FORTH - INFO

Mitteilungsblatt der Fachgruppe FORTH im Kulturbund der DDR

Ausgabe 2/89 vom 1.9.1989

Lieber Freund!

Korrekt wäre natürlich "Bundesfreund" (Kulturbund!) gewesen, aber das wollen wir mal nicht übertreiben. Jedenfalls hat unser Info-Blatt über den Urlaub ein neues Layout bekommen und wir wollen versuchen, den Umfang und die Gestaltung beizubehalten, solange es die Vervielfältigungsmöglichkeiten zulassen. Ich melde mich jedenfalls aus dem Urlaub zurück und hoffe, daß niemand die Sommerpause über eingerostet ist. Denn es geht nun munter wieder los. Am 28.9.89 ist Klubtreff im Leipziger Jugendklubhaus "Walter Barth" und am 7.10.89 findet unsere Herbsttagung in Dresden statt. Dazu liegt diesem Info ein gesondertes Blatt bei. Die Vorbereitungszeit ist diesmal etwas kurz, so daß ich im Moment auch noch nicht über alle Einzelheiten informiert bin. Trotzdem wollten wir den Oktobertermin halten, damit wir nicht wieder aus dem Zeitrythmus kommen und aus der Herbst- eine Wintertagung wird. Auf jeden Fall empfehle ich allen Teilnehmern, Datenträger zu ihrem Computer für den Softwaretausch mitzubringen. Wahrscheinlich werden folgende Computer vorhanden sein: die KC's, C64, Z1013, PC1715, MS-DOS-Kompatibler. Soviel dazu. Die nächste Frühjahrstagung soll dann im April 90 stattfinden. Ort und Organisator werden noch gesucht! Für diesmal werden Tagungsankündigungen auch an den Z1013-Computerclub und an die Gäste der letzten Tagung im Juni 89 verschickt.

Ansonsten gibt es infolge der Sommerpause wenig Neuigkeiten. Wichtig und erfreulich ist, daß nun das langerwartete Buch "FORTH" von G.-U. Vack endlich auch im "VD" (Vorankündigungsdienst) stand. Preis ca. 35.- M. Man kann also Bestellungen aufgeben!

In diesem Info setze ich das Programm zur Dateiverwaltung fort. Es wird mit komfortableren Ein- und Ausgabemöglichkeiten ausgestattet. Ich nehme an, daß noch ca. 3 Fortsetzungen folgen werden. Außerdem haben wir diesmal auch Beiträge aus der Feder von Klubmitgliedern. Davon wünsche ich mir noch viel mehr!

In diesem Sinne! Herzliche Grüße,

Michael Baltys

PS: Dank der Aktivitäten von Thomas Noßke sind jetzt zwei Bände von "Dr. Dobb's Toolbook of Forth" verfügbar geworden. Sie stehen in der Hochschulbibliothek der Technischen Hochschule "Carl Schorlemmer" Leuna-Merseburg. Ein Überblick über die Inhalte ist auf der Rückseite dieses Blattes zu finden!

The Toolbook of Forth

Volume I

Kim Harris: The Forth Philosophy
John S. James: Teaching Forth as a First Language
George W. Shaw II: Forth-83 and Vocabularies
C. H. Ting: GO in Forth
Glen B. Haydon: Elements of a Forth Data-Base Design
Mark I. Manning: The Forth Sort
Robert Taylor: SEND and RECV
Ray Duncan/Rick Wilton: Interfaces for a Mouse
Joe Barnhart: Relocating Loader in Forth
Ray Duncan: Forth Decompiler
Henry Laxen/Michael A. Perry: Screen-Oriented Editor Re-Visited
Wendall C. Gates: Evolution of a Video Editor
Albert S. Woodhull: H-19 Screen-Editor
John S. James: The Conference Tree
Wendall C. Gates: Series Expansion in Forth
Alfred J. Monroe: Forth Floating-Point Package
Robert L. Smith: Signed Integer Division
Ralph Deane: A Proposal for Strings in Forth
Louis L. Odette: Non-Deterministic Control Words
Ed Wischmeyer: Some Forth Coding Standards
Harvey Glass: Towards a More Writable Forth Syntax
Joe Barnhart: Forth and the Motorola 68000
Michael A. Perry: A 68000 Forth Assembler
William F. Ragsdale: A Forth Assembler for the 6502
Louis L. Odette: Z8000 Forth
Ray Duncan: Converting fig-FORTH Programs for FORTH-83

Volume II

Kim Harris, Michael Ham: An Approach to Reading Programs
Michael Ham: Forth Philosophy, Standards, and Practical Advice
Michael Ham: Factoring in Forth
Michael Ham: Think Like a User, Write Like a Fox
Wil Baden: Charting Forth
Wil Baden: Escaping Forth
Wil Baden: Hacking Forth
Wil Baden: Leaping Forth
C. H. Ting: F83 Word Usage
Klaus Schleisiek: Error Trapping and Local Variables
Kim R. Harris: Using the Structure-Tool Program to Automatically Construct Charts of Forth Programs
Kim R. Harris: Analyzing Large Forth Programs Using the Structure-Tool Program
Raymond Buvel: A Forth Native-Code Cross Compiler
Guy T. Grotke and Guy M. Kelly: A Simple Metacompiler
C. H. Ting: Zapping the F83 Dictionary
Martin Tracy: A Forth Standard Prelude
Joe Barnhart: Forth and the Fast Fourier Transform
Marin Tracy: Zen Floating Point
Nathaniel Grossman: A Forth Slide Rule
Craig A. Lindley: Forth Windows for the IBM PC
Wil Baden: Quicksort and Swords
Craig A. Lindley: A Forth Spreadsheet
Phil Koopman, Jr.: Bresenham Line-Drawing Algorithm
Phil Koopman, Jr.: Fractal Landscapes
David L. Jaffe: Forth in Rehabilitation Applications
Dana Redington: A Forth-Oriented, Real-Time Expert System
Martin Tracy: A Forth Lisp
Lance Collins: LOGO in Forth
Dana Redington: Knowledge Representation in Forth
Stephen D. Lindner: How and Why - Multiple Inheritance Object Systems
Jack Park: An Approach to Natural Language Parsing

Datenverwaltung, Teil II

Bevor ich die Beschreibung der Datenverwaltungs-Worte fortsetze, muß ich anmerken, daß das Echo auf den ersten Teil meinen Erwartungen nicht entsprach. Ich hatte vorsätzlich einen - ich gebe zu: nicht leicht zu findenden - Fehler eingebaut, aber es hat sich niemand bei mir beschwert. Nun ja, heute kommt die unverlangte Auflösung des Rätsels, und die paßt auch genau zum heutigen Thema: Vokabulare. Sicher wird sich der eine oder andere zumindest gewundert haben, warum ich so darauf erpicht war, die Worte zur Datenbeschreibung - und nur die! - in ein extra dafür eröffnetes Vokabular zu verfrachten. Aber man braucht nur einmal "DATEI WORDS" einzugeben, dann liegt die Lösung auf der Hand. Unabhängig davon, was in diesem Vokabular vereinbart ist - es werden alle Worte darin getreulich aufgelistet. Und das nun wollen wir benutzen, um einen kompletten Datensatz ein- oder auszugeben. Dabei spielt es also keine Rolle, wie die einzelnen Datenbeschreibungen heißen, es könnten genausogut Materialbestandsdateien oder Literaturverzeichnisse sein, die wir in DATEI verschlüsselt haben. Wie macht man das nun? Dazu braucht man sich eigentlich mit "SEE" nur mal das Wort "WORDS", das alle im CONTEXT-Vokabular vorhandenen Worte ausgibt, anzusehen - genauso machen wir das auch. Dazu wird das Wort "TRIM" benötigt, das im Vokabular "HIDDEN" versteckt ist (hidden - versteckt!). Deshalb beginnt Screen 7 (b.w.) damit, das Vokabular HIDDEN mit in die Suchreihenfolge einzubinden. Nach der Nennung von HIDDEN ist HIDDEN das transiente Vocabular innerhalb der Gruppe der CONTEXT-Vokabulare. Seht in Screen 1 nach! Dort steht "ONLY FORTH ALSO DEFINITIONS". Damit war FORTH sowohl transientes als auch residentes Vokabular geworden. Das transiente haben wir nun durch HIDDEN ersetzt. Da wir weiterhin in FORTH hineinkompilieren, müssen wir noch dafür sorgen, daß HIDDEN resident wird, denn bei Beginn einer Doppelpunktdefinition wird das CURRENT-Vokabular gleichzeitig als transientes Vokavbular eingesetzt und HIDDEN wird an dieser Stelle durch FORTH überschrieben, deshalb also "ALSO" nach "HIDDEN"! Dieses Gleichsetzen von CURRENT- und transientem (CONTEXT-) Vokabular hat pragmatische Gründe. Das gab es schon im fig-Forth. Man muß es hinnehmen und sich danach richten. Das Wort **-STEP** "steppt" durch das transiente Vokabular. Mit jedem Aufruf wird die CFA des nächsten Eintrages ermittelt. Es ist praktisch aus "WORDS" abgeschrieben. Man achte darauf, daß CONTEXT nicht mehr nur <u>eine</u> Adresse darstellt, sondern mehrere - soviel, wie im Wörterbuch getrennte Ketten (Fäden) vorhanden sind. In unserem Falle findet man ihre Anzahl in der Konstanten #THREADS. Da die nun zu beschreibenden Worte **INFO** und **NEU** immer "DATEI" in "CONTEXT" voraussetzen, muß man also dafür sorgen, daß das transiente Vokabular auch wirklich "DATEI" heißt. Natürlich könnte man die Benennung dieses Vokabulares unter dieser Prämisse ebenfalls freistellen, aber ich glaube, das würde die Erklärung noch schwerer faßbar machen als sie es ohnehin schon ist.
Das Wort **INFO** geht also durch das Vokabular und beginnt dabei mit Hilfe von "TOP" am letzten, neuesten Eintrag. Deshalb sollte er (siehe Teil 1) - der wichtigste, namensgebende sein. Das Wort TOP wird weiter unten erklärt! Es liefert die CFA des zuletzt eingetragenen Wortes. Nach Ausgabe der aktuellen Aktennumer wird nun jeweils in einer Zeile die Bezeichnung des Eintrages (Items) und der dazugehörige Dateninhalt ausgegeben. Zeile 6 zeigt übrigens, wie man in einfacher Weise Tabulatoren setzen kann. Man könnte sich dafür auch ein Wort **TAB** definieren:

```
SCR # 7
 0 \ Datenverwaltung II. Teil
 1 HIDDEN ALSO  \ Items aufsuchen
 2 : -STEP ( cfa -- cfa f )
 3    >LINK     \ lfa fuer TRIM
 4    CONTEXT @ HERE #THREADS 2*
 5    CMOVE     \ Fadenzeigerfeld
 6    HERE TRIM \ Faeden kuerzen
 7    HERE #THREADS LARGEST NIP
 8    DUP LINK> SWAP 0= ;
 9
10
11
12
13
14
15
```

```
SCR # 8
 0   \ Akteninhalt anzeigen
 1 : INFO ( -- ) \ Bildschirmabh.!
 2    12 EMIT \ Bildschirm loeschen
 3    TOP     \ erstes Item
 4    ." Akte Nr. " AKTE# ? CR
 5    BEGIN DUP >NAME .ID ." :"
 6     BEGIN SPACE #OUT @ 12 >
 7     UNTIL  DUP .FELD CR -STEP
 8    UNTIL DROP ;
 9
10
11
12
13
14
15
```

```
SCR # 9
 0   \ Akteninhalt neu eingeben
 1 : NEU ( -- ) \ Anforderung!
 2    CR
 3    TOP        \ erstes Item
 4    ." Akte Nr. " AKTE# ? CR
 5    BEGIN DUP >NAME .ID ." : "
 6     QUERY  SPAN @
 7     IF DUP ABLEGEN \ Uebernahme
 8     ELSE DUP .FELD \ Ausgabe
 9     THEN  CR
10     -STEP
11    UNTIL DROP ;
12 \ nach Abarbeitung von "NEU"
13 \ wird wegen "QUERY" stets eine
14 \ neue Kommandozeile angefordert
15 \ ( egal, was nach "NEU" steht!)
```

```
SCR # 10
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
```

arcus SOFTWARE Michael Balig

gedruckt auf PRÄSIDENT 6313 - Z1013

```
Scr # 3
  0   \ Position & Bewegung in Akten
  1 : WAS ( -- adr ) PAD 80 + ;
  2 : AKTE ( -- adr ) AKTE# @
  3    A/BLK /MOD DATEN + BLOCK
  4    SWAP C/AKTE * + ;
  5 : FELD ( cfa -- adr count )
  6    >BODY 2@ AKTE + SWAP ;
  7 : .FELD ( cfa -- ) FELD
  8    -TRAILING TYPE SPACE ;
  9 : TOP ( -- cfa ) \ 1. Eintrag!
 10    CONTEXT @ #THREADS LARGEST
 11    NIP LINK> ;
 12 : .FIRST ( -- ) \ Item 1 ausgeb.
 13    TOP .FELD ;
 14 \ vor Aufruf der Worte muss
 15 \ DATEI aufgerufen werden!
```

: TAB (n --) BEGIN SPACE #OUT @ OVER > UNTIL DROP ;
Die Tabulatorposition ist hierbei (n) sogar wählbar.
Das Wort in Screen 9 stellt die Eingabe dar und ist der Ausgabe recht ähnlich. Nur daß hier nach der Bezeichnung des Datenfeldes sein Inhalt angefordert wird. Um Kompatibilität zur Kommandozeilenarbeit der schon beschriebenen Worte (z.B. SUCHE, AENDERE) zu erhalten, erfolgt die Eingabe über QUERY. Damit kann man dann ABLEGEN als Eingabewort benutzen. Das hat aber den Nachteil, daß die Interpretation der Zeile, in der NEU aufgerufen wird, mit diesem Wort beendet ist. Normalerweise ruft man also folgendermaßen auf: DATEI FREI? NEU
FREI? sucht eine freie Akte - mit NEU gibt man Werte ein. Man kann aber auch eine bereits eingegebene Akte mit Hilfe von NEU korrigieren, denn wenn man auf eine Eingabeanforderung nur die Returntaste (bzw. Enter) eingibt (Test von SPAN), wird der vorhandene Inhalt belassen und protokolliert. Damit kann man bequem korrigieren, wenn in einer Akte mehrere Angaben zu ändern sind.
Nun noch einmal zu Teil I! Wer sich das Wort .FIRST einmal genauer ansieht und mit dem vergleicht, was ich oben über Vokabulare geschrieben habe, wird verstehen, daß es zwar im gegebenen Falle richtig funktioniert - hätten wir aber als ersten Eintrag z.B. statt NAME "MITGLIED" gewählt, so würde es nicht klappen. Wir wären nämlich in der falschen Link-Kette. Man muß also hier genauso wie in INFO vorgehen, nur daß man wirklich nur das letzte Wort des Vokabulares braucht. Deshalb wurde Screen 3 noch einmal (wie bereits angekündigt!) geändert und das Wort TOP eingeführt, das die CFA des letzten (oder 1. - das ist Ansichtssache) Eintrages im transienten Vokabular zur Verfügung stellt. Im Falle der Eingabe von "FORTH TOP" würde beim jetzigen Stand der Dinge also die CFA von "NEU" ausgegeben werden. Warum?

m.b.

PS: Diese ganze Vokabularproblematik kann man recht gut im Buch von R. Zech "Forth 83" oder in der Übersetzung des Standards Forth 83 von G.-U. Vack ("Der Standard Forth 83", KdT Suhl) nachlesen.

Liebe Freunde!

Einige Worte zur Zahlungsmoral in der Fachgruppe scheinen angebracht: Mit Stand vom 1.9.89 haben weniger als die Hälfte der Fachgruppenmitglieder ihren Beitrag überwiesen. Das geht natürlich nicht so weiter. Die Fachgruppenleitung gibt Euch noch bis 15.10. Gelegenheit, die finanziellen Dinge in Ordnung zu bringen. Danach werden Nicht-Zahler von der Fachgruppenliste gestrichen. Bitte überweist Euren Beitrag möglichst am 10.1., 10.4., 10.7. und 10.10. jeden Jahres für das jeweilige Quartal. Das macht es leichter, die eingegangenen Beträge zu überprüfen. Ich habe ohnehin damit sehr viel Mühe.

Herzlich Computergrüße

Otfried Müller

Bodo Bachmann:

G L E I T K O M M A P A K E T E

1. 5-Byte Gleitkommarechnung

EXTENDED FIG-FORTH arbeitet mit dem in sich geschlossenen Gleitkommakalkulator des ZX-Spectrum. Dieser Kalkulator konnte auf den gleichen Adreßbereich übernommen werden.
Die Anbindung an Forth wird in dem Artikel "Gleitkomma in Forth" (MIKROPROZESSORTECHNIK 6/87) beschrieben. Hier werden noch einige zusätzliche Informationen gegeben:

```
Adreßbereich:       2CØØ - 39ØØ
    speziell:       2CØØ - 2C87  Systemvariablen
                    2C88 - 386D  Kalkulatorprogramme
                    386E - 39ØØ  Anbindung an Forth

verwendete Restarts:        RST ØØ   JMP GETNXT
                            RST Ø8   JMP ERROR
                            RST 1Ø   JMP PRTOUT2
                            RST 18   JMP GETAKT
                            RST 28   JMP KAL (335B)
```

Adr	UP
386E	GETNXT
3877	GETAKT
387C	ehem. 2AB6
3892	ERROR
38A9	Fehlercode
38AB	PRTOUT1
38AE	PRTOUT2
38B2	ehem. 1A1B
38DØ	ehem. 192A

Interne Änderungen: MEMBOT 5C92 - 5CAF --> 2C6A - 2C87

Adressen, auf denen Anpassung notwendig war:

2CA2	2CC3	2CCB	2CDA	2CE8	2CF4	2CFE	2D2B	2D4A	2D56
2E2E	2E32	2E44	2E4B	2E62	2E85	2E95	2E98	2EAB	2EAE
2EBA	2EC1	2EC6	2EDF	2EFE	2FØ2	2FØ7	2FØD	2F1Ø	2F2A
2F2D	2F36	2F3A	335F	3365	339D	33A3	33B4	33BB	341Ø
342F	35BF	367C	2D34	2DC8	33AE	2CCF			
33FB	(DE=ØØBØH)								
2E24 - 2E55	ABS(X<1)								

Jede Kalkulatorfunktion ist durch einen Aufrufcode (siehe kommentiertes SPECTRUM-Listing) gekennzeichnet.
Noch nicht verwendet werden können:

```
Ø9 - ØE
11 - 1A
1C - 1E
2B - 2F
```

Dies sind vor allem Stringoperationen.

****

INIT ( )
die Restarts werden gesetzt und der G-Stack initialisiert.

PKONT (adr -- f)
testet den Ziffernstring ab adr auf Beinhalten eines Punktes.
Flag =1, für Punkt.

KONV (adr --)
die Zahl (String) ab adr wird konvertiert und auf den G-Stack gelegt.

GPAK (n --)
Form: n GPAK cccc
die Funktion des Kalkulators mit dem Makrocode n wird als Forth-Wort cccc definiert.

GNUMBER (adr -- ?)
berücksichtigt auch die Eingabe von Gleitkommazahlen, sonst wie NUMBER.

G->S (-- n)
5-Byte-2-Byte Wandlung; die Zahl kommt vom G-Stack auf den P-Stack.

S->G (n --)
Umkehrung von G->S.

Wort	analog
GLIT	LIT
G,	,
GLITERAL	LITERAL
G.	.
G@	@
G!	!
GCONSTANT	CONSTANT
GVARIABLE	VARIABLE
G+	+
G-	-
G*	*
G/	/
GSWAP	SWAP
GDUP	DUP
GDROP	DROP

Diese Worte (G...) beziehen sich auf Gleitkommazahlen und gegebenenfalls auf den G-Stack.

Folgende Funktionen verlangen einen Wert auf dem G-Stack und legen den Funktionswert dafür dort ab:
SIN COS TAN LN EXP GVZ G<Ø G>Ø GABS GANZ

Bei den Funktionen
GOR GAND X^Y
werden 2 Werte auf dem Stack verlangt.

PI/2 hinterläßt den Wert 'PI/2' auf dem G-Stack.

****

2. 6-Byte Gleitkomma-FORTH-Modul

Das Modul befindet sich in dem File FLOAT.BLK und ist inzwischen komplett lauffähig gemacht worden. Entsprechende Erläuterungen befinden sich in den Schatten-Screens.

Anmerkung:
Das oben benannte File FLOAT.BLK liegt auf Diskette für Z-80-Computer vor, läßt sich aber leicht durch Umprogrammierung von 2 Primitiven auf andere Prozessoren umstellen. Screenformat ist 64*16, ca. 28 Programmscreens. Ernsthafte Interessenten können mir eine bereits formatierte Diskette (nicht CP/A!) zuschicken. Bei Andreas Driesel ist ein entsprechendes Kassettenfile für den Z1013 zu haben. Andere U880-Computer werden schnell folgen.

Michael Balig

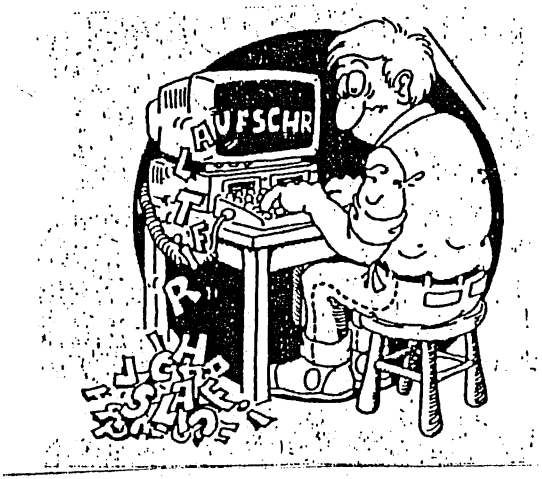


Bild: Herstellung der FORTH - INFO's

Betr.: Unser Softwarevertrieb

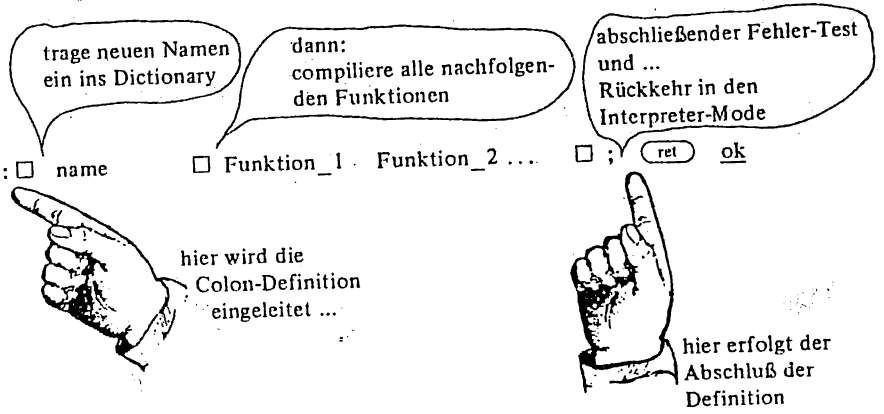
Ich möchte an dieser Stelle einmal die Gelegenheit nutzen und im Namen der Leitung der Fachgruppe allen Rechnerbetreuern für Ihre Hilfe recht herzlich danken und hoffe, daß wir in dieser Weise weiterarbeiten können. Allerdings wäre es wünschenswert, wenn sich noch ein paar Freunde für diese Aufgabe fänden, damit wir die Arbeit auf mehr Schultern verteilen können.
Bisher haben folgende Freunde am Vertrieb mitgearbeitet:
Andreas Driesel, Springerstr. 31, Leipzig 7022 (Z1013)
Steffen Schwips, Nordstr. 5, Zwenkau 7114 (Schneider CPC)
Michael Jungmann, Försterweg 1, Erkner 1250 (ZX Spectrum)
Wolf-Rüdiger Jürgens, Str. d. Einheit 9, Staßfurt 3250 (AC1)
Helmut Rauschenbach, Zeitzer Str. 55 / PF 01-012,
Meuselwitz 7404 (MZ-800)
Matthias Langer, Dr.-Otto-Nuschke-Str. 8, Lugau 9159 (KC85/1)
Den KC85/3 mußte ich übernehmen, da es dort mit der Organisation nicht klappte, aber ich würde mich freuen, wenn mir jemand dieses Amt abnähme, da die Fachgruppe auch ohnedem meine Freizeit schon arg strapaziert. Also nochmals vielen Dank!

Michael Jung

Sollte es jemand schon vergessen haben: ...

Colon-Definitionen (secondary-Worte)

Die Eröffnung eines high-level-Forth Programmes erfolgt mit dem Befehl: (‚colon'). Es wird ein Name für die neue Funktion erwartet und eingetragen. Sodann befindet sich der Rechner im Compile-Mode. Nachfolgender Text wird als Programm (source text) behandelt, auf die Compile-Adressen bekannter Adressen reduziert, und diese Adressen (CFAs) werden dann compiliert. Die Anwendung erfolgt in der allgemeinen Form:



Vektoren

Im Forth-83 unserer IG wurde ein neues, leistungsfaehiges Hilfsmittel breit verwendet: Vektoren. Worum handelt es sich dabei, und wie kann man diese in der eigenen Arbeit verwenden? Dieser kleine Artikel soll dies erklaeren.

Die Vektorisierung, sprich Umleitung von Ausfuehrungsroutinen ist eigentlich nichts neues. Vielen duerfte das alte Verfahren von FIG bekannt sein, die Codefeldadresse eines Wortes in einer Variable zu speichern und deren Inhalt spaeter zur Ausfuehrung zu bringen. Dies ermoeglicht es, nach erfolgter Compilation das Verhalten des Systems zu aendern, z.B. die Ausgabe vom Bildschirm auf Drucker umzuleiten. Das sah etwa so aus:

```
0 VARIABLE ZEIGER     ' EMIT CFA  ZEIGER !

: AUSGABE   ZEIGER @  EXECUTE  ;
```

Wenn man dann den Inhalt von ZEIGER mit der Ausgaberoutine fuer den Drucker fuellte, gingen alle Ausgaben des angedeuteten Codes auf den Drucker hinaus.

Diesen Mechanismus hat man nun zur Schaffung einer neuen Wortklasse, eben der Vektoren, genutzt. Das Definitionswort, mit dem alle Vektoren angelegt werden und das deren Laufzeitverhalten bestimmt ist das Wort DEFER. Es sieht in Hochsprache wie folgt aus:

```
: DEFER   CREATE  ['] CRASH ,   DOES> @ EXECUTE ;
```

Es wird also ein Wort,der Vektor, angelegt, das in seinem Parameterfeld die CFA des zugewiesenen Codes enthaelt. Dieser wird bei Aufruf des Vektors ausgefuehrt. Unser Beispiel saehe dann so aus:

```
   DEFER AUSGABE
```

Dies wars erstmal schon. Vorbelegt ist dieser Vektor mit dem Code CRASH, welcher nach Ausgabe der Meldung "uninitialized Execution Vector" nach QUIT verzweigt und die Abarbeitung beendet. Dies verhindert einen Systemabsturz, solange dem Vektor noch kein gueltiger Code zugewiesen wurde.
Diese Zuweisung geschieht mit dem Wort IS (sowohl im Interpret- als auch im Compilemode) wie im folgenden gezeigt:

```
  ' EMIT  IS AUSGABE   oder

  : name .....     ['] EMIT  IS AUSGABE  ... ;
```

Der Aufruf des Wortes AUSGABE wird schliesslich die entsprechende Routine zur Ausfuehrung bringen.

Es bleibt noch zu sagen, dass es einen zweiten Typ Vektoren gibt, die mit der Sequenz USER DEFER 'name' angelegt werden. Im Unterschied zur ersten Variante stellen sie analog den USER-Variablen fuer jedes Task eine eigene Ausfuehrungsroutine bereit. Auf Sinn und Zweck wird im folgenden eingegangen.

Am Beispiel der im Grundsystem vorhandenen Vektoren soll gezeigt werden, wo diese neue Struktur sinnvoll einsetzbar ist. Nebenbei sollen damit Anregungen zur Nutzung der im Grundsystem eingebauten Aenderungsmoeglichkeiten gegeben werden.

Folgende Worte sind als Vektor bzw. USER-Vektor definiert:

EMIT CR KEY KEY? EXPECT READ-BLOCK WRITE-BLOCK NUMBER?
STATUS HANDLE WHERE ?ERROR BOOT PUT GET

Einige weitere werden in Editor und Assembler benutzt.

Nun etwas detailierter.
EMIT, CR, KEY und KEY? zeigen im Normalfall auf die Routinen (EMIT), CRLF, (KEY) und (KEY?). Dabei sind EMIT und CR User-Vektoren, d.h. jedes Task kann ein anderes Ausgabegeraet benutzen.
Durch diese Vektorisierung koennen Ein- und Ausgaben bequem umgeleitet werden. Anwendungsbeispiele sind etwa die Einbindung von Spezialtastaturen oder Joysticks fuer Spiele, der Anschluss einer seriellen Schnittstelle, um Fernbedienung zu ermoeglichen oder die Umkodierung der Steuerzeichen. Ebenso laesst sich dies zur Ausfilterung bestimmter Zeichen, Umwandlung von Klein- in Grossbuchstaben u.ae. verwenden.

EXPECT als Wort zur Eingabe einer Textzeile fuehrt im Urzustand die Routine (EXPECT) aus. Diese verwertet nur die Steuerzeichen Backspace 08H, Delete 7FH und CR 0DH. Wer bessere Editier-moeglichkeiten der Eingabezeile wuenscht muss nur ein newEXPECT schreiben und die Eingabe mit ' newEXPECT IS EXPECT auf sein neues Wort umleiten. Dazu noch eine Bemerkung: Natuerlich muss das neue Wort das gleiche Stackverhalten wie sein Vorgaenger haben, siehe dazu unser Glossarium.

READ-BLOCK und WRITE-BLOCK wurden als Vektoren definiert um dem Benutzer im Nachhinein die Moeglichkeit zu geben, eine andere Organisation der Datenspeicherung zu waehlen. Als Beispiele seien der Anschluss eines Diskettenlaufwerkes, die komprimierte Speicherung der Bloecke im RAM und die Auslagerung der simulierten RAM-Disk in einen erweiterten Adressraum (echte RAM-Disk-Karte) genannt.

NUMBER? verwertet normalerweise nur Integerzahlen. Es benutzt dazu die Routine INUMBER?. In einem Gleitkommapacket kann an dessen Stelle auf ein Wort zur Behandlung von Gleitkommazahlen umgeleitet werden.

STATUS wird jedesmal aufgerufen, nachdem eine Eingabezeile interpretiert wurde. Es fuehrt im Urzustand nur CR aus. In einer Anwendung mit Bedienerfuehrung koennten dort z.B. Hinweise zum Systemstatus gegeben werden. Ich nutze es auch sehr oft beim Test neuer Definitionen, und lasse mir mit ' .S IS STATUS nach jeder Aktion automatisch den Stack zeigen. (Nicht vergessen dies zum Schluss wieder auf CR zu stellen).

Zu HANDLE moechte ich nicht allzuviel sagen. Es dient der bequemen und einfachen Umschaltung zwischen Interpreter- und Compilermode des Systems. Ich empfehle, keine Experimente damit anzustellen, da dieser Vektor eine Schluesselrolle im System einnimmt. Wer wissen moechte, wozu er dient und wie er funktioniert sollte sich mit dem Decompiler SEE die Worte

HANDLE, INTERPRET, [und] naeher betrachten. Dies ist auch fuer die anderen genannten Vektoren sinnvoll.

WHERE kommt bei der Fehlerbehandlung beim Laden von Screen zum Einsatz. Es tut normalerweise nichts. Beim Laden des Editors wird es auf (WHERE) umgestellt und verwendet dann Blocknummer und Fehlerposition auf dem Stack, um den Editmode aufzurufen und den Kursor an die richtige Stelle zu setzen.

?ERROR dient ebenfalls der Fehlerbehandlung. Wer eine andere Behandlung als die installierte benoetigt (z.B. Auschluss bestimmter Son. faelle) kann sich eine eigene Routine schreiben und den Vektor dorthin umleiten. (Studiert dazu ABORT" und ?ERROR).

GET und PUT sind Vektoren zum Lesen bzw. Schreiben der RAM-Disk von bzw. auf Band. Damit wurde es moeglich, nach der Fertigstellung des Systems dieses an die unterschiedlichen Aufzeichnungsverfahren der Computer anzupassen.

BOOT ist das erste Wort, welches nach dem Kaltstart des Systems ausgefuehrt wird. Es nimmt bis jetzt nur die Vorstellung des Systems und gewisse Voreinstellungen vor. Da es ein Vektor ist, kann man es benutzen um sogenannte Turn-Key-Programme zu schaffen, d.h. Programme, die nach dem Starten sofort mit ihrer Aufgabe beginnen und nicht erst in den Forth-Interpreter springen. Dies ist fuer Spiele, Programme mit Bedienerfuehrung usw. recht nuetzlich.
Ein weiteres Anwendungsbeispiel ist denkbar: Falls jemand verschiedene periphere Geraete an seinen Computer angeschlossen hat, kann man diese beim Einschalten des FORTH automatisch initialisieren lassen, die Schnittstellentreiber einrichten u.ae.

Mit den Vektoren ist es wie mit allen anderen Dingen auch. Man muss sich erst etwas mit ihnen beschaeftigen ehe man ihre volle Leistungsfaehigkeit erkennt. Dann jedoch wird man viele eigene Anwendungsfaelle finden und sich fragen, wie man das Problem eigentlich frueher ohne Vektoren geloest hat.