Handbuch

volksFORTH
CP/M 2.2 und CP/M 3.Ø

volksFORTH für CP/M 2.2 und CP/M 3.Ø

Nach den Implementierungen von volksFORTH auf dem 65Ø2 (C64) und dem 68ØØØ (Atari ST), liegt hier nun die dritte Implementierung, die auf dem 8Ø8Ø/Z8Ø, vor. Sie stützt sich dabei auf das CP/M 2.2 Betriebssystem, sodaß volksFORTH damit auf einer großen Zahl von Mikrocomputersystemen zur Verfügung steht.

volksFORTH ist ein public domain Produkt, d.h. volksFORTH darf beliebig kopiert und weiterverbreitet werden, solange dies nicht kommerziell geschieht, und das Urheberrecht der FORTH-Gesellschaft e.V. beachtet wird.

Das volksFORTH für CP/M-Systeme wird momentan mit dem Handbuch zur C64'er-Version vertrieben. Da aber einige systemspezifische Teile erheblich verändert sind, werden zum Handbuch einige zusätzliche Blätter mit den entsprechenden Erklärungen mitgeliefert. Dieser Teil stammt vom CP/M-volksFORTH-Autor Ulrich Hoffmann und umfassen folgende Kapitel:

Kapitelüberschrift	Seitenanzahl
Einleitung	3
README.TXT	1
Allgemeines	1
Installierung	2
Editor	3
8Ø8Ø-Assembler	2
Fileinterface	11
Vokabular	6

FORTH und damit auch das volksFORTH ist eine sehr lebendige Programmiersprache. Wir (die FORTH-Gesellschaft e.V. und der Vertrieb) freuen sich über alle Erweiterungen und Verbesserungen, die an uns herangetragen werden. Fragen werden umgehend beantwortet, da sie auch evtl. auf schlecht verständliche oder unvollständige Dokumentation hindeuten. Bitte senden Sie alle Anfragen und Bestellungen, die speziell das volksFORTH betreffen, direkt an die nachfolgende Adresse. Dadurch wird einmal Zeit gespart und zum zweiten das ohnehin stark frequentierte FORTH-Büro entlastet.

Michael & Klaus Kohl
Pestalozzistr. 69
8Ø05 Mering

Die FORTH-Gesellschaft e.V. hat sich zur Aufgabe gestellt, die Programmiersprache FORTH populär zu machen. Diese "Öffentlichkeitsarbeit" umfaßt dabei folgende Punkte:

1. Ausgabe der Vereinszeitschrift "Vierte Dimension"
Diese Zeitschrift erscheint viermal im Jahr und wird an Mitglieder der FORTH-Gesellschaft kostenlos versandt. Es ist aber auch für Nichtmitglieder über das FORTH-Büro erhältlich und kostet momentan DM 7.5Ø pro Einzelexemplar.

2. Treffen in lokalen Gruppen
In mehreren Städten wie München, Hamburg oder Darmstadt treffen sich monatlich lokale FORTH-Gruppen. In diesen Gruppen werden zum einen bestimmte Projekte mit FORTH bearbeitet, aber auch Schulungen oder Informationsveranstaltungen zu FORTH durchgeführt. Eine vollständige Liste dieser Gruppen und die Adresse der Gruppenleiter kann über das FORTH-Büro erfahren werden.

3. Jahrestreffen der FORTH-Gesellschaft e.V.
Jedes Jahr findet einmal eine Vereinsversammlung statt. Dabei werden neben der vereinsinternen Mitgliederversammlung auch begleitende Informationsveranstaltungen von Einzelpersonen und Firmen durchgeführt.

4. euroFORML
Die FORTH-Gesellschaft e.V. ist ein Verein, der sehr enge Beziehung zu anderen FORTH-Clubs überall auf der Welt unterhält. Neben der alljährlichen amerikanischen Konferenz findet auch in Europa ein Treffen statt. Alle 2-3 Jahre (1987, 1989, ...) ist dieses Treffen in Deutschland, hat aber als Konferenzsprache Englisch.

5. Kopierservice
Ab Januar 1989 gibt es neben den Bibliotheken der lokalen Gruppen auch ein Kopierservice in Hamburg, bei den Artikel über FORTH bezogen werden können. Eine Liste der verfügbaren Artikel erhalten sie von:

FORTH-Gesellschaft e.V. - Kopierservice
Roter Hahn 42
2ØØØ Hamburg 72
Sonderkonto K / Postgiroamt Hamburg
Nr. 5226 46-2Ø3 / BLZ 2ØØ 1ØØ 2Ø

6. Mailbox
Auch für Freunde eines heißen Drahtes wird etwas geboten. Seit dem Januar 1989 ist wieder der FORTH-Tree in Hamburg unter der Nummer Ø4Ø-6444636Ø erreichbar. Es können mit 3ØØ Baud/8 Bit/no Parity Informationen und Programme abgerufen werden.

Die Adresse der Forth-Gesellschaft e.V. lautet:

FORTH-Gesellschaft e.V.
Postfach 111Ø
8Ø44 Unterschleißheim

Änderungen im CP/M-volksFORTH von Version 3.8Ø zu Version 3.8Øa:

(Ulrich Hoffmann Ø4 März 88)

Die Unverträglichkeit des ursprünglichen CP/M-volksFORTHs mit CP/M+ und die damit verbundene Vielzahl von unterschiedlichen Versionen hat eine allgemeine Überarbeitung des CP/M-volksFORTHs notwendig gemacht. Bei dieser Gelegenheit wurden gleich einige Fehler beseitigt und einige neue Funktionen eingeführt.

1. Änderungen im Kern (SOURCE.SCR)

- Die Terminal-Ein- und Ausgabe wurde auf ein Mindestmaß begrenzt, sodaß auch unmittelbar mit dem Kern gearbeitet werden kann. Es gibt keinen Zeileneditor für die Eingabezeile mehr, dieser wurde zusammen mit der "**Terminal:**"-Funktion in das File XINOUT.SCR ausgelagert.

- Der Kern enthält kein Fileinterface mehr, sondern arbeitet nur in dem File, daß bei Aufruf in der Kommandozeile mit angegeben wird (Defaultfile). Typischerweise wird mit diesem Mechanismus zuerst das File-Interface geladen.

- Direkter Diskettezugriff wird im Kern nicht mehr unterstützt, da er unter CP/M+ nicht problemlos zu implementieren ist. Außerdem kann in Ermangelung eines CP/M+ Systems der Code hier nicht getestet werden. Diskettenzugriff findet nur noch über das BDOS statt.

- Zahlreiche Funktionen des Kerns wurden neu überarbeitet und in Code geschrieben. Als wichtige neue Funktion des Kerns ist "**search** hinzugekommen, das eine schnelle Suche mit Berücksichtigung der Groß/Kleinschreibung ermöglicht.

- Die Funktion **CAPITALIZE** ist durch die ähnliche Funktion **UPPER** ersetzt worden. Das **EXIT** in **NAME** verschiebt sich dadurch.

- Der Kern gibt beim Verlassen eine Größenangabe in (256 Byte)-Seiten aus. Diese Angabe kann direkt benutzt werden, um mit dem CP/M SAVE Kommando das System auf Diskette zu schreiben. (Forth: **SAVE** nicht vergessen!)

- **SAVE-BUFFERS** ist um ein defered Wort **SAVE-DOS-BUFFERS** erweitert worden. Damit sollte der lästige CP/M+ Fehler ausgeschaltet sein.

- Das defered Wort **POSTLUDE** regelt die letzte Handlung des Systems vor dem CP/M Warmstart (Cursor anschalten, Bildschirm löschen oder Systemgröße ausgeben...)

- Die Kommandozeile des Aufrufs wird in den TIB kopiert und kann dort interpretiert werden. Das Öffnen des default-Files löscht allerdings den TIB wieder, sodaß diese Funktion erst ausgenutzt werden kann, wenn das Fileinterface geladen ist. (**DRVINIT** öffnet nicht mehr das Defaultfile.)

- Die Interpret-Loop wurde überarbeitet und um das Wort **PROMPT** erweitert. Das Sonderwort **>INTERPRET** ist weggefallen. Seine Funktion übernimmt jetzt das (normale) defered Wort **PARSER**.

- Die Kontrollstruktur-Anweisungen (**IF**, **WHILE** ...) sind jetzt auch interaktiv verwendbar.

- Diverse kleinere Änderungen haben stattgefunden.

2. Änderungen im Editor (EDITOR.SCR, STRING.SCR)

- Das Markieren der Screens wurde korrigiert und geschieht jetzt auch beim Suchen/Ersetzen und bei showload richtig.

- **VIEW** wurde geändert und sucht nun nach dem in Blanks eingerahmten Wort.

- Es wird nun zusätzlich das Associative File angezeigt.

- Beim Suchen/Ersetzen wird die Screennummer hochgezählt, um eine Kontrolle über das Suchen zu geben.

- Die Textsuche ist nun schon im Kern definiert, die elementaren Stringfunktionen sind mit in das EDITOR.SCR genommen worden. STRING.SCR ist daher entfallen.

3. Änderungen im Multi-Tasker (TASKER.SCR)

- Das Wort **TASK** wurde geändert: Die Konstante ist nun vor der Task definiert. Man kann also nun mit **FORGET** <taskname> tatsächlich die Task vergessen.

- Der PAUSE/WAKE/STOP-Mechanismus wurde geändert. In der Benutzung ergibt sich daraus keine Änderung.

4. Änderungen im Fileinterface (FILEINT.SCR)

- Das Fileinterface wurde überarbeitet und einige Fehler beseitigt. Die Namen zahlreicher Worte haben sich geändert, sind dadurch aber systematischer geworden. Die Funktionen sind im Wesentlichen gleich geblieben.

5. Terminal-Installation (Zusatz zu Anpassung von volksFORTH an den Computer)

- Da der Kern kein Fileinterface mehr enthält, muß dies noch vor dem Primitivst-Editor geladen werden. Es ergibt sich also die Kommandosequenz:

```
A>      kernel fileint.scr
        1 load
        use primed.scr  1 load
        use terminal.scr
```

6. Erstellen eines Standard-Systems

- Mit folgender Kommandosequenz wird aus KERNEL.COM das File VOLKS4TH.COM gemacht:

```
A>      kernel fileint.scr
        1 load
        include startup.scr
```

7. Neue Files auf der Diskette

READ.ME	dieses File
XINOUT.SCR	Terminalfunktionen und Zeileneditor für Eingabe
COPY.SCR	Die Funktionen **COPY** und **CONVEY** (früher im Kern).
STRING.SCR	Entfällt, da in EDITOR.SCR und SOURCE.SCR integriert.

Willkommen zu volksFORTH!

Damit das volksFORTH-System auf dem weitverbreiteten Diskettenformat Osborne SS DD untergebracht werden kann, mußten die Files mit einem Kompaktifizierungsverfahren behandelt werden.

Um die Files wieder auf normale Größe zu bringen sind folgende Schritte zu unternehmen:

- Das Public Domain Programm NSWP starten. (Mit ? kann die Liste der NSWP-Befehle dann abgerufen werden.)
- mit T nun alle Files markieren, die expandiert werden sollen. (Komprimierte Files haben ein Q als zweiten Buchstaben in der Extension.)
- mit Q, folgendem U und Angabe des Ziellaufwerks nun die Files expandieren.

Beispiel: Expandieren des Source-Codes für den Assembler und den Disassembler

```
A>   nswp

NSWEEP   -   Version 2.07   07/17/1984
(c) Dave Rand, 1983, 1984
Edmonton, Alberta

Drive A0:????????.???    54K in   12 files.   858K free.
   1. A0: ASS8080 .SQR    6K : t  Tagged files =    6K (   5K).
   2. A0: ASSTRAN .SQR    2K :
   3. A0: COPY    .SQR    2K :
   4. A0: DISASS  .SQR    6K : t  Tagged files =   12K (  10K).
   5. A0: DOUBLE  .SQR    2K : q
  Squeeze, Unsqueeze or Reverse (S,U,R)? u
  Copy to drive/user? B0:
SQ/USQ    --> A0: ASS8080 .SQR to B0:(ASS8080.SCR)
SQ/USQ    --> A0: DISASS  .SQR to B0:(DISASS.SCR)

   5. A0: DOUBLE  .SQR    2K : x
A>
     ...
```

Um aus dem File KERNEL.COM ein (neues) VOLKS4TH.COM zu erzeugen, als erstes in STARTUP.SCR (STARTUP.SQR expandieren) nachsehen, welche Files benötigt werden, dann diese zusammen mit KERNEL.COM auf eine Diskette bringen. Desweiteren der Beschreibung im Handbuch bzw. in READ.ME folgen.

Viel Spaß,

UH März 88

volksFORTH und der Platzmangel auf Disketten

Das volksFORTH Softwaresystem, so wie es zur Zeit in der Version 3.8Ø für CP/M 2.2 ausgeteilt wir, hat einen Umfang von ca. 4ØØ kB, zu groß für einige Diskettenformate.

Um diesem Dilemma zu entgegnen haben wir dem volksFORTH System das Public-Domain-Programm **NSWP.COM** (inclusive Doku) beigefügt, und die Screenfiles damit bearbeitet (Huffman-Kompression bringt Screenfiles typischer Weise auf 1/3 bis 1/2 der Größe).

Um nun mit dem System arbeiten zu können, müssen die Screenfiles zuerst wieder expandiert werden. Dies geschieht auch mit Hilfe von NSWP.COM.

1) NSWP aufrufen.
2) mit W *.SQR alle komprimierten Screenfiles anwählen
3) mit Q und U und der Angabe einer leeren (!) Diskette Files expandieren.

Als dann viel Erfolg und Spaß,

Ulrich Hoffmann

volksFORTH fÜr CP/M 2.2 und CP/M 3.Ø

Nach den Implementierungen von volksFORTH auf dem 65Ø2 (C64) und dem 68ØØØ (Atari ST), liegt hier nun die dritte Implementierung, die auf dem 8Ø8Ø/Z8Ø, vor. Sie stützt sich dabei auf das CP/M 2.2 Betriebssystem, sodaß volksFORTH damit auf einer großen Zahl von Mikrocomputersystemen zur Verfügung steht.

Das volksFORTH unter CP/M 2.2 und CP/M 3.Ø kostet DM 85,- bzw. DM 9Ø,- (für 3"-Disketten). Für Schneider Computer ist es auf 3"-Disketten oder 5 1/4"-Diskette (Vortex-Format) erhältlich.

Für andere CP/M-Computer kann es z.Zt. nach Absprache des zu verwendenden Formats auf 51/4" Disketten ausgeliefert werden. Alle Versionen werden mit Handbuch geliefert.

Die allgemeine CP/M-Version von volksFORTH enthält

- den volksFORTH-Kern,
- den Multitasker,
- den 8Ø8Ø-Assembler,
- einen Full-Screen-Editor,
- das Printerinterface,
- den Tracer und
- das CP/M-Fileinterface

Speziell für die Schneider-Computer sind:

- das Grafikpaket (Linien, Punkte, Vielecke...),
- die Turtle-Grafik und
- Grafik-Demos

Um volksFORTH auf einem beliebigen CP/M-Computer laufen zu lassen (alle Möglichkeiten außer der Schneider-Grafik) sind lediglich die Bildschirmfunktionen (wie etwa Bildschirmlöschen) anzupassen.

Auch die Befehlstasten des Full-Screen-Editors lassen sich leicht den eigenen Bedürfnissen entsprechend verändern. Dieser Vorgang der Installation ist von uns für die Schneider-Computer schon vorgenommen worden, für andere Computer steht eine ausführliche Anleitung zur Verfügung.

Das CP/M-Fileinterface läßt ein bequemes Arbeiten mit CP/M-Files zu, so können auch hier Files, die von anderen Programmen erzeugt wurden, weiterverarbeitet werden.

Wie fange ich an?

In diesem Text soll der Vorgang der Installation von volksFORTH an ein CP/M-Computersystem (Bildschirm, Tastatur, Drucker) beschrieben werden.

Auf der ausgelieferten Diskette finden sich folgende Files:

ASS8Ø8Ø	SCR	Der volksFORTH 8Ø8Ø-Assembler
ASSTRAN	SCR	Zum Laden des Assembler auf den Heap
DISASS	SCR	Ein Z8Ø-Disassembler für volksFORTH
DOUBLE	SCR	Definitionsn für doppeltgenaue Zahlen
EDITOR	SCR	Der volksFORTH Full-Screen Editor
FILEINT	SCR	Das volksFORTH Fileinterface zu CP/M 2.2
HASHCASH	SCR	Ein schnelles Dictionary-Suchverfahren
INSTALL	SCR	Der Installer für die Editor-Befehlstasten
KERNEL	COM	Der volksFORTH Kern (Terminal unabhängig)
PORT8Ø8Ø	SCR	Definitionen für 8Ø8Ø Portzugriff
PORTZ8Ø	SCR	Definitionen für Z8Ø Protzugriff
PRIMED	SCR	Der primitivst Editor zum Installieren
PRINTER	SCR	Anpassung von volksFORTH an den Drucker
RELOCATE	SCR	Das Utility-Wort BUFFERS
SAVESYS	SCR	Das Utility-Wort SAVESYSTEM
SEE	SCR	Der automatische Decompiler
SIMPFILE	SCR	Ein einfaches Filesystem für Direktzugriff
SOURCE	SCR	Der Quelltext des volksFORTH Kerns
STARTUP	SCR	Macht aus KERNEL.COM das VOLKS4TH.COM
STRING	SCR	Definitionen für Stringoperationen
TASKER	SCR	Der volksFORTH Multitasker
TERMINAL	SCR	Definitionen für das installierte Terminal
TIMES	SCR	Die Utility-Worte OFTEN und TIMES
TOOLS	SCR	Der manuelle Decompiler, DUMP und Tracer
VOLKS4TH	COM	Das volksFORTH Standard-System

```
************************ A C H T U N G ************************
*                                                             *
*     Bevor Sie irgend etwas mit dem System ausprobieren :    *
*    Machen Sie Sicherheitskopien von den Orginaldisketten.   *
*                                                             *
***************************************************************
```

Ø) Drei wichtige Worte: USE, LIST und LOAD

volksFORTH bearbeitet seine Programmtexte in sogenannten Screen Files (Nachname: .SCR), das sind Files, die in 1 kB große Screens aufgeteit sind, die wiederum in 16 Zeilen mit je 64 Zeichen strukturiert sind. Um ein schon existentes File als aktuelles File anzuwählen wird das Wort **USE** <filename> benutzt. (Beispiel: **USE TERMINAL.SCR**, wählt TERMINAL.SCR als aktuelles File.) Um sich nun einen bestimmten Screen anzusehen, wird nn LIST benutzt.
(Beispiel: 1 **LIST**, zeigt Screen 1 des aktuellen Files.)

Mit **nn LOAD** wird ein bestimmter Screen geladen:
Die Definitionen in diesem Screen werden in eine für den Computer ausführbare Form gebracht.
(Beispiel: **1 LOAD**, lädt Screen 1 des aktuellen Files.)

Per Konvention soll der Screen null eines jeden Files eine Erklärung des Inhaltes des Files enthalten. Wird Screen eins, der sogenannte LOAD-Screen, geladen, so soll er das Laden der gesamten Definitionen des Files veranlassen.
Zeile Null eines jeden Screens soll Auskunft über den Inhalt des Screens geben.

1) Die Anpassung von volksFORTH an den Computer

Damit das volksFORTH in vollem Umfang benutzt werden kann, ist zunächst eine Installation erforderlich. Für Schneider-Computer ist diese schon von uns vorgenommen worden, sodaß es gleich richtig losgehen kann.

Die Anpassung an einen anderen Computer beinhaltet:

a) Anpassung der Bildschirmfunktionen

In dem File TERMINAL.SCR werden die notwendigen Bildschirmfunktionen definiert. Diese müssen auf den neuen Bildschirm angepaßt werden.
Da der Editor erst nach erfolgreicher Anpassung benutzt werden kann, müssen diese Screens auf andere Art und Weise geändert werden.
Dazu kann der Primitivst-Editor im File PRIMED.SCR benutzt werden.
Die normalerweise zu benutzenden Escape-Sequenzen, sind dem entsprechenden Terminal-Handbuch zu entnehmen. Mit **USE PRIMED.SCR 1 LOAD** den primitivst Editor laden. (Screen Ø enthält Anleitung, Screen 2 ein Beispiel). Dann mit **USE TERMINAL.SCR** dieses File zur Benutzung anwählen. **PRIMED** arbeitet dann auf diesem File.

b) Anpassung der Editor-Befehlstasten

Im File EDITOR.SCR gibt es eine Tabelle mit Namen **KEYTABLE**, in der die Tasten zu den in der Tabelle **ACTIONTABLE** definierten Befehlen angegeben werden. Durch Ändern der Tabelle **KEYTABLE** können die Befehlstasten des Editors verändert werden.
Zum einfachen Anpassen des Editors gibt es das File INSTALL.SCR, indem interaktiv die neuen Befehlstasten abgefragt werden.
(Achtung!: Der Sourcetext wird nicht mitgeändert!!)

2) Die Anpassung von volksFORTH an den Drucker

In dem File PRINTER.SCR wird die Ansteuerung des Druckers (hier Epson FX8Ø) definiert. Sollte kein Epson-kompatibler Drucker vorliegen, müssen auch hier die Escape-Sequenzen geändert werden. (Siehe Druckerhandbuch!)
Dies sollte aber möglichst erst dann geschehen, wenn die restlichen Anpassungen laufen!

volksFORTH-Editor für CP/M 2.2 Implementation

Der für die CP/M 2.2 Version von volksFORTH benutzte Editor enthält im wesentlichen die gleichen Funktionen wie die des Editors auf dem C64, beschrieben im Handbuch Seite 147ff.

Im Gegensatz zum C64-Editor benutzt der neue Editor das Forth-Screen-Standardformat von 16 Zeilen a 64 Zeichen.

Aufgerufen wird der Editor mit **<screennummer> L.** Den zuletzt edititierten Screen erhält man mit: V, und mit **VIEW <name>** kann man sich ansehen, wo <name> definiert worden ist.

Die Tastenbelegung ist neu organisiert. Im folgenden wird die Tastenbelegung für die allgemeine CP/M-Version und für die Schneider Version beschrieben. Mit Hilfe des Files INSTALL.SCR können die Tasten neu angepaßt werden.
(Siehe Installationshinweis: "Wie fange ich an?")

Tastenbelegung für die allgemeine CP/M Version:

Komando (siehe Handbuch S. 152ff)	Taste
Cursor up	Control E
Cursor left	Control S
Cursor down	Control X
Cursor right	Control D
push-line	Control I
push-char	Control J
pull-line	Control O
pull-char	Control K
copy-line	Control P
copy-char	Control L
backspace	Control H
backspace	delete
delete-char	Control G
insert-char	Control T
delete-line	Control Y
insert-line	Control N
insert-mode-on insert-mode-off	Control V
clear-to-right	Control Z
new-line	return
+tab	Control F
-tab	Control A
search	Control \ = Control \
undo	Control U
update-exit	Control Q
flushed-exit	escape
shadow-screen	Control W
next-screen	Control C
back-screen	Control R
alter-screen	Control [= Control [
mark-alter-screen	Control B

Tastenbelegung für die Schneider CP/M Version:

Komando (siehe Handbuch S. 152ff)	Taste
Cursor up	Pfeil nach oben
Cursor left	Pfeil nach links
Cursor down	Pfeil nach unten
Cursor right	Pfeil nach rechts
push-line	shift Pfeil nach oben
push-char	shift Pfeil nach links
pull-line	shift Pfeil nach unten
pull-char	shift Pfeil nach rechts
copy-line	Control Q
copy-char	Control Z
backspace	Control H
backspace	delete
delete-char	Control P (clr)
insert-char	copy
delete-line	Control D
insert-line	Control T
insert-mode-on	Control I
overwrite-mode-on	Control O
eraser-line	Control C
clear-to-right	Control E
new-line	return
+tab	Control Pfeil nach rechts
-tab	Control Pfeil nach links
home	Control Pfeil noch oben
to-end	Control Pfeil nach unten
search	Control F
undo	Control U
update-exit	Control X
flushed-exit	escape
show-load	Control L
shadow-screen	Control W
next-screen	Control N
back-screen	Control B
alter-screen	Control A
mark-alter-screen	Control R

Der volksForth-8Ø8Ø-Assembler

Die CP/M-Version von volksFORTH ist mit einem Assembler für den Intel 8Ø8Ø ausgestattet. Dieser Assembler kann aber auch unter den anderen Versionen geladen werden und so als Cross-Assembler arbeiten.

Diese Beschreibung enthält kein vollständiges Glossar, da die Mnemonics des Assemblers den meisten Programmierern vertraut sein dürften. Sie dient als Ergänzung der Beschreibung des 65Ø2-Assemblers im UltraForth83-Handbuch Seite 175ff.

Eine genaue Darstellung der Funktionsweise findet sich in dem Artikel von John J. Cassady in den FORTH-Dimensions (Jahrgang III/6 Seite 18Øf), an dessen Implementation sich die volksFORTH-Version anlehnt.

Der 8Ø8Ø-Assmebler erlaubt strukturierte Programmierung. Er verwendet die gleichen Strukturelemente, wie der 65Ø2-Assembler. Vor den Kontrollstrukturen sind folgende Condition Codes zulässig:

```
cØ=    cØ<>   cs   Ø=   Ø<>   pe   Ø<   Ø>=
```

Sie entsprechen den Flags im Processor Status Word des 8Ø8Ø. Neben den Kontrollstrukturen gibt es auch noch absoluten Sprünge (jc, jm, jmp, jnc, jnz, jp, jpe, jpo, jz).

Beispiele für die Verwendung des 8Ø8Ø-Assemblers:

volksFORTH	Intel
A xra	xra A
A L mov	mov L,A
Ø H mvi	mvi H,Ø
H pop	pop H
vector lxi	lxi vektor
D dad	dad D
...	...

Die Belegung der Forth-Register sieht folgendermaßen aus:

IP	im BC-Registerpaar
W	im DE-Registerpaar
SP	im SP
UP	im Speicher
RP	im Speicher

Die beiden 8-Bit-Hälften von IP und W können auch getrennt angesprochen werden durch (IP und IP', bzw. W und W'). Zum Ansprechen der 8Ø8Ø-Register dürfen die FORTH-Namen sowie die Intel Namen benutzt werden.

Zusätzlich enthält das System noch mehrere Macros:

R rpop	Hole das 16-Bit-Register R (R<>H) vom Returnstack.
R rpush	Bringe das 16-Bit-Register R (R<>H) zum Returnstack.
R1 R2 mvx	Kopiere 16-Bit-Register R1 nach R2.
Next	Springe zum Address-Interpreter.
;c:	Schalte den Assembler ab und den Forth-Compiler an.

Vordefinierte Labels sind:

Hpush	Adresse der Routine, die das H-Register auf den Stack bringt und dann zu Next springt.
Dpush	Adresse der Routine, die das D- und H-Register auf den Stack bringt und dann zu Next springt.
>Next	Adresse des Address-Interpreters.
UP	Adresse der Speicherzelle für den User-Pointer
RP	Adresse der Speicherzelle fürden Returnstack-pointer
IPsave	Adresse einer Hilfszelle um den IP zwischenzuspeichern

Neue Labels können mit **>LABEL** und **LABEL** erzeugt werden, wie im 65Ø2-Assembler.

Fileinterface für das volksFORTH83 auf CP/M Computer

Wie geht es los?

Bevor Sie das Glossar lesen, sollten Sie diese kleine Einführung lesen und auf einer leeren Diskette die Beispiele ausprobieren.

Wie erzeuge ich ein File, in das ich ein Programm eingeben kann?

Geben Sie bitte folgendes ein: **MAKEFILE test.scr**

Das File test.scr wird auf der Diskette erzeugt, auf dem Sie das Forth gebootet haben.

Als nächstes schätzen Sie bitte ab, wie lang Ihr Programm etwa wird. Beachten Sie dabei bitte, daß der Screen Ø eines Files für Hinweise zur Handhabung Ihres Programms und der Screen 1 für einen sog. Loadscreen (das ist ein Screen, der den Rest des File lädt) reserviert sind. Wollen Sie also z.B. 3 Screens Programm eingeben, so muß das File 5 Screens lang sein;

Sie geben also ein: **5 MORE**

Fertig! Sie haben jetzt ein File, das die Screens Ø..4 enthält.

Geben Sie jetzt **1 L** ein.

Sie editieren jetzt den Screen 1 Ihres neuen Files test.scr. Sie können, falls der Platz nicht ausreicht, Ihr File später einfach mit MORE verlängern. Ein File kann leider nicht verkürzt werden.

Wie spreche ich ein bereits vorhandenes File an?

Das geht noch einfacher. Geben Sie einfach den Filenamen ein. Reagiert das System mit der Meldung "Haeh?", so kennt das Forth dieses File noch nicht. Sie müssen in diesem Fall das Wort **USE** vor dem Filenamen eingeben, also z.B. **USE test.scr**

Jetzt können Sie wie oben beschrieben mit **1 L** (oder einer anderen Zahl) das File editieren. Das Wort USE erzeugt übrigens im Forthsystem das Wort **TEST.SCR**, falls es noch nicht vorhanden war. Wissen Sie also nicht mehr, ob Sie ein File schon benutzt haben, so können Sie mit **WORDS** nachsehen oder das Wort **USE** voranstellen.

Wie erzeuge ich ein File auf einem vorgegebenem Laufwerk?

Durch Voranstellen des Laufwerks etwa:

MAKEFILE a:test.scr

Oder durch Eingabe von:

A:

Hierbei wird A: zum aktuellen Laufwerk gemacht. Files ohne Laufwerksangabe werden immer auf dem aktuellen Laufwerk erzeugt.

Ø) Allgemeines

Im folgenden wird die Benutzung des Fileinterfaces beschrieben. Dieses Fileinterface benutzt die Files des CP/M.

Benutzt man ein File von Forth aus, so wird es in Blöcke zu je 1Ø24 Bytes aufgeteilt, die in gewohnter Wiese anzusprechen sind. Dies trifft auch für Files zu, die nicht vom Forth aus erzeugt wurden. Als Konvention wird vorgeschlagen, daß Files, die Forth-Screens, also Quelltexte, enthalten, mit **.SCR** erweitert werden. Files, die Daten enthalten, die nicht unmittelbar lesbar sind, sollten auf **.BLK** enden.

Zum Umschalten vom Filesystem auf Direktzugriff und umgekehrt gibt es das Wort:

DIRECT (--) "direct"
Schaltet auf Direktzugriff um. Auf den Filezugriff schalten wir durch das Nennen eines Filenamens.

1) Die Laufwerkswahl

Files werden immer auf dem aktuellen Laufwerk erzeugt, solange der Filename nicht ausdrücklich ein anderes Laufwerk vorsieht. Als Betriebssystemname wird dann der vollständige Filename eingetragen, also mit eindeutig festgelegtem Laufwerk.

Zum Ändern des aktuellen Laufwerks stehen die folgenden Worte zur Verfügung:

A: (--) "a-colon"
Macht Diskettenstation A: zum aktuellen Laufwerk entsprechend der Funktion im CCP.
Siehe **SETDRIVE**.

B: (--) "b-colon"
Macht Diskettenstation B: zum aktuellen Laufwerk entsprechend der Funktion im CCP.
Siehe **SETDRIVE**.

SETDRIVE (n --) "setdrive"
Macht die Diskettenstation mit der Nummer n zum aktuellen Laufwerk. Hierbei entspricht n=Ø der Diskstation A, n=1 der Diskstation B usw.

Um sich den Inhalt einer Diskette anzusehen, gibt es die Worte:

FILES (--) "files"
Listet den Inhalt des aktuellen Laufwerks (siehe **SETDRIVE**) auf dem Bildschirm auf. Dieses Wort, zusammen mit dem Wort **FILES"** entspricht dem Kommando DIR des CCP. In anderen volksFORTH-Filesystemen wird **DIR** benutzt um Direktories umzuschalten (MS-DOS, GEM-DOS).

FILES" (--) "files-quote"
Benutzt in der Form **FILES" cccc"**
Listet die Files auf, deren Name cccc ist. Der String cccc darf die bekannten Wildcards ('?','*') sowie eine Laufwerksbezeichnung enthalten. Wird kein Laufwerk angegeben, so werden die Files des aktuellen Laufwerks ausgegeben.

2) Files

Files bestehen aus einem Forthname und einem Betriebssystemnamen, die nicht übereinstimmen müssen.

Ist das Forthwort, unter dem ein File zugreifbar ist, gemeint, so wird im folgenden vom Forthfile gesprochen. Ist das File auf der Diskette gemeint, das vom CP/M-BDOS verwaltet wird, so wird vom DOS-File gesprochen. Durch das Nennen des Forthnamens wird das Forthfile (und das zugehörige DOS-File) zum aktuellen File, auf das sich alle Operationen wir **LIST**, **LOAD**, **CONVEY** usw. beziehen. Beim Bekanntmachen des Files mit **USE**, **MAKEFILE** und **ASSIGN** u.a. wird das File auf dem aktuellen Laufwerk gesucht, wenn kein Laufwerk im Namen angegeben wird. Danach darf das aktuelle Laufwerk beliebig geändert werden, ohne daß das File dann auf einem anderen Laufwerk gesucht wird. Mit **FORTHFILES** können die aktuellen Zuordnungen zwischen Forthfile und DOS-File angezeigt werden.

FILE (--) "file"
Wird in der Form **FILE <name>** benutzt.
Erzeugt ein Forthwort mit Name <name>. Wird <name> später ausgeführt, so vermerkt es sich als aktuelles File. Ebenso vermerkt es sich als **FROMFILE**, was für **CONVEY** wichtig ist. Einem Forthfile wird mit **MAKE** oder **ASSIGN** ein DOS-File zugeordnet.

MAKE (--) "make"
Wird in der Form **MAKE** cccc benutzt.
Erzeugt ein DOS-File mit Namen cccc auf dem aktuellen (oder angegebenem Laufwerk) und ordnet es dem aktuellen Forthfile zu. Das File wird auch gleich geöffnet. Es hat die Länge Null (siehe **MORE**).
Beispiel FILE ausgabe
ausgabe MAKE test.scr
erzeugt ein Forthwort **AUSGABE** und ein File mit dem Namen A:TEST.SCR. (Angenommen A: ist aktuelles Laufwerk.)
Alle Operationen wie **LOAD**, **LIST** usw. beziehen sich nun auf den entsprechenden Screen in A:TEST.SCR. Beachten Sie bitte, daß dieses File noch leer ist, und daher eine Fehlerbedingung besteht, wenn Zugriffsoperationen ausgeführt werden sollen.

MAKEFILE (--) "makefile"

Wird in der folgender Form benutzt:

MAKEFILE <name>

Erzeugt ein Forthfile mit dem Namen <NAME> und erzeugt abschließend ein DOS-File mit demselben Namen (und eindeutiger Laufwerksangabe). Die folgende Sequenz würde genau dasselber bewirken:

FILE <name>
<name> MAKE <name>

SAVEFILE (addr len --) "savefile"

Wird in der folgenden Form benutzt:

SAVEFILE <name>

Schreibt den String, der an der Adresse addr begint und die Länge len hat als File mit dem Namen <name> auf die Diskette.

KILLFILE (--) "killfile"

Löscht das aktuelle File. Unschön, da dann das Forthfile noch existiert, das Dosfile aber gelöscht ist, sodaß es bei dem nächsten Diskettenzugriff einen Fehler gibt, wenn nicht ein anderes File angewählt wird.

ASSIGN (--) "assign"

Wird in der Form ASSIGN cccc benutzt.
Ordnet dem aktuellen File das DOS-File mit Namen cccc (mit eindeutiger Laufwerksangabe) zu. Eine Fehlerbedingung besteht, wenn das File nicht gefunden werden kann.

USE (--) "use"

Dieses Wort ist das wichtigste Wort zum Auswählen von Files.
Es wird in der folgenden Form benutzt:

USE <name>

Dieses Wort macht das File mit Namen <NAME> zum aktuellen File, auf das sich **LOAD**, LIST usw. beziehen. Es erzeugt ein Forthfile mit Namen <NAME>, falls der Name noch nicht vorhanden war. Anschließend wird das File auf dem aktuellen (oder angegebenem) Laufwerk gesucht. Wird das File nicht gefunden, so wird eine Fehlermeldung ausgegeben. Das (automatisch) erzeugte Forthfile verbleibt im Dictionary und muß ggf. mit **FORGET** vergessen werden.

CLOSE (--) "close"

Schließt das aktuelle File. Dabei wird das Inhaltsverzeichnis (Directory) der Diskette aktualisiert. Es werden die zu diesem File gehörenden geänderten Blöcke auf Diskette zurückgeschrieben und alle zu diesem File gehörenden Blöcke in den Block-Puffern gelöscht.

OPEN (--) "open"

Öffnet das aktuelle File. Eine Fehlerbedingung besteht, wenn das File nicht gefunden werden kann. Die Benutzung dieses Wortes ist in den meisten Fällen überflüssig, da Files automatisch bei einem Zugriff geöffnet werden.

EMPTYFILE (--) "emptyfile"

Kürzt das aktuelle File auf die Länge null.

FROM (--) "from"

Wird in der folgenden Form benutzt:

FROM <name>

<name> ist der Name eines Forthfile, aus dem beim Aufruf von **CONVEY** und **COPY** Blöcke herauskopiert werden sollen.

Beispiel: filea 1 FROM fileb 3 COPY

Kopiert den Block 1 aus FILEB auf den Block 3 von FILEA. Dieses Wort benutzt **USE** und das File auszuwählen. Das bedeutet, daß FILEB automatisch als Forthfile angelegt wird, wenn es noch nicht im System vorhanden ist.

LOADFROM (n --) "loadfrom"

Wird in der folgenden Form benutzt:

LOADFROM <name>

<name> ist der Name eines Forthfiles, aus dem der Block n geladen wird.

Beispiel: 15 LOADFROM filea

Lädt den Block 15 aus FILEA. Dieses Wort ist wichtig, wenn während des Ladens eines Files Teile eines anderen Files geladen werden sollen. Dieses Wort benutzt USE, um FILEA zu selektieren. Das bedeutet, daß automatisch ein Forthfile mit Namen FILEA erzeugt wird, falls es im System noch nicht vorhanden war.

Beachten Sie bitte, daß dieses Wort nichts mit FROM oder FROMFILE zu tun hat, obwohl es ähnlich heißt!

INCLUDE (--) "include"

Wird in der folgenden Form benutzt:

INCLUDE <name>

<name> ist der Name eines Forthfiles, das vollständig geladen wird. Dabei ist Voraussetzung, daß auf Screen 1 dieses Files Anweisungen stehen, die zum Laden aller Screens dieses Files führen.

Siehe auch **LOADFROM**.

CAPACITY (-- u) "capacity"

u ist die Länge des aktuellen Files in Forth-Blöcken (1Ø24 Bytes). Beachten Sie bitte, daß die Länge des Files um eins größer ist als die Nummer des letzten Blocks, da der Block Ø mitgezählt wird.

FORTHFILES (--) "forthfiles"

Druckt eine Liste aller Forthfiles, zusammen mit den Namen der zugehörigen DOS-Files, deren Länge und deren Status (geöffnet / geschlossen).

FROMFILE (-- addr) "fromfile"

Addr ist die Adresse einer Variablen, die auf das Forth-File zeigt, aus dem **COPY** und **CONVEY** Blöcke lesen. Bei Nennen eines Forthfiles wird diese Variable gesetzt.

Siehe auch **FROM**

LOADFILE (-- addr) "loadfile"

Addr ist die Adresse einer Variablen, die auf das Forthfile zeigt, das gerade geladen wird. Diese Variable wird bei Aufruf von **LOAD**, **THRU** usw. auf das aktuelle File gesetzt.

ISFILE (-- addr) "isfile"
Addr ist die Adresse einer Variablen, die auf das aktuelle Forthfile zeigt. Sie wird bei Ausführung eines Forthfiles gesetzt.

FILE? (--) "file-question"
Druckt den Namen des aktuellen Forthfiles.

MORE (n --) "more"
Verlängert das aktuelle File um n Screens. Die Screens werden hinten angehängt. Anschließend wird das File geschloseen.

EOF (-- f) "end-of-file"
f ist ein Flag, das wahr ist, falls über das Ende des Files hinausgelesen wurde. f ist falsch, falls auf den zuletzt gelesenen Block noch weitere folgen.

3) Verschiedenes

Beim Vergessen eines Forth-Files mit Hilfe von **FORGET**, **EMPTY** usw. werden automatisch alle Blockpuffer, die aus diesem File stammen, gelöscht, und, wenn sie geändert waren, auf die Diskette zurückgeschrieben. Das File wird anschließend geschlossen.

Bei Verwendung von **FLUSH** werden alle Files geschlossen. **FLUSH** sollte VOR jedem Diskettenwechsel ausgeführt werden, und zwar nicht nur, um die geänderten Blicke zurückzuschreiben, sondern auch damit alle Files geschlossen werden. Sind nämlich Files gleichen Namens auf der neuen Diskette vorhanden, so wird sonst eine abweichende Länge des neuen Files vom Forth nicht erkannt.
Nach dem Diskettenwechsel verlangt CP/M das "einloggen" der neuen Diskette.
Dies geschieht mit DOS RESET. Wenn dies vergessen wird, so erhält man nach einem Schreibversuch auf die neue Diskette "BDOS-ERROR ON xx R/O" und landet zu allem Überfluß im CCP. Warum?? Fragen Sie Digital Research!

Bei Verwendung von **VIEW** wird automatisch das richtige File geöffnet.

4) CP/M 2.2. interne Worte des Filesystems (Implementation)

In diesem Abschnitt findet sich das Glossary für die Worte, die zur Implementation des Filesystems benutzt werden. Da das Filesystem noch recht neu ist, sind noch fast alle Namen sichtbar. Das kann sich aber ändern, wenn klar ist, welche Worte man nicht mehr benutzt.

Im Glossary wird oft von Forth-FCB (File-Control-Block) gesprochen. Das sind Speicherbereiche, mit denen Files beschrieben werden. Auch CP/M kennt FCBs. Die CP/M Filefunktionen erwarten alle einen DOS-FCB zur Beschreibung der Files. Die Worte, die diese Funktionen auslösen erwarten aber einen Forth-FCB, die im volksFORTH-Filesystem übliche Beschreibung von Files. Wenn die Gefahr der Verwechselung besteht, so wird ausdrücklich von Forth-FCBs und DOS-FCBs gesprochen. Allgemein ist mit der Angabe von FCB ein Forth-FCB gemeint. Seine Struktur ist aus dem Quelltext ersichtlich.
(Befehlsfolge: **DOS VIEW B/FCB**)

!fcb (fcb --) "store-f-c-b"
Interpretiert das als nächstes in der Eingabe sthende Wort als Filename und weist es dem fcb zu.

!name (addr len fcb --) "store-name"
addr gibt die Anfangsadresse eines Strings an, der die Länge len hat und einen Filenamen enthält. Dieser Name wird in den fcb eingetragen.
Enthält er keine Laufwerksangabe, so wird das aktuelle Laufwerk benutzt und in den FCB geschrieben.

(capacity (forthfcb -- n) "paren-capacity"
n ist die Filegröße des durch forthfcb beschrieben Files in Forth-Blöcken.

(close (fcb --) "paren-close"
Schließt das File, das durch fcb beschrieben wird. Schreibt alle veränderten Blöcke dieses Files auf die Diskette zurück und löscht alle Blöcke dieses Files in den Blockpuffern.

(closefile (forthfcb -- f) "paren-closefile"
Schließt das durch den Forth-FCB angegebene File. f=$FF bedeutet, daß das File nicht gefunden werden konnte.
(Siehe CP/M Operating System Manual)

(createfile (forthfcb -- f) "paren-createfile"
Erzeugt ein File, das durch den angegebenen Forth-FCB beschrieben wird. f=$FF bedeutet, daß im Inhaltsverzeichnis der Diskette kein Platz mehr ist.
(Siehe CP/M Operating System Manual)

(dir (addr len --) "paren-dir"
addr ist die Anfangsadresse eines Strings der Länge len, der ein Suchmuster enthält. **(dir** zeigt die Files an, die auf dieses Suchmuster passen. Siehe **SEARCHØ**, **SEARCHNEXT**, **FILES**, **FILES"**.

(file-read (forthfcb -- f) "paren-file-read"
Liest den im Record-Feld des angegebenen Forth-FCB's bestimmten Sektor in den Sektorpuffer ein. f<>Ø bedeutet, daß Daten fehlen.
(Siehe CP/M Operating System Manual)

(file-write (forthfcb -- f) "paren-file-write"
Schreibt den Sektorpuffer auf den im Record-Feld des angegebenen Forth-FCB's bestimten Sektor. f<>Ø bedeutet, daß die Diskette voll ist.
(Siehe CP/M Operating System Manual)

(killfile (forthfcb -- f) "paren-killfile"
Löscht das durch den Forth-FCB angegebene File. f=$FF bedeutet, daß das File nicht gefunden werden konnte.
(Siehe CP/M Operating System Manual)

(makeview (-- n) "paren-make-view"
n ist eine Zahl die aus dem momentanen Block (BLK) und dem aktuellen File (LOADFILE) berechnet wird. Sie wird in das VIEW-Feld einer neuen Definition geschrieben, und dient dazu später mit **VIEW** den Definitions-Ort zu bestimmen.

(open (fcb --) "paren-open"
Öffnet das durch den FCB angegebene File und trägt dessen Länge ein. Meldet einen Fehler, falls das File nicht gefunden werden konnte.

(openfile (forthfcb -- f) "paren-open-file"
Öffnet das durch den Forth-FCB angegebene File. f=$FF bedeutet, daß das File nicht gefunden werden konnte.
(Siehe CP/M Operating System Manual)

(read-seq (forthfcb -- f) "paren-read-sequential"
Liest den nächsten Sektor aus dem durch den Forth-FCB angegebene File in den Sektorpuffer ein. f<>Ø bedeutet, daß keine Daten mehr zur Verfügung stehen.
(Siehe CP/M Operating System Manual)

(view (viewblk -- blk') "paren-view"
blk' ist die relative Blocknummer zum Anfang des in viewblk enthaltenen Files. viewblock hat die Form: ffffffbbbbbbbbb. Wobei f Bits für die Filenummer, b Bits für den Block angeben. Das File wird von (VIEW automatisch geöffnet.

(write-seq (forthfcb -- f) "paren-write-sequential"
Schreibt den nächsten Sektor aus dem Sektorpuffer in das durch den Forth-FCB angegebene File. f<>Ø bedeutet, daß die Diskette voll ist.
(Siehe CP/M Operating System Manual)

.buffers (--) "dot-buffers"
Gibt eine Liste der Block-puffer aus, die angibt, welchen Block aus welchem File die Puffer enthalten, und ob sie als UPDATEd markiert sind.

.dosfile (fcb --) "dot-dosfile"
Gibt den Dos-Namen des durch fcb angegebenen Files aus.

.fcb (fcb --) "dot-f-c-b"
Gibt den Forth-Namen, den Dos-Namen, die Filegröße und den Status (geöffnet / geschlossen) des durch fcb angegebenen Files aus.

.file (fcb --) "dot-file"
Gibt den Forth-Namen des durch fcb angegebenen Files aus.

b/fcb (-- n) "bytes-per-f-c-b"
n gibt an, wieviele Bytes ein Forth-FCB belegt.

b/rec (-- n) "bytes-per-record"
n gibt an, wieviele Bytes in die Sektoren passen, die vom Betriebssystem benutzt werden. Bei CP/M 2.2 sind dies 128 Bytes.

bdos (arg fun# -- res) "bdos"
Veranlaßt einen Sprung ins BDOS. fun# ist der Wert, der ins C-Register geladen wird, die Nummer der aufzurufenden Funktion. arg ist der Wert, der ins DE-Register geladen werden soll, und res ist der Wert, der vom BDOS im A-Register zurückgeliefert wird. CP/M BDOS-Aufrufe sind im Operating System Manual beschrieben.

createfile (fcb --) "createfile"
Erzeugt ein File, daß durch den FCB beschrieben wird. Meldet einen Fehler, falls dies nicht möglich ist.

default-buffer (-- addr) "default-buffer"
addr ist die Adresse des Standard Sektorpuffers des BDOS.

Dos (--) "dos"
Das Vocabulary, indem die meisten Definitionen des Filesystems gemacht werden.

dos-error? (n -- f) "dos-error-question"
f ist TRUE, wenn n=$FF ist, denn das ist das Kennzeichen des BDOS für einen Fehler.

drive (forthfcb -- addr) "drive"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, unter der das Laufwerk eingetragen ist.

extension (forthfcb -- addr) "extension"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der die Extension beginnt.

fcbØ (-- addr) "f-c-b-zero"
addr ist die Adresse, des vom CCP-benutzten Standard-File-Control-Blocks, so verändert, daß er einen Forth-FCB halten kann.

file-link (-- addr) "file-link"
addr ist die Adresse einer User-Variablen, die auf den Anfang der Forth-file-liste zeigt.

file-r/w (buffer block fcb r/wf -- f) "file-r-w"
Liest oder schreibt einen Forth-Block von der / auf die Diskette.
r/wf gibt an, ob gelesen (rw/f<>FALSE) oder geschrieben (rw/f=FALSE) werden soll.
block ist die Nummer des Blocks, buffer die Adresse des Puffers.
fcb bestimmt, ob ein File benutzt wird (fcb<>Ø ist dann die Adresse eines FCB) oder ob im Direktzugriff gearbeitet werden soll (fcb=Ø).
f ist TRUE, falls ein Fehler aufgetreten ist.
Vergleiche R/W.

filename (forthfcb -- addr) "filename"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der der Filename beginnt.

filenamelen (-- n) "filenamelen"
n gibt die Länge der im Betriebssystem benutzten Filenamen an. Bei CP/M sind dies 11 Zeichen (8 Name + 3 Extension)

fileno (forthfcb -- addr) "file-number"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der die Filenummer abgelegt ist.

filesize (forthfcb -- addr) "filesize"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der die Filegröβe (in Sectoren) abgelegt ist.

in-range (block fcb --) "in-range"
Testet, ob der Forth-Block block in dem durch fcb angegebenen File liegt, und gibt eine Fehlermeldung aus, falls dies nicht der Fall ist.

opened (forthfcb -- addr) "opened"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der das open-Flag abgelegt ist.

read-seq (--) "read-sequential"
Liest den nächsten Sektor aus dem aktuellen File in den Sektorpuffer und liefert einen Fehler, falls dies nicht möglich ist.

rec/blk (-- n) "bytes-per-record"
n gibt an, wieviele logische CP/M-Sectoren (128 Bytes) in einen Forth-Block passen. Nach dem Forth-83 Standard ist ein Forth-Block 1Ø24 Bytes groβ, B/REC ist dann also 8.

record (forthfcb -- addr) "record"
Berechnet aus der Adresse eines Forth-FCBs die Adresse, an der der Recordzähler für Random-Access-Files beginnt.

reset (--) "reset"
Initialisiert das Diskettensystem des BDOS. Muβ nach einem Disketten-wechsel benutzt werden!
(Siehe CP/M Operating System Manual)

searchØ (forthfcb -- f) "search-zero"
Sucht im Inhaltsverzeichnis der Diskette nach dem ersten Vorkommen, des durch den Forth-FCB angegebenen Files. f=$FF bedeutet, daβ das File nicht gefunden werden konnte.
(Siehe CP/M Operating System Manual)

searchnext (forthfcb -- f) "serach-next"
Sucht im Inhaltsverzeichnis der Diskette nach dem nächsten Vorkommen, des durch den Forth-FCB angegebenen Files. f=$FF bedeutet, daβ das File nicht gefunden werden konnte.
(Siehe CP/M Operating System Manual)

setdma (dma --) "set-d-m-a"
dma ist die Adresse des Sektorpuffers, der beim nächsten Diskettenzugriff benutzt werden soll.
(Siehe CP/M Operating System Manual)

size (forthfcb --) "size"
Berechnet die Filegröße in dem durch den Forth-FCB beschriebenen File und trägt sie in das Feld record ein.
(Siehe CP/M Operating System Manual)

tab (--) "tab"
Geht auf die nächste Tabulatorposition (alle 2Ø Spalten).

write-seq (--) "write-sequential"
Schreibt einen Sektor aus dem Sektorpuffer als nächsten Sektor des aktuellen Files und meldet einen Fehler, falls dies nicht möglich ist.

CP/M 2.2 - spezifische Worte

Befehl	Stack	Aussprache

#bs (-- n) "number-backspace"
n ist der Ascii-Wert für Backspace.

#cr (-- n) "number-c-r"
n ist der Ascii-Wert für Carriage-Return.

#esc (-- n) "number-escape"
n ist der Ascii-Wert für Escape.

#lf (-- n) "number-linefeed"
n ist der Ascii-Wert für Linefeed.

(at (row col --) "paren-at"
Positioniert den Cursor in die Zeile row, Spalte col und setzt **OUT**. Benutzt dabei **LOCATE**.
Siehe auch **AT**.

(at? (-- row col) "paren-at-question"
row ist die aktuelle Zeilennummer, col die aktuelle Spaltennummer.
Vergleiche **AT?**

(blk/drv (drv -- blocks) "paren-blocks-per-drive"
blocks gibt an wieviele Forth-Blöcke (1kB) auf dem Laufwerk drv sind. Ist blocks=Ø, dann existiert dieses Laufwerk nicht.
Siehe **BLK/DRV**.

(cr (--) "paren-c-r"
Setzt den Cursor in die erste Spalte der nächsten Zeile. **PAUSE** wird ausgeführt.

(decode (addr pos1 key -- addr pos2) "paren-decode"
Wertet key aus. key wird in der Speicherzelle addr+pos1 abgelegt und als Echo auf dem Bildschirm ausgegeben. Die Variable **SPAN** und pos1 werden inkrementiert.
Folgende Tasten werden besonders behandelt: Control-S und Control-D beeinflussen nur pos1 und den Cursor. Ctrl-G löscht das Zeichen unter dem Cursor und dekrementiert **SPAN**.
Backspace (Control-H) und Delete ($7F) löschen das Zeichen links vom Cursor und dekrementieren pos1 und **SPAN**. Control T fügt an der Cursorposition ein Leerzeichen ein. **SPAN** wird inkrementiert. Return positioniert den Cursor auf das letzte Zeichen.
Vergleiche **INPUT:** und **(EXPECT**.

(del (--) "paren-del"
Löscht ein Zeichen links vom Cursor. Benutzt dabei **CURLEFT**.
Vergleiche auch **DEL**.

(emit (8b --) "paren-emit"
Gib 8b auf dem Bildschirm aus. Ein **PAUSE** wird ausgeführt. Alle Werte werden als Zeichen ausgegeben. Steuercodes sind nicht möglich, d.h. alle Werte < $2Ø werden als Punkt "." ausgegeben.
Vergleiche **CON!** und **EMIT**.

(expect (addr len --) "paren-expect"
Erwartet len Zeichen vom Eingabegerät, die ab addr im Speicher abgelegt werden. Ein Echo der Zeichnen wird ausgegeben. Return beendet die Eingabe vorzeitig. Ein abschließendes Leerzeichen wird immer ausgegeben. Die Länge der Zeichenkette wird in der Variablen **SPAN** übergeben.
Vergleiche **EXPECT**.

(key (-- char) "paren-key"
Wartet auf einen Tastendruck. Während der Wartezeit wird **PAUSE** ausgeführt. Die untersten 7 Bit von char enthalten den Ascii-Code der gedrückten Taste. Steuerzeichen werden nicht ausgewertet, sondern unverändert abgeliefert.
Vergleiche **KEY**.

(key? (-- flag) "paren-key-question"
flag ist TRUE, wenn eine Taste gedrückt wurde, sonst FALSE.
Vergleiche auch **KEY?**.

(page (--) "paren-page"
Löscht den Bildschirm, positioniert den Cursor in die linke obere Ecke und setzt **OUT** auf Null.
Siehe auch **LOCATE** und **PAGE**.

(r/w (adr blk file r/wf -- flag) "paren-r-w"
Ist r/wf<>FALSE, wird der Forth-Block mit der absoluten Blocknummer blk von der Diskette gelesen. Ist r/wf=FALSE so wird er geschrieben. adr gibt die Addresse des Block-Puffers an. file muß Null sein, da **(r/w** den Zugriff auf Files nicht unterstützt. flag ist TRUE wenn ein Diskettenfehler vorlag.

(type (addr len --) "paren-type"
Gibt den String, der im Speicher bei addr beginnt und die Länge len hat, auf dem Blidschirm aus. Genau ein **PAUSE** wird nach der Ausgabe ausgführt.
Vergleiche **TYPE**, **OUTPUT:** und **(EMIT**.

/drive (blk -- blk' drv) "per-drive"
blk gibt die absolute Nummer eines Forth-Blocks an. **/DRIVE** berechnet, auf welchem Laufwerk (drv) dieser Block zu finden ist, und welche relative Blocknummer (blk') er zum Anfang dieses Laufwerks hat.
Siehe **DRV?**, **>DRIVE**.

>drive (blk drv -- block') "to-drive"
blk gibt die relative Blocknummer eine Forth-Blocks bezüglich des Anfangs von Laufwerk drv an. **>DRIVE** berechnet daraus, unter welcher Blocknummer dieser Block beim momentanen Stand von **OFFSET** erreicht werden kann (block'). In gewisser Weise Umkehrung von **/DRIVE**.

?drive-error (f --) "question-drive-error"
Ist f=FALSE, so wird "beyond capacity" als Fehlermeldung ausgegeben.

?drive (n -- n) "question-drive"
Überprüft, ob das Laufwerk n existiert, und gibt "beyond capacity" als Fehlermeldung aus, wenn dies nicht der Fall ist.

b/blk (-- b/blk) "bytes-per-block"
Eine Konstante die angibt, wieviele Bytes in einen Forth-Block passen. Nach dem Forth-83 Standard ist **B/BLK** = &1Ø24.

bios (-- addr) "bios"
Adresse eines 8Ø8Ø-Unterprogramms, das einen Sprung ins BIOS ausführt. Das Low-Byte der Einsprungadresse steht dabei in HL. Wird von **con!**, **(key?**, **getkey** und read/write benutzt.

blk/drv (-- #blk) "blocks-per-drive"
#blk gibt die Kapazität des aktuellen Laufwerks (bestimmt durch **OFFSET**) in Forth-Blöcken (1kB) an. Siehe **(BLK/DRV**.

con! (8b --) "con-store"
Gibt 8b auf die CONsole (Bildschirm) aus. Ascii-Werte < $2Ø werden als Steuercodes interpretiert.

curleft (--) "cur-left"
Bewegt den Cursor ein Zeichen nach links. Eine der vordefinierten Terminalfunktionen.

curoff (--) "cur-off"
Schaltet den Cursor aus. Eine der vordefinierten Terminalfunktionen.

curon (--) "cur-on"
Schaltet den Cursor an. Eine der vordefinierten Terminalfunktionen.

currite (--) "cur-right"
Bewegt den Cursor ein Zeichen nach rechts. Eine der vordefinierten Terminalfunktionen.

dark (--) "dark"
Löscht den Bildschirm. Eine der vordefinierten Terminalfunktionen.

display (--) "display"
Ein mit **OUTPUT:** definiertes Wort, das den Bildschirm als Ausgabegerät anwählt, wenn es ausgeführt wird. Die Worte **EMIT**, **CR**, **TYPE**, **DEL**, **PAGE**, **AT**, und **AT?** beziehen sich dann auf das aktuelle Terminal.
Siehe **TERMINAL:**.

dma! (addr --) "d-m-a-store"
addr ist die Adresse des Diskettenpuffers, der beim nächsten Diskettenzugriff verwendet werden soll.

drive (n --) "drive"
Wählt n als aktuelles Laufwerk an. Ändert **OFFSET** entsprechend.
Siehe **BLK/DRV**.

drv! (drv f -- dph) "drive-store"
drv ist die Nummer des Diskettenlaufwerks, das als nächstes verwendet werden soll. f=Ø gibt an, ob es sich um den erste Zugriff nach einem CP/M Warmstart handelt. dph ist die Adresse des CP/M Disk-Parameter-Headers. (Siehe CP/M Operating System Manual). Ist dph=Ø, so ist das angesprochene Laufwerk in diesem Computersystem nicht unterstützt.

drv? (blk -- drv) "drive-question"
blk gibt die absolute Nummer eines FORTH-Blocks an, **DRV?** berechnet daraus das Laufwerk (drv) auf dem er zu finden ist.
Siehe **/DRIVE**, **>DRIVE**.

drvØ (--) "drive-zero"
Wählt Laufwerk Ø (A) als aktuelles Laufwerk für **R/W** an.
Siehe **DRIVE** und **>DRIVE**.

drv1 (--) "drive-one"
Wählt Laufwerk 1 (B) als aktuelles Laufwerk für **R/W** an.
Siehe **DRIVE** und **>DRIVE**.

drvinit (--) "drive-init"
Initialisiert das volksFORTH-Disk-System. Die im Computer-System vorhandenen Laufwerke werden der Reihe nach selektiert und deren Kapazität berechnet. Dann wird das CP/M default-Laufwerk selektiert.

dumb (--) "dumb"
Ein mit **TERMINAL:** definiertes Wort, das ein ignorantes Terminal anwählt, wenn es ausgefährt wird. **CURON**, **CUROFF**, **CURLEFT**, **CURRITE**, **RVSON**, **RVSOFF**, **DARK** und **LOCATE** haben dann keine Wirkung. Mit ihnen auch die sie benutzenden Worte **(PAGE**, **(AT**, **(DEL**. Wenn **DISPLAY** eingeschaltet ist, sind also auch **PAGE**, **AT** und **DEL** wirkungslos. DUMB ist als aktuelles Terminal angewählt, bis die Installierung eines leistungsfähigeren Terminals abgeschlossen ist.

getkey (-- char) "getkey"
Die unteren 7 Bit von char enthalten den Ascii-Code des letzten Tastendrucks. Ist noch keine Taste gedrückt, dann wartet **GETKEY**.
Siehe auch **KEY?** und **KEY**.

home (--) "home"
Der Kopf des momentan selektierte Diskettenlaufwerks wird auf Spur Null gefahren. Spur Null wird als nächste Spur angewählt, die verwendet werden soll.
Siehe **TRK!**, **DRV!**.

index (from to --) "index"
Liest die Blocks from bis einschließlich to und gibt deren erste Zeilen aus. Index kann mit einer beliebigen Taste angehalten werden und mit RETURN-Taste abgebrochen werden. (Siehe **STOP?**) Die ersten Zeilen von Screens enthalten typischer Weise Kommentare, die den Inhalt charakterisieren.

keyboard (--) "keyboard"
Ein mit **INPUT:** definiertes Wort, das die Tastatur als Eingabegerät anwählt. Die Worte **KEY**, **KEY?**, **DECODE** und **EXPECT** beziehen sich nun auf die Tastatur.
Siehe **(KEY**, **(KEY?** **(DECODE**, **(EXPECT**.

locate (row col --) "locate"
Bewegt den Cursor absolut auf Spalte col, Zeile row. Eine der vordefinierten Terminalfunktionen.

out (-- addr) "out"
Adresse einer Variablen, die die Anzahl der ausgegebenen Zeichen enthält.

read/write (r/wf sponti -- f) "read-write"
Bewirkt das physikalische Lesen (r/wf = FALSE) und Schreiben (r/wf=TRUE) eines Sektors (=128 Bytes) von der/auf die Diskette. Das Laufwerk, die Spur , der Sektor sowie der Sektor-Puffer sind vorher mit **DRV!**, **TRK!**, **SEC!** und **DMA!** gewählt worden. sponti gibt an, ob beim Schreiben unmittelbar auf die Diskette geschrieben werden soll (sponti=TRUE) oder, ob der geschriebene Sektor im BIOS zwischengepuffert werden darf (sponti=FALSE).

rvsoff (--) "reverse-off"
Schaltet die Inversdarstellung aus. Eine der vordefinierten Terminalfunktionen.

rvson (--) "reverse-on"
Schaltet die Inversedarstellung ein. Eine der vordefinierten Terminalfunktionen.

sec! (sec --) "sec-store"
sec ist der beim nächsten Diskettenzugriff zu verwendende Sektor.

Term: (offset -- offset') "term-colon"
Ein definierendes Wort für Terminalfunktionen. Wird benutzt um die einzelnen Komponenten eines Terminal-Vektors zu definieren. Vordefinierte Terminalfunktionen sind **CURON**, **CUROFF**, **CURLEFT**, **CURRITE**, **RVSON**, **RVSOFF**, **DARK** und **LOCATE**.
Siehe auch **TERMINAL:**

Terminal: (--) "terminal-colon"
Ein definierendes Wort für Terminals. Benutzt in der Form:

```
Terminal: <name>
        newCURON newCUROFF newCURLEFT newCURRITE
        newRVSON newRVSOFF newDARK    newLOACTE ;
```

TERMINAL: erzeugt einen Kopf für <name> im Dictionary und kompiliert einen Vektor von Zeigern auf Worte die für die Ausführung von Terminalfunktionen zuständig sind. Wird <name> ausgeführt, so werden die Terminalfunktionen von <name> zu den aktuellen Terminalfunktionen gemacht, das Terminal <name> ist damit aktiv. Terminalfunktionen werden von **AT**, **PAGE**, **DEL** ausgeführt, wenn die Ausgabe auf **DISPLAY** geschaltet ist.
Siehe **OUTPUT:**, **DISPLAY**, **DUMB**.

trk! (trk --) "track-store"
trk ist die beim nächsten Diskettenzugriff zu verwendende Spur.