VIERTE DIMENSION

3/1996

12. Jahrgang 1996, 3. Quartal, DM 10.-

- Kurzbericht über die Echtzeit 96
- Ein Rahmen für modular aufgebaute Forth-Systeme
- Forth-Server bei Forth-Host-Target-Interfaces
- 32-Bit 386-Prefix-Assembler mit AT&T Syntax
- Serie PC-Meßtechnik, Teil V: Der Joystick
- Forth International
- Who is Who? What is TOS?
- Ini-File für WinView.exe
- Wenn alle Brünnlein fließen
- XMODEM
- Ein heiliger Krieg - das THEN im Forth
- Forth Online

FORTH MAGAZIN

Organ der Forth Gesellschaft e.V.

Dienstleistungen und Produkte von Forthlern und/oder für Forthler (Anzeige)

Möchten auch Sie oder Ihre Firma hier aufgeführt werden? Bitte wenden Sie sich an die Anzeigenverwaltung (s. Impressum).

Ihre Anzeige plus 3 Zeilen je 45 Zeichen Text kosten 90.-DM (incl. 20.-DM Einrichtung/ Änderung, je Zusatzzeile 10.-DM) und das komplett für ein ganzes Jahr.

Dienstleistungen und Produkte von Forthlern und/oder für Forthler (Anzeige)

IMPRESSUM

Name der Zeitschrift

FORTH MAGAZIN - VIERTE DIMENSION
Organ der Forth-Gesellschaft e. V.

Herausgeberin

FORTH-Gesellschaft e. V.
Postfach 1110
85701 Unterschleißheim
Tel./Fax: 089 / 3173784
Mail: secretary@admin.FORTH-eV.de

Redaktion & Layout

Claus Vogt
Katzbachstr. 23
Tel.: 030 / 786 84 60 (Fax & B
Mail: vd@FORTH-ev.de

Neue Adresse ab Juli '96

Anzeigenverwaltung: Ulrike Schnitter c/o Forth-Ges.; PF 1110; 85701 Unterschleißheim.
ANS-Forth: Ulrich Hoffmann; uho@informatik.uni-kiel.de; Sehestädter Str. 26; 24340 Eckernförde.
Forth international: Fred Behringer; Planegger Str. 24; 8124 1 München.
Zeichnungen: Rolf Kretzschmar;
Titelbild: Hans-Georg Schmid

Redaktionsschluß '96

Erste Januar-, April-, Juli-, und Oktoberwoche.

Erscheinungsweise

Viermal im Jahr.

Preis

Einzelpreis: DM 10,-

Von Sommer- und anderen -löchern

Im Sommer ist im Baubereich und in der Landwirtschaft Hochkonjunktur. Urlauber räkeln sich in der Sonne. Nur die Presse hat Saure-Gurken-Zeit. Eigentlich. Dieses Jahr war es etwas anders. Gehetzte Urlauber fliehen vor Gewitterwolken, der Baubereich ist kurz vor Schlechtwetter und die Journaille hat den Mund voll mit vergiftetem Rindfleisch und rekordverdächtigen Arbeitslosenzahlen, gekürzten Sozialleistungen und nicht zuletzt mit dem schlechten Wetter und natürlich der Olympiade.

Das Forthmagazin ist völlig unsportlich, unpolitisch und hat auch kein geeignetes Programm für's Wetter zu bieten. Obendrein ist es grade umgezogen und residiert in zentralen Teilen immer noch in staubigen Umzugskartons. Dafür hat es sich in weiser Vorraussicht einen 'Sommerspeck' angesammelt, so tritt es auch mitten im Sommerloch mit feisten 40 Seiten an. Vielleicht eine kleine Entschädigung für verregnete Programmiererurlaube und gestrichene Staatsknete.

Nach der Forthtagung im April und der 'Echtzeit' im Juni, auf der Winfried Clemens die Forth-Gesellschaft dieses Jahr vertreten hat (vielen Dank dafür), wird das Kongreßjahr merklich dünner. Die französischen Nachbarn schicken sich an, dieses Loch zu füllen und laden uns zum Januar nach Paris ein. Das gibt uns Gelegenheit, zum Abschluß eine just gelesene Notiz von Alfred Kerr von 1929 zu zitieren:

"Ich rate jedem, der sich schlecht fühlt, eine Reise nach dieser Stadt zu machen. Weil das allgemeine Empfinden, weil das Glücksgefühl dort gehoben wird. Eine Reise solcher Art ist sehr zu empfehlen; mitten im Winter; für wenige Tage; plötzlich. Ohne jemandem was mitzuteilen steigt man in die Bahn, nachdem man im Zustande der Erleuchtung ein paar Sachen in den Koffer geschmettert hat. Es ist sehr schön, mitten im Winter von Mittwoch bis Freitag in Paris zu sein."

- Claus Vogt, Editor des Forth Magazins Vierte Dimension -

Sie schenken uns einen Artikel..

... da möchten wir Ihnen auch ein kleines Präsent überreichen

Jeder Autor eines Artikels hat einen Wunsch frei!

Einige Geschenke sind schon gebraucht, aber gut erhalten - manchmal längst vergriffen und damit richtige Raritäten. Sollte man ein Geschenk mal nicht in sein Herz schließen können, kann man es ja wieder spenden.

Jupiter ACE

FORTH Programm-Handbuch
S. Vickers 1983, deutsch, 159 Seiten

Das Handbuch des einzigen Homecomputers, der FORTH als Betriebsystem hatte. Für Minimalisten und Oldtimer.

- gestiftet von Rafael Deliano

... aus Nebaj

Schal 2m lang, rot und gelb

Wer Nebaj nicht kennt, sollte es kennenlernen. Als Vorgeschmack gibt's einen Schal. Die Muster und Farben der Kleidung ist typisch für jedes Dorf des guatemaltekischen Hochlandes.

- gestiftet von Holger Dyja

Steve Jobs

J.S.Young 1988, deutsch, 495 Seiten

Von der Presse zum Kultbuch hochstilisiert, ist es jedenfalls ein sehr gut geschriebenes Buch. Beleuchted die Microcomputerbranche zwischen 1975 und 1985 aus der Sicht von Apple.

- gestiftet von Rafael Deliano

Das kleine Zeitgeistpaket

Wer den Timer des PC mit der Trial-And-Error-Methode bezwingen möchte, braucht dieses Geschenk nicht. Doku-Freunde finden hier: Timer-Routinen des AT-Rom-Bios, 8254-Datenblatt, Register der RT-Clock...

- gestiftet von Claus Vogt

Programmers at Work

S.Lammers 1986, englisch, 385 S.

Enthält Interviews, die Lammers mit 20 Top-Programmierern geführt hat. U.a. Gates, Bricklin, Hertzfeld, Kildall, Page, Warnock, Raskin,... Interessanter Einblick in die Innereien der PC-Softwareindustrie der 80er. Sowie in Ansichten und Arbeitstechnik derer, die Erfolg hatten. Auffällig wieviele von ihnen mal FORTH programmiert haben.

- gestiftet von Rafael Deliano

The Macintosh Way

Guy Kawasaki 1988 englisch 209 Seiten

Der Inhalt befaßt sich mit Marketing von Standardsoftware für Tischcomputer. Lesenswert wird es durch seine umfassende Sammlung industriespezifischer Jokes.

- gestiftet von Rafael Deliano

Vom Mythos des Mann-Monats

F.P.Brooks 1972 (1987), dtsch, 165 S.

Der Titel wurde zum Schlagwort. Das Buch wirkt auf den ersten Blick antiquiert, weil sich der Autor auf Erfahrungen bei IBM aus den 60er Jahren stützt. Die Probleme der Softwareentwicklung haben sich jedoch kaum geändert.

- gestiftet von Rafael Deliano

''Fisch''

Linolschnitt 21 x 17 cm

Aus der Hand des Alteditors stammt dieses Kunstwerk.

- gestiftet von Rolf Kretzschmar

F-PC - Das Buch zum Forth

Das F-PC stammt wie das ZF aus der Hand von Tom Zimmer und hat der Forth-Gemeinde neue Dimensionen geöffnet. Vom Forth-Zweizeiler bis zur Optimierung der *win.ini* kann hier jeder was lernen.

- gestiftet vom Autor Jörg Staben

Forth lernen mit dem ZF

Das ideale Geschenk für Anfänger

Friederich Prinz hat ein wirklich schönes Päckchen gepackt. Es enthält das ZF, ein Forth sich gut für Anfänger mit DOS-Erfahrung eignet. Dazu zahlreiche Aufsätze, die sich in den Anfängerkursen der Moerser Forthgruppe bewähren mußten.

- gestiftet von Friederich Prinz

Forth-Magazin VD

von hohem archivarischen Wert

.. einige längst verschollen geglaubte Jahrgänge dieses Blattes sind bei jüngsten Ausgrabungen im Keller des Forth-Büros gefunden wurden. InteressentInnen sollten die gewünschte Nummer angeben.

- gestiftet vom Forth-Büro

Rubriken

Inserenten

ANS Forth

Forth-Gesellschaft

Forth-Systeme

Prozessorgeflüster

Forth inside

Firmen

Forth online

Zeitschriften

Was fehlt

Die Rubriken Bücher, Anfänger, Produktinfo, 'Was noch' und die Kolumne ANS Forth

1	31.383	28.02.96	17:44	assemble\assemble.f
2	3.596	5.04.96	0:00	modular\coldbye.cfw
3	3.157	27.07.96	14:40	transput\screened.fth
4	12.404	25.05.96	10:33	tropfen\dropplay.seq
5	10.056	25.05.96	10:33	tropfen\tropfen.seq
6	4.513	7.06.96	20:49	wasser\new_numb.seq
7	7.239	23.07.96	18:40	wasser\sinus.seq
8	25.216	7.06.96	20:49	wasser\vga_emit.seq
9	22.020	27.07.96	12:01	wasser\wasservd.seq
10	3.335	24.01.94	15:47	whoiswho\fastinfo.seq
11	3.029	28.07.96	17:21	winview\winv-ini.f
12	7.806	7.05.96	12:33	xmodem\xmod.lst
13	10.549	23.07.96	19:19	xmodem\xmod.src

Anzahl Dateien: 13 Anzahl Bytes: 144.303

= *Listing auf Diskette*

Leserforum

Internationales Echo ...

Hallo Friederich,

The 2/1996 VD came yesterday. I read your article on Holon, and then Joerg's article on the Jahrestagung. You must be the one up front in the picture, with the sunglasses, but without a cigarette. Is the tall one standing in the back Joerg Ploewe? Can't be as tall as Schleisiek, though?

... I do not quite understand what this Holon is. Where does the name come from? No more screens, no files, only words? I wonder whether anyone has uploaded HOLON-LT onto GEnie. If so, I'll definitely want to look at it.So far I am not equipped yet to browse the WWW. I tried to sign up on CompuServe, but canceled the subscription after wasting several evenings with its front-end (WINCIM) and browser (Mosaic) interfaces.

I wish I had more time for Forth. I still need to develop a multitasking program that plays music together with my stepper motor demo (in other words, beeps the PC's speaker while bits are going out of the parallel port). It is possible that I get a third invitation to the annual Embedded Systems Conference this coming fall. I would like to improve my demo for it.

Speaking of pictures, were you in the group picture in the 1994 Tagung in Malente? You are right--your group has fewer members coming to the yearly Forth days than the Silicon Valley FIG has in the monthly meetings.

But I like the contents of your magazine much better than ... . My compliments to the group that puts the Vierte Dimension together!

Until next time, Henry
v.vinerts@genie.com, Juni96

Reaktionen auf VD 1/96

Übrigens ist die letzte VD, die inzwischen bei mir eingetroffen ist, wieder gut geworden. Weiter so ;-)...

tb@tb.forth-ev.de (Thomas Beierlein), Mai96

Das Titelbild ist gut gewählt und der Begleittext hat die richtigen Proportionen. Farbe macht sich natürlich immer besser als s/w. Jedoch kommt das Titelbild nicht voll zur Wirkung, weil die Reste des alten Covers zu aufdringlich wirken. Wenn schon Neuerungen, dann konsequent durchgreifen.

Die Mischung von News und Inhaltsangabe auf Seite 4 und 5 halte ich für nicht optimal. Besser wäre es, die gesamten Inhaltsangaben auf Seite 5 zu konzentrieren.

Schlau gemacht, den Rand vom Foto der "Canon Cat" (Seite 25) wegzuschneiden.

Wahrscheinlich die einzige Möglichkeit das widerspenstige Farbfoto reproduzierbar zu machen.

Das Foto hatte auch die richtige Größe. Ein optischer Aufhänger muß dem Leser ins Gesicht springen, wenn er die Seite aufschlägt. Negativbeispiel auf Seite 28: das Bild ist viel zu klein und wird nur noch als eckiger Schmutzfleck wahrgenommen.

Die Zusammenfassung aus c.l.f und d.c.l.f fehlte leider in dieser Ausgabe.

Rafael Deliano (jrd) April'96

zu Peter Knaggs 'Forth auf eine professionelle Basis heben', VD 1/96, Seite 19

Der Einschätzung, daß FORTH derzeit ein mieses Image hat, kann ich nur beipflichten. Wenn Knaggs an seiner Universität öffentlich für FORTH eintritt, schädigt er damit seine beruflichen Aussichten. Goppold würde dem wohl beipflichten. Knaggs bürokratische Ideen, FORTH als Loge aufzuziehen, scheitern hauptsächlich an den finanziellen Gegebenheiten. Weiterbildung, (schlagkräftige) Organisationen und Tagungen kosten Geld. Solange niemand mit FORTH Geld verdient, wird niemand für FORTH Geld ausgeben. Und in dem Augenblick wo mit FORTH Geld verdient wird, hat es auch ein positives Image. Money talks. Erst dann wäre die von Knaggs gewünschte Bürokratisierung durchführbar. Ist dann aber nicht mehr nötig.

Rafael Deliano (jrd) Mai'96

zu Fred Behringer 'Forth braucht keinen qualitätsüberwachenden Fachverband', VD 1/96, Seite 21

Hobbyisten sind leider sehr naiv. Es dürfte sehr schwierig werden, den Franzis-Verlag dazu zu bewegen, die Bücher von Zech nochmal aufzulegen. Der Lektor sieht in die Zeitschriften, stellt fest, daß keine FORTH-Compiler angeboten werden und keine Artikel zu FORTH erscheinen und vermutet nicht zu Unrecht, daß FORTH kaum noch Anwender hat und er auf den Büchern sitzenbleibt.

Aus den gleichen Gründen wird sich auch kein Leiter einer Bibliothek exotische FORTH-Zeitschriften ins Regal stellen: er kann nicht davon ausgehen, daß es Leser für sie gibt.

Auch die Redakteure der ELRAD veröffentlichen Artikel mit FORTH-Source nur sehr widerwillig. Ihnen wäre 6502-Assembler oder C beträchtlich lieber. Sie haben Recht. Die Masse der Leser kennt FORTH nicht und kann mit den veröffentlichten Listings nichts anfangen. Die Redakteure werden auch keine Artikel über Hobby-PD-Forths veröffentlichen. Sie müssen zusehen, daß genügend Anzeigen geschaltet werden. Der Schwerpunkt liegt deshalb auf der Besprechung kommerzieller Compiler, denn nur für die wird Geld in Werbung investiert.

Einige Programmiersprachen wie COBOL, Fortran und Ada haben stabile / senile Benutzergruppen die sie am Leben erhalten. Die meisten anderen brauchen jedoch Artikel in Zeitschriften und Bücher damit sie dem Publikum bekannt werden. Damit man sie verwenden kann, braucht es Compiler, die ein sinnvolles Preis/Leistungsverhältnis haben, dokumentiert und benutzbar sind. Ferner ist Werbung in den Zeitschriften nötig, die die Brücke zwischen Anwender und Hersteller schlägt und dem Publikum gegenüber die Glaubwürdigkeit von Anbieter und Sprache unterstreicht.

Ohne dieses Umfeld kann kaum eine Programmiersprache auf Dauer existieren. Alle Komponenten dieses Umfelds bedingen sich gegenseitig. Wenn die Programmiersprache in der Öffentlichkeit unbekannt ist, kann niemand Compiler verkaufen. Ohne Compiler gibt's nicht genügend Anwender. Und damit keinen Grund sekundäre Produkte wie Literatur anzubieten.

Dieses Umfeld hat für FORTH nie sehr üppig existiert, bzw es existiert augenblicklich nicht mehr. Einigen Programmiersprachen wie z.B. PILOT oder MUMPS ging es ähnlich und sie fristen deshalb ihr Dasein nur noch als Fußnote in Lexikas.

Eine Sprache verbreitet sich entweder nach den Gesetzen der Marktwirtschaft (= Darwinismus) oder sie siecht dem Exitus entgegen. Für FORTH gilt derzeit letzteres.

Der einzelne Hobbyist macht da keinen Unterschied. Denn konstruktive Arbeit, nicht lautstarker Enthusiasmus wären nötig.

Auch die FORTH-Gesellschaft macht keinen Unterschied. Sie hat sich nicht vom Hobbyismus lösen können und wird deshalb von der Mehrzahl der ernsthaften Anwender von FORTH gemieden, weil schlechtes Image abfärbt.

Rafael Deliano (jrd) Mai'96

Reaktionen auf
VD 2/96

Leserbriefe:
Am liebsten kurz.Sonst trifft uns die Pflicht zur Kürzung.
Die Redaktionsadresse lautet:

Forth-Magazin 'Vierte Dimension'
Claus Vogt, Ebersstr.10,
D-10827 Berlin, vd@FORTH-ev.de

Forth. Kurz und Knapp. Das

VD 2/96 war ziemlicher Durchschnitt. Habe wenig gefunden, was ich kritisieren könnte, aber auch nichts, was eine Verbesserung wäre.

Der PI-Artikel war gut geschrieben und illustriert, hatte aber ein unergiebiges Thema.

Beim SPS-Artikel wurde ein gutes Thema verstoppselt. Mehrere gute Zeichnungen wären wohl nötig gewesen, die Grundfunktion einer SPS dem Leser nahezubringen.

Bei meinen beiden Artikeln sind die Illustrationen gegenüber der Textfläche zu klein gewesen, weil du den Text großräumiger als üblich verteilt hast. Hatte natürlich den Vorteil, daß man den Text ohne Augenschmerzen lesen konnte. Bei Behringers folgender Seite ist der Text wieder VD-üblich gestaucht worden.

Rafael Deliano (jrd), Juni96

Die VD 2/96 ist wieder ausgezeichnet. Eine gute Mischung aus Handfestem und vielen kleinen nützlichen Informationen.

Fred Behringer (beh)

Hallo Claus,

Du wolltest “was hören” zur neuen VD...

Das Titelblatt wirkt wieder etwas “grau”, ist aber sehr informativ.

Martin Bitters Beitrag zum Tropfenalgorithmus kannte ich natürlich schon aus seinem “life”-Vortrag hier in Moers. Dazu würde mich sehr interessieren, wie solche “spielerischen Leckerbissen” bei anderen Lesern der VD ankommen. Es gibt viele “ganz praktische” Lösungen zu relativ komplexen, mathematischen “Philosophien”, die die Mathematik interessant machen, wie z.B. den Kreisalgorithmus. Ich würde gerne mehr davon sehen.

Einfach SUPER fanden meine Frau und ich Michael Kalus’ Bericht über den Drachenrat. Ich würde ihm das gerne selbst schreiben, kenne aber seine E-Mail Adresse nicht. Vor allem interessiert uns, welche Fantasy Autoren er außer dem Zech bevorzugt ;-)

Klaus Kohls Artikel über die RS232 findet hier in Moers als “Fortsetzungsroman” von Anfang an großen Anklang. Christel, die als Lehrerin immer nach Literatur sucht, die ihr hilft, ihren Schülerinnen und Schülern einfache Dinge wirklich einfach zu erklären, kopiert sich Klaus’ Artikel jedesmal. Klar, daß ich hinter Thomas Beierleins Beitrag stehe und den jederzeit unterschreibe ;-)

Glückauf
der Fritz
F.PRINZ@MHB.gun.de (Friedrich Prinz), Mai96

zu 'Steter Tropfen höhlt den Stein' in VD 2/96, Seite 12

Martin Bitter hat in seiner “Rätsellösung” in der VD 2/96 einiges über das Stellenprinzip und die Zahlendarstellung mit veränderlicher Basis geschrieben. Seine hornerschema-artige Schreibweise eignet sich hervorragend zur Verdeutlichung des Begriffs der stellenabhängigen Basis. In seinem “Vorspiel” hat er allen (also auch mir) die Frage gestellt, ob in der VD “Platz für Algorithmen” sei. Ich habe seinen Artikel gelesen und finde ihn gut. “Unbedingt”, lautet meine Antwort. Nichts ist praktischer als eine gute Theorie, und ich würde in den zukünftigen Heften der VD gern mehr zu diesem Thema lesen.

Natürlich drängen sich sofort ein paar Fragen auf: Konvergenz, Eindeutigkeit, Ausschöpfung der gesamten Menge der reellen Zahlen (“Basis”!), zulässiges Ziffernmaterial (sollte auch 1,123456789A... bei Zehnerbasis erlaubt sein?), Zuwächse kleiner als 1 bei stellenabhängiger Basis (?) usw.

Hier eine kleine Sammlung von Anmerkungen, die mir unmittelbar einfallen: e = 2,11111... bei Basis $(n+1)^{-1}$ ergibt sich aus der Taylorentwicklung der Exponentialfunktion um 0 für x=1. cosh(1) = 1,101010... entsprechend. Desweiteren gilt (siehe Taylorentwicklung) sinh(1) = 1,010101... bei derselben Basis. Soll man auch cosh(1) = 0,1101010...und sinh(1) = 0,101010... bei Basis n^{-1} schreiben dürfen? Oder sinh(1) = 1,101010... bei Basis $(n+2)^{-1}$? Noch schnell zur Eindeutigkeit: 1,1105 = 1,1110 und 1,03 = 1,10 bei Basis $(n+1)^{-1}$.

Bei der von Martin Bitter so benannten Basis $(n+1)^{-1}$ hätte ich eigentlich (n+1) erwartet, so wie man von der Basis 10 (Dezimalbasis) spricht, und nicht etwa von 10^{-1} (siehe oben). Aber das sind natürlich untergeordnete Fragen der Bezeich-

Fortsetzung auf Seite 8 ...

Korrekturen /Ergänzungen

zu 'Steter Tropfen höhlt den Stein' in VD 2/96, S. 12

In dem Artikel wird auf eine Literaturstelle verwiesen, die im Quelltext TROPFEN.F untergebracht ist. Da das Listing aufgrund seiner Länge nicht mit abgedruckt wurde, hier der Literaturhinweis, in dem sich auch weitere interessante Quellenangaben befinden:

Ian Stewart: Mathematische Unterhaltungen. In: Spektrum der Wissenschaft. Heidelberg 12/1995 Seite 10-15.

Im zweiten Kasten ist eine Formel verzerrt wiedergegeben. Richtig muß es heißen:

$$4*\frac{1}{10}+5*\frac{1}{100}+6*\frac{1}{1000}$$

Die Spaltenüberschriften beim Rechenschema zur Ermittlung von PI lauten richtig: 1/10 , 1/3, 2/5, 3/7, 4/9, 5/11, 6/13, 7/15, 8/17, 9/19, 10/21.

Der zum Schluß des Artikels erwähnte Quelltext zum Spielen DROPPLAY.SEQ sowie TROPFEN.SEQ (eine leicht überarbeitete Version von TROPFEN.F zur Darstellung von PI, das auf der VD-Listingdiskette 2/96 enthalten war) finden sich in der MHB 02841/57325 in dem Brett SUPPORT/FORTH, in der KBBS und auf der **VD-Listingdiskette 3/96**.

Martin Bitter

zu 'HOLON - Das ganz andere Forth' in VD 2/96, S.33

Aufgrund praktischer Probleme bei der Drucklegung erreicht der zughörige Screendump leider erst jetzt die LeserInnen. Dadurch wird wohl auch der Text des Artikel verständlicher:

In drei Spalten werden die bereits vorhandenenen Definitionen, ebenso wie alle dazukommenden, zunächst auf MODULS aufgeteilt. In der Modulspalte finden sich Begriffe wie BASIS, FORTH, DOS&EXT, TASKING usw.. Applikationen beginnen mit einem neuen Moduleintrag, z.B. HALLO.

In der Spalte GROUPS werden die Module sinnfällig in Gruppen unterteilt. Für das Modul FORTH existieren Gruppen wie 16-Bit, ... ,FormattedOutput usw.. Und erst diese Gruppen enthalten die einzelnen Worte, bzw. Definitionen - so wie sie in anderen FORTH-Systemen gnadenlos ungeordnet nach dem Aufruf von WORDS über den Bildschirm huschen.

Zwischen Modulen, Gruppen und Worten schaltet der Entwickler mit den Cursortasten hin und her und kann sich auf einfache und effiziente Weise gezielt beim System nach einzelnen Definitionen erkundigen. Welches Wort in welcher Gruppe und in welchem Modul gerade in dem darunter befindlichen Editfenster angezeigt wird, läßt sich einfach an der andersfarbigen Ausgabe (gelb statt weiß) von Wort, Gruppe und Modul ablesen. Selbstverständlich stellt HOLON dabei alle seine Definitionen bis in den Kern hinein zur Verfügung, so daß auch dem Wißbegierigsten keine Fragen offen bleiben. Und selbstverständlich ist alles zu jeder Zeit veränderbar. Metakompilation ist eine natürliche Fähigkeit von HOLON !

Editfenster und Kommentarfenster sind zwei voneinander getrennte Bereiche auf HOLONs Oberfläche. Dabei kann man den schmalen, für Kommentare vorgesehenen Bereich aber eigentlich nicht als Fenster bezeichnen, bestenfalls als Oberlicht.

Die Redaktion entschuldigt sich bei allen Geschädigten /clv

nung, die an der Sache selbst nicht rütteln.

In der Literatur wurde die von Martin Bitter verwendete Basis $(n+1)^{-1}$ als "factorial base" bezeichnet: "Error-free Fractions" von Peter Wayner in Byte 6/88. Wie Wayner erwähnt, hat schon Georg Cantor nachgewiesen, daß man in der "factorial base" (Fakultätsbasis(?)) alle rationalen Zahlen in endlicher Form darstellen kann. Wayner liefert in seinem Byte-Artikel einen Beweis.

Auf Forth bezogen erscheint dieses Thema als "Variable-Base-Arithmetic" von S.Y. Tang und wurde als solches in F-PC 3.5 aufgenommen. Ich habe das diesbezügliche Programm in Turbo-Forth überprüft: Wenn man CC (beispielsweise durch CCC) ersetzt, da Turbo-Forth CC anderweitig verwendet, so funktioniert alles.

Ich fasse zusammen: Ich bin sehr daran interessiert, in der VD mehr über diese Dinge zu lesen. Weiter so, Martin Bitter!

Fred Behringer, München

zu 'Forth und der Anfänger' in VD 2/96, Seite 31

Ich bin ein bekennender Anfänger im Sinne des Artikels von Thomas Beierlein in der VD 2/96 auf den Seiten 31f und versuche, FORTH für ein privates Projekt zu erlernen. Tatsächlich jedoch fehlt es mir an einem geeigneten und auf meine mageren Kenntnisse wohlwollend nachsichtig reagierenden System sowie an Quellen, die meinen Lernprozeß unterstützen oder ihn sogar erst ermöglichen. In Sachen ANS-Forth ist die Lage mit der Literatur nur als "desolat" zu bezeichnen und anderes Material habe ich noch nicht lokalisiert.

Das heißt zusammengefaßt: Die Forderung, neue Leute ("frisches Blut") in die Forth-Gesellschaft zu bringen, neue Anhänger dieser beeindruckenden Sprache zu gewinnen und damit die Sprache selbst am Leben zu erhalten ist im Augenblick nur eine Absichtserklärung, der noch keine Taten gefolgt sind. Wie kann man diesem Zustand abhelfen? Ich denke, es sollte ein Kurs konzipiert werden, der die im Artikel beschriebene Linie verfolgt und der in der VD veröffentlicht wird. In diesem Zusammenhang wäre es auch von Vorteil zu erfahren, wieviele der Mitglieder sich nach eigener Einschätzung als "Einsteiger" bezeichnen würden und ob sie glauben, dass ein solcher Kurs hilfreich sein könnte. Dieser Kurs könnte durch den Dialog mit eben jenen Anfängern sogar um Beispiele, spezielle Probleme etc. erweitert werden und abschließend in gesonderter und gebundenen Form zusammengefasst über die üblichen Kanäle bezogen werden. Dies erschlägt nämlich zwei Probleme:

- Wenn auch nur in geringem Umfang wäre ein Stück Literatur verfügbar, das geeignet sein sollte, den reinen Text des Standards z.B. mit Gehalt anzureichern, so dass dem Einsteiger ein Überblick über und eine Bewertungsgrundlage für die Sprache Forth gegeben ist.

- Das Erlernen der Sprache selbst kann reibungslos und mit moderatem Widerstand ermöglicht werden. Insbesondere die erwähnten Beispiele könnten hilfreich sein; wie oft findet man Lehrbücher ohne dieses machtvolle Werkzeug?

Das ganze ist natürlich nur ein Vorschlag und ich weiß, daß es bei der Realisierung eines solchen Planes eo ipso eine Menge Probleme gibt. Ich glaube jedoch auch, insbesondere aus der Situation heraus, in der ich nun als Anfänger einmal bin, daß gerade jetzt und gerade in der Forth-Gesellschaft dieses Problem angegangen werden kann und auch muß.

os@cs.tu-berlin.de (Olaf Stoyke), Juni'96

zu 'C auf Stackprozessoren' in VD 2/96, Seite 36

Rafael Deliano behauptet in der VD, daß es offen sei, ob C auf Stackprozessoren überhaupt effizient verarbeitet werden könne. Dem muß widersprochen werden. Denn Stackprozessoren beschränken sich nicht nur auf das Forth-Feld, auch wenn umgekehrt Forth-Prozessoren natürlich immer Stackprozessoren sind.

Stackprozessoren für Algolartige Sprachen haben eine lange Tradition, die ersten (von Borroughs) gab's schon vor der Zeit von Forth. Die waren allerdings nicht für C, sondern für Algol gebaut. Von HP gab's sogar Workstations mit Stack-Prozessoren (die 3000er Reihe), natürlich auch für C. Der Transputer ist eigentlich auch ein Stack-Prozessor und der konnte ebenfalls C (obwohl man ihn für Occam gebaut hat). Und dieser ganze Java-Hype dreht sich nicht um eine Forth-artige Sprache, sondern um eine C-Variante, die auf einem Stackprozessor läuft. Obwohl Sun so tut, als sei alles ganz neu, gibt's sogar schon Prozessoren, die für Java wie geschaffen sind (der Hobbit von AT&T etwa), und die von Anfang an C konnten.

Will sagen: Die einzigen Stackprozessoren für die's kein (oder nur ein lahmes) C gibt sind die für Forth. Das legt zwei Vermutungen nahe: Entweder sind die Forther so arrogant und boykottieren C mit Absicht, oder sie haben etwas eingebaut (oder ausgebaut), was C behindert. Wahrscheinlich trifft beides zu, und deshalb gucken wir mal, was die Forth-Prozessoren von dem Rest der Stackprozessoren unterscheidet:

Forth-Prozessoren haben zwei Stacks, der Rest hat immer nur einen. Dafür haben alle C-philen Stackprozessoren eine Unterstützung von Framepointern in Hardware, entweder auf dem Stack direkt (Hobbit, Borroughs, Java) oder über den Hauptspeicher (Transputer, HP 3000), wobei letzterer mit besonderen Caches durchaus beschleunigt wird. IMHO ist letztere Variante zu empfehlen. Was muß man also in seinen Forth-Prozessor einbauen, damit er C kann? Lediglich einen Pointer, der mit Offset angesprochen werden kann. In Forth kann man den dann als Objektpointer verwenden, für die objektorientierte Erweiterung, in C als Framepointer. Dann erübrigt sich auch die Frage, ob der Datenstack tief genug ist: Der Transputer kommt mit 3, die 3000er von HP mit 4 Stackelementen aus, beides für Forth jenseits von Gut und Böse, für C aber offenbar ausreichend.

Von einer anderen Vorstellung sollte man sich auch rechtzeitig verabschieden, nämlich daß embedded-Control-Mikroprozessoren besonders klein sind. Eine gewisse Chipgröße kann man nur mit Mühe unterschreiten (Bondpads mal Padgröße), und da auf heutigen Prozessoren die Bondpads über dem Rest der Metallayers angebracht werden, hat darunter schon eine ordentliche CPU Platz. Der niedrigere Preis einer veralteten Technologie gilt nur solange, bis sie weggeworfen wird. Man wird sich also auch im Low-End-EC-Bereich an 32-Bit-CPUs mit Caches und sowas gewöhnen müssen, auf denen dann auch C vernünftig läuft. Wer also noch Stackprozessoren baut, sollte das "besonders klein" vergessen, weil sich daraus kein Vorteil mehr schlagen läßt. Das einzige, was einem bleibt ist "bei besonders hoher Performance immer noch relativ klein".

Durch die ganze Internet-Java-Hysterie ist dadurch tatsächlich ein ungeahnter Markt für Forth-Prozessoren entstanden. Chuck Moore etwa baut einen seiner Prozessoren für iTV, eine Firma, die Fernsehen und Internet in einer Kiste bringen will. Wenn diese Firma Erfolg hat (und nicht nur von den Anlegern lebt), wird die Frage "Warum gibt's keinen Web-Browser in Forth" bald der Vergangenheit angehören. Auch wenn der Forth-Prozessor in so einem Fernseher dann vielleicht hauptsächlich mit Java beschäftigt wird.

Bernd Paysan in de.comp.lang-.forth im Juni`96

Holon unter Novell Dos

Holon von Wolf Wejgaard (vgl. VD 2/96, Seite 33) besteht aus einem von der Target-Anwendung getrennt gleichzeitig laufenden Host, in dem das Programm geschrieben wird.

Der Parallel-Lauf kann bei Anwendern ohne Zweit-PC nur durch Multitasking unter Windows oder OS/2 erreicht werden. Das wirkt behelfsmäßig in Anbetracht der Tatsache, daß Holon ein DOS-Programm ist. Unter Windows und OS/2 werden im Hinblick auf Holon gesehen überflüssig viel Ressourcen verbraucht.

Das muß jedoch nicht so sein: Holons Host und Target multitasken einwandfrei auch unter DOS, nämlich Novell-DOS 7. Das bringt Geschwindigkeitsvorteile und ist natürlich naheliegender. Novell DOS 7 kostet rund 70 DM und ist u.a. erhältlich beim Tewi-Verlag.

Folgende Einstellungen sind erforderlich, damit Host und Target miteinander kommunzieren:

In der TASKMGR.INI muß die Priorität eingestellt sein mit:

```
FOREGROUND=1.
```

In der CONFIG.SYS muß folgender Eintrag vorhanden sein:

```
DEVICE=C:\DOS\EMM386.EXE  MULTI DPMI=ON
FRAME=AUTO  VxD=C:\DOS
```

Holon sollte mit einer Batchdatei etwa folgenden Inhalts aufgerufen werden:

```
@echo off
taskmgr
cd c:\holon
taskmgr /c c:\holon\win-mon.exe
cd c:\holon
c:\holon\holon-lt.exe
cls
pause
cls
taskmgr /k:2
echo.
echo.
echo.
echo.
pause
```

Mit CTRL ESC den Taskmanager aufrufen und ihn dann mit DEL-Taste löschen.

Frank Schwanitz, Mai96

zu 'kleinstes Forth' im Leserforum VD 2/96, Seite 6

Gerhard Raisig hat in seinem Leserbrief in der VD 2/96 völlig recht: Ich habe Peirce als synonyme Funktionsbezeichnung unachtsamerweise dem NAND statt dem NOR zugeordnet und mit “ie” geschrieben. Ich werde mich bessern und wieder dazu übergehen, Geschriebenes erst noch ein paar Tage liegenzulassen, bevor ich es wegschicke. Am Inhalt des von mir Gesagten ändert sich nichts, wie Gerhard Raisig mit seiner Formulierung bestätigt. Ich habe das Ganze als schnell skizziertes erläuterndes Beispiel gedacht. Einer der beiden Hinweise, Peirce oder NAND, schien mir genug. Danke für das aufmerksame Lesen. Was ist nun aber zu meinem eigentlichen Anliegen, der Frage nach einem “minimalen Forth”, zu sagen? Forth als Sprache, nicht als System. Die gängigen Datenstackoperationen lassen sich beispielsweise durch PICK und PLOP erledigen, wenn 0 PLOP dem DROP entspricht, 1 PLOP dem NIP usw. Geht es noch “einfacher”? Um nicht ins Blaue hinein zu reden, hier eine konkrete Zusammenstellung:

```
: PLOP ( n -- ) SP@ DUP CELL+ ROT
CELLS CELL+ MOVE DROP;

: DROP 0 PLOP ;
: DUP 0 PICK ;
: OVER 1 PICK ;
: ROT 2 PICK 3 PLOP ;
: SWAP 1 PICK 2 PLOP ;
: -ROT 2 PICK 2 PICK 3 PLOP 3 PLOP ;
: NIP 1 PLOP ;
: TUCK 1 PICK 2 PLOP 1 PICK ;
: 2DUP 1 PICK 1 PICK ;
: 3DUP 2 PICK 2 PICK 2 PICK ;
: 2OVER 3 PICK 3 PICK ;
: 2DROP 0 PLOP 0 PLOP ;
: 2NIP 2 PLOP 2 PLOP ;
: 2SWAP 3 PICK 3 PICK 4 PLOP 4 PLOP ;
: ROLL >R R@ PICK R> 1+ PLOP ;
```

Das Wort -ROLL ist etwas umständlicher. Ich lasse es zur Übung. Worte, die in ein Minimalsystem aufgenommen werden sollen, würde man wohl nach Möglichkeit als Code-Definitionen fassen. In meinem Transputer-Forth könnte das für PLOP wie folgt aussehen:

```
CODE PLOP ( n -- )
   SP LDL 0 LDNL EREG STL \ n
   SP LDL 4 ADC SP STL \ Stack kürzen
   EREG LDL BCNT BREG STL \ Länge
   SP LDL 3 ADC CREG STL \ Ziel-1
   SP LDL -1 ADC DREG STL \ Quelle-1
   DREG LDL BREG LDL ADD LB \ Von hinnen
   CREG LDL BREG LDL ADD SB \ nach dannen
   AREG LDLP 0D LDC LEND
   SP LDL 4 ADC SP STL \ Nochmal kürzen
   NEXT END-CODE
```

Diese Dinge habe ich vorsichtshalber alle ausprobiert (und dabei meine Überraschungen erlebt). Selbstverständlich geht es auch mit nur einem einzigen Wort:

```
: PLUFF ( n fl -- )
   IF PICK ELSE PLOP THEN ;
```

Aber das berührt schon die Grundfrage nach einer geeigneten Bestimmung des Begriffs “Wort”. Wort ist nicht gleich Wort. Welchen Kriterien muß ein Wort genügen, um in einem “Minimalforth” Aufnahme zu finden? Schließlich könnte man ja nach dem obigen Schema sämtliche Forth-Worte über eine geeignete Parameter-Auswahl aus einem einzigen Urwort heraus erzeugen. (Das liefe auf CASE: hinaus.) Der Einwand, PLUFF erscheine hier als zusammengesetztes Wort, das von PICK und PLOP Gebrauch macht, läßt sich entkräften: Man nehme die Code-Definitionen von PICK (bei mir eine solche) und PLOP und verarbeite sie zu einer Code-Definition aus einem Guß für PLUFF.

Fred Behringer, München (beh)

iX fährt Profibus

Zur Zeit starte ich ein kleines Entwicklungsprojekt, in dem ich den IX von delta-t als Protokoll-Chip für den Profibus einsetzen möchte. Ich würde mich deshalb gerne mit Anwendern unterhalten, die praktische Erfahrungen im Umgang mit der CPU und/oder dem Forth-Entwicklungssystem haben.

Danke, Winfried Clemens@clforth.forth-ev.de (Winfried Clemens), Juli96

Gesucht: Billigste Schrittmotorsteuerung

In einem Informatikkurs ergab sich für eine Kollegin folgendes Problem: Entweder waren Schrittmotoren billig zu bekommen (bipolar- ca. 1,50 DM) und die Steuerungshardware teuer (bipolare Treiber ca. 24 DM) oder die Schrittmotoren waren teuer (unipolar ca. 16 DM) und die Treiber billig (“normale” Treiber ca. 1-2 DM). Beide Lösungen übersteigen den Finanzierungswillen der an Lehrmittelfreiheit gewohnten Schüler. Überlegungen mit Widerstandsnetzen, Spannungsteilern etc. bringen Unübersichtlichkeit und hohen Leistungsverlust mit sich.

Ratschläge bitte an Forth-Gruppe-Moers via: F.PRINZ@MHB-.GUN.DE

Neuer Name?

Rafael Deliano macht einige gute Vorschläge zur Neugestaltung des VD-Covers. Die Beispiele sind übersichtlich und ansprechend.

Ob jedoch ein neuer Titel sein muß, bezweifle ich. Selbst wenn das Wortspiel (Forth-Dimensions) durch die Übersetzung gelitten hat, so ist doch der Titel VD historisch gewachsen und etabliert (mein Gott was bin ich konservativ). Wenn ”Forth” im Titel gewünscht wird, böte sich m.E. eine Formulierung an, die der Tradition folgend ein Wortspiel enthält und gleichzeitig progressiv ”Forth” enthält:

Forth-Schritt, Forth-wärts, Forth-Gang, For(th)um Forth-ruhm, For(th)mat, Der Forth-Kommer, Forth-Schreiben, Forth-Schreibung... .

Martin Bitter

“N’ bißchen wat Kritik”

Was ich früher als “Salz in der Suppe” betrachtete, wenn ich ein Computermagazin las, waren die Listings. Es machte mir einfach Spaß, Problemlösungen nachzuvollziehen oder mal was Kurzes, Spaßiges abzutippen und anzusehen oder zu verbessern. Dabei kam es natürlich auf den praktischen Nutzen an. Ohne Nutzen - kein Interesse, wobei ich eine Spielerei, die mir Spaß bringt, durchaus als nützlich ansehe. So etwas fehlt hier in der VD, glaube ich.

Außerdem finden Anfänger nicht genügend Unterstützung. So fragte ich vor einiger Zeit nach Floatingpoint Quellen, da ich einen kleinen UPN-Taschenrechner schreiben wollte, aber helfen konnte man mir nicht. Da hört man nur das übliche: “Das kann man doch auch mit Integerarithmetik machen.” Und dann noch: “Aber wenn du was findest, kannst du ja mal Bescheid sagen.”

“Bescheid!” kann ich da nur sagen. Oder wenn man nach Stringbefehlen sucht: “Die sind doch ganz leicht zu implementieren!”

“Is ja alles richtig!” - liebe Cracks und langjährige Schreiber von Artikeln, die ihr eure FORTH-Systeme seit Jahren bis in die hinterste Ecke erforscht habt. Aber kommen nicht die zu kurz, die einfach nur mal schnell ein Problem lösen, oder eine Idee umsetzen wollen (z.B. von BASIC nach FORTH). Da kommt es doch nicht auf den trickreichsten und schnellsten Code, sondern auf Durchschaubarkeit und schelle Umsetzung an, und da spielen die beiden oben genannten Themen eine wichtige Rolle.

Die VD macht oftmals den Eindruck, als wenn immer nur “Eingeweihte” in “Eingeweiden” wühlen (man verzeihe die derbe Wortspielerei), statt sich mal an der Schönheit eines Programms oder dessen Ausführung zu erfreuen. Wie soll denn der Club wachsen, wenn nicht auch mal ein paar kleine Nettigkeiten (Programmäßig) in die VD eingestreut werden, die auch ein Anfänger nachvollziehen kann und somit ein Erfolgsergebnis hat - ein Grundstein für dauerhaftes Interesse.

Mein Aufruf gilt deshalb auch an all die netten Amateure unter uns - “Schreibt mal wieder !”

Nicht das Perfekte ist gefragt, sondern das was Raum für Verbesserungen läßt, denn das spornt zu eigenem Handeln und Nachdenken (über/mit FORTH) an.

Wolfgang Führer, 45663 Recklinghausen, Idastr. 19, Tel. 02361/891344, E-mail: wfuehrer@uni-upn.forth-ev.de oder 100415.3227@compuserve.com

P.S. “Kritiker haben nie recht!”

Echt Zeit '96 | iNet '96
18. - 20. Juni 1996
Karlsruher Kongreß- und Ausstellungszentrum, Stadthalle

Kurzbericht über die

Echtzeit 96

von Birgit Steffenhagen und Ulrich Hoffmann

Wie in jedem Jahr war die Forth-Gesellschaft e. V. auch in diesem Jahr auf der Echtzeit vertreten und hat versucht, Anfragen zur Programmiersprache, zu ihren Möglichkeiten und Anwendungsschwerpunkten und natürlich auch zur Gesellschaft selbst zu beantworten. Leider hatten wir uns ein bißchen spät, nämlich nach der Mitgliederversammlung, entschlossen auch in diesem Jahr wieder präsent zu sein, so daß unser Stand ein wenig unglücklich lag.

Stichworte: Echtzeit96

Die Vorbereitungen hatte diesmal Winfried Clemens von der Firma EchtzeitSysteme W.Clemens in die Hand genommen. Dafür an dieser Stelle unser Dankeschön. An der Standbetreuung waren wieder einmal Malte Köller und Birgit Steffenhagen beteiligt. Malte stellte das Modell des Programmierwettbewerbes des vergangenen Jahres namens “Sisyphus” vor. Er steuerte es mit einem 68332 an, auf dem ein prioritätsgesteuertes Multitask-Forth implementiert war. Dieses kommunizierte über eine serielle Schnittstelle mit einem intelligenten Terminal, welches auf comFORTH für Windows basierte. Neben dem Modell wurden noch eine ganze Reihe anderer Gerätschaften von dem kleinen 68332 angesteuert und damit das Multitasksystem ganz schön ausgereizt. Da sich an unserem Stand also etwas bewegte, blieben auch reichlich Leute stehen, die so einiges wissen wollten oder/und zu wissen bekamen.

Ein zentrales Thema der Messe und des Kongresses war die Vereinbarkeit von Echtzeit und Windows95 bzw. Windows-NT. Dazu gab es auch eine Podiumsdiskussion, auf der festgestellt wurde, daß Windows-NT nur weiche Echtzeitforderungen erfüllen kann. Einige Firmen stellten auch Hard- und Softwarelösungen zu diesem Thema vor. Für die Hardware wäre die Firma LP-Elektronik zu nennen, die eine kleine ISA-Einsteckkarte vorstellte, die Windows determiniert unterbrochen hat. Als Softwarelösung wäre das allen bekannte RMOS for Windows zu nennen.

Weitere Kongreßthemen waren:

- Verteilte Systeme in der Automatisierungstechnik
- Testumgebungen
- Fallstudien
- Objektorientierte Entwurfstechniken
- Betriebssystemaspekte
- PC-Plattformen in der Automatisierung
- Innovative Ansätze in der Praxis
- Entwicklungswerkzeuge und Standards
- Anwendungen von Fuzzy-Logik

Am Donnerstag fand der Programmierwettbewerb statt. Das Modell hatte keinen Namen - ich nenne es den Zappelphilip. Es war ein Aluminiumpendel, an dessen Ende ein Permanentmagnet befestigt war. Bei genügend weitem Ausschlag hat er rechts bzw. links je einen Reed-Kontakt ausgelöst. Am Scheitel (bei 6 auf der Uhr) befand sich eine Spule, durch die man keinen oder aber so oder andersrum Strom schicken (800mA) konnte und sich damit je nach Orientierung ein Magnetfeld aufgebaut hat, das den Permanentmagneten angezogen oder abgestoßen hat.

Als kleine Schikane durfte die Spule nur etwa 30% der Zeit belastet werden, oder aber man disqualifizierte sich selbst durch eine aufrauchendes Modell.

Aufgabe war, das Pendel in dauerhafter gleichmäßiger Pendelbewegung mit großer Amplitude zu halten. Dies zu erreichen, war angeblich nur mit einer Regelung möglich - stimmte aber nicht.

Wer hat gewonnen?

Hubert Jäger’s Team hat das als erstes erkannt und sie waren schon nach 10 Minuten fertig (Die haben sich ja den ganzen Spaß verdorben).

Wir anderen haben ein wenig länger gebraucht:
- Malte Köller und Winfried Clemens (60Min)
- Andreas Dobbertin und Ulrich Hoffmann (90Min)

Ein weiteres Forth-System wurde von Egmont Woitzel und Stephan Lange benutzt. Sie benutzten ein 68HC11-Board mit fieldFORTH und sind sechster geworden.

Zu gewinnen gab es DM 1200,-, einen Hitex 68K-InCircuit-Emulator, LynxOS, QNX, einen C-Crosscompiler, Ada und viele Buchpreise aus dem Franzis Verlag, der auch den Wettbewerb ausgerichtet hat.

Alles in allem war es wieder ein Riesenspaß.

❑

Ein Rahmen für modular aufgebaute Forth-Systeme

von Malte Köller / Uni Rostock
Tel.: 0381 / 4983552
mk@atnw1.e-technik.uni-rostock.de
WWW: http://www.e-technik1.uni-rostock.de/at/home_pages/malte

In diesem Beitrag soll es nicht vordergründig darum gehen, wie man ein Forth-System modular aufbaut. Das ist ein ganzes Kapitel für sich. Vielmehr soll gezeigt werden, wie man die Eingliederung der einzelnen Komponenten (Module) des Forth-Systems elegant gelöst bekommt.

.Stichworte: Objekt Modul Ressourcen Konstruktoren Destruktoren comFORTH_Win m68332

Modulrahmen

Die Idee für ein Modulrahmen ist teilweise der objektorientierten Programmiertechnik entlehnt, nämlich dem Definieren von Konstruktoren und Destruktoren für Objekte (siehe C++).

Konstruktoren dienen dem Initialisieren von Objekten (Rücksetzen von Variablen, Öffnen von Ressourcen, Allokieren von Speicher...), Destruktoren dagegen zum korrekten Beseitigen von Objekten (Zurückgeben von Speicher, Schließen von Ressourcen...).

Diesen Mechanismus kann man nun auch auf die Forth-Module übertragen. Jedes Modul bekommt seinen eigenen Konstruktor und Destruktor.

Die Frage stellt sich nur, wann werden diese aufgerufen. Beim Konstruktor läßt sich die Frage noch relativ leicht beantworten. Er wird im `COLD` des Forth-Systems aufgerufen bzw. für nachladbare Module sofort nach dem Ladevorgang. Der Destruktor dagegen kann im `BYE` aufgerufen werden. Das reicht jedoch nicht aus, denn man will nachladbare Module ja vielleicht auch während der Laufzeit des Forthsystems wieder entladen (`FORGET`) , wobei angenommen wird, daß Module immer als Ganzes betrachtet werden und somit immer vollständig geladen und entladen werden. Daraus folgt, daß der Destruktor auch aufgerufen werden muß, wenn das Modul mit `FORGET` aus dem Dictionary entfernt wird.

Diese Idee wurde einmal in comFORTH für Windows für nachladbare Module und außerdem für ein modular aufgebautes Forth-System für den m68332-Controller umgesetzt.

.

Implementierung

Die Definitionen für die einzelnen Konstruktoren und Destruktoren wurden jeweils über eine Liste verkettet, die dann entsprechend im COLD, BYE bzw. FORGET (da nur teilweise) abgearbeitet wird. Für das Definieren der Aktionen wurden spezielle Definitionswörter eingesetzt.

Anwendungsbeispiel:

```
... Code des Modules ...

:COLD MODUL_KONSTRUCTOR_CODE ;

:BYE MODUL_DESTRUCTOR_CODE ;
```

Die Definition der beiden Wörter ist im abgedruckten Quelltext zu finden.

Es muß beachtet werden, das die BYE-Liste mit dem Ankerfeld `BYE^` andersherum aufgebaut werden muß als die COLD-Liste mit dem Ankerfeld `COLD^`. Damit wird erreicht, daß die COLD-Liste entsprechend der Ladereihenfolge der Module abgearbeitet wird, die BYE-Liste dagegen entsprechend der Entladereihenfolge.

Die Definitionen von `COLD` und `BYE` gestalten sich dadurch trivial (wenn das Forthsystem von Anfang an mit dieser Mimik implementiert wird wie beim m68332-Forth).

```
: BYE ( ps: ===> )
   ( Verlassen des Systems )
    BYE^ EXECLIST RETURN ;

: COLD ( ps: ===> )
   ( Start des Systems )
    COLD^ EXECLIST BYE ;
```

Für das Entladen der einzelnen Module über FORGET stehen die beiden Wörter `FORGETCOLDLIST` und `FORGET-BYELIST` zur Verfügung, die jeweils die Anfangs- und

Endadresse des zu vergessenden Bereiches übergeben bekommen. Der Einbau dieser Wörter in das FORGET hängt dann natürlich vom jeweiligen System ab (wenn es denn überhaupt möglich sein sollte).

Die FORGET-Wörter rufen für ein Modul im zu vergessenden Bereich den Destruktor auf und ketten ihn und den Konstruktor aus der Liste aus. Dabei wurde darauf Rücksicht genommen, daß Module auch über eine Heapverwaltung dynamisch geladen werden könnten bzw. allgemein in getrennten Wörterbuchbereichen zu liegen kommen und damit der Entladevorgang nicht mehr stackartig erfolgt.

Der sofortige Aufruf des Konstruktors nach dem Laden von Modulen müßte in den Lademechanismus des jeweiligen Systems eingebunden werden. Das wurde bei der vorliegenden Variante jedoch nicht eingesetzt, da man den `'MODUL_KONSTRUCTOR_CODE'` auch einfach in den Quelltext schreiben kann, wodurch er beim Laden ausgeführt wird.

Dieser Applikations-Rahmen läßt sich zum Aufbau eines kompletten Forth-Systems einsetzen. Dabei wird der äußere Interpreter (das `QUIT`) einfach als Konstruktor des letzten Moduls eingebunden.

```
:COLD QUIT ;
```

Soll eine Turnkey-Applikation (nicht das Forth-System startet sondern die eigene Applikation) erzeugt werden, muß nur das Interpreter-Modul (oder QUIT-Modul oder...) durch ein Applikationsmodul mit eigenem Konstruktor ersetzt werden.

Anwendungen

Die vorliegende Konstruktion kam im Zusammenhang mit einer Overlay-Technik unter *comFORTH für Windows* zum Einsatz, mit der einzelne Module (Overlays) zur Laufzeit geladen und auch wieder entladen werden können, jedes Modul besitzt dabei einen eigenen Konstruktor sowie Destruktor.

Einfachste Anwendung in einem solchen Modul ist z.B. das Verbiegen von Systemvektoren über `DOER` oder defered words. Beim Entladen des Moduls werden die Vektoren damit sauber wieder zurückgesetzt.

Außerdem kam die Konstruktion in einem modularen Forth für den *m68332* zur Anwendung. Hier wurde die gesamte Systeminitialisierung über die Rahmenkonstruktion realisiert.

Da das Forth-System ROM-fähig implementiert ist, werden auch sämtliche Variablen bzw. DOER-Initialisierungen darüber abgewickelt.

Literatur:

Stroustrup, Bjarne: Design und Entwicklung von C++,
Addison Wesley, 1994

❑

File: coldbye.cfw vom 5.4.1996

```
\ COLDBYE
\ ======================================
\
\ stellt Konstruktoren und Destruktoren
\ zur Verfügung
\
\ ----------------------------------------
\
\ Bearbeiter: M. Köller
\         Stand: 13-Feb-96 / korr.vd 5.4.96
\
\ ? 1995 Uni Rostock
\ ======================================

DECIMAL

CR
.( Warte ...lade Konstruktoren --
       Destruktoren )
CR

\  ----------------------------------------
\ comFORTH - Anpassungsschicht (nur laden,
      wenn nicht vorhanden)
\  ----------------------------------------

: RDROP   ( rs: n ===> )
          R> R> DROP >R ;

: UWITHIN   ( ps: u umin umax ===> ? )
                ( ?t wenn umin<=u<umax )
         2 PICK U> >R U<_>0= R> AND ;

: DELINK   ( ps: addr1 ===> addr2 )
              ( 1=Vorgaenger 2=alt )
         DUP @ DUP @ ROT ! ;

: CALL ( ps: addr ===> )
           ( Fadencode-Stück aufrufen )
         >R ;

: PRE   ( ps: addr link ===> pred | 0 )
           ( Vorgaenger in Liste )
         BEGIN ?DUP IF    2DUP @ =
                          IF    NIP TRUE
                          ELSE @ FALSE THEN
                     ELSE DROP 0 TRUE   THEN
       UNTIL ;

: LINK   ( ps: addr1 addr2 ===> )
            ( 1=neu 2=Vorgaenger )
         DUP @ 2 PICK ! ! ;

: !CSP   ( ps: ===> )
       ( Überwachung von Colon-Definitionen )
         ;

: RETURN   ( ps: ===> )
              (  Verlassen des Forthsystems )
           ( hängt vom System ab      )
         ;

\ Anker der Listen
```

```
\ ----------------
VARIABLE  COLD^
VARIABLE  BYE^        \ Listenanker
\ können zur Laufzeit erweitert werden
COLD^ OFF
BYE^   OFF

\ Hilfsdefinitionen
\ -------------------
: EXECLIST ( ps: 'list ===> )
                ( arbeitet die Liste 'list ab )
           BEGIN  @  ?DUP
           WHILE  R! CELL+ CALL R> REPEAT ;


\ für die FORGET-Mimik
\ ---------------------
: FORGETLIST
        ( 'action 'list aaddr eaddr ===> )
     ( entfernt Listenstück zwischen aaddr )
        ( und eaddr mit Ausführung )
        ( von 'action auf jeden ausgelinkten )
        ( Knoten )
( Verhalten von 'action: ps: 'node ===> )
    2>R
    BEGIN DUP @ ?DUP
    \ Vorgänger Nachfolger
    WHILE DUP 2R@ UWITHIN         \ Nachfolger im Bereich?
          IF DROP DUP DELINK \ dann entfernen
             2 PICK EXECUTE   \ Aktion ausführen
          ELSE NIP THEN         \ Nachfolger wird Vorgänger
    REPEAT 2DROP RDROP RDROP ;


: FORGETCOLDLIST
      ( ps: aaddr eaddr ===> aaddr eaddr )
   ( kürzt die COLD-Liste beim Vergessen )
           ['] DROP COLD^ 2SWAP 2R! FORGETLIST
           2R> ;

: DEFACTION ( ps: 'node ===> )
( ruft Destruktorroutine eines Knotens auf )
           CELL+ CALL ;

: FORGETBYELIST
   ( ps: aaddr eaddr ===> aaddr eaddr )
   ( kürzt die BYE-Liste beim Vergessen )
   ( und ruft gegebenfalls Destruktor-Routine
   ( auf )                       )
           ['] DEFACTION BYE^ 2SWAP 2R!
           FORGETLIST 2R> ;


\ Anwenderschnittstelle
\ ----------------------

: :COLD ( ps: ===> )( ib: ... ; )
           ( definiert neuen Konstruktor )
           HERE 1 CELLS ALLOT 0 COLD^ PRE LINK
           !CSP ] ;


: :BYE ( ps: ===> )( ib: ... ; )
          ( definiert neuen Destruktor )
           HERE 1 CELLS ALLOT BYE^ LINK !CSP ] ;    \ Liste and
ersherum aufbauen


\ Definition von COLD und BYE
\ ---------------------------

\ System-Rahmen

: BYE ( ps: ===> )( Verlassen des Systems )
           BYE^ EXECLIST RETURN ;

: COLD   ( ps: ===> )( Start des Systems )
           COLD^ EXECLIST BYE ;


.(          ...fertig )
CR

```
Längste Zeile hat 76 Zeichen

zu Seite 14:

File: screened.fth vom 27.7.1996

```
\ SCREENED.FTH

HEX

70 STRING CHAR-CURS@$                    \ Belegung folgt und bleibt

" BASE @ HEX " CHAR-CURS@$ APPEND$   \ Host auf HEX schalten, Basis retten.
" 40 62 LC@ "   CHAR-CURS@$ APPEND$   \ Bildschirmseite holen.
" FLIP "         CHAR-CURS@$ APPEND$   \ Liegt im unteren Byte, muß ins obere.
" CODE XXX "    CHAR-CURS@$ APPEND$   \ CODE-Definition XXX aufbauen.
" BX POP "       CHAR-CURS@$ APPEND$   \ Bildschirmseite jetzt in BH.
" 8 # AH MOV " CHAR-CURS@$ APPEND$   \ Funktionsnummer für Interrupt 10h.
" 10 INT "       CHAR-CURS@$ APPEND$   \ Interrupt 10h ausführen.
" 1PUSH "        CHAR-CURS@$ APPEND$   \ Gelesenes Zeichen (AL) und Attribut
" END-CODE "    CHAR-CURS@$ APPEND$   \ (AH) auf Stack und Definition beenden.
" XXX "          CHAR-CURS@$ APPEND$   \ XXX im Host ausführen.
" FORGET XXX " CHAR-CURS@$ APPEND$   \ XXX wieder beseitigen.
" 40 0FE L! "   CHAR-CURS@$ APPEND$   \ In "BIOS-Kommunikationsbereich" legen.
" BASE ! "       CHAR-CURS@$ APPEND$   \ Basis im Host wiederherstellen.

\ Das Ganze sieht nur so gefährlich aus. Es erfordert aber wirklich nur
\ etwas mehr als die 70h Bytes für den String, die auch dann angefallen
\ wären, wenn ich den String über " ... " $EXECUTE-HOST direkt in das Wort
\ CHAR-CURS@ hineingeschrieben hätte.

: CHAR-CURS@ ( -- xy )                   \ xy = 16 Bit, x=Attribut, y=ASCII-Code
      CHAR-CURS@$ $EXECUTE-HOST          \ String im Host ausführen.
      40 0FE @-H ;                       \ xy vom BIOS-Zwischenbereich holen.

9A STRING CHAR-CURS!$                    \ Belegung folgt und bleibt

" BASE @ HEX " CHAR-CURS!$ APPEND$   \ Basis retten, Host auf HEX schalten.
" 40 0FC L@ "   CHAR-CURS!$ APPEND$   \ Erst Zeichen mit Attribut, dann Anzahl
" 40 0FE L@ "   CHAR-CURS!$ APPEND$   \ aus "BIOS-Zwischenbereich holen.
" 40 62 LC@ "   CHAR-CURS!$ APPEND$   \ Bildschirmseite holen.
" FLIP "         CHAR-CURS!$ APPEND$   \ Liegt im unteren Byte, muß ins obere.
" CODE XXX "    CHAR-CURS!$ APPEND$   \ CODE-Definition XXX aufbauen.
" BX POP "       CHAR-CURS!$ APPEND$   \ Bildschirmseite jetzt in BH.
" CX POP "       CHAR-CURS!$ APPEND$   \ Anzahl nach CX.
" 9 # AH MOV " CHAR-CURS!$ APPEND$   \ Funktionsnummer für Interrupt 10h.
" DX POP "       CHAR-CURS!$ APPEND$   \ Attribut-Byte und ASCII-Byte.
" DH BL MOV "   CHAR-CURS!$ APPEND$   \ Attribut-Byte nach BL.
" DL AL MOV "   CHAR-CURS!$ APPEND$   \ ASCII-Byte nach AL.
" 10 INT "       CHAR-CURS!$ APPEND$   \ Interrupt 10h ausführen.
" NEXT "         CHAR-CURS!$ APPEND$   \ Von Code-Definition
" END-CODE "    CHAR-CURS!$ APPEND$   \ wieder zurück.
" XXX "          CHAR-CURS!$ APPEND$   \ XXX im Host ausführen.
" FORGET XXX " CHAR-CURS!$ APPEND$   \ XXX wieder beseitigen.
" BASE ! "       CHAR-CURS!$ APPEND$   \ Basis im Host wiederherstellen.

: CHAR-CURS! ( xy n -- )                 \ x=Attr-Byte, y=ASCII-Byte, n=Anzahl
        40 0FE !-H                       \ n   in den BIOS-Zwischenbereich legen.
        40 0FC !-H                       \ xy in den BIOS-Zwischenbereich legen.
        CHAR-CURS!$ $EXECUTE-HOST ;  \ String im Host ausführen.

\ P.S.: F-TP, mein Transputer-Forth, gibt es höchstwahrscheinlich auf der
\         kommenden Weihnachts-CD.
```
Längste Zeile hat 76 Zeichen

Forth-Server bei Forth-Host-Target-Interfaces

Fred Behringer
Planegger Str. 24; D_81241 München

Situation: Im Host (bei mir ein 486-DX/2-PC) läuft ein Forth-System (Host-Forth), genauer gesagt, ein in diesem programmierter "Server". Im Target (bei mir Transputer T800) läuft ebenfalls ein Forth-System (Transputer-Forth oder Target-Forth). Der Server sitzt ständig am "Linkadapter" (seriell, 10 MBit/sec) und lauscht. Auf ein bestimmtes Zeichen (Token) vom Transputer-Forth hin tritt der Server in Aktion: Er empfängt weitere Zeichen zur Verarbeitung im Host oder sendet seinerseits Zeichen vom Host zum Target.

Stichworte: Turbo-Forth F83 PC Transputer Host-Target-Interface

auf Seite 13:
SCREENED.FTH3.157 27.07.96

Beweggrund

In der ebengenannten Situation kommt es nicht darauf an, daß ein Transputer als Target wirkt. Jeder andere Mikroprozessor, jeder Nicht-on-Chip-Gleitkomma-Prozessor, jeder Mikrocontroller, der am Host-PC hängt, um auf dessen Ressourcen zurückgreifen zu können, wirft ähnliche Fragen auf. Vielleicht, denke ich daher, könnten auch andere Leser an meinen Überlegungen Interesse haben.

Aufgabe

Es sollen zwei Worte im Target definiert werden, die es gestatten, ein Zeichen mit Attribut unter den Cursor zu schreiben oder das Zeichen an der momentanen Cursorposition mit Attribut auszulesen. (An sich ist es nur das letztere, was mich interessiert, sozusagen ein Gegenstück zu EMIT. Das erstere läßt sich mit EMIT und (ATTRIB) - in Turbo-Forth jedenfalls, und mit ANSI.SYS im Speicher - erledigen.) Die Entwicklung meines Servers ist abgeschlossen. Ich habe wenig Lust, bei jeder neuen Idee, die ich dem Target-Forth nachträglich hinzufügen möchte, einen neuen Baustein in den Server einzubauen, der auf ein weiteres Token vom Target reagieren soll. Was ich suchte, war eine Methode, die beliebige weitere Vorhaben im Host (Aufrufen von BIOS- oder DOS-Interrupts) allein durch Hinzufügen neuer Worte im Target (im compilierten, nicht mehr metacompilierten, zweiten Teil meines Transputer-Forth-Systems oder sogar erst im Anwendungsprogramm) zu verwirklichen gestattet (offener, target-manipulierbarer Server).

Idee

Ich nütze aus, daß Forth-Programme in den Compiler eingreifen können: Ein Forth-Programm kann sich aus dem Lauf heraus selbst erweitern, reduzieren oder anderswie verändern. In Turbo-Forth gibt es das Wort $EXECUTE (führe einen String aus). In mein Transputer-Forth (F-TP) habe ich das Wort $EXECUTE-HOST eingebaut (führe einen String im Host-System aus). Natürlich sind noch ein paar andere grundlegende Worte für den Host-Server-Target-Betrieb eingebaut: Hier interessiert höchstens @-H und !-H (hole 16 Bit von einer Langadresse im Host und speichere 16 Bit an eine Langadresse im Host). Für die Zwischenspeicherung verwende ich den BIOS-Interkommunikationsbereich 40:F0 bis 40:FF. Mit einem solchen Mechanismus kann ich alles erschlagen, da es ja für das in den String Hineinzuschreibende, abgesehen von der 255-Byte-Länge, keine Grenzen gibt. Ich übertrage Parameter in den Zwischenbereich, definiere ein Wort (Code XXX ... NEXT END-CODE oder : XXX ... ;), rufe das Wort (XXX, natürlich im Host) auf, beseitige das Wort wieder (FORGET XXX) und lege die anfallenden Ergebniswerte in den BIOS-Zwischenbereich. Nach Rückkehr vom Host, wenn also der gesamte String von dem im Target laufenden $EXECUTE-HOST bearbeitet ist, hole ich mir im ummantelnden Target-Wort, das auch $EXECUTE-HOST enthält, die Ergebniswerte vom BIOS-Zwischenbereich.

Verwirklichung

Ich gebe im folgenden die kommentierten und sich selbst erklärenden Programme wieder. Abgesehen von $EXECUTE-HOST, @-H und !-H aus dem Target-System (Besprechung siehe oben), enthält das Listing nur PC-Forth/Assembler-Worte und ist damit für alle diejenigen, die den PC als Host verwenden, verständlich. Um genau zu sein, ist es der 80286-Assembler aus Turbo-Forth, der mit dem aus F83 und einigen anderen Systemen, die Forth 83 verwenden, übereinstimmt. Natürlich wird hier vom Programmlauf her mit dem ständigen Neucompilieren und wieder FORGETten im Host ein Riesenaufwand getrieben, der unnötig Zeit kostet. Aber bei solchen Aufgaben, wie dem gelegentlichen Abholen eines Zeichens vom Bildschirm, und bei der Schnelligkeit der beteiligten Geräte spielen derartige Trödelzeiten (Overheads) keine Rolle.

32-Bit 386-Prefix-Assembler mit AT&T Syntax

von Bernd Beuster
B.Beuster@bbepoint.FORTH-ev.de
Carl-Benz-Str. 1A; 55131 Mainz

*Bei der Entwicklung eines 32-Bit-Forth für OS/2, fiel die Wahl auf den Assembler **as**, welcher integraler Bestandteil der C/C++-GNU-Portierung von Eberhard Mattes ist (emx). Damit lassen sich native 32-Bit PM-Anwendungen erstellen.*
Die Metacompilierung des Forthkernels erfolgte mit diesem Assembler, so daß es nahe lag, dieselbe Syntax auch für den integrierten Forthassembler zu verwenden. Ziel der Implementierung war es, in der Forth-Implementierung möglichst wenig von der vorgegebenen Syntax abzuweichen, das bedingt einen Prefix-Assembler.
Die Quelle zu diesem Beitrag befindet sich auf der Listingdiskette als ASSEMBLE.F

Stichworte: Assembler 386 AT&T GNU OS/2

AT&T Syntax vs. Intel Syntax

Der GNU-Assembler verwendet die *AT&T SystemV/386*-Assemblersyntax. Diese unterscheidet sich von der *Intel*-Syntax und ist wesentlich ortogonaler. Die Unterschiede bestehen in:

Immediateoperanden
Registeroperanden
Operand-Delimiter

AT&T-Immediateoperanden werden durch ein vorangestelltes **$** gekennzeichnet. *Intel*-Immediateoperanden werden ohne spezielle Delimiter angegeben (*Intel* **push 4** entspricht *AT&T* **pushl $4**). *AT&T*-Registeroperanden besitzen ein vorangestelltes **%**, wogegen die *Intel*-Registeroperanden wiederum ohne Sonderzeichen notiert werden (*Intel* **eax** entspricht *AT&T* **%eax**). Damit entfallen automatisch Probleme, die z.B. durch die Forth-Konstante und dem Register **BL** auftreten können.

Quelle-Ziel Operanden

AT&T und *Intel* verwenden die entgegengesetzte Reihenfolge für die Quelle-Ziel Operanden. *Intel* **add eax, 4** wird zu *AT&T* **addl $4, %eax**. Diese Quelle-Ziel Konvention ist auch ,,forthiger".

Opcode Suffixe

In der *AT&T*-Notation wird die Größe der Speicheroperanden durch das letzte Zeichen im Opcode bestimmt. Die Opcode Suffixe **b**, **w** und **l** spezifizieren jeweils **byte** (8 Bit), **word** (16 Bit) bzw. **long** (32 Bit) Speicherzugriffe. In der *Intel*-Syntax wird das durch Prefixe der *Speicheroperanden* (nicht durch den *Opcode* selbst), wie **byte ptr, word ptr** und **dword ptr,** erreicht.

Somit wird *Intel* **mov al, byte ptr foo** zu *AT&T* **movb foo, %al.** Die Verlegung der Größe der Speicheroperanden in den Opcode vereinfacht die Implementierung wesentlich und trägt zur Eindeutigkeit und Lesbarkeit der Programme bei.

Opcode Bezeichnungen

Alle Opcodes, bis auf wenige Ausnahmen, haben dieselben Namen im *Intel*- bzw. *AT&T*-Format. Die *Sign-Extended*-und die *Zero-Extended*-Instruktionen benötigen zwei Größen, um spezifiziert zu werden. Sie benötigen jeweils eine Bezeichnungsgröße für **from** und eine für **to**. Die Basisbezeichnungen für *Sign/Zero-Extended*sind **movs...** und **movz...** in *AT&T*-Syntax bzw. **movsx** und **movzx** in *Intel*-Syntax. Die Opcode Suffixe werden an diese Basisbezeichnung angehängt, der **from**-Suffix vor dem **to**-Suffix. Daher steht **movsbl %al, %edx** in *AT&T*-Syntax für **move sign extended *from byte* %al *to word* %edx**. Mögliche Suffixe sind **bl** (from byte to long), **bw** (from byte to word) und **wl** (von word to long).

Die Opcodes in *Intel*-Syntax**:**

cbw erweitere Byte in **%al** zu Word in **%ax**,
cwde erweitere Word in **%ax** zu Long in **%eax**,
cwd erweitere Word in **%ax** zu Long in **%dx:%ax**,
cdq erweitere Long in **%eax** zu Quad in **%edx:%eax**,

werden zu:

cbtw (convert byte to word),
cwtl (convert word to long),
cwtd (convert word to double) bzw.
cltd (convert long to double) in der *AT&T*-Syntax.

Speicheraddressierung

Eine *Intel*-Syntax für indirekte Speicheraddressierung der Form

section: **[***base + index*scale+ disp***]**

wird in *AT&T*-Syntax zu:

section: disp **(***base, index, scale***)**

wobei *base* und *index* die optionalen 32-Bit Base- bzw. Indexregister sind, *disp* ist der optionale Speicheroffset und *scale* (1,2,4 bzw. 8) ist der Multiplikator für *index*, um die Adresse des Operanden zu berechnen. Ein Überschreiben des Default-Registers (z.B. **%ss** für Speicherzugriffe via **%esp** bzw. **%ebp**) kann durch den optionalen *section*-Prefix erfolgen.

Intel	*AT&T*
[ebp - 4]	**-4 (%ebp)**
[foo + eax*4]	**foo (, %eax, 4)**
[foo]	**foo** oder **foo (,1)**
gs: [foo + edi]	**%gs: foo (%edi)**

Unterprogrammaufrufe und Sprünge

Es gibt Unterprogrammaufrufe und Sprünge mit direkter bzw. indirekter Adressierung jeweils mit *near*- bzw. mit *far*-Adressierung.

Intel	*AT&T*	*Forth-Assembler*
call near foo	**call foo**	**call foo**
call dword ptr [foo]	**call *foo**	**call* foo**
jmp near foo	**jmp foo**	**jmp foo**
jmp dword ptr [foo]	**jmp *foo**	**jmp* foo**

Die Versionen mit *far*-Addressierung, hier der Vollständigkeit halber erwähnt, werden von dem Forth-Assembler z.Z. nicht unterstützt:

Intel	*AT&T*
call far foo	**lcall foo**
call tword ptr [foo]	**lcall *foo**
jmp far foo	**ljmp foo**
jmp tword ptr [foo]	**ljmp *foo**

Floating Point

Alle Datenypen, außer gepackte BCD, werden unterstützt. Gepackte BCD kann ohne große Schwierigkeiten nachgerüstet werden. Gültige Datentypen sind *short* (16 Bit), *long* (32 Bit) und *quad* (64 Bit) Integer, sowie *single* (32 Bit), *long* (64 Bit) und *temporary real* (80 Bit) Floating Point.

Der Quad-Integer-Datentyp wird vom 80387 nur über die Opcodes **fildq** (load quad integer to stack top) bzw. **fistpq** (store quad integer and pop stack) unterstützt. Entsprechendes gilt für den Temporary-Real-Datentyp mit den Opcodes **fldt** (load temporary real to stack top) und **fstpt** (store temporary real and pop stack).

Register-Register-Operationen erhalten den Suffix **l** (z.B. **faddl %st, %st(1)**).

Implementierung in Forth

Die Addressierungstechniken des 80386 sind gegenüber seinen 16-bittigen Vorgängern stark verallgemeinert worden. Durch die Verwendung von 32-Bit-Registern kann man nun problemlos große lineare Speicherbereiche addressieren. Die Codierung erfolgt mittels *Scaled Index Base*-Byte (SIB).

Daraus ergibt sich eine große Anzahl von Variationen Um die nötigen Definitionen auf ein Minimum zu beschränken, wurden folgende Klassen eingeführt:

$	Immediate Operanden
,	Separator, wertet Immediate oder Speicheroperanden aus
%reg	Zielregister
%reg,	Quellregister oder zweiter Teil der allgemeinen SIB-Adressierung
n(%reg)	indirekte Addressierung, Zieladresse
n(%reg),	indirekte Addressierung, Quelladresse
n(%reg,	erster Teil der allgemeinen SIB-Adressierung mit Baseregister
n(,	erster Teil der allgemeinen SIB-Adressierung ohne Baseregister
n)	dritter Teil der allgemeinen SIB-Adressierung, Zieladresse
n),	dritter Teil der allgemeinen SIB-Adressierung, Quelladresse
%freg	80387-Zielregister
%freg,	80387-Quellregister

Daraus ergeben sich einige Besonderheiten im Vergleich zum GNU-Assembler:

Es muß bei den Speicheradressierungen über Register immer ein Displacement angegeben werden, da sonst unterschiedliche Stacktiefen bei diesen Worten vorliegen würden. Bei einem Displacement von Null tritt somit ein Sonderfall auf, der extra behandelt werden muß. Beim derzeitigen GNU-Assembler (v. 2.3) wird im Gegensatz dazu bei **decl (%eax)** nicht derselbe Code erzeugt wie bei **decl 0 (%eax).**

Bei der allgemeinen SIB-Adressierung muß immer der Indexmultiplikator angegeben werden. Das kann durch eine Erweiterung der Klassen mit **: %reg) %reg, 1) ;** bzw. **: %reg), %reg, 1), ;** vermieden werden, kostet aber zusätzliche Wörterbucheinträge.

Es sind immer Leerzeichen zwischen den einzelnen Klassen notwendig, das ist durch die Verwendung des Forthinterpreters begründet.

Wie bei Forth-Assemblern üblich, können Forthworte ausgeführt werden (CELLS + @ usw.), um die Operanden zu berechnen.

Der Zustandsautomat

Um das Prinzip der Codegenerierung zu erläutern, gehen wir zunächst von einem Postfix-Assembler aus, d.h. aus **addl 0 (%ebx), %eax** wird **0 (%ebx), %eax addl.** Damit wird ersichtlich, daß die Operanden, ausgehend von einem Initialisierungszustand, den Opcode vorbereiten, und der dann mit dem abschließenden **addl** ins Wörterbuch eingetragen wird. Die nötige Prefixgenerierung wird durch eine Verzögerung von **addl** um eine Instruktion erreicht, doch dazu weiter unten mehr.

Für die Verwaltung der einzelnen Zustände wird ein Automat verwendet, dessen Zustand in der Variable **RSTATE** enthalten ist. Der Initialisierungszustand wurde mit festgelegt, die anderen Zustände sind aus der Zustandstabelle ersichtlich.

	-4	-3	-2	-1	0	1	2	3	4	10	20	30	100	200	300	400	401	1000
$	-	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
,	-	-	-	401	-2	-	-	-	-	-	-	-	100	200	300	-	-	-
%reg	-	30	20	10	-	-	-	-	-	-	-	-	-	-	-	4	-4	-
%reg,	-	-	-	-3	-	-	-	-	-	-	-	-	100	200	300	-	-	-
n(%reg)	-	3	2	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
n(%reg),	-	-	-	400	-	-	-	-	-	-	-	-	-	-	-	-	-	-
n(%reg,	-	300	200	100	-	-	-	-	-	-	-	-	-	-	-	-	-	-
n(,	-	300	200	100	-	-	-	-	-	-	-	-	-	-	-	-	-	-
n)	-	-	-	-	-	-	-	-	-	-	-	-	1	2	3	-	-	-
n),	-	-	-	-	-	-	-	-	-	-	-	-	400	-	-	-	-	-
%freg	-	-	-	1001	-	-	-	-	-	-	-	-	-	-	-	-	-	1002
%freg,	-	-	-	1000	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Dem aufmerksamen Leser werden die Dummy"-Zustandsänderungen der Klasse **,** bei den Zuständen 100/ 200/300 aufgefallen sein. In der verwendeteten Forthimplementierung traten Mehrdeutigkeiten bei den Namen **(%esp,**=**(%ebp)** bzw. **(%ebp,**=**(%esp)** auf, da die Namen im Wörterbuch mittels Hashalgorithmus auf 32 Bit komprimiert wurden. Daher wurden nur **(%esp** sowie **(%ebp** mit durch Leerzeichen getrenntem Komma verwendet, so daß eine Extrabehandlung des Kommas notwendig wurde.

Zustände, welche einen negativen Wert enthalten, dienen zur Kennzeichnung von direkten Speicheroperanden. In der ersten Spalte steht der Endzustand des Parserautomaten.

Ein Operand

-1	**instruction**	*mem*
10	**instruction**	*%reg*
1	**instruction**	*disp (%reg)*
1	**instruction**	*disp (, index, scale)*
1	**instruction**	*disp (base, index, scale)*
1001	**instruction**	*%freg*

Beispiel: *AT&T* `incb 0 (%ebx)`

Intel `inc byte ptr [ebx]`

Immediate Operand

-2	**instruction**	**$** *immed , mem*
20	**instruction**	**$** *immed , %reg*
2	**instruction**	**$** *immed , disp (%reg)*
2	**instruction**	**$** *immed , disp (, index, scale)*
2	**instruction**	**$** *immed , disp (base, index, scale)*

Beispiel:
AT&T `addl $ 100 , 2000 (%eax, %ebx, 4)`
Intel `add dword ptr [2000+eax+ebx*4], 100`

Register-Memory und Register-Register

-3	**instruction**	*%reg, mem*
30	**instruction**	*%reg, %reg*
3	**instruction**	*%reg, disp (%reg)*
3	**instruction**	*%reg, disp (, index, scale)*
3	**instruction**	*%reg, disp (base, index, scale)*
1002	**instruction**	*%freg, %freg*

Beispiel:
AT&T `addw %bx, %ax`
Intel `add ax, bx`

Memory-Register

-4	**instruction**	*mem , %reg*
4	**instruction**	*disp (%reg), %reg*
4	**instruction**	*disp (, index, scale), %reg*
4	**instruction**	*disp (base, index, scale), %reg*

Beispiel:
AT&T `movl 0 (, %ebx, 4), %eax`
Intel `mov eax, dword ptr [ebx*4]`

Implementierung des Prefixassemblers

Folgende Definitionen bilden die Basis für den Prefixassembler:

```
clear flags and state variable
: RESET ( -- ) ... ;

*** PREFIX OPERATIONS ***
holds pfa and ip of previous operand made with DOES>
CREATE 'PRIOR 0 , 0 , ( pfa ip )

execute previous operand and defer current operand
: PRIOR ( pfa1 -- pfa2 ) R> 'PRIOR 2@ 2SWAP 'PRIOR 2! >R ;

dummy operator used as 1st operand
: DUMMY-OP ( x -- ) DROP RESET ;

init with dummy operator and reset
(>BODY for threaded code)
: A: ( -- ) 0 ['] DUMMY-OP ( >BODY ) 'PRIOR 2! RESET ;

finish previous operand and reset
: A; ( -- ) 0 PRIOR DROP RESET ;
```

PRIOR wird immer als erstes Wort im DOES>-Teil der Opcodeworte aufgerufen, und führt den vorherigen Opcode aus, während es gleichzeitig die PFA und den IP der eigenen

Definition für die nachfolgenden **PRIOR** speichert. **A:** wird z.B. von **CODE** verwendet, wogegen **A;** einen Abschluß der vorherigen Instruktion erzwingt und in **END-CODE** sowie in den Makros verwendet wird. **RESET** setzt den Zustandsautomaten in den Initialisierungszustand.

Die Klasse **1MI** ist z.B. folgendermaßen definiert:

```
: 1MI
CREATE [ FORTH ] C,
DOES> PRIOR C@ [ ASSEMBLER ] C, RESET ;

HEX

37 1MI aaa
3F 1MI aas
0F8 1MI clc
0FC 1MI cld
0FA 1MI cli
```

Weitere Implementierungsdetails

Im Sourcecode sind die Instruktionen vermerkt, welche zur Zeit nicht unterstützt werden. Es sind entweder Instruktionen, welche priviligiert sind und somit dem Betriebssystem vorbehalten bleiben oder Instruktionen, die so selten verwendet werden, daß sich eine Implementierung nicht lohnt.

Bei der Verwendung von bedingten Sprüngen wurde eine Einschränkung der Sprungweite auf -128...+127 vorgenommen (*short jumps).* Obwohl der 80387 *near jumps* im Bereich -32768...32767 unterstützt, gilt das nicht für die Befehle **jcxz/jecxz** bzw. die **loop...-**Befehle, welche weiterhin nur 8-bittig sind. Deswegen wurden im Interesse der einfachen Implementierung der struktierierten Sprungbefehle grundsätzlich nur *short jumps* zugelassen. Alle Sprünge bzw. Schleifen müssen über diese strukturierten Sprunganweisungen vorgenommen werden. Dabei können Schleifen mit mehreren Ausgängen, wie in ANS-Forth zugelassen, verwendet werden, so daß sich keine wesentlichen Einschränkungen ergeben dürften.

necxz if, ... else, ... then,
```
jecxz L1
...
jmp L2
```
L1: ...
L2:

begin, ... nbe while, ... repeat,
L1: ...
```
jbe L2
...
jmp L1
```
L2: ...

begin, ... loop until,
```
L1: ...
loop L1
```

Verwendung in anderen Forth-Systemen

Der verwendete Assembler ist sicher für diejenigen interressant, die mit den GNU-Werkzeugen programmieren. Das zugrunde gelegte Forthsystem ist Subroutine-Threaded mit dem Register **%eax** als TOS. Als Stackpointer dient **%ebp** und als Returnstackpointer **%esp**.

Für abweichende Systeme (indirect/direct threaded, kein bzw. anderes TOS-Register) sind die entsprechenden Worte abzuändern:

A: für Threaded-Code muß >BODY verwendet werden

NEXT, PUSHTOS, POPTOS,CODE ;CODE END-CODE LABEL INLINE END-CODE END-LINE muß dem jeweiligem System angepaßt werden

Die im Abschnitt „**„Der Zustandsautomat**" behandelten Sonderfälle für **(%esp, (%esp) (%ebp,** und **(%ebp)** können entfallen, wenn man ein Forthsystem benutzt, indem die dort erwähnten Mehrdeutigkeiten bei den Wörterbucheinträgen nicht auftreten.

Anwendungsbeispiel

Als Anwendungsbeispiel wird eine Definition des Stringvergleichwortes **COMPARE** gezeigt.

```
CODE COMPARE ( c-addr1 u1 c-addr2 u2 -- n )
movl 0 cells (%ebp), %edi \ %edi=c-addr2 %eax=u2
movl 1 cells (%ebp), %ebx \ %ebx=u1
movl 2 cells (%ebp), %esi \ %esi=c-addr1
addl $ 3 cells , %ebp

\ %ecx=min(u1,u2)
movl %ebx, %ecx
cmpl %ebx, %eax
b if, movl %eax, %ecx then,

\ assume %es=%ds!
repz
cmpsb
nz if,
   s if, movl $ -1 , %ecx
   else, movl $ 1 , %ecx
   then,

\ test if substring is equal and compare length
else,
   cmpl %ebx, %eax
   nz if,
      b if, incl %ecx else, decl %ecx then,
   then,
then,

movl %ecx, %eax
NEXT,
END-CODE
```

❑

Serie: PC-Meßtechnik, Teil V

Der Joystick

Autor: Klaus Kohl / BMC Systeme GmbH

Zeppelinstr. 10; 86406 Mering; Tel.: 08233 / 30524

Bei den früher begehrten Heimcomputern war für Spiele der Joystick, meist in digitaler Form, ein unbedingtes Muß. Bei den ersten PC´s fehlte diese Schnittstelle und wurde erst mit dem PC junior, diesmal in teilweise analoger Form, wieder eingeführt. Heute besitzt fast jeder PC diese Schnittstelle, weil sie bei Multi-IO- oder Soundkarten zum Lieferumfang gehört.

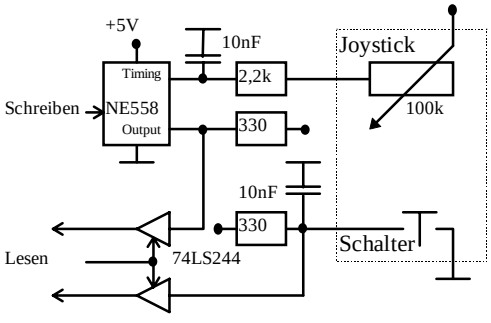
Stichworte: Meßtechnik, PC, Joystick

1. Aufbau und Programmierung

An einem Joystick-Interface können 4 digitale Schalter und 4 “analoge” Werte abgefragt werden. Da ein handelsüblicher Joystick nur zwei Schalter und einen zweiachsialen Schieber besitzt, könnten bis zu zwei Stück an einer Karte angeschlossen werden.

1.1 Interner Aufbau

Obwohl ich in der letzten Folge erwähnt hatte, daß der “analoge” Joystick ein echter A/D-Wandler ist, kann er nicht für die direkte Messung von Spannungen verwendet werden. Der Grund ist der interne Aufbau der Karte:



Beim Schreiben auf den Joystick-Port (meist $0201) wird der Kondensator am **Timing**-Pin nicht mehr auf Masse gezogen und gleichzeitig **Output** des NE558 auf 1 gesetzt. Über internen (zur Strombegrenzung) und externen Widerstand lädt sich dieser Kondensator wieder auf, bis die Schaltschwelle des NE558 erreicht und damit **Timing** auf Masse gezogen und **Output** wieder auf 0 gesetzt wird. Die Ermittlung des Wertes erfolgt über die Messung der Zeit vom Start bis zum Zurückschalten. Verwendet man die vom Betriebssystem zur Verfügung gestellten Routinen, so wird nur 0..255 (8 Bit) als Wert zurückgeliefert. Mit eigenen Abfragen oder anderen Widerstandswerten können aber auch höhere Auflösungen erreicht werden.

Der digitale Schalter kann direkt abgefragt werden. Hier ist nur zu beachten, daß ein gedrückter Schalter den Eingang auf Masse zieht und dadurch mit 0 gekennzeichnet ist.

1.2 Pinbelegung

Physikalische Schnittstelle ist eine unverwechselbare 15polige, zweireihige Buchse. Bemerkenswert ist hier, daß dies die einzige PC-Schnittstelle ist, bei der auch die +5V-Versorgung herausgelegt ist. Man muß deshalb besonders darauf achten, daß hier kein Kurzschluß entsteht.

Pin	Bezeichnung	Verwendung
1	+5V	Versorgung für Potentiometer
2	Taste A1	1. Taste des Joystick A
3	Stick-AX	X-Achse des Joystick A
4	Gnd	Masse
5	Gnd	Masse
6	Stick-AY	Y-Achse des Joystick A
7	Taste A2	2. Taste des Joystick A
8	+5V	Versorgung für Potentiometer
9	+5V	Versorgung für Potentiometer
10	Taste B1	1. Taste des Joystick B
11	Stick-BX	X-Achse des Joystick B
12	Gnd	Masse
13	Stick-BY	Y-Achse des Joystick B
14	Taste B2	2. Taste des Joystick B
15	+5V	Versorgung für Potentiometer

1.3 Portbelegung

Obwohl der Joystick nur eine Portadresse zum Start der Messung und Abfrage des Zustandes benötigt, werden mindestens 8 Adressen belegt. Üblicherweise wird die Adresse $0201 zur Ansteuerung verwendet. Die Belegung der Bits ist dabei wie folgt:

Bit	Verwendung
Schreiben auf Portadresse $0201	
Wert beliebig	Alle Joystick-Monoflops zurücksetzen
Lesen von Portadresse $0201	
0	Joystick AX-Koordinate
1	Joystick AY-Koordinate
2	Joystick BX-Koordinate
3	Joystick BY-Koordinate
4	Taster A1
5	Taster A2
6	Taster B1
7	Taster B2

1.4 Unterstützung durch (MS)DOS

Am leichtesten erkennt man im BIOS des PC´s durch Abfrage der BIOS-Variable auf Adresse $0040:$0011 (High-byte des Systemkonfigurationswort), ob ein Joystick-Port vorhanden ist. Bit 5 ist dann in diesem Byte gesetzt. Darüber hinaus zeigen auch die beiden über Interrupt $15/Funktion $84 zur Verfügung stehenden Abfrageroutinen bei fehlendem Joystickport einen Fehler durch gesetztes Carry-Flag an. Folgende, in KK-FORTH/8086-Assember geschriebene Beispiele, zeigen dessen Aufruf.

```
Code ms_getkeystate ( -- f 0 | -1 )
    tos push, $84 # ah mov, $00 # dx mov, $15 int,
    u< IF, -1 # tos mov,
    ELSE, $00f0 # ax and, ax push, 0 # tos mov,
    THEN, next,
End-Code
```

Unterfunktion $00 liefert neben dem Fehlerflag den aktuellen (invertierten) Status der Tasten in Bit 4..7 zurück. Wenn z.B. die erste Taste des Joysticks A gedrückt wird, wechselt Bit 4 von 1 auf 0. Fehlerflag -1 deutet auf eine fehlende Joystick-Karte hin, und es werden keine zusätzlichen Daten geliefert.

```
Code ms_getjoy ( -- y2 x2 y1 x1 0 | -1 )
    tos push, $84 # ah mov, $01 # dx mov, $15 int,
    u< IF, -1 # tos mov,
    ELSE, dx push, cx push, bx push, ax push,
        0 # tos mov,
    THEN, next,
End-Code
```

Falls kein Fehler auftrat, werden beide Joysticks gleichzeitig in beiden Achsen abgefragt und gemessene Werte zurückgeliefert. Solange kein Joystick angeschlossen ist, wird der Wert 0 zurückgeliefert. Ansonsten sind Werte zwischen 1 und 255 möglich und so berechnet, daß die Werte proportional der Meßzeit von 22us (Spannung von 5.0V über internen 2,2k-Widerstand an 10nF) bis 1ms (5V liegt über externen 100kOhm an) ist. Da die Meßzeit auf diese 1ms begrenzt wird, sind größere externe Widerstände nicht sinnvoll.

1.5 Eigene Abfrageprogramme

In eigenen Programmen, die auf schnellen Rechnern laufen, kann eine höhere Auflösung erreicht werden. Folgendes Beispiel für die X-Achse des Joystick A zeigt den prinzipiellen Ablauf dieser Abfrage.

```
Code get_ax ( -- x ) \ x=0 bei fehlenden Joystick
    tos push, $0201 # dx mov, 1 # bx mov,
    dx byte out, \ Start der Messung
    BEGIN, dx byte in, $01 # al and, \ Test Bit 0
        0= IF, bx tos mov, next, THEN,
        1 # bx add,
    0= UNTIL, bx tos mov, next,
End-Code
```

Beispielsweise werden auf einem 133MHz-Pentium in einer DOS-Box unter Windows95 durch diese Routine und einen Standard-Joystick Werte von 18 bis 1191 mit sehr geringem Rauschen (<±2Digit) geliefert, was einer effektiven Auflösung von 10Bit entspricht. Dabei ist zu beachten, daß das Timing durch Interrupt beeinflußt wird und dann einzelne Werte deutlich nach unten abweichen.

2. Einsatzmöglichkeiten

Die Verwendung der vier Digitaleingänge pro Joystick werde ich hier nicht näher erläutern, da es sich wie auch beim Druckerport um direkt abfragbare Eingänge handelt, welche intern mit Pull-Up und Glättungskondensator bestückt sind. Man sollte bei deren Verwendung nur darauf achten, daß die externen Schalter nicht mit einer zusätzlichen Spannung gespeist werden.

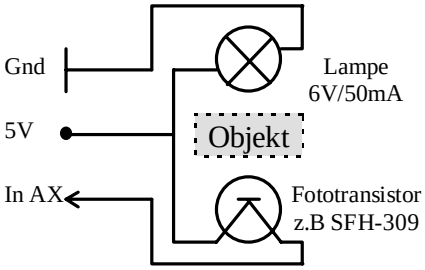
Bei den “analogen” Eingängen gibt es zwei prinzipielle Möglichkeiten zur Verwendung für die Meßtechnik. Die erste Version verwendet extern nur einen veränderlichen Widerstand wie beim Joystick. Für Sonderanwendungen kann es (wie im 3. Beispiel) notwendig sein, die internen Bauteile zu überbrücken. Die zweite Möglichkeit ist die Verwendung des gesamten 8Bit-Ports des Joysticks als Digitaleingang. Es werden dann auch an den 4 Widerstandseingängen eine digitale Spannung (0V oder 5V) z.B. aus einem 8Bit-A/D-Wandler angelegt. Man muß aber dann die Schaltung so

auslegen, daß pro “Analogpin” ständig die 2,3mA (5V/2.2k) fließen können.

Im folgenden möchte ich noch einige Beispiele für die Verwendung des Gameport erwähnen, die alle aus einer Artikelserie der Zeitschrift TOOLBOX von 1991 stammen.

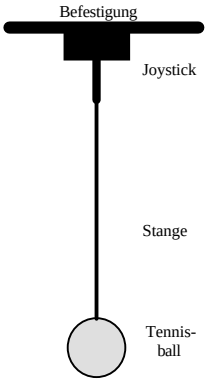
2.1 Pulsmessung

Fototransistoren haben die Eigenschaft, daß der Widerstand proportional zur auftreffenden Lichtmenge ist. Dies kann in einer einfachen Schaltung zur indirekten Messung des Pulsschlages genutzt werden. Sowohl in diesem als auch bei nachfolgenden Beispielen muß man immer darauf achten, daß der Mensch nicht mit spannungsführenden Teilen in Kontakt kommen kann. Es ist möglich, daß bei einem PC das Gehäusepotential nicht auf Masse ist oder durch einen internen Kurzschluß die Netzspannung nach außen geleitet wird.



Im Artikel wird einfach auf einer Wäscheklammer eine Lampe auf der einen Seite und der Fototransistor auf der anderen Seite angebracht. Diese Klammer wird dann so am Ringfinger befestigt, daß das Licht durch Fingernagel und Finger auf den Fototransistor fällt.

Großer Aufwand ist dann allerdings in der Software notwendig, da der gemessene Wert sehr stark von der Person und dessen Zustand abhängt. Vor der eigentlichen Messung ist erst eine Kalibrierung des Meßbereichs notwendig. Die Pulsfrequenz kann dann durch Auswertung der Meßkurve erfolgen.



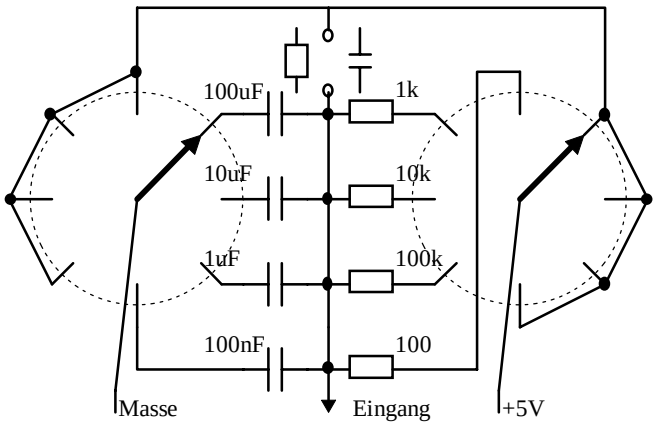
2.2 Windmessung

Ein gern aufgegriffenes Problem der Meßtechnik ist die Erfassung der Wetterdaten. Neben der einfach zu bestimmenden Temperatur sind vor allem windbezogene Werte wie Richtung und Geschwindigkeit oft nur mit teuerer Mechanik zu lösen. Eine billigere Lösung mittels eines umgebauten Joysticks wurde ebenfalls in der Zeitschrift TOOLBOX vorgestellt und dort mit entsprechenden Berechnungsformeln und Programm abgedruckt.

Dazu wird einfach über einen Verlängerungsstab ein Tennisball an einem umgekehrt aufgehängten Joystick befestigt. Da die Auslenkung beider Achsen von Windstärke und Windrichtung abhängt, können sie aus den gemessenen Werten errechnet werden.

2.3 Widerstand oder Kapazität

Falls man einen Lötkolben und eine übrige Joystick-Karte zur Hand hat, kann man das intern verwendete R/C-Glied nach außen verlagern und zur Messung eines unbekannten Widerstandes bzw. einer unbekannten Kapazität bei vorgegebenen Gegenpart (Kondensator bzw. Widerstand) über die



benötigte Schaltzeit heranziehen. Weil man hier schon beim Löten ist und seine Spiele weiterverwenden will, kann man (wenn nicht gerade eine Multi-IO-Karte verwendet wird) auch die Basisadresse der Testkarte ändern.

Die Wahl des Meßbereiches geschieht dabei über einen zweikanaligen, mehrpoligen Schalter. Referenzwiderstände sind relativ genau (<0.1%) erhältlich. Bei Kondensatoren ist diese Genauigkeit nicht erhältlich, und es muß deshalb der genaue Wert mittels Referenzmessungen ermittelt und berücksichtigt werden.

Literatur

(1) Tischler
PC Intern 1..5
Data Becker

(2) Thom Hogan
Die PC-Referenz für Programmierer
Microsoft Press

(3) Blank/Bernstein
PC-Schaltungstechnik in der Praxis
ISBN 3-89090-914-0
Markt-und-Technik-Verlag 1990

(4) Serie in Zeitschrift TOOLBOX
4´89 : Analoge Joysticks
9’90 : Joystick-Abfrage für Quick Basic
1’91 : Pulsmessung mit Joystick
3’91 : Widerstand und Kapazität messen
4’91 : Mehrere Schalter mit Joystick-Karte abfragen
5+6’91 : Gehirnstrom-Messung
7/8’91 : Windmessung

❑

Forth International

von Fred Behringer

Planegger Str. 24; D-81241 München

Das Wetter macht nicht an der Grenze halt. Und auch das Sommerloch ist international. Diesmal erhielt unser Kolumnist leider keine Antworten aus dem europäischen Ausland.

❑

Gehaltvolles

zusammengestellt und übertragen
von Fred Behringer

Forth Dimensions der Forth Interest Group, USA

März/April 1996

7 Coordinating Pygmy and C
Frank Sergeant

Mit ein paar Abänderungen kann Pygmy-Forth in C eingebunden werden. C übergibt dabei Strings, die von Pygmy INTERPRETiert werden. Pygmy kann C-Bibliotheksfunktionen oder Funktionen, die in C definiert wurden, aufrufen. Das Ganze läuft im wesentlichen wie folgt ab: Das einbindende C-Programm reserviert RAM-Speicherplatz, in das ein Abbild des Forth-Systems gelegt wird. Das umhüllende Programm in C enthält eine Tabelle, die die Adressen der Funktionen oder Strukturen (Variablen) aufnimmt, auf die Forth zugreifen können soll. C ruft das Forth-Abbild als eine Funktion auf und übergibt die Adresse der Tabelle. Über diese Tabelle kann Forth die C-Funktionen aufrufen oder die C-Strukturen lesen oder verändern. Wenn Forth seine Tätigkeit beendet hat, gibt es die Regie wieder an das einhüllende C-Programm ab.

21 Differential File Comparison
Wil Baden

"Aus Forth mehr herausholen": Es gibt kaum einen Programmierer, der nicht ein Hilfsprogramm zum Vergleich von Dateien gebrauchen könnte, ein Programm, das herausfinden kann, ob und an welchen Stellen zwei Quelltext-Dateien verschieden sind. Im Verlaufe einer längeren Reihe von kleinen Veränderungen, die ein schlecht funktionierendes Anwendungsprogramm schließlich zum Laufen bringen sollen, kann man leicht den Überblick über das gerade Vollbrachte verlieren. Dann muß ein Dateivergleichsprogramm her, das die abgeänderten Stellen anzeigt. Ein solches Vorhaben richtig anzugehen, ist keine leichte Aufgabe. Die Methode, an die man zunächst denkt, versagt schneller, als man sich versieht. Der Trick besteht vielmehr darin, nicht nach Abweichungen zu suchen, sondern die längste gemeinsame Zeichenteilfolge aufzuspüren - den längsten Block von Zeilen, die in beiden Dateien übereinstimmen, mit den nicht übereinstimmenden Teilen in unveränderter Reihenfolge dazwischen. Was übrig bleibt, sind die nicht übereinstimmenden Teile. Der Autor hat seine Version dieser Methode seit 1976 ständig verbessert und läßt uns in diesem Artikel an den Früchten seiner Bemühungen teilhaben.

30 Getting to the Hardware from Linux
Skip Carter

"Forthware": Leuten, die ohne Erfahrung mit Minicomputern und Workstations zu Linux überwechseln, wird es sicher einen Schlag versetzen, wenn sie erkennen, daß die Maschine nicht mehr vom Benutzer, sondern vom Betriebssystem beherrscht wird. Im vorliegenden Artikel werden dazu ein paar wesentliche Fragen angegangen: Welches Forth soll man nehmen? Wie erhält man Zugang zur Parallel-Schnittstelle?Wie lassen sich Gerätetreiber einbinden? Im Artikel enthalten ist der Linux-Code zum Thema Schrittmotoren aus dem letzten Heft.

❑

Ich möchte in dieser Kolumne in unregelmäßiger Folge über Forth-Aktivitäten in anderen Ländern berichten. Ich bin fleißig am Sammeln, würde mich aber natürlich auch über Zusendung von Material von anderer Seite freuen.

Verständlich sind für mich die Sprachen Englisch, Holländisch, Italienisch und Französisch.

Fred Behringer

Who is Who ? What is TOS?

von Arndt Klingelnberg

Straßburger Str. 12; D-52477 Alsdorf

Was enthält die Adresse an der Adresse von ... ? Ist es der pointer zu ... ?

Nicht nur wärend ·DEBUG ist man wißbegierig, was sich hinter TOS verbirgt.

Top Of Stack DEBUG F-PC tools

Einmal kann der Wert selbst von Interesse sein, dann sagt ein HEX-Wert manchesmal viel (Adresse oder Maske), manchmal wenig (Verkaufspreis, negativer Meßwert ...). Es kann auch ein (zwei) Character sein (CrLf, a, ...). TOS kann aber auch ein execution token (cfa) sein (’ DUP , ’ <run>) oder das einer ·DEFER-red Funktion (' UNIBUTTON) oder eine Adresse, und zwar einer Variablen (pfa == body) (FUDGE) oder eines Pointers sein.

Wesentlich ist also die Darstellung als (unsigned) HEX, als signed (!) decimal, als double-char, als cfa (·’) und pfa (·&>) eines Forth Wortes (Dekompilierung der Wortinhalte).

Mann/Frau sucht immer das, was man nicht findet. Hier sollte auch all das gefunden werden, was man/frau gar nicht zu finden wagte.

Noch weitere Ideen...

... zu möglichen Verbesserungen oder Verkomplizierungen:

* Automatische Analyse, ob es sich möglicherweise um eine cfa ODER pfa handelt, und ob eine weitere Analyse des jeweils anderen Wertes (>body oder body>) ein mehr an Information bringt.
* Automatische Analyse, ob es eine cfa oder pfa zu einem USER-Wort ist, dann werden entsprechend die USER -Inhalte ermittelt/ausgewertet.
* Automatische Analyse, ob der body auf eine cfa zeigt, die wiederum ein deferred word ist, ... (aber nicht schrankenlos rekursiv).
* Dazu sollte ein mögliches Cashen abgefangen werden, da z.B. ein $FFFF @ (also das bei flagWorten mögliche: TRUE @) NICHT zulässig ist !
*

Eine recht aufwendige und informative Version ist in F-PC-ak ab 4.23 integriert. Dort wird auch z.B. bei

’ aRUNsave ??

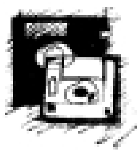
oder

’ EMIT ??

eine nutzbare Information geliefert. Die Darstellung in Hex, Decimal und ASCII und (!) Double (32bit) ist allerdings dort bereits im PullDownStack (Ctrl-RShift) vorhanden.

❑

File: fastinfo.seq vom 24.1.1994



```
\ fastinfo.seq   for easy debugging during ·DEBUG  or interactive Forth    \ akg93dec17

comment:    vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv\ akg94jan24

                        SOME RESULTS of ·?? ·$?? ·"??
                    --------------------------------

( a variable )
          fudge ??
          $367E =13950 =~6= ’ NO-NAME = &> FUDGE
          ( @ ==> $420 =1056 =   = ’ NO-NAME = &> NO-NAME )

( a cfa of a deferred word )
          ’ emit ??
          $ADE =2782 =?= ’ EMIT = &> NO-NAME
          ( @ ==> $35E8 =13800 =?5= ’ NO-NAME = &> NO-NAME )

( a pfa of a deferred word )
          ’ run >body ??
          $1D7C =7548 =|= ’ NO-NAME = &> RUN
          ( @ ==> $1D74 =7540 =t= ’ <RUN> = &> NO-NAME )

( a counted string )
          c"    123456    " $?? ?   123456    ?

( an AddressLength string )
          s" abc" "?? ?abc?

          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
comment;

\\  //  \\  //  \\  //  \\  //  \\  //  \\  //  \\  //  \\  //    \ akg94jan24

   get information about TopOfStack, very comfortable/informative during·DEBUG

   all NONdestructive words use '??'-names

   for NON- F-PC   change            fEMIT              to    EMIT
                                     '$'                to    ASCII $
                               or    '$'                to    [char] $
                               or    '?' femit          to    ." ?"

   >name.id          ( cfa -- )        \ prints name for cfa , uses ·>name

  old mini version
  \ : ??              ( n -- n )                        \ ..       \ akg93aug05
  \        dup @
  \        BASE @ >R  dup '$' emit HEX U. '=' emit DECIMAL .  R> BASE ! ;   \ akg93aug05

{

: .=              ( n -- )          \ print in different formats                 \ akg93dec17
         '$' emit  HEX dup U.  '=' emit  DECIMAL dup .  '=' emit
         sp@ 2 type  ." = ' "  dup >name.id  ." = &> "  body> >name.id ;

: ??              ( n -- n )                         \ ..      \ akg93aug05    \ akg93dec17
         BASE @ >R
  \        dup @
  \        BASE @ >R  dup '$' emit HEX U. '=' emit DECIMAL .  R> BASE ! ;   \ akg93aug05
         cr
         dup      .=
         50 ?line  ." ( @ ==> "
         dup  @  .=
         ." ) "
         50 ?line  ." (( >body @ ==> "
         dup  >body @  .=
         ." )) "
         cr
         R> BASE ! ;

: ??              ( n -- n )                         \ ..      \ akg93aug05    \ akg93dec17
         BASE @ >R
         cr
         >R
         R@ >body @                        \ contents at body ( at pfa )
         R@ @                              \ contents
         R@                               \ value itself
         \ we get the values first, might change during following display ...
         .=
         50 ?line  ." ( @ ==> " .= ." ) "
         50 ?line  ." (( >body @ ==> " .= ." )) "
         cr
         R>
         R> BASE ! ;

: "??             ( A L - A L )   \ NONdestructively display ·ALstring
    \     2dup type ;              \ "..
         space '?' femit  2dup type  '?' femit ;          \ "..                \ akg93dec17
                   \ show heading and trailing spaces

: $??             ( $ -- $ )        \ NONdestructively display ·$string
    \     dup count type ;          \ $..
         space '?' femit  dup count type  '?' femit ;     \ $..                \ akg93dec17
```

Längste Zeile hat 86 Zeichen

Sorry, aber manche Dos-Zeichen werden als '?' dargestellt /clv

INI-File für WinView.exe

Einstellungen unter Windows 3.1 dauerhaft speichern

von Martin Bitter
Möllenkampweg1 1a, 46499 Hamminkeln

Tom Zimmers (und anderer) neuestem Kind WIN32FOR liegt als Editor und Demonstration der Möglichkeiten das mit WIN32FOR erstellte Programm WINVIEW.EXE bei. Darin können einige Einstellungen gemacht werden, die u.a. Fontgröße, Zugriffspfade und Dateiendungen betreffen. Ursprünglich wurde WIN32FOR für WindowsNT geschrieben, es läuft aber unter Win95 und sogar unter Windows3.1. Letzteres erfordert den Windows Zusatz WIN32.S Das ganze ist so laufsicher und "gnädig", daß API-Aufrufe, die es unter Windows3.1 nicht gibt und die von Win32.s nicht nachgebildet werden, im positiven Sinne folgenlos bleiben. Dazu gehören auch die Befehle mit denen WinView normalerweise seine Einstellungen in einer REGISTRY-Struktur unter WIN95 oder WindowNT speichert. Sie bleiben folgenlos. Das System stürzt nicht ab (toll!), aber die Einstellungen werden ebenso nicht behalten. Das abgedruckte File WinV-Ini.F behebt dieses Manko.

*Stichworte: Win32For WinView *.INI_Files*

File: winv-ini.f vom 28.7.1996

```
(( M. Bitter 30.6. 1996

    Windows95 hat ein anders Konzept programmspezifische Informationen zu speichern,
    als es Windows 3.1 und damit Windows 3.1S haben. Unter Windows 95 wird in eine
    Registry geschrieben, Windows 3.1 kennt *.INI-Files.
    Win32for (Vers. 3.1 Build: 0160) verwendet die Registry, deshalb ist eine
    Adaption für Windows 3.1S notwendig.
    Die Aufrufe RegSetString bzw. RegGetString von Tom Zimmer bekommen hier die
    gleichartigen Aufrufe Write>ini bzw. Get>ini zur Seite gestellt.
    Insbesondere habe ich auf ein gleiches Parameterpassing geachtet, so daß
    z.B. in WinView.f "GetDefault und "SetDefault sehr einfach angepasst werden
    können:

    aus : "GetDefault ( a1 n1 -- a2 n2 )
                        s" Settings" RegGetString ;
    wird nun:

   : "GetDefault ( a1 n1 -- a2 n2 )
      s" c:\win32for\winview.ini" INI-File place INI-File +null
      s" Settings" getini ;

    aus: : "SetDefault ( a1 n1 a2 n2 -- )
                      s" Settings" RegSetString ;
    wird:

   : "SetDefault ( a1 n1 a2 n2 -- )
     s" c:\win32for\winview.ini" INI-File place INI-File +null
     s" Settings" setini ;


))

wincon also

Create INI-File max-path here ," c:\win32for\Test.ini" 0 , here - + allot


: setini ( dadr dlen vadr vlen sadr slen -- )
   INI-file count                                \
   pad dup >R place r@ +null R> count 1+ +       \ F-name
       dup >R place r@ +null R> count 1+ +       \ section
       dup >R place r@ +null R> count 1+ +       \ key
       dup >R place r> +null                     \ data
   pad count 1+ over + swap rel>abs swap         \ F-name
       count 1+ over + swap rel>abs swap         \ section
       count 1+ over + swap rel>abs swap         \ key
             1+              rel>abs             \ data
   swap rot                                      \ richtige Reihenfolge herstellen
   call WritePrivateProfileString drop ;         \ Win-function


: getini ( vadr vlen sadr slen -- dadr dlen )
   pad off                                       \ sicher ist sicher
   INI-file count                                \
   pad dup >R place r@ +null R> count 1+ +       \ F-name
       dup >R place r@ +null R> count 1+ +       \ section
       dup >R place r> +null                     \ key
   pad count 1+ over + swap rel>abs swap         \ F-name
       60 swap                                   \ maximale Stringlänge
   pad rel>abs dup rot                           \ Zieladr. hier = Vorgabewert
   count 1+ over + swap rel>abs swap             \ section
         1+                rel>abs               \ key
   swap                                          \ richtige Reihenfolge herstellen
call GetPrivateProfileString pad swap ;          \ Win-funktion



: test>ini
45 s>d (d.) s" fünfundvierzig" s" Einstellungen" setINI ;

: test<ini
  s" fünfundvierzig" s" Einstellungen" getini ;


L„ngsteZeilehat 84 Zeichen
```

Wenn alle Brünnlein fließen ...

Grafische Darstellung der Fließrichtung von Grundwässern

von Martin Bitter
Möllenkampweg 1a, 46499 Hamminkeln,

Die Ermittlung der Fließrichtung eines Grundwassers scheint eine prinzipiell einfache Aufgabe zu sein. Gelingt ihre Lösung ohne Zuhilfenahme von Sinusfunktion, Floating Point Operationen und Matrizen- (Vektor)rechnung?

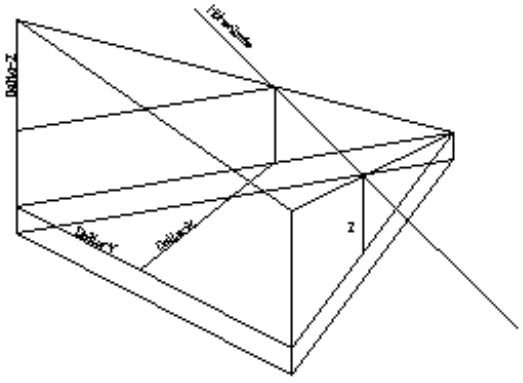
Stichworte: ZF Skalierungbefehl Strahlensatz Grundwasser

WASSERVD.SEQ 22.020 27.07.96
VGA_EMIT.SEQ 25.216 07.06.96
NEW_NUMB.SEQ 4.513 07.06.96
SINUS.SEQ 7.239 23.07.96

Innerhalb der Forth-Gruppe-Moers hatte (oder stellte?) Fritz Prinz folgende Frage: Gelingt es mit den Mitteln eines einfachen Forth, die Fließrichtung von Grundwässern grafisch darzustellen, so daß sie halbautomatisch in eine Kartendarstellung umgewandelt werden könnte?

Wo fließen sie denn?

Zur Ermittlung der Fließrichtung der Grundwässer wird das entsprechende Gebiet in ein Netz von Dreiecken untergliedert. An den Eckpunkten der Dreiecke werden Bohrungen bis zum Grundwasserspiegel niedergebracht und die Höhe des Grundwassers gemessen. Unter der Annahme, daß innerhalb des jeweiligen Dreiecks die Fließverhältnisse homogen sind, wird für jedes Dreieck eine Linie ermittelt, unter der das Grundwasser gleich hoch steht - die Höhenlinie. Rechtwinklig zu dieser Linie verläuft der Grundwasserfluß in Richtung der größeren Tiefe.



Strahlemann und Söhne

Die notwendigen Berechnungen lassen sich mit Hilfe des Strahlensatzes durchführen. (Für ähnliche Dreiecke sind die Verhältnisse entsprechender Seiten gleich.) Zu einer Seite des Dreiecks wird der Mittelpunkt ermittelt, indem der Mittelwert der Koordinaten der Eckpunkte gebildet wird. Zu diesem Punkt (Höhenpunkt) wird der Gegenpunkt berechnet. Das ist der Punkt einer zweiten Seite, der die gleiche Höhe hat: Durch geeignete Auswahl wird die Seite bestimmt, die diese Höhe schneidet. Senkrecht unter dieser Seite spannt sich ein Dreieck auf, das beschrieben wird durch die geographischen Koordinaten und durch den bekannten Höhenunterschied des Grundwassers an diesen Koordinaten.

Dieser Höhenunterschied wird zu der Höhe des Höhenpunktes ins Verhältnis gesetzt,. Mit diesem Verhältnis und der Kenntnis des Strahlensatzes lassen sich nun die geographischen Koordinaten dieses Punktes bestimmen.

$$z / \Delta Z = x / \Delta x = y / \Delta y$$

(z=Höhe, x, y=geographische Koordinaten)

Die Verbindung der beiden Punkte ist die Höhenlinie.

Eine Gerade im rechten Winkel zu einer bekannten Geraden, erhält man, indem die Steigung der bekannten Geraden umgedreht wird, d.h. ihr Kehrwert gebildet wird. Die Fließrichtung wird auf diese Art ermittelt und dargestellt. Auf eine Darstellung ihrer Steilheit in Z-Richtung habe ich, ebenso wie auf ein Clipping, verzichtet. Die Länge der Geraden, die die Fließrichtung angibt, zeigt also nicht die Steilheit des Gefälles an.

Hinfort(h) mit dir ...

Der Strahlensatz drückt eine geometrische Regel durch algebraische Verhältnisbildung aus. Daher bietet es sich an, mit Skalierungsbefehlen zu arbeiten. Sehr leicht tauchen bei verschiedenen Dreiecken ungünstige Zahlen auf, so daß das Zwischenprodukt oder gar das Endprodukt den gültigen Wertebereich überschreitet. Dem wurde durch ein neues Wort s*/ (symmetric-times-divide) begegnet, das zum einen Divisionen durch Null und Wertüberschreitungen gutmütig behandelt, und zum anderen von der üblichen floored Rundung abweicht, indem es zu Null hin rundet (-0,5->0).

Die Ermittlung der Höhenlinie (und der Fließrichtung) ist fast trivial, aber ihr Clippen, also die Verlängerung bis zu den Fenstergrenzen, erweist sich als sehr verzwickt. Glücklich all die, bei denen das Betriebsystem oder das GUI diese Aufgabe erledigt!

Der X-Wert eines Punktes der Verlängerung zu einem bestimmten Y-Wert kann mit Hilfe des Strahlensatzes leicht gefunden werden. Fragt man jedoch alle vier Seiten ab, so kommt es immer dann, wenn diese Verlängerung durch eine Ecke verläuft, zu doppelten Werten, denn es werden dort ja zwei Rahmenseiten gleichzeitig geschnitten. Durch Rundungsfehler, die, wenn auch nicht so stark, selbst bei 68-bit floating-point auftreten, tritt dieser Effekt nicht nur in den Ecken, sondern schon nahe den Ecken auf. Von potentiell vier gültigen Lösungen müssen also bis zu zwei Lösungen ignoriert werden.

Alles weitere steht im (hoffentlich) ausreichend kommentierten Quelltext.

Horch was kommt von draußen rein ...

WASSERVD.SEQ ist in ZF-Forth geschrieben.

Zur Darstellung wurde der externe Grafikbefehl LINIE benutzt (F.Prinz GRAPH.SEQ Forth-Gruppe-Moers), der, da auf Geschwindigkeit optimiert, wenig Sicherheit bietet. (Clipping, Start- und Endpunkt gleich). Vermißt wurde die Möglichkeit, Text und Grafik auf einem Bildschirm auszugeben. In der Forthgruppe Moers sind zur Zeit drei verschiedene Möglichkeiten der Grafiktextausgabe in Erprobung. Eine wird hier verwendet, um die Höhen der Dreieckspunkte, ihre Namen und programminterne Werte anzuzeigen.(GEM-TEXT.SEQ, VGA-TEXT.SEQ Forth-Gruppe-Moers) Wer über diese oder ähnliche Funktionen nicht verfügt, muß die entsprechenden Zeilen auskommentieren.

Bei der Definition von s*/ werden Präfixe zur Anzeige der Basis verwendet, $7FFF bzw. $8001 (NEW-NUMB.SEQ M.Bitter Forth-Gruppe-Moers). Besorgen oder mit HEX arbeiten.

Zum Testen des Codes werden Dreiecke mittels SINUS getaumelt (SINUS.SEQ M.Major nach Zech Forth-Gruppe Moers). Besorgen, selbst Schreiben oder anders testen.

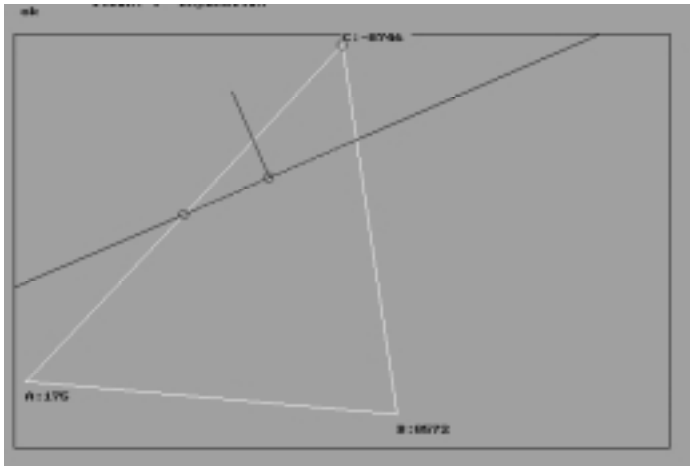
WASSERVD.SEQ NEW_NUM.SEQ und VGA_EMIT.SEQ befinden sich auf der Listingdiskette. ❑

File: wasservd.seq vom 27.7.1996

```
Wasser.seq vom 6.7 1996



\ einige Worte für Fritzens Wasserprogramm
\ lt. Fritz sind die Koordinaten der Eckpunkte eines Dreiecks gegeben.

fload vga_text
Fload sinus


\ In diesem Programm wird gelegentlich Gebrauch von Skalierungsbefehlen
\ gemacht. Leider führt dabei ein Überschreiten des gültigen Zahlenbe-
\ reiches (vor dem ja */ schützen soll) zu einer Division durch Null mit
\ der Folge eines Absturzes.
\ Ausprobieren: 32000 32000 32000 */ --> 32000
\                    32000 32000 15000 */ --> Absturz
\ Ist das Ergebnis größer als mit 16 bit darstellbar, geht's in Nirwana.
\ Abhilfe schafft S*/. Sprich Symmetric-times-divide.
\ Zwar liefert auch das normale m/mod Informationen über einen Ergebnisüber-
\ lauf (bsp. Rest absolut größer als der Teiler, bzw. nicht passende Vor-
\ zeichen bei Rest und Teiler) aber auch das müßte abgefragt werden. Zudem
\ wurde ein */ gewünscht, daß um Null symmetrisch ist.

: s*/ ( n n n - n )
   dup 0=                                  \ Division durch Null?
   IF drop *d drop 0<_>                    \ Zwischenprodukt negativ?
      IF $8001 ELSE $7FFF THEN               \ ja --> kleinstes; nein --> größtest
   ELSE                                    \ Egebnis zurück
   3dup                                    \ Kopie der drei obersten Einträge
   dup abs /   -rot                        \ vorzeichen Teiler ermitteln
   *d nip 0<_>IF -1 ELSE 1 THEN * >r       \ mal Vorzeichen Zwischenprodukt; zum TOR
   abs -rot *d dabs rot                    \ ab jetzt wird absolut gerechnet
   um/mod swap 0<_>                          \ Rest neg.? (Fehler!=Überlauf)
   IF drop $7FFF                           \ ja --> größtmögliches Ergebniss
   ELSE dup $7FFF u>                       \ nein --> Ergebnis größer als gewünscht?
      IF drop $7FFF THEN                   \ ja --> größmöglichstes Ergebnis
   THEN                                    \ Verzweigung ende (TOS= 15bit absolut)
   R> *                                    \ mit Vorzeichen mul. (16bit)
   THEN ;
\ ------------------------------------------------------------------+
\  Definition und Zugriff auf Punkte mit drei Koordinaten (x,y,z)      |
\ ------------------------------------------------------------------+

3variable create 6 allot ;

: 3! ( n n n adr -- )
   tuck 4 + ! tuck 2+ ! ! ;

: 3@ ( adr -- n n n )
   dup @ swap 2+ dup @ swap 2+ @ ;

: x@   ( adr -- n )    @ ;
: y@   ( adr -- n )    2+ @ ;
: xy@ ( adr -- x y ) 2@ swap ;
: z@   ( adr -- n)     4 + @ ;
: z!   ( n adr -- )    4 + ! ;

\ ------------------------------------------------------------------+
\  Variablen und Konstanten einrichten und vorbesetzen                |
\ ------------------------------------------------------------------+

3variable pa                \ Punkt a
3variable pb                \ Punkt b
3variable pc                \ Punkt c
3variable pmitte            \ Mittelpunkt der niedrigsten Seite

3variable tiefster        \ tiefster Punkt  \
3variable hoechster       \ höchster Punkt    > --- entweder A B oder C
3variable mittlerer       \ mittlerer Punkt /

 10 Constant links         \ linker Rand des Ausgabefensters
 80 Constant oben          \ oberer Rand des Ausgabefensters
610 Constant rechts        \ rechter Rand
460 Constant unten         \ unterer Rand

: Breite ( -- n )          \ Breite des Ausgabefensters
   rechts links - ;

: Hoehe   ( -- n )         \ Höhe des Ausagabefensters
   unten oben - ;

1 constant masstab         \ Faktor der Verkleinerung

: masst/ ( n -- n ) masstab / ; \ maßstäblich verkleinern
: masst* ( n -- n ) masstab * ; \ maßstäblich vergrößern

: skal_xy ( vx vy -- x y )        \ Umwandlung: virtuell xy nach Bildschirm-xy
   masst/ oben  + swap               \ durch Maßstab; plus y-Offset
   masst/ links + swap ;             \ dito -x-

: skal_linie ( x y x y -- )       \ zeichnet eine maßstabgerechte Linie
   skal_xy 2swap skal_xy             \ x-y-Koordinaten umrechnen
   linie ;                           \ Linie zeichen

: rahmen ( -- )
   blk farbe !                                \ Rahmenfarbe schwarz
   rechts oben links oben     linie          \ obere
   rechts unten links unten   linie          \ untere
   rechts oben rechts unten   linie          \ linke
   links oben links unten     linie ;        \ untere Begrenzungslinie
```

```
104
105 : dreieck                                        \ zeichnet das Dreick ABC
106   pa xy@ pb xy@ skal_linie                       \ Seite c
107   pb xy@ pc xy@ skal_linie                       \ Seite a
108   pc xy@ pa xy@ skal_linie ;                     \ Seite
109
110 : gross ( n -- )                                 \ vergrößert das Dreieck um n
111   pa 3@ >R 2 pick * swap 2 pick * swap r> pa 3!
112   pb 3@ >R 2 pick * swap 2 pick * swap r> pb 3!
113   pc 3@ >R 2 pick * swap 2 pick * swap r> pc 3! drop ;
114
115 : fachgenau ( n -- )              \ vergrößert Dreieck und maßstab um n
116   dup gross =: masstab ;           \ d.h. die Ausgabe verändert sich nicht,
117                                      \ es wird aber mit größeren Zahlen gerechnet
118                                      \ Rundungfehler wirken sich weniger aus.
119 \ ************** Ende der Zeichenroutinen ***********************************
120
121 : sortiere_punkte ( -- adr adr adr )          \ sortiert die Punkte nach Höhe
122   pa z@ pb z@ > IF pa pb ELSE pb pa THEN       \ Höhe A,B vergleichen
123   pc z@ pb z@ > pc z@ pa z@ > and               \ C niedriger als A und B?
124       IF pc -rot EXIT THEN                     \ falls ja: unter A B legen: Schluß
125   pc z@ pb z@ <_>pc z@ pa z@ <_>and             \ C höher als A und B?
126       IF pc EXIT THEN                          \ falls ja: C zum TOS; Schluß
127   pc swap ;                                    \ ansonsten C zwischen A und B
128
129
130 : zuordnen ( -- )          \ speichert die höhensortierten Punkte A,B und C
131   sortiere_punkte          \ Adressen der Punkte nach Höhe sortiert zum Stack
132   3@ tiefster 3!           \   \
133   3@ mittlerer 3!          \    >-- Koordinaten holen, neuen Punkten zuordnen
134   3@ hoechster 3! ;        \   /
135
136 : calc_mittelpunkt ( x y z x y z -- x y z )
137   >R rot >R rot             \ jeweils z zum R-Stack, ( x x1 y1 y )
138   + 2/ -rot                 \ y-Mittel; unter die beiden x-werte legen
139   + 2/ swap                  \ x-Mittel; x-Werte nach unten legen ( x y )
140   r> r> + 2/ ;              \ z-Werte holen; Mittel bilden
141
142 : steilste ( -- p1 p2 )           \ Punkte mit dem zweitgrößten Höhenunterschied
143   mittlerer                        \ dieser Punkt ist immer dabei
144   hoechster z@ mittlerer z@ -    \ Höhenunterschied höchster und mittlerer
145   mittlerer z@ tiefster z@  -    \ bzw. mittlerer und tiefster Punkt
146   > IF hoechster                   \ vergleichen und max. unterschiedliche
147     ELSE tiefster                  \ Punkte auf den Stack legen
148     THEN ;
149
150 : mittelpunkt ( -- )             \ Mittelpunkt der steilsten Seite errechnen
151   steilste >r 3@ r> 3@
152   calc_mittelpunkt               \ Mittelpunkt berechnen
153   pmitte 3! ;                    \ und merken
154
155 : pi ( -- x y )                   \ ISO-punkt (Punkt gleicher Höhe d. Gegenseite)
156   pmitte      z@ tiefster z@ -     \ delta z-z'
157   hoechster z@ tiefster z@ -     \ delta z
158   2dup                             \ Kopie für weitere Verwendung
159   hoechster x@ tiefster x@ -     \ delta x
160   -rot */ tiefster x@ + -rot     \ delta-x * ( delta-z / delta z-z')
161   hoechster y@ tiefster y@ -     \ delta y
162   -rot */ tiefster y@ + ;        \ delta-y * ( delta-z / delta z-z')
163
164 : hoehenlinie   ( -- )           \ zeichnet Isomere (Linie der gleichen Höhe)
165   weiss                            \ Zeichenfarbe ist weiss
166   zuordnen mittelpunkt             \ Punkte nach Höhe ordnen, Startpunkt
167   pmitte xy@ pi skal_linie ;     \ Koordinaten holen; dito Höhenpunkt; zeichnen
168
169 \ ***************************************************************************
170 \ Hier endet die Berechnung der Höhenlinie. Es folgen:                      |
171 \ - Die Berechnung einer Cliplinie, das ist die Linie, die die Höhen-      |
172 \    linie bis zum Rand des Darstellungsrahmens verlängert. Hier erwies    |
173 \    sich das Aussieben überflüssiger Werte in den Ecken als sehr verzwickt.|
174 \ - Die Darstellung der Fließrichtung, das ist eine Gerade, die im Mittel- |
175 \    punkt der Höhenlinie rechtwinklig abzweigt und zu der tieferen Seite  |
176 \    des Dreiecks zeigt, d.h. vom höchsten Punkt weg.                      |
177 \ ***************************************************************************
178
179 : steigung_mi ( -- dx dy )       \ Steigung der Höhenlinie als Zahlenpaar
180   pmitte xy@ pi rot swap -       \ delta x
181   -rot swap - ;                  \ delta y
182
183 : 0dabei? ( n n -- flag )        \ ist einer der Werte Null? (divison/0!)
184   0= swap 0= or ;                  \ d.h. Steigung waagerecht oder senkrecht
185
186 : waagerecht? ( -- flag )       \ ist die Gerade waagerecht?
187   steigung_mi drop 0= ;
188
189 : senkrecht? ( -- flag )        \ ist sie senkrecht?
190   steigung_mi nip 0= ;
191
192 : y(xn) ( x -- x y )             \ errechnet y-Wert zu gegebenem x
193   dup                              \ x-Wert kopieren
194   pmitte xy@                       \ x y des Mittelpunktes holen
195   >r swap - r> swap steigung_mi   \ x-Differenz bilden; x y der Steigung holen
196   s*/ + ;                          \ Delta-Y bilden, zu y-pmitte addieren
197
198 : x(yn) ( y -- x y)               \ errechnet x-Wert zu gegebenem y
199   dup                              \ y-Wert kopieren
200   pmitte xy@                       \ x y des Mittelpunktes holen
201   rot - steigung_mi swap          \ y-Differenz bilden; x y der Steigung holen
202   s*/ + swap ;                     \ Delta-x bilden, zu x-pmitte addieren
203
204 : x_innen? ( x y -- x y f )      \ liegt der x-Wert innerhalb des Rahmens?
205   over 0 breite masst* between ;
206
207 : y_innen? ( x y -- x y f )      \ dito für den y-Wert
208   dup 0 hoehe masst* between ;
209
210 : x_clippunkt? ( y -- FALSE / x y TRUE )
211   masst* x(yn) x_innen? dup not IF -rot 2drop THEN ;
212
213 : y_clippunkt? ( x -- FALSE / x y TRUE )
214   masst* y(xn) y_innen? dup not IF -rot 2drop THEN ;
215
216 \ Clippoints berechnet jeweils einen Schnittpunkt der Höhenlinie mit
217 \ einer Rahmenlinie, überprüft, ob die X-Y-Werte genehm sind und wirft
218 \ sie entweder weg, oder behält sie auf dem Stack.
219 \ Bspl.: Eine Abfrage auf x=600 liefert richtig y=380,
220 \          Eine Abfrage auf y=380 liefert ebenfalls richtig x=600
221 \ Liegt ein Punkt genau in einer Ecke, so gibt es zweimal eine richtige
222 \ Wertepaarung, das ist eine zuviel.
223 \ Daher werden die gültigen Punkte gezählt, sind es drei muß einer wegge-
224 \ worfen werden, sind es vier, müssen zwei Punkte weggeworfen werden.
225
226 5 Constant Rundungsfehler
227
228 : toleranz ( n -- n1 n2 )         \ bildet zu n die Fehlergrenzen
229   dup Rundungsfehler masst*       \ Rundungsfehler maßstäblich berücksichtigen
230   dup negate                       \ Plus-Minus des Fehlers
231   rot + -rot + ;                   \ Anfang-Ende des Toleranzbereiches
232
233 : zu_nahe? ( x y x1 y1 -- f )    \ liegen zwei Punkte nahe beieinander?
234   rot                              \ y zum TOS holen
235   toleranz between -rot            \ ist y1 innerhalb der Toleranz um y?; merken
236   toleranz between and ;           \ dito x; traf das für x und y zu?
237
238 : doppel_weg ( x y x1 y1 x2 y2 -- xn yn xn1 yn1 ) \ Doubletten entfernen
239                 4dup zu_nahe? IF 2drop THEN          \ zwei oberste Punkte vergl.
240           >R >R 4dup zu_nahe? IF 2drop THEN R> R>    \ zwei untere Punkte vergl.
241   2swap >R >R 4dup zu_nahe? IF 2drop THEN R> R> ; \ ober-unter Punkte verg.
242
243 : clippoints ( -- x y x y )              \ Koordinaten gültig. Punkte
244   0                                        \ Dummywert für Zähler
245   0       y_clippunkt? IF rot 1+ THEN     \ Schnitt linker Rand; evtll. zählen
246   breite y_clippunkt? IF rot 1+ THEN      \ Schnitt rechter Rand; zählen
247   0       x_clippunkt? IF rot 1+ THEN     \ Schnitt oberer Rand; zählen
248   hoehe  x_clippunkt? IF rot 1+ THEN      \ Scnitt unterer Rand; zählen
249   dup 2 = IF drop                          \ Zähler=2--> ok, Stack aufräumen
250           ELSE 4 =                         \ nein --> neuer Frage: sind es vier
251             IF   doppel_weg doppel_weg    \ ja --> zwei Doubletten vernichten
252             ELSE doppel_weg                \ nein (=3) eine Doublette vernichten
253             THEN                           \
254           THEN ;                           \ fertig
255
256 : cliplinie ( -- )                  \ zeichnet eine Verlängerung der Höhenlinie
257   schwarz                            \ Zeichenfarbe ist schwarz
258   steigung_mi 0dabei?                \ ist die Höhenline senkrecht oder waagerecht?
259   IF waagerecht?                     \ ja --> ist sie waagerecht?
260     IF   0 pmitte xy@ nip           \ ja --> x-linker Rand; y vonPmitte
261          breite masst* over         \ x rechter Rand maßtabgerecht; y Pmitte
262     ELSE pmitte xy@ drop 0          \ nein --> x Pmitte; y oberer Rand;
263          over hoehe masst*          \ x-Pmitte, y unterer Rand maßstabgerecht
264     THEN                             \
265   ELSE clippoints                    \ wenn nicht: hole zwei Randpunkte
266   THEN                               \
267   skal_linie ;                       \ zeichne die Linie
268
269 \ ----------------------------------------------------------------------+
270 \ Für das weitere ist wichtig: liegt die Strecke Mittelpunkt-höchster Punkt |
271 \ rechts von der Strecke Mittelpunkt-Fusspunkt?                           |
272 \ - Liegen beide (vom Mittelpunkt aus gesehen) im gleichen Quadranten?    |
273 \    Wenn ja, liefert der Vergleich der beiden Steigungen (Steilheit) die  |
274 \    Lösung.                                                               |
275 \    Wenn nein, gilt folgender Zusammenhang:                              |
276 \    Die Gerade M-H(höchster Punkt) liegt rechts von der Geraden M-PI (ISO- |
277 \    punkt), wenn der Quadrant der Geraden M-H dem Quadranten der Geraden  |
278 \    M-PI voreilt.                                                        |
279 \ - Wegen des 'gekippten' (Bildschirm-)Koordinatensystems (y-Werte wachsen |
280 \    nach unten) verteilen sich die Vorzeichen der Quadranten anders, als  |
281 \    von kartesischen Koordinatensystem her gewohnt.                      |
282 \    Q1--> +-  Q2--> ++  Q3--> -+   Q4--> --                              |
283 \ ----------------------------------------------------------------------+
284
285 Create voreil_tab      \ Tabelle: zu jedem Quadranten den voreilenden Quad.
286   -1 ,  1 , 1 ,  1 ,  \ Vorzeichen geswapt passend zum zukünftigen
287   -1 , -1 , 1 , -1 ,  \ Vergleich
288
289 : fusspunkt ( -- x y )           \ Fußpunkt der Fließline=Mittelpunkt Höhenl.
290   pi 0 pmitte 3@                   \ x-y PI und Dummy-Z; x-y-z von Pmitte
291   calc_mittelpunkt drop ;          \ Mittelpunkt berechnen; Z-Wert wegwerfen
```

```
292
293 : steigung_mh                    \ Steigungsdupel  mitte-höchster Punkt
294   pmitte xy@                     \ x-y von Pmitte
295   hoechster xy@                  \ x-y des höchsten Punktes
296   rot swap - -rot swap - ;       \ x-y des resultierenden Mittelpunktes
297
298 : +-? ( n -- +1/-1)              \ isoliert das Vorzeichen von N
299   dup 0= IF    drop 1            \ Null soll positiv sein
300          ELSE dup abs /          \ alle anderen Zahlen werden durch
301          THEN ;                  \ ihren Absolutwert geteilt
302
303 : quadrant ( n n - n n )         \ in Richtung welchen Quadranten?
304   +-? swap +-? swap ;            \ Vorzeichen Delta-x Delta-Y isolieren
305
306 : quadrant_gleich?  ( -- flag ) \ zeigen beide Steigungen in die gleiche Richtung?
307   steigung_mi quadrant           \ Vorzeichen des Steigungsdupels
308   steigung_mh quadrant           \ Vorzeichen des Steigungsdupels
309   d= ;                           \ vergleichen
310
311 : voreilend? ( 1n-n 2n-n -- f)  \ eilt Quadrant1 dem Quadranten2 vor?
312   2* + 3 + 2* voreil_tab +       \ Eintragadresse in der Voreiltabelle
313   2@ d= ;                        \ Dupel holen, mit Quadrantendupel vergleichen
314
315 : rechts? ( -- flag )            \ liegt der höchste Punkt rechts der Höhenlinie?
316   quadrant_gleich?               \ liegen beide Vektoren im gleichen Quadranten?
317   IF    100 steigung_mi s*/      \ ja --> Steigung mi skaliert errechnen
318         100 steigung_mh s*/      \        Steigung mh skaliert errechnen
319         >                        \          auf Größe vergleichen
320   ELSE steigung_mh quadrant      \ nein --> Quadrant zur Steigung_mh ermitteln
321         steigung_mi quadrant     \   dito für steigung_mi
322         voreilend?               \   liegt höchster Punkt links der Höhenlinie?
323   THEN ;                         \   fertig
324
325 : fliessrichtung ( -- )          \ zeichnet die Fliessrichtung ein
326   fusspunkt 2dup                 \ Fußpunkt errechnen, Kopie zum TOS
327   steigung_mi 2/ swap 2/         \ Steigung mit fünf skalieren
328   rechts?                        \ liegt der höchste Punkt rechts?
329   IF    -rot - -rot -            \ ja --> y-fuß - x-Steig.; x-Fuß - y-Steig.
330   ELSE -rot + -rot +             \ nein --> y-fuß + x-Steig.; x-fuß + y-Steig.
331   THEN swap                      \ Stack ordnen
332   dblau skal_linie ;             \ Zeichenfarbe Dunkelblau; zeichnen
333
334 \ ------------------------------------------------------------------------+
335 \ Jetzt ist alles fertig: aber, um den Code zu überprüfen, folgen einige    |
336 \ Testroutinen.                                                            |
337 \ Verschiedene Dreiecke werden gezeichnet, und die Höhen der Eckpunkte      |
338 \ kontinuierlich verändert.                                                |
339 \ Die Benuzung von gr_emit (VGA_EMIT.SEQ) gestattet es, auf einfache Weise |
340 \ interessierende Werte auf dem Grafikbildschirm auszugeben.               |
341 \ Wer nicht über diese Möglichkeit verfügt, muß die Worte .HOEHEN und      |
342 \ INTERNES entsprechend auskommentieren.                                   |
343 \ ------------------------------------------------------------------------+
344
```



```
345 Hier     Screenshot: Taumel in Aktion
346
347   0 Constant Drehwinkel
348 10 Constant Schrittweite          \ Schreittweite hier 0 Grad 10 Minuten
349
350 : drehwinkel+ ( n -- )           \ erhöht den Drehwinkel um Schrittweite
351   Schrittweite * Drehwinkel +    \
352   360 60 * mod =: Drehwinkel ;   \ interne Werte in Minuten
353
354 : .hoehen                        \ gibt die Höhen der Punkte aus
355   pa 3@ -rot skal_xy 10 + xy ." A:" .   \ jeweils Cursor auf x-y des Punktes
356   pb 3@ -rot skal_xy 10 + xy ." B:" .   \ auf Bildschirm umrechnen; um 10 Pixel
357   pc 3@ -rot skal_xy 10 - xy ." C:" . ; \ verschieben; Z-Wert ausgeben
358
359 : gefaelle? ( -- flag )          \ vergleicht die Höhen miteinader
360   pa z@ pb z@ = pb z@ pc z@ = and ;
361
362 : Winkel>hoehe                   \ Drehwinkel in Höhen umrechnen
363   Drehwinkel  60 /mod swap 60 *  \ Drehwinkel in Grad Minuten
364   2dup              sinus pa z!  \ Höhe bei Drehwinkel + 0 Grad
365   2dup 120 0 d+     sinus pb z!  \ ... plus 120 Grad
366        240 0 d+     sinus pc z! ; \ ... plus 240 Grad
367
368 : zeige ( adr -- )              \ zeichent einen Kreis an x y
369   xy@ skal_xy 4 kreis ;
370
371 : Bezugspunkte ( -- )           \ hebt wichtige Punkte hervor
372         drot                     \ Zeichenfarbe ist dunkelrot
373         pmitte       zeige       \ Mittelpunkt der höchsten Seite
374         fusspunkt skal_xy 4 kreis \ Fußpunkt der Fliesslinie
375         tiefster     zeige ;     \ tiefster Eckpunkt
376
377 \ INTERNES half bei der Entwicklung. Es gibt die Steigungungen der
378 \ Verbindungslinien Fußpunkt-Mittelpunkt und Höhenpunkt-Mittelpunkt
379 \ aus, ob beide Geraden im selben Quadranten liegen, in welcher
380 \ Richtung die Fliessrichtung liegt, und last not least die momentane
381 \ Stacktiefe. Voraussetzung ist vga_emit.
382
383 : internes ( -- )
384   10 2 at  ." Steigung: M-F "
385   100 steigung_mi 2dup swap 6 .r ." :" 6 .r  s*/  6 .r
386   10 3 at  ." Steigung: M-I "
387   100 steigung_mh 2dup swap 6 .r ." :" 6 .r  s*/  6 .r
388   10 4 at  ." Quadrant: "
389   quadrant_gleich? IF ." gleich " ELSE ." ungleich " THEN
390   10 5 at  ." Richtung: "
391   rechts? IF ." rechts " ELSE ." links " THEN
392   50 2 at ." CP1: "
393   clippoints 6 .r ." /" 6 .r
394   50 3 at ." CP2: "  6 .r ." /" 6 .r
395   10 6 at ." Stack: " depth . ;
396
397 : taumel ( -- )                 \ holt Veränderungen bei Drehung
398   Winkel>hoehe                   \ neue Höhen berechnen
399   gefaelle?                      \ liegt das Dreieck eben?
400   IF beep                        \ Melden
401   ELSE hoehenlinie               \ neue Höhenlinie
402         cliplinie                \ neue cliplinie
403         fliessrichtung           \ neue fliessrichtung
404         Bezugspunkte             \ neue Bezugspunkte
405         .hoehen                  \ neue Höhenwerte ausgeben
406   THEN ;
407
408 : taumeltest ( -- )              \ Taumelt das Dreieck nach rechts oder links
409   Begin                          \ Schleifenbeginn
410   dark                           \ Bildschirm neu zeichnen
411   schwarz rahmen                 \ Rahmen schwarz zeichnen
412   weiss dreieck                  \ Dreick weiss zeichen
413   taumel                         \ verändertes Zeichnen
414   internes                       \ Internes ausgeben
415   key                            \ auf Tastendruck warten
416   dup  4 = IF  1 ELSE -1 THEN    \ welche Taste wurde gedrückt
417   drehwinkel+                    \ Drehwinkel entsprechend ändern
418   27 =                           \ ESC gedrückt?
419   UNTIL ;                        \ ja --> Schleifenende;
420
421 : dauertest ( -- )               \ taumelt das Dreieck 'ewig' nach rechts
422   BEGIN                          \ Schleifenstart
423   dark                           \ Bildschirm neu zeichnen
424   schwarz rahmen                 \ Rahmen schwarz zeichnen
425   weiss dreieck                  \ Dreieck weiss zeichnen
426   taumel                         \ Veränderte Linien zeichen
427   internes                       \ interne Werte ausgeben
428   ?keypause                      \ ist ein Pause, ein Abbruch gewünsacht?
429   drehwinkel+                    \ Drehwinkel um 1 sek. erhöhen
430   again ;                        \ unbedingt wieder von vorne
431
432 : Fall1                          \ Testdreieck Nummer 1
433   10   320 -5299 Pa 3!
434   350 350   9994 Pb 3!
435   300   10 -4695 Pc 3!
436    0     0 0 pmitte 3! ;
437
438 : Fall2                          \ Testdreieck Nummer 2
439   0    320 10 Pa 3!
440   350 350 20 Pb 3!
441   550  90 30 Pc 3!
442    0     0 0 pmitte 3! ;
443
444 Fall1
445
446 grtext
447 \ grafik
448
449
450
```

LängsteZeilehat 82 Zeichen

XMODEM

von Rafael Deliano
Steinbergstr. 37; D-82110 Germering

Einplatinencomputer werden während der Entwicklung normalerweise über V24 von PCs aus gesteuert. Im PC wird meist ein handelsübliches Terminalprogramm verwendet das entweder TTY oder VT52 emuliert. Damit können nur ASCII-Zeichen übertragen werden, Datenpakete werden nicht unterstützt und es findet keine Fehlersicherung statt. Will man Binärdaten als Paket übertragen, muß man ein Filetransferprogramm verwenden. Das älteste und verbreitetste ist XMODEM. Es wird von den meisten PC-Terminalprogrammen unterstützt. Zudem ist es einfach zu implementieren und braucht im Einplatinencomputer wenig Programm- und Datenspeicher.

Stichworte: XMODEM Datenübertragung nanoFORTH

Es war einmal

Der Vorläufer von XMODEM hieß schlicht MODEM und stammt von Ward Cristensen. Keith Petersen verbesserte das Programm und taufte es XMODEM. 1977 wurde es in die Public Domain übergeben und verbreitete sich rasch. Es war eigentlich nur dazu gedacht, die Datenübertragung zwischen CP/M-Rechnern mit 300 Baud Modems zu ermöglichen. Daraus ergibt sich auch die Länge des XMODEM-Datenfelds von 128 Byte. Sie entspricht direkt dem Datenblock auf einer CP/M-Diskette. Im praktischen Betrieb ergaben sich mit XMODEM bald Probleme. Sobald die Modems schneller wurden, begann die kurze Länge des Datenfelds den Durchsatz zu begrenzen. Die schwache Fehlersicherung durch eine Prüfsumme erlaubte auf Telefonleitungen keinen sicheren Betrieb, da schnellere Modems mehr Bitfehler produzieren.

Bald entstanden deshalb Varianten von XMODEM. In XMODEM-1K wurde die Länge des Datenfelds auf 1024 Bytes erhöht. In XMODEM/CRC wird das Prüfsummenbyte durch eine 16 Bit CRC ersetzt. Chuck Forsberg faßte die Vorteile dieser und anderer XMODEM-Erweiterungen in das neue Protokoll YMODEM zusammen. Leider löste das die Probleme nicht, da von YMODEM viele inkompatible Implementierungen entstanden. Wirklich befriedigend war erst Forsbergs ZMODEM. Für Datenübertragung zwischen PCs über Modem ist XMODEM heute überholt und ZMODEM besser geeignet. Für Anwendung auf Einplatinencomputern ist ZMODEM jedoch zu aufwendig. Speziell für die Übertragung von Binärfiles genügt hier XMODEM.

Protokoll

Obwohl nur in eine Richtung Daten übertragen werden, wird eine Vollduplexverbindung benötigt, damit im Rückkanal Handshakesignale empfangen werden können. Diese Quittungssignale sind ASCII-Steuerzeichen die in Tabelle 1 aufgeführt sind. Alle Zeichen werden mit 8 Bit übertragen. Der Empfänger startet die Datenübertragung indem er ein NAK-Byte sendet (Bild 1). Er zeigt damit an, daß er bereit ist, und wiederholt das NAK bis zu 10 mal. Als Pause zwischen diesen Wiederholungen ist in XMODEM

Tabelle 1: Verwendete Steuerzeichen mit Code in Hex

ACK	“Acknowledge”	06h
NAK	“Negative Acknowledge”	15h
SOH	“Start of Header”	01h
STX	“Start of Text”	02h
EOT	“End of Transmisson”	04h
CAN	“Cancel”	18h
^Z		1Ah
C		43h

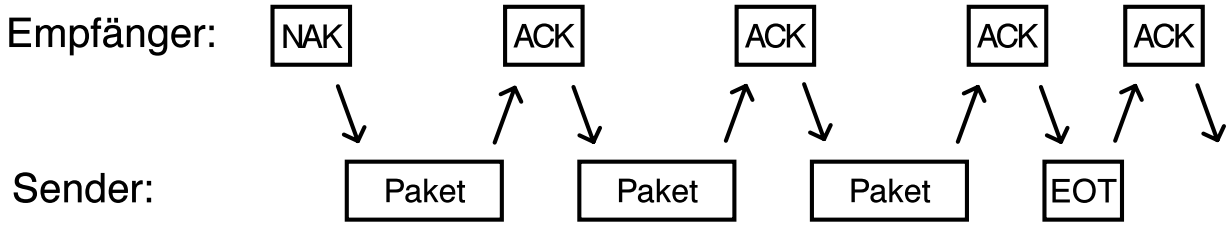


Bild1: Handshake

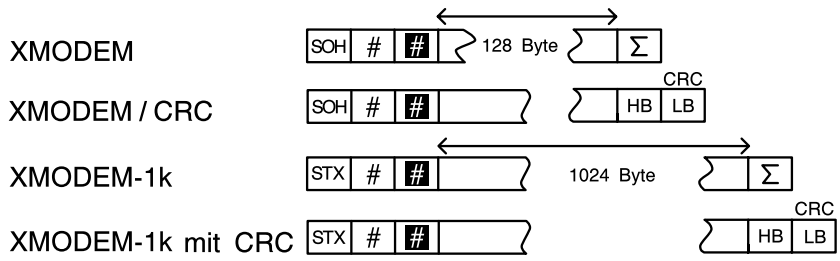


Bild 2: Pakete

ursprünglich 10sec vorgesehen. Bei hohen Baudraten ist jedoch 3sec sinnvoller. Hat der Sender das NAK erkannt, antwortet er mit dem ersten Datenpaket.

Der Empfänger quittiert jedes erfolgreich empfangene Paket mit einem ACK-Byte. Bei einem Fehler sendet er ein NAK. Danach muß der Sender dieses Paket nocheinmal wiederholen. NAKs die nicht beantwortet werden, werden maximal 10 mal wiederholt. Dann bricht der Empfänger die Verbindung ab. Wenn alle Pakete erfolgreich übertragen wurden, schickt der Sender nach dem letzten Datenpaket ein EOT-Byte, das vom Empfänger mit ACK quittiert werden muß. Sonst wiederholt der Sender dieses EOT-Zeichen bis zu 10 mal.

Im ursprünglichen XMODEM nicht vorhanden, heute aber meist unterstützt, ist das CAN-Byte mit dem sowohl Sender als auch Empfänger einen Abbruch der Verbindung anzeigen können. Will der Sender abbrechen, sollte er das Steuerzeichen nach der Übertragung eines vollständigen Datenpakets senden. Innerhalb eines Pakets kann es der Empfänger nicht von den binären Daten unterscheiden, wird es nicht erkennen und erst nach Timeout die Verbindung abbauen.

Paket

Das Datenfeld hat im ursprünglichen XMODEM eine feste Länge von 128 Byte (Bild 2). Als Startzeichen für ein derartiges kurzes Paket dient ein SOH-Byte. In XMODEM-1k hat das Datenfeld die feste Länge von 1024 Byte. Lange Pakete werden durch das STX-Startbyte gekennzeichnet. Jedes Paket hat eine Nummer, die im ersten Byte der Paketnummer übertragen wird. Das erste Paket hat die Nummer 01. Diese Nummer wird mit jedem weiteren Paket inkrementiert. Unabhängig davon ob damit 128 oder 1024 Byte Daten übertragen werden. Nachdem FF erreicht ist, erfolgt Überlauf auf 00. Erst im nächsten Paket wird auf 01 weitergezählt. Das zweite Byte entspricht dem ersten Byte, ist aber bitweise invertiert. Damit hat die Paketnummer ihre eigene Fehlersicherung. Der Empfänger akzeptiert Pakete mit kontinuierlich steigender Paketnummer wie sie bei fehlerfreier Übertragung auftreten. Er akzeptiert auch Pakete die nacheinander identische Paketnummern haben, wie das bei Wiederholung einer Übertragung der Fall ist. Trifft keiner dieser beiden Fälle auf die Folgenummer zu, haben Sender und Empfänger ihre Synchronisation verloren. Dieser Fehler

File: xmod.lst vom 7.5.1996

```
 1 <|  \ XMODEM.F74
 2 HEX
 3 06 CONSTANT 'ACK'   15 CONSTANT 'NAK'   01 CONSTANT 'SOH'
 4 02 CONSTANT 'STX'   04 CONSTANT 'EOT'   18 CONSTANT 'CAN'
 5 1A CONSTANT '^Z'    43 CONSTANT 'C'
 6 2 ZVARIABLE LIMIT          \ Addr beyond Buffer
 7 2 ZVARIABLE >BUFFER        \ Pointer in Buffer
 8 2 ZVARIABLE >BUFFER-1      \ Start of last Packet
 9 2 ZVARIABLE SMALL/BIG      \ Flag (TX) or Count of Bytes (RX)
10 1 ZVARIABLE SUM/CRC        \ Flag: Checksum or CRC
11 1 ZVARIABLE PACKET#        \ Packet-Number
12 \ ----------------------------------------------------------
13 \                          Calculate Checksum   MSB first
14 1021 CONSTANT POLY            \ CRC-CCITT
15 7E00 CONSTANT CRC-TABLE    \ Table for 512 Bytes
16 : CRC+                     \ ( CRC Data --- CRC' ) Version 2
17 8<SHIFT XOR 7 0 DO   \ Loop 8 x
18 DUP 8000 AND IF 1<SHIFT POLY XOR ELSE 1<SHIFT THEN
19 LOOP ;
20
21 : FILL-TABLE            \ (   ---  )
22 FF 0 DO                  \ 256x 00 ... FF
23 I 8<SHIFT 00 CRC+ I 1<SHIFT CRC-TABLE + ! LOOP ;
24
25 FILL-TABLE
26
27 : CRC+        \ ( CRC Data   ---    CRC ) Version 3
28 OVER 8SHIFT> XOR 1<SHIFT CRC-TABLE + @ SWAP 8<SHIFT XOR ;
29
30 : FCS+                \ ( FCS Data Flag1  --- FCS' ) Update FCS
31 \ Use either Sum or CRC ; Flag1 is 0 in the Headerbytes
32 SUM/CRC C@ IF DROP + ELSE IF CRC+ ELSE DROP THEN THEN ;
33 \ ----------------------------------------------------------
34 :CODE 1SEC-KEY?            \ (    --- Data 1 ) data received
35                            \ (    ---        0 ) no data for 1 sec
36 ... CODE; \ coded in assembler, device-dependant
37 \ ----------------------------------------------------------
38 \ Transmit XMODEM
39 : NAK/C?         \ (   --- Flag ) 0 = disconnect 1 = ok
40 0 3C 0 DO       \ Wait up to 60 sec for handshake
41 1SEC-KEY? IF DUP 'NAK' = IF        SUM/CRC  C! 1+ LEAVE ELSE
42                  DUP 'C'    = IF LNOT SUM/CRC  C! 1+ LEAVE ELSE
43                      'CAN'  = IF                     LEAVE THEN
44             THEN THEN THEN
45 LOOP ;
46
47 : ACK/NAK?   \ (   --- Token ) 0 = disconnect 1 = NAK 2 = ACK
48 0 3C 0 DO    \ Wait up to 60 sec for handshake
49 1SEC-KEY? IF DUP 'ACK' = IF DROP 2+ LEAVE ELSE
50                  DUP 'NAK' = IF DROP 1+ LEAVE ELSE
51                      'CAN' = IF          LEAVE THEN
52             THEN THEN THEN
53 LOOP ;
54
55 : TX-PACKET                          \ ( --- )
56                                     \ Send SOH / STX
57 SMALL/BIG @    LIMIT >BUFFER - 400 U> LNOT OR \ ( --  Flag )
58 IF 80 'SOH' ELSE 400 'STX' THEN          \ ( --- 400/80 )
59 DUP EMIT 0000 SWAP 0 FCS+                \ ( --- 400/80 FCS )
60                                     \ Send PACKET-Nummer
61 PACKET# C@ SWAP OVER 0 FCS+ OVER EMIT \ ( -- 400/80 # FCS )
62 SWAP 00FF XOR DUP EMIT 0 FCS+           \ ( --- 400/80 FCS )
63                                     \ Send Data-Bytes
64 SWAP 1 DO                                  \ (   --- FCS )
65 >BUFFER @ DUP LIMIT @ =
66 IF DROP '^Z' ELSE C@ 1 >BUFFER +! THEN   \ ( --- FCS Data )
67 DUP EMIT 1 FCS+
68 LOOP                                       \ (   --- FCS )
69                                     \ Send Checksum
70 SUM/CRC C@ IF EMIT ELSE DUP 8SHIFT> EMIT EMIT THEN ;
71
72 : (TX)                          \ ( Addr # S/B-Flag   --- Flag )
73 SMALL/BIG !     01 PACKET# C!
```

```
 74 OVER DUP >BUFFER ! >BUFFER-1 ! + LIMIT !
 75 NAK/C? IF
 76 BEGIN
 77    9 0 DO                                \ Try up to 10 NAKs
 78    TX-PACKET  ACK/NAK?    \ (   --- Token ) Token: 0 , 1 , 2
 79    DUP IF 1- IF >BUFFER @ >BUFFER-1 !             \ ACK
 80                  1 PACKET# +C! 1 LEAVE
 81               ELSE >BUFFER-1 @ >BUFFER ! THEN   \ NAK
 82          ELSE LEAVE THEN                          \ disconnect
 83    LOOP              \ (   --- Flag ) 0 = disconnect  1 = ok
 84    IF >BUFFER @ LIMIT @ = IF 1 1              \ all done
 85                             ELSE 0 THEN       \ next packet
 86    ELSE 0 1 THEN                              \ disconnect
 87 UNTIL ELSE 0 THEN
 88 IF 'EOT' EMIT 0 9 0 DO ACK/NAK?
 89      DUP IF 1- IF    1 OR LEAVE               \ ACK
 90               ELSE 'EOT' EMIT THEN            \ NAK
 91          ELSE DROP LEAVE THEN                   \ disconnect
 92      LOOP
 93 ELSE 0 'CAN' EMIT THEN ;
 94
 95 \ Addr = Start of Data , # = Number of Bytes
 96              \ Send only 128 Byte Packets
 97 : TX         \ ( Addr #   --- 1 ) Transmisson ok
 98 1 (TX) ;    \              --- 0 ) failed
 99              \ Send 1024 & 128 Byte Packets
100 : TX-1k      \ ( Addr # --- 1 ) Transmisson ok
101 0 (TX) ;    \              --- 0 ) failed
102 \ ------------------------------------------------------
103 \                                                  Receive  XMODEM
104 \ N/A-Flag:  0000 = ok              -> ACK
105 \              0001 = Timeout       -> NAK
106 \              0010 = Data-Error    -> WAIT -> NAK
107 \              0100 = Synch-Error   -> disconnect
108 \              11xx = 100sec-Timeout  -> disconnect
109
110 : TX-HS-10X      \ ( 'HS'  --- Data N/A-Flag )
111                   \ Send 'HS' every 10 seconds up 10 times
112                   \ 'HS' = 'NAK'  or 'C' or, if 00, idle
113 FFFF 64 0 DO    \ (  ---  'HS' # )
114    1+ DUP LNOT                          \ output 'HS' every 10 sec
115    IF DROP DUP IF DUP EMIT THEN FFF6 THEN
116    1SEC-KEY? IF 0 2SWAP 2DROP LEAVE THEN   \ input Data
117 LOOP ;
118
119 : (WAIT)              \ (   ---   ) wait till line clear for 1sec
120 BEGIN   1SEC-KEY? IF DROP  0 ELSE 1 THEN UNTIL ;
121
122 : (PACKET#)                          \ ( FCS   --- FCS'  N/A-Flag )
123 1SEC-KEY?
124 IF SWAP  OVER 0 FCS+                \ ( --- Data1 FCS' )
125    1SEC-KEY?
126    IF SWAP  OVER 0 FCS+ SWAP ROT \ ( --- FCS'  Data2 Data1 )
127       OVER FF XOR =
128       IF FF XOR PACKET# C@ 2DUP =
129          IF 2DROP >BUFFER @ >BUFFER-1 ! 0    \ Expected Packet
130          ELSE 1- =
131             IF                          \ ( --- FCS ) Retransmisson
132             FFFF PACKET# +C!          \ decrement PACKET#
133             >BUFFER-1 @ >BUFFER ! 0
134             ELSE 4 THEN                      \ Synch  lost
135          THEN
136       ELSE DROP 2 THEN                \ Error: Data2 ><_>Data1
137    ELSE DROP 1 THEN
138 ELSE 1 THEN ;
139
140 : (DATA)                    \ ( FCS  --- FCS'  N/A-Flag )
141 0 SWAP SMALL/BIG @ 1 DO   \ (   --- 0 FCS )
142 1SEC-KEY?
143 IF SWAP  OVER 1 FCS+ SWAP \ (   --- 0 FCS  Data )
144    >BUFFER @ LIMIT @ =    \ overrun ?
145    IF    DROP                 \ ignore new data
146    ELSE >BUFFER @ C! 1 >BUFFER +! THEN \ store data
147 ELSE 2DROP 1 0 LEAVE THEN    \ timeout -> NAK
148 LOOP SWAP ;
149
150 : (FCS)                    \ ( FCS   --- N/A-Flag )
151 1SEC-KEY?
152 IF SUM/CRC  C@
153    IF SWAP  FF AND = IF 1 PACKET# +C! 0 ELSE 2 THEN   \ SUM
154    ELSE 8<SHIFT 1SEC-KEY?
155       IF                          \ ( --- FCS  Data1 Data2 ) CRC
156       OR = IF 1 PACKET# +C! 0 ELSE 2 THEN
157       ELSE 2DROP DROP 1 THEN
158    THEN
159 ELSE DROP 1 THEN ;
160
161 : (PACKET)              \ ( Data  SMALL/BIG  --- N/A-Flag )
162 SMALL/BIG ! 0000 SWAP 0 FCS+    \ (   --- FCS              )
163                       (PACKET#)       \ (   --- FCS  N/A-Flag )
164 DUP LNOT IF DROP (DATA) THEN    \ (   --- FCS  N/A-Flag )
165 DUP LNOT IF DROP (FCS)   THEN ; \ (   ---       N/A-Flag )
166
167 : RX-PACKET      \ ( Data   ---   0 ) send ACK
168                  \              ---   1 ) send NAK
169 DUP 'SOH' = IF   80 (PACKET) ELSE    \  128 Byte
170 DUP 'STX' = IF 400 (PACKET)          \ 1024 Byte
171               ELSE DROP 2 THEN THEN   \ garbage: NAK
172 DUP 02 = IF (WAIT) THEN ;
173
174 : (RX)               \ ( Addr # 'HS'  SUM/CRC-Flag  --- Flag )
175 SUM/CRC  C!    01 PACKET#  C!
176 SWAP ROT DUP >BUFFER ! DUP >BUFFER-1 ! + LIMIT !
177 TX-HS-10X
178 IF DROP  0 ELSE \ Send 'NAK' or 'C' every 10sec
179 BEGIN    RX-PACKET  \ ( Data  --- N/A-Flag )
180 DUP 04 =
181 IF 0 SWAP
182 ELSE IF 'NAK' ELSE 'ACK' EMIT 0 THEN
183    TX-HS-10X            \ Send either idle or 'NAK' every 10sec
184    IF DROP                                  0 1  \ (  ---   0    1 )
185    ELSE DUP 'EOT' = IF DROP 'ACK' EMIT 1 1  \ (  ---   1    1 )
186                        ELSE 0              THEN    \ (  --- Data 0 )
187    THEN
188 THEN
189 UNTIL IF 1 ELSE 'CAN' EMIT 0 THEN
190 THEN ;
191
192 \ Addr = Start of Receivebuffer , # is Size of Buffer
193                   \ XMODEM with checksum:
194 : RX              \ ( Addr #   --- 1 ) File received
195 'NAK' 1 (RX) ; \              --- 0 ) Transmission  failed
196                   \ XMODEM with CRC:
197 : RX-CRC          \ ( Addr #   --- 1 ) File received
198 'C'    0 (RX) ; \ (            --- 0 ) Transmission  failed
199
200 |>
LängsteZeilehat 60 Zeichen
```

Zur Anpassung ...

... dieses nanoFORTH-Listings an Forth83 und andere:

- Die Zahl vor ZVARIABLE gibt die Anzahl der Bytes an, die der Variable zugewiesen werden
- LOOP macht einen Durchlauf mehr als im Forth83: 3 0 DO ... LOOP läuft 0, 1, 2, 3 und bricht dann ab.
- Flags werden durch LAND , LOR , LNOT verknüpft. Forth83 benutzt AND , OR , NOT .

❑

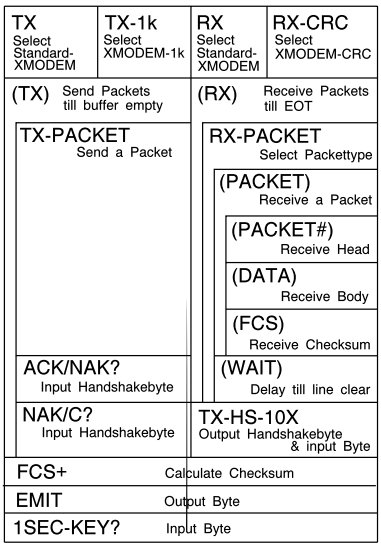


Bild 3: Source

kann nicht behoben werden. Der Empfänger bricht dann die Verbindung ab. Eventuell sendet er vorher noch ein CAN-Zeichen.

Das Datenfeld hat in XMODEM immer eine feste Länge. Die Daten werden direkt im Binärformat gesendet. Sind für das letzte Paket nicht mehr genau 128 Byte Daten vorhanden, wird der Rest des Datenfelds aufgefüllt. Verwendet wird dazu das Steuerzeichen ^Z, das bei CP/M das Ende eines Files anzeigt. Wenn im letzten Block genau 128 Byte Daten vorhanden sind, senden manche ältere, besonders CP/M-konforme Implementierungen deshalb einen weiteren Block, der nur ^Z-Zeichen als Daten enthält. Bei ASCII-Texten kann man diese Endezeichen problemlos im Empfänger eliminieren. Bei Binärfiles können jedoch Probleme auftreten, weil das genaue Ende des Datensatzes nach der Übertragung unklar ist. Im schlimmsten Fall verlängert sich das File um 128 Byte.

Fehlersicherung

Zur Bildung der Prüfsumme werden in XMODEM alle vorangegangen Bytes, also Kopf und Daten addiert. Die untersten 8 Bit der Summe werden als Prüfzeichen übertragen. Innerhalb des Datenblocks wartet der Empfänger 1sec auf jedes Byte. Wird dieser Timeout überschritten, sendet er ein NAK. Wird ein Fehler entdeckt, während ein Paket empfangen wird, wartet der Empfänger bis das Paket komplett gesendet ist. Z.B. kann er darauf warten, daß 1sec lang keine weiteren Daten empfangen wurden. Erst dann sendet er das NAK. Das bremst zwar die Übertragung, verhindert aber den Verlust der Synchronisation.

XMODEM/CRC

Diese Option wird vom Empfänger gewählt. Er sendet dazu als Eröffnung statt des NAK das ASCII-Byte C. Die Pakete werden jedoch dann wieder konventionell mit ACK und NAKs quittiert. Hat der Sender die CRC-Option jedoch nicht implementiert, kann er nur noch die Verbindung abbrechen. Die Prüfsumme wird hier nur über die Datenbytes, d.h. nicht über den Kopf gebildet. Die 16-Bit CRC verwendet das Generatorpolynom CRC-CCITT. Die Daten werden mit MSB zuerst verarbeitet [2]. Vom Prüfwort wird erst das obere, dann das untere Byte übertragen. Das Paket wird damit um ein Byte länger .

XMODEM-1k

Diese Option wird vom Sender gewählt. Hat der Empfänger sie nicht implementiert, kommt auch hier wieder keine Übertragung zustande. STX ersetzt hier SOH als Startzeichen des Pakets. Das Datenfeld eines langen Pakets enthält 1024 Bytes. Lange und kurze Pakete können jedoch gemischt werden. Ein File von 1,5k Byte Umfang zerlegt der Sender dementsprechend in ein langes und 4 kurze Pakete.

Viele Terminalprogramme akzeptieren XMODEM-1k nur mit CRC als Fehlersicherung. Im Konfigurationsmenü der Terminalprogramme wird XMODEM-1k oft nicht explizit aufgelistet. Trotzdem können sie mit der Einstellung XMODEM auch lange Pakete empfangen, da sie anhand der Startzeichen diese Option automatisch erkennen können.

Implementierung

Das in nanoFORTH geschriebene Programm im Listing ermöglicht Senden und Empfangen von XMODEM, XMODEM-CRC und XMODEM-1k. Es ist für die Verwendung auf einem Einplatinencomputer gedacht. Empfang erfolgt durch die Befehle RX, bzw. RX-CRC. Dem Befehl wird auf dem Stack die Adresse und der Umfang der zu übertragenden Daten übergeben. Er läßt auf dem Stack ein Flag zurück, das anzeigt, ob die Übertragung erfolgreich war. Senden erfolgt durch TX, bzw. TX-1k. Hier wird Adresse und Umfang des Empfangspuffers übergeben. Zugriff auf die UART erfolgt durch EMIT, das Bytes sendet und den Befehl 1SEC-KEY?, der den Empfangskanal 1sec lang pollt .

Das komplette Programm belegt etwa 3 kByte Programmspeicher zuzüglich 512 Byte für die CRC-Tabelle. Verzichtet man auf die CRC, oder benötigt man nur Senden oder nur Empfangen, kann es erheblich abmagern. Auf einem Mitsubishi-6502, der mit 2,5 MHz läuft, verarbeitet es bis zu 9600 baud.

Résumé

Die Liste von XMODEMs Pferdefüßen ist lang. Fehlende Feature-Negotiation: trifft eine erweiterte Version von XMODEM auf eine Grundversion, kommt keine Verbindung zustande. Denn die erweiterte Version kann sich nicht automatisch zurückschalten. Deshalb sollte man die beiden gängigen Erweiterungen implementieren, selbst wenn man sie nicht unbedingt benötigt. Für Kanäle die nur für 7 Bit transparent sind, müssen andere Protokolle, z.B. Kermit [3] verwendet werden. Bei Einplatinencomputern sollte das Problem jedoch selten auftreten. Flowcontrol durch XON/XOFF ist nicht möglich. Will man nicht auf Hardwarehandshake mit RTS/CTS ausweichen, muß die XMODEM-Software auf dem Einplatinencomputer also so schnell sein, daß sich Flowcontrol erübrigt.

Die unkontrollierte Verlängerung von Binärfiles stellt kein Problem dar, wenn man sich darauf beschränkt die Daten in fester Länge von 128-Bytes zu übertragen. Und gegebenenfalls den letzten Block untersucht, ob er nur aus ^Z-Steuerzeichen besteht.

Ward Christensen hat zu den praktischen Problemen mit XMODEM angemerkt, daß komplexe Protokolle existieren, die diese Macken nicht haben. XMODEM hat sich aber durchgesetzt, weil es einfach ist. Deshalb konnte es von vielen Leuten schnell und kompatibel implementiert werden.

Literatur:

[1] Chuck Forsberg “XMODEM/YMODEM Protocol Reference” 1988 (Als EMAIL in vielen Mailboxen vorhanden. Z.B. KBBF 0431/5339898)

[2] “Auf Fehlersuche” ELRAD 96/2

[3] Frank da Cruz "Kermit - A File Transfer Protocol". Digital Press 1987

Ein heiliger Krieg

das THEN im Forth

von M.Kalus
Malente, Plönerstr24a

Am 27. Juni dieses Jahres entbrannte ein heiliger Krieg in /comp/lang/forth. Einige Jahre lang herrschte Friede, ANS sei Dank. Es begann mit einer harmlosen Mitteilung von M.L. Gassanenko ...

Stichworte: THEN

Gassanenko schrieb: “Suddenly I have come to understanding that the Forth IF..ELSE.. THEN syntax is quite natural, and, probably, THEN is the best word to be in that place:

```
: FOO ( n -- )
  dup 0< 0= \ If n is positive or 0,
  if 200 + \ add 200,
  else -200 + \ else add -200.
  then . ; \ Then, print the result.
```

So, Forth THEN means ‘after that’. (...)
How can native speakers comment this?”

Daß er damit grundlegende sprachliche Problem des Programmierens aufgerührt hatte, wurde ganz schnell deutlich. Es strömten so an die 100KB Text in knapp 14 Tagen in c/l/f hinein.

Nun, jeder Mensch, der auch nur einmal das allerkeinste Computerprogramm in Form von Bitfolgen in Pappkarten knipsen mußte, kennt das elementare Bedürfnis nach höheren Programmiersprachen. Dieses Streben nach Kommandos, die jede Menge Code ohne unser Zutun compilieren können, ist so etwas wie ein Trieb, ist wie Agression und Libido.

Mit den heutigen Computersprachen, den modernen Oberflächen und Betriebssystemen sind wir in dieser Sache auch schon gut voran gekommen. Wir können in einer gigabytegroßen Umgebung mit megaschnellen Maschine n“Hallo World” in multitaskingfähige Ausgabefenster flimmerfreier Farbmonitore in vielhundert Farbabstufungen und zahlosen Fonts schreiben ohne die dazu nötigen 22 Megabytes Code von Hand eingeben zu müssen. Diese Programmierumgebung ist mächtig wie noch nie. Die Visionen fliegen hoch - ganz weg von Programmlistings. VisualBasic! Aber lassen wir das, darum geht es ja jetzt nicht - VisualForth ist auch noch nicht in Sicht.

Nur einer ganz kleinen Gruppe von Programmierern ist es wohl noch erlaubt ganz nahe an den Bits und Bytes zu sein. Obschon auch sie dann und wann mächtigere Routinen herzunehmen suchen, welche allerlei unbequeme Arbeit ersparen. Forthler sind solche Programmierer.

Aber sie sind schlimmer zu hüten als ein Sack Flöhe. Wir wissen das ja nun aus eigener Erfahrung. Wie schnell ist mit Forth alles auch ganz anders gemacht! Also Vorsicht. Das ist wie beim Turmbau zu Babel. Sprachverwirrung ist von Übel. Es war und ist daher wichtig, die Sprache Forth wenigstens in ihrem Kern stabil zu halten. Forth wurde bekanntlich inzwischen im ANS fixiert.

Und dabei wurde auf die besondere forthige Etikette geachtet, wie ‘gute mnemonische Namen’ - also sowas wie ‘natural language programming’, um lesbare Listings zu gewinnen und ‘selbsterklärender Code’ natürlich und überhaupt die ‘Eleganz des Programms’ schlechthin. Strukturiertes Programmieren gilt als selbstverständlich (multiples EXIT ist gradezu ein Sakrileg) dazu `pretty printing’ von Quellen, vorzugsweise Screens und so weiter. Ein ganzer langer alter Zopf an Historie war zu wahren. Oder an gediegener Erfahrung einer ganzen Generation von Programmierern. Das kann man so oder so sehen.

Leo Brodie hatte sich 1984 die Mühe gemacht, über diese Tugenden des Programmierens ein ganzes Buch zu schreiben, darin ein ganzes Kapitel nur darüber, wie man in Forth Kontrollstrukturen vermeiden oder minimieren kann. Um Mißverständnissen an dieser Stelle vorzubeugen: Ein vortreffliches Buch, auch heute noch sehr erhellend in diesen Dingen und bei der Gelegenheit als Lektüre zu unserem Thema hier sehr empfehlenswert.

Der Konflikt

Und nun sind wir wieder angekommen beim zugrunde liegenden Konflikt, den M.L. Gassanenko mit seiner einfachen Feststellung berührt hat. Was ist ‘gutes Forth’, ist ‘mnemonisch’ und ‘richtig’ bei der Namensgebung von Controllstrukturen? Was dem Einen treffend ausgedrückt zu sein scheint, muß einem anderen noch lange nicht gefallen! Und hier kommt etwas technikfremdes ins Spiel: Gefühl. Sprachgefühl in diesem Falle.

In der Praxis haben Routinen schnell einen Namen ‘weg’. Und auch wenn der Name später als mißlungen eingestuft werden muß, das Sprachgefühl damals also dürftig gewesen war, wissen die Beteiligten oft auch so noch recht gut was gemeint war. Aber irgendwann stolpern die Schüler doch über diese unglücklichen Benennungen und fordern Nachbesse-

rung. Und die Diskussion entzündet sich dann gerne an Forthworten, die auch in der gesprochenen natürlichen Sprache vorkommen, also das Sprachgefühl herausfordern.

Warum sind diese Elemente denn dann überhaupt vorhanden? Nun, die Hoffnungen der Spracherfinder gingen wohl dahin, durch natürlich klingende Anweisungen das Coding gewissermaßen wie von selbst geschehen zu lassen. Wenn ein Computer schon nicht nachdenken kann, soll er doch wenigstens dem Programmierer das Nachdenken erleichtern, ihn vor Unfällen schützen und warnen. Und da ist es bequemer, wenn der Computer die Menschensprache spricht statt umgekehrt. Und so werden diese Kontrollanweisungen gerne mit “natürlichen” Worten belegt. Ich glaube es war Wil Baden der diese Elemente der Computersprachen einmal “syntactical sugar” genannt hat.

IF... ELSE... THEN ist so ein Zuckerbonbon. Es vermeidet das lästige Nachdenken über die Sprungziele bei Verzweigungen - strukturiertes Programmieren eben. Damit produzieren dann auch unordentliche Programmierer Code, den man noch sehr viel später verstehen und sogar warten kann und der dann immer noch funktioniert. Denn sonst hätte man ja auch bei GOTO bleiben können, oder?

Der Kampf

Was ist also nun mit diesem Konstrukt? Ist es ok oder nicht? Einige andere Sprachen kommen ohne ein THEN aus. Doch wenn THEN vorkommt, wird es in folgender Form gebraucht:

```
IF <test> THEN <true clause>
```

Das kommt einem einfachen englischen Satz schon sehr nahe. Doch befolgt man nun konsequent UPN findet man sogleich Gründe das ganze umzudrehen und bekommt das, was wir von FORTH kennen:

```
<test> IF <true clause> THEN
```

Und das klingt dann zunächst gar nicht mehr wie Englisch. Und daher wurde THEN eine Zeitlang verbannt, figForth nannte es ENDIF. Damit wurde immerhin klar, was an der Stelle geschieht. Der bedingte Sprung des IF wird hierhin gelenkt. Doch der FORTH-79 Standart setzte THEN durch, so wie Chuck Moore es in den späten ‘60igern oder frühen ‘70igern nun mal erfunden hatte. Und damit wurde immerhin viel existierender kommerzieller Code gerettet.

Brodie (1) schrieb später die Kunst der Namensgebung nieder und befand darin “Ein Name soll ausdrücken ‘was’ und nicht ‘wie’ es geschieht.” Und “geben sie den Worten Namen, die im Zusammenhang einen Sinn ergeben”. Wendet man diese Regel auf IF...ELSE... THEN an, findet man erstaunliches.

- IF leitet zwar eine Entscheidung ein, drückt aber nicht richtig aus ‘was’ nun tatsächlich geschieht, doch es versteckt das ‘wie’.
- ELSE kündigt die Ausführung der anderen Möglichkeit an, drückt also schon besser aus ‘was’ an dieser Stelle geschieht und versteckt auch das ‘wie’.
- THEN ist im Englischen mehrdeutig und ohne Kontext gar nicht zu verstehen. Entfernt es sich zu sehr von IF oder ELSE, geht sein möglicher Sinn vollends verloren. ‘Was’ hier geschieht ist nicht mehr zu erkennen, das ‘Wie’ allerdings auch nicht.

So sieht es also danach aus als hätte Chuck Moore solche ehrenwerten Regeln nicht beachtet, sondern IF...THEN nur in Anlehnung an andere Computersprachen gewählt und dann einfach UPN draus gemacht. Doch versteht man THEN in dieser Weise als “after that”, entsteht unweigerlich das Bedürfnis die Syntax auch so haben zu wollen.

```
<if_test_is_true> THEN <do_this> ELSE <do_that>
```

Nun stünde THEN an der Stelle von IF und um das Chaos nicht perfekt zu machen, muß es nun natürlich einen anderen Namen bekommen. Und so lautete z.B. der Vorschlag von Bruce R. McFarling dazu:

```
IF{ <test-state> ?THEN <conditional-act>
ELSE <test-state> ?THEN <conditional-act>
ELSE <test-state> ?THEN <conditional-act>
    ...
}ENDIF
```

Bei dieser Regelung des Problems der Vorwärtsreferenz benötigt er aber zusäzlich Marker für die Einleitung und Auflösung der Referenzen auf dem Stack. Und natürlich hagelte es Schelte von allen Seiten, nicht zuletzt von Elisabeth Rather, der Hüterin des Standard im Forth.

Andere Vorschläge sahen dagegen die Lösung des Problems in einer einfache Redefinition. Dabei wurden allerdings auch wieder Brodies ‘Was’-Regel verletzt. Im folgenden Beispiel wird AFTERWARDS dem Sinn nach endgültig zum bloßen Marker und damit überflüssig im Forth. Überflüssig zu sagen, daß auch dieses Beispiel kräftig Schelte bekam. Es sei hier stellvertretend für weitere Redefinitionen aufgeführt.

```
: IF-SO-THEN POSTPONE IF ; IMMEDIATE
: OTHERWISE POSTPONE ELSE ; IMMEDIATE
: AFTERWARDS POSTPONE THEN ; IMMEDIATE
```

Also, wir wissen nun, an der THEN-Stelle geschieht etwas anderes als einfach nur eine Überleitung zu den danach folgenden Anweisungen. Und es wäre wichtig, genau das *was* dort geschieht auch in einen Begriff zu fassen.

Der Sieger

Somit bedeutet THEN schon mal etwas anderes als “after that”. THEN löst dort genau an der Stelle die Vorwärtsreferenz der Verzweigung auf und genau an der Stelle muß ein Wort hin, welches so etwas tut. Es kann nicht einfach weggelassen werden. Und hierin liegt der Irrtum von M.L. Gassenkow.

Sagt THEN denn nun aus, *was* dort geschieht? Versuchen wir es einmal. M.L.Gassanenko selbst kommt uns dabei zur Hilfe. Denn er stellte in seiner Anfrage bereits Überlegungen dazu an, welche weitern Bedeutungen THEN im Englischen haben kann.

"The Concise Oxford Dictionary says:

then [@en] adv., adj., & n. --adv.
_1 at that time; at the time in question (...).
_2 _a next, afterwards; after that (then he told me to come in).
_b and also (then, there are the childen to consider).
_c after all (it is a problem, but then that is what we are here for).
_3 _a in that case; therefore; it follows that (...).
_b if what you say is true (...)
_c (...) if you must have it so (...)
_d used parenthetically to resume a narrative etc. (...).
--adj. that or who was such at the time in question (the then Duke).
--n. that time (until then).
(...)"

So wie Forth sein THEN immer verwendete, hat es mehr den Sinn von Fall 1, "at that time (resolve)", wie ich finde. Wir könnten es mit "zu der Zeit (lösen)" übersetzen. Vielleicht würde Chuck Moore heute in diesem Sinne anders formuliert haben. RESOLVE würde den Regeln von Brodie genügen und wohl passen.

```
<test> IF <do_this_and> RESOLVE
```

Doch die Tradition gebietet es uns, das Wort dafür nun nicht mehr zu verändern. Daher müssen wir uns begnügen, THEN 'richtig' zu verstehen. Die Bedeutung ist "at that time resolve THEN"

```
<test> IF <do_this_and_resolve_reference> THEN
```

Nun, es hat Spaß gemacht, all die Argumente in c.l.f zu lesen, den 'heiligen Krieg' über THEN zu beobachten. Wie er ausgegangen ist? Unentschieden, so 50 zu 50, meinten einige, die die Anzahl der Argumente gezählt hatten. Doch nach den Gewichten der Argumente ist wohl ANS der Sieger: Es wird nichts geändert.

Abschließende weise Worte:
"When I write Forth, I like to hear Forth.
IF...ELSE...THEN sounds like Forth to me.
This too: BEGIN...WHILE...WHILE...REPEAT...THEN .
But if you don't agree, use ENDIF then."

[LEO WONG MABS85B@prodigy.com
New York State Department of Civil Service
Albany, NY 12239]

Mein Dank geht an die Diskussionsteilnehmer des c.l.f Forum. Speziell an M.L.Glassanenko für die Anregung des Themas. Und an John O. Comeau der den Titel zu diesem Beitrag erfand.

"The Holy War was fun to watch, ..."
[jcomeau@world.std.com (John O Comeau)]

Literaturhinweise

Betreff : "IF..ELSE..THEN*is* natural" in comp/lang/forth im Juni 1996
(1) Leo Brodie, Thinking Forth; A language and Philosophy for solving Problems, 1984. (Kap.8, Minimierung von Steuerstrukturen).

❑

Glossar

c.l.f

/comp/lang/forth

Diskussionsforum über Forth, wird 2x täglich auch in der Mailbox der Forthgesellschaft abgebildet.

UPN

Umgekehrte polnische Notation. Englisch: RPN reverse polish notation.

IF

Forth Wort. Es compiliert einen bedingten Sprung, weiß aber zu der Zeit noch nicht, wohin der Sprung einmal gehen wird und trägt daher einen Dummy als Ziel ein. An dessen Stelle wird später das richtige Sprungziel angegeben. Der Vorgang wird "lösen einer Vorwärtsreferenz" genannt. Englisch: "Resolve forward reference".

ELSE

Forth Wort. Es setzt einen unbedingten Sprung mit Dummy als Sprungziel ein und löst dann die Vorwärtsreferenz des IF auf, so das der bedingte Sprung beim IF nun hier hin geht, wo die andere Alternative steht.

Forth-Gesellschaft: **Arbeitsgruppe Marc 4 im Aufwind**

Zwar sind wir noch nicht ganz einsatzfähig, aber da bei den potentiellen Interessenten frühzeitige Planung nötig ist, sei es jetzt schon angekündigt: Die Arbeitsgruppe wird MARC4-OTPs und (leihweise) Programmiergeräte für Diplomarbeiten zur Verfügung stellen. Inzwischen sind neun Muster des OTP eingetroffen. Wir bedanken uns bei der Firma Enatechnik.

Insbesondere für E-Technik Diplomarbeiten an Fachhochschulen hat der MARC4 die richtige Kragenweite. Viele Professoren akzeptieren eigene Vorschläge der Studenten für Themen von Diplomarbeiten. Wer selbst keine guten Ideen hat, kann die üblichen Applikationen angehen: Funkuhren, Waagen, Telefone, elektronischer Kompass usw. Besonders also Geräte die aus Batterien oder Solarzellen betrieben werden. Wobei von uns technische Hilfe nicht nur bezüglich des MARC4, sondern auch bei den Applikationen zu erwarten ist.

Im Gegenzug wollen wir die Diplomarbeit in einer eigenen Ausgabe des Newsletters veröffentlichen. Wir brauchen dazu die Unterlagen noch während die Diplomarbeit durchgeführt wird, damit der Newsletter unmittelbar danach zur Verfügung steht. Es ist nicht anzunehmen, daß der Professor den Newsletter als Dokumentation akzeptiert. Er wird sich jedoch ausgezeichnet in der Bewerbungsmappe machen, mit der der frischgebackene Ing. auf Stellungssuche geht.

Da von der TEMIC eventuell ein Emulator leihweise beschaffbar ist, muß man nicht unbedingt auf die Geräte Arbeitsgruppe warten und kann nominell sofort anfangen.

Rafael Deliano (jrd) April'96

Forth-Gesellschaft: **Microcontrollerverleih**

Neu im Verleih sind ein RSC-Board mit Rockwell 65 F 12 und ein 68 F 11. Dank an Rafael Deliano. Das 65 F 12 Board ging nicht. Hier hat mir Dirk Brühl unbürokratisch mit einer Ersatzkarte, Chips und Rat sofort geholfen.

Das 68 F 11 Kit konnte ich noch nicht testen, u.a. weil die Beschreibung in Englisch ist. Die gefädelte Karte müßte überarbeitet oder ausgetauscht werden (z.B. Franzis Buch + Platine).

Anfragen habe ich erhalten für Boards mit dem 8096 von Intel und dem 80166 von Siemens. Leider habe ich die nicht.

Wer dem Verleih Geräte zur Verfügung stellen könnte, sei hiermit ermuntert. Interessant wären auch ältere EMUF, MOPS aber mit Unterlagen, Software etc.

Nach Möglichkeit möge man mich zwischen 19:00 Uhr und 22:00 Uhr kontaktieren, außer Montag und am 1. Mittwoch jeden Monats.

Thomas Prinz
Microcontrollerverleih der
Forth Gesellschaft e.V. und
FG-e.V. Rhein Nechar-Gruppe
Adalbert-Stifter-Str.15
D-69412 Eberbach a/N
Tel.: 06271 / 2380
Fax: 06220 / 7065

Forth-Gesellschaft: **Jahrestagung '97**

Die Jahrestagung 1997 der Forth-Gesellschaft e.V. findet voraussichtlich vom 25. bis 27.April 1997 im Rhein Neckar-Raum statt.

Thomas Prinz, 06271 / 2830

Prozessorgeflüster: **Koopmann: 'Stackcomputer'**

Phil Koopman gab kürzlich in c.l.f bekannt, daß sein Buch über die·Architekturen von Stackcomputern jetzt on-line vorliegt. Grund: Der Druck wurde eingestellt. In dem Buch verglich er 7 Stackcomputer wie sie in den späten 80igern üblich waren miteinander.

Bezugsadresse:

http://www.cs.cmu.edu/~koopman/stack_computers/index.html

mka (Michael Kalus) nach einer News von Phil Koopman (koopman@cs.cmu.edu) in comp/lang/forth, Juli'96·

Forth inside: **Weiss Entwicklungspaket**

In der Franzis-"Elektronik" 9/96 findet sich eine Anzeige der Firma Dr. Weiss, in der ein Einplatinencomputer und Software angeboten werden. Wortreich werden "drastisch reduzierte Entwicklungszeit", "Interaktives Arbeiten aus der Kommandozeile", "komplettes Enwicklungspaket mit Compiler, Interpreter ..." angepriesen. Hört sich alles bekannt und verdächtig an. Doch mit keinem Wort erwähnt die Anzeige den Namen dieser Wundersoftware. Man läßt sich also Infomaterial schicken. Auch die vierseitige Hochglanzbroschüre gibt sich wortreich-vage. Doch halt: beim zweitenmal Durchlesen findet sich im Abschnitt Integer-Arithmetik im Unterabsatz genau einmal das gefürchtete F-Wort. Hat die Schere im Kopf mal wieder geklemmt? Wir fordern strenge Bestrafung des Schreiberlings für diesen Lapsus.

Rafael Deliano (jrd), Mai'96

Forth-Systeme: ***NEWZ jetzt Public Domain***

Tom Zimmers großer Editor NEWZ ist seit Juni 1996 frei benutzbar. Wir danken Tom, dem dieser Weg der praktischste erschien, um Fehlerkorrekturen - wie sie zum Beispiel von Ulrich Hoffmann vorgenommen wurden - so einfach wie möglich einer breiten Öffentlichkeit zugänglich zu machen. Der NEWZ-Editor ähnelt stark dem F-PC-Editor, hat allerdings einige Erweiterungen.

Beispielsweise in Zusammenhang mit Hypertext, oder wenn es darum geht, ihn eine Workshell (ZZ) einzubinden.

Claus Vogt hat sich bereit erklärt, für die nächste Zeit die Wartung zu übernehmen, Fehlermeldungen und Verbesserungsvorschläge zu sammeln und - vorraussichtlich in Zusammenarbeit mit dem Forthvertrieb Klaus Kohl - einen geeigneten Vertriebsweg zu finden. Eine schnelle Grundversorgung der Republik mit möglichst vielen verschiedenen Quelltextversionen ist allerdings nicht geplant.

Beta-Tester für die erste deutsche Version werden noch gesucht. Interessenten sollten eine kurze Mail oder einen mit 3.-DM frankierten Rückumschlag an meine neue Adresse senden.

Claus Vogt, Aug'96
clv@FORTH-eV.de
Katzbachstr.23; D-10965 Berlin

Forth-Systeme: **DOOF 0.1.2 released**

Dynamic Object-oriented Forth Version 0.1.2 wurde jetzt von Andras (h9290246@hkuxa.hku.hk) veröffentlicht und auf dem Taygeta-Server unter: taygeta:ftp://taygeta.com/pub/Forth/Linux/doof-0.1.2.tgz abgelegt. Diese neue Version ist bereits für Applikationen geeignet. Die Vorläufer dienten der Erprobung des Prinzips. Die Entwicklung läuft weiter. Objekte für die Anbindung an Betriebssysteme stehen bereit und für 'shared libraries', also den Zugriff auf z.B. C-Libraries. Das System läuft derzeit auf LINUX (nur Intel x86) und es gibt eine Portierung nach Solaris auf der Sun Sparc. Der Autor ist interessiert an Rückmeldungen zu seinem System. Andreas wurde bereits darauf hingewiesen, daß im deutschsprachigen Raum die von ihm gewählte Abkürzung für sein System ein Schimpfwort darstellt. DOOF klingt eben doof hier. Aber eine bessere Idee liegt noch nicht vor. Seis drum. Wer's hierzulande brauchen kann, dem ist's wahrscheinlich eh wurscht wie's heißt, oder?

mka (Michael Kalus) nach comp.lang.forth im Mai96

Forth online: **Forth Bretter**

Bekanntlich sind Forth Bretter kein geschnittenes Tropenholz. Ein 'Brett' ist hier die Verkürzung der etwas unglücklichen deutschen Übersetzung des englischen Begriffs 'bulletin board'. Ein 'bulletin' ist meinem penguin english student's dictionary nach ein "short public statement". Das 'board' dazu nenne ich gern 'Pinnwand'. Und ein 'bulletin system' wiederum. ach lassen wir das. Ihr wißt schon was gemeint ist. In unserer Mailbox in Kiel (Kiel bulletin board system, KBBS) bei Holger Petersen sind derzeit vier Bretter mit Forth drin zu haben:

- /comp/lang/forth – english, global, 10-30 news per day
- /comp/lang/forth/mac – english, global, 00-03 news per day
- /de/comp/lang/forth – deutsch, europa, 0-2 Nachrichten pro Tag
- /z-netz/sprachen/forth – deutsch, wird auch als Forum geführt.

Ferner haben wir die Bretter der Forthgesellschaft. Das sind /forthev/forum und forth-ev/allgemein. Das Brett /forth-ev/test ist ja lediglich ein Automat, der jede Nachricht sogleich an den Absender zurück schickt, damit mann/frau Modem und XP für den weiteren Datenverkehr fit machen kann, ohne echte Bretter zumüllen zu müssen.

Quelle: Brettliste Juli96 KBBS.
mka

Forth online:
Frequently asked questions

Seit die Forth Interest Group (FIG) im letzten Jahr ein neues 'Board of Directors' gewählt hat, hat sich einiges getan. Die FAQs - so heißen Zusammenfassungen häufig gestellter Fragen - des internationalen Netzbrettes comp/lang/forth sind zu ungeahnter Größe angeschwollen. Wir können selbst die Lite-Fassung der Inhaltsangabe nur noch teilweise zitieren:

"Here is a brief summary and description of the Forth language Frequently Asked Questions files. They are posted monthly to the comp.lang.forth USENET group, and are updated regularily. These files are available for reading, or downloading, at:

http://www.taygeta.com/fig.html
ftp://ftp.forth.org/pub/Forth/FAQ

...

- forthfaq.1 'general':General/ Miscellaneous. A brief historical background, a discussion of standards, and an overview of the FAQ
- forthfaq.2 'online': Online Resources, a relatively comprehensive guide to Forth info online
- forthfaq.3 'vendors': Forth Vendors, a listing of Forth vendors and authors
- forthfaq.4 'systems': Forth Systems: Commercial, Shareware, and Freeware, a large listing of various Forth and Forth-like systems for many different architectures (& budgets...)
- forthfaq.5 'books': Books, Periodicals, and Tutorials,- a bibliography of printed resources available for Forth users
- forthfaq.6 'groups': Forth Groups & Organizations, a listing of organizations for the Forth professional, and hobbyist ... "

Steht die Forth Gesellschaft eigentlich schon in forthfaq.6? Oder müssen wir da erst noch Bescheid sagen?

clv (Claus Vogt) nach comp/ lang/forth, Juli96

Medien:
SIGPLAN sucht Autoren

I invite the Forth community to participate in the Special Interest Group on Programming Languages (ACM SIGPLAN) by submitting Forth articles to:

Paul Frenger
Associate Editor (Forth)
P.O. Box 820506
Houston, TX 77282-0506

Voice: +1-713-293-9484
Fax: +1-713-293-9446

70410.1173@compuserve.com

pfrenger@ix.netcom.com
Regards, Irving Montanez

Past Chair SIGForth
[comp/lang/forth im Juli'96, korrigierte Fassung]

Presseschau: DDJ
Preemptive Multitasking in Forth

Any Yuen beschreibt im Dr. Dobbs Journal vom März'96 im Zusammenhang mit EFORTH (hier speziell für PC) einen interruptgesteuerten Multitasker - 'A Tiny Preemptive Multitasking Forth'. Dazu wird der im Timerinterrupt mit 18.2Hz aufgerufene Interruptvektor $1C auf eine eigene Routine umgeleitet, die dann die Taskumschaltung ausführt. Der Artikel enthält nur einen Auszug des Taskerlistings. Über FTP kann von Dr. Dobb's (ftp.mv.com, /pub/ddj) neben Beispielprogrammen auch das angepaßte EFORTH (insgesamt 88K) geholt werden.

KK@forthver.forth-ev.de
(Klaus Kohl), Mai'96

Presseschau: ELRAD
Qualität mit CRC

Immer wieder unterlegt Rafael Deliano in ELRAD seine meist auf praktische Computeranwendung bezogenen Artikel mit FORTH-Sourcen. Im Heft 2/96 befaßt er sich auf Seite 30ff unter dem Titel "Auf Fehlersuche - Qualitätssicherung beim Datentransfer mit Software-CRC" mit der Generierung der CRC(Cyclic Redundancy Check)- Prüfsummen, wie sie zur Sicherung von Datenübertragungen genutzt werden.

Neben Hintergrundinformationen über verschiedene CRC-Verfahren und deren Programmierung wird für das FORTH-Beispielprogramm zum M37451-Einplatinencomputer auch die benötigte Zeit angegeben.

KK@forthver.forth-ev.de
(Klaus Kohl), Mai96

Presseschau: Franzis
FORTH-Tagung'96

Die Franzis-"Elektronik" 4/96 vom 20.Februar hatte die Tagung vorbildlich aufgeführt.

Es war eine Ankündigung, d.h. man konnte noch an der Tagung teilnehmen. Die Meldung kam rechtzeitig, nämlich im Februar. Und der Text war kurz und präzise. Die Zeitschriften haben nämlich wenig Platz für weitscheifige Selbstdarstellung unbekannter Vereine.

Der Text in der VD 2/96 war eine Nachlese und weitschweifig. So unprofessionell, daß sie die Kröte schluckt, ist die ELRAD denn doch nicht.

Rafael Deliano (jrd), Juni96

Seminare

FORTH-Tagung '96

Die 12. Jahrestagung der FORTH-Gesellschaft e.V. findet vom 19. bis 21. April 1996 in Mittweida (Sachsen) statt. Thematische Schwerpunkte bilden „Embedded Control", „Applikationen" und „FORTH in Hard- und Software". In einer begleitenden Ausstellung präsentieren Firmen und Anwender Produkte und Applikationen.

Weitere Informationen bei der FORTH-Gesellschaft e.V., Postfach 1110, 85701 Unterschleißheim, Tel./Fax 089/3 17 37 96.

ANS Forth:
Forth wird ISO-Standard

Die Abstimmung der SC22 Gruppe der International Standardization Organization ISO über eine sog. Fast-Track Standardisierung auf Basis von ANS-Forth ist einstimmig angenommen worden (16:0:8). Forth ist damit Draft International Standard DSI und, sobald die Formalitäten abgeschlossen sind, international standardisiert.

uho (Ulrich Hoffmann), Aug'96

Firmen:
JW-Datentechnik expandiert

Die Firma JW-Datentechnik expandiert und hat ein neues Gebäude bezogen. Die Telefonnummern bleiben wie gehabt. Der Überhang an Bürofläche in München und der Niedergang der DASA machen's möglich! In ihrem stillgelegten alten Werk in Neuabing wird Bürofläche nicht mehr nach Quadratmetern, sondern nur noch nach kompletten Gebäuden abgegeben. Firmeninhaber Jens Wilke hat die Chance ergriffen und sich ein komfortables neues Domizil mit Blick ins Grüne gesichert. Trotz Umzug läuft die Entwicklung von ISDN-Routern mit Bernd Paysans BigForth schon wieder auf Hochtouren.

Die neue Adresse:
JW-Datentechnik
Brunhamstr.21
D-81249 München
Tel.: 089 / 897 68 90
Fax: 089 / 871 45 48

jrd/clv Aug96

Forth Online

Olaf Stoyke

os@cs.tu-berlin.de

In der heutigen Kolumne geht es um Stack-Prozessoren und die Optimierung von Programmcode für diese Umgebungen, seien sie nun in Hardware oder in Form sogenannter “Virtueller Maschinen” wie die Java-VM als Software realisiert. Ferner geht es um die Geschichte von FORTH, um eine FORTH-Umgebung sowie eine FORTH-Web-Seite im Aufbau und - last but not least - um ein Beispiel, wie man FORTH einfacher in die Ganglien eines Einsteigers bekommen könnte - bei mir funktioniert es jedenfalls!

http://www.cl.cam.ac.uk/users/rrt1001/ ist die Adresse eines Studenten der Cambridge University namens Reuben Thomas - Studienschwerpunkt: Virtuelle Prozessoren (siehe Java Virtual Machine) und Interpreter. Unter „Documents and Programs“ findet man vier Texte, die einen solchen Prozessor (Beetle Forth Virtual Processor), dessen Implementierung in ANSI-C sowie einen darauf basierenden FORTH-Compiler vorstellen. Der Quellcode ist dazu noch nicht verfügbar; nach einer Aussage vom Autor (persönliche Mail vom 4. Juni 1996) ist dies jedoch kein Zustand von Dauer mehr. Es könne sich nur noch um wenige Tage handeln, bis alles unter genannter Adresse verfügbar ist.

Unter http://www.cs.cmu.edu/afs/cs.cmu.edu/Web/People/koopman/ findet man die Seiten von Philip J. Koopman, Jr., der auch eine Seite zu Forth und Stack-Computern eingerichtet hat. Neben einer kurzen Einführung in die Sprache findet man dort einen Bericht zum Thema Optimierung bei Ausführungsumgebungen, die auf einem Stack aufbauen, also keine Register zum Speichern von Zwischenergebnissen o.ä. benutzten - Titel des Berichts: „A Preliminary Exploration of Optimized Stack Code Generation“. (Siehe auch: Journal of Forth Applications and Research, 1994, 6(3), S. 241ff)

Manchmal ist es ja recht interessant, etwas über die Geschichte und die Entwicklung einer Programmiersprache zu erfahren, um zum Beispiel herauszufinden, warum etwas so und nicht anders gelöst wurde. Eine solche Darstellung findet sich bei FORTH, Inc. und kann unter http://websites.earthlink.net/~forth/History_HTML/History1.html studiert werden. Der letzte Abschnitt dieses kurzen, historischen Überblicks enthält eine Literaturliste.

http://cuiwww.unige.ch/OSG/Vitek/Resources/Languages/Year95/languages/msg00045.html enthält die Beschreibung einer forth-ähnlichen Sprache namens 4t-H. Nach oberflächlicher Lektüre dieser nicht eben kurzen Textseite, handelt es sich dabei um ein System, daß aus einem C-Programm heraus aktiviert wird, um einen Zwischencode (H-Code genannt) zu erzeugen, der dann interpretiert wird. Ein Compiler ist ebenfalls Bestandteil dieses Pakets, welches sich derzeit in der letzten Testphase befindet. Partizipieren kann man durch Kontaktaufnahme mit der Autorin/dem Autor: hbezemer@vsngroep.nl. Der Klartext ihres/seines Namens war der Seite leider nicht zu entnehmen.

Da gerade der Begriff „Testphase“ gefallen ist: Eine neue Forth-Seite soll im Web unter http://www.ionet.net/~saylor/pronet/forth.shtml erstellt werden. Derzeit ist da noch gar nichts zu sehen, da der verantwortliche Autor (Steve Aylor) noch nach Inhalten sucht - ein Unterfangen, bei dem er übrigens um Unterstützung bittet: Vorschläge, was dort eingebaut werden sollte, können ihm über die Adresse saylor@ionet.net per eMail zugesandt werden. (Sieht man sich auf diesem Rechner etwas genauer um, stellt man fest, daß da noch eine ganze Reihe von Seiten aufgebaut werden (sollen): Assembler, 4th Dimension, Java, etc.)

Die Firma Mosaic Industries, Inc. (Newark, US-Bundesstaat Kalifornien) beschäftigt sich mit allerlei Controller-Karten und anderer Hardware aus diesem Umfeld und bietet unter anderem das „QED Board“ an, welches wohl eine I/O-Karte auf der Basis des MC68HC11F1 von Motorola ist; wie dem auch sei (Merkt man, daß ich kein Hardware-Mensch bin?), das Teil kann mit einem Forth programmiert werden, dem QED-Forth, zu dem eine Einführung auf http://www.mosaic-industries.com/gsf-ch4.html zu finden ist. Die Homepage der Firma erreicht man, wenn „gsf-ch4.html“ aus der Adresse entfernt wird.

In der letzten Ausgabe der VD habe ich die Seiten der FIG-Gruppe im US-Bundesstaat Maryland (http://www.bcpl.lib.md.us/~nbuck/fig.html) vorgestellt und den Terminus „farblos und trist“ verwandt. Nun ich bin recht froh und glücklich, daß ich dieses Urteil zumindest in großen Teilen revidieren kann: An diesem Angebot wird offensichtlich gearbeitet! Berichte von den dortigen Gruppentreffen, FORTH-Beispielcode, Lektionen zur Sprache und eben jenes Betriebssytemprojekt sind nun auf der Seite vertreten. Nach wie vor nicht bunt, aber immerhin…

Zum Abschluß noch ein Tip von einem Anfänger an alle anderen Mitglieder der FORTH-Gesellschaft (oder solche in spe) gleichen Bildungsstands: Wer sich der Sprache FORTH lieber mit einfachen und übersichtlichen Diagrammen nähern will, sollte einen Blick in die Seiten ab http://www.forth.org/forth_intro/stackflo.htm werfen. Dort wird mit einer grafischen Metapher namens „StackFlow“ gearbeitet, um eben den Datenbewegungen auf dem Stack habhaft zu werden. Diese äußerst hilf- und bildreiche Einführung verdankt der Leser Gordon Charlton, einem FIG-Mitglied, das sich seit 1985 mit FORTH beschäftigt. (Mein Tip zu der genannten Seite: Das Browser-Fenster horizontal etwas ausdehnen, einige der Grafiken interagieren sonst recht unschön mit dem umlaufenden Text!)

❑

Forth-Gruppen regional

Rhein-Ruhr
Jörg Plewe
Tel.: +208 -49 70 68 p
Treffen: jeden 1. Samstag im Monat im S-Bahnhof Derendorf Münstererstr. 199,

Moers
Friederich Prinz
Tel.: +2841 -5 83 98 p
Treffen: jeden Samstag 14:00 Arbeitslosenzentrum, Donaustr. 1, Moers

Darmstadt
Andreas Soeder
Tel.: +6257 -27 44

Mannheim
Thomas Prinz
Tel.: +6271 -28 30 p
Ewald Rieger
Tel.: +6239 -86 32 p
Treffen: jeden 1. Mittwoch im Monat, Vereinslokal Segelverein Mannheim e. V., Flugplatz Mannheim-Neuostheim

µP-Controller Verleih

Thomas Prinz
Tel.: +6271 -28 30 p

Gruppengründungen, Kontakte

Regional

Stuttgart
Wolf-Helge Neumann
Tel.: +711 -8 87 26 38 p

Fachbezogen

8051 ... (Forth statt Basic, e-FORTH)
Thomas Prinz
Tel.: +6271 -28 30 p

Forth-Hilfe für Ratsuchende

Forth allgemein
Jörg Plewe
Tel.: +208 -49 70 68 p
plewe@mpi-dortmund.mpg.de

Karl Schroer
Tel.: +2845 -2 89 51 p

Jörg Staben
Tel.: +2103 -24 06 09 p

Spezielle Fachgebiete

Arbeitsgruppe Marc4
Rafael Deliano
Tel./Fax.: +89 -841 83 17 pf

Anfänger und Wiedereinsteiger
Gerd Limbach
Tel.: +2051 -25 51 12 p
Mo.+Di. 20:00 - 22:00

32FORTH (Atari)
Rainer Aumiller
Tel.: +89 -6 70 83 55 gp

FORTHchips (FRP1600, RTX, Novix...)
Klaus Schleisiek-Kern
Tel.: +40 375 008-03 g

F-PC & TCOM, ASYST (Meßtechnik), embedded controller(H8/5xx// TDS2020, 8051 ... eFORTH...), FUZZY
Arndt Klingelnberg
Tel.: +2404 -6 16 48 agp

HS/Forth (Harvard Softworks)
Wigand Gawenda
Tel.: +30 -44 69 41 p

KI (Künstliche Intelligenz), OOF (Object Oriented Forth)
Ulrich Hoffmann
Tel.: +4351 - 712 217 pf

Birgit Steffenhagen
Tel.: +38204 - 129 33 pa
+381 - 498 35 52 g

Forth-Vertrieb
volksFORTH/ultraFORTH, RTX/FG/Super8/KK-FORTH
Ingenieurbüro Klaus Kohl
Tel.: +8233 -3 05 24 p
Fax: +8233 -99 71 f

Forth-Mailbox (KBBS)
+431-533 98 98 *:8N1*
Sysop: Holger Petersen
Fax: +431-533 98 97 f
Tel: +431-533 98 96 p *bis 22 Uhr*
Mail: hp@kbbs.org

Hinweise

Zu den Telefonnummern

f == FAX
a == Anrufbeantworter, hier können Sie Ihren Ansprechpartner eventuell vorinformieren, erwarten Sie bitte keinen (kostspieligen) Rückruf
g == geschäftlich, zu erreichen innerhalb typischer Arbeitszeiten
p == privat, zu erreichen außerhalb typischer Arbeitszeiten

Die Adressen des Forth e. V. (Forth Büro) und der Redaktion/ finden Sie im Impressum

FORTecH Software

- Forth-Entwicklungsumgebung comFORTH unter DOS oder Windows
- interaktive Crossentwicklungssysteme für Mikroprozessoren von Intel, Motorola, Zilog, TI ...
- Softwareentwicklung für PC und Mikrocontroller
- System- und Anwendungsprogrammierung unter Windows

comFORTH

- Forth-Entwicklungsumgebung für Windows
- interaktive Benutzbarkeit aller Windows-API-Funktionen und -Strukturen
- kombinierbar mit anderen Programmiersprachen
- Unterstützung von DDE, DLL, VBX, ...

fieldFORTH

- Forth-Entwicklungssystem für eingebettete Systeme
- interaktive Programmierung off-line und on-line
- verfügbar für diverse 8-, 16- und 32-Bit Mikrocontroller und -Prozessoren (TMS320C40, M68332, M68HC11,...)
- **NEU!!!** Evaluation-Kit M68HC11 inclusive Board MINI-HC11 296,70 incl. MwSt.

FORTecH Software GmbH
J.-Jungius-Str. 9 • D-18059 Rostock • Tel: (03 81) 4 05 94 72 • Fax: (03 81) 4 05 94 71