Teil 1

In die Geheimnisse der Floppy eingetaucht

Das Diskettenlaufwerk VC 1541 ist der Renner unter den Massenspeichern. Doch mit der passenden Literatur hapert es. Deshalb beschränken sich die meisten Anwender auf das Speichern und Laden von Programmen. Mit diesem Kurs lernen Sie, Ihre Floppy effektiv auszunützen und schließlich zu manipulieren.

Daß die 1541 ein sehr wandelbares Gerät ist, werden die meisten Benutzer wohl wissen oder zumindest erahnen. Man denke ja nur an den »Kleinkrieg« zwischen Softwareherstellern und Softwarepiraten, die sich gegenseitig das Leben schwer machen. Die meisten »Schlachten« liefert man sich hier im Inneren der Floppy, die viel raffiniertere Methoden des Programmschutzes anbietet als der Commodore 64.

Aber auch solche Programme, wie das HYPRA-LOAD, das Sie ebenfalls in dieser Ausgabe als Listing des Monats finden, beweisen die Flexibilität der 1541. Doch wie bei so vielen Dingen in der Commodore-Welt sind auch hier die Informationen rar, beziehungsweise in den Handbüchern gar nicht vorhanden. So wollen wir uns mit Ihnen an die Floppy heran- und in sie hineintasten. Angefangen bei grundlegenden Informationen über den Diskettenaufbau und den Befehlssatz der Floppy werden wir Schritt für Schritt in deren Möglichkeiten zur Programmierung und Manipulation hinabtauchen. Was wird benötigt?

Nun, außer einem C 64 und einer VC 1541, »nur« Basic-Erfahrungen, grundlegende Kenntnisse in Maschinensprache (für spätere Folgen) und ein wenig Geduld.

Bevor wir jedoch mit unserer ersten Tauchfahrt beginnen, tippen Sie bitte das beigefügte Programm EDDI (Listing 1) ein, sofern Sie nicht über einen eigenen Disk-Monitor verfügen. Auf die Bedienung von EDDI wird im Einzelnen eingegangen.

Sehen wir uns jetzt erst einmal so eine Diskette an; die folgenden Erläuterungen beziehen sich auf eine formatierte Diskette.

Aufbau einer Diskette

Die Diskette ist in 35 konzentrische Spuren (englisch: Tracks) aufgeteilt. Diese Spuren enthalten wiederum jede eine bestimmte Anzahl von Sektoren, die von außen nach innen abnimmt. Die genauen Zahlenverhältnisse stehen in Tabelle 1.

Die Spuren sind, beginnend mit der äußeren Spur, von 1 bis 35 durchnumeriert. Die Sektoren sind auf den Spuren in numerischer Reihenfolge gegen den Uhrzeigersinn angeordnet. Jeder Sektor enthält einen Block, das sind 256 Bytes, an Information. Es kann jeder der 683 Blöcke auf der Diskette durch Angabe der jeweiligen Spur- und Sektornummer aufgerufen werden. Allerdings stehen davon dem Benutzer normalerweise nur 664 Blocks zur Verfügung, da das Betriebssystem der Floppy die Spur 18 für sich beschlagnahmt.

Für die nun folgenden Versuche wäre es sinnvoll, eine Diskette neu zu formatieren, mit der wir ein bißchen »spielen« können. Sehen wir uns nun erst einmal das Directory an (LOAD "$",8).

In der ersten Zeile stehen die Drivenummer (hier immer 0) und der Name der Diskette, sowie die ID und das Formatkennzeichen (genaueres später).

Die zweite Zeile enthält, da sich kein File auf der Diskette befindet, die Meldung »664 BLOCKS FREE«.

Erste Versuche mit EDDI, dem Disk-Monitor/Editor

Da sich diese Informationen auf der schon erwähnten Spur 18 befinden, wollen wir uns diese Spur mit EDDI gleich einmal etwas genauer ansehen. Laden Sie den Editor und legen Sie unsere »Spieldiskette« ein; danach starten Sie mit RUN.

Als Kommando tippen Sie F3 für »BLOCK LESEN«. Danach geben Sie, durch Komma getrennt, die Spur und Sektornummer des gewünschten Blocks ein; in unserem Fall »18,0«.

Nach dem Ladevorgang meldet sich EDDI mit Byte 0 der ersten von 16 Seiten, zu je 16 Bytes. Drücken Sie jetzt RETURN, um die erste Seite anzuzeigen, welche wir nun betrachten wollen.

Es sollte vielleicht erwähnt werden, daß die Zählung von Blöcken und Bytes grundsätzlich bei Null beginnt. Den eingeladenen Block bezeichnet man als BAM (Block Availability Map), auf deutsch etwa »Blockbelegungsplan«. Dieser Plan gibt an, welche Blöcke auf der Diskette frei und welche schon beschrieben sind. Ferner enthält er den Namen der Diskette, die ID, das Formatkennzeichen und den Beginn des Directory.

Die ersten beiden Bytes (0,1) dieses Blocks enthalten Spur und Sektor des ersten Directoryblocks; normalerweise »18,1« (siehe auch Tabelle 2).

Byte 2 enthält das Formatkennzeichen (hier 65, beziehungsweise »A«). Zur Erklärung: Commodore stellt ja verschiedene Laufwerke her, zum Beispiel 1541, 3040, 8050, 8250... Diese Laufwerke unterscheiden sich fast alle im Aufzeichnungsformat, das heißt Anzahl und Verteilung der Spuren und Sektoren; so hat die CBM 8050 77 Spuren mit bis zu 29 Sektoren, was deren höhere Speicherkapazität zur Folge hat. Solche Disketten können verständlicherweise von der 1541 weder gelesen noch beschrieben werden. Am Formatkennzeichen »A« erkennt die 1541 nun Disketten ihres eigenen Formats; ist dieses nicht identisch, so beschwert sich die Floppy mit einer Fehlermeldung. Eine Ausnahme dieser Regel bildet die Lesekompatibilität, die besagt, daß eine »fremde« Diskette zwar gelesen, aber nicht beschrieben werden kann (zum Beispiel 3040 auf 1541).

Byte 3 steht generell auf Null, da es bei der 1541 keine Funktion erfüllt.

In die Geheimnisse der Floppy eingetaucht

Die Bytes 4 bis 143 enthalten nun die eigentliche BAM, deren Format ein wenig kompliziert ist: Für jede Spur sind 4 Bytes reserviert, wobei das jeweils erste Byte die Anzahl der noch freien Blöcke auf dieser Spur angibt. Die folgenden drei Bytes müssen wir als eine Gesamtheit von 24 Bits betrachten, wobei jedes gesetzte Bit einen freien Block signalisiert; siehe auch Tabelle 3.

Um auch die folgenden Seiten des Blocks zu betrachten, drücken Sie zum Vorwärtsblättern F1; die weitere Bedienung ist analog zur oben beschriebenen. Rückwärtsblättern ist durch Drücken von F2 möglich.

Fahren Sie nun bis zum Byte 144 vor und sehen Sie sich die Seite an.

Die Bytes 144 bis 161 enthalten den Namen der Diskette, der beim Formatieren festgelegt wird. Direkt im Anschluß daran folgen die Bytes 162,163, die die ID im ASCII-Code beinhalten, gefolgt von einem »Shift Space«. An der ID erkennt die Floppy, ob die Diskette gewechselt wurde; deshalb sollte jede Diskette eine andere ID haben.

Byte 165 und 166 enthalten DOS-Version und Formatkennzeichen, hier normalerweise »2A«, wiederum gefolgt von einem »Shift Space«.

Die Bytes 171 bis 255 haben normalerweise keine Bedeutung und können unterschiedlich gefüllt sein.

Wie sieht das Inhaltsverzeichnis aus?

Auf unserer Entdeckungsreise durch Spur 18 folgen wir jetzt der Angabe in den ersten beiden Bytes und laden den ersten Directoryblock (F3; 18,1). Das Format des Blocks ist der Tabelle 4 zu entnehmen. Jeder Direktoryblock enthält acht File-Einträge und den Zeiger auf den nächsten Direktoryblock (Byte 0 und 1); ist die Tracknummer des nächsten Blocks 0, so war der gelesene Directoryblock der letzte, und das zweite Byte zeigt die Anzahl der hier benutzten Bytes an. In unserem Fall stehen hier 0 und 255.

Nun zu Tabelle 5, die das Format eines Directoryeintrags darlegt: Jeder dieser Einträge besteht aus 30 Bytes, wobei das erste den Filetyp (siehe Tabelle 6), die beiden nächsten Spuren und Sektoren des ersten Fileblocks und die 16 folgenden Bytes den Filenamen enthalten. Die folgenden 3 Byte werden nur bei relativen Dateien verwendet; sie werden später im einzelnen noch besprochen.

Byte 26 und 27 enthalten Track und Sektor des neuen Files, falls das alte mit »@« überschrieben wurde. Die Bytes 28,29 schließlich geben die Anzahl der belegten Blöcke dieses Files an.

Die einzelnen Datei-Typen der Floppy

Diese bis jetzt beschriebenen Angaben werden vom Betriebssystem der Floppy, also vom DOS (englisch: Disk Operating System) verwaltet.

Beschäftigen wir uns nun mit den restlichen Blöcken auf der Diskette, die dem Anwender zur freien Verfügung stehen, denn dort werden die einzelnen Files abgespeichert, deren Aufbau uns jetzt interessiert.

DEL-Files:

Diese Fileanzeige existiert normalerweise nicht im Directory; wird ein File gelöscht, so wird dieses nicht mehr angezeigt; das Byte des Filetyps steht dann auf 0. Durch setzen des Filetyps auf 128 (hex. $80) kann eine DEL-Anzeige jedoch erzwungen werden.

SEQ-Files:

Dieser Filetyp dient zur Speicherung von Daten auf Diskette (im Gegensatz zur Programmspeicherung). Der Aufbau dieses Filetyps ist relativ einfach: Die ersten beiden Bytes eines Datenblocks zeigen jeweils auf den nächsten Block im File; so erfolgt eine beliebig lange Blockverkettung auf der Diskette. Da aber auch das schönste File einmal zu Ende geht, muß der letzte Block gekennzeichnet sein. Dies erfolgt, wie schon beim Directory, durch eine 0 als Spurnummer. Die Sektornummer bezeichnet jetzt die Anzahl der belegten Datenbytes dieses Blocks. Diese Art der Verkettung von Blöcken wird bei allen Filetypen vorgenommen! Die restlichen 254 Bytes jedes Blocks enthalten die Daten.

USR-Files

USR-Files stimmen im Aufbau exakt mit den SEQ-Files überein, sie haben jedoch noch Zusatzfunktionen im DOS, auf die ein anderes Mal eingegangen werden soll.

PRG-Files

PRG-Files stellen den häufigsten Filetyp dar. Sie dienen der Speicherung von Programmen auf der Diskette und haben nahezu den selben Aufbau wie SEQ-Files. Der einzige Unterschied besteht in den Bytes 2 und 3 des ersten Blocks, welche die Startadresse des Programms im Computer enthalten. Ist diese Adresse gleich der Adresse des Basic-Anfangs, also 2049 ($0801), so können die Programme mit »LOAD"Name",8« geladen werden; dieser Modus ignoriert die Anfangsadresse auf Diskette und lädt die Programme generell an den Basic-Anfang (sogenanntes relatives Laden). Sollen Programme jedoch an anderen Stellen im Speicher stehen, zum Beispiel Maschinenprogramme, so muß diese angegebene Adresse als Startadresse benutzt werden; man lädt hier absolut mit »LOAD"Name",8,1«.

REL-Files:

Dieser Filetyp ist im Aufbau ungleich komplizierter als die eben besprochenen; es soll daher zuerst kurz auf die Arbeitsweise von REL-Files eingegangen werden. Sequentielle Files haben den Nachteil, daß sie praktisch nur aus einem Datensatz bestehen. Sucht man nun, zum Beispiel in einer Kartei, eine bestimmte Hausnummer oder einen bestimmten Namen, so muß der gesamte Datensatz durchgelesen werden, um die entsprechende Stelle zu finden. In einer relativen Datei geht man deshalb einen anderen Weg, um jede Stelle schnell auffinden zu können.

Es existiert eine beliebige Anzahl (zum Beispiel 100) von Datensätzen, wobei alle Datensätze die gleiche Länge haben müssen (maximal 254 Zeichen).

Das DOS legt jetzt einen sogenannten Side-Sektor an, der aus bis zu sechs Blöcken bestehen kann. Diese Blöcke enthalten nun die Zeiger auf sämtliche Datenblöcke, in

denen die Datensätze gespeichert sind (1 Datensatz hat maximal 1 Block Länge). Auch hier sind die Datenblöcke wieder durch Zeiger in den Bytes 0 und 1 verkettet. Den Aufbau eines Side-Sektor-Blocks zeigt Tabelle 7. Zum besseren Verständnis hier ein kleines Beispiel:

```
10 REMEDDI - DISKMONITOR/EDITOR
20 REM          VON
30 REMKARSTEN SCHRAMM      1984
40 :
50 PRINT"■":POKE53280,14:POKE53281,14
60 GOSUB10000
70 OPEN1,8,15,"I0":OPEN2,8,2,"#"
80 PRINT"□ E D D I  -  HAUPTMENUE"
85 HE$="BYTE      DEC   HEX   BIN
 ASC":POKE650,128
90 PRINT" ──────────────────────"
100 PRINT:PRINT:PRINT
110 PRINT"(F1) - SCROLLING VORWAERTS":PR
INT
120 PRINT"(F2) - SCROLLING RUECKWAERTS":
PRINT
130 PRINT"(F3) - BLOCK LESEN":PRINT
140 PRINT"(F4) - BLOCK SCHREIBEN":PRINT
150 PRINT"(F5) - EDITOR EINSCHALTEN":PRI
NT
160 PRINT"(F6) - DISKETTE WECHSELN":PRIN
T
170 PRINT"(F7) - RUECKKEHR INS MENUE":PR
INT
180 PRINT"(F8) - PROGRAMMENDE"
190 PO=1:GOTO 9000
1000 REM EDDI AN
1010 X=0:Y=0
1020 FORY=ETO255STEP16
1030 PO=2:PRINT"□EDITOR-MODUS FUER TRACK
"T" SEKTOR"S
1040 PRINT:PRINTHE$:PRINT
1050 FORX=YTOY+15:PRINTX:NEXTX
1060 PRINT"■■■■":FORX=YTOY+15
1065 DA=PEEK(50000+X):GOSUB7030:PRINTX,O
U$
1070 INPUT"□■■■■■■■■";IN$:IFIN$=""THEN10
90
1072 IFLEFT$(IN$,1)="↑"THENPRINT"■■■■■■■
":GOTO9000
1073 IFLEFT$(IN$,1)="←"THENPRINT"■■■■■■■
":GOTO1125
1075 DA=VAL(LEFT$(IN$,3)):IFDA>255ORDA<0
THENPRINT"□□":GOTO1065
1080 POKE50000+X,DA
1120 NEXTX:PRINT
1125 PRINT"EINGABE ?";
1130 GETA$:IFA$=""THEN1130
1140 IFA$="■"THEN1200
1150 IFA$="■"THEN1300
1160 IFA$<>" "THENNEXTY
1170 PO=1:GOTO9000
1200 PRINT"■■■■":PRINTE".....???"
1210 GETA$:IFA$=""THEN1210
1215 IFA$="■"THEN1300
1220 IFA$<>"■"THEN1020
1230 E=E+16:IFE>255THENE=0
1240 GOTO1200
1300 PRINT"■■■■":PRINTE".....???"
1310 GETA$:IFA$=""THEN1310
1315 IFA$="■"THEN1200
1320 IFA$<>"■"THEN1020
1330 E=E-16:IFE<0THENE=240
1340 GOTO1300
2000 REM DISKETTENWECHSEL
2010 PRINT"□BITTE NEUE DISKETTE EINLEGEN
"
2020 GETA$:IFA$=""THEN2020
2030 RUN
3000 REM BLOCK READ
3010 PO=2:PRINT"□   BLOCK LESEN":PRINT:P
RINT
3020 INPUT"TRACK, SEKTOR ";T,S
3025 IFT<1ORT>35THEN3010
3030 PRINT#1,"U1 2 0"T;S
3035 IFST<>0THENPRINT:GOTO9000
3040 PRINT#1,"B-P 2 0"
3050 SYS49152:E=0:X=0:Y=0:GOTO5010
3060 FORY=ETO255STEP16
3070 PRINT"□TRACK"T" SEKTOR"S
3080 PRINT:PRINTHE$:PRINT
3090 FORX=YTOY+15:DA=PEEK(50000+X):GOSUB
7030:PRINTX,OU$:NEXTX
3100 GOTO9000
4000 REM BLOCK WRITE
4010 PO=1:PRINT:PRINT:INPUT"□■TRACK, SEK
TOR";T,S:PRINT"■"
4020 PRINT#1,"B-P 2 0"
4030 SYS49177
4040 PRINT#1,"U2 2 0"T;S
4050 GOTO9000
5000 REM SCROLL FORWARD
5010 E=X:IFE>255THENX=0:E=0
5020 PRINT"□TRACK"T" SEKTOR"S
5030 PRINT:PRINTHE$:PRINT
5040 DA=PEEK(50000+E):GOSUB7030:PRINTE,O
U$
5050 X=X+16
5060 GETA$:IFA$=""THEN5060
5070 IFA$="■"THEN5010
5075 IFA$="■"THENX=X-16:GOTO6010
5077 IFA$="■"THEN1000
5080 GOTO3060
6000 REM SCROLL BACKWARD
6010 E=X:IFE<0THENE=240:X=240
6020 PRINT"□TRACK"T" SEKTOR"S
6030 PRINT:PRINTHE$:PRINT
6040 DA=PEEK(50000+E):GOSUB7030:PRINTE,O
U$
6050 X=X-16
6060 GETA$:IFA$=""THEN6060
6070 IFA$="■"THEN6010
6075 IFA$="■"THENX=X+16:GOTO5010
6077 IFA$="■"THEN1000
6080 GOTO3060
7000 REM BEREITSTELLUNG DES STRINGS
7010 REM DA/DA$ SIND AUSGABEWERTE
       H$,D$,B$,C$ SIND ZWISCHENWERTE
7020 REM OU,OU$ SIND ENDERGEBNISSE
7030 IFDA>31ANDDA<128ORDA>159ANDDA<256TH
ENC$=CHR$(DA):GOTO7040
7035 C$="."
7040 XX$="000":D$=RIGHT$(STR$(DA),LEN(ST
R$(DA))-1)
7045 D$=LEFT$(XX$,3-LEN(D$))+D$
7050 XX$="123456789ABCDEF":H$=""
7060 HH=INT(DA/16):HL=DA-HH*16
7070 IFHHTHENH$=H$+MID$(XX$,HH,1):GOTO70
80
7075 H$=H$+"0"
7080 IFHLTHENH$=H$+MID$(XX$,HL,1):GOTO70
90
7085 H$=H$+"0"
7090 B$="":FORQ=7TO0STEP-1
7100 IF(DAAND(2↑Q))<>0THENB$=B$+"1":NEXT
:GOTO7110
7105 B$=B$+"0":NEXT
7110 OU$=D$+"   "+H$+"    "+B$+"   "+C$
7120 RETURN
8999 END
9000 REM GET KOMMANDO
9010 PRINT:PRINT"KOMMANDO ?  ";
9020 PRINT"■■";:FORW=1TO75:GETKO$:IFKO$<
>""THEN9090
9030 NEXTW
9040 PRINT"■■■■";:FORW=1TO75:GETKO$:IFKO
$<>""THEN9090
9050 NEXTW
9060 GOTO9020
9090 IFKO$="@"THEN9200
9100 IF ASC(KO$)>140ORASC(KO$)<133THEN90
20
9110 KO=ASC(KO$)-132
9120 ON PO GOTO9130,9140,20000
9130 ON KO GOTO9020,3000,1000,80,9020,40
00,2000,20000
9140 ON KO GOTO5000,3000,1000,80,6000,40
00,2000,20000
9200 PRINT
9210 GET#1,A$:PRINTA$;:IFST<>64THEN9210
9220 GOTO 9000
9999 END
10000 DATA160,0,169,8,32,9,237,169,98,32
,199,237,32,19,238,153,80,195,200
10010 DATA208,247,32,239,237,96,160,0,16
9,8,32,12,237,169,98,32,185,237
10020 DATA185,80,195,32,221,237,200,208,
247,32,254,237,96,0,0
10030 RESTORE:FORZ=1TO51:READA:POKE49151
+Z,A:NEXT
10040 REM GET:49152; WRITE:49177
10050 RETURN
20000 PRINT:PRINT:PRINT"□AUF WIEDERSEHEN
 !!!":PRINT:POKE53280,14:POKE53281,6
20001 PRINT"UND DANKESCHOEN !"

READY.
```

Listing 1. EDDI, ein Disk-Monitor/Editor

Wir haben eine relative Datei mit 250 Datensätzen à 127 Zeichen. Diese Datei benötigt also 125 Datenblöcke und zwei Side-Sektor-Blöcke. Im Directory-Eintrag finden wir jetzt die schon erwähnten zusätzlichen Bytebelegungen: Byte 19 und 20 jedes Eintrags enthalten jetzt Spur und Sektor des ersten Side-Sektor-Blocks; Byte 21 gibt die Datensatzlänge (Recordlänge) an.

Wir wollen jetzt auf den 248. Datensatz zugreifen; das DOS arbeitet nun folgendermaßen: Ein Datensatz enthält 127 Byte, das heißt, es passen zwei Datensätze in einen Block; dadurch errechnet sich der Block, auf den jetzt zugegriffen wird, aus (248-1)/2 = 123.5. (Minus 1, da immer von 0 an gezählt wird). Da ein Side-Sektor-Block nur 120 Einträge aufnehmen kann, ist der Zeiger auf den Datenblock im Side-Sektor-Block Nummer 2 zu finden. Dieser wird jetzt anhand des Verzeichnisses in Block 1 gelesen und dann auf Zeiger Nummer 3 (Bytes 22,23) zugegriffen. Wir kennen also jetzt Spur und Sektor des Blockes, in dem unser Datensatz steht; die Position des ersten Datenbyte berechnet sich jetzt aus dem Nachkommaanteil der obigen Division (0.5*254 = 127). Der Datensatz beginnt also beim 127 + 2 = 129ten Byte.

Der Aufbau von relativen Dateien ist also, wie schon erwähnt, ziemlich kompliziert; diese Art der Datenspeicherung hat aber einige Vorteile gegenüber der 'normalen' mit SEQ-Files.

Bedienungshinweise für EDDI, dem Disk-Monitor/Editor

Da unserem U-Boot auf dieser schwierigen Fahrt der Sauerstoff ausgegangen ist, wollen wir uns nun erst einmal erholen. Hier noch ein paar Anregungen zur Arbeit mit EDDI: EDDI kann nicht nur Blöcke lesen und anzeigen; Sie können auch Bytes verändern und diesen Block danach wieder abspeichern.

Dazu laden Sie den zu verändernden Block und fahren auf die Seite, die Sie interessiert; hier tippen Sie als Kommando F5, und der Editor-Modus startet. Sie können jetzt Bytes dezimal abändern, indem Sie den jeweils neuen Wert eingeben und »RETURN« drücken. Wollen Sie aus dem Eingabemodus aussteigen, so

In die Geheimnisse der Floppy eingetaucht

Spuren	Sektoren	Anzahl Total
01-17	00-20	21
18-24	00-18	19
25-30	00-17	18
31-35	00-16	17

Tabelle 1. Spuren und Sektoren des VC 1541-Diskettenformates

Aufbau des Blocks 18,0:	
BYTE(s)	Bedeutung:
000-001	Spur und Sektor des ersten Directory-Blocks
002	Formatkennzeichen »A«
003	unbenutzt
004-007	BAM der Spur 1
008-011	BAM der Spur 2
012-143	BAM der Spuren 3-35
144-161	Name der Diskette, aufgefüllt mit 160
162-163	ID der Diskette
164	unbenutzt
165-166	DOS und Formatkennzeichen
167-255	nicht benutzt

Tabelle 2. Aufbau und Inhalt der BAM (Block-Belegungs-Plan) in Spur 18, Sektor 0

Aufbau eines 4 Byte-Eintrages in der BAM (eine Spur)	
BYTE(s)	Bedeutung:
000	Anzahl der freien Blöcke dieser Spur
001-003	Belegplan der Spur Jedes Byte ist zuständig für 8 Sektoren: Byte 1 für 0-7 Bit 7 für Sektor 0 Bit 6 für Sektor 1 und so weiter Byte 2 für 8-15 Byte 3 für 16-23

Tabelle 3. Für jede Spur reserviert die BAM 4 Bytes

Aufbau eines Directory-Blocks:	
BYTE(s)	Bedeutung
000-001	Spur und Sektor des nächsten Dir.-Blocks
002-031	Eintrag Nr. 1
032-033	unbenutzt
034-063	Eintrag Nr. 2
064-066	unbenutzt
067-225	Einträge Nr. 3-7 bzw. unbenutzt
226-255	Eintrag Nr. 8

Tabelle 4. Aufbau der Directory der VC 1541

tippen Sie entweder »RETURN« und können weiterblättern ohne den Editor zu verlassen, oder Sie tippen »↑ RETURN«, um in den Kommando-Modus zu kommen. Nach einigem Probieren wird Ihnen EDDI sehr schnell vertraut werden; wir gehen auch in den folgenden Ausgaben noch darauf ein.

Wichtig:

Beim Wechseln einer Diskette muß die Funktionstaste F6 getippt und nach dem Austausch eine Taste gedrückt werden, sonst reagiert die Floppy mit einer Fehlermeldung. Diese können übrigens mit »@« abgerufen werden. Das Zurückschreiben eines Blocks auf Diskette erfolgt mit F4, wobei Spur und Sektornummer angegeben werden müssen. Hier noch ein paar Vorschläge zum Ausprobieren: Ändern Sie doch einmal auf Ihrer Versuchsdiskette (!) das Formatkennzeichen (Spur 18, Sektor 0, Byte 2 auf 66 statt jetzt 65 und speichern den Block an die gleiche Stelle auf die Diskette zurück. Versuchen Sie nun einmal ein kleines Programm auf diese Diskette zu schreiben. (Die genauen Vorgänge in der Floppy werden beim nächstenmal erläutert.) Oder ändern Sie einmal die Bytes im Directory, die den Filetyp angeben, entsprechend Tabelle 6 und laden Sie es danach. Mit dem klugen Satz »Probieren geht über Studieren« verabschieden wir uns für diese Ausgabe. Nächstesmal beginnt dann ein praktischer Teil unserer Expedition, nämlich die Vorstellung des Befehlssatzes der 1541 mit vielen Beispielen und Anregungen.

(K. Schramm/B. Schneider/gk)

Aufbau des Filetyp-Bytes					
BIT	Bedeutung, in Klammern jeweiliger Inhalt				
0	(0)	(1)	(0)	(1)	(0)
1	(0)=DEL	(0)=SEQ	(1)=PRG	(1)=USR	(0)=REL
2	(0)	(0)	(0)	(0)	(1)
3	unbenutzt				
4	unbenutzt				
5	unbenutzt				
6	(0)=normal; (1) = File kann durch SCRATCH nicht mehr gelöscht werden				
7	(0)= File noch offen (1)= File ordnungsgemäß geschlossen				

Tabelle 6. Die Bedeutung des ersten Bytes eines Directory-Eintrages

Aufbau eines Directory-Eintrags:	
BYTE(s)	Bedeutung
000	Filetyp, siehe ges. Tab
001-002	Spur und Sektor des ersten Datenblocks
003-018	Filename, aufgefüllt mit 160
019-020	REL-Files: Spur und Sektor des ersten Side-Sektor-Blocks
021	REL-Files: Datensatzlänge
022-025	unbenutzt
026-027	Spur und Sektor beim Überschreiben mit @ (nur Zwischenspeicher)
028-029	Anzahl der von diesem File belegten Blocks

Tabelle 5. Bedeutung der einzelnen Byte des Directory

Aufbau eines Side-Sektor-Blocks:	
BYTE(s)	Bedeutung
000-001	Spur und Sektor des nächsten Side-Sektor-Blocks
002	Nummer des Side-Sektor-Blocks
003	Datensatzlänge
004-005	Spur und Sektor des Side-Sektor-Blocks 1
006-007	Spur und Sektor des Side-Sektor-Blocks 2
008-015	Spur und Sektor der Side-Sektor-Blöcke 3-6
016-017	Spur und Sektor des ersten Datenblocks für den der Side-Sektor-Block zuständig ist. (Datenblock 0) Spur und Sektor des zweiten Datenblocks (Nr. 1)
018-255	Spur und Sektor der Datenblocks Nr. 2 bis Nr. 119

Tabelle 7. Relative Dateien benutzen Side-Sektor-Blöcke um Datensätze gezielt anzuspringen