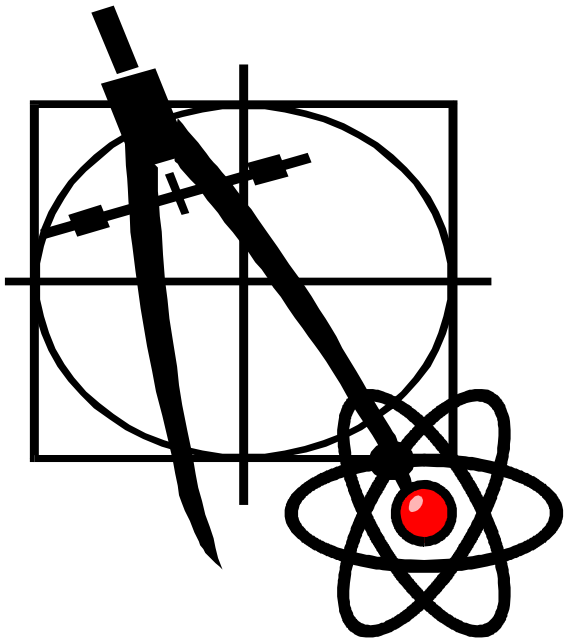
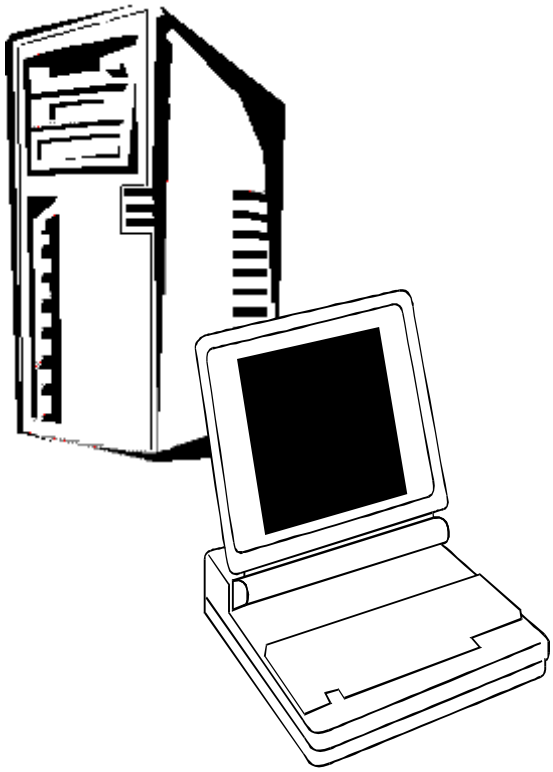
VIERTE DIMENSION

**für Wissenschaft und Technik, für kommerzielle EDV,
für MSR-Technik, für den interessierten Hobbyisten.**





In dieser Ausgabe:

14. Jahrgang - Ausg. 2/’98 - 10,- DM

Organ der FORTH-Gesellschaft e.V.

Dienstleistungen und Produkte fördernder Mitglieder des Vereins

Inhalt

In der nächsten Ausgabe finden Sie voraussichtlich:

- einen Bericht über die diesjährige Mitgliederversammlung der Forthgesellschaft e.V.
- die Frage nach einem MODERNEN Forth

Leserbriefe

Neue Adressen...
(Auszüge aus E-Mails Fred Behringer -> Friederich Prinz)

Mein Transputer-Forth-System f-tp 1.00 liegt jetzt auch auf dem Ftp-Server des Leibniz-Rechenzentrums der Bayerischen Akademie der Wissenschaften:

ftp://ftp.leo.org/pub/comp/platforms/pc/msdos/programming/forth/transputer/f-tp-100.zip

Marc Petremann, der Schöpfer des Turbo-Forth-Systems aus Frankreich, schreibt mir, daß sein System, und auch FASTGRAF, jetzt auch auf dem amerikanischen Server taygeta liegt:

ftp://www.taygeta.com/pub/Forth/MSDOS/turbof.zip
ftp://www.taygeta.com/pub/Forth/MSDOS/tfgraf.zip

Mehr darüber auf Marcs Homepage:

http://ourworld.compuserve.com/homepages/mp7

Meine E-Mail-Adresse hat sich geändert:

behringe@sunstatistik1.mathematik.tu-muenchen.de

hier noch ein Nachtrag...:

Ich habe es nicht ganz verstanden, aber bei meinen Versuchen hat es sich herausgestellt, daß die Programme turbof. zip und tfgraf.zip von Marc Petremann in einem anderen Unterverzeichnis liegen. Irgendwie habe ich dabei aber das Gefühl, daß es sich bei meinem Aufruf um eine Spiegelung von taygeta.com auf dem Ftp-Server des Leibniz-Rechenzentrums der Bayerischen Akademie der Wissenschaften handelte. Die Programme (je etw 400 KB) lagen in:

ftp://www.taygeta.com/pub/Forth/Compilers/native/dos/

übrigens liegt auch pygmy-forth und hforth (natürlich nicht von Marc Petremann) dort.

Fred Behringer

Neues aus der FIG SV...

Hallo, Forthfreunde!
Das November-Treffen der Silicon Valley Forth Interest Group kam eine Woche frueher dran -- um die FORML Forth Modification Conference am darauffolgende Wochenende in Asilomar, California, zu ermoeglichen und um keinen Konflikt mit dem Nationalfeiertag des Thanksgiving zu verursachen. Als Konsequenz werden wir wohl erwarten muessen, das im naechsten Jahr weniger Truthaehne die Forthgemeinschaft unterstuetzen werden.
Wie dem auch sei, das Novembermeeting war traditionell immer das groeszte des Jahres; wir nennen es den Forth-Tag.
Und es war auch dieses Jahr das groeszte Treffen, und das obwohl es regnete und Dr. Ting unter dem Vordach des kleinen Patio des Cogswell College grillen muszte. Ich habe ganz am Anfang 27 Leute gezaehlt und ich bin sicher, dasz die Zuhoererschaft die 40 im Laufe des Tages ueberschritten hat. Nach meinen Aufzeichnungen haben mindestens 9 Leute Vortraege gehalten:

1) John Rible beschrieb seinen SP16, eine Forth-Maschine fuer Ausbildungszwecke, die in einem 4000-Gate Quick-Logik FPGA laeuft und die er fuer den Unterricht in einem Hardware-Design Kurs benutzt.
2) Andy Korsak hat zu Hause keinen Computer, auf dem Windows laeuft (genau die Sorte Leute wie ich!), aber er konnte uns ueber seine Erfahrungen mit Win32Forth, welches er auf Arbeit benutzt, berichten. Bob Smith gab als Information bekannt, dasz Bug Updates fuer Win32Forth nach wie vor kostenlos sind und alle vorherigen Updates einschlieszen.
3) Bob Reiling lud alle zur kommenden FORML Konferenz in Asilomar ein, auf der Spitze von Kaliforniens wunderschoener Halbinsel Monterey. Das klingt gut: Eine gute Firma, interessante Gespraeche und technische Praesentationen, Geplauder am Feuer, Exkursionen, Wein- und Kaese-Parties (allerdings musz man dieses Jahr Korken-Trinkgeld fuer den Wein geben). Kein wahrer Forth'er wird diese Konferenz auslassen.
4) Dwight Elvey verlautbarte einige Worte ueber den TCOM Compiler fuer AD2100 DSP-Chips. Wie ueblich, verstand er seine Sache, aber ich war wohl nicht auf der selben Verstandes-Ebene. Damit ist das leider alles, was ich im Moment darueber berichten kann.
5) John Hall sprach ueber seine Arbeit am IEEE 1394 Firewire Serial Bus. Habe ich das richtig verstanden, dasz erwartet wird, dasz dieser Bus das 6-Draht SCSI Coax ersetzen wird?
6) John Carpenter fasste seinen Vortrag ueber einen Fuzzy-Interpreter zusammen, den er auf der kuerzlichen Euro-Forth Konferenz in Oxford gehalten hat. Lofti Zadeh war allerdings nicht da, um ein paar Fragen dazu zu stellen.
7) Bob Nash sprach ueber die Programmierung des Atmel 2051 Mikrocontrollers. Ich habe mir eine Notiz ueber seine Bemerkung gemacht, dasz "Forth im eigenen Kopf und nicht so sehr in der Implementation" ist.
8) Al Mitchell brachte eine Maschine zur Ueberwachung von Verunreinigungen mit, die zum groeszten Teil sein eigenes Design ist und die groszen, kommerziellen Geraete um Groeszenordnungen in Preis und Leistung schlaegt. Sein 'Periskop' kann den Softwareprozess bei laufendem System modifizieren. 40 Mikrosekunden fuer eine I2C-Uebertragung. Ich fragte ihn, ob er von Wolf Wejgaards Holon gehoert hat. Er sagte: "Ja".
9) Zum Schlusz offerierte schlieszlich Dr. Ting seine Version eines User Manuals fuer Windows Forth. "Es ist nur ein Anfang. Der beste Rat besteht darin, die Beispiele in Win32Forth zu nutzen und sie fuer die eigene Anwendung zu modifizieren."

Fortsetzung: Seite 6

Editorial

IMPRESSUM

Name der Zeitschrift

Vierte Dimension
Organ der Forth-Gesellschaft e. V.

Herausgeberin

Forth-Gesellschaft e.V.
Postfach 1110
85701 Unterschleißheim
Tel./Fax: 089-317 37 84
E-Mail:
SECRETARY@ADMIN.FORTH-EV.DE

Redaktion & Layout

Friederich Prinz
Hombergerstraße 335
47443 Moers
Tel.: 02841-58 3 98
E-Mail:
F.PRINZ@MHB.GUN.DE
FRIEDERICH.PRINZ@T-ONLINE.DE

Anzeigenverwaltung

Büro der Herausgeberin

Redaktionsschluß 1997

März, Juni, September, Dezember
jeweils in der letzten Woche

Erscheinungsweise

1 Ausgabe / Quartal

Einzelpreis
DM 10,- zzgl. Porto u. Verp.

Liebe Leser,

Wir, die Moerser Forther, haben für die erste in Moers zusammengestellte Ausgabe der VD Kritik und Lob bekommen. Das Lob hat deutlich überwogen und die Kritik war in allen Fällen konstruktiv. Die Ausgabe 01/98 hat Ihnen gefallen. Das freut uns natürlich sehr.

Wir wollen das Lob als Ansporn nehmen, und Ihnen weitere Ausgaben unserer Vereinszeitschrift zusammenstellen. **Dazu benötigen wir** nach wie vor **Ihre Beiträge.** Ihre Kritik soll uns natürlich ebenfalls Ansporn sein - zum Beispiel darauf zu achten, daß NIE WIEDER auf der Titelseite die Angabe zur Erscheinungsnummer der Ausgabe fehlt. Die von uns gewählte "Gestaltung" der Titelseite wollen wir aber nur dann aufgeben, wenn die Mitgliederversammlung das ausdrücklich so wünscht.

Die "Druckqualität" der VD hat ein wenig nachgelassen, wurde aber in allen Briefen und E-Mails an uns als durchaus akzeptabel bezeichnet. Hierzu ein Zitat: "*Ich finde Eure Arbeit ausgezeichnet, die Präsentation entspricht besser der kargen Wirklichkeit, und wenn man bei DEN Druckkosten DIE Lesbarkeit erreichen kann, ist das Direktorium auf dem besten Weg.*"

Inzwischen halten Sie die zweite "Moerser Ausgabe" in den Händen, die Sie zumindest schon einmal durchgeblättert - hoffentlich bereits vollständig gelesen haben. Haben Ihnen die einzelnen Beiträge gefallen ? Schreiben Sie uns doch einfach einmal ein paar Zeilen dazu. Oder besser: diskutieren Sie das mit uns auf der diesjährigen Jahrestagung.

Für das "Editoriat"
Friederich Prinz

VD im Internet

Die VD auch im Internet wiederfinden zu können, wird in den Netzwerkforen, in E-Mails an die Redaktion und auch in persönlichen Gesprächen immer wieder als Wunsch geäußert. Unsere Zeitschrift ist nicht nur für die Mitglieder der Forthgesellschaft interessant ! Es spricht auch wenig dagegen, die VD - mit entsprechender, zeitlicher Verzögerung - 'allgemein zugänglich' zu machen. Woran es scheitert ist einmal mehr der Wille zur Tat, sprich: Wir brauchen Jemanden, der uns 4 Mal im Jahr die dazu notwendige Arbeit machen will !

fep

Quelltext Service

Die Quelltexte in der VD müssen Sie nicht abtippen. Sie können diese Texte auch direkt bei uns anfordern und sich zum Beispiel per E-Mail schicken lassen. Schreiben Sie dazu einfach eine E-Mail an die Redaktionsadresse.

fep

Chuck war auch da, aber wir konnten ihn nicht zu einem Vortrag waehrend des Treffens ueberreden, und hinterher war ich nicht mit zum Essen. So, meine lieben Freunde, ist das alles, was ich im Moment darueber sagen kann.
Doch wartet... Ich lernte etwas, das ich Euch noch mitteilen moechte - von Dr. Morgenstern. Wie man mit einem Rechenschieber addiert: a+b=(a/b + 1)b.
Wunderbar. Jedoch, wenn es da unter Euch einige Forth'er gibt, die nicht wissen, was ein Rechenschieber ist, dann mag das ein gutes Zeichen sein. Das sind die Neulinge, die wir an Forth heranbringen wollen.
Froehliche Weihnachten und Gutes Neues Jahr!

Henry Vinerts

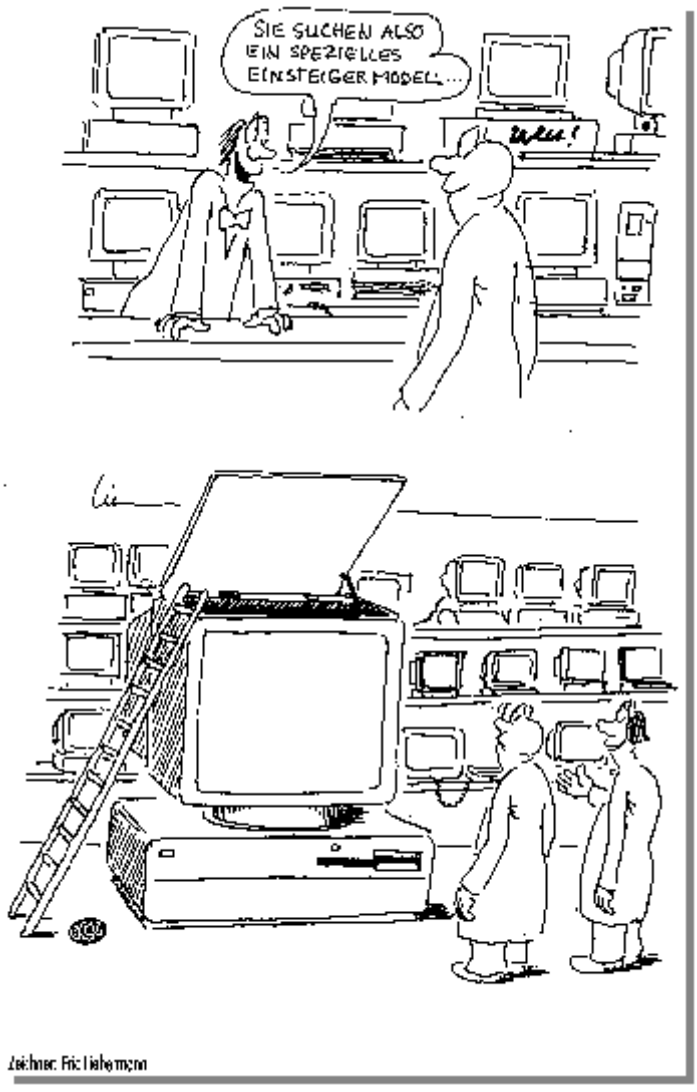
Die Übersetzung aus dem Amerikanischen hat Thomas Beierlein vorgenommen. Bis zum nächsten Weihnachten dauert es natürlich wieder ein wenig. Aber zu Henry's Gruß passt sicher die nachfolgende Bildergeschichte, die uns Klaus Kohl gesandt hat:

Platz da?! - "Nachtrag" zum ZF

Tom Zimmers ZfForth ist ein transparentes Werkzeug mit vielen Möglichkeiten. Als T. Zimmer es 1987 entwickelte, sah die PC-Hardwarewelt aus heutiger Sicht recht beschränkt aus (linear zu adressierender Speicher > 1MB war „Spielecomputern" wie dem ATARI und AMIGA vorbehalten). Festplatten des Normalverbrauchers bewegten sich in Größenordnungen um 20 bis 40 MB. Das hat sich (glücklicherweise?) bis heute geändert. Erschwingliche Festplatten liegen heute bei Kapazitäten zwischen 1 und 5 GB.

Tom Zimmers Editor zum ZFForth führt beim Öffnen der *.SEQ Files eine Überprüfung des freien (Disketten) Festplattenplatzes durch:

```
: ?diskfull ( --- f1 )
    shndl @ >nam 1+ c@ ascii : =
    if shndl @ >nam c@ bl or 96 -
  else 0
  then
  ( ----> ) getdiskfree * 0 128 um/mod nip *D
  toff @ tend @ negate + 0 128 um/mod swap
  if 1+
then
  0 D< dup
  if creeol ." WARNUNG !!"
     creeol ." Diskette voll !!"
     beep 1 seconds
     beep 1 seconds
then ;
```



„Getdiskfree" meldet die freien Cluster, die Anzahl der Bytes pro Sector und die Anzahl der Sektoren pro Cluster. Diese Angaben werden durch: „* 0 128 um/mod nip *D" weiterverarbeitet.

Die heutige Größe von Clustern und Sektoren kann bei großen Festplatten(partitionen) dazu führen, dass ein Überlauf stattfindet und der freie Platz falsch berechnet wird. Der Editor erzeugt eine Warnmeldung und gegebenenfalls beim Speichern des veränderten Files die Aufforderung eine neue Diskette einzulegen. Beides kann ohne weitere Folgen (bisher bei M.B) ignoriert werden. Lästig ist es aber allemal. Dem Überlauf und den lästigen Warnungen kann vorgebeugt werden, indem der Code von ?diskfull in der markierten (---->) Zeile leicht geändert wird:

getdiskfree * 0 128 um/mod nip U*D

M.Bitter; Mehrhoog

ETT (ED) und ÖMMES

Fred Behringer hat in der letzten VD ein kleines Floating-Point-Paket vorgestellt, das mit der vorgeschlagenen Änderung FSAVE, wegen eines Namenskonfliktes mit dem Wort FSAVE der Fileroutinenen von ZF u.a. Forthen, in FSTOR umzubenennen, fast problemlos in ZF lief. Fast: bei der Definition von FLDLN2 (ist hier FLD1LN2 gemeint?) steht „ CODE FLDLN2 (--- .....) D9 C, ED C, NEXT END-CODE" . Ungücklicherweise startet das Wort ED in ZF den Editor. Anstatt den Hexwert $ED zu speichern, startet der Interpreter den Editor und man findet sich überraschend in selbigem wieder.
Umformulieren in „CODE FLDLN2 (--- .....) D9 C, 0ED C, NEXT END-CODE" behebt dieses. Alternativ könnte man auch ED in ÖMMES umbenennen.
(ETT und ÖMMES (ruhrg. Ausdruck) -- „die da und der da ")

M.Bitter; Mehrhoog

Ein neues DOS ??

Viele können sich an das alte DR-DOS erinnern. Von vielen belächelt, hatte es meiner Ansicht nach einige Vorteile gegenüber dem Rest. Dieses wurde dann an verschiedene Hersteller verkauft und landete dann bei Caldera. Genau dort ist es jetzt als OPEN DOS erhältlich. Es kann gratis zum Download bezogen werden (Coldera.com). Das besondere dabei ist, dass der Quelltext ebenfalls erhältlich ist. (Lizenzhinweis beachten!). Ein DOS mit Netzunterstützung benötigt 5 Disketten.

Die Installation auf der (frisch formatierten) Platte hat es in sich. Vollständig menügeführt, anscheinend auch problemlos. Das System erscheint stabil. Tastatur und Codepage sind korrekt eingestellt. Unterstützung aller Windows ist enthalten. Ein Novell Client ist auch dabei. Ein HDD-Stacker kann zugeschaltet werden. Viele wichtige Tools sind auch vorhanden, unter anderem Debug und Exetobin. Aufgefallen ist mir ein stark verbessertes FDISK, dass lange Filenamen unterstützt werden, das BIOS-Jahr 2000 Problem wird gelöst, CDRoms werden automatisch erkannt und integriert, die EMM Konfiguration geht automatisch, ein Taskmanager, Security-Operationen, und Remotecomputing sind fest integriert.

Der vollständig vorhandene Quelltext in C und Assembler ist vernünftig strukturiert und erlaubt Einblicke und Experimente. Die Liste der integrierten Befehle zeigt, dass sich hier nicht viel geändert hat. Es scheint aber möglich, zusätzliche Befehle zu definieren. Ein DOS mit Forth-Batch-Sprache hat es noch (?) nicht gegeben. Denkbar wäre aber auch eine F-Shell.

Caldera stellt noch einige zusätzliche Tools zur Verfügung, die zum Erstellen der Exe Files notwendig sind. Auch das IBMIO.SYS kann erstellt werden. Leider fehlen die anderen Anweisungen.

Was mich besonders gefreut hat, ist die Hilfefunktion. Sie ist sehr viel besser als ihre Vorgänger, hier wird endlich mal erklärt (nix abbrechen/ ignorieren, ignorieren geht nicht). Allerdings muß hingenommen werden, daß alles auf englisch erklärt wird.

Dieses Dos ist wegen des Quelltextes sehr interessant, um Einblicke in die Programmierung dieses DOS zu erhalten. Wer noch einen älteren Rechner herumstehen hat, hat mit OPEN DOS ein gescheites Betriebssystem zur Hand.

Robert Freitag
Formesstraße 31 a
Köln

Nachtrag zu “Von der Stirne heiß...”,

Ausgabe 01/98, Seite 28

Im Quelltext zu THERMOMETER werden die Pfade absolut angegeben. Zur Veröffentlichung habe ich sie etwas gekürzt. Dadurch hat sich die Länge der Pfade und damit die Patchadresse für verschiedene Zifferbilder geändert. Das führt ohne Korrekturen zu Fehlern ! Änderungen bei :

```
25  Constant im_pfad    \ an Position 25 im Filepfad
                        \ steht die jeweilige Ziffer

Für ( 7seg_x.pcx) den richtigen Offset - hier 9 -
angeben, also:

9 Constant im_pfad     \ oder (besser) Forth selbst
                       \ die richtige Patchadresse
                       \ finden lassen:

Create Pcx ," .pcx"

: patch_ziffer ( n -- )
  Ascii 0 +
  PCX count bildpfad count search
  not IF abort" Konnte notwendige Bilddatei nicht
         finden!" THEN
  bildpfad + c! ;

: Ziffer ( n pos -- )
    swap " 7seg_0.pcx" bildpfad place
    patch_ziffer   2* positionen + @
    ecke display_y + 40 + -rot    + swap
    einblenden ;
```

M.Bitter

Das hat die Redaktion via E-Mail erreicht:

Hi,
Friederich, möchte mich hier mal für die VD bedanken. Schön das es weitergeht. Habt ihr gut gemacht - obwohl ihr das Titelblatt ja auch einer "Radikalkur" unterzogen habt :-)

Michael Kalus

Aus berufenem Munde tut jedes Lob doppelt gut. Die Verantwortung für unsere Zeitschrift hat auch schon in Michaels Händen gelegen.

Vielen Dank, Michael. Wir werden Dein Lob zum Ansporn nehmen.

fep

Hi all Mopsters, and anyone else who's interested, Yes, Mops v. 3.0 is here. The easiest way to get it is via the web site:

<URL:http://www.netaxs.com/~jayfar/mops.html>

The manual has been revised for 3.0, and is now uploaded in both Microsoft Word 5 for Mac format, and PDF (Adobe Acrobat). It may be a couple of days before it's available for downloading, but in the meantime, Mops 3.0 itself is ready to go. PowerMops (native for PowerPC) is included, but doesn't have floating point yet. I'm getting started on this now, and it will be in v. 3.1.

Diese interessante Meldung hat *Claus Vogt* “aus dem Netz gefischt” und der Redaktion gesandt.
Cheers, Mike

Mike Hore mikeh@zeta.org.au

Die Entwicklung von HolonForth, Teil 2: Wie man Forth echt interaktiv macht.

Wolf Wejgaard
Forth Engineering, CH-6045 Meggen <wejgaard@acm.org>
+41 (0)41 377 3774 Fax +41 (0)41 377 4774

Forth ist eine interaktive Programmiersprache, aber die meisten Forth-Systeme sind nur interaktiv auf der Wortebene, nicht auf der Programmebene. Das ist schade, denn Forth bietet vom Konzept her die Funktionen für eine volle Interaktivität und damit einen weiteren grundsätzlichen Vorteil vor anderen Programmiersprachen. HolonForth ist voll interaktiv, das bestehende Programm kann unmittelbar nach einer Änderung getestet werden. Mit HolonForth können Sie am "lebenden" Programm arbeiten. Ich beschreibe in diesem Artikel, wie ein Forth-System echt interaktiv wird.

Stichworte: ***Forthsysteme, Interaktivität, Holon, Direktheit, Codeaustausch, Sofortheit***

Interaktiv

Forth ist, wie jeder weiss, ein interaktive Programmiersprache. Man kann einzelne Worte laden und sofort testen. Und man kann ein Wort korrigieren, neu laden und sofort interaktiv testen. So weit, so gut. Als Programmierer bin ich aber nicht nur an *Worten* sondern am *Programm* interessiert! Und am Programm kann ich im konventionellen Forth-System *nicht* interaktiv arbeiten.

Können Sie das? Wenn Sie im Programm ein Wort ändern, wie lange dauert es, bis Sie das Programm mit der Korrektur testen können? Das Programm muss neu geladen werden, dann gestartet und schliesslich wieder in den Zustand gebracht werden, den Sie testen wollen. Schätzungsweise 1 Minute, bis Sie wieder so weit sind? Und wenn Ihr Editier-Test-Zyklus nur zehn Sekunden dauert: ist das *interaktiv*?

Die schlichte Erkenntnis ist diese: Konventionelles Forth ist interaktiv auf der Wortebene, aber durchaus nicht auf der Programmebene. Abgesehen von einigen Ausnahmen. Vielleicht erinnern Sie sich an Fifth? Dieses schöne Forth-System von Mitte der achtziger Jahre konnte das Programm interaktiv ändern, indem es das korrigierte Wort neu lud und den entsprechenden Code im Programm austauschte. Genau so macht es Holon.

Forth kann es

Forth ist prädestiniert zur Interaktivität auch auf der Programmebene. Man muss das System nur ein bisschen umorganisieren. Fifth und Holon unterscheiden sich von konventionellen Forth Systemen darin, dass sie das Dictionary, diese wunderbare Datenbank in Forth, voll ausnutzen und nicht nur den Programmcode sondern auch den Programmtext damit verwalten. Anders gesagt, der Quelltext steht nicht in Textdateien oder Textblökken, sondern wird als Datenbank organisiert.

Das hat bereits auf der Textseite erhebliche Vorteile, wie ich im ersten Teil dieses Artikels beschrieben habe. Man hat immer das ganze Programm vor Augen, klar strukturiert und bestens lesbar. Man kann direkt auf jedes Wort zugreifen, ohne in einer Unzahl von Blöcken oder Dateien blättern zu müssen, Hypertext-Verknüpf-ungen fallen als Nebenprodukt automatisch an. Das heisst, in der Definition eines Wortes lässt sich jedes Unterwort per Funktionstaste nachschlagen (Definition und Kommentar).

Programmcode austauschen

Die Datenbank hat aber auch für die Verwaltung des Programmcodes interessante Aspekte. Wenn ich bei jedem Wort neben der Codeadresse auch die Codelänge im Dictionary speichere, ist es ein Einfaches, Code im Programm auszutauschen. Wenn der Code am alten Ort Platz hat, wird er dort abgelegt. Falls der neue Code am alten Ort keinen Platz hat, wird er hinten angehängt, und am alten Ort wird ein Sprung zum neuen Ort eingesetzt.

Viele Forth-Programmierer haben bereits von Hand Code in einem Programm geflickt, auch in lebenden Programmen. Dazu muss das Programm nur als ein Task und der Interpreter als ein anderer Task laufen. Wenn der neue Code am alten Ort Platz findet, wird er einfach ausgetauscht. Ich würde das allerdings an einem lebenden Programm nur machen, wenn ich sicher bin, dass das Programm sich nicht gerade in diesem Wort oder einem Unterwort davon befindet. Aber das kann ich wissen, denn ich bestimme ja selbst durch die Lage des Taskwechselwortes (oft PAUSE genannt), wo sich das Programm in dem Moment befindet, in dem ich den Code wechsle.

Wenn der Code hinten angehängt wird, wird am alten Ort ein Sprung zum neuen Ort eingesetzt. Wie dieser Sprung genau aussieht, hängt vom Worttyp und der konkreten Forth-Maschine ab.

Bei einem Code-Wort ist es klar. Hier ist der Sprung ein "Jump" zum neuen Ort .

Bei Colon-Worten spielt die Fädelungsart eine Rolle. Bei Subroutinen-Fädelung und bei direkter Fädelung können wir ebenfalls einen "Jump" als Maschinenbefehl einsetzen. Diese Worte beginnen ja mit ausführbaren Maschinencode und werden direkt vom Prozessor angesprungen. Wir leiten den Prozessor nur sanft zum neuen Ort um.

Bei Token-Fädelung wird einfach die Adresse in der Tokentabelle umgesetzt.

Die indirekte Fädelung verlangt etwas mehr Nachdenken. Hier beginnt das Colon-Wort nicht mit einem Maschinenbefehl, sondern mit der Adresse von NEST (auch DOCOLON oder ähnlich genannt). NEST gibt bekanntlich, nach Zaubereien mit dem IP, die Kontrolle an das Wort weiter, das auf NEST folgt. Und dieses Wort gibt gelegentlich die Kontrolle an das folgende Wort ab, und so weiter. Gerade das aber wollen wir hier nicht, denn damit wird ja das alte Worte ausgeführt, das eigentlich ersetzt werden soll.

Die Kontrolle lässt sich elegant an den neuen Code weitergeben mit folgender Methode. Wir bilden ein neues Code-Wort NESTFAR und setzen dieses an den Anfang des alten Wortcode zusammen mit einem Zeiger zum neuen Wortcode. NESTFAR macht zunächst dasselbe wie NEST, aber übergibt die Kontrolle dann nicht an das folgende Wort, sondern an *das Wort, das im neuen Wortcode auf das Anfangs-NEST folgt*. Das kann NESTFAR machen, denn die Adresse vom neuen Wortcode liegt ja jetzt im alten Code hinter dem Aufruf von NESTFAR.

Im neuen Wortcode steht natürlich das normale NEST am Anfang. Damit ist der neue Wortcode voll wirksam. Programmworte, die die alte Version anspringen, werden zur neuen Version umgelenkt. Neue Worte, die nach dieser Wortkorrektur geladen werden, benutzen direkt die neue Version. Das Dictionary zeigt jetzt auf die neue Version.

Jedes Forth-System kann interaktiv sein

Vielleicht haben Sie es bereits entdeckt. Das automatische Auswechseln des Codes und damit die volle Interaktivität in Forth können Sie auch in einem konventionellen Forth-System verwirklichen. Es ist mir ein Rätsel, warum das noch keiner macht. Die Rostocker Gruppe hat einmal in der VD beschrieben, wie man in COMFORTH Code flicken kann (leider finde ich das Heft nicht mehr, und kann deshalb keine Referenz angeben - vielleicht fügt der liebe Redakteur die Referenz hier ein -), aber das war harte Handarbeit. Es geht auch automatisch.

Wenn Sie darauf verzichten, Code am alten Ort einzusetzen, brauchen Sie nicht einmal das Dictionary zu verändern. Sonst müssten Sie ein neues Feld einführen, in dem die Codelänge der Worte gespeichert wird. Auch kein Problem. Dann wird der Interpreter so erweitert, dass er beim neuen Laden eines bestehenden Wortes den Code ersetzt oder den besprochenen Sprung zum neuen Code am alten Ort einsetzt. Der Interpreter weiss ja, dass das Wort neu geladen wird, da es bereits im Dictionary existiert. Also setzt er den Code neu ein und korrigiert den Codezeiger. Im Detail gibt es da noch je nach Forth-System ein paar Aufgaben zu lösen, aber wer diese Interaktivität wirklich will, und den vollen Quelltext und einen Cross-Compiler für sein Forth-System hat, sollte das eigentlich ohne fundamentale Schwierigkeiten realisieren können. Eine nette Aufgabe für den fortgeschrittenen Forth-Programmierer.

Es lohnt sich

Der Lohn ist ein neues Programmiergefühl. Man arbeitet ohne Ladepausen in voller Konzentration. So wie im normalen Leben bei einer normalen Arbeit. Jeder Handwerker will direkt sehen, was er macht. Die Sofortigkeit ist wichtig! Was halten Sie von einem Textverarbeitungsprogramm, das jeweils 10 Sekunden braucht, um zu zeigen, was Sie schreiben oder ändern? Wetten, dass Sie das Programm sehr schnell aus Ihrem Leben verbannen würden?

Nur Programmierer leben in einem merkwürdigen Zustand. Sie wollen die schnellsten Editoren und sie bauen ihren Kunden Programme, die sofort reagieren – Ehrensache! Sie verwenden optimierende Compiler, die jedes Quentchen Speed aus dem Programm herausholen. Jedoch beim Arbeiten am Programm selbst sind sie unheimlich geduldig. Warten und warten und warten. Verstehen Sie mich recht, ich habe nichts gegen Pausen beim Programmieren. Denkpausen müssen sein. Nur die Wartepausen vermeide ich gerne.

Wie Sie sich an Hand von Holon schnell überzeugen können, ist das Warten nicht notwendig. Forth wurde das automatische Auswechseln von Code in die Wiege gelegt, doch leider haben es fast alle bisher auch dort liegen lassen. Bei Forth ist der Codeaustausch prinzipiell leicht möglich, weil erstens der Codezeiger im Dictionary steht und zweitens Forth den Quelltext eines Wortes direkt in lauffähigen Code verwandelt. Bei C und anderen Sprachen braucht das Codetauschen einiges mehr an Aufwand.

Wieder ein grundsätzlicher Vorteil von Forth, der von den meisten Forth-Systemen nicht ausgenutzt wird. Eigentlich schade.

Wolf Wejgaard bezieht sich vermutlich auf den Artikel:
Ein Rahmen für modular aufgebaute Forth-Systeme
von ***Malte Köller***
VD 3/1996; Seite 11
die Redaktion

Real-Mode-32-Bit-Erweiterung für Turbo-Forth

von Fred Behringer
Planegger Str. 24, 81241 München

Die Segmentgröße des Real Mode unter DOS von 64 KByte auf 4 GByte aufblähen. Nach Vorarbeiten von Harald Albrecht aus c't 1/90 von mir für Turbo-Forth (zunächst noch 16 Bit) zurechtgestutzt. Turbo-Forth stammt von Marc Petremann und Mitarbeitern aus der französischen Forth-Gruppe JEDI. Es können nach dem Umschalten alle 32-Bit-Befehle des 486 verwendet werden, ohne daß bei Zugriffen jenseits der 1MB-Grenze das System streikt. Als Anwendung habe ich ein Primzahlensieb eingebaut, das den gesamten RAM-Bereich (bei mir 64 MB) ausnützt. Ist 63963131 eine Primzahl und ist 63963077 63963079 ein Primzahlzwillingspaar? Nach 30 Sekunden Wartezeit Antwort auf beide Fragen: Ja . - Wieviele Primzahlzwillinge gibt es eigentlich? Unendlich viele?

Stichworte: ***Turbo-Forth-16, 80486, DOS, 4 GB linear, Primzahltest*** *über 64 Millionen in 30 sec.*

Ziel: Ich habe 64 MB RAM. Ich arbeite mit einer RAM-Disk von 12 MB. Mit PC-TOOLS kann ich die einzelnen Dateien auf der RAM-Disk untersuchen, kann aber nicht erfahren, an welcher RAM-Adresse die Informationen liegen. Außerdem möchte ich die Organisation der RAM-Disk im laufenden Betrieb untersuchen. Ich möchte den gesamten RAM-Speicher von Turbo-Forth aus mit einem DUMP-ähnlichen Befehl sichten können. Beim Arbeiten mit Turbo-Forth liegen die 64 MB RAM brach. Ich könnte also für Experimentierzwecke Riesenmengen an Daten gezielt irgendwo dorthin legen, zwischenspeichern, ändern, wieder abrufen usw. Mit einem Suchbefehl möchte ich, immer unter DOS von Turbo-Forth aus, schnellstens bestimmte Stellen in dem 64-MB-RAM-Bereich aufspüren.

Lösung: Die Adreßleitung A20 wird über FREE-A20 freigeschaltet und bleibt freigeschaltet. Es wird eine General Descriptor Table (GDT) eingerichtet und in bestimmter Weise belegt (siehe gleich). Es wird SIZEUP aufgerufen. Dadurch wird das System zunächst in den Protected Mode geschaltet (einfach durch Setzen des dafür zuständigen Bits im Register CR0). Es wird jetzt (im Protected Mode) die GDT aufgerufen. Dadurch werden in die Segment-Cache-Register 4-GB-Grenzen (bei 4-KB-Granularität) eingetragen. Sodann wird wieder in den Real Mode zu-rückgeschaltet. Wegen Feinheiten siehe SIZEUP. Die Adressen werden zwar weiterhin in der üblichen Form SEGMENT:OFFSET angesprochen, OFFSET läßt sich aber jetzt (nach wie vor im Real Mode!) durch Verwendung der erweiterten Register (EAX usw.) in 32-Bit-Breite ansetzen.

Alle Assembler-Befehle, in denen erweiterte Register vorkommen, werden über ein Präfixbyte 66h angesprochen. In die betreffende CODE-Definition setzt man hierzu einfach 66 C, ein. Alle Befehle, die sich auf 32-Bit-Adressen beziehen, erhalten das Präfixbyte 67h (in der betreffenden CODE-Definition also 67 C,).
Interessant ist, daß beim 80486 auch noch die beiden Seg-mentregister FS und GS zur freien Verfügung stehen. Ich setze sie (siehe unten) durch FS! und GS! und lese sie durch FS@ und GS@. Alle 32-Bit-Speicherbefehle (C@-32 CC@-32 @-32 C!-32 CC!-32 !-32) nehmen ihre 32-Bit-Adressen als Offset zum FS-Wert an. Setzt man 0 FS! , dann entspricht dieser Offset der physikalischen (linearen) Adresse.

Erklärung und Anleitung: Die Bildschirmausgaben bei DUMP-32 erscheinen hexadezimal. Die Adressen bei DUMP-32 werden als Offset zum Selektor FS angegeben. Vorgabe für FS ist 0. Will man Übereinstimmung mit dem eigentlichen DUMP des 16-Bit-Betriebs haben, so muß man per DSEGMENT FS-DUMP-32 ! das Segment FS für die Zeit des DUMP-Aufrufs auf den Wert von DS legen. Angenommen, DS hat den Hex-Wert 2775 . Dann wird bei Eingabe der Hex-Adresse 1000000., was 16MB entspricht, die effektive Adresse 1027750. angesprochen. Über die Taste "+" kann man (beliebig oft) um einen Bildschirm (256 Bytes) weiterschalten, mit der Taste "-" zurück. Die Taste "Return" beendet DUMP-32. Die Einstellung ASCFILT OFF bewirkt, daß beim Bildschirmausdruck rechts alle darstellbaren Zeichen, z.B. auch das lachende Gesicht für ASCII-2, ausgegeben werden. Bei Einstellung ASCFILT ON erscheinen alle Nicht-ASCII-Zeichen als Punkt. Das erhöht die Übersicht, wenn man bestimmte Stellen im Forth-System sucht. ?ASCII sorgt in DUMP-32 dafür, daß nicht oder nur schlecht wiedegebbare Zeichen, wie CR, als Leerstelle ausgegeben werden.

Nach Eingabe von 12345678. 1000 2! erscheinen in Turbo-Forth an den Adressen 1000 bis 1003 (bezogen auf das Segment DS) die Werte 34 12 78 56 . Diese Umstellung tritt z. B. bei Belegung von 2Variablen oder auf dem Stack auf. In den erweiterten Registern (EAX...), bei Weiterverarbeitung mit arithmetischen Operationen, würde sich diese Umstellung unnatürlich ausnehmen. Ich habe es daher, zumindest für die Zwecke der vorliegenden Arbeit, als natürlich angesehen, bei Eingabe von 12345678. 1000. !-32 an den Adressen 1000. bis 1003. die Werte 78 56 34 12 auftreten zu lassen. Man prüfe das mit Hilfe von 1000. @-32 nach. (Man beachte die Punkte bei den 32-Bit-Adreß-Angaben.)

Aus der gemischten Programmierung in den diversen CODE-Definitionen sollte folgendes klar werden: Man braucht sich

nicht mühsam einen eigenen Cross-Assembler zu konstruieren, um eine 32-Bit-Erweiterung von Turbo-Forth "cross"-zuassemblieren. Ein paar schnell ergänzbare Erweiterungen genügen, wenn man im übrigen bereit ist, sich die wirklich benötigten Befehle aus den Unterlagen herauszusuchen und per C, aufzunehmen. Andererseits kann man sich aber auch getrost schrittweise an das Projekt eines Cross-Assemblers wagen. Er braucht ja nicht sofort alles zu überdecken.

Bei PRIMES werden die Zahlen dezimal durchnumeriert, und zwar als Summe aus links stehender und obenstehender Zahl. Eine "1" im Kreuzungspunkt bedeutet, daß die betreffende Zahl eine Primzahl ist. Sie ist im Sieb von Eratosthenes hängengebleiben. "0" bedeutet "keine Primzahl". Jede andere Ziffer deutet darauf hin, daß etwas schiefgelaufen ist. Das Programm kommt ohne Multiplikation und Division aus, die beide je 40 Takte benötigen würden. Addition und Multiplikation benötigen nur je einen einzigen Takt.

Schnelleinstieg: INCLUDE TFORTH32 0 300000. 4000000. SIEVE DECIMAL 3400. 500. PRIMES Von Turbo-Forth aus aufgerufen, liefert das die Primzahlen von 3400 bis 4000 auf dem Bildschirm. (Primzahlen haben eine "1", Nichtprimzahlen eine "0". Ich habe 64 Megabyte RAM. Bei anderer Bestückung muß die Eingabe "4000000." geändert werden. "300000." legt den Anfang der Primzahlentabelle auf 3 Megabyte. Damit bleibt meine RAM-Disk intakt. 0 FS! setzt das Segmentregister FS auf 0. FS@ holt dessen Wert. 12345678. 300000. !-32 legt den Wert 12345678 an die 4 Bytes ab Adresse 300000 (bezogen auf FS), 300000. @-32 liest diesen 4-Byte-Wert wieder aus. 300000. 100. dump-32 liefert ein Hex-Dumping von etwas 100 Bytes ab Adresse 300000 . 200000. 1000000. 12345678. FIND-VALUE durchsucht den angegebenen Bereich nach dem Wert 12345678 . Mit 55AA. gelangt man so zum BIOS-Bereich.

Caveat: Nur für 80486 (und höher). Darunter zwecklos. EMM386.EXE stört das vorliegende Programm. HIMEM. SYS schaltet A20 frei und beläßt es so. Mit HIMEM.SYS im System läßt sich A20 nicht blockieren. Wäre ja auch unlogisch, da HIMEM.SYS das "HIMEM" zur Verfügung stellt. Bootet man das System ohne HIMEM.SYS in CONFIG.SYS, dann kann man über FREE-A20 und LOCK-A20 die Adreßleitung A20 lösen oder blockieren. Im MS-DOS-Fenster unter Windows 95 funktioniert das vorliegende Programm (natürlich) nicht. Schaltet man beim Booten von Windows 95 im Menü zuerst nach DOS 7.0 , dann läuft das Programm ausgezeichnet. Geht man über BYE wieder aus Turbo-Forth raus, dann gelangt man anschließend ohne weiteres durch Eingabe von WIN [RET] zur Benutzer-Oberfläche von Windows 95. **Achtung:** Auch aus CONFIG.SYS von Windows 95 (CONFIG.W40, von DOS aus gesehen) muß EMM386.EXE entfernt werden.

Manifest: Es ist noch gar nicht so lange her, da der 80286 als Non-plus-ultra angepriesen wurde. Man hatte zwar als Computer-Freak auch schon mit dem 68000 herumexperimentiert, man liebäugelte dann aber ebenfalls mit dem 80286, da alle Welt das tat und die Zeitschriften nur noch über diesen einen berichteten. Man wartete. Irgendwann, als in Amerika schon längst die nächste Prozessor-Generation angesagt war, rutschte der 80286 in die bürgerliche Preisklasse ab und man schlug zu. Mit Hilfe der c't und anderer guter Quellen versuchte man, die Defizienzen auszutricksen. "Zwischen den Adaptern", "Above Board" und was man sich nicht alles einfallen ließ. Heutzutage (1998) erntet man in den Computergeschäften bei den immer wieder nachwachsenden stets viel zu jungen Verkäufern verständnisloses Kopfschütteln, wenn man gesteht, daß man "noch" mit einem 80486 arbeitet. "Längst passé". Den 80286 oder den 80386 wagt man gar nicht erst zu erwähnen. Das heißt also, es ist jetzt an der Zeit, den 80486 als Ausgangspunkt der Betrachtug zu wählen. Das soll im folgenden geschehen.

Was ist aber am 80486 besser ? Der Wirrwarr mit dem Protected Mode, mit den Selektoren und Deskriptoren, mit den Zulassungsebenen und den "nicht behebbaren allgemeinen Schutzverletzungen" ist eine Zumutung. Beim Arbeiten mit Windows 95 steigt mir nur allzu oft das Gesamtsystem aus - nicht nur ein Teil davon, das gesamte System. Wenn das nun aber schon so ist und man nichts dagegen unternehmen kann, dann soll es mir auch recht sein, wenn beim Arbeiten mit Forth das Gesamtsystem zusammenbricht. Bei einer undurchsichtigen Windowskomponente, auf die ich nicht den geringsten Einfluß habe, stört mich das. Bei Forth nicht. Bei Forth macht es mir Spaß. Ich lerne dabei. Bei Windows lerne ich nichts.

Ich brauche keinen "Protected Mode". Ich bin kein Multi-User. Ich bin mein eigener Super-User. Ich habe eine Maschine, die noch vor kurzem der Stolz eines jeden Rechenzentrums gewesen wäre. Diese Maschine gehört mir allein und ich will sie in voller Breite von 32 Bit ohne künstlich aufgezwungene Behinderung linear adressieren - über einen RAM-Bereich von zur Zeit 64 Megabyte hinweg.

Über Windows 95, OS/2-Warp und Linux freue ich mich, wenn ich per Modem meine E-Mail abhole oder wenn ich mir mit Hilfe von Corel Draw einen mathematischen TTF-Zeichensatz nach eigenem Gusto erschaffe. Die hierzu nötigen Werkzeuge könnte ich mir nicht selbst herstellen, oder besser, würde ich nicht selbst erstellen wollen. Ich habe andere Interessen. Und zu deren Verwirklichung hätte ich gern den vollen Zugriff auf meine Maschine. Es stört mich, wenn ich daran gehindert werde.

Andere 32-Bit-Forth-Quellen und Primzahlen:
(Quellen, die keine Metacompilation und CODE-Definitionen zulassen, interessieren mich nicht. Quellen, die Geld verlangen, auch nicht. Die Ehre der Anerkennung sollte reichen. In der Wissenschaft ist das seit Jahrhunderten so.)

VanNorman, Rick: 32-Bit Protected-Mode Subroutine Threaded Forth. Liegt in 2 Versionen auf ftp://www.taygeta.

com/... : DOS über DPMI und OS/2 im Textmodus.

Astle, Richard: Forth in 32-bit Protected Mode. Forth Dimensions Jan. 1995, S. 8-20. Verwendet DPMI.

Schröder, Michael: Extending Forth - Ein Protected-Mode-Experiment. Vierte Dimension 11 (1995), Heft 1, S. 19-22. Verwendet DPMI.

Petremann, Marc: Turbo-Forth-32. Liegt auf ftp://www.taygeta.com/ Läßt sich beim Metacompilieren auf Prozessoren über und unter dem 80386 einstellen (16- oder 32-Bit-Betrieb). Die zur Metacompilation benötigten Quelltexte werden auf taygeta nicht mitgeliefert. Es lassen sich zwar die Register in erweiterter Form ansprechen (EAX ..), ein Adreßzugriff oberhalb der 1-MB-Grenze ist jedoch nicht möglich.

Albrecht, Harald: Grenzenlos - Vier Gigabyte im Real Mode des 80386 adressieren. c't 1990, Heft 1, S. 212-220. Dieser wirklich sehr interessante Artikel war der Ausgangspunkt für die vorliegenden Untersuchungen. Dinge, die hier undurchsichtig bleiben, können dort nachgelesen werden. Mir war eine sofort nachvollziehbare Forth-Realisation wichtiger als eine 100prozentige Beschreibung dessen, was auch woanders steht.

An DPMI (DOS Protected Mode Interface) gelangt man z.B. in der DOS-Box von Windows 3.11 . Mein Programm kommt ohne DPMI aus.

Behringer, Fred: Forth on the IBM; Prime Numbers. JEDI, le Journal qui pedale Forth 48, Dezember 1988, S.9 . Das Programm zur Bestimmung der größten ganzen Zahl <= der Quadratwurzel einer gegebenen ganzen Zahl doppelter Genauigkeit, [SQR], eine Art sukzessiver Approximation mit Binärentwicklung, ist dort enthalten. Man findet hier auch Vorschläge zur RAM-sparenden Komprimierung des Primzahlensiebs.

Behringer, Fred: Forth on the IBM; Prime Numbers. CompUser International Computing 2 (1989), H. 3, S. 38.

Flipo, Daniel: More About Eratosthenes' Sieve; Prime Numbers with Forth. CompUser International Computing 2 (1989), Heft 4, S. 29.

JEDI und CompUser sind Zeitschriften der gleichnamigen Vereine, JEDI in Frankreich, CompUser in Holland. Beide Vereine waren jahrelang sehr aktiv. Beide Vereine existieren leider nicht mehr.

Scheid, Harald: DUDEN, Rechnen und Mathematik. Mannheim 1985. Auf den Seiten 498 und 499 stehen die Primzahlen im Bereich von 2 bis 4500. Ich habe mein Programm anhand dieser Tabelle überprüft. Das beweist natürlich nicht, daß es korrekt arbeitet, es erhöht aber das Vertrauen in seine Richtigkeit.

Ab hier Programmteil

```
CR .( 32-Bit-Assembler-Erweiterung )

HEX ONLY FORTH ALSO ASSEMBLER DEFINITIONS

: FS:      64 C, ;  : GS:      65 C, ;  : OPSIZE:  66 C, ;  : ADRSIZE: 67 C, ;
: EAX OPSIZE: AX ;  : ECX OPSIZE: CX ;  : EDX OPSIZE: DX ;  : EBX OPSIZE: BX ;
: ESP OPSIZE: SP ;  : EBP OPSIZE: BP ;  : ESI OPSIZE: SI ;  : EDI OPSIZE: DI ;

CR .( Freischalten der Adreßleitung A20 )

FORTH DEFINITIONS

CODE FREE-A20 ( -- )                  \ Adreßleitung A20 bereitstellen
     CLI BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
         0D1 # AL MOV  64 # AL OUT        \ Kommando "Write Output Port"
         BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
         0DF # AL MOV  60 # AL OUT        \ A20 freigeben
         BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
     STI NEXT END-CODE

CODE LOCK-A20 ( -- )                  \ Adreßleitung A20 sperren
     CLI BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
         0D1 # AL MOV  64 # AL OUT        \ Kommando "Write Output Port"
         BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
         0DD # AL MOV  60 # AL OUT        \ A20 sperren
         BEGIN 64 # AL IN ( Statusport ) 02 # AL AND ( Input-Buffer-Flag ) 0= UNTIL
     STI NEXT END-CODE

: A20? ( -- fl )  0FFFF 10 L@ 0. L@ <> ;               \ fl = 0  -->  A20 gesperrt

CR .( Segmentgrenzen auf 4 Gigabyte setzen )

VARIABLE >GDT 4 ALLOT  VARIABLE GDT  0E ALLOT  2VARIABLE FARJMP  DSEGMENT FARJMP 2 + !
```

```
18 >GDT      !   0. GDT      2!   0. GDT 04 + 2!   0FFFF GDT 08 +  !
 0  GDT 0A + !   0  GDT 0C + C!   92 GDT 0D + C!     0CF GDT 0E + C!  0 GDT 0F + C!

CODE SIZEUP ( -- )                    \ Segmentgrenzen auf 4 GB setzen
     CS AX MOV ( Lage der GDT berechnen ) OPSIZE: 0F C, 0B7 C, 0C0 C, ( AX EAX MOVZX )
     OPSIZE: 0C1 C, 0E0 C, 04 C, ( 4 # EAX SHL ) GDT # BX MOV
     OPSIZE: 0F C, 0B7 C, 0DB C, ( BX EBX MOVZX ) OPSIZE: 03 C, 0C3 C, ( EBX EAX ADD )
     OPSIZE: 0A3 C, >GDT 2 + , ( EAX [>GDT+2] MOV )
     PUSHF DS PUSH ES PUSH                                  \ Flags und
     0F C, 0A0 C, ( FS PUSH ) 0F C, 0A8 C, ( GS PUSH )      \ Segment-Register sichern
     CLI                                                    \ Kein Interrupt mehr
     2E C, 0F C, 01 C, 16 C, >GDT ,                         \ GDT setzen
     \ -----------
     0F C, 20 C, 0C0 C, ( CR0 EAX MOV ) 66 C, 83 C, 0C8 C, 01 C, ( EAX 1 OR )
     0F C, 22 C, 0C0 C, ( EAX CR0 MOV )                     \ Jetzt im Protected Mode
     0EB C, 0 C,                                            \ JMP auf nächsten Befehl
     \ -----------
     08 # AX MOV   AX DS MOV   AX ES MOV                    \ 4-GB-Grenze jetzt in
     8E C, 0E0 C, ( AX FS MOV ) 8E C, 0E8 C, ( AX GS MOV ) \ die Schattenregister
     \ -----------
     0F C, 20 C, 0C0 C, ( CR0 EAX MOV ) 66 C, 83 C, 0E0 C, 0FE C, ( NOT 1 EAX AND )
     0F C, 22 C, 0C0 C, ( EAX CR0 MOV )                     \ Raus aus Protected Mode
     2E C, 0FF C, 2E C, FARJMP , ( JMP CS:[FARJMP] )
     HERE FARJMP !
     0F C, 0A9 C, ( GS POP ) 0F C, 0A1 C, ( FS POP ) ES POP DS POP POPF \ Restaurieren
     NEXT END-CODE

CR .( Datenverkehr über den gesamten Systembereich )

CODE FS@ ( -- cc ) 0F C, 0A0 C, ( FS PUSH ) NEXT END-CODE  \ Segmentwert holen
CODE GS@ ( -- cc ) 0F C, 0A8 C, ( GS PUSH ) NEXT END-CODE
CODE FS! ( cc -- ) 0F C, 0A1 C, ( FS POP  ) NEXT END-CODE  \ Segmentwert setzen
CODE GS! ( cc -- ) 0F C, 0A9 C, ( GS POP  ) NEXT END-CODE

CODE C@-32 ( ad. -- c ) EBX POP       \ Von 32-Bit-Adresse ad. Byte c holen
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL ) AX AX XOR ( AH = 0 )
     FS: ADRSIZE: 8A C, 03 C, ( FS:[EBX] AL MOV ) 1PUSH END-CODE
CODE C!-32 ( c ad. -- ) EBX POP       \ Byte c nach 32-Bit-Adresse ad. speichern
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL ) AX POP
     FS: ADRSIZE: 88 C, 03 C, ( AL FS:[EBX] MOV ) NEXT END-CODE
CODE CC@-32 ( ad. -- cc ) EBX POP     \ Von 32-Bit-Adresse ad. Doppelbyte cc holen
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL )
     FS: ADRSIZE: 8B C, 03 C, ( FS:[EBX] AX MOV ) 1PUSH END-CODE
CODE CC!-32 ( cc ad. -- ) EBX POP     \ D-Byte cc nach 32-Bit-Adresse ad. speichern
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL ) AX POP
     FS: ADRSIZE: 89 C, 03 C, ( AX FS:[EBX] MOV ) NEXT END-CODE
CODE @-32 ( ad. -- d ) EBX POP        \ Von 32-Bit-Adresse ad. 32-Wert d holen
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL )
     OPSIZE: FS: ADRSIZE: 8B C, 03 C, ( FS:[EBX] EAX MOV  )
     OPSIZE: 0C1 C, 0C0 C, 10 C, ( 10 # EAX ROL ) EAX PUSH NEXT END-CODE
CODE !-32 ( d ad. -- ) EBX POP       \ 32-Bit-Wert d nach 32-Bit-Adresse ad. speichern
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL ) EAX POP
     OPSIZE: 0C1 C, 0C0 C, 10 C, ( 10 # EAX ROL )
     OPSIZE: FS: ADRSIZE: 89 C, 03 C, ( EAX FS:[EBX] MOV ) NEXT END-CODE

CR .( Speicherbereich DUMPen )

CODE 2TUCK ( d1 d2 -- d2 d1 d2 ) EAX POP ECX POP EAX PUSH ECX PUSH EAX PUSH NEXT END-CODE

: D0<= ( d -- fl ) 0. D> NOT ;

VARIABLE ASCFILT ASCFILT ON           \ Bei DUMP nur ASCII-Zeichen ausgeben ?

VARIABLE FS-DUMP-32 0 FS-DUMP-32      \ Segment-Wert für DUMP-32

: ?ASCII ( n1 -- n2 ) ASCFILT @       \ Nicht-ASCII-Zeichen --> Punkt ?
     IF   DUP 20 7E BETWEEN NOT ( nicht ASCII ? ) IF DROP 2E THEN ( dann Punkt )
     ELSE DUP 7 = ( Bell ) OVER 8 = OR ( Del ) OVER 0A = OR ( LF ) OVER 0D = OR ( CR )
          OVER 1B = OR ( ESC ) OVER 0FF = OR IF DROP BL THEN ( dann Blank )
     THEN ;
```

```
: DUMP^ ." 0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F  0123456789ABCDEF" ;

: DUMP-32 ( ad. len. -- ) \ Stop: SPACE Weiter: SPACE Exit: CR CR \ Mehr: + Nochmal zurück: -
   CR >R >R >R >R FS@ FS-DUMP-32 @ FS! BASE @ HEX R> R> R> R> 2OVER 10 MU/MOD 2DROP >R
   2SWAP R@ 0 D- R> DUP 3 * DUP 17 > IF 1+ THEN 0A + (CURSOR) NIP 0B SWAP AT DUMP^
   (CURSOR) NIP AT BOLD ." \/" 3C + (CURSOR) NIP AT ." V" ATTOFF CR
   BEGIN
     BEGIN 2DUP <# # # # # # # # # #> TYPE SPACE SPACE \ Adresse anzeigen
        8 0 DO 2DUP C@-32 0 <# # # #> TYPE SPACE 1. D+ LOOP SPACE
        8 0 DO 2DUP C@-32 0 <# # # #> TYPE SPACE 1. D+ LOOP SPACE  10. D-
       10 0 DO 2DUP C@-32           ?ASCII  EMIT 1. D+ LOOP CR
       2SWAP 10. D- 2TUCK D0<= STOP? OR                \ Ende oder Abbruch ?
     UNTIL
     (CURSOR) DUP 18 = IF 0 0 AT 0B SPACES DUMP^ THEN AT
     BOLD (CURSOR) -1 0 (FRAME) ." Weiter: +   Zurück: -   Exit: RET" ATTOFF AT KEY DUP
       ASCII + = IF  DROP 2SWAP 2DROP 100. 2SWAP                        FALSE ELSE
       ASCII - = IF 2SWAP 2DROP 100. 2SWAP 200. D- DARK 0B 0 AT DUMP^ CR FALSE ELSE
       2DROP 2DROP BASE ! FS! TRUE  THEN  THEN
   UNTIL 0C 0D (FRAME) CR ;

CR .( 32-Bit-Wert suchen )

CODE (FIND-VALUE) ( ad1. ad2. d -- ad3. )
     ECX POP ( ECX =  d   = Suchwert )   OPSIZE: 0C1 C, 0C1 C, 10 C, ( 10 # ECX ROL )
     EDX POP ( EDX = ad2. = Suchende )   OPSIZE: 0C1 C, 0C2 C, 10 C, ( 10 # EDX ROL )
     EBX POP ( EBX = ad1. = Suchanfang ) OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL )
     BEGIN OPSIZE: BX DX CMP ( EBX EDX CMP ) 74 C, 0E C, ( Raus, wenn ad2. erreicht )
       OPSIZE: FS: ADRSIZE: 8B C, 03 C, ( FS:[EBX] EAX MOV -- 32-RAM-Wert )
       OPSIZE: AX CX CMP                ( EAX ECX CMP )                    0<>
     WHILE EBX INC ( EBX = EBX+1 )
     REPEAT
     OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL )   EBX PUSH ( ad3. auf Stack legen )
     NEXT END-CODE                      ( ad2. = Adresse, an der der Suchwert liegt )

2VARIABLE AD1 ( Suchanfang )  2VARIABLE AD2 ( Suchende )   2VARIABLE VAL ( Suchwert )

: FIND-VALUE ( ad1. ad2. d -- )       \ Von ad1. bis ad2. nach d suchen
    CR DEPTH 6 < ABORT" Zu wenig Eingaben" VAL 2! AD2 2! AD1 2!
    BEGIN AD1 2@ AD2 2@ VAL 2@ (FIND-VALUE) 2DUP AD1 2! AD2 2@ D=
     IF   AD1 2@ @-32 VAL 2@ D=
       IF   AD1 2@ D. ELSE CR ." Wert bis " AD2 2@ D. ." nicht oder nicht mehr gefunden "
       THEN
     FALSE
     ELSE AD1 2@ D. TRUE
     THEN
    WHILE AD1 2@ 1. D+ AD1 2!
    REPEAT ;

CR .( Primzahlen )

: UD> ( ud1 ud2 -- fl )  DUP 3 PICK = IF   ROT 2DROP U>
                                      ELSE SWAP DROP ROT DROP U>
                                      THEN ;

: [SQR] ( ud -- u ) HERE 2! 0 PAD !   \ Größte ganze Zahl < oder = sqr{ud}
     1 2 4 8 10 20 40 80 100 200 400 800 1000 2000 4000 8000  \ Zweierpotenzen
     10 0  DO DUP PAD @ + DUP UM* HERE 2@ UD>  IF DROP 0 THEN  PAD +! LOOP  PAD @ ;

CODE FILL-32 ( ad. len. d -- )        \ len. 32-Worte ab ad. mit d füllen.
   EAX POP ( EAX = d = Füllwert, 32-Bit ) OPSIZE: 0C1 C, 0C0 C, 10 C, ( 10 # EAX ROL )
   ECX POP ( ECX = len. = Länge, 32-Bit ) OPSIZE: 0C1 C, 0C1 C, 10 C, ( 10 # ECX ROL )
   EBX POP ( EBX = ad. = Anfang, 32 Bit ) OPSIZE: 0C1 C, 0C3 C, 10 C, ( 10 # EBX ROL )
     BEGIN ECX DEC  0>=                \ Zähler-1 >= 0 ?
     WHILE OPSIZE: FS: ADRSIZE: 89 C, 03 C, ( EAX FS:[EBX] MOV ) 4 # EBX ADD
     REPEAT NEXT END-CODE

CODE (SIEVE) ( ad1. ad2. sqr{ud} -- ) \ ad1. = Anfang , ad2. = Ende
     EDI POP ( EDI = sqr{ud} ) OPSIZE: 0C1 C, 0C7 C, 10 C,  ( 10 # EDI ROL )
     EDX POP ( EDX = ad2. )    OPSIZE: 0C1 C, 0C2 C, 10 C,  ( 10 # EDX ROL )
     OPSIZE: B9 C, 2 , 0 ,                                  ( 2. # ECX MOV )
     EBX POP ( EBX = ad1. )    OPSIZE: 0C1 C, 0C3 C, 10 C,  ( 10 # EBX ROL )
```

```
     EBX PUSH                                          ( EBX = ad1. SWAP )
     BEGIN EBX POP  EBX PUSH ( EBX = ad1. ) OPSIZE: 03 C, 0D9 C, ( ECX EBX ADD )
       FS: ADRSIZE: 8B C, 03 C, ( FS:[EBX] AX MOV , 16-RAM-Wert ) 0 # AL CMP 0<>
       IF  BEGIN OPSIZE: DX BX CMP <  ( EDX EBX CMP )
           WHILE OPSIZE: 03 C, 0D9 C, ( ECX EBX ADD )   AX AX XOR ( AX = 0 )
                 FS: ADRSIZE: 88 C, 03 C,                ( AL FS:[EBX] MOV )
           REPEAT
       THEN
       ECX INC ( ECX = ECX+1 )  OPSIZE: DI CX CMP  <         ( EDI ECX CMP )
     WHILE
     REPEAT
     EBX POP ( Stack säubern ) NEXT END-CODE

2VARIABLE SIEVE-BEG ( Siebanfang )          2VARIABLE SIEVE-END ( Siebende )

: RAMTEST ( ad. -- ) 2DUP @-32 2SWAP  \ Obere RAM-Grenze eingehalten ?
     5A5A5A5A. 2OVER !-32 2DUP @-32 5A5A5A5A. D= NOT ABORT" Zu wenig RAM ?" !-32 ;

: SIEVE ( fs ad1. ad2. -- )           \ ad1. = Anfang , ad2. = Ende
    SIEVE-END 2! SIEVE-BEG 2! FS!  SIEVE-BEG 2@ SIEVE-END 2@
    2DUP 4. D- RAMTEST ( Genügend Platz ? ) 2OVER 2OVER 2OVER D- D2/ D2/ ( ad1. len. )
    01010101. FILL-32 ( Gesamtes Sieb mit 1 belegen ) 2DUP [SQR] S>D ( [sqr{ad2.}], 32 Bit )
     (SIEVE) ( Primtabelle jetzt fast fertig ) 0 SIEVE-BEG 2@ CC!-32 ; ( 0 und 1 nicht prim )

: PRIMES^ ." 0 1 2 3 4 5 6 7 8 9  0 1 2 3 4 5 6 7 8 9  0 1 2 3 4 5 6 7 8 9" ;

: PRIMES ( beg. len. -- )  \ Stop: SPACE Weiter: SPACE Exit: CR CR Mehr: + Nochmal zurück: -
     CR >R >R BASE @ -ROT DECIMAL R> R> 2OVER 1E MU/MOD 2DROP >R 2SWAP R@ 0 D- R> 2 *
     DUP 12 > IF 1+ THEN DUP 27 > IF 1+ THEN 0C +
     (CURSOR) NIP 0C SWAP AT PRIMES^ (CURSOR) NIP AT BOLD ." V" ATTOFF CR
     BEGIN 2DUP D0<= IF 2DROP 0. THEN \ Rückwärtsblättern nicht unter 0
       BEGIN (CURSOR) 4C 0 DO BL EMIT LOOP AT
        2DUP <# # # # # # # # # # # #> TYPE SPACE SPACE SIEVE-BEG 2@ D+
        0A 0 DO 2DUP SIEVE-END 2@ D< IF 2DUP C@-32 0 <# # #> TYPE SPACE THEN 1. D+ LOOP SPACE
        0A 0 DO 2DUP SIEVE-END 2@ D< IF 2DUP C@-32 0 <# # #> TYPE SPACE THEN 1. D+ LOOP SPACE
        0A 0 DO 2DUP SIEVE-END 2@ D< IF 2DUP C@-32 0 <# # #> TYPE SPACE THEN 1. D+ LOOP SPACE
        CR SIEVE-BEG 2@ D- 2SWAP 1E. D- 2TUCK D0<= STOP? OR       \ Ende oder Abbruch ?
       UNTIL
       (CURSOR) DUP 18 = IF 0 0 AT 0C SPACES PRIMES^ THEN AT BOLD (CURSOR) -1 0 (FRAME)
       2OVER SIEVE-BEG 2@ D+ SIEVE-END 2@ D<
       IF   0C SPACES ." Weiter: +                 Zurück: -                  Exit: RET"
       ELSE 0C SPACES ." Ende der Primzahlentabelle!        Zurück: -        Exit: RET"
       THEN
       ATTOFF AT KEY DUP
       ASCII + = IF  DROP 2SWAP 2DROP 258. 2SWAP                            FALSE ELSE
       ASCII - = IF  2SWAP 2DROP 258. 2SWAP 4B0. D- DARK 0C 0 AT PRIMES^ CR FALSE ELSE
       2DROP 2DROP BASE ! TRUE  THEN  THEN
     UNTIL 0C 0D (FRAME) CR ;

CR .( 32-Bit-System einschalten )

SIZEUP  0 FS!                         \ Segmentgrenzen hochschalten, Vorgabe für FS = 0

: XXX ( -- ) FREE-A20 ( A20 freischalten ) CR ." Adreßleitung A20" A20?  \ Wirklich frei ?
     IF ."  jetzt frei." ELSE ."  noch blockiert." THEN ; XXX FORGET XXX \ Ausführen und weg
```

Date: Thu, 26 Feb 1998 11:05:10 +0100
From: Fred Behringer
<**behringe@mathematik.tu-muenchen.de**>

...
der Alan Wenham hat mich auf einen Fehler in meinem Zen-Floating-Point-Artikel aus der VD 98/1 aufmerksam gemacht. Könnten wir die folgende Korrekturmeldung in die VD 98/2 setzen?

...aber selbstverständlich ! Dafür sind wir doch da ;-)
die Redaktion

Korrektur zu Zen Floating Point aus VD 98/1

In meinem Gleitkomma-Artikel aus der VD 98/1 muß es unter FPATAN heißen:

D9 C, F3 C, , und nicht *D9 C, F2 C,* . Auf diesen Fehler hat mich Alan Wenham von der FIG UK aufmerksam gemacht. Alan hat in der Forthwrite über den Zen-Floating-Point-Artikel von C.H. Ting geschrieben und beschäftigt sich weiter mit diesem Thema, speziell in Hinsicht auf F83.

Fred Behringer

Projekt: Ein “gutes” Forth

Was ist ein wirklich „gutes“ Forth ?

Diese Frage haben sich nicht nur die Implementierer der bisher verfügbaren Forth-Systeme gestellt, die als Antwort jeweils ihre eigene Version eines „guten“ Forth anbieten. Die Frage nach einem guten oder einfach nur „brauchbaren“ Forth stellen sich auch diejenigen Forther, die ein für Ihre Aufgaben oder Interessen geeignetes Werkzeug aussuchen sollen oder müssen. Die unterschiedlichen Aufgaben, die das gesuchte Werkzeug bewältigen helfen soll, bedingen in der Regel ebenso unterschiedliche Anforderungen.

„Natürlich“ muß ein gutes Forth möglichst schnell sein, sowohl während der Compile-Zeit als auch zur Laufzeit. Selbstverständlich erwartet der Forther, das das Werkzeug seiner Wahl eigene Werkzeuge wie zum Beispiel einen Debugger mitbringt. Und der Editor sollte unbedingt mit sequentiellen Dateien arbeiten, wegen der Bequemlichkeit. Diese Forderung wird aber schon nicht mehr von allen Forthern unterstützt. Blockorientierte Editoren sind längst nicht passe, wie unter anderem das recht moderne HOLON zeigt.

Die Meinungen über ein „gutes“ Forth gehen spätestens nach diesen ersten Punkten bei den Forthern weit auseinander. Klein soll das System sein, am besten auf einer Diskette Platz finden - samt aller dazugehörigen Werkzeuge und Quellen. Die Quellen des Systems selbst müssen einschließlich Metacompiler ebenfalls auf der gleichen Diskette Platz finden. Vor allem die Forther der „Windows-Fraktion“ stört es allerdings wenig, wenn das entpackte WIN32FOR 5,2 MByte auf der Festplatte belegt. Ihr Argument, daß zum Beispiel Borland’s C++
mehr als 145 MByte Plattenspeicher für sich beansprucht, zeigt deutlich genug, daß WIN32FOR unter den modernen Entwicklungssystemen relativ genügsam ist.

Die „Nicht Windows-Fraktion“ setzt häufig auf Systeme, die auf DOS basieren. Sie hat gute Gründe dafür, die selbstverständlich von den „Windowsfans“ nicht akzeptiert werden. Der erschwerte Zugriff auf den (vor einigen Jahren) teuer erkauften Arbeitsspeicher war aber bisher nur Michael Schröder und Fred Behringer Herausforderung genug, um zu zeigen, daß daran die Entscheidung gegen DOS eigentlich nicht festgemacht werden sollte.

Die „Dos-Fraktion“ steht allerdings nicht allein gegen die „ Windowsfans“. Die „Unixer“ (meist Linuxer) können nichts akzeptieren, was nicht unter diesem System entwickelt wurde. Und die wenigen kommerziellen Entwickler für „Embedded Controller“ stellen Anforderungen an ‘ihre’ Forthe, die kein (?) Windowsforth erfüllen kann. Ihnen sind graphische Oberflächen ohnehin ein Greuel das Resourcen frißt.

Selbst an der (meist ohnehin nicht vorhandenen) Dokumentation scheiden sich die Geister. Den ‘versierten’ Forthern macht es eher Spaß, sich in einem unübersichtlichen Haufen ungeordneter Quellen mit einem mehr schlecht als recht funktionierenden Hypertextsystem zu tummeln, um herauszufinden, wie denn der Implentierer die Definition <WasIstDennDas> „gemeint“ hat. Der Einsteiger vermag diesen „Spaß“ nicht nachzuvollziehen. 4.191 Words in WIN32FOR sind eher dazu angetan, ihn ein für allemal abzuschrecken oder zumindest für lange Zeit heillos zu verwirren. Um Informationen aus dieser Fülle zu ziehen, bedarf es durchaus ein wenig Übung. „Much is beautiful“ sagt dagegen der Optionenjäger, den es nicht im geringsten stört, daß ein solches System einen größeren Wortschatz hat, als er selbst im Alltag nutzt.

Durch bloßes Gegenüberstellen von vermeintlichen und/oder echten Vor- und Nachteilen läßt sich aus der Menge der verfügbaren Forthe kein wirklich gutes System selektieren ! Zumindest wird diese Selektion zu keinem Konsens führen, den eine breitere Mehrheit ‘der Forther’ als ‘ihr’ System akzeptieren würde. Tatsächlich ist die Entscheidung für ein System eben doch in großem Maße von Aufgaben abhängig - und von der Psycholgie.

Nimmt man diejenigen Implementierer, die von **ihren** Systemen und deren Verkauf leben wollen oder müssen, aus den nachfolgenden Überlegungen heraus, dann bleiben immer noch reichlich Menschen übrig, die ganz selbstverständlich jeweils das Forth für „das Beste“ halten, das sie selbst implementiert oder nach vielen Jahren der Arbeit ‘verstanden’ haben. Um bewertenden Diskussionen vorzubeugen, sei an dieser Stelle gesagt, daß der Autor das nicht nur für verständlich hält, sondern durchaus auch für legitim. Ihm geht es mit den über viele Jahre hinweg gewonnenen Erfahrungen zum ZF nicht anders. Selbstverständlich ist der Mensch bemüht, die meist doch recht große Arbeits- und/oder Lernleistung zu schützen - und sei es dadurch, daß er nach ‘Haaren in der Suppe’ anderer Systeme sucht. Das gelingt dem Forther auch recht gut. 2 CPU Takte, die im NEXT zuviel ‘verbraten’ werden, ein S“ das nicht ‘Statesmart’ definiert wurde, fehlende Hypertextfähigkeit und andere Dinge sind schnell zusammengetragen. DAS hat der Forther gelernt. Und - ein Schelm wer Böses dabei denkt - welcher Grund zur Ablehnung könnte größeres Gewicht haben, als die Tatsache, daß ein System nicht mit dem eigenen Namen verknüpft ist ?

Diese menschliche Komponente der Entscheidungsfindung könnte aber, bei aller Unvollkommenheit, auch Vorteile mit sich bringen. Zwar trägt sie nicht dazu bei, ein „gutes“ Forth zu benennen, aber die Summe aller gesammelten ‘Suppenhaare’ zeigt in jedem Fall, was ein gutes Forth sich nicht leisten darf ! Daraus sollte sich etwas machen lassen !

In der Vergangenheit hat die Unzufriedenheit mit den Produktionen einzelner Implementierer immer wieder zu Modifikationen ihrer Systeme durch ‘Dritte’ geführt. Das ZF ist über viele Jahre hinweg in Moers ausgesprochen liebevoll gepflegt und an die besonderen Bedürfnisse der Moerser Forthgruppe angepaßt worden. Dies war vor allem deshalb möglich, weil ZF’s Autor sein Werk in die Public Domain gegeben und alle Quellen seines Systems allen Interessierten zur freien Verfügung gestellt hat. Andere PD Forthe sind bis zu einer semikommerziellen Vermarktung ‘gereift’, und haben immens an Funktionalität und Umfang ‘gewonnen’. Michael Schröder

hat dem ZF eine Schnittstelle zu dem DPMI von Windows und OS/2 spendiert, die es diesem Forth ermöglicht, den gesamten, im Rechner verfügbaren, realen und virtuellen Arbeitsspeicher zu nutzen. Fred Behringer zeigt in dieser Ausgabe der VD am Beispiel von TurboForth, wie sich auch im REAL-Mode unter DOS der Arbeitsspeicher 32-Bit breit linear adressieren läßt, ohne ein DPMI nutzen zu müssen.

Andere Forther haben sich in der jüngsten Vergangenheit mit ganz anderen Wegen und Projekten befaßt - und tun es teilweise zur Zeit noch. Friedhold Birnkammerer (in 1997 verstorben) hat an einem „Betriebssystem auf der Basis von Forth“ gearbeitet. Dirk Zoller hat die internationale Forthgemeinde mit seinem PFE (Portable Forth Environment) überrascht, das sich unter nahezu jedem Betriebssystem nutzen läßt. Wolf Wejgaard ist mit seinem HOLON einen ganz eigenen Weg zu einem „interaktiven Targetsystem“ gegangen, über den seit einiger Zeit in der VD berichtet wird. Die Schnittstelle zum FASTGRAF ©, die der Autor dem ZF und dem HOLON geschrieben hat, hat großen Anklang bei den Nutzern von ZF und F-PC gefunden. Bernd Paysan hat auf der Jahrestagung der Forthgesellschaft 1997 in Ludwigshafen die Anwesenden mit einer Vorführung seines „Visual gForth“ überrascht. Und last but not least experimentiert der Autor - aufbauend auf die Erfahrungen mit dem WIN32FOR - aktuell mit einem neu gestalteten Lademodul (C-Kompilat) und einem völlig neuen Kernel zu einem schnelleren, an die Fähigkeiten der Pentium CPU angepassten „WIN32FOR“.

Mit anderen Worten: „die Szene lebt“. An den verschiedensten Orten experimentieren und basteln Forther aus den verschiedensten Motivationen heraus an Modifikationen und/ oder neuen Forthen, die ihren jeweiligen Vorstellungen mehr entsprechen als das, was sie von den bekannten Implementierern angeboten bekommen.
Dabei werden einmal mehr weitere Systeme und Subsysteme herauskommen, die zum einen deshalb keine breitere Verwendungsbasis finden können, weil die Zahl der potentiellen Nutzer nicht allzu groß ist, und zum anderen gewiß sein können, daß die oben beschriebenen, psycholgischen Aspekte eine breitere Akzeptanz zuverlässig verhindern werden. Es scheint keinen Ausweg zu geben, aus der spöttischen Kritik an Forth, daß 5 Forther = 10 Implementierer = 20 (Sub) Systeme bedeutet.

Und doch sollte es möglich sein, die verschiedenen Aktivitäten und Interessen zu bündeln. Das Wort Kompromiß, das allenthalben als kleinster, gemeinsamer Nenner aufgefaßt wird, kann eine ausgesprochen positive Bedeutung bekommen, wenn es den Forthern gelingt, alle ‘Suppenhaare’ in den nicht präferierten Systemen zusammenzutragen und eine Liste aller technischen Eigenschaften vorzugeben, die ein von einem bereiten Konsens getragenes Forth haben muß ! Entsprechend dem Vorbild der GNU Foundation sollte es auch den Forthern (zumindest in der BRD) möglich sein, sich via DFÜ (und anderen Medien) auszutauschen und all das zusammen zu tragen, was in ihren Augen ein „gutes“ Forth ausmacht. Hier wären ALLE Forther gefragt.

Mit diesem Wissen versehen, sollte es weiterhin möglich sein, in einem einzigen, großen Projekt DAS Forth zu ‘bauen’, das uns im Geiste vorschwebt. Jeder der vielen Spezialisten (im positivsten Sinne des Wortes) könnte zu diesem Projekt das beitragen, was seinen Erfahrungen und Kenntnissen und/oder seinen Neigungen entspricht. Zu tun gäbe es ‘für alle’ reichlich. Metacompiler und Compiler, Assembler, Userschnittstelle (äußerer Interpreter), Tools (View, Debug usw.), Dokumentation... die Arbeit eines solchen Projektes wäre nicht von einer einzelnen Person zu leisten.

Viel braucht es nicht, wenn ein solches Projekt angegangen werden sollte. Einen Projektmanager müßte sich die Forthgesellschaft ‘ausgucken’, jemanden der mit Langzeitprojekten Erfahrung hat, der selbst ein versierter Forther ist, und - daran mag es scheitern - der es akzeptieren könnte, daß die Verwaltungsarbeit des Projektes ihn so in Anspruch nimmt, daß er kaum oder gar nicht dazu kommt, selbst am technischen Part mitzuwirken. Dann braucht es ‘nur’ noch Projektmitglieder, die sich VERBINDLICH verpflichten, über einen noch zu spezifizierenden Zeitraum oder für eine bestimmte Arbeitsleistung an der Aufgabe mitzuwirken. Das Ergebnis könnte ein Forth sein, das der Forthgesellschaft „gehört“, in das der gebündelte Sachverstand der Mitglieder unseres Vereins geflossen ist und das vielleicht die weltweit breiteste Nutzerbasis fände.

Ein lesenswerter Auszug aus einer Mail von **Claus Vogt** in **DE\COMP\LANG\FORTH**

Neulich habe ich ein Excelbasic-Listing gelesen. Excel übersetzte es auf deutsch und ich lag unterm Tisch vor Lachen. Das können wir besser. Etwa so:

```
Handgriff reinkomm
Handgriff rauswerf

: rübertu ( hre hra -- )
      rüber handgrifföffne
      schwappe handgrifföffne
      FANGAN
            rüber hz@ 2 picke rotiere hz!
            rüber handgriffAlle
      BIS
      NOCHMAL ;

reinkomm rauswerf rübertu
```

Vielleicht wäre statt Handgriff besser und kürzer: GRKK für Gefädelte Reihe KontrollKlotz ? Aber das können dann die Deutschlandvertretungen von LWG (Labor Winzig Gesellchaft) und WW (Winzig Weich) unter sich ausmachen.
Jedenfalls eine witzige Idee, die unbedingt in den KneipenIndex gehört :-)
Ciao,
- Claus -

Anmerkung der Redaktion: Der von Claus angesprochene ‘Kneipenindex’ ist eine ‘Mail-Thread’, die sich in den vergangenen Monaten in der oben bezeichneten Newsgroup entwickelt hat.

fep

Gehaltvolles

zusammengestellt und übertragen
von ***Fred Behringer***

VIJGEBLAADJE der HCC Forth-gebruikers-groep, Niederlande

Nr. 8, Februar 1998

Vijgeblaadje 8
Albert Nijhof

Der Autor beschreibt und kommentiert einen großen Teil seines Forth-Programms, mit dessen Hilfe er die in der Überschrift "VIJGEBLAADJE" verwendeten Blockbuchstaben erzeugt und auf einem HP 520 ausdruckt. Das vollständige Programm liegt als VIJG8.F auf der BBS der Forth-gebruikersgroep.

Het Forth-BBS
Willem Ouwerkerk

Wie gelangt man in das (rund um die Uhr erreichbare) BBS-Bulletinboard der Forth-gg in Arnhem unter 026 442 2164 ? Der Autor gibt ausführliche Antworten. Schwerpunkte: 8051-ANS-Forth und CHForth.

Forth-gg produkten

Und hier noch aus einer Liste von 13 aufgezählten Forth-Erzeugnissen der Forth-gebruikersgroep ein paar von mir willkürlich ausgewählte. Bestellungen bei Willem Ouwerkerk, Boulevard Heuvelink 126, 6828 KW Arnhem. Tel: 026 443 1305 .

- **CHForth** ... 60,-- Gulden. Ein 16-Bit-Multisegment-ANS-Forth für den PC/AT mit gedrucktem Handbuch in englischer Sprache.
- **8051-ANS-Forth** ... 90,-- Gulden. EPROM mit einem 16-Bit-ANS-Forth für die B+-Karte der Sektion Den Haag mit einem 80C535 und einem ausführlichen (englischen) Handbuch mit Beispielen. Jetzt auch für den 80C32 und den 80C552 lieferbar.
- **B+-Karte** ... 130,-- Gulden. Ohne B+-EPROM.
- **ByteForth** ... 100,-- Gulden. Entwicklungsumgebung für den Microcontroller AT89Cx051. Enthält: AT89C2051 (2x), Programmiergerät und Kabel, gedruckte Karte, ByteForth auf Disk und Handbuch. Benötigt B+ mit Forth.
- **AT51-Karte** ... 15,-- Gulden. Lose Platine zur Verwendung mit ByteForth.
- **AT89C2051-CPU ... 15,-- Gulden.**Controller für die AT51-Platine.
- **JinForth** ... 25,-- Gulden. 32-Bit-ANS-Forth für den Atari-ST auf Disk.
- **C64-Prä-ANS-Forth** ... 15,-- Gulden. 16-Bit-beinahe-ANS-Forth auf Disk.

FORTHWRITE der FIGUK, Großbritannien

Nr. 92, August 1997

1 Editorial
Chris Jakeman

Mit einigen Monaten Verzögerung ist FORTHWRITE wieder da. 36 DIN-A-4-Seiten auf DIN-A-5 verkleinert. Nachdem Gil Filbey, als 1-Mann-Team, die ersten 91 Ausgaben gemanaget hat, teilen sich jetzt Doug Neale (Druck und Vertrieb) und Chris Jakeman (Redaktion) die Aufgaben. Es wird sich "einiges ändern", "anderes bleibt". Die Mitgliedschaft in der FIGUK kostet nur 10 englische Pfund pro Jahr und ist damit weitaus billiger als in Amerika oder bei uns.

2 The End of an Era
Chris Hainsworth

Chairman (Vorsitzender) Chris Hainsworth stellt mit Erstaunen fest, daß eine Ausgabe von FORTHWRITE erscheinen konnte, die nicht von Gil Filbey herausgegeben, verlegt und versandt wurde. Man dankt Gil Filbey, der seit den ersten Anfängen im April 1981 die Fäden in der Hand hielt.

3 Forth News
Chris Jakeman

Marty McGowan (AT&T): Diskussions-Gruppe über die Verwendung von C/C++-Bibliotheken von Forth aus. Siehe http://www2.cybernex.net/ ~mcgowan . Eric Marsden, Universität Toulouse, Frankreich, errichtet eine Bibliothek von Public-Domain-ANS-Forth-Quellen. - Den Inhalt von ftp.forth.org gibt es jetzt auf CD. Frank Sergeant (Pygmy Forth) verkündet mit Stolz, daß er seit Mai 1997 Master of Science in Computer Science ist. Auch wir gratulieren! Es werden desweiteren 8 Forth-Systeme mit voller Internet-Bezugsadresse angegeben, darunter EForth für Linux 2.0 von Marcel Hendrix.

6 Building a New Inner Interprter Interactively
Jack Brien <jackbrien@zetnet.co.uk>

Im Forthwrite-Heft 90 hat der Autor einen neuen äußeren Interpreter für F83 vorgestellt. Heute geht es darum, aus einem F83-System heraus eine direktgefädelte Z80-Implementation hochzuziehen. Mit Cross-Assembler usw. Der Autor orientiert sich an CamelForth (ANSI-Standard-Z80) von Brad Rodriguez. Siehe ftp://ftp.forth.org/pub/Forth/Camel/cam09-10.zip .

10 Deutsche Forth-Gesellschaft
Chris Jakeman

Chris berichtet darüber, daß wir seit einiger Zeit in unserer VD über Forthwrite berichten. "Als gute Europäer sollten wir gleiches tun. Ist jemand sattelfest in Deutsch, um die Vierte Dimension für Forthwrite zu rezenzieren?", fragt Chris. Das ist wirklich gut! So habe ich mir das vorgestellt. Prompt hat sich auch jemand gefunden: Seit Januar 1998 rezenziert Alan Wenham unsere VD für Forthwrite. - In Amerika ist unser Mittelsmann Henry Vinerts, in England jetzt Alan Wenham mit Unterstützung von Chris Jakeman. Das nenne ich Zusammenarbeit! Wir brauchen ja gar nicht alle dieselbe Sprache zu sprechen. Warum denn auch? Es reicht ja, wenn

wir bereit sind, auf den jeweils anderen einzugehen und ihm zuzuhören. Die Sprache der anderen verstehen, können viele. Forthler schon allemal. Die Sprache der anderen dauernd aktiv sprechen zu müssen, ist dagegen unnatürlich und auch gar nicht so sehr nötig.- Vielleicht könnte man mal die Meinung anderer Mitglieder hören? Schließlich leben wir ja nicht nur von, für und mit Foth (?) Übrigens habe ich inzwischen mehrere E-Mails mit Dr. Wenham ausgetauscht: Er ist organischer Chemiker und war längere Zeit bei der Tochtergesellschaft von BP Chemicals in Köln tätig.

11 Forth for the Transputer

Vier Seiten Abdruck der Beschreibung meines Forth-Systems für den T800. Thanks a lot, Chris.

15 Reader Survey
Chris Jakeman

Zwei Seiten vorgefertigte Fragen an den p.p. Leser. Als Rückkopplung für Redaktion und Direktion gedacht. Könnten wir auch machen (?).

17 Welcome Disk - Bon Mots
Chris Jakeman

"Pet words" (Lieblingsworte) war der Ausdruck, der mir kürzlich in comp.lang.forth auffiel. Offensichtlich wird bei unseren englischen Freunden eine Sammlung solcher Worte für die "Begrüßungsdiskette" für Neuankömmlinge vorbereitet. Sie nennen sie "Bon Mots", mit, soviel ich weiß, der auch uns bekannten Nebenbedeutung. Beispiele: : OFF? @ 0= ; : ON? @ 0<> ; Könnte man unter "Bon Mots" nicht eine Gemeinschaftsaktion starten? Vielleicht auch die holländischen Freunde zum Mitwirken einladen? Und vor allen Dingen viele von uns, die keine Zeit für große Artikel haben, zu schnellen unkomplizierten Einsendungen animieren?

18 Pattern-Matching 2/3
Chris Jakeman

"Forth ist wie ein Chamäleon und läßt sich nur schwer beschreiben. Eine mögliche Beschreibung wäre: Forth ist eine Sprache, mit der man Sprachen schreiben kann"...
Auf 7 Seiten entwickelt Chris seine "Chamäleon-Auffassung" anhand von FoSM (Forth String Matcher), das von Gordon (Gordon Charlton?) stammt und speziell für Stringvergleiche geschaffen wurde. Als Anwendungen bespricht er ein Programmstück zur Bestimmung der Tageszeit und einen Taschenrechner mit Auswertung von Infix-Ausdrücken. Enthalten ist ein ausführlicher Internet-Hinweis auf Gray, einen ANS-Forth-Parser von Anton Ertl.

26 EuroForth ´97
Chris Jakeman

Vier Seiten Vorschau auf die Embedded-Communications-Tagung EuroForth ´97 in Oxford vom 26. bis 28. September 1997. Für mich interessant zu erfahren, daß Ulrich Hoffmann, Anton Ertl und Wolf Wejgaard im Programm-Komitee waren. Markus Dahm von der RWTH-Aachen ebenfalls. Aber Bernd Paysan hat in der VD 1/98 ja über das Drum und Dran der inzwischen schon wieder der Vergangenheit angehörenden Tagung ausführlich berichtet.

30 Forthwrite Index ´97
Chris Jakeman

Und wieder ist es Chris Jakeman, der sich da hervortut: Die Titel der 224 Artikel aus Forthwrite von 1990-1997. Sie sind auch auf der Homepage der FIG UK zu finden und sollen alle Jahre ergänzt und neu eingeordnet werden:
http://www.forth.org/fig/uk/homepage.html. Sollten wir das nicht auch machen? Ich greife willkürlich zwei Titel heraus: algorithms Pochin, David 94-10 First attempts at Fuzzy Logic tools Franin, Julio 95-02 MC51 Forth debugging

FORTHWRITE der FIGUK, Großbritannien

Nr. 93, November 1997

1 Editorial
Chris Jakeman

Christopher (Chris) Jakeman kündigt jemanden an, der ab der nächsten Ausgabe die "German Vierte Dimension" rezensieren wird. Wir wissen es schon: Es wird Alan Wenham sein. Jack Brien hat die Aufgabe des Web-Masters für die FIG-UK-Homepage übernommen.

2 Forth News
Chris Jakeman

Drei Seiten Neuigkeiten. Zwei bekannte Forthler haben promoviert: Brad Rodriguez und Simon Read. Glückliche Jugend von heute! Vor 30 Jahren konnte man mit Forth nicht promovieren. Ähnliche Ideen konnte man aber damals durchaus schon haben - und hatte sie auch. Bernd Paysan kündigt MINOS (mit Internet-Bezugsadresse) an. Zwei weitere "Graphiknachrichten". Demnächst wird eine ANS-erweiterte Ausgabe von "Starting Forth" herauskommen. Julian Noble (Autor von "Scientific Forth") hat ein ANS-Forth-Programm für einen FORTRAN-2-FORTH-Übersetzer veröffentlicht. Forth-Chips MC143150 und MC143120 von Motorola. 4 Millionen sind davon seit 1991 in Umlauf. 7 Forth-Systeme mit Internet-Bezugsquellen.

5 FIG UK Web Site
Jack Brien <jackbrien@zetnet.co.uk>

"Die Verantwortung liegt jetzt bei mir", sagt Jack. Links zu anderen Quellen zum Herunterladen: www.zetnet.users.co.uk/aborigine/forth.htm . Die Texte der Homepage liegen auch auf der Begrüßungsdiskette für Neumitglieder.

7 Multi-precision Stack Operators
Ed Hersom

Ed hat sich schon in den Fünfzigern mit Computern beschäftigt. Jetzt gehört er zur Gruppe der Ruheständler. Ed hat Frank Sergeants Pygmy in seinem HP-200LX-Palmtop installiert. "In Forth fällt es nicht schwer, ein bißchen Assembler beizumischen. CODE-Definitionen machen es einem leicht." Recht hat er! TASM kann dies nicht, MASM kann das nicht... Die Code-Definitionen in Forth können alles! Man geht einfach hin und macht es. Und dazu braucht man nicht einmal groß einen Assembler (siehe meinen Beitrag im vorliegenden Heft - der Rezensent).

10 Welcome Disk - Bon Mots

Gehaltvolles

Chris Jakeman

Wieder 8 nützliche kleine Worte, diesmal von Ed Hersom, Alan Wenham und John Payne. Diese Idee sollten wir wirklich aufgreifen. Keith Methews macht Chris Jakeman in der nächsten Ausgabe darauf aufmerksam, daß doch wohl "Bons Mots" geschrieben werden sollte.

12 Pattern-Matching 3/3
Chris Jakeman

Der letzte von drei Artikeln über Stringvergleiche, 7 Seiten. Hier ist, was der Autor sagt: Der letzte Teil dieser Reihe baut mit Hilfe von Gordon Charltons FoSM (Forth String Matcher) ein Dienstproramm zur Textsuche auf, genannt Rex. Bei dem zu durchsuchenden Text darf es sich um das Forth-Dictionary handeln, aber auch um einen String oder um eine komplett einzulesende Textdatei. Der Einfachheit halber ist die hier wiedergegebene Fassung des Programms für eine Suche ab Adresse 0 bis HERE eingerichtet.

19 From the ´Net - More on Square Roots
Chris Jakeman

Die Summe der ersten n ungeraden Ganzzahlen ist gleich dem Quadrat von n. Jack Brien gibt eine anschauliche Erklärung. Er führt einen weiteren Wurzelalgorithmus an, der aus Wil Badens ThisForth stammt. Es folgt ein algebraischer Beweis von Chris Jakeman.

21 From the ´Net - DEFER and IS
Ray Allwright

Sechs Seiten Internet-Diskussionen über das Für und Wider von DEFER und IS . .... DEFERierte Worte sind schlimmer als GOTOs..... I love ‘em ...usw....Der Rezensent: Ich bin baß erstaunt zu hören, daß es im ANS-Standard gar kein DEFER und IS mehr gibt. Und nach den hier abgedruckten Stimmen muß man sich ja schämen, DEFER und IS zu verwenden. Natürlich verwende ich sie, ausgiebig!

27 EuroForth ‘97 Conference
Paul E. Bennett

Das Tagungsprogramm kann im Internet eingesehen werden oder man fordere es als Kopie von der FIG-UK-Bibliothek (Chris oder Silvia) an. Beiträge von Nick Nelson, Wolf Wejgaard, Peter Knaggs, Howard Oakford, Stephen Pelc, Paul Bennett, Paul Frenger, Duncan Louttit and Jonathan Morrish. Große Aufmerksamkeit widmet der Reviewer (P.E. Bennett) der Vorführung des auf Linux aufsetzenden MINOS-Systems von Bernd Paysan: "Ad-hoc-Konstruktion eines einfachen Taschenrechners in 3 Minuten". Meinung des Rezensenten der Rezension: Geballte Forth-Kraft mit wissenschaftsähnlichem Flair. (Eingereichte Arbeiten durchlaufen in Zukunft das von der Wissenschaft her bekannte Gutachterverfahren: Zwei unabhängige und voneinander unabhängige Gutachter empfehlen die Annahme oder die Ablehnung). Hat da derjenige, der Forth nur so zum Vergnügen betreibt, eigentlich überhaupt eine Chance mitzuhalten? Von Anfängern ganz zu schweigen. Peter Knaggs macht folgende Ankündigung: Die elektronische Ausgabe des "Journal of Forth Applications and esearch" ist auf folgender Adresse zu finden: http://cis.paisley.ac.uk/forth/jfar/ Binnen kurzem wird es folgende Adresse sein: http://www.jfar.com .

32 Letters
Chris Jakeman

Ray Allright möchte gern wissen, ob ihm jemand den Begriff der Finite State Machine, des endlichen Zustandsautomaten, erklären kann. Graeme Dunbar verspricht das für demnächst. Alan Wenham kramt Tings Zen Floating Point aus Dr Dobbs Werkzeugkiste und fragt, ob ihm jemand die Gleitkommabefehle des 80486 für Forth schmackhaft machen kann. Wir werden in der nächsten Ausgabe von FW mehr davon hören. Aber auch in der vorliegenden VD wird einiges dazu gesagt.

FORTHWRITE der FIGUK, Großbritannien

Nr. 94, Januar 1998

1 Editorial
Chris Jakeman

Chris versucht, Mitglieder gleichen Interesses zusammenzubringen und reicht Telephonnummern und E-Mail-Adressen ohne Rückfragen von einem zum anderen weiter. Wenn einer das nicht will, muß er es sagen. (Könnten wir uns auch zu eigen machen.) Alan Wenham rezensiert ab jetzt unsere VD und Doug Neale schickt Kopien von Forthwrite auch an die holländischen und amerikanischen Freunde. Motto: Go international. Jack Brien arbeitet unter dem gleichen Vorzeichen mit Anton Ertl über Forth-Datenstrukturen zusammen. Forth kann fast so schnell wie optimiertes C gemacht werden (sagt Chris Jakeman).

2 Gil’s Gold Watch
Chris Hainsworth

Man dankt noch einmal Gil Filbey für seine langjährige aufopferungsvolle Arbeit als Herausgeber, Redakteur, Verleger und Versandbüro der Forthwrite. Frage, die Chris stellt: Wer kann Gils gerngelesene Tutorials weiterführen?

3 Forth News
Chris Jakeman

Forth-Bibliographie von Peter Knaggs (Forth-Bücher, Konferenzberichte, 573 Artikel aus JoFAR): www-cis.paisley.ac.uk/forth/bib/forth.zip . Pygmy für DOS und Kermit von Frank Sergeant: www.eskimo.com/~pygmy/pfkerm.zip und www.eskimo.com/~pygmy/pygmy15.zip . Homepage des Institute for Applied Forth Research: http://www.jfar.org . Julian Noble hat seinen FORTRAN-2-FORTH-Übersetzer verbessert. Bernd Paysan verbessert MINOS: www.informatik.tu-muenchen.de/~paysan/bigforth.html . Thomas Worthington verbessert Aztec Forth. "Genetix Turing Machine" von Bernard Hodson, ein neuartiger gefädelter Interpreter, älter als Forth und womöglich besser: Da scheint sich was anzubahnen, auch an Diskussionen.

5 Aztec - A Forth For Windows 95
Thomas Worthington

Auf 7 Seiten ein wirklich interessanter Bericht darüber, warum jemand ein 32-Bit-Forth für Windows 95 hochzieht, wenn es doch Win32Forth von Tom Zimmer schon gibt. Eine wahre Lebens- und

Leidensgeschichte. "Die Azteken haben das Rad nicht erfunden. Was sie also brauchten, war Forth." Soviel zum Namen . "Windows 95 ist großer Mist, da man keine Low-Level-Informationen bekommt. Ran kommt man trotzdem. Die Homepage von Intel ist empfehlenswert, die von Microsoft ist es nicht. Nach 12 Stunden Surfen erfährt man, daß man 1600 Dollar braucht, um ranzukommen. SDK und Borland-Produkte allgemein sind eigentlich nur für C-Programmierer genießbar"...

12 Welcome Disk - Bons Mots
Chris Jakeman

Wieder 5 schöne kleine Worte, diesmal von Alan Wenham und Dwight Elvey.

13 Vierte Dimension
Alan Wenham

Zwei Seiten sehr gute Rezension unserer VD. Ich gehe davon aus, daß die Seitennummer 13 kein allzu schlechtes Vorzeichen ist und daß die Rezension der VD jetzt zur festen Einrichtung wird. Die Geschichte (Clausens Erfindung?) vom DAU, dem dümmsten anzunehmenden User, mit seiner großen Anfrage an die Hotline auf der verzweifelten Suche nach der Eniki-Taste (CNTRL ist es nicht) wurde in englischer Version wiedergegeben und hat als "German Humour" großen Eindruck gemacht.

15 Not the AGM
Chris Jakeman

Die Finanzen sind in Ordnung. Hauptposten ist die Zeitschrift, Forthwrite. Der Beitrag bleibt bei 10 englischen Pfund pro Jahr. Dafür gibt es 6 FW-Hefte. "In den 80ern waren wir viele, dann ging die Mitgliederzahl zurück. Jetzt steigt sie wieder und wir sind 123. Alte FW-Hefte? Schreibt an Silvia Hainsworth."

17 Building Forth Structures
Jack Brien

Einleitung, die wohl vom Redakteur stammt: "Anton Ertl hat viel zu Forth beigetragen - Gforth, die RAFTS-Native-Compiler-Technik und eine OO-Erweiterung für ANS-Forth, die von Worten zur Bildung von Datenstrukturen Gebrauch macht. In Zusammenarbeit mit Anton hat Jack Brien über die Datenstrukturen geschrieben und ein interessantes Beispiel und einige eigene Variationen beigesteuert." Es folgen 5 Seiten Programmbeschreibungen. Enthalten ist die Bezugsadresse für Anton Ertls Paket. Fußnote: "Jack betreibt Forth seit 10 Jahren als Hobby."

22 Readers Survey
Chris Jakeman

Von 123 Mitgliedern haben 17 den Fragebogen ausgefüllt. Viele davon sind mehr als 10 Jahre bei Forth. Die Zahl der Ruheständler (mit viel Zeit) wächst. Viele Mitglieder haben E-Mail-Anschluß.

23 From the 'Net - Speed Demons
Chris Jakeman

Zehn Seiten aus dem Internet über die Frage, wie schnell Forth ist und wie schnell es gemacht werden kann. Tom Zimmers Antwort: "Windows bringt die meiste Zeit mit Systemaufrufen zu. Wenn man in Forth Geschwindigkeit braucht, kann man immer Assembler einbauen. Machen Sie das mal mit Visual Basic. Viele glauben fest daran, daß Forth schneller ist als alles andere. Ist es natürlich nicht. Aber kann es über Assembler-Einsprengsel gemacht werden. Ich verwende Forth, weil es mich produktiver macht. Will man Geschwindigkeit, so lasse man vor allen Dingen die Finger von Windows. Ganz egal, welches. Das klingt hart, ist aber begründet." Viele Geschwindigkeitstests verschiedener Systeme wurden von Anton Ertl unternommen. Fazit: "Forth war schon immer schneller als BASIC. Und die jüngsten Forths sind um ein Vielfaches schneller als VisualBASIC 4."

33 Letters
Chris Jakeman

Jeremy Fowell macht auf den Artikel von Professor C.H. Ting in FD 97/1 über Zen Floating Point für den 80486 aufmerksam. Nun ja, der ist ja auch wirklich gut. Ein Hammer, wie man heute zu sagen pflegt. Jeremy erwähnt sodann "Scientific Forth" von Julian Noble und die Forth Scientific Library auf http://www.taygeta.com . Alan Wenham gibt ebenfalls einige Hinweise zum Artikel von C.H. Ting. Ed Hersom liefert ein weiteres Bon Mot: RROLL . P.H. Tanner outet sich als "Anarchist". Es liebt Forth, weil er da machen kann, was er will. Wenn's nicht klappt, ist es eben seine eigene Schuld. (Bravo!) John Phillips sucht Hilfe für Forth auf dem Amiga.

39 Corrections
Chris Jakeman

Die genaue Homepage-Adresse der FIG UK ist:
http://www.users.zetnet.co.uk/aborigine/forth.htm

Presseschau c't 5/98, S.32
Java auf Silikon-Forth

Am Rande der 'Embedded' in Sindelfingen ist dem c't-Redakteur das erste Evaluationsboard für den PSC1000 aufgefallen. Immerhin 40 Prozent des Java-Byte-Codes kann dieser RISC-Prozessor in Hardware ausführen.

Der Hersteller Patriot Scientific (www.ptsc.com) und sein deutscher Distributor Ineltek (www.ineltek.com) planen ihrem 1000-DM-Kit mit Java und C noch in diesem März einen JIT-Compiler hinzufügen zu können.

Chuck Moore, "Vater einiger Forth-Prozessoren" hat am Design dieses Prozessores mitgewirkt, dessen "Muttersprache" leicht zu raten ist.

Zukunftsweisend "entstand ein Baustein, der tatsächlich die Operandenübergabe via Stack erwartet, bevor die Operation kommt". Daher ist nunmehr der Programmierer nur noch für die "geschickte Abfolge der Operationen" zuständig, weil er sich nicht mehr um "das Sichern von Zwischenergebnissen kümmern muß". Da freut sich der Programmierer und der (ea) von der c't und natürlich der ...

(clv) von der VD im Feb'98.

Verschiedenes

Wollen Sie Fred Behringers Vorschläge zur 32 Bit Adressierung unter DOS mit dem ZF ausprobieren ? Dann müssen folgende Änderungen vornehmen:

```
hex

\ ------- Korrekturen und Änderungen für ZF (M.B.) ----------
\ Korrekturen:
\ Zeile:       Variable FS-DUMP-32 0 FS-DUMP-32
\ ändern in: Variable FS-DUMP-32 0 FS-DUMP-32 !

\ Änderungen:
\ Folgende Worte überdefinieren:
 : Dsegment  ?cs: ;
 : attoff        >norm ;
 : (cursor)     #out @ #line @ ;
 : frame        2drop ;
 : stop?        key? dup
                IF 0<>  drop key 1b =
           THEN ;
```

Anmerkung: Das Beispiel im Text zu FIND-VALUE funktioniert so nicht. Wegen dem Hick-Hack mit High-Byte-Low-Byte-High-Word-Low-Word (Big-endian) muss man nach AA55.0000 suchen lassen. Gefunden wird dann korrekt: 000055AA .

Martin Bitter

Redaktionelles:

Hallo,
gibt's beim neuen Editorat der VD keine FAX Nr ??

Microcontroller Verleih Ecke

Auf der Jahrestagung 1997 in Ludwigshafen wurde dem Microcontroller-Verleih angeboten ein 68 HC 11 Board (oder ein Derivat) zu ueberlassen. Bisher hier aber noch nicht eingetroffen.

Weiter wurde ein ein weiteres RTX-2000 Board in noch unbekannter Ausstattung angeboten.

Weitere Spenden sind gerne willkommen.

Thomas Prinz

Hallo Thomas, und alle anderen faxwilligen Forther,

das FAX Gerät liegt ganz in der Nähe bereit. Eigentlich müßte es nur 'Irgendjemand' in Mülheim abholen. Bisher habe ich das 'irgendwie' einfach noch nicht hinbekommen. Sorry, aber das wird bis zur nächsten Ausgabe erledigt - versprochen

fep

Jörg Arndt & Christoph Haenel:

Pi
Algorithmen, Computer, Arithmetik

Springer-Verlag 1998
191 Seiten (deutsch)
mit CD-ROM mit C++-Quelltext
ISBN 3-540-63419-3
DM 78,--

Eine sehr locker geschriebene interessante Untersuchung in Pi, um Pi und um Pi herum. Mir war nie so richtig bewußt, wieviel man zu Pi sagen kann. Das Buch ist von zwei deutschen Autoren in gutem Deutsch geschrieben, keine Pidgin-Übersetzung, ganz anders als beispielsweise das eine Linux-Buch aus Berlin. Man finde spaßeshalber die Stelle heraus, an der die Autoren "brauchen" ohne "zu" gebrauchen! Diese beiden Autoren (die anderen nicht), die ein Buch so lesbar geschrieben haben, können sich das ohne weiteres gelegentlich erlauben. Das spricht nicht gegen sie. Das spricht allenfalls gegen die Formalismen der Sprache, in die sie ihre Gedanken zwängen müssen, wenn sie gleichzeitig die nicht gleichlaufenden Formalismen der anderen, allesbeherrschenden Sprache im Kopf haben. Das Buch behandelt ein an sich hoch-mathematisches Thema, die Berechnung von Pi auf immer mehr Stellen und die dazu dienlichen Methoden, kommt aber ohne das von der mathematischen Literatur her gewohnte Schema "Voraussetzung-Behauptung-Bewei" aus. So geht es also auch, wenn man ansonsten lebendig schreibt und die wesentlichen Beweisideen in den Text mit einfließen läßt. Fürs erste jedenfalls. Wer mehr will, findet genügend Literaturhinweise. Einer der beiden Autoren hatte selbst einmal, für kurze Zeit, den Weltrekord in der Berechnung von Pi inne. Die Suche nach Methoden und die Bekanntmachung von Ergebnissen scheint sich hier im Internet abgespielt zu haben. Da bahnt sich eine neue Art der wissenschaftlichen Kommunikation an. Das Buch enthält sehr viele WWW-Hinweise zum "Herunterladen". Man erfährt im Buch viel über geschichtliche Hintergründe. Beschrieben werden unter anderem der Tröpfelalgorithmus, mit dem sich ja Martin Bitter kürzlich in der VD hervorgetan hat, und die schnelle Fourier-Multiplikation mit der schnellen Fourier-Transformation. Das Buch ist nicht gerade billig. DM 78,-. Das sind etwa 40 Pfennige pro Seite. Manche Seite ist das nicht unbedingt wert. Andere dagegen, wie die mit Pi auf 2500 Stellen genau, dezimal und hexadezimal sehr. Rechnet man die CD mit den C++-Programmen im Quelltext dazu, dann stimmt der Preis wieder.

beh

Liebe Leser,

kennen Sie ebenfalls Bücher, rund um das Thema Computer herum, die Ihnen gefallen haben, die Sie empfehlen, oder auf die Sie einfach nur aufmerksam machen möchten ? Oder hat Sie ein Artikel in einer Fachzeitschrift besonders angesprochen ? Dann schreiben uns das bitte ! Die Leser der VD werden es Ihnen danken.

Das "Thema Computer" sollten Sie dabei getrost sehr weit gefaßt sehen.

fep

Gehaltvolles

zusammengestellt und übertragen
von Fred Behringer

Forth Dimensions der Forth Interest Group, USA

November/Dezember 1997

7 Forth in Control: Analog/Digital Converters
Ken Merk

Wenn Sie angekoppelte Gerätschaften über den Parallelport steuern, haben Sie sicher manchmal den Wunsch, die Geräte zu überwachen und nachzusehen, ob sie richtig funktionieren. Die hierzu benötigten Rückgabesignale können digitaler Art sein, so daß sie der Computer direkt lesen kann, sie können aber auch in analoger Form vorliegen, so daß sie erst in digitale Signale gewandelt werden müssen, bevor sie der Computer verstehen kann. Dieser Artikel liefert eine allgemeine Übersicht darüber, wie Forth in die Steuerung eingebracht werden kann, und geht in diesem Zusammenhang auch auf Hardwarefragen ein.

11 MicroFont: 4x5
Rob Chapman

Eingebettete Steuervorrichtungen (oder müssen wir im Deutschen unbedingt "embedded systems" sagen?) und ähnliche Kleinsysteme kommen nicht selten mit ganz kleinen Flüssigkristall- oder ähnlichen -Anzeigen daher. Beim Autor war es eins von den Ausmaßen 128x64. Solche Anzeigevorrichtungen haben zum Teil nur eine beschränkte Anzahl von Zeichen im Zeichensatz oder sind sonstwie für die beabsichtigte Anwendung nicht optimal eingerichtet. In diesem Artikel wird ein 4x5-Zeichensatz vorgestellt, der alle darstellbaren ASCII-Zeichen enthält und gut lesbar ist. Und der Forth-Code kann ganz leicht verändert werden, um die dargebotenen Zeichen zu erweitern oder sonstwie zu bearbeiten.

22 Misty Beach Forth: An Implementation in Java
Mark Roulo

Dieser Artikel liefert eine Übersicht über Misty Beach Forth, eine Forth-Implementation, die unter Java läuft. Auf den ersten Blick scheinen sich Forth und Java leicht aneinanderzufügen. Die virtuelle Java-Maschine hat jedoch einige Eigenheiten, die das Aneinanderfügen nicht ganz so einfach gestalten. Außerdem hat der Autor, ganz im Sinne der wohldefinierten Java-Semantik, ein paar zusätzliche Beschränkungen eingebaut, die die Misty-Beach-Forth-Implementation unnötig erschweren.

26 Factors Influencing the Use of Forth for Large Projects
Dr. Everett F. Carter, Jr.

Der Autor beschäftigt sich mit den Faktoren, die bei großen Projekten für oder gegen die Wahl von Forth sprechen. Häufig wird die Wahl einer bestimmten Sprache von Überlegungen der Geschäftsleitung, des Kunden oder des Betriebswesens diktiert und ist eine geschäftspolitische und keine technisch bedingte Entscheidung. Von Projekt zu Projekt gesehen, kann man an solchen Vorgaben meist wenig ändern. Auf längere Sicht lassen sich solche Entscheidungen aber durch Aufklärung beeinflussen und durch die Einsicht, daß es keinen technischen Grund gibt, die eine oder andere Sprache vorzuziehen. Der vorliegende Artikel beschäftigt sich hauptsächlich mit den technischen Randbedingungen, die bei großen Systemen wichtig werden.

Suchsel - Auflösung **Forts.: Seite 30**

Lösung des Rätsels aus der VD 1/98

Suchsel ist ein Hilfsmittel für einige Lehrer der GHS-Borth. Eine Hauptschule, an der nun im vierten Jahr behinderte und nicht behinderte Kinder gemeinsam unterrichtet werden.
Das Programm Suchsel generiert aus einer vorgegebenen Wortliste quadratische Legekarten, die mit einer Vielzahl von Buchstabenfeldern bedeckt sind. In diesen matrixartig verteilten Buchstaben ist das vorgegebene Wort versteckt. Es muss nicht erraten werden, wie bei einem Rätsel, sondern (nur) gesucht werden. Deshalb der Name Suchsel. Erlaubte Leserichtungen sind: waagrecht in beiden Richtungen, senkrecht in beide Richtungen, in allen vier Diagonalen. Insgesamt ergeben sich so pro Karte 12 mögliche Leseanordnungen.
Die Idee zu diesem Legespiel, stammt nicht aus der GHS-Borth, sondern wurde einem käuflichen Material entnommen. Allerdings erlaubt es das Programm, eigene Lernwörter zu verwenden und sie gezielt auszuwählen.

Ein Königreich für ein Pferd

Hilfefunktion für die Arbeit mit Fastgraf unter ZF

von Martin Bitter
Möllenkampweg 1a, 46499 Hamminkeln,

Oftmals möchte man beim Programmieren die Konventionen eines Befehls schnell zur Hand haben. Langwieriges Blättern erspart eine kleine Hilfefunktion.

***Stichworte*: ZF, Fastgraf, Hilfe**

F. Prinz hat Ted Grubers Grafikroutinen (FGDRIVER) für ZF und Holon implementiert. Im Schlachtengetümmel mit den teilweise recht komplexen Aufrufen benötige ich immer wieder ein Pferd, sprich: tiefe Blicke in die mitgelieferte Datei FGDRVREF.TXT.
Um mir die Zeit des Nachschlagens zu verkürzen habe ich eine Hilfefunktion geschrieben.
Alles was benötigt wird, ist eine Kopie von FGDRVREF.TXT im Ascii-Format (das kann wohl jede bessere Textverarbeitung erledigen), die ich FGDRVREF.ASC genannt habe.

Gebrauchanweisung:

Vor dem Laden von FGHELP.SEQ muss in der gekennzeichneten Zeile der Zugriffspfad für FGDRVREF.ASC an das jeweilige System angepasst werden.

Nach Laden von FGHELP.SEQ muss einmalig das Wort FILL_INDEXTAB aufgerufen werden. Dadurch wird eine Datei FGDRVREF.TAB erzeugt, die zu jedem ZF-Fastgraf-Befehl die Adresse im File FGDRVREF.ASC auflistet. Das ist möglich, weil zum Einen alle Fastgrafbefehle glücklicherweise mit einem FG_ beginnen und zum Anderen innerhalb des FGDRVREF.ASC alle Erläuterungen mit dem entsprechenden Stichwort direkt am Zeilenanfang beginnen. Das erlaubt es, nach der Kombination „Zeilenende+FG_“ zu suchen, seine Adresse zu merken und in FGDRVREF.TAB zu speichern.

Um die Hilfe zu benutzen, muss unter ZF ein FG? und der entsprechende ZF-Fastgrafbefehl eingegeben werden. **Bspl: fg? svgainit** zeigt den Text zu FG_SVGAINIT. Mit RETURN wird die nächste Zeile angezeigt. Mit ESC gelangt man zum Ausgangsbildschirm zurück.

Auf komfortablere „Interaktionen“ Seite rauf und runter etc. habe ich verzichtet.

Ich habe mich bemüht, das Quellfile so zu kommentieren, dass es hoffentlich, selbsterklärend ist (medium heavily documentet out). An einigen Stellen taucht $ als Präfix für Hexadezimalzahlen auf. Entweder benutzt man mein NewNumber.SEQ oder schaltet jedesmal in den Hex-Modus und wieder zurück in den Dezimalmodus.

```
\ eine Hilfefunktion, die in seperaten Files nach Einträgen
\  sucht -  hier: in einem Textfile zu FASTGRAF

empty              \ erlaubt vielfaches laden

handle tabfile     \ zwei Filehandle werden gebraucht eines für
handle helpfile    \ die Stichworttabelle, eines ist der Hilfetext

32 Constant Item_len    \ so lang ist ein Eintrag in der Tabelle

\ hier wird der Pfad für die beiden Files definiert ...
\ Bitte an eigene Dateien anpassen.

Create help_path
    ," F:\ALT_H1\FASTGRAF\FGDRVREF.ASC"

help_path tabfile $>handle \ ... und hier dem DOS bekannt
                           \ gemacht
Create $tab ," TAB"        \ einen neuen Extender definieren
       $tab tabfile $>ext  \ und an DOS weitergeben
                           \ (FGDRVREF.TAB)

help_path helpfile $>handle \ mit dem originalen Extender
                            \ (ASC) ein weiteres File bei
                            \ DOS anmelden

2variable helppointer \  zu jedem File benötigen wir einen
                      \ Zeiger,
2Variable tabpointer  \ der die aktuelle Lese/Schreibposition
                      \ hält

: helppointer@ ( -- d )  \ Zeiger der Hilfedatei holen
  helppointer 2@ ;

: helppointer! ( d -- )  \ Zeiger der Hilfedatei setzen
  helppointer 2! ;

: helppointer+! ( n -- )  \ Zeiger der Hilfedatei um n erhöhen
  s>d helppointer@        \ s>d erlaubt es negative Werte zum
  d+ helppointer! ;       \ Zeiger zu addieren

: tabpointer@ ( -- d )   \ Zeiger der Stichworttabelle holen
  tabpointer 2@ ;
```

```
: tabpointer! ( d -- )    \ Zeiger der Stichworttabelle setzen
  tabpointer 2! ;

: Tabpointer+! ( n -- ) \ Zeiger der Stichworttabelle um n er
                        \ höhen
  s>d tabpointer@       \ n vorzeichenrichtig auf double erwei
                        \ tern ...
  d+ tabpointer! ;      \ addieren und wieder speichern

0      Constant I/OBuffer \ der Puffer für die Files liegt im
                          \ Scratchbereich, dadurch wird
  \ Speicher gespart. Klappt solange genug Raum im
  \ dictionary ist!

Create Buffer $0A0D ,  40 allot  \ Einen Stichwortpuffer anle
                                 \ gen mit einem CR-LF am
                                 \ Anfang

\ hier wird der Anfang des Suchstrings Zeichen für Zeichen
\ in den Speicher gelegt. Grund: Steuerzeichen wie CR-LF
\ sind so einfach zu handhaben

Create first$ 5 c, $0A0D , Ascii f c, Ascii g c, Ascii _ c,

: make_help$ ( $adr -- )  \ wandelt den String addr von ZF-
                          \ Konvention in FG-Konvention
first$ count buffer place  \ Anfangsstring (CR-LF fg_) zum
                          \ Puffer kopieren
count                     \
\ 2 /string               \ man braucht nicht mehr das fg
                          \ eingeben
dup >R buffer count + swap move  \ Stinglänge sichern,
                          \ String hinter den Puffer kopieren
buffer c@ r> + buffer c! ;\ Countbyte im Puffer anpassen

\ ——————————————————————————————————
\ Die Fehlermeldungen, die DOS beim Filehandling
\ erzeugt, werden im Folgenden sichtbar gemacht.
\ Es folgen dann die Worte um Dateien zu öffnen, zu
\ lesen und zu verwalten.
\ ——————————————————————————————————

: .dos-Error ( n -- )       \ gibt einen Text aus
   ." Dosfehler Nr. " . ;

: ?DOS-Error ( n -- )       \ passender Text zu Fehler;
                            \ Programmabbruch!
 dup IF
 dup 1 = IF .dos-error abort" Ungültiger DOS-Aufruf!"THEN
 dup 2 = IF .dos-error abort" Datei nicht gefunden!"    THEN
 dup 3 = IF .dos-error abort" Pfad nicht gefunden!"     THEN
 dup 4 = IF .dos-error abort" Zu viele offene Dateien!" THEN
 dup 5 = IF .dos-error abort" Zugriff verweigert! (Share?)"
                                                        THEN
 .dos-error abort" Ein unbekannter Fehler ist aufgetreten!"
                                                        THEN
  drop ;

: help_open                 \ Dateien öffnen
  helpfile hopen ?dos-error \ Hilfetext OK?
  tabfile  hopen ?dos-error \ Stichworttabelle OK?
  pad 80 + =: I/OBuffer ;   \ Puffer adresse (dynamisch)
                            \ ermitteln

: help_close                \ Dateien schliessen
  helpfile hclose           \
  tabfile hclose 2drop ;    \ Rückmeldung ignorieren

\ ——————————————————————————————————
\ Hier die Routinen, um in eine Stichwortdatei die Adressen
\ des zugehörigen Textes einzutragen.
\ In der Datei stehen nach einem Zeilenende (das ist
\ gleichbedeutend an einem Zeilenanfang) die Schlüssel
\ wörter zum Aufbau der Indextabelle. Ich habe sie aus dem
\ Inhaltverzeichnis mit einer Texverarbeitung gebildet.
\  Eine Zeile ist 30 (mit CR-LF 32) Zeichen lang. Die letzten
\ zehn Buchstaben bilden den Ziffernstring, der Offsetadresse
\ des Schlüsselwortes. Das erlaubt es, die Datei mit den
\ Stichwortinformationen im Klartext zu lesen (und zu
\ schreiben).
\ ——————————————————————————————————

: tab_read ( -- n )     \ liest bis zu 200 Bytes ein. n=gelesene
                        \ Bytes
  tabpointer@ tabfile movepointer   \ Zeiger holen, via
                                    \ Handle DOS bekannt
                                    \ machen
  I/OBuffer 200 tabfile hread ;     \ in den Puffer .. 200
                                    \ Bytes .. aus tabfile lesen

: next_tab_item ( -- n ) \ liest nächstes Stichwort ein n=geles.
                         \ Bytes
  tabpointer@ tabfile movepointer   \ Zeiger im File positio
                                    \ nieren
  I/OBuffer 40 tabfile hread        \ 40 Bytes lesen
  I/OBuffer 40 bl scan nip 40 swap - dup \ bis zum ersten
                                    \ Leerzeichen lesen
  I/OBuffer buffer 1+ rot move buffer c!   \ in den Suchpuffer
                                    \ schreiben
  item_len tabpointer+! ;           \ Zeiger anpassen

: tab! ( d -- )    \  32bit-zahl in Stichworttab. schreiben
  <# # # # # # # # # # # #> pad 60 + swap move
                    \ d in String bei pad+60 wandeln
  tabpointer@ item_len 10 + 0 d- tabfile movepointer
                    \ Zeiger positionieren
  pad 60 + 10 tabfile hwrite drop ;
                    \ Ziffernstring eintragen

: help_read ( -- n ) \ bis zu 200 Byte aus dem Hilfetext lesen
  helppointer@ helpfile movepointer  \ Zeiger positionieren
  I/OBuffer 200 helpfile hread ;     \ max 200 Bytes in den
                                     \ I/O-Puffer lesen
```

```
6452. 2Constant helpstart  \ in meinem Textfile werden die
                           \ ersten Seiten übersprungen
False constant Tab_ende?   \ Tab_ende? ist ein Flag

\ ------------------------------------------------------
\ Eine Stichwortdatei mit den richtigen Adressangaben zu
\ versehen kann einige Zeit dauern.  Damit man weiss, dass
\ alles gut läuft gibt es hier eine kleine Anzeige, die zeigt:
\ Das System arbeitet !
\ ------------------------------------------------------

0 Constant (busy               \ Eine Laufvariable (Constant!)

\ hier werden vier Ascii-Zeichen in einer Tabelle abgelegt

Create busy_tab ascii \ c, ascii | c, ascii / c, ascii - c,

: working ( -- ) \ gibt bei jedem Aufruf eins der 4 Zeichen aus
  (busy dup busy_tab + c@     \ (busy als Index in die busy_tab,
                              \ Zeichen holen
   8 emit       emit          \ ein Schritt zurück, Zeichen
                              \ ausgeben
   1+ 4 mod =: (busy          \ Lauf"variable" erhöhen,
                              \ (zwischen 0 und 3)
  -2 #out +! ;                \ FORTHs Zeichenzähler anpassen

: in_I/Obuffer? ( n -- n flag ) \ da (n), wenn (fl) ist der Such-
                              \ string im I/O-Puffer?
  buffer count                \ Suchstring (adr, Lnge)
  rot I/OBuffer swap          \ Puffer ( adr, Lnge)
  search ;                    \ durchsuchen

: write_index ( n flag -- n )  \ schreibt evtll. an relativ.
                               \ Position n
  IF next_tab_item 0> =: tab_ende?  \ nächstes Stichwort-
                                    \ lesen, Erfolg merken
    dup s>d helppointer@ d+ tab!    \ Adresse im Hilfetext
                                    \ errechnen; schreiben
    Buffer count nip +        \ Länge des Suchstrings zum TOS
  ELSE abs      \ SEARCH kann auch negative Werte liefern!..
  THEN ;        \ ... falls Suchstring länger als Puffer

: fill_indextab ( --) \ füllt die Indextabelle aus
  1 spaces            \ eine kleine Lücke ausgeben (wg. working)
  help_open           \ Dateien öffnen
  TRUE =: tab_ende?   \ Tabellenendeflag vorbesetzen
  helpstart helppointer!  \ Zeiger im Hilfetext positionieren
  0.0 tabpointer!     \ Zeiger im Stichwortverzeichnis
                      \ positionieren
  next_tab_item  drop \ erstes Stichwort lesen
                      \ (drop=klappt immer)
  BEGIN               \ solange:
  help_read dup 0>    \ aus Hilfetext lesen; wurde mehr
                      \ als Null gelesen?
  tab_ende? and       \ und das Ende der Tabelle noch
                      \ nicht erreicht?
  WHILE               \ mache folgendes:
  in_I/OBuffer?       \ ist der Suchstring im I/O-Puffer?
  write_index         \ bei Erfolg in die Tabelle schreiben
  helppointer+!       \ Zeiger aktualisieren
  working             \ beruhigende Anzeige: Ich arbeite!
  REPEAT drop         \ und wiederhole:
  help_close ;        \ wenn fertig: Files schliessen!

\ ------------------------------------------------------
\ Es folgen die Worte, um einen String der im BUFFER liegt
\ in dem Stichwortverzeichniss zu suchen, die Adresse des
\ zugehörigen Textes im Textfile auslesen und den Text
\ anzuzeigen.
\ ------------------------------------------------------

: find_index ( -- )    \ sucht den Suchstring im Stichwort
                       \ verzeichis
  0.0 tabpointer!      \ Zeiger auf Null setzen
  buffer count pad 50 + place \ Suchstring aus dem Puffer zum
                              \ Scratchbereich
  BEGIN next_tab_item 0=  \ Wiederhole: nächstes Stich
                          \ wort lesen
    IF -1. tabpointer! EXIT THEN \ war da keines? also Ende
                                 \ flag setzen: Fertig!
    buffer count pad 50 + count \ Stringpuffer mit Stichwort
    search nip                  \ vergleichen (Adresse weg,
                                \ Flag behalten)
  UNTIL                \ bis gefunden
  -10 tabpointer+! ;   \ 10 Zeichen zurück (liegt die Adresse
                       \ des Textes)

: read_index ( -- )    \ liest die Adresse zum Stichwort aus
  tab_read drop        \ aus dem Stichwortverz. ab der akt.
                       \ Adr. lesen
  10 i/OBuffer 1- c!   \ 10 als Count davor setzen
    0 0 I/OBuffer      \ das ist ein ERsatz für Number !
    convert convert drop \ d.h. den Ziffernstring in eine Zahl
                         \ wandeln
  helppointer! help_read drop ;  \ Zeiger im Text setzen und
                                 \ Text einlesen

: rand_oben ( -- )  \ Zeichnet den oberen Rand eines Kastens
  0 0 at  ." +----------------------------------------------------------------------"
          ." ----------------------------------------------------------------------"
          ." ----------------------------------------------------------------------+"
  5 0 at bl emit buffer count 2-   \ Suchstring lesen
  swap 2+ swap qtype bl emit ;     \ im oberen Rand ausgeben

: rand_unten ( -- )          \ Zeichnet den unteren Rand
        ." +----------------------------------------------------------------------"
        ." ----------------------------------------------------------------------"
        ." ----------------------------------------------------------------------+" ;

: rand ( -- )          \ Setzt Farben und zeichnet oberen und
                       \ unteren Rand
  #out @ #line @       \ Cursorposition zum TOS
  blk >fg              \ Schwarz als Zeichenfarbe
  rand_oben at rand_unten ;    \ Ränder ausgeben
```

```
: rechts-links ( -- )   \ den rechten und linken Rand ausgeben
  blk >fg               \ wiederum Schwarz als Zeichenfarbe
  0 #out !  179 qemit   \ am Zeilenanfang ein 3 ausgeben
  78 qspaces            \ Leerzeile ausgeben
  179 qemit 1 #out ! ;  \ 3 ausgeben und zum Zeilenanfang
                        \ gehen

Variable padpointer 80 padpointer !     \ Ein Zeiger auf die
                                        \ aktuelle Stelle
                                    \ im Textpuffer (I/O-Buffer)

: show_line ( -- )          \ gibt eine Zeile des Textes aus
  rechts-links  brn >fg     \ Ränder zeichenen, Textfarbe braun
  padpointer @ dup pad + swap  \ Zeiger holen zu PAD
                               \ addieren NOS,
  280 swap -                \ Differenz Zeiger-Maxlänge als
                            \ Zähler
  0 ?DO count dup $0D = \ ein Zeichen einlesen auf CR testen
     IF drop 1+ 2 padpointer +! cr LEAVE    \ ja? --> Zeiger
                                            \ aktualisieren; fertig
     ELSE qemit 1 padpointer +!             \ ansonsten Zeichen
                                        \ drucken; Zeiger setzen
       #out @ 77 =              \ Zeilenende?
       IF cr rechts-links       \ ja --> neue Leerzeile
         brn >fg 3 qspaces      \ Textfabe erneuern, und Rand
                                \ bereich ausgeben
       THEN
     THEN                       \
    LOOP drop                   \ Stack aufräumen
  rand ;                        \ oberen und unteren Rand ausgeben

: read_lines ( -- )         \ liest neue Zeilen aus dem Text ein
  padpointer @ 195 >        \ sind wir schon gefährlich nah am
                            \ Ende (I/O-BUF.)
  IF padpointer @ 80 -      \ ja? --> gelesene Zeichen ermitteln
                            \ (TOS)
    80 padpointer !         \ Zeiger auf 80 entspr. NULL
                            \ gelesene Zeichen
    helppointer+!           \ TOS zum Text(file)zeiger addieren
    help_read drop          \ einen neuen Block lesen
  THEN ;

: new_line? ( -- flag )     \ wünscht der Anweder eine weitere
                            \ Zeile?
  key dup $1b = IF drop restscr true exit THEN
                        \ esc --> Ende, Bildschirm restaurieren
     dup $1c = IF drop 2drop #out @ #line @
                        \ Ende-Taste --> Ende, Text
                       true exit THEN
                        \ bleibt sichtbar
       $18 = IF read_lines  false exit THEN
                        \ Abwärtspfeil --> neue Zeile
       beep recurse ;   \ jede andere Taste --> Piep und von
                        \ vorne!
```

```
: show_index ( -- )   \ zeigt den Text zum gefunden Stichwort
  attrib c@           \ Bildschirmfarben retten      (TOS)
  #out @ #line @ \ Cursorposition retten (TOS)
  0 0 at statoff      \ zum Seitenanfang, Statuszeile ausschalten
  savescr dark        \ Bilschirminhalt speichern, leerer Schirm
  lgy >bg brn >fg\ Text- und Hintergrundfarbe setzen
  80 padpointer ! \ Zeiger setzen
  show_line show_line show_line
                      \ zu Anfang drei Zeilen zeigen
  BEGIN               \ "Endlos"schleife: solange
  show_line           \ eine nächste Zeile zeigen
  NEW_line?           \ wenn noch eine Zeile gewünscht wird
  UNTIL               \ sonst Schluss!
  staton at attrib c! ;   \ Statuszeile einschalten, Cursor setzen,
                          \ Farben restaurieren

: fg? ( / fg_name -- )\ zeigt den Text zum folgenden Stichwort
  help_open           \ Dateien öffnen
  bl word make_help$ \ Eingabestring isolieren, in Suchstring
                      \ wandeln
  find_index          \ zugehörige Adresse im Text ermitteln
  read_index          \ zugehörigen Text einlesen
  show_index          \ zugehörigen Text anzeigen
  help_close ;        \ Dateien schliessen
```

Lieber Friederich,

das war jetzt eine lange Zeit... Eure VD 14/#1 ist gerade angekommen und sie gefällt mir gut (Insbesondere, da sie einige Witze über Windows-95 enthält). Bitte leite meinen Dank an Dr. Beierlein für seine Übersetzung meiner 'Berichte' weiter. Du weißt, ich kann alles ohne Deutsch-Englisches Wörterbuch problemlos lesen (außer den Worten "viweniger verstdaedlich"). *(In meiner Version hieß das noch: "viel weniger verstaendlich" TB).* Und vielleicht noch eine kleine Korrektur: "Nord Carolina" ist an der anderen Seite der U.S.A., wo, soviel ich weiß, Wolf eines seiner Holon Systeme in einer Textil-Fabrik laufen hat. Hier sind wir in "Nord Kalifornien", und es hat die ganze letzte Zeit hier geregnet!

Soll ich ein paar Notizen über das SVFIG Januar-Treffen schicken und auch für den Februar (an dem ich nächsten Samstag teilnehmen will)? Wie ich bemerkt habe, werde ich immer bequemer je älter ich werde. Falls Leute die Bulletins vom SVFIG-Treffen von der FIG-Webseite bekommen, dann macht das für mich nicht viel Sinn zu wiederholen, was ich auf dem Treffen beobachtet habe.

Wie ist die Lage im Bergbau? Hast Du Zeit etwas an Holon zu arbeiten? Ich werde Wolf nach dem nächsten SVFIG-Treffen nochmal schreiben. Ich weiß, daß er ziemlich damit beschäftigt ist, sein Holon zu polieren und zu vermarkten.

Ich gehe immer noch voll auf Arbeit, obwohl ich mich auf einer sehr kleinen Sozial-Pension ausruhen könnte. Ich kann damit hier nicht leben, nun, so ist es besser auf Arbeit zu bleiben.

Bis demnaechst, alles Gute!

Henry

übersetzt von tb

Win32Forth: Mein erstes Programm

Bericht eines Forth-Anfängers über ein größere Arbeit mit Win32Forth.

Von Ulrich Richter
Oberwallstraße 4 ; D 47441 Moers

Über das hier zur Diskussion stehende Programm ist bereits in der letzten VD berichtet worden.-- Es handelt sich um dieses Schiebespiel. -- Der letzte Aufsatz betraf nur ein Detail der Arbeit, nämlich die Bereitstellung von Strings in der jeweils erforderlichen Landessprache. -- Dieser neue Beitrag soll nun die 'allgemeinen Erfahrungen des Autors mit Win32Forth' zusammenfassen. Die richtige Adresse für das Programm auf dem Taygeta-Server lautet:

http://www.taygeta.com/pub/forth/Applications/Win32For

Die Wahl des Themas gestattet mir, aus der Fülle der Schwierigkeiten, die ich bei der Erstellung meines Programmes hatte, Beliebiges herauszugreifen, das ich für bemerkenswert halte; Und ich hoffe, meine Auswahl wird auch Andere interessieren:

Bewegung der Steine mit BitBlt.
Erfahrungen mit dem OOP.
Arbeiten mit Freeware Win32Forth.

Über jedes dieser Themen kann ich natürlich nur meine Erfahrungen berichten; Ein Vergleich mit anderen Systemen, ist mir nicht möglich: Ich kenne sie nicht.

'Bewegung' der Steine mit BitBlt

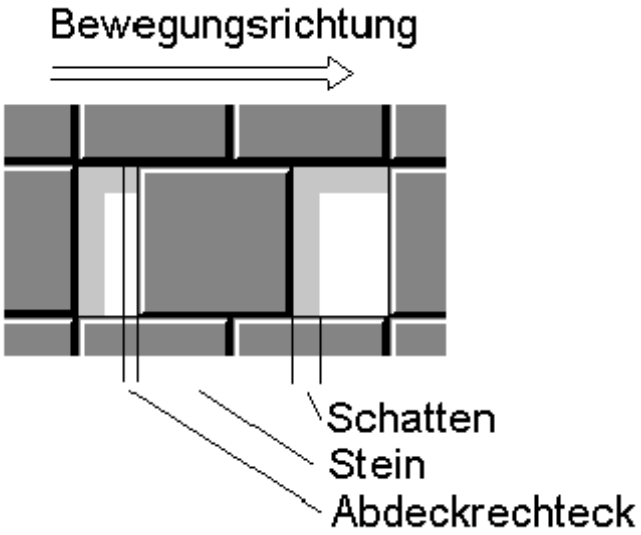
Für mich war es wichtig, *ein Gefühl dafür zu bekommen, wieviel Bewegung man unter 'normalem Windows' realisieren kann, ohne daß das Motiv während der Bewegung zu sehr leidet*. In unwissendem Mut hatte ich behauptet, daß die Beschriftungen der Steine auch lesbar bleiben sollten, *während* sie sich bewegen.

Dieses Bewegen der Steine ist *keine Scrollaufgabe*; Würde man den Stein nämlich mit Scroll anfassen, dann würde Windows auf dem frei werdenden Bildschirmstreifen die Hintergrundfarbe des Fensters einfüllen. Beim vorliegenden Problem sind aber *zwei* Farben einzufüllen: Kastengrund und Kastengrund mit Schatten. Auch wenn man sofort ein FillRect nachschiebt, bleibt ein 'Blitzen' in der Hintergrundfarbe für das Auge sichtbar. Außerdem wird die Bewegung lahm, wenn man mehr als einen Call pro gezeigtes Bild absetzt.

Deshalb ist das Bewegen der Steine eine *BitBlt-Aufgabe*. Die zu bewegende Bitmap besteht aus drei verschiedenen Bereichen:

Dem blauen Stein,
Dem Schatten,
Einem zusätzlichen 'Abdeckrechteck', das hinter dem sich bewegenden Stein den normalen Kastenhintergrund herstellt.

Dieses Abdeckrechteck besteht aus zwei Bereichen: dem Schatten und dem normalen Kastengrund. Seine Breite hängt davon ab, um wieviele Pixel man vorspringt, wenn man das nächste Bild ausgibt. -- Die Farbe des schattenlosen Teils ändert sich beim Austritt aus dem Schatten. –Schatten und Abdeckrechteck werden während der Steinbewegung auf- bzw. abgebaut, wie aus der folgenden Skizze hervorgeht.



Die Skizze zeigt deutlich, daß Windows bei dieser Lösung nur das *Minimum an Pixeln* zu transportieren hat. –

Beim Experimentieren mit dem Programm stellt man bald fest, daß es überhaupt nicht auf die Anzahl der codierten Zeilen oder Worte ankommt; Sobald die Loop für die Ausgabe eines Bildes nur *einen einzigen Call* enthält, steht die Geschwindigkeit des Steins praktisch fest.

Das heißt: *Wenn sich jetzt die Steine zu langsam bewegen, liegt es am Windows, nicht an der Programmierung*!

Ich habe auch versucht, den Bildschirm während der Bewegung des Steins nach Ausgabe jedes einzelnen Bildes frei zu machen und dann wieder neu zu sperren (EndPaint ... BeginPaint, also zwei Calls mehr). Das Ergebnis war viel zu langsam, also unbrauchbar.
Das Programm bietet jetzt vier verschiedene Geschwindigkeiten an:

Für jedes Pixel wird ein Bild abgesetzt (1-fach),

Für jedes dritte Pixel wird ein Bild abgesetzt (3-fach),

Es werden 6 Bilder abgesetzt (max),

Zwischen zwei nebeneinander liegenden Pixeln wird eine zusätzliche Verzögerung herbeigeführt, indem in einer Loop bis 400 gezählt wird (Zeitlupe).

Wenn der verwendete Computer keine 'Beschleuniger-Grafikkarte' (also noch Chip Trident 8900C, ET4000 o. Ä.) hat, wird das Spiel zu einer Qual. Die Farben kommen nicht richtig, weil ich habe keine Paletten mehr programmiert habe. Und die Steine ruckeln langsam auf ihre Plätze.

Tempo der Bewegung testen

Will man jedoch mit einem modernen Computer bis an die Leistungsgrenze testen, dann geht man wie folgt vor:

Umschalten auf 1024*768 Pixel oder mehr,

Ein Spiel zB. 3*3 aufmachen,

Das Leerfeld nach links oben holen,

Das ganze Spielfenster nach rechts unten *schieben* und dann als *Vergrößerung* wieder nach links oben hochziehen, bis nur noch zwei Steine auf dem Monitor sind,

Und dann den supergroßen Stein mit verschiedenen Geschwindigkeiten schieben.

Erfahrungen mit dem OOP

Im Folgenden habe ich ein paar Ziele, die zur Anwendung eines OOP-Systems führen können, kurz kommentiert:

Die *Beschränkung des OOP nur auf das Arbeiten mit Windows* ist zwar ein gemeinsamer Nenner für die hier Diskutierenden, ist aber sicher keine geeignete Anwendung! Das OOP muß auf die Bearbeitung eines richtigen Problems angesetzt werden!

Die *Verminderung des zu einem Programm gehörigen Codes* ist heute nicht mehr interessant. Windows-Rechner sind groß genug, um umfangreiche, voll auskommentierte Programme zu speichern und bei Bedarf zusammenzustellen. Seit der Einführung der langen Filenamen ist auch die Gliederung unproblematisch.

Mit dem Win32Forth ist es möglich, ein Programm derart *arbeitsteilig* zu erstellen, daß der Programmierer der Anwendung *nur die letzten* zwei oder drei *Vererbungsebenen richtig verstanden* hat. Diese sind mit minimalen Kenntnissen aus der Quelle zu entnehmen.

Die OOP-Programmierung kann *Träger einer sinnvollen Normung* sein, indem bestimmte (genormte) Teile eines fertigen Produkts (als Methoden) bezogen werden müssen, und nicht neu erstellt werden dürfen.

Eine *Verminderung der Programmierarbeit durch OOP* scheint mir allerdings bei den ersten Anwendungen nicht erzielbar, weil -- nach dem Verstehen des OOP überhaupt -- vor jeder Benutzung erst ein Einarbeiten in die benutzten Klassen nötig ist. Das OOP wird sinnvoll da eingesetzt, wo gleichbleibende Moduln öfter verwandt werden sollen.

Arbeiten mit der Freeware Win32Forth

Zu bedenken ist, daß dieses Angebot Win32Forth sich von üblicher Software dadurch unterscheidet, daß es *mehrere Sachgebiete überspannt*:

-- Objektorientierte Programmierung --
Dieser Teil ist von einem Forther leicht zu durchschauen. Für das Arbeiten genügt zunächst, wenn das Prinzip bekannt ist.

-- Forth --
Dieser Teil ist in der Art des ZF-Forth dokumentiert. Gute neue Zusätze vereinfachen das Arbeiten auf diesem Sachgebiet. Es sind 'Anwendungskenntnisse' nötig.

-- Programmierung für Windows --
Von großem Vorteil (beinahe auch Voraussetzung !) ist, wenn der Benutzer des Win32Forth vorher bereits eine Windows-Programmierung (zB. in C++) verstanden hat.

-- Windows API --
Dieses Sachgebiet ist geschützt. Bei der Arbeit muß der User diesen Stoff parat haben, verstanden haben und permanent anwenden. Win32Forth benutzt Variablennamen der API. Wer sie nicht zuordnet, versteht es nicht.

Tom Zimmer hat in seiner Arbeit -- abgesehen von einem wirklich kurzen Readme und etlichen Beispielen -- auf eine gesonderte Beschreibung von Details verzichtet. Statt dessen bekommt jeder Benutzer die vollständigen, kurz aber klar

kommentierten Quellen.

Wenn die oben beschriebenen Sachgebiete hinreichend bekannt sind und wenn sich der Win32Forther einen Überblick über die ausgelieferten Quellen verschafft hat, kann mit der eigentlichen Programmierung begonnen werden.

Während dieser ergibt sich folgendes Bild:

> Man kann keine Zeit dafür verlieren, etwas in der Dokumentation zu suchen, denn dieses System ist zwar spärlich, aber *am logischen Ort dokumentiert*.
>
> Weil Quelle und Dokumentation beieinander stehen, gibt es nichts Vergleichbares, das so *vollständig beschrieben* ist.
>
> Weil das OOP für Einfügungen aller Art offen ist, und weil die Quelle vorhanden ist, gibt es *Nichts, das im System nicht vorgesehen ist*.
>
> Das System wurde seit seinem Erscheinen von einer größeren Anzahl fremder User benutzt. Heute kann man von einem *weitgehend fehlerfreien* System ausgehen.
>
> Falls noch irgend etwas nicht gut oder richtig sein sollte, *kann man es sofort selbst in die gewünschte Form bringen*.

Für mich war die Entdeckung völlig unerwartet, daß dieses Win32Forth ein ausgezeichnetes Hilfsmittel ist, um *Windows zu lehren*. – Das ergibt sich aus der Interaktivität des Forth. Man kann die Strukturen des Aufrufs am Bildschirm aufbauen und die Funktionen einzeln vorführen.

So dürfte das Arbeiten mit diesem System nach Überwindung der Anfangsschwierigkeiten äußerst effektiv sein. Die Zeitersparnis ergibt sich später mehr und mehr, wenn laufend mit der Quelle gearbeitet wird. –

Das System von Tom Zimmer deckt das Bedürfnis 'Forth für Windows' vollständig ab, deshalb finde ich es sehr gut.

Suchsel ist mein erster Versuch, das WIN-API zu nutzen. In der Hauptsache wurde Code aus WIN32FOR abgewandelt und angepasst. Bei vielem verstehe ich gar nicht so genau, was ich mache - aber es läuft (leidlich). (Kontakte, kritische Quelltextleser gesucht).
Die momentane Version bezeichne ich als Gamma. D.h. ich kann sie interaktiv mit WIN32FOR bedienen, eine stand-alone-applikation ist es (noch) nicht.
Doch nun zur Auflösung: weiße Karten 1=**FORTH** 2=**TAGUNG** 3=**MOERS** 4=**ANMEL** (-) 5=**DEN!** (lesbare) graue Karten 1=LAND 2=MOND 4=SALZ 6=OFEN 7=MAUS 8=SIEB 9=LÖWE 10=RABE 11=TANZ 20=TIER. *M.*

His Masters Voice: Lean is beautiful ?

Ausgabe von Waves unter DOS mit ZF

Von Martin Bitter
Möllenkampweg 1a, 46499 Hamminkeln

Stichworte: Soundausgabe, ZF, Lean-PC

Die aktuelle Lage:

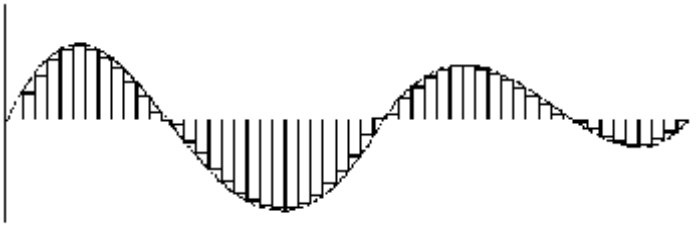
Ich unterrichte als Sonderschullehrer an einer Hauptschule. Hauptschulen sind in vieler Hinsicht Stiefkinder der Nation. Dies gilt trozt solcher Initiativen wie „Schulen ans Netz" auch für „meine" Schule.
Die Grundausstattung unser Informatikraumes besteht aus 6 Epson-PC (8088 CPU, 4 MHZ-Takt; 20 MB-Festplatte, MDA-Grafikkarte!). Das benutze ich nicht (der Informatikraum ist eh weit vom Schuss).
Seit einiger Zeit hat meine Klasse einen „modernen" (Reste)-PC (386 CPU 33 MHZ, 8MB, 250MB-Festplatte, VGA-Grafik). Das benutze ich!

Ein Problem - Forth ist die Lösung

Unter vielem anderen stellte sich das Problem, Sprach- und Geräuschesamples auszugeben. Ohne Soundkarte! Selbst der Kauf einer Soundkarte hülfe wenig, da heimische Versuche, eine solche unter dem Gespann DOS-ZF zu betreiben fehlschlugen (weiss jemand Hilfe?).
Die Erinnerung an selige ATARI-Zeiten half weiter: Dort war es gelungen, den Mini-Monitor-Lautsprecher durch gezieltes, genau getimtes Belegen mit unterschiedlichem **Rauschen** zur Ausgabe von beliebigen Samples zu bewegen.
Ähnlich verfahre ich bei meiner „Soundausgabe".

Brown'sche Molekularbewegung, Treibball, Speaker und ein Tritt in den Allerwertesten

Theoretisch ist es recht einfach: Ein Sample ist eine Folge von abgestuften (diskreten) Werten, die den Druckverlauf eines Geräusches wiedergeben.
Aus dem ursprünglichen glatten Bild eines solchen Druckverlaufes wird durch die Beschreibung mit diskreten Werten eine gestufte Reihe. (Bild 1)



Bei der Wiedergabe eines Samples durch z.B. eine Lautsprechermembran, folgt die Membran diesen gestuften Werten. Sie rekonstruiert den ursprünglichen Druckverlauf nicht kontinuierlich sondern in Sprüngen.
Daraus ergeben sich zwei wesentliche Erkenntnisse:

1. Je feiner die Abstufungen sind, mit denen der Druckverlauf beschrieben wird, umso geringer (weniger störend) sind die Sprünge, die die Lautsprechermembran macht. In dem Bild-

chen sind es 144 verschiedene Höhenstufen, die zur Verfügung stehen.

2. Je dichter die Werte (in der Zeit) beieinander liegen, umso feiner kann der zeitliche Verlauf abgebildet werden (die höchste hinreichend genau darstellbare Frequenz ist die Hälfte der Samplefrequenz). In dem Bild liegen die einzelnen „Streifen“ ca. 15 Pixel beieinander.

Für den PC-Speaker gilt ein wesentliches Handicap, soll er eine solche Bewegung ausführen: Er kennt streng binär nur zwei Zustände: Ein - Aus! 0V - 5V!

Es ist ein leichtes, mit der vorhanden PC-Hardware (Timer-Chip und Gatter) Rechteckschwingungen in einem weiten Frequenzbereich zu erzeugen - aber es war bei deren Entwurf nicht vorgesehen, den Speaker mit kontinuierlichen Werten zwischen 0 bis 5V zu betreiben.

Einige Analogien: Bei der Brown’schen Molekularbewegung sieht man nicht die anprallenden kleinen Moleküle, sondern das unregelmäßige Erzittern der großen Moleküle. Bei dem Spiel Treibball darf der Spielball (Medizinball) nicht direkt berührt werden, sondern er muss durch gezieltes Werfen mit kleinen Gymnastikbällen in das gegnerische Tor getrieben werden.

Problemlösung: Wenn sich also die Lautsprechermembran nicht direkt auslenken lässt, so vielleicht doch indirekt, durch viele kleine elektromagnetische Stösse, die sich in der Summe so auswirken, wie das Einwirken einer elektromagnetischen Kraft bestimmter Stärke.
Die Lausprechermembran mit ihrer Trägheit kann vielen kurzen Impulsen in ihrem Auf und Ab nicht folgen, sondern nimmt tatsächlich eine Mittelstellung ein.

Das war mein erster Ansatz (vgl. PLAYPWM und TIMER2 im Sourcecode). Durch den Timer-Chip des PC kann eine Rechteckeckschwingung von (fast beliebigem) Tastverhältnis ausgegeben werden. Der Erfolg enttäuschte. Zwar ist die Wiedergabe recht gut in der Qualität, aber leider auch sehr, sehr leise.

Wenn die Lautsprechermembran den Mittelwert einer Impulsfolge einnimt (ich glaube das nennt man PWM; Puls Weiten Modulation?), so kann sie, wenn der Timerbaustein als Rechteckgenerator benutz wird, niemals ihre höchste Auslenkung (5V) erreichen, da die Rechteckfolge ein maximales (je nach Sichtweise minimales) Impulsverhältnis von 1:1 liefert (Bilder 2,3,4).

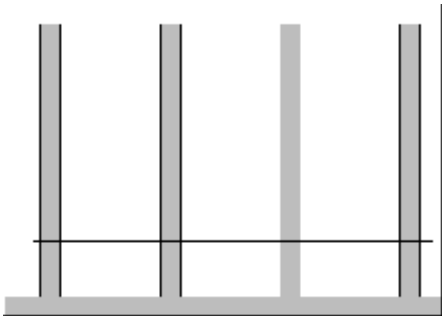


Bild 2: Der Timer-Chip liefert alle 5 Timer-ticks einen Impuls, das durchschnittliche Niveau liegt bei ca. 0,8 V.

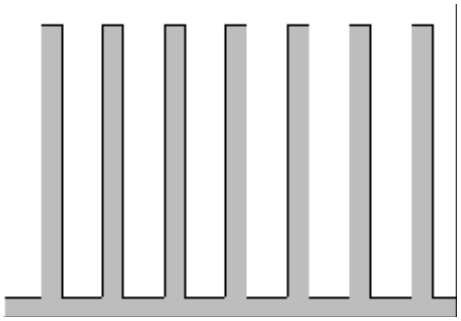


Bild 3: Der Timer-Chip liefert alle 2 Timer-ticks einen Impuls, das durchschnittliche Nieveau liegt bei ca. 1,6 V.

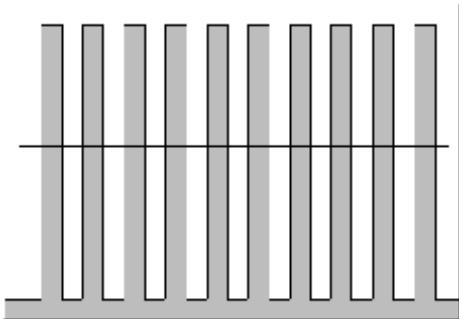


Bild 4: Der Timer-Chip liefert nach jedem Timer-tick einen Impuls, das durchschnittliche Nieveau liegt bei ca. 2,5 V.

Es muss also ein anderes Verfahren her: Wenn viele kleine Stöße (Spannungsspitzen) nicht reichen, vielleicht tut es dann ein starker Stoß?

Analogie: Ein Ball kann zwar mit vielen kleinen Tritten über den Rasen bewegt werden, aber wenn er hoch und höher fliegen soll, hilft nur ein wirklich fester Tritt.

Umgesetzt bedeuted das, dass der Timer-Chip nicht mehr als Rechteckgenerator programmiert wird, sondern als Monoflop, der einen einzigen Impuls variabler Länge liefert. Abhängig von der Länge dieses Impulses wird die Lautsprechermembran mehr oder weniger ausgelenkt. (Bilder 5,6,7).

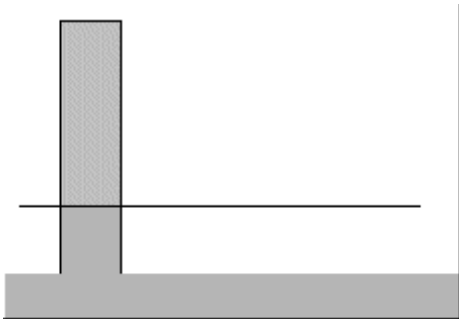


Bild 5: Einem kurzen Impuls kann die Membran nicht ganz folgen. Sie erreicht nur einen Teil ihres Vollausschlages.

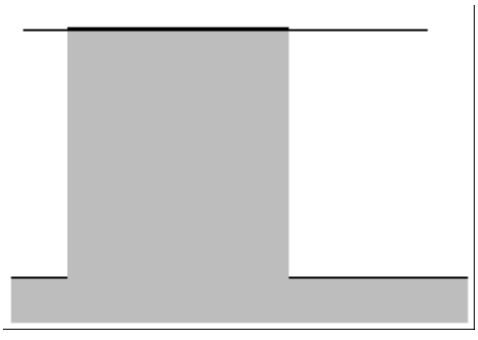


Bild 6: Bei einem Impuls von hinreichender Länge erreicht die Membran ihren maximalen Ausschlag.

Bild 7: Eine weitere Verlängerung des Impulses hat keinen höheren Ausschlag zur Folge, die Membran ist am „Anschlag“.

Nachteil: mit einem solchen Tritt in den Allerwertesten, erreicht die Lautsprechermembran ihren jeweiligen Sollausschlag nur für einen kurzen Moment, für längere Zeit läßt sie sich nicht auf einer Höhe halten. Das kommt allerdings beim Abspielen von Samples kaum vor.

Zum Programm

Im Quelltext wird in Hochsprache gezeigt, wie mit einer Frequenz von ca. 11 KHZ die jeweiligen Samplewerte ausgelesen und der Timerchip entsprechend programmiert wird (Rechteckgenerator --> PWM; Monoflop --> ballistisch), um die Lautsprechermembran zu der erwünschten Bewegung zu veranlassen.
Ein vorzeichenbehaftetes 8-bit Sample liefert z.B. einen Wert von 127, das bedeutet Maximalausschlag in positive Richtung. Da der Speaker nur positive Werte kennt, entspricht dieser Samplewert einem einzustellenden Wert von 255. Ein Samplewert von -127 (Vollausschlag in negative Richtung) entspricht folgerichtig einem Wert von 0. Ein Samplewert von 0 (Ruhestellung, keine Auslenkung) führt zu einem Wert von 127. Durch viele Impulse der Maximallänge wird versucht, die Membran auf einem mittleren Niveau (2,5V) zu halten. Gute Ohren nehmen das als Pfeifen war, dieses Pfeifen (Diskretisierungspfeifen) ist jeder Samplewiedergabe überlagert (nicht nur in der Ruhelage).*

Da die notwendigen Frequenz- und Impulswerte für den Timerchip empfindlich von den jeweiligen physikalisch-mechanischen Bedingungen des einzelnen Lautsprechers abhängen (Dicke=Masse der Membran und Steifheit, Kraft der Spule, usw.) muss die Soundausgabe kalibriert werden.
Zum einen gibt es einen Parameter „TRAEGHEIT“ der die maximale Impulslänge wiederspiegelt. Bei meinem System sind das 111 Timerticks. Bei dieser Länge ist der Lautsprecher am Anschlag.
Zum anderen gibt es einen Parameter „SPREIZUNG“. Dieser setzt schlecht ausgesteuerte Samples , die nur einen Teil des 8-bit-Umfanges ausnutzen, in „echte“ 8-bit Systeme um. Er dient gleichzeitig der Lautstärkeregelung.
Aus der gewünschten Lautstärke und dem Maximalwert des Samples wird ein Skalierungsfaktor gebildet, der den Wertebereich eines 8-bit-Samples (255) auf den Wertebereich des physikalischen Systems (hier 211) abbildet. Die so errechneten Werte werden in einer Tabelle abgelegt.
Im Quelltext wird eine lineare Umrechnung gezeigt. Die Art der Umrechnung bietet ein weites Feld für Experimente und hat einen merkbaren Einfluss auf die Qualität der Wiedergabe.
Leider bietet der Timer-chip in den verwendeten Modi nicht die Möglichkeit, interruptgesteuert zu arbeiten. Die Sampleroutine muss selbst für das entsprechende Timing sorgen.
In den beiden Hochsprachevariationen PlayPWM bzw. PlayBall wird eine Pause von 3 MS zwischen den einzelnen Aufrufen eingelegt. Achtung: die ZF-Worte SET-FUDGE und FUDGE sind für „alte“ Hardware ausgelegt. Ein ZF MS entspricht je nach CPU-Takt bis zu 0,000025 sek und weniger!
Genauer, aber auch stark systemabhängig sind Regelungen über DO LOOP Schleifen.
Die Version PlayBall liegt auch in einer Codedefinition vor, die in den Pausen auch umfangreichere andere Arbeiten erlaubt. Mir ist es gelungen, auf einem AMDK5 System parallel zur Soundwiedergabe, das Sample grafisch auszugeben .

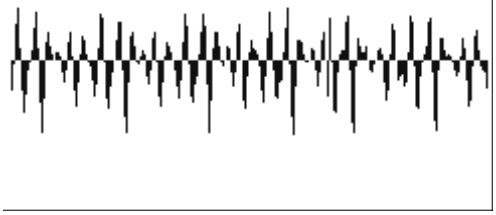


Bild 8: Sampleausschnit aus „YOU“ Screendump 640x480 Pixel

Lean is beautiful??

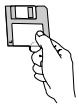
Zu einem gewissen Teil ist diese „Soundausgabe“ ein Schritt zurück in die Computersteinzeit.
Solange aber Soft- und Harwarehype verhindert, dass sich sich stabile Software entwickelt, die auf Maschinen läuft, die älter als 6 Monate sind, solange wird es zumindest in (armen) Schulbereichen Bedarf an Routinen und Algorithmen geben, die auf Rechnern laufen, die einige Jahre alt sind. Denn Rechner zählen zum festen Inventar und sind nicht Verbrauchsmaterial!
In diesem Sinne ist LEAN nicht nur BEAUTIFUL sondern auch NOTWENDIG.

** Das Diskretisierungspfeifen läßt sich durch eine kleine Schaltung mit einem Kondensator und einer Spule dämpfen. (Kondensator parallel zum Speaker, Spule in Serie, Werte empirisch ermittelt ca. 2.2 æF 1mH, bestimmt kann das jemand genau ausrechnen.)*

Was Sie schon immer über intelligente Softwareagenten wissen wollten, finden Sie in diesem Buch. In leicht verständlicher Sprache veranschaulichen die Autoren die Funktionen und den Einsatz intelligenter Hilfssoftware zur Erledigung wiederkehrender Aufgaben.
Gleichzeitig erhalten Sie einen Überblick darüber, welche Software in diesem Bereich zur Zeit auf dem Markt ist, wer sie anbietet, und wo sie eingesetzt wird. Als verantwortlicher Mitarbeiter Ihrer Datenverarbeitung, erhalten Sie einen Einblick darin, welchen Nutzen eine Einbindung intelligenter Software Agenten Ihrem Unternehmen bringt, wie Sie diese Software neu in Ihrer DV-Landschaft einführen und integrieren können.
Überraschend gut beschrieben sind die Einsatzmöglichkeiten in den unterschiedlichen Bereichen einer Unternehmung, die Vorteile eines Einsatzes und die Vergleiche der einzelnen Softwareagenten untereinander.
Auch die versierten ProgrammiererInnen werden in diesem Buch eingehend mit den Grundlagen über den Aufbau, die eigene Entwicklung und den Einsatz dieser Softwareagenten, in leicht verständlicher Weise eingeführt. Interessant sind in diesem Zusammenhang auch die Hinweise auf Adressen verschiedener Anbieter bereits bestehender Softwareagenten, sowie entsprechenden Softwaretools zur eigenen Softwareerstellung. Obwohl dieses Buch sich in der Hauptsache an DV-Profis wendet, ist es auch für sogenannte Einsteiger durch den gut strukturierten Aufbau und die leichte Lesbarkeit empfehlenswert.
Die Einführung in die Grundlagen der Intelligenten Softwareagenten, und deren praktischer Anwendung im Unternehmen, ist den Autoren mit diesem Werk gelungen.
Ein Muß für jeden, der sich mit dieser zukunftsweisenden Softwaretechnologie befaßt.

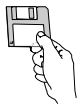
HANSER Verlag
ISBN 3-446-19269-7
98,- DM

up



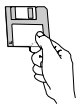
```
\ erster Versuch der Soundausgabe                        M.Bitter Sept. 95
\ gestützt auf H.P. Messmer: PC_Harwarebuch, Addison-Weslesy 1995/3
\ Seite 695 ff
\ Anmerkung: Ich halte es für lesbarer und einfacher, Zahlenwerte passend zur
\ jeweiligen Basis zu schreiben. In meinem System wird das von NewNumber
\ erledigt.
\ NewNumber gibt es in der MHB (02841-57325 : /SUPPORT/FORTH/ZF
\ Alternativ kann Base jeweils von Hand gesetzt werden, oder aber die passenden
\ Dezimalwerte werden errechnet und eingesetzt.

\ ------------------------------------------------------------------------------
\ Die Soundfähigkeiten eines PC sind von Hause aus sehr beschränkt. In der
\ Regel gibt der Lautsprecher nur ein Signal aus, das er von einem Timer-
\ chip bekommt. D.h. um einen bestimmten Ton auszugeben, muß erst die Fre-
\ quenz des Timers - hier ist es der 1. PIT  - eingestellt werden, die dann
\ über ein Gatter (Bit 1 des Ports $61) verknüpft, an den Lautsprecher wei-
\ tergeleitet wird. Im Klartext: Es können nur periodische Signale gleicher
\ Amplitude (Rechteckschwingungen) ausgegeben werden, eine Lautstärkenein-
\ stellung ist nicht möglich. Oh, du toller marktbeherrschender PC!
\ ------------------------------------------------------------------------------

Vocabulary Sound Sound Definitions also

$0061 Constant speaker_port     \ Adresse des Lautsprecherports

code speaker_on                 \ Lautsprecher einschalten
     speaker_port # AL IN       \ Lautsprecher adressieren und lesen
              %11 # AL or       \ diese Bits schalten ein und verknüpfen ...
     speaker_port # AL out      \ ... den Timer mit dem Speaker; schreiben
  next end-code

code speaker_off                \ Lautsprecher ausschalten
    speaker_port # AL IN        \ Register lesen
       %11111100 # AL and       \ signifikante Bits ausmaskieren
    speaker_port # AL out       \ und schreiben
  next end-code

$0042 Constant Timer2_port      \ Adresse des PIT 8253 Zähler Nr. 2
$0043 Constant Steuer_reg       \ Steuerregister des PIT 8253

\ ------------------------------------------------------------------------------
\ Bit     | 76 | 54 | 321 | 0 |
\ Name    | SC | RW | Mod |BCD|
\ SC Select Counter = Zählernummer
\ RW READ/WRITE 00=Zähler-Latch 01=nur Low Byte 10=nur High Byte
\               11=erst Low Byte dann Hight Byte lesen/schreiben
\ Modus= 0-101 ( sechs verschiedene Zählmodi)
\ BNC  = Zählen im Hexadezimalsystem (0) oder BNC (1) = 0 -9999
\ ------------------------------------------------------------------------------

code timer2 ( n -- )    \ Timer gibt alle n Ticks einen Impuls
  %10110110 # AL mov    \ Zähler 2, low-high Byte, Modus 3, Hex.
  Steuer_reg # AL out   \ in das Steuerregister schreiben
  AX pop                \ Zählerwert vom Stack nach AX holen
  Timer2_port # AL out  \ Lowbyte schreiben
  AH AL mov             \ Highbyte zum Lowbyte schieben
  Timer2_port # AL out  \ Highbyte schreiben
  next end-code         \ fertig

code timer2_flop ( n-- ) \ Timer gibt einen Impuls von n Ticks Länge
  %10110000 # AL mov    \ Zähler 2, low-high Byte, Modus1, Hex.
  Steuer_reg # AL out   \ in das Steuerregister schreiben
  AX pop                \ Zählerwert vom Stack nach AX holen
  Timer2_port # AL out  \ Lowbyte schreiben
  AH AL mov             \ Highbyte zum Lowbyte schieben
  Timer2_port # AL out  \ Highbyte schreiben
  next end-code         \ fertig
```

```
code timer_stop ( -- )  \ Timer gibt keinen Impuls
  %10110000 # AL mov    \ vgl. oben aber Modus 0
  Steuer_reg # AL out   \ ins Steuerregister
  next end-code         \ fertig


\ ------------------------------------------------------------------------
\ Leider läßt sich der Lautsprecher nur mit 5 Volt oder 0 Volt beschalten.
\ Das bedeutet, ein Wert von z.B. 2 Volt und damit eine Regelung der Laut-
\ stärke ist (eigentlich) nicht möglich. Eigentlich!
\ Ein Ausweg besteht darin, mit unterschiedlich langen Stromimpulsen zu
\ arbeiten, denen die Lautsprechermembran nur beschränkt folgen kann.
\ Ein einzelner Impuls ist so kurz, daß die Membran gar nicht reagieren
\ kann. Man (ich) hört nichts. Eine Wartezeit von 100 Schleifen läßt der
\ Membran genug Zeit, um sich zu bewegen. Man (ich) hört ein leises Knacken.
\ Jede Verlängerung der Wartezeit (Tick) wirkt sich als Lautstärkeerhöhung
\ aus.
\ Natürlich gibt es eine Obergrenze für diese Wartezeit, ab der keine Aus-
\ wirkungen auf die Lautstärke mehr resultieren. Die Membran ist sozusagen
\ an ihrem Anschlag angelangt. Eine weitere Verlängerung der Schleife be-
\ wirkt,daß nun zwei Knackgeräusche zu hören sind: das Einschwingen und das
\ Ausschwingen der Membran.
\ Wie lang die Schleifen, bzw. die Wartezeiten sein müssen, um unterschied-
\ liche Lautstärken aufzulösen, hängt entscheidend von den physikalischen
\ Gegebenheiten des Lautsprechers und der Geschwindigkeit des PC-Systems,
\ sowie von den Hörfähigkeiten des jeweiligen Anwenders ab.
\ ------------------------------------------------------------------------
\ ------------------------------------------------------------------------
\ Falls es gelingt eine Routine zu schreiben, die ein Sample abspielt, so
\ muss unter anderem viel Speicher verwendet werden.
\ Deshalb hier zwei Worte, die ein Sample in einen externen Speicher laden
\ und diesen nach Gebrauch wieder freigeben.
\ Leider, leider ist der Speicherplatz beschränkt. Lange Samples gehen also
\ (noch) nicht!
\ ------------------------------------------------------------------------

handle sample                   \ Dos-Forth-Handle für eine File

1.2 2Constant Samp_len          \ Länge des Samplefiles vorbesetzen
0 Constant Samp_start           \ Zeiger auf Startsegm. des Samplebereichs
0 Constant Samp_segment         \ Zeiger auf die Samplesegementadresse
0 Constant Samp_offset          \ dito für die Offsetadresse


Create temp_exlist ," COMEXEARJBINOBJBAKSEQMAPOVRBLKTXTDOC" \ Files mit diesen
                                \ Endungen nicht anzeigen

: open_sample ( -- )            \ öffnet ein File
  [ HIDDEN ]                    \ 'versteckte' Worte sichtbar machen
  exlist @                      \ Adr. der 'originalen' Ausschlußliste holen
  temp_exlist exlist !          \ Adr. der eigenen Ausschlußliste setzen
  getfile rot exlist !          \ Dateiauswahlbox, Ausschlußliste restaurieren
     IF sample $>handle         \ Datei ausgewählt?
     ELSE abort" Doch nicht?"   \ evtll: Kommentar
     THEN                       \ weiter im Text
  sample hopen 0<>              \ Datei mit dem Handle SAMPLE versehen + öffnen
  IF abort" Konnte Sample nicht öffnen!" THEN \ evtll. Fehlermeldung
  0 =: samp_offset              \ Sampleoffset (Zeiger) auf NULL setzen
  sample endfile                \ Datei(sample)länge holen
  ['] samp_len >body 2!         \ und als 32-bit-Konstante! speichern
  0.0 sample movepointer ;      \ DOS-Lesezeiger auf NULL setzen

: alloc_sample ( -- )           \ Speicher für das Sample anfordern
  samp_len 16 um/mod nip        \ Bytes in Paragraphen umrechnen  (16 Byte)
  dup -1 = IF abort" nicht genug (DOS)Speicher!" THEN        \ ging das?
  1+ alloc  dup                 \ (ja) Speicher anmelden
  8 =                           \ meldet DOS-Rückgabewert einen Fehler?
```



```
  IF abort" Nicht genug Speicher für das Sample vorhanden!" THEN \ ja=Meldung
  dup =: samp_segment           \ Segmentadresse des Speichers merken
      =: Samp_start ;           \ ... als Startzeiger setzen

: free_sample                   \ Samplefile schließen und Speicher freigeben
  sample hclose drop            \ Sampledatei schließen; Meldungen ignorieren
  samp_start                    \ Segmentadr. des Speichers holen
  0 =: samp_start               \ auf NULL setzen
  dealloc                       \ Speicherbereich beim DOS abmelden
  9 = IF ABORT" Samplespeicher konnte nicht freigegeben werden!" THEN ;

: load_sample ( -- )            \ ein Samplefile laden
  samp_start 0<>                \ ist (noch) Speicher reserviert?
  IF free_sample THEN           \ ja --> freigeben
  open_sample                   \ Datei öffnen
  alloc_sample                  \ Speicher reservieren
  samp_len 4095 16 * um/mod     \ Samplelänge in Blocks zu 4095 Byte ...
  0 ?DO                         \ ... und Paragraphen umrechnen; pro Block
  Samp_offset 4095 16 * Sample  \ 4095 Paragraphen aus der Datei SAMPLE ...
        Samp_segment exhread    \ zur Speicherstelle (Samp_segment) lesen
        4095 16 * = not IF abort" Fehler beim Einlesen!" THEN \ evtll. Fehler?
        samp_segment 4095 + =: Samp_segment  \ Segmentzeiger hochzählen
     LOOP                       \ wiederhole
  dup Samp_offset swap Sample   \ restliche Datei (Blockrest) ...
  Samp_segment exhread          \ ... einlesen
  = not IF abort" Fehler beim Einlesen des Samples!" THEN ; \ geklappt?

\ ------------------------------------------------------------------------
\ Nun der Versuch ein Sample abzuspielen. Auflösung ist 8 Bit, d.h. es sind
\ theoretisch 255 Abstufungen möglich.
\ Für jede Abstufung (im Idealfall ein Voltwert zwischen 0 - 5 V) wird eine
\ Frequenz des Timerbausteins eingestellt, die einer Tabelle entnommen wird.
\ Fürs erste ist das eine linear aufgebaute Tabelle. Die bestimmt nicht zu
\ einer linearen Wiedergabe führt (Verzerrung!).
\ ------------------------------------------------------------------------

70 100 2constant Spreizung

177 Constant max_wert          \ mittels Analyse für das Testsample erhalten
 92 Constant min_wert          \ dito
134 Constant mittel_wert       \ kann man vielleicht mal brauchen

: >spreizung ( n  -- )
  100 ['] spreizung >body 2! ;

: read_wert ( d -- n )          \ liest den (Sample-) Wert an Position d
  16 um/mod                     \ Position in Segment und Offset wandeln
  samp_start + =: samp_segment  \ als Segment und ..
  =: samp_offset                \ Offset speichern
  samp_segment samp_offset c@l  ; \ Wert aus Speicher lesen

Create Werte_tab 512 4 + allot  \ eine Tabelle für die Umrechnungswerte ...
                                \ länger als gebraucht: evtll. für 16-bit

: analyse                       \ liefert den Extremwerte eines Samples
  0 =: max_wert                 \ Maximalwert mit Null vorbesetzen
  255 =: min_wert               \ Minimalwert mit $FF vorbesetzen
  0.60                          \ evntll. *.WAV-Header kühn überspringen
  BEGIN                         \ Endloschleife starten
  2dup                          \ Leseadresse (absolut) kopieren
  read_wert                     \ Samplewert lesen
  dup 0=                        \ ist er Null?
  IF drop                       \ dann den Schrott? ignorieren
  ELSE                          \ ansonsten
```

```
  dup Werte_tab + -1 swap c!    \ Stelle in der Werttabelle markieren
  dup max_wert max =: max_wert  \ größter Wert
      min_wert min =: min_wert  \ kleinster Wert
  THEN                          \
  1. d+ 2dup                    \ Leseadresse um eins erhöhen
  samp_len d>                   \ sind wir schon am Ende des Samples
  UNTIL                         \ Falls ja: Fertig! sonst nochmal.
  2drop                         \ Stack aufräumen
  max_wert min_wert + 2/        \ Mittelwert errechnen
  =: mittel_wert ;              \ und merken

110 Constant traegheit     \ Physikalische Konstante, abhängig vom jeweiligen
                           \ Lausprecher-Computer-System usw.

: tab_lin ( -- )                \ Umrechnung linear zwischen den Extremwerten
  traegheit  100                \ Trägheit der Lausprechermembran
  max_wert min_wert - */        \ zu überstreichender Wertebereich
  >spreizung                    \ anpassen
  Werte_tab 512 erase           \ vorsorglich Tabelle leeren
  max_wert 1+  min_wert         \ für jeden Wert zwischen den Extremen
  DO  I min_wert - Spreizung */  \ entsprechenden Wert errechnen
      I Werte_tab + c!          \ in der Tabelle speichern
  LOOP ;

: tab_zer ( n -- )              \ Umrechnung mit Holzhammer verzerrt
  depth 1 < abort" Bitte einen Verzerrungswert angeben!" \ Parameter da?
  max_wert min_wert rot         \ alte Extremwerte auf dem Stack merken
  max_wert over - =: max_wert   \ neue Werte berechnen ....
  min_wert      + =: min_wert   \     ... und speichern
  tab_lin                       \ Tabelle berrechnen
  =: min_wert =: max_wert ;     \ alte Extemwerte restaurieren

3 Constant speed                \ Wartewert für Pause zwischen den 'Stufen'

: playpwm                       \ spiele ein Sample mit der 'PWM' Methode ab
  speaker_on                    \ Lautsprecher einschalten
  0.0                           \ an Position NULL
  BEGIN                         \ wiederhole (unendlich oft)
  2dup                          \ aktuelle Adresse merken (TOS)
  read_wert                     \ Samplewert lesen
  Werte_tab + c@                \ zugehörigen Wert aus der Tabelle lesen
  timer2                        \ an den Timer-Chip weiterreichen
  speed ms                      \ eine Pause einlegen
  1. d+ 2dup                    \ Position um eins erhöhen
  samp_len d>                   \ Ende erreicht?
  UNTIL                         \ falls ja --> Schleife ende
  2drop                         \ Stack aufräumen
  speaker_off ;                 \ Lautsprecher ausschalten

: playball                      \ Sample ballistisch abspielen
  speaker_on                    \ Lautsprecher einschalten
  0.0                           \ an Position NULL
  BEGIN                         \ wiederhole (unendlich oft)
  2dup                          \ aktuelle Adresse merken (TOS)
  read_wert                     \ Samplewert lesen
  Werte_tab + c@                \ zugehörigen Wert aus der Tabelle lesen
  timer2_flop                   \ an den Timer-Chip weiterreichen
  speed ms                      \ eine Pause einlegen
  1. d+ 2dup                    \ Position um eins erhöhen
  samp_len d>                   \ Ende erreicht?
  UNTIL                         \ falls ja --> Schleife ende
  2drop                         \ Stack aufräumen
  speaker_off ;                 \ Lautsprecher ausschalten
```



```
\ für langsame Rechner hier die Hauptausgaberoutine in Code

2Variable     sample_ende       \ Achtung Format nicht frei!
                                \ Hier wird das Ende des Samples
                                \ mit dem Offset Modulo $7FF0 gespeichert,
                                \ das Segment ist entsprechend anzupassen

code play_sample  ( d -- d1 f n ) \ liest einen Samplewert, spielt ihn ab ...
                  \ ... setzt Dateizeiger neu, hinterläßt ein Flag und ...
                  \ ... eine Kopie des Playwertes
                  \ vormals: 2dup read_wert 2* werte_tab + @
                  \ ....
                  \ 1. d+ samp_len d>
\ ********************************************************************
\ **** Samplewert lesen, wird in AX hinterlegt ***********************
\ ********************************************************************

  BX pop                \                         (Offset)
  DS pop                \ ins Datensegment      (Segment)
  DS CX mov             \ Kopie in CX aufbewahren

  0 [BX] AL mov         \ Byte lesen
  CS DX     mov         \ Forth-Code-Segment via ...
  DX DS     mov         \ ... DX restaurieren, d.h. DS=CS
  AH AH     xor         \ High-byte löschen
\ ********************************************************************
\ **** Ofsett und Segmentadresse aktualisiert, liegen auf dem Stack ***
\ ********************************************************************
  BX        inc         \ BX (Offset) um eins erhöhen
  $7FF0 # BX   cmp      \ vieviel Bytes aus aktuellem Segment gelesen?
  0= IF                 \ wenn $7FF0 Bytes gelesen ...
       0 # BX mov       \ BX (Offset) auf Null setzen und
    $7FF # CX add       \ CX (Segment) entsprechend erhöhen
    THEN                \
  CX        push        \ neues Segment und ...
  BX        push        \ ... neues Offset zum TOS
  BX DX     mov         \ Offsetkopie zur späteren Verwendung
\ ********************************************************************
\ **** Überprüfen, ob das Ende Samples erreicht ist, Flag zum Stack ***
\ ********************************************************************
  sample_ende #) BX mov \ Segment des Sampleendes holen
  CX BX cmp             \ mit akt. Segment vergleichen
  0= IF                 \ falls es gleich ist ...
    sample_ende 2+ #) CX mov    \ Offset des Endes holen
    DX CX cmp           \ mit akt. Offset vergleichen
    0= IF               \ falls es gleich ist ...
      BX BX mov         \   ---\
      ELSE false # BX mov \       \ ____  Flag je nachdem in BX
      THEN              \       /
    ELSE false # BX mov   \  ---/
    THEN                \
   BX push              \ Flag zum TOS
\ ********************************************************************
\ ****  mit CPU-Befehl XLAT, Wert in benötigten Wert übersetzen *******
\ ********************************************************************
  Werte_tab # BX mov    \ Startadresse der Wertetabelle holen
                xlat    \ "Übersetzen"
  AX        push        \ Sample-Byte zum Stack
  AX        push        \ (Kopie zur weiteren Verwendung)
\ ********************************************************************
\ *** Timer "ballistisch" programmieren, es wird nur ein Byte gebraucht
\ ********************************************************************
  %10010000 # AL mov    \ Zähler 2, low Byte, Modus1, Hex.
```

```
  Steuer_reg # AL out   \ in das Steuerregister schreiben
  AX pop                \ Zählerwert vom Stack nach AX holen
  Timer2_port # AL out  \ Lowbyte schreiben
  next end-code         \ fertig

defer dazwischen        \ in der Wartepause kann evtll. etwas getan werden
' drop is dazwischen    \ vorerst wird der Sampleplaywert verworfen

560 Constant (speed     \ Wartewert für die Codeversion

: playcode              \ spielt ein Sample im ballistischen Modus
  speaker_on            \ Lautsprecher ein
  samp_len $7FF0 um/mod $7ff * \ Samplelänge in 'Blocks' umrechnen
  samp_start + sample_ende 2!  \ zum Start addieren: Sampleende merken
  samp_start 0          \ akt. Segment und Offset setzen
  BEGIN                 \ Start der 'Endlosschleife'
  play_sample  dazwischen \ Wert_spielen und in der Pause etwas tun
  (speed 0 ?DO LOOP     \ warten
  UNTIL                 \ Wiederhole, flag von play_sample auswerten
  2drop                 \ Stack aufräumen
  speaker_off ;         \ Lautsprecher aus

\ ************************************************************

: testball ( -- ) load_sample playball ;
: testpwm  ( -- ) load_sample playpwm ;
: testcode ( -- ) load_sample playcode ;

load_sample analyse tab_lin

\ ____________________________________________________________________
\ Bei meinem Gehör und bei meiner Hardware ist die ballistische Methode die
\ mit den besten Ergebnissen. Das Pfeifen, das durch die Diskretisierungs-
\ schritte entsteht, ist bei einem Speedweert von 3 (eine Abtastrate von ca.
\ 11 KHz) für mich nicht mehr hörbar - für einige Schüler allerdings doch.
\____________________________________________________________________
```

Forth-Gruppen regional

Moers
Friederich Prinz
Tel.: 02841-58398 (p) (Q)
(Bitte den Anrufbeantworter nutzen !)
(Besucher: Bitte anmelden !)
Treffen: (fast) jeden Samstag,
14:00 Uhr, MALZ, Donaustraße 1
47443 Moers

Mannheim
Thomas Prinz
Tel.: 06271-2830 (p)
Ewald Rieger
Tel.: 06239-8632 (p)
Treffen: jeden 1. Mittwoch im Monat
Vereinslokal Segelverein Mannheim e.V.
Flugplatz Mannheim-Neuostheim

München
Jens Wilke
Tel.: 089-89 76 890
Treffen: jeden 4. Mittwoch im
Monat, China Restaurant XIANG
Morungerstraße 8
München-Parsing

µP-Controller Verleih

Thomas Prinz
Tel.: 06271-2830 (p)

Gruppengründungen, Kontakte

Fachbezogen 8051 ... (Forth statt Basic, e-Forth)
Thomas Prinz
Tel.: 06271-2830 (p)

Forth-Hilfe für Ratsuchende

Forth allgemein Jörg Plewe
Tel.: 0208-49 70 68 (p)

Jörg Staben
Tel.: 02103-24 06 09 (p)

Karl Schroer
Tel.: 02845-2 89 51 (p)

Spezielle Fachgebiete

Arbeitsgruppe MARC4	Rafael Deliano Tel./Fax: 089-841 83 17 (p)
FORTHchips (FRP 1600, RTX, Novix)	Klaus Schleisiek-Kern Tel.: 040-375 008 03 (g)
F-PC & TCOM, Asyst, emb.Contr., Fuzzy...	Arndt Klingelnberg Tel.: 02404-6 16 48 (p) (g) (Q)
KI, Object Oriented Forth, Sicherheitskritische Systeme	Ulrich Hoffmann Tel.: 04351 -712 217 (p) Fax: -712 216
Forth-Vertrieb	volksFORTH / ultraFORTH RTX / FG / Super8 / KK-FORTH Ingenieurbüro Klaus Kohl Tel.: 08233-3 05 24 (p) Fax : 08233-99 71
Forth-Mailbox (KBBS)	0431-533 98 98 (8 N 1) Sysop Holger Petersen hp@kbbs.org Tel.: 0431-533 98 96 (p) bis 22:00 Fax : 0431-533 98 97 Helsinkistraße 52 24109 Kiel

Möchten Sie gerne in Ihrer Umgebung eine lokale Forthgruppe gründen, oder einfach nur regelmäßige Treffen initiieren ? Oder können Sie sich vorstellen, ratsuchenden Forthern zu Forth (oder anderen Themen) Hilfestellung zu leisten ? Möchten Sie gerne Kontakte knüpfen, die über die VD und das jährliche Mitgliedertreffen hinausgehen ?

Schreiben Sie einfach der VD - oder rufen Sie an - oder schicken Sie uns eine E-Mail !

Hinweise zu den Angaben nach den Telefonnummern:

Q = Anrufbeantworter
p = privat, außerhalb typischer Arbeistzeiten
g = geschäftlich

Die Adressen des Büros der Forthgesellschaft und der VD finden Sie im Impressum des Heftes.

Jahrestagung der Forthgesellschaft

24. - 26. April 1998

Hotel Dampfmühle
Krefelder Straße 9
47506 Neukirchen-Vluyn

Haben Sie sich schon angemeldet ?