeugen, geben Sie ein:

MKDIR TROMME

Um ein Subdirectory in einem anderen Teil der Directory-Baumstruktur zu erzeugen, geben Sie ein:

MKDIR <Pathname><Name des neuen Subdir.>

Wechseln des aktiven Directories:

Es ist sehr einfach, vom aktiven Subdirectory in ein anderes zu wechseln. Der Befehl dazu lautet:

```
CHDIR <Pathname>
```

Der Befehl

```
CHDIR ..
```

bringt das MS-DOS immer ins Vorgängerdirectory des aktiven Directories.

```
CHDIR \
```

wechselt immer ins System-Directory

Der Befehl

```
CHDIR \TEXT\BERICHT
```

wechselt vom aktiven Directory auf BERICHT über.

Löschen eines Subdirectories:

Der Befehl zum Löschen eines Subdirectories lautet:

```
RMDIR <Name der Directory>
```

(RMDIR = ReMove DIRectory)

Ist das Subdirectory in einem anderen als dem aktiven Directory definiert, muß ein Weg beschrieben werden.

Der Befehl lautet also vollständig:

RMDIR <Pathname><Directoryname>

Der Befehl arbeitet nur, wenn die zu löschende Subdirectory vollständig leer ist. Ausgenommen sind allerdings die '.' und '..' Einträge.

Um alle Files im Subdirectory zu entfernen, benutzen Sie den Befehl:

DEL <Wegname>

Um z.B. alle Files in BERICHT zu löschen, tippen Sie :

DEL \TEXT\BERICHT

MS-DOS fragt:

'Are you sure'(Y/N) ?

Die '.' und '..' Eintragungen können nicht gelöscht werden. MS-DOS erzeugt diese als ein Teil der Directorystruktur.

6. Einführung in die DOS-Befehle

6.1 Allgemeine Hinweise

Ein 'Befehl' ist ein Kommando, welches der Benutzer dem System gibt. Die DOS-Befehle werden benutzt, um das System folgende Aufgaben ausführen zu lassen:

- Kopieren und Formatieren von Disketten;
- Kopieren, Vergleichen, Anzeigen und Löschen von Files;
- Kopieren von DOS-Files auf eine andere Diskette;
- Laden und Ausführen von Systemprogrammen wie EDLIN;
- Laden und Ausführen von Anwenderprogrammen;
- Laden und Ausführen Ihrer eigenen Programme;
- Filenamen und Inhaltsverzeichnisse auflisten;
- Datum und Zeit eingeben;
- Zahlreiche Drucker- und Bildschirmoptionen ein- und ausschalten.

6.2 Interne und externe Befehle

Interne Befehle sind die am häufigsten benutzten Befehle. Sie werden sofort vom DOS ausgeführt.

MS-DOS kennt folgende interne Befehle:

BREAK
CHDIR
CLS
COPY
CTTY
DATE
DEL(ERASE)
DIR
ECHO
EXIT
FOR
GOTO
IF
MKDIR(MD)
PATH
PAUSE
PROMPT
REM
REN(RENAME)
RMDIR(RD)
SET
SHIFT
TIME
TYPE
VER
VERIFY
VOL

Die Befehle in Klammern erfüllen denselben Zweck und können ebenso verwendet werden.

Externe Befehle werden als Programmfiles auf der Diskette gespeichert. Das DOS muß sie erst von Diskette laden, bevor sie ausgeführt werden können. Wenn der entsprechende File sich nicht im aktiven Directory des aktiven Laufwerks befindet, kann MS-DOS ihn nicht lesen und gibt folgende Fehlermeldung aus:

'Bad command or file name'

Alle Filenamen mit den Erweiterungen .COM , .EXE und .BAT sind externen Befehlen zugeordnet.

So sind z.B. FORMAT.COM und FIND.EXE externe Befehle.

Weil alle externen Befehle als Files auf der Diskette stehen, können Sie selber neue Befehle entwerfen und diese in das vorhandene System einfügen. Die meisten Programme, die von Programmiersprachen abgespeichert werden, also Benutzerfiles sind, sind .EXE-Files. (EXE kommt von 'execute' (ausführen)).

Wenn Sie einen externen Befehl aufrufen, werden keine Erweiterungen mit eingegeben.

Externe Befehle des MS-DOS:

ASSIGN
CHDSK
DEBUG
DISKCOMP
DISKCOPY
EXE2BIN
EDLIN
FC
FDISK *
FIND
FORMAT
GERMAN
GRAPHPAT
LINK
MODE
MORE
PRINT
RECOVER
SET40
SET80
SHIPZON1 *
SHIPZON2 *
SORT
SYS

* Hinweis:

Diese Befehle können nur in Verbindung mit einer installierten Harddisk benutzt werden und sind daher in diesem Handbuch nicht beschrieben.

Befehlsoptionen:

Einem DOS-Befehl können zusätzliche Informationen durch eine sog. Befehlsoption hinzugefügt werden. Wird diese Möglichkeit ausgelassen, setzt DOS einen sog. Defaultwert voraus.

Tippen Sie z.B.:

DIR

nimmt MS-DOS den Buchstaben des aktuellen Laufwerks als Defaultwert an. Wird jedoch mehr Information gegeben, wie z.B. :

DIR B:

so wird der Defaultwert übergangen und B: als Wert eingesetzt.

Die Defaultwerte für die unterschiedlichen Befehle werden in einem späteren Kapitel aufgeführt.

Hinweis zu allen DOS-Befehlen:

- Befehle werden in der Regel von einer oder mehreren Optionen gefolgt.
- Befehle und Optionen können in Groß- oder Kleinschrift eingegeben werden.
- Befehle und Optionen werden durch sog. Begrenzer (engl. Delimiters) voneinander getrennt (z.B. , ; =).
- Filespecs (Laufwerk:Filename.Erw) enthalten schon Begrenzer (Doppelpunkt und Punkt). Unterteilen Sie diese drei Teile also nicht weiter mit Begrenzern.
- Wenn ein File eine Erweiterung besitzt, muß diese mit dem Filenamen zusammen angegeben werden.

- Die meisten Befehle, die einen Filenamen erfordern, erlauben auch die Angabe einer Path-Beschreibung vor dem Filenamen. Wenn Sie keine Subdirectories verwenden, brauchen Sie keine Path-Beschreibung.
- Befehle, die sich in Arbeit befinden, können durch folgende Tastenkombination abgebrochen werden:

<Ctrl><Break>

- Das DOS führt Befehle nur aus, wenn anschließend <ENTER> gedrückt wird.
- Wildcards (? und *) und Gerätenamen können nicht als Befehle, wohl aber als Befehlsoptionen benutzt werden.
- Wenn ein Programm eine große Menge an Daten auf dem Schirm darstellt, rollt der Bildschirm automatisch auf die nächste Seite. Um dieses 'Scrollen' zu unterbrechen, können die Tastenkombinationen

<Ctrl><Num Lock>

oder

<Ctrl><S>

benutzt werden.

Um fortzufahren, drücken Sie eine beliebige Zahlen- oder Buchstabentaste.

- Die DOS-Editier- und Kontrolltasten können während der Eingabe einer DOS-Kommandozeile benutzt werden.

- Diskettenlaufwerke werden auch als Quell- (engl. source) und Ziellaufwerke (engl. destination) bezeichnet. Das Quellaufwerk ist dasjenige Laufwerk, von dem die Information kommt, das Ziellaufwerk ist dasjenige Laufwerk, auf das die Information geschrieben wird.
- Das gewöhnliche Prompt, das zur Eingabe eines Befehls auffordert, ist der momentane Laufwerksbuchstabe und das Größer-Zeichen (>).
A> ist also ein Prompt.
- Wenn ein Befehl ausgeführt wurde, meldet sich DOS mit dem Prompt zurück. Wird keine Fehlermeldung ausgegeben, wurde der Befehl erfolgreich beendet.

6.3 Batchverarbeitung

Wenn Sie eine Reihenfolge von Befehlen oft verwenden, werden Sie es für nützlicher befinden, einen sog. Batch-File anzulegen. Dieser File führt dann die gesamten Befehle durch, indem man nur seinen Namen aufruft.

Batchfilenamen:

Jeder Batchfile muß mit der Erweiterung .BAT versehen werden. Es wird später jedoch nur noch der Filename eingegeben, um den File aufzurufen. Die Erweiterung .BAT ist also Defaultwert.

Erzeugen von Batchfiles:

Es gibt zwei Möglichkeiten, einen Batchfile zu erzeugen.

1. Mit Hilfe des Zeileneditors (EDLIN).
Informationen über die Benutzung von EDLIN finden Sie in einem späteren Kapitel.
2. Durch Benutzen des COPY-Befehls direkt von der Tastatur aus.

Batchbefehle:

Es stehen zwei DOS-Befehle zur ausschließlichen Verwendung in einem Batchfile zur Verfügung:

REM und PAUSE

REM erlaubt das Einfügen von Kommentaren in einen Batchfile.

PAUSE hält das System an und erlaubt ein Abbrechen oder Fortfahren.

Beispiel:

Ein sinnvoller Batchfile z.B. formatiert eine Diskette und überprüft diese. Dieser Batchfile würde folgendermaßen aussehen:

1. REM Formatieren und Überprüfen von neuen Disketten.
2. REM Der Name dieses Files ist NEWDISK.BAT.
3. PAUSE (Legen Sie die neue Diskette in Laufwerk B)
4. FORMAT B:
5. DIR B:
6. CHKDSK B:

Um diesen File auszuführen, geben Sie einfach ein:

```
NEWDISK <ENTER>
```

Das Resultat ist dasselbe, als wenn Sie die einzelnen Befehle direkt von der Tastatur aus eingegeben hätten.

Die drei Schritte sind:

1. Eine Batch-Datei erstellen (NEWDISK z.B.)
2. Die Datei mit .BAT versehen und abspeichern
3. Die Datei kann jetzt wie ein neuer Befehl verwendet werden.

Hinweise zur Batchverarbeitung:

- Nur der Filename wird eingegeben, nicht die Erweiterung.
- Nur Files der Form <Filename>.BAT werden ausgeführt.
- Werden <Ctrl> und <Break> während der Ausführung gedrückt, erscheint folgende Meldung:

Terminate batch job (Y/N)?

Wenn Sie mit Y antworten, wird die Ausführung des Batchfiles abgebrochen und das Prompt 'A>' erscheint wieder.

Bei der Eingabe N wird nur der gerade bearbeitete Befehl abgebrochen, und das System fährt mit der Ausführung des Batchfiles fort.

- Entfernen Sie während der Ausführung eines Batchfiles die Diskette, auf der sich der File befindet, verlangt das DOS das Einlegen der Diskette, bevor der nächste Befehl gelesen wird.

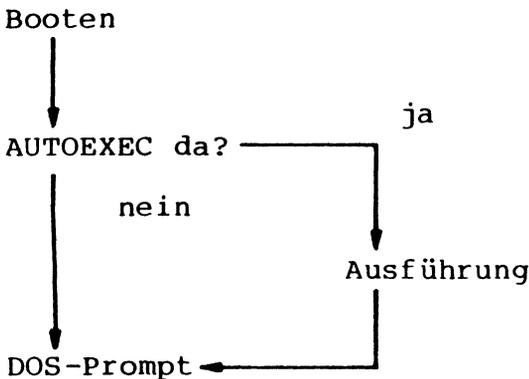
- Der letzte Befehl eines Batchfiles kann der Name eines anderen Batchfiles sein. Diese Möglichkeit erlaubt es Ihnen, mehrere Batchfiles aneinanderzuhängen.

Der AUTOEXEC.BAT File:

AUTOEXEC steht für AUTOMATIC program EXECution bzw. automatische Programmausführung. Der AUTOEXEC.BAT-File erlaubt es Ihnen, Programme automatisch zu starten, wenn Sie das DOS booten. Das ist sinnvoll, wenn Sie ein bestimmtes Programmpaket automatisch starten wollen.

Wird DOS geladen, sucht der Rechner auf der DOS Diskette nach AUTOEXEC.BAT. Findet er das File, wird es sofort ausgeführt. Die Eingaben für Zeit und Datum werden dann übersprungen.

Wird jedoch kein AUTOEXEC.BAT gefunden, geht DOS in den normalen Modus über. Das folgende Diagramm zeigt, wie DOS ein AUTOEXEC.BAT-File bearbeitet.



Erzeugen eines AUTOEXEC.BAT-Files mit COPY:

Angenommen, Sie wollen nach dem Booten sofort BASIC laden und ein Programm namens MENU starten. Dazu können Sie folgendes AUTOEXEC.BAT File erzeugen:

1. Geben Sie ein:

```
COPY CON: AUTOEXEC.BAT
```

Dieser Befehl weist DOS an, alle folgenden Informationen von der Tastatur in das genannte File zu kopieren.

2. Tippen Sie

```
BASIC MENU
```

Dieser Befehl geht sofort in den AUTOEXEC.BAT-File.

3. Tippen Sie nun <F6> und ENTER, um den Befehl zu übernehmen (<ENTER>) und die Eingabe zu beenden (<F6>).
4. MENU wird jetzt automatisch nach jedem Booten starten.

(Um Ihr eigenes BASIC-Programm zu benutzen, tippen Sie Ihren Filenamen anstelle von MENU).

Hinweise zu AUTOEXEC.BAT-Files:

- Das AUTOEXEC.BAT-File muß im Systemdirectory einer DOS-Diskette stehen.
- Sie können jeden beliebigen DOS-Befehl oder jede DOS-Befehlsfolge in den File einbauen.
- Das DOS fragt Sie nicht nach Datum und Zeit, es sei denn, Sie verwenden die TIME- und DATE-Befehle im AUTOEXEC.BAT-File. Dies empfiehlt sich, da DOS das Datum und die Zeit verwendet, um Ihr Directory zu ergänzen.

Erstellung eines .BAT-Files mit ersetzbaren Parametern:

Es kann sein, daß Sie ein Anwenderprogramm mit unterschiedlichen Daten laufen lassen wollen. Diese Daten können in verschiedenen DOS-Files gespeichert sein. Ein .BAT-File mit ersetzbaren Parametern kann dieses Problem lösen.

Ein Parameter ist eine Option, die an Ihre Befehle angehängt werden kann und die das DOS mit zusätzlichen Informationen versorgt. AT-DOS bietet Ihnen die Möglichkeit, einen Batchfile mit ersetzbaren (sog. dummy-) Parametern zu erzeugen. Diese werden dann durch die Werte ersetzt, die beim Ausführen des Files angegeben werden.

Die Dummyparameter heißen %0, %1 ... bis %9.

Wenn Sie z.B. folgende Befehlszeile eingeben:

```
COPY CON: MYFILE.BAT
```

werden die folgenden von der Tastatur aus eingegebenen Daten in den File MYFILE.BAT kopiert.

```
A>COPY CON: MYFILE.BAT <ENTER>  
COPY %1.MAC %2.MAC <ENTER>  
TYPE %2.PRN <ENTER>  
TYPE %0.BAT <ENTER>
```

Drücken Sie nun <F6> und anschließend <ENTER>. DOS antwortet mit der Meldung

```
1 File(s) copied  
A>
```

Der File MYFILE.BAT, bestehend aus drei Befehlen, befindet sich nun auf der Diskette im aktiven Laufwerk A.

Die Dummyparameter %1 und %2 werden nun von den Werten ersetzt, die Sie beim Ausführen des Files angeben. Der Dummyparameter %0 wird immer vom Laufwerksbuchstaben und vom Filenamen des Batchfiles ersetzt.

Wichtiges über das Erzeugen eines .BAT-Files mit ersetzbaren Parametern:

- Bis zu zehn Dummyparameter können angegeben werden.

- Wenn Sie ein Prozentzeichen (%) als Teil eines Filenamens innerhalb eines Batchfiles benutzen wollen, müssen Sie dieses zweimal eingeben.

Beispiel:

Um den Filenamens DNE%.EXE innerhalb eines Batchfiles zu benutzen, muß dieser als DNE%%.EXE eingegeben werden.

Ausführung eines Batchfiles:

Um den File MYFILE.BAT auszuführen und um die Werte anzugeben, die die Dummyparameter ersetzen sollen, müssen Sie den Filenamens (nicht die Erweiterung) gefolgt von den einzusetzenden Parametern eingeben.

Um MYFILE.BAT auszuführen, tippen Sie also

```
MYFILE A:PROG1 B:PROG2
```

%0 wird jetzt durch MYFILE, %1 durch A:PROG1 und %2 durch B:PROG2 ersetzt, anschließend wird der Batchfile ausgeführt.

Wenn anstatt der Dummyparameter die angegebenen Werte im Batchfile stehen würden, wäre das Ergebnis identisch.

6.4 Input und Output

Das DOS geht normalerweise davon aus, daß alle Eingaben von der Tastatur kommen und alle Ausgaben an den Bildschirm gehen. Es besteht jedoch die Möglichkeit, den Input bzw. Output "umzulenken".

Der Input z.B. kann von einem File kommen und der Output kann z.B. an den Drucker oder auch in ein File gehen.

Weiterhin können noch sogenannte PIPES erstellt werden, die den Output des einen Befehls zum Input des nächsten werden lassen.

Umleiten des Output:

Um die Ausgaben eines Programms nicht auf den Schirm, sondern in ein File gehen zu lassen, benutzen Sie das 'Größer als'-Zeichen '>'.
>

Beispiele:

DIR erzeugt z.B. üblicherweise ein Listing des Inhaltsverzeichnisses auf dem Bildschirm.

Derselbe Befehl DIR, nun als DIR >FILE.TXT, schickt die Ausgaben in das File FILE.TXT. Existiert noch kein File dieses Namens, wird es von DOS erzeugt. Existiert ein File FILE.TXT schon, wird dessen Inhalt überschrieben.

Die Angabe

```
DIR >PRN
```

druckt ein Listing des Directories auf dem Drucker.

Wenn Sie ein vorhandenes File nicht löschen, sondern ergänzen wollen, so benutzen Sie einfach zwei '>' Zeichen.

Der Befehl

```
DIR >>LIST.TXT
```

hängt an den File LIST.TXT ein Directory-listing an.

Unter bestimmten Bedingungen sollen die Eingaben an ein Programm nicht von der Tastatur kommen, sondern von einem File.

Das MS-DOS erlaubt dieses unter Benutzung eines 'Kleiner als'-Zeichens '<'.

Beispiel:

```
SORT <NAMEN >LIST1
```

Sortiert den Inhalt des Files NAMEN alphabetisch und speichert das sortierte Ergebnis im File LIST1. Die Eingaben erfolgen also vom File NAMEN und nicht von der Tastatur.

Wenn Sie diese Möglichkeit ausschöpfen, um ein Programm mit Daten zu versorgen, gehen Sie sicher, daß alle vom Programm verlangten Daten im File verfügbar sind. Sollten weniger Daten im File sein, als das Programm verlangt, so ist das DOS nicht in der Lage, das Programm mit weiteren Daten zu versorgen und das System wird sich "aufhängen". Sie können in diesem Fall zum DOS-Prompt 'A>' zurückkehren, indem Sie <Ctrl> und <Break> drücken.

Filter:

Ein Filter ist ein Programm, welches Ihre Eingaben 'liest', verarbeitet und anschließend wieder ausgibt. Die eingegebenen Daten werden also 'gefiltert'.

SORT ist z.B. ein Filter. Es liest Ihre Eingaben, sortiert diese und gibt sie dann wieder aus. Filter können in vielen verschiedenen Variationen zusammengesetzt werden. So können Sie oft wenige Filter benutzen, um eine ganze Reihe von Befehlen zu ersetzen.

Auf der MS-DOS Diskette befinden sich drei Filter:

FIND Sucht nach einer bestimmten Buchstaben- oder Zahlenfolge in einem Programm.

MORE Anzeige eines vollen Bildschirms, Pause mit der Anzeige '-MORE-', wenn weitere Anzeigen vorhanden sind.

SORT Sortiert Textdaten

PIPES:

Wenn Sie dem System mehr als einen Befehl gleichzeitig geben wollen, so können Sie PIPEn. Es ist z.B. sinnvoll zu PIPEn, wenn Sie den Output eines Befehls als Input für einen anderen benötigen.

Das PIPEn geschieht mit dem sogenannten PIPE-Trennzeichen (|).

Das PIPE-Zeichen (|) wird nach laden des deutschen Tastaturtreibers folgendermaßen erreicht:

Drücken Sie <Alt> und halten Sie diese Taste gedrückt. Geben Sie dann auf dem numerischen Tastenfeld die Zahl 124 ein und lassen die Alt-Taste los.

Der Befehl `DIR |SORT` gibt z.B. ein alphabetisch sortiertes Inhaltsverzeichnis aus. Der Output von DIR wurde also an SORT als Input weitergeleitet.

Eine PIPE-Zeile kann auch aus mehr als zwei Befehlen bestehen. `DIR |SORT|MORE` z.B. zeigt Ihnen das Directory alphabetisch sortiert an und hält jedesmal an, wenn ein Bildschirm gefüllt ist. Es erscheint dann `'-MORE-'`, wenn weitere Daten verfügbar sind.

7. Die DOS-Befehle

=====
Befehlsformate:

Folgendes ist wichtig bei der Eingabe von DOS-Befehlen:

- Alle Befehls Worte (hier großgeschrieben) können in Groß- oder Kleinschrift eingegeben werden.
- Argumente stehen in eckigen Klammern <>. Anstelle dieses beschreibenden Textes (z.B. <filename>) müssen Sie ein entsprechendes Argument (i.o. Beispiel also einen Filenamem) angeben.
- Daten, die in Nummernzeichen (#) eingefasst sind, können wahlweise eingegeben werden. Ihre Eingabe ist also nicht unbedingt erforderlich. (Dabei werden nur die Daten, nicht aber die Nummernzeichen eingegeben !!!)
- Runde Klammern bedeuten, daß deren Inhalt beliebig oft wiederholt werden kann.
- Sämtliche Sonderzeichen wie Punkt oder Komma müssen unbedingt mit eingegeben werden. (außer < und #)
- 'd' in der Formatbeschreibung bezieht sich immer auf einen Laufwerksbuchstaben.
- Durch einen Strich (|) getrennte Daten können wahlweise eingegeben werden. Bei ON OFF kann also entweder ON oder OFF eingegeben werden.

Befehl: ASSIGN

Typ: Extern

Funktion: Veranlaßt das DOS das vorgegebene Laufwerk auf ein anderes umzuleiten.

Format: ASSIGN #d=e#

Kommentar: Der ASSIGN-Befehl wird benutzt um alle Anfragen auf ein Laufwerk auf ein anderes umzuleiten. Dies kann z.B dann von Nutzen sein, wenn ein Anwenderprogramm, das fest auf ein Laufwerk zugreift, ohne Änderungen auf einem anderen Laufwerk (z.B. Harddisk) lauffähig gemacht werden soll. Beachten Sie, daß bei der Eingabe der Laufwerksangaben keine Doppelpunkte mit eingegeben werden dürfen.

Beispiel: ASSIGN B=A

Bewirkt, daß alle Zugriffe auf Laufwerk B auf Laufwerk A umgeleitet werden.

Wird nur

ASSIGN

eingegeben, wird MS-DOS in den Urstand, ohne Umleitungen zurückversetzt.

Befehl: BREAK

Typ: Intern

Funktion: Bei jedem DOS-Aufruf durch ein Programm wird die Ctrl-Break Funktion überprüft.

Format: BREAK #ON|OFF#

Kommentar: Wenn Sie ein Anwenderprogramm benutzen, das Ctrl-Break als Funktionstaste benutzt, ist es sinnvoll, die DOS Ctrl-Break Funktion außer Betrieb zu setzen. BREAK ON aktiviert und BREAK OFF inaktiviert diese Funktion. Die Eingabe von BREAK ohne Parameter bewirkt die Ausgabe des aktuellen Status (ON oder OFF).

Befehl: CHDIR (CD)

Typ: Intern

Funktion: Wechselt vom aktiven Directory auf ein anderes Directory oder zeigt das aktive Directory an.

Format: CHDIR #pathname#
bzw.
CD #pathname#

Kommentar: Wird kein Laufwerk angegeben, nimmt DOS das aktive Laufwerk.

Beispiele: CD

Zeigt das aktive Directory an.

CD \

Wechselt vom aktiven Directory ins Systemdirectory.

CD ..

Wechselt vom aktiven Directory in dessen Vorgängerdirectory.

Wenn Ihr aktuelles Directory \TEXT\BERICHT ist und Sie ins Directory

\TEXT\BERICHT\ADDRESS

wollen, lautet der Befehl:

CD \TEXT\BERICHT\ADDRESS

oder

CD ADDRESS

Befehl: CHKDSK

Typ: Extern

Funktion: Liest das Directory auf der angegebenen Diskette und gibt einen Bericht über den Zustand von Diskette und Speicher.

Format: CHKDSK #d:# <filespec> #/F# #/V#

Kommentar: Wird ein Filename angegeben, zeigt CHKDSK die Zahl der nicht durchgehenden Bereiche (non contiguous blocks) auf der Floppy an, die vom File belegt werden. Es handelt sich dabei um keine Fehlermeldung, sondern lediglich um einen Hinweis. Je höher die Zahl der nichtdurchgehenden Bereiche ist, desto länger wird die Ladezeit. CHKDSK überprüft nur das aktive Directory.

CHKDSK sollte gelegentlich auf jeder Diskette eingesetzt werden, um diese nach Fehlern im Directory zu überprüfen. Werden Fehler gefunden, gibt CHKDSK einen Zustandsbericht.

Beispiel: Ausgabe eines Statusberichtes
nach der Eingabe von CHKDSK

```
362496 bytes total disk space
 22528 bytes in 3 hidden files
194560 bytes in 22 user files
145408 bytes available on disk
```

```
131072 bytes total memory
105312 bytes free
```

Hinweise: CHKDSK korrigiert keine Fehler,
wenn nicht die /F (fix) Option
eingesetzt wurde.

Eine Eingabe von /V bewirkt die
Ausgabe weiterer Meldungen
während des Programmablaufs.

Wird die Ausgabe von CHKDSK umge-
leitet, so darf die /F Option
nicht benutzt werden.

Folgende Fehler werden automa-
tisch beseitigt, wenn die /F-Op-
tion benutzt wurde:

- Invalid drive specification
- Invalid parameter
- Invalid sub-directory entry
- Cannot CHDIR to <filename>
Tree past this point not
processed
- First cluster number is inva-
lid, entry truncated
- Allocation error, size adjusted
- Has invalid cluster, file trun-
cated

- Disk error reading FAT
- Disk error writing FAT
- <filename> contains non-contiguous blocks
- All specified file(s) are contiguous

Folgende Fehler können nicht automatisch beseitigt werden:

'Incorrect DOS version'

CHKDSK benötigt mindestens die MS-DOS Version 2.11.

'Insufficient memory
Processing cannot continue'

Ihr Rechnersystem verfügt nicht über ausreichenden Speicherplatz, um CHKDSK auf dieser Diskette anzuwenden.

'Errors found, F parameter not specified
Corrections will not be written to disk'

Die /F Option muß angegeben werden, wenn die Fehler korrigiert werden sollen.

'Invalid current directory
Processing cannot continue'

Booten Sie das System erneut und
starten Sie CHKDSK nochmal.

'Cannot CHDIR to root
Processing cannot continue'

Die Diskette ist beschädigt. Ver-
suchen Sie, das DOS zu booten und
rufen Sie RECOVER auf.

'<filename> is cross linked on
cluster'

Ein Datenblock auf der Diskette
ist mehreren Files zugeordnet.
Kopieren Sie den File, den Sie
behalten wollen, auf einen an-
deren Namen, dann löschen Sie al-
le zusammenhängenden Files von
der Diskette.

'X lost clusters found in Y
Chains Convert lost chains to
files (Y/N)?'

Datenblöcke sind als belegt ge-
kennzeichnet, obwohl sie keinem
File zugeordnet sind. Beantworten
Sie die Frage mit Y (yes), so
wird ein Directoryeintrag er-
zeugt, der diese Blöcke enthält.

Löschen Sie diesen dann, werden die Blöcke wieder als frei gekennzeichnet. (Von CHKDSK erzeugte Files werden mit FILEnnnn benannt)

CHKDSK wird jetzt anzeigen:

'X bytes disk space freed'

Wird die Frage mit N beantwortet und /F wurde nicht angegeben, so werden die Blöcke von CHKDSK freigegeben und es wird angezeigt:

'X bytes disk space would be freed'

'Probable non-DOS disk
Continue (Y/N) ?'

Die eingelegte Diskette ist wahrscheinlich keine DOS-Diskette. Geben Sie an, ob CHKDSK weiterarbeiten soll oder nicht.

'Insufficient room in root
directory Erase files and
repeat CHKDSK'

CHKDSK benötigt mehr Platz im Systemdirectory. Löschen Sie einige Files und starten Sie CHKDSK neu.

'Unrecoverable error in directory
Convert directory to file (Y/N)?'

Wird diese Frage mit Y beantwortet, wird die Directory in einen File verwandelt und kann in dieser Form verbessert werden.

Befehl: CLS

Typ: Intern

Funktion: Löscht den Bildschirm

Format: CLS

Befehl: COPY

Typ: Intern

Funktion: Kopiert einen oder mehrere Files auf eine andere Diskette. Wenn gewünscht, können die zwei Files unterschiedliche Namen erhalten. Soll auf die gleiche Diskette kopiert werden, so ist es unbedingt erforderlich, den Files verschiedene Namen zu geben, es sei denn, es werden verschiedene Unterdirectories benutzt.

Format: COPY <filespec>#filespec##pathname##pathname##/V#

Kommentar: Wird die zweite Filespec-Option nicht angegeben, so wird sich die Kopie auf dem aktiven Laufwerk befinden und denselben Namen tragen. Befindet sich das erste File auf dem aktiven Laufwerk, und wird der zweite File nicht angegeben, so wird der Kopiervorgang abgebrochen.

Das DOS wird jetzt anzeigen:

```
'File cannot be copied onto itself  
0 File(s) copied'
```

Der zweite Filespec kann in drei Formen angegeben werden:

1. Nur durch die Laufwerks-
angabe d:

Der File wird unter demselben Namen auf die die angegebene Diskette kopiert.

2. Wird nur der Filename eingegeben, wird der File unter diesem Namen auf das aktive Laufwerk kopiert.

3. Wird ein vollständiger Filespec angegeben, so wird der Originalfile entsprechend den Angaben im Filespec kopiert.

Die Option /V bewirkt ein nachträgliches Überprüfen aller geschriebenen Sektoren.
Bei Anwendung dieser Option wird COPY langsamer ablaufen.

Der COPY-Befehl erlaubt ebenfalls ein 'Aneinanderhängen' von Files. Dieses erfolgt einfach mit dem Zeichen '+':

```
COPY A.TXT+B.TXT+C.TXT BIGFILE.ABC
```

Die Files A.TXT, B.TXT und C.TXT werden aneinandergehängt und unter dem Namen BIGFILE.ABC abgespeichert.

Es lassen sich auch alle Files mit einer bestimmten Erweiterung aneinanderhängen.

```
COPY *.LST COMBIN.PRN
```

Alle Files mit der Erweiterung .LST werden aneinandergehängt und unter COMBIN.PRN abgespeichert.

Im folgenden Beispiel werden alle Files, deren Name sowohl mit der Erweiterung .LST als auch mit der Erweiterung .REF auftaucht, zusammengefaßt und unter demselben Namen, aber mit der Erweiterung .PRN, abgespeichert.

```
COPY *.LST+*.REF *.PRN
```

FILE1.LST wird also mit FILE1.REF zu FILE1.PRN zusammengefügt, XYZ.LST und XYZ.REF zu XYZ.PRN etc.

Benutzen Sie nie ein COPY mit +, wenn die Möglichkeit besteht, daß Quellfile und Zielfile den selben Namen haben.

Der Befehl

COPY *.LST ALL.LST

würde z.B. so lange funktionieren, bis das DOS versuchen würde, den File ALL.LST an sich selber anzuhängen. In diesem Fall findet das DOS nie das Ende des Files, da jeder Sektor, der kopiert wird, den File verlängert.

COPY vergleicht den Filenamen des Inputfiles mit dem Filenamen des Outputfiles. Liegt derselbe File-name vor, so wird der Inputfile übergangen und es wird die Fehlermeldung:

'Content of destination lost
before copy'

ausgegeben. Das weitere Anhängen erfolgt normal.

So kann z.B. der Befehl

COPY ALL.LST+*.LST

benutzt werden, um alle .LST Files im File ALL.LST zu vereinen. Dieser Befehl erzeugt keine Fehlermeldungen und ist der korrekte Weg, um Files aneinanderzuhängen.

Befehl: CTTY

Typ: Intern

Funktion: Erlaubt es, das Gerät zu ändern, von dem die Kommandos eingegeben werden. (TTY ist die Tastatur).

Format: CTTY <Gerät>

Kommentar: <Gerät> ist das Gerät, über das Eingaben an das DOS erfolgen. Dieser Befehl wird sinnvoll, sobald dieses nicht mehr die Tastatur sein soll.

Der Befehl

CTTY AUX

leitet alle Funktionen der Tastatur auf den AUX-Port um.

Der Befehl

CTTY CON

hingegen schaltet wieder auf das ursprüngliche Gerät um (hier die Tastatur um).

Befehl: DATE

Typ: Intern

Funktion: Erlaubt Eingabe oder Veränderung des ursprünglich benutzten Datums. Dieser Befehl kann natürlich auch in einem Batchfile eingesetzt werden. Bei Benutzung von AUTOEXEC.BAT ist dies besonders zu empfehlen, da sonst keine Abfrage stattfindet.

Format: DATE #<mm>-<dd>-<yy>#

Kommentar: Wird DATE ohne Datum eingegeben, so erscheint:

```
Current date is <mm>-<dd>-<yy>
Enter new date:
```

Drücken Sie jetzt <ENTER>, so wird das angegebene Datum übernommen.

Das Datum kann jedoch auch direkt eingegeben werden. Die entsprechende Syntax ist:

```
DATE 3-9-81
```

Das neue Datum darf nur mit Zahlen eingegeben werden. Dabei ist folgendes zu beachten:

```
<dd> : zwischen 1 und 31
<mm> : zwischen 1 und 12
<yy> : zwischen 80 und 79 oder
       zwischen 1980 und 2079
```

Die Eingaben müssen entweder mit
'-' oder '/' getrennt werden.
Beim Weiterstellen des Datums
berücksichtigt das DOS, die Länge
des momentanen Monats. Das DOS
erkennt auch Schaltjahre.

Ist die Datumseingabe ungültig,
so erscheint:

Invalid date
Enter new date:

Es wird also die erneute Eingabe
des Datums verlangt.

Befehl: DEL (ERASE)

Typ: Intern

Funktion: Löscht alle Files mit dem angegebenen Filespec.

Format: DEL #pathname##filespec#

Kommentar: Sollen alle Files auf der Diskette gelöscht werden, so ist *.* einzugeben. Das DOS erwartet dann jedoch eine Bestätigung. Obwohl die Wildcards * und ? bei DEL benutzt werden können, sollten Sie diese Möglichkeit nur vorsichtig anwenden. Eine falsche Eingabe genügt, um einen ganzen Satz Programme zu löschen.

Der Befehl ERASE kann anstelle von DEL benutzt werden.

Beispiele: DEL *.*

Löscht alle Files in der aktiven Directory.

DEL #d:# #pathname#

Löscht alle Files im angegebenen Directory.

DEL A:DPFILE.TXT

Löscht den angegebenen File auf Drive A.

Befehl: DIR

Typ: Intern

Funktion: Listet das Directory.

Format: DIR #pathname# #filespec# #/P# #/W#

Kommentar: Wird nur DIR eingegeben, kommt es zur Ausgabe des Inhaltsverzeichnis der aktiven Diskette. Wird hingegen hinter DIR eine Laufwerksangabe d: vorgenommen, so werden alle Einträge des angegebenen Laufwerks angezeigt.

Wird ein Filespec angegeben, so werden alle Files, die diesen Namen haben, gelistet.

In jedem Fall werden alle Files mit ihrer Größe und Datum ihrer Erzeugung aufgelistet.

Die Wildcards ? und * können benutzt werden.

Zwei Optionen können mit DIR angegeben werden. Die /P Option bedeutet Seiten-Modus. Sobald eine Seite vollgeschrieben ist, stoppt das System, es muß eine beliebige Taste gedrückt werden, um fortzufahren.

Wird /W angegeben, werden nur die Filenamen ausgegeben, es paßt somit mehr Information auf den Schirm. Es werden 5 Filenamen pro Zeile an gezeigt.

Befehl: DISKCOMP

Typ: Extern

Funktion: Vergleicht zwei ganze Disketten miteinander. Die Anwendung dieses Befehls ist z.B. als Kontrolle nach einem DISKCOPY sinnvoll.

Format: DISKCOMP #d:# #d:# #/l# #/8#

Kommentar: Als Laufwerksangabe kann auch nur ein Laufwerk angegeben werden. In diesem Fall müssen die zu vergleichenden Disketten nach Aufforderung gewechselt werden. Wird die /l-Option gewählt, werden nur die Vorderseiten der beiden Disketten verglichen, auch wenn die Disketten doppelseitig beschrieben sind. Analog beschränkt die /8-Option den Vergleich auf 8 Sektoren pro Spur. DISKCOMP zeigt Vergleichsfehler mit Spur- und Sektornummerangabe an. Abschließend wird gefragt, ob weitere Disketten verglichen werden sollen (Compare more diskettes (Y,N)?)

Befehl: DISKCOPY

Typ: Extern

Funktion: Kopieren einer ganzen Diskette.

Format: DISKCOPY #d:# #d:#

Kommentar: Das erste eingegebene Laufwerk gibt die Quell-, das zweite die Zieldiskette an.

Der Rechner überprüft zunächst, ob die Zieldiskette formatiert ist. Findet er keine Formatierung vor, führt er diese vor dem Kopieren selbsttätig aus.

Quell- und Ziellaufwerk können identisch sein. Sie werden dann im Verlauf des Kopiervorgangs mehrmals aufgefordert, die Disketten zu wechseln. DISKCOPY fordert Sie abwechselnd auf, die Quelldiskette (SOURCE DISK) und die Zieldiskette (DESTINATION DISK) in das Laufwerk einzulegen.

Nach Beenden des Kopiervorganges fragt DISKCOPY, ob noch eine Kopie gemacht werden soll. Wird diese Frage mit Y beantwortet, so erfolgt der Kopiervorgang auf denselben Laufwerken, wie bei der ersten Kopie.

- Wichtig:
1. Werden beide Optionen weggelassen, erfolgt eine Kopie auf dem aktiven Laufwerk.
 2. Wird die zweite Option weggelassen, so erfolgt eine Kopie auf das aktive Laufwerk.
 3. Beide Disketten müssen auf dieselbe Art formatiert worden sein. Ansonsten erfolgt automatisch eine neue Formatierung.
 4. Eine Diskette, auf der schon viele Files erzeugt, verschoben oder gelöscht wurden, ist intern meist sehr zerstückelt. Die Programme befinden sich in Fragmenten überall auf der Diskette. Es sind Verzögerungen beim Laden dieser Files zu erwarten. Benutzen Sie in diesem Fall lieber den COPY-Befehl:

```
COPY A:*. * B:
```

kopiert alle Files von Laufwerk A nach B.

5. Bei einem Fehler während des DISKCOPY-Befehls meldet das DOS:

```
Disk error while reading drive  
Abort, Ignore, Retry?
```

Befehl: EXE2BIN

Typ: Extern

Funktion: Macht aus EXE(cutable)-Files binäre Files, wodurch Diskettenplatz gespart und Ladezeiten verkürzt werden.

Format: EXE2BIN <filespec> #d:##<filename>#<.ext>##

Kommentar: Dieser Befehl konvertiert EXE-Files in das Format der COM-Files.

Der erste Filespec gibt den Quellfile an.
Wird keine Erweiterung angegeben, setzt das DOS .EXE als Erweiterung voraus.

Wird kein Laufwerk angegeben, so wird das Outputfile auf dem Laufwerk des Inputfiles erzeugt.

Wird kein Filename für das Outputfile angegeben, so wird der Filename des Inputfiles benutzt.

Wird beim Outputfile keine Erweiterung angegeben, so erhält dieser die Erweiterung .BIN.

Der residente Code und der Datenteil des Inputfiles dürfen die Länge von 64K nicht überschreiten, und es darf kein STACK-Segment enthalten sein.

Zwei Arten von Bearbeitungen sind möglich, jenachdem, wie der ursprüngliche CS:IP (Code Segment: Instruction Pointer) im EXE-File enthalten ist.

1. Ist CS:IP im EXE-File nicht angegeben, wird eine reine Binärwandlung ausgeführt. Werden Segmentänderungen benötigt (das Programm enthält Befehle, die eine absolute Segmentadressierung vornehmen), werden Sie nach dem sog. fixup value gefragt. Dieser Wert gibt das Segment an, in das der EXE-File geladen wird. Das File, das EXE2BIN dann erzeugt, ist nur lauffähig, wenn es wieder in genau dieses Segment geladen wird. Vom DOS aus ist es nicht lauffähig.
2. Wird CS:IP als 0000:100H angegeben, so wird angenommen, daß dieser File als COM-File lauffähig sein soll. Folgende Bedingungen müssen erfüllt sein:

Die ersten 100H Bytes enthalten keine für das Programm relevante Information.

Das Programm verändert das CS-Register nicht (COM-Files müssen segmentrelokätierbar sein).

Sind diese Bedingungen erfüllt, ist das erzeugte COM-File vom DOS aus aufrufbar.

Sollten diese Bedingungen nicht erfüllt werden, so gibt EXE2BIN folgende Fehlermeldung aus:

'File cannot be converted'

Diese Meldung wird auch ausgegeben, wenn es sich beim Quellfile nicht um einen gültigen, ausführbaren File handelt.

Sollte bei der weiteren Ausführung ein Fehler auftreten, gibt EXE2BIN eine der folgenden Fehlermeldungen aus:

'File not found'

Der File befindet sich nicht auf der Diskette.

'Insufficient memory'

Es ist nicht genügend Speicher vorhanden.

'File creation error'

Das Zielfile kann nicht erzeugt werden. Benutzen Sie CHKDSK, um festzustellen, ob das Directory voll ist o.ä.

'Insufficient disk space'

Es ist nicht genug Platz auf der Zieldiskette vorhanden.

'Fixups needed-base segment
(hex):'

Das Quellfile enthält Informationen, wonach ein Load-Segment für das File benötigt wird. Geben Sie die Segmentadresse an, ab die das File geladen werden soll.

'WARNING - Read error on EXE
file.'

Es ist weniger eingelesen worden, als im Vorspann angegeben wurde. Es handelt sich dabei nur um eine Warnung.

Befehl: EXIT

Typ: Intern

Funktion: Verläßt den Kommandoprozessor COMMAND.COM und kehrt auf die vorherige Ebene zurück (sofern diese existiert).

Format: EXIT

Kommentar: Dieser Befehl kann eingesetzt werden, wenn Sie von einem Programm aus den DOS-Kommandoprozessor aufrufen, um verschiedene Kommandos auszuführen. Geben Sie EXIT ein, kehrt das System wieder in das aufrufende Programm zurück. Der Aufruf des Kommandoprozessors erfolgt mit dem Befehl

COMMAND

Befehl: FIND

Typ: Extern

Funktion: Sucht nach einem angegebenen File oder in einer Gruppe von Files.

Format: FIND #/V|/C|/N# <string> #(<filename>)#

Kommentar: FIND durchsucht den File nach dem angegebenen Text (String). Wird kein Filename angegeben, nimmt FIND den Input von der Tastatur.

Die Optionen von FIND sind:

/V weist FIND an, alle Zeilen anzuzeigen, die NICHT den angegebenen String enthalten.

/C weist FIND an, nur die Anzahl der Zeilen anzuzeigen, in denen eine Übereinstimmung vorlag.

/N weist FIND an, vor jeder Zeile deren relative Zeilennummer im File anzuzeigen.

Der String muß in Anführungszeichen eingefaßt sein.

Beispiel: FIND "Trommeschläger" RECH1.TXT RECH2.TXT

zeigt alle Zeilen an, in denen
"Trommeschläger" vorkommt.

DIR B: |FIND /V "DAT"

Zeigt alle Filenamen an, die
nicht DAT enthalten.

Um einen String, der bereits An-
führungszeichen enthält, zu su-
chen, müssen Sie doppelte Anföh-
rungszeichen eingeben.

FIND "Er sagte: ""Hallo Peter!"" BUCH1.TXT

Tritt ein Fehler auf, antwortet
FIND mit einer der folgenden Feh-
lermeldungen :

'Incorrect DOS version'

FIND arbeitet erst ab Version
2.11 des MS-DOS.

'FIND: Invalid number of para-
meters'

Es wurde kein String angegeben.

'FIND: Syntax error'

Der angegebene String ist fehler-
haft.

'FIND: File not found <filename>'

Der Filename, der angegeben wurde, befindet sich nicht auf der angegebenen Diskette.

'FIND: Read error in <filename>'

Es ist ein Lesefehler aufgetreten.

'FIND: Invalid parameter <Option-Name>'

Es wurde eine Option verlangt, die FIND nicht erkennt.

Befehl: FORMAT

Typ: Extern

Funktion: Formatiert die angegebene Diskette, sodaß das DOS sie benutzen kann.

Format: FORMAT #d#: #/l# #/8# #/0# #/v# #/s#

Kommentar: Wird kein Laufwerk angegeben, wird die Diskette im aktiven Laufwerk formatiert. Die Optionen müssen in der oben angegebenen Reihenfolge angegeben werden.

Die /l Option erzeugt eine einseitige Diskette.

Die /8 Option erzeugt eine Diskette mit 8 Sektoren pro Spur.

Die /0 Option (0 nicht Null) veranlaßt FORMAT, eine zum IBM PC DOS Version 1.X kompatible Diskette zu formatieren. Option /0 verändert die Directory der Diskette, sodaß es von PC DOS und MS-DOS gelesen und bearbeitet werden kann. Diese Möglichkeit sollte aber nur angewendet werden, wenn die Diskette auch wirklich mit diesen Betriebssystemen benutzt werden soll, da jeder DOS-Zugriff unter MS-DOS auf solche Disketten wesentlich länger dauert.

Die /V Option fragt nach dem Namen der Diskette, wenn das Formatieren beendet ist.

Wird /S angegeben, so werden nach dem Formatieren die Systemfiles auf die neue Diskette kopiert.

Dies sind die Files

ERSOIO.BIN,

MSDOS.SYS

und

COMMAND.COM

Hinweis: Um eine MS-DOS-Diskette auf einem GENIE 16 C bootfähig zu machen, muß in jedem Fall noch der File DIOS.COM darauf kopiert werden.

Befehl: GRAPHPAT

Typ: Extern

Funktion: Belegt den Zeichensatz des
Grafikmodus mit dem kompletten
PC-Zeichensatz.

Format: GRAPHPAT

Kommentar: Die Ausführung dieses Kommandos
ist z.B. dann interessant, wenn
im BASIC im Grafikbildschirm
(SCREEN 1 oder SCREEN 2) mit Um-
lauten oder Grafikzeichen gear-
beitet werden soll, da diese
sonst nicht vordefiniert sind.

Befehl: MKDIR (MD)

Typ: Intern

Funktion: Erzeugt ein neues Directory

Format: MKDIR <pathname>
oder
MD <pathname>

Kommentar: Dieser Befehl wird verwendet, um eine hierarchische Directorystruktur aufzubauen. MKDIR erzeugt ein Unterdirectory des aktiven Directories.

Beispiel: MKDIR BERICHT

Erzeugt im Rootdirectory ein Unterdirectory mit dem Namen BERICHT.

Befehl: MODE

Typ: Extern

Funktion: Setzt verschiedene Video-, Dru-
ker und Kommunikationsoptionen.

Format: 1. MODE #40|#80|#W40|#C040|#C080|#MONO# #,#L|#R# #,T#
 2. MODE LPTn:=-COMn
 3. MODE LPTn: #80|132#,#6|##,P#
 4. MODE COMn:#baud##,parity##,databits##,stopbits##,parity##,P#
 5. MODE LPTn:

Kommentar: 1. Setzt den Bildschirmmodus.

Beispiele: MODE 80

80 Zeichen pro Zeile

MODE BW40

40 Zeichen/Zeile Schwarzweiß

MODE ,L

Bildschirm eine Spalte nach links

MODE ,R

Bildschirm eine Spalte nach
rechts

MODE ,L,T

Bildschirm nach links und Testmuster anzeigen.

Kommentar: 2. Leitet die Ausgabe vom Parallelport n zum seriellen Port n.

Beispiel: MODE LPT1:=COM1

Alle Ausgaben auf den Parallelport 1 werden auf den seriellen Port 1 umgeleitet.

Kommentar: 3. Bestimmt für den Drucker die Anzahl der Zeichen pro Zeile und der Zeilen pro Zoll. Es kann noch angegeben werden, ob bei Auftreten eines Druckerfehlers kontinuierlich weitere Versuche unternommen werden sollen:

Beispiel: MODE LPT1:132,8,P

setzt den Drucker 1 auf 132 Zeichen pro Zeile, 8 Zeilen pro Zoll und bei eventuellen Druckerfehlern wird weiterprobiert, nicht abgebrochen.

Kommentar: 4. Angabe der RS 232 Parameter.

Baudrate: 110, 150, 300, 600
1200, 2400, 4800 und 9600.

Parität: N(keine), O(ungerade bzw. odd) E(gerade bzw. even), wobei EVEN die Normal Einstellung ist.

Die Anzahl der Stopbits kann entweder 1 oder 2 betragen. Wird ,P angegeben, werden bei auftretenden Fehlern kontinuierlich Wiederholungen gemacht.

Kommentar: 5. Entwertet die Umleitung von Punkt 2.

Befehl: MORE

Typ: Extern

Funktion: Unterbricht alle 25 Zeilen die
Bildschirmausgabe.

Format: MORE

Kommentar: MORE ist ein Filter, der den In-
put liest und diesen seitenweise
ausgibt.
Es wird dann '-- MORE --' ange-
zeigt, und das DOS wartet auf Be-
tätigung von <ENTER>.
Dieser Befehl ist sinnvoll, wenn
eine große Datenmenge auf dem
Bildschirm dargestellt werden
soll.

Beispiel: TYPE FILE.TXT |MORE

Der Inhalt von FILE.TXT wird sei-
tenweise dargestellt.

Befehl: PATH

Typ: Intern

Funktion: Gibt dem DOS den Path an, auf dem nach externen Befehlen gesucht werden soll.

Format: PATH #<pathname>#(<pathname>)#

Kommentar: Um z.B. das Directory
TROMME\COMPUTER\SOFTWARE
nach externen Befehlen durchsu-
chen zu lassen, geben Sie ein:

```
PATH\TROMME\COMPUTER\SOFTWARE
```

Dieser Path wird solange benutzt,
bis ein anderer angegeben wird
oder das DOS außer Betrieb ge-
setzt wird. Es können auch mehre-
re Paths angegeben werden, wenn
diese durch Semikolons getrennt
werden:

```
PATH EXTERN\BEFEHL1; EXTERN\BEFEHL2
```

durchsucht BEFEHL1 und BEFEHL2
nach externen Befehlen. Dabei
wird in der durch PATH angegege-
benen Reihenfolge vorgegangen.
PATH ohne Optionen gibt den aktu-
ellen Path aus. PATH ; setzt den
Path wieder auf die Systemdirec-
tory.

Befehl: PRINT

Typ: Extern

Funktion: Druckt einen Textfile auf dem Drucker aus, während mit dem System gearbeitet werden kann. Diese Art von Drucken nennt man background printing bzw. spooling.

Format: PRINT (#<filespec># #/T# #/C# #/P#)

Kommentar: Der PRINT-Befehl ist natürlich nur dann sinnvoll, wenn auch ein Drucker an das System angeschlossen ist. Die Optionen sind:

/T Diese Option löscht alle Files aus der Warteschlange für den Drucker und räumt dem angegebenen File damit Priorität ein.

/C Diese Option entfernt das angegebene File und alle folgenden Files aus der Warteschlange. Folgt ein Filename mit /P, so werden ab diesem wieder alle Files in die Schlange eingefügt.

/P Fügt die angegebenen Files in die Warteschlange ein.

PRINT ohne Optionen zeigt den Inhalt der Warteschlange an.

PRINT /T leert die Warteschlange.

PRINT /T *.ASM leert die Warteschlange und fügt alle .ASM Files ein.

PRINT A:TEMP1.TST/C A:TEMP2.TST A:TEMP3.TST

entfernt die angegebenen Files aus der Warteschlange.

PRINT TEMP1.TST/C TEMP2.TST/P TEMP3.TST

entfernt TEMP1.TST aus der Schlange und fügt TEMP2.TST und TEMP3.TST ein.

Fehler: Tritt ein Fehler auf, wird eine der folgenden Fehlermeldungen angezeigt:

Name of list device #PRN:#

Diese Meldung erscheint, wenn PRINT zum erstenmal aufgerufen wird. Jedes angeschlossene Gerät kann angegeben werden, die Ausgabe erfolgt dann auf dieses Gerät. Wird kein Gerät angegeben, wird auf den Drucker ausgegeben.

List output is not being assigned
to a device

Diese Fehlermeldung wird ausgegeben,
wenn das oben angegebene Gerät ungültig ist.

PRINT queue is full

In die Warteschlange passen 10
Files. Wird diese Zahl überschritten,
erscheint obige Fehlermeldung.

PRINT queue is empty

Die Warteschlange ist leer.

No files match d:XXXXXXXXX.XXX

Unter dem angegebenen Filespec
wurde kein File gefunden.

Hinweis:

Soll ein File aus der Warteschlange
gelöscht werden, das gar nicht in der
Schlange steht, wird kein Fehler gemeldet.

Drive not ready

Die Diskette war nicht betriebsbereit, als PRINT versucht auf sie zugreifen wollte. PRINT versucht es solange, bis die Diskette bereit ist.

All files cancelled

Wurde /T angegeben, wird diese Meldung auf dem Drucker ausgegeben. Wird das File, das gerade ausgedruckt wird, aus der Schlange gelöscht, erscheint die Meldung:

File cancelled by operator.

Befehl: PROMPT

Typ: Intern

Funktion: Ändert das DOS-Prompt

Format: PROMPT #(<prompt-text>)#

Kommentar: Dieser Befehl erlaubt es, das DOS-Prompt (d>) gegen ein anderes auszutauschen. Wird kein Text eingegeben, wird d> benutzt. Folgende Zeichenfolgen erzeugen besondere Prompts:

Zeichen	Effekt
-----	-----
\$\$	\$'-Zeichen
\$t	akt. Zeit
\$d	akt. Datum
\$p	Directory des akt. Laufwerks
\$v	Versions-Nr.
\$n	Default-Laufw.
\$g	'>'-Zeichen
\$l	'<'-Zeichen
\$b	' '-Zeichen
\$_	Zeilenvorschub
\$s	Leerzeichen (nur führend)
\$h	Backspace
\$e	lBH (Escape)

Beispiele: PROMPT \$n\$g

setzt das normale DOS-Prompt.

PROMPT Time=\$t

gibt als PROMPT aus:

Time=(aktuelle Zeit)

Ist der ANSI.SYS Treiber geladen,
können Sie auch Escape-Sequenzen
zur Bildschirmformatierung benutzen.

Befehl: RECOVER

Typ: Extern

Funktion: Stellt Files bzw. ganze Disketten mit beschädigten Sektoren wieder her.

Format: RECOVER <filename|d:>

Kommentar: Wenn einer der Sektoren auf Ihrer Diskette beschädigt ist, können Sie entweder RECOVER direkt auf den betroffenen File anwenden, oder Sie wenden RECOVER auf die ganze Diskette an.
Um einen File zu reparieren, geben Sie ein:

```
RECOVER <filename>
```

Der File wird sektorweise gelesen und die beschädigten Sektoren werden markiert, sodaß sie nicht mehr benutzt werden.

Um RECOVER auf eine ganze Diskette anzuwenden, tippen Sie

```
RECOVER <d:>
```

entsprechende Laufwerk angibt.

Sollte im Systemdirectory nicht genug Platz sein, so gibt RECOVER eine Fehlermeldung aus.

Befehl: REM

Typ: Intern

Funktion: Zeigt die Bemerkungen, die hinter REM stehen, während der Ausführung eines Batchfiles an.

Format: REM #Kommentar#

Kommentar: Die einzigen erlaubten Trennzeichen in einem Kommentar sind Leerzeichen, Tabulatoren und Kommata.

Beispiel: 1: REM File formatiert neue Disks
2: REM und heißt NEWDISK.BAT
3: PAUSE Disk in Lw. B einlegen
4: FORMAT B: /S
5: DIR B:
6: CHKDSK B:

Befehl: REN (RENAME)

Typ: Intern

Funktion: Ändert den Namen eines Files

Format: REN <filespec> <filename>
oder
RENAME <filespec> <filename>

Kommentar: Laufwerksangaben in <filename> werden ignoriert, da der File nicht kopiert, sondern nur umbenannt wird, also auf der Diskette bleibt, auf der er sich ursprünglich befunden hat. Wildcards dürfen angewendet werden.

Beispiel: REN *.LST *.PRN

macht aus allen .LST Files
.PRN Files.

Beim Versuch, einen File mit einem Namen zu versehen, der sich bereits im Directory befindet, erscheint die Fehlermeldung:

Duplicate Filename or File not found

Befehl: RD (RMDIR)

Typ: Intern

Funktion: Entfernt ein Directory in einer hierarchischen Directorystruktur.

Format: RMDIR <pathname>

Kommentar: Dieser Befehl entfernt ein Directory. Das zu entfernende Directory muß leer sein, d.h. es darf keine Files mehr enthalten.

Beispiel: RMDIR \TROMME\TCS\TEXTE

Entfernt das angegebene Directory aus der Directorystruktur.

Befehl: SET

Typ: Intern

Funktion: Definiert einen String als äquivalent zu einem anderen, sodaß beide dasselbe bewirken.

Format: SET #<string=string>#

Kommentar: Dieser Befehl ist nur sinnvoll, wenn er in Zusammenhang mit selbstgeschriebenen Programmen verwendet wird. Ein Anwenderprogramm kann alle mit SET angegebenen Optionen durch Angabe von SET ohne Optionen abfragen.

Beispiel: SET TTY=VT52

Setzt den TTY-Wert auf VT52, bis dieser wieder geändert wird.

Weiterhin ist es auch möglich, den SET Befehl bei der Batchverarbeitung sinnvoll einzusetzen. Es können dann auch Namen als Parameter verwendet werden. Diese werden dann einfach mit SET umdefiniert.

In einem Batchfile steht z.B.
LINK %FILE%.

Mit SET FILE=HUGO kann jetzt der Parameter FILE ersetzt werden.

Bemerkung: Wird Text anstelle von Zahlen in einem Batchfile verwendet, so muß dieser in Prozentzeichen eingefaßt werden.

Befehl: SET40

Typ: Extern

Funktion: Umschalten auf 40 Zeichen/Zeile

Format: SET40

Befehl: SET80

Typ: Extern

Funktion: Umschalten auf 80 Zeichen/Zeile

Format: SET80

Befehl: SORT

Typ: Extern

Funktion: Sortiert eine Eingabe und gibt das Ergebnis aus.

Format: SORT #/R# #/+n#

Kommentar: SORT kann benutzt werden, um ein File nach einer bestimmten Spalte zu sortieren.

Es gibt zwei Optionen:

/R kehrt den Sortiervorgang um, es wird also von Z-A, nicht von A-Z, sortiert.

/+n Gibt die Spalte an, nach der sortiert werden soll. Wird dieser Parameter nicht eingegeben, sortiert SORT nach Spalte 1.

Beispiele: SORT R <UNSORT.TXT >SORT.TXT

Der Inhalt des Files UNSORT.TXT wird in absteigender Reihenfolge sortiert und in den File SORT.TXT geschrieben.

DIR | SORT /+14

Das Directory wird nach seiner 14ten Spalte sortiert und angezeigt (Die 14te Spalte enthält die Filelänge).

Befehl: SYS

Typ: Extern

Funktion: Übertragen der Systemfiles von der aktiven Diskette auf die angegebene Diskette.

Format: SYS <d>:

Kommentar: SYS wird normalerweise benutzt, um die Systemfiles auf eine andere formatierte Diskette zu kopieren.

Wenn sich ERSOIO/BIN und MSDOS.SYS bereits auf der Zieldiskette befinden, müssen Sie genausoviel Platz auf der Diskette belegen, wie das neue System belegen wird.

Die Zieldiskette muß leer sein bzw. darf nur die Files ERSOIO/BIN und MSDOS/SYS enthalten.

ERSOIO/BIN und MSDOS/SYS sind versteckte Files, die nicht mit DIR gelistet werden können. Der Kommandoprozessor COMMAND.COM, sowie DIOS.COM werden nicht mit SYS übertragen. Diese müssen mit COPY übertragen werden.

Tritt ein Fehler auf, wird eine der folgenden Fehlermeldungen ausgegeben:

No room for system on destination disk

Auf der Zieldiskette ist nicht genug Platz für die Systemfiles vorhanden.

Incompatible system size

Die Systemfiles auf Quell- und Zieldiskette unterscheiden sich in ihrer Größe.

Befehl: TIME

Typ: Intern

Funktion: Anzeigen und Stellen der Zeit

Format: TIME #<SS>#:<MM>##

Kommentar: Wird TIME ohne zusätzliche Angaben eingegeben, erscheint die aktuelle Uhrzeit und es wird die Eingabe einer neuen Uhrzeit verlangt.

Beispiel: Current time is 14:51:53.03
Enter new time:

Soll keine neue Zeit eingegeben werden, drücken Sie die <ENTER>-Taste.

Die neue Zeit kann auch direkt eingegeben werden. Es gilt z.B.:

TIME 8:20

Als Zeit dürfen nur Ziffern eingegeben werden, keine Buchstaben.

<SS> ist erlaubt von 00-24

<MM> ist erlaubt von 00-59

Stunden und Minuten müssen durch Doppelpunkte getrennt werden. Sekunden bzw. Hundertstelsekunden müssen nicht eingegeben werden.

Ist die Eingabe fehlerhaft, er-
scheint:

Invalid time
Enter new time:

DOS wartet jetzt auf die Eingabe
einer gültigen Zeit.

Befehl: TYPE

Typ: Intern

Funktion: Zeigt den Inhalt eines Files auf dem Schirm an.

Format: TYPE #pathname# <filespec>

Kommentar: Die einzige Anzeigeformatierung, die TYPE vornimmt ist, daß Tabulatorsteuerzeichen (CHR\$(9)) Leerzeichen erzeugen (Stop bei jeder 8 Spalte). Es ist zu beachten, daß Binärfiles (.BIN, .COM oder .EXE) Sonderzeichen enthalten, es können also Seitenlöschcodes, Bell-Zeichen oder Escapesequenzen die Bildschirmformatierung stören.

Befehl: VER

Typ: Intern

Funktion: Gibt die Versionsnummer des MS-DOS an.

Format: VER

Befehl: VERIFY

Sinn: Gibt an, ob bei Schreiboperationen auf Diskette eine Überprüfung stattfinden soll.

Funktion: VERIFY #ON|OFF#

Kommentar: Der Sinn dieses Befehls ist derselbe, wie die /V Option bei COPY.

Es wird dem DOS mitgeteilt, daß jeder geschriebene Datensatz nach dem Schreiben zu Überprüfen ist. Fehlermeldungen werden nur ausgegeben, wenn der Inhalt des zurückgelesenen Datensatzes nicht mit dem Inhalt des geschriebenen Datensatzes übereinstimmt.

VERIFY ON bleibt aktiv bis es durch VERIFY OFF ausgeschaltet wird. Durch Eingabe von VERIFY ohne Parameter wird der aktuelle Stand ausgegeben.

Befehl: VOL

Typ: Intern

Funktion: Zeigt den Diskettenamen an.

Format: VOL #d:#

Kommentar: Es wird der Name der Diskette im angegebenen Laufwerk ausgegeben. Wird kein Laufwerksbuchstabe angegeben, so wird der Name der Diskette im aktiven Laufwerk ausgegeben. Hat die Diskette keinen Namen, so wird ausgegeben:

Volume in drive x has no label

8. Batchverarbeitungsbefehle

=====

Die nun folgenden Befehle ECHO, FOR, GOTO, IF und SHIFT sind nur in Batchfiles zu benutzen und machen diese wesentlich leistungsstärker.

Befehl: ECHO

Typ: Intern

Funktion: Schaltet die ECHO-Funktion ein und aus.

Format: ECHO #ON OFF text#

Kommentar: Normalerweise werden die Befehle in einem Batchfile auf dem Bildschirm angezeigt, wenn der Kommandoprozessor sie bearbeitet. Diese Option läßt sich mit ECHO OFF ausschalten. ECHO ON aktiviert die Ausgabe wieder. Wird ECHO ohne Optionen eingegeben, wird der augenblickliche Stand (ON oder OFF) ausgegeben. ECHO <text> zeigt den Text an.

Befehl: FOR

Typ: Intern

Funktion: Befehlserweiterung zum Gebrauch in Batch- und interaktiver Fileverarbeitung.

Format: FOR %%<c> IN <set> DO <Befehl>

- Zur Batchverarbeitung

FOR %<c> IN <set> DO <Befehl>

- Zur interaktiven Verarbeitung

Kommentar: <c> kann jedes beliebige Zeichen außer den Ziffern 0 bis 9 sein, um Verwechslungen mit den Batchparametern %0 bis %9 zu vermeiden.

<set> ist eine Menge von Ausdrücken beliebiger Länge, die durch Leerzeichen getrennt sind. Diese Ausdrücke müssen in Klammern eingeschlossen sein. Ein solcher Ausdruck darf auch die Wildcards '?' und '*' enthalten, in diesem Falle werden alle passenden Filenamen auf der Diskette Elemente des Sets. Alle Ausdrücke, die nach einem solchen Wildcard-Ausdruck folgen, werden nicht bearbeitet!

Beispiele für <set>:

(* .ASM)

Alle .ASM-Files auf der aktiven Diskette.

(TEST.ASM BRIEF.TXT FLUSIM.GAM)

Die drei angegebenen Files.

Der Variablen %%<c> werden nun nacheinander die Elemente des Sets zugeordnet. Ein Kommando hinter DO, das diese Variable enthält, wird in der Art ausgeführt, als ob statt der Variablen dort der entsprechende Name stände.

```
FOR %%f IN (*.ASM) DO TYPE %%f
```

Dieser Befehle gibt nacheinander den Inhalt aller auf der aktiven Diskette zu findenden .ASM-Files aus.

```
FOR %%e IN (T.TXT M.BRF) DO BATCH %%e
```

Führt das Batchfile BATCH.BAT mit den Parametern T.TXT und M.BRF aus.

Soll eine FOR-Konstruktion in einem Batchfile angewandt werden, müssen alle Variablen mit '%%' beginnen, da '%f' beispielsweise als Fehler erkannt würde (Nach '%' dürfen als Variablen in Batchfiles nur Ziffern folgen). '%%f' hingegen wird richtig erkannt.

Benutzen Sie FOR als normalen DOS-Befehl, geben Sie nur ein '%' an, da dann '%0' bis '%9' ohnehin nicht erlaubt ist.

Befehl: GOTO

Typ: Intern

Funktion: Sprungbefehl für die Batchverarbeitung.

Format: GOTO <label>

Kommentar: GOTO <label> bewirkt einen Sprung zu einer bestimmten Zeile im Batchfile. Wird der entsprechende Label nicht gefunden, wird die Ausführung des Batchfiles abgebrochen.

```
:foo  
REM in der Schleife .....  
GOTO foo
```

Dieses Batchfile bewirkt den laufenden Ausdruck von :
REM in der Schleife

Ein Label wird definiert, indem man vor den Namen des Labels einen Doppelpunkt schreibt. Der Label wird dann vom Kommandoprozessor ignoriert.

Befehl: IF

Typ: Intern

Funktion: Vergleichsbefehl bei der Batch-
verarbeitung

Format: IF <Bedingung> <Befehl>

Kommentar: Die <Bedingung> kann eine der
folgenden sein:

ERRORLEVEL <Zahl>

Nur dann wahr, wenn das vorher
von COMMAND ausgeführte Programm
einen 'Exit code' von <Zahl> oder
mehr hatte.

<string1> == <string2>

Nur wahr, wenn <string1> und
<string2> exakt identisch sind.

EXIST <Filespec>

Nur dann wahr, wenn der angegebene
File existiert.

NOT <Bedingung>

Genau dann wahr, wenn die Bedin-
gung nicht erfüllt ist.

Beispiel: IF NOT EXIST test.txt ECHO Can't find file
IF NOT ERRORLEVEL 3 LINK test.rel

Befehl: PAUSE

Typ: Intern

Funktion: Unterbricht die Ausführung eines Batchfiles.

Format: PAUSE #Kommentar#

Kommentar: Es kann notwendig sein, die Batch-Ausführung zu unterbrechen, um z.B. die Diskette zu wechseln o.ä.
Trifft der Kommandoprozessor auf den Befehl PAUSE, so wird auf dem Schirm ausgegeben:

'Strike a key when ready'

Drücken Sie nun <Ctrl><Break>, fragt das System:

'Abort batch job (Y/N)?'

Geben Sie nun 'Y' ein, wird die Ausführung des Batchfiles abgebrochen.

Drücken Sie nach 'Strike a key when ready' eine andere Taste als <Ctrl><Break>, wird die Ausführung des Batchfiles fortgesetzt.

Die <Kommentar> Option gibt Ihnen die Möglichkeit, vor der 'Strike ..' Meldung noch einen eigenen Text einzufügen.

Befehl: SHIFT

Typ: Intern

Funktion: Erlaubt die Benutzung von mehr als 10 ersetzbaren Parametern bei der Batchverarbeitung.

Format: SHIFT

Kommentar: Im Normalfall können Sie in einem Batchfile maximal 9 ersetzbare Parameter benutzen (%0 enthält immer den Namen des Batchfiles). Es kann jedoch der Fall auftreten, das dies nicht ausreicht. Gehen wir von folgenden Beispiel aus:

Ihr Batchfile lautet:

```
MODE COM1:%1,%2,%3,%4,%5
MODE COM2:%6,%7,%8,%9...
```

Sie möchten mit diesem Batchfile zwei serielle Schnittstellen konfigurieren. Da jeder MODE COM-Befehl bis zu 5 Parameter haben kann, kommen Sie mit 9 Batchparametern nicht aus. Hier hilft der SHIFT-Befehl. Bei seiner Ausführung wird der Inhalt der einzelnen Parameter verschoben. %0 enthält nun die Bedeutung von %1 usw.

Auf diese Art und Weise wird %9
frei, kann also mit dem elften
Parameter aufgefüllt werden. Ihr
Batchfile müßte also so aussehen:

```
MODE COM1:%1,%2,%3,%4,%5  
SHIFT  
MODE COM2:%5,%6,%7,%8,%9
```

Geben Sie nun ein:

```
SETCOM 300,0,8,2,P,1200,N,5,1,P
```

werden die seriellen Schnittstel-
len konfiguriert.

9. DOS Editier- und Funktionstasten

=====
Die DOS-Editiertasten werden benutzt, um Verbesserungen innerhalb einer Zeile zu machen. Um größere Änderungen in einem File zu machen, muß der Zeileneditor EDLIN angewendet werden.

Es ist zu bemerken, daß die meisten Anwenderprogramme sich eigene Editierregeln aufstellen, und daß dadurch die unten beschriebenen Regeln ungültig werden. Auch ist es möglich, sich in BASIC eigene Editierregeln zu definieren.

Das DOS schreibt die eingegebene Zeile zuerst in einen Eingabepuffer. Dieser Puffer wird dem DOS durch Druck auf die <ENTER>-Taste zugänglich gemacht. Die Zeile bleibt jedoch trotzdem im Eingabepuffer und kann so als Schablone für eine neue Eingabe benutzt werden.

Die Benutzung dieser Schablone in Verbindung mit den DOS-Editiertasten bieten sich folgende Vorteile:

- Eine Befehlszeile kann unendlich oft wiederholt werden indem zwei Tasten gedrückt werden.
- Haben Sie einen Fehler bei der Eingabe einer Befehlszeile gemacht, können Sie diesen verbessern, ohne die gesamte Zeile neu eintippen zu müssen.

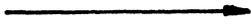
- Eine Befehlszeile, die der Schablone ähnelt, kann mit wenigen Tastendrücken verändert werden. Eine völlig neue Eingabe ist nicht nötig.

Das Verhältnis zwischen Befehlszeile und Schablone läßt sich folgendermaßen beschreiben:

Benutzereingabe



Befehlszeile



Schablone



Kommandoprozessor

Bei Eingabe eines Befehls gelangt dieser in den Eingabepuffer. Durch Drücken von <ENTER> wird dieser Befehl an den Kommandoprozessor weitergeleitet. Gleichzeitig gelangt jedoch noch eine Kopie des Befehls zur Schablone. Von dort aus kann der Befehl wieder abgerufen bzw. editiert werden.

Die folgende Tabelle enthält eine vollständige Liste der besonderen Editiertasten.

Taste -----	Editierfunktion -----
<F1> bzw. <→>	Kopiert einen Buchstaben von der Schablone in die Kommandozeile
<F2>	Kopiert bis zum angegebenen Zeichen von der Schablone in die Kommandozeile
<F3>	Kopiert alle verbleibenden Zeichen von der Schablone in die Kommandozeile
	Überspringt (kopiert nicht) das nächste Zeichen in der Schablone.
<F4>	Überspringt (kopiert nicht) die Buchstaben in der Schablone bis zum angegebenen Zeichen.
<ESC>	Entwertet die aktuelle Eingabe. Verändert nicht die Schablone.
<INS>	An- und Ausschalten des Insert-Modus.
<F5>	Bricht die Eingabe ab, übernimmt das bisher getippte aber in die Schablone.
<F6>	Setzt ein CTRL-Z (1AH) an das Ende der Eingabe (EOF).

Wird z.B. der Befehl:

```
DIR PROG.COM
```

eingegeben, zeigt das DOS die Information über den File PROG.COM an. Der Befehl wird auch als Schablone gespeichert. Um den Befehl erneut auszuführen, drücken Sie <F3> und <ENTER>.

Um aus dem Text PROG.COM PROG.ASM zu machen, drücken Sie die Tasten <F2> und <C>. Dies kopiert den Inhalt der Schablone bis zum ersten Auftauchen des Zeichens C in die Eingabezeile. Es erscheint also:

```
DIR PROG.
```

Tippen Sie jetzt ASM <ENTER>. Es steht auf dem Schirm :

```
DIR PROG.ASM
```

Die Befehlszeile wird nun neu in die Schablone übernommen und kann an den Kommandoprozessor weitergegeben werden. Soll als Befehl eingegeben werden:

```
TYPE PROG.ASM
```

Drücken Sie TYPE<Ins><Leertaste><F3><ENTER>.

CTRL-Zeichen Kombinationen:

Eine CTRL-Zeichen Tastenkombination beeinflusst die Kommandozeile. Sie kennen bereits die Funktionen <Ctrl>-<Break> etc.

Wichtig!

Wenn Sie eine CTRL-Zeichen Kombination eingeben, so halten Sie die Taste <Ctrl> fest und drücken dann die entsprechende Taste.

CTRL-Zeichen -----	Funktion -----
<Ctrl>-<Break>	Bricht die Ausführung des aktuellen Befehls ab.
<Ctrl>-<H> bzw. <←>	Bewegt den Cursor um eine Position nach links und löscht den letzten Buchstaben.
<Ctrl>-<ENTER>	Einsetzen eines Zeilenendzeichens. Die Kommandozeile wird nicht gelöscht.
<Ctrl>-<P> oder	Leitet Bildschirmausgabe auf den <Ctrl>-<PrtSc> Druckertreiber um.
<Ctrl>-<NumLock>	Unterbricht die Bildschirmausgabe. Durch drücken einer beliebigen Taste wird mit der Ausführung fortgefahren.
<Esc>	Löscht die aktuelle Zeile, löscht die Kommandozeile. Gibt einen umgekehrten Schrägstrich (\), einen Wagenrücklauf (CR) und einen Zeilenvorschub (LF) aus. Die Schablone wird nicht geändert.

10. Der Zeileneditor (EDLIN)

=====

EDLIN ist der zum Eingeben und Ändern von Files benötigte Zeileneditor.

EDLIN kann benutzt werden, um:

- Neue Quellfiles zu erzeugen und abzuspeichern.
- Alte Files zu verändern
- Zeilen zu löschen, zu verändern und anzuzeigen.
- Text innerhalb von Files zu suchen und auszutauschen.

Der von EDLIN erzeugte oder bearbeitete Text wird in Zeilen unterteilt. Die Zeilennummern dienen nur als Hilfe und werden nicht mit abgespeichert.

Werden Zeilen eingefügt, rückt der nachfolgende Text automatisch um die entsprechende Zeilenzahl auf. Die Zeilen sind also automatisch immer kontinuierlich durchnummeriert. Ebenso verhält es sich beim Löschen von Zeilen.

Starten von EDLIN:

Um EDLIN zu starten, geben Sie ein:

EDLIN <filename>

Wird der angegebene Filename nicht auf der Diskette gefunden, so wird er von EDLIN neu erzeugt. EDLIN antwortet dann mit der Meldung:

New File

*

Es können jetzt Textzeilen eingegeben werden. Dazu geben Sie ein 'I' ein. Der I-Befehl wird später in diesem Kapitel besprochen. Um ein bestehendes File zu editieren, muß der Filename des Files eingegeben werden. Dieser File wird dann von EDLIN in den Arbeitspuffer geladen. Paßt der File vollständig in den Speicher, erscheint folgende Meldung:

End of Input File

*

Der File kann jetzt mit Hilfe der EDLIN-Editiertasten editiert werden.

Ist der File zu lang, um in einem Stück von der Diskette geladen zu werden, wird der Puffer zu 3/4 gefüllt. Dieser Teil des Files kann jetzt editiert werden. Um den restlichen Teil des Files zu editieren, müssen einige schon editierte Zeilen abgespeichert werden.

Lesen Sie dazu bei den WRITE und APPEND-Befehlen in diesem Kapitel nach.

Ist Ihre Arbeit beendet, speichern Sie mit dem END-Befehl den veränderten File ab.

Der ursprüngliche File wird mit der Erweiterung .BAK versehen. Der neue File bekommt die mit EDLIN angegebene Erweiterung. Der Originalfile wird nicht gelöscht, bis EDLIN beendet wird oder Platz auf der Diskette benötigt.

Es darf auf keinen Fall ein Filename mit der Erweiterung .BAK benutzt werden. Benutzen Sie gegebenenfalls zuvor den DOS-Befehl RENAME.

Die EDLIN-Tasten:

EDLIN-Taste <F1>

Funktion: Kopiert ein Zeichen von der
Zeile in die Kommandozeile.

Beispiel: Die letzte Zeile enthält
z.B.:

'Dies ist eine Befehlszeile'

Beim Prompt

l:*

erzeugt die mehrfache Betä-
tigung der <F1>-Taste :

l:*D
l:*Di
l:*Die
l:*Dies
etc.

EDLIN-Taste <F2>

Funktion: Kopiert die Zeile bis zu einem bestimmten Zeichen.

Beispiel: Die Zeile ist wieder:

'Dies ist eine Zeile'

Eingabe von <F2> und t
erzeugt den Ausdruck:

l:*Dies is

Es wird also nur bis zum angegebenen Zeichen kopiert, das Zeichen selbst wird nicht kopiert.

EDLIN-Taste <F3>

Funktion: Kopiert alle verbleibenden
Buchstaben aus der Zeile.

Beispiel: Die Zeile lautet:

'Dies ist eine Zeile'

Druck auf <F3> ergibt nun:

l:*Dies ist eine Zeile

Wird z.B. vor F3 noch
gedrückt, steht auf dem
Schirm:

l:*ies ist eine Zeile

EDLIN-Taste: <F4>

Funktion: Überspringt mehrere Zeichen
bis zum angegebenen Zeichen.

Beispiel: Die Zeile ist:

'Dies ist eine Zeile'

Die Tastenfolge <F4> t <F3>
erzeugt:

l:*t eine Zeile

Das angegebene Zeichen wird
also nicht übersprungen.

EDLIN Taste <Esc>

Funktion: Löscht die aktuelle Zeile

Beispiel: Die Zeile enthält:

'Dies ist eine Zeile'

Sie geben jetzt ein:

'Neue Zeile'

So steht auf dem Bildschirm:

l:*Dies ist eine Zeile

l:*Neue Zeile

Soll jetzt wieder die alte
Zeile editiert werden, tip-
pen Sie <Esc>

Es erscheint:

l:*

Ein Druck auf <F3> läßt
erscheinen:

l:*Dies ist eine Zeile

EDLIN-Taste: <F5>

Funktion: Neue Schablone übernehmen

Beispiel: Die alte Zeile ist:

'Dies ist eine Zeile'

Wird jetzt

'Neue Zeile'

eingegeben und anschließend
<F5> gedrückt, so zeigt
der Schirm:

l:*Neue Zeile@

Das @ am Ende der Zeile
bedeutet, daß diese Zeile
als neue Schablone übernom-
men wurde.

Befehlsinformationen:

1. Es dürfen Pathnamen mit EDLIN angegeben werden. Es ist also zulässig:

EDLIN <Pathname><Filename>

2. Zeilen können auch relativ aufgerufen werden. +10 bezieht sich z.B. auf eine Zeile, die sich 10 Zeilen vor der aktuellen Zeile befindet. -10 bezieht sich auf eine Zeile, die zehn Zeilen zurück liegt.
3. Es können mehrere Befehle gleichzeitig gegeben werden, wenn diese durch ein Semikolon getrennt sind. Im Falle eines SEARCH-Befehls kann <F6> verwendet werden, um das Ende des zu suchenden Textes zu kennzeichnen.
4. EDLIN-Befehle können mit oder ohne Leerzeichen eingegeben werden. Der Befehl 6D gleicht also dem Befehl 6 D.
5. <Ctrl-Zeichen> können in den Text eingefügt werden, indem man vor dem Zeichen ein <Ctrl><V> eingibt. <Ctrl><V> weist den Computer an, das nächste Zeichen als <Ctrl>-Zeichen zu erkennen.

Beispiel:

S<Ctrl-V>Z

sucht nach dem ersten Vorkommen von <Ctrl-Z> in einem File.
<Ctrl-V> kann in einen Text durch Eingabe von <Ctrl-V>V eingefügt werden.

6. Die Taste <F6> sagt EDLIN normalerweise "Hier ist das EOF" (End Of File).
Haben Sie <Ctrl-Z> auch anderswo in Ihrem File, müssen Sie die /B Option benutzen, um EDLIN anzugeben, daß es sich hierbei nicht um EOF's handelt.

Zusammenfassung der EDLIN-Befehle:

In der folgenden Tabelle sind die EDLIN-Befehle kurz aufgeführt. Auf den folgenden Seiten werden diese dann näher beschrieben.

Befehl	Sinn
-----	----
<n>	Editiert Zeile Nr. n
A	Weitere Zeilen von Disk laden
C	Kopiert Zeilen
D	Löscht Zeilen
E	Beendet Editierung
I	Fügt Zeilen ein
L	Listet Text
M	Verschiebt Zeilen
P	Listet seitenweise
Q	Beendet Editierung
R	Ersetzt Zeilen
S	Sucht Text
T	Überträgt Text

EDLIN Befehlsoptionen:

Einige EDLIN-Befehle erlauben die Eingabe einer oder mehrerer Optionen. Der Effekt der Befehlsoptionen variiert von Befehl zu Befehl.

Folgende Liste beschreibt die Optionen:

<n> <n> gibt eine Zeilennummer an. Zeilennummern werden durch Komma-ta von anderen Zeilennummern, Befehlen und Optionen getrennt. <n> kann auf drei verschiedene Arten angegeben werden:

Zahl Jede beliebige ganze Zahl unter 65534. Wird eine Zeile angegeben, die hinter der letzten existierenden Zeile liegt, wird die erste Textzeile angesprochen.

. Wird ein Punkt für <n> angegeben, so ist die aktuelle Zeile angesprochen. Diese ist die letzte editierte Zeile, nicht jedoch unbedingt die letzte geschriebene Zeile. Die aktuelle Zeile ist durch einen Stern (*) zwischen der Zeilennummer und dem ersten Buchstaben gekennzeichnet.

£ Das Pfund-Zeichen ruft die nächste Zeile auf. Der Effekt ist derselbe, als wenn Sie die aktuelle Zeilennummer + 1 angeben.

<ENTER> ohne eine der oben genannten Angaben, veranlaßt EDLIN, einen Standardwert, der je nach Befehl variiert, anzunehmen.

? Das Fragezeichen veranlaßt den Computer, nachzufragen, ob der korrekte String gefunden wurde. Das Fragezeichen kann nur in Verbindung mit den R und S Befehlen eingesetzt werden.

<string> Repräsentiert einen Text, der gefunden, ersetzt oder eingesetzt werden soll. Die <string>-Option tritt nur mit den R und S-Befehlen in Aktion. Jeder String muß entweder mit einem <Ctrl-Z> oder einem <ENTER> beendet werden. Es dürfen keine Leerzeichen zwischen einem String und dessen Befehlszeichen (R oder S) stehen, da diese sonst bei der Suche mitzählen.

EDLIN-Editierbefehle:

Die folgenden Seiten beschreiben ausführlich die EDLIN-Editierbefehle.

EDLIN: (A)nhängen

Funktion: Hängt eine angegebene Anzahl Zeilen von der Diskette an den File im Speicher an.

Format: #<n>#A

Kommentar: Der Befehl ist nur sinnvoll, wenn die Länge des Files die Größe des zur Verfügung stehenden Speichers überschreitet. EDLIN wird so viele Zeilen in den Speicher lesen, bis dieser zu 3/4 voll ist. Die Zeilen können dann bearbeitet werden. Schon bearbeitete Zeilen lassen sich abgespeichern und an ihrer Stelle können neue Zeilen mit dem A-Befehl in den Speicher geladen werden.

- Bemerkung:**
1. Ohne Angabe einer Zahl werden so viele Zeilen in den Speicher gelesen, bis er zu 3/4 voll ist. Sollte er schon zu 3/4 voll sein, hat der A-Befehl keinen Effekt.
 2. Die Meldung 'End of Input File' wird angezeigt, wenn EDLIN die letzte Zeile in den Speicher gelesen hat.

EDLIN: (C)opy

Funktion: Kopiert einen bestimmten Bereich von Zeilen an eine andere Stelle.

Format: #<n>#, #<n>#, <n>, #<Zähler>#C

Kommentar: Wird bei der Option <Zähler> eine Angabe gemacht, wird der Zeilenbereich einmal kopiert. Wird die erste oder die zweite Option <n> weggelassen, wird die aktuelle Zeile angenommen. Die Zeilennummern dürfen sich auf keinen Fall überlappen, in diesem Fall wird ein 'ENTRY ERROR' ausgegeben. Der folgende Befehl würde also zu einer Fehlermeldung führen:

3,20,15C

Die Zeilennummern werden nach dem C-Befehl neu geordnet.
Setzen wir voraus, daß folgender File existiert und zur Editierung bereit ist:

- 1: Dieser File ist
- 2: zum Editieren bereit
- 3: im EDLIN

Der Befehl 1,3,4C würde den File folgendermaßen verändern:

- 1: Dieser File ist
- 2: zum Editieren bereit
- 3: im EDLIN
- 4: Dieser File ist
- 5: zum Editieren bereit
- 6: im EDLIN

Die drei <n> bedeuten also VON, BIS und NACH. Es wird angegeben, wo der zu kopierende Bereich beginnt, endet und wohin er kopiert werden soll.

EDLIN: (D)elete

Funktion: Löscht einen angegebenen Bereich von Zeilen in einem File.

Format: #<n>#, #<n>#D

Kommentar: Die <n>'s bedeuten: VON und BIS. wird die VON-Option weggelassen, so wird die aktuelle Zeile angenommen. Wird die BIS-Option weggelassen, so wird nur eine Zeile gelöscht. Wurden Zeilen gelöscht, so wird die erste Zeile nach dem gelöschten Bereich die aktuelle Zeile und bekommt die Zeilennummer der ersten Zeile des gelöschten Bereiches.

Beispiel: Folgender File existiert und ist bereit zum Editieren:

```
1:*Dies ist Zeile 1
2: Dies ist Zeile 2
3: Dies ist Zeile 3
4: Dies ist Zeile 4
```

Der Befehl 2,3D erzeugt jetzt folgenden File:

```
1: Dies ist Zeile 1
2:*Dies ist Zeile 4
```

Die Zeilennummern werden automatisch geordnet.

EDLIN: <n>

Funktion: Editiert eine bestimmte Zeile.

Format: #<n>#

Kommentar: Um eine Zeile zu editieren, wird einfach deren Zeilennummer eingegeben. EDLIN gibt nun diese Zeilennummer, gefolgt vom Cursor, aus. Sie können diese Zeile nun editieren. Der existierende Text kann als Schablone benutzt werden.

Wird keine Zeilennummer angegeben, also nur die <ENTER>-Taste gedrückt, so wird die Zeile hinter der aktuellen Zeile editiert. Soll diese in ihrem ursprünglichen Zustand bleiben, so ist erneut <ENTER> zu drücken.

Warnung: Wird die <ENTER>-Taste gedrückt, wenn sich der Cursor in der Mitte der Zeile befindet, so wird automatisch der Rest der Zeile gelöscht.

EDLIN: (E)nde

Funktion: Beendet die Editierung

Format: E

Kommentar: Dieser Befehl speichert den editierten File auf Diskette und versieht den Quellfile mit der Erweiterung BAK. Wurde der File von EDLIN neu erzeugt, so wird kein .BAK-File erzeugt. Anschließend erfolgt ein Rücksprung ins DOS.

Der E-Befehl akzeptiert keine Optionen. Es ist also auch nicht möglich, EDLIN zu sagen, auf welchem Laufwerk der Zielfile abgespeichert werden soll. Dieses muß bereits beim Aufruf von EDLIN erfolgen. Wurde keine Laufwerksangabe gemacht, so erfolgt die Abspeicherung auf dem aktuellen Laufwerk. Es ist jedoch möglich, den File nachträglich mit COPY auf eine andere Diskette zu übertragen.

Es muß auf jeden Fall genügend freier Platz auf der Zieldiskette sein. Sollte der Platz nicht ausreichen, so wird das Schreiben abgebrochen und der File geht verloren. Es ist jedoch möglich, daß der File schon teilweise auf Diskette geschrieben wurde.

EDLIN: e(I)nfügen

Funktion: Fügt Text direkt vor der angegebenen Zeile ein.

Format: #<n>#I

Kommentar: Der I-Befehl muß angewendet werden, wenn ein neuer Text erzeugt werden soll. Die erste Zeilennummer ist 1, alle folgenden werden durch Druck von <ENTER> aufgerufen.

EDLIN verbleibt im INSERT-Modus, bis <Ctrl><Break> gedrückt wird. Nach Beendigung des Einfügens wird die Zeile nach dem eingefügten Block zur aktuellen Zeile. Alle folgenden Zeilennummern werden um die Zahl der eingefügten Zeilen erhöht.

Wird <n> nicht angegeben, so wird die aktuelle Zeile angenommen, es wird also vor der aktuellen Zeile eingefügt. Ist <n> größer als die Zeilennummer der letzten Zeile, so werden die Zeilen an das Ende des Files angehängt. In diesem Fall wird die letzte eingefügte Zeile zur aktuellen Zeile.

Beispiel: Angenommen folgender File existiert und befindet sich im Speicher:

```
1: Dies ist Zeile 1
2: Dies ist Zeile 2
3: Dies ist Zeile 3
4: Dies ist Zeile 4
5: Dies ist Zeile 5
```

Wird nun der Befehl 3I ausgeführt, erscheint:

3:

Geben Sie nun ein:

3: Dieser Text wurde eingefügt

Durch Druck auf <ENTER> wird die Zeile übernommen und mit <Ctrl><Break> der Editiermodus verlassen.

Tippen Sie nun L <ENTER>, um den File zu listen. Es erscheint:

```
1:Dies ist Zeile 1
2:Dies ist Zeile 2
3:Dieser Text wurde eingefügt
4:Dies ist Zeile 3
5:Dies ist Zeile 4
6:Dies ist Zeile 5
```

EDLIN: (L)ist

Funktion: Listet einen Zeilenblock.

Format: #<n>##,<n>#L

Kommentar: Die Standardwerte werden angenommen, wenn entweder eine oder beide Optionen nicht angegeben werden. Wird z.B. die erste Option nicht angegeben, wird also

,<n>L

eingegeben, so beginnt die Ausgabe ll Zeilen vor der aktuellen Zeile und endet bei der angegebenen Zeile. Das führende Komma ist notwendig, um anzuzeigen, daß es sich um die zweite Option handelt.

Hinweis: Ist bei einer Eingabe in dieser Form die angegebene Zeile mehr als ll Zeilen vor der aktuellen Zeile, so erfolgt die Ausgabe, als ob beide Optionen weggelassen wurden.

Lassen Sie die zweite Option weg, geben also ein:

<n>L

so werden 23 Zeilen ab der angegebenen Zeile ausgegeben.

Werden beide Optionen weggelassen, d.h., es wird eingegeben:

L

wird der Text ab der 11. Zeile vor bis zur 11. Zeile nach der aktuellen Zeile ausgegeben. Sollten sich vor der aktuellen Zeile weniger als 11 Zeilen befinden, so wird entsprechend mehr hinter der aktuellen Zeile ausgegeben. In jedem Fall werden 23 Zeilen dargestellt.

Werden beide Optionen angegeben, so wird VON, BIS gelistet. Die erste Option besagt also, ab wo, die zweite Option sagt, bis wo gelistet werden soll.

Beispiel: Es existiere folgender File:

```
1: Dies ist Zeile 1
2: Dies ist Zeile 2
3: Dies ist Zeile 3
4: Dies ist Zeile 4
5: Dies ist Zeile 5
6: Dies ist Zeile 6
```

Die Eingabe von 2,5L gibt aus:

```
2: Dies ist Zeile 2
3: Dies ist Zeile 3
4: Dies ist Zeile 4
5: Dies ist Zeile 5
```

EDLIN: (M)ove

Funktion: Verschiebt einen Zeilenblock.

Format: #<n>#,<n>,<n>M

Kommentar: Der Befehl M verschiebt VON, BIS, NACH.

Die erste Option gibt also an, wo der Block startet, die zweite, wo er endet und die dritte, wohin er verschoben werden soll.

20,30,100M

verschiebt also

VON 20
BIS 30
NACH 100

Wird die erste Option weggelassen, so wird ab der aktuellen Zeile verschoben:

,+25,100M

verschiebt VON der aktuellen Zeile BIS zur aktuellen Zeile +25
NACH Zeile 100

EDLIN: (P)age

Funktion: Listet Zeilen seitenweise.

Format: #<n>##,<n>#P

Kommentar: Wird die erste Option weggelassen, so wird die aktuelle Zeile vorausgesetzt. Ohne zweite Zeile werden 23 Zeilen gelistet. Die letzte gelistete Zeile wird zur aktuellen Zeile und markiert mit einem Stern (*). Das Format ist also VON, BIS. Die erste Option gibt an, ab wo gelistet wird, die zweite Option gibt an, bis wo gelistet wird.

EDLIN: (Q)uit

Funktion: Beendet das Editieren, speichert den File NICHT ab und kehrt zum DOS zurück.

Format: Q

Kommentar: EDLIN fragt Sie sicherheitshalber, ob Sie den File nicht abspeichern wollen.

Drücken Sie 'Y', wenn Sie keine Abspeicherung wollen und 'N', wenn Sie weitereditieren wollen.

Wichtig: Wird EDLIN gestartet, löscht es jede .BAK-Version des Files. Verlassen Sie EDLIN also mit Q, so existieren keine .BAK-Versionen mehr auf der Diskette.

EDLIN: (R)eplace

Funktion: Ersetzt einen bestimmten String durch einen anderen.

Format: #<n>##,<n>##?#R<string1><F6><string2>

Kommentar: Jeder im Text vorkommende <string1> wird durch <string2> ersetzt. Jede Zeile, in der eine Änderung vorgenommen wird, wird angezeigt.
Wird kein <string1> angegeben, so wird der <string1> des letzten R-Befehls verwendet. Wurde 'R' zum erstenmal aufgerufen, so wird der Befehl sofort abgebrochen. Wird <string2> weggelassen, kann <string1> mit <ENTER> beendet werden.
Die <n> Optionen beziehen sich auf VON, BIS. Die Optionen geben also den Bereich an, in dem gesucht werden soll.
Wird die erste Option weggelassen, so wird ab der Zeile nach der aktuellen Zeile nach <string1> gesucht, wird die zweite Option weggelassen, so wird bis zum Ende des Files gesucht. Wird <string1> mit <F6><ENTER> beendet, und wird kein <string2> angegeben, so wird für <string2> der Leerstring angenommen.

Beispiel: R<string1><F6><ENTER>

löscht alle <string1>-Vorkommen
im File.

Verwendet man im Befehl die '?'-
Option, kommt es bei jedem gefun-
denen <string 1> zur Abfrage, ob
er durch <string2> ersetzt werden
soll.

Anwortet man bei der 'O.K.?'-
Frage mit Y, so wird <string1>
durch <string2> ersetzt. Wird
jedoch mit N geantwortet, bleibt
der alte <string1> erhalten.

Beispiel: Setzen wir voraus, daß folgender
File existiert:

1:Diese Zeile ist am Anfang
2:Computer sind nützlich
3:ich bin hier
4:Das war's

Der Befehl

lRich<F6>du

würde folgende Ausgabe erzeugen:

2:Computer sind nützldu
3:du bin hier

Dieser Befehl zeigt die Notwen-
digkeit nach einer einzelnen
Überprüfung der Veränderungen.
Der entsprechende Befehl ist:

l?Rich<F6>du

EDLIN: (S)earch

Funktion: Sucht in einem Zeilenblock nach einem String.

Format: #<n># #,<n># #?# S<string><ENTER>

Kommentar: Die <n> Optionen bedeuten VON, BIS und geben den Bereich des Zeilenblocks an. Wird VON weggelassen, so sucht der Rechner ab der Zeile nach der aktuellen Zeile. Wird NACH weggelassen, so sucht der Rechner bis zum Ende des Files. Wird <string> weggelassen, so nimmt der Rechner den alten String. Liegt kein alter String vor, wurde S also zum ersten Mal aufgerufen, so wird die Funktion sofort beendet.

Wird die '?'-Option benutzt, so fragt der Rechner bei jedem Vorkommen des Strings 'O.K.'. Wird 'Y' eingegeben, so wird die Suche beendet und die Zeile, in der der String gefunden wurde, wird zur aktuellen Zeile. Wird jedoch 'N' eingegeben, so wird weitergesucht.

Beispiel: 1: Dies ist eine Zeile
2: TCS GENIE 16 C
3: Der Personalcomputer
4: Die vierte Zeile

Der obige File existiere und sei
zum editieren bereit.
Die Eingabe von

1?SDie

gibt aus

1: Dies ist eine Zeile
O.K.?

Drücken Sie jetzt 'N', erscheint:

4: Die vierte Zeile
O.K.?

EDLIN: (T)ransfer

Funktion: Fügt den Inhalt eines Files an der angegebenen Stelle ein.

Format: #<n>#T<Filename>

Kommentar: Es wird der Inhalt des angegebenen Files gelesen und ab der angegebenen Zeilennummer eingefügt. Die Zeilen werden anschließend neu geordnet.

EDLIN: (W)rite

Funktion: Schreibt eine Anzahl Zeilen vom Speicher auf die Diskette.

Format: #<n>#W

Kommentar: Dieser Befehl ist nur dann sinnvoll, wenn der zu editierende File größer als der zur Verfügung stehende Speicher ist. Wird EDLIN gestartet, lädt dieser den Speicher zu 3/4 voll. Um den Rest des Files zu editieren, muß der Anfang des Files wieder auf Diskette abgespeichert werden. Dies geschieht mit dem W-Befehl. Die <n> Option gibt an, wieviele Zeilen auf Diskette geschrieben werden sollen.

Wird <n> nicht angegeben, werden alle Zeilen auf Diskette geschrieben.

EDLIN-Fehlermeldungen:

Wenn EDLIN einen Fehler findet, so wird eine der folgenden Fehlermeldungen ausgegeben:

'Cannot edit .BAK file-rename file'

Ursache: Sie versuchen einen .BAK-File zu editieren. Diese Files können nicht editiert werden, da die BAK-Erweiterung für Sicherheitskopien vorgesehen ist.

Beheben: Sie können entweder den .BAK File mit einer anderen Erweiterung versehen oder ihn mit COPY duplizieren.

'No room in directory for file'

Ursache: Als Sie versucht haben, den File abzuspeichern, war entweder das Directory voll oder Sie haben einen illegalen Filenamen bzw. einen illegalen Laufwerksbuchstaben eingegeben.

Beheben: Überprüfen Sie die Befehlszeile, mit der EDLIN aufgerufen wurde, auf Fehler. Sollte diese sich nicht mehr auf dem Schirm befinden, so läßt sie sich mit <F3> erneut aufrufen. Sollten keine Fehler zu finden sein, so wenden Sie CHKDSK an. Wenn CHKDSK Ihnen mitteilt, daß das Directory voll ist, formatieren Sie eine neue Diskette und benutzen diese.

'Entry Error'

Ursache: Der letzte eingetippte Befehl enthielt einen Syntaxfehler.

Beheben: Geben Sie den Befehl erneut mit richtiger Syntax ein.

'Line too long'

Ursache: Während eines R-Befehls wurde ein String eingesetzt, der die Zeile auf über 253 Zeichen erweitert hat. Der R-Befehl wurde abgebrochen.

Beheben: Teilen Sie die Zeile in zwei Teile auf, und wenden Sie R zweimal an.

'Disk full- file write not completed'

Ursache: Es ist nicht genug Platz auf der Diskette, um den File abzuspeichern. EDLIN hat den Befehl abgebrochen und Sie an das Betriebssystem zurückgeleitet.

Beheben: Es wurde eventuell ein Teil des Files auf Diskette abgespeichert. Dieser Teil ist jedoch meistens wertlos. Gehen Sie beim nächsten Mal sicher, daß noch genügend Platz auf der Diskette vorhanden ist.

'Incorrect DOS version'

Ursache: Sie haben versucht, EDLIN auf einer älteren DOS Version als 2.11 zu starten.

Beheben: Sicherstellen, daß Sie mindestens mit Version 2.11 arbeiten.

'Filename must be specified'

Ursache: Sie haben beim Starten von EDLIN keinen bzw. einen ungültigen Filenamen angegeben.

Beheben: Geben Sie einen gültigen Filenamen an.

'Invalid Parameter'

Ursache: Es wurde versucht, eine andere Option als /B anzugeben, als EDLIN gestartet wurde.

Beheben: Geben Sie die /B Option an, wenn Sie EDLIN starten.

'Insufficient memory'

Ursache: Es ist nicht genügend Speicherplatz frei, um EDLIN zu starten.

Beheben: Wenn sich mehrere Programme im Speicher befinden, speichern oder löschen Sie diese.

'File not found'

Ursache: Der beim T-Befehl angegebene
Filename existiert nicht.

Beheben: Geben Sie den richtigen Filenamen
bei T ein.

'Must specify destination number'

Ursache: Es muß beim C- und M-Befehl eine
NACH-Zeilenummer angegeben werden.

Beheben: Geben Sie eine NACH-Zeilenummer
an.

'Not enough room to merge the entire file'

Ursache: Es liegt nicht genug Speicherplatz
vor, um den gesamten File der beim
T-Befehl angegeben wurde, zu laden.

Beheben: Schaffen Sie sich genügend Spei-
cherplatz, indem Sie Files, die
sich im Speicher befinden, löschen.

'File creation error'

Ursache: EDLIN hat Schwierigkeiten beim Er-
zeugen seines Zwischenspeicher-
Files.

Beheben: Überprüfen Sie, ob die Directory
genug Platz für den File aufweist.
Kontrollieren Sie auch, ob eventu-
ell eine Unterdirectory gleichen
Namens existiert.

11. File-Vergleichs-Programm (FC)

=====
Das Filevergleichsprogramm (FC) vergleicht den Inhalt von zwei Files. Die Unterschiede zwischen den Files können auf dem Schirm, dem Drucker oder in ein drittes File ausgegeben werden. FC kann benutzt werden, um Textfiles oder Binärfiles (Files, die vom Macroassembler, dem Linker etc. ausgegeben werden) zu vergleichen.

FC kann auf zwei verschiedene Arten vergleichen, auf einer Zeile für Zeile-Basis oder byteweise. Wird Zeile für Zeile verglichen, isoliert FC Zeilenblöcke, die unterschiedlich sind, und gibt diese aus. Wird byteweise verglichen, werden die unterschiedlichen Bytes ausgegeben.

Einschränkungen beim Filevergleich:

FC benutzt einen großen Teil des RAM Speicherbereichs als Zwischenspeicher für die Textfiles. Sollten die Textfiles zu groß sein, um in den Speicher zu passen, vergleicht FC soviel Text, wie in den Zwischenspeicher paßt. Sollte keine Übereinstimmung zu finden sein, gibt FC nur folgende Meldung aus:

*** files are different ***

Wird ein byteweiser Vergleich durchgeführt, verarbeitet FC Files beliebiger Länge, indem nur Teile der Files geladen, verglichen und wieder überschrieben werden.

Das FC-Format ist wie folgt:

```
FC #/n /B /W /C# <filename1> <filename2>
```

FC vergleicht den ersten File mit dem zweiten und berichtet über alle Übereinstimmungen. Beide Filenamen können natürlich auch Pathnamen enthalten.

FC-Optionen:

Es gibt vier Optionen, die mit FC angegeben werden können:

/B Erzwingt einen binären Filevergleich. Die Files werden Byte für Byte verglichen, ohne einen Versuch zur erneuten Synchronisation nach einem Fehler zu machen. Ein Unterschied zwischen den Files wird wie folgt kenntlich gemacht:

```
-- ADDR5 ----- F1 ----- F2 --  
      xxxxxxxx      yy          zz
```

Hierbei ist xxxxxxxx die relative Adresse des Bytes vom Fileanfang an gerechnet, yy und zz sind die Bytes in File1 und in File2. Sollte einer der Files länger als der andere sein, meldet FC einen Fehler. Ist z.B. File2 länger als File1, so meldet FC:

```
***Data left in F2***
```

/n n steht für eine Zahl zwischen 1 und 9. Diese Option gibt die Anzahl von Zeilen an, die übereinstimmen müssen, damit nach einem Fehler wieder Übereinstimmung erkannt wird. Wird diese Option nicht angegeben, so wird 3 angenommen. /n ist nur bei zeilenweisen Vergleichen sinnvoll.

/W läßt FC jede Folge von Leerzeichen, jedes TAB-Zeichen usw. durch ein einzelnes Leerzeichen ersetzen. So würde zum Beispiel:

"Dies ist eine Zeile"

nicht mit

"Dies ist eine Zeile"

Übereinstimmen, wenn die /W Option nicht angegeben wird.

/C Diese Option veranlaßt FC, Groß- und Kleinschreibung nicht zu beachten. So würde z.B.

"dies IST eine Zeile"

nicht mit

"DIES IST EINE ZEILE"

übereinstimmen, wenn die /C Option nicht angegeben wurde.

Wie FC Unterschiede anzeigt:

FC zeigt den Unterschied zwischen beiden Files folgendermaßen an:

```
...
...
-----<Filename1>
<Unterschied>
<Erste passende Zeile in File 1>
-----<Filename2>
<Unterschied>
<Erste passende Zeile in File 2>
-----
...
...
```

Auf diese Art und Weise wird jeder Unterschied der beiden Files angezeigt.

Treten zu viele Unterschiede auf, wird das Programm abgebrochen und es teilt mit, daß es sich um zwei verschiedene Files handelt.

Wird nach dem ersten Unterschied keine Übereinstimmung mehr gefunden, so teilt das Programm mit:

```
*** Files are different ***
```

und die Programmausführung wird abgebrochen.

Umleiten der FC-Ausgabe:

Die Unterschiede zwischen beiden Files werden auf dem Schirm ausgegeben. Es besteht jedoch auch die Möglichkeit, den Output auf ein File umzulenken. Dieses geschieht auf dieselbe Art und Weise, wie bei einem DOS-Befehl.

Um die Unterschiede von File1 und File2 in den File UNTER.TXT umzuleiten, geben Sie folgendes ein:

```
FC FILE1 FILE2 >UNTER.TXT
```

Beispiele für den Gebrauch von FC:

Beispiel 1:

Folgende beiden ASCII-Files befinden sich auf der Diskette:

ALPHA.TXT FILE1 -----	BETA.TXT FILE2 -----
A	A
B	B
C	C
D	G
E	H
F	I
G	J
H	1
I	2
M	P
N	Q
O	R
P	S
Q	T
R	U
S	V
T	4
U	5
V	W
W	X
X	Y
Y	Z
Z	

Um diese Files zu vergleichen und deren Unterschiede auf dem Bildschirm anzuzeigen, tippen Sie:

FC ALPHA.TXT BETA.TXT

FC vergleicht ALPHA.TXT mit BETA.TXT und zeigt alle Unterschiede auf dem Schirm an:

-----ALPHA.TXT
D Anmerkung: ALPHA enthält DEFG,
E BETA nur G
F
G

-----BETA.TXT
G

-----ALPHA.TXT
M Anmerkung: ALPHA enthält MNOP,
N BETA J12P
O
P

-----BETA.TXT
J
1
2
P

-----ALPHA.TXT
W Anmerkung: ALPHA enthält W,
-----BETA.TXT BETA 45W

4
5
W

Beispiel 2:

Dieses Beispiel erzwingt einen byteweisen Vergleich zwischen den Files.

Geben Sie ein:

```
FC /B ALPHA.TXT BETA.TXT
```

Die Bildschirmausgabe ist wie folgt:

```
--ADDRS----F1---F2--  
00000009    44    47  
0000000C    45    48  
0000000F    46    49  
00000012    47    4A  
00000015    48    31  
00000018    49    32  
0000001B    4D    50  
0000001E    4E    51  
00000021    4F    52  
00000024    50    53  
00000027    51    54  
0000002A    52    55  
0000002D    53    56  
00000030    54    34  
00000033    55    35  
00000036    56    57  
00000039    57    58  
0000003C    58    59  
0000003F    59    5A  
00000042    5A    1A
```

FC-Fehlermeldungen:

Wird von FC ein Fehler gefunden, so erscheint eine der folgenden Fehlermeldungen:

'Incorrect DOS version'

FC läuft nur unter DOS-Version 2.11 oder späteren.

'Invalid parameter:<Option>'

Eine der angegebenen Optionen ist ungültig.

'File not found:<Filename>'

FC hat den angegebenen Filenamen nicht gefunden.

'Read error in:<Filename>'

Während des Lesens von <Filename> trat ein Fehler auf, der File konnte deswegen nicht vollständig gelesen werden.

'Invalid number of parameters'

Sie haben zuviele oder nicht zusammenpassende FC-Optionen gewählt.

12. BASIC

=====

12.1 Für den Anfänger in BASIC

Dieses Kapitel ist eine Einführung in die Programmiersprache BASIC, mit der Sie Ihr GENIE 16 C programmieren können.

Es ist nicht als Lehrgang gedacht, sondern als eine Hilfe für diejenigen, die BASIC schon kennen. Falls Sie BASIC lernen wollen, benutzen Sie bitte einen der vielen BASIC Lehrgänge, die es gibt. Bei einem solchen Lehrgang werden Sie feststellen, daß es viele Befehle gibt, die sich von GENIE 16 C BASIC unterscheiden. Dieses Kapitel wird Ihnen als Hinführung zu diesen GENIE 16 C-spezifischen Befehlen dienen.

Wie auch immer, wenn Sie mit BASIC anfangen wollen und keine Möglichkeit haben, an ein Handbuch zu kommen, können Sie es zumindest versuchen. Dieser kurze Abschnitt erklärt Ihnen einen kleinen Teil von BASIC am Beispiel des Programms HILO.

BASIC ist eine Programmiersprache, d.h. man kann damit den Computer mit bestimmten Befehlen dazu bringen, Probleme zu bearbeiten. BASIC (Beginners All-purpose Symbolic Instruction Code) ist eine Sprache, von der viele Versionen auf verschiedenen Rechnern laufen. Meistens hat sogar jeder Computertyp einen eigenen BASIC-'Dialekt', aber es gibt genügend Ähnlichkeiten, die es erlauben, von einer Sprache zu reden.

Wenn wir BASIC untersuchen, so stellen wir fest, daß es bestimmte Schlüsselfunktionen gibt: Input (Eingabe der Daten in den Rechner), Output (Anzeigen der Informationen), rechnen und vergleichen. Wir müssen außerdem darauf achten, wie die Reihenfolge der Befehle in einem Programm, abhängig von den Umständen, bestimmt werden kann.

Das läßt sich alles an einem einfachen Programm erklären. Wir wollen ein einfaches Zahlenrate-Programm entwerfen. Ein solches Spiel ist z.B., wenn sich jemand eine Zahl zwischen 1 und 100 ausdenkt, und ein anderer versucht, diese zu erraten. Dabei wird ihm immer gesagt, ob er zu hoch oder zu niedrig geraten hat.

Zuerst müssen wir einen Weg finden, nach dem das Spiel gespielt werden soll, so daß alle Möglichkeiten durch eine problemlösende Methode gedeckt sind. Beim Programmieren nennt man den Weg, nach dem wir ein Problem lösen, Algorithmus.

Für unser Beispiel müssen wir zuerst eine geheime Nummer wählen (Schritt 1), die Zahl der Versuche auf 1 setzen (Schritt 2, nicht unbedingt notwendig, aber hilfreich), den Spieler nach der Zahl fragen (Schritt 3), herausfinden ob die Zahl stimmt (Spielende) oder ob die Zahl zu hoch oder zu tief liegt (Schritt 4), dann erhöhen wir die Zahl der Versuche um 1 und kehren zu Schritt 3 zurück.

In diesem Algorithmus verwenden wir also Input (welche Zahl?), Output (Meldung) Berechnung (welche Anfangszahl?), vergleichen (wo liegt die geratene Zahl?) und Kontrolle (was geschieht hängt von dem Rateversuch und der gewählten Zahl ab).

Der Algorithmus ist als folgendes BASIC-Programm entwickelt worden:

```
10 REM High/Low - Zahlenraten
20 REM
30 REM
40 REM
50 PRINT "HILO - Rate ein Zahl zwischen 1 und 100"
55 REM
60 LET computerzahl=INT(RND*(100+1))
65 REM
70 LET versuch=1
75 REM
80 INPUT "Gebe deine Zahl ein ";n
85 REM 1. Vergleich
90 IF n>computerzahl THEN PRINT "Zu hoch geraten"
95 REM 2. Vergleich
100 IF n<computerzahl THEN PRINT "Zu tief geraten"
105 REM Letzter Vergleich
110 IF n=computerzahl THEN GOTO 140
120 LET versuch=versuch+1:REM erhöhe die Zahl der Versuche
130 GOTO 80:REM Rücksprung in die Eingabezeile
135 REM *** Erfolg ***
140 PRINT "Erfolg nach";versuch:REM letzte Zeile
150 END
```

Jetzt können Sie sehen, wie ein BASIC-Programm aussieht. Jede Zeile fängt mit einer Nummer an, die die Reihenfolge der Befehle festlegt und Sprünge in andere Zeilen erlaubt. Eine Programmzeile kann auch wesentlich länger als eine Bildschirmzeile sein, da sich der Rechner nicht an den Bildschirmzeilen orientiert.

Das Programm fängt mit der Zeile 10 an. Der Zeilennummer folgt das Wort REM (Abkürzung für REMark = Bemerkung), d.h. der Rest der Zeile wird vom Computer ignoriert, er ist nur für das bessere Verständnis der Abläufe im Programm notwendig. Die Zeilen 20, 30 und 40, sowie alle Zeilen, die mit 5 enden, erhöhen nur die Übersichtlichkeit und haben auf das Programm keinen Einfluß.

Die erste Zeile, in der tatsächlich etwas passiert, ist die Zeile 50: Zeige die Einleitung auf dem Bildschirm. In Zeile 60 wird eine Zufallszahl vom Computer berechnet und unter dem Namen 'computerzahl' abgespeichert. Das ist dann die Zahl, die der Spieler erraten soll. Bevor wir den Spieler raten lassen, muß noch die Anzahl der Versuche gleich 1 gesetzt werden (Zeile 70).

In Zeile 80 wird der Spieler nach seiner Zahl gefragt und gibt diese ein. Falls er zu hoch (Zeile 90) oder zu tief (Zeile 100) geraten hat, wird ihm das mitgeteilt. Ist das nicht der Fall (Zeile 110), so wird ein Sprung in Zeile 140 gemacht, wo dem Spieler sein Erfolg und die Zahl der Versuche mitgeteilt werden. Wenn kein Sprung nach 140 stattgefunden hat, wird die Zahl der Versuche um 1 erhöht (Zeile 120) und aus Zeile 130 wird nach Zeile 80 zurückgesprungen.

Dieses Programm ist eine mögliche Lösung der Aufgabe, es gibt noch viele andere. Wenn man bedenkt, daß mit einem Menschen gespielt wird, so ist es vorteilhaft, das Programm wirklich idiotensicher zu machen. Wenn wir eine Zahl zwischen 1 und 100 wählen, so sind nur ganze natürliche Zahlen gemeint.

Auch könnte jemand statt 13 'DREIZEHN' eingetippen, was eine Fehlermeldung zur Folge hätte. Bei unserem Beispiel ist das nicht so schlimm, aber bei einem anderen Programm kann eine falsche Eingabe zu einem falschem Ergebnis führen.

Falls Sie noch bestimmte Befehle interessieren, wie z.B. `INT(RND*(100+1))`, werden Sie diese bei den BASIC-Befehlen mit einer entsprechenden Erklärung finden. Jetzt müssten Sie im Stande sein, den Rest dieses Kapitels zu verstehen. Sie sollten den Teil über den Bildschirmeditor durchgehen und die Programme unter 'Töne erzeugen' ausprobieren, um zu sehen wie sie funktionieren.

12.2 GENIE 16 C BASIC

GENIE 16 C BASIC ist eine Version von GW-BASIC der amerikanischen Firma Microsoft. Das bringt mehrere Vorteile mit sich: GENIE 16 C BASIC ist mit dem IBM Personal Computer und vielen anderen 16-bit Microcomputern kompatibel. Außerdem stammt dieses BASIC vom alten Standard 'Microsoft BASIC' ab, was das Übertragen der Programme von anderen Rechnern sehr leicht macht. Wenn Sie BASIC auf dem GENIE 16 C lernen, werden Sie auch keine Probleme haben, andere Computer zu benutzen.

12.3 Laden von BASIC

Beim GENIE 16 C muß BASIC von der Diskette geladen werden. Es ist unter dem Namen BASIC.COM abgespeichert. Um es zu laden geben Sie nach dem DOS Prompt (A>) BASIC ein und drücken <ENTER>.

12.4 BASIC-Prompt

GENIE 16 C BASIC meldet sich mit "Ok" und einem blinkenden Cursor eine Zeile tiefer. Das bedeutet, daß der Rechner für Ihre Eingaben bereit ist. Am unteren Rand des Bildschirms finden Sie eine Reihe weißer Kästchen. Diese zeigen die momentane Belegung der Funktionstasten <F1> bis <F10> an.

Drücken Sie <F1>, erscheint das Wort LIST auf dem Bildschirm. Der Befehl KEY OFF schaltet diese Zeile aus, mit KEY1, "text" können Sie <F1> neu definieren. Es gibt bei BASIC zwei Ebenen, die aktive Befehlsebene, vor allem zum Testen und Entwickeln von Programmen, und die Programmausführungsebene, bei der Sie nur sehen, was das Programm anzeigt. Sie kommen auf die aktive Befehlsebene, wenn das Programm zu Ende ist, wenn ein Fehler auftritt oder wenn Sie das Programm abbrechen.

Die Break-Taste:

Um ein laufendes BASIC-Programm abzubrechen, vor allem wenn es in einer Endlosschleife läuft, drücken Sie <Ctrl> und <Scroll Lock> gleichzeitig.

Jetzt erscheint die BASIC "Ok"-Meldung und der Computer befindet sich in der aktiven Befehlsebene.

12.5 Programme in BASIC

Eine Zeile im Programm kann bis zu 240 Zeichen lang werden, Zeilennummern dürfen im Bereich von 0 bis 65529 liegen. Es können mehrere Befehle in einer Zeile eingegeben werden, sie müssen dann mit einem Doppelpunkt (:) getrennt werden.

Bevor Sie ein neues Programm eingeben, löschen Sie das vorherige mit NEW. Neue Zeilen können eingegeben werden, sie werden von Computer automatisch an die richtige Stelle im Programm eingesetzt. Zeilen können einfach durch Eingabe der Zeilennummer, gefolgt von <ENTER>, gelöscht werden. Mehrere Zeilen werden mit dem DELETE-Befehl gelöscht. Schon vorhandene Zeilen werden durch die Eingabe der neuen Zeile ersetzt. GENIE 16 C hat außerdem einen Bildschirmeditor, der wirkungsvolles und einfaches Editieren von Programmen erlaubt.

12.6 Typen und Namen von Variablen

GENIE 16 C BASIC ist bei Variablennamen sehr flexibel. Es müssen aber einige Regeln beachtet werden.

Der Name

- muß mit einem Buchstaben anfangen
- in Großbuchstaben definiert werden. Das BASIC übersetzt Klein- in Großbuchstaben. So ist "franz" derselbe Name wie "FRANZ"
- darf kein reserviertes Wort sein, wie z.B. LIST, RUN, PRINT... Sie können aber diese Worte beinhalten, z.B. LETTER oder LISTE.
- darf höchstens 40 Buchstaben lang sein. Wenn Sie länger sind, benutzt das BASIC nur die ersten 40 Zeichen, um Namen voneinander zu unterscheiden.

BASIC arbeitet mit vier verschiedenen Typen von Variablen.

Der Typ kann im Namen oder mit DEF INT, DEF DBL, DEF SNG und DEF STR definiert werden.

Die Typen sind:

Einfache Genauigkeit:

Bis zu sieben Stellen lang. Im Programm werden Variablen mit weniger als acht Stellen als einfache Genauigkeit definiert.

Sie können den Typ selbst definieren, wenn Sie dem Namen ein Ausrufezeichen anfügen. Mathematische Funktionen wie SIN, COS oder TAN ergeben Werte einfacher Genauigkeit.

Ganzzahlig (engl. integer):

Ganze Zahlen im Bereich von -32768 bis +32767 werden durch Anfügen eines Prozentzeichens (%) an ihren Namen definiert. LET A%=10/3 ergibt also eine 3 in A%.

Doppelte Genauigkeit:

Bis zu 16 Stellen lang. Variablen doppelter Genauigkeit werden durch Anfügen eines '#'-Zeichens an ihren Namen gekennzeichnet.

Zeichenketten (strings):

Beinhalten Zeichen wie Buchstaben, Zahlen, Symbole, usw. Die Zeichen werden als sogenannte ASCII-Codes abgespeichert. Dieser Code wird bei den meisten Microcomputern verwendet. Eine ASCII-Liste finden Sie im Anhang. Eine Zeichenkettenvariable wird durch Anfügen eines '\$'-Zeichens an ihren Namen gekennzeichnet. Zeichenketten sind beliebige Folgen von Zeichen, die in Anführungszeichen (") eingeschlossen sind:

```
"Das ist ein Testtext"  
"%######!!!!$$"  
"12345"
```

Zeichenketten können durch Addition (+) zusammengefügt werden.

```
LET NAME$="JOSEF"+" "+"MÜLLER" weist der Variablen NAME$ den Text "JOSEF MÜLLER" zu. Zeichenketten dürfen bis zu 255 Zeichen lang werden.
```

Wissenschaftliche Schreibweise:

Meistens werden Nummern von BASIC so geschrieben, daß sie leicht zu lesen sind. Sehr kleine oder sehr große Zahlen werden in gesonderter Form (ähnlich wie beim Taschenrechner) angegeben. Die Zahl wird in zwei Teile zerlegt:

Mantisse (der Koeffizient reduziert auf eine Zahl zwischen 0 und 10) und Exponent (die Zehnerpotenz, mit der die Mantisse zu multiplizieren ist).

Beispiele:

6.23E7 entspricht also der Zahl 62300000, denn es bedeutet nicht anderes als 6.23 mal 10 hoch 7.

4.356E-09 ist 0.000000004356, also eine sehr kleine Zahl.

Die wissenschaftliche Schreibweise ist sowohl bei einfacher als auch bei doppelter Genauigkeit erlaubt. Letztere wird zur durch ein D anstelle des E gekennzeichnet.

12.7 Datenfelder

Alle bis jetzt behandelten Variablen waren einfache Variablen, d.h. einem Namen wird ein Wert zugeordnet. In BASIC gibt es auch sogenannte Datenfelder. Dabei werden dem Namen noch eine oder mehrere Indexzahlen beigefügt. So können unter demselben Namen verschiedene Werte gespeichert werden. Das Datenfeld NAME\$ besteht dann aus den Elementen NAME\$(1), NAME\$(2), NAME\$(3) usw. Die Anzahl der Elemente muß mit DIM definiert werden.

Einzelne Elemente können immer noch gesondert behandelt werden, der eigentliche Vorteil der Datenfelder ist aber, daß der Index durch eine Variable ersetzt werden kann und so alle Daten in einer Schleife ausgedruckt, sortiert oder nach einem bestimmten Wert untersucht werden können. Die einfachste Form der Datenfelder ist eine Liste. Sie hat nur eine Dimension. Z.B. erzeugt DIM NAME\$(20) eine Liste von Elementen:

```
NAME$(1)
NAME$(2)
NAME$(3)...
bis
NAME$(20)
```

Ein Datenfeld kann aber auch zwei Dimensionen haben. DIM A(3,4) erzeugt die Tabelle:

```
A(1,1)   A(2,1)   A(3,1)
A(1,2)   A(2,2)   A(3,2)
A(1,3)   A(2,3)   A(3,3)
A(1,4)   A(2,4)   A(3,4)
```

Es können auch drei, vier oder mehr Dimensionen definiert werden, obwohl das selten nützlich ist. Die absolute Grenze ist 255 mit höchstens 32767 Elementen in jeder Dimension. Aber Sie werden, durch die begrenzte Speicherkapazität kaum an diese Grenzen stossen.

Wenn ein Datenfeld definiert ist, hat es eine feste Größe. Wenn Sie später im Programm einige Elemente beifügen wollen, dann definieren Sie das Feld vorher etwas grösser.

Ein Datenfeld kann durch ERASE wieder gelöscht werden. Standardmäßig sind alle Datenfelder Listen mit elf Elementen. Außerdem fängt jede Liste bei Null an.

A(0), A(1), A(2), ... A(10).

Ein DIM X(100) definiert 101 Elemente, von X(0) bis X(100). Sie können das einfach ignorieren. Wenn es Sie aber sehr stört, vor allem wenn durch viele mehrdimensionale Felder Kapazitätsmangel entsteht, können Sie mit OPTION BASE das Feld mit eins anfangen lassen.

12.8 Arithmetische Funktionen

Bei BASIC gibt es viele Befehle, mit denen man Berechnungen ausführen kann. Das BASIC rechnet wie ein Mathematiker oder ein guter Taschenrechner, kennt also die Punkt vor Strich-Regel. $2+5*3$ ergibt also 17, nicht 21. Hier ist eine Liste der Befehle in der Reihenfolge, wie sie ausgeführt werden:

1. BASIC Funktionen Befehle wie ABS oder SQR werden immer zuerst berechnet.
2. Klammern Der Teil des Ausdruckes, der in Klammern steht, wird vor dem Teil außerhalb berechnet. So ist der Wert von $(2+10)*4$ gleich 48.
3. Potenzieren das (^) Zeichen wird zum Potenzieren verwendet.
 $4^2=4*4=16$.
 $8^{(1/3)}$ ist die dritte Wurzel aus 8, also 2.
4. Negation -4 ist minus 4.
5. *,/ Multiplizieren und Dividieren.

6. Integer dividieren funktioniert wie normales Dividieren, das Ergebnis ist aber immer ganzzahlig.
7. MOD berechnet den Rest bei einer Division. 25 MOD 6 ist also 1.
8. +,- Addieren und Subtrahieren.
9. Vergleichen Zwei Werte werden verglichen. Das Ergebnis ist 0, wenn die Bedingung nicht erfüllt ist und -1 wenn sie wahr ist. Normalerweise werden Bedingungen mit IF-THEN-Konstruktionen genutzt. Folgende Vergleiche sind erlaubt:
- | | |
|------------|----------------|
| = | ist gleich |
| <> oder >< | nicht gleich |
| < | kleiner als |
| > | größer als |
| <= oder =< | kleiner gleich |
| => oder >= | größer gleich |

10. Logische Verknüpfungen werden benutzt, um logischen und booleschen Ausdrücken Numerische Werte zuzuordnen. Meist werden Sie in IF-THEN-Konstruktionen verwendet:

IF X<0 OR X>10 THEN ...

Sie können auch zum bitweisen Verknüpfen zweier Zahlen benutzt werden. Sie arbeiten nach folgendem Schema:

	X Y	X Y	X Y	X Y
	F F	F W	W F	W W
X AND Y	F	F	F	W
X OR Y	F	W	W	W
X XOR Y	F	W	W	F
X EQV Y	W	F	F	W
X IMP Y	W	W	F	W

F steht für falsch (0), W für wahr. Jeder nicht-null-Wert wird bei der Berechnung wie wahr behandelt, als Ergebnis eines Vergleichs ist wahr aber immer -1. Mit NOT können bei jeder Verknüpfung F und W vertauscht und damit die Verknüpfung negiert werden.

12.9 Der BASIC-Editor

GENIE 16 C BASIC hat einen sehr leistungsstarken Bildschirmeditor. Der Vorteil dieses Editors ist, daß jede BASIC-Zeile, die auf dem Bildschirm steht, direkt mit dem Cursor korrigiert werden kann. Er ist sehr einfach zu bedienen.

Der Editor benutzt die Pfeiltasten auf der rechten Seite der Tastatur. Dazu sollte die <Num Lock> Taste ausgeschaltet sein. Um das festzustellen, drücken Sie auf der Zahlentastatur die Taste 6. Erscheint eine 6 auf dem Bildschirm, so betätigen Sie die <Num Lock> Taste. Jetzt werden die Editorbefehle anstelle der Zahlen ausgegeben.

Folgende Tasten werden vom Editor benutzt:

< ↑ >	(8)	Eine Zeile höher
< ↓ >	(2)	Eine Zeile tiefer
< ← >	(4)	Ein Zeichen nach links
< → >	(6)	Ein Zeichen nach rechts
<Home>	(7)	In die erste Zeile setzen
<End>	(1)	An das Ende der Zeile setzen
<Esc>		Löscht die gesamte Zeile
<Backspace>		Löscht das Zeichen links vom Cursor
		Löscht das Zeichen rechts vom Cursor
<Ins>		Neue Zeichen einsetzen
<Enter>		Übernimmt die Zeile

Um eine Zeile zu verbessern, muß diese auf dem Bildschirm sein. Das erreichen Sie mit dem LIST Befehl. Falls die Zeile schon auf dem Bildschirm ist, sei sie schon gelistet oder gerade eingegeben, brauchen Sie das nicht zu machen.

Wollen Sie eine Zeile korrigieren, die nicht auf dem Bildschirm steht, benutzen Sie den EDIT-Befehl. Er listet die Zeile und setzt den Cursor an deren Anfang. Bewegen Sie den Cursor mit den Pfeiltasten, <Home> und <End> an die Stelle, an der Sie eine Korrektur vornehmen wollen. Tippen Sie neue Zeichen über die alten, oder benutzen Sie <Ins>, und <Backspace> um etwas zu verändern. Wenn die Zeile richtig ist, drücken Sie <ENTER>.

Gebrauch von <Ins>:

Wenn Sie <Ins> drücken, bekommt der Cursor ein anderes Zeichen, um zu signalisieren, daß der INSERT-Modus eingeschaltet ist. Beim Eintippen wird der Text nach rechts verschoben, um neuen Eingaben Platz zu machen.

Um den INSERT-Modus auszuschalten, drücken Sie erneut <Ins> oder benutzen Sie einen der anderen Editor-Befehle.

Gebrauch von :

Mit können Sie das Zeichen, auf dem der Cursor steht, löschen. Die Lücke wird mit dem nachfolgenden Text gefüllt.

Gebrauch von <Backspace>:

Löscht das Zeichen, das links vom Cursor steht. Dabei wird der Cursor nach links bewegt und der nachfolgende Text nachgerückt.

Alle Veränderungen werden nur dann akzeptiert, wenn auch anschließend <ENTER> gedrückt wurde. Manchmal ist es leichter, mehrere Verbesserungen in mehreren Zeilen vorzunehmen und dann alle Zeilen mit <ENTER> durchzugehen. Vergessen Sie außerdem nicht, daß die Editorbefehle auch dann arbeiten, wenn die BASIC-Zeile länger als eine Bildschirmzeile ist.

Wenn Sie einmal viel mit dem Bildschirmeditor arbeiten, werden Sie feststellen, daß Sie oft neue Befehle über eine schon vorhandene Zeile schreiben. Um das zu vermeiden benutzen Sie besser <Esc>, bevor Sie auf einem vollen Bildschirm etwas neues eintippen. Das löscht die Zeile vom Bildschirm und vermeidet, daß Ihre Eingaben mit irgendwelchen alten Informationen auf dem Bildschirm vermischt werden. Viele benutzen auch den Pfeil nach unten, um am untersten Rand neue Befehle einzugeben. Wenn Sie viele Verbesserungen gemacht haben, löschen Sie den Bildschirm (CLS) und LISTen das Programm noch einmal.

Tricks mit dem Bildschirmeditor:

Es gibt eine Menge nützlicher Tricks mit dem Bildschirmeditor. Wenn Sie erstmal an das System gewöhnt sind, werden Sie sicherlich auch noch weitere herausfinden. Hier sind einige der bekanntesten:

Alle Änderungen vergessen:

Wenn Sie sich stark vertippt haben, tippen Sie <Esc> und editieren Sie die Zeile erneut.

Einen Befehl wiederholen:

Stellen Sie den Cursor auf die entsprechende Zeile und tippen Sie <ENTER>.

Eine Zeile kopieren:

Um eine Zeile zu kopieren fahren Sie mit dem Cursor auf die entsprechende Stelle und ändern die Zeilennummer.

Zeile zurückholen:

Haben Sie eine Zeile aus Versehen gelöscht, haben Sie aber noch auf dem Schirm, fahren Sie mit dem Cursor auf die entsprechende Stelle und tippen Sie <ENTER>.

12.10 Der freie Speicherplatz

Um nachzusehen, wieviel Speicher Sie im Augenblick zur Verfügung haben, tippen Sie 'PRINT FRE (0)'. Der GENIE 16 C meldet Ihnen etwa 61 K freien Speicherplatz, obwohl Sie ja mindestens 256 K an Speicher besitzen.

Diese Beschränkung wird durch die interne Systemarchitektur des 8088 Prozessors aufgestellt, der den Speicher in 64K-Blöcke einteilt. Würde BASIC diese Blöcke ignorieren, wäre es dadurch sehr langsam. Sie werden jedoch im seltensten Fall diese Grenze erreichen.

Wollen Sie das restliche RAM erreichen, müssen Sie mit dem DEF SEG= Befehl und den Befehlen PEEK, POKE, CALL und USR arbeiten.

Der GENIE 16 C bietet eine sehr große Speicherkapazität, die von BASIC gar nicht ausgenutzt werden kann. Andere Sprachen, wie C oder Assembler erlauben ohne weiteres den Zugriff auf einen Speicher dieser Größe. Der Hauptgrund für einen derartigen Speicher ist jedoch, professionellen Programmsystemen genügend Speicher zu bieten.

12.11 Format von gespeicherten Programmen.

Das GENIE 16 C BASIC speichert Programme auf Diskette normalerweise in einem komprimierten Format ab. Dieses Format ist zwar sehr platzsparend, es ist jedoch nur für BASIC verständlich. Es ist aber möglich, ein BASIC-Programm auch in reinem ASCII-Format abzuspeichern. Dadurch kann es z.B. von EDLIN bearbeitet werden etc. Um ein Programm in ASCII abzuspeichern, benutzen Sie folgendes Format:

```
SAVE "filename",A
```

Viele andere Microcomputer benutzen BASICs, die dem des GENIE 16 C ähnlich sind. Es ist jedoch unmöglich, von diesen Computern Programme zu überspielen, weil sie komprimiert abgespeichert sind. Speichern sie jedoch im ASCII-Format, sollte es durchaus möglich sein, daß ein Programm von einem ganz anderen Rechner bei Ihnen läuft. Benutzt dieser Rechner dann auch noch MS-DOS, können Sie wahrscheinlich dessen Disketten direkt lesen. Andernfalls können Sie beide Rechner über eine serielle Leitung oder über ein Modem koppeln.

Wollen Sie ein ASCII-File laden, können Sie dieses genauso tun, als wenn es sich um einen komprimierten File handeln würde.

Es gibt noch eine zweite Option, um Programme abzuspeichern. Die P-Option wird genauso eingesetzt wie die A-Option und hat folgenden Effekt:

Ein mit der P-Option abgespeichertes Programm wird durch einen ausgeklügelten Verschlüsselungscode verschlüsselt und läßt sich danach weder listen noch editieren.

Seien Sie also gewarnt:

Wenn Sie die P-Option anwenden ist es unmöglich, den ursprünglichen Zustand des Programms wiederherzustellen. Machen Sie also nur verschlüsselte Versionen von Programmen, von denen Sicherheitskopien vorhanden sind.

12.12 Der Zeichensatz

Der GENIE 16 C hat einen vollständigen Zeichensatz aus 256 verschiedenen Zeichen, die nach dem für Microcomputer üblichen ASCII-Code geordnet sind. Eine vollständige Liste des ASCII-Codes für den GENIE 16 C finden Sie im Anhang. Auf diese können Sie von BASIC aus durch `CHR$(x)` zugreifen. Mit dieser Zeichenvielfalt können Sie praktisch alles darstellen, was Sie brauchen. Sie können die Zeichen für mathematische Formeln benutzen, Zeichnungen erstellen und sogar einfache Spiele damit programmieren. Befriedigen Sie diese Möglichkeiten nicht, so können Sie noch eigene Zeichen definieren.

12.13 Abfragen der Tastatur

Das GENIE 16 C BASIC verfügt über alle Möglichkeiten des normalen BASIC, um die Tastatur abzufragen, einschließlich INPUT und LINE INPUT. Zusätzlich verfügt es noch über eine Reihe von nützlichen Befehlen, um möglichst viele Anwendungen zu ermöglichen.

INPUT\$

INPUT\$ liest eine bestimmte Anzahl von Zeichen von der Tastatur. Dabei können auch Zeichen verwendet werden, die normalerweise bei INPUT nicht verwendet werden dürfen, wie <ENTER> und <BACKSPACE>. Die eingetippten Buchstaben erscheinen nicht auf dem Bildschirm.

Zum Beispiel:

```
10 CLS
20 PRINT "Geben Sie ein Codewort ein:"
30 PRINT "Maximal 3 Zeichen!"
40 IF INPUT$(3) <> "BUH" THEN 10
50 PRINT "BUH ist richtig"
```

Die einzige Möglichkeit, um INPUT\$ abubrechen, ist die <BREAK>-Taste.

INKEY\$

Diese Funktion liest einen einzelnen Tastendruck von der Tastatur, ohne zu warten, ob eine Taste gedrückt wurde. Die Tasten des GENIE 16 C geben nicht immer einen einzelnen Tastendruck wieder. Die Tasten F1 bis F10 z.B. geben Ihren Inhalt immer Buchstabe für Buchstabe wieder, wenn sie aktiviert sind. Sind Sie nicht aktiviert, gibt das System nur einen zwei-Byte Tastencode zurück. Das erste Byte ist Null und das zweite enthält dann einen besonderen Tastencode, der bei jeder Taste anders ist.

Das System mit den verlängerten Tastencodes versetzt den Benutzer in die Lage, Funktionen von der Tastatur abzufragen, ohne dabei mit den normalen ASCII-Codes in Konflikt zu kommen.

Folgende Tabelle enthält den kompletten Satz von erweiterten Tastencodes:

Code	Taste
----	-----
3	NUL
15	<SHIFT><TAB>
16-25	<ALT> und Q,W,E,R,T,Y,U,I,O,P
30-38	<ALT> und A,S,D,F,G,H,J,K,L
44-50	<ALT> und Z,X,C,V,B,N,M
59-68	F1-F10 wenn nicht aktiviert
71	<Home>
72	<Pfeil hoch>
13	<Pg Up>
75	<Pfeil links>
77	<Pfeil rechts>
79	<End>
80	<Pfeil unten>
81	<Pg Dn>
82	<Ins>
83	
84-93	<Shift> F1 - F10
94-103	<Ctrl> F1 - F10
104-113	<Alt> F1 - F10
114	<Ctrl> <Prt Sc>
115	<Ctrl> <Pfeil links>
116	<Ctrl> <Pfeil rechts>
117	<Ctrl> <End>
118	<Ctrl> <Pg Dn>
119	<Ctrl> <Home>
120-131	<Alt> und 1,2,3,4,5,6,7,8,9,0,-,=
132	<Ctrl> <Pg Up>

12.14 Der Zufallszahlengenerator

BASIC besitzt die Möglichkeit, mittels eines Zufallgenerators die Erzeugung von Zufallszahlen zu simulieren. Echte Zufallszahlen kann der Computer nicht erzeugen, die Simulation nähert sich jedoch stark der Wirklichkeit.

Die GENIE 16 C-Zufallszahlenfunktion ist $RND(n)$, wobei das Ergebnis eine Zufallszahl zwischen 0 und 1 ist. Ist n größer als 0, so liefert RND die nächste Zufallszahl aus der internen Zufallszahlentabelle. Ist $n = 0$, so wird die letzte erzeugte Zufallszahl erneut ausgegeben. Dies kann bei der Fehlersuche sehr nützlich sein.

Eine sinnvolle Formel, um eine Zufallszahl zwischen m und n zu erzeugen, ist:

```
DEF FN R(X)=INT(RND(1)*(n-m))+m
```

12.15 'Soft keys' und Tastenfallen

Die zehn Funktionstasten des GENIE 16 C können auf zwei verschiedene Arten benutzt werden. Unter BASIC haben Sie die Möglichkeit, die Funktionstasten als 'soft keys' zu definieren. Den Tasten kann also eine bestimmte Zeichenfolge zugeordnet werden, die sich dann auf Druck einer einzigen Taste abrufen läßt.

BASIC bietet weiterhin die Möglichkeit, den Tasten Interruptfunktionen zuzuordnen. Ein Programm kann damit unterbrochen werden und ein bestimmtes Unterprogramm ausführen.

Die Benutzung der 'Soft Keys' ist recht einfach. Der Befehl KEY ON zeigt die Tastenbelegung in der untersten Bildschirmzeile. Im 40-Zeichen Modus sind die ersten 5, im 80-Zeichen Modus sind alle Belegungen sichtbar. Mit KEY OFF wird diese Anzeige wieder abgeschaltet. Jede Taste kann mit einer Zeichenfolge von bis zu 15 Zeichen belegt werden. Dennoch werden nur die ersten 6 Zeichen jeder Belegung angezeigt. Um eine vollständige Anzeige der Belegung zu erhalten, benutzen Sie den Befehl KEY LIST.

Um eine Funktionstaste zu belegen, benutzen Sie folgende Syntax:

```
KEY n,string
```

Wollen Sie z.B. der Funktionstaste F10 die Funktion zum Löschen des Bildschirms zuordnen, so tippen Sie

```
KEY 10,"CLS"+CHR$(13)
```

Das CHR\$(13) ist der Code für <ENTER> und bewirkt, daß die Funktion sofort ausgeführt wird.

Um einen 'Soft Key' von seiner Funktion zu befreien, ordnen Sie ihm den Nullstring zu. Tippen Sie also

```
KEY n,""
```

Die so entwertete Taste wird bei der Abfrage mit INKEY\$ einen zwei Byte Tastencode erzeugen. Das erste Byte ist 00H, das zweite Byte ist der Code für die Taste (siehe Tabelle). Es ist sinnvoll, den Tasten eine besondere Funktion zuzuordnen, wenn z.B. Programme entwickelt werden.

Wird BASIC gestartet, sind die Funktionstasten mit zehn sinnvollen Befehlen versehen. Sie können jedoch auf die beschriebene Art die Tastenbelegung Ihren Bedürfnissen anpassen. Die Funktionstasten können auch in Ihren eigenen Programmen benutzt werden, indem Sie diesen die entsprechenden Werte zuordnen. Bedenken Sie jedoch, daß bei einer Abfrage durch INKEY\$ die Zeichen einzeln durchgegeben werden.

Tastenfallen:

Die zweite Möglichkeit zur Tastenabfrage steht nur unter Disk-BASIC zur Verfügung. Es handelt sich dabei um sogenannte Tastenfallen (engl. key traps), die ähnlich arbeiten wie der bekannte ON ERROR GOTO Befehl.

Es besteht die Möglichkeit, Befehle der Form ON KEY(n) GOSUB aufzustellen. Dadurch springt das Programm bei jedem Druck auf die entsprechenden Taste in eine Unterroutine.

Die Tastencodes sind wie folgt:

1 bis 10	Die Funktionstasten
11	Pfeil hoch
12	Pfeil links
13	Pfeil rechts
14	Pfeil nach unten

Zum Beispiel würde die Zeile

```
ON KEY(1) GOSUB 1000
```

am Anfang eines Programms jedesmal in ein Unterprogramm ab Zeile 1000 springen, wenn die Taste <F1> gedrückt wird. Sobald der ON KEY(n) GOSUB Befehl aktiviert ist, kann mit KEY (n) ON (nicht zu verwechseln mit KEY ON) die Abfrage aktiviert werden. Sobald jetzt die Taste <F1> gedrückt wird, springt BASIC in Zeile 1000 - egal, was es gerade macht.

Die Tastenabfrage kann, z.B. bei einem Programmteil mit höchster Priorität, mit

```
KEY (n) OFF
```

abgeschaltet werden.

Eine andere Variation dieses Befehls ist

KEY (n) STOP

Mit KEY (n) STOP wird der Tastendruck von BASIC zwar bemerkt, aber es wird nicht in Zeile 1000 gesprungen. BASIC wartet erst, bis wieder KEY (n) ON gegeben wurde. Dann wird in Zeile 1000 gesprungen.

Jede 'gefangene' Taste braucht Ihre eigene Routine, mit der sie bearbeitet wird. Nachdem der ON KEY GOSUB Befehl ausgeführt wurde, ist es unmöglich, mit dem INKEY\$-Befehl o.ä. festzustellen, welche Taste gedrückt wurde.

BASIC führt einen automatischen KEY (n) STOP aus, sobald eine Taste gedrückt wurde und behandelt wird. Es wäre ja auch ziemlich sinnlos, wenn eine Taste in der eigenen Bearbeitungsroutine 'gefangen' wird. Sobald wieder in das Hauptprogramm zurückgesprungen wird, führt BASIC einen KEY (n) ON aus.

Beispiel:

```
10 ON KEY(1) GOSUB 1000
20 KEY (1) ON
30 PRINT I
40 I=I+1
50 GOTO 30
1000 REM Bearbeitungsroutine für <F1>
1010 PRINT:PRINT "<F1> ist gedrückt !"
1020 RETURN
```

12.16 Töne

Der GENIE 16 C hat einen kleinen eingebauten Piezo-Lautsprecher, über den auf einfache Weise Töne erzeugt werden können. BASIC verfügt zu diesem Zweck über diverse Tonbefehle.

BEEP

Der BEEP-Befehl ist in beiden Versionen von GENIE 16 C BASIC verfügbar und läßt den Lautsprecher einmal kurz aufpiepen. Der BEEP Befehl ist identisch dem BELL (CHR\$(7)) auf alten Terminals. Sie können BEEP in Ihren Programmen verwenden, um einen warnenden Ton zu erzeugen.

SOUND

SOUND ist ein wesentlich vielseitigerer Befehl. Er wird zusammen mit der Frequenz der zu spielenden Note und der Zahl der zu spielenden Taktimpulse angegeben. Es gibt 18.2 Impulse pro Sekunde und die Zeitdauer kann zwischen 0 und 65535 liegen.

Das Programm fährt fort, auch wenn der Ton noch nicht zu Ende ist, hält aber vor dem nächsten an und wartet, bis der noch laufende Ton zuende ist.

Beispiel:

RANDOM-Sound:

```
10 SOUND INT(RND(1)*1000)+130,RND(1)
20 GOTO 10
```

SOUND DEMO 2:

```
10 REM SOUND DEMO 2
20 FOR F=523 TO 261 STEP-10
30 SOUND F,F/1000
40 NEXT F
50 GOTO 20
```

PLAY

Es handelt sich hier sicherlich um den komfortabelsten Sound-Befehl. Nach PLAY steht ein String mit Buchstaben-Codes für diverse Funktionen.

Ein Beispiel:

```
PLAY "CDEFGABC"
```

Die komplette 'Musik-Sprache' ist folgende:

A-G Spielt die entsprechende Note in der momentanen Oktave. Jede Note kann von einem '+' oder einem '-' gefolgt werden, um den Notenwert um eine Halbnote zu erhöhen. '+' und '-' arbeiten jedoch nur auf einer normalen Tonleiter. Wird B+ angegeben, so weiß BASIC nicht, daß C gemeint ist.

Jede Note kann von einer Zahl gefolgt werden. Diese Zahl gibt an, wie lange die Note spielen soll. Es gilt $1/\text{Zahl}$. Für eine ganze Note ist $n=1$, für eine halbe $n=2$ etc. n kann von 0 bis 64 gehen.

On Wählt die Oktave aus. n kann von 0 bis 6 gehen. BASIC startet normalerweise immer auf Oktave 4. Jede Oktave beginnt mit C und endet mit B.

Nn Dies ist ein alternativer Weg, um Noten zu spielen. n muß zwischen 0 und 84 liegen. Dieser Befehl ist vor allem dann sinnvoll, wenn nur für eine oder zwei Noten die Oktave gewechselt werden muß.

Ln L setzt die Länge der folgenden Noten. Dieser Befehl arbeitet genauso, als würden Sie für eine Note eine Länge festlegen. Diese Länge gilt dann allerdings auch für alle folgenden Noten.

Pn Definiert eine Pause. Wird genauso angegeben, wie ein Ton ($\text{Länge} = 1/n$).

Tn Bestimmt das allgemeine Tempo der Musik, indem die Zahl der Halbnoten pro Minute angegeben wird. n kann einen Wert zwischen 0 und 255 annehmen.

MF Schaltet in den Vordergrundmodus. Ein neuer Ton läßt das Programm anhalten und warten, bis der alte Ton beendet wurde. BASIC startet in diesem Modus.

MB Schaltet in den Hintergrundmodus. Die Musik (von einem PLAY oder einem SOUND-Befehl) kommt in einen Zwischenspeicher und BASIC kann weiterarbeiten. Die Noten stehen also in einer Warteschlange. Diese Warteschlange kann maximal 32 Noten lang sein.

MN Wählt den 'normalen' Musikmodus aus.

ML Wählt den Legato-Modus aus.

MS Wählt den Staccato-Modus aus.

X variable:

Dieser Befehl bietet die Möglichkeit, dem PLAY Befehl einen String zuzuführen, den dieser dann bearbeiten kann. XTlässt also den Inhalt von Tspielen.

Auf diese Weise können Tonfolgen, die sich öfter wiederholen, nicht immer wieder eingetippt zu werden.

Zusammensetzen von PLAY-Strings:

Die Strings können beliebig viele Befehle enthalten und in jeder gewünschten Reihenfolge abgespielt werden. Wird der String zu lang, zerlegen Sie ihn in mehrere kleine Strings und X'en diese anschließend zusammen. Zur Übersichtlichkeit können Sie die Befehle auch mit Leerzeichen trennen. Wo immer ein n verlangt ist, kann es durch eine Konstante (wie in G4A2) oder durch eine Variable ersetzt werden. Um das n durch eine Variable bestimmen zu lassen, muß dem Befehl ein Gleichheitszeichen, die Variable und ein Semikolon folgen.

Ein Befehl wäre also G=Y;A=Y;

12.17 Der Bildschirm des GENIE 16 C

Der GENIE 16 C besitzt eine Reihe von sehr effektvollen Befehlen, um mit dem Bildschirm umzugehen.

TEXT screens

40 x 25 Buchstaben,
16 Farben + Blinkend,
8 Schirme

80 x 25 Buchstaben,
16 Farben + Blinkend,
4 Schirme

Mittlere Auflösung:

320 x 200 Graphik,
40 x 25 Text,
4 Farben,
2 Paletten

Hochauflösende Graphik

640 x 200 Graphik,
80 x 25 Text,
Schwarzweiß

Zusätzlich können Sie die Farbe der Schirmumrandung bestimmen. Benutzer, die mit dem IBM vertraut sind, sollten bemerken, daß der GENIE 16 C nur über einen Farbgraphikadapter verfügt. Probleme wie mit dem Schwarzweißadapter von IBM treten also nicht auf.

Der SCREEN Befehl kann benutzt werden, um zwischen den verschiedenen Bildschirmmodi umzuschalten. Innerhalb der Bildschirmmodi kann man dann mit COLOR zwischen den zahlreichen Farben umschalten. Der SCREEN Befehl sieht wie folgt aus:

SCREEN modus,burst,aseite,vseite

'modus' wählt die Graphikauflösung an, die Sie benutzen wollen. 0 wählt den Textmodus an (keine Graphik), 1 wählt die mittlere Graphikauflösung an (320 x 200 Punkte) und 2 wählt die hochauflösende Graphik an.

'burst' schaltet das Farbsignal ein oder aus. Das Auftreten eines Farbsignals kann zur Bildverschlechterung bei einfarbigen Monitoren führen. Im Textmodus führt ein Burstwert von 1 zu einem Farbsignal und 0 zu keinem Farbsignal. Bei mittlerer Auflösung ist dies genau umgekehrt. Bei hochauflösender Graphik ist der Wert von Burst irrelevant, da es sich dabei nur um Schwarzweißgraphik handelt.

'aseite' und 'vseite' geben zwei unterschiedliche Bildschirmseiten an. Die 'aseite' ist diejenige Bildschirmseite, die verändert werden kann (aktive Seite),

'vseite' ist diejenige Bildschirmseite, die sichtbar ist. Auf der aktiven Bildschirmseite werden sämtliche Manipulationen wie CLS, PRINT etc. vorgenommen.

Die sichtbare Bildschirmseite ist diejenige Seite, die auf dem Schirm zu sehen ist. Ein CLS löscht also nur die aktive Seite, es muß aber nicht unbedingt etwas auf dem Bildschirm passieren.

Durch diese Technik ist es z.B. möglich, etwas anzuzeigen, obwohl der Computer an einem der nicht sichtbaren Bildschirme arbeitet. Die Parameter müssen nicht unbedingt alle angegeben werden. Sie können Befehle übergehen, indem Sie einfach mehr Kommas machen. Bei Optionen am Ende müssen Sie also die Kommata weglassen:

```
SCREEN 0,0,0    Schwarzweißschirm, Seite 0
                aktiv
SCREEN ,,1,1    Seite 1 sichtbar und aktiv
SCREEN 2        320 x 200 Punkte Graphik
etc.
```

sind also geltende Befehle.

12.18 Der Textschirm

Der Textschirm ist am einfachsten zu behandeln. BASIC startet immer im Textmodus, Sie sollten durch SCREEN 0 jedoch immer sicherstellen, daß Sie sich im Textmodus befinden. Die unterschiedlichen Bildschirmbreiten werden mit WIDTH 40 bzw. WIDTH 80 eingestellt. Das Ausführen dieser Befehle löscht jedoch den Bildschirm und setzt die Bildschirmrandfarbe auf Schwarz.

Auswahl der Textfarben:

Die Syntax des COLOR-Befehls lautet:

COLOR Vordergrund,Hintergrund

'Vordergrund' muß zwischen 0 und 31 liegen, wobei die Farben von 16 bis 31 die blinkenden Farben 0 bis 15 sind.

'Hintergrund' muß zwischen 0 und 7 liegen.

Eine Rahmenfarbe läßt sich über den Befehl OUT &H3D9,Rand festlegen. Der Wert von "Rand" muß zwischen 0 und 15 liegen.

Die Text-Farben sind:

0 Schwarz	8 Grau
1 Blau	9 Hellblau
2 Grün	10 Hellgrün
3 Cyan	11 Helles Cyan
4 Rot	12 Hellrot
5 Magenta	13 Helles Magenta
6 Braun	14 Helles Braun
7 Weiß	15 Leuchtendes Weiß

Ein einfaches Beispiel für einen COLOR-Befehl ist COLOR 4,7 was einen roten Text auf einem weißen Schirm festlegt.

Das GENIE 16 C BASIC hat zwei Möglichkeiten, den Cursor zu kontrollieren. Der Befehl LOCATE bietet direkte Kontrolle über die Position des Cursors. Weiterhin besteht die Möglichkeit, den Cursor mittels besonderer Steuerzeichen zu bewegen. Diese Steuerzeichen sind:

- 09 Tabulator auf die nächste Position
- 10 Zeilenvorschub (Cursor auf den Anfang der nächsten Zeile)
- 11 Cursor auf die obere, linke Bildschirm-ecke
- 12 Formfeed (löscht den Bildschirm)
- 13 CR (Cursor auf den Anfang der nächsten Zeile)
- 28 Cursor nach rechts
- 29 Cursor links
- 30 Cursor hoch
- 31 Cursor runter

Diese Steuerzeichen können auf einfache Weise benutzt werden. PRINT CHR\$(29) bewegt z.B. den Cursor um eine Position nach links.

LOCATE Zeile,Spalte,Cursor

LOCATE bewegt den Cursor auf eine bestimmte Bildschirmposition. Die Befehle POS (0) und CSRLIN dienen dazu, die Position des Bildschirms abzufragen. In beiden Fällen ist die obere, linke Ecke des Schirms 1,1.

Normalerweise können nur 24 Zeilen des Bildschirms benutzt werden, denn die 25te Zeile enthält die Belegung der Funktionstasten. Sie können jedoch die 25te Zeile mit Information belegen, wenn Sie KEY OFF ausführen und anschließend mit LOCATE den Cursor in die 25te Zeile bewegen. Dort können Sie dann mit PRINT Informationen ablegen. Die 25te Zeile wird nicht wie die anderen Zeilen gescrollt, sondern bleibt immer erhalten. Sie eignet sich daher ausgezeichnet dazu, Statusmeldungen und Hilfsinformationen für Ihren eigenen Programme aufzunehmen. Wird der Cursor-Wert auf 0 gesetzt, ist der Cursor abgeschaltet. Wird 1 eingegeben, so blinkt der Cursor. Dies gibt Ihnen die Möglichkeit, den Cursor während einer Abfrage mit INKEY\$ zu aktivieren.

12.19 Der unsichtbare Bildschirm

Wie schon besprochen, verfügt der GENIE 16 C über 8 bzw. 4 unabhängige Bildschirme. Wird BASIC gestartet, so sind der aktive und der sichtbare Schirm auf Schirm 0 gesetzt. Eine andere Konfiguration läßt sich leicht mit SCREEN einstellen. BASIC verfügt jedoch nur über einen Cursor. Wird in einen anderen Bildschirm gesprungen, müssen die Koordinaten des Cursors abgespeichert werden. Dies muß etwa in der beschriebenen Weise erfolgen:

```
100 REM Umschalten auf Schirm 2
110 CX1=POS(0):CY1=CSRLIN:REM alte Position
120 SCREEN ,,2,2
130 LOCATE CY2,CX2:REM alte Position wieder
    herstellen.
.
.
.
200 REM Umschalten auf Schirm 1
210 CX2=POS(0):CY2=CSRLIN:REM alte Position
220 SCREEN ,,1,1
230 LOCATE CY1,CX1
usw.
```

12.20 Der Graphik-Bildschirm

Der GENIE 16 C muß sich im Graphik-Modus befinden, bevor Sie mit dem Graphik-Bildschirm arbeiten können. Befinden Sie sich in der hochauflösenden Graphik, können Sie mit WIDTH 40 in die mittlere Auflösung umschalten, mit WIDTH 80 können Sie von der mittleren in die hohe Auflösung schalten.

Auswahl der Palette:

Diese Technik bezieht sich nur auf die mittlere Auflösung. Hochauflösende Graphik ist nur in Schwarzweiß und der COLOR Befehl erzeugt hier einen SYNTAX ERROR. Die mittlere Auflösung bietet die Auswahl einer von zwei Paletten. Diese Paletten bestehen jeweils aus vier Farben:

Farbe	Palette 0	Palette 1
0	Hintergrund	Hintergrund
1	Grün	Cyan
2	Rot	Magenta
3	Braun	Weiß

Wie Sie sehen, wird bei Farbe 0 immer dieselbe Farbe angewählt, wie sie sich auf dem Bildschirmhintergrund befindet. Die SYNTAX für den COLOR Befehl ist in diesem Fall:

COLOR Hintergrund,Palette

Der Hintergrund kann aus den 16 verfügbaren Farben ausgewählt werden (siehe Text-Bildschirme). Ist Palette eine gerade Zahl, wird Palette 0 angewählt, ist es eine ungerade Zahl, wird Palette 1 angewählt. Alle folgenden PSET, PRESET etc. Befehle beziehen sich jetzt auf die neue Palette.

Eine Kurztour durch die Graphikbefehle.

Bringen Sie das System durch einen in die mittlere Auflösung (SCREEN 1,0). Der Bildschirm wird jetzt aufgeteilt in 320 Elemente in horizontaler und 200 Elemente in vertikaler Position. Wenn wir mit der mittleren Auflösung arbeiten, können wir jeden einzelnen Punkt in diesem Feld setzen, zurücksetzen bzw. ihm eine Farbe zuordnen. Um BASIC mitzuteilen, welcher Punkt auf dem Bildschirm gemeint ist, benutzen wir ein einfaches Koordinatensystem. Der Punkt ganz oben links hat die Koordinaten (0,0). Der Punkt genau in der Mitte des Schirms hat die Koordinaten (160,100) und der Punkt in der unteren Ecke hat die Koordinaten (319,199).

PSET (x,y),Farbe

Dieser Befehl zeichnet einen Punkt auf eine bestimmte Stelle des Bildschirms. Mit x,y werden die Koordinaten des Punktes angegeben.

Mit 'Farbe' kann die zu benutzende Farbe definiert werden. Es werden die Farben aus den Paletten entnommen, die bei COLOR definiert wurden. 0 ist dann die entsprechende Hintergrundfarbe. Wird die hohe Auflösung benutzt, wird bei Angabe von 1 ein Punkt gezeichnet, bei Angabe von 0 wird kein Punkt gezeichnet.

Des weiteren gibt es einen sehr ähnlich lautenden Befehl, PRESET. Dieser verhält sich auch ähnlich. Der Punkt (x,y) wird nur zurückgesetzt bzw. auf die Hintergrundfarbe gesetzt, wenn keine Farboption angegeben wurde. Wird eine Farboption angegeben, so verhält sich PRESET genau wie PSET.

LINE

Zeichnungen mit dem PSET-Befehl ausführen zu lassen, würde zu lange dauern. Ein Befehl, der das Verbinden zweier Punkte erlaubt, heißt LINE.

Die Syntax des LINE-Befehls ist folgende:

LINE (x1,y1)-(x2,y2),Farbe,Option

x1 und y1 sind die Koordinaten des Anfangspunktes, x2 und y2 die Koordinaten des Endpunktes der Linie.

'Farbe' gibt die Farbe der zu zeichnenden Linie an (siehe COLOR Befehl).

Wird B als 'Option' angegeben, so wird ein Rechteck mit gegebenen Eckpunkten gezeichnet.

Wird BF als 'Option' angegeben, so wird ein Rechteck gezeichnet und ausgefüllt.

Die einfachste Form zur Angabe einer Linie sieht so aus:

```
LINE -(x2,y2)
```

Dieser Befehl zeichnet eine Linie vom letzten angesprochenen Punkt zur angegebenen Bildschirmposition.

Weiterhin können die Koordinaten nicht nur absolut, sondern auch relativ angegeben werden, dies geschieht mit der STEP-Option.

```
LINE -STEP(10,20)
```

zieht eine Linie vom letzten angesprochenen Punkt zu einem Punkt, der 10 Punkte in X- und 20 Punkte in Y-Richtung entfernt ist.

CIRCLE

Der CIRCLE-Befehl bietet die Möglichkeit, einfach und schnell Bögen, Kreise und Ellipsen zu zeichnen. Die Syntax des Befehls ist:

```
CIRCLE (x,y),r,Farbe,von,bis,Form
```

x,y gibt die Koordinaten der Kreismitte an. Wie üblich gibt 'Farbe' einen Wert zwischen 0 und 3 an.

'von' und 'bis' sind zwei Winkel, die angeben, ab welchem Winkel und bis zu welchem Winkel der Kreis gezeichnet werden soll. Dabei müssen die Winkel im Bogenmaß angegeben werden. Der anzugebende Bereich liegt zwischen 0 und 2π .

Ein negativer Winkel verbindet den Bogen mit dem Kreismittelpunkt.

'Form' gibt an, wie zusammengedrückt bzw. wie gestreckt die Ellipse sein soll. Ein Wert von 1 erzeugt einen Kreis, ein Wert kleiner 1 staucht den Kreis in vertikaler Richtung, ein Wert größer 1 staucht den Kreis in horizontaler Richtung.

PAINT

PAINT wird benutzt, um eine Form auf dem Bildschirm auszufüllen. Je komplexer die zu füllende Form ist, desto mehr Speicher wird der GENIE 16 C benötigen, um diese Form auszufüllen. Gehen Sie also sicher, daß Sie genügend Speicherplatz mit CLEAR reserviert haben.

Die Syntax des-PAINT Befehls ist wie folgt:

PAINT (x,y),Füllfarbe,Randfarbe.

'Füllfarbe' gibt die Farbe an, mit der ausgefüllt werden soll.

'Randfarbe' gibt die Farbe an, bis zu der ausgefüllt werden soll. Es kann also z.B. ein grüner Kreis mit rotem Rand gezeichnet werden.

Hier zwei Beispiele:

```
10 REM PAINT DEMO
20 SCREEN 1,0:KEY OFF:CLS
25 F=1
30 DEF FN R(X)=INT(RND(1)*X)+1
40 LINE (0,0)-(319,199),,B
100 X=30+FNR(260):Y=30+FNR(50):A=FNR(10)
105 IF A<5 THEN A=A/5
110 CIRCLE (X,Y),R,,,,A
120 PAINT (X,Y),FNR(4)-1,3
125 IF RND(1)>.9799999 THEN COLOR 0,F:
    F=ABS(F-1)
130 GOTO 100
```

```
10 SCREEN 1,0:CLS
20 FOR A=0 TO 2 STEP .15
30 CIRCLE (160,100),95,,,,A
40 NEXT A
50 PI=3.1415926
60 R=INT(RND(1)*100):A=RND(1)*2*PI
70 PAINT (160+COS(A)*R,100+SIN(A)*R),
    INT(RND(1)*3)+1
80 GOTO 60
```

DRAW

Mit DRAW kann auf dem Bildschirm gezeichnet werden. Die Syntax ist der des PLAY-Befehls sehr ähnlich. Die Daten werden DRAW in einem String zur Verfügung gestellt.

```
DRAW"M160,100U5R5D5L5"
```

zeichnet z.B. Rechteck auf dem Schirm. Die Syntax der DRAW-Anweisung ist wie folgt:

Un	n Einheiten hoch
Dn	n Einheiten runter
Ln	n Einheiten nach links
Rn	n Einheiten nach rechts
En	n Einheiten nach rechts oben
Fn	n Einheiten nach rechts unten
Gn	n Einheiten nach links und unten
Hn	n Einheiten nach links und oben

Mx,y Bewegung zur Schirmposition (x,y).
Soll eine relative Bewegung erfolgen, sind Vorzeichen vor die Koordinaten zu setzen.

B Wird dieser Befehl vor beliebigen Bewegungsbefehlen angegeben, so erfolgt die Bewegung, es wird jedoch nicht gezeichnet.

N Wird dieser Befehl vor beliebigen Bewegungsbefehlen angegeben, wird die Bewegung ausgeführt, doch hinterher wird an den Ausgangspunkt der Bewegung zurückgekehrt.

- An n kann zwischen 0 und 3 liegen. n besagt, ob die Zeichnung um 0, 90, 180 oder 270 Grad gedreht werden soll.
- Cn Gibt den Farbwert an. 0 bis 3 bei mittlerer Auflösung, 0 oder 1 bei hoher Auflösung.
- Sn Definiert den SCALE-Faktor. Darf zwischen 0 und 255 liegen. Beeinflußt die Größe der benutzten Einheiten und somit die Größe der Zeichnung.

X variable

Führt die angegebene Stringvariable als DRAW Kommando aus. Erlaubt das Zusammenfügen von Zeichnungen aus mehreren Unterstrings.

n kann entweder eine Konstante oder eine Variable sein. Wird n als Variable angegeben, müssen Befehl und Variablenname durch ein '=' getrennt werden.

Beispiele:

```

10 REM PYRAMIDE
20 SCREEN 1,0:KEY OFF:CLS
40 DRAW "bm200,10"
50 OT$="FDGLHUER"
60 FOR S=10 TO 255 STEP 5
70 DRAW "S=S;XOT$;"
80 NEXT S

```

```

10 REM DRAW Tester
20 SCREEN 1,0:KEY OFF:CLS
30 ON ERROR GOTO 150
40 PRINT "DRAW Tester"
50 PRINT:PRINT "Dieses Programm gibt Ihnen
55 PRINT "die Möglichkeit, verschiedene
58 PRINT "DRAW-Kommandos auszuprobieren.
60 PRINT "Sie können auch CLS eingeben,
62 PRINT "um den Bildschirm zu löschen,
65 PRINT "oder END, um das Programm
68 PRINT "zu beenden.
70 PRINT:PRINT "Bitte drücken Sie die
75 PRINT "Leertaste, um anzufangen";
80 IF INPUT$(1) <> " " THEN 80
90 CLS
100 LOCATE 1,1:PRINT SPACE$(40);
110 LOCATE 1,1:LINE INPUT A$
120 IF A$<>"CLS" OR A$="cls" THEN GOTO 130
125 CLS:DRAW "bml60,100":GOTO 100
130 IF A$="END" OR A$="end" THEN END
140 DRAW A$:GOTO 100
150 BEEP:RESUME 100

```

GET und PUT

GET und PUT sind eine leistungsstarke Befehlskombination.

Sie erlauben das 'Abnehmen' eines Teils des Graphikschirms und das Bewegen und Kopieren dieses Teils.

GET (x1,y1)-(x2,y2),Feldvariable

(x1,y1) und (x2,y2) geben zwei diagonale Ecken eines Rechtecks an, dessen Inhalt gespeichert werden soll.

'Feldvariable' ist der Name eines Variablenfeldes, in das die Graphik geschrieben werden soll. Dieses kann von jeder beliebigen numerischen Sorte sein. Verwenden Sie ein Feld aus ganzzahligen Variablen, können Sie eventuell eigene Wege finden, um das Bild in diesem Array zu manipulieren. Die Anzahl der zu reservierenden Bytes läßt sich nach folgender Formel berechnen:

$Y * \text{INT}((2 * X + 7) / 8) + 4$ für mittlere Auflösung.

$Y * \text{INT}((X + 7) / 8) + 4$ für hochauflösende Graphik

Das Ergebnis der Rechnung gibt an, wieviele Integer-Variablen reserviert werden müssen, um den Graphikblock abzuspeichern.

Um das mit GET gespeicherte Bild dann wieder auf den Bildschirm zu bringen, wird der Befehl PUT verwendet. Die Syntax von PUT ist:

`PUT (x,y),Feldvariable,Modus`

x,y sind die Koordinaten, an denen das Bild begonnen werden soll. Sie geben dieselbe Stelle des Rechtecks an, wie xl,yl dies bei GET getan haben, war xl,yl also die obere linke Ecke, so wird das Bild nun mit x,y als oberer linker Ecke gezeichnet. Die Feldvariable gibt an, wo das Bild abgespeichert wurde und Modus kann eine der folgenden Optionen sein:

`PSET,PRESET,XOR,OR,AND`

PSET schreibt das Bild erneut auf den Bildschirm, PRESET invertiert das Bild vollständig (Farbe 0 wird 3 und umgekehrt.)

XOR setzt das neue Bild auf ein altes und invertiert dieses, wenn beide Punkte gesetzt sind. Durch zweimaliges XOR wird also der alte Hintergrund wieder hergestellt. Dadurch kann sehr einfach eine Bewegung simuliert werden, ohne den Hintergrund zu zerstören.

OR schreibt das neue Bild einfach über das alte Bild.

AND setzt die Punkte nur, wenn diese im alten Bild genauso vorhanden waren.

Einige Beispiele:

```
10 REM GET und PUT Demo
20 DIM A$(125):SCREEN 1,0:CLS
50 REM Speichern eines Mondgesichts
60 CIRCLE (15,15),15
70 CIRCLE (10,12),4:CIRCLE (20,12),4
80 CIRCLE (15,13),10,3,3.927,5.498
90 GET (0,0)-(30,30),A$
100 REM Überall hinschieben
110 CLS:FOR I=1 TO 500:NEXT I
120 FOR Y=0 TO 150 STEP 30
125 FOR X=0 TO 270 STEP 30
130 PUT (X,Y),A$,PSET
140 NEXT X,Y
150 REM Pause, dann nochmal
160 CLS:FOR I=1 TO 500:NEXT I
170 FOR Y=0 TO 150 STEP 30
175 FOR X=0 TO 270 STEP 30
180 PUT (X,Y),A$,PSET
190 NEXT X,Y
200 GOTO 100
```

```

10 REM Bewegung mit GET & PUT
20 DIM A%(125):SCREEN 1,0:CLS
30 DEF FN R(X)=5*INT(RND(1)*3-1)
40 REM Speichern eines Mondgesichts
50 CIRCLE (15,15),15
60 CIRCLE (10,12),4:CIRCLE (20,12),4
70 CIRCLE (15,13),10,3,3.927,5.498
80 GET (0,0)-(30,30),A%
90 REM Hintergrund aufbauen
100 CLS
110 FOR I=5 TO 15:LOCATE I,8
115 PRINT "Bewegung mit GET & PUT":NEXT I
120 REM Bewegen des Gesichts
130 X=160:Y=100:REM Start in der Mitte
140 PUT (X,Y),A%,XOR
150 NEWX=X+FNR(5):NEWY=Y+FNR(Y)
160 IF NEWX<0 THEN NEWX=0
170 IF NEWY<0 THEN NEWY=0
180 IF NEWX>290 THEN NEWX=290
190 IF NEWY>160 THEN NEWY=160
200 PUT (X,Y),A%,XOR:X=NEWX:Y=NEWY
210 GOTO 140

```

```

10 REM Nochmehr mit GET & PUT
20 DIM A%(125):SCREEN 1,0:CLS
30 REM Speichern eines Kreises
40 CIRCLE (15,15)
50 GET (0,0)-(30,30),A%
60 REM Versuchen Sie mal verschiedene
62 REM Schrittweiten und verschiedene
64 REM Modi mit PUT
70 CLS
80 FOR X=0 TO 2*3.1415926 STEP .05
90 PUT (X*40,70+SIN(X)*70),A%,PSET
100 NEXT X

```

12.21 GENIE 16 C BASIC-Kurzreferenzliste

Auf den folgenden Seiten finden Sie eine Liste aller Befehle, die für den GENIE 16 C verfügbar sind.

Die Form ist folgende:

Befehl, Angabe

ob es sich um einen Befehl oder eine Funktion handelt, Schreibweise (wenn erforderlich), Bemerkungen (wenn erforderlich).

Alles, was in Nummernzeichen (#) eingeschlossen ist, kann wahlweise eingegeben werden.

ABS-Funktion:

Format: ABS(x)

Berechnet den Absolutwert von x.

ASC-Funktion:

Format: ASC(x\$)

Berechnet den ASCII-Code des ersten Buchstaben von x\$.

ATN-Funktion:

Format: ATN(x)

Berechnet den Arcustangens von x, wobei x im Bogenmaß angegeben sein muß.

AUTO-Befehl:

Format: AUTO #a# #,b#

Automatische Zeilenummerierung. Die Numerierung beginnt mit a (Default = 10) und wird in Schritten von b fortgesetzt (Default=10). Ist eine Zeile schon belegt, erfolgt eine Warnung durch ein '*' hinter der Zeilennummer. Abbruch mit <Ctrl><Break>.

BEEP-Befehl:

Format: BEEP

Läßt einen kurzen 800 Hz Warnton ertönen.

BLOAD-Befehl:

Format: BLOAD <Filename> #,<Adresse>#

Laden eines Maschinensprachefiles in den Speicher. <Filename> gibt den Namen des Files an, <Adresse> die Ladeadresse. Achten Sie darauf, daß das File das BASIC und das DOS nicht überlädt.

BSAVE-Befehl:

Format: BSAVE <Filename>,<Start>,<Länge>

Speichert ein Maschinensprachefile auf Diskette ab. Es werden <Länge> Bytes ab Adresse <Start> aufgezeichnet.

CALL-Befehl:

Format: CALL <Ausdruck> #(<Argument>)#

Ruft ein Maschinenspracheunterprogramm auf. Der angegebene Ausdruck bildet die Startadresse des Unterprogramms und danach kann eine Liste von Argumenten folgen, die an das Unterprogramm übergeben werden sollen.

CDBL-Funktion:

Format: CDBL(x)

Macht aus x eine Zahl doppelter Genauigkeit.

CHAIN-Befehl:

Format: CHAIN #MERGE# <Filespec.>
#, #<Zeilennummer># #, ALL# #,
DELETE <Bereich># #

Ruft ein anderes Programm auf und übergibt die Kontrolle und Variablen. <Zeilennummer> gibt an, wo das neue Programm starten soll. Wird diese Option weggelassen, so startet das neue Programm in seiner ersten Zeile.

ALL gibt an, daß jede Variable des alten Programms an das neue Programm übergeben werden soll. Wird die ALL-Option weggelassen, muß das alte Programm eine Liste der zu übergebenden Variablen (siehe COMMON) enthalten.

Wird die MERGE-Option verwendet, so wird das alte Programm nicht gelöscht. Es können so z.B. Overlays verwendet werden. Das neue Programm kann dann nach Benutzung mit DELETE gelöscht werden. Es ist unbedingt notwendig, daß es sich bei dem neuen Programm um ein ASCII-File handelt, wenn MERGE benutzt werden soll.

CHDIR-Befehl:

Format: CHDIR"pathname"

Wechselt vom aktiven Directory auf ein anderes Directory. Vergleiche auch den DOS-Befehl CHDIR. Die Abkürzung CD ist jedoch nicht zulässig. Ferner muß immer ein pathname angegeben werden.

CHR\$-Funktion:

Format: CHR\$(i)

Erzeugt ein Zeichen mit dem ASCII-Code i.

CINT-Funktion:

Format: CINT(x)

Wandelt die Zahl x in eine ganzzahlige Zahl um.

CIRCLE-Befehl:

Format: CIRCLE (<x-Mitte>,<y-Mitte>),<Radius> #,
<Farbe># #,<Anfang>,<Ende># #,
<Form>#

Zeichnet eine Ellipse mit angegebenem Mittelpunkt und Radius.

Die Angaben <x-Mitte> und <y-Mitte> geben die Koordinaten des Ellipsenmittelpunktes an, der Radius wird durch <Radius> angegeben, die Farbe wird mit <Farbe> angegeben (siehe COLOR-Befehl), <Anfang> und <Ende> sind zwei Winkel im Bogenmaß, die den Anfangs- und den Endpunkt der Ellipse bezeichnen.

<Form> gibt die Streckung der senkrechten Halbachse an. Ist <Form> kleiner als 1, liegt die Ellipse waagrecht, ist <Form> größer als 1, liegt sie senkrecht. <Form> gleich 1 ergibt einen Kreis.

CLEAR-Befehl:

Format: CLEAR #, #<Ausdr.1># #,<Ausdr.2># #

<Ausdr. 1> gibt die höchste Speicheradresse an, die noch vom GENIE 16 C BASIC benutzt werden darf. <Ausdr.2> reserviert Platz für den BASIC-Stapelspeicher. CLEAR schließt alle Files, löscht alle COMMON-Variablen, setzt alle Variablen auf 0, setzt alle Disk-Puffer zurück und löscht alle DEF FN-Definitionen.

CLOSE-Befehl:

Format: CLOSE #fn1,fn2,fn3.....#

Schließt die angegebenen Files. Wird nur CLOSE angegeben, so werden alle Files geschlossen. Bei sequentiellen Ausgaben schreibt CLOSE noch den restlichen Pufferinhalt auf die Diskette.

CLS-Befehl:

Format: CLS

Löscht den Bildschirm.

COLOR-Befehl:

Format:

Das Format ist verschieden je nach Graphikoption.

1. Im Textmode:

```
COLOR #<vordergrund## #,  
      #<Hintergrund>#
```

Dabei gilt:

<Vordergrund> von 0 bis 31

<Hintergrund> von 0 bis 7

2. Bei mittlerer Auflösung:

COLOR Hintergrund, Palette.

Bestimmt die Farbe des Hintergrunds und die Farbe der Palette. Jede der beiden Optionen kann weggelassen werden, wobei der letzte Wert angenommen wird.

COM-Befehl:

Format: COM (n) ON
 COM (n) OFF
 COM (n) STOP

Schaltet die Abfrage auf den Kommunikationskanälen 1 oder 2 ein und aus.

Ist COM (n) ON gegeben worden, überprüft BASIC die Aktivitäten auf dem entsprechenden Kommunikationskanal. Findet Kommunikation statt, wird der entsprechende ON COM-Befehl ausgeführt.

COMMON-Befehl:

Format: COMMON <Variablenliste>

Übergibt Variablen an ein nachgeladenes Programm. Es ist zu empfehlen, den Befehl an den Anfang des Programms zu setzen. Dieselbe Variable darf nicht nochmal in einem COMMON-Befehl verwendet werden. Feldvariablen können durch hinzufügen von '()' an die Variablenliste angegeben werden.

CONT-Befehl:

Format: CONT

Nimmt die Programmausführung wieder auf, wenn ein <BREAK>, ein STOP oder ein END aufgetreten ist. Dieser Befehl wird unmöglich, wenn das Programm während der Unterbrechung editiert wurde.

COS-Funktion:

Format: COS (x)

Berechnet den Cosinus der Zahl x. x muß im Bogenmaß angegeben werden.

CSNG-Funktion:

Format: CSNG (x)

Macht aus der Zahl x eine Zahl einfacher Genauigkeit.

CSRLIN-Funktion:

Format: x=CSRLIN

Speichert die aktuelle Zeilenposition des Cursors in einer numerischen Variable.

CVI, CVS, CVD-Funktionen:

Format: CVI (<2-Byte-String>)
CVS (<4-Byte-String>)
CVD (<8-Byte-String>)

Wandelt Stringwerte in Zahlenwerte um. Zahlenwerte, die sich in einem RANDOM-ACCESS File befanden, müssen mit diesen Funktionen in Zahlenwerte umgewandelt werden, da sie dort in Stringform gespeichert waren.

DATA-Befehl:

Format: DATA (Konstantenliste)

Mit diesem Befehl können Sie Daten für ein Programm zur Verfügung stellen. Diese können dann später mit READ gelesen werden.

DATE\$-Befehl:

Format: DATE\$=<String>

Setzen des aktuellen Datums. <String> muß das Datum in der folgenden Form enthalten:

MM-TT-JJ bzw. MM-TT-JJJJ

Der '-' kann auch durch einen Schrägstrich (/) ersetzt werden.

DATE\$-Funktion:

Format: x\$=DATE\$

Das aktuelle Datum wird in der Form MM-TT-JJJJ in die Variable x\$ eingelesen.

DEF FN-Befehl:

Format: DEF FN Name #(<Parameterliste>)#
= <Funktionsdefinition>

Definiert eine Funktion. "Name" muß ein legaler Variablenname sein.

<Parameterliste> enthält die Variablen, die an die Funktion übergeben werden sollen.
<Funktionsdefinition> enthält das, was die neue Funktion berechnen soll.

DEF INT/SNG/DBL/STR-Befehl:

Format: DEFINT <Buchstabenbereich>
 DEFSNG <Buchstabenbereich>
 DEFDBL <Buchstabenbereich>
 DEFSTR <Buchstabenbereich>

Deklariert alle Variablen, deren Name mit einem Buchstaben aus dem Buchstabenbereich (A-Z oder A,B,D,T-Z etc.) beginnt, als Integer-, Single-, oder Doubleprecision oder als Stringvariablen.

DEF SEG-Befehl:

Format: DEF SEG #=<Adresse>#

Definiert eine neue Segmentadresse, auf die alle folgenden PEEKs, POKEs, CALLs und USR's zugreifen sollen. <Adresse> darf zwischen 0 und 65535 liegen.

DEF USR-Befehl:

Format: DEF USR n=<Adresse>

Definiert die Startadresse eines Maschinenspracheunterprogrammes.
n darf zwischen 0 und 9 liegen, Default ist mit USRn aufgerufen werden.

DELETE-Befehl:

Format: DELETE <von> #-<bis>#

Löscht die angegebenen Zeilennummern.

DIM-Befehl:

Format: DIM <Variable>(a,b,c,...) ,

Definiert ein Variablenfeld mit dem Namen <Variable> und ordnet diesem maximale Indizes (a,b etc.) zu.

DRAW-Befehl:

Format: DRAW <String>

Zeichnet Linien. <String> enthält Informationen über die Zeichenbewegungen. Die möglichen Befehle sind: D, L, R, F, G, H, Mxy, B, N, An, Cn, Sn, X. Diese Optionen sind bereits ausführlich erklärt worden.

EDIT-Befehl:

Format: EDIT <Zeilennummer>

Erlaubt das Editieren der angegebenen Zeile.

END-Befehl:

Format: END

Das Programm wird beendet, alle Files werden geschlossen.

EOF-Funktion:

Format: EOF(<Filenummer>)

Testet, ob im angegebenen File das EOF (End of File) erreicht wurde. Es wird -1 (wahr) zurückgegeben, wenn EOF erreicht ist.

ERASE-Befehl:

Format: ERASE <Variablenliste>

Löscht die angegebenen Variablen aus dem Speicher.

ERR und ERL-Variablen:

Die Variable ERR enthält den Code des letzten aufgetretenen Fehlers, ERL enthält die Zeilennummer, in der der letzte Fehler aufgetreten war.

ERROR-Befehl:

Format: ERROR <n>

Simuliert einen Fehler von BASIC aus. <n> kann eine Zahl zwischen 0 und 255 sein. Der Code ist derselbe, wie bei ERR.

EXP-Funktion:

Format: EXP(x)

Berechnet e (Eulersche Zahl) hoch x . x muß kleiner oder gleich 88.02969 sein.

FIELD-Befehl:

Format: FIELD <Filenummer>,
 <Länge>AS<Stringvar.>...

Reserviert Platz für Variablen in einem File mit wahlfreiem Zugriff. Bevor GET und PUT ausgeführt werden können, muß der Puffer mit FIELD formatiert werden. <Filenummer> gibt an, welcher Puffer formatiert werden soll. <Länge> gibt an, wieviele Bytes den einzelnen Stringvariablen zugeordnet werden sollen. Die Gesamtzahl der zugewiesenen Bytes darf die Recordlänge, die bei OPEN angegeben wurde, nicht überschreiten. Es können mehrere FIELD-Befehle für ein und denselben File ausgeführt werden.

FILES-Befehl:

Format: FILES #"<Filespec.>"#

Entspricht dem DIR-Befehl im DOS. Listet alle Files auf einer Diskette. Die Wildcards '*' und '?' können benutzt werden.

FOR..NEXT-Befehl:

Format: FOR <Variable>=x TO y #STEP z#
 .
 .
 .
 NEXT #<Variable# #,<Variable>....#

Erlaubt das mehrmalige Ausführen eines Befehls bzw. eines Befehlsblocks.

<Variable> wird auf x gesetzt, und der Programmblock (vom FOR bis zum NEXT) wird ausgeführt. Beim NEXT wird überprüft, ob <Variable> größer als y ist. Falls nicht, wird <Variable> um 1 (bzw. um z, wenn angegeben) erhöht und es wird zum FOR zurückgesprungen. Das geht solange weiter, bis <Variable> größer als y wird. Dann wird nicht mehr zurückgesprungen, sondern die Ausführung des Programms mit dem Befehl nach dem NEXT fortgesetzt.

FRE-Funktion:

Format: FRE(x) oder FRE(x\$)

Es kann entweder eine Zahl oder ein String als Argument angegeben werden. Wird eine Zahl angegeben, so kehrt das System mit der Anzahl der noch nicht belegten Bytes zurück. Wird ein String als Argument angegeben, so wird auch die Zahl der freien Speicherbytes angegeben, vorher wird aber der Stringspeicher aufgeräumt, das sogenannte 'garbage collecting'.

GET-Befehl:

Dieser Befehl erfüllt mehrere Funktionen. Bei Files:

Format: GET <Filenummer> #,<Recordnummer>#

Liest ein 'Record' von einem File mit wahlfreiem Zugriff. Wird die zweite Option weggelassen, so wird der nächste Record gelesen. Die bei FIELD genannten Stringvariablen enthalten dann den gelesenen Record.

Mit Graphik:

Format: GET (x1,y1)-(x2,y2),<Feldname>

Bringt einen Teil des Bildschirmes in ein Variablenfeld.

Mit PUT (x1,y1),<Feldname> #,Verb# läßt er sich dann wieder auf den Bildschirm zurückschreiben.

'Verb' kann sein:

- PSET : Schreibt das Bild genau wie es vorher war.
- PRESET: Schreibt das Bild als Negativ.
- AND : Schreibt das Bild über ein schon existierendes Bild. Dabei werden ungleiche Punkte gelöscht.
- OR : Überschreibt das alte Bild, ohne es zu löschen.
- XOR : Setzt nur ungleiche Punkte.

GOSUB-RETURN-Befehle:

Format: GOSUB <Zeilennummer>

.
.
.

Zeilennummer

.
.

RETURN #Zeile#

Ruft ein Unterprogramm ab Zeile <Zeilennummer> auf. Das Unterprogramm muß mit RETURN beendet werden. Wahlweise wird nicht hinter GOSUB, sondern zu einer beliebigen Zeile zurückgekehrt.

GOTO-Befehl:

Format: GOTO <Zeilennummer>

Springt in die Zeile <Zeilennummer> und fährt dort mit der Programmausführung fort.

HEX\$-Funktion:

Format: HEX\$(x)

Wandelt die Zahl x in einen String um, der den Wert von x in hexadezimaler Schreibweise wiedergibt. Ist x nicht ganzzahlig, so wird es zuerst ganzzahlig gemacht.

IF....THEN....ELSE-Befehl:

Format: IF <Bedingung> THEN <Befehl(e)> ELSE
<Befehl(e)>

IF <Bedingung> GOTO <Zeilennummer>
ELSE <Befehl(e)>

Dieser Befehl dient dazu, bedingte Verzweigungen auszuführen. Ist die angegebene Bedingung erfüllt, werden die Befehle nach THEN ausgeführt bzw. es wird zur Zeilennummer nach GOTO gesprungen. Ist die Bedingung nicht erfüllt, werden die Befehle nach ELSE ausgeführt. Wurde kein ELSE angegeben, so wird die Ausführung in der nächsten Zeile wieder aufgenommen.

IF...THEN...ELSE Konstruktionen können ineinander verschachtelt werden. Die Verschachtelungstiefe wird nur durch die maximale Zeilenlänge begrenzt.

INKEY\$-Funktion:

Format: INKEY\$

Enthält das letzte auf der Tastatur gedrückte Zeichen. Wurde keine Taste gedrückt, enthält INKEY\$ den Nullstring ("").

INP-Funktion:

Format: INP(p)

Liest einen Wert vom angegebenen Port p. p muß zwischen 0 und 65535 liegen.

INPUT-Befehl:

Format: INPUT #<"String">;# <Variablenliste>

Liest Daten von der Tastatur ein. Bei einem INPUT hält der Rechner an, gibt ein '?' aus und wartet auf die Eingabe von Daten. Die Daten werden anschließend in den angegebenen Variablen gespeichert. Wird <String> angegeben, so wird dieser vor dem '?' ausgegeben. Soll das '?' bei der Abfrage unterdrückt werden, muß ein Komma anstelle des Semikolons angegeben werden. Die angegebenen Daten müssen zu dem Variablentyp, dem sie zugewiesen werden sollen, passen.

INPUT#-Befehl:

Format: INPUT#<Filenummer>, <Variablenliste>

(Das # muß hier mit eingegeben werden!) Arbeitet genau wie INPUT, liest aber aus einem File anstatt von der Tastatur. Ist das erste gefundene Zeichen ein ", so wird das nächste " als Ende der Zeile interpretiert. Werden numerische Werte eingelesen, so werden alle führenden Leerzeichen, Zeilenvorschübe etc. ignoriert, dasselbe gilt für Stringvariablen. Das erste gefundene Zeichen wird dann als Textanfang angenommen.

INPUT\$-Funktion:

Format: INPUT\$ (x,y)

Liefert einen String von x Zeichen Länge aus dem File y. Wird y weggelassen, so erfolgt die Eingabe von der Tastatur, es erfolgt jedoch keine Ausgabe auf dem Bildschirm. Alle Zeichen werden eingelesen, <Ctrl><Break> bricht die Ausführung von INPUT\$ ab.

INSTR-Funktion:

Format: INSTR (#i,# x\$,y\$).

Sucht nach dem ersten Auftreten von y\$ in x\$. Liefert die Position, wo y\$ in x\$ gefunden wurde. Wird i angegeben, so beginnt die Suche erst bei Position i. Wird y\$ nicht in x\$ gefunden, wird 0 ausgegeben, ist y\$ ein Nullstring, wird 1 bzw. i ausgegeben.

INT-Funktion:

Format: INT(x)

Berechnet die größte ganze Zahl, die kleiner oder gleich x ist.

KEY-Funktion:

Format: KEYn,x\$
KEY LIST
KEY ON
KEY OFF

Weist den Funktionstasten ihre Belegung zu (KEY1,"LIST").

Listet die Belegung der Funktionstasten (KEY LIST).

Schaltet die 25te Zeile an/aus (KEY ON/KEY OFF).

KEY (n)-Befehl:

Format: KEY(n)ON
KEY(n)OFF
KEY(n)STOP

Schaltet die Tastenfalle für die entsprechende Taste ein und aus. Bei KEY(n) ON wird die Tastenfalle für die Taste n aktiviert, bei KEY(n) OFF wird die Tastenfalle für Taste n deaktiviert und die Tasten werden nicht mehr berücksichtigt. KEY(n) STOP verhält sich so wie KEY(n)OFF, aber die in der Zwischenzeit gedrückten Tasten werden gespeichert und beim nächsten Auftreten von KEY(n)ON wird der entsprechende ON KEY-Befehl ausgeführt.

KILL-Befehl:

Format: KILL <Filespec>

Löscht den angegebenen File von der Diskette.

LEFT\$-Funktion:

Format: LEFT\$(x\$,i)

Kehrt mit i Zeichen (von links angefangen) aus x\$ zurück. Ist i größer als die Länge von x\$, wird x\$ ganz zurückgegeben.

LEN-Funktion:

Format: LEN (x\$)

Gibt die Länge von x\$ an.

LET-Befehl:

Format: LET <Variable>=<Konstante>

Weist einer Variablen einen Wert zu. LET selbst ist wahlweise anzugeben, die Angabe ist nur sinnvoll, wenn die Programme kompatibel zu Rechnern sein sollen, die LET verlangen.

LINE-Befehl:

Format: LINE (x1,y1)-(x2,y2),<Farbe>,bf

Zeichnet eine Linie bzw. ein Rechteck. (x1,y1) gibt den Anfangspunkt der Linie an, (x2-y2) den Endpunkt.

Wird <Farbe> weggelassen, so erfolgt das Zeichnen in Farbe 3.

Die b-Option zeichnet ein Rechteck, die bf-Option zeichnet ein Rechteck und füllt dieses aus. Sind die Koordinaten außerhalb des möglichen Bereiches, wird der nächste legale Punkt angenommen.

Die Koordinaten können auch als relative Koordinaten mit der STEP-Option angegeben werden. LINE STEP (10,5) würde z.B. einen Punkt definieren, der die Koordinaten x+10 und y+5 hat. Wird STEP bei der zweiten Koordinate angewendet, bezieht dieses sich auf die Koordinaten des Anfangspunktes.

LINE INPUT-Befehl:

Format: LINE INPUT #"String";#<Stringvariable>

Ermöglicht die Eingabe eines Strings, in dem alle Zeichen zulässig sind. (INPUT benutzt z.B. ',' als Trennzeichen). Sämtliche Zeichen gehen in die angegebene Stringvariable, bis ein CR (CHR\$(13)) gefunden wird.

LINE INPUT#-Befehl:

Format: LINE INPUT #<Filenummer>,<Stringvar.>

Arbeitet genau wie LINE INPUT, liest aber aus einem File.

LIST-Befehl:

Format: LIST <Zeilennummer>

LIST <von>-<bis> <,Filespec>

Listet das gesamte Programm oder Teile von diesem. Wird LIST ohne Optionen angegeben, wird das gesamte Programm gelistet. Wird die <bis>-Option weggelassen, wird ab der <von>-Zeile bis zum Programmende gelistet. Wird die <von>-Option weggelassen, aber die <bis>-Option angegeben, wird vom Anfang bis zur <bis>-Zeile gelistet. Werden <von>- und <bis>-Option angegeben, wird dieser Bereich gelistet. Wird <Filespec> angegeben, wird das erzeugte Listing in diesem File abgespeichert.

LLIST-Befehl:

Format: LLIST <von>-<bis>

Arbeitet genau wie LIST, gibt aber auf den Drucker aus. Es wird ein 132 Zeichen/Zeile Drucker vorausgesetzt.

LOAD-Befehl:

Format: LOAD <Filespec>,R

Lädt ein Programm von Diskette bzw. Cassette in den Speicher. Die R-Option startet das Programm nach dem Laden automatisch. LOAD schließt alle offenen Files und löscht alle Variablen und das im Speicher befindliche Programm. Wird die R-Option angegeben, werden alle Files offengelassen.

LOC-Funktion:

Format: LOC (<Filenummer>)

Bei der Benutzung von Dateien mit wahlfreiem Zugriff ergibt LOC die Nummer des letzten geschriebenen bzw. gelesenen Records. Bei einem Kommunikationsfile dient LOC(x) zur Überprüfung, ob noch Zeichen in der Warteschlange auf Abfrage warten.

LOCATE-Befehl:

Format: LOCATE <Zeile>,<Spalte>,<Cursor>,
<Von>,<Bis>

Bewegt den Cursor auf die durch <Zeile>,<Spalte> angegebene Position.

Ist <Cursor>=1, ist der Cursor sichtbar, ist <Cursor>=0, ist der Cursor unsichtbar. <Von> gibt die Cursor-Startzeile, <Bis> die Cursor-Endzeile an. Beide Werte dürfen zwischen 0 und 31 liegen. Werden Werte weggelassen, so wird der entsprechende Parameter nicht geändert.

LOF-Funktion:

Format: LOF(<Filenummer>)

Ergibt die Länge des Files in Bytes.

LOG-Funktion:

Format: LOG (x)

Berechnet den natürlichen Logarithmus von x.
x muß größer als 0 sein.

LPOS-Funktion:

Format: LPOS(x)

Ergibt die aktuelle Stellung des Druckkopfes im Druckerpuffer. Bei der Benutzung von Disk-BASIC ist x die Nummer des Druckers.

LPRINT und LPRINT USING-Befehle:

Format: LPRINT <Ausdr.>

bzw.

LPRINT USING<Formatstring>;<Ausdr. >

Druckt Daten auf einem angeschlossenen Drucker. <Ausdr.> sind die auszugebenden Daten (Strings, Numerische Variablen etc.). <Formatstring> gibt das Format an, in dem gedruckt werden soll. Dazu werden besondere Zeichen verwendet.

Druckpositionen:

Die Position eines Ausdruckes auf dem Papier wird durch die Zeichensetzung in der Ausdrucksliste bestimmt. BASIC unterteilt eine Zeile in Druckzonen je 14 Zeichen. Werden die Ausdrücke durch Kommas getrennt, wird bei jedem neuen Ausdruck eine neue Druckzone begonnen. So ist tabelliertes Ausdrucken möglich. Befindet sich am Ende der Ausdrucksliste ein Komma oder ein Semikolon, beginnt der nachfolgende Ausdruck in derselben Zeile.

Besondere Formatierzeichen bei Benutzung der USING-Option:

Ein Formatstring besteht grundsätzlich aus zwei Teilen:

- a) Normalen Zeichen, in die die zu formatierenden Daten eingesetzt werden sollen.
- b) Spezielle Formatierungszeichen, die angeben, wie die Daten formatiert werden sollen.

Der Formatstring

"Dies ist das Ergebnis: ####.##"

würde den Text 'Dies ist das Ergebnis: ', gefolgt von der angegebenen Zahl im angegebenen Format ausgeben.

Die im folgenden genannten Formatierzeichenkombinationen können also nicht in Ihren Texten auftreten. Wollen Sie ein solches Zeichen dennoch in einem Text ausgeben, muß vor dem Zeichen ein Unterstreichungszeichen '_' angegeben werden.

1. Formatierung von Strings

"!" druckt nur das erste Zeichen jedes Strings.

Ein String aus n Leerzeichen, eingeschlossen in inverse Schrägstriche, gibt an, daß $2+n$ Zeichen des Strings ausgegeben werden sollen. Ist der String länger, werden alle folgenden Zeichen ignoriert. Ist der String kürzer als $2+n$, wird er ausgedruckt, das Feld wird dann bis zur Länge $2+n$ mit Leerzeichen aufgefüllt.

"&" druckt jeden String so aus, wie er ist.

2. Formatierung von numerischen Ausdrücken:

Jedes "#" definiert eine Ziffer.#### gibt also an, daß jede Zahl vierstellig und rechtsbündig auszugeben ist. Hat die Zahl weniger als vier Ziffern, wird links mit Leerzeichen aufgefüllt.

Ein Formatstring wie "##.###" druckt eine Zahl immer auf drei Nachkommastellen gerundet. Sind die Vorkommastellen alle 0, wird trotzdem eine Null vor dem Komma ausgegeben.

"+####" gibt an, daß vor jeder Zahl ihr Vorzeichen angegeben werden soll, entsprechend bedeutet "####+", daß das Vorzeichen hinter der Zahl stehen soll.

"####-" druckt hinter jede negative Zahl ein Minuszeichen.

"**####" füllt das Formatfeld anstatt mit Leerzeichen mit '*' auf. Die beiden '*' in der Formatdefinition halten zusätzlich Platz für zwei weitere Ziffern frei.

"\$\$####" bewirkt die Ausgabe eines '\$'-Zeichens vor der ersten Stelle jeder Zahl. Negative Zahlen können nur mit "\$\$####-" ausgegeben werden, damit '-' und '\$' sich nicht beeinflussen. Die '\$\$'-Option hält Platz für eine weitere Ziffer und für das '\$'-Zeichen frei.

"**\$####" kombiniert den Effekt der '**'-Option mit dem der '\$\$'-Option, d.h. alle führenden Zeichen werden als '*' ausgegeben, dann folgt ein '\$' und die Zahl. '**\$' hält drei weitere Stellen frei, eine davon wird vom '\$'-Zeichen belegt. Wird eine negative Zahl ausgegeben werden, wird das Minuszeichen links vom '\$' ausgegeben.

"####,####" bewirkt, daß die Tausenderstellen jeweils durch ein Komma getrennt werden (10000.57 = 10,000.57). Das Komma hält eine weitere Stelle für eine Ziffer frei.

"####----" stellt jede Zahl in Exponentialform dar. Die vier Zeichen halten den Platz für die Darstellung des Exponenten in der Form 'E+xx' bzw. 'E-xx' frei.

Wird eine auszugebende Zahl größer als das angegebene Feld, wird vor der Zahl ein '%' ausgegeben. Wird sie durch Rundung zu groß, wird ',%' ausgegeben.

LSET und RSET-Befehle:

Format: LSET <Stringvariable>=<Stringvar.>
RSET <Stringvariable>=<Stringvar.>

Bringt Daten aus einem String in einen Stringblock, der mit FIELD im Pufferspeicher definiert worden ist. Die erste <Stringvariable> ist der definierte Block, die zweite ist die Variable, die die Daten enthält. LSET setzt den String an den linken, RSET an den rechten Rand. Ist der String zu lang, werden die restlichen Zeichen von rechts weggelassen. Numerische Werte müssen zuerst zu Strings konvertiert werden.

MERGE-Befehl:

Format: MERGE <Filename>

Lädt ein Programm zusätzlich in den Speicher. Das Programm muß in ASCII-Form vorliegen. Programmzeilen des alten Programms, die in beiden Programmen vorkommen, werden beim Nachladen überschrieben. Nach einem MERGE kehrt BASIC auf die Kommandoebene zurück.

MID\$-Befehl:

Format: MID\$(<String Ausdr. 1>,n,m)
=<String Ausdr. 2>

Ersetzt einen Teil eines Strings durch einen anderen. m und n sind ganzzahlig, <String Ausdr. 1> und <String Ausdr. 2> sind Strings. Die Zeichen des ersten Strings werden durch die Zeichen des zweiten Strings ersetzt. m gibt die Zahl der zu ersetzenden Zeichen an, wird es weggelassen, werden alle Zeichen bis zum Ende des ersten Strings ersetzt. n gibt das erste zu ersetzende Zeichen im ersten String an.

MID\$-Funktion:

Format: x\$=MID\$(y\$,n,m)

Erzeugt einen Teilstring von y\$. n gibt an, an welcher Position der Teilstring beginnen soll, m gibt die Länge des Teilstrings an.

MKDIR-Befehl:

Format: MKDIR "pathname"

Erzeugt ein neues Directory. Vergleiche auch DOS-Befehl MKDIR. Die Abkürzung MD kann nicht verwendet werden.

MKI\$,MK\$\$,MKD\$-Funktionen:

Format: MKI\$(<Ganzzahliger Ausdruck>)
MK\$\$(<Ausdr. einfacher Genauigkeit>)
MKD\$(<Ausdr. doppelter Genauigkeit>)

Macht aus Zahlenwerten Strings. Diese können dann in den Puffer für Dateien mit wahlfreiem Zugriff gebracht werden.

NAME-Befehl:

Format: NAME <Filename 1> AS <Filename 2>

Umbenennen eines Diskfiles. Filename 1 muß existieren, Filename 2 darf nicht existieren. <Filename 2> darf keine Laufwerksangabe enthalten, da er immer auf demselben Laufwerk entstehen muß, auf dem sich <Filename 1> befindet.

NEW-Befehl:

Format: NEW

Löscht das Programm im Speicher, setzt alle Variablen auf 0, schließt alle Files und schaltet das Tracing aus.

OCT\$-Funktion:

Format: OCT\$(x)

Gibt einen String zurück, der den Wert von x im Oktalsystem enthält. x wird vor der Berechnung ganzzahlig gemacht.

ON COM-Befehl:

Format: ON COM(n) GOSUB <Zeilennummer>

Führt ein Unterprogramm ab <Zeilennummer> aus, wenn Signale auf dem betreffenden Kommunikationskanal auftreten. Das Unterprogramm wird entsprechend den COM ON Befehlen ausgeführt. (Siehe COM ON, COM OFF etc.) Verhält sich ähnlich wie der ON KEY-Befehl.

ON ERROR GOTO <Zeilennummer>-Befehl:

Springt in die angegebene Zeile, wenn ein Fehler auftritt. Fehler können so abgefangen werden.

ON ... GOSUB und ON ... GOTO-Befehle:

Format: ON <Ausdruck> GOSUB <Liste von Zeilennummern>

ON <Ausdruck> GOTO <Liste von Zeilennummern>

Springt in eine der in der <Liste von Zeilennummern> angegebenen Zeilen. Dieser Sprung ist abhängig vom Wert von <Ausdruck>. Ist <Ausdruck>=1, so springt der Rechner zur ersten Zeilennummer, ist <Ausdruck>=2, zur zweiten ...

Ist <Ausdruck> negativ oder größer als 255, so tritt ein ILLEGAL FUNCTION CALL Fehler auf.

ON KEY-Befehl:

Format: ON KEY (n) GOSUB <Zeilennummer>

Der ON KEY-Befehl fragt ab, ob eine der Funktionstasten bzw. eine der Cursorstasten gedrückt wurde. Wird Taste (n) gedrückt, erfolgt ein Sprung in ein Unterprogramm ab Zeile <Zeilennummer>.

ON PEN-Befehl:

Format: ON PEN GOSUB <Zeilennummer>

Gibt die erste Zeile eines Unterprogramms an, das aufgerufen werden soll, wenn der Lightpen aktiviert wurde. Eine <Zeilennummer> = 0 schaltet die ON PEN-Funktion ab.

Der ON PEN-Befehl wird nur dann ausgeführt, wenn ein PEN ON-Befehl gegeben wurde. Dieser Befehl aktiviert die Abfrage des Lightpens. PEN OFF verhindert, daß der Lightpen abgefragt wird. Eine weitere Möglichkeit besteht bei Angabe eines PEN STOP-Befehls. Bei PEN STOP wird der Lightpen nicht abgefragt, eine Aktivierung wird jedoch bemerkt und gespeichert. Beim nächsten PEN ON wird sofort in die Behandlungsroutine gesprungen. Befindet sich der Rechner in der Behandlungsroutine, erfolgt automatisch ein PEN STOP, damit keine rekursiven Sprünge erfolgen können. Nach dem RETURN erfolgt automatisch ein PEN ON. Wenn mit RETURN <Zeilennummer> zurückgesprungen werden soll, ist zu beachten, daß der Rechner sich innerhalb einer FOR...NEXT-Schleife befunden haben könnte, als der Lightpen aktiviert wurde. Es können also FOR WITHOUT NEXT-Fehler o.ä. auftreten.

ON STRIG (n)-Befehl:

Format: ON STRIG (n) GOSUB <Zeilennummer>

Gibt die Zeile an, in die gesprungen werden soll, wenn der Feuerknopf des Joysticks gedrückt wurde.

Der ON STRIG-Befehl wird nur dann ausgeführt, wenn ein STRIG ON-Befehl gegeben wurde. Dieser Befehl aktiviert die Abfrage des Joysticks. STRIG OFF verhindert, daß der Joystick abgefragt wird. Eine weitere Möglichkeit besteht bei Angabe eines STRIG STOP-Befehls. Bei STRIG STOP wird der Joystick nicht abgefragt, eine eventuelle Aktivierung wird jedoch bemerkt und gespeichert. Beim nächsten STRIG ON wird sofort in die Behandlungsroutine gesprungen.

Befindet sich der Rechner in der Behandlungsroutine, erfolgt automatisch ein STRIG STOP, damit keine rekursiven Sprünge erfolgen können. Nach dem RETURN erfolgt automatisch ein STRIG ON. Wenn mit RETURN <Zeilennummer> zurückgesprungen werden soll, ist zu beachten, daß der Rechner sich innerhalb einer FOR..NEXT Schleife befunden haben könnte, als der Feuerknopf gedrückt wurde. Es können also FOR WITHOUT NEXT-Fehler o.ä. auftreten.

OPEN-Befehl:

Format: 1. OPEN <Modus>, #<Filenummer>, <Filespec>, <Record-Länge>
2. OPEN Filespec FOR <Modus> AS # <Filenummer> LEN=<Record-Länge>

Öffnen eines Files.

<Modus> kann eine der folgenden Optionen sein:

1.	2.	Bedeutung

O	OUTPUT	Sequentielle Ausgaben an das File
I	INPUT	Sequentielle Eingaben vom File
R	<keine Angabe>	Wahlfreies Ein- und Auslesen
A	APPEND	Sequentielles Anhängen an ein File

<Filenummer> gibt die Nummer des zu öffnenden Files an. <Record-Länge> gibt an, wie lang jeder Record bei einem File mit wahlfreiem Zugriff sein soll. Diese Angabe ist bei Files mit sequentiellm Zugriff nicht erlaubt.

Ein Diskfile muß geOPEND werden, bevor jegliche Ein- und Ausgaben erfolgen können.

OPEN COM-Befehl:

Format: OPEN "COMn:<Baud>,<Parität>,<Data>,
<stop>,RS,CSn,DSn,CDn,BIN,ASC,LF" AS
#<Gerätenummer>

Initialisiert einen Kommunikationskanal.

Die Parameter bedeuten:

<Baud> ist die Baudrate (Bits pro Sekunde).

<Parität> ist entweder N (keine), O (ungerade) oder E (gerade)

<Data> gibt die Nummer der Datenbits an: 5,6,7 oder 8

<stop> gibt die Anzahl der Stoppbits an: 1 oder 2

RS unterdrückt die Ausgabe von RTS (Request To Send)

CS(n) kontrolliert CTS (Clear To Send), wobei n die Zeit in ms angibt, nach der ein Fehler gemeldet wird.

DS(n) kontrolliert DSR (Data Set Ready). Ein 'Device timeout error' wird ausgegeben, wenn kein DSR erkannt wird.

CD(n) kontrolliert Carrier Detect

<Gerätenummer> ist die Nummer des zu öffnenden Gerätes.

<Baud><Parität><Data><stop> müssen in dieser Reihenfolge angegeben werden. Die Eingabe der restlichen Parameter kann in beliebiger Reihenfolge erfolgen.

LF gibt an, daß ein nach jedem CR ein LF zu senden ist.

BIN gibt an, daß der Kanal im Binärmodus zu öffnen ist, d.h die Daten werden ohne Veränderung übertragen. Der Kanal kann nur durch CLOSE geschlossen werden.

OPTION BASE-Befehl:

Format: OPTION BASE n

Gibt den kleinsten möglichen Index für Feldvariablen an. n kann entweder 0 oder 1 sein.

OUT i,j-Befehl:

Schickt ein Byte j an den Systemport i.

PAINT-Befehl:

Format: PAINT (x1,y1),Farbe,Rand

Dieser Befehl wurde schon ausführlich erklärt.

PEEK-Funktion:

Format: PEEK(i)

Gibt den Inhalt (0 - 255) der Speicherzelle i (0 - 65535) zurück.

PEN-Befehl/Funktion:

Format: PEN ON,
PEN OFF,
PEN STOP,
x=PEN(n)

PEN ON, OFF und STOP verhalten sich entsprechend zu den KEY PEN u.ä. Befehlen. Die PEN-Funktion x=Pen(n) ($1 \leq n \leq 9$) gibt folgende Daten aus:

n	Wert	Bedeutung
0	0	Pen wurde seit der letzten Abfrage nicht mehr auf den Bildschirm gesetzt.
0	-1	Pen wurde seit der letzten Abfrage auf den Bildschirm gesetzt. 1
1	x	Gibt die x-Koordinate der Position an, auf die der Pen gesetzt ist.
2	y	Gibt die y-Koordinate der Position an, auf die der Pen gesetzt ist.
3	0	Pen ist nicht auf dem Bildschirm.
3	-1	Pen ist auf den Bildschirm gesetzt.

- | | | |
|---|---|---|
| 4 | x | Gibt die letzte x-Koordinate an, auf die der Pen gesetzt war. |
| 5 | y | Gibt die letzte y-Koordinate an, auf die der Pen gesetzt war. |
| 6 | y | Gibt die momentane Bildschirmzeile an, in der der Pen steht. |
| 7 | x | Gibt die momentane Bildschirmspalte an, in der der Pen steht. |
| 8 | y | Gibt die letzte bekannte Zeile an, in der der Pen stand. |
| 9 | x | Gibt die letzte bekannte Spalte an, in der der Pen stand. |

PLAY-Befehl:

Format: PLAY <Stringausdruck>

Spielen einer Notensequenz. Dieser Befehl wurde bereits ausführlich erklärt.

POINT-Funktion:

Format: POINT (x,y)

Liest die Farbe des angegebenen Punktes auf dem Schirm. Befindet sich der angegebene Punkt außerhalb der legalen Grenzen, wird -1 zurückgegeben.

POKE-Befehl:

Format: POKE i,j

Schreibt den Wert j (zwischen 0 und 255) in Speicherzelle i des Speichers.

POS-Funktion:

Format: POS (i)

Gibt die aktuelle Spaltenposition des Cursors an. Die Spalte ganz links hat die Nummer 1.

PRESET-Befehl:

Format: PRESET (x,y),Farbe

Setzen eines bestimmten Punktes auf die Hintergrundfarbe. Dieser Befehl wurde bereits ausführlich erklärt.

PRINT-Befehl:

Format: PRINT <Ausdrucksliste>

Der Hauptausgabebefehl in BASIC. Die Teile der <Ausdrucksliste> können Variablen, Strings etc. sein.

Wird keine <Ausdrucksliste> angegeben, wird eine leere Zeile gedruckt. Ein angehängtes Semikolon verhindert die Ausgabe eines anschließenden Zeilenvorschubs.

PRINT USING-Befehl:

Format: PRINT <Formatstring>;<Ausdruckliste>

Formatierte Ausgabe von Daten. Das Format ist prinzipiell dasselbe wie beim Befehl LPRINT USING. Schlagen Sie bitte dort nach.

PRINT#- und PRINT USING#-Befehle:

Format: PRINT#<Filenummer>
USING<Formatstring>;
Ausdruckliste

Ausgabe von Daten wie bei PRINT, aber in den File <Filenummer>, eine sequentielle Datei.

PSET-Befehl:

Format: PSET (x,y),Farbe

Setzen eines bestimmten Punktes auf dem Bildschirm. Dieser Befehl wurde schon ausführlich erklärt.

PUT-Befehl:

Format: PUT #<Filenummer>,<Recordnummer>
oder
PUT (x1,y1),<Feldname>,<Modus>

Dieser Befehl kann auf zwei verschiedene Arten angewendet werden.

Bei der Verarbeitung von Dateien mit wahlfreiem Zugriff schreibt er einen Record vom Puffer auf die Diskette:

PUT #<Filenummer>,<Record-Nummer>

Der Puffer wird also in den File <Filenummer> als Record <Recordnummer> geschrieben.

PUT kann auch im Zusammenhang mit Graphik eingesetzt werden. Die Syntax ist:

PUT (x1,y1),<Feldname>,<Modus>

Dieser Befehl wurde schon erklärt.

RANDOMIZE-Befehl:

Format: RANDOMIZE <Numerischer Ausdruck>

Setzt den Zufallsgenerator auf einen neuen Anfangswert. Der <numerische Ausdruck> kann zwischen -32768 und 32767 liegen. Wird immer der gleiche Ausdruck verwendet, werden auch immer dieselben Zufallszahlen erzeugt.

READ-Befehl:

Format: READ <Variablenliste>

Liest unter DATA abgelegte Konstanten in die angegebenen Variablen ein. Die Zahl der vorhandenen Daten muß ausreichen, um alle angegebenen Variablen zu füllen. Sind zu wenig Daten vorhanden, wird ein OUT OF DATA-Fehler ausgegeben. Siehe auch RESTORE.

REM-Befehl:

Format: REM <Kommentar>

Einfügen von Kommentaren in ein Programm. Nach dem REM wird der Rest der Zeile ignoriert.

RMDIR-Befehl:

Format: RMDIR"pathname"

Dieser Befehl entfernt ein leeres Directory. Siehe auch DOS-Befehl RMDIR. Die Abkürzung RD ist jedoch im BASIC nicht zulässig.

RENUM-Befehl:

Format: RENUM <a>,,<Zeilenabstand>

Alle Zeilen ab werden neu nummeriert. Die Numerierung beginnt mit Zeilennummer <a>, die Zeilennummern werden jeweils um <Zeilenabstand> erhöht.

Defaults sind: <a> = 10, = erste Programmzeile, <Zeilenabstand> = 10

RENUM ändert alle Sprungbefehle innerhalb des Programms (GOTO, GOSUB, ON x GOTO etc.), so daß sie weiterhin richtig springen. Das renumerierte Programm ist also weiterhin ausführbar.

RESET-Befehl:

Format: RESET

Schließt alle offenen Files auf allen angeschlossenen Laufwerken.

RESTORE-Befehl:

Format: RESTORE <Zeilennummer>

Setzt den DATA-Zeiger (er zeigt auf den nächsten zu lesenden DATA-Wert) auf den Beginn der ersten DATA-Zeile im Programm. Wird <Zeilennummer> angegeben, wird der Zeiger auf den Beginn der DATA-Zeile in der angegebenen Zeile gesetzt.

RESUME-Befehl:

Format: RESUME,
RESUME NEXT,
RESUME <Zeilennummer>

Springt aus einer Fehlerbehandlungsroutine zurück. RESUME kehrt zu dem Befehl zurück, bei dem der Fehler aufgetreten ist. RESUME NEXT kehrt zum nächsten Befehl zurück und RESUME <Zeilennummer> kehrt in die angegebene Zeile zurück.

RETURN-Befehl:

Siehe GOSUB ... RETURN

RIGHT\$-Funktion:

Format: RIGHT\$(x\$,i)

Erzeugt einen String, der die letzten i Zeichen von x\$ enthält.

RND-Funktion:

Format: RND (x)

Berechnet eine Pseudo-Zufallszahl zwischen 0 und 1.

RUN-Befehl:

Format: RUN <Zeilennummer>
 RUN <Filespec>,R

Ausführen eines Programmes. <Zeilennummer> gibt die Zeile an, bei der gestartet werden soll. <Filespec> ist der Name eines Files, das geladen und gestartet werden soll. RUN schließt alle offenen Files. Wird ,R angegeben, so werden offene Files nicht geschlossen.

SAVE-Befehl:

Format: SAVE <Filespec>,A oder ,P

Abspeichern eines BASIC Programmes. Die A-Option speichert das Programm im ASCII-Format ab, die P-Option schützt das Programm, so daß es nach dem Laden nicht mehr gelistet oder editiert werden kann.

SCREEN-Funktion:

Format: x=SCREEN (Zeile,Spalte,z)

Gibt den ASCII-Wert des Zeichens bei (Zeile,Spalte) aus. Wird zusätzlich z (ungleich 0) angegeben, so wird statt des ASCII-Codes die Farbe dieser Position ausgegeben.

SCREEN-Befehl:

Format: SCREEN <Modus>,<Burst>,<Aseite>,<Vseite>

Dieser Befehl wurde schon ausführlich besprochen.

SGN-Funktion:

Format: SGN(x)

Berechnet das Vorzeichen von x. Das Ergebnis ist -1, falls x negativ ist, 0, falls x Null ist und 1, wenn x positiv ist.

SHELL-Befehl:

Format: SHELL"DOS-Befehl"

SHELL ermöglicht vom BASIC aus DOS-Befehle aufzurufen, bzw. auszuführen.

SHELL"CHKDSK" ruft beispielsweise des DOS-Befehl CHKDSK auf.

Nach Rückkehr ins BASIC wird der Bildschirm gelöscht.

SHELL funktioniert nur einwandfrei mit Befehlen die die Speicheraufteilung nicht verändern, so daß bei der Benutzung Vorsicht geboten ist.

SIN-Funktion:

Format: SIN(x)

Berechnet den Sinuswert von x, wobei x im Bogenmaß angegeben werden muß.

SOUND-Befehl:

Format: SOUND <Frequenz>,<Dauer>

Dieser Befehl wurde schon in Abschnitt 6.20 ausführlich erläutert.

SPACE\$-Funktion:

Format: SPACE\$(x)

Erzeugt einen String aus x Leerzeichen. x muß zwischen 0 und 255 liegen.

SPC-Funktion:

Format: SPC(i)

Wird als Argument von PRINT, LPRINT u.ä. benutzt. Stellt den Cursor bzw. den Drucckopf um i Zeichen weiter.

SQR-Funktion:

Format: SQR (i)

Berechnet die Quadratwurzel aus i.

STICK-Funktion:

Format: x=STICK (i)

Abfrage der Joysticks:

i=0 x-Koordinate von Joystick A
i=1 y-Koordinate von Joystick A
i=2 x-Koordinate von Joystick B
i=3 y-Koordinate von Joystick B

STOP-Befehl:

Format: STOP

Wird bei der Programmausführung ein STOP erreicht, verhält sich der Rechner so, als wäre die <Break>-Taste gedrückt worden. Die Programmausführung kann mit CONT fortgesetzt werden.

STR\$-Funktion:

Format: STR\$(x)

Gibt eine String-Darstellung des Wertes x aus.

STRIG-Befehl/Funktion:

Format: STRIG ON,
STRIG OFF,
STRIG STOP,
x=STRIG (n).

Die ersten drei Möglichkeiten verhalten sich entsprechend zu KEY bzw. PEN.

x=STRIG(n) gibt aus:

n=0 Feuerknopf wurde seit der letzten Abfrage gedrückt.

n=1 Feuerknopf im Moment gedrückt ?

n=2 Wie n=0, aber Knopf B

n=3 Wie n=1, aber Knopf B

STRING\$-Funktion:

Format: STRING\$(i,j) oder STRING\$(i,x\$)

Erzeugt einen String der Länge i, dessen Zeichen den ASCII-Code j haben bzw. das erste Zeichen von x\$ sind.

SWAP-Befehl:

Format: SWAP <Variable1>,<Variable2>

Vertauscht den Inhalt der angegebenen Variablen miteinander.

SYSTEM-Befehl:

Format: SYSTEM

Schließt alle offenen Files und kehrt ins DOS zurück.

TAB-Funktion:

Format: TAB(i)

Bewegt den Druckkopf bzw. den Cursor in Spalte i. Die Spalte am linken Rand ist Nummer 1, die größte zulässige Spaltennummer ist 255.

TAN-Funktion:

Format: TAN(x)

Berechnet den Tangens von x, x muß im Bogenmaß angegeben werden.

TIME\$-Variable:

Format: TIME\$

Enthält die aktuelle Uhrzeit.

TRON / TROFF-Befehle:

Ein- und Ausschalten des TRACE-Modus. Mit Hilfe des TRACE-Modus kann die Fehlersuche in einem Programm sehr vereinfacht werden, da die Nummer jeder durchlaufenen Zeile auf dem Bildschirm ausgegeben wird.

TRON schaltet den TRACE-Modus ein, TROFF schaltet ihn wieder aus.

USR-Funktion:

Format: USR <n>(<Wert>)

Aufruf eines Maschinenspracheunterprogrammes. <n> ist die Nummer des aufzurufenden Unterprogramms, welches vorher mit DEF USR definiert worden ist. <Wert> ist ein Wert, der an das Unterprogramm zu übergeben ist.

VAL-Funktion:

Format: VAL(x\$)

Wandelt einen String, der eine Zahl enthält, in einen numerischen Wert um.

VARPTR-Funktion:

Format: 1. VARPTR(<Variablenname>)
2. VARPTR(#<Filenummer>)

1. Ergibt die Adresse des ersten Bytes von <Variablenname> im Speicher. Wird eine Stringvariable benutzt, wird die Adresse des ersten Bytes des Stringdescriptors zurückgegeben.

2. Gibt die Adresse des Ein- Ausgabepuffers zurück, der zu <Filenummer> gehört.

VARPTR\$-Funktion:

Format: VARPTR\$(<Variablenname>)

Keht mit der Adresse der Variablen in Stringform zurück. Das erste Byte enthält den Variablentyp, das zweite Byte das LSB und das dritte Byte das MSB der Variablenadresse.

WAIT-Befehl:

Format: WAIT <Portnummer>,i,j

Liest den Inhalt von Port <Portnummer>. Dieser Inhalt wird mit j geXORed und mit i geANDed. Ist das Ergebnis 0, wird der Befehl erneut ausgeführt. Ist das Ergebnis nicht 0, wird mit der Programmausführung fortgefahren.

WHILE..WEND-Befehle:

Format: WHILE <Ausdruck.>

.
. .
. .
. .
WEND

Führt die Befehle und Funktionen zwischen WHILE und WEND solange aus, bis <Ausdruck> den logischen Wert 0 hat. Tritt dieser Fall ein, wird nach dem WEND fortgefahren.

WIDTH-Befehl:

Format: 1. WIDTH LPRINT <Breite>
2. WIDTH <Filenummer>,<Breite>
3. WIDTH <Gerät>,<Breite>

Stellt die Breite des Ausdrucks auf dem Schirm bzw. auf dem Drucker ein. <Breite> darf zwischen 0 und 255 liegen. <Filenummer> darf zwischen 1 und 15 liegen. <Gerät> ist ein String, der den Namen des Gerätes enthält, das benutzt werden soll. Wird bei 1. LPRINT angegeben, bezieht sich <Breite> auf den Drucker. Wird es nicht mit angegeben, so kann <Breite> entweder 40 oder 80 sein und bezieht sich auf den Bildschirm.

WRITE-Befehl:

Format: WRITE <Liste von Ausdrücken>

Gibt Daten auf dem Schirm aus. Der Befehl verhält sich genau wie PRINT.

WRITE#-Befehl:

Format: WRITE#<Filenummer>,
<Liste von Ausdrücken>

Verhält sich genau wie WRITE, gibt aber in den File <Filenummer> aus.

12.22 BASIC-Fehlermeldungen.

Diese Liste enthält alle Fehlermeldungen, die auftreten können und deren Fehlercode, den ERR nach Auftreten dieses Fehlers enthält.

NEXT without FOR (1)

Ein NEXT wurde gefunden, ohne daß ein passendes FOR vorhanden ist.

FOR...NEXT-Schleifen dürfen zwar verschachtelt, aber nicht über Kreuz beendet werden.

SYNTAX ERROR (2)

Schreibfehler.

RETURN WITHOUT GOSUB (3)

Es wurde ein RETURN-Befehl erreicht, ohne das vorher ein GOSUB ausgeführt wurde.

OUT OF DATA (4)

Ein READ wurde versucht, ohne daß sich genug Daten in DATAzeilen befanden.

ILLEGAL FUNCTION CALL (5)

Die angegebenen Parameter sind nicht erlaubt.

OVERFLOW (6)

Eine Zahl ist zu groß, um in eine Variable zu passen. Ganzzahlige Variablen dürfen nicht größer als 32767 werden.

OUT OF MEMORY (7)

Es ist nicht genug Speicher für das Programm vorhanden. Es wurden zuviele Felder definiert oder versucht, zu komplexe Formen mit dem PAINT-Befehl auszumalen.

UNDEFINED LINE NUMBER (8)

Es wurde versucht, in eine Zeile zu springen, die nicht existiert.

SUBSCRIPT OUT OF RANGE (9)

Beim Zugriff auf ein Element einer Feldvariablen wurde ein größerer Index angegeben, als mit DIM definiert wurde.

DUPLICATE DEFINITION (10)

Es wurde versucht, ein Variablenfeld mehrmals zu definieren.

DIVISION BY ZERO (11)

Bei einer Division war der Divisor 0.

ILLEGAL DIRECT (12)

Der Befehl, den Sie im direkten Eingabemodus ausführen wollten, kann nur im Programm ausgeführt werden (INPUT).

TYPE MISMATCH (13)

Es wurde versucht, einer Variable Daten eines falschen Typs zuzuordnen.

OUT OF STRING SPACE (14)

Die Strings, die Sie im Moment benutzen, belegen mehr Speicher, als mit CLEAR reserviert wurde. Reservieren Sie mehr Speicher.

STRING TOO LONG (15)

Es wurde versucht, einem String mehr als 255 Zeichen zuzuordnen.

STRING FORMULA TOO COMPLEX (16)

Eine Stringberechnung ist zu kompliziert, um auf einmal durchgeführt zu werden.

CAN'T CONTINUE (17)

Nach einem BREAK wurde editiert o.ä. Es ist nun unmöglich, CONT auszuführen.

UNDEFINED USER FUNCTION (18)

Eine Funktion muß erst mit DEF FN definiert werden, bevor sie mit FN aufrufbar ist.

NO RESUME (19)

Aus einer Fehlerbehandlungsroutine muß mit RESUME zurückgesprungen werden.

RESUME WITHOUT ERROR (20)

Ein RESUME-Befehl wurde erreicht, ohne das ein Fehler aufgetreten war.

MISSING OPERAND (22)

Bei einer Berechnung fehlt ein Operand

LINE BUFFER OVERFLOW (23)

Es wurde eine Zeile eingegeben, die länger als 240 Zeichen ist.

DEVICE TIMEOUT (24)

Es wurde keine Meldung von einem Peripheriegerät empfangen. Überprüfen Sie das entsprechende Gerät.

DEVICE FAULT (25)

Fehler bei einem Peripheriegerät.

FOR WITHOUT NEXT (26)

Es wurde ein FOR, jedoch kein NEXT gefunden.

OUT OF PAPER (27)

Es befindet sich kein Papier im Drucker.

WHILE WITHOUT WEND (29)

Es wurde ein WHILE, jedoch kein WEND gefunden.

WEND WITHOUT WHILE (30)

Es wurde ein WEND gefunden, ohne das ein WHILE vorhanden ist.

FIELD OVERFLOW (50)

Ein Puffer für Files mit wahlfreiem Zugriff ist zu klein.

INTERNAL ERROR (51)

Ein interner Fehler ist aufgetreten. Überprüfen Sie alle Anschlüsse, finden Sie keinen Fehler, wenden Sie sich an Ihren GENIE-Händler.

BAD FILE NUMBER (52)

Eine Filenummer ist ungültig.

FILE NOT FOUND (53)

Es wurde versucht, auf einen nicht existenten File zuzugreifen.

BAD FILE MODE (54)

Sie haben z.B. versucht, einen sequentiellen File wie einen File mit wahlfreiem Zugriff zu behandeln.

FILE ALREADY OPEN (55)

Es wurde versucht, einen File, der schon offen ist, nochmals zu eröffnen.

DEVICE I/O Error (57)

Ein Kommunikationbaustein, z.B. das RS232-Interface hat einen Fehler in der Datenübertragung gemeldet.

FILE ALREADY EXISTS (58)

Sie haben versucht, einen File mit NAME umzubenennen, dessen neuer Name schon auf der Diskette vorhanden ist.

DISK FULL (61)

Es ist kein Platz mehr auf der angesprochenen Diskette.

INPUT PAST END (62)

Es wurde versucht, nach dem Ende des Files weiterzulesen.

BAD RECORD NUMBER (63)

GET und PUT erlauben nur die Anwendung von Recordnummern zwischen 0 und 32767.

BAD FILE NAME (64)

Es wurde versucht, ein File mit einem ungültigen Namen aufzurufen.

DIRECT STATEMENT IN FILE (66)

Das soeben geladenen Programm enthält Teile, die keine BASIC-Zeilen sind.

TOO MANY FILES (67)

Es ist unmöglich, noch weitere Files zu öffnen.

DEVICE UNAVAILABLE (68)

Das angesprochene Gerät ist entweder nicht existent oder kann momentan nicht angesprochen werden.

COMMUNICATIONS BUFFER OVERFLOW (69)

Ein Kommunikationsgerät lieferte Daten, obwohl der Datenpuffer schon voll war. Wenn Sie die Daten nicht schnell genug aus dem Datenpuffer auslesen können, müssen Sie eine niedrigere Übertragungsgeschwindigkeit wählen.

DISK WRITE PROTECT (70)

Die Diskette, auf die geschrieben werden sollte, ist schreibgeschützt.

DISK NOT READY (71)

Das angesprochene Diskettenlaufwerk ist nicht betriebsbereit.

DISK MEDIA ERROR (72)

Die Diskette, mit der Sie arbeiten, ist wahrscheinlich beschädigt. Kopieren Sie alle noch lesbaren Files auf eine neue Diskette und mustern Sie die beschädigte Diskette aus.

UNPRINTABLE ERROR (??)

Ein Fehler ist aufgetreten, der keinen legalen Errorcode hat. Tritt meistens in Verbindung mit dem ERROR-Befehl auf.

13. Erweiterungsmöglichkeiten.

=====
Das DOS und BASIC sind die fundamentalen Bestandteile der Kommunikation mit dem GENIE 16 C. In diesem Kapitel jedoch soll kurz auf die Erweiterungsmöglichkeiten des GENIE 16 C eingegangen werden.

Die interessanteste Möglichkeit, den Rechner zu erweitern, besteht in einer Erweiterung auf Softwarebasis. Der GENIE 16 C ist in der Lage, eine große Palette an Textverarbeitungsprogrammen, Kalkulationsprogrammen etc. zu betreiben. Diese verwandeln den Rechner in ein universelles Werkzeug. Mit einer großen Softwarebibliothek kann das System universell eingesetzt werden. Weitere interessante Möglichkeiten bestehen in einer Erweiterung auf Hardwarebasis.

13.1 Der Drucker

Der Drucker wird wohl das erste Gerät sein, das zusätzlich zur Rechneinheit angeschafft wird. Es ist sinnlos, ein Textverarbeitungssystem zu betreiben, ohne die Texte auch ausdrucken zu können. Bei der Wahl des Druckers ist darauf zu achten, welcher Drucker benötigt wird. Ein Matrixdrucker z.B. druckt sehr schnell, hat aber meistens eine schlechtere Druckqualität, so daß dieser kaum zur Korrespondenz benutzt werden kann, von Ausnahmen natürlich abgesehen. Der Matrixdrucker ist also mehr ein Gerät für Leute, die Programme und Daten ausgeben wollen. Ein Typenraddrucker dagegen ist teuer und langsam, er druckt aber in einer optisch hervorragenden Druckqualität.

Drucker mit IBM-Zeichensatz und Centronics-Parallel-Schnittstelle können problemlos am GENIE 16 C betrieben werden.

13.2 Plotter

Ein Plotter ist eine Art kleine Zeichenmaschine:

ein Arm führt einen (oder mehrere) Stift(e) über ein Papier und zeichnet. Die Vorteile eines Plotters sind die gute Graphik und die Möglichkeit, verschiedene Farben zu benutzen.

Plotter sind in den verschiedensten Preis- und Qualitätsklassen erhältlich. Besonders ist beim Kauf eines Plotters auf seine Zeichengenauigkeit Wert zu legen. Wenn ein Plotter über eine Centronics-Parallel-Schnittstelle verfügt, ist er problemlos an den GENIE 16 C anschließbar.

13.3 Monitore

Bei Monitoren müssen Sie sich im klaren darüber sein, ob Sie einen Farbmonitor oder einen S/W Monitor benötigen.

An Farbmonitoren gibt es verschiedene Ausführungen. Das beste Bild erhalten Sie mit einem RGB-Monitor, doch die sind naturgemäß auch die teuersten.

Am GENIE 16 C sind Anschlüsse für NTSC-, RGB- und monochrome Monitore.

13.4 Zusätzlicher Speicher

Der GENIE 16 C verfügt bekanntlich über 256K internen Speicher. Dieser Speicher kann im System auf der Platine auf 640 K aufgestockt werden.

13.5 Hard Disk

Eine Harddisk arbeitet prinzipiell ähnlich wie eine Floppy, sie bleibt jedoch im Gehäuse, ist schneller und hat wesentlich mehr Speicherkapazität. Mit einer Harddisk lassen sich durchaus 10-30 Millionen Bytes ansprechen.

Anhang A

=====

Diskettenfehler

Tritt ein Fehler beim Lesen oder Schreiben mit einer Diskette auf, so gibt das DOS eine Fehlermeldung aus:

<yyy> ERROR WHILE <READING oder WRITING> ON
DRIVE x

Die Meldung <yyy> ist dann eine der folgenden:

WRITE PROTECT
BAD UNIT
NOT READY
BAD COMMAND
DATA
BAD CALL FORMAT
SEEK
NON-DOS DISK
SECTOR NOT FOUND
NO PAPER
WRITE FAULT
READ FAULT
DISK

Das DOS wartet dann auf eine Eingabe:

A Beendet laufende Programm.

I Ignoriert den Fehler.

R Erneuter Versuch. Kann z.B. eingegeben werden, wenn der Fehler inzwischen behoben wurde.

Eine andere Fehlermeldung, die auftreten kann, ist:

FILE ALLOCATION TABLE BAD FOR DRIVE x

In der FILE ALLOCATION TABLE befindet sich ein Zeiger auf einen nicht existenten Block. Die Diskette wurde entweder falsch oder gar nicht formatiert.

Konfigurieren des Systems

In vielen Fällen muß das DOS genau auf ein System abgestimmt werden.

Das Konfigurationsfile CONFIG.SYS erlaubt es Ihnen, das DOS genau auf Ihr System abzustimmen.

Im File CONFIG.SYS sind Daten abgelegt, die beim Booten des DOS die Konfiguration des Systems angeben.

Der Boot-Prozess erfolgt:

1. Der Boot-Sektor der Diskette wird gelesen. Dieser Sektor enthält ein kurzes Programm, welches den restlichen Teil des DOS lädt.
2. Das DOS und BIOS werden geladen.
3. Das BIOS nimmt eine Vielzahl von Initialisierungen vor.
4. Eine Systemroutine lädt das File CONFIG.SYS und anhand der darin enthaltenen Daten Systemtreiber etc.
5. Der Kommandoprozessor wird gestartet. Das Booten ist damit beendet.

Änderungen im File CONFIG.SYS:

Befindet sich kein entsprechendes File auf der Systemdiskette, kann dieses mit EDLIN erzeugt werden. Die Befehle, die dafür verwendet werden können, sind:

`BUFFERS=<Zahl>`

Dies ist die Zahl der Sektorenpuffer, die die Systemliste enthalten. Ist hier noch kein Eintrag vorhanden, empfiehlt es sich, 10 anzugeben.

`FILES=<Zahl>`

`<Zahl>` ist die Zahl an offenen Files, auf die XENIX Systemaufrufe zugreifen können. Ist noch kein Eintrag vorhanden, empfiehlt sich die Eingabe von 10.

`DEVICE=<Filename>`

Installiert den Gerätetreiber mit dem Namen `<Filename>` in der Systemliste.

`BREAK=<ON oder OFF>`

Wird ON angegeben (der Standard ist OFF), fragt der Tastaturreiber laufend `<Ctrl>-<C>` ab. Es besteht dadurch eine zusätzliche Möglichkeit, ein Programm abubrechen.

`SHELL=<Filename>`

Die Ausführung des Kommandoprocessors (SHELL) beginnt bei `<Filename>`.

Eine typische Konfiguration würde so aussehen:

```
BUFFERS=10  
FILES=10  
BREAK=ON  
SHELL=A:\BIN\COMMAND.COM A:\BIN /P
```

Die letzte Zeile des Konfigurationsfiles sagt dem DOS, daß sich der Kommandoprozessor COMMAND.COM in der Directory \BIN befindet. Das anschließende A:\BIN teilt COMMAND.COM mit, wo zu suchen ist, wenn Teile von COMMAND.COM nachgeladen werden müssen. /P teilt mit, daß es sich bei COMMAND.COM um das erste Programm handelt, was geladen wird, so daß der DOS EXIT Befehl richtig zurückspringen kann.

Anhang C

Die ASCII-Codes des GENIE 16 C

Graphikset

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	ç	ü	é	è	ä	à	á	ç	ø	ë	ë	ï	ÿ	ì	ñ	á
9	é	z	ñ	ó	ö	ó	û	ü	ý	ø	ü	ç	è	¥	ñ	í
A	á	í	ó	ú	ñ	ñ	é	é	í	í	ç	ç	í	«	»	
B	§	§	§	l	l	l	§	§	§	§	§	§	§	§	§	§
C	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l
D	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l	l
E	α	β	γ	δ	ε	ϕ	μ	τ	ξ	θ	Ω	δ	ω	φ	€	Π
F	≡	±	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫

Graphikset

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
1	▶	4	‡	!!	¶	§	∞	±	†	‡	†	†	†	†	†	†
2		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	P	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	

Anhang D

Steckerbelegungen:

Tastaturanschluß:

Pin	Belegung
1	Tastatur Clock-Signal
2	Tastaturdaten
3	reserviert
4	Masse
5	+5 Volt

Paralleler Druckerport:

Pin	Belegung
1	Strobe-Signal. 0 wenn aktiv
2-9	Daten. Pin 2 ist Bit 0, Pin 9 ist Bit 7
10	Acknowledge. 0 wenn aktiv
11	Busy. 1 wenn aktiv
12	Paper end. 1 wenn aktiv
13	Printer selected. 1 wenn aktiv
14	Print LF after CR. 0 wenn aktiv
15	Printer error. 0 wenn aktiv
16	Initialize Printer. 0 wenn aktiv
17	Select in. 0 wenn aktiv
18	Masse
25	Masse

RGB-Monitoranschluß:

Pin	Belegung
1	Masse
2	Masse
3	Rot
4	Grün
5	Blau
6	Intensität
7	nicht belegt
8	Horizontal sync
9	Vertical sync

Alle Signale sind TTL-kompatibel.

Lichtgriffelanschluß:

Pin	Belegung
1	Lichtgriffel-Signal
3	Lichtgriffel Indikator
4	Masse
5	+5 Volt
6	+12 Volt

Serielle RS 232 Schnittstelle:

Pin	Belegung
2	TX Data
3	RX Data
4	RTS
5	CLR TO SND
6	DATA SET RDY
7	GND
8	CARRIER DET -
20	DTR

Anhang E

=====
Das GENIE 16 C BIOS:

Das BIOS gibt dem System Kontrolle über die I/O- Geräte auf der Systemplatine.

Die BIOS-Routine gibt dem Programmierer eine festgelegte Verbindung zur Systemhardware. Der größte Teil des BIOS ist mit der Kommunikation beschäftigt - mit Bildschirm, Tastatur, Drucker, usw.

Zusätzlich erfüllt das BIOS eine Reihe von Systemfunktionen, wie eine Echtzeituhr, Selbsttest, Speichertest usw.

Obwohl das BIOS von nahezu jeder auf dem GENIE 16 C verfügbaren Sprache aufgerufen werden kann, ist sein Einsatz jedoch nur einem erfahrenen Programmierer zgedacht, der über das System genaue Kenntnisse besitzt.

Da das BIOS nur sehr wenige Fehlerüberprüfungen durchführt, ist es sehr leicht möglich, das System "aufzuhängen", wenn die Routinen des BIOS mißbraucht werden.

Der Gebrauch des BIOS:

Das BIOS besteht aus einem Set von Unterprogrammen. Jedes dieser Unterprogramme kann verschiedene Aufgaben erfüllen.

Um diese Funktionen auszuführen, laden Sie die entsprechenden Daten in die Register des 8088. Durch einen Software-Interrupt wird dann die entsprechende Routine aufgerufen.

Setzen wir z.B. voraus, daß wir ein Zeichen auf dem parallelen Druckerport geben wollen. Der Parallelausgang wird mit INT 17H angesteuert. INT 17H unterstützt drei verschiedene Funktionen, die durch das Laden vom Register AH mit 0, 1 oder 2 ausgewählt werden. Wird AH mit 0 geladen, geht das Zeichen auf den Parallelport. Der Code in Assembler sieht also folgendermaßen aus:

```
MOV AH,0 ; Auswahl von Funktion 0
MOV AL,13 ; Code für Zeilenvorschub
MOV DX,0 ; Auswahl des ersten Druckers
INT 17H ; Aufruf des Unterprogramms
; Der Druckerstatus wird ausgegeben
```

Die BIOS-Routinen erhalten den Inhalt sämtlicher Register außer AX und den Flags. Bitte nehmen Sie Notiz, daß manche Routinen Statuswerte in die anderen Register übergeben.

<u>Interruptnummer</u>	<u>Funktion</u>
10H	Video I/O
11H	Geräteüberprüfung
12H	Ermitteln des Speicherplatzes
13H	Disketten I/O
14H	RS 232 - Kommunikation
15H	Cassettenroutinen
16H	Tastaturroutinen
17H	Paralleler Drucker
1AH	Echtzeituhr

Für Video I/O Interrupt 10H:

AH=0	Einstellmodus
AL=0	40 x 25 S/W-Text
AL=1	40 x 25 Farbtext
AL=2	80 x 25 S/W-Text
AL=3	80 x 25 Farbtext
AL=4	320x200 Farbgrafik
AL=5	320x200 S/W-Grafik
AL=6	640x200 S/W-Grafik
AH=1	Art des Cursors definieren
CL	Bits 0-4 Endzeile für Cursor
CH	Bits 0-4 Startzeile für Cursor

AH=2 Cursorposition einstellen

DL Spaltennummer

DH Zeilennummer

BL Seitennummer (0 für Grafik)

AH=3 Lesen der Cursorposition

BH Seitennummer (0 für Grafik)

Zurückgegeben werden:

DL Spalte

DH Zeile

CL Bits 0-4 Endzeile für Cursor

CH Bits 0-4 Startzeile für Cursor

AH=4 Lesen der Lightpenposition

BH Seitennummer (0 für Grafik)

Zurückgegeben werden:

AH Lightpenschalterstatus
(0-nicht gedrückt, 1-gedrückt)

DL Spalte

DH Zeile

CH Rasterlinie (0-199)

BX Pixelspalte (0-319 oder 0-639)

AH=5 Verändern der Seite im Textmodus

AL Seitennummer
(0-7 im 40 Zeichenmodus,
0-3 im 80 Zeichenmodus)

AH=6 Scrollen der Bildschirmseite nach oben

AL Anzahl der Zeilen,
0 bedeutet das gesamte Fenster
CL Spalte der obersten, linken Ecke
CH Zeile der obersten, linken Ecke
DL Spalte der untersten, rechten Ecke
DH Zeile der untersten, rechten Ecke
BH Bildschirmattribut auf unbenutzten Zeilen

AH=7 Scrollen der Bildschirmseite nach unten

AL Anzahl der Zeilen,
0 bedeutet das gesamte Fenster
CL Spalte der obersten, linken Ecke
CH Zeile der obersten, linken Ecke
DL Spalte der untersten, rechten Ecke
DH Zeile der untersten, rechten Ecke
BH Bildschirmattribut auf unbenutzten Zeilen

AH=8 Ausgabe des Zeichens/Attributes auf dem Cursor

BH Seite

Zurückgegeben wird:

AL Zeichen der Cursorposition
AH Attribute des Zeichens auf der Cursorposition

AH=9 Ausgabe eines Zeichens auf der
 Cursorposition

BH Bildschirmseite
 CX Zahl der zu schreibenden Zeichen
 AL ASCII-Code des zu schreibenden
 Zeichens
 BL Attribute des zu schreibenden
 Zeichens

AH=10 Drucken eines Buchstabens auf der
 Cursorposition

BH Bildschirmseite
 CX Zahl der zu schreibenden Zeichen
 AL ASCII-Code des zu schreibenden
 Zeichens

AH=11 Setzen der Farbpalette

BH Farbpalette
 BL Farbwert

AH=12 Setzen eines Punktes

AL Farbwert. Wenn Bit 7=1, dann
 Schirm=Schirm XOR AL
 CX Spaltennummer
 DX Zeilennummer

AH=13 Lesen eines Punktes

CX Spaltennummer

DX Zeilennummer

Zurückgegeben wird:

AL gelesener Punkt

AH=14 Schreibe einen Buchstaben

AL ASCII-Code des Buchstabens

BL Vordergrundfarbe im Grafikmodus

BH Seite im Textmodus

AH=15 Abfragen des Videostatus

Zurückgegeben wird:

AL Augenblicklicher Schirm-Modus

AH Zahl der Buchstabenspalten auf dem Schirm

BH Augenblickliche Bildschirmseite

Gerätecheck INT 11H:

Zurückgegeben wird:

AX an das System angeschlossenen Geräte

Emitteln der Speichergröße INT 12H:

Zurückgegeben wird:

AX Zahl der 1K Blocks

Tastatur INT 16H:

AH=0 nächsten Buchstaben von der Tastatur
 lesen

Zurückgegeben werden:

AL Buchstabe
AH Scan Code

AH=1 ist ein Buchstabe verfügbar?

Zurückgegeben werden:

Z Flag = 1 Kein Code verfügbar
Z Flag = 0 Code verfügbar
wenn Z=0 AX=Buchstabe im Puffer

AH=2 Übergabe des aktuellen SHIFT-Status

Zurückgegeben wird:

AL Aktueller SHIFT-Status

Paralleler Drucker INT 17H:

AL zu druckender Buchstabe
DX Druckernummer 0, 1 oder 2

Zurückgegeben wird:

AH=1 falls Zeitfehler

AH=1 Initialisiere den Druckerport

DX Druckernummer

Zurückgegeben wird:

AH Druckerstatus

AH=2 Abfragen des Druckerstatus

DX Druckernummer

Zurückgegeben wird:

AH Druckerstatus

Echtzeituhr INT 1AH:

AH=0 Lesen der Uhr

Zurückgegeben wird:

CX:DX Uhr

AL=0 falls die 24 Uhr nicht überschritten
wurden.

AH=1 Uhr stellen

CX:DX Uhrzeit

Anhang F

Zusätzliche Programme auf Ihrem AT-DOS:

ANSI.SYS:

ANSI.SYS ist ein Bildschirmverwaltungsprogramm, das Ihnen, einmal geladen, eine Vielzahl von zusätzlichen Ausgabemöglichkeiten bietet. Laden Sie ANSI.SYS, indem Sie ein File CONFIG.SYS (siehe oben) erzeugen, das die Zeile DEVICE=ANSI.SYS enthält. So ist nach jedem Booten der ANSI.SYS Treiber geladen.

Die Möglichkeiten von ANSI.SYS:

Alle im folgenden angegebenen Kontrollsequenzen funktionieren nur, wenn Sie über die normalen Zeichenausgaberoutinen des DOS ausgegeben werden.

Schreibweise: Der Defaultwert für anzugebende Zahlen ist 0.
meint eine Zahl.

Für alle Kontrollsequenzen gilt:
ESC bezeichnet den ASCII-Wert 27 und nicht die Zeichenfolge E S C.

1. Cursorsteuerung

Cursor positionieren:

ESC [#;#H oder ESC [#;#f

Stellt den Cursor auf die angegebene Position. Der erste Parameter gibt die Zeile (Default: oberste Zeile), der zweite gibt die Spalte (Default: linke Spalte) an.

Cursor aufwärts:

ESC [#A

Bewegt den Cursor n Zeilen nach oben (Default: 1). Steht der Cursor schon in der obersten Zeile, wird die Kontrollsequenz ignoriert.

Cursor abwärts:

ESC [#B

Bewegt den Cursor n Zeilen nach unten (Default: 1). Steht der Cursor bereits in der untersten Zeile, wird die Sequenz ignoriert.

Cursor rechts:

ESC [#C

Bewegt den Cursor n Spalten nach rechts (Default: 1). Steht der Cursor schon ganz rechts, wird die Sequenz ignoriert.

Cursor links:

ESC [#D

Bewegt den Cursor n Spalten nach links (Default: 1). Steht der Cursor bereits ganz links, wird die Sequenz ignoriert.

Cursorposition anfordern:

ESC [6n

Nach Ausgabe von ESC [6n gibt ANSI.SYS die momentane Cursorposition an. Dies geschieht, indem ANSI.SYS folgende Tastendrücke simuliert:

ESC zeile;spalterR

Cursorposition speichern:

ESC [s

Die momentane Cursorposition wird gespeichert.

Cursorposition wiederherstellen:

ESC [u

Der Cursor wird an die zuletzt gespeicherte Position gesetzt.

2. Löschen des Bildschirms

Gesamten Bildschirm löschen:

ESC [2J

Löscht den Bildschirm und positioniert den Cursor auf Zeile 1, Spalte 1 (Home).

Zeile löschen:

ESC [k

Löscht den Rest der Zeile ab der Cursorposition.

3. Bildschirmausgabe-Modus einstellen

Zeichenausgabe definieren:

ESC [#;#;...;#m

Parameter Bedeutung

0	Alle Attribute abschalten (Zeichen weiß auf schwarz)
1	Zeichen hell darstellen
5	Zeichen blinkend darstellen
7	Zeichen invers darstellen
8	Zeichen nicht darstellen
30	Schwarzer Vordergrund
31	Roter Vordergrund
32	Grüner Vordergrund
33	Gelber Vordergrund
34	Blauer Vordergrund
35	Magenta Vordergrund
36	Zyaner Vordergrund

- 37 Weißer Vordergrund
- 40 Schwarzer Hintergrund
- 41 Roter Hintergrund
- 42 Grüner Hintergrund
- 43 Gelber Hintergrund
- 44 Blauer Hintergrund
- 45 Magenta Hintergrund
- 46 Zyaner Hintergrund
- 47 Weißer Hintergrund

Bildschirmmodus definieren:

- ESC [=#h
- ESC [=h
- ESC [?7h

Parameter Bedeutung



- 0 40 x 25 schwarz-weiß
- 1 40 x 25 Farbe
- 2 80 x 25 schwarz-weiß
- 3 80 x 25 Farbe
- 4 320 x 200 Farbgrafik
- 5 320 x 200 Schwarz-weiß-Grafik
- 6 640 x 200 Schwarz-weiß-Grafik
- 7 Eingaben, die länger als eine Bildschirmschirmzeile sind, werden in der nächsten Zeile fortgeführt.

Bildschirmmodus rücksetzen:

```
ESC [=#I  
ESC [=I  
ESC [=0I  
ESC [?7I
```

Die Parameter dieses Befehls entsprechen denen des vorherigen, Parameter 7 schaltet den Modus aber ab, statt an.

3. Neubelegung von Tasten:

```
Kontrollsequenz: ESC [#,#;..;#p  
                  ESC ["string";p  
                  ESC [#;"string";##;  
                  "string";#p  
                  oder jede andere Kombina-  
                  tion von ASCII-Werten oder  
                  Strings
```

Diese Kontrollsequenz ermöglicht es, bestimmte Tasten umzudefinieren, sodaß sie andere Zeichen oder ganze Strings erzeugen. Der erste ASCII-Wert beschreibt die Taste, die umdefiniert werden soll, alle anderen Werte geben den String an, den diese Taste erzeugen soll. Ist der erste Wert jedoch Null, so ergeben die ersten beiden Werte die neuzubelegende Taste an (Funktionstasten etc.).

Beispiele:

Die Tasten Q und q sollen ein A und a erzeugen und umgekehrt:

```
ESC [65;81p    A wird Q
ESC [97;113p   a wird q
ESC [81;65p    Q wird A
ESC [113;97p   q wird a
```

Die Funktionstaste F10 (Code: 0 68) soll das Kommando DIR, gefolgt von einem <ENTER> (Chr\$(13)), erzeugen:

```
ESC [0;68;"DIR";13p
```

GERMAN.COM

=====

GERMAN.COM ist ein Tastaturtreiber, der Ihr GENIE 16 C an die deutsche Tastatur anpaßt. Geladen wird GERMAN.COM im DOS durch Eingabe von

GERMAN <ENTER>

Sie können nun über die Tastatur die Umlaute und das ß eingeben, außerdem entsprechen die ausgegebenen Zeichen jetzt den auf den Tastenköpfen aufgedruckten Bezeichnungen. Alle engl. Sonderzeichen sind weiterhin mit der Alt-Taste erreichbar.

Drücken Sie die Alt-Taste und halten Sie sie gedrückt, während Sie auf dem numerischen Tastenblock den ASCII-Wert des gewünschten Zeichens eingeben. Sobald Sie die Alt-Taste loslassen, erscheint das gewünschte Zeichen auf dem Bildschirm.

Die eckigen Klammern, der inverse Schrägstrich und der Klammeraffe sind auch weiterhin auf der Tastatur aufgedruckt. Sie erreichen diese Zeichen, indem die Tastenkombination CTRL, Alt und die entsprechende Taste betätigt wird.

Eine weitere Änderung ergibt sich bei der Benutzung des Hochkommas. Wenn Sie diese Taste drücken, tut sich zunächst einmal nichts. Folgt als nächstes jedoch ein Zeichen wie e, a oder ähnl., so erscheint das entsprechende Zeichen aus einem anderen internationalen Zeichensatz (z.B. é o. ù).

Drücken Sie nach dem Hochkomma die Leertaste, erscheint das Hochkomma. Jede andere Taste erzeugt einen Ton und gibt das Zeichen aus.

LINK.COM

=====

Der Linker ist zunächst völlig nutzlos für Sie. Er wird erst dann wichtig, wenn Sie den Compiler, den Assembler oder andere Programme von MICROSOFT benutzen. Alle diese Programme benutzen den Linker, deshalb ist es sinnvoll, ihn auf der DOS-Diskette zu haben.

Anhang G

===== Definieren von Zeichen im Grafik-Modus: -----

In den beiden Grafikauflösungen des GENIE 16 C werden die ASCII-Zeichen von 128 bis 255 nicht mehr aus dem Zeichengenerator ausgelesen, sondern aus einem anderen Speicherbereich, der sich frei festlegen läßt. Beim Einschalten wird dieser Speicherbereich auf einen willkürlichen Wert gesetzt, so daß im Grafikmodus nicht mehr die Umlaute und die gewohnten Sonderzeichen dargestellt werden. Diesem kann abgeholfen werden, wenn man die Zeichen, die gebraucht werden, selbst definiert. Es können alle ASCII-Codes von 128 bis 255 benutzt werden. Da jedes Zeichen in einer 8x8 Matrix dargestellt wird, werden pro Zeichen acht Bytes benötigt. Hierbei stellt jedes Byte eine Punktrasterzeile mit je acht Punkten dar. Das höchste Bit zeigt auf die linke Spalte.

Um nun die Punktmatrixtabelle für insgesamt 128 Zeichen abzuspeichern zu können, braucht man genau 1024 Bytes (1kByte), die aus dem Arbeitsspeicher zur Verfügung gestellt werden müssen. Die Wahl dieses Speicherbereiches ist, abgesehen von dem Platz den das Betriebssystem und das BASIC belegt, völlig frei.

Der Beginn dieser Tabelle wird in vier aufeinanderfolgenden Bytes im Speicher vermerkt:

Seg.	Adr.	Byte
0000:	007C	LSB der Tabellenadresse
0000:	007D	MSB der Tabellenadresse
0000:	007E	LSB des Tabellensegments
0000:	007F	MSB des Tabellensegments

Da das Segment null ist, sind die vier Bytes des Tabellenzeigers als Absolutadressen des Speichers anzusehen.

Sehr praktisch ist die Tatsache, daß der Tabellenpointer innerhalb eines Programms verändert werden kann. Man kann sich daher mehrere Tabellen mit bis zu 128 definierten Zeichen anlegen und hat somit Zugriff auf "fast beliebige" Mengen von eigenen Zeichen, da beim "Umbiegen" des Tabellenpointers der aktuelle Bildschirminhalt ja nicht verloren geht.

Nachfolgend ist ein kurzes BASIC-Programm aufgelistet, das im Grafikmodus die Umlaute und das 'ß' definiert. Die Tabelle legt sich ans obere Ende des RAM-Speicherplatzes. Die Zeichen können bei einer deutschen Tastatur und aktiviertem GERMAN.COM direkt eingegeben werden. In allen anderen Fällen bekommt man die Zeichen mit "PRINT CHR\$(n)" oder durch Festhalten der 'Alt'-Taste und nachfolgender Eingabe des ASCII-Codes auf dem Numeric-Block. Der jeweils erste Wert in den DATA-Zeilen beinhaltet den betreffenden ASCII-Code des definierten Zeichens.

Programmlisting zur Umlautdefinition:

```
10 DATA 129,CC,CC,CCCC,7E00
20 DATA 132,CC00,780C,7CCC,7E00
30 DATA 142,C638,6CC6,FEC6,C600
40 DATA 148,CC,78,CCCC,7800
50 DATA 153,C638,6CC6,C66C,3800
60 DATA 154,C600,C6C6,C6C6,7C00
70 DATA 225,78,CCD8,CCD8,C0C0
100 RESTORE 10
110 CLEAR,-20764:DEFINT I-M:ADR=-20764
115 REM Bei 256K und mehr muß -20764 durch -1024 ersetzt werden
120 K=ADR:GOSUB 410:DEF SEG=0:POKE 124,K:POKE 125,J
130 POKE 126,PEEK(1296):POKE 127,PEEK(1297):DEF SEG
140 FOR L=1 TO 7:READ I:I=(I-128)*8+ADR:FOR M=0 TO 7 STEP 2
150 GOSUB 400:POKE M+I,J:POKE M+I+1,K:NEXT:NEXT
160 SCREEN 1,0:CLS:PRINT CHR$(129)" "CHR$(132)" "CHR$(142)" ";
170 PRINT CHR$(148)" "CHR$(153)" "CHR$(154)" "CHR$(225)
180 PRINT:END
400 READ S$:K=VAL("&H"+S$)
410 J=INT(K/256)AND 255:K=K AND 255:RETURN:REM J=LSB, K=MSB
```

Bei 256 KByte Speicher stimmt in der Zeile der angegebene Wert von -1024 in jedem Falle.

Bei 128 KByte RAM ändert sich die Speichergröße je nachdem ob Sie den GERMAN.COM-Tastaturtreiber oder auch ANSI.SYS usw. benutzen.

Der Wert -20764 stimmt bei Benutzung von GERMAN.COM in Verbindung mit BASIC. Bei zusätzlich aktiven .COM- oder .SYS-Programmen muß der Wert um etwa die Filelänge dieser Programme verringert werden. (Steigt der Wert hierbei unter -32768, so muß 65536 hinzuaddiert werden).

Anmerkung: Die PEEK's in Zeile 130 holen sich den Wert des Default-Segments des BASIC. Dieser Wert wird in den Adressen 0000:0510H und 0000:0511H vermerkt.

Anhang H

=====

DEBUG:

Aufruf von DEBUG:

Um DEBUG zu starten, geben Sie folgendes ein:

DEBUG d:pathname filename.ext param. param.

Wenn Sie einen Dateinamen eingeben, so lädt DEBUG die angegebene Datei in den Speicher. Wird kein Dateiname angegeben, so arbeitet man mit dem augenblicklichen Speicherinhalt, oder man muß mittels der Befehle NAME und LOAD eine Datei in den Speicher laden. Die Parameter repräsentieren die möglichen Parametereingaben des jeweiligen auszuführenden Programms.

Beispiele:

DEBUG CHKDSK.COM
DEBUG DISKCOPY.COM A: B:

Beim Laden von DEBUG werden den Flags und Registern folgende Werte zugewiesen:

- Die Segmentregister CS, DS, ES und SS werden auf den Beginn des freien Speichers über DEBUG gesetzt.

- Der Instructionpointer (IP) wird auf 0100H gesetzt.
- Der Stackpointer (SP) wird auf das obere Ende des Segments gesetzt (FFFFH), falls dieses nicht zu Kollisionen mit COMMAND.COM führt. In diesem Fall wird SP unter COMMAND.COM gesetzt.
- Die übrigen Register (AX, BX, CX, DX, BP, SI und DI) werden auf 0000H gesetzt. Wurde DEBUG mit einem Dateinamen gestartet, enthalten BX und CX die Länge der Datei in Bytes. Dabei gibt BX die Anzahl der voll belegten 64K-Blöcke und CX die Anzahl der benutzten Bytes im letzten 64K-Block an.
- Die Flags werden gelöscht.
- Der Sektor I/O-Buffer wird auf Adresse 0080H im Codesegment initialisiert.

Anmerkung:

1. Wird eine .EXE-Datei von DEBUG geladen, so werden alle notwendigen Positionierungen von DEBUG durchgeführt. Die Segmentregister, der Stackpointer und der Instructionpointer werden entsprechend der Angaben in der Datei gesetzt. DS und ES zeigen auf das niedrigste benutzte Segment, BX und CX enthalten die Programmlänge.
2. Wird eine .HEX-Datei geladen, so nimmt DEBUG an, daß diese Hex-Bytes in ASCII-Darstellung enthält. Diese werden während des Ladens in ihre Binärform umgewandelt.

Eingabeparameter:

In diesem Handbuch werden bestimmte Begriffe zur Kennzeichnung verschiedener Befehlsargumente benutzt. Die Bedeutungen dieser Begriffe sollen hier erläutert werden.

Anmerkung: Alle Zahlen, die DEBUG übergeben werden, werden als hexadezimal angenommen.

Adresse - Eine Adresse besteht aus der optionalen Angabe eines Segments und einer Speicheradresse in diesem Segment:

Die Segmentangabe kann entfallen, z.B. 1000

Als Segment kann der Registername verwandt werden, z.B. CS:1FFF

Eine Segmentadresse kann direkt angegeben werden, z.B. 048A:2877

Byte - Eingabe eines ein- oder zweistelligen Hexadezimalwertes.

Laufwerk - Eingabe einer Nummer (0 oder 1 für Laufwerk A bzw. B)

Dateiname - Eingabe eines Dateinamens im üblichen Format, z.B. A:DISKCOPY.COM

Liste - Eingabe einer Folge von Hex-Bytes und/oder ASCII-Zeichen, die von Hochkommata oder Anführungszeichen eingeschlossen werden müssen.
Beispiel: F3 'ABC' 4 "xy"

Portadresse - Eingabe einer ein- bis vierstelligen Portadresse.

Bereich - Eingabe einer Anfangs- und Endadresse oder einer Anfangsadresse und einer Länge.
Beispiele: CS:177 4711
CS:177 L 100

Anmerkung: Wenn eine Endadresse angegeben wird, darf diese keine Segmentangabe enthalten.

Registername - Siehe REGISTER-Befehl

Sektor Seknr.- Eingabe von zwei ein- bis dreistelligen hexadezimalen Zahlen zur Angabe eines Sektors auf der Diskette und der Anzahl der zu schreibenden bzw. zu lesenden Sektoren.
Siehe Befehle LOAD und WRITE.

Anmerkung: Die Sektoren einer Diskette werden von 000H bis 13FH auf der Vorderseite und von 140H bis 27FH auf der Rückseite der Diskette numeriert.

Zeichenkette - Zeichenfolge, die in Hochkommata oder Anführungszeichen eingeschlossen ist. Wollen Sie Hochkommata bzw. Anführungszeichen in der Zeichenfolge verwenden, so schließen Sie sie in das jeweils andere Zeichen ein.

Beispiele:

'Ich heiÙe "Peter"'

"Dies ist ein Hochkomma: '"

Wert - Eingabe einer ein- bis vierstelligen Hexadezimalzahl.
Beispiel: 1000

Befehlsliste:

Ein DEBUG-Befehl besteht aus einem Buchstaben, gefolgt von verschiedenen Parametern. Die Befehle können in Klein- oder Großbuchstaben eingegeben werden, auch Mischung beider Schriftarten ist erlaubt. Die einzelnen Parameter müssen durch Leerzeichen oder Kommata voneinander getrennt werden, wobei der erste Parameter auch direkt hinter dem Befehl folgen darf.

Im folgenden Beispiel sind alle Eingaben gleichwertig:

```
DCS:100 110  
dCS:100,110  
d,cs:100,110
```

Die Ausführung jedes Befehls kann durch Betätigen von <Ctrl><Break> abgebrochen werden. Jeder Befehl wird erst ausgeführt, wenn die <ENTER> gedrückt wurde. Erzeugt ein Befehl viele Ausgaben auf dem Bildschirm, so kann die Ausführung durch <Ctrl><NumLock> angehalten werden. Durch Druck auf eine beliebige Taste wird die Ausführung fortgesetzt. Die Funktionstasten <F1> bis <F10> funktionieren wie im DOS-Handbuch beschrieben. DEBUG meldet sich immer mit einem Bindestrich ('-'), wenn es eine Eingabe erwartet.

ASSEMBLE - Mnemonische Befehle in Opcodes
übersetzen

FORMAT: A Adresse

FUNKTION: Mit Hilfe des A-Befehls können Sie Maschinenprogramme in den Speicher schreiben, ohne vorher die einzelnen Befehle (wie MOV AX,3333) mit Hilfe einer Liste in die Opcodes zu übersetzen. Die einzelnen Befehle werden ab der angegebenen Adresse (Default ist CS:100) im Speicher abgelegt. Die Eingabe wird beendet, indem Sie eine leere Zeile mit <ENTER> abschließen.

BEISPIEL: A 100
A DS:765

Anmerkung: Wenn Sie genaueres über die Maschinensprache des 8088-Prozessors wissen wollen, empfehlen wir Ihnen:

'Das 8086/8088 Buch'

von Russell Rector und George Alexy, erschienen im TE-WI Verlag

COMPARE - Speicherbereiche vergleichen

FORMAT: C Bereich Adresse

FUNKTION: Der Inhalt des durch 'Bereich' beschriebenen Speicherblocks wird mit dem Speicherinhalt ab 'Adresse' verglichen. Werden Unstimmigkeiten gefunden, so werden diese in folgendem Format ausgegeben:

```
adr1  bytel  byte2  adr2
```

adr1 gibt die Speicheradresse von bytel an, adr2 die Adresse von byte2. Werden bei der ersten Adresse von 'Bereich' und/oder bei 'Adresse' keine Segmentadressen angegeben, wird der Inhalt des DS-Register als Segmentadresse angenommen.

BEISPIEL: C 100L20,200

Die 32 Bytes von Adresse DS:0100 bis Adresse DS:011F werden mit den 32 Bytes ab Adresse DS:0200 verglichen.

DUMP - Speicherbereich in Hex- und ASCII-
Format auslisten

FORMAT: D Adresse
oder
D Bereich

FUNKTION: Wird nur eine Startadresse ange-
geben, so werden ab dieser 128
Bytes aus dem Speicher auf dem
Bildschirm gelistet. Wird ein Be-
reich angegeben, so wird der ge-
samte Bereich gelistet. Die Aus-
gabe erfolgt zeilenweise, jede
Zeile zeigt 16 Bytes in Hexadezi-
mal- und ASCII-Form an. Zeichen
kleiner als 32 dezimal werden in
der ASCII-Darstellung als Punkte
ausgegeben.

Wird bei der Startadresse keine
Segmentadresse angegeben, wird
der Inhalt des DS-Registers als
Segmentadresse benutzt.

Wird nach D nichts eingegeben, so
wird als Startadresse die Adresse
die erste Adresse gewählt, die
nach dem letzten D-Befehl nicht
mehr angezeigt wurde. Auf diese
Weise können ein Speicherlisting
durch D <ENTER> fortgesetzt wer-
den.

BEISPIEL: D0100
D1024:0100 01FF

ENTER - Änderung des Speicherinhalts

FORMAT: E Adresse Liste

FUNKTION: Wird die Adresse ohne Segmentadresse angegeben, wird der Inhalt des DS-Registers als Segmentadresse benutzt. Wird keine Liste angegeben, wird der Inhalt einer Speicherzelle ausgegeben und kann durch Eingabe eines Bytes überschrieben werden. Eine der folgenden drei Tasten muß betätigt werden:

Leertaste: Übernimmt das eingegebene Byte und geht eine Adresse weiter. Wurde kein Byte eingegeben, so wird der Inhalt der angezeigten Speicherstelle nicht verändert.

Bindestrich (-): Geht eine Adresse zurück.

<ENTER>: Beendet die Eingabe.

Wird eine Liste angegeben, so wird diese Liste in den Speicher ab der angegebenen Adresse übertragen.

BEISPIEL: E 0100
E CS:0100
E 0200,54,'Text',4

FILL - Speicherbereich füllen

FORMAT: F Bereich Liste

FUNKTION: Der angegebene Bereich (wurde keine Segmentadresse angegeben, wird DS angenommen) wird mit der Liste gefüllt. Enthält die Liste weniger Bytes, als der Bereich lang ist, wird sie wiederholt, bis der Bereich voll ist.

BEISPIEL: F 100,200,45
F 45A:100L20,'T'
F CS:300,500,'text',0

GO - Ausführen von Programmen

FORMAT: G =Adresse,Adresse,...

FUNKTION: Startet ein im Speicher befindliches Programm bei der angegebenen '=Adresse'. Wird '=Adresse' nicht angegeben, beginnt die Programmausführung bei der Adresse im IP-Register. Die restlichen angegebenen Adressen (bis zu 10), geben Punkte an, bei deren Erreichen die Programmausführung abgebrochen werden soll. Wird ein solcher Punkt erreicht, kehrt das System in die DEBUG-Eingabe zurück und gibt die Registerinhalte aus (Format wie beim R-Befehl beschrieben). Werden Adressen ohne Segmentadresse angegeben, wird der Inhalt des CS-Registers als Segmentadresse benutzt.

BEISPIEL: G=100
G=CS:4323,500
G600

HEXARITHMETIK - Addition und Subtraktion
zweier Hexzahlen

FORMAT: H Wert Wert

FUNKTION: Gibt Summe und Differenz der
beiden angegebenen Hexzahlen aus.

BEISPIEL: H 100,432

INPUT - Lesen eines Ports

FORMAT: I Portadresse

FUNKTION: Liest den Wert, der auf dem angegebenen Port liegt und gibt ihn aus.

BEISPIEL: I 3D8

LOAD - Laden von Programmen und Daten

FORMAT: L Adresse, Laufwerk, Sektor, Seknr.

FUNKTION: Lädt Daten oder Programme von Diskette in den Speicher. Werden Laufwerk, Sektor und Seknr. nicht angegeben, so wird ein Programm, dessen Name mit dem N-Befehl festgelegt sein muß, geladen. Wird Adresse angegeben, so wird das Programm ab dieser Adresse geladen. Segmentdefault ist dabei der Inhalt des CS-Registers. Wird Adresse nicht angegeben, wird das Programm nach Adresse CS:0100 geladen. Werden Laufwerk, Sektor und Seknr. angegeben, werden von dem angegebenen 'Laufwerk' 'Seknr.' Sektoren ab der Nummer 'Sektor' eingelesen und ab der angegebenen Adresse im Speicher abgelegt. 'Adresse' muß beim Sektorlesen angegeben werden, Segmentdefault ist auch hier der Inhalt des CS-Registers.

BEISPIEL: L 4BA:100 1 0F 6D
L

MOVE - Speicherbereich verschieben

FORMAT: M Bereich Adresse

FUNKTION: Verschiebt den angegebenen Bereich auf die neue Adresse. Dabei dürfen sich der alte und der neue Bereich auch überlappen. Segmentdefault ist beim M-Befehl der Inhalt des DS-Registers.

BEISPIEL: M 100,110,500
M 100L20,726

NAME - Filenamen eingeben

FORMAT: N Filespec Filespec

FUNKTION: Der erste angegebene Filespec wird formatiert im FCB ab Adresse CS:005C abgelegt, falls ein zweiter Filespec angegeben wird, wird dieser formatiert ab Adresse CS:006C. Dieses formatierte Ablegen ist für die Befehle LOAD und WRITE nötig, wenn sie Programme laden bzw. schreiben sollen.

BEISPIEL: N B:TEST.COM
N MYFILE

OUTPUT - Byte auf einen Port ausgegeben

FORMAT: O Portadresse Byte

FUNKTION: Gibt das Byte auf den angegebenen
 Port aus.

BEISPIEL: O 3D8 23

QUIT - DEBUG verlassen und ins DOS zurück-
kehren

FORMAT: Q

FUNKTION: DEBUG wird beendet und kehrt ins
Betriebssystem zurück. Alle DOS-
Parameter, wie aktives Laufwerk
etc., bleiben erhalten.

BEISPIEL: Q

REGISTER - Anzeigen und Modifizieren von Registers und Flags

FORMAT: R Registername

FUNKTION: Zeigt Registerinhalte an und erlaubt, diese zu modifizieren. Wird kein Registername angegeben, werden nur alle Registerinhalte, Flagzustände, sowie der Befehl ab Adresse CS:IP angezeigt. Die Anzeige erfolgt in folgendem Format:

1. Zeile: Inhalt der Register AX, BX, CX, DX, SP, BP, SI, DI
2. Zeile: Inhalt der Register DS, ES, SS, CS, IP sowie Zustand der Flags.
3. Zeile: Befehl ab Adresse CS:IP in hexadezimaler und mnemonischer Form.

Die Registerzustände werden folgendermaßen angezeigt:

Überlauf	Ja:	OV, Nein:	NV
Richtung	Erhöhen:	UP, Erniedrigen:	DN
Singlestepper			
Interrupt	An:	EI, Aus:	DI
Vorzeichen:	Positiv:	PL, Negativ:	NG
Null:	Ja:	ZR, Nein:	NZ
Hilfsübertrag:	Ja:	AC, Nein:	NA
Parität:	Gerade:	PE, Ungerade:	PO
Übertrag:	Ja:	CY, Nein:	NC

Wird als 'Registername' einer der Namen AX, BX, CX, DX, SP, BP, SI, DI, DS, ES, SS, CS oder IP eingegeben, so wird nur der Inhalt dieses Registers ausgegeben, dann kann ein neuer Wert für dieses Register angegeben werden. Wird als 'Registername' ein F angegeben, so wird der Zustand der Flags angezeigt. Diese können dann durch Eingabe der entsprechenden Kürzel (siehe Tabelle auf der letzten Seite) verändert werden. Die Kürzel können in jeder beliebigen Reihenfolge angegeben werden.

BEISPIEL: R
R IP
R F

SEARCH - Speicher durchsuchen

FORMAT: S Bereich Liste

FUNKTION: Durchsucht den angegebenen Bereich nach der angegebenen Liste. Wird sie gefunden, wird die Anfangsadresse der Liste im Speicher ausgegeben. Segmentdefault ist der Inhalt des DS-Registers.

BEISPIEL: S 100 400 'text'
S CS:100,200,42

TRACE - Programme in Einzelschritten ausführen

FORMAT: T =Adresse Wert

FUNKTION: Führt ab 'Adresse' 'Wert' Maschinenbefehle aus. Nach Ausführung eines jeden Befehls werden die Registerinhalte ausgegeben (Format siehe R-Befehl). Wird 'Wert' nicht angegeben, wird ein Befehl ausgeführt. Segmentdefault ist der Inhalt des CS-Registers, wird '=Adresse' nicht angegeben, wird der Befehl ab Adresse CS:IP ausgeführt. IP wird nach Ende des TRACE-Vorgangs auf den nächsten Befehl positioniert.

BEISPIEL: T=100,5
T 27
T=485:267

UNASSEMBLE - Programm in mnemonischer Form
auslisten

FORMAT: U Bereich

FUNKTION: Listet den Speicherinhalt ab der
angegebenen Adresse in hexadezi-
maler und mnemonischer Form. Wird
nur eine Adresse angegeben, wer-
den ab dieser 32 Bytes disassem-
bliert. Wird ein Bereich angege-
ben, werden alle Bytes in diesem
Bereich disassembliert. Segment-
default ist der Inhalt des CS-Re-
gisters.

BEISPIEL: U 100
U DS:387
U 123,7777

WRITE - Schreibt Programme und Daten

FORMAT: W Adresse Laufwerk Sektor Seknr.

FUNKTION: Werden Laufwerk, Sektor und Seknr. angegeben, so wird der Speicherinhalt ab der angegebenen Adresse auf Diskette geschrieben. Werden Laufwerk, Sektor und Seknr. nicht angegeben, so werden ab der angegebenen Adresse (Default ist CS:100) soviele Bytes unter dem mit dem N-Befehl angegebenen Namen auf Diskette geschrieben, wie das BX- und das CX-Register angegeben.

BEISPIEL: W 100 1 70 20
W
W DS:77

Zusammenfassung der DEBUG-Befehle:

Befehl	Format	Beschreibung
Assemble	A Adresse	Übersetzt Mnemonics in Opcodes
Compare	C Bereich Adresse	Vergleicht Speicherbereiche
Dump	D Adresse oder D Bereich	Listet den Speicherinhalt
Enter	E Adresse Liste	Speicherinhalt verändern
Fill	F Bereich Liste	Speicherblock füllen
Go	G =Adresse,...	Programm starten
Hexarithmetik	H Wert Wert	Summe u. Differenz bilden
Input	I Portadresse	Lesen eines Ports
Load	L Adresse lw...	Laden von Diskette

Move	M Bereich Adresse	Speicherbereich verschieben
Name	N Filespec	Filenamen festlegen
Output	O Portadr. Byte	Byte auf Port ausgeben
Quit	Q	DEBUG beenden
Register	R Registername	Register anzeigen/verändern
Search	S Bereich Liste	Speicher durchsuchen
Trace	T =Adresse Wert	Einzel-schrittausführung
Unassemble	U Adresse oder U Bereich	Disassemblieren
Write	W Adresse lw...	Schreiben auf Diskette

Anhang I

=====

GENIE 16 C BASIC-Fehlermeldungen:

Entdeckt das GENIE 16 C BASIC einen Fehler im Programm, wird eine der folgenden Fehlermeldungen ausgegeben. Diese lassen sich mit ON ERROR GOTO abfangen bzw. mit ERROR simulieren.

Nummer	Meldung
1	NEXT without FOR
2	SYNTAX Error
3	RETURN without GOSUB
4	Out of data
5	Illegal function call
6	Overflow
7	Out of memory
8	Undefined line number
9	Subscript out of range
10	Duplicate definition
11	Division by zero
12	Illegal direct
13	Type mismatch
14	Out of string space
15	String too long
16	String formula too complex
17	Can't continue
18	Undefined user function
19	No RESUME
20	RESUME without Error

21	Unprintable error
22	Missing operand
23	Line buffer overflow
24	Device timeout
25	Device fault
26	FOR without NEXT
27	Out of paper
28	Unprintable error
29	WHILE without WEND
30	WEND without WHILE
51	Internal error
52	Bad file number
54	Bad file mode
55	File already open
57	Device I/O error
62	Input past end
64	Bad filename
66	Direct statemant in file
67	Too many files
68	Device unavailable

Anhang J

=====

Kurzeingabe von Schlüsselwörtern:

Das GENIE 16 C BASIC erlaubt die schnelle Eingabe von Schlüsselwörtern mit Hilfe der Altund einer Buchstabentaste.

Die Belegung ist wie folgt:

A - AUTO
B - BSAVE
C - COLOR
D - DELETE
E - ELSE
F - FOR
G - GOTO
H - HEX\$
I - INPUT
J - J
K - KEY
L - LOCATE
M - MOTOR
N - NEXT
O - OPEN
P - PRINT
Q - Q
R - RUN
S - SCREEN
T - THEN
U - USING
V - VAL
W - WIDTH
X - XOR
Y - Y
Z - Z

Anhang K

=====

GENIE 16 C BASIC-Befehlstabelle:

ABS(X)	- Absolutwert von X z.B. ABS(-20)=20 etc.
AND	- UND-Verknüpfung
ASC(A\$)	- ASCII- Wert des ersten Buchstaben von A\$
ATN(X)	- Arcus Tangens von x
AUTO A , B	- Automatische Zeilen- numerierung
BEEP	- Erzeugt einen Signal- ton
BLOAD "name",offset	- Wie LOAD, nur für Maschinensprache
BSAVE "name",offset,Länge	- dto.
CALL variable,argument	- Ruft ein Maschinen- programm auf
CDBL(X)	- X wird eine Zahl mit doppelter Genauigkeit
CHAIN parameter	- Die Steuerung des System wird an ein anderes Programm gegeben, wobei die Variablen jedoch erhalten bleiben

CHR\$(X)	- Ergibt Zeichen mit ASCII-Code x
CINT(X)	- X wird eine ganzzahlige Zahl
CIRCLE (x,y),r	- Zeichnet eine Ellipse mit r als Radius und x,y als Mittelpunkt
CLEAR msp,stsp	- Löscht Variablen, reserviert Speicher
CLOSE fnl,..	- Schließt die angegebenen Files
CLS	- Löscht den Bildschirm
COLOR vor,hin	- Farbbefehl (Vordergrund, Hintergrund)
COM(n) ON/OFF/STOP	- Definiert, ob Unterbrechungen über den dangegebenen Anschluß bei DFÜ zugelassen sind oder nicht
COMMON v,v	- Über gibt die angegebenen Variablen an ein aufgerufenes Programm
CONT	- Setzt Programm nach BREAK fort
COS (X)	- Cosinus von X wird ermittelt
CSNG (X)	- X wird eine Variable einfacher Genauigkeit

CSRLIN	- Ermittelt Zeilennummer der Cursorposition
CVI	- Wandelt 2-Byte-String in numerischen Wert
CVS	- Wandelt 4-Byte-String in numerischen Wert
CVD	- Wandelt 8-Byte-String in numerischen Wert
DATA Liste	- Kennzeichnet Datenfeld. Lesen mit READ
DATE\$	- Fragt das Datum ab oder definiert es
DEF FNname(p)	- Definiert einzeilige Funktionen
DEF SEG=adr.	- Definiert Speichersegment f. POKE, etc.
DEFINT v,v,v	- Vorabdefinition für ganzzahlige Variablen
DEFDBL v,v,v	- Vorabdefinition für doppelt genaue Variablen
DEFSTR v,v,v	- Vorabdefinition für Stringvariablen
DEFSNG v,v,v	- Vorabdefinition für Variablen mit einfacher Genauigkeit

DEFUSRn=adr.	- Definition einer Startadresse für Maschinensprache
DELETE a - b	- Löscht alle Zeilennummern von a bis b
DIM a(n1,n2,..)	- Dimensioniert Matrixfelder n1 x n2 x n3
DRAW "zeichenkette"	- Zeichnet im Grafikmodus das, was in der Zeichenkette definiert ist
EDIT n	- Zeigt eine Zeile zur Korrektur an
END	- Ende des BASIC-Programms
EOF n	- Zeigt das Ende des Files n an
ERASE varliste	- Löscht die angegebenen Matrixfelder
ERR	- Ermittelt Fehlercode des letzten Fehlers
ERL	- Ermittelt Zeilennummer des letzten Fehlers
ERROR (X)	- Simuliert einen Fehler
EXP (X)	- Exponentialfunktion zur Basis e

- FIELD parameter - Bereitet die Aufteilung eines Puffers für eine Random-Datei vor
- FILES pathname - Zeigt das gewünschte Directory an
- FIX (X) - Schneidet alle Nachkommastellen von X ab
- FOR I=a TO b STEP c - Beginn einer BASIC-schleife
- NEXT I - Endpunkt der BASIC-Schleife
- FRE (X) - Zeigt freien Speicherplatz an
- FRE (X\$) - dto. erzwingt garbage collection
- GET dateiparameter - List aus einer Random-Datei einen Satz in den Puffer
- GET grafikparameter - List Punkte aus dem Bildschirmspeicher
- GOSUB Zn - Springt in Unterprogramm ab Zeile Zn
- GOTO Zn - Springt in Zeile Zn
- HEX\$ (X) - Ermittelt hexadezimale Zahl zu X
- IF..THEN..ELSE - Bedingte Programmverzweigung

IF..GOTO..ELSE	- Bedingte Programmverzweigung
INKEY\$	- Holt ein Zeichen von der Tastatur
INP (X)	- Holt den Wert auf dem I/O Port
INPUT"Text";var,var	- Holt Eingabestring von der Tastatur
INPUT#n, var,var,..	- Holt Daten von einem File
INPUT\$ len,#n	- Liest Text von der Länge n aus einem File
INSTR (i,A\$,B\$)	- Sucht nach dem Vorkommen von A\$ in B\$
INT	- Bildet den ganzzahligen Wert von X
KEY n,text	- Belegt die Funktionstaste n mit text
KEY list	- Listet die aktuelle Tastenbelegung auf
KEY ON	- Unterste Bildschirmzeile zeigt F-Belegung
KEY OFF	- Unterste Bildschirmzeile normal
KILL dateiname	- Löscht angegebene Datei von der Diskette

LEFT\$ (A\$,n)	- Ergibt die ersten n Zeichen von A\$
LEN (A\$)	- Ergibt die Länge der Zeichenkette A\$
LET	- Weist einer Variable einen Wert zu
LINE ...	- Grafikbefehl
LINE INPUT "..",var	- Eingabe einer Zeile von der Tastatur
LINE INPUT #n,..	- Entsprechend eine Zeile vom File
LIST a - b	- Ausgeben der Zeilen von a bis b
LLIST	- LIST auf dem Drucker
LOAD dateiname	- Lädt ein Programm von Diskette
LOAD dateiname,R	- Lädt ein Programm von Diskette und startet es
LOC (dateinummer)	- Zeigt die letzte Position in der angegebenen Datei an
LOCATE...	- Bestimmen der Cursorposition
LOF (dateinummer)	- Gibt die Länge der angegebenen Datei an

LOG	- Ermittelt den natürlichen Logarithmus von X
LPOS (X)	- Kopfposition eines angeschlossenen Druckers
LPRINT	- Wie PRINT nur auf einem angeschlossenen Drucker
LPRINT USING	- Wie PRINT USING nur auf einem angeschlossenen Drucker
LSET	- Datenübergabe in den Puffer einer Datei.
MERGE	- Verknüpfen von 2 BASIC-Programmen
MID\$ (a\$,b1,n2)	- Isolieren/Einfügen v. Teilen aus A\$
MKI\$	- Wandelt ganzzahligen numerischen Wert Zeichenkette
MKS\$	- Wandelt numerischen Wert einfacher Genauigkeit in Zeichenkette
MKD\$	- Wandelt numerischen Wert doppelter Genauigkeit in Zeichenkette
NAME altname AS neuname	- Ändert Dateinamen

NEW	- Löscht das BASIC-Programm
OCT\$ (X)	- Erzeugt einen String mit einer Oktalzahl von X
ON COM(n) GOSUB	- Sobald eine Information auf dem angegebenen DFÜ-Anschluß erscheint, wird ein Unterprogramm aufgerufen
ON ERROR GOTO	- Verzweigung, wenn ein Fehler auftritt
ON X GOTO a,b,c,d	- Verzweigung je nach X-Wert
ON X GOSUB ...	- Unterprogrammaufruf je nach X-Wert
ON KEY(n) GOSUB	- Springt ins Unterprogramm bei Betätigung der angegebenen Funktionstaste
ON PEN GOSUB	- Geht ins Unterprogramm sobald der Lightpen aktiviert wird
OPEN	- Eröffnung von Files bzw. Datenfeldern
OPEN"COM..."	- Eröffnung einer Datei für DFÜ
OPTION BASE (0 oder 1)	- Bestimmt Zählstart eines Feldes

OR	- ODER-Verknüpfung
OUT n,x	- Schreibt Wert x an Port n
PAINT (x,y),f,r	- Ausfüllen eines Bereiches im Bildschirm mit Farbe
PEEK (n)	- Liest Inhalt von Speicherstelle n
PEN ON/OFF/STOP	- Steuert angeschlossenen Lichtgriffel
PEN (X)	- Liest Information vom Lichtgriffel
PLAY zeihenkette	- Gibt Töne entsprechend der angegebenen Zeichenkette aus
POINT (x,y)	- Liest Farbe des Grafikpunktes auf x,y
POKE n,x	- Schreibt x in die Speicherstelle n
POS (X)	- Ermittelt Spalte der Cursorposition
PRESET (x,y),Farbe	- Setzt einen Punkt auf dem Schirm
PRINT ...	- Schreibt Daten auf den Bildschirm

- ? ... - Schreibt Daten auf den Bildschirm
- PRINT USING - Schreibt Daten formatiert auf den Schirm
- PRINT #n ... - Schreibt Daten in ein File
- PRINT #n USING - Schreibt Daten formatiert in ein File
- PSET - Grafikbefehl ähnlich PRESET
- PRESET (x,y),farbe - setzt einen Punkt auf den Bildschirm
- PUT dateinummer - Schreibt aus dem Puffer einen Satz in eine Random-Datei
- PUT grafikparameter - Schreibt in den angegebenen Bereich des Bildschirmspeichers
- RANDOMIZE - Zufallsgenerator neu starten
- READ a,b,c.. - Lesen von Daten aus DATA Feldern
- REM xyz - Einfügen von Kommentaren in den Programmtext

RENUM	- Umnummerieren von BASIC Zeilen
RESET	- Löscht Systempuffer und schließt alle offen Dateien
RESTORE Zn	- Setzt Zeiger eines DATA-Feldes
RESUME/RESUME NEXT	- Rückkehr aus der Fehlerbehandlung
RETURN	- Rückkehr aus Unterprogramm
RIGHT\$ (A\$,n)	- Ergibt die letzten n Zeichen von A\$
RND (X)	- Erzeugt eine Zufallszahl
RSET	- Datenübergabe in den Puffer einer Datei
RUN	- Startet ein BASIC Programm
SAVE" "	- Speichert ein Programm ab
SCREEN	- Umschalten zwischen den 8 Bildschirmen
SGN (X)	- Ermittelt das Vorzeichen von X
SIN (X)	- Ermittelt den Sinuswert von X

SOUND f,d	- Erzeugt einen Ton mit Freq. f und Dauer d
SPACE\$ (n)	- Erzeugt n Leerzeichen
SPC (n)	- Überspringt n Leerzeichen in PRINT u.a.
SQR (X)	- Ermittelt die Wurzel von X
STOP	- Unterbricht ein laufendes Programm
STR\$ (X)	- Wandelt x in einen String um
STRING\$(n,a\$)	- Erzeugt einen String nach Angabe
SWAP var1,var2	- Vertauscht zwei Variablen
SYSTEM	- Kehrt vom BASIC ins DOS zurück
TAB (X)	- Tabulator
TAN (X)	- Ergibt Tangens von X
TIMES\$	- Setzen oder Abfragen der Zeit
TRON	- Schaltet die Spurverfolgung ein
TROFF	- Schaltet die Spurverfolgung aus

USR (n,arg)	- Aufruf eines Maschi- nensprache-Unterpro- gramms
VAL (A\$)	- Wandelt den String A\$ in eine Zahl um
VARPTR...	- Ermittelt Variablen- adressen
WAIT (n,x)	- Wartet bis an Port n der Wert x anliegt
WHILE-WEND	- Besondere Schleifen- form
WIDTH	- Befehl zur Umschaltung der Bildschirm-Ausgabe
WRITE ...	- Allgemeiner Ausgabebe- fehl auf dem Schirm
WRITE #n,...	- Allgemeiner Ausgabebe- fehl auf eine Datei

Operatoren:

+	- Addition
-	- Subtraktion
*	- Multiplikation
/	- Division
^	- Potenzieren
(ausdruck)	- Klammerausdruck mit Vorrang
=	- Gleichheit
<>	- Ungleichheit
>	- größer als
<	- kleiner als
>=	- größer gleich
<=	- kleiner gleich

Anhang L

Erweiterte Tastencodes:

Der INKEY\$-Befehl ergibt einen zwei-Zeichen String wenn einige Tasten gedrückt werden. Wenn das erste Zeichen des Strings ASCII Null (00) ist, so ist es sinnvoll das zweite Zeichen zu untersuchen. Folgende Tabelle zeigt den Wert des zweiten Zeichens und die dazugehörige Taste.

Code	Taste
3	NULL
15	Tabulator links
16-25	Alt- QWERTYUIOP
30-38	Alt- ASDFGHJKL
44-50	Alt- ZXCVBNM
59-68	F1 - F10
71	Home
72	Cursor aufwärts
73	PgUp
75	Cursor links
77	Cursor rechts
79	End
80	Cursor abwärts
81	PgDn
82	Ins
83	Del
84-93	Shift F1 - F10
94-103	Ctrl F1 - F10
104-113	Alt F1 - F10
114	Ctrl PrtSc
115	Ctrl Cursor links
116	Ctrl Cursor rechts
117	Ctrl End
118	Ctrl PgDn
119	Ctrl Home
120-131	Alt 1,2,3,4,5,6,7,8,9,0,-,=
132	Ctrl PgUp

Anhang M

Die Schalter des GENIE 16C:

Zur Anpassung der Hauptplatine des GENIE 16C an verschiedene Peripherie-Geräte gibt es zwei sogenannte DIP-Schalter (von "Dual Inline Package"), die so heissen, weil sie die Größe von IC-Fassungen haben. Jeder der DIP-Schalter (im weiteren mit S1 und S2 bezeichnet) besteht aus acht Ein/Aus-Schaltern.

Die DIP-Schalter des GENIE 16C befinden sich auf der Hauptplatine vorne links, unter dem Platz für das 3. und 4. Diskettenlaufwerk bzw. für die Harddisk.

Jeder der Mikroschalter kann zwei Stellungen einnehmen, die Stellung "ein" ist mit "ON" bezeichnet (bei den jetzigen Geräten die rechte Stellung). Die Schalter sind von hinten nach vorne mit den Zahlen 1 bis 8 gekennzeichnet (Schreibweise von nun an S1-1 bis S1-8 bzw. S2-1 bis S2-8).

Einstellung der Schalter auf der Systemeinheit des GENIE 16C:

Die Schalter auf der Hauptplatine des GENIE 16C müssen immer richtig eingestellt sein, sonst arbeitet der Rechner nicht einwandfrei. Die folgende Tabelle gibt einen Überblick über die Funktion der einzelnen Schalter, die im Anschluß daran genauer erläutert werden:

Schalter	Funktion	ein	aus	ab Werk
S1-1	Diskettenlaufwerke angeschlossen?	nein	ja	aus
S1-2	8087-Koprozessor angeschlossen?	nein	ja	ein
S1-3	Größe des auf der Systemeinheit	siehe		*
S1-4	installierten Arbeitsspeichers	Tabelle 2		*
S1-5	Art (und Anzahl) der Bildschirme	siehe		ein
S1-6	an der Systemeinheit	Tabelle 3		aus
S1-7	Anzahl der Diskettenlaufwerke	siehe		aus
S1-8	an der Systemeinheit	Tabelle 4		ein
S2-1	- nicht	-	-	ein
S2-2	- benutzt	-	-	ein
S2-3	- (für spätere	-	-	ein
S2-4	- Erweiterungen	-	-	ein
S2-5	- vorgesehen)	-	-	ein
S2-6	Werden 256KBit-RAMs benutzt?	ja	nein	*
S2-7	Serieller Adapter eingeschaltet?	ja	nein	ein
S2-8	Druckeradapter eingeschaltet?	ja	nein	ein

(*) Diese Einstellungen hängen von der Hauptspeicher-Ausstattung des Rechners ab.

Tabelle 1: Schalterstellungen auf der System-Einheit

Schalter S1-1:

Dieser Schalter legt fest, ob an die Systemeinheit überhaupt Diskettenlaufwerke angeschlossen sind. Da dies beim GENIE 16C (und XC) der Fall ist, steht dieser Schalter in Stellung "aus".

Schalter S1-2:

Mit Schalter S1-2 wird festgelegt, ob der Interrupt (Unterbrechungssignal) des 8087-Koprozessors anerkannt werden soll (Stellung "ein") oder nicht. Ist kein 8087 auf der Systemeinheit enthalten, so muß dieser Schalter auf "aus" stehen. Da MSDOS und auch die meiste andere Software diesen Interrupt nicht unterstützt, wird empfohlen, diesen Schalter auch mit Koprozessor ausgeschaltet zu lassen. Andernfalls kann es zu unkontrollierten Systemabstürzen kommen.

Schalter S1-3, S1-4 und S2-6:

Zunächst einige Informationen über die Möglichkeiten, den GENIE 16C (XC) mit Arbeitsspeicher (RAM) zu bestücken:

Wie Sie aus den Daten des GENIE 16C wissen, läßt sich der Arbeitsspeicher des GENIE 16C bereits auf der Systemeinheit auf bis zu 640 KBytes ausbauen. Dies ist dadurch möglich, daß die Steckplätze auf der Hauptplatine mit verschiedenen Sorten von Speicherbausteinen bestückt werden können.

Die 256 KByte-Version des GENIE 16C besitzt vier Reihen ("Speicherbänke") mit Bausteinen mit einer Kapazität von je 64 KBit. Diese Speicherchips heißen 4164 oder 4864. Eine Bank dieser Chips ergibt also 64 KByte, alle vier zusammen folglich 256 KByte.

Die 640 KByte-Version des GENIE 16C besitzt zwei Reihen mit Chips von je 256 KBit (41256 genannt) und zwei Reihen mit 4164 bzw. 4864. Aus den obigen Betrachtungen ergibt sich, daß alle Geräte mit mehr als 256 KByte RAM mit den 41256-Chips bestückt werden müssen (oder aber jede Speichererweiterung über 256 KByte hinaus mit Hilfe einer in einen Erweiterungssteckplatz steckbaren Karte geschehen muß, mit dem Nachteil, daß damit ein Platz belegt ist). Der Schalter S2-6 bestimmt nun, ob 64 KBit-Chips oder 256 KBit-Chips (Stellung "ein") für die ersten beiden Bänke verwendet werden. Mit den Schaltern S1-3 und S1-4 wird die Anzahl der aktivierten Speicherbänke selbst bestimmt.

Die Anzahl der eingeschalteten Arbeitsspeicher-Bänke in Abhängigkeit von den Schaltern S1-3, S1-4 und S2-6 und die daraus resultierenden Speichergrößen gehen aus der folgenden Tabelle hervor:

S1-3	S1-4	eingeschaltete Bänke	S2-6 ein	S2-6 aus
ein	ein	1	64K	256K
aus	ein	1 + 2	128K	512K
ein	aus	1 + 2 + 3	192K	576K
aus	aus	1 + 2 + 3 + 4	256K	640K

Tabelle 2: Einstellung der Hauptspeichergröße

Schalter S1-5 und S1-6:

Wie schon oben erwähnt, wählen diese beiden Schalter die Art der verwendeten Bildschirm-Adapter-Karte (Video-Karte) aus. Für das GENIE 16C gibt es (auch von anderen Herstellern) verschiedene Arten von Video-Karten, wie die Monochrom-Karte (monochrome Zeichendarstellung mit sehr gutem Schriftbild, aber ohne Grafikmöglichkeit) oder die Farbgrafik-Karte (wahlweise Zeichendarstellung mit 40 oder 80 Zeichen pro Zeile, hochauflösende Farbgrafik), die standardmäßig beim GENIE 16C mitgeliefert wird.

Je nach verwendeter Bildschirm-Adapter-Karte (Standard für den GENIE 16C ist die Farbgrafik-Karte) müssen die Schalter S1-5 und S1-6 folgendermassen gesetzt werden:

S1-5	S1-6	eingestellter Bildschirm-Typ
ein	ein	kein Bildschirm
aus	ein	Farbgrafik-Karte 40x25 Zeichen
ein	aus	Farbgrafik-Karte 80x25 Zeichen
aus	aus	Monochrom-Karte 80x25 Zeichen

Tabelle 3: Einstellung des Bildschirm-Typs

Schalter S1-7 und S1-8:

In Abhängigkeit von der Anzahl der angeschlossenen Diskettenlaufwerke werden die Schalter S1-7 und S1-8 gesetzt. Normal beim GENIE 16C sind zwei Diskettenstationen, beim GENIE 16XC (Harddisk-Version) eine Diskettenstation. Die nächste Tabelle gibt die Schalterstellungen dafür an:

S1-7	S1-8	eingestellte Laufwerks-Anzahl
ein	ein	1 Diskettenlaufwerk angeschl.
aus	ein	2 Diskettenlaufwerke angeschl.

Tabelle 4: Einstellung der Laufwerks-Anzahl

Schalter S2-1 bis S2-5:

Diese Schalter werden momentan noch nicht benutzt, sind aber für spätere Erweiterungen des Systems reserviert.

Schalter S2-6:

Siehe oben.

Schalter S2-7:

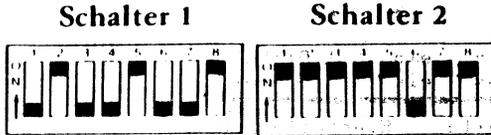
Dieser Schalter aktiviert die eingebaute asynchrone serielle Schnittstelle des GENIE 16C (COM1, liegt auf System-Board Nummer 03F8H). Standardeinstellung für diesen Schalter ist "ein", das Ausschalten ist bei der Verwendung von Zusatzkarten mit zwei seriellen Schnittstellen manchmal sinnvoll.

Schalter S2-8:

Schalter S2-8 ist normalerweise "ein", da er die eingebaute parallele Druckerschnittstelle (LPT2, für Programmierer: Port 0378H) aktiviert. Das Ausschalten dieses Schalters ist nur dann sinnvoll, wenn Druckerschnittstellen auf Zusatzkarten benutzt werden sollen.

Beispiele:

1) Für ein GENIE 16C mit zwei Diskettenlaufwerken, Farbgrafik-Karte und 256 KByte RAM ohne 8087-Koprozessor und zusätzliche Erweiterungskarten sollten die DIP-Schalter so eingestellt werden:



2) Für ein GENIE 16XC mit einem Diskettenlaufwerk, Farbgrafik-Karte, 640 KByte RAM und 8087-Koprozessor (unter MSDOS) und unter Verwendung der eingebauten Schnittstellen sieht die Einstellung der Schalter am Besten so aus:

