VIERTE DIMENSION

Vierte Dimension
Volume II/Nr.1

VOLKSFORTH
STACK
STRINGSTACK
KOMMUNIKATION

FORTH MAGAZIN

5,00 DM

I N H A L T
==========

+++++ OK +++++ OK +++++ OK +++++ OK +++++ OK +++++ OK +++++

==----> ERNEUERN SIE IHREN BEITRAG JETZT <----==
==----> ERNEUERN SIE IHREN BEITRAG JETZT <----==
==----> ERNEUERN SIE IHREN BEITRAG JETZT <----==

Und das Forth Magazin
Vierte Dimension
kommt weiterhin
zu Ihnen nach Hause

==========
Mit den Mitgliederlisten des Vereins
==========

* * *

In diesem zweiten Jahr
wird sich zeigen, ob Forth genug Anhänger hat,
um in der Computerwelt der Bundesrepublik Deutschland
eine Rolle in der öffentlichkeit zu spielen, und
ob Forth eine Sprache wird, die allen gehören kann,
weil Forth Systeme freie Software bleiben.

Das ist unsere Aufgabe.

Wenn Sie mit Forth arbeiten,
wenn Sie mit Forth lernen,
wenn Sie mit Forth Geld verdienen,
wenn Sie mit Forth anfangen wollen,

werden Sie Mitglied in der Forth Gesellschaft eV.

* * *

Renew Now

EDITORIAL

Ein Forth Magazin, wozu?

M.Kalus, Editor

Um mit Forth warm zu werden, sollte man in Tuchfühlung mit erfahrenen Forthlern bleiben. Das öffnet den Blick für die simplen Lösungen und schult den Gebrauch von Forth an konkreten Problemen. Und weil die Sprache so lebendig ist, sich wandelt und entwickelt, behalten diejenigen den Anschluß, die die Verbindungen aufnehmen. Kommunikation über Forth ist unabdingbar für einen erfolgreichen Umgang mit dieser Sprache. Das Forth Magazin VIERTE DIMENSION wurde 1985 in Hamburg gegründet, um im deutschspachigen Raum diese Beziehungen knüpfen zu helfen. Ich wünsche uns einen erfolgreichen zweiten Jahrgang.

*

Was wüschen wir uns im Magazin? Anschriften und Termine, Anzeigen aus der Forthwelt natürlich, Anwendungsberichte, Forthtechnik, Forthstil, Quellenhinweise, Utilities, Grundlagenthemen, Word-Review, Tools, State of the Art, Forthmaschinen und tausend und eine Sache mehr. Aber unser Magazin ist ein Vereinsblatt, das von den Mitgliedern gemacht wird. Es lebt nur, wenn die Besten in Forth uns zuschauen lassen, wie sie mit Forth umgehen. Die Redaktion alleine kann keine Artikel herbeizaubern.

Wer darf für die VIERTE DIMENSION schreiben? Jederman kann es versuchen. Sie können sich dabei als Forthprogrammierer bekannt machen. Honorar gibt es leider nicht. Platz für einen Artikel kann man nicht kaufen (Platz für Anzeigen natürlich schon..). Aber wer durch Forth gutes Geld verdient hat, darf sich dessen rühmen. Und wem das noch nicht vegönnt war, der soll nicht neidisch "Schleichwerbung" rufen.

Wenn Sie einen Beitrag für die VD haben, schreiben Sie mir. Der Forth Code wird von gestandenen Forthlern gesichtet. Besprechungen Ihrer Programme und Produkte in Soft- oder Hardware mit Bezug zu Forth werden wir mit Vorliebe wiedergeben. Der 'ideale' Artikel hat eine Seite Text für zwei Seiten Code. Skizieren Sie Ihre Aufgabe durch ein Beispiel. Welche Teile wurden vorteilhaft in Forth gemacht? Schreiben Sie, welches Forth Sie eingesetzt haben. Aber wir wollen nicht nur Code bringen. Meinungen und Entwicklungsvorschläge zu Forth oder der Forth Gesellschaft interessieren uns ebenso wie Kritik.

* * *

GAST EDITORIAL

Elegant, Clever oder Trickreich?

M.Ouverson, Editor, Forth Dimensions, USA

Forth ist formbar wie Wachs. Wegen seiner Ausdruckskraft spiegelt ein Stück fertiger Code den kreativen Denkprozeß, und vielleicht sogar emotionale Charakteristika des Programmierers wieder. Diese Ansicht ist fundamentaler als die Debatte darüber, ob jemand Fix- oder Floatingpoint-Arithmetik benutzt. Aber da so etwas leichter zu verstehen ist, wird es mehr beredet als die Feinheiten und die Eleganz, mit der ein echtes "Forth-like" Programm arbeitet. -->

Wie kommt es, daß einige Forthprogramme weniger lesbar als Assembler Code aussehen, während andere sehr leicht verstanden werden? Wie ist es möglich, daß ein Stück Forth Code aussieht wie ein Autobahnverkehrsknoten in Los Angeles nach Büroschluß, während andere hübsch friedlich anzusehen sind wie ein Picknik im Grünen?

Wir wollen keine langatmigen Vergleiche von Forth mit irgendwelchen anderen Sprachen anstellen. Das führt ins Abseits. Es ist schnell gesagt: Wenn man die Besten anschaut, sieht es so aus, als ob gute Forthprogramme erst dann entstehen, wenn die Grundlagen wirklich verstanden worden sind und man im Geiste nicht mehr aus Konzepten von Pascal oder C oder Basic oder irgendeiner anderen Sprache übersetzt. Es hat die nötige Eleganz, wenn die Lösung ähnlich wie ein Hologramm zuerst im Kopf entsteht, bevor sie in Forth niedergeschrieben wird, den Standardkriterien entspricht und charakterisiert ist durch die natürliche Direktheit, die dem Leser das Aha-Erlebnis vermittelt: "Das isses!"

Was nun ein "Forth-like" Programm ist, läßt sich so schwer definieren, wie generelle Regeln darüber zu finden, was eine Lösung elegant macht, anstatt einfach nur schlau oder was noch schlimmer ist, trickreich zu sein. Um gut zu schreiben, wird man mehrere Anläufe brauchen, in denen zunächst nur die Leistung Beachtung findet, aber denen noch der eine Schritt weiter zur Eleganz fehlt. Glücklicherweise unterstützt Forth Annäherungsverfahren und Modifikationen. Wenn wir wollen, erlaubt Forth uns, zunächst eine arbeitende Lösung, ein Modell zu verfassen und daran die logischen Grundlagen des Problems zu studieren, um diese schließlich in einfacher und natürlicher Form zum Ausdruck zu bringen.

Mimikrie, die Fähigkeit sich das Aussehen eines anderen zu geben, ist ein Weg, der bei so vielen Firmen und Herstellern in der Computerindustrie einschlagen wird, um aus "ich-habs-auch" und "sieht-so-aus-wie" schnelles Geld zu machen. Vorbilder jedoch, insbesondere die mit evolutionären Werkzeugen wie Forth, halten Ausschau nach deutlich besseren Lösungen von alten Problemen. Und sie achten auch darauf, was Forth selbst gut stünde.

Um den Bedürfnissen von Firmen zu genügen, die eigene Programmierer beschäftigen, muß Forth das haben, was Professionelle brauchen an Tools, Utilities, Dokumentation und Service, und wenn ein Anbieter oder Berater sich einen dauerhaften Namen machen will, muß er etwas zu dem Strom der Innovationen, die periodisch in Wellen der Begeisterung auf den Markt schwappen, beitragen. Er muß in neue Richtungen weisen. Das erfordert den Dialog mit den Benutzern, und den Willen, etwas Besseres zu machen. Diejenigen, die darüber hinaus gehen, einfach nur lauffähigen Code zu verfassen und Beispielhaftes schaffen, einen Grundstock legen, der "Forth-like" ist in jeder Hinsicht, werden eine führende Rolle spielen können und werden dies sicherlich in materiellen Erfolg umsetzen können. Wir sollten in unserem eigenen Betrieb diese Qualitäten sorgfältig ausbilden, so sorgfältig wie der Künstler das Wachsmodell, bevor es abgegossen wird.

* * *

BEMERKUNGEN HINWEISE IMPRESSUM

Der Zeichensatz

Wir verwenden den Ascii-Zeichensatz. Da der Codevorrat nicht für alle Zeichen der Welt reicht, sind einige Schriftzeichen des ASCII-Zeichensatzes von Land zu Land unterschiedlich belegt.

Zum Beispiel unsere Umlaute ÄÖÜ sind in den USA [\]. Forthcode wird - soweit das geht - als Original-Listings einkopiert, um keine Tipfehler einzuschleppen. Und da kann es passieren, daß ein Autor keinen 'amerikanischen' Zeichensatz drucken kann. Wir lassen das dann so. Andererseits ist es so, daß wir Listings im 'amerikanischen' Zeichensatz präsentieren und Sie dann beim eintippen auf Ihrer 'deutschen' Tastatur einige Zeichen nicht finden.

Deshalb geben wir Ihnen in den nachfolgenden Tabellen die gebräuchlichsten 'Übersetzungen' der Zeichen an. In anderen europäischen Ländern finden sich noch andere Codebelegungen. Und manche Computer-Tastaturen sind nicht 'vollständig' ausgerüstet (der alte AppleII z.B.). Doch oft wird es sich schon irgendwie machen lassen. (MK)

U.S.A. Matrixdrucker

```
 !"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~
```

B.R.D. Matrixdrucker

```
 !"#$%&'()*+,-./0123456789:;<=>?
§ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ^
`abcdefghijklmnopqrstuvwxyzäöüß
```

B.R.D. Typenrad

```
 !"#$%&'()*+,-./0123456789:;²=³?
§ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖÜ°_
àbcdefghijklmnopqrstuvwxyzäöüß
```

===Impressum==========

Titel:	VIERTE DIMENSION Magazin für die Mitglieder der Forth Gesellschaft eV
Herausgeber:	Forth Gesellschaft eV
Redaktion & Anzeigenleitung:	Michael Kalus, Präsidentenstraße 40 5830 Schwelm, Telefon: 02336-82204
Druck:	Eckart Schmidt, Kodak
Forth:	Klaus Schleisiek und die Mitglieder des Review-Boards sowie alle namentlich genannten Autoren
Auflage:	500 Stück europaweit
Erscheinungsweise:	In jedem Quartal
Redaktionsschluß:	Der mittlere Quartalsmonat

ARTIKEL BITTE NUR AN DIE REDAKTIONSANSCHRIFT EINSENDEN!

 Freie Mitarbeit ist erwünscht. Eingesandte Artikel müssen jedoch frei sein von Ansprüchen Dritter. Eingesandte Artikel und Programme gehen, sofern nicht anders vermerkt, in die Public Domain über. Manuskripte bitte als Maschinentext einreichen oder per DFü (300Bd,ASCII) übergeben. Listings werden nur veröffentlich, wenn sie mit frischem Farbband ausgedruckt wurden, da wir diese fotomechanisch nachdrucken, um Fehlerquellen auszuschließen.

==========

LESERBRIEFE

Forth und Graphik

Die Computergraphik hatte bis noch vor wenigen Jahren einen leicht unseriösen Beigeschmack. Doch spätestens seit die enorme Wichtigkeit einer leistungsfähigen Schnittstelle bei der Mensch-Maschine-Kommunikation erkannt wurde und in immer höherem Maße alltägliche visuelle Präsentationen (z.B. ARD-Logo) rechnergestützt erzeugt werden, ist die Computergrafik ein fester und allgegenwärtiger Bestandteil der Computerei.

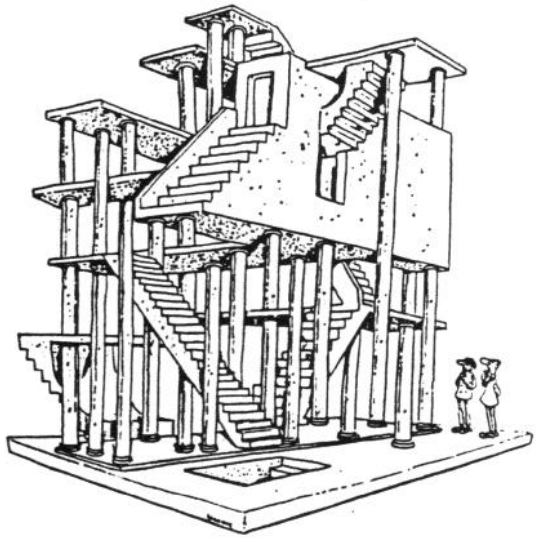
Auf der EuroFORML'85 wurde deutlich, daß auch in Forth-Kreisen großes Interesse an Computergraphik besteht, jedoch kaum wesentliches Know-How vorhanden ist. Weiterhin verlangt Forth, zieht man klassische 'Graphiksprachen' zum Vergleich heran, ein Vorgehen nach anderen Kriterien , sowohl bei der Implementation der elementaren Grafikfähigkeit, als auch auf der Applikationsebene.

Um nun die Kommunikation aller an Graphik interessierten Forthler zu starten, biete ich mich als zentrale Ansprechstelle an. Es ist keine feste AG geplant, sondern vielmehr ein lockerer Informations- und Erfahrungsaustausch auf allen Ebenen, vom Anfänger bis zum Experten. Ich selbst arbeite speziell in den Bereichen 3D-Animation, Ray-Tracing und Image Processing. Kontaktgesuche bitte möglichst telefonisch oder via DFÜ, da der Briefwechsel zu kompliziert ist. DFÜ wird über den 'ComuniTree' der Forthgesellschaft erfolgen.

Marco Fauck, 2000 Hamburg, Friedensallee 92, Tel.:040-3900139

Dictionary Struktur

Zum Leserbrief von U. Kloss VD I/3 möchte ich noch kurz sagen: Aufgepaßt! Wenn man an der Dictionary-Verwaltung rumdreht, kann man leicht an einen Punkt gelangen, wo im Bezug auf Kompatibiltät mit anderen Systemen auf anderen Rechnern nichts mehr läuft. Dies würde aber mit Sicherheit die Sprache für viele uninteressant und damit kaputtmachen. Auch könnte man dann auf die Veröffentlichung von Listings vezichten, da sie sowieso nur noch auf XY-Forth laufen. Und man darf nicht außer acht lassen, daß Forth nunmal nicht Basic oder Pascal ist, wo man leicht zwischen verschiedenen Dialekten übersetzen kann. Forthworte, die den Compiler erweitern oder direkt im Dictionary herummanipulieren, sind mit Sicherheit für viele Anwender nicht so leicht vollständig zu durchschauen und anzupassen. Hier muß der Grundaufbau der Sprache auf jeden Fall erhalten bleiben, damit nicht eine babylonische Sprachvewirrung geschaffen wird, die jedem Einsteiger die Sprache Forth vermiesen wird.

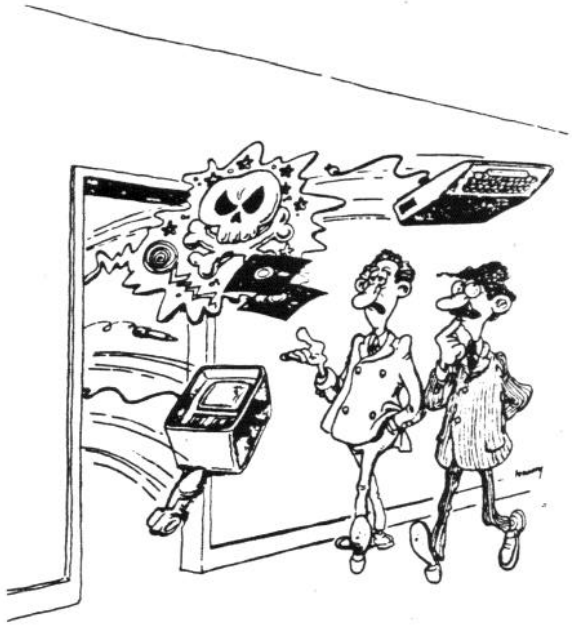


R. Hamann, Braunschweig

Standard für 16-bit-Systeme?

Etwa zwei Jahre habe ich mit einem Z80-figForth'1.1 gearbeitet. Nun habe ich mir aber einen 16-bit Rechner zugelegt (Atari 520 ST) und zwei Forthversionen angeschaut. Die eine war fig-orientiert, verarbeitete aber nur 64K meiner 500K. Eine andere bediente den großen Speicher, guter Editor, schnell und prima an die Maschine angepasst. Das Erwachen kam dann, als ich ein Programm in F83 aus Dr. Dobbs abtippen wollte: Die Wörter sind total anders aufgebaut, nichts ging.

Gibt es für 16-bit-Forth und die großen unsegmentierten Speicher bereits einen 'Standard'?
Peter Jodda, Celle



(Vierte Dimension: Vermutlich nicht. Zwar macht bei den 16-bit CPU's z.Z. jeder was er will, doch sind auch schon solide Versionen auf dem Markt, die sich gemäß dem 83-Standard verhalten. Hier auf dem Kontinent wird hoffentlich das VOLKSFORTH aus der Public Domaine das Rennen im 16-bit-83-Standard machen. Wir berichten darüber in diesen Heft. Im Forth-Standard-Team in und bei den FORML-Konferenzen in den USA liegen zuhauf Vorschläge zu diesem Thema und auch jüngst auf der EuroFORML'85 in Stettenfels wurde dazu vorgetragen. Eine überzeugende Strömung auf einen "eigenen" 16-bit-Standard hin ist bis jetzt jedoch nicht zu erkennen. Wer zu diesem Thema etwas beitragen möchte, ist hiermit herzlich eingeladen.)

Hofacker Forthgruppe München

Ich beabsichtigte, der Forth Gesellschaft beizutreten und erhielt vor einigen Tagen die Ausgabe I,3/4 der VIERTEN DIMENSION. Beim blättern stieß ich auf die Seite 48 und da überkam mich doch wieder einmal das kalte Grausen. Wir beim Hofacker Verlag sind es gewöhnt, daß immer wieder auf unseren Produkten herumgehackt wird (VD: Es ging um's Forth für C64...). Wenn der Schreiber dieser Beurteilung nur ein kleines bischen Ahnung von Forth gehabt hätte, so hätte ihm auffallen müssen, daß der verwendete Editor nicht der Forth Line Editor aus dem Installationsmanual (VD: des figForth), sondern das es der Forth Inc. Editor ist, wie er in den Forth Dimensions, VolIII, No.3, pp80-89 beschrieben ist (...).

Es spricht für sich, wenn dieser Editor als eine "einfallslose Implementierung" bezeichnet wird, da er in dieser Form lange Zeit beim Erfinder vom Forth, Mr. C.Moore und der Forth Inc. verwendet wurde, und wie er, mit einer kleinen Erweiterung, auch in F83 verwendet wird (...).

Die erste Implementierung unserer fig-Version erfolgte im Frühjahr 1982 auf einem AppleII und wurde von dort über eine Zweidrahtleitung in die Assembler für den Atari 800 und C64 überspielt. Ich selbst programmiere seit 1979 mit Forth und schreibe nahezu alle Anwenderprogramme in dieser Sprache (...).

Meine Absicht war es, hier in München einen Zusammenschluß

der Forthanwender zu organisieren. Ich werde dies trotzdem tun, aber außerhalb der Forth Gesellschaft.
Ekkehard Flögel, Hofackerverlag, Holzkirchen

(Vierte Dimension: Vielen Dank für die Hinweise. Der Vergleich der verschiedenen C64 Forth's war von Ing. Othmar Kreil in östereich, figForth- und 6502-Benutzer. --- Einfache Editoren haben natürlich ihren Sinn, z.B. in kleinen Systemen mit knappen Speichern. Das hält den Forth-"Kern" klein und doch noch immer komfortabler als in Monitoren oä. Doch der C64 ist ja schon keine "8K-Maschine" mehr. Das es heute schon netteres für ihn gibt, sieht man am ultraForth83. --- Schade das Sie das Kind gleich mit dem Bade ausschütten. Nicht Herr Kreil, auch nicht die Vierte Dimension, ist "die Forthgesellschaft". Die Vierte Dimension ist ein Forth Magazin, das Beiträge aus der Forth Gemeinde verbreitet. Wir freuen uns auf Ihren Beitrag über Ihr Produkt! Vielleicht ist die Forth Gesellschaft als gemeinnütziger Verein auch für Sie und Ihre Gruppe im Münchener Raum bedeutsam.)

* * *

PARIS, le 18 mars 1986

ASSOCIATION JEDI
8, rue Poirier de Narçay
F-75014 PARIS

FRANCE / FRANKREICH

A l'attention de Mr Michael KALUS FIG Wuppergebiet
Ludorfstraße 5
D-5600 WUPPERTAL 2

Monsieur,

Notre association, loi 1901 (but non lucratif) se propose de développer la connaissance des nouveaux langages informatiques. C'est pourquoi nous prenons contact avec vous. Nous avons relevé vos coordonnées dans la rubrique "CLUB" du magazine CHIP.

Le langage le plus développé pour le moment par notre association est le langage FORTH. A ce titre, nous pensons que vous seriez certainement intéressé par un échange suivi, et régulier entre votre club et le nôtre.

Vous trouverez ci-joint un exemplaire de notre magazine mensuel JEDI (May be have the FORTH with you !!). Si votre club diffuse lui aussi des programmes ou articles, nous serions intéressés par leur contenu. Nous tenons à vous faire remarquer que nous n'avons pas de copyright en ce qui concerne le contenu de notre magazine si ce n'est pas à des fins commerciales. Par conséquent, si nous trouvons un terrain d'entente, vous trouverez certainement dans JEDI des trucs et astuces dont vous pourrez faire profiter vos adhérents, la réciproque étant vraie.

Dans l'attente d'une réponse de votre part, nous vous souhaitons une bonne lecture de JEDI et vous prions d'agréer l'expression de nos salutations FORTHiennes.

LE SECRETAIRE
M.PETREMANN

Association JEDI
8, rue Poirier de Narcay
F-75014 Paris

den 18 März 1986

Monsieur,

unsere Gesellschaft gemäß Gesetz 1901 (aber nicht mit Gewinnabsicht) widmet sich der Entwicklung der Kenntnis neuer Programmiersprachen. Wir haben Ihre Adresse der Rubrik "Club" der Zeitschrift CHIP entnommen.

Die durch unsere Gesellschaft zur Zeit am meisten entwickelte Programmiersprache ist FORTH. Deshalb nehmen wir an, daß Sie an einem regelmäßigen Gedankenaustausch zwischen Ihrem und unserem Club interessiert sind.

Beiliegend finden Sie ein Exemplar unseres Monats-Magazines JEDI (Möge Forth mit dir sein!). Wenn Ihr Club auch Programme und Artikel verbreitet, wären wir an den Inhalten interessiert. Wir möchten bemerken, daß wir kein Copyright auf unserem Magazin haben, da wir keine kommerziellen Interessen verfolgen. Deshalb finden Sie in JEDI zu unserem gemeinsamen Gebiet sicherlich Tips und Tricks, von denen Ihre Mitglieder profitieren können, und das gleiche gilt umgekehrt.

In Erwartung Ihrer Antwort wünschen wir Ihnen eine angenehme Lektüre von JEDI und wir verbleiben mit den besten FORTH-Grüßen.
M.Petremann, Le Secretaire

(Vierte Dimension: Herzliche Grüße nach Paris! Viel Spaß mit unserem Forth Magazin. Auf eine gute Zusammenarbeit. --- Was war in JEDI? Zu Forth ein Artikel von Daniel Grossman über das Cordic-schema, ins französische übertragen; ein gründlich ausgearbeiteter Vorschlag für die Syntax einer Floating-Point-Standard-Erweiterung (12 Seiten), von Tracy und Grossman; ein Artikel über den Gebrauch von Vektoren im 83-Standart von Petreman. Zudem etwas über MUMPS und die Sprache LPB.)

* * *

VERSCHIEDENES

Nennt bitte Eure Telefonnummern!

Bitte haben Sie Verständnis. Ich beantworte alle Briefe gewissenhaft. Aber ich brauche dafür Zeit. Da kann es vorkommen, daß mal 2 bis 3 Wochen ins Land gehen, bis Sie an der Reihe sind. Je konkreter eine Frage ausfällt, umso eher haben Sie eine konkrete Antwort. Natürlich: Manchmal ist eine Briefantwort nicht möglich, weil eher eine allgemeinere Beratung gesucht wird. Schreiben Sie dann doch bitte Ihre Telefonnummer dazu. Dann kann ich Sie beraten oder vermitteln. Ihr Editor

Forth für den Atari

In Ermangelung käuflicher 16 Bit Forth Systeme für meinen Atari 520 ST habe ich mir die Mühe gemacht, dieses selbst zu implementieren. Und da im Bereich von 16Bit Forth Systemen Schwierigkeiten in der Einhaltung des Standards auftreten, so bei Arithmetik Operationen, wäre ich an einer Zusammenarbeit hierüber interessiert. Mein Forth System stelle ich gerne jedermann zur Verfügung, der daran interessiert ist. Mit herzlichen Grüßen an die Forthgemeinde.
R. Hamann, Zimmerstr.27
Braunschweig

(Vierte Dimension: Herzlichen Dank. Wer kann eine Arbeitsgruppe dazu anbieten?)

دكتور
محمد ابو شادي
أخصائي باطني - جامعة القاهرة
اطفال جامعة مونبليه بفرنسا
ت ٥١٠٦٥٤

Graphik-Freaks,
Arabien wartet auf
Euren Textprocessor!

Forth-83-Standard im mc-Format

Herr Petrikowski in Nürnberg bietet sein FORTH-83 als freie Software an. Schicken sie Ihm eine formatierte 5"-Diskette im mc-Format oder in (fast) jedem anderen CP/M-Format nebst Diskparametern. Gerade eben hat er dafür auch eine 32Bit-Worte Erweiterung fertig. Wir danken für die Erlaubnis, dies bekannt machen zu dürfen. (mk)
Günter Petrikowski, Füll 10, 8500 Nürnberg 1

(Vierte Dimension: Es ist immer gut, zunächst mitteinande zu telefonieren um genauer abzusprechen, wie ein Programmtausch geschehen soll und wie man sich entgegenkommen kann. Es ist Ehrensache, daß dem Verteiler von Public Domain Forth zumindest seine Versandkosten und Disketten ersetzt werden.)

Der Forth Markt in den USA

In der letzten FORTH DIMENSIONS, Forth Magazin der Forth Interest Group (fig) in den USA, wurde wieder allerlei angeboten. Ein kleiner Streifzug durch den Markt zeigt, daß auch in den Staaten die Tendenz vorherrscht, Basisversionen der Forthsysteme praktisch Public Domain abzugeben. Die etablierteren Anbieter glänzen zwar noch ehr mit gesalzenen Preisen für simple Standards, aber es ist Bewegung in die Preise gekommen. Was gab es da also alles schönes?

Mach1 ist ein 32Bit-Forth für den Macintosh von Apple. Hat "Multi-tasking, headerless code, macro substitution, vectored I/O, named parameters, manual". Zwei interessante Worte tauchen da auf, einmal WORKSPACE, das Projekte in der Entwicklung abspeichert, die dann mit den Ikonen beim nächsten booten wieder da sind. Und TURNKEY, welches ein fertiges Projekt stand allone Applikation anlegt, das nur noch 16K Overhead durch das multi-tasking operating system hat. Dabei "Macin Talk for words that speak". Für den Atari und Amiga soll Mach1 ab Frühjah '86 zu haben sein; angeboten für 50$.

Der Mac mit 10 MB Paradise Hard Disk wird angeboten für 699$, mit 20 MB für 899$.

Bei LMI gibt es jetzt Forth-83 in 16-bit und 32-bit Implemen-tation und Cross-Compilern für 8080, Z80, 8086, 68000, 6502; ohne Preisangabe.

Die Forth Inc. bietet PolyforthII auf IBM-PC mit 8087-Mathematik-Koprocessor-Support und Graphik-Paket an; ohne Preisangabe.

Harvard Softworks ist wieder mit HS/Forth für IBM-PC, Compac und Tandy dabei , einem Forth-79 und Forth-83 für 8088, 80189 und 8087; ohne Preisangabe.

Micro Computers Applications Ltd bietet ein figForth für den IBM-PC; für 35$.

Micromotion's MasterForth für Macintosh, IBM-PC, AplleII-serien, CP/M und - man höre und staune, in USA eine Seltenheit - Commodore 64; 100$ bis 125$ für das Standard-83-Paket.

Next Generation Systems' NGS Forth bietet für IBM-PC ein Forth-79-Standard mit allerlei Extras an; ohne Preisangabe.

SOTA hat Forth-79 und figForth für IBM-PC, TRS-80, CP/M für $90 zu bieten.

Talbot Microsystems bietet Forth-83-Standard für 6809 Systeme unter dem FLEX Disk System, für OS9/6809, und für 680x0 wie Macintosh und CP/M-68K; kostet 150$.

UBZ Software hat das UBZ-Forth für den Amiga fertig, "forth-83 compatible, 32-bit stack, multi tasking, separate headers,

(sorry, der Sätze)

full screen editor, assembler, Amiga DOS support, ROM kernel support, graphics and sound support, complete documentation, assembler source code included, monthly newsletter" und kostet 85$.

MMS Forth bietet 'seinen Standard' für 180$ an und verkauft als Module dazu Datahandler, Forthwrite-wordprocessor, forth-comunications, general ledger, games, expert-2, graphics als zusäzliche Module für je 50$ bis 100$. Das ganze klingt wie eine MVP-Version des 79-Standart.

*

Bei der Hardware hat die HITech Equipment Corporation die ForthCard mit Part #STD65F11-05 und Download Software für den IBM-PC ab $199 anzubieten.

Software Composers bietet das SC-1000 Delta Board mit dem super schnellen Novix NC4000P Forthchip für $895 an.

* * *

AUS DEM VEREIN

Über die Forth Gesellschaft eV

H.G. Lynsche, Hamburg, BRD

Anfang 1984 wurde die Forth Gesellschaft in Norddeutschland gegründet. Die 24 Gründungsmitglieder bekamen im selben Jahr noch 100 weitere Forthler hinzu. Am Ende des Jahres 1984 wurde aus der Forth Gesellschaft (BGB) dann die Forth Gesellschaft eV. Wiederum ein Jahr später wurde dem Verein die Gemeinnützigkeit zuerkannt. Zum Jahresende 1985 hatte die Forth Gesellschaft 200 Mitglieder in ganz Europa.

Das Hauptziel der Forth Gesellschaft ist es,
die Verbreitung der Programmiersprache Forth zu fördern und auch die Sprache selbst weiterzuentwickeln, wobei die FG sich lediglich als Instrument zur Förderung von solchen Aktivitäten versteht. Dies geschieht hauptsächlich, indem die Forth Gesellschaft ein Forum für die aktiv Interessierten bildet und einen Kanal zwischen ihnen schafft. Die Plattformen hierfür bilden das Forth Magazin VIERTE DIMENSION, die euroFORML-Konferenz, welche erstmalig 1985 veranstaltet wurde und auf der sich international anerkannte Experten ebenso wie angehende Forth-Programmierer zum freien Gedankenaustausch treffen und schließlich die Mitgliedertreffen der Forth Gesellschaft eV und Ihrer lokalen Gruppen.

Die Forth Gesellschaft hat Ihren Sitz in Hamburg. Von hier aus werden alle bundesweiten Aktivitäten koordiniert, die Vierte Dimension versendet und Anfragen bearbeitet. Hier trifft sich auch das Direktorium der Forth Gesellschaft, um die auf der jährlichen Mitgliedeversammlung getroffenen Beschlüsse durchzuführen und über sonstige aktuelle Ereignisse zu beraten. Die 'wahre Arbeit' der Forth Gesellschaft findet jedoch in den lokalen Gruppen bzw. den Fach-Gruppen statt. Diese wählen einen Gruppenkoordinator, der die Aktivitäten verantwortlich leitet und koordiniert. Gleichzeitig ist dieser der Vertreter für ein weiteres Organ der Forth Gesellschaft, den Krisenrat. Dieser wird vom Direktorium angerufen, wenn es nicht zu einer einstimmigen Entscheidung kommt.

Oft werden wir gefragt, ob die Forth Gesellschaft Listings vekauft oder selber Forth vetreibt. Dem ist nicht so. Das machen die lokalen Gruppen in eigener Regie. Sie finden die Anschriften dieser Gruppen in jeder VIERTEN DIMENSION, außerdem Anzeigen (VD:

Hoffentlich ab Juni) und Produktbesprechungen von Forth-Anbietern. Sowohl an die Gruppen, als auch an die Anbieter können Sie sich mit Ihren konkreten Fragen wenden. Insbesondere die Probleme von Neulingen des FORTH lassen sich in persönlichen Gesprächen und durch Ausprobieren von FORTH zusammen mit anderen, also in der Gruppe, oft schnell lösen.

* * *

Aus dem Direktorium

Klaus Schleisiek, Hamburg

Ich glaube, man muß erst einmal feststellen, daß sich die Forthgesellschaft in einer Kriese befindet; dies wohl zum größten Teil deshalb, weil H.G.Lynsche sich wegen einer beruflichen Umorientierung nicht mehr um die Forth Gesellschaft so intensiv kümmern konnte, wie das in der Aufbauphase der Fall gewesen war. Resultat: Auf einmal ist die Gesellschaft nicht mehr am Telefon ereichbar. Die Vierte Dimension erscheint, statt sporadisch, gar nicht mehr bis selten. Die Beantwortung von Briefen wird noch schleppender als sie eh schon immer war. Kurz, man fragt sich zu Recht, warum man denn Mitglied in solch einem toten Verein bleiben soll.

Mich hat das gelehrt, daß das Funktionieren der Gesellschaft nicht von der freiwilligen Arbeit einer einzigen Person abhängig sein darf; statt dessen müssen die vorhandenen Aufgaben auf mehrere Schultern verteilt werden.

Der Bewegungsspielraum ist aber auch nicht allzu groß: So sind die Einnahmen aus den Mitgliederbeiträgen nicht ausreichend, eine hauptamtliche Stelle für die Vereinsorganisation einzurichten; die Forth Gesellschaft ist also auf ehrenamtliches Engagement angewiesen. Leider ist dieses Engagement aber entweder nicht weit verbreitet, oder (hoffentlich) bisher lediglich unzulänglich koordiniert.

Grundsätzlich waren wir bei der Gründung der Gesellschaft davon ausgegangen, daß die stärksten Impulse von den lokalen Gruppen ausgehen sollten, die sich - bitte schön - massenhaft zusammenfinden sollten. Auch in dieser Beziehung ist Stagnation zu vermelden; außer den Gruppen in Hamburg, Wuppertal, Paderborn, und Karlsruhe scheint sich nicht viel neues zu tun - erfreulicherweise sind diese Gruppen aber nach wie vor aktiv; insbesondere die Wuppertaler Region macht immer wieder durch ihren eigenen Rundbrief auf sich aufmerksam.

Ich glaube jedoch, daß es langsam wieder bergauf geht. Seitdem Michael Kalus die Redaktion der Vierten Dimension übernommen hat, erscheint diese auch wieder. Ab Mitte April wird der 'forthTREE' (VD: = 'CommuniTree' --- Das Kind hat noch keinen guten Namen...) den 'Probebetrieb' aufnehmen; eine Mehlbox, die bei Marco Pauck unter Tel# 040-3904204 erreichbar sein wird; dortselbst wird ein südostasiatischer PC Nachbau mit 20MByte-Disk drauf lauern, von 300baud Modems angepiept zu werden. Dies wird hoffentlich die Kommunikation innerhalb der Gesellschaft entscheidend verbessern - aber natürlich nur für diejenigen, die neben einem Komputer auch ein Modem und ein Kommunikationsprogramm haben. Für die restliche Menschheit planen wir, wöchentlich an 2-3 Stunden dieses Telefon von einem kompetenten 'home forthius' beantworten zu lassen (VD: Uns fällt ein Stein vom Herzen. --- Echte regelmäßige Erreichbarkeit und Support dürfte die Gesellschaft hoch bringen. Gepriesen sei die

Einsicht der Präsidenten!)

Was die Verbreitung von Forth - und damit eine Vergrößerung der Mitgliedschaften in der Gesellschaft - angeht, so hoffe ich darauf, daß im Laufe dieses Jahres VolksFORTH einschlagen wird. Es ist inzwischen auf dem C64 und dem Atari 520ST implementiert und wird von allen, die Erfahrung mit anderen Systemen haben, gelobt. In den nächsten Monaten dürften auch die ersten Besprechungen von VolksFORTH in den Kioskzeitschriften auftauchen und hoffentlich zu einem run auf das System führen.
Hummel Hummel, Forth Forth.

(Vierte Dimension: Drei Präsidenten hat die FG. Es amtieren 1986: Lynsche, Schleisiek, Storjohan; alle Hamburg)

* * *

FORTHQUELLEN

Das Volksforth Softwarepaket

B. Pennemann, Hamburg

Die Arbeitsgruppe Volksforth in Hamburg hatte es sich zum Ziel gesetzt, eine Lehrversion des Forth-83 für die Public Domain zu schreiben. Wir haben das "Gläserne Forth" fertiggestellt. Volksforth wird mit allen Quelltexten ausgeliefert. Dabei ist das System, wie wir finden, leistungsfähig und zukunftssicher ausgefallen. Mit diesem "Lehrstück", das ein komplettes System mit vielen Utilities darstellt, kann sich jeder Computerbesitzer in die Lage versetzen, nicht nur Spielzeugcompiler eintippen zu müssen, sondern vollständige Entwicklungssysteme zu verstehen, zu erweitern und selbst zu schreiben. Das VOLKSFORTH SOFTWAREPAKET ist zur Weitergabe und zum Kopieren gedacht. Wir möchten dazu ausdrücklich ermuntern.

Es handelt sich um ein vollständiges Forthsystem gemäß dem Standard von 1983. Daher werden Werte auf dem Stack in 16 Bit dargestellt. Wir haben dazu etliches an guten Ideen von verschiedenen Forthprogrammierern hinzugegeben. Als Beispiel seien die DEFER Worte, die Funktion VIEW und die Kontrollstrukturen erwähnt. Aber wir haben auch bisher unbekannte und sehr leistungsfähige Konstruktionen mit eingebaut. So ist die Umschaltung von Ein- und Ausgabe auf Drucker etc. sehr viel leichter als in anderen Forthsystemen. Eine andere fortschrittliche Lösung wurde für die Erzeugung namenloser Worte gefunden. Namen von Worten, die nur während der Compilation benötigt werden, können einfach markiert und dann später ohne Seiteneffekte gelöscht werden. Diese Eigenschaft ist eine Voraussetzung zur Modularisierung von Forth. Wir wissen, daß das System nicht besonders schnell arbeitet. Wir haben es aber absichtlich leicht verständlich gehalten und auch zeitkritische Passagen deswegen nicht mit List und Tücke optimiert. Immerhin compiliert das Volksforth auf dem C64 trotzdem noch doppelt so schnell wie bekannte figForth-Versionen. Die Ausführungszeiten sind etwa gleich. figForth läßt nur etwa 100 Werte auf dem Stack zu, das ultraFORTH83 (so haben wir die C64-Version unseres Volksforth genannt) jedoch sehr viel mehr.

Das System ist mit einem Multitasker ausgerüstet, der es erlaubt, im Hintergrund weitere Prozesse ausführen zu lassen. Allerdings dürfen diese Prozesse nicht den Interpreter benutzen, d.h. sie dürfen nicht compilieren. Sie können aber z.B. Zeichen

zum Drucker oder Bildschirm senden und auf den Massenspeicher zugreifen. Das System enthält sogenannte Semaphore, die Probleme verhindern, wenn zwei Prozesse gleichzeitig auf den Massenspeicher oder Drucker zugreifen wollen. Natürlich gibt es verschiedene Worte zur Prozeßkontrolle.

Zum System gehört natürlich ein Assembler, der auch echte Label mit beliebigen Namen, die keinen Speicherplatz benötigen, kennt. Der 6502-Assembler kennt alle illegalen Kombinationen von Befehlen und Adressierungsarten. Der 68000-Assembler erkennt dies jedoch nicht. Der Assembler kann auch "transient" geladen werden, d.h. er wird nach Benutzung entfernt und benötigt dann keinen Speicher mehr.

Beim C64 gibt es eine Fülle von Worten für die Farbwahl, das Zeichnen von Linien, Verändern und Bewegen von Sprites sowie eine Turtlegraphik, die wesentlich schneller als Logo läuft. Beim Atari 520 ST wird GEM unterstützt, d.h. Geraden, Kreise und Fenster sind kein Problem. Wir haben Demoprogramme dazugegeben, an denen die Benutzung der Graphik studiert werden kann.

Der Editor ist ein wirkliches Schmuckstück. Man kann mit ihm ganz einfach Zeilen und Zeichen löschen, wieder hervorzaubern, kopieren, Worte suchen und ersetzen und Mengen von Screens durch die Gegend schaufeln. Beim C64 wurden wegen des anderen Bildschirmes Screens in 25 Zeilen mit 40 Zeichen, statt wie üblich in 16 Zeilen mit 64 Zeichen, dargestellt.

An Hilfsmitteln wird eine Menge geboten. Der zum System gehörende Single-Step-Tracer ist sehr leistungsfähig. Man kann während des Tracens den Stack oder Speicher ändern, in andere Worte umsteigen, Screens listen, ja sogar compilieren. Wir haben uns beim Tracer wirklich Mühe gegeben, weil wir der Ansicht sind, daß der Tracer das wichtigste Werkzeug des Forthprogrammierers ist. Außerdem gibt es zwei Decompiler, Worte zum Ändern der Speicherverteilung und eine Menge an "Kleinzeugs", die man als Programmierer so benötigt. Hierzu gehören Programme zum ansteuern verschiedener Drucker und zum auslisten von Programmen mit und ohne Shadow-Screens.

Und natürlich ist ein umfangreiches Handbuch dabei. Die Struktur und die Benutzung des Systems, der Utilities, des Editors, des Assemblers, der Graphik, des Multitaskers werden erklärt. Im Handbuch finden Sie das gegliederte Glossar der Worte des Systems. Eine Definition der verwendeten Begriffe und eine Liste mit den Abweichungen der Beispiele aus dem Buch "Programmieren in Forth". Das Handbuch ist in deutscher Sprache verfaßt und ist 200 Seiten dick. Das System ist kompakt aber vollständig beschrieben. Ein Forthlehrbuch ersetzt das Handbuch natürlich nicht.

Die Forthgruppe arbeitet weiter. Ein Floating-Point-Paket wird realisiert, der Tracer wird intelligenter gemacht und eine V24-Schnittstelle wird eingebunden. Unter Verwendung des Multitaskers werden sich dann relativ einfach sehr gute Terminalprogramme schreiben lassen. Die Mailboxer können sich schon mal die Hände reiben. Das VOLKSFORTH wird auch für weitere Rechner angepaßt werden. Die nächsten Versionen nach dem C64 und dem ATARI-520ST werden für's MSDOS und CP/M sein. Wir wollen das Volksforth zumindest im deutschsprachigen Europa weit verbreiten. Dazu möchten wir mit den lokalen Gruppen und fig-Chaptern kommunizieren, Kritik, Ideen und Programmteile austauschen. Die bisherigen Kontakte haben sich gut angelassen. Unsere Erfahrungen damit sind recht positiv. Auf eine gute Zusammenarbeit!

* * *

Stand des Volksforth

Die Entwicklung des ultraFORTH83 (das ist das Volksforth auf dem C64) und des Volksforth für den Atari ST 520 ist beendet.
Unsere Preisgestaltung war etwas verworren, aber nun ist definitiv klar:

Das ultraFORTH83 für den C64 mit Handbuch kostet 45,- DM.
Das Volksforth für den Atari kostet 65,- DM.

Wer Mitglied der Forth Gesellschaft werden und das ultraFORTH83 kaufen will, bezahlt 95,- DM.
Mitgliedschaft und Volksforth für den Atari kosten zusammen 115,- DM.

Überweist das Geld an folgende Adresse :

Bernd Pennemann, Sonderkonto U
Postgiroamt Hamburg BLZ 200 100 20 , Kto. 317 87 - 204

Adresse nicht vergessen !

Um es noch einmal zu wiederholen: Das ultraFORTH83/Volksforth ist Public Domain und komplett mit allen Quelltexten (Assembler, Screeneditor, Multitasker, Tracer, Printeransteuerung, Turtle- (nur C64) und Linegraphic sowie Utilities) versehen.

* * *

Ein universelles Stackwort

Doneil Hoekman, Santa Clara, California

Normalerweise ist guter Forth Code charakterisiert durch einen unkomplizierten Parameter Stack. Manchmal jedoch wird er unübersichtlich. Das kommt vor bei Arithmetik in gemischtem Modus, Floating Point, vierfacher Genauigkeit oder Graphikroutinen. Hier wird eine einzelne Funktion mit dem Namen STACK vorgestellt, die jedes andere Wort, das den Stack manipuliert, oder jede hintereinander liegende Sequenz solcher Worte ersetzen kann. Es erlaubt die unmöglichsten Dinge auf dem Stack in einem Schlag zu erledigen, ohne sich Sorgen darüber machen zu müssen, ob die Dinge auch wirklich da landen, wo man sie haben will. Dieses Kunststück gelingt, indem STACK eine Information über das Aussehen des Stacks vorher/nachher erhält.

Hier einige Beispiele, wie man STACK benutzen könnte, um einige gängige Stack Manipulatoren zu implementiere. Die Zeichen links vom Strich zeigen den Parameter Stack vor der Ausführung von STACK, und die Zeichen rechts vom Strich zeigen ihn danach.

```
: DROP    STACK A/ ;
: DUP     STACK A/AA ;
: SWAP    STACK AB/BA ;
: ROT     STACK ABC/BCA ;
: 2SWAP   STACK ABCD/CDAB ;
```

Die hier vorgestellte Implementation von STACK muß einige Regeln beachten:

1. Wenn der vorliegende Stack n Items enthält, muß sein Abbild ebenfalls n aufeinander folgende ASCII-Zeichen enthalten. Dies begrenzt den Stack normalerweise auf 26 Items.
2. Das zu erzeugende Stackbild darf nur solche Zeichen enthalten, die im Abbild des vorliegenden Stack vorkommen. Der erzeugte Stack ist begrenzt auf 127 Zeichen.

In Scr#1 finden Sie zwei Versionen für den Laufzeit-Code und Scr#2 zeigt das Compilezeit-Verhalten von STACK. Die meisten Applikationen von STACK werden wohl die Implementation in Assembler benötigen. Zum Experimentieren wurde die Implementation in Forth ebnfals angegeben. Die Assembler Version läuft ca. zehnmal schneller, zumindest auf meinem 8088-System und compiliert etwa 200 Bytes Code, während die Forth-83 Version ca. 240 Bytes benötigt. Der Assembler Code für (STACK) sollte auf den meisten 8088/8086-Systemen laufen und der Forth Code für (STACK) sollte auf den meisten Forth-83-Standard Versionen gehen.

Zur Compilezeit legt STACK als erstes die Laufzeit-Adresse von (STACK) ab. Es zählt sodann die Anzahl der Items 'vorher' und 'nachher' und compiliert für jedes ein einzelnes Byte. STACK compiliert sodann ein Byte für jedes Zeichen des zu erzeugenden Stackbildes. Zur Laufzeit bewegt (STACK) die Items des Parameter Stack zur Zwischenspeicherng nach HERE. (STACK) läuft sodann durch das Abbild des zu erzeugenden Stacks, ruft die Items aus dem Buffer bei HERE auf, und schiebt sie zurück auf den Stack. Um den Laufzeit-Code zu vereinfachen, werden die Items in einem speziellen Format abgelegt. Wenn n Items auf dem Stack vorliegen, wird Item m als (n-m)*2 abgelegt. Dieser Wert stellt die Offset-Adresse von HERE aus dar. Diese wird benötigt, um das nächste Item für den neuen Stack zu finden. In der compilierten Form nimmt STACK immer vier Bytes für sich und je ein Byte für jedes Item des zu erzeugenden Stacks in Anspruch.

Scr#3 zeigt einige Beispiele, wie STACK verwendet werden kann im Gegensatz zum sonst üblichen Weg. Die BOX-Worte zeichnen ein Rechteck, wenn die Koordinaten der oberen linken und unteren rechten Ecke gegeben sind. STAR-Worte zeichnen einen Stern, wenn sie fünf Werte erhalten. In diesen Beispielen benötigt die STACK-Implementation weniger Objekt Code, weniger Quell Code und läuft ein bißchen schneller beim Test in einer Schleife. Und es ging sehr viel schneller die Beispiele mit Hilfe des Wortes STACK zu erzeugen.

Man sollte STACK bevorzugt dort einsetzen, wo der Umgang mit komplizierten Stack Manipulationen vereinfacht werden muß. Für vergleichsweise einfache Stackmanipulationen hat das Wort STACK Nachteile in der Laufzeit.

> "However, when a programmer feels he has painted himself into a corner, STACK may be a good way to spell relief".

```
Screen # 1
  0 ( run-time stack words                                        05/06/85 )
  1
  2 : (STACK) ( Run-time STACK in forth-83 )
  3           R@ C@ 0
  4     DO    I 2*  HERE + !  LOOP    R@ 1+ DUP C@ + 1+ R@ 2+
  5     ?DO   I C@  HERE + @  LOOP    R> 1+ COUNT  + >R  ;
  6
  7 CODE (STACK) ( Run-time STACK in 8086 assembler )
  8           CL, [SI]     MOV        CH, CH    XOR
  9           DI, DP       MOV        BX, DI    MOV
 10    1$:    AX           POP        WORD      STOS      1$  LOOP
 11           SI           INC        CL, [SI] MOV
 12           AH, AH       XOR        SI        INC       3$  JCXZ
 13    2$:    BYTE         LODS       DI, AX    MOV
 14           DX, [BX+DI] MOV         DX        PUSH      2$  LOOP
 15    3$:    NEXT

Screen # 2
  0 ( char>   stack                                               05/06/85 )
  1
  2 : CHAR> ( -- <n> (Get next ascii character from input stream) )
  3      >IN @ 1 >IN +! BLK @ ?DUP IF BLOCK ELSE TIB THEN + C@ ;
  4
  5 : STACK ( -- abcd:abcd (Perform stack rearrangement) )
  6           ?COMP COMPILE (STACK)          \ compile run-time word
  7    BEGIN  CHAR> BL <> UNTIL              \ find stack picture
  8           0                              \ counter for #input
  9    BEGIN  1+ CHAR> ASCII : = UNTIL       \ compile #input items
 10           DUP >R C,   >IN @   -1         \ remember where we are
 11    BEGIN  1+ CHAR> BL = UNTIL  C,        \ compile #output items
 12           >IN !                          \ back to output items
 13    BEGIN CHAR> DUP BL <>      WHILE      \ while valid output
 14           64 - R@ SWAP - 2* C,  REPEAT   \ compile them
 15           R> 2DROP ; IMMEDIATE           \ clr stacks

Screen # 3
  0 ( box1 box2   star1  star2                                    05/06/85 )
  1
  2 : BOX1 ( x1 y1 x2 y2 -- draw box ) ( 43 bytes : 20.9 seconds )
  3         2OVER 3 PICK OVER 2DUP 2ROT LINE 2OVER LINE
  4         3 PICK OVER 2DUP 2ROT LINE LINE ;
  5
  6 : BOX2 ( x1 y1 x2 y2 -- draw box ) ( 35 bytes : 20.1 seconds )
  7         STACK ABCD:ABCBCBCDADCDABAD   4 LINES ;
  8
  9 : STAR1 ( pt1...pt5 -- draw star ) ( 88 bytes : 43.6 seconds )
 10       9 PICK 9 PICK 7 PICK 7 PICK 2DUP 2ROT LINE 2OVER LINE
 11       7 PICK 7 PICK 2DUP 2ROT LINE 2OVER LINE 7 PICK 7 PICK
 12       LINE 2DROP 2DROP 2DROP ;
 13
 14 : STAR2 ( pt1...pt5 -- draw star ) ( 42 bytes : 41.8 seconds )
 15       STACK ABCDEFGHIJ:ABEFEFIJIJCDCDGHGHAB   5 LINES ;
```

Den Compiler steuern

Michael Kalus, Wuppertal

Das folgende Beispiel soll dazu anregen, tiefer in die inneren Strukturen des Forth einzudringen. Es zeigt, wie man den Compiler selbst kontrollieren kann. Als Vorlage diente das Forth-83-Standard Model 'F83' von M.Perry und H.Laxen (1984). Es läuft bei mir auf einem AppleII Compatiblen Rechner mit Z80-Karte unter CP/M Version2.2.

Will man den eigenen Drucker einstellen, muß man die entsprechenden Control Codes an den Drucker abschicken. In den Handbüchern wird die Code-Sequenz detailiert beschrieben und in einem hübschen Programmierbeispiel in BASIC vorgeführt. Das liest sich dann etwa so:

```
LPRINTCHR$(27);"*";CHR$(6);CHR$(§#Grrmpf1!!)
```

In Forth hat man es da leichter. Man legt die Werte hintereinander auf den Stack und sagt sooft (PRINT) wie der Vorrat an Items es verlangt - das ganze dann etwas automatischer und in der angegebenen Reihenfolge natürlich. Jedes (PRINT) überträgt ein Byte an den Drucker. (PRINT) steckt zusammen mit (KEY) hinter dem Wort KEY. (PRINT) wird ausgeführt, z.B. wenn der Drucker der Terminalausgabe parallel geschaltet worden ist - unter CP/M also mittels Contol-P.

Noch aus guter alter figForth-Tradition gibt es die User-Variable "check-stack-pointer" CSP und einige fertige Worte dazu. CSP dient als Zeiger auf ein bestimmtes Stack-Item. Der stack pointer SP hingegen zeigt auf das TOP-Item (im F83 jedenfalls). Rettet man nun SP nach CSP indem man !CSP ausführt, legt danach die Control Codes in normaler Folge auf den Stack und vergleicht SP mit CSP, so läßt sich die Anzahl neu hinzugekommener Items feststellen. Der Rechner 'weiß' somit selbst, wieviele Bytes er senden muß. Bei meinem F83 wächst der Stack im Adressraum abwärts und die sendende Schleife in PR! läuft daher in Zwei-Byte-Schritten rückwärts. Der Schleifenindex I wird zur Adresse des Items. Werden Bytes in dieser Weise gesendet, bleibt der Stack Pointer SP unverändert und muß zurückgesetzt werden, damit der Stack hinterher wieder leer erscheint. Der alte Stack Pointer aus CSP wird nach SP zurückgeschrieben. Die Komandozeile soll dann so aussehen, wie hier am Beispiel "Proportionalschrift-EIN" gezeigt wird:

```
!CSP  ESC p  1  PR!
```

Wie wird das realisiert? ESC interpretiert das nachfolgende Zeichen und legt dessen Wert auf den Stack. ESC ist IMMEDIATE. Darauf will ich näher eingehen. Sehen Sie sich zuvor den Code in Scr#1 an. Die Definition von ESC hat zwei logische Teile. Im ersten Teil werden Stackitems erzeugt. Einmal wird einfach die 27 auf den Stack gelegt. Zum anderen wird durch BL WORD 1+ C@ das nächste Zeichen im input stream isoliert und sein Ascii-Wert auf dem Stack hinterlassen. Würde ESC nur im Interpreter verwendet, wäre die Arbeit damit getan.

Aber es soll auch möglich sein, das gleiche Verhalten durch Compilieren zu erzeugen. Beim Compilierlauf würde nun aber WORD lediglich im Diktionary eingetragen und erst zur Laufzeit ausgeführt. Doch soll schon jetzt beim compilieren von WORD das auf ESC folgnde Zeichen der Quelle interpretiert werden. Deswegen muß ESC, statt compiliert zu werden, unmittelbar (engl.

immediate) den Compiliermodus verlassen. Es interpretiert dann als Input Stream die Quelle, findet das gewünschte Zeichen und hinterläßt dessen Ascii-Wert auf dem Stack.

Diese Werte sollen aber nun ihrerseits compiliert werden und dann später zur Laufzeit wieder auf dem Stack erscheinen. Das kann man mit dem Wort LITERAL bewerkstelligen. LITERAL soll nun aber erst zur Laufzeit von ESC wirken, es muß also in ESC hinein compiliert werden, obwohl es ein IMMEDIATE Wort ist. Das macht man mit [COMPILE] <Name>. Damit wird das nächste Wort des Input Streams ins Dictionary compiliert. [COMPILE] ist natürlich selbst wieder IMMEDIATE. In ESC entscheidet schließlich der STATUS, ob compiliert oder interpretiert werden muß.

Was meinen Sie mit: "Ich glaube, Basic ist gar nicht so schlecht..." ??! Als Designvorlage für ESC diente übrigens das Wort ASCII. Und wenn man's erst mal kann, eröffnen sich ungeahnte Möglichkeiten.

```
Scr # 1          B:PR.BLK
  0 \ sende control codes an ritemanF+                          14.mar86mk
  1 : PR!  ( n...m -- )
  2   SP@ CSP @ 2-
  3   DO I @ (PRINT) -2 +LOOP
  4   CSP @ SP! ;
  5 : ESC   ( C  ( -- 27 c )
  6    27 BL WORD 1+ C@
  7    STATE @ IF SWAP [COMPILE] LITERAL [COMPILE] LITERAL THEN ;
  8   IMMEDIATE
  9
  A EXIT
  B
  C \ Extensible Layer              Defining Words
  D : !CSP   (S -- )   SP@ CSP !   ;
  E : ?CSP   (S -- )   SP@ CSP @ <> ABORT" Stack Changed"   ;
  F

Scr # 2          B:PR.BLK
  0 \ Angefuegt zum besseren Verstaendniss                      14mar86mk
  1 \ 16 bit Stack Operations
  2 CODE SP@     (S -- n )
  3      0 H LXI   SP DAD   HPUSH JMP   END-CODE
  4 CODE SP!     (S n -- )
  5      H POP    SPHL   NEXT   END-CODE
  6 CODE RP@     (S -- addr )
  7      RP LHLD   HPUSH JMP   END-CODE
  8 CODE RP!     (S n -- )
  9      H POP   RP SHLD   NEXT   END-CODE
  A
  B
  C
  D
  E
  F
```

```
Scr # 3          B:PR.BLK
  0 \ Angefuegt zum besseren Verstaendniss                    14mar86mk
  1 \ System VARIABLEs
  2 DEFER     EMIT          ( TO ALLOW PRINT SPOOLING )
  3 VARIABLE  SCR         ( SCREEN LAST LISTED OR EDITED )
  4 VARIABLE  PRIOR       ( USED FOR DICTIONARY SEARCHES )
  5 VARIABLE  STATE       ( COMPILATION OR INTERPRETATION )
  6 VARIABLE  WARNING     ( GIVE USER DUPLICATE WARNINGS IF ON )
  7 VARIABLE  DPL         ( NUMERIC INPUT PUNCTUATION )
  8 VARIABLE  R#          ( EDITING CURSOR POSITION )
  9 VARIABLE  LAST        ( POINTS TO NFA OF LATEST DEFINITION )
  A VARIABLE  CSP         ( HOLDS STACK POINTER FOR ERROR CHECKING )
  B VARIABLE  CURRENT     ( VOCABULARY WHICH GETS DEFINITIONS )
  C 8 CONSTANT #VOCS      ( THE NUMBER OF VOCABULARIES TO SEARCH )
  D VARIABLE  CONTEXT     ( VOCABULARY SEARCHED FIRST )
  E
  F
```

* * *

Status Anzeigen in Forth

Bernd Pennemann, Hamburg

Hier stelle ich eine Datenstruktur vor, die Status-Meldungen auf den Bildschirm ausgeben und wieder löschen kann. Das Wort STATUS schreibt in diesem Beispiel eine kurze einzeilige Meldung an eine vorher definierte Stelle des Bildschirms oder löscht diese Meldung, je nachdem ob das Flag, das von STATUS verbraucht wird, TRUE oder FALSE war. Die Statusanzeige erfolgt mit fixer Position unabhängig vom Arbeits-Cursor. Um die Länge der Meldung braucht man sich nicht kümmern, wenn man sie löschen will.

Die Definition von STATUS benutzt zwei Primitives. Das eine Primitive ist AT. Dies positioniert den Cursor in die Zeile ROW und in die Spalte COL. Das andere Primitive ist AT? und ermittelt die aktuelle Position des Cursors, indem es die Zeilen- und Spaltennummern auf den Stack legt. Die Funktion der Worte ist in dem Glossar in Bild 1 beschrieben. Die Definitionen von AT und AT? sind übrigens dem ultraforth83 Pennemann entnommen. Gegenüber dem F83 von H.Laxen und M.Perry sind im Ultraforth83 leider ROW und COL vertauscht. Ich bitte dies zu beachten. Ich finde diese Definition recht praktisch. Wer will, kann das Wort ," in STATUS vor DOES> einbauen. Man darf dann aber nur ein Leerzeichen zwischen dem Namen und dem Text lassen.

```
Scr # 0          STATUS.BLK
  0 \ Status Anzeigen                                        14.mar86mk
  1
  2 : STATUS    ( row col -- )
  3    CREATE , ,
  4    DOES> AT? >R >R
  5      DUP 2+ @ OVER @ AT
  6      4+ COUNT ROT
  7      IF TYPE ELSE SPACES DROP THEN
  8      R> R> AT ;
  9 : ,"    ( -- )
  A    ASCII " WORD C@ 1+ ALLOT ;
  B
  C
  D
  E
  F

Scr # 1          STATUS.BLK
  0                                                          14.mar86mk
  1
  2 STATUS  ist ein definierendes Wort, benutzt in der Form:
  3 row col STATUS <name> ," ccc"
  4 STATUS erzeugt einen Kopf fuer <name> und kompiliert die Werte
  5 row und col. Wird <name> spaeter ausgefuehrt, so verbraucht es
  6 ein Flag. Ist Flag wahr, so wird der mit ," kompilierte
  7 String in der Zeile row und der Spalte col ausgedruckt. Ist
  8 das Flag falsch, so wird der String geloescht.
  9
  A ,"  speichert einen counted String im Dictionary ab HERE.
  B Dabei wird die Laenge des Strings in dessem ersten Byte, das
  C nicht zur Laenge hinzugezaehlt wird, vermerkt.
  D
  E
  F

Scr # 2          STATUS.BLK
  0                                                          14.mar86mk
  1
  2 Beispiel:
  3
  4 decimal 23 70 STATUS printer  ," Printer ein"
  5
  6 true PRINTER    ( schreibt "Printer ein" in die Statuszeile)
  7 false PRINTER   ( loescht die Statuszeile )
  8
  9
  A Zur Information:
  B AT     ( row col -- )   positioniert die Schreibstelle eines
  C Ausgabegeraetes in die Zeile row und die Spalte col.
  D AT?    ( -- row col )   ermittelt die Schreibstelle und hinter-
  E laesst die Zeilen- und Spaltennummer auf dem Stack.
  F
```

Stringstack

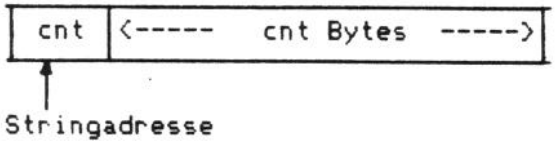
Klaus Schleisiek, Hamburg

Strings sind das Rückgrat der kommerziellen Datenverarbeitung, ja man kann heute sagen, daß sinnvoller Komputereinsatz hauptsächlich darin besteht, Texte von Hüh nach Hott zu bewegen, verdichten, exzerpieren, aufbereiten etcpp. In der Werbung für PC's werden hauptsachlich die Textverarbeitungsaspekte in den Vordergrund gerückt. Daneben ist die ursprüngliche Aufgabe der Komputer - die Berechnung von Balistiktabellen für die Artillerie - sehr in den Hintergrund gerückt.

Die heilige atomare Einheit ist das Zeichen, das in Zeiten einer heraufdämmernden Norm für 'Extended-Ascii' acht Bits zum überleben benötigt.

Die nächst größere Einheit ist der gecountete String; charakterisiert durch eine Adresse, an der man ein bytecount findet. Daher kann ein gecounteter String nur maximal 255 Bytes lang sein - ausreichend, um darin Zeilen eines Textes 'einzupacken'.

Der gecountete String:



Strings haben ein solch eigentümliches Format, daß man am besten einen eigenen Stack nebst den notwendigen Operatoren anlegt, um Strings bequem und flexibel zu manipulieren. An Operatoren benötigt man folgende:

a) Operatoren zur Verwaltung des Stacks, die notwendig sind, um die nachfolgenden Operatoren zu implementieren, jedoch nicht vom Benutzer des Stringstacks benutzt werden sollen.
b) Stringstack Operatoren (DUP etc.)
c) Operatoren, die Strings manipulieren.
d) Operatoren, die es ermöglichen, Strings auf den Stack zu bekommen.
e) String-Datentypen und zugehörige Operatoren (Variable, Konstante).

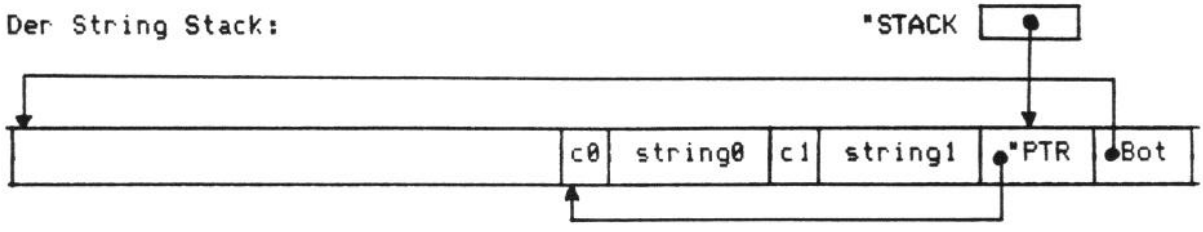
Die nächst höhere Dateneinheit nach dem String ist der Record, der in modernen Datenbasen als Zusammenfassung mehrerer Felder variabler Länge aufgefasst wird und damit selber variable Länge besitzt. Zur Verarbeitung solcher Records ist ein Stringstack unbedingt notwendig, da der Record selber - vorausgesetzt er ist nie länger als 255 Bytes - als String 'verpackt' werden kann.

Zur effizienten Implementation eines Stringstacks ist es notwendig, die 'richtige' Struktur zu wählen, so daß alle notwendigen Operatoren verhältnismäßig 'einfach' realisiert werden können. Es gibt in der Forthliteratur mehrere Ansätze für einen Stringstack, die sich oft dadurch auszeichnen, daß sie so kompliziert sind, daß, nachdem der Code für den Stringstack geladen ist, kaum noch Platz für ein Anwendungsprogramm bleibt. -->

Folgende Bedingungen müssen erfüllt sein:

a) Die Strings auf dem Stringstack müssen genauso aufgebaut sein, wie gecountete strings auch sonst im System dargestellt sind.
b) Der Stringstack wächst von hohen gegen niedrige Speicheradressen.
c) Der Stringstack soll multitaskingfähig sein.

Der Stringstack selber ist eine kompakte Datenstruktur, die im Dictionary allokiert wird, und beinhaltet einen eigenen Stackpointer. Die Stringstackoperatoren beziehen sich jeweils auf den Stringstack, der durch die (User)variable "STACK lokalisiert wird. Damit ist es sowohl möglich, lokale Stringstacks für Hintergrundprozesse anzulegen, als auch mit mehreren Stringstacks 'zugleich' zu jonglieren.



---- aufsteigende Speicheradressen ---->

Es wäre interessant, einmal ein Forthsystem auf den Stringstack aufzubauen. Sämtliche Operationen, die die Manipulation von Strings beinhalten (WORD EXPECT TYPE etc.) liefen in wesentlich geordneteren und flexibleren Bahnen ab. Solche ein Forth wäre sicher sinnvoll, wenn es sich in 32bit Systemen und Gigabyteadreßräumen breitmachen dürfen.

Noch eine Bemerkung zur Wahl der Namen: Wie leicht zu erkennen ist, wurde " als Präfix für die Stringoperatoren benutzt. Schuld daran ist Chuck Moore und sein ." Operator. In der natürlichen Sprache wird der 'Tüttel' (VD: Hamburgisch für 'Anführungszeichen'.) benutzt, um zwei verschiedene Ebenen von Zeichenketten voneinander zu unterscheiden. Ich habe versucht, möglichst eindrucksvolle oder bereits gebräuchliche Namen zu wählen.

(Lexikon: counted String (engl.). Eine abgezählte Reihe von Bytes, welche die Codes von Ascii-Zeichen enthalten. Die Länge der Zeichenkette ist im ersten Byte des gecounteten String abgelegt. Das erste Byte wird bei der Länge des String nicht mitgezählt.)

640

```
0 ( stringstack load screen                          ks 16 mar 86 )
1  Forth definitions  decimal
2  064 Constant C/L                                  Zeilenlaenge eines Screens. ( Characters-per-Line )
3  : \     >in @  C/L /  1+ C/L *  >in ! ; immediate  Kommentar bis zum Ende der Zeile.
4  : under     ( n0 n1 -- n1 n0 n1 )  swap over ;    Oft gebrauchter Stackoperator.
5  : move      ( from to len -- )                    Dieses MOVE stellt sicher, dass ein String sich nicht selber
6    >r  2dup u<  IF  r> cmove> exit  THEN  r> cmove ; ueberschreibt.
7  : place     ( addr len to -- )                    Plaziert einen String an ADDR mit Laenge LEN auf die Adresse
8    over >r  rot  over 1+  r> move  c! ;            TO als gecounteten String.
9
10 : ,"     Ascii " word  count here  over 1+ allot  place ;  Legt den naechsten durch " begrenzten String im Dictionary ab.
11
12 : ("  [compile] ( ; immediate                     Kommentarklammer fuer die Stringstack Kommentare.
13
14 Blk @ 1+  dup 7 +  thru
15
```

641

```
0 \ string stack allocation                          ks 16 mar 86  Die Stringoperatoren beziehen sich auf den Stringstack, auf
1                                                                  den "STACK verweist. Bei Multitaskinganwendung wird "STACK als
2  Variable "stack                                                 Uservariable definiert, so dass jeder Task ein eigener String-
3                                                                  stack zur Verfuegung stehen kann.
4  : "ptr      ( -- addr )  "stack @ ;                             Diese Speicherstelle haelt den Stringstack Pointer.
5
6  : "top      ( -- addr )  "ptr @ ;                               Auf dieser Adresse liegt das Countbyte des obersten Strings.
7
8  : "allot    ( len -- )                                          Allokiert einen Stringstack im Dictionary. Am Ende des Stack-
9    here  swap allot  here "stack !  here ,  , ;                  bereichs wird der Pointer und dahinter die Adresse des unteren
10                                                                 Endes des Stacks abgelegt fuer Ueberlauftest.
11 1000 "allot                                                     Nun steht ein Stringstack von 1000 Bytes zur Verfuegung.
12
13 : "clear                "ptr  dup ! ;                           Leert den Stringstack
14                                                                 ADDR0 weist auf das Countbyte eines Strings; ADDR1 ist dann die
15 : "skip     ( addr0 -- addr1 )      count + ;                   Adresse des ersten Bytes hinter dem String.
```

642

```
0 \ string access primitives                         ks 16 mar 86
1
2  : ?string   ( len -- len )  [ hex ]                             Ein String kann maximal 255 Bytes lang sein. (Hex FF)
3    dup 0FF00 and abort" no string" ;  decimal
4
5  : ?overflow ( addr -- addr )                                    Dieser Adressvergleich ist moeglich, da beim Allokieren
6    dup  "ptr 2+ @  u< abort" overflow" ;                         die Adresse des unteren Stackendes hinter dem Stackpointer
7                                                                  abgelegt wurde.
8  : "push     ( addr len -- )  (" -- s )    ?string              LEN Bytes, die in aufsteigender Reihenfolge ab ADDR stehen,
9    "top  over - 1-  ?overflow  dup "ptr !  place ;               werden auf den Stack gepackt.
10
11 : ?empty    ( addr -- addr )  dup "ptr = abort" empty" ;       Wenn der Stringstack leer ist, dann verweist, nach Konstruktion
12                                                                 der Stackverwaltung, der Stackpointer auf sich selber.
13 : ter       ( +n -- addr )  (" s -- s )  "top                   Liefert die Adresse des "n+1-ten" Strings auf dem Stringstack.
14   BEGIN  ?empty  swap ?dup  WHILE  1- swap  "skip  REPEAT ;     Abbruch mit Fehlermeldung, wenn der n+1-te String nicht
15                                                                 existiert.
```

643

```
0 \ string stack operators                          ks 16 mar 86
1
2  : "count  ( -- addr len )  (" s -- s )  0 ter count ;
3
4  : "length ( -- len )       (" s -- s )  0 ter c@ ;
5
6  : "@      ( addr -- )      (" -- s )    count "push ;
7
8  : "drop                    (" s -- )    "count + "ptr ! ;
9
10 : "pop    ( -- addr len )  (" s -- )    "count "drop ;
11
12 : "pick   ( +n -- )   (" sn ... s0 -- sn ... s0 sn ) ter "@ ;
13
14 : "roll   ( +n -- )   (" sn ... s0 -- sn-1 ... s0 sn )
15    ter  dup "@  "top under -  "pop +  swap cmove) ;
```

Liefert Adresse und Laenge des obersten Strings auf dem Stack.

Nur Laenge des obersten Strings.

Holt einen gecounteten String an ADDR auf den Stack.

Entfernt den obersten String vom Stack und legt seine Adresse und Laenge auf den Datenstack. ** Nachfolgende Stackoperationen koennen diese Speicherstellen veraendern **

Kopiert den n+1ten String nach oben auf den Stack.

Holt den n+1ten String nach oben auf den Stack. Die darunterliegenden Strings werden in den freigewordenen Raum geschoben.

644

```
0 \ string stack operators                          ks 17 mar 86
1
2  : "dup              (" s -- s s )            0 "pick ;
3  : "over             (" s0 s1 -- s0 s1 s0 )   1 "pick ;
4
5  : "-roll  ( +n -- )   (" sn ... s0 -- s0 sn ... s1 )
6     ter "skip dup   "count + under  -  "top swap
7     "dup  cmove   "pop   rot over - 1-  place ;
8
9  : "swap             (" s0 s1 -- s1 s0 )      1 "roll ;
10 : "rot              (" s0 s1 s2 -- s1 s2 s0 ) 2 "roll ;
11
12 : "depth  ( -- +n )   (" sn-1 ... s0 -- sn-1 ... s0 )
13    0  "top  BEGIN  dup "ptr - WHILE "skip  swap 1+ swap
14             REPEAT  drop ;
15
```

Packt den obersten String unter den n+1ten String.
-roll ist Komplementaer zu roll, so dass die Phrase
*** n dup "roll "-roll *** den Stack unveraendert laesst.

Anzahl der Strings auf dem Stack

645

```
0 \ string operators                                ks 17 mar 86
1
2  : "join                      (" s0 s1 -- s2 )
3     "top   dup >r   "pop   dup "length +  r> c!
4     over "ptr !   1+ cmove)  ;
5
6  : "extract   ( +n0 +n1 -- )   (" s0 -- s1 )
7     "pop   rot umin   rot over umin
8     rot over +   -rot   - "push ;
9
10 : "split     ( +n -- )        (" s0 -- s1 s2 )
11    "dup   0 swap   dup "length "extract   "swap "extract ;
12
13 \ : "split     ( +n -- )        (" s0 -- s1 s2 )
14 \    "count  rot over umin   under - >r
15 \    "" "top place   r> 1 ter c! ;
```

Konkateniert die beiden obersten Strings, so dass der oberste String den Anfang des konkatenierten Strings bildet.

Extrahiert einen Teilstring. Der Teilstring beginnt mit dem n0+1-ten Byte und endet mit dem n1-ten Byte.
Ist n0 >= n1, so ergibt sich der leere String. Siehe: ""

Zerschneidet eine String in zwei Teile, so dass der vordere Teil zum obersten String wird; dabei ist das n-te Byte sein Ende.
Der hintere Teil ist das zweite Element auf dem Stack.
Eine generische Version von "SPLIT. Prinzipiell laesst sich "SUB auch aus "SPLIT ableiten. Aber dann sind die Eigenschaften von "SUB in Ueberlaufsituationen weniger sinnvoll.

646

```
0 \ string I/O                                        ks 17 mar 86
1   : ""      (" -- s )   0 0 "push ;
2   : )string ( char -- )       (" -- s )
3     0 1 "push   "count drop c! ;
4
5   : (")     (" -- s )   r) dup "skip )r "@ ;
6
7   : "       ( string" )     (" -- s )
8     State @ IF  compile (") ," exit  THEN
9     Ascii " word "@ ; immediate
10
11  : "expect ( +n -- )     (" -- s )
12    dup "push   "count expect   "pop   drop Span @   "push ;
13
14  : ".      (" s -- )   "pop type ;
15  : ".s     "top  BEGIN  "depth  WHILE  cr ".  REPEAT  "ptr ! ;
```

Der leere String, bestehend aus dem Countbyte mit dem Wert Null.
Konvertiert ein Byte auf dem Datenstack in einen String der Laenge eins.

Laufzeitroutine fuer Kompilierte Inline-Strings.
Implementationsabhaegig. Hier fuer postinkrementierendes System.
Bei der Kompilation wird der naechste durch " begrenzte String im Eingabestrom hinter der Laufzeitroutine ins Dictionary abgelegt. Beim Interpretieren wird der folgende String auf dem Stack abgelegt.
Maximal n Bytes werden von der Tastatur entgegengenommen und als String abgelegt. Setzt die Variable SPAN voraus, die angibt, wieviele Bytes tatsaechlich empfangen wurden.
Zeigt den obersten String an.
Zeigt alle Strings auf dem Stack zerstoerungsfrei an.

647

```
0 \ string data types                                 ks 17 mar 86
1
2   : "constant               (" s -- )
3     Create   "pop here   over 1+ allot   place
4     does)                  (" -- s )   "@ ;
5
6   : "variable ( +n -- )
7     Create   dup c,   0 c,   allot
8     does)    ( -- addr )   1+ ;
9
10  : ?fits     ( addr -- addr )
11    dup 1- c@ "length u< abort" too long" ;
12
13  : "!        ( addr -- )     (" s -- )
14    ?fits  "pop rot place ;
15
```

Legt den obersten String als Konstante ins Dictionary ab.
Wird die Stringkonstante ausgefuehrt, so wird der String auf dem Stack abgelegt.

Legt eine Stringvariable an, indem Speicherplatz fuer Strings bis zur Laenge n allokiert und der leere String abgelegt wird.
Nach Ausfuehrung liegt die Adresse des Countbytes des der Variablen zugewiesenen Strings auf dem Datenstack.
Vor der Stringadresse liegt das Laengenbyte einer Stringvariablen. Abbruch und Fehlermeldung, wenn der String auf dem Stack laenger ist als Platz in der Variablen vorhanden ist.
Legt den obersten String auf dem Stack in einer Stringvariablen ab.

648

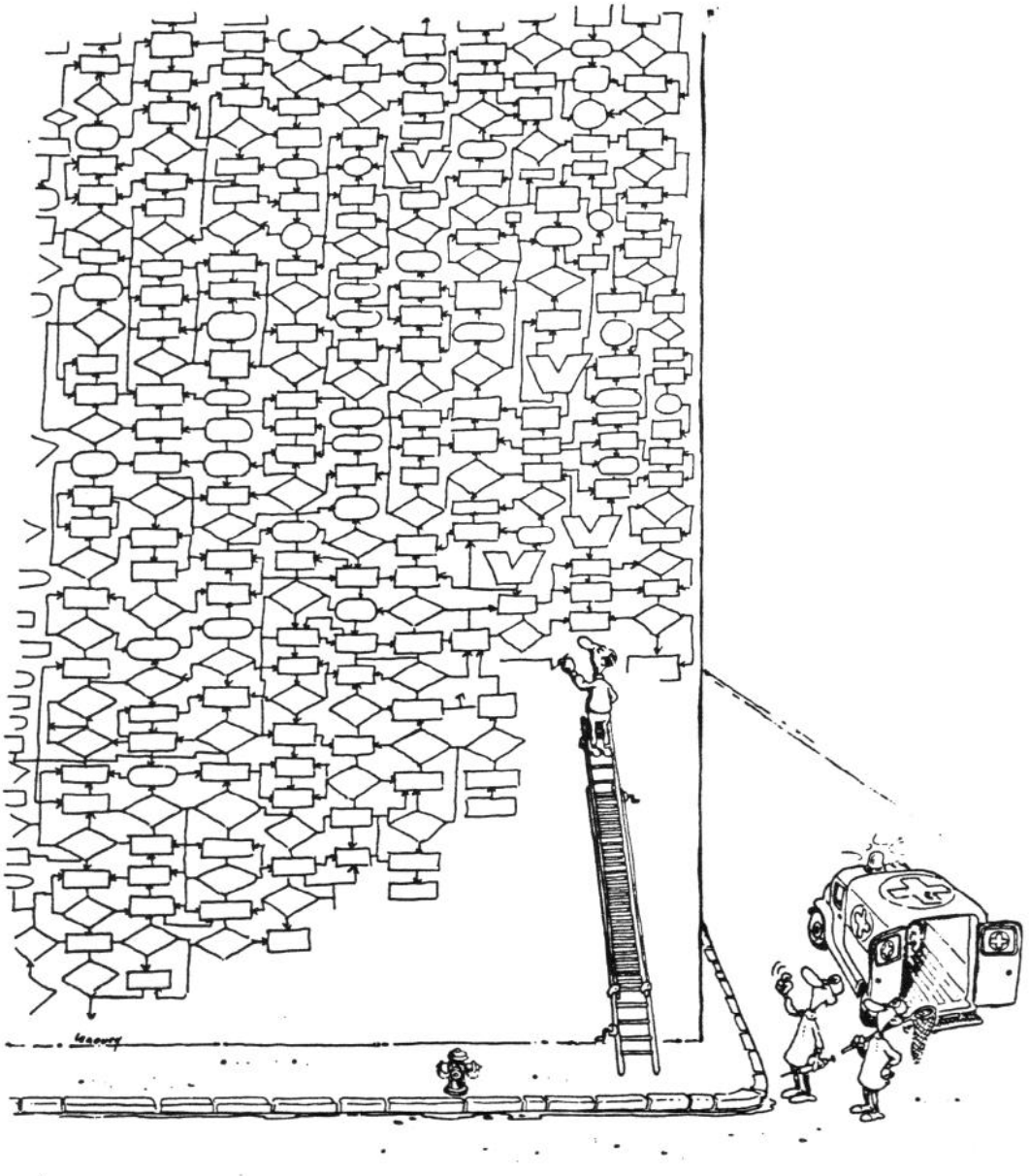
```
0 \ lexikalischer ascii stringvergleich               ks 17 mar 86
1
2   : compare   ( addr0 len0 addr1 len1 -- n )
3     rot   2dup - )r   umin
4     BEGIN  ?dup WHILE  1- )r   count rot   count rot  - ?dup
5            IF  r) r) 2drop  -rot  2drop  negate exit  THEN
6        swap r) REPEAT  2drop   r) ;
7
8   : "compare  ( -- n )    (" s0 s1 -- )
9     "pop "pop compare negate ;
10
11  : "=        ( -- f )    (" s0 s1 -- )     "compare 0= ;
12  : "(        ( -- f )    (" s0 s1 -- )     "compare 0( ;
13  : "(=       ( -- f )    (" s0 s1 -- )
14    "compare   dup 0(  swap 0= or ;
15
```

Lexikalischer Groessenvergleich zweier Strings.
string0) string1 ==> n (0
string0 = string1 ==> n = 0
string0 (string1 ==> n) 0
Sortierfolge ist Ascii, so dass Umlaute falsch sortiert werden.

Die obersten Strings werden miteinander lexikalisch verglichen.

649

```
0 \\ an alternative *expect                          ks 17 mar 86
1   hex 08 Constant #BS     0D Constant #CR   decimal          Funktionscodes der Tastatur.
2
3   : case?  ( n0 n1 -- n0 ff / tf )                          Ist n0 gleich n1, so wird Wahr hinterlassen.
4      over = dup IF  swap drop  THEN ;                        Ist n0 ungleich n1, so wird n0 unter Falsch hinterlassen.
5
6   : *append  ( char -- )    (* s0 -- s1 )  )string *swap *join ; Fuegt CHAR auf dem Datenstack an den obersten String auf dem
7                                                              Stringstack als letztes Byte an.
8   : del   0 *length 1- *extract   #BS emit space #BS emit ;
9
10  : decode   ( key -- flag )   False swap
11     #BS case? IF  *length IF  del  THEN  exit  THEN         Der Backspace key loescht den vorher eingegebenen key.
12     #CR case? IF  cr   not             exit  THEN          Mit (cr) wird die Eingabe des Strings beendet.
13     dup emit  *append ;                                     Alle anderen keys werden an den obersten String des Stacks ange-
14                                                            haengt.
15  : *expect   (* -- s )   ** BEGIN  key decode  UNTIL ;      Eine Folge von Eingaben wird bis zum (cr) als String abgelegt.
```



Rechner sprechen Forth miteinander

Adolf Krüger, Schwelm

Was ist eine Eingabe für den Computer? Eine endlose Programmschleife: Einen Strom von Zeichen empfangen, interpretieren, ausführen und dann von Vorn beginnen. In Forth heißt diese Schleife INTERPRET. Um zu verstehen, wie Computer in Forth miteinander reden können, wäre es gut, wenn Sie sich schon einmal mit diesem Textinterpreter beschäftigt hätten.

Zur Einstimmung in mein eigentliche Thema will ich kurz erzählen, wie es kam, das die Computer es lernen sollten, miteinander zu sprechen. Das Projekt war eine Installation mit mehreren bildschirmlosen Computern. Sie messen den Stromverbrauch in einem großen Gebäudekomplex mit mehreren hundert Verbrauchern, schalten diese Ströme und verteilen die Kosten auf die Abnehmer. Die Muttersprache dieser Computer ist LOLA, die 'Logger Language', eine Forth-Extension, die ich zum bequemen Umgang mit Meßdaten entwickelt habe. Wir wollten von unserer Firma aus die Daten abfragen, die Software pflegen und weiterentwickeln können, ohne dazu jedesmal in die weit entfernte Stadt reisen zu müssen. Die Entwicklung der Teile für mein 'lokales Netz' mit Telefonanschluß, Soft- und Hardware zusammen, hat dann nur etwa fünf Stunden gebraucht, wohlgemerkt von der Idee bis zur Platine und dem Protokoll in Forth.

Die Rechner selbst mit Forth, Applikation und seriellen Schnittstellen waren zu der Zeit schon installiert. Sie sollten alle an nur ein Telefonmodem angeschlossen werden, jedoch ohne teure Multiplexer und sich dennoch nicht gegenseitig stören. Sie sollten alle von einem dummen Terminal aus ohnen Aufwand bedient werden können, also ohne besondere Protokolle für die Zeichenübertragung, und der jeweilige Computer sollte sich dann so benehmen, als sei nur er allein mit dem Terminal verbunden. Weitere oder neue Computer sollten einfach anzuschließen sein. Und sie sollten sogar gegenseitig Daten austauschen können. Das waren die Designvorgaben für die folgende Lösung.

Vorbild war die bekannte Vollduplex-Schaltung zwischen zwei Rechnern. Die Sendeleitung TXD des einen Computers ist an die Empfangsleitung RXD des anderen angeschlossen und umgekehrt. (Abb.1). Wird ein dritter Rechner angeschlossen und soll nun abwechselnd jeder mit jedem reden können, entsteht das Problem, abwechselnd verschiedene Kreuzungen der Leitungen schalten zu müssen (Abb.2 und Abb.3). Der KREUZER zusammen mit dem kleinen Stück Forth machts möglich.

Jeder Computer im lokalen Verbund ist über den KREUZER ageschlossen (Schaltbild-1). Der Kreuzer hat rechnerseitig die Anschlüsse RXD, TXD sowie den Selekt-Eingang SEL für die Auswahl des Betriebs-Zustandes. Die Kreuzer sind untereinander durch die Verdrahtung der Anschlüsse R+ und R- sowie T+ und T- verbunden (Abb.2). R und T sind V24-Stromschleifen mit eingeprägtem Strom aus einer Konstantstromquelle. Ein Strom I<0,3mA bedeutet LOW- , und ein Strom 12mA<I<20mA bedeutet HIGH-Pegel. Zur galvanischen Trennung wurden die Optokoppler-Typen HCPL4100 als Sender und HCPL4200 als Empfänger mit Freigabeeingang verwendet. Die Selectlogik wurde durch eine einfache Inverterstufe und die Dioden von den Freigabeleitungen zu den TXD-Leitungen realisiert. Die Schaltung ist auf einer kleinen Platine untergebracht (1). Das Telefonmodem (2) ist an die Stromschleife durch einen einfachen Umsetzer angeschlossen.

Die Unterhaltung findet in Forth statt. Der eine Rechner schickt dem anderen Forthcode in den Terminal-Input-Buffer TIB.

Der Interpreter arbeitet diesen Inputstream dann ab. Das ist nichts besonderes für Forth - alles wie gewohnt. Änderungen sind nicht nötig, denn dieses Verhalten entsteht sowieso, wenn KEY und EMIT den Input- und Outputstream über die serielle Schnittstelle leiten. Es muß lediglich noch ereicht werden, daß die Rechner sich ihren Gesprächspartner auswählen können. Daten empfangen dürfen alle gleichzeitig, aber sie dürfen nicht alle gleichzeitig senden, da sonst undefinierter 'Datensalat' auf der gemeinsamen Sendeleitung entstehen würde. Daten senden darf pro Leitung nur einer, bei zwei Leitungen im Voll-Duplex-Betrieb also zwei Rechner gleichzeitig; diese beiden 'reden' miteinander.

Ein Beispiel soll dies veranschaulichen und dabei den Forthcode "Netzchen" erklären. Von einem Terminal aus sende ich per Telefon über das Modem ein Selektwort und dann weitere Worte an meine Rechner im lokalen Verbund. Da im Ruhezustand alle auf Empfang geschaltet sind, nehmen alle dieses erste Wort auf. Ich habe SEL#2 in den Verbund geschickt, um den Rechner#2 anzurufen. Nur er 'kennt' dieses Selektwort und verlässt damit seinen Ruhezustand im versiegelten Vocabular LTG und begiebt sich ins FORTH. Alle übrigen Computer finden in ihrem LTG kein SEL#2. Sie finden nichts, führen deshalb ein ABORT aus und interpretieren den Rest ihres TIB nicht mehr. Von nun an arbeite ich mit Rechner#2 von meinem Terminal aus im Vordergrund wie mit jedem Forthsystem sonst auch. Auch wenn alle anderen jetzt im Vordergrund 'schweigen', laufen parallel dazu die Programme der Meßdatenerfassung und die Steuerung im Hintergrung weiter, angestoßen durch Interrupts. Sie tun also 'ihren Job' weiterhin. Die Mitteilungen im Forthvordergrund können nun alles sein: Daten für oder von einer bestimmten 'Task', wecken von 'Jobs', übermitteln neuer 'Jobs', völliges neukompilieren der gesammten Anwendung usw. Im FORTH selber findet sich der Deselektor SEL#_. Er dient dazu, den angerufenen Rechner 'auflegen' zu lassen und einen anderen 'anrufen' zu können. Ich sende einfach ein anderes Selectorwort aus, z.B. SEL#3. Rechner#2 kennt dieses Wort in FORTH nicht, erkennt aber die ersten vier Zeichen S-E-L-# und führt SEL#_ aus. Damit wird der bisher selektierte Rechner wieder in sein Vocabular LTG eingesperrt, deselektiert. (Der Trick mit diesem Wort ist im Forthcode erklärt.)

Dieses Verfahren ist simpel, aber für unsere Zwecke nützlich. Einschränkungen in der Anwendbarkeit sind daher bewußt in Kauf genommen worden: Man muß beachten, daß in der Zeit zwischen zwei Selects auf den Computern im Vordergrund nichts mehr geht. Da bei jedem Select der Stack gelöscht und das Vocabular umgeschaltet wird, muß die Kompilierung vorher abgeschlossen sein. Und SEL#n darf nicht das erstes Wort in einer Zeile sein, wenn es kompiliert werden soll. Vorausgesetzt in diesem Beispiel wurde ein Highlevel-Vektor für EMIT.

Für einen sichereren 'Forth-Handshake' könnte man das 'Null-Wort' des figForth so redefinieren, daß es eine Quittung zurückschickt.

(1) A.Krüger, Microcomputer Systeme, Schwelm. Zwei Versionen des Kreuzers und Umsetzers: V24/20mA und V24/RS422-485.
(2) Ramelow, Informations- und Kommunikationstechnik GmbH, Hamburg. Telemodem 1200 iAWD.

Schaltbild 1
Kreuzer

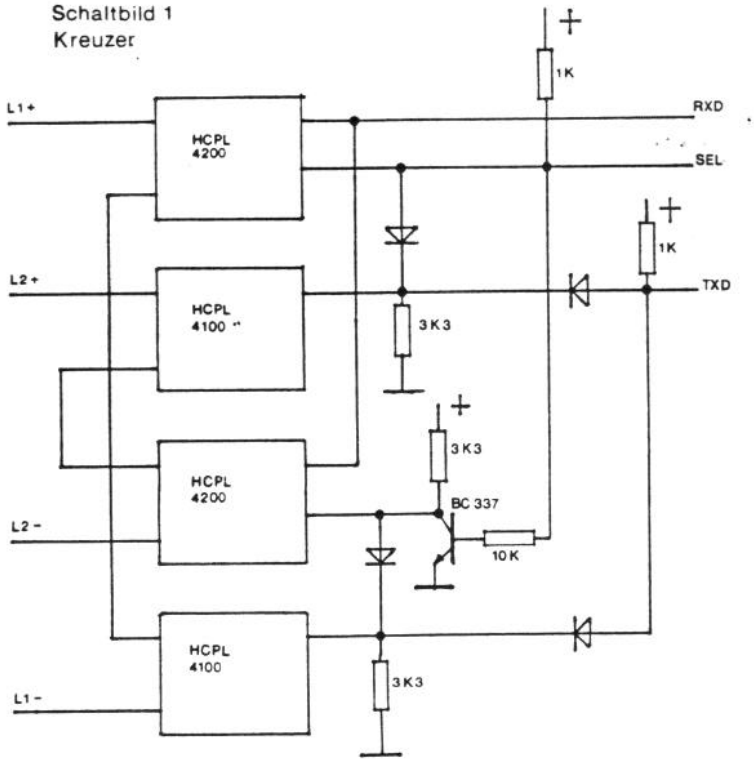


Abb.1

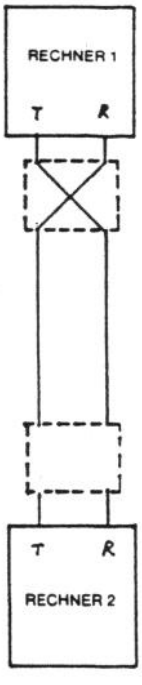


Abb.2

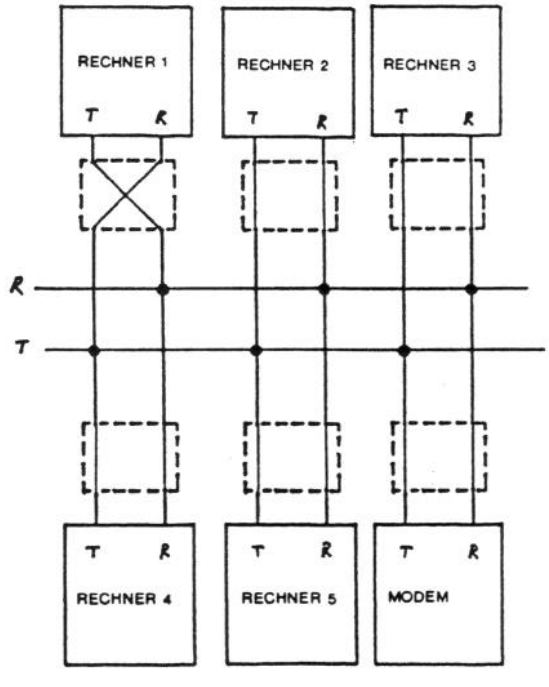
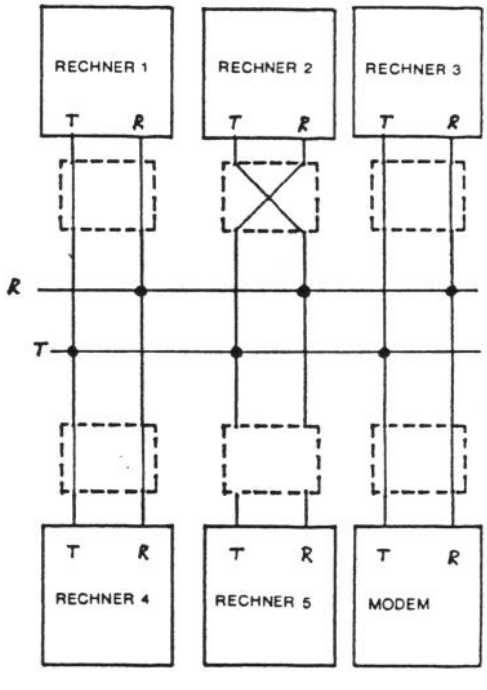


Abb. 3



```
( NETZCHEN                                  akr     1.4.86)

HEX A980 CONSTANT VIA

: KREUZEN  ( setze sel pin des Kreuzers )
  VIA 2 + C@ 04 OR VIA 2 + C!
  VIA C@  FB AND VIA C! ;

: ENTKREUZEN  ( loesche sel pin des Kreuzers )
  VIA C@ 04 OR VIA C! ;

: TEST
  KREUZEN
  CR ." SEL#7 MACH-WAS SEL#X " CR
  ENTKREUZEN ;

FORTH DEFINITIONS

: SEAL
  0 [COMPILE] ' LFA !
  [COMPILE] FORTH DEFINITIONS
  ;

VOCABULARY LTG IMMEDIATE

WIDTH @
4 WIDTH !       ( nur vier Zeichen des Namens in den Header )
: SEL#_
  ' DROP CFA UEMIT !            ( nicht mehr senden )
  [COMPILE] LTG   DEFINITIONS   ( nicht mehr interpretiren)
 ; IMMEDIATE                    ( da LTG gesealt ist )
WIDTH !

LTG DEFINITIONS

: SEL#0
 ' UEMIT CFA UEMIT !            ( jetz wieder senden )
 [COMPILE] FORTH  DEFINITIONS   ( auch was finden  )
 ;  IMMEDIATE

( SEL#0 muss erstes wort der Zeile sein , damit es wirkt )

SEAL SEL#0

FINIS
```

Glossar:

KREUZEN (---)
 Hardwareabhaengig

ENTKREUZEN (---)
 Hardwareabhaengig

SEAL (Name) (---)
  an 0000 in einer lfa erkennt das FIND , dass
  die Suche hier zuende ist .
  Nur Worte oberhalb von Name werden gefunden .

SEL#_   (---)
  deselectiert den Rechner , indem es den Interpreter
  im LTG Vocabular einsperrt .

FORTH GRUPPEN

Bundesrepublik Deutschland

Hamburg
Treffen jeden vierten Samstag im Monat ab 16:00Uhr in der Berufsfachschule für Radio- und Fernsehtechnik, Eimsbüttelerstr.64-66
Kontakt: Bernd Pennemann, 040-6900539

Karsruhe
Treffen jeden dritten Mittwoch im Monat ab 19:00Uhr im Jugend- und Begegnungszentrum, Krohnenplatz.
Kontakt: Michael Weiss, 0721-854994

Paderborn
Treffen in der Gruningerstr.20 nach Verabredung.
Kontakt: Thomas Asche, 05251-26496

Wuppertal
Treffen jeden vierten Samstag in Monat ab 20:00Uhr im Bahnhof Ottenbruch, Funckstrasse, W'tal-Elberfeld.
Kontakt: Michael Kalus, 02336-82204

Amateurfunker & Forth
Kontakt: Bernd Zimmermann 04105-52068

Darmstadt (*)
Kursus in der Volkshochschule.
Kontakt: Andreas Soeder, 06257-2744

Berlin (*)
Treffen nach Verabredung.
Kontakt: Hans Madlung, 030-4141831

Münchener Raum (*)
Treffen nach Verabredung.
Kontakt: Ekkehard Flögel, 08021-8414

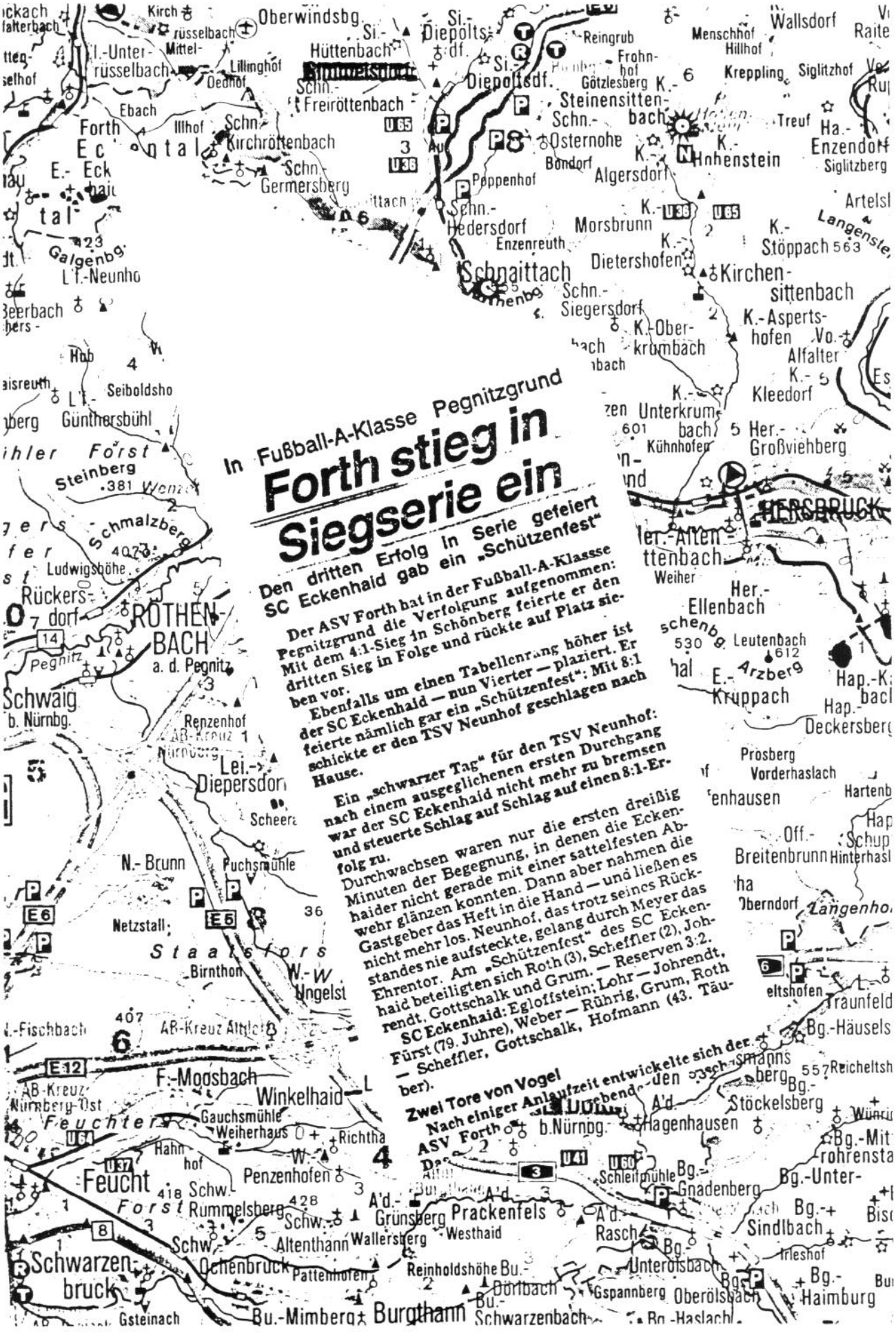
Braunschweiger Raum (*)
Treffen nach Verabredung.
Kontakt: Eckhard Heyne, 05352-58087

Moers (*)
Treffen nach Verabredung.
Kontakt: Hans Chrapia, 0203-3793274

östereich

Villach (*)
Forth-Club in Kärnten. Treffen nach Verabredung.
Kontakt: Heinz Klambauer, 04242-33566

(*) Forth-Enthusiasten oder Gruppen, die noch nicht offiziell als Gruppe der FG anerkannt sind, aber zu dem Zweck noch Verbindungen zu weiteren Forthlern suchen. Wenn Sie interessiert sind, kann hier auch IHR Name stehen. Schreiben Sie an die VIERTE DIMENSION



In Fußball-A-Klasse Pegnitzgrund

Forth stieg in Siegserie ein

Den dritten Erfolg in Serie gefeiert SC Eckenhaid gab ein „Schützenfest"

Der ASV Forth hat in der Fußball-A-Klasse Pegnitzgrund die Verfolgung aufgenommen: Mit dem 4:1-Sieg in Schönberg feierte er den dritten Sieg in Folge und rückte auf Platz sieben vor.

Ebenfalls um einen Tabellenrang höher ist der SC Eckenhaid — nun Vierter — plaziert. Er feierte nämlich gar ein „Schützenfest": Mit 8:1 schickte er den TSV Neunhof geschlagen nach Hause.

Ein „schwarzer Tag" für den TSV Neunhof: nach einem ausgeglichenen ersten Durchgang war der SC Eckenhaid nicht mehr zu bremsen und steuerte Schlag auf Schlag auf einen 8:1-Erfolg zu.

Durchwachsen waren nur die ersten dreißig Minuten der Begegnung, in denen die Eckenhaider nicht gerade mit einer sattelfesten Abwehr glänzen konnten. Dann aber nahmen die Gastgeber das Heft in die Hand — und ließen es nicht mehr los. Neunhof, das trotz seines Rückstandes nie aufsteckte, gelang durch Meyer das Ehrentor. Am „Schützenfest" des SC Eckenhaid beteiligten sich Roth (3), Scheffler (2), Johrendt, Gottschalk und Grum. — Reserven 3:2. **SC Eckenhaid:** Egloffstein; Lohr — Johrendt, Fürst (79. Juhre), Weber — Rühriq, Grum, Roth — Scheffler, Gottschalk, Hofmann (43. Täuber).

Zwei Tore von Vogel

Nach einiger Anlaufzeit entwickelte sich der ASV Forth [illegible] **den**

Antrag auf Mitgliedschaft in der Forth Gesellschaft eV

Antragsteller:

NAME:_____

Strasse:_____

PLZ/ORT:_____

TEL:_____/_____

Absenden an:
Forth Gesellschaft eV
Schanzenstrasse 27
2000 Hamburg 6

überweisungen bitte auf
Postgiroamt Hamburg
Forth Gesellschaft
Kto: 56 32 11 - 28
BLZ: 200 100 20

freiwillige Angaben für die interne Mitgliederliste:
(um neue Kontakte zu knüpfen und den Erfahrungsaustausch zu fördern)

Alter: ____ Beruf: _____ Forth-System: _____

verw. Computer:_____ Betriebssystem: _____

sonstiges: _____

Beitrag:

( ) --- DM 32,- / Jahr (Studenten + Arbeitslose nur mit Ausweis)
( ) --- DM 64,- / Jahr (Ordentliches Mitglied / Auslandsadresse)
( ) --- DM 128,- /Jahr (Förderndes Mitglied=Firmen/Institutionen)

Ausserdem unterstütze ich die Forth Gesellschaft eV mit einer Spende von _____ DM.

UNTERSTÜTZUNG:

( ) - Ich möchte eine Gebietsgruppe gründen. Daher bin ich mit der Veröffentlichung meiner Adresse in der VIERTE DIMENSION einverstanden.

( ) - Ich möchte eine Fachgruppe gründen . Daher bin ich mit der Veröffentlichung meiner Adresse in der VIERTE DIMENSION einverstanden. Das Thema der Fachgruppe lautet:

_____

_____

( ) - Ich möchte die Forth Gesellschaft eV unterstützen in dem ich Artikel / Buchbesprechungen / Produktnews für die VIERTE DIMENSION schreibe. Das Material (liegt bei) (ist unterwegs zur Redaktion)

( ) - ich habe ausserdem noch folgende Anregungen / Kritiken :

_____

_____

Den Gesamtbetrag von DM _____ ,- habe ich am _____

auf euer Konto überwiesen ( )
als Verrechnungsscheck beigelegt ( )

DATUM & ORT : _____ Unterschrift:_____