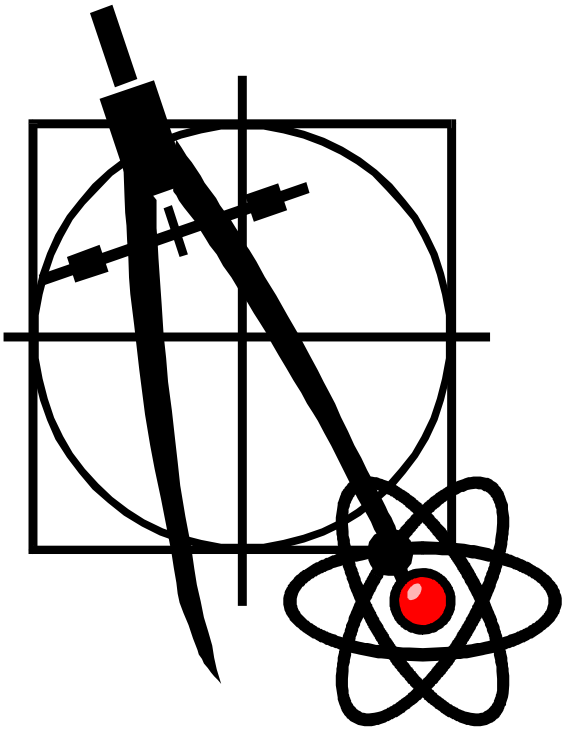
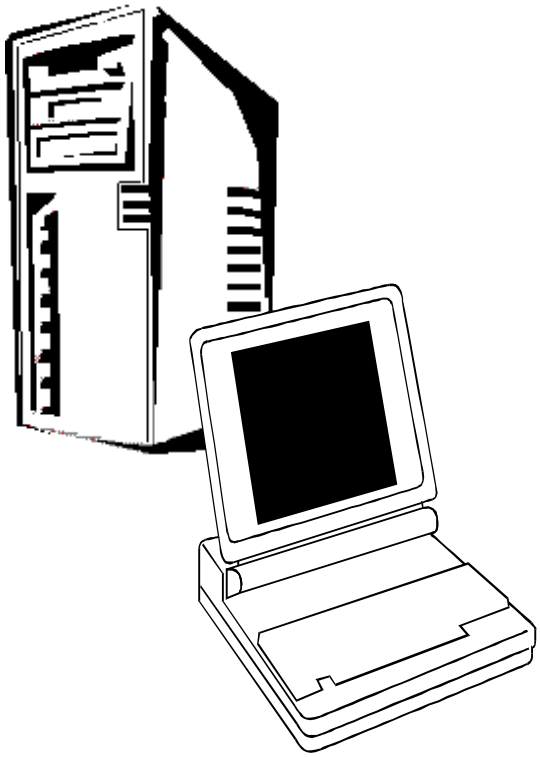
VIERTE DIMENSION

1 / 2001

Das FORTH-Magazin

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





In dieser Ausgabe

Dienstleistungen und Produkte fördernder Mitglieder des Vereins

Inhalt

Editorial

IMPRESSUM

Name der Zeitschrift

Vierte Dimension

Herausgeberin

Forth-Gesellschaft e.V.
Postfach 16 12 04
D-18025 Rostock
Tel.(Anrufbeantw.): 0381-400 78 28
Fax: 0381-400 78 28
E-Mail:
SECRETARY@FORTH-EV.DE
DIREKTORIUM@FORTH-EV.DE

Bankverbindung: Postbank Hamburg
BLZ 200 100 20
Kto. 563 211 208

Redaktion & Layout (vorübergehend)

Martin Bitter
Möllenkampweg 1a
46499 Hamminkeln
Tel.: 02857-1419
E-Mail: VD@FORTH-EV.DE
mbitter@bigfoot.de

Anzeigenverwaltung

Büro der Herausgeberin

Redaktionsschluß 2001

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

Erscheinungsweise

1 Ausgabe / Quartal

Einzelpreis

DM 10,- zzgl. Porto u. Verp.

Manuskripte und Rechte

Berücksichtigt werden alle eingesandten Manuskripte. Leserbriefe können ohne Rücksprache gekürzt wiedergegeben werden. Für die mit dem Namen des Verfassers gekennzeichneten Beiträge übernimmt die Redaktion lediglich die presserechtliche Verantwortung. Die in diesem Magazin veröffentlichten Beiträge sind urheberrechtlich geschützt. Übersetzung, Vervielfältigung, Nachdruck sowie Speicherung auf beliebigen Medien ist auszugsweise nur mit genauer Quellenangabe erlaubt. Die eingereichten Beiträge müssen frei von Ansprüchen Dritter sein. Veröffentlichte Programme gehen - soweit nichts anderes vermerkt ist - in die Public Domain über. Für Fehler im Text, in Schaltbildern, Aufbauskizzen u.ä., die zum Nichtfunktionieren oder eventuellem Schadhaftwerden von Bauelementen oder Geräten führen, kann keine Haftung übernommen werden. Sämtliche Veröffentlichungen erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes. Warennamen werden ohne Gewährleistung einer freien Verwendung benutzt.

Liebe Leserinnen und Leser,

Leider ist Friederich Prinz recht kurzfristig als Redakteur/Editor der Vierten Dimension ausgefallen (falls Sie sich Sorgen machen: dazu besteht kein Grund.) und ich bin ebenso kurzfristig für ihn eingesprungen. Dies und mein Kampf mit der Technik, insbesondere verschiedenen Drucktreibern führte dazu, dass Sie diese Vierte Dimension verspätet erhalten. Dafür bitte ich Sie im Namen aller Beteiligten um Entschuldigung.

Fritz Prinz wird nie müde, eine Aussage immer wieder zu wiederholen: Die Vierte Dimension ist das Kind der Forth-Gesellschaft! Das heißt, sie lebt von den Beiträgen, die Sie - liebe Leser – als Autoren einschicken.

In der Januarausgabe der c't befindet sich eine Analyse über die derzeitige Absatzflaute von Computerhardware im PC-Bereich. Dort wird schlussgefolgert, es sei ein Ende des ständigen Aufrüstungskarusells erreicht. Wer braucht's schon schneller?! Nebenbei wird behauptet, dass die programmtechnischen Möglichkeiten nicht wirklich mit dieser Hardwareentwicklung Schritt gehalten hätten (nur wenige Programme reizten die vorhandenen Prozessortechnologien aus).

Jörg Staben geht mit seinen insgesamt vier Beiträgen in diesem Heft auf eine solche fehlende? Softwareentwicklung auch bei Forth! ein. Er regt ein attraktives Projekt mit und für Forth an (s.h. Ein Lehrstück?) und zeigt gleichzeitig, dass eine 'modernes' Forth durchaus grafikfähig ist, wenn es vorhandene Resourcen (DLLs u.ä.) in seinen Code integriert.

Fred Behringer steht stellvertretend für den anderen Schwerpunkt von Forth. Die textbasierte (manchmal sogar 'hard-codierte') Programmierung 'simpler' Mikrochips (nun ja, embedded ist noch etwas anderes ;-).

Das Forth-Spektrum reicht von Kleinstrechnern, bis hin zu Workstations. Mitglieder der Forth-Gesellschaft sind Profis und Hobbyisten. Hoffen wir alle, dass von diesem Spektrum sich auch weiterhin ein repräsentativer? Querschnitt in der vierten Dimension findet. Ich weiß, dass es nicht immer einfach ist, neben der eigentlichen Arbeit über diese Arbeit zu berichten. Dennoch: Es gibt nichts Gutes – es sei den man tut es!

In diesem Sinne: Schicken Sie Beiträge!

Haben Sie ein Einsehen mit den armen Redakteuren!

Martin Bitter

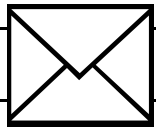
MB

(PS Wenn ich tippe, gibt es immer Rechtschreibfehlel! Drüber weglesen ;)

Quelltext Service

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

MB



Atomzeit und PC 4/2000 S. 38
Das Netzteil 4/2000 S.34

Hi Friederich,
ich habe gerade in der VD einen kleinen Artikel gelesen von wegen Atomzeit und dem PC.
Ich habe hier mehrere Rechner, die meisten unter Linux, aber auch 2 unter Windoof (98 und 2000). Vielleicht interessiert es irgendjemand, dass man dann die Sache recht einfach machen kann:
Man fügt auf einem der Linux-Rechner im cron.daily einfach einen Aufruf von ntp ein, der einmal am Tag sich mit einem Zeitserver verbindet. Auf den Windoof-Kisten verwende ich NetLab (ist Freeware), das neben anderen sehr nützlichen Sachen auch eine automatische Zeitanpassung bietet. Da auf einem PC unter Win die Uhr ja chronisch falsch geht, fragen diese Rechner (per NetLab) einmal pro Stunde den Linux-Rechner nach der korrekten Zeit.
Das geht einwandfrei und der große Vorteil ist der, dass nicht jeder Rechner in meinem Netz getrennt nach der Zeit bei der PTB (oder einem anderen NTP-Server) fragt (Telefonkosten).
Etwas anderes: Die Glosse "Das Netzteil" ist einfach göttlich! Wo kann ich die in elektronischer Form kriegen? Ich möchte sie an ein paar Freunde mailen.
CU Uli --
Ulrich Paul , Paul Elektronik

Hier kann geholfen werden: die Glosse „Das Netzteil“ ist (e-)nachzulesen unter: www.witze.de (dort in der Suche „netzteil“ eintippen, oder gleich eingeben http://www.witz.de/cgi-bin/warp/warp.pl?zone=witze&navigation=witze&seite=computer/netzteil.htm ohne Trennungsstriche in 'netzteil' und 'computer'). Sollten alle Stricke reißen: Kurze E-Mail an mbitter@bigfoot.de. Wer einen weiteren netten Humorbeitrag findet: die Redaktion freut sich darüber.

RCX Selbstbau 4/2000 S. 34

Sehr geehrte Damen und Herren,
mit Interesse habe ich die gerade erhaltene neue VD gelesen. Ihr Artikel zum RCX-Modul auf Seite 34 regte mich zu den folgenden Zeilen an:
Ihr Ärger über die komplizierte Demontage und die Preisgestaltung des RCX-Modules ist verständlich. Auch ich denke mit Wehmut an die Zeiten zurück, in denen man beim Kauf eines Druckers oder Oszilloskops (von TEKTRONIX habe ich damals sehr viel analoges Schaltungsdesign gelernt!) eine umfassende Dokumentation mit Schaltbildern, Lageplänen, Stücklisten, bis hin zu Abgleich- und Reparaturanleitungen erhielt.
Man muss sich jedoch über einige Dinge im Klaren sein, die solche Entwicklungen haben entstehen lassen:
1. Ist es mit Sicherheit der verständliche Wunsch der Firma LEGO, den eigenen Wettbewerbsvorteil zu wahren, indem man den Nachbau erschwert (theoretisch hätte ich das TEK 560 nachbauen können. Sic!).
2. Ist es eine unmittelbare Folge des Produkthaftpflichtgesetzes. Als Produzent oder Importeur müssen Sie heute dem größten Dummdepp technisch kompliziertes Gerät in die Hand geben und dabei sicherstellen, dass nichts Schlimmes passieren kann, selbst wenn der Anwender das Gerät völlig fehl bedient. Gerade Produkte der Firma LEGO sind vorwiegend für Kinder und Jugendliche gedacht, bei denen von vornherein mit -spielerisch- missbräuchlich Anwendung zu rechnen ist.
Aus Gründen der Produzentenhaftung wäre es völlig falsch (vermutlich sogar gesetzwidrig), dem Anwender eine missbräuchlich Benutzung durch entsprechend leichte Zugangsmöglichkeit zum Inneren des Gerätes oder Offenlegung von Internas zu ermöglichen.
Wir könnten jetzt noch lange darüber filosofieren (oder philosophieren? ich komme mit der neuen Rechtschreibung noch nicht gut zurecht (leider? immer noch philosophieren der Sätzä)), dass die Menschen sich nicht gerne den vorsichtigen Umgang mit potentiell gefährdenden Dingen vorschreiben lassen, aber im Schadensfall den Schuldigen dann lieber woanders suchen (Jeder beansprucht sein Recht auf Selbstverantwortung, aber nur wenige wollen die Verantwortung dann auch wirklich "selbst" übernehmen).
Übrigens finden Sie aus den vorgenannten Gründen heutzutage auch in besseren Fernsehgeräten meist keinen Schaltplan mehr und das dicke Handbuch, welches bei Ihrem nächsten neuen Drucker mitgeliefert wird, enthält auf mehreren Seiten die vielfältigsten Warnhinweise in allen europäischen Sprachen nebst lokalen Unterdialekten, aber bestimmt keinen Schaltplan.

Mit freundlichem Gruß
Peter Güttler
APS Software Engineering GmbH

Kommentar: Herr Güttler hat in weiten Teilen recht. Wer einmal versuchte, ein elektrisches Gerät 'narrensicher' über den TÜV etc. zu bringen, weiß, wie das ins Geld gehen kann. Ergänzend dazu: Den RCX gibt es in zwei Versionen. Einmal mit Anschlussbuchse für ein externes Netzteil (Modell 1.0), einmal ohne einen solchen Anschluss (Modell 1.5). Legomitarbeiter haben meiner Vermutung, das gehe auf schlechte Erfahrungen mit dem amerikanischen Verbraucher(schutz) zurück, widersprochen. Vielmehr handle es sich bei dem Modell 1.0 (mit Netzteilanschluss) um ein Modell, das nur für den Schulgebrauch entwickelt und vertreiben werde. Warum wohl nur da?

========= Leserbrief ==========

Kann mir jemand erklären, was bei IP (Intentional Programming - iX 11/2000, s.142) so revolutionär ist, dass es in Forth nicht schon eingebaut wäre und ständig verwendet würde? "Der mittlere Teil des C-Compilers sollte dem Entwickler zugängig gemacht werden", steht da. Für uns doch eigentlich eine Selbstverständlichkeit? Der vordere und hintere Teil des Metacompilers auch, so wie wir das gewöhnt sind? Oder gibt es das in C-Derivaten gar nicht? (Ein bisschen Provokation darf sein?) Der Entwickler soll Befehle und Daten situationsbedingt nach eigenem Geschmack umstrukturieren dürfen. Oder habe ich das falsch verstanden? Könnte mal jemand einen Artikel über "Forth und die Intentionelle Programmierung" schreiben? Absichtliche Programmierung? Programmierung nach eigenem Geschmack? Oder vielleicht doch einfach nur "Programmieren in Forth"?

beh.

Küstenforth gegründet!

Hallo Friedrich,
mein Name ist Klaus Zobawa. Wir kennen uns von der letzten FORTH-Tagung in Hamburg. Ich bin Teilnehmer in der neugegründeten regionalen FORTH-Gruppe in HAMBURG.
Darum geht's: Die Küstenforth-er treffen sich nun regelmäßig jeden 4. Freitag im Monat um Uhr 16:30 in den Räumen der Fa. SEND, Stubbenhuk 10 in 20459 Hamburg.
Ich bitte Dich diese Info in die VD mit aufzunehmen. Wichtig dabei ist - darauf legt Klaus Schleisiek viel Wert - die Bezeichnung Küstenforth. Sie muss unbedingt genannt werden. Ansprechpartner ist Klaus Schleisiek, Tel. 040 / 375008-13.
Viele Grüße im Namen aller Küstenforther
Klaus

Dem haben wir sofort und gerne entsprochen, wie ein Blick auf die vorletzte Seite zeigt.

Eine Bitte:

Der hoffentlich vorübergehende Wechsel der Redaktion brachte auch einen Wechsel des Satzprogrammes bzw. der Textverarbeitung mit sich. Alle Formate, Anzeigentexte usw. wurden vom Microsoft Publisher 98, in manchmal mühevoller Handarbeit nach Star Office 5.2 übertragen. (Bin ich selbst schuld, warum verwende ich auch keinen Publisher!)
Dabei kann es vorgekommen sein, dass nicht alle Anzeigen und periodische Mitteilungen etc. fehlerfrei geblieben sind. Bei Fehlern bitte ich um kurze Benachrichtigung.

MB

Bericht vom Küstenforth

Hallo Friederich,
das Hamburger 4th-Treffen hat Folgen gehabt:
Inzwischen trifft sich regelmäßig jeden 4. Freitag im Monat ab 16:30 die "Küsten4th-Gruppe" zumeist in den Räumen der Fa. SEND GmbH, Stubbenhuk 10, 2. Stock, 20459 Hamburg (U3 "Baumwall").
Regelmäßige Teilnehmer: Ulrich Hoffmann, Andrej Kostrov, Manfred Mahlow, Klaus Schleisiek, Klaus Zobawa
Zur Zeit wird emsig an Objekten und Methoden mit Vererbung gearbeitet, bzw. die 4thige Lösung ist bereits gefunden und zur Zeit vereinfacht und wird bei nächster Gelegenheit nicht nur vorgetragen werden, sondern auch veröffentlicht.
Letzten Freitag (27.10.) haben wir uns ausnahmsweise in der TU-Harburg bei Prof. Fritz Mayer-Lindenberg getroffen, der uns sein "Fifth" vorgeführt hat. Das basiert zwar nicht direkt auf Forth, sieht aber sehr "forthig" aus. Einzigartig daran ist, dass es prinzipiell davon ausgeht, dass jeder Problemlösung ein System mehrerer miteinander kommunizierender Prozessoren zugrunde liegt. Eine solche Struktur wird in einer Art Deklarationsteil definiert und dann werden alle Prozessoren gleichzeitig im Rahmen eines einzigen Programmes programmiert. Sehr eindrucksvoll.
:-)
Klaus Schleisiek
SEND Signal-Elektronik und Netz-Dienste GmbH

(Hoppla. Ich kenne wohl ein anderes 'fifth'. Eine 'forthähnliche IDE-Applikation' für das Betriebssystem V2OS, das sich anschickt, das schnellste Betriebssystem auf 386 und 486 PCs zu werden. Liegt da ein Namenskonflikt vor? Oder handelt es sich um zwei Seiten einer Medaille? (s.h. http://v2os.v2.nl/~vp/)

MB)

strongForth 0.03 liegt vor

Dr. Stephan Becher lebt in Dänemark. Er entwickelt strongForth, ein Forth das zwar nicht ANSI konform ist, aber diesem Standard weitgehend entspricht. Es ist neben anderem der Versuch, ein streng (strong) typendeklariertes Forth zu entwerfen, bei dem sowohl Compiler als auch Interpreter die Werte auf dem Stack auf ihre Typverträglichkeit mit dem aufrufenden Wort überprüfen. Dies bringt nicht nur eine geringere Fehleranfälligkeit, sondern erlaubt auch die mehrfache (Wieder)Verwendung von Wortnamen, da gleichnamige Worte sich durch ihre Typzugehörigkeit unterscheiden.
Ein Ansatz, über den sich viel philosophieren lässt. strongForth 0.03 läuft unter Dos, bzw. in diversen Dos-Boxen. Es steht bei Dr. Becher zum Download bereit.
http://home.tonline.de/home/s.becher/forth/index.htm

MB

AMD Athlon Tachometer von Marcel Hendrix

Wer schon immer mal wissen wollte, wie schnell denn sein Athlon ist und wie 'man' das herausbekommt, der kann bei Marcel Hendrix, dem Vater des IForth, vorbeisehen und dort erfahren welche Register (und welche nicht) bei dieser CPU von AMD dazu ausgelesen werden können.
Der Performancecheck ist nicht für Intelbassierte CPUs geschrieben und läuft (leider?) 'nur' unter WIN2000 und WIN-NT. http://home.iae.nl/users/mhx/perf.html

MB

Jörg Staben baut deutschsprachige Win32for Website auf

Jörg Staben, in dieser Ausgabe der Vierten Dimension mit drei Beiträgen vertreten, baut eine Website zum Programmieren mit Win32for von Tom Zimmer auf. Ein Besuch lohnt sich jetzt schon (allein wegen der Links-Seite). Die Seite ist für Jörg Staben kostenlos, dafür aber mit Werbebannern belastet. Wer eine bessere Lösung kennt, möge sich an Jörg Staben wenden (Auf der Homepage unter Kommentar)
http://forth-programmieren.webjump.com/

MB

KForth für Linux und Windows 95/98/NT

Fast wie eine 'eierlegende Wollmilchsau' hört sich das an, was Krishna Myneni, wissenschaftliche Assistent an der Universität von Loisville, da plant. Ein Forth, unter zwei Betriebssystemen mit freien Quellen (Cygwin), entwickelt. Sinn und Zweck des Ganzen ist es, Studenten naturwissenschaftlicher Fächer Grundlagen der Informatik bis hin zur Implementation einer Programmiersprache nahe zu bringen. Nicht so ganz nebenbei fallen einige Tools und ein Tutorial ab.
http://ourworld.compuserve.com/homepages/krishnamyneni/-ccre/ccre.htm (kein Trennungsstrich zwischen / und 'ccre')

Bibliotheks- und Sicherheitskonzepte in Forth und anderen Programmiersprachen

von Jörg Staben, Langenfeld

Stichworte:

JavaBeans, ActiveX, Bibliotheken, binäre Softwarekomponenten, Komponentenarchitektur, Internet, Sicherheitskonzepte, signing, sandboxing, Java, Forth

Bibliotheken?

Wenn Sie in die örtliche Bibliothek gehen, dann möchten Sie gerne von dem profitieren, das andere bereits geleistet haben.

Das kann für Sie in der Leihbücherei kostenfrei sein; in der Buchhandlung wird der Verkäufer Geld erwarten; das Gefundene kann Ihren Bedürfnissen und Erwartungen entsprechen oder auch nicht – im Idealfall haben Sie durch eine Bibliothek mit wenig Aufwand ein Problem gelöst, das Sie hatten.

Das ist die Erwartung, die hinter dem Wunsch nach Bibliotheken steht. Auch in der Programmierung.

Gestern

Schon in der Götterdämmerung der PC-Programmierung, in den Zeiten seit 1984, haben Bibliotheken eine Rolle gespielt. Schauen wir uns ein Borland-Produkt, den PACAL-Compiler ´TurboPASCAL´ an:

Die Firma Borland hat – fast als historische Leistung - erste Bibliothekskonzepte für ihr Produkt TurboPASCAL entwickelt:

Quelltextbibliotheken (INCLUDE Konzept)

Binäre Bibliotheken (UNIT-Kozept)

Solche Bibliotheken (ToolBoxen) konnten von Drittanbietern wie ´TurboPower´ hinzugekauft werden, standen lizenzfrei in Toolsammlungen zur Verfügung oder waren zunehmend mehr im Lieferumfang des Produktes enthalten (TurboVision für die Programmierung von Bedienoberflächen).

Die Vorteile von Bibliotheken liegen auf der Hand:

Ohne sich selbst jeweils immer wieder neu in die Grundlagen einarbeiten zu müssen, können Sie beispielsweise 3-dimensionale grafische Darstellungen verwirklichen, auf dBase-Datenbanken zugreifen oder Schwachpunkte Ihres Programmierwerkzeuges ausgleichen.

Nachteile gibt´s natürlich auch:

Versionsabhängigkeiten: die frisch gekaufte ToolBox läuft mit der neuen Compilerverson nicht mehr. Man ist auf die Produktpflege durch den Drittanbieter angewiesen.

Herstellerabhängigkeiten: Fehler in binären Bibliotheken können nicht – ohne den Quelltext zu besitzen – eigenständig korrigiert werden.

Compilerabhängigkeiten: die Bibliothek für Borland Compiler ist nicht für Microsoft-Produkte geeignet.

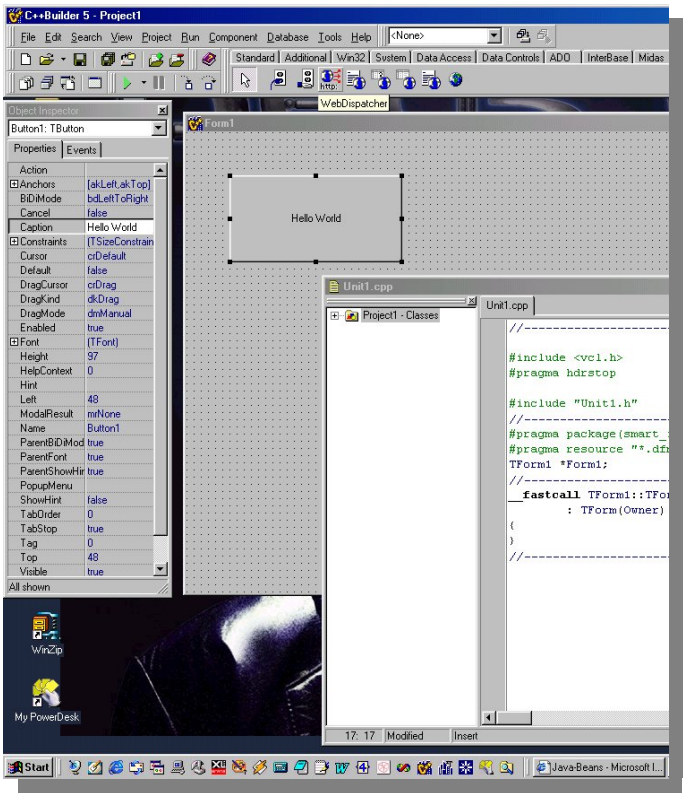
Sprachabhängigkeiten: eine Bibliothek für PASCAL ist beim Einsatz eines C-Compilers nutzlos.

Und deshalb wurde nur sehr wenig Software wiederverwendet; und wenn, dann meist in Form von mehr oder weniger gepflegten Quelltextbibliotheken.

Heute

Bleiben wir bei der Firma Borland.

Während der letzten zehn Jahre ist aus ´TurboPASCAL´ die Entwicklungsumgebung ´Delphi´ geworden, um in PASCAL Programme zu entwickeln. Für den C-Programmierer bietet Borland eine nahezu identisch aussehende (!) Entwicklungsumgebung ´C++Builder´ an.



Auch bei anderen Anbietern haben sich die Produkte weiterentwickelt; so wurde aus ´Quick-C´ von Microsoft das ´Visual Studio´ und Sun hat ´Forte for Java´ auf dem Markt.

Formularbasierte Programmierung

All´ diesen Entwicklungsumgebungen ist gemeinsam, dass sie mit einer Komponentenpalette kommen, aus der der Programmierer einzelne sichtbare Komponenten oder auch unsichtbare Komponenten auswählt.

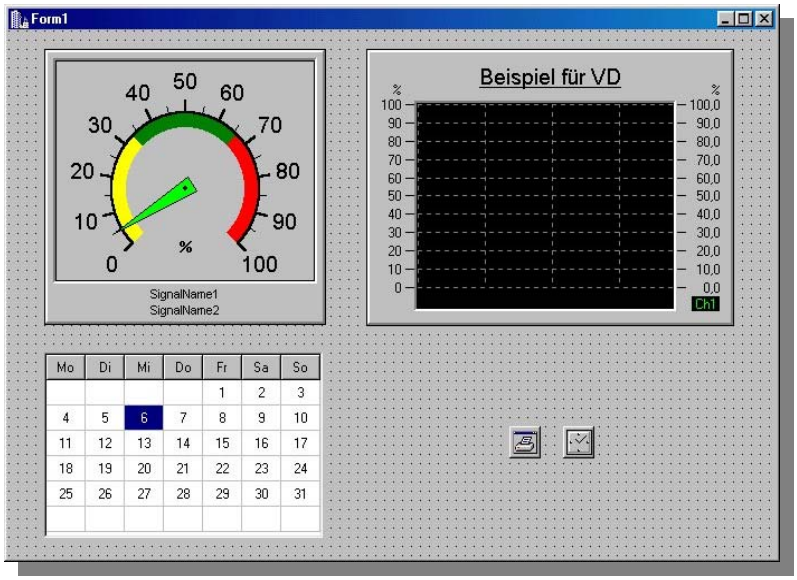
Sichtbare Komponenten können Buttons, Eingabefelder oder ein Kalender sein, unsichtbare können Zeitgeber, Datenbankmodule, aber auch Internetanbindung sein.

Komponentenbibliotheken

Alle genannten Entwicklungsumgebungen ermöglichen es dem Programmierer, seine Anwendung visuell zu entwickeln, indem er benötigte Komponenten auf einem Formular platziert und diese Komponenten in Beziehung zu einander bringt.

Diese hier gezeigten Komponenten sind interaktiv; schon während der Programmentwicklung bewegt sich der Zeiger, sobald Daten kommen, werden Linien gezeichnet und der Kalender liefert das aktuelle Tagesdatum zurück.

Die beiden kleinen Komponenten sind später unsichtbar; es sind ein Druckerdialog und ein Zeitgeber.



Stellen Sie sich vor, Sie hätten das alles selbst programmieren müssen..

Komponenten können mit oder ohne Quelltext kommen; Komponenten, auch von Drittanbietern wie diese hier, werden einfach in der Entwicklungsumgebung angemeldet und stehen Sekunden später zur Verfügung.

Und so entwerfen der C-Programmierer mit ´C++Builder´ und der PASCAL-Programmierer mit ´Delphi´ ihre Programme – und zwar mit identischen, gemeinsam nutzbaren (!), sprachunabhänigen (!), wiederverwendbaren (!), ausgetesteten (!) Komponenten, die vom Lieferumfang oder von Drittanbietern zur Verfügung gestellt werden.

Morgen

Wenn komponenten-basierte Programmierung die nächste große Welle in der Entwicklung von Software ist, und viele glauben das, dann wird sich derjenige durchsetzen, der über die dominierende Komponenten-Architektur verfügt.

Zur Zeit (2000) wetteifern zwei Firmen darum, wer die dominierende Komponenten-Architektur auf dem zunehmend wichtiger werdenden Desktop-PC bestimmt:

Sun mit ihrer JavaBeans Technologie und Microsoft mit den ActiveX Controls, häufig auch COM-Objekte genannt.

Was sind denn überhaupt Komponenten?

JavaBeans und COM-Objekte (=ActiveX-Controls) sind sprachunabhänige Komponenten für alle Programmiersprachen – auch für Forth.

Eine Komponente ist ein abgeschlossenes Programmteil, das innerhalb eines größeren Ganzen abläuft; also keine ausführbare Anwendung, sondern eben ein Bibliotheksmodul.

Eine Komponente

läuft innerhalb eines Containers: eines Visual Basic oder eines Web Browsers wie Netscape Navigator oder Internet Explorer.

stellt dem Container Eigenschaften und Methoden zur Verfügung.

stellt dem Container eine Anwenderschnittstelle zur Verfügung, in der die Eigenschaften typischerweise visuell vom Programmierer verändert werden.

sendet dem Container Ereignisse.

wird dem Anwender in ausführbarer Form zur Verfügung gestellt.

Komponenten, die auf Servern laufen statt auf dem Desktop-PC (und daher keine Anwenderschnittstelle haben), werden ebenfalls ActiveX Komponenten genannt; wenn der Container, in dem sie ablaufen, ein Microsoft Transaction Server ist, heißen sie MTS Komponenten oder MTS Objekte.

Die Java Nomenklatur ist einfacher. Desktop Komponenten nennt man JavaBeans, die Komponenten für den Servereinsatz sind die Enterprise JavaBeans.

Und welche sind nun die besseren Komponenten?

ActiveX Controls und JavaBeans haben einen großen Vorteil gemeinsam: Es ist nicht allzu schwer, die Schnittstellen des einen in die des anderen zu übertragen und sowohl Microsoft als auch Sun bieten Software an, die das leistet.

Der entscheidende Unterschied ist, daß ActiveX Controls fast immer 80x86-Code, also Intel Binärcode ist, der nur auf Windows und Windows NT Plattformen läuft, während JavaBeans prinzipiell auf vielen Systemen laufen kann, also unabhängig von der verwendeten Plattform sind. JavaBeans sind also sprach- und plattformunabhänige Komponenten!! (Denken Sie jetzt bitte an die Situation früher zurück.)

Das sieht zuerst nach einem offensichtlichen Vorteil für JavaBeans aus. Dasselbe Programm auf unterschiedlichen Plattformen laufen zu lassen, ist doch ein Vorteil?

Die Wirklichkeit seht anders aus. Zuerst einmal bedingt die Fähigkeit, denselben Code auf jeder Kombination von Betriebssystem und Prozessor auszuführen, irgendeine Art von Überstzung. Für JavaBeans wird diese Übersetzung von der Java Virtual Machine und der Java Umgebung geleistet.

Nur – eine allgemeine Betriebsumgebung kann vielleicht langsam sein, sie ist aber bestimmt uneffizient.

Wie Forth – ein portables Standard-Forth-System nutzt die Maschine nicht aus, kann sie nicht ausnutzen, weil es portabel ist. Ein schnelles effizientes Forth ist nicht mehr portabel und kann nicht mehr standardkonform sein.

So können auch JavaBeans nicht die hochspezialisierten Dienste eines bestimmten Betriebssystems nutzen; durch die Portabilität sind sie immer auf den kleinsten gemeinsamen Nenner beschränkt.

Gut. Portabilität könnte diesen Preis wert sein, wenn sie überhaupt nötig ist.

Und das ist die zweite große Herausforderung für JavaBeans: Die überwältigende Mehrheit der Schreibtisch-PCs sind Wintel-Maschinen: Intel drin, Windows drauf.

Viele Firmen haben ausschließlich Wintel Desktops. Warum sollte man unter dieser Voraussetzung eine Komponentenarchitektur mit Nachteilen einsetzen, wenn man die Vorteile nicht braucht? Für Firmen mit vorrangig Wintel-Desktop-Maschinen machen JavaBeans Komponenten wenig Sinn.

Der Markt für Komponenten spiegelt diese Wirklichkeit auch wieder.

Es sind Tausende von ActiveX Controls von Hunderen von Anbietern verfügbar. Dagegen ist der Markt für JavaBeans viel kleiner. Dies liegt z.T. daran, daß JavaBeans relative neu sind. Aber es ist auch eine wirtschaftliche Entscheidung unabhängiger Software-Anbieter. Wenn man mit seinen Komponenten 90% des Marktes bedienen kann, warum noch um die letzten 10% kümmern?

Gibt es noch Hoffnung für JavaBeans oder beherrscht Microsoft alles?

Die erste Antwort: Ja. Die zweite Antwort:Vielleicht.

Die Chance für JavaBeans liegt im Internet und auf den Servern.

Im öffentlichen Internet können Server weniger Annahmen darüber machen, wer ihre Clients sind und was ihre Clients können. Hier machen plattformunabhänige Komponenten Sinn, weil man von hier eher von unterschiedlichen Clients ausgehen kann, als im Intranet weltweiter Firmen.

Ein zweites Argument macht JavaBeans für Internet-basierte Anwendungen attraktiv: **Sicherheit**.

Es ist viel über die Sicherheitsprobleme mit ActiveX Controls geschrieben worden, dabei läßt sich die Diskussion auf die beiden Konzepte **signing** und **sandboxing** beschränken.

Java kann beides, während ActiveX Controls ihr Hauptaugenmerk auf signing legen.

In vielen Fällen ist signing auch die richtige Lösung (deshalb hat Java auch diese Fähigkeit bekommen). Beim zufälligen Herunterladen völlig unbekannter Komponenten aus dem Internet auf den eigenen Desktop-PC reicht signing nicht.

Nur jemand, dem seine Firma oder sein Unternehmen völlig egal ist, wird eine Komponente aus dem öffentlichen Internet auf seinen Desktop-PC herunterladen, die von jemandem gezeichnet worden ist, den er nicht kennt und dem er nicht vertraut.

Sprachsicherheit in Java

Java wurde von Anfang mit höheren Sicherheitsansprüchen entworfen, als dies üblicherweise bei Programmiersprachen der Fall ist. Einer der Hauptgründe dafür war der Wunsch, den Aufruf von Applets, die aus dem Internet geladen wurden, so sicher wie möglich zu machen. Selbst bösartige Applets sollten nicht in der Lage sein, ernsthafte Angriffe auf den Computer, das Betriebssystem oder die Ressourcen des Anwenders auszuführen.

Sicherheit beginnt in Java schon bei der Implementierung der Sprache. Anders als in C oder C++ gibt es beispielsweise keine direkten Zugriffe auf den Hauptspeicher und keine Pointerarithmetik. Das Memory-Management arbeitet vollautomatisch. Sicherheitslücken, die durch (provozierte) Speicherüberläufe verursacht werden, sind damit nicht ohne weiteres möglich.

ActiveX steht am anderen Ende des Spektrums. Programmieren mit ActiveX ist wie Windows programmieren – man kann machen, was man will. Wenn Sie auf eine Web-Seite klicken, die eine ActiveX Komponente auf Ihren Rechner lädt, hat dieses Control vollen Zugriff auf die Dateien Ihres Rechners. Dies ist der Traum eines jeden Virus-Programmierers.

Eine Lösung für dieses Problem schien das signing, die "digitale Unterschrift", die zeigt, welcher Autor den Programmcode geschrieben hat. Dies basiert auf der Idee, das ein Virus nur arbeiten kann, wenn der Autor anonym ist. Klar, ansonsten würde man ihn für sein Handeln verantwortlich machen.

Signing - Digitale Unterschriften

Ein großer Vorteil der Kryptosysteme auf Public-Key-Basis ist, daß sie Möglichkeiten zum Erstellen und Verifizieren von *digitalen Unterschriften* bieten. Eine digitale Unterschrift besitzt folgende wichtige Eigenschaften:

> Sie stellt sicher, daß eine Nachricht von einem ganz bestimmten und eindeutig identifizierbaren Absender stammt.
>
> Sie stellt sicher, daß die Nachricht intakt ist und nicht während der Übertragung verändert wurde.

Beide Eigenschaften sind für den elektronischen Datenverkehr so fundamental wie die Verschlüsselung selbst. Technisch basieren sie darauf, daß die Funktionsweise eines Public-Key-Kryptosystems sich umkehren läßt, daß es also möglich ist, Nachrichten, die mit einem privaten Schlüssel verschlüsselt wurden, mit Hilfe des korrespondierenden öffentlichen Schlüssels zu entschlüsseln.

Im Prinzip funktioniert eine digitale Unterschrift so:

Will A eine Nachricht signieren, so verschlüsselt er sie mit seinem privaten Schlüssel. Jeder, der im Besitz des öffentlichen Schlüssel von A ist, kann sie entschlüsseln. Da nur A seinen eigenen privaten Schlüssel kennt, *muß* die Nachricht von ihm stammen. Da es keinem Dritten möglich ist, die entschlüsselte Nachricht zu modifizieren und sie erneut mit dem privaten Schlüssel von A zu verschlüsseln, ist auch die Integrität der Nachricht sichergestellt. Den Vorgang des Überprüfens der Integrität und Authentizität bezeichnet man als *Verifizieren* einer digitalen Unterschrift.

Das Sandbox-Konzept

Traditionell wurde in Java zwischen lokalem Code und solchem, der aus dem Netz geladen wird, bezüglich seiner Sicherheitsanforderungen rigoros unterschieden. Während lokalem Code (also Applikationen und von der Festplatte geladenen Applets) der Zugriff auf alle verfügbaren Ressourcen gestattet ist, dürfen Applets, die aus dem Netz geladen wurden, nur einen kleinen Teil davon verwenden.

Sie halten sich gewissermaßen in einem Sandkasten auf, in dem sie nur ungefährliche Spielzeuge verwenden und keinen ernsthaften Schaden anrichten können (daher der Name »Sandbox«=Sandkasten).

Zu den "gefährlichen Spielzeugen", die nicht verwendet werden dürfen, zählen:

> der lesende und schreibende Zugriff auf Dateien des lokalen Computers,
>
> das Öffnen von TCP/IP-Verbindungen zu einem anderen als dem Host, von dem das Applet geladen wurde,
>
> das Akzeptieren von TCP/IP-Verbindungen auf privilegierten Portnummern,
>
> das Lesen benutzerbezogener System-Properties wie "user.name", "user.home", "user.dir" oder "java.home",
>
> das Erzeugen eines Top-Level-Fensters ohne Warnhinweis,
>
> das Ausführen externer Programme,
>
> das Laden von System-Libraries,
>
> das Beenden der virtuellen Maschine

Beim Herunterladen einer anonymen Komponente aus dem öffentlichen Internet auf den Desktop-PC ist Sandboxing eine absolut notwendige Sicherheitsmaßnahme. Hier – wenn es um Internet-basierte Anwendungen geht - haben JavaBeans einen ganz klaren Vorteil über ActiveX controls.

P.S.: Ach so, ich habe ganz vergessen zu erwähnen:

Applets sind Anwendungen, die in einem Web-Browser ausgeführt werden.
Forth weiß von all dem hier nichts und beschränkt sich auf Quelltextbibliotheken in ANSI-standardisiertem Forth.

Der Chefprogrammierer einer großen Softwarefirma zieht sich für ca. eine Woche zu einer 'kreativen Phase' in sein kommunikationsloses, einsam gelegenes Wochenendhaus zurück, um in aller Abgeschiedenheit, den programmphilosophischen Ansatz seines Hauses zu überdenken. Als er nach einer Woche nicht erscheint, wundert man sich. Nach zwei Wochen macht sich die Firmenleitung Sorgen und schickt Mitarbeiter des Sicherheitsdienstes zu dem Wochenendhaus.
Die Wachleute finden den Programmierer tot unter der laufenden Dusche. In der Hand hält er noch eine Flasche Super-Haarwuchs-Schampoo. Darauf steht: „Anleitung: Einseifen – Auswaschen – Wiederholen."

UUENCODE und UUDECODE

von Wil Baden
*(*Aufgefangen von Fritz Prinz bei: sftalk@forth.com, der offiziellen NG der Swiftforthuser, eingedeutscht von *MB)*

Geschichtliches

„uuencode“ ist ein UNIX-Programm, das binäre Daten ins ASCII-Format konvertiert (enkodiert). „uuencode“ wurde urspünglich zusammen mit „uucp“ verwendet, um Binärdateien über serielle Verbindungen zu versenden, die das höchste Bit eines Buchstabenbytes ignorierten. Zur Zeit jedoch ist sein Haupteinsatzgebiet die Versendung von Binärdateien per E-Mail oder als Posting in Newsgroups.

Das Programm „uudecode“ bildet den Umkehreffekt zu „uuencode“ und restauriert exakt die ursprünglichen Binärdateien.

Das Format

Mit „uuencode“ gewandelte Daten beginnen mit einer Zeile folgenden Formats:
begin <mode> <file>
gefolgt von den eigentlichen Datenzeilen. In der letzten Zeile steht: end

<mode> beinhaltet drei Oktalziffern, die die Lese/Schreib-/Ausführungsrechte repräsentieren und <file> ist der Dateiname, den die zu rekonstruierende Datei bekommt.

Die 'Kodierung'

„uunecode“ liest die ganze Datei in Gruppen von drei Bytes ein. Sollten zwei oder ein Byte übrig bleiben, so wird mit Null-Bytes aufgefüllt. Die sich so ergebenden 24 Bits werden in vier Gruppen zu je sechs Bits aufgeteilt und als Dezimalzahlen zwischen 0 und 63 behandelt. Zu jeder Zahl wird (der Dezimalwert) 32 addiert und das Ganze dann als ASCII-Zeichen zwischen 32 (Leerzeichen) und 32+63 = 95 (Unterstrich) ausgegeben.

Je sechzig dieser ASCII-Zeichen (entsprechend 45 ursprünglichen Bytes) werden als einzelne Zeile ausgegeben, die mit einem M (ASCII-Wert 77 = 32+45) beginnt.

Den n Restbytes, die nach der letzten vollständigen sechziger Zeile übrig bleiben, wird das ASCII-Zeichen vorangestellt, das sich ergibt, wenn man zu der Anzahl der Bytes 32 addiert (n+32).

Schlussendlich folgt eine Zeile, die ein einzelnes Leerzeichen, gefolgt von der Zeichenkette 'end' enthält.

Dieses Verfahren wandelt 0 in ein Leerzeichen und die Werte 1 bis 26 in !"#$%&'()*+,-./0123456789: um. Andere Versionen wandeln 0 in ` und 1 bis 26 in abcdefghijklmnopqrstuvwxyz. Die Rekonstruktion mit "uudecode" funktioniert bei beiden Variationen!

Mache Decoder hängen an jede Zeile ein zusätzliches 'Dummy'-Zeichen an. Dadurch werden Probleme mit Mail-Programmen vermieden, die Leerzeichen am Zeilenende abschneiden. Solche 'Dummy'-Zeichen 'übersieht' uudecode.

In eigener Sache:

Aus gegebenem Anlass bitten wir die Mitglieder der FG, dem Forthbüro Adressen- und/oder Anschriftenwechsel und Änderungen der Telefonnummern mitzuteilen. Ohne diese Informationen kann die rechtzeitige Zusendung der VD nicht gewährleistet werden.

secretary@forth-ev.de
Ute Woitzel

Listing 1 uuencode – uudecode Teil 1

```
{ ============================================================
Umgebungsabhängigkeiten:
 1 CHARS ist 1 Byte
 1 CELLS ist 4 Byte oder größer

      * * * * * * * * * * * * * * * * * * * * * * *
      *                                           *
      * "uuencode"                                *
      *                                           *
      * Systemabhängige Worte (selbst definieren):  *
      * GET-UNENCODED-FILE ( -- str len more )     *
      * PUT-ENCODED-LINE ( str len -- )            *
      *                                           *
      * * * * * * * * * * * * * * * * * * * * * * *
============================================================ }

{ ------------------------------------------------------------
   Häufige Worte. Falls vorhanden auskommentieren
 ------------------------------------------------------------ }

    : SKIP  ( s n c -- s+k n-k )
       >R  BEGIN DUP WHILE  OVER C@ R@ = WHILE
           1/STRING REPEAT THEN
       R> DROP ;
```

Listing 1 uuencode -uudecode Teil 2

```
    : SCAN  ( s n c -- s+k n-k )
        >R  BEGIN DUP WHILE  OVER C@ R@ - WHILE
            1/STRING REPEAT THEN
        R> DROP ;

    : TRIM  ( s n -- s n-k )
        BEGIN DUP WHILE
            1- 2DUP CHARS + C@ 33 - 0<
        0= UNTIL 1+ THEN ;

    : STARTS?  ( str len pattern len2 -- str len flag )
        DUP >R 2OVER R> MIN COMPARE 0= ;

    : STUFF-PAD  ( c -- )  PAD COUNT  DUP 1+ PAD C!  + C! ;
{ ------------------------------------------------------------

FILE-SPAN  enthält die Länge der Binärzeile.

BIN>ASCII  schreibt 6-bit binär als 7-bit Ascii.        ( n -- )

3BIN>4ASCII  enkodiert eine Binärzeile zu 7-bit Ascii.  ( str len -- )

------------------------------------------------------------ }

VARIABLE FILE-SPAN

    CREATE FILENAME  256 ALLOT  S" Untitled" FILENAME PLACE

    : WRITE-ASCII-HEADER  ( -- )
        S" begin 666 " PAD PLACE
        FILENAME COUNT PAD APPEND
        PAD COUNT PUT-ENCODED-LINE ;

    : WRITE-ASCII-LINE  ( -- )
        PAD 1+  FILE-SPAN @ 2 + 4 3 / 1-  ( str len)
        PUT-ENCODED-LINE ;

: BIN>ASCII            ( n -- )
    63 AND
    DUP 27 < IF  64 +  THEN                         \  Variation
    32 +
    STUFF-PAD ;

: 3BIN>4ASCII       ( str len -- )
    2DUP + >R  0 R@ C!  0 R> 1+ C!                  \ Pad vorbesetzen; zwei Nullbytes
    0 ?DO                          ( str)           \ immer 3 Bytes auf einmal
        COUNT 16 LSHIFT >R         ( str+1)( R: x)
        COUNT  8 LSHIFT R> OR >R   ( str+2)( R: xx)
        COUNT R> OR                ( str+3 xxx)( R: )
        DUP 18 RSHIFT BIN>ASCII
        DUP 12 RSHIFT BIN>ASCII
        DUP  6 RSHIFT BIN>ASCII
        BIN>ASCII                  ( str+3)
    3 +LOOP  DROP ;
{ ------------------------------------------------------------

ENCODE-LINE  uuenkodiert eine Zeile.  ( str len -- )
ENCODE-FILE  uuenkodiert eine Datei   ( -- )

------------------------------------------------------------ }

: ENCODE-LINE   ( str len -- )
    DUP FILE-SPAN !                          \  Länge aufheben
    0 PAD !  DUP BIN>ASCII                   \  Zeilenlänge enkodieren
    3BIN>4ASCII ( )
    WRITE-ASCII-LINE ;

: ENCODE-FILE              ( -- )
    WRITE-ASCII-HEADER
    BEGIN  GET-UNENCODED-FILE
    WHILE  ( str len) ENCODE-LINE  REPEAT
    2DROP  ( )
    S"  " PUT-ENCODED-LINE
    S" end" PUT-ENCODED-LINE ;
```

Listing 1 uuencode -uudecode Teil 3

Rätsel: Zahlendarstellung

10001111110
10000101010
10000010100 0100

In Forth sind wir sehr stolz auf die Möglichkeit, mitten aus dem Programm heraus im Zahlensystem von einer Basis zur anderen umschalten zu können. Was heißt das aber schon? Es gibt eine ganze Menge mehr Darstellungsmöglichkeiten, als wir es uns selbst in Forth überhaupt träumen lassen können. Es folgt ein und dieselbe (positive ganze) Zahl in drei verschiedenen Darstellungen. Die Zahl selbst ist uns aus der Geschichte Englands bekannt. Wie lautet sie im dezimalen Positionssystem und welche Zahlendarstellungen wurden verwendet? Welche unmittelbare Kompressionsmöglichkeit gibt es für die dritte Darstellung, sodass die dritte Zahl nur um 1 "länger" wird als die beiden anderen?

Fred Behringer
behringe@mathematik.tu-muenchen.de

```
{ ============================================================

        * * * * * * * * * * * * * * * * * * * * *  *
        *                                          *
        * "uudecode"                               *
        *                                          *
        * Systemabhängige Worte (selbst definieren): *
        *                                          *
        * GET-ENCODED-LINE  ( -- str len more )     *
        * PUT-DECODED-FILE  ( str len -- )          *
        *                                          *
        * * * * * * * * * * * * * * * * * * * *  * *

============================================================ }

{ ------------------------------------------------------------

ASCII>BIN    konvertiert 7-bit enkodiertes Ascii in 6-bit Binärwert.

4ASCII>3BIN  komprimiert 4 Teile 7-bit Ascii in 3 Teile 6-bit Binärwert.

DECODE-LINE  dekodiert eine Zeile ins Binärformat.  ( str len -- flag )

DECODE-FILE  dekodiert eine Datei ins Binärformat.  ( -- )

------------------------------------------------------------ }

: ASCII>BIN  ( n -- b ) 32 -  63 AND ;

: 4ASCII>3BIN   ( str -- str' )
    COUNT ASCII>BIN 18 LSHIFT >R        ( str+1)( R: b)
    COUNT ASCII>BIN 12 LSHIFT R> OR >R  ( str+2)( R: bb)
    COUNT ASCII>BIN  6 LSHIFT R> OR >R  ( str+3)( R: bbb)
    COUNT ASCII>BIN R> OR               ( str+4 bbb)( R: )
    DUP 16 RSHIFT  STUFF-PAD
    DUP  8 RSHIFT  STUFF-PAD
    STUFF-PAD                           ( str+4)
    ;

: DECODE-LINE  ( str len -- flag )
    DUP 0=                  IF NIP  EXIT THEN             \ Leerzeile überspringen
    2DUP S" end" COMPARE 0= IF DROP EXIT THEN             \ Dateiende?
    DROP COUNT ASCII>BIN ( str #chars)                    \ Länge der uuenkodierten Zeile
    DUP FILE-SPAN !                                       \ aufheben
    0 PAD C!                                              \ Pad löschen
    0 ?DO ( str) 4ASCII>3BIN  3 +LOOP                     \ die Zeile uudekodieren
    DROP 0                                ( flag)
    PAD 1+ FILE-SPAN @ PUT-DECODED-FILE ;
```

Listing 1 uuencode -uudecode Teil 4

Gehaltvolles

zusammengestellt und übertragen
von Fred Behringer

VIJGEBLAADJE der HCC Forth-gebruikers-groep, Niederlande

Nr. 22, Oktober 2000

Controlestructuren in Forth I
De Schoolmeester

Prima, was sich der "Schulmeister" da an Erklärungen der Kontrollstrukturen ausgedacht hat! Haben Sie gewußt, daß man ... IF ... ELSE ... THEN auch als ... IF ... AHEAD THEN ... THEN schreiben kann? Natürlich nicht ohne 1 CS-ROLL ! Vonwegen der Strukturiertheit! Endlich mal jemand, der erklärt, wozu man AHEAD (ANS-Forth) gebrauchen kann! Ein sehr lehrreicher Artikel! - Ich (der Rezensent) habe herausbekommen, dass sich hinter dem "Schulmeister" Albert Nijhof, der Redakteur, versteckt. Aber bitte nicht weitersagen!

Draadloze RF-datacommunicatie
Willem Ouwerkerk <w.ouwerkerk@kader.hobby.nl>

A/D-Wandler, Mikrocontroller AT89C2051 und 433-MHz-Datenfunksender. Entsprechend für die Empfangsseite. Kleines Programm in ByteForth.

Nr. 23, Dezember 2000

Controlestructuren in Forth II
De Schoolmeester <a.nijhof@kader.hobby.nl>

Diesmal erklärt der "Schulmeister" dem Anfänger WHILE und DO-LOOP. (... REPEAT entspricht AGAIN THEN ... , DO-LOOP kann über LEAVE vorzeitig verlassen werden ..., man kann in der DO-LOOP auch mit WHILE und IF ELSE THEN arbeiten, hat dann aber UNLOOP einzusetzen ...) CASE hält der Schulmeister in Forth für total überflüssig. "Es wurde höchstwahrscheinlich nur mit der Absicht eingeführt, Quereinsteigern aus anderen Sprachen das Leben leichter zu machen".

Afstandsmeting
Willem Ouwerkerk <w.ouwerkerk@kader.hobby.nl>

Hard- und Softdemonstration zum Einsatz des Infrarot-Sensors GP2D02 von Sharp zur berührungslosen Abstandsmessung. Gibt es bei Conrad für 43 Gulden. (Der Rezensent: Ich dachte eigentlich, daß es da bei Conrad zur berührungslosen Abstandsmessung per Infrarot schon Bausteine für DM 8,50 gibt (?) ...) Demonstrationsaufbau mit dem AT89C2051.

Angeregt durch den von der Zeitschrift Elektor ausgeschriebenen Roboter-Wettbewerb, wird die Fgg im Jahre 2001 einen Roboter mit dem AT89C2051 entwickeln. In ByteForth. Höchstwahrscheinlich mit Platine. Beschreibung schon im nächsten Vijgeblaadje.

Die nächste Jahresversammlung der Fgg findet am 10. Februar 2001 statt.

Niederländisch ist gar nicht so schwer. Es ähnelt sehr den norddeutschen Sprachgepflogenheiten. Und außerdem ist Forth sowieso international. Neugierig ? Werden Sie Förderer der

HCC-Forth-gebruikersgroep.

Für 20 Gulden pro Jahr schicken wir Ihnen 5 oder 6 Hefte unserer Vereinszeitschrift 'Het Vijgeblaadje' zu. Dort können Sie sich über die Aktivitäten unserer Mitglieder, über neue Hard- und Softwareprojekte, über Produkte zu günstigen Bezugspreisen, über Literatur aus unserer Forth-Bibliothek und vieles mehr aus erster Hand unterrichten. Auskünfte erteilt:

Willem Ouwerkerk
Boulevard Heuvelink 126
NL-6828 KW Arnhem
E-Mail: ***w.ouwerkerk@kader.hobby.nl***

Oder überweisen Sie einfach 20 Gulden auf das Konto 525 35 72 der HCC-Forth-gebruikersgroep bei der Postbank Amsterdam. Noch einfacher ist es wahrscheinlich, sich deshalb direkt an unseren Vorsitzenden, Willem Ouwerkerk zu wenden.

```
: DECODE-FILE     ( -- )
    BEGIN  GET-ENCODED-LINE 0= ABORT" Missing `begin` " ( str len)
         S" begin " STARTS? NOT
    WHILE 2DROP REPEAT
    BL SCAN                                      \  hinter begin springen (Leerzeichen)
    BL SKIP  BL SCAN  BL SKIP                    \  dito " 666 ".
    2DUP BL SCAN NIP - FILENAME PLACE    ( )
    ( You may open decoded output file here. )
    BEGIN  GET-ENCODED-LINE 0= IF  2DROP  EXIT THEN ( str len)
        TRIM  DECODE-LINE               ( end)
    UNTIL ;

(
--
Wil Baden   Costa Mesa, California   WilBaden@Netcom.com
)
```

RCX
Wie kommt Forth in den Roboter? - Erste Schritte

Fred Behringer

<behringe@mathematik.tu-muenchen.de>

"Mindestanforderung: Pentium 166, 70 Megabyte freier Plattenspeicher ...", steht auf dem "Robotics-Invention-System-1.5"-Kasten von Lego-Mindstorms. Für das voll automatisch arbeitende graphische Bedien- und Programmiersystem (das aber durchaus nicht alle Möglichkeiten des im Roboter steckenden wundervollen Micro-Controller-Bausteins ausnutzt).

NQC (Not Quite C) von Dave Baum benötigt als ausführbarer Compiler nur einige zig Kilobytes auf dem PC und einen kleinen Editor (z.B. NotePad) zum Erstellen der Programme. Allerdings wird die 32-Bit-Eigenschaft von Windows 95 (im DOS-Fenster) benötigt und NQC ist eine neue, bisher unbekannte Programmiersprache, die eigens für den Roboter erlernt werden muss. Und sie ist nicht interaktiv. Man muss erst das fertige Programm erstellen, es auf den Roboter-Speicher übertragen und kann es erst dann abarbeiten lassen. Jeder kleinste Fehler beim Programmieren erfordert eine Neufassung des Programms.

Forth ist eine bekannte, seit jeher für eingebettete Systeme, also auch für Roboter-Experimente, verwendete Programmiersprache, die für alle Plattformen zur Verfügung steht. Forth ist klein. Forth ist erweiterbar. (Jeder baut sich für seine speziellen Probleme eigene Zusatzbefehle, die er/sie auch nach Jahren noch in Erinnerung behält. Wenn man es denn unbedingt wollte, könnte man sich die Forth-Befehle, die in der Grundausstattung "natürlich" englisch sind, auch sofort selbst eindeutschen. Alles kein Problem.) Forth ist schnell. Forth ist ein Interpreter (ähnlich wie weiland BASIC - jeder neue Befehl kann schnell interaktiv ausprobiert werden) und gleichzeitig ein Compiler (wie C und andere starre nur compilierende Sprachen auch).

pbForth (Programmable Brick Forth) von Ralph Hempel ist ein seit längerem bekanntes Forth für Micro-Controller (hForth), das speziell für die Lego-Roboter um einige Zusatzbefehle erweitert wurde. Der (vorcompilierte) pbForth-Kern ist nur einige Kilobyte lang. Es gibt mehrere Arten, ihn in den Roboter-Speicher zu laden. Von Ralph Hempel wird derzeit noch empfohlen, das über das NQC-Compiliersystem von Dave Baum zu machen. Alle Ein- und Ausgabeoperationen (Definition neuer Forth-Befehle, auch im laufenden Betrieb, Austesten einzelner Programmteile u.v.a.m.) spielen sich über die mit dem Lego-Bausatz mitgelieferte Infrarot-Strecke ab. Die "Firmware" des (programmierbaren und mit Speicher versehenen) Roboter-Controllers wird außer Kraft gesetzt. Ein vorcompilierter Forth-Kern wird über die Infrarot-Strecke vom Computer (Tastatur oder ASCII-Datei) zum Roboter-Controller übertragen. pbForth ist in (PC-)Assembler geschrieben und liegt als Binärdatei für den Roboter-Controller vor (Cross-Compilation).

Ich zeichne im folgenden ein Protokoll meiner ersten Schritte auf, so wie ein Physiker oder Chemiker sein Experiment protokolliert, um später daraus Schlüsse ziehen zu können. Das Herunterladen bestimmter Programmteile vom Internet braucht natürlich nur einmal ausgeführt zu werden.

(1) Im Buch "Lego-Mindstorms-Roboter" von Dave Baum steht vieles über den Zusammenbau der Roboter und über NQC. Außerdem sind 11 wichtige Internet-Adressen angegeben.

(2) Die dem Buch beiliegende CD scheint nur ab Windows 95 lesbar zu sein. Im Verzeichnis "NQC Documentation" finden sich die Dateien "NQC Programmer's Guide.pdf" und "NQC User Manual.pdf". Im Verzeichnis "AcrobatReader" findet sich ein Leseprogramm für die pdf-Dateien. Die Datei "nqc.exe" befindet sich im Verzeichnis "Tools\NQC for Windows".

(3) Die Informationen aus (1) und (2) scheinen sich auch unter http://www.enteract.com/~dbaum/nqc/ zu finden. Ich habe es nur oberflächlich, nicht im einzelnen, nachgeprüft, da ich mit den Dateien aus dem Buch vollauf zufrieden war. Will man in Forth programmieren, braucht man über NQC überhaupt nichts zu wissen. Man missbraucht den NQC-Compiler von Dave Baum lediglich dazu, das pbForth-System in den RAM-Speicher des RCX-Bausteins zu bringen. Wichtig hierzu ist einzig und allein die Datei nqc.exe .

(4) Die pbForth-Site ist http://www.hempeldesigngroup.com/-lego/pbFORTH/.... (kein Trennungsstrich zwischen / und 'lego' *MB)* Interessant sind die Informationen unter den Mausklicks "pbForth FAQ" , "pbForth Tutorial" (pdf-Datei) und "Saving pbForth Systems - How To" . Die beiden wichtigen Controller-Binärdateien "pbforth.srec" (Minimalsystem) und "pbmax.srec" (Maximalsystem) findet man über den Mausklick "current version of pbForth", in einer zip-Datei (mit vielen weiteren Hilfsdateien) verpackt. - Diese ganzen Internet-Geschichten laufen bei mir unter Windows 95. Andere Möglichkeiten habe ich nicht untersucht.

(5) Achtung: pbForth unterscheidet zwischen Groß- und Kleinschreibung. Wenn etwas nicht geht, Eingaben überprüfen! (Bei vielen anderen Forth-Systemen werden die Eingaben vor ihrer Weiterverarbeitung erst einmal in Großbuchstaben umgewandelt.) Dass die Worte ("Befehle") des vorgelegten pbForth-Systems in Großbuchstaben eingegeben werden müssen, war für mich nicht selbstverständlich und hat mich etwas Zeit des Herumexperimentierens gekostet.

(6) Ich arbeite für die Zwecke dieses Berichtes mit Windows 95. Das gleich benötigte nqc.exe läuft nicht im DOS 7.0, das beim Windows-Hochfahren im Menü gewählt werden kann. Es benötigt ein "DOS-Fenster" (DOS-Eingabeaufforderung unter Windows 95). Ich habe mir (Sinn wird gleich deutlich) das Command.com , das üblicherweise im Verzeichnis WINDOWS liegt, als Verknüpfung ("Ikone") auf die Bildschirm-Oberfläche (Desktop) gelegt. Dort kann ich es beim Hochfahren (Booten) von Windows, auch ohne Maus, über die Cursor-Tasten anwählen und über die Eingabetaste aktivieren.

Ab dann bin ich in einem "DOS-Fenster", so als wenn es Windows gar nicht gebe. Aus Windows ganz am Schluss dann wieder herauszukommen, ist kein Problem: Mit CTRL-ALT-DEL gelingt das immer, und in keiner Weise "unsauber".

(7) Um jedweder eventueller Zusatzüberlegung aus dem Weg zu gehen, lege ich die gleich benötigten Programme nqc.exe (siehe (2)), pbforth.srec und pbmax.srec (siehe (4)) auf eine Diskette ins Laufwerk A:.

(8) Der Infrarot-Transmitter wird über eine 9-Pin-SubD-Steckbuche an eine serielle Schnittstelle des PCs geschaltet. "Normalerweise" ist dafür COM1 vorgesehen. Es geht auch beispielsweise über COM2. An COM2 (aus persönlich bedingten historischen Gründen mit einer 9-Pin-SubD-Steckbuchse versehen) liegt bei mir die Maus. Ich arbeite an zwei verschiedenen Anlagen, an verschiedenen Stellen im Haus, mal hier, mal dort. Ich spare mir das Aufschrauben der Rechner, um herauszubekommen, welche COMx-Schnittstelle denn an welchem Rechner wo gerade frei ist, welche Verbindungsleitung von der betreffenden Einsteckkarte an welche PC-Blende gelegt wurde, welche Steckbrücken (Jumper) umgesteckt werden müssen und ob ich nicht einen Adapter brauche. Ich ziehe einfach die Maus aus der COM2-Verbindung und stecke den Infrarot-Transmitter an (449 Mark, aber keine Feststellschrauben!). Windows 95 werde ich (weiter unten) ohne Maus hochfahren.

(9) 9-Volt-Batterie in den Infrarot-Transmitter legen und Transmitter an den PC stecken (siehe (12)).

(10) 6 Mignon-Zellen in den RCX-Baustein legen. Aus bestimmten Gründen "empfiehlt" es sich laut "Handbuch" (Händchenbuch), Akkus zu verwenden. 6 Akkus ergeben normalerweise 7,2 Volt. 6 Batterie-Zellen ergeben, ebenfalls normalerweise, 9 Volt. Und das sollte nichts ausmachen? Glauben will ich das gern, aber das muss man mir erst einmal erklären! Ich verwende für das hier zu beschreibende erste Experiment nicht-wiederaufladbare Batterie-Zellen (aus der Wühlkiste von Conrad-Elektronik).

(11) PC anschalten und (über den Bootmanager und das Windows-Menü) Windows 95 hochfahren. Das System beschwert sich darüber, dass es keine Maus vorfindet. Per "[ret]" ignorieren. Mit den Cursor-Tasten auf das Sinnbild ("Icon") der DOS-Eingabeaufforderung ("MSDOS-Prompt") gehen und über die Eingabetaste ("Return") aktivieren.

(12) Ich platziere den Infrarot-Transmitter etwa 15 cm vor das Gegenstück am RCX-Baustein und setze den Schalter auf "kurze Reichweite".

(13) Ich schalte (über die mitgelieferte Steckverbindungsschnur) einen Motor an den Ausgang A des RCX-Bausteins (siehe Handbuch). Die Drehrichtung ist mir egal. Sobald sich der Motor nach Eingabe eines entsprechenden Forth-Befehls bewegt (siehe unten), betrachte ich meine Aufgabe als erledigt. Die "ersten Schritte" sind dann getan.

(14) Zum unmittelbaren Ausprobieren, ob der RCX-Baustein mit dem angeschlossenen Motor funktioniert, folgendes tun: Rote Taste "On" drücken. Es ertönt ein zweimaliges Piepsignal. In der Anzeige sieht man ein stehendes Männchen und die Ziffer 1, die anzeigen soll, dass das Programm 1 angeschaltet ist. Notfalls per grauer Taste "Prgm" korrigieren. Grüne Taste "Run" drücken. Es ertönt wieder ein zweimaliges Piepen und der Motor dreht sich. Erneut auf "Run" drücken, und der Motor kommt abrupt zum Stillstand. Rote Taste "Off" drücken, und der RCX-Baustein ist abgeschaltet. Zum totalen Abschalten kann man später vorsichtshalber (mindestens) eine Batterie-Zelle herausnehmen.

(15) Computer ist eingeschaltet (siehe (11)). Diskette mit nqc.exe usw. in Laufwerk A: einlegen.

(16) Zu Laufwerk A: gehen über

```
a:      [ret]
```

(17) RCX-Baustein durch Drücken der roten Taste "on" einschalten.

(18) Folgendes eingeben:

```
nqc -SCOM2 -firmware pbforth.srec      [ret]
```

(19) Alles dies geht bei mir auch mit:

```
nqc -firmware pbforth.srec      [ret]
```

wenn der Transmitter an COM1 liegt. (Die Maus darf dann - bei mir - ruhig an COM2 bleiben.) Für einen solchen Nebenversuch habe ich mir extra einen Mausadapter (9 Pin zu 25 Pin) gekauft.

Am Bildschirm erscheint nach der Eingabe unter (18) die Meldung "Downloading firmware:" und es werden während des Lagevorgangs etwa drei Minuten lang in gleichmäßigen Abständen Punkte ausgegeben (bei mir sind es 57). Abgeschlossen wird der Ladevorgang mit einem akustischen Signal vom RCX-Baustein.

(20) Auf dem Bildschirm erscheint die Meldung "No reply from RCX". "Das ist ganz normal", wie mir Martin Bitter auf Anfrage mitteilte. Der NQC-Compiler arbeitet mit einem ganz anderen Übertragungs-Protokoll als der RCX-Baustein. Für die weitere Kommunikation mit dem RCX (vom PC aus) benötigt man ein "Terminalprogramm". "Irgendeins". Was, Sie wissen nicht, was ein Terminalprogramm ist? Macht nichts! Wusste ich auch nicht. Zu DOS-Zeiten habe ich kaum etwas über die COM-Schnittstelle laufen lassen. Das hat mich aber jetzt sehr viel Zeit gekostet. Wer konnte denn ahnen, dass der Übertragungs-Server (Lauschen - Frage - Antwort) nicht schon im NQC-Compiler enthalten ist? Das eigentliche pbForth-System wird ja vom NQC-Compiler-und-Server auch anstandslos zum RCX übertragen. Aber acht E-Mails mit unserem Roboter-Experten Martin Bitter (der Martin weiß wirklich viel) haben schließlich die Aufklärung gebracht. Ich verwende das von Windows 95 mitgelieferte Programm HyperTerminal. (In meinem Transputer-Forth-System, das ja

auch eine Host-Target-Übertragung und einen entsprechenden "Server" benötigt, habe ich das alles in Forth (auf beiden Seiten der Übertragungsstrecke) schnell selbst erledigt - und mir keine Gedanken über ein Terminal-Programm "von der Stange" gemacht.)

Das HyperTerminal-Programm befindet sich bei mir in h:\Programme\Zubehör\HyperTerminal\ . Ich gebe zum Aufrufen von HyperTerminal jetzt ein:

(21) h:\progra~1\zubehör\hypert~1\hypertrm [ret]

Jetzt erscheint das Konfigurationsmenü des Programms HyperTerminal. Es ist nicht einfach, das HyperTerminal-Programm ohne Maus zu bedienen. Es geht aber mit einiger Überlegung und einigem Geschick. Es sind folgende Einstellungen vorzunehmen:

In die Fensterzeile für "Name"

(22) rcx [ret]

Es erscheint ein Konfigurationsmenü mit vier einzeiligen Fenstern. (Fast) alles ignorieren.

(23) Die Tab-Taste drücken.

Die Zeile "Verbinden über" sollte jetzt blau unterlegt sein.

(24) Mit der Cursor-abwärts-Taste zweimal klicken, bis "Direktverbindung über COM 2" erscheint.

(25) Return-Taste drücken.

Es erscheint ein weiteres Menü. Entsprechend der Vorgehensweise von eben die folgenden Einstellungen vornehmen:

(26)	Bits pro Sekunde:	2400
	Datenbits:	8
	Parität:	Keine
	Stopbits:	1
	Protokoll:	Hardware

(27) Return-Taste drücken.

Jetzt ist das HyperTerminal-Bildfeld für Forth-Kommandozeilen-Eingaben (fast) bereit. Eingeben:

(26) hi [ret]

Wenn "Salat" kommt, Eingabe wiederholen. Es erscheint die pbForth-Begrüßungsmeldung und das Bildfeld ist für die Eingabe von pbForth-Befehlen endgültig bereit. Die Bereitschaftsmeldung ("Prompt") von pbForth lautet (wie bei Forth allgemein üblich) "ok".

Achtung: Werden die einzelnen Zeichen von der Tastatur aus zu schnell eingegeben, so kann sich das System "verschlucken". Eingabe dann wiederholen. Außerdem müssen die (einem erfahrenen Forth-Benutzer) bekannten Forth-Befehle in Großbuchstaben eingegeben werden (also DUP, und nicht Dup oder dup).

Bevor irgend etwas Vernünftiges gemacht werden kann, muss der RCX initialisiert werden. Hierzu eingeben:

(27 RCX_INIT [ret]

Die drei Befehle, die ich gleich ausprobieren möchte, verlangen ihre Parameter (so, wie ich sie eingeben werde) als Hexadezimalzahlen. Ich veranlasse also zunächst einmal das System, in den Hexadezimal-Modus umzuschalten. Hierzu gebe ich ein:

(28) HEX [ret]

Jetzt probiere ich einen Sound-Befehl aus:

(29) 3 4003 SOUND_PLAY [ret]

Es ertönt ein anschwellender Akkord. Und nun zum Motor:

(30) 7 1 0 MOTOR_SET [ret]

schaltet den Motor am Ausgang A an (im Weigerungsfall Anschluss überprüfen!).

Und (hart abgebremst) abschalten:

(31) 7 3 0 MOTOR_SET [ret]

Das darf für die vorliegenden Zwecke reichen. Zum Verlassen von HyperTerminal:

(32) ALT-Taste drücken.

(33) Mit der Cursor-abwärts-Taste auf "Beenden" gehen.

(34) Die Return-Taste drücken.

Es erscheint ein Fenster mit der Meldung "Es besteht noch eine Verbindung! Verbindung unterbrechen?" und der Ja-Knopf ist aktiviert (wenn nicht, Cursor-Taste betätigen).

(35) Durch Drücken der Return-Taste "Ja" bestätigen.

(36) Im Fenster "Sitzung rcx speichern?" mit der Cursor-rechts-Taste auf "Nein".

(37) Return-Taste drücken.

Jetzt befindet man sich wieder auf der Windows-Oberfläche (dem "Desktop"). Von da aus ohne Maus herauszukommen, ist gar nicht so einfach. Aber mit CTRL-ALT-DEL gelingt das immer.

Weiteres zu pbForth findet man in der unter (4) angegebenen Quelle.

Fazit: So geht es (noch) nicht. Das erinnert mich alles an die C16-Zeiten. Das hatten wir alles schon mal. Es ist wirklich

nicht einzusehen, warum ich erst "einundzwanzig, zweiundzwanzig, dreiundzwanzig" zählen müssen soll, bevor ich den nächsten Buchstaben per Tastatur eingeben darf, weil im Nichtbeachtensfall nichts Gescheites herauskommt. Aber das läßt sich (in unseren Zeiten der superschnellen Rechnern) ganz bestimmt ändern. Davon bin ich überzeugt. Vielleicht habe ich ja auch nur ganz einfach etwas falsch gemacht. Man müsste halt von seiten der Forth-Gesellschaft Anstrengungen bündeln und mit dem Erfinder von pbForth, Ralph Hempel, eng zusammenarbeiten. Martin Bitter, der die Lego-Roboter-Idee bei uns publik gemacht hat, tut das seit nun fast einem Jahr. Unterstützen wir Martin weiter!

Over the Big Teich

Part 1 (October)

Lieber Friederich,
hier bin ich wieder, 2 Monate älter und nicht viel weiser. Wie ich hörte, waren zum Septembertreffen des SVFIG, John Hall, der den Aplle G4 Cube vorführte, und Dwight Elvey da, der sein letztes Projekt zur Verjüngung eines NIC-80 Mini-Computers vorstellte. Dr. Ting weilte in Taiwan und war zu unserem Treffen am 28. Oktober wieder zurück. Er füllte den größten Teil des Vormittags mit einer Fachbesprechung über die Umformung des alten eFORTH-Systems in die neue P16-basierte Architektur. Er arbeitet mit einem Xilinx VCX300 FPGA Board um einen Chip zu entwerfen, der 32 Befehle haben wird und ein Forth mit 2 zusätzlichen Stacks abarbeiten kann. Soweit ich verstanden habe, hat das mit der Konkurrenz zu den alten 6502 Mikroprozessoren, die weltweit noch verbreitet im Einsatz sind, zu tun.
Unser Sonderseit beim Oktobertreffen war Wil Baden, auch bekannt als "Neil Bawd". Er sprach über all die "Goodies" die er im Laufe der Jahre beim "erforschen" von Forth entworfen hat.
"Goodies" sind nützliche Werkzeuge, populäre, wünschenswerte Computer-Routinen; "Erforschen" bedeutet für ihn "damit herumspielen". Unter diesen Dingen war u.a. sein "Differential File Compare", welches er 1976 ursprünglich in FORTRAN schrieb und welches er jetzt in FORTH und HTML übersetze, so dass man es, wie die anderen "Goodies" als Public Domain über die Web-Seite home.earthlink.net/~neilbawd erhalten kann. Das ist der Platz, an dem man die Früchte der Arbeit aus der Werkstatt eines Mannes sehen kann.
Bis zu 23 Leute zählte ich an diesem Tage. Ein anderer Oldtimer, der uns mit seinem Besuch beehrte, war Bill Ragsdale. Er kam mit einer Dia-Vorführung über eine FORML-Konferenz in China in 1884 (oder doch 1984? der Sätzä). Etwa 17 Personen aus den USA nahmen damals daran teil und die Konferenz hatte insgesamt etwa 150 Teilnehmer. Zum Erstaunen und zur Freude der Einheimischen hielt Wil Baden seinen Vortrag in Mandarin. Er sagte heute, dass er nicht mehr die Courage hätte, dies nocheinmal zu tun. Ich denke, dass es nicht nur mein Eindruck ist, dass sich die Forth-Gemeinschaft aus einer Reihe äußert talentierter, intelligenter und kreativer Leute zusammensetzt.
Am Tag nach dem SVFIG-Treffen erreichte mich ein netter Brief von John Hall, stellvertretender Präsident der FIG, in dem er mir meinen Check zurücksandte und mir mitteilte, dass der Beitrag momentan nicht erhoben wird, da die FIG nicht in der Lage ist, eine Zeitschrift herauszugeben oder Bücher zu liefern. Ich sehe einen Hoffnungschimmer von Reorganisation, durch die Einladung die FIG Web-Seiten auf http://www.forth.org für weitere Ankündigungen zu verfolgen. Ich weiß, dass John, der seinen Enthusiasmus schon während der früheren Präsidentschaft gezeigt hat, die geeignetste Person ist, die FIG aus der gegenwärtigen "Niedergeschlagenheit" zu ziehen.
Unser nächstes SVFIG-Treffen steht schon kurz bevor (nächsten Sonnabend), wegen der Ferien zum Thanksgiving (Erntedankfest) und der erwarteten jährlichen FORML-Konferenz. Aber ich hörte gerade, dass dieses Jahr keine FORML stattfindet. Ist das ein weiteres trauriges Zeichen natürlicher Alterung und Schwunds?
Gaudeamus igitur, iuvenes dum sumus!
Henry.

Ein Lehr-Stück?

von Jörg Staben, Langenfeld

Stichworte:

Snake, Lehrstück, Win32for

Ziele der Zukunft – Oder: Wie alles anfing...

1984 – Orwell´s Jahr brachte uns glücklicherweise noch nicht Big Brother; das kam erst 15 Jahre später auf RTL2. 1984 brachte uns nach dem PET von Commodore eine Welle von 64K Heimcomputern auf Basis der Z80 und 6502 Prozessoren: C64, MSX, Atari und wie sie alle hießen.

Ich hatte einen ORIC-64 und ein Forth auf Cassette. Die anderen hatten BASIC und Assembler und die ersten Spiele in Blöckchengrafik tauchten auf. **SNAKE** wurde erfunden und wie beliebt das Spiel auf dem Handy ist, kann man jeden Tag in Bussen und S-Bahnen beobachten.

Da sind wir wieder:

Wenn wir allgemein zugänglich, d.h. im Internet auf einer Homepage den Spaß am Programmieren und die Grundlagen des Programmierens in Forth vermitteln wollen, müssen wir den Leuten was zeigen und die Leute wollen dabei Spaß haben.

Also: *SNAKE* als Programmierprojekt im Internet-ANS-Forth-Kurs

Und das Programm nicht mit DUPpel, SWAPpel, Pröppel spicken, sondern „schön“ geschrieben – es ist immerhin unser Ausstellungsstück. Und der hoffnungsfrohe Interessent lernt damit die Grundbegriffe von Forth kennen, weil er sich die Forth-Entwicklungsumgebung direkt von der Homepage ziehen kann. Ich bin für Win32For, aus persönlichen historischen Gründen; aber sieht man dann ja, was sich die Besucher herunterziehen. Abstimmung per Download ☺

Flashback:

1984 hatten wir einige Alleinstellungsmerkmale: Forth war zwar nicht portabel, aber im FIG-79 Standard exakt beschrieben; das Blockinterface ermöglichte uns, Massenspeicher unabhängig vom Medium (Band oder RAM) zu begreifen und – Zech wie auch den anderen sei Dank – es gab Literatur über Forth.

15 Jahre sind vergangen – haben wir in Forth noch Alleinstellungsmerkmale ? Mal schau´n.

Interaktivität – der Kommandozeile: Ja.

Interaktivität - allgemein: Können wir vergessen; Interaktivität haben die anderen auch und besser. Wenn man einen Borland C++Builder komfortabel einrichtet, ihm ´n bisschen Plattenplatz für vorkompilierte Header spendiert, ist das Ding schnell, so richtig schnell. Wird sogar in der Statuszeile angezeigt, dass es mal wieder nur 0,08 Sek. gedauert hat. Dann ist auch ein Compiler interaktiv.

Und sobald der Anwender die erste visuelle Komponente aus der Komponentenpalette in das Formular zieht und die Entwicklungsumgebung den C-Quelltext automatisch um den Quelltext der Komponenten ergänzt, stehen wir in der Interaktivität ganz weit hinten. Das ist einfach so.

Kompakte schnelle Programme – Wie gerne denke ich an Tom Zimmer´s 8086-TargetCompiler zurück: Das kleinste Programm in 16 Byte. Gut, hatte aber auch keine graphische Bedienoberfläche, sondern gab den Bildschirminhalt auf dem Drucker aus. Ein Forthprogramm – über savesystem erzeugt – war mit einigen 100KB bei weitem nicht so schlank. Die aktuellen C++Compiler compilieren die üblichen graphischen Übungsprogramme aus Büchern zu etwa 400K Code; ich denke, das ist in Ordnung.

Zudem war eine Entwicklungsumgebung mit Forth-Interpreter und Target-Compiler erst nach mehreren Versionsanläufen einigermaßen zu handhaben. Der Quelltext für die gemeinsame Entwicklungsumgebung aus Forth-Interpreter und Target-Compiler war mit Abhängigkeiten gespickt und das Ganze war genauso wenig transparent wie der Quelltext, den heutige Codegeneratoren als Rahmen für bequeme Windowsprogrammierung erzeugen.

Also: Konzentrieren auf´s Kerngeschäft, die **Interaktivität der Kommandozeile** in der Interpreter-Umgebung in den Vordergrund stellen. Der Rest ist für Spezialisten und kommt später.

Quelltextverwaltung war mal eine Zeitlang eine Domäne von Forth – als wir noch die Blöckchen hatten. Es gab über die **stamps**, das Einbringen von Änderungsdatum und Kürzel des Autors eine rudimentäre Versionskontrolle.

Schade, es gab in Forth immer wieder einzelne Ansätze wie SMan – Hallo, Frank Stüss – oder die ´grenzenlosen´ Blöcke des DeltaT-Forth – Hallo, Ulrich Hoffmann. Keine dieser Lösungen hat Einzug in ein Forth-System mit Breitenwirkung gefunden, es ist in Forth rudimentär geblieben; das Organisieren von Projekten beherrschen die dicken IDE´s von Forte für Java oder Visual Studio tausendmal besser.

Hab´ ich noch was vergessen? Ach ja, von PASCAL und C redete 1984 noch keiner.

Pascal räumte dann als TurboPASCAL den Markt auf und bot den Anwendern die IDE, die Integrierte Entwicklungsumgebung. Von da ab kann man an den TurboPASCAL-Versionen sehr schön sehen, was es braucht, um am Markt erfolgreich zu sein.

Erfolg? Forth verspielte seine Ersterfolge damit, nicht die Erwartungen der Anwender zu erfüllen.

Die Anwender kannten TurboPASCAL, jeder kannte das, und sie wollten den Editor, aus dem heraus compiliert wird. Ob man das braucht oder nicht und wie toll die Kommandozeile ist, entscheidet nicht der Forth-Entwickler – das hat dann der Anwender durch Nicht-Benutzen ganz deutlich entschieden.

Dies macht auch den vergleichsweise großen Erfolg der Entwicklungswerkzeuge aus dem Hause „Tom Zimmer and friends“ aus. Die boten und bieten das, was der Anwender erwartet. Nach dem Anwender haben wir uns zu richten.

Den Anwender interessiert nicht, dass man aus dem Müller-Forth das Meier-Forth metacompiliert hat, um irgendwelchen Heckmeck zu machen. Den Anwender interessiert, ob das System (ein System!) vernünftig und komfortabel zu bedienen ist, der Sprachumfang groß genug für eine bequeme Programmentwicklung ist – kurz: Was hinten ´rauskommt.

Dafür müssen wir uns nach Standards richten, sie kennen und akzeptieren, solange wir selbst keine Standards setzen. Wir müssen mit Standards umgehen können.

Windows ist so ein Standard, Unix/Linux ist auch einer.

Grafik-**Bibliotheken** wie OpenGL und DirectX sind ebenfalls Standard.

In der Forth (neudeutsch: community – Gemeinde klingt so komisch) sind exzellente Einzelleistungen erbracht worden,

die uns eine Zeitlang geholfen haben. Der Anwender erwartet heute aber den Zugriff auf Standards, eben weil da die Dokumentation, die Literatur, das Lehrmaterial vorhanden ist.

Also können und müssen wir zeigen, wie SNAKE seine Töne einer Windows-Standardbibliothek entlockt. Und wenn wir dann noch zeigen, wie SNAKE nicht mehr als Klötzchengrafik umherrennt, sondern bunt und fein in OpenGL daher kommt, sieht der Anwender, dass er oder sie auch Forth nehmen könnte.

Zeigen wir´s nicht, nimmt der Anwender halt was anderes.

Weil die anderen Sprachen und Entwicklungsumgebungen täglich zeigen, dass man´s damit kann.

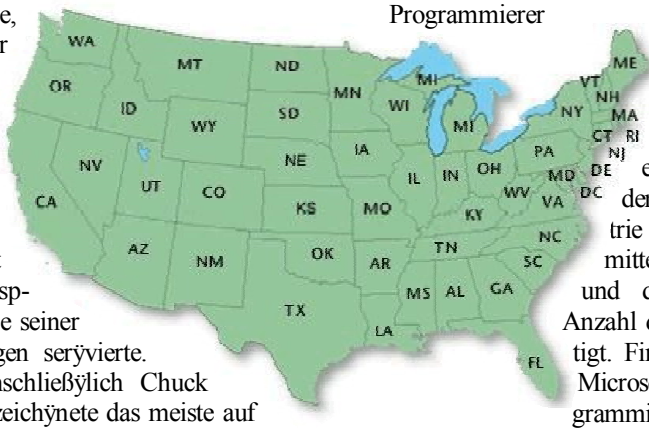
Und das führt dann zur Höchststrafe für Forth: Zum Nicht-Benutzen.

From the Big Teich

Part 2 (November)

Hallo, Friederich,

Ich komme gerade vom SVFIG-Treffen und mir kam die Idee, meinen Bericht sofort zu schreiben. Vielleicht hauptsächlich deshalb, weil das Wetter kalt geworden ist und ich nicht gerade das Verlangen danach habe, einige Dinge außerhalb des Hauses oder in der Garage zu tun. Ja, 10 Grad Celsius um 5 Uhr Nachmittags ist kalt für diesen Teil Kaliforniens im November. Heute war der jährliche Forth-Tag mit mehr als der üblichen Anzahl Redner, die fünf Stunden mit Gesprächen und Präsentaÿtionen füllten. Geselligkeit war dadurch nur während der Mitÿagspause erlaubt, in der Dr. Ting wieder eine seiner traÿditionellen Barbecue-Überraÿschungen seröÿvierte. Es kamen mindestens 30 Leute, einschließÿlich Chuck Moore, des Vaters von Forth. Jeff Fox zeichÿnete das meiste auf Video auf und wird es voraussichtlich auf eine weitere CD brennen, die über seine Website (www.ultraÿtechnology.com) verfügbar ist. Dave Jaffe machte Digitalbilder, die möglicherweise auf der SVFIG-Seite (www.forth.org/svfig) erscheinen werden. Die Redner am Morgen (in Reihe ihres Aufÿtretens) waren Dave Jaffe, Chuck Moore, Kevin Appert, Jeff Fox, Dr. Ting, John Rible, George Perry, John Rible (nochmal), "Neil Bawd" (ein Anagramm des nächsten Sprechers -- Wil Baden) (der sagte, dass man ihn in der Schweiz, knapp unter 47.5 Grad Nord finden kann; s. seine Webseite auf www.home.earthlink.net/~neilbawd), Charley Shattuck, Bob Nash, und schließlich Chuck Moore nochmal, um den Tag mit seinen Neuigkeiten und Gedanken abzuÿschließen. Die meisten Gespräche drehten sich um die Nutzung und die Ideen von Forth, aber ich werde die technischen Details hier nicht ausarbeiten, ich möchte nur einige Ideen von allgeÿmeinem Interesse an Dich weitergeben.



Als erstes, weder einer der früheren Präsidenten, noch der 'geschäftsführende' Präsident der nationalen FIG, kam zu diesem Treffen, Silicon Valley ist sehr lebendig und kann autonom ohne irgendwelche Abhängigkeiten von nationalen Organisationen arbeiten. Es scheint mir, dass solange wir Oldtimer dazu in der Lage sind, wir uns weiter treffen und das genießen werden. Jeff Fox sagte dass die FIG nicht tot ist, auch wenn die Forth Dimensions und der FIG Bücherladen nicht funktionieren. Er befürwortet die Fortsetzung der FIG-Kommunikation über eine Website. Freiwillige könnten, als minimaler Ersatz, für die Erstellung von Artikeln für die Forth Dimensions verfügbar sein. John Rible (der Schatzmeister der SVFIG) berichtete dass die SVFIG immer noch ein Bankkonto hat und die Mitglieder drängt, es weiter zu unterstützen. Und wir haben immer noch Chuck!
Und hier sind einige meiner Notizen von Chuck's Rede:
Forth hat Stacks, hat Definitionen, ist frei von Syntax. Es macht Spaß, da sein Code elegant, lesbar und hübsch sein kann. Zufriedenheit entsteht, wenn man Dinge auf elegante Weise macht. Er schuf Forth, weil er eine einfache Programmiersprache wünschte, die auf vielen Computern laufen würde. Auf eine gewisse Weise gibt es eine Ähnlichkeit zwischen Forth und den File-Komprimierungs-Algorithmen. Datenstücke werden ausfaktorisiert, bis man bei der kompaktesten Lösung des Problems herauskommt. Natürlich, kann man das alles auch in C tun, wie man es mit Forth tut, aber die Leute tun dies eben üblicherweise nicht. Dazu braucht es gute Programmierer, jedoch beginnend in den 70ern, als automatische Compiler verfügbar wurden, hat die Industrie die Tür für eine Menge unausgebildeter Codierer (die man wohl nicht als Programmierer bezeichnen kann) geöffnet. Es ist nicht wirklich billiger, kompetente Leute durch eine größere Anzahl leicht anzuheuernder Arbeiter zu ersetzen, aber das ist die Tendenz in der Industrie. Die Industrie möchte zuverlässige Systeme mittels unzuverlässiger Teile bauen, und daher wird eine sehr große Anzahl der unzuverlässigen Teile benötigt. Firmenphilosophien, wie z.B. die Microsoft's, fordern eine gute Programmierung, aber üblicherweise ist es wichtiger, Termine einzuhalten. Dadurch wurde eine Menge überflüssiger Code über die Jahre erzeugt. Einigen Quellen zufolge hängt eine typische US-Bank von einigen 90 Millionen Zeilen Code ab; die nationale Luftfahrtbehörde FAA von 23 Millionen Zeilen, Windows NT von 40 Millionen und Chuck's ColorForth hat 400 Zeilen Code für ein 6 kByte Objekt. Die Idee sich zurückzuziehen und jemanden anzubieten, seinen Code um zwei Größenordnungen zu reduzieren, klingt verlockend für Chuck, aber auf den zweiten Blick: würde er jemals die Zeit

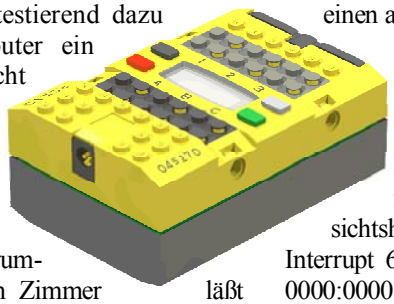
haben, diese Millionen Zeilen Code zu lesen? Es ist wirklich einfacher, neuen Code zu generieren, anstatt alten umzuschreiben. Es ist eine furchterregende Zukunftsvorstellung an alten Code gebunden zu sein. Das ist nicht der Weg der uns vorwärts führt. Vielleicht ist es unsere Mission die Welt davor zu warnen. Vielleicht sollte mehr Forth Code mit eleganten Beispielen auf den Websites verfügbar sein. Denkt darüber nach.

Ich werde mit Appert's Kommentar schließen: "Forth ist wie Sex. Wenn Du keinen richtigen Spaß daran hattest, muss Du etwas falsch gemacht haben."
Bis zum nächsten Mal,
Henry 11 Nov. 2000

ASM2COM über Turbo-Forth: Warum meldet sich der RCX nicht?

Fred Behringer <behringe@mathematik.tu-muenchen.de>

Forth ist maschinennah. maschinennäher ist Assembler. Am maschinennächsten ist Code unter Forth. Die gesamte flexible Forth-Organisation steht zur Verfügung und den Assembler kann man nach Belieben zurechtbiegen. Man braucht sich mit keinen LABELn herumzuschlagen, man kann alle in Forth eingebauten oder leicht einbaubaren Elemente der strukturierten Programmierung einsetzen. (Die vielen "Verbote" im GUI-Konstrukt (Transputer-Assembler) unter OCCAM2 hatten mich seinerzeit protestierend dazu bewogen, mir auch für den Transputer ein Forth zu bauen.) Programmieren macht mir immer erst dann richtigen Spaß, wenn es um Dinge geht, die vermeintlich nicht gehen.



Für viele Anwendungen ist es lästig, den gesamten Forth-Überbau mit sich herumschleppen zu müssen. TCOM von Tom Zimmer läßt den Überbau schrumpfen, COMPACT von der JEDI-Gruppe tut das auch. Am besten geeignet wäre für manche Zwecke eine kleine COM-Datei für die DOS-Kommandozeile. Ich zeige hier anhand eines Beispiels (im Beruf habe ich gelernt, dass Beispiele besser sind als abstrakte Erklärungen - das Abstrahieren kann man dem Hörer oder Leser überlassen), wie man unter Forth (ich darf wieder einmal Turbo-Forth verwenden) schnell eine solche COM-Datei (wenn gewünscht, sogar resident) erzeugt.

Die Dinge sind bekannt, es ist aber nicht leicht, eine Literaturstelle zu finden, wo sie erklärt werden. Wenn davon die Rede ist, dass wir geeignete Forth-Literatur brauchen, denke ich nicht nur an Brodie-ähnliche Tutorials, sondern auch an aufbereitete weiterreichende eigentliche Lösungsvorschläge, die man bei Bedarf schnell übernehmen kann. Hätte ich ein geeignetes Forth-Buch zum "Abschreiben" gefunden, hätte ich mir Arbeit erspart.

Als Beispiel (das den Ausschlag für diese Niederschrift gab) habe ich eine speicherresidente COM-Datei von etwa 600 Bytes gewählt, die dafür sorgt, dass der Infrarot-Transmitter bei den Lego-Roboter-Baukästen sein allfünfsekündiges In-den-Stromsparmodus-Schalten aufgibt. In der RCX-Literatur steht: "Der PC erteilt dem RCX (über den Transmitter) Befehle. Der RCX kann dem PC (siehe Transmitter-Verhalten) keine (unverlangten) Meldungen oder Befehle übermitteln." Nach Aufruf von PULSCOM1.COM (unten) kann er es.

(Übrigens steht im Sensoren-Artikel von Elektor 9/2000: "Dies ist der sogenannte RAW-Wert, mit dem der RCX-Mikrocontroller intern operiert. Der Wert ist in der originalen Mindstorms-Software nicht editierbar, der Zugriff ist aber in den alternativen Programmiersprachen leicht möglich.**" - Forth ist "alternativ"!)**

Das unten stehende Programm, ich nenne es PULSCOM1.FTH, das unter Turbo-Forth per INCLUDE hinzuzuladen ist, erzeugt im laufenden DOS-Verzeichnis eben dieses Programm PULSCOM1.COM, das sich bei Aufruf von der DOS-Ebene aus resident im RAM-Speicher einnistet. PULSCOM1 nimmt den Timer-Tick über Interrupt 1Ch auf, zählt die Ticks bis 50h hoch und gibt dann über das Ausgaberegister der seriellen Schnittstelle COM1 einen nichts (oder zumindest wenig) bewirkenden Code (ich verwende momentan 20h) aus. Dadurch wird der Transmitter für weitere etwa 5 Sekunden "angestoßen". Ein- oder ausschalten kann man diesen Mechanismus von der DOS-Ebene aus über das (ebenfalls über Turbo-Forth erzeugte und nur 6 Byte lange) Programm PULS-E-A.COM, welches lediglich den zuvor geeignet "verbogenen" Software-Interrupt 60h aufruft. Wenn bei einem interessierten Leser der Interrupt 60h nicht frei ist, möge er einen anderen einsetzen.

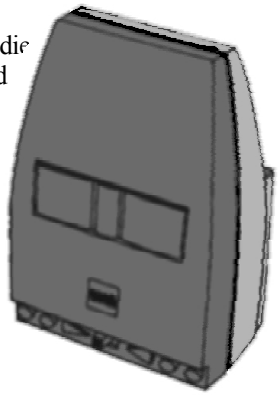
Der Interrupt 1Ch endet auf meinem System mit einer Sprungadresse auf einen Platz im BIOS, an welchem ein IRET-Befehl liegt. Ich habe den IRET-Befehl der Einfachheit halber in unsauberer Weise gleich in mein residentes Programm eingebaut. DS habe ich vorsichtshalber gerettet und anschließend wiederhergestellt. Der Interrupt 60h endet auf meinem System mit der Sprungadresse 0000:0000 , ist also offensichtlich unbelegt. Der Einfachheit halber habe ich auch hier den IRET-Befehl gleich ins residente Programm gelegt.

Das Programm ist noch nicht perfekt, was man erkennt, wenn man auf dem RCX in pbForth beispielsweise die Hex-Schleife **: XXX 700 0 DO I . LOOP ;** ausprobiert. Das Echo vom Transmitter stört. Ich stehe mit Martin Bitter, unserem RCX-Kenner, in reger Diskussion. Wir "arbeiten daran". Es gelingt mir aber auf die aufgezeigte Weise zumindest, auf dem RCX eine Abfrageschleife für die Berührungssensoren ablaufen zu lassen, die erst nach dem Drücken eines Tasters verlassen wird und auf dem Bildschirm die Meldung "Taster 1 gut!" ausgibt. Ohne die hier aufgezeigte Methode eines dem Transmitter verliehenen "Herzschlags" wäre das nicht möglich, jedenfalls nicht, ohne in regelmäßigen Abständen immer wieder eine Taste drücken zu müssen.

Möchte jemand statt der COM1-Schittstelle die COM2-Schittstelle verwenden, so hat er einfach nur die Portadressen 3F8 und 3FD durch 2F8 und 2FD zu ersetzen.

Vorgehensweise:

Aus den untenstehenden Programmen die beiden Dateien PULSCOM1.FTH und PULS-E-A.FTH machen.



Turbo-Forth aufrufen.

INCLUDE PULSCOM1 [ret] eingeben.

Turbo-Forth per BYE verlassen.

Vom DOS-Prompt aus PULSCOM1 genau einmal aufrufen.

Turbo-Forth aufrufen.

INCLUDE PULS-E-A [ret] eingeben.

Turbo-Forth per BYE verlassen.

Per PULS-E-A [ret] kann der Pulsschlag hinfort an- und abgeschaltet werden.

ORG (als Konstante) ist diejenige Offset-Adresse, die zur Zeit des Assemblierens (per INCLUDE xxxxx.FTH) an der Stelle des Hauptlabels herrscht, abzüglich 100h. COM-Dateien beginnen ihre Arbeit (zur Laufzeit) an der Offset-Adresse 100h.

Dass man mit dem hier beschriebenen Beispiel, per Turbo-Forth eine kleine COM-Datei herzustellen und resident in den Speicher zu legen, auch ein ganzes Turbo-Forth-System, einschließlich einer hinzugefertigten Anwendung, resident ins RAM legen kann, ist selbstverständlich. Mit dem hier verwendeten Mechanismus des An-/Abschaltens per Interrupt 60h wird man dann allerdings nicht mehr mit den "normalerweise" nur zulässigen bedingten Sprüngen von nur 128 Bytes auskommen. Mit den in VD 4/99 und 2/00 mitgeteilten Verfahren ist jedoch auch dieses Problem kein Problem mehr. JNEs, BEGIN- und DO-Schleifen und selbst CASE-Konstrukte über 32K im Intel-Code lassen sich ab dem 80386 leicht bewältigen.

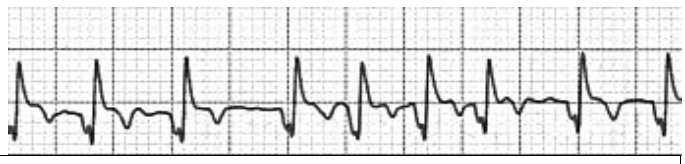
Listing2: PulseCom1 Teil 1

```
HEX

FORTH ALSO
ASSEMBLER DEFINITIONS

: PUSHA 60 C, ;                     \ PUSH AX, CX, DX, BX, SP, BP, SI, DI
: POPA  61 C, ;                     \ POP  DI, SI, BP, SP, BX, DX, CX, AX

FORTH DEFINITIONS

: TO ( <name> n -- ) ' 2+ ! ;       \ legt n in die Konstante <name>
                                    \ nur zur interpretativen Verwendung!
0 CONSTANT ORG
0 CONSTANT ANFANG
0 CONSTANT PULS-E-A

LABEL PULS
   PULS 100 - TO ORG                \ ORG = PULS-100
   PULS 05 + #) JMP                 \ 0100 : auf 0105 springen
   4F  C,                           \ 0102 : Schleifenzaehler-Voreinstellung
   0   C,                           \ 0103 : Ein/Aus-Schalter
   0   C,                           \ 0104 : Initialisierungsindikator
   0104 #) AL MOV  AL AL OR 0<>     \ 0105 : AL = 0, dann ininitalisieren
   IF
      HERE ORG - TO ANFANG          \ ANFANG zur Laufzeit
      PUSHF PUSHA
      DS PUSH
      CS AX MOV  AX DS MOV          \ DS an CS anpassen
      BYTE 0103 #) AL MOV
      AL AL OR  0<>                 \ Nicht 0, dann Mechanismus anschalten
      IF
         BYTE 0102 #) INC
         50 # BYTE 0102 #) CMP 0=   \ ja, dann Transmitter anstossen
         IF
            0 # BYTE 0102 #) MOV    \ Schleifenzaehler zuruecksetzen
            CLI
```

Listing 2: PulseCom1 Teil 2

```
          BEGIN
             3FD # DX MOV            \ Leitungsstatusregister von COM1
             0 AL IN
             20 # AL AND
             AL AL OR  0<>           \ warten, bis bereit
          UNTIL
          020 # AL MOV               \ Zeichen, das wenig bewirkt, ins
          3F8 # DX MOV               \ Empfangs-/Senderegister von COM1
          0 AL OUT
          STI
        THEN
     THEN
     DS POP  POPA  POPF             \ DS und Register zurueckholen
     IRET
\ ---------------------------------
     HERE ORG - TO PULS-E-A         \ Einsprung fuer Ein-/Aus-Schalter
     DS PUSH
     CS AX MOV  AX DS MOV           \ DS an CS anpassen
     0103 #) AL MOV
     0FF # AL XOR                   \ Von FF nach 0 hin- und herschalten
     AL 0103 #) MOV
     DS POP
     IRET                           \ Zurueck aus INT 60
\ ---------------------------------
   THEN
   HERE ORG -\ "Offset" fuer INT 27
   BYTE 0104 #) INC                 \ ab jetzt als initialisiert angezeigt
   ANFANG # DX MOV                  \ INT-1C-Vektor auf ANFANG setzen
   CS AX MOV  AX DS MOV             \ DS mit CS belegen
   25 # AH MOV
   1C # AL MOV  21 INT
\ ---------------------------------
   PULS-E-A # DX MOV                \ INT-60-Vektor auf PULS-E-A setzen
   CS AX MOV  AX DS MOV             \ DS mit CS belegen
   25 # AH MOV
   60 # AL MOV  21 INT
\ ---------------------------------
   # DX MOV  27 INT                 \ bis "Offset" resident machen

PULS HERE SAVE PULSCOM1.COM         \ aufrufbare DOS-Datei erzeugen
```

Und hier das Programm zur Erzeugung der DOS-Datei PULS-E-A.COM , die das An- und Abschalten des Pulsschlagmechanismus gestattet.



Listing 3: Pulse-E-A

```
HEX

LABEL PULS-E-A
   60 INT  0 # AH MOV 21 INT        \ INT 60h aufrufen und wieder zu DOS

PULS-E-A HERE SAVE PULS-E-A.COM     \ aufrufbare DOS-Datei erzeugen
```

From the Big Teich

Part 3 (December)

Lieber Friederich,
Deine letzte Ausgabe der Vierten Dimension für das Jahr 2000 kam hier ungefähr zwei Wochen vor dem letzten SVFIG Treffen dieses Jahres an. Ich war dadurch in der Lage, sie der Gruppe zu zeigen, sie nach ihren Namen in meinen Berichten suchen zu lassen und auch Deine Wertschätzung für Kevin Appert's Späße direkt an Kevin weiterzuleiten. Die Leute waren vom Umfang und Inhalt Eures Magazins beeindruckt, welches von weniger Mitstreitern unterstützt wird, als wir normalerweise Leute auf unseren monatlichen SVFIG Treffen sehen.
Sogar diesen Monat zeigten sich knapp an die zwanzig Leute und ich glaube kaum, dass zu viele kamen, um dem schmerzlichen Weihnachtseinkäufen zu entgehen.
Wie ihr alle wisst, war unsere große FIG nun eine ganze Weile schon nicht mehr in der Lage, eine Zeitung fertig zustellen, und es ist nicht nur eine Frage der Autoren sondern

Bundesgerichtshof
Mitteilungen der Pressestelle

Nr. 49/2000

Microsoft unterliegt vor dem Bundesgerichtshof im Streit um OEM-Vertrieb

Gesonderter Vertrieb für OEM-Produkte urheberrechtlich nicht durchsetzbar

Der u.a. für das Urheberrecht zuständige I. Zivilsenat des Bundesgerichtshofes hat entschieden, daß ein Softwareunternehmen keine Ansprüche gegen einen mit ihm vertraglich nicht verbundenen Händler geltend machen kann, wenn dieser ausdrücklich als OEM-Software gekennzeichnete Ware – also Software, die nur mit einem neuen PC vertrieben werden soll – isoliert an einen Verbraucher veräußert.

Die klagende Microsoft Corporation unterhält für die von ihr entwickelte und vertriebene Software – wie auch sonst in der Branche üblich – einen gespaltenen Vertrieb: Auf der einen Seite bietet sie sog. Fachhandelsversionen ihrer Programme an, die zum isolierten Erwerb durch Endverbraucher bestimmt sind. Davon getrennt vertreibt sie ihre Programme zur Erstausrüstung neuer Computer in einer einfacheren Ausstattung zu einem wesentlich günstigeren Preis. Diese OEM-Versionen (OEM = Original Equipment Manufacturer) werden von hierzu autorisierten Unternehmen hergestellt und entweder unmittelbar oder über Zwischenhändler an die Hardwarehersteller ausgeliefert. Nach den Verträgen, die Microsoft mit den Herstellern sowie mit den Zwischenhändlern und den großen PC-Herstellern schließt, dürfen die OEM-Versionen nur zusammen mit einem neuen PC vertrieben werden. Einen entsprechenden Hinweis läßt die Klägerin auf die Verpackung der Software aufdrucken.

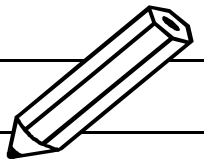
Die Beklagte, ein in Berlin ansässiger Hardwarehersteller, hatte von einem Zwischenhändler OEM-Versionen des Betriebssystems der Klägerin (MS-DOS & MS Windows for Workgroups) erworben. Sie veräußerte ein Exemplar isoliert, d.h. ohne einen PC, an einen Endverbraucher. Die Klägerin nahm sie daraufhin wegen einer Verletzung der ihr zustehenden Urheberrechte an der Software auf Unterlassung und Schadensersatz in Anspruch. Da sie – so ihr Standpunkt – die OEM-Version nur für die gleichzeitige Veräußerung mit einem PC zugelassen habe, sei die von ihr gegebene Erlaubnis zur Weiterverbreitung beschränkt erteilt. Auch der beklagte PC-Hersteller habe nur ein solches beschränktes Nutzungsrecht erhalten und durch den isolierten Weiterverkauf in das der Klägerin zustehende Verbreitungsrecht eingegriffen. Die Beklagte berief sich demgegenüber auf den Erschöpfungsgrundsatz, nach dem ein urheberrechtlich geschütztes Werk – wie ein Computerprogramm – ohne Beschränkung weitervertrieben werden könne, wenn es erst einmal mit Zustimmung des Berechtigten in Verkehr gebracht worden sei.

Mit seinem gestern verkündeten Urteil hat der Bundesgerichtshof – im Gegensatz zu den Vorinstanzen – die Klage von Microsoft abgewiesen. Dabei ist der BGH ohne weiteres davon ausgegangen, daß das in Rede stehende Programm Urheberrechtsschutz genießt. Nachdem das von der Beklagten isoliert vertriebene Exemplar des Betriebsprogramms mit Zustimmung der Klägerin in den Handel gelangt sei, könne diese aber den weiteren Vertrieb nicht mit Hilfe des Urheberrechts kontrollieren. Eine Befugnis des Urhebers, durch eine beschränkte Rechtseinräumung Einfluß auf den weiteren Vertrieb zu nehmen, sei dem deutschen Recht fremd. Der Urheber habe die Möglichkeiten, die Umstände des ersten Inverkehrbringens zu bestimmen. Im Interesse der Verkehrsfähigkeit der Waren sehe das Gesetz dann aber eine Erschöpfung des Verbreitungsrechts vor. Die sachliche, räumliche oder zeitliche Beschränkung der Rechtseinräumung könne die Wirkungen dieser Erschöpfung nicht verhindern, wenn das fragliche Werkstück – wie hier – mit Zustimmung des Berechtigten in den Handel gelangt sei.

Der Bundesgerichtshof hat im übrigen das Argument der Klägerin nicht gelten lassen, sie sei im Interesse der Bekämpfung der Softwarepiraterie auf einen gespaltenen Vertrieb angewiesen. Wenn die Klägerin ihre Programme verbilligt an PC-Hersteller abgebe, um eine Erstausrüstung der PC mit Microsoft-Produkten zu fördern, sei nicht einzusehen, warum nicht auch Interessenten an einer isolierten Programmkopie in den Genuß des günstigeren Preises kommen sollten. Das Interesse eines Herstellers, verschiedene Marktsegmente mit unterschiedlichen Preisen zu bedienen, werde auch sonst von der Rechtsordnung nicht ohne weiteres geschützt.

Urteil des Bundesgerichtshofs vom 6. Juli 2000 – I ZR 244/97 –

Karlsruhe, den 7. Juli 2000



FIGUK

(Englische Forth-Gesellschaft)

Treten Sie unserer Forth-Gruppe bei.
Verschaffen Sie sich Zugang zu unserer umfangreichen Bibliothek.
Sichern Sie sich alle zwei Monate
ein Heft unserer Vereinszeitschrift.
(Auch ältere Hefte erhältlich)
Suchen Sie unsere Webseite auf:
www.users.zetnet.co.uk/aborigine/Forth.htm
Lassen Sie sich unser Neuzugangs-Gratis-Paket geben.
Der Mitgliedsbeitrag beträgt 12 engl. Pfund.
Hierfür bekommen Sie 6 Hefte unserer
Vereinszeitschrift Forthwrite.
Beschleunigte Zustellung (Air Mail)
ins Ausland kostet 20 Pfund.
Körperschaften zahlen 36 Pfund,
erhalten dafür aber viel Werbung.

Wenden Sie sich an:

Dr. Douglas Neale
58 Woodland Way
Morden Surrey
SM4 4DS
Tel.: (44) 181-542-2747
E-Mail: dneale@w58wmorden.demon.co.uk

Danke für Deine persönlichen Bemerkungen, Friederich. Um es klar zu machen, man feiert auch in den USA keine Namenstage. Das ist noch ein Brauch unter den Letten hier. Soweit es Tom Zimmer betrifft... hast Du mich veranlasst, meine Notizen zu durchsuchen. Ich glaube nicht, dass ich ihn seit dem Sommer 1996 gesehen habe, als er und Adrew McKewan nach Austin, Texas, zogen. Und Austin ist weiter weg von hier als Moskau von München... Tempus fugit, Leute verschwinden.
Carpe diem! Frohe Weihnachten und ein frohes Neues Jahr!
Henry

auch des Geldes. Dennoch ist die FIG nicht tot, wie uns John Hall, amtierender Präsident der FIG, mitteilte. Im wesentlichen hat er freiwillig die Aufgaben des Büros übernommen und sucht nach einer besseren Repräsentation der Forth Interest Group im Web. Er sagte uns, dass die FIG eine nicht-kommerzielle Organisation in Californien mit einem Direktoriumsvostand ist, aber im Moment sieht er keine Rechtfertigung dafür Mitgliedsbeiträge zu erheben, da es für die Mitgleider keine reelle Gegenleistung gibt. Um Hilfe bittet er, aber er würde gern ohne "konstruktive Kritik" auskommen. Viel Glück, John!
Wir hatten 3 Sprecher über technische Themen auf dem Dezembertreffen. John Rible sprach wieder über Mikroprozessor Design mit einem anderen Blick darauf wie man die P16 Architektur in FPGA's implementiert. Jeff Fox gab eine Erklärung des historischen Hintergrundes wie ihn unterschiedliche Forthsysteme -- von 1968 bis heute -- zur Entwicklung seines 'Aha', einer neuen Forth-Umgebung für Chuck's F21 Prozessor, führten. Auf Jeff's Webseite WWW.Ultratechnology.com kann man seine neuen Ideen nachlesen, von denen einige schon von anderen Entwicklern rund um die Welt aufgegriffen wurden. Dr. Ting beschloss das Treffen mit einer schnellen Beschreibung darüber, wie und warum er sich in Richtung Win32Forth bewegt und den P24 in FPGA entwickelt (ein 24-Bit-Word Abkömmling des P21). Es gibt einige Leute sowohl in Taiwan, als auch in unserem Land, die am Ergebnis von Dr. Ting's Arbeit interessiert sind.

Win32Forth und Grafik

von Jörg Staben, Langenfeld

Stichworte: Win32for, GDI, OpenGl, 3D-Grafik, 2D-Grafik

2D Grafik

Win32Forth ist als Programmierumgebung für Win32 in der Lage, das Windows GDI (Graphic Device Interface) anzusprechen.

Damit steht Ihnen die Windows-eigene 2D-Grafik zur Verfügung, die Grafiken wie dieses Beispiel von Dave Pochin ermöglicht.

Deutlich umfangreicher sind die Juliamengen des Niederländers Jos v.d.Ven und wecken uralte Erinnerungen an den Fraktalgenerator Fraktint, der in minutenschnelle farbenfrohe Fraktale auf den Bildschirm brachte.

Und genau dies macht das Programm Julia; zu finden unter: www: http://home.wxs.nl/~josv email: josv@wxs.nl

3D-Grafik

Ein korrekt installiertes Windows9x/NT bietet neben DirectX immer auch standardmäßig OpenGL als 3D-Grafikschnittstelle an, den Grafikstandard, der seit dem Erfolg von Quake3 und den GeForce-Grafikkarten wieder in aller Munde ist.

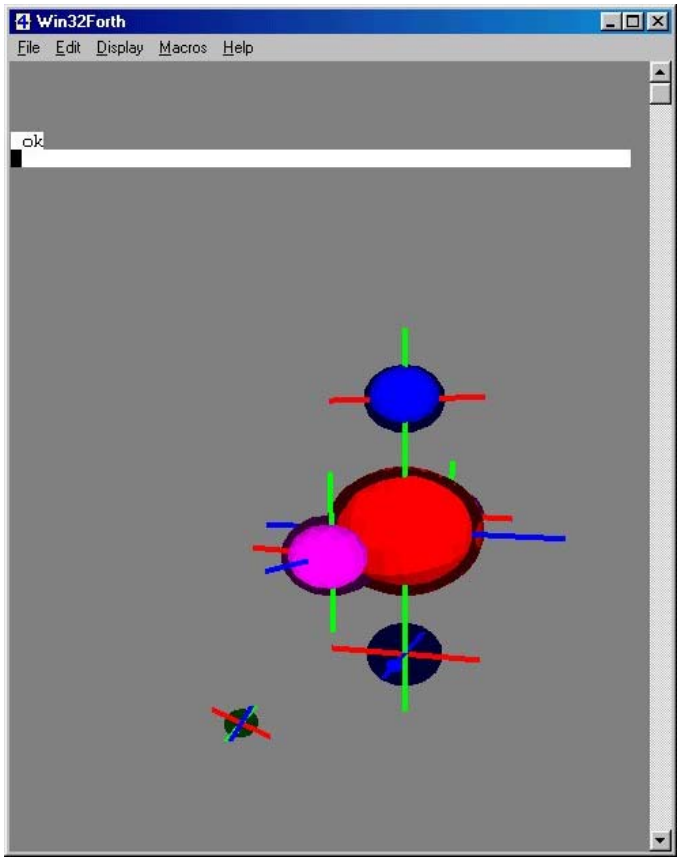
Im Internet kursiert ein kleines Archiv namens SCENE.ZIP, das neben einem Beispielprogramm auch das Forth-File OPENGL.F für den Zugriff auf die OPENGL-Schnittstelle enthält.

Darin findet der interessierte Nutzer diese magischen Zeilen:

```
\ linking to the needed standard openGL Dll's
winlibrary opengl32.dll
winlibrary glu32.dll
```

An der OpenGL-Anbindung sieht man auch deutlich, dass die ganze Diskussion um Bibliotheken in Forth für umsonst war - besonders, wenn man die beiden Zeilen oben und die damit erzielte Funktionalität über eine Grafikbibliothek wie OpenGL mit dem Aufwand und der Mühe vergleicht, mit der ´früher´ Grafikfunktionen für Forth-Systeme implementiert wurden.

Ich denke da an die exzellenten, aber auch sehr aufwändigen Arbeiten von Mark Smiley und Ingo Mathyl, die wirklich extensive Grafik für´s F-PC möglich gemacht haben.



Dennoch zeigt allein schon das kleine Beispielprogramm schnell rotierender Achsen, wieviel mehr Möglichkeiten der Zugriff auf die OpenGL-Bibliothek bietet.

Ein kleiner Wermutstropfen: Irgendwo hat sich einer beschwert, er bekäme auf einem NT-System keinen Code zum Laufen, obwohl in der Win98Box alles einwandfrei ausgeführt wird. Nun gut, Microsoft hat ja auch ´ne ganze Zeit gebraucht, bis D3D-Grafikprogramme sowohl auf NT als auch auf Win98 liefen...

Ach so, OpenGL muss man natürlich können und ein bisschen Windows, aber so isses nun mal. Zu OpenGL gibt es bannig Informationen im Internet, bei 3,5 Pfennig die Minute über Internet-by-call wohl auch kein Thema...

(Da muss ich, Vater zweier Söhne, heftig widersprechen, TK-Gebühren sind nach wie vor ein Thema!

xbigforth von Bernd Paysan enthält inzwischen 'festverdrahtet" einen Menüpunkt 'dragon'. Wenn man darauf klickt, bekommt man (auch unter Windows98!!) seinen berühmten Swapdrachen zu sehen. OpenGL natürlich! bigForth ist (frei) erhältlich: http://www.jwdt.com/~paysan/bigforth.html

MB)

Neulich am Moerser Stammtisch ...

von Martin Bitter
Mehrhoog

Stichworte: Russische Multiplikation, Ägyptische Multiplikation, Hardwaremultiplikation

Vor einiger Zeit erschien Friederich Prinz zu unserem Moerser Forthstammtisch und präsentierte uns folgende 'seltsame' Art der Multiplikation:

313 * 45 = ?

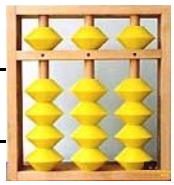
Schritt eins: Man lege eine Tabelle mit drei Spalten an: die erste Spalte heiße M1 (Multiplikand), die zweite heiße R (Rest) und die dritte M2 (Multiplikator).

Nun, das schien nicht besonders schwer:

M1	R	M2

In einem zweiten Schritt trage man Multiplikand und Multiplikator ein, wobei es sich als nützlich erweist, diese so zu vertauschen, dass der kleinere in die Spalte M1 zu stehen kommt.

M1	R	M2
45		313



Für die folgenden Schritte benötigt man 'nur' die Kenntnis des kleinen Ein-mal-Eins mit zwei.

Die Anweisung selbst ist schnell formuliert:

Teile M1 durch 2, merke dir den Rest (0/1) in der Spalte R und notiere das Ergebnis, als 'neuen' Wert M1 eine Zeile tiefer. M2 multipliziere mit 2 und trage das Ergebnis ebenfalls eine Zeile tiefer ein.

M1	R	M2
45	1	313
22		626

Wiederhole diese Schritte, bis M1 gleich Null ist. Die letze Zeile von M2 braucht man nicht auszurechnen.

M1	R	M2
45	1	313
22	0	626
11	1	1252
5	1	2504
2	0	5008
1	1	10016
0		

Streiche im vorletzten Schritt alle Zeilen, in denen R = Null ist.

M1	R	M2
45	1	313
~~22~~	~~0~~	~~626~~
11	1	1252
5	1	2504
~~2~~	~~0~~	~~5008~~
1	1	10016
0		

Letztendlich: Addiere die übrig gebliebenen Zahlen aus M2!

M1	R	M2
45	1	313
~~22~~	~~0~~	~~626~~
11	1	1252
5	1	2504
~~2~~	~~0~~	~~5008~~
1	1	10016
0		14085

Das Ergebnis wurde 'klassisch' überprüft und wir sahen amüsiert und erstaunt, dass es stimmte. Die schnelle Berechnung einiger weiterer Zahlen überzeugte uns vollends: So kann man das auch rechnen.

Die 'Russische Multiplikation'

Inzwischen habe ich mich weiter mit diesem Verfahren, das Fritz uns als 'Russische Multiplikation' bekannt machte, beschäftigt und heraus gefunden (eher: nachgelesen), dass dieser Algorithmus mit jeder Zahlenbasis funktioniert.

Jede Zahlenbasis bedeutet, dass im Schritt zwei eben durch die jeweilige Basis dividiert bzw. mit ihr multipliziert wird. (In den folgenden Beispielen wird aber weiterhin im Dezimalsystem notiert). Die entstehenden M2 werden nun aber mit den jeweiligen Resten multipliziert:

Basis = 5 M1 = 45 M2 = 313

M1	R	M2	M2*R
45	0	313	0
9	4	1565	6260
1	1	7825	7825
			14085

Naja! Mit fünf rechnet sich das ganze nicht mehr so einfach aber schau'n wir mal, wie es mit der Zehn steht:

Basis = 10; M1 = 44; M2 = 313

M1	R	M2	R*M2
45	5	313	1565
4	4	3130	12520
0			14085

Das sieht doch unserer 'traditionellen' (s.u.) Rechnungsart schon recht ähnlich!

3 1 3 * 4 5

	1	2	5	2	
		1 $_1$	5	6	5
1	4	0	8	5	

Der 'Historie' nach soll die russische Multiplikation ehemals von mathematisch ungebildeten, einfachen Leuten in den weiten Landschaften Russlands angewendet worden sein, die das kleine Ein-mal-Eins nicht vollständig beherrschten. Irgendwann muss es auch einmal so einen russischen Bauern nach Ägypten verschlagen haben. Oder wie sonst soll man sich erklären, dass das gleiche Rechenverfahren auch unter dem Namen 'Ägyptische Multiplikation' bekannt ist? Vielleicht kam er ja über die Krim? Die Wege der Mathematik sind oft seltsam!

Wieder andere behaupten, diese Art zu rechnen sei schon 16 Jahrhunderte vor Christus in Indien bekannt gewesen.

Geschwindigkeit ist keine Hexerei

Nun aber zurück zu Fritz. Schon dass die Zahl zwei bei der 'russischen Multiplikation' eine so herausragende Rolle spielt, machte uns hellhörig. Und Fritz spannte uns auch nicht lange auf die Folter: „Mit diesem Wissen kann man einen schnellen Hardwaremultiplizierer bauen.“ Bald war ein Registerschema aufgezeichnet und durchgetestet:

Erklärung: Der Multiplikand wird in ein Register geladen, gleichzeitig wird er (eben per Hardwareverknüpfung) in so viele Spiegelregister kopiert, wie im Hardwareentwurf vorgesehen sind. Die maximale Bitbreite des Multiplikators bestimmt die Anzahl dieser Spiegelregister. Ebenfalls per Hardwareverknüpfung ist dafür gesorgt, dass alle diese Register um eine Stelle nach links verschoben sind. Das kommt einem wiederholten 'Linksshift' gleich. Zur Erinnerung: ein 'Linksshift' entspricht jeweils einer Multiplikation mit zwei.

Senkrecht dazu wird in einem weiteren Register der Multikator geschrieben. Die Auswertelogik verknüpft jetzt das Multiplikandenregister additiv mit den Spiegelregistern, wobei die jeweiligen Register nur berücksichtigt werden, wenn im Multiplikatorregister eine Eins gesetzt ist. Die im Beispiel grau hinterlegten Register werden also nicht mitaddiert.

Solch eine reine Hardwaremultiplikation müsste unschlagbar schnell sein!

Warum haben wir PC-User sie dann (noch) nicht?

Sie benötigt verhältnismäßig viele Resourcen. Pro Bit des Multiplikators ein Register, das ebenso viele Bits umfasst wie der Multiplikand. Dazu noch eine gehörige Menge an Bits im Ergebnisregister, denn die maximale Überlaufmenge muss ebenfalls verkraftet werden. Bei binären Rechnungen ergibt sich die Anzahl der maximalen Ergebnisbits durch eine einfache Addition der Bits von Multiplikator und Multiplikand.

Bei einer Rechenbreite von 64 Bit hieße das: 64 Multiplikandenregister, ein 64 Bit tiefes Multiplikatorregister und ein 128 Bit breites Ergebnisregister (plus ein paar Zwischenregister für die Überträge).
All dies ist ein Aufwand der zur Zeit für den 'Normalanwender' unwirtschaftlich erscheint. Einige Hochgeschwindigkeitsrechner, die für spezielle Aufgaben gebaut werden, verfügen über entsprechende CPUen. Dort nimmt ein Registerarray zur Hardwaremultiplikation bis zu einem Viertel der Chipfläche ein und produziert neben schnellen und genauen Produkten auch noch jede Menge Hitze.

Faszinierend erscheint mir, dass ich – diesmal durch Fritz – ein Rechenverfahren kennen gelernt habe, das auf binärem Denken beruht und schon vor fast 3000 Jahren eine optimale? Lösung für eine heute hochmoderne binär-digitale Maschine vorweggenommen hat.

45 = %101101 313=%100111001

							1	0	1	1	0	1		1
						1	0	1	1	0	1			0
					1	0	1	1	0	1				0
				1	0	1	1	0	1					1
			1	0	1	1	0	1						1
		1	0	1	1	0	1							1
	1	0	1	1	0	1								0
1	0	1	1	0	1									1
				1		1								X
	1	1	1	1	0	1	0	1	1					X
1	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	

(Die beiden mit X gekennzeichneten Zeilen habe ich der Lesbarkeit halber dazu gefügt. Sie enthalten die Überträge der binären Addition.)

1 1 0 1 1 1 0 0 0 0 0 1 0 1 = 14085 (qed)

Meine Achtung vor den(m) 'Ahnen' wächst mit jedem solchen Erlebnis.

Die Leistungsfähigkeit heutiger CPUs ist vor allem durch eine Steigerung der Geschwindigkeit gestiegen. Die Entwicklung der Rechengenauigkeit hat nicht in dem gleichen Maße zugenommen. Es gibt Argumente dafür, dies drastisch zu ändern: Vektorrechner. Das sind sehr vereinfacht gesagt Rechner, die in beliebiger Bitbreite mit Integers bzw. Festkommazahlen rechnen und auf den Geschwindigkeitsvorteil von floting-point Rechnungen verzichten. Sie rechnen ohne Rundungsfehler! Bis (und ob?) sich diese Art von Rechenwerken durchgesetzt

hat, wird es noch dauern. Deshalb muss jeder, der mit wirklich langen Zahlen rechnet (vgl. vierte Dimension 4/2000: Digitale Signaturen ... von Ulrich Hoffman) entsprechende Softwarealgorithmen verwenden. In der nächsten Ausgabe werde ich das 'zweitschnellste' Multiplikationsverfahren für lange Zahlen vorstellen. In gewissem Sinn ist es ein Gegenstück zur russischen Multplikation, wurde es doch erst in den 60er Jahren des 20. Jahrhunderts entdeckt.

Fünf Jahre später – eine Standortbestimmung

von Jörg Staben, Langenfeld

Ich weiß es noch wie gestern: Jörg (Plewe) und ich (Jörg Staben) sitzen im Auto auf dem Weg nach Hamburg zur Echtzeit´94 und natürlich zum Echtzeitwettbewerb. Waren wir aufgeregt...

Der Rest ist Geschichte: Die Aufgabe, ein Teelicht von vier Ventilatoren sich im Kreis bewegen zu lassen; die spannende Endausscheidung; der stolze dritte Platz und die Siegesprämie: Ein ADA-Compiler für MS-DOS.

War das spannend!

Ein halbes Jahrzehnt ist in´s Land gegangen; beruflich hat sich bei beiden Jörgs einiges verändert; Firmen werden schneller verkauft, als man gucken kann und - Programmieren hat sich verändert:

Programmieren ist nicht mehr Forth.

Braucht jemand zufällig einen validierten ADA-Compiler für MS-DOS ? Ehrliche Antwort: NEIN.

Genau so wenig, wie man ein F-PC, ein TurboPASCAL oder ein Quick-C braucht. Denn:

Programmieren ist heute ereignisorientiert und visuell.

Ich weiß, ich weiß, jetzt geht das Gejaule wieder los: „Wenn ich auf meinem Z80 aber in 4 KB RAM ein OpenGL unterbringen will und damit die Münchener U-Bahn steuern muss, dann brauch´ ich aber...“
Quatsch! Gar nix braucht er.

Wie PC´s heute aussehen, wissen wir alle.

Meine Grafikkarte hat viermal mehr RAM als mein 486 vor 5 Jahren und das gesamte F-PC würde komplett im 1st-level cache des Prozessors ablaufen. Die Interaktivität des Entwicklungssystems wird nicht mehr vom Interpreter geleistet, sondern einfach von der Systemgeschwindigkeit.

Die Schnelligkeit, mit der ich Anwendungen entwickele, hängt nur noch von der Schnelligkeit ab, mit der ich die Dokumentation lesen und verstehen kann.

Da sind wir wieder:

Dokumentation muss da sein!

Mein Sohn Fabian, mittlerweile 14 Jahre alt, hat seit fünf Monaten Informatik. Der Lehrer hat die Kids mit TurboPASCAL 6.0 bekannt gemacht. Integrierte IDE, prozedurale Programmentwicklung, einfache Schleifen und so jet. War das romantisch!! Die öffentliche Leihbücherei hat mir übrigens ihr TurboPASCAL-Buch einfach geschenkt, es war in fünf Jahren nur einmal ausgeliehen worden.

Dann hat mein Sohn mich gefragt: „Wenn Jörg Plewe programmiert, sieht das dann aus wie TurboPASCAL?“ Nein! Tut es nicht. Jörg programmiert in einer der beiden aktuellen Sprachen: C++ und Java. Und das sieht anders aus.

Und wenn sich jemand heute für eine dieser beiden Sprachen entscheidet, bekommt er Doku und Werkzeuge vom Feinsten.

Die Werkzeuge, Entwicklungsumgebungen und Compiler, gibt´s an jeder Ecke nachgeschmissen und die Werkzeuge sind gut.

Wir waren in einer Buchhandlung, Ergebnis: ´C++ für Kids´ für keine 50DM. Ein gutes Buch, der C++Builder als visuelle Entwicklungsumgebung von Borland (Inprise) und drei Tage später kannten wir **Forms** und **Properties**, hantierten mit **Labels** und **Buttons** und das Compilieren dauerte keine Sekunde.

FORTE für Java sieht genauso aus wie der Borland Builder und leistet genauso viel. Und Microsoft Visual Studio auch.

Und da sind wir wieder:

Werkzeuge müssen verfügbar und gut sein!

Das ist Fakt:

Für die beiden aktuellen Sprachen C++ und Java gibt es Werkzeuge vom Feinsten und Dokumentation im Überfluss; ein Abend im Internet und man hat, sei es C++ oder sei es Java, mindestens zwei gute elektronische Bücher gefunden, die man wunderbar unter die frei schwebenden Fenster seiner Entwicklungsumgebung legen und beim Programmieren lesen kann. Der darunter liegende finster blickende junge Mann ist übrigen J.C. Denton aus einem Ego-Shooter.

Aktuelle visuelle Entwicklungsumgebungen haben halt eine Projektverwaltung, vervollständigen Codesequenzen im Editor, bieten eine Komponentenpalette und und und... zudem laufen bei uns mp3´s im Hintergrund, wenn programmiert wird. So isses halt.

Am Ende der ersten Woche hatten wir TRY und CATCH durch und freuen uns nun auf Grafik. Natürlich 3D – OpenGL oder D3D. Denn:

Wir wollen was sehen – es muss Grafik sein!

Nun gut, wissen wir jetzt alles.

Wo stehen wir mit Forth heute? Ziemlich weit hinter.

Können wir das ändern? Nein.

Uns gehört nicht SUN Microsystems.

(Für den Kalender: Am 25. Mai 2000 feierte SUN den fünften Geburtstag von Java.)

Wir hätten keine Chance, ein Forth mit einer Projektverwaltung oder einer Komponentenpalette zur Verfügung zu stellen. Sun kann das, Microsoft kann das, Borland vielleicht noch, wir können das nicht.

Was können wir denn tun?

Wir können die Arbeit der letzten zehn Jahre von ´84 bis ´94 retten; gucken, was von unserer Arbeit allgemein gültig ist; uns über ANS-Forth freuen und den Kram öffentlich machen. Ins Netz stellen, auf Resonanz hoffen, aber keine erwarten.

In Forth machen wir dann die **Grundlagen**; gucken, wie's geht; die **Anwendungen** selbst machen wir mit einer der fetten Entwicklungsumgebungen für C++ oder Java.

Ist ja auch nicht schlimm, so lernen wir einfach nur ´ne Sprache mehr.

Ciao

Jörg

Auf den vierten Platz geschlängelt

Robotik-AG der GHS-Borth gewinnt vierten Preis

Martin Bitter
Mehrhoog

Am 20.11. 2000 fand in Westerkappeln bei Osnabrück die Endausscheidung in dem Wettbewerb „Mensch-Natur-Technik“ statt, der von Lego©-Dacta für die beiden Bundesländer Nordrhein-Westfalen und Niedersachsen ausgeschrieben worden war. Schirmherr war MdB Dr. Scheer, die fünfköpfige Jury war mit Medien- und Technikprofessoren besetzt. Als erster Preis winkte eine Rechnerraumausstattung im Wert von über 80.000 DM. Die Preise zwei bis fünf fielen dem gegenüber deutlich ab.

Zu unserer aller Überraschung waren wir, das ist die Robotik-AG der Gemeinschaftshauptschule-Borth, zu dieser Endausscheidung eingeladen. Das bedeutete, wir gehörten nicht nur zu den zwanzig von ehemals siebzig gemeldeten Schulen, die durchgehalten und ein Projekt abgegeben hatten, sondern unser Projekt hatte die Vorauswahl der besten fünf erreicht.

Die Verführerin oder: wie es dazu kam

Gegen Ende des letzten Schuljahres (99/00) erreichte mich die Einladung, mit einer Schüler-AG (AG = Arbeitsgemeinschaft) an diesem attraktiven Wettbewerb teilzunehmen. Die Schüler der damaligen Robotik-AG waren recht erfahren im Umgang mit Legosteinen und der Software zur Programmierung des Lego-Mindstorm Systems (s.h. Kasten) also meldete ich nach kurzer, gemeinsamer Beratung diese AG zur Teilnahme an.

Eine Woche lang überlegten wir, welches Thema wir bearbeiten sollten. Mein Vorschlag: „Wir sind eine integrative Schule, also wäre ein automatischer Rollstuhl ganz nett.“ wurde höflich, aber bestimmt abgelehnt.

Im Internet kann man Videos von Robot-Schlangen bestaunen. Davon waren die Schüler so fasziniert, dass sie beschlossen so etwas auch zu versuchen, trotz meiner Warnung ob des hohen Anspruchs.

Wie gesagt, das war gegen Ende des Schuljahres. Deshalb ließ ich, ließen wir es gemütlich angehen. Ein zwei Schüler bastelten ein wenig an einem Schlangenmodell und erprobten verschiedene Möglichkeiten, Biegungen (Knicke) im Schlangenkörper zu realisieren. Nach und nach schieden erste technische Ansätze aus und zum Beginn des Sommers war noch keine zufrieden stellende Lösung in Sicht. Aber bis Oktober war ja noch sooo weit hin. Dann traf mich langsam die Erkenntnis...

Vertreibung aus dem Paradies

Mitten im Schuljahresendestress (das gibt's! Ehrlich! Auch für Lehrer!) merkte ich, dass mit dem Schuljahr 'meine' Robotik-AG sterben würde. Man möge mir das verzeihen, aber es war das erste mal, dass ich so hohe Jahrgänge unterrichtete. Alle Schüler dieser AG hatten es geschafft, entweder einen Ausbildungsplatz zu bekommen oder in die (besser qualifizierende) 10B zu wechseln. Und die 10B hat nunmal soviel zu lernen, dass für Arbeitsgemeinschaften kein Raum mehr ist.

Die letzten Wochen der Schulzeit vergingen (zumindest vom Robotikstandpunkt aus) wie im Fluge. Für mich stand fest:

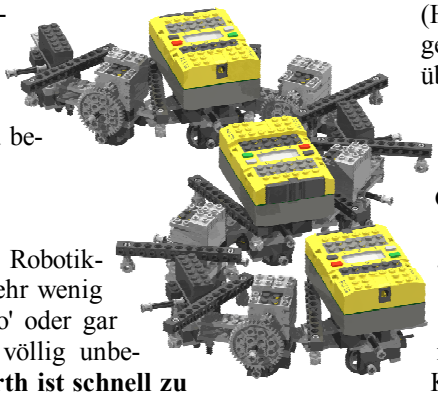
1. Der Abgabezeitpunkt ist ungünstig gewählt. **Nach** den Ferien!

2. Bis Ferienanfang schaffen wir das nie!

3. Die Zeit nach den Ferien, mit neuen Schülern, würde knapp werden.

Womit ich nicht gerechnet hatte: Zwei Schüler hatten Feuer gefangen und so kam es, dass ein Lehrer (ich) und zwei Schüler sich in den Ferien regelmäßig trafen und versuchten ein Schlangenmodell zu bauen. Versuche, mit Zahnrädern und Getrieben in den Segmentgelenken scheiterten. Aber pünktlich zu Schuljahresanfang war das erste Modell, ausgestattet mit 'Sehnen', soweit lauffähig, dass wir Bewegungen erahnen

konnten. 'Großzügig' erlaubten die beiden Schüler der neuen Arbeitsgemeinschaft 'ihr' Schlangenmodell als Ausgangspunkt zu benutzen.

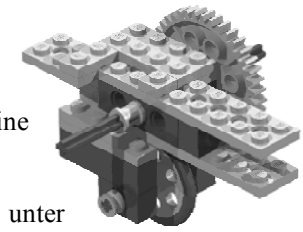


Von Grund auf

Die Mitglieder der neuen Robotik-AG 2000-2001 brachten sehr wenig Vorkenntnisse mit. 'Makro' oder gar 'Batchdatei' waren ihnen völlig unbekannte Begriffe. Aber: **Forth ist schnell zu erlernen und durch seine Interaktivität unmittelbar erfahrbar.** Sobald die Schüler (diesmal alles junge Männer) erst einmal begriffen hatten, wie die PC-seitige Software zu bedienen war, gingen sie mit großem Eifer daran, den RCX-Steinen Reaktionen zu entlocken.

Motorbefehle, rechts – links, Leerlauf und Blockieren wurden gesehen. Unterschiedliche Kraftparameter mit den Fingern gespürt. Parameterübergabe anhand der Soundbefehle (1-6) gehört. Einige wagten sich sogar an Schleifen und einfache Verzweigungen.

Wesentlich schwieriger gestaltete es sich, die Schlange fertig zu bauen und zu steuern. Eine Schlange besteht im echten Leben aus einer Vielzahl von ähnlichen Segmenten (Rippenabschnitte) die gegeneinander beweglich sind. Das sollte auch bei unserem Modell so sein. Doch die 'Vielzahl' von Segmenten musste aus mehreren Gründen auf 'elf' reduziert werden. Zum einen ist die Anzahl der Motoren, die wir haben, beschränkt, zum anderen gilt dies auch für unsere RCX-Steine, von denen ja jeder 'nur' über drei Eingänge und drei Ausgänge verfügt. Und, letztendlich braucht man ja auch Legosteine!



Was uns als erstes ausging? Die Legosteine! Von den Motoren her hätten wir noch vier Segmente mehr bauen können, die RCX-Steine hätten für sechs Segmente gereicht.

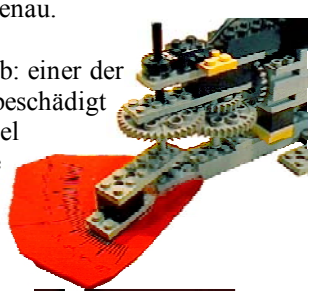
Bald bildete sich ein 'harter Kern' unter den Schülern, der sich fast jeden zweiten Nachmittag in der Schule traf und an dem Modell arbeitete. Je länger die Schlange wurde, umso 'friemeliger' wurde die Arbeit: - sechs, acht Hände auf solch engem Raum – wurde vorne montiert, brach hinten etwas ab.

Obendrein tauchten anscheinend unüberwindbare Schwierigkeiten in der Bewegungsmechanik auf. Der Zwirn, der die Rolle von Sehnen in unserem Modell übernahm, musste irgendwie 'progressiv' aufgewickelt werden (s.h. Kasten). Lange wurde mit Zwangsführungen, Fadenspannern etc. experimentiert, bis der Bruder eines Schülers aus dem Kindergarten 'Gummizwirn' mitbrachte. Das gibt es: Eine Mischung aus Viskose und Gummifaden zum Nähen! Schnell drei Rollen davon gekauft, und diese Sorge war vorbei. Als dann noch durch einen Einbaufehler, die Zwangsführung (Holzperlen) für dieses Gummiband an einem Segment vergessen wurde, stellten die Schüler fest, dass diese jetzt völlig überflüssig war. (Änderungen am Segmentaufbau bedeutet: Schlange in elf Segmente zerlegen, alle elf Segment umbauen, Schlange neu montieren, verspannen und justieren.) Neue Abstandhalter erfüllten den Zweck, die Gummisehnen vor dem Verheddern zu schützen.

Jetzt war die Mechanik funktionsfähig – abgesehen davon, dass Legosteine nicht für so große Zugspannungen und Belastungen ausgelegt sind, wie sie in dem Modell manchmal auftauchten. Hier fanden die Schüler einen guten Kompromiss bei der Programmierung der Motorkraft.

Denn sie wissen nicht, was sie tun ... (oder Messen und Regeln)

Die Winkelgeber, die ich aus Potis so gebaut hatte, dass sie sehr einfach auf Lego-Achsen zu befestigen sind, messen alle unterschiedliche Werte. In Verbindung mit der (internen) Messroutine des RCX-Steines gibt das einen deutlich nicht linearen Verlauf. In der ersten Version hatte ich eine Funktion geschrieben, die ausgehend von der Kirchhoffschen Regel, etwas Mathematik und Informationen aus dem Internet zum Selbstbau von Lego-Sensoren, diese unterschiedlichen Werte 'normalisierte'. Trotzdem erwies sich die Messung und Regelung der Winkel als sehr ungenau.



Eine genauere Untersuchung ergab: einer der Sensoren war beim Umbasteln beschädigt worden. Er hatte zu großes Spiel und lieferte ungenaue Werte (Hysterese). Er wurde durch einen Gott-Sei-Dank vorhandenen anderen ersetzt.

Jetzt maßen die Schüler jeden Poti individuell aus und hielten die Ergebnisse in der Software durch mehrere Tabellen fest. Das hatte den positiven Nebeneffekt, dass die Messroutinen nicht nur genauer, sondern durch die Schüler auch verstehbar wurden.

Wenn die Linke nicht weiß, was die Rechte tut

Jeder RCX-Stein steuert zwei mal zwei Segmente. Um zu koordinierten Bewegungen zu gelangen, bedarf es einer Verständigung der jeweiligen RCX-Steine untereinander. Leider scheiterten hier unsere Versuche, dazu die vorhandenen Infrarot-Schnittstellen zu benutzen. Zu unzuverlässig war dieser Kanal, denn wegen der Schlängelbewegung 'schauen' die RCX-Steine in ständig wechselnde Richtungen.

Der RCX kann seine Ausgänge als Widerstände schalten und die Eingänge ebenfalls auf Widerstände abfragen. Also musste jeder RCX-Stein einen Eingang und einen Ausgang für eine drahtgebundene Kommunikation opfern. Zwar war das Repertoire eingeschränkt (high-low) aber dafür zuverlässig!

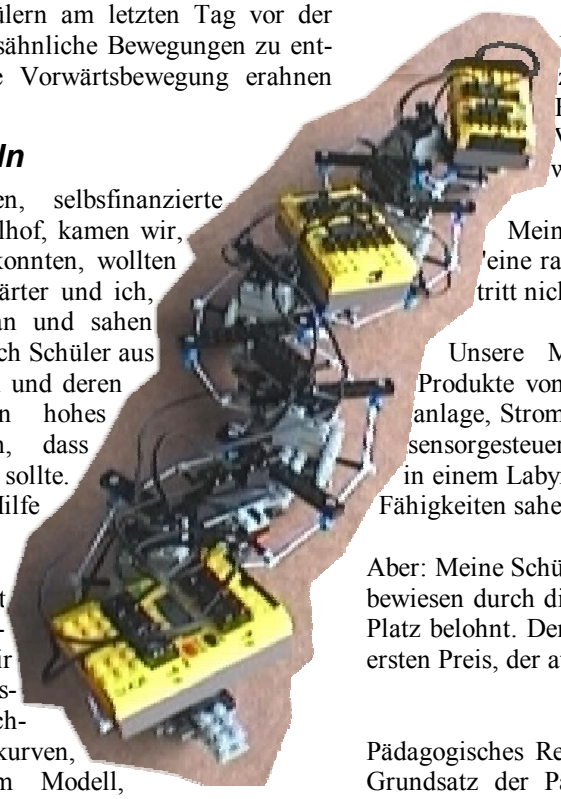
Einzelne Segmente zu einer regelmäßigen Rechts-links-Bewe-

gung (Abknicken im Gelenk) zu veranlassen, war programmtechnisch nicht besonders schwer, hatte man erst einmal den Befehl MOD verstanden. Aber jetzt galt es herauszufinden, in welchem Abstand zueinander die Segmente ihre Bewegungen vollführen sollten. Hier war Phantasie, Vorstellungsvermögen und Experimentieren gefragt, denn für die beteiligten Schüler sind Begriffe wie Sinus, Kosinus oder Schwingung physikalische Fremdwörter.

Immerhin gelang es den Schülern am letzten Tag vor der Abreise der Schlange windungsähnliche Bewegungen zu entlocken, die eine schlängelnde Vorwärtsbewegung erahnen ließen.

D-Day in Westerkappeln

Nach einem frühen Morgen, selbsfinanzierte Abfahrt war 6.30 Uhr ab Schulhof, kamen wir, vier Schüler, die mit fahren konnten, wollten und durften, ein Lehramtsanwärter und ich, pünktlich in Westerkappeln an und sahen unsere Mitbewerber, vornehmlich Schüler aus Leistungkursen an Realschulen und deren Modelle. Ein ausgesprochen hohes Niveau! Und wir erfuhren, dass mündlich vorgetragen werden sollte. Zwölf Minuten lang!!! Ohne Hilfe durch irgendeinen Lehrer!



Das Reden ist nicht so unbedingt die Stärke 'meiner' Robotikschüler. Deshalb hatten wir einen schönen Präsentationsstand mit der Einmessvorrichtung, den passenden Messkurven, großen CAD-Bauplänen zum Modell, ausgewählten Quelltexten und dem Logikdiagramm zur Kommunikation vorbereitet. Den konnten wir zwar gemeinsam aufbauen, aber er wurde (finde ich) unter Gebühr beachtet.

Glücklicherweise bestimmte das Los unsere Gruppe dazu, den Reigen der Vorträge zu eröffnen. Zwar gelähmt wegen der Größe des Publikums, aber tapfer schlugen sich die vier! Das Logikdiagramm wurde in die Kamera der Großleinwand gehalten, ein wenig (viel zu wenig!) über die technischen Probleme gesprochen und dann mit einem (vorbereiteten) Knopfdruck das Modell gestartet.

Unüberhörbar rasselnd entrollte sich die Schlange und begann sich zu winden. Jetzt steigerte sich dass rasselnde Geräusch – wurde lauter – noch lauter! Es gab einen Knacks. Das Modell stand mit laufenden Motoren.

Enttäuschte Rufe aus dem mitfühlenden Publikum. Ich suchte mein Mauseloch, winkte gleichzeitig die Schüler von der Bühne.

Diese liefen jetzt zu ihrer Hochform auf: vor der laufenden Kamera drückten sie erst einmal die Notausknöpfe, begutachteten die Lage und kamen zu dem Schluss: Das läßt sich richten! Sie befestigten einen abgesprungen Motor, justierten einige Zahnräder, montierten Abstandhalter. Dann bediente einer souverän den geliehenen Lap-top, initialisierte die Software und ...

Diesmal klappte es!! Die Schlange wand sich auf der Stelle, mit etwas Phantasie war eine Vorwärtsbewegung zu erkennen. Sie ging kein zweites mal kaputt. Und last not least: Die Schüler waren im Zeitlimit geblieben.

Donnernder Applaus vom Publikum: an die zweihundert Schüler, eine Jury aus zwei Hochschulprofessoren und einigen Vertretern aus der Industrie. Die Stimmung war hervorragend bis euphorisch.

Meine Schüler? Weg! Nach dem Stress, erst mal 'eine rauchen'. Und als Erfolg sahen sie ihren Auftritt nicht. Wir sprachen Mut zu.

Unsere Mitbewerber waren sehr 'eloquent'. Ihre Produkte von hohem Standard: Selbsttätige Tafelwaschanlage, Stromgewinnung aus Meereswellen, verschiedene sensorgesteuerte Greifsysteme und ein Fahrzeug, das sich in einem Labyrith zurechtfindet. Neben ihren sprachlichen Fähigkeiten sahen wir richtig 'alt' aus.

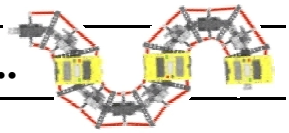
Aber: Meine Schüler hatten Glück. Ihre hohe Sachkompetenz, bewiesen durch die 'Live'-Reparatur, wurde mit einem vierten Platz belohnt. Der dazugehörige Preis war der, der nach dem ersten Preis, der attraktivste war.

:-)

Pädagogisches Resumee: Diese Projektarbeit bestätigte einen Grundsatz der Pädagogik: Prozessorientierung! vulgo: Der Weg ist das Ziel! Wir haben zwar nicht den ersten Preis gewonnen, aber die Schüler (nicht nur die vier, die nach Westerkappeln mitfuhren) haben: freiwillig, über einen langen Zeitraum, außerhalb der Schulzeit, gemeinsam an einem Ziel gearbeitet. Und das ist eine Erfahrung, die nicht nur an Hauptschulen, selten ist?

Lego-Mindstroms

heißt eine Produktfamilie der Firma Lego, die sich zum Ziel gesetzt hat, Kinder spielerisch an das Programmieren von (aus Legosteinen gebauten) Robotern heranzuführen.
Die im Spielwarenhandel für ca. 450 DM erhältlichen Sets enthalten einige Hundert Lego-Techniksteine, sowie einen RCX-Stein. Das ist ein vollständiger Rechner mit Display, vier Tasten, drei Eingängen, drei Ausgängen und einer Infrarot-Schnittstelle.
Mit einer grafischen Oberfläche (RoboLab) kann programmiert werden.
Ich benutze in meinem Unterricht pbForth von Ralph Hempel und eigens für den Schulbedarf zusammengesetzte Sets von Lego-Dacta (LPE).
Weitere Informationen über die Redaktion oder
www.forth-ev.de.



Für weitere Nachfragen stehe ich gerne zur Verfügung. Anschrift s.h. Redaktionsadresse

MB

Die Mitbewerber, ihre Projekte und Preise:

1. Realschule Westerkappeln
   Tafelputzvorrichtung
   Rechnerraum, 10 PC, Flachbild schirme, Peripherie, Netzwerk,

2. Realschule Lathen
   verschiedene **Greifsysteme**
   Solarmodul für den Physikunterricht

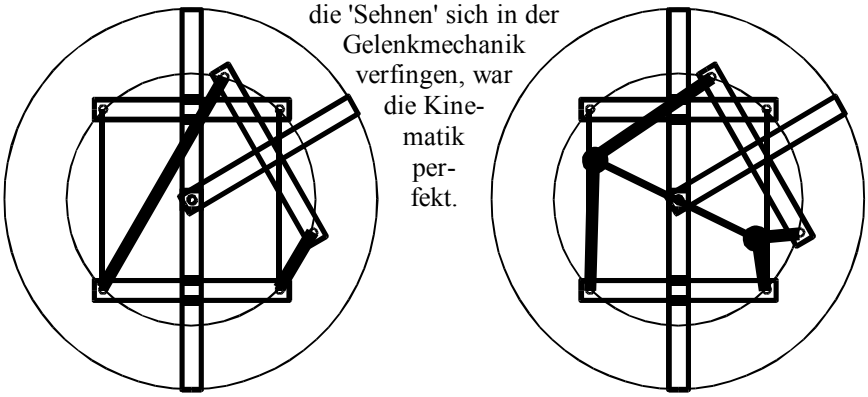
3. NAWI-Haus Oldenburg
   Strom aus Meereswellen
   Tagesfahrt nach Legoland-München

4. Gem. Hauptschule Borth
   Schlange
   4 RCX-Steine und Zubehör,
   1 Klassenset RoboLab

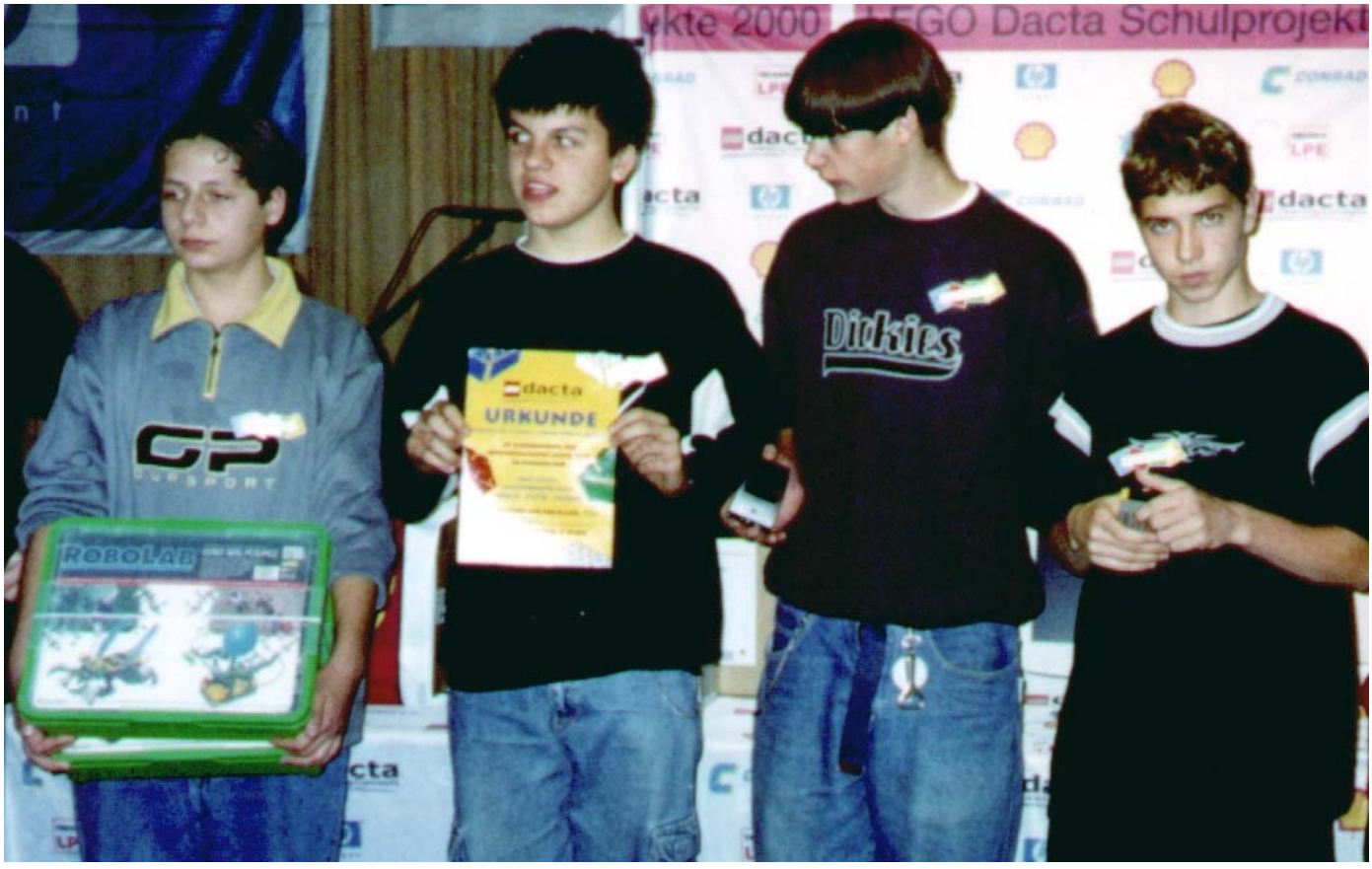
5. Kooperative Gesamtschule Sehnde
   Selbst**orientierendes Fahrzeug**
   Doppelset Legotechnik

Teuflische Geometrie:
Wie die kleine Zeichnung und etwas Überlegungen zeigen, stimmt die Länge der jeweils abzuwickelnden 'Sehnen' nicht mit der Länge der im Moment aufzuwickelnden 'Sehnenabschnitte' überein. Hier können sich Leute, die mit Sinus; Cosinus etc. per Du sind umtun. Wir jedenfalls waren glücklich über die 'Gummibänder'.

Als dann noch zusätzlich Abstandhalter eingebaut wurden, die verhinderten, dass die 'Sehnen' sich in der Gelenkmechanik verfingen, war die Kinematik perfekt.



Die 'Sieger'



Ridvan Sevimli Johannes Wolf Daniel Nürenberg Christoph Drötboom

Brückenbauers Traum

oder

ein echt statisches Spiel.

Von ihrer letzten Netzwerknacht kamen meine Söhne (Sie erinnern sich, die mit den Telefonkosten) mit einem neuen Spiel zurück: „Echt Klasse. Dürfte Dir gefallen, ganz ohne Blut und Ballerei.“

Worum geht's?

Brücken bauen. Nicht mehr und nicht weniger. Ich kenne jemanden, der hat einen Teil seines Lebens damit zugebracht, Brücken zu bauen, und er schwärmt jetzt noch davon.

Bridgebuilder ist mit einer sehr mächtigen Physik ausgestattet. In einem Planungsraster sind Widerlager, Fundamentpunkte und verschiedene Topographien vorgegeben. Es gilt, mit möglichst wenig Stahlstreben über ein Gewässer eine Brücke zu konstruieren.

Das ist schon interessant genug, ist doch der Stahlvorrat (gemessen in Dollar) und die maximale Länge einer einzelnen Strebe beschränkt.

Ist die Brücke fertig geplant, kommt der spannende Teil: sie wird in die Landschaft ge'beamt'. Sofort beginnt sie auf die Belastungen durch die Schwerkraft zu reagieren und biegt sich, Schwingungen bauen sich auf. Hat die Brücke diesen Einschwingvorgang überstanden und hält (noch), muss sie die Belastungsprobe durch eine Güterzug bestehen. Erst danach geht's weiter zum nächsten Level.

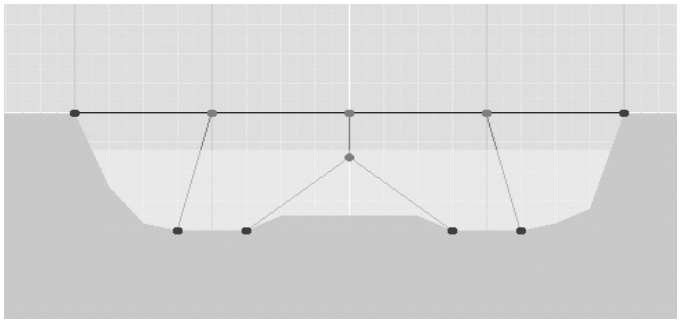
Die berechnete Physik ist gut. Belastungen werden auf Wunsch farbig angezeigt, gebrochene Streben schwingen um Drehpunkte, unter Wasser erfährt der Güterzug Auftrieb und Reibung, manchmal schleudern Bruchstücke durch die Gegend. Kurzum: allerlei 'realer' Unfug ist möglich, soweit physikalisch vertretbar. Statik macht erst richtig Spaß, wenn sie sich in Dynamik (ver)kehrt.

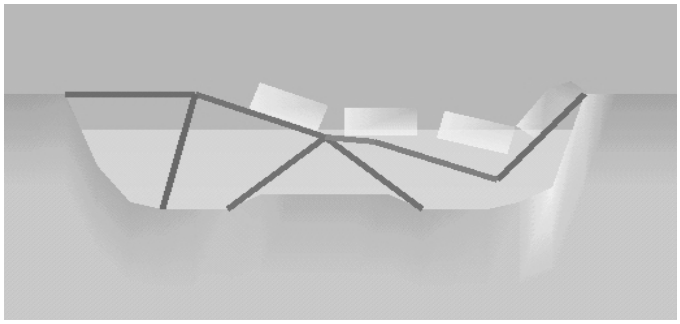
Nicht nur für Menschen mit 'Ingenieurherz' hat das Suchtpotential.'

Einen Wermutstropfen gibt es doch. Die aufwendigen Berechnungen (infinite Elemente?) bremsen gewaltig. Ein schneller Rechner muss also sein. Am Besten noch mit einer OpenGL- Grafikkarte. (Seit Papa nicht mehr den schnellsten Rechner im Hause hat ...läuft's nur beim Sohnemann so richtig.)

Bridgebuilder gib es bei: http://bridgebuilder.8m.com/ (Win95/98/2000/NT)

MB





„Ich habe gehört, dass, wenn man die NT-4.0-CD rückwärts abspielt, eine satanische Nachricht ertönt."
"Das ist gar nichts. Wenn man sie vorwärts abspielt, installiert sie Windows NT!“

Wenn jemand will, dass seine Server von Idioten bedienbar sind, bekommt er was er verdient: Server bedient von Idioten.

Windows -- ist kein Virus. Viren sind klein und effizient.

Die zwei bekanntesten Produkte aus Berkley sind BSD Unix und LSD. Das kann kein Zufall sein.

Rechnen in Binärcode ist so einfach wie 00, 01, 10, 11 ...

Was ist schon der Raub eines OS gegen das Schreiben eines OS? (frei nach B.B.)

Das Kleingedruckte: Die Markennamen 'Windows', 'NT' stehen stellvertretend für eine ganze Produktgruppe. Bitte nach Belieben andere Betriebssystemnamen eintragen. Übrigens: Brauchen sie eine Lupe?

Forth-Gruppen regional

Hamburg **Küstenforth**
Klaus Schleisiek
Tel. 040 / 375008-13 g
kschleisiek@send.de
Treffen jeden 4. Freitag im Monat
16:30 Uhr, Fa. SEND, Stubbenhuk 10
20459 Hamburg

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

Mannheim **Thomas Prinz**
Tel.: 06271-2830 (p)
Ewald Rieger
Tel.: 06239-920 185 (p)
Treffen: jeden 1. Mittwoch im Monat
Vereinslokal Segelverein Mannheim e.V.
Flugplatz Mannheim-Neuostheim

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

µP-Controller Verleih

Thomas Prinz
Tel.: 06271-2830 (p)
micro@forth-ev.de

Gruppengründungen, Kontakte

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

Forth-Hilfe für Ratsuchende

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

Jörg Staben
Tel.: 02173-75708 (p)

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

Spezielle Fachgebiete

Arbeitsgruppe MARC4	Rafael Deliano Tel./Fax: 089 -841 83 17 (p)
FORTHchips (FRP 1600, RTX, Novix)	Klaus Schleisiek-Kern Tel.: 040 -375 008 03 (g)
F-PC & TCOM, Asyst (Meßtechnik) embedded Controller (H8/5xx//, TDS2020, TDS9092),Fuzzy	Arndt Klingelnberg, Consultants akg@aachen.kbbs.org Tel.: ++32 +87 -63 09 89 pgQ (Fax -63 09 88)
KI, Object Oriented Forth, Sicherheitskritische Systeme	Ulrich Hoffmann Tel.: 04351 -712 217 (p) (Fax: -712 216)
Forth-Vertrieb	volksFORTH / ultraFORTH RTX / FG / Super8 / KK-FORTH Ingenieurbüro Klaus Kohl Tel.: 08233-3 05 24 (p) Fax : 08233-99 71 mailorder@forth-ev.de
Forth-Mailbox (KBBS)	0431-533 98 98 (8 N 1) Sysop Holger Petersen hp@kbbs.org Tel.: 0431-533 98 96 (p) bis 22:00 Fax : 0431-533 98 97 Helsinkistraße 52 24109 Kiel

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

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

Hinweise zu den Angaben nach den Telefonnummern:

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

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

Not Yet!

(Nein, aber in der nächsten Ausgabe.)

Last Order!

Schickt Beiträge!

Die nächste Tagung der „Forthgesellschaft e.V.“ steht an.

Jahrestagung 2001
27. bis 29. April 2001

Diese findet mal wieder am schönen Niederrhein statt. In fast schon therapeutisch ruhiger Atmosphäre. Betten, Essen, Tagungsräume, Beamer, OHP und Videoanlagen sind vorhanden. Was leider immer noch fehlt, sind Tagungsbeiträge!
Wie schön ist es doch, von diesem Treffen mit einzigartigen Menschen etwas Gediegenes mit nach Hause zu nehmen.
Welch sinnliches Erlebnis einen **Tagungsband** zu sehen, zu riechen, zu tragen!

Sie haben bestimmt Interessantes zu berichten. Flexible Zuhörer mit hoher Motivation warten auf Ihren Beitrag! Selbst eine Lohnsteuererklärung kann fesseln, wenn sie in oder mit Forth erstellt wird.

Medizintechnik? Prozessorinformationen? Embedded Systeme? Lernsoftware?
Verfahrenstechnik? Neue Forthdialekte? ... Ihr Spezialgebiet? ...

Geben Sie sich einen Ruck! Setzten Sie sich dieses Ziel: Mein Vortrag für die Tagung! Und viele werden es Ihnen Danken (vor allem Ich ;-).

Für den Anfang reicht ein Titel oder ein Thema über das Sie berichten wollen. Teilen Sie mir das mit. Ideal ist natürlich ein Tagungspapier, das bis zum Ende des Monats März bei mir eintrifft.

Letztendlich: Forth-Vortragende: Setzt Euch mit mir in Verbindung: mbitter@bigfoot.de oder

Martin Bitter

Möllenkampweg 1a
46499 Hamminkeln

mit forthlichen Grüßen