

Das Programm FORMAT.COM (Version 3.8)

FORMAT.COM dient der Formatierung von Disketten auf dem Schneider PCW/JOYCE.

FORMAT basiert auf dem Programm XFORMAT (von M. Anton in PCI 7/87) und der Erweiterung XXFORMAT (von A F S C in PCI 6/88). FORMAT ist als Ergänzung und Ersatz von DISCKIT gedacht. Es bietet folgende Formate an:

Format	Laufwerk	Spuren	Sektoren	Speicher	
-----	-----	-----	-----	-----	
CPC System	A	40	9	169k	
CPC Data only	A	40	9	178k	
CF2	A	40	9	173k	
CF2	A	43	9	187k	
CF2	A	40	10	193k	*
CF2	A	43	10	208k	*
CF2DD	B	80	9	706k	
CF2DD	B	84	9	740k	
CF2DD	B	80	10	784k	*
CF2DD	B	84	10	824k	*

(* = neu in V3.5)

Programmaufruf

FORMAT kann sowohl ohne als auch mit Parametern aufgerufen werden.

Ein Aufruf ohne Parameter bringt ein Menü auf dem Bildschirm. Dieses Menü zeigt die vom Programm unterstützten Formate an und gibt dem Anwender die Möglichkeit sie per Tastendruck auszuwählen.

Die für den jeweiligen Menüpunkt einzugebende Taste steht gefolgt von einer schließenden runden Klammer ')' vor dem Menüpunkt. Zur Eingabe der gewählten Option des Menüs muß nur die jeweilige Taste eingegeben werden, eine Bestätigung mit der RETURN-Taste ist nicht erforderlich.

Beim Aufruf mit Parametern erwartet FORMAT eine Formatangabe und eine optionale Angabe der GAP-Length-Werte.

Die Angabe der Formate kann in unterschiedlicher Form erfolgen. Entweder wird die Größe der Diskette in kByte angegeben (z.B. FORMAT 173 oder FORMAT 784) oder die Anzahl der gewünschten Spuren und Sektoren (z.B. FORMAT 40,10) getrennt durch ein Komma. Eine weitere Möglichkeit ist die Angabe der Standardformate mit Namen (z.B. FORMAT CF2) sowie von Laufwerksnamen (z.B. FORMAT B:) wobei das jeweils größte unterstützte Format benutzt wird. Diskettengrößen sowie Spur- und Sektorangaben die vom Programm nicht unterstützt werden führen zu einem Fehler.

FORMAT erkennt automatisch zu welchem Laufwerk das angegebene Format gehört, eine zusätzliche Angabe des Laufwerksnamens ist also nicht erforderlich und auch nicht erlaubt.

Im Folgenden eine Liste aller möglichen Parameter für FORMAT. Die Angaben in Klammern hinter den Formatnamen deuten die Anzahl der Spuren und Sektoren an. Die Parameter sind natürlich ohne die hier gezeigten einschließenden Häkchen einzugeben.

Formate in Laufwerk A:

CPC System-Format	'169'	'SYS'
CPC Data-Only-Format	'178'	'DATA'
CF2-Format (40/9)	'173'	'40,9' 'CF2'
CF2-Format (43/9)	'187'	'43,9'
CF2-Format (40/10)	'193'	'40,10'
CF2-Format (43/10)	'208'	'43,10' 'A:'

Formate in Laufwerk B:

CF2DD-Format (80/9)	'706'	'80/9' 'CF2DD'
CF2DD-Format (83/9)	'740'	'83/9'
CF2DD-Format (80/10)	'784'	'80/10'
CF2DD-Format (83/10)	'824'	'83/10' 'B:'

Hilfstexte anzeigen

Kurzinformationen	'?' '/?' '-?'
Ausführliche Hilfestellung	'H' '/H' '-H'

Da es auf einigen JOYCE, aufgrund von Laufwerksunterschieden, Probleme mit der vorgegebenen GAP-Length gibt, ist diese einstellbar. In der Kommandozeile wird eine geänderte GAP-Length mit zusätzlichen Parametern angegeben:

```
FORMAT <format> GAP <gap-rw> <gap-fmt>
```

Wobei <format> einer der oben angegebenen Formatbezeichner ist während <gap-rw> und <gap-fmt> die Zahlenwerte für die GAP-Length beim Lesen/Schreiben bzw. beim Formatieren sind. Die Werte für <gap-rw> sollten im Bereich von ca. 6 bis 12, die Werte für <gap-fmt> doppelt so hoch liegen. Vorgegeben sind <gap-rw>=12 und <gap-fmt>=23.

Zusätzlich gibt es noch die Möglichkeit eine bestehende Diskette im CF2- Standardformat (40 Spuren, 9 Sektoren) auf 43 Spuren zu erweitern, eine Nachformatierung vorzunehmen. Dies geschieht mit dem Aufruf

```
FORMAT /N
```

Aber Achtung: Dieser Aufruf prüft nicht, ob in Laufwerk A: wirklich eine normale CF2-Diskette vorliegt. Die Spuren 40 bis 42 (Gezählt wird ab Spur 0) werden mit 9 Sektoren pro Spur im CF2-Format formatiert. Sollte die Diskette ein CPC-Format oder ein Format mit 10 Sektoren haben, dann gibt's Datenmüll! Nach dem Umformatieren weist der Bootsektor die Diskette als CF2-Diskette mit 43 Spuren und 9 Sektoren aus.

Mit der Version 3.8 verifiziert FORMAT die Diskette nach der Formatierung. Dadurch kann FORMAT jetzt endlich fehlerhafte Sektoren auf der Diskette erkennen. Leider hat diese Verbesserung aber auch, wie so oft, einen kleinen Haken: FORMAT wird dadurch deutlich langsamer. Wer sich sicher ist, dass die Diskette keine Fehler enthält, der kann die Verifizierung mit

```
FORMAT <format> -V
```

abschalten. Damit erhält FORMAT seine gewohnte Geschwindigkeit, aber auch seine Unzuverlässigkeit, zurück.

Systemvoraussetzungen

Schneider/Amstrad PCW xxxx JOYCE mit CP/M plus. Ich hoffe allerdings, dass auch CPC-6128-Besitzer etwas damit anfangen können. Besitzer anderer Computer sollten sich keine großen Hoffnungen machen, da das XBIOS sehr Amstrad-spezifisch ist.

Entwickelt und getestet habe ich FORMAT auf einem Schneider PCW 8256 mit einem 5 1/4" Zweitlaufwerk und einer Speichererweiterung auf 512 kByte unter Turbo-Pascal 3.0.

Erläuterungen

Die in XFORMAT bzw. XXFORMAT gegebene Möglichkeit die CPC-Formate mit mehr als 40 Spuren zu formatieren ist hier aus folgendem Grund nicht implementiert.

Der JOYCE kann die CPC-Formate mit 40 Spuren bearbeiten, mehr als 40 Spuren in den CPC-Formaten würden eine Änderung an der *.EMS-Datei erfordern.

Die CF2-Formate hingegen liest der JOYCE auch mit mehr Spuren und/oder Sektoren ohne Änderung an der EMS-Datei. Dann muss allerdings der Bootsektor der Diskette an das 43/83-Spur- bzw. 10-Sektor-Format angepasst werden. Dies wird von FORMAT durchgeführt.

In der Dokumentation zu Version 3.4 habe ich behauptet, der JOYCE könne Formate mit mehr als 80 Spuren nicht ohne einen Eingriff in die EMS-Datei verarbeiten. Das ist ein Irrtum! Bei Disketten mit höherer Speicherkapazität muss nur die Blockgröße von 2k auf 4k geändert werden, da CP/M andernfalls mit der Anzahl der Blöcke nicht mehr klarkommt.

FORMAT schreibt einen eigenen Bootsektor auf Disketten die im CF2- oder CF2DD-Format formatiert werden. Im CF2DD-Format ist der Sektor bis auf die Kopfinformationen, die das Format beschreiben, schlicht leer. Im CF2-Format ist der Bootsektor in der Lage die erste *.EMS-Datei von Diskette zu laden und zu starten. Der CF2-Bootsektor funktioniert auch mit den erweiterten Formaten mit mehr Spuren und/oder Sektoren.

Die CPC-Formate werden nicht am Bootsektor, sondern an ihren abweichenden Sektornummern erkannt. Beim Data-Only-Format verbietet sich ein Bootsektor von selbst, da dort keine reservierte Spur zur Verfügung steht. Im System-Format gibt es zwei reservierte Spuren in denen das CP/M 2.2 untergebracht ist. Da ich es erstens nicht habe und es zweitens zu Copyrightproblemen führen würde, es weiterzugeben, ist auch auf den systemformatierten Disketten kein Bootsektor drauf. CPC-Besitzer haben aber, falls nötig, sicher die Möglichkeit nachträglich ein System auf die Diskette zu bringen.

Wenn ich richtig informiert bin, dann wird das CPC-Systemformat ohne System, aber mit den reservierten Spuren als sogenanntes Vendor-Format bezeichnet.

Warum 4 kByte/Block bei den großen Formaten?

Die Blockgröße beim Standard CF2DD-Format beträgt 2 kByte, alle meine Versuche ein Format mit mehr als neun Sektoren oder mehr als achtzig Spuren bei einer Blockgröße von 2 kByte zu formatieren und anschließend auch zu benutzen sind kläglich gescheitert. Das heißt natürlich nicht, dass es unmöglich sein muss! Im CP/M 3 Programmers Guide steht, dass mit einer Blockgröße von 2 kByte eine Laufwerksgröße von bis zu 64 MByte (!) möglich ist, ich krieg's dem JOYCE nur nicht untergejubelt.

Meine vorläufige Erklärung dafür ist folgende: Im CP/M existiert ein sogenannter Disk-Parameter-Header (DPH) für jedes Laufwerk. In diesem DPH steht unter anderem ein Zeiger mit

dem Namen Allocation-Vector (ALV). Dieser Allocation-Vektor zeigt auf einen Bereich in dem die Belegung der Diskette durch die Dateien festgehalten wird. Die Länge des Bereiches ist auf die Diskettengröße abgestimmt. Pro Block auf der Diskette ist in der Allocation-Table ein bit reserviert. D. h. für Laufwerk B: mit Standardmäßig 357 Blöcken ist die Allocation-Table mindestens 45 Byte lang. Um Speicher zu sparen ist sie wahrscheinlich kein Byte länger, so dass bei 360 Blöcken spätestens Schluss ist.

Bei größeren Formaten ist man damit, bis jemand eine Lösung findet, auf eine höhere Blockgröße von 4 kByte angewiesen. Um auch die größeren Formate mit einer Blockgröße von 2 kByte benutzen zu können braucht man einen größeren Speicherbereich für die Allocation-Table, dann muss man den Zeiger ALV im DPH umbiegen und außerdem muss man noch herausbekommen wo der JOYCE sich die Länge der Allocation-Table merkt um auch diese zu ändern. Der JOYCE stürzt bei Formaten mit zu vielen Blöcken nicht einfach ab, sondern stellt schon beim Login der Diskette fest, dass er das Format nicht bearbeiten kann.

Hat man viele kleine Dateien kann aus dem scheinbaren Gewinn an Speicherplatz bedingt durch die höhere Blockgröße ein Verlust werden. Eine kurze Überschlagsrechnung hat ergeben, dass sich der zusätzliche Speicherplatz erst bei einer bestimmten mindesten Durchschnittsgröße der Dateien positiv bemerkbar macht. Darunter ergibt sich, bedingt durch die höhere Blockgröße, ein teilweise derber Verlust an Speicherplatz. Eine Gegenüberstellung der Formate ergibt folgende Werte:

Format	Größe	Mindestdateigröße
-----	-----	-----
80,9	706k	Referenzformat (mit 2k statt 4k Blockgröße)
83,9	740k	16k mittlere Dateigröße
80,10	784k	8k mittlere Dateigröße
83,10	824k	5k mittlere Dateigröße

Dramatische Werte ergeben sich bei mittleren Dateigrößen kleiner als ca. 2 kByte. Hier geht die Anzahl der Dateien, die auf die Diskette passen teilweise bis zu 25% zurück. Dafür passen dann bei einer durchschnittlichen Dateigröße über 8 kByte beim 828-kByte-Format vier Dateien mehr auf die Diskette. Das 740-kByte-Format sollte man lieber gleich vergessen.

Der Diskettencontroller

Im PCW treibt ein Floppycontroller vom Typ uPD765 sein Unwesen. NEC schlägt diesen Floppycontroller unter anderem für Z80-Systeme vor. Auch in den IBM-kompatiblen PC's ist er eingebaut. NEC schlägt ein Format mit 8 Sektoren zu 512 Byte vor. Eine andere Sektorgröße als 512 Byte braucht man, meiner Meinung nach, auf dem PCW gar nicht in Betracht ziehen (Wer will kann's ja trotzdem mal ausprobieren). Auf den meisten Computersystemen ist man, wie auch auf dem PCW, zu 9 Sektoren pro Spur übergegangen. Die Obergrenze liegt (512 Byte/Sektor vorausgesetzt) bei 10 Sektoren. Dann wird's auf der Spur aber schon recht eng.

Theoretisch passen, ausgehend von 5 Umdrehungen pro Sekunde und MFM-Aufzeichnung (Werte, die beim PCW vorgegeben sind) genau 6250 Byte auf die Spur. Für den Spurkopf reserviert der uPD765 146 Byte, für jeden Sektor weitere 62 Byte als Kopfinformation sowie 512 Datenbytes. Bei neun Sektoren ergibt das 5312 Byte, bei zehn Sektoren 5882 Byte reine Daten. Um die unvermeidlichen Schwankungen in der Laufwerksgeschwindigkeit abfangen zu können muss an Sektorende und am Spurende eine Sicherheitszone (GAP) eingerichtet werden. Zwischen der Länge der GAP beim Formatieren und beim Schreiben/Lesen muss unterschieden werden. Eine Nahrungsrechnung (nach HC-Sonderheft 13) liefert:

Sektoranzahl	Toleranz	GAP Schreib/Lese	GAP Format
8	0,0600	68	145
9	0,0348	39	81
10	0,0127	14	29

NEC hat nur einen Vorschlag für 8 Sektoren pro Spur zu bieten.

8	42	80
---	----	----

Amstrad benutzt auf dem PCW

9	42	82
---	----	----

Auf mit 10 Sektoren formatierten Disketten, die mir zuflogen, fand ich

10	6	12	für Laufwerk A:
10	12	23	für Laufwerk B:

Warum die GAP's sich bei den vorgefundenen 10-Sektor-Formaten unterscheiden ist mir nicht einsichtig. Die Näherungsrechnung spricht für den letztgenannten Fall. Auf meinem Rechner gibt's mit beiden Werten keine Probleme. Im Programm benutze ich (anders als in Version 3.6) den letztgenannten Wert.

Wie aus dem obigen Ausführungen zu ersehen ist erlaubt das Format mit 9 Sektoren Schwankungen in der Laufwerksgeschwindigkeit bis ca. 3,5%, bei 10 Sektoren pro Spur reduziert sich die erlaubte Toleranz auf 1,2%. Unter diesem Grenzwert muss das Laufwerk in jedem Fall bleiben damit die Aufzeichnung sicher erfolgen kann. Es ist also durchaus einzusehen, warum sich fast alle Betriebssysteme auf 9 Sektoren pro Spur beschränken, um eine höhere Datensicherheit zu erreichen. Wer genug Vertrauen in die Toleranz seines Laufwerks hat und nicht gerade lebenswichtige Daten auf seiner Diskette aufbewahrt der kann aber, wie man sieht, durchaus noch einige kByte aus seiner Diskette herauskitzeln.

Die zweite Methode, die von FORMAT angewandt wird, um die Diskettenkapazität zu erhöhen besteht darin zusätzliche Spuren auf dem Laufwerk einzurichten. Praktisch alle Laufwerke haben dafür noch Luft. Der Schlitz durch den der Schreib-/ Lesekopf auf die Diskette zugreift ist lang genug und auch die Spindel im Laufwerk die den Kopf auf die Spuren positioniert verträgt einige Umdrehungen mehr. Höhere Anforderungen an die Toleranz der Laufwerksdaten treten hier nicht auf.

Wie erkennt der PCW die geänderten Formate

Um das Format der eingelegten Diskette zu erkennen haben die Rechner- bzw. Betriebssystemhersteller sich verschiedene Wege einfallen lassen. Viele gehen einfach von einem bestimmten vorgegebenen Format aus und verweigern andere Disketten oder liefern dann schlicht Müll. Andere (z.B. MS-DOS) werten ein Byte im ersten Sektor der Diskette aus und schließen daraus auf das Format.

Die CPC's unterscheiden die Formate an unterschiedlichen Sektornummern, die Sektoren in den Spuren werden hier nicht ab 01h sondern ab 41h (System) bzw. C1h (Data only) nummeriert. An diesen Sektornummern erkennt auch der PCW die CPC-Formate und geht dann von jeweils 40 Spuren zu 9 Sektoren aus.

Bei PCW-eigenen Formaten werden Kopfinformationen im ersten Sektor der Diskette ausgewertet. Hier ist unter anderem auch die Zahl der Spuren und der Sektoren pro Spur abgelegt. Die grundsätzliche Entscheidung, ob es sich überhaupt um ein PCW-Format handeln kann, wird anhand einer Prüfsumme entschieden. Diese Prüfsumme ist eine Quersumme des ersten Diskettensektors. Bei Änderung der Spur- und Sektorinformationen ist also auch das Prüfbyte

im ersten Sektor anzupassen. Ein (leidlich kommentiertes) Listing des von FORMAT benutzten Bootsektors befindet sich in der Datei BOOT.MAC.

Der Name Bootsektor für den ersten Sektor leitet sich aus der Tatsache her, dass fast alle Betriebssysteme in diesem Sektor ein kurzes Programm erwarten das den Systemstart (das Booten) ermöglicht. Da es auch ohne genaue Kenntnis der Spur- und Sektorzahl der Diskette sowie des verwendeten Dateisystems immer möglich ist die erste Diskettenspur anzufahren und von dort der ersten Sektor einzulesen bietet er sich hierfür geradezu an.

Dateien

```
FORMAT.DOC - Eine (gekürzte) englische Dokumentation
FORMAT.DOK - Diese Dokumentation
FORMAT.PAS - Der Quelltext des Programmes
FORMAT1.PAS - Include-Datei dazu
FORMAT2.PAS - Include-Datei dazu
FORMAT3.PAS - Include-Datei dazu
BOOT.MAC   - Kommentiertes Listing des benutzten Bootsektors
FORMAT.COM - Das ausführbare Programm
```

Die Datei FORMAT.PAS musste aufgeteilt werden, da der Platz im Editor zu klein wurde. Ich habe versucht eine sinnvolle Aufteilung zu finden. Weiterhin habe ich verstärkt darauf geachtet, dass die Textdateien eine Breite von weniger als 80 Spalten erhalten. Ich weiß nicht, ob es mir immer gelungen ist.

Bis zur Version 3.5 hieß diese Datei FORMAT.DOC, jetzt FORMAT.DOK. Das ist zwar keine wesentliche Änderung, ich will sie dennoch kurz erläutern. Im Bereich der PD und ShareWare bin ich über viele DOC- und DOK-Dateien gestolpert. DOC wird vor allem für englische Dokumentationsdateien benutzt. Im deutschsprachigen Raum hat sich die Extension DOK durchgesetzt. Dieser durchaus sinnvollen Konvention will ich mich natürlich nicht verschließen. Bei dieser Gelegenheit habe ich auch gleich die ae's, oe's und so weiter in richtige ä's und ö's umgewandelt. In den Programmquelltexten werde ich aber weiterhin bei ae etc. bleiben. Dies hat einen einfachen Grund, da ich in Pascal die eckigen und geschweiften Klammern brauche, arbeite ich dort mit dem englischen Zeichensatz.

Neuübersetzung

Soll FORMAT nach einer Änderung neu übersetzt werden, dann ist darauf zu achten, dass die Endadresse des Compilers auf eine Adresse kleiner als 'TableAdr' (im Listing definiert, z.Zt. \$D000) gesetzt wird.

Was fehlt noch?

Das Nachformatieren der Spuren 40...42 auf Laufwerk A: kümmert sich nicht um das bisher auf der Diskette vorhandene Format. Eine Data-only-, System- oder 10-Sektor-Diskette wird also gnadenlos übergebügelt, hier sollte man noch was verbessern.

Beim Programmstart sollte geprüft werden, ob der mit ABSOLUTE-Deklarationen festgelegte Speicherbereich für 'MyTable' und 'MyXDPB' noch frei ist. Im Fehlerfalle sollte das Programm abgebrochen werden.

Was darf noch hinzukommen?

Dem Autoren spukt immer noch eins im Kopf herum: Um den Bootsektor anzulegen reserviert CP/M eine vollständige Spur auf der Diskette. Davon sind 8 (bzw.9) Sektoren völlig ungenutzt, damit müsste man doch etwas (mehr oder weniger) sinnvolles anfangen können. Wie wär's mit einem Logo das während des Bootvorganges angezeigt wird? Dazu müsste eigentlich nur der Aufbau des provisorischen Video-RAM's ein wenig aufgebohrt werden. Na, wenn ich mal wieder viel Zeit habe, dann vielleicht. Oder hat jemand anders Lust sich das vorzunehmen? Im Erfolgsfalle bitte ich um Nachricht.

Literatur

PC Schneider International 7/87
DMV-Verlag, 1987

PC Schneider International 6/88
DMV-Verlag, 1988

M. Anton
Im Herzen von CP/M Plus
XBIOS zerlegt
SONDERHEFT Joyce Nr. 1 1/87
DMV-Verlag, 1987

Borland
Turbo-Pascal-Handbuch

Rollke, Karl-Hermann
Das Turbo Pascal Buch
Sybex, 1986

Digital Research
CP/M Plus(tm) Operating System Programmer's Guide
Markt&Technik, 1986

Disketten - eine runde Sache
6. Schneider Sonderheft
Happy Computer Sonderheft 13
Markt&Technik, 1986

NEC Electronics (Europe) GmbH
1982 CATALOG
Firmenschrift
NEC, 1982

Schlussbemerkung

Der Autor ist offen für Anregungen zum Programm und Rückmeldung von Änderungen durch deren Autoren.

Allen, die das Programm gebrauchen können, wünsche ich viel Spaß mit FORMAT.COM. Wenn's euch besonders gut gefällt, dann schickt doch aus dem nächsten Urlaub mal eine Karte als Dankeschön.

Andreß Freystatzky, Braunschweig