ATARI ST COMPUTER

Die Fachzeitschrift für den ATARI-ST Anwender.

November 89

DM 7,- Ös. 56,- Sfr. 7,-

11

DFÜ

Mailboxen & Modems

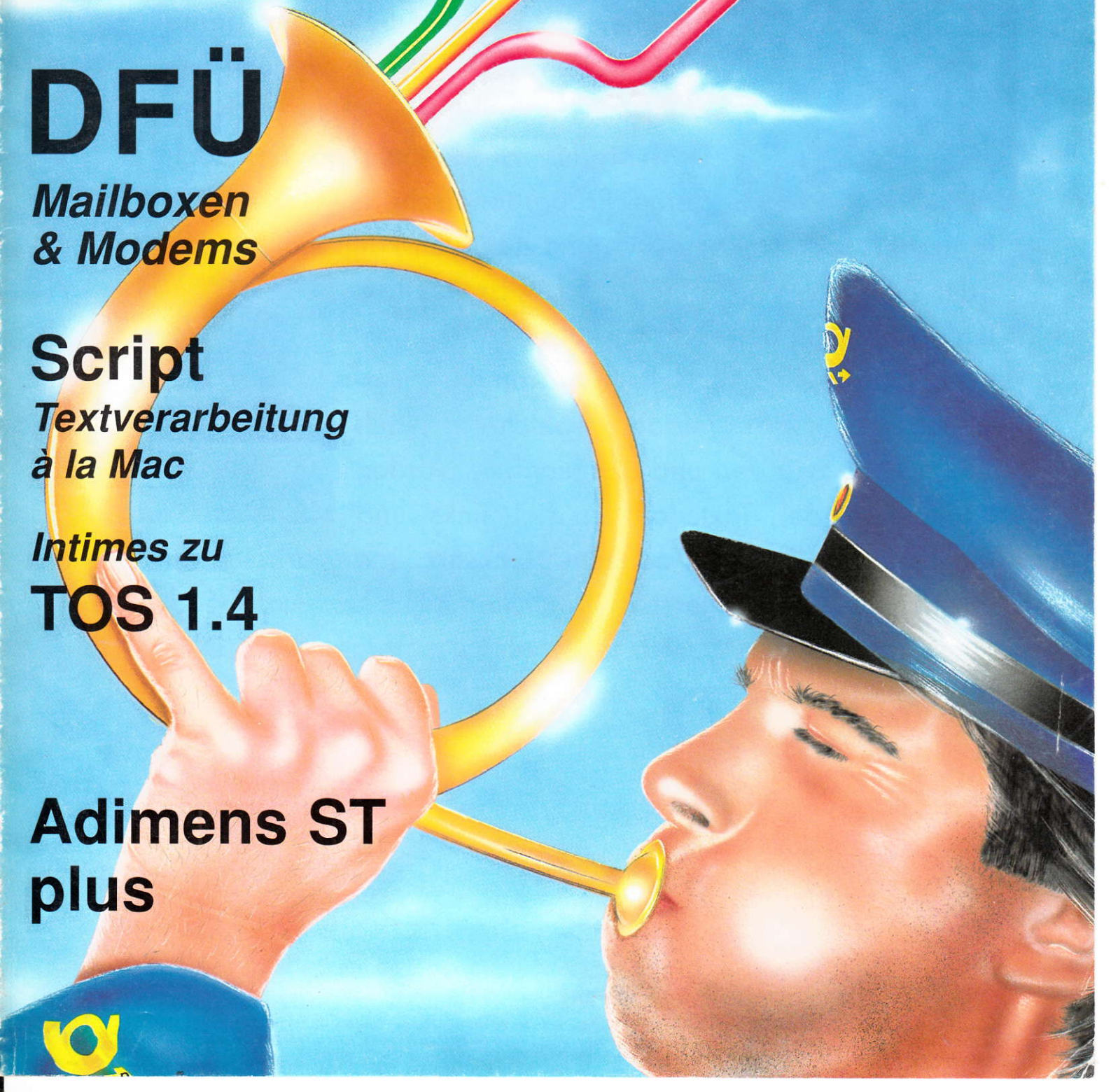
Script

Textverarbeitung à la Mac

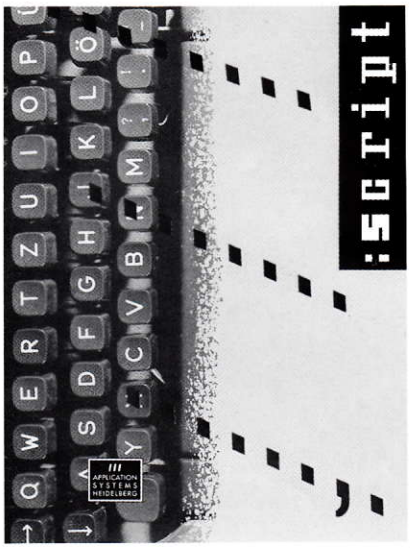
Intimes zu

TOS 1.4

Adimens ST plus



Wer viel oder auch nachts script, äh schreibt. Nanu?



script ist neu. *script* ist für Mengen. *script* ist einfach und modern. Textverarbeitung, leicht, wie Brause. Für Vielschreiber und für Nachtschreiber. Überhaupt für alle, die es beim Schreiben gerne leicht haben. Es kann alle *Signum*!-Zeichensätze verarbeiten, aber auch im Draft-Modus ausdrucken (dann auch gerne Proportionalschriften). Natürlich formatiert *script* automatisch und natürlich kann man in Zoll oder Zentimeter bemaßen und bis zu vier Texten gleichzeitig bearbeiten. Der puren Bequemlichkeit dient die Blockselektion: Kopieren, verschieben, einsetzen, laden, speichern. Auch ASCII kann geladen und ausgegeben werden (Hallo Freunde), Texte ordnen sich links- und rechtsbündig, zentriert oder im Blocksatz. *script* arbeitet im Grafikmodus und unterstützt verschiedene Bildformate (Signum!, Doodle, STAD). Wir tauschen *script* gegen198,-- DM. Nun aber!



APPLICATION SYSTEMS HEIDELBERG

Die Kunst der ST Software.

MMAD Kommunikation

Application Systems Heidelberg Software GmbH, Englerstraße 3, Postfach 10 26 46, D-6900 Heidelberg 1, Telefon (0 62 21) 30 00 02, Fax (0 62 21) 30 03 89 **in Österreich:** Reinhart Temmel Ges.m.b.H., Markt 109, A-5440 Golling, Telefon (0 62 44) 7081-0, Fax 7 18 83. **in der Schweiz:** DTZ DataTrade AG, Langstraße 94, CH-8021 Zürich, Telefon (01) 2 42 80 88, Fax 291 05 07.

EDITORIAL

Von Vergleichen und Marktführern

Zuweilen muß man sich fragen, wohin es mit der Pressefreiheit noch gehen soll. Ein Gerichtsurteil (2 HK 0 2284/89) gibt mir momentan stark zu denken. Nach diesem Urteil muß sich ein Redakteur ernsthaft fragen, wie ein Testbericht auszusehen hat. Wird nämlich ein Vergleichstest verfaßt, muß man sich jetzt auf alle sogenannten "Marktführer" konzentrieren, um eine vollständige Übersicht zu produzieren. Dazu einige Zitate aus dem Urteilsspruch:

"Die Beklagte nimmt mit ihrer Anzeige eine vergleichende Testwerbung vor, wobei sie für sich sogar eine Alleinstellung in Anspruch nimmt. Eine derartige Werbung würde auf jeden Fall voraussetzen, daß das Testergebnis, auf das sich der Werbende bezieht, ordnungsgemäß zustandegekommen ist (...)"

"Von einem ordnungsgemäßen Zustandekommen eines Testergebnisses kann schon nach dem eigenen Vortrag der Beklagten nicht ausgegangen werden. Die Zeitschrift (...) hat nämlich nach Auffassung der Kammer die Warenauswahl zum Test (...) nicht ordnungsgemäß vorgenommen."

"(...) daß das (...) Magazin das Programm der Beklagten als das beste (...) für den ST bezeichnet. Dies ist unzulässig, nachdem die Beklagte selbst einräumen muß, daß das Programm eines weiteren Marktführers (...) nicht getestet wurde."

Natürlich sollte man in einem Testbericht nicht gleich mit dem Superlativ um sich werfen, aber nach Studium des Urteils kommen sofort einige Fragen auf: Wer sagt mir definitv, wer "Marktführer" in einer bestimmten Sparte ist? Wie wird ein "Marktführer" überhaupt definiert? Ist es der- bzw. diejenige, der (die) die meisten Programme verkauft - oder die größte Zahl an Angestellten hat - oder die größte Werbung schaltet - oder aber am längsten im Geschäft ist?

Nehmen wir an, trotz dieser Fragen sind die "Marktführer" ausfindig gemacht worden. Was passiert, wenn einer dieser "Marktführer" sein (ihr) Programm nicht kostenlos zu einem Test zur Verfügung stellen möchte? Zitat aus dem Urteilsspruch: *"Wenn aber bei fünf Marktführern ein Programm überhaupt nicht in den Test einbezogen wurde, ist die Bezeichnung eines Programmes als bestes nach Auffassung der Kammer nicht zulässig."* Muß der Tester sich dann das Programm im freien Handel besorgen, damit der Vergleichstest vollständig wird? Sicher, die Stiftung Warentest kauft auch Testgeräte, doch ein Computer-Magazin ist keine Stiftung, die ihren Sinn einzig und allein darin sieht, Endverbrauchern Testergebnisse zu präsentieren - noch dazu kostet ein Programm erheblich mehr als eine Kaffeemaschine. Was sagt das Urteil dazu: *"Wenn das (...) Magazin schon einen Vergleichstest durchführen und die Ergebnisse veröffentlichen wollte, hätte es sich gegebenenfalls dieses Programm auf dem Markt beschaffen müssen."*

Wir können die Situation auch noch weiter verfolgen: Die - gekauften oder zum Test zur Verfügung gestellten - Programme aller "Marktführer" werden getestet. Nach dem Test möchte ein Hersteller aber nicht, daß der Bericht veröffentlicht wird, weil er denkt, daß sein Programm dabei zu schlecht abschneidet. Was ist zu tun? Wird der Test jetzt unvollständig? Darf der Redakteur noch behaupten, daß alle "Marktführer" berücksichtigt wurden, wenn er das Programm nicht in die Bewertung mit aufnimmt? Oder soll man den Test trotzdem veröffentlichen und wird dann schon fast bedroht, daß keine Werbung mehr geschaltet wird?

Es geht noch weiter: Der Test ist, trotz all dieser Schwierigkeiten, abgeschlossen und veröffentlicht. Zwangsläufig muß bei einem Vergleichstest mindestens ein Programm als Testsieger hervorgehen. Der Hersteller verwertet das Testergebnis bei seiner Werbung. Was wird unter Umständen passieren? Eine Anzeige gegen ihn erfolgt, da ein Mitbewerber (ein "Marktführer"?) diese Werbung als "irreführend" oder "unzulässige Alleinstellungswerbung" bezeichnet. Im Urteil mit dem Aktenzeichen 2 HK 0 2284/89 ist eine Firma dazu verurteilt worden, knapp 35.000,- DM Strafe zu zahlen, weil Sie geworben hatte mit *"Der Testsieger. (...) Vergleichstest (...) Magazin in 10/88 (...)"*. Das Gericht entschied, daß der Vergleichstest kein solcher sei.

Hat nun der Redakteur *"die Warenauswahl nicht ordnungsgemäß vorgenommen"* (laut Anklageschrift), oder fühlt sich lediglich ein Hersteller (der Kläger) ausgebootet? Darf man als gewissenhafter Redakteur noch wertende Bezeichnungen wie "sehr gut" oder "mangelhaft" benutzen, ohne gleich mit einer Klage vor Gericht oder fehlender Werbung (die dem Entzug der Arbeitsgrundlage entspricht) rechnen zu müssen? Gilt ein Schlußsatz des Redakteurs ("Fazit: ...") als subjektive oder objektive Wiedergabe des Testergebnisses? Darf man überhaupt noch ein Programm als das aus einem Test hervorgehende "beste Programm" dieses Tests bezeichnen? Ich kann für mich aus diesen Überlegungen nur einen Schluß ziehen: Entweder lasse ich mich von solchen Urteilen einschüchtern - was ausschließt, daß ich objektive Testberichte verfassen kann -, oder ich schreibe weiter wie bisher - und sollte mir unter Umständen schon jetzt einen Rechtsanwalt "reservieren"...

Martin Pittelkow

I N H A L T

SOFTWARE

Adimens ST plus
Das "PLUS" macht den Unterschied 38

GD-Fibu
- Finanzbuchhaltung leichtgemacht? 41

Mailbox-Programme
- Die Verbindung zur Außenwelt 25

Relax
Aktuelle Spiele 164

Scarabus
- Der Profi-Fonteditor für Signum! und Script 49

Script
- Textverarbeitung à la Mac 16

SPC-AdiProg
- Daten bunkern mit Modula-2 55

TeX
- TeX für den Alltag 60

GRUNDLAGEN

Komfortables Dialog-Handling
in GFA BASIC 3.0 100

Lovely Helper - Ein Desk-Accessory
- Teil 7 124

Modula-2-Kurs Teil 10 110

Numerische Mathematik
- Integration und Differentation 118

Programme unter GEM Teil 2 137

Somewhere over the Rainbow
Intimes zu TOS 1.4 Teil 1 151

Wartezyklen beim ATARI ST 148



Modems
Der Anschluß zur Außenwelt

Um Ihnen eine Übersicht des heutigen Stands bei Modems zu geben, haben wir die Modelle der bekanntesten Anbieter getestet. Es wurden lediglich diejenigen getestet, die auch mit 2400 Baud (gleich zirka 240 Zeichen pro Sekunde) Übertragungsgeschwindigkeit arbeiten, da langsamere Baud-Raten aus der "Mode" kommen und kaum mehr verlangt werden.

Seite 20

ADIMENS ST plus
Das "PLUS" macht den Unterschied

Gerne schmücken die Software-Häuser ihre neuen Produkte mit einem kleinen unscheinbaren Zeichen, dem "+". Andere schreiben das Wort sogar aus: "plus" (z.B. 1st_Wordplus, ST-PASCAL plus, 1st-Proportional plus usw.). Die Firma ADI-Software aus Karlsruhe hat schon auf der Düsseldorfer ATARI-Messe '89 einen Einblick in die neuen Funktionen ihres Produktes "ADIMENS ST plus" (Version: 3.0) gegeben. Was bringt das "Plus" den Anwendern jetzt?

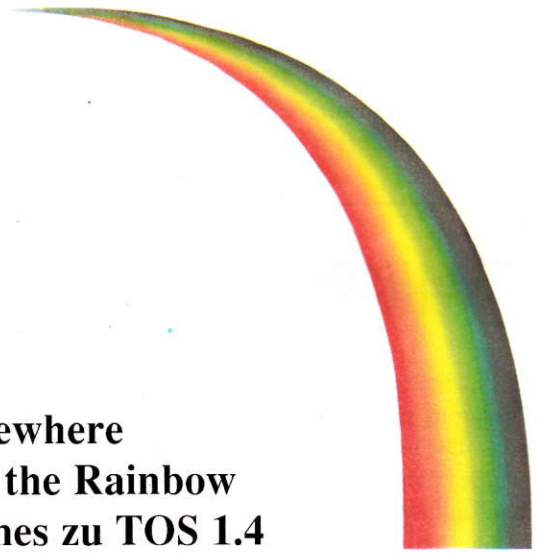
Seite 38



Script
Textverarbeitung à la Mac

Wer ein ST-Anwender der ersten Stunde ist, dem ist vielleicht noch der Rechtsstreit zwischen ATARI und Apple wegen der ähnlichen Benutzeroberfläche, dem Desktop, in Erinnerung. Bald darauf kamen die ersten Mac-Emulatoren, die es erlaubten, Mac-Programme auf dem ST laufen zu lassen. Nun scheint die nächste Generation der Gemeinsamkeiten eingeläutet zu werden: Es werden auf dem ST Programme geschrieben, die Mac-Programmen doch sehr ähnlich sehen. Bewährte Mac-Standards auf dem ST? Das neueste Produkt von Application Systems scheint es zu beweisen, denn wer das Mac-Programm WriteNow kennt, wird sich sofort in Script heimisch fühlen.

Seite 16



Somewhere over the Rainbow Intimes zu TOS 1.4 Teil 1

Nun ist es endlich soweit - die von vielen lang erwartete neue Version unseres Lieblings-Betriebssystems TOS ist fertig. Das neue TOS 1.4 (Rainbow-TOS) wurde auf der Düsseldorfer ATARI-Messe offiziell vorgestellt und endgültig freigegeben. Dies ist Grund genug, es auch unseren Lesern häppchenweise vorzustellen.

Seite 151

HARDWARE

Modems
- Die Verbindung zur Außenwelt 20

MGP-Prommer - programmierte Logik Teil 2
- Einige Schaltungsbeispiele im GAL 173

PROGRAMMIERPRAXIS

Tastenunterstützung in Drop-Down-Menüs 80

Menüleisten total 86

Eigener Desktop in Pascal 91

Eingabe von Termen 93

AKTUELLES

Editorial 3

Immer up to date 192

Kleinanzeigen 182

Leserbriefe 184

NEWS 6

Public Domain. 188

Vorschau 194

RUBRIKEN

Einkaufsführer 71

Inserentenverzeichnis 192

Impressum 194

NEWS

Lispas jetzt als PD

Das Softwarehaus Tommy-Software aus Berlin veröffentlicht jetzt in einer PD-Deluxe-Serie seinen bekannten GEM-Lisp-Interpreter LISPAS II als PD-Deluxe Nummer 2. Damit soll einem noch größeren Kreis von Interessenten die Programmiersprache LISP nahegebracht werden. Kopien des PD-LISPAS II-Interpreters gibt es überall, oder bei TommySoftware gegen DM 20,- für Porto und Verpackung. Kunden, die sich registrieren lassen (DM 20,-), erhalten als Dankeschön ein gedrucktes Handbuch (solange Vorrat reicht) und für DM 99,- den kompletten Sourcetext in Pascal.

Tommy Software
Selchower Straße 32
1000 Berlin 44
Tel. 030/621406-3

Symbolisches Algebra- und Programmiersystem RIEMANN

Von der Firma Begemann & Niemeyer Softwareentwicklung gibt es jetzt "RIEMANN", das sowohl ein Mathematikprogramm für symbolische Algebra und Analysis als auch eine vollständige KI-nahe und LISP-ähnliche Programmiersprache ist. Im Bereich der Mathematikfunktionen ist Riemann zu etwa 98% kompatibel zu muMATH-83 und muSIMP-83. Die interne Darstellung von Riemann ist LISP sehr ähnlich, nicht so sehr jedoch die äußere Syntax, die eher an z.B. PASCAL orientiert ist. Die Programmiersprache Riemann ist ebenfalls zu etwa 95% kompatibel zu muSIMP-83, der Programmiersprache, in der muMATH-83 geschrieben ist. Riemann kostet DM 238,- (für Studenten DM 168,-).

Begemann & Niemeyer
Softwareentwicklung GbR
Schwarzenbrinker Straße 91
4930 Detmold

Neuheiten zur UKW-Tagung in Weinheim

Der Verlag Afusoft hat zur UKW-Tagung 1989 sein Vertriebsprogramm für Produkte der Kommunikationstechnik erweitert. Für alle Packet-Radio- und RRTY-Fans erscheint jetzt die Version 2.0 des Kommunikationsprogramms Radio-Writer ST. Gegen Einsendung der Originaldiskette erhalten alle registrierten Benutzer gegen Zahlung einer geringen Handlingsgebühr die neue Version. Ebenfalls im Programm sind die Afusoft MPSK Empfänger-Option (ein neues Option-Board für den Standardkonverter MPSK), die MPSK Satelliten-Demodulator-Option (ein AM Option-Board für den Standardkonverter MPSK), ein Morse-Tutor PC (ein Morsetrainer für IBM-kompatible Rechner), sowie der Radio-Manager ST, der aus einem Stations-Management und einer Datenbank besteht.

Verlag Afusoft
Steiner Str. 5
7531 Eisingen
Tel. 07232/81664

OMIKRON.Assembler

Omikron liefert seinen Assembler nun in der Version 1.86 aus. Die benutzeroberfläche wurde komplett überarbeitet und mit Menüleiste und Maussteuerung ausgestattet. Menüfunktionen sind jedoch weiterhin auch über Tasten erreichbar. Der Zehnerblock läßt sich als erweiterter Cursorblock oder zur Hex-Eingabe verwenden. Neue Einstellmöglichkeiten, wie z.B. das Ausschalten von Alertboxen, wurden implementiert. Das Programm arbeitet jetzt auch mit dem linkfähigen DRI-Format, und der Sourcetext kann platzsparend gepackt abgespeichert werden. Auf Tastendruck werden automatisch Optimierungen von Sprüngen und Subroutine-Aufrufen durchgeführt. Gegen Einsendung der Programmdiskette und DM 30,- erhalten registrierte Benutzer den neuen Assembler incl. komplett überarbeitetem Handbuch.

Omikron
Erlachstraße 15
7534 Birkenfeld
Tel. 07082/50048

PAM'S NET

Seit über einem Jahr liefert PAM Software das schnelle Netzwerk PAM's NET, mit dessen Hilfe ATARI ST-Computer professionell vernetzt werden können. Die Software erlaubt die gemeinsame Nutzung von im Netz verteilten Festplatten (incl. File/Record-Locking für Datenbanken), Druckern, Streamern, V.24-Schnittstellen etdc., wobei auf die Netzlaufwerke 100% TOS-transparent zugegriffen werden kann. Die Server bleiben weiterhin als Arbeitsplatz nutzbar. Die bisherigen PAM's NET/s-Adapter wurden inzwischen durch Standard-Ethernet (IEEE 802.3, 10 MBit/s) Module in mehreren Ausführungen ergänzt. Verfügbar sind Adapter sowohl für den DMA-Port als auch den ROM-Port und den Prozessorbus des MegaST - alle mit integriertem Cheapernet-Transceiver. Die Software wurde um eine TCP/IP-Shell mit Tektronix/VT100-Emulation erweitert, so daß nun komfortable Kommunikation mit UNIX-, VMS- und MS-DOS-Systemen etc. möglich ist.

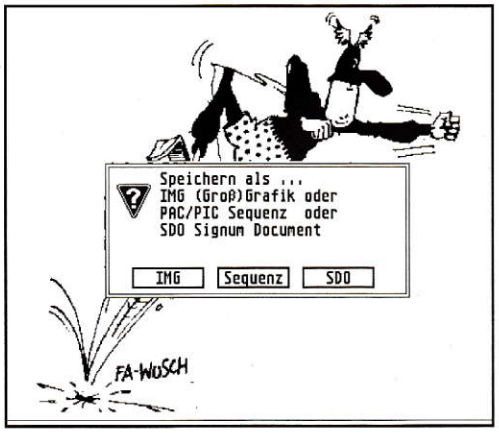
PAM Software
Carl-Zuckmayer-Straße 27
6500 Mainz 33
Tel. 06131/476402

MNP 5 und Mailboxen bei BIELING

In einem Monat beginnt die Firma BIELING Computersysteme aus Recklinghausen mit der Auslieferung ihrer MNP 5-Modems. Sie beherrschen die Übertragungsstandards V.21, V.22, V.22bis, Bell 103 und Bell 212a und werden zum für MNP 5-Modems überaus günstigen Preis von 649,- DM verkauft werden. Interessenten sollten sich schon jetzt bei BIELING melden, um sich eines der Modems zu sichern! Ebenfalls von BIELING angeboten wird jetzt das Mailboxprogramm MagicBOX ST und die Point-Version. Eine Demo-Version des Programms liegt jedem verkauften ATARI ST-Modem bei.

BIELING Computersysteme
Spitzweg 11
4350 Recklinghausen
Tel. 02361/181485

Bildkonvertierung



Dem lästigen Konvertieren von Bildern wird nun endgültig das Garaus gemacht: Mit *Convert* lassen sich mehr als 60 Bildschirmformate einladen und als PIC, PAC, SDO oder IMG-Format wieder abspeichern. Zu den einladbaren Formaten zählen auch Bilder vom AMIGA oder dem MacIntosh. Nach dem Einladen kann gar ein Bildschirmausschnitt gewählt werden, der abgespeichert werden soll! Convert verarbeitet aber nicht nur einzelne Bilder, sondern auch Bildsequenzen. Sowohl beim Laden, als auch beim Speichern, ist die Pixelgröße nicht beschränkt auf 640 x 400 Punkte. Ist das Bild größer, kann es gescrollt werden. Mit Convert kann man endlich alles in einem Programm machen. Die konvertierten Bilder lassen sich rückspeichern als STAD-Format, Signum-Dokument (!) oder Image-Bild für Calamus, Pagestream, Timeworks, Ventura etc. (!!). Convert läuft auf allen ATARI ST mit mindestens 1 MB. Im Lieferumfang enthalten sind über 1 MB Beispielgrafiken, gespeichert auf zwei doppelseitigen Disketten. Convert kostet 95,- DM bzw. 95,- SFr bzw. 790,- öS.

Andreas Pirner Software
Bundesallee 56
1000 Berlin 31
Tel. 030/8534350

Neues vom C.Itoh C-610

Ergänzend zu unserem Testbericht über den C-610 in der September-Ausgabe hat uns C.Itoh mitgeteilt, daß der Drucker jetzt tatsächlich in Grau lieferbar ist. Es ist abzuwarten, wie das neue Outfit bei den Kunden ankommt - bei C.Itoh erwartet man eine große Nachfrage. Die in unserem Bericht kritisierten Probleme mit der Druckbarkeit einiger IBM-Sonderzeichen im Epson-Modus seien mittlerweile behoben. Mit den neuen ROMs, mit denen der Drucker jetzt ausgestattet ist, gibt es diese Probleme nicht mehr.

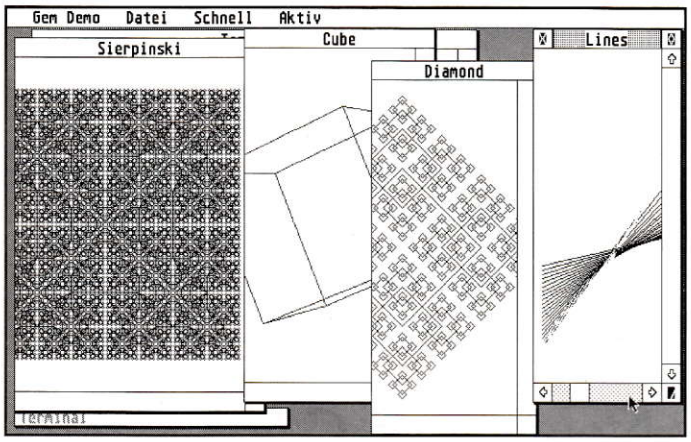
BC-Fortan 77

BC-Fortran77 ist nun um einen symbolischen Debugger und die Unterstützungsmöglichkeit durch einen 68881-Koprozessors erweitert worden. Beim Debugger wird die normale Bildschirmausgabe nicht gestört, da er einen eigenen Bildschirmspeicher verwendet. Der obere Teil des Ausgabebereichs enthält Daten (Variablen, Felder, Register), der untere Teil zeigt Quellprogramme, auf Wunsch mit Angabe der Laufzeit pro Fortran-Anweisung oder Disassemblerausgabe. Da der Debugger relativ umfangreich ist (ca. 100 kB), wird er bei Bedarf nachgeladen. Für "Notfälle" steht weiterhin der alte Debugger zur Verfügung. Der Koprozessor kann jetzt wahlweise über die Library oder direkt angesprochen werden. Eine Compileroption für den 68020 ist vorhanden, allerdings unterstützt das TOS diesen Prozessor bisher nicht. Weiterhin wurde das Laufzeitsystem von BC-Fortran77 komplett überarbeitet: Es sollte in zukünftigen Versionen nicht mehr nötig sein, die Objektdateien neu zu übersetzen.

BC-Softwaretechnik André Köstli
Martin-Luther-Straße 63
7000 Stuttgart 50
Tel. 0711/564736

Viczena-News

Die Firma advanced applications Viczena bietet einige Neuigkeiten an. *MASKWIN* ist ein Maskengenerator. Er ist Bestandteil des SPC AdiPROG und erlaubt den Aufbau von Masken auf unterliegende Grafiken. Maskwindow ist das erste einer Serie von objektorientierten Windows, die auf der Windowschnittstelle SSWiS aufsetzt. *ICONWINDOW* ist das zweite objektorientierte Konstrukt dieser Serie. Mit diesem Programm besitzt man eine recht große Flexibilität, ohne in die Tiefen des AES vordringen zu müssen. Die Verwaltung von Objekten geschieht hier ebenfalls über SSWiS. *WPMail* ist eine Ergänzung zum WPProg-Paket des SPC AdiPROG und erlaubt auf recht einfache Weise die Serienbrieferstellung mit Wordplusdokumenten. Die hier aufgeführten Neuerungen gehören ab sofort zum Standardlieferumfang von SPC Modula-2 bzw. SPC AdiPROG und werden den Kunden im Rahmen eines kostenlosen Updaters zur Verfügung gestellt. Weiterhin wurde eine *GFALIB* entwickelt, um allen GFA BASIC-Programmierern den Umstieg auf Modula-2 zu erleichtern. In der Library sind *alle* GFA BASIC 2.0 und einige GFA BASIC 3.0-Befehle enthalten. So können auch größere Programme durch weitgehend syntaktische Ersetzungen in Modula-2 gewandelt werden.



advanced applications
Viczena GmbH
Sperlingweg 19
7500 Karlsruhe 31
Tel. 0721-700912

Streamer- und Belichtungsservice in der Schweiz

Die vir einem Jahr in Luzern gegründete Firma Aha EDV Support AG, deren Spezialgebiet in der Schulung und Beratung liegt, bietet den ersten Belichtungsservice für die Schweiz an. Calamus- und Wordplus-Dokumente sowie ASCII-Dateien werden mit einer maximalen Auflösung von 2450 dpi je nach Wunsch auf Film oder Papier mit einer Linotronic 300 im Format A4 oder A3 belichtet. Wem die Datensicherung mit Disketten zu aufwendig und ein eigener Streamer zu teuer ist, kann seine Festplatte von der Firma Aha EDV Support AG streamen lassen. Da die Firma den Kunden besucht, entstehen für diesen keine Umtriebe oder Demontagen an seiner EDV-Anlage.

Aha EDV Support AG
Suchulung, Beratung
und Anwendung
Zähringerstraße 21
CH-6003 Luzern
Tel. 041 22 51 61

Schädliche Lüfter

Nachdem in letzter Zeit bereits einige Hersteller dem Lüfterkrach der ATARI-Festplatten ein Garaus bereiten wollten, gesellt sich nun als weiterer Hersteller die Firma digital data deicke aus Hamburg hinzu. Sie bietet einen Lüfter an, mit dem verhindert wird, daß die Festplatten den zulässigen Temperaturanstieg von höchstens 10 Grad Celsius pro Stunde erfahren. Dies sei nämlich, laut digital data deicke, "ein entschiedener Konstruktionsfehler" vieler auf den Markt gekommener Noise-Reduction-Kits. Für DM 49,- bekommt man einen Lüfter, das benötigte Werkzeug zum Einbau und eine genaue Einbauanleitung. Wem der Einbau zu mühselig ist, kann sich auch die Arbeit vom Hersteller abnehmen lassen: Im Preis von 49,- DM ist alles enthalten.

digital data deicke
Nordring 9
3000 Hannover 1

Update von CADja

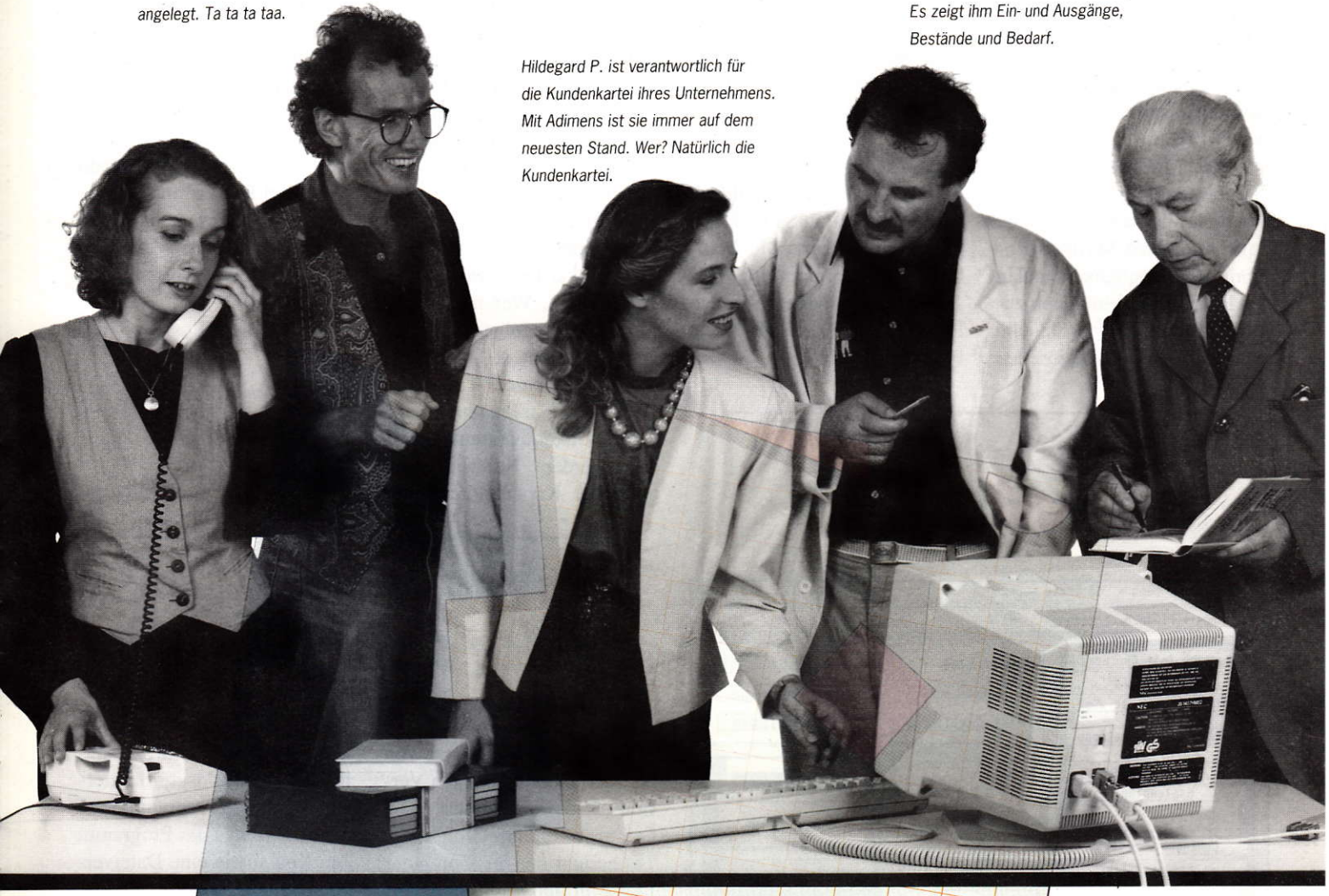
Unser ausführlicher Test in der Juni-Ausgabe befaßte sich auch mit dem CAD-Programm CADja von Computer Technik Kiekbusch. An der Version 1.0 fand sich der eine oder andere Kritikpunkt. Mittlerweile gibt es eine überarbeitete Version 1.02. Einige neue Features: Vor allem die Schraffurfunktion ist sehr viel schneller und flexibler geworden. Der Status der Schraffur läßt sich löschen. Dadurch kann man die Schraffurlinien wie bei TechnoCAD auch als Konstruktionslinien verwenden. Texte lassen sich nun auch nach dem Einfügen in die Zeichnung edieren. Die Linienstärke kann zu Kontrollzwecken angezeigt werden. Damit sind die hauptsächlichen Kritikpunkte mittlerweile allesamt beseitigt. Ferner ist mittlerweile eine umfangreiche Sammlung an Symbolen aus den Bereichen Elektronik und Pneumatik enthalten. Weitere sollen folgen. Doch damit nicht genug - Autor Michael Rauch plant eine sogenannte 'Megaversion', die tatsächlich nur auf Rechnern mit wenigstens 2MB Hauptspeicher laufen soll. Features dieser geplanten Version: Doppellinie, Kreis über zwei Punkte, nachladbare Fonts, Textattribute, Ellipsenbögen, Bereiche drehen, komplette Tastatur- oder CRP-Graphiktablett-Bedienung. Für diese Version ist allerdings noch kein Liefertermin avisiert.

Adimens. Perfekt in Anwendungen.

Ganz privat ist Peter T. Musikfan. Und Sammler. Den Katalog seiner oft seltenen Liebhaberstücke, seiner LPs und CDs hat er mit Adimens angelegt. Ta ta ta taa.

Hildegard P. ist verantwortlich für die Kundenkartei ihres Unternehmens. Mit Adimens ist sie immer auf dem neuesten Stand. Wer? Natürlich die Kundenkartei.

Herr K. leitet die Materialverwaltung in einem Großunternehmen. Sein wichtigstes Werkzeug heißt Adimens. Es zeigt ihm Ein- und Ausgänge, Bestände und Bedarf.



Das relationale Datenbanksystem Adimens hat viele Freunde, das ist bekannt. Bemerkenswert bei diesen Freunden ist, daß sie so unterschiedlich sind. Da gibt es Privatleute und Selbständige, kleine Unternehmen und Großunternehmen, Abteilungen und Verwaltungen, Wissenschaftler und Sacharbeiter. Gemeinsam ist allen, daß sie ihre Datenbestände mit Adimens verwalten, weil es vielseitig und leicht zu handhaben ist und natürlich auch, weil Adimens sich allen Bedürfnissen anpaßt.

Kurz: eine perfekte Datenbank. Über 50 000 verschiedene Anwender haben das schon gemerkt und sind daher perfekt in ihren Anwendungen.

Adimens läuft auf dem Atari und auf dem PC, aber auch auf Großrechenanlagen. Fordern Sie Informationen über das relationale Datenbanksystem Adimens an, und fragen Sie uns.

ADI Software GmbH
Hardeckstraße 5, D-7500 Karlsruhe 1
Telefon (0 72 1) 57 000-0

ADI Software GmbH

MMAD Kommunikation

Cappuccino-Mailbox

In Bottrop ist jetzt die Cappuccino-Mailbox online. Cappuccino ist ein neues Jugendmagazin, das mit der Mailbox Artikel der im Bundesgebiet verteilten Lokal- und Regionalredaktionen sammelt und verwaltet. Trotz dieser Aufgabe ist die Mailbox auch für "Nichtredakteure" frei zugänglich. Sie bietet zusätzlich zum Standard-Mailboxprogramm die Möglichkeit, mal in den Betrieb der Zeitung reinzuschnuppern und sich aktuelle Artikel der Zeitung anzusehen und vielleicht auch selbst einmal an der redaktionellen Arbeit teilzunehmen. Über das CTS-Brett der Mailbox, die dem MagicNET angeschlossen ist, ist es auch auch möglich, Bestellungen oder Informationen über aktuelle Software, z.B. Fremdsprachensysteme für Griechisch, Kyrillisch und Hebräisch auf ST und MS-DOS, aufzugeben. Die Mailbox hat die Rufnummer 02041/27778 und arbeitet mit den Parametern 8 Datenbits, keine Parität, 1 Stopbit.

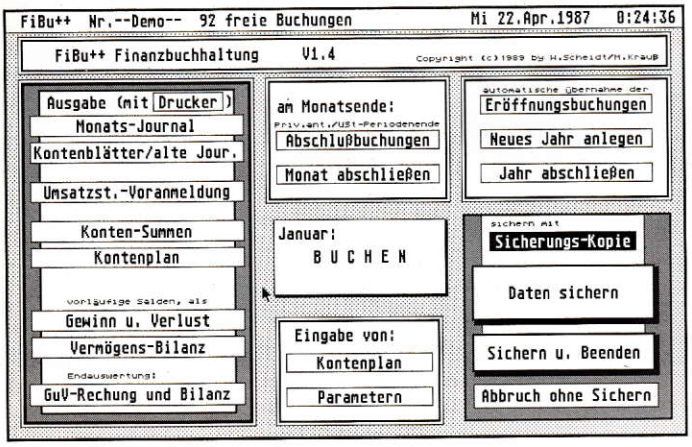
Modula-News

Vielleicht haben sich viele Modula-2-Programmierer(innen) schon mal gewünscht, Kontakte mit anderen Modula-2-Benutzern zu knüpfen. Doch wie soll man an interessierte Leute herankommen? Zwei Informatikstudenten aus München haben sich vorgenommen, eine Art Diskettenzeitung, mit Artikeln über Modula-Compiler, Tips und Tricks, Testberichten etc. herzustellen. Wer mitmachen möchte, sollte folgende "Spielregeln" beachten: Der Quelltext muß beigefügt werden. Am besten sollte auch eine kurze Dokumentation beigelegt werden. Natürlich darf auch die Kontaktadresse für Nachfragen nicht fehlen. Soll die Diskette wieder zurückgeschickt werden, bitte Rückporto beifügen. Fehlen darf auf keinen Fall eine kurze Mitteilung in folgendem Stil: "(nicht)kommerzielle Nutzung gestattet, frei von Rechten Dritter, etc. pp". Sobald genügend Programme gesammelt sind, werden auch alle Routinen auf einer MAXON-Sonderdiskette veröffentlicht.

Andreas Schallmaier
bei Golgath
Waskestraße 6
8000 München 19

Andreas Kisslinger
Haberstraße 10
8000 München 50

FiBu++



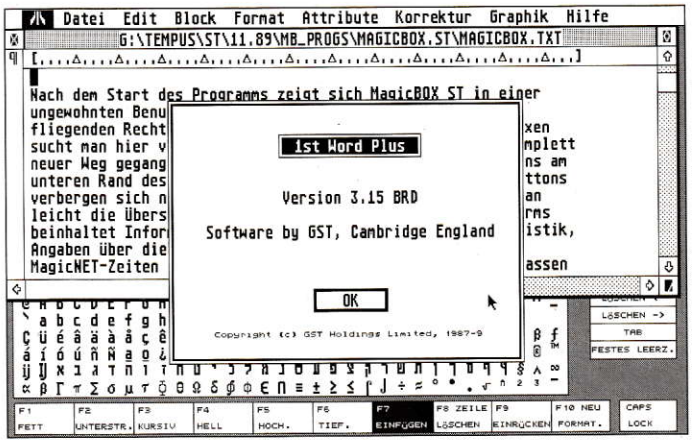
Mit FiBu++ kann man jetzt ein Finanzbuchhaltungsprogramm kaufen, das man auch ohne lange Einarbeitungszeit bedienen kann. Das Programm hält sich streng an das Bilanzrichtlinien-Gesetz. Ein langes Warten auf den Drucker fällt hier flach: Ein Drucker-Spooler ist eingebaut. Erfreulich ist, daß das gesamte Programm mausgesteuert ist. So können Sie beispielsweise zwar eine Kontonummer per Hand eingeben, sie aber auch einfach anklicken. FiBu++ kann wahlweise monatlich automatisch oder per Hand buchen. Alle Listen können auf dem Drukker ausgegeben werden. FiBu++ kostet 498,- DM. Wer erst eine Demoversion haben möchte, kann diese für 40,- DM ersteigern, der Preis wird auf die Vollversion angerechnet.

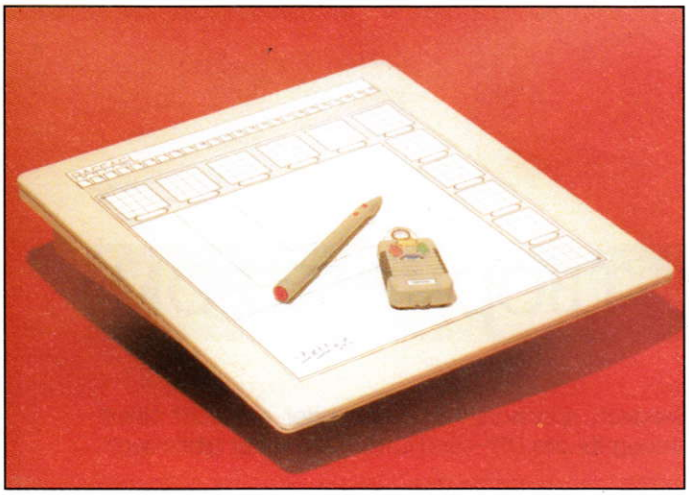
Softwareentwicklung und
Computergrafik
Matthias Krauß
Espanstraße 76
8510 Fürth
Tel. 0911/794584

Wordplus 3.15

1st_Wordplus wird jetzt von ATARI in der offiziellen Version 3.15 ausgeliefert. Die Features entsprechen dem Testbericht der ST Computer 12/88, in der bereits ausführlich auf die Vorabversion 3.11 eingegangen wurde. Wesentliche Neuerungen sind Tastaturkombinationen für die Menüleiste, Großbildschirmunterstützung. Dem Programm werden neuerdings das Programm 1st_Xtra sowie eine Dateiverwaltung beigelegt.

Alle ATARI-Fachhändler





RAFFAEL

Mit Raffael bietet die Firma Zeller ComputerS nun ein Grafiktablett an. Es besitzt eine 30x30 cm große aktive Fläche, ausreichend für DIN A 4 hoch und quer. Als Eingabesystem dient entweder ein 4-Button-Cursor oder ein 3-Tasten-Stift, die beide kabellos angeschlossen werden. Raffael wird an der seriellen Schnittstelle des STs angeschlossen und über ein mitgeliefertes Treiberprogramm vom System angesprochen und emuliert die Maus. Weiterhin können bis zu 23 Funktionstasten und 176 aktive Flächen mit Maus- und Tastaturmakros belegt werden. Der Preis für Raffael liegt bei 1798,- DM inkl. Software, Interface, einem Eingabegerät (Stift oder Cursor) nach Wahl und Bedienungsanleitung.

Zeller ComputerS
Hauptstraße 53
7230 Schramberg
Tel. 07422/1606

Technobox CAD/1 Studentenversion

Das CAD-System für den ST, das zu einem Preis von 1698,- DM angeboten wird, ist ab dem 15.10.89 für Studenten und Schulen zu einem Preis von 698,- DM erhältlich. Neu im Lieferumfang der Technobox und speziell für die CAD-Ausbildung existiert vom Technobox CAD/1-System ein Paket mit 10 Nutzungslizenzen zum Preis von 3990,- DM. Die Aktion ist bis zum 15.12.89 beschränkt und soll je nach Resonanz eine Dauereinrichtung werden. Die Studentenversion, die keinerlei Unterschiede zu der Originalversion zeigt, ist bei jedem Technobox-Fachhändler zu beziehen.

Technobox Software GmbH
Kornharpener Straße 122a
4630 Bochum 1
Tel. 0234/503060

Eickmann-Wechselplatte

Die für ihre leisen und schnellen Festplatten bekannte Firma Eickmann bietet ab sofort auch eine Wechselplatte an. Sie hat ein leises Betriebsgeräusch [kleiner 41 db(A)], wird mit dem neuen HDPlus- und Cache-Programm geliefert (beide unterstützen den Media-Wechsel, auch bei unterschiedlicher Partitionsgröße), die auch MS-DOS-Medien verarbeiten, besitzen einen durchgeschliffenen DMA-Bus, sind autobootfähig und von außen adressierbar. Im Lieferumfang enthalten ist das schon bekannte HDPlus von Eickmann, das HDU von Application Systems, der Optimizer von FPS, das Virus Filter Set und das Cache-Programm von Eickmann.

Eickmann Computer
In der Römerstadt 249
6000 Frankfurt 90
Tel. 069/763490

WordPerfect ST

Ab Ende Oktober wird die WordPerfect-Version 4.1 für den ATARI ST ausgeliefert, die dem Funktionsumfang der WordPerfect 4.2 entspricht. Die ATARI-Version bietet nicht nur die gesamte Funktionsvielfalt der WordPerfect-Textverarbeitung, sondern nutzt auch die Vorteile der 68000er-Rechner. Natürlich kann das Programm über Maus und Tastatur gesteuert werden. WordPerfect wird exklusiv über ATARI vertrieben.

WordPerfect Software GmbH
Frankfurter Straße 33-35
6236 Eschborn
Tel. 06196/904-0

Falsche Nummer

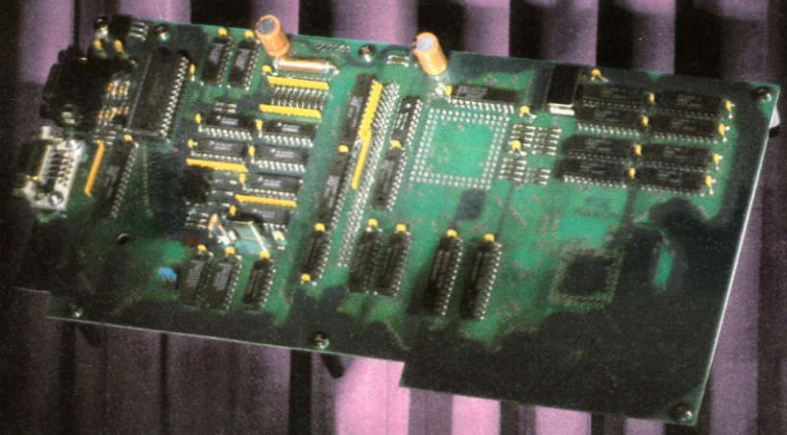
In unserer letzten Ausgabe haben wir auf Seite 10 die Nummer der schweizer AAV-Mailbox falsch abgedruckt. Die richtige Nummer lautet 41-1-3837054.

"Dem Floppycontroller Dampf gemacht"

Leider hat sich ein Fehler in den Artikel in der ST 7/8 '89 eingeschlichen. Unter 'Aufbauhinweise' findet sich der Satz: Zum Schluß stellen Sie die Masse- (Pin 14 *Ym 2149*) und +5 Volt-Verbindung (Pin 15 *Ym 2149*) her. Wie auch richtig aus dem Schaltplan hervorgeht, sind die IC-Bezeichnungen leider vertauscht, d.h. die Masseverbindung geht nach Pin 14 *WD 1772*, die +5 Volt-Verbindung nach Pin 15 *WD 1772*. Das IC Ym 2149 wird also nur an einer Stelle (Pin 15) mit der Schaltung verbunden, die Betriebsspannung wird nur am FDC WD 1772 abgegriffen.

MGE

Ein Grafikstandard stellt sich vor



Mit ihrem schnellen Intel 82786-Grafikprozessor und ihren vielfältigen Auflösungen bietet die MAXON Graphic Expansion (MGE) die ideale Profi-Grafikerweiterung für alle Mega ST-Besitzer mit geeignetem Monitor (z.B. MultiSync oder Großbildschirm).

Sie besticht durch ihren Leistungsumfang und ihre Flexibilität. Dank des GDOS-VDI-Treibers sind GEM-Programme lauffähig.

Egal ob Sie Desktop Publishing (z.B. Calamus) oder anspruchsvolle Farbgrafik einsetzen wollen, die MGE verwandelt Ihren Mega ST in eine Grafik-Workstation!

Auflösungen:

1. 13,75 MHz, 800 x 600 Bildpunkte, maximal 256 Farben interlace, 50 Hz Halbbildfrequenz (für Video- und Genlockanwendungen);
2. 27,5 MHz, 640 x 480 Bildpunkte, maximal 256 Farben aus 262 144 non-interlace, 66 Hz Bildfrequenz;
3. 27,5 MHz, 800 x 600 Bildpunkte, maximal 256 Farben aus 262 144 interlace, 80 Hz Halbbildfrequenz;
4. 55 MHz, 896 x 684 Bildpunkte, maximal 16 Farben aus 262 144 non-interlace, 67 Hz Bildfrequenz;
5. 55 MHz, 1280 x 1024 Bildpunkte, maximal 16 Farben aus 262 144 interlace, 65 Hz Halbbildfrequenz;
6. 110 MHz, 1280 x 960 Bildpunkte, monochrom non-interlace, 60 Hz Bildfrequenz.

Mitgelieferte Software:

- Hardware-Treiber
- GDOS-VDI-Treiber
- Bindings für TURBO-C
- Bindings für Assembler

Weitere Informationen erhalten Sie bei:

MAXON Computer GmbH
Schwalbacherstr. 52
6236 Eschborn
Tel.: 0 61 96 / 48 18 11
FAX: 06196/41885

MAXON

Gamma - VEKTOR-Plus

Die Firma Astrosoft liefert für ihr pixelorientiertes Grafikprogramm Gamma_04 jetzt auch das Programm Gamma-VEKTOR-Plus aus. Damit kann jedermann Rastergrafiken mit dem Programm-Modul ALTERNATE-VEKTOR einfach und schnell in Vektorgrafiken umwandeln. Ein Originalprogramm Gamma_04 auf zwei Disketten, komplett mit Handbuch und Box, kostet 169,- DM. Die Vorläufer-Version 03 (weniger Sonderfunktionen) ist weiterhin für DM 119,- erhältlich. Gamma-VEKTOR-Plus auf zwei Disketten, komplett mit Handbuch und Box, kostet DM 249,-. Für Schulen und Universitäten etc. werden Sonderkonditionen und Mehrfachlizenzen gewährt.

Astrosoft + Optik
Altvaterstraße 7
8728 Haßfurt

SIMULA

Allen, die eigene Erfahrungen mit objektorientierter Programmierung sammeln wollen, aber nicht mit dem bisher dafür erforderlichen Kleingeld gesegnet sind, bietet sich nun eine attraktive Alternative: SIMULA, das bisher nur auf größeren Rechnern erhältlich war, ist nun auch für den ST verfügbar. Zusätzlich zur objektorientierten Programmierung enthält SIMULA komfortable Werkzeuge für die prozeßorientierte diskrete Simulation, mit deren Hilfe auch große und komplexe Systeme modelliert werden können. Weitere wesentliche Merkmale sind die volle Typ-Prüfung zur Übersetzungszeit, die automatische Speicherverwaltung durch einen "Garbage-Collector" und die aufgrund der Standardisierung gute Portabilität von SIMULA-Programmen. SIMULA kann für DM 198,- bezogen werden über

SIMULA-Team
Universität Dortmund / IRB
Postfach 500 500
4600 Dortmund 50

e-mail: simula@unido.uucp
simula@unido.bitnet

Neuer NEC-Drucker

Mit dem Pinwriter P2plus bietet NEC ab November den Nachfolger des P2200 an. Viele Optionen wurde gegenüber dem Vormodell verbessert, so druckt der P2plus jetzt mit 192 Zeichen pro Sekunde. Außerdem wurden sogenannte "High-Speed-LQ-Fonts" entwickelt, die Korrespondenzdruck mit bis zu 160 Zeichen pro Sekunde erlauben. Natürlich hat auch der P2plus eine Auflösung von 360x360 Punkten pro Inch. Der eingebaute Traktor arbeitet wahlweise mit Zug oder Schub. Außerdem ist eine Paper-Park-Funktion eingebaut. Mit 5 zusätzlichen Schriften wird das Gerät eine echte Bereicherung für jeden Druckertisch. Ein Test folgt in einer der nächsten Ausgaben.

NEC Deutschland GmbH
Postfach 801309
8000 München 80

GFA-BASIC-Shell v1.5

Wem die von GFA mitgelieferte Shell zum GFA-BASIC 3.0 zu wenig Komfort bietet, kann nun für DM 20,- eine professionelle Shell bekommen: Sie verwaltet alle GFA-BASIC-Unterprogramme ab der Version 2.0x, unterstützt die Optionen für die 3.0x-Version des Compilers/Linkers, bietet sämtliche Dateioptionen des Desktops, beinhaltet einen Accessory-Compiler für Version 2.0x und vieles mehr. Die Shell kann bezogen werden über

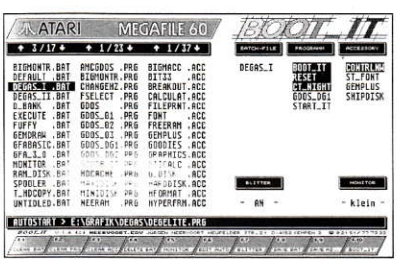
DAR's
Daniel Redanz
Kirchstraße 91
4353 Oer-Erkenschwick

Vortex plus 20-MB Festplatte	DM	849,00
Vortex plus 60-MB Festplatte	DM	1399,00
Turbo-C mit Ass. + Debugger V1.1 dt.	DM	259,00
Signum II deutsch	DM	339,00
Megamax Modula II deutsch	DM	309,00
Stad V1.3 deutsch	DM	139,00
PC-Speed MS-Dos-Emulator	DM	499,00
BTX-Term an Postmodem deutsch	DM	249,00
N-N-Disk 3.5-Z DD . DM 1,99 Psion Chess	DM	59,95
LDW Power Calc dt. DM 209,00 Cyber Paint 2	DM	109,00
Amstrad 24-Nadeldrucker LQ 3500 dt.	DM	599,00
TDI-Modula V3.01 Standard englisch	DM	149,00

Kostenlose Prospekte,
auch für Amiga und IBM von

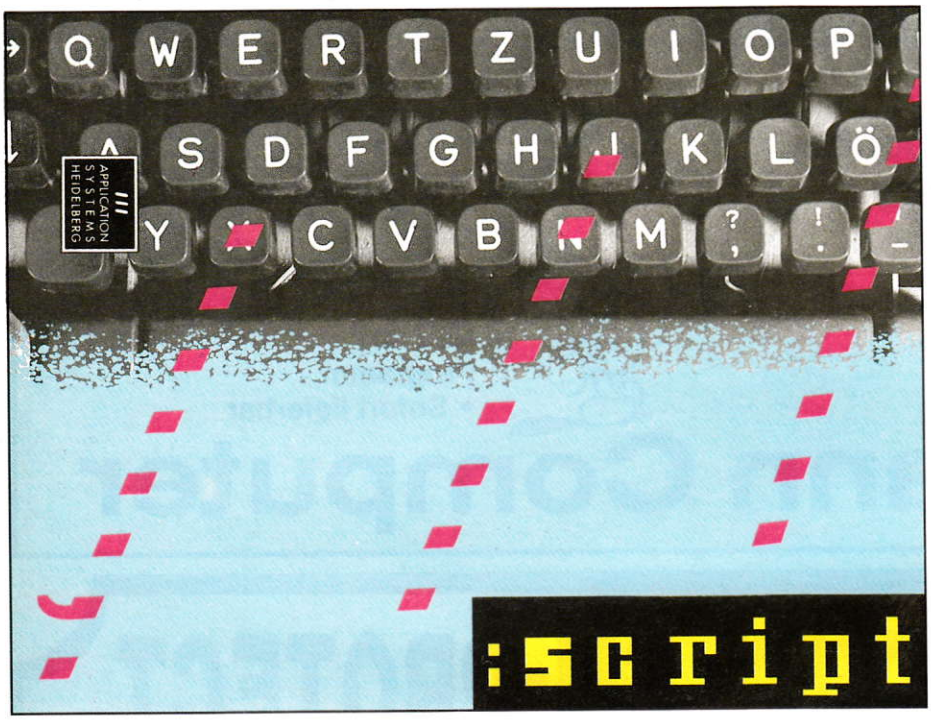
CWTG **C W T G Joachim Tiede**
Bergstraße 13 ★★★ 7109 Roigheim
Tel./BTX 0 62 98 / 30 98 von 17-19 Uhr

BOOT_IT
... das PLUS für jede Festplatte !



- *GEM-Programme mit allen TOS-Versionen automatisch starten*
- *Auswahl der ACC's und PRG's bei jedem Bootvorgang möglich*
- *Anlegen, ändern oder löschen der Batch-Dateien in BOOT IT - ohne separaten Editor*
- *Batch-Files mit - Accessories*
  - *AUTO-Ordner-Programme*
  - *Zugriffspfad u. Name der AUTOSTART-Anwendung*
  - *BLITTER-Einstellung*
  - *Auflösung im Farb-Modus*
- *Verwaltung von vier auflösungs-abhängigen DESKTOP's - auch für Großbildschirm*
- *resetfestes Datum / Uhrzeit bei Systemen ohne batterie-gepufferte Echtzeit-Uhr*
- *Auswahl über Maus oder Tastatur*
- *mit Handbuch, Update- u. Hot-Line-Service*
- *DM 69,- zuzügl. DM 5,- Versandkosten bei V-Scheck/Vorkasse (Nachnahme: DM 7,50)*
- *Bezug über ATARI-Systemfachhändler oder direkt bei :*

NEERVOORT . EDV Jürgen Neervoort Neufelder Str. 21 4152 Kempen 3
Telefon : 0 21 51 / 77 73 22 *(kein Ladenverkauf !)*



Script

Textverarbeitung à la Mac

Wer ein ST-Anwender der ersten Stunde ist, dem ist vielleicht noch der Rechtsstreit zwischen ATARI und Apple wegen der ähnlichen Benutzeroberfläche, dem Desktop, in Erinnerung. Bald darauf kamen die ersten Mac-Emulatoren, die es erlaubten, Mac-Programme auf dem ST laufen zu lassen. Nun scheint die nächste Generation der Gemeinsamkeiten eingeläutet zu werden: Es werden auf dem ST Programme geschrieben, die Mac-Programmen doch sehr ähnlich sehen. Bewährte Mac-Standards auf dem ST? Das neueste Produkt von Application Systems scheint es zu beweisen, denn wer das Mac-Programm *WriteNow* kennt, wird sich sofort in Script heimisch fühlen.

Nachdem 1987 Signum! seinen Siegeszug auf dem ATARI ST begonnen hat, kommt jetzt die zweite Textverarbeitung von Application Systems auf den Markt. Sie soll wohl diejenigen ansprechen, denen Signum! zu teuer und/oder zu umständlich in der Bedienung ist. Gleichzeitig muß man Script bei einem Preis von DM 198,- aber auch als Konkurrenz zu Wordplus ansehen, da sie im gleichen Preisbereich angesiedelt sind.

Wichtigster Punkt von Script ist in jedem Fall seine leichte Bedienbarkeit, denn die Funktionen sind übersichtlich und fast alle Menüpunkte auf Tastenkombinationen gelegt (s. Bild 1). Es besteht sogar die Möglichkeit, die Tastenkombinationen mittels eines Resource-Construction-Sets individuell zu verändern, da Script sich die entsprechenden Buchstaben aus den Pull-Down-Menüs holt.

Script kann maximal vier Texte gleichzeitig verwalten, zwischen denen man mittels Tastendruck hin- und herblättern kann. Accessories sind im Gegensatz zu Signum! zugelassen. Eins wird sogar gleich mitgeliefert, denn es konvertiert Wordplus- und MS-Word-(RFT-Format)-Dateien in sogenannte Script-Import-Dateien (STI), die mittels der Script-Funktion *ASCII einfügen* eingelesen werden können. Der besondere Witz daran ist, daß Textattribute wie fett, kursiv, unterstrichen usw. automatisch übernommen werden.

Script verfügt über eine schnelle Scrollgeschwindigkeit, die zwar nicht an Tempus herankommt, aber es handelt sich hier auch nicht um einen Editor zum Programmieren. Leider läuft der Cursor derzeit noch etwas nach, so daß man auch schon mal über sein Ziel hinausschießen kann. Dies kommt ja leider bei vielen Programmen vor und läßt sich wohl GEM-bedingt nicht sauber lösen.

Signum!s Erbe

Die Textdarstellung erfolgt je nach Zeichensatz proportional oder unproportional, wobei wir bei einem weiteren wichtigen Vorteil Scripts angekommen sind, denn es arbeitet mit Signum!-Zeichensätzen. Damit stehen Script über 900 verschiedene Zeichensätze zur Verfügung, die ja teilweise schon professionell eingesetzt werden (s. Bild 2). Mitgeliefert werden drei Zeichensätze der Rokwel-Familie (11 Punkt prop., 11 Punkt unprop. und 15 Punkt). Genau wie in Signum! lassen sich diese Zeichensätze in den Arbeitsspeicher laden und in einer Dialogbox anzeigen (Bild 3), so daß man einen schnellen Überblick über das Aussehen der Fonts bekommt. In Script wurde die Box sogar insoweit erweitert, daß man zwischen den Zeichensätzen blättern kann.

Im Gegensatz zu Signum!, das maximal sieben Fonts verwalten kann, stehen einem in Script maximal 256 zur freien Auswahl. In Anbetracht des ewig währenden Speicherplatzmangels der ST-Besitzer sollte man sowieso nicht so viele Zeichensätze laden, vom rein typografischen Standpunkt einmal abgesehen. In der Regel wird man fast immer dieselben Zeichensätze verwenden. In der Menülei-

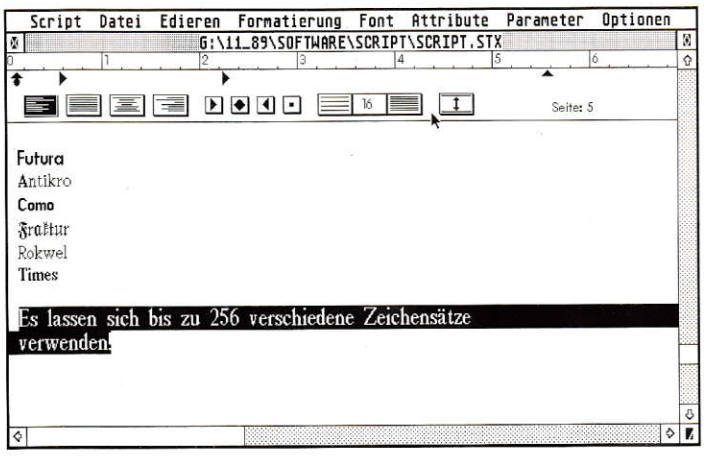


Bild 2: In Script lassen sich bis zu 256 Signum!-Zeichensätze verwenden.

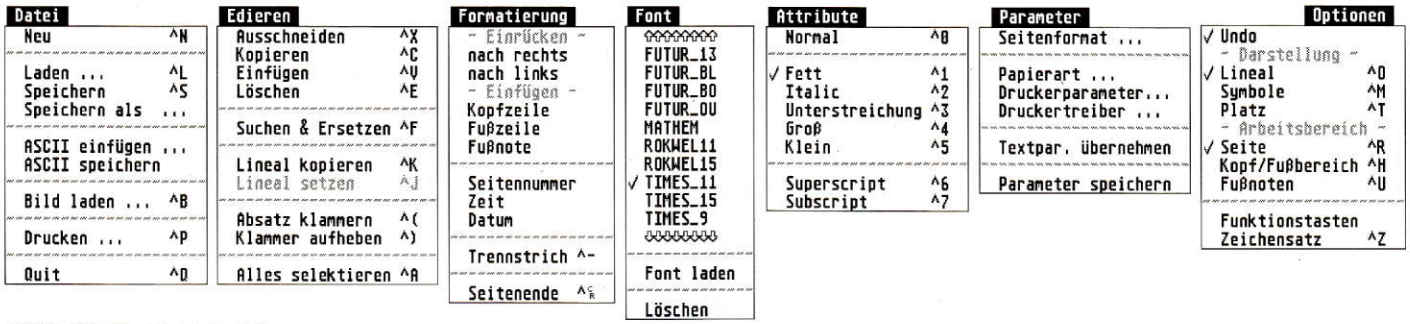


Bild 1: Die Menüleiste Scripts

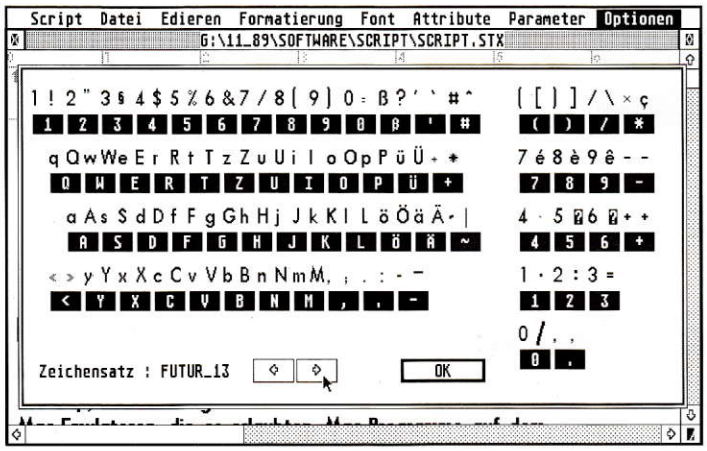


Bild 3: Die Zeichensätze lassen sich übersichtlich anzeigen.

ste werden maximal zehn von ihnen angezeigt. Ab elf erscheinen in der Leiste Pfeile, über die weitergeblättert werden kann. Ein Manko ist, daß die Menüleiste dabei immer wieder zuklappt und von neuem heruntergeklappt werden muß. Aber damit läßt sich leben; es ist sowieso nicht sinnvoll, mehr als zehn Zeichensätze in einem Dokument zu verwenden. Zeichensatzeinstellungen lassen sich wahlweise zusammen mit Attributen wie z.B. fett, groß, klein usw. auf die zehn Funktionstasten legen, damit man schnell zwischen ihnen wechseln kann (Bild 4). Hat man einen festen Fontstamm, den man ständig benötigt, kann man sich ein leeres Dokument mit den Fonts zusammen abspeichern und spart sich so das ewige Laden.

Übrigens lassen sich auch die übrigen Parameter abspeichern, durch die u.a. der Pfad für die Zeichensätze, das Seitenformat, der Druckertreiber usw. bestimmt werden.

Leicht markiert

Mit Script lassen sich Texte mit der gedrückten Maustaste leicht zeichenweise markieren. Will man dagegen größere Abschnitte selektieren, markiert man einfach den Beginn des Abschnittes, geht an dessen Ende und drückt Control-Shift. Schon hat man den gewünschten Bereich. Nebenbei gibt es auch eine Funktion *Alles selektieren*, durch die der gesamte Text markiert wird. Dies ist besonders dann notwendig, wenn man z.B. seinen Text umformatieren will. Auch das Löschen von Texten geht ähnlich einfach vonstatten. Dazu muß man den zu löschenden Text markieren und drückt anschließend einfach die Backspace-Taste. Sollte man aus Versehen, oder weil man es sich anders überlegt haben sollte, seine Entscheidung bereut haben, hilft ein Druck auf die Undo-Taste, und der Text ist wieder da. Dies funktioniert aber nur einen Arbeitsschritt rückwärts (1 Undo-Ebene).

Das Lineal

Ein wichtiger Bestandteil Scripts ist ohne Zweifel das Lineal. Es enthält u.a. alle Tabulatoreinstellungen. Dabei kann man einen Tabulator als links-, rechtsbündig, zentriert oder numerisch definieren. Er wird einfach mittels der Maus an der gewünschten Stelle der Maßskala plaziert und läßt sich dort auch leicht verschieben. Ebenso kann man den linken und rechten Rand und Einrückungen festlegen. Praktisch ist, daß beim Anklicken des Lineals der aktuelle Absatz gleich selektiert wird, da ein Lineal nur für diesen Bereich gilt. Natürlich lassen sich verschiedene Linealeinstellungen in einem Text benutzen. Die Maßskala kann zwischen Zoll und cm umgestellt werden, wodurch ein exaktes Arbeiten möglich wird.

Nicht zu vergessen ist, daß man im Lineal auch zwischen links- und rechtsbündigem Flatter-, Blocksatz und zentrierter Darstellung wählen kann. Auch den Zeilenabstand kann man hier pixelgenau (5-99) verstellen. Warum hier die Einheit Pixel und nicht Punkt Cicero gewählt wurde, ist allerdings unbegreiflich, da man, wenn man schon diese Möglichkeit einbaut, dies auch noch hätte berücksichtigen können, denn nur so wäre ein typografisch genaues Schreiben möglich. Leider verfügt Script über keine automatische Trennhilfe, d.h. man muß gegebenenfalls jedes Wort von Hand trennen. Dazu ist eine weiche Trennung als Tastenkombination vorhanden. Schöner wäre hier ein Trennlexikon, das die normalen Trennregeln beherrscht und für das man eine erweiterbare Ausnahmedatei erstellen könnte. Zwar verfügt Script über keinen Spaltensatz, doch auch schon bei normalem Blocksatz kann eine Trennhilfe von Nutzen sein.

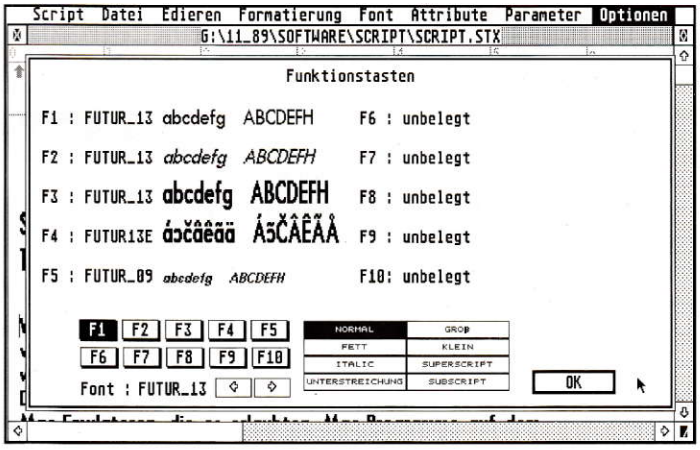


Bild 4: Zeichensätze kann man inklusive Schriftattribute auf Funktionstasten legen.

Will man bestimmte Textabschnitte auf keinen Fall durch den automatischen Seitenumbruch unterbrechen lassen, kann man Absätze klammern, d.h. sie werden dann auf einer Seite komplett dargestellt. Gegebenenfalls wird eine neue Seite angefangen.

STX stellt sich vor

Natürlich gehört zu einer neuen Textverarbeitung auch ein neues Dateiformat; in diesem Falle handelt es sich um das Script-Text-Format (STX). In ihm sind die verwendeten Zeichensätze und natürlich der gesamte Text mit Textattributen usw. enthalten. Darüber hinaus kann Script auch ASCII-Dateien hinzuladen. Leider ist eben nur ein Hinzuladen möglich, wodurch man gezwungen ist, zuerst einen STX-Text anzulegen, wenn man eine ASCII-Datei laden möchte. Hier ist Wordplus weniger wählerisch.

Script bietet auch die Möglichkeit, Bilder im Text zu positionieren. Hier ist es aber ein großer Nachteil, daß man dabei vernachlässigt hat, eine freie Positionierbarkeit eines Bildes als Objekt zu implementieren. Ein Bild wird von Script leider nur als quasi großer Buchstabe angesehen. Ein Formsatz bzw. freies Umfließen eines Bildes ist nicht möglich, da der Cursor die Bildhöhe annimmt (s. Bild 5). Diese Funktion ist also nur für Bilder ohne nebenstehenden Text verwendbar.

Positiv dagegen fällt auf, daß man Bildausschnitte frei bestimmen kann. Dazu ist nach dem Laden des Bildes ein Gummiband vorhanden. Ferner kann man die Bildgröße über eine Dialogbox oder mit der Maus verändern. Für Freunde spezieller Effekte kann man das Bild natürlich auch unproportional vergrößern, sprich verzerren. Script akzeptiert als Bildformate das 32k-Screen- (PIC od. DOO), das Signum!- (I??), das STAD- (PAC) und das GEM-Image-Format (IMG). Letzteres ist allerdings auf 640 x 400 Pixel beschränkt.

Zeigt her eure Füße

Fußnoten sind für professionelle Texte unabkömmlich. Script unterstützt kontinuierlich durchnumerierte, bei denen keine Nummernwiederholungen vorkommen dürfen. Es läßt sich aber eine Startnummer angeben. Bilder sind dort nicht zugelassen, was ja auch keinen Sinn ergäbe. Sind Fußnoten zu lang, werden sie automatisch auf die nächsten Seiten über-

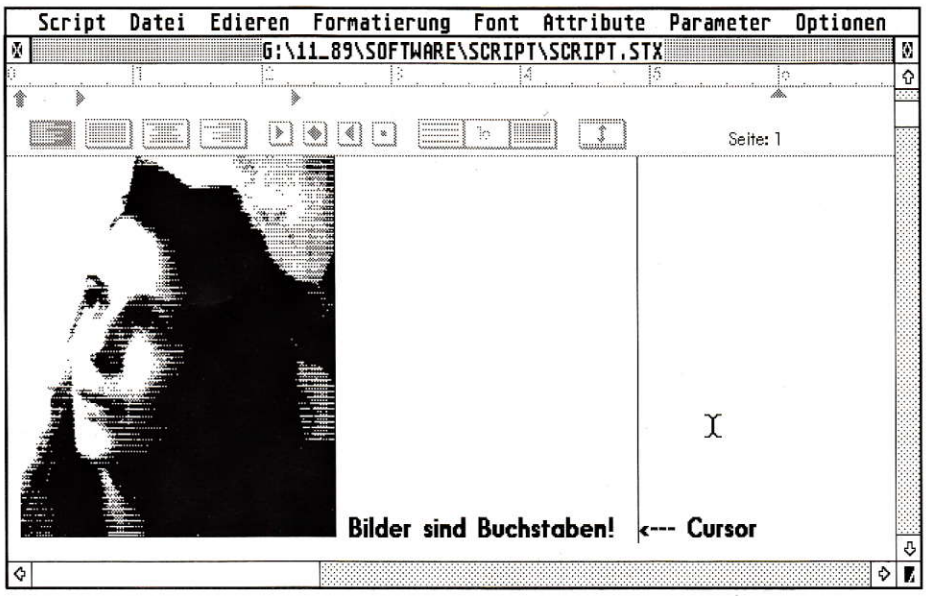


Bild 5: Bilder sind nur als Buchstaben ladbar. Dadurch ist kein Umfließen des Bildes möglich, da der Cursor die Bildhöhe annimmt.

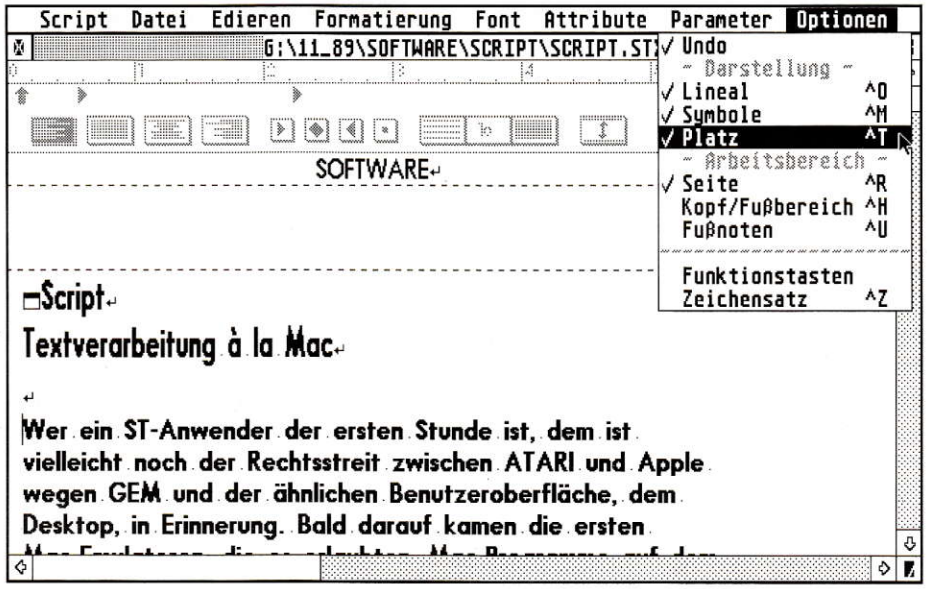


Bild 6: Zwischenräume, Zeilenende, Symbole usw. lassen sich ebenfalls wahlweise darstellen.

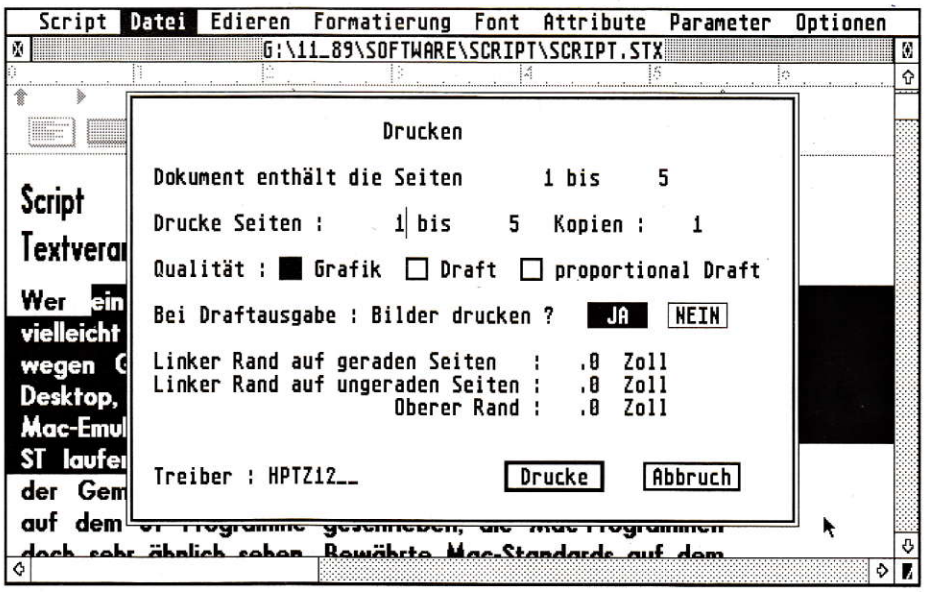


Bild 7: Das Druckermenü Scripts

nommen. Zum Edieren der Fußnote öffnet sich ein Fußnotenfenster, in dem auch ein Lineal mit seinen Formatierungsmöglichkeiten zur Verfügung steht.

Neben den Fußnoten kann man eine automatische Seitennumerierung einstellen und Fuß- und Kopfzeilen eingeben. Bei letzteren besteht die Möglichkeit, sie wahlweise auf der rechten, linken oder auf beiden Seiten auszugeben. Auch hier steht ein Lineal zur Verfügung.

Eine praktische Idee ist das Einfügen von aktueller Zeit und Datum an der Cursorposition durch Anklicken in der Menüleiste. Man kann dabei zwischen einer Text- oder Symbolausgabe wählen. Bei letzterer erscheint ein Symbol auf dem Bildschirm, das ein Platzhalter ist, und wodurch erst beim Ausdruck das aktuelle Datum bzw. Zeit eingefügt wird. So kann man sich z.B. ein leeres Briefformular erstellen und hat automatisch immer das richtige Datum.

Bei leeren Formularen empfiehlt sich übrigens immer der Name *NAMENLOS.STX*, da bei diesem Namen nur die Funktion *Speichern als* zum Abspeichern des Textes benutzt werden kann, wodurch ein Überschreiben des Leerformulars verhindert wird.

Die Funktionen Undo, Lineal, Symbole und Platz lassen sich übrigens auch abschalten. Schaltet man die Symbole ein, werden neben Uhrzeit und Datum auch Fuß- und Kopfzeilen, Seitennummern und Trennsymbole angezeigt (s.Bild 6). Bei Platz werden Tabulatoren, Seiten-, Zeilenende usw. als Zeichen dargestellt.

Schwarz auf weiß

Beim Drucken kann Script natürlich wieder auf die riesige Auswahl von Druckerzeichensätzen Signum!s zurückgreifen. Es stehen eine große Anzahl von Druckertreibern zur Verfügung, mit denen so ziemlich alle gängigen Drucker angesteuert werden können. Ähnlich wie bei anderen Application-Programmen können Papierart und Druckerparameter für 9-Nadel-, 24-Nadel- und Laserdrucker eingestellt werden. Es läßt sich zum Drucken zwischen Grafik- (schön) und Draftausdruck (schnell) wählen, wobei bei letzterem noch die Wahl zwischen proportionalem und unproportionalem Druck besteht (s. Bild 7). Ferner kann man seine Werke mit und ohne Bilder zu Papier bringen. Zuletzt gibt es noch einen Leckerbissen für alle Handbuchschreiber: Man kann den linken Rand für gerade und ungerade Seiten verschieden einstellen.

Fazit

Script bietet für sein Geld schon eine ganze Menge. Ob es allerdings gegen den übermächtigen Konkurrenten Wordplus ankommen wird, sei noch dahingestellt. Seine Vorteile liegen eindeutig bei den Zeichensätzen und Formatierungsmöglichkeiten, den besseren Ausdruck nicht zu vergessen. Nachteile zeigen sich allerdings dagegen in der Bildpositionierung, der Trennhilfe und nicht zuletzt auch im Lieferumfang, denn Wordplus wird neuerdings mit Dateiverwaltung, Serienbrieferstellung und 1st-Xtra ausgeliefert. Trotzdem glaube ich, daß Script seinen Weg machen wird, da mit Sicherheit, wie bereits oben erwähnt, viele, die von Signum! abgeschreckt sind, auf es einsteigen werden. Hinzu kommt, daß sich mit Script dank seiner Scrollgeschwindigkeit und seiner Bedienungsfreundlichkeit auch recht flott arbeiten läßt.

Martina Pfahl

Bezugsadresse:

Application Systems /// Heidelberg
Englerstr. 3
6900 Heidelberg
Tel.: 06221/300002

AMIGO ST DIE ERSTE.

Ludwig fun!

Tamtam. Es ist soweit. Sie ist da: die SoundMachine II ST, der Nachfolger der legendären Version I. TommySoftware hat sich mal wieder mächtig in die Spendierhosen gegriffen und einen Leckerbissen nach dem anderen produziert. Voila:
Neue Oberfläche, Drumcomputer, Mini-SoundMachine, MusiX32-Converter, Shapeconverter und noch einen ganzen Berg weiterer Sensationen, den es zu feiern gilt. Registrierte Anwender erhalten selbstverständlich ein Upgrade. Und auch an die, die lieber etwas zögern, haben wir gedacht: den idealen Fieps-pieps-Programmvernichter gibt's auch als Dröhndemo für leise DM 10.- (Schein). Damit Ludwig fun hat.

TommySoftware®
Überlegen durch Kreativität

Selchower Str. 32
D-1000 Berlin 44
Tel. 0 30/ 621 40 6-3
Fax 0 30/ 621 40 6-4

Modems

Der Anschluß zur Außenwelt

Um Ihnen eine Übersicht des heutigen Stands bei Modems zu geben, haben wir die Modelle der bekanntesten Anbieter getestet. Es wurden lediglich diejenigen getestet, die auch mit 2400 Baud (gleich zirka 240 Zeichen pro Sekunde) Übertragungsgeschwindigkeit arbeiten, da langsamere Baud-Raten aus der "Mode" kommen und kaum mehr verlangt werden.

Wenn Sie Bildschirmtext betreiben möchten, sollten Sie sich Ihre Entscheidung nochmal gründlich überlegen: Möchten Sie wirklich? Wenn Sie dann immer noch möchten, suchen Sie sich, sofern Sie nicht in einer Großstadt wohnen, ein Modem bzw. einen Akustikkoppler aus, der auch 1200/75 Baud hat [1200 Baud empfangen, 75 Baud senden (gähn)].

Beim Test ist zu beachten, daß nur die wichtigsten Ergebnisse in einer übersichtlichen Tabelle festgehalten sind. Dadurch können Sie sich einen schnellen Überlick über die Ergebnisse des Tests verschaffen. Alle anderen Testergebnisse, die nicht in die Tabelle passen, können Sie in diesem Artikel finden.

Was ist wichtig?

Zuerst sollten Sie sich überlegen, ob Sie nur ab und zu Datenfernübertragung betreiben möchten, oder ob Ihr Hobby ausgiebiger betrieben werden soll. Wenn Sie weniger als ein Mal pro Woche Daten übertragen wollen, reicht auch ein Akustikkoppler. Hier steht Ihnen mit 2400 Baud nur das Dataphon 2400 B zur "Auswahl". Das Dataphon hat einen entscheidenden Nachteil: Man kann sich nicht in Mailboxen bzw. Datensysteme einwählen, die mit einem Discovery 2400 C-Modem arbeiten. Erfahrungsgemäß machen das jedoch ca. 40-50% aller Mailboxen. Mit 300 Baud konnte keine Mailbox aufgestöbert werden, die das Dataphon 2400 B nicht gemeistert hätte. Ein weiterer Nachteil: Sobald die Verbindung über mindestens eine Fernvermittlungsstelle geht, wird die Störanfälligkeit des Kopplers so hoch, daß mitunter nur noch Datenmüll auf dem Bildschirm erscheint.

Der Löwenanteil der getesteten Modems ist nicht durch die Post zugelassen. Wenn Sie aber unbedingt auf dem Vorhaben beharren wollen, ein postalisch zugelassenes Modem zu ersteigern, bleiben Ihnen nur wenige Möglichkeiten: Das Fury- und das Mikrolink-Modem. Natürlich können Sie auch direkt bei der Post ein Modem bestellen (wenn Sie Krösus sind). Ein Wort dazu: Wir wollten auch Modems von der Post testen, die uns jedoch nicht kostenlos zur Verfügung gestellt worden wären. Für jedes getestete Modem hätten 65,- DM Anschlußgebühr und eine Grundgebühr von 27,- DM plus 5,- DM monatlich bezahlt werden müssen. Sollten Sie also aus irgendeinem Grund ein postzugelassenes Modem benötigen, entscheiden Sie sich besser für eines der beiden getesteten.

MNP-Standard

Einige Modemhersteller bieten bereits Modems an, die nach dem MNP 5-Standard arbeiten. In diesen Geräten ist eine Datenkomprimierungsroutine eingebaut, die die effektive Übertragungsgeschwingkeit um ein Vielfaches erhöht. Da noch nicht alle Hersteller diese Modems anbieten können, sind die Ergebnisse des MNP 5-Tests nicht in die Bewertung eingeflossen. Wir werden in einem späteren Heft einen reinen MNP 5-Test durchführen.

Natürlich sollten Sie auch auf "Kleinigkeiten" wie gute Lautsprecher und einen externen Lautstärkeregler achten, damit Sie komfortabel mit Ihrem Modem arbeiten können. Es ist ungemein nervenaufreibend, wenn die Lautstärke des Modems nicht vernünftig oder gar nicht geregelt werden kann.

Mailbox-Betreiber

Wenn Sie eine eigene Mailbox betreiben möchten, müssen Sie zwischen drei Modems Ihre Auswahl treffen: dem Lightspeed 2400 C, dem Discovery 2400 C und dem Best 2400+. Nur diese Modems beherrschen die Baud-Raten 300, 1200 und 2400 Baud auch gleichzeitig mit automatischer Erkennung im Answer-Modus. Als "Normal-Benutzer" dürften Sie den Answer-Modus kaum benötigen und sollten deshalb hier auch keine Priorität setzen.

Lassen Sie sich nicht durch die Post oder anderslautende Berichte ins Bockshorn jagen: Die Neuordnung der Post kommt bald. Dann werden auch solche Modems zugelassen sein, die in anderen europäischen Ländern eine Zulassung erhalten haben. Erkundigen Sie sich also nach Zulassungen in anderen europäischen Ländern, wenn Sie für die Zukunft vorsorgen möchten.

Ein Unglücksfall!

Ich darf Ihnen leider keine Anleitung zum Anschluß eines Modems geben. Nehmen wir jedoch an, eine Gummipalme stürzt auf Ihre Telefonanschalteeinrichtung (amtsdeutsch), also die häßliche Dose, aus der Ihr Telefonkabel kommt. Während Sie die Dose wieder festschrauben, werden Ihnen zwei Kabel auffallen, die hier ankommen. Würden Sie jetzt versuchen, diese Kabel an das grüne und rote Kabel des Modems anzuschließen (welches an welches angeschlossen würde, wäre egal), würden Sie mit Verwunderung feststellen, daß das Modem bereits angeschlossen wäre...

MP



	Lightspeed 2400 [13]	Lightspeed 2400 C	Best 2400+	Discovery 2400 C	GVC Supermodem [11]	Tornado 2400 [12]
Baudraten	1200, 2400	300,1200,2400 1200/75	300, 1200, 2400, 1200/75	300, 1200, 2400	300, 1200, 2400	300, 1200, 2400
Baud bei Autoanswer	1200+2400	300+1200+2400 [1]	300+1200+2400	300+1200+2400 [2]	300, 1200+2400	300, 1200+2400
Busy-Erkennung	ca. 10%	ca. 10%	ca. 90%	100%	0%	0%
DIP-Schalter	ja	ja	ja [17]	nein	nein	nein
nichtflüchtiges RAM	nein	nein	ja [17]	ja	ja	ja
Befehlssatz	erw. Hayes [3]	erw. Hayes [3]	erw. Hayes	erw. Hayes	erw. Hayes	erw. Hayes
Anschluß	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]
Netzteil	sehr gut	sehr gut	gut	gut	gut	gut
Lautsprecher	sehr gut	sehr gut	sehr gut	gut	ausreichend	ausreichend
Lautstärkeregler	sehr gut	sehr gut	keiner	sehr gut	gut	gut
Wärmeentwicklung	bleibt kühl	bleibt kühl	bleibt kühl	bleibt kühl	bleibt kühl	bleibt kühl
Zuverlässigkeit	sehr gut	sehr gut	sehr gut	sehr gut	gut	gut
Zulassung	nein	nein	nein	nein	nein	nein
Ausstattung	HNK [5]	HNK [5]	HNK [5]	HNK [5]	HNK [5]	HNK [5]
Hilfsfunktion "AT?"	ja [18]	ja [18]	nein	nein	nein	nein
Preis inkl. MWSt. (DM)	389,-	479,- / 679,- [10]	449,-	399,- / 469,-	567,- DM	399,-

[1] Lightspeed 2400 C: bestes Autoanswer aller getesteten Modems

[2] Discovery 2400 C: Autoanswer nur gut bei Bestriebssystemversion 4 und 8

[3] erweiterte Hayes-Befehle, jedoch ohne nichtflüchtiges RAM

[4] erweiterte Hayes-Befehle, jedoch Befehl ATA nur möglich nach RING

[5] H = Handbuch, N = Netzteil, K = RJ 11-Anschlußkabel

[6] Supra-Modem in unterschiedlichen Versionen: Mit oder ohne Terminalprogramm und RS232-Kabel

[7] TAE 6 = Telefonanschalteeinrichtung (amtsdeutsch) = Anschlußdose für Telefonkabel

[8] die mitgelieferte Telix-Version 3.11 ist lizensiert und deutsch

[9] Dataphon 2400 B läßt sich über die Schnittstelle in Klartext konfigurieren

[10] Compuscience: 479,- DM

[11] GVC Modem baugleich mit Tornado Modem

[12] Tornado Modem baugleich mit GVC Modem

[13] Lightspeed 2400 baugleich mit Lightspeed 2400 C bis auf 300-Baud-Modus

[14] RJ 11-Anschlußkabel für amerikanische Dosen. Benötigte Kabel sind grün und rot

[15] Supra-Lautsprecher klingt blechern

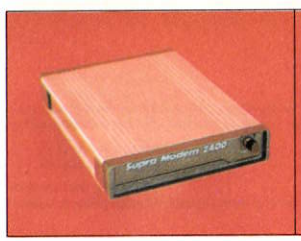
[16] Smartlink hat zwar Lautsprecher, jedoch keinen Regler! Lautsprecher ist zu laut

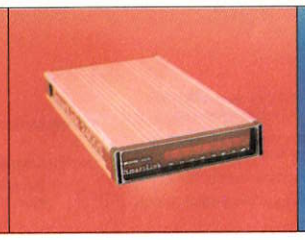
[17] Grundeinstellungen über DIP-Schalter, Rest über nichtflüchtiges RAM

[18] Ausführung beim Lightspeed 2400 (C) zwar praktisch, jedoch nicht nach Hayes

[19] Procomm 2.42 in lizensierter, deutscher und dem Modem angepaßter Version

[20] 300-Baud-Modus bei Fury in Arbeit








	Supra Modem 2400	SmarTeam 2400	SmartLink 2400	Fury 2400 TI	Mikrolink 2400 T2	Dataphon 2400 B
Baudraten	300, 1200, 2400	300, 1200, 2400	300, 1200, 2400	1200, 2400 [20]	300, 1200, 2400	110, 300, 1200, 2400, 1200/75
Baud bei Autoanswer	300, 1200+2400	300, 1200+2400	300, 1200+2400	1200+2400	300+1200+2400	300, 1200+2400
Busy-Erkennung	ca. 30%	ca. 30%	ca. 50%	90%	100%	keine
DIP-Schalter	nein	nein	nein	ja	ja	nein
nichtflüchtiges RAM	ja	ja	ja	ja	ja	ja
Befehlssatz	erw. Hayes	erw. Hayes	erw. Hayes	erw. Hayes [4]	erw. Hayes [4]	keiner
Anschluß	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	RJ11/Selbstbau [14]	Post/Gebühr	Post/Gebühr	über Hörer
Netzteil	gut	ausreichend	gut	sehr gut	gut	gut
Lautsprecher	krächzt [15]	mangelhaft	viel zu laut	keiner	sehr gut	keiner
Lautstärkeregler	ausreichend	gut	fehlt! [16]	entfällt	sehr gut	
Wärmeentwicklung	wird heiß	wird warm	bleibt kühl	bleibt kühl	bleibt kühl	bleibt kühl
Zuverlässigkeit	gut	sehr gut	sehr gut	sehr gut	sehr gut	ausreichend
Zulassung	nein	nein	nein	ja	ja	ja
Ausstattung	HNK [6]	HNK [5]	HNK [5]	HN, Postkabel, Procomm 2.42 [19], TAE 6 [7], RS-Kabel, Amaris BTX Demo, Postantrag [5]	HN, Telix 3.11 [8], Postkabel, Postantrag , RS-Kabel [5]	HN [5]
Hilfsfunktion "AT?"	nein	nein	nein	nein	nein	ja [9]
Preis inkl. MWSt. (DM)	398,- / 448,- [6]	798,-	399,-	2277,72	2109,-	598,- / 698,-

Händleradressen:

Compuscience Computerhandels GmbH, Steinwegpassage 5, 2000 Hamburg 36, Tel. 040/353941: Smartlink 2400, Lightspeed 2400, Lightspeed 2400 C

Wörlein Gmbh & Co. KG, Gewerbestr. 12, 8501 Cadolzburg, Tel. 09103/8294: Dataphon 2400 B

BIELING Computersysteme, Spitzwegstraße 11, 4350 Recklinghausen, Tel. 02361/181485: BEST 2400+

DFÜ-Shop, Kolonnenstraße 33, 1000 Berlin 22, Tel. 030/7827118: Discovery 2400 C

CSH Ingenieurbüro, Schilderring 19, 8751 Großwallstadt, Tel. 06022/24405: Supra Modem 2400

resco electronic, Kobelweg 68a, 8900 Augsburg, Tel. 0821/407027: GVC Modem 2400

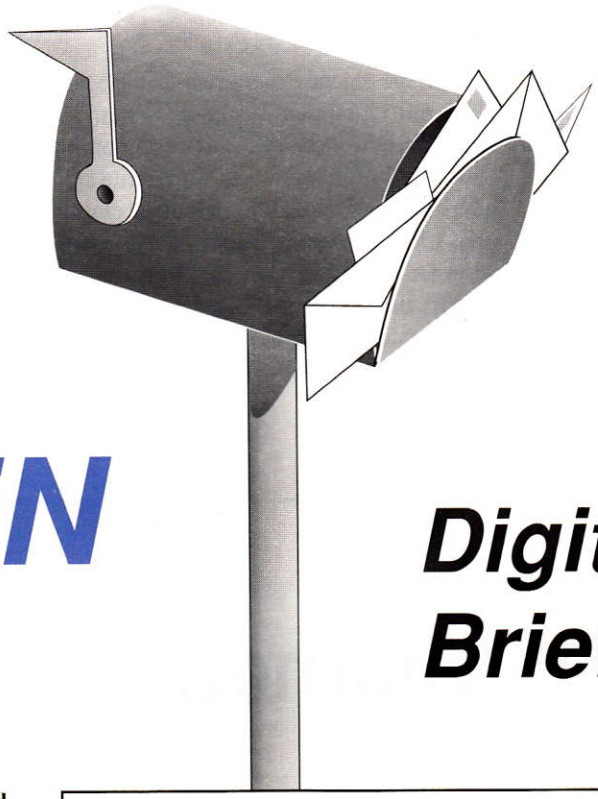
Rechnersysteme Information und Technik GmbH, Freunder Weg 106, 5100 Aachen, Tel. 0241/59035: Lightspeed 2400 C

RVS Datentechnik GmbH, Hainbuchenstraße 2, 8000 München 45, Tel. 089/3510071: Team 2400 ET Modem

Carl Schewe, Essener Straße 97, 2000 Hamburg 62, Tel. 040/5270321: Tornado 2400

Elsa GmbH, Monheimsallee 53, 5100 Aachen, Tel. 0241/477890: Mikrolink 2400 T2

Dr. Neuhaus Mikroelektronik GmbH, Haldenstieg 3, 2000 Hamburg, Tel. 040/553040: Fury 2400 TI



MAIL-BOXEN

Digitale Briefkästen

Wer in die DFÜ-Szene einsteigt, wird sicherlich schnell bemerken, daß er/sie sich ein kostenintensives Hobby ausgesucht hat (viele Grüße an die POST). Um die Kosten etwas zu mindern, kann man sich eine eigene Mailbox anlegen. So muß man nicht mehr andere Systeme anrufen, sondern stellt nur eine Box zu Hause bereit und läßt sich stattdessen anrufen. Da zum einen schwer an solche Programme zu gelangen und zum anderen kaum bekannt ist, welche es überhaupt gibt und was sie können, bieten wir Ihnen eine kleine Übersicht der momentan "vier großen" Programme.

Was ist eine Mailbox?

Sicher können nicht alle mit dem Begriff "Mailbox" etwas anfangen. Eine Mailbox ist ein elektronischer Briefkasten. Sofern Sie ein Modem oder einen Akustikkoppler besitzen, können Sie mit einer Mailbox kommunizieren, sich dort Informationen besorgen, Nachrichten hinterlassen, persönliche Briefe schreiben oder beispielsweise auch an einem Netzverbund teilnehmen. In einem Netzverbund sind mehrere Mailboxen zusammengeschlossen und tauschen in bestimmten regelmäßigen Abständen neue Nachrichten aus. Je nach Größe des Verbunds können Sie dann täglich mehr oder weniger viele neue Nachrichten abrufen.

	MagicBOX ST	Picobox	Skylink	Starmail
Netzverbund	ja [1]	ja [2]	nein	ja [3]
Netzebenen	3	2	0	1
Netztransfer	täglich	täglich	keiner	täglich
Netzkosten	gut	gut	-	ausreichend
Netzstruktur	ringförmig	ringförmig	-	sternförmig
Steuerung	Befehle	Menüs+Befehle	Menüs	Menüs+Befehle
Protokolle	X, Z [4]	X, Kermit [5]	X	X, Y [6]
Point-Systeme	ja	nein [7]	nein	nein
Emulationen	52, 100 [8]	Fullscreen 52	Fullscreen 52	Fullscreen 52
Befehlsumfang	über 80	über 80	26	ca. 30
Lieferumfang	gut	ausreichend	gut	gut
Preis	349,-/29,- [9]	400,- DM	198,- DM	398,- DM

[1] 56 Systeme im MagicNET, Netzverbund mit Piconet und EIS, Gateway zu Zerberus
[2] 5 Systeme im Piconet, Netzverbund mit MagicNET
[3] 11 Systeme im Starnet
[4] Y-Modem und Kermit folgen in der nächsten Version
[5] Y-Modem und Z-Modem folgen in der nächsten Version
[6] Z-Modem in einer der nächsten Versionen
[7] Als Point kann ein MagicNET-Point benutzt werden
[8] zusätzliche Emulation für C64-VipTerm-VT52-Anwender
[9] Vollversion 349,- DM; Pointversion 29,- DM

Tabelle: Alle Programme in der Übersicht

Was muß beachtet werden?

In einem Mailbox-Programm sollte die Möglichkeit gegeben sein, komfortabel Nachrichten in verschiedene Unterabteilungen (sogenannte "Bretter") zu schreiben bzw. daraus zu lesen. Dazu sollten verschiedene genormte Übertragungsprotokolle zur Verfügung stehen. Möglich sind beispielsweise X-Modem, X-Modem 1k, X-Modem CRC, Y-Modem, Y-Modem Batch, Y-Modem G, Z-Modem, Bimodem, Q-Modem, Kermit und

viele andere. Je mehr Protokolle zur Auswahl stehen, desto besser können die Möglichkeiten der Mailbox genutzt werden. Wichtig ist auch, daß beliebige Dateien in einem bestimmten Bereich der Mailbox abgelegt werden können. Das kann entweder in normalen Brettern, in extra dafür angelegten Up- und Download-Bereichen sowie über die "Private Mail" geschehen. Natürlich sollte all das komfortabel durch möglichst viele Befehle und/oder Parametereinstellungen steuerbar sein. Sofern Sie Wert auf ein Mailbox-Netz legen, achten Sie darauf, daß Ihnen die Größe des Netzes sowie dessen Ausbaufähigkeit ausreicht. Es bringt wenig, in einem Mailbox-Netz teilzunehmen, das schlechte Überlebenschancen hat.

Wenn Sie eigene Programme in die Mailbox mit einbinden möchten, sollten Sie auf die Erweiterbarkeit des Programms achten. Einige Mailbox-Programme sind dermaßen geschlossen, daß nicht einmal das kleinste eigene Programm eingebaut werden kann. Ein wichtiges Kriterium ist auch die Geschwindigkeit des Programms. Dabei sollten Sie nicht die Diskettengeschwindigkeit messen, da es wenig sinnvoll ist, eine Mailbox auf Diskettenstation(en) zu betreiben (sorry: einige AMIGA-Boxen verstehen es nie...). Auch die Zeit, die Sie täglich mit dem Programm bei der Wartung und der Pflege verbringen möchten, ist ein wichtiges Kriterium. Hier sollten Sie einen Kompromiß zwischen Benutzerfreundlichkeit für die Anrufer und Ihrem Arbeitsaufwand schließen, da Sie kaum beides in einem Programm vereinigen können. Nicht zuletzt zählt natürlich Ihre Finanzlage als Kaufentscheidung...

MP

Starmail (Starnet)

Auch dieses Programm besitzt einen Anschluß an ein Netzwerk. Im Starnet sind momentan knapp 10 Mailboxen vernetzt. Das Programm ist grundsätzlich verschieden von MagicBOX ST konzipiert.

Starmail wartet mit den üblichen Menüleisten auf, in denen wir diverse Funktionen finden. Interessant: Im Programm ist ein Bildschirmschoner eingebaut; eine recht sinnvolle Funktion für ein Mailboxprogramm. Der Schoner läßt sich natürlich bei Bedarf auch wieder ausschalten. An den Menüleisten (Bild 1) kann man bereits deutlich erkennen, daß Starmail weniger für die Benutzer, als vielmehr für den Systembetreiber konzipiert wurde. Hier kann man in bester GEM-Manier alles nach Herzenslust verändern, was zu verändern ist. So können Nachrichten, die an den Systembetreiber adressiert sind, gelesen, neue Nachrichten betrachtet oder Anrufprotokolle von verschiedenen Tagen bearbeitet werden. Hier läßt sich beispielsweise auch feststellen, welcher User wann wo welches Programm in die Mailbox geschickt hat. Weiterhin läßt sich überprüfen, ob ein Gast sich als Benutzer in der Mailbox eintragen möchte. Hier zeigt sich ein Manko: Es ist nicht einstellbar, daß sich neue User frei eintragen können; sie müssen grundsätzlich den Umweg über den Brief an den Betreiber gehen. An diesem Problem wird aber schon gearbeitet. Was passiert aber momentan in diesem Falle weiter: Man schaut sich den Antrag an und schreibt ihn ab oder läßt ihn ausdrucken, um ihn dann per Hand wieder einzugeben. Bei einer der nächsten Programmversionen wird der Eintrag direkt nach dem Lesen zu übernehmen sein. Wie lassen sich nun die User gliedern? Starmail unterscheidet zwischen Usern, "Kategorie I"- und "Kategorie II"-Usern. Normale User sind beispielsweise Gäste und solche Benutzer, die nicht allzu häufig anrufen. In der Kategorie I landet man, wenn man zwar hin und wieder mal anruft, aber nicht gerade der aktivste ist. Kategorie II wird also meistens dann vergeben, wenn ein User recht aktiv teilnimmt. Natürlich kann der Betreiber all diese Einteilungen frei vornehmen. Mit Kategorie II stehen dem User dann auch noch einige Befehle zusätzlich zur Verfügung.

Im System

Sehen wir uns das Login an (Bild 2). Hier wird erst recht deutlich, daß Starmail für Sysops konzipiert ist, denn dem Benutzer stehen deutlich weniger Befehle als beispielsweise bei MagicBOX ST zur Verfügung. Noch deutlicher wird es anhand des Menüeintrags, mit dem man sich einloggen kann: Er nennt sich "Test", nicht etwa "Login". Nachdem man sich mit seiner recht unpersönlichen User-Nummer eingeloggt hat, erscheint das Prompt, das zur Eingabe eines Befehls auffordern soll ("*>"). Man kann sich auch unter seinem Namen einloggen. Wenn das System den

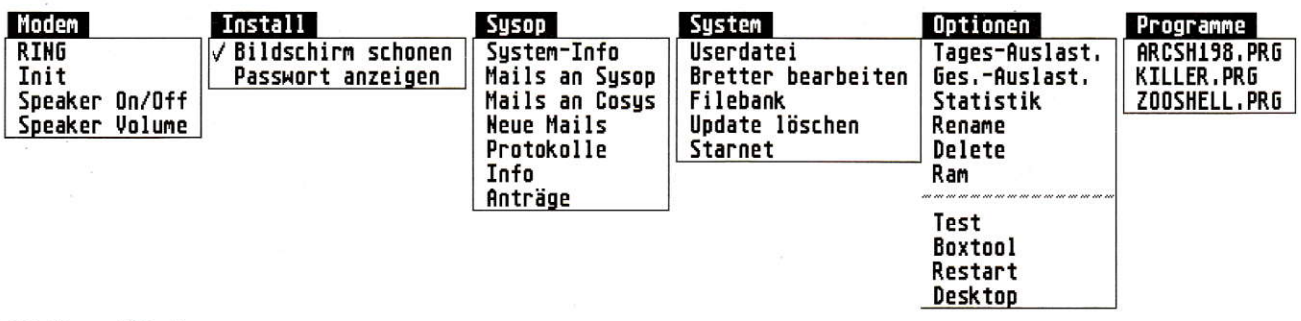


Bild 1: Die Starmail-Menüs

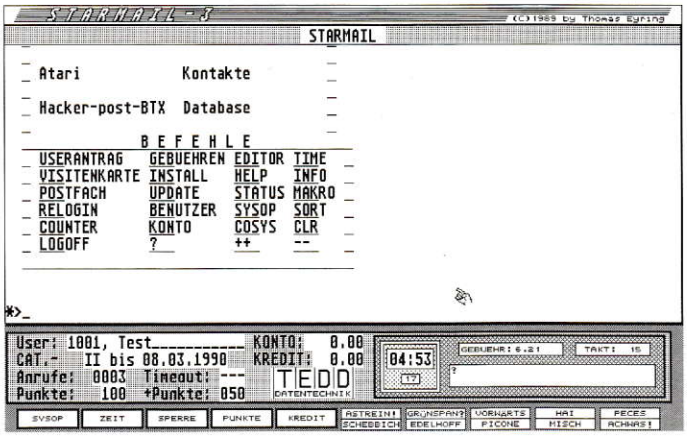


Bild 2: Starmail online

Namen gefunden hat, wird die User-Nummer bekanntgegeben, und man muß sich erneut mit seiner User-Nummer einloggen. Wie in anderen Programmen auch, hat der Sysop die Möglichkeit, Bretter nach seinen Wünschen einzurichten. Den Zugang kann man nach einigen Kriterien bestimmen: entweder einen Zugang für alle User (auch Gäste), einen Zugang für eingetragene User (Kategorie I und II), oder nur für Kategorie II-User.

Bretter

Möchte man ein Brett nur einigen Usern freigeben, so ist auch das möglich: Durch Zugriffsschlüssel kann jedes Brett für jeden User gesperrt werden - das allerdings nur bei bis zu 16 Brettern. Möchte man 17 Benutzergruppen eintragen, ist das nicht möglich. Im normalen Boxbetrieb sollte allerdings eine Unterscheidung nach 16 Benutzergruppen ausreichend sein. Ein Brett kann auch hier als ein Up-/Download-Brett eingetragen werden, in das nur Programme oder ähnliche Dateien eingespielt werden können. Dabei kann man per X- oder Y-Modem übertragen, bei Starmail wird noch an einer Z-Modem-Routine gearbeitet. Während ein User in der Mailbox ist, kann der Sysop natürlich auch diverse Funktionstasten drücken, woraufhin dann ein vorher definierter Text erscheint. Der gesamte Anruf des Users läuft in einem Window ab, das ungefähr 2 Drittel des Bildschirms einnimmt. Dadurch wird es leider unmöglich, eine vernünftige VT52-Steuerung aufzubauen: Inverse Buchstaben erscheinen hier unterstrichen, leider ist auch nicht der ganze Bildschirm zu überblicken. Auch das soll jedoch in einer der nächsten Versionen verbessert werden, so daß man zwischen dem Fenster- und einem ganzen Bildschirm hin- und herschalten kann: Das ist wohl die beste Lösung. Nun zur Bedienung: Möchte man in ein Brett gelangen, so ist einfach der Name des Bretts einzugeben. Hier treten unter Umständen beim Anlegen von Unterbrettern noch einige Probleme auf. Hat man beispielsweise ein Brett "Hardware" im Oberbegriff "Atari", geht man folgendermaßen vor: Zuerst gibt man "Ata" ein und befindet sich sodann im Atari-Bereich. Hier wird, vollkommen korrekt, die weitere Auswahlmöglichkeit angezeigt, z.B. "Hardware" und "Software". Nun möchte man das Brett "Hardware" anwählen. Die Eingabe von "Ha" unterscheidet sich eindeutig von "Software", also gibt man "Ha" ein - und landet im Brett "Hacker". Hier sollte eventuell etwas geändert werden. Leider kann auch beispielsweise kein Brett namens "Install" eingerichtet werden, da das System "Install" als Befehl erkennt. So sollten dann auch keine Bretter wie beispielsweise "Inserate" eingebaut werden, da bei Eingabe von "Ins" das System ebenfalls den Befehl "Install" erkennt. Grundsätzlich sollte also kein Brett eingerichtet werden, das in den ersten 3 Buchstaben einem Befehl entspricht. Wählt man ein Brett an, so wird auch sofort in die Leseroutine gesprungen. Das kann recht entnervend sein, wenn man nur schreiben möchte. Starmail hat auch einen Scanner eingebaut, der alle neuen Nachrichten seit dem letzten Login anzeigt. Diese überaus wichtige Funktion ist mit dem Befehl "INSTALL UPDATE ALL" aufrufbar (bei MagicBOX ST mit "+"), wenn man alle neuen Bretter angezeigt bekommen möchte. Mit dem Update ist es auch möglich, sich den Inhalt der Bretter anzeigen zu lassen.

Punktekonto

Der Kategorie I-User wird hier allerdings stark beschnitten, denn er kann nur den Inhalt aufrufen lassen. Möchte er die Nachrichten dann lesen, muß er noch per Hand das entsprechende Brett aufrufen. Beim Lesen und Schreiben von Mails ist eine Funktion eingebaut, die in bestimmten Fällen sehr nützlich sein kann: Ein sogenanntes "Punktekonto". Für jede Nachricht, die ein Benutzer liest, bekommt er einen Punkt abgezogen. Schreibt er eine Mail, so bekommt er wieder Punkte auf sein Konto addiert. Ist sein Punktekonto erloschen, kann er entweder nichts mehr lesen oder bekommt vor jeder gelesenen Nachricht eine Meldung ausgegeben, daß das Punktekonto nun erloschen ist, je nachdem, wie der Sysop es eingestellt hat. Da man auf diese Art und Weise natürlich seine User eher vergrault als einlädt, kann man diese Funktion auch abstellen. Eine wirklich nette Funktion ist die Möglichkeit, mit dem User zu sprechen, ohne den Sysop-Ruf zu betätigen. Dazu muß einfach vor der Eingabe einer Nachricht ein Semikolon eingegeben werden, um die (dann lästige) Meldung "Keinen Befehl erkannt" zu unterdrücken.

Fazit

Starmail ist ein sehr gutes Mailboxprogramm, das überaus absturzsicher und unschlagbar komfortabel ist - allerdings nur für den Betreiber des Systems. Ihm wird aller nur erdenkliche Komfort geboten, den keines der anderen Programme erreichen kann. Leider wurde aber bei den Funktionen für die User gespart. Wünschenswert wäre auch ein Point-Programm für das Starnet. Starmail mit Starnet kostet DM 398,- plus Porto und Verpackung.

MP/HE

Bezugsadresse:

TEDD-Datentechnik
Dieter Deppe & Thomas Eyring
Gladiolenweg 19
4792 Bad Lippspringe
Tel. (05252) 4525
Mailbox: (05252) 3413

STARMAIL MAILBOX

+ sehr gute Sysopoberfläche
+ Y-Modem-Protokoll eingebaut
+ Netzverbund
+ arbeitet mit Postmodems
+ Brettschutzmöglichkeiten
+ gute Geschwindigkeit von Diskette

- Z-Modem-Protokoll fehlt (noch)
- wenig Befehle für den Benutzer
- Netzverbund sternförmig: u.U teuer

Skylink
Das universelle System

Beim Mailbox-Programm Skylink ist man einen gänzlich anderen Weg gegangen - die Befehlsstruktur läßt sich vom Sysop komplett selbst bestimmen.

Nach dem Start von Skylink präsentiert sich dem Benutzer eine eher schlichte Oberfläche: Ein weißes Desktop. Doch die Einschaltmeldung kann nicht über die vielfältigen Möglichkeiten des Programms hinwegtäuschen. Eine Übersicht der Funktionen, die mit der Menüleiste ausgeführt werden können, sehen Sie in Bild 1. Gehen wir gleich zu den wichtigen Funktionen des Programms über, den Befehlen für den Benutzer.

Befehle

Skylink kann nicht ohne Probleme mit den anderen Mailbox-Programmen verglichen werden. Das rührt daher, daß alle Befehle, die man als User eingeben darf, selbst gestaltet werden können. Und selbst hier sind dem Sysop sehr gute Möglichkeiten gegeben, denn es können 4 unterschiedliche Befehlsstrukturen in das Programm eingebaut werden. So kann man beispielsweise eine Menü-, eine Bildschirmtext-ähnliche, eine Befehlsstruktur (ohne Parameter) und ähnliche Befehlslisten einbauen, zwischen denen der aktive User auswählen kann. Daß alle Befehlsstrukturen sich ähneln und im Grunde das gleiche mit unterschiedlichen Befehlen bewirken, fällt dabei nicht auf. Grundsätzlich stehen 26 Befehle zur Verfügung, denen man 4 oder mehr verschiedene Befehlsnamen zuweisen kann. Man sollte allerdings bedenken, daß 26 Befehle nicht gerade viel sind. Alle wichtigen Funktionen werden jedoch zufriedenstellend dadurch abgedeckt. Mit den 26 Befehlen läßt sich auf 4 Ebenen jeweils eine menü- bzw. befehlsgesteuerte Mailbox erstellen. Leider läßt sich auf diese Art und Weise keine Befehlssteuerung mit Argumenten bewerkstelligen (etwa "INHALT *" oder "BRETT ALL*"), obwohl sich Benutzeroberflächen wie Geonet in weiten Teilen des Bundesgebiets bereits als Standard durchgesetzt haben.

Bretter

Auch Skylink arbeitet, wie die beiden anderen Programme, mit Brettern bzw. Pinboards. Bei Skylink ist auch die Möglichkeit gegeben, sich mit einem Druck auf die Insert-Taste einzuloggen - bei den anderen Programmen muß entweder ein Button oder ein Menüeintrag angewählt werden. Vorteilhaft: beim Sysop-Login von der Konsole kann man die Baudrate einstellen, mit der man gerne die Box betrachten möchte. Wie bei den anderen Programmen auch, können hier "normale" Bretter zum Lesen und Schreiben eingerichtet werden, aber auch Up-/Download- sowie paßwortgeschützte Bretter. Dabei ist man nicht, wie bei Starmail, auf 16 Paßwörter beschränkt, sondern kann für jedes Brett ein eigenes Paßwort wählen. Da sich ein Paßwort unter Usern schnell herumspricht, hat man hier ähnliche Möglichkeiten wie bei MagicBOX ST, denn jedes Brett kann mit einer Zugangsberechtigung für jeden einzelnen User belegt werden, so daß das Wissen um das Paßwort allein nichts nützt, wenn man nicht in der Zugangsdatei eingetragen ist. Leider können Bretter aber grundsätzlich nicht vom Programm aus installiert, sondern müssen grundsätzlich mit einem externen Texteditor bearbeitet werden (s.u.). Dadurch ist es nicht möglich, die gesamte Mailbox per Fernwartung instandzuhalten.

Tastenbelegung

In Skylink hat der Sysop, wie auch bei den anderen Programmen, die Möglichkeit, mit Funktionstasten in das Geschehen während einer Verbindung einzugreifen. Dabei ist es allerdings das einzige Programm, bei dem man beispielsweise alle Eingaben, die über das Modem kommen, sperren, oder alle Befehle, die eingegeben werden, sofort über den Drucker ausgeben lassen kann. Sehr gut ist auch die Möglichkeit, sich den Zustand der Funktionstasten anzeigen zu lassen, indem man die HELP-Taste drückt.

Während bei den anderen Mailbox-Programmen komfortable Brett- und Usereditoren eingebaut sind, muß man hier etwas improvisieren: Bretter können nur über die Bretterdatei mit einem Texteditor geändert oder gelöscht werden. Weder im Box- noch im externen Programm existiert ein solches Programm. Möchte man im Wartestatus einen User eintragen oder edieren, muß man sich mit den Funktionen des Online-Usereditors zufriedengeben. Auch hier wurde kein komfortablerer Editor eingebaut.

Zusätzliche Funktionen

Interessante Möglichkeiten hat auch der Sysop: Durch eine Art von "Weiterleit"-Funktion können Nachrichten in eine temporäre Datei kopiert werden, von wo aus man sie weiterverarbeiten kann. Dabei wird der kopierte Text wieder unter dem Namen des Originalautors abgespeichert. Skylink bietet unter anderem aber auch sehr gute Statistikfunktionen für den Sysop: Alle Postleitzahlen, die beim User-Antrag durch neue Benutzer angegeben werden, werden mit dem entsprechenden Ort in einer Datei gespeichert. Außerdem hat das Programm, und das sollte eigentlich schon Standard sein, einen eingebauten Telefongebührenzähler, der anhand der Vorwahl des Benutzers die Tarifzone berechnet. Dabei werden vom Programm sogar Feiertage beachtet. Man muß sie zwar vorher in eine Datei schreiben, hat somit aber immer die

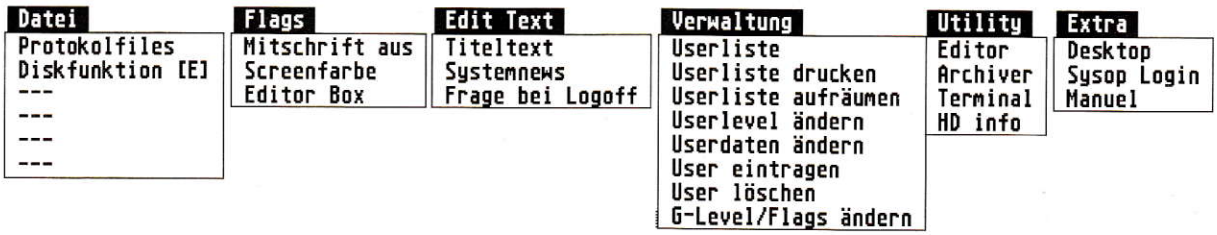


Bild 1: Die Skylink-Menüs

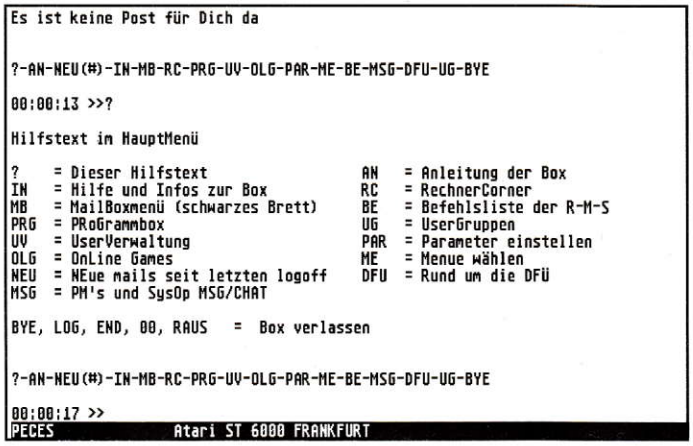


Bild 2: Skylink online

richtige Gebührenberechnung. Damit kann sich der User ein genaues Bild über seine nächste Telefonrechnung machen. Auch die Computertypen, die bei Neueinträgen angegeben werden, werden in einer extra dafür eingerichteten Datei abgespeichert. So ist man immer über die Verteilung der verschiedenen Computertypen auf dem Markt informiert.

Mehr Programme

Nicht nur das Mailboxprogramm wird geliefert, sondern auch eine komfortable Shell zur Behandlung von Disketten- und Festplattenoperationen. Diese Shell läßt sich aber leider nicht aufrufen, wenn man von außen anruft, da sie mit Menüleisten gesteuert wird. In der Shell lassen sich Dateien löschen, umbenennen, kopieren, Directories anlegen oder löschen, aber auch Programme in Up- und Download-Bretter kopieren. Um dies zu bewerkstelligen, muß man sich also nicht immer in die Mailbox einloggen, es reicht auch, wenn man die externe Shell aufruft. Auch ein Texteditor, "Edimax", wird mitgeliefert. Natürlich kann man aber auch seinen eigenen Texteditor wie Tempus einbinden. ARC, ZOO, LHARC und AR werden natürlich auch mitgeliefert, um einen einwandfreien Mailbox-Betrieb zu garantieren und seine Programme effektiv packen zu können.

Fazit

Mit Skylink erhält man ein menügesteuertes Mailbox-Programm, das man ganz nach seinen eigenen Wünschen gestalten kann. Es stehen zwar nicht viele Befehle zur Verfügung, sie sollten aber im normalen Mailbox-Betrieb zu einer recht komfortablen Steuerung ausreichen. Ein DFÜ-Neuling wird sicherlich Probleme haben, das Mailbox-Programm nach seinen Wünschen zu installieren, da die Befehlszuweisung für die einzelnen Menüs recht kompliziert ist. Doch es ist so wie bei anderen Programmen: Wenn man den Bogen einmal raus hat, klappt es gut. Leider ist das Programm auch noch nicht netzwerkfähig, dies soll in einer späteren Version jedoch folgen. Dafür ist es das Programm mit dem niedrigsten Preis, es kostet DM 198,-. Wer eine menügesteuerte Mailbox betreiben möchte, ist sicherlich mit Skylink recht gut bedient.

MP/HE

Bezugsadresse:

Thomas Rapp EDV-Beratung
Cutiusweg 21
2000 Hamburg 26
Tel. 040/218917
Mailbox: 040/217906

SKYLINK MAILBOX

+ sehr variable Menügestaltung
+ Help-Bildschirm für Sysop eingebaut
+ Eingaben lassen sich ausdrucken
+ temporäre Textdatei zum Kopieren
+ gute Statistikfunktionen
+ niedriger Preis
+ 4 verschiedene Menüs zur Auswahl

- kein Bretteditor
- (noch) nicht netzwerkfähig
- Menüaufbau für den Sysop kompliziert
- unkomfortabel von Sysopseite

Picobox

Aus der Schweiz kommt ein Mailbox-Programm namens Picobox, das am besten mit Skylink zu vergleichen ist: veränderbarer Menüaufbau, jedoch mit vielen komfortablen Befehlen.

Auch Picobox präsentiert sich nach dem Start eher schlicht. Lediglich eine Meldung auf die Möglichkeit zum Login oder zum "Shutdown" (Programmende) ist angegeben (Bild 1). Verwundert über nur 2 Möglichkeiten drückt man die Escape-Taste, um einen Login zu machen, und wird eines besseren belehrt: Hier kommen alle Möglichkeiten der Mailbox voll zur Geltung. Das System läßt sich in ausnahmslos allen Bereichen voll fernwarten. Deshalb sind auch keine aufwendigen Menüs zur Konsolenwartung nötig. Picobox arbeitet mit Hayes-kompatiblen Modems mit 300, 1200 oder 2400 Baud. In der Schweiz besteht bereits ein im Aufbau befindliches Netzwerk, genannt "Piconet". Das Piconet ist ab Mitte Oktober auch mit dem MagicNET vernetzt, so daß dadurch eine noch größere Auswahl an Nachrichten für den Benutzer erreicht wird. Picobox ist, wie alle anderen getesteten Mailbox-Programme, in GFA-BASIC geschrieben. Wie bei MagicBOX ST wird auch hier die Version 3.07 mit dem Compiler 3.02 benutzt. Dadurch wird das Programm recht schnell.

Befehle

Ähnlich wie bei Skylink lassen sich auch hier die Befehle schwer beschreiben, da man allen Kommandos einen eigenen Namen geben kann. Doch damit nicht genug: Es können auch eigene Menüs entworfen werden. Damit ist man nicht mehr auf die Voreinstellung angewiesen, sondern kann sich sein eigenes System

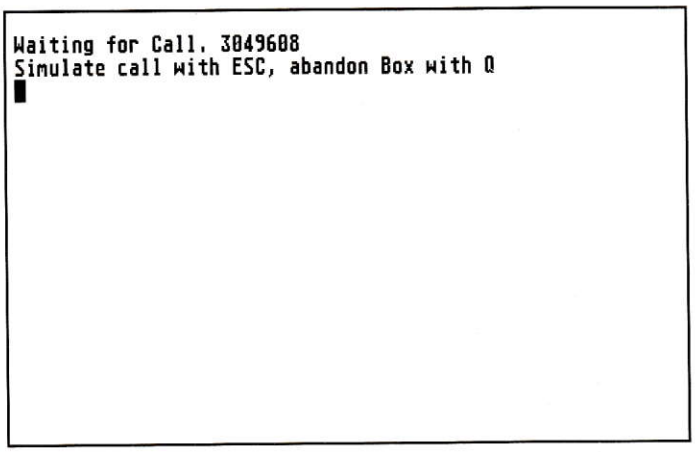


Bild 1: Die Picobox-Mailbox im Wartemodus

von oben bis unten so einrichten, wie man es gerne möchte. Dabei können mehr als 70 Befehle verwertet werden, die auch recht komfortabel sind. Doch nicht nur die Befehle sind sehr vielfältig, sondern auch die User der Box können nach 10 unterschiedlichen Kategorien in bis zu 9 Abstufungen eingeteilt werden. Level 0 ist für Gäste, Level 1-9 für "Newcomer", Level 10 bis 19 sollte für normale User reserviert sein. Ab User-Level 20 lassen sich dann immer mehr Privilegien einstellen, die ein "normaler" User nicht hat, beispielsweise der Zugriff auf die PD-Bretter oder der Löschzugriff in Brettern. Die User-Level haben bei Picobox jedoch so gut wie nichts zu bestimmen, da man jeden Befehl ab einem bestimmten Level freigeben kann. Doch damit immer noch nicht genug:

Terminalemulationen

Bei den Terminal-Emulationen sind Picobox keine Grenzen gesetzt! Alle können ebenfalls über eine Konfigurationsdatei eingestellt werden. Dadurch kann jede noch so exotische Terminalemulation perfekt emuliert werden. Auf diese Art und Weise kann man sich hervorragend seinen Usern anpassen, da nicht überall die gleichen Emulatoren benutzt werden. Diese Art der Steuerzeichenbeschreibung scheint mir zukunftsweisend. Die Installation der entsprechenden Steuerzeichen ist so einfach gemacht, daß man sie innerhalb von 2 Minuten bewerkstelligen kann.

Alle Aktionen des anrufenden Benutzers werden in 3 unterschiedlichen Protokollen gespeichert. Dabei besitzen sie jeweils eine unterschiedliche Ausführlichkeit. Im ersten Protokoll wird nur gespeichert, wer um welche Uhrzeit angerufen hat. Im zweiten Protokoll werden Zugriffe auf Bretter mit Datum und Uhrzeit gespeichert, im dritten, dem ausführlichsten Protokoll, wird alles mitgespeichert. Diese Vorgehensweise läßt sich leider nicht abstellen. Lassen Sie mich rechnen: 3 Anrufer erzeugen ungefähr 100 kB Protokolldatei. 20 Anrufer pro Tag erzeugen 650 kB Datei. In einer Woche (die Woche, in der der Sysop Urlaub macht) entspricht das knapp 7 Megabyte Protokolldatei... Hier sollte ein Schalter eingeführt werden, um diese Funktion auszuschalten. Man kann natürlich auch über eine Zeitschaltuhr eine Batchfile-Prozedur im Auto-Ordner aufrufen, die den Protokolltext täglich automatisch löscht.

Watchdog

Picobox ist bisher das einzige Programm, das einen Carrier-Watchdog mitliefert. Wird der Watchdog nicht mindestens alle 15 Minuten zurückgesetzt, oder findet er nach 15 Minuten keinen Carrier mehr an der Schnittstelle, führt er einen Reset aus und startet dadurch das Programm neu. Dadurch ist es unmöglich, daß das Programm für längere Zeit (s.o.: Urlaub) abstürzt, es sei denn, die VBL-Queue gerät derart durcheinander, daß der ST nicht mehr oben und unten erkennt. In Zusammenarbeit mit einer Zeitschaltuhr, die das System täglich einmal aus- und wieder einschaltet, ist eine absolut einwandfreie Funktion des Programms garantiert.

Wie in (fast) allen anderen Mailbox-Programmen auch, können bei Picobox Nachrichten in Bretter (hier heißen sie "Boards") oder in das persönliche Fach geschrieben werden. Dabei können die Bretter wieder unterteilt werden nach dem Zugriffslevel (von 0 bis 99). Durch die weiträumige Level-Verteilung ist es zwar nicht notwendig, ein Brett durch ein Paßwort zu schützen, die Möglichkeit ist aber trotzdem eingebaut.

Natürlich existiert auch die Möglichkeit, Programme in extra dafür angelegte Bretter zu veschicken. Leider stehen dem Programm dazu aber nur die Protokolle X-Modem und Kermit zur Verfügung. Da Kermit heutzutage kaum noch genutzt wird, ist der Einbau einer Z-Modem-Routine bereits in Planung. Hier haben sich einfach die Autoren von Picobox und MagicBOX ST zusammengeschlossen und tauschen ihre Routinen aus.

Auch die Möglichkeiten beim Lesen sind sehr umfangreich für ein menügesteuertes System (Bild 2): Es können bestimmte Dateien gelesen werden, alle Nachrichten ab und bis zu einer bestimmten Position oder alle Nachrichten von einer bestimmten Position bis zu einer anderen. Dabei kann jeweils noch unterschieden werden, ob die anzuzeigenden Nachrichten in absteigender oder aufsteigender Reihenfolge gezeigt werden sollen. Wenn eine Nachricht gelesen wird, wird automatisch alle 24 Zeilen ein "Weiter Taste" ausgegeben, um die Textausgabe zu stoppen. Diese Ausgabe kann durch Control-D

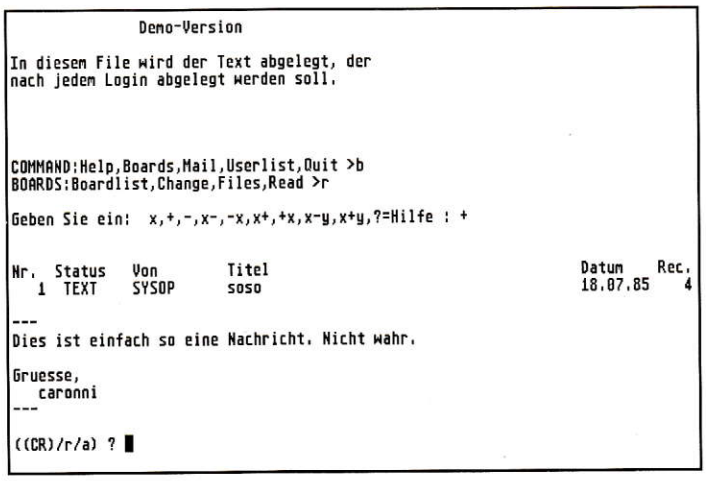


Bild 2: Auch online gibt's einiges zu sehen

unterbunden werden. Besser wäre eine Lösung, die andere Programme bieten, indem zum einen die Bildschirmlänge eingestellt, zum anderen die "Weiter Taste"-Ausgabe komplett unterbunden werden kann. So muß man nicht bei jedem Login wieder Control-D drücken.

Auch mit Picobox kann man an einem Netzverbund teilnehmen, dem Piconet. An diesem im Aufbau und seit zwei Monaten im Test befindlichen Netz sind zur Zeit 5 schweizerische Mailboxen angeschlossen, unter anderem auch die Firma DTZ, die in der Schweiz für etliche Programme und Hardware-Produkte den Vertrieb übernommen hat. Außerdem ist ab Mitte Oktober ein Gateway zum MagicNET in tätig, das alle Nachrichten zwischen den beiden Netzen austauscht. Die Verbindung zur Firma DTZ dürfte für viele Mailbox-Betreiber sicherlich ein guter Grund sein, sich dieses Programm anzuschaffen. Sollte sich dieses Mailbox-Netz auch in anderen Ländern als der Schweiz verbreiten, wäre das wirklich eine schöne Sache. Außerdem besteht in Deutschland die einfache Möglichkeit, durch das MagicNET eine Verbindung zu DTZ und dem Piconet zu schaffen.

Alsdann

Es erübrigt sich zu sagen, daß Picobox das beste menügesteuerte Programm ist. Das im Aufbau befindliche Netz läuft seit zwei Monaten fehlerfrei und dürfte somit die Serienreife erlangt haben. Die Möglichkeiten der eigenen Menü- und Befehlsgestaltung sind derart ausgereift, daß sich so manches Programm diverse Scheiben abschneiden könnte. Wenn Sie also ein menügesteuertes System einem befehlsorientierten vorziehen, sind Sie mit Picobox sehr gut beraten. Das Programm kostet SFr 400,- bzw. DM 400,-.

MP/HE

Bezugsadresse:

DTZ DataTrade AG
Langstrasse 94
Postfach 413
CH-8021 Zürich
Tel. (0041) (0) 1/2428088

PICOBOX MAILBOX

+ Netzwerk
+ Gateway zum MagicNET
+ alle Befehle selbst bestimmbar
+ X-Modem und Kermit
+ großer Befehlsumfang
+ komplett fernwartbar

- (noch) kein Y- und Z-Modem
- keine offline-Sysopfunktionen

MagicBOX ST
(mit MagicNET)

Die Überschrift zeigt bereits, daß man sich mit MagicBOX ST an eines der größten deutschen Mailboxnetze, das MagicNET, anschließen kann. Da das Programm recht viele Features hat, werde ich versuchen, Ihnen die wichtigsten in Kurzform darzustellen, da sonst der Rahmen dieses Artikels gesprengt würde.

Nach dem Start des Programms zeigt sich MagicBOX ST in einer ungewohnten Benutzeroberfläche. Die üblichen Menüleisten, fliegenden Rechtecke und hin- und herschwirrenden Dialogboxen sucht man hier vergeblich, bei der Oberfläche wurde ein komplett neuer Weg gegangen (Bild 1): Alle Optionen sind über Buttons am unteren Rand des Bildschirms erreichbar. Hinter einigen Buttons verbergen sich noch weitere, die anwählbar sind. So kann man leicht die Übersicht behalten. Der obere Teil des Bildschirms beinhaltet Informationen über das System wie die Tagesstatistik, Angaben über die Anzahl der Benutzer, Online-Zeiten, MagicNET-Zeiten u.ä. Erfreulich: Alle änderbaren Angaben lassen sich auch verändern, indem man mit der Maus einfach dort hinklickt, wo sie im oberen Bereich des Bildschirms stehen. Hier kann man nun beispielsweise einstellen, ob das System Public (Gäste dürfen sich frei eintragen), Non-Public (vor dem Eintrag muß der Systembetreiber konsultiert werden) oder offline sein soll, wieviele Laufwerke angeschlossen sind, Sysop- und Systemname können verändert werden etc. Doch der wichtigste Teil einer Mailbox überhaupt ist der Teil für die Benutzer, denn ohne sie kann kein System existieren. MagicBOX ST arbeitet mit allen Hayeskompatiblen Modems mit 300, 1200, 2400 und/oder 9600 Baud. Bei 9600 Baud wird eine effektive Baudrate von etwa 9000 Baud erreicht. Nun aber zur wichtigsten Sache, der Benutzeroberfläche für den Anrufer. Man drücke also mit dem Mausknopf auf "Login", und schon ist man in der Mailbox.

Von innen

Nach dem Login präsentiert sich dem Sysop eine Mailbox mit weit mehr als 80 Befehlen, die nach dem IMCA/Geonet-Standard gehalten sind. Für den Gast, der das erste Mal anruft, dürften sicherlich 80 Befehle eine Überforderung sondergleichen darstellen. Aus diesem Grunde kann für jeden Benutzer einzeln jeder Befehl zugeteilt und natürlich auch wieder gesperrt werden. So kann man dem Gast nur die wichtigsten Befehle geben, dem neueingetragenen Benutzer ein Drittel usw., bis er sich gut mit dem System auskennt. Auf diese Art und Weise ist sichergestellt, daß die reichhaltige Auswahl an Befehlen auch wirklich angewendet werden kann und keine Überforderung des Benutzers darstellt.

Auch an Terminal-Emulationen hat MagicBOX ST einiges zu bieten. VT52, ANSI/DEC VT100 und "VIP52" sind eingebaut. Zunächst zur VIP52-Emulation: Das ist eine abgemagerte VT52-Emulation für VipTerm-Benutzer auf dem C64. So können endlich auch diese DFÜ-Fans "ihre" Emulation voll ausnutzen. Die VT52-Emulation entspricht keinesfalls der eingebauten ATARI VT52-Version, sondern ist stark erweitert und an DEC VT100 angepaßt. Beim Login von der Console wird nicht nur auf inverse Zeichen erweitert, sondern es sind auch Fettschrift, Schrägschrift, unterstrichene, helle Schrift und natürlich weiterhin inverse Zeichen möglich. Beim Login von außen (also über die Telefonleitung) wird aus Geschwindigkeitsgründen auf die

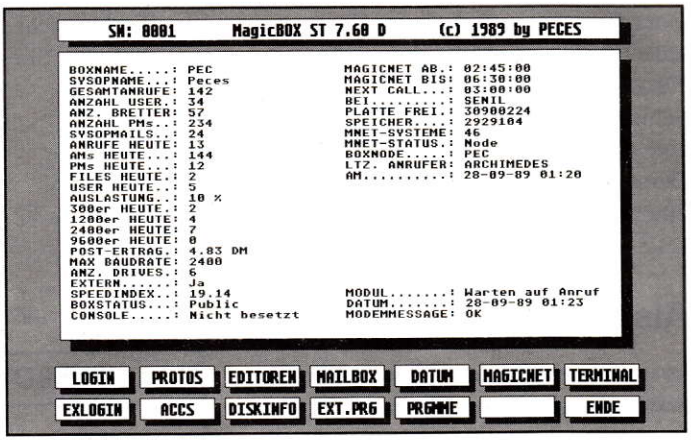


Bild 1: MagicBOX ST im Wartemodus

"normale" VT52-Steuerung umgeschaltet. Die eingebaute DEC VT100-Emulation wird für den Sysop umgewandelt in VT52-Steuersequenzen, damit er nicht mit Steuersequenzen überflutet wird. Interessante Effekte ergeben sich bei der Inhaltsübersicht, bei der die Zeilen 7 bis 23 zum Scroll-Bereich bestimmt werden, während alle anderen Zeilen auf dem Bildschirm stehenbleiben. Dies natürlich nur, wenn eine der drei Emulationen eingestellt ist. Da die erweiterte VT52-Steuerung natürlich Zeit in Anspruch nimmt (effektiv werden es etwa 4000 Baud), kann auch ein "Fast-Login" durchgeführt werden, das im günstigsten Fall ca. 60000 Baud (mit Blitter und Turbo ST) zustande bringt, dann allerdings nur mit "einfacher" VT52-Steuerung. Nun auf alle Befehle einzugehen, würde bis übermorgen dauern. Lassen Sie mich nur noch festhalten, daß die gesamte Mailbox auch per Fernwartung in Schuß zu halten ist: Ein Konsolen-Login ist dazu nicht notwendig.

Unterteilung

Das Mailboxsystem läßt sich grob in drei Bereiche einteilen: in den Lokalbereich, den Netbereich und die Gatemail. Lokale Bretter sind, wie der Name schon sagt, nur in der eigenen Box, also lokal, vorhanden, und werden nicht an andere Boxen des MagicNETs weitergeleitet. Alle Bretter, die im Netbereich sind, sind auch in allen anderen MagicNET-Mailboxen. Außerdem ist ab Mitte Oktober ein Gateway zum Piconet lauffähig. Dadurch ist gewährleistet, daß ungefähr 7000 Benutzer aus verschiedenen Ländern auf diese Bretter zugreifen können. In der Gatemail sind Bretter enthalten, die aus anderen Systemen stammen, etwa dem Fido- oder ZerebrusNet, und natürlich auch dem Piconet im normalen Netzteil. Hier wird also, und das ist einmalig bei den deutschen Mailboxnetzen, ein Übergang zu anderen Netzen geschaffen, damit man noch mehr Menschen erreichen kann. Es existiert noch eine weitere Art von Brettern: die Up-/Download-Bretter. In diesen können lediglich Programme abgelegt werden. Natürlich kann man auch gemischte Bretter anlegen, in denen Texte UND Programme untergebracht sind, diese Methode ist allerdings eleganter. Der Zugriff auf einzelne Bretter kann auf vielfältige Weise geschehen. So kann man zunächst festlegen, ab welchem Level ein Brett beschrieben, ab welchem es gelesen werden und ab welchem man sich den Inhalt anzeigen lassen kann. Damit nicht genug: Für jedes Brett kann ein eigenes Paßwort vergeben werden. Damit die Hacker gar keine Chance mehr haben, kann man jedem Benutzer einzeln den Zugriff für ein Brett geben oder nicht.

Lieferumfang

Nicht nur das Mailbox-Programm selbst wird geliefert, sondern auch einige andere nützliche externe Programme. Da sind zum einen die drei externen Editoren für User, Bretter und MagicNET-Systeme. Diese Programme sind in der gleichen Benutzeroberfläche gehalten wie das Mailbox-Programm selbst. Auch hier können alle Einstellungen durch einfaches Anklicken der entsprechenden Zeile verändert werden. Wichtige globale Funktionen sind wiederum in der unteren Button-Leiste zu finden. Ein Bild des Usereditors finden Sie in Bild 2.

Nützliche Utilities sollten bei keinem Mailbox-Programm fehlen. So werden auch einige Packprogramme mitgeliefert: ARC, ZOO und auch LHARC. Für Übertragungen wird auch eine externe ZModem-Routine mitgeliefert, die alle Netztransfers zu anderen MagicNET-Mailboxen schnell und sicher abwickelt. Desweiteren liegt ein kleines Terminal-Programm bei, ebenfalls mit der Benutzeroberfläche des Mailbox-Programms und der drei Editoren.

Übertragungsprotokolle

MagicBOX ST unterstützt die Übertragungsprotokolle XModem und ZModem. An einer Einbindung von YModem wird bereits gearbeitet. Die Übertragungsprotokolle können für reine Textübertragungen, aber auch zur Übertragung von Programmen in Up-/Download-Bretter genutzt werden. Nachdem ein Text in den eingebauten Editor geladen ist, kann man auch seine Befehlsvielfalt begutachten. Nützliche Funktionen wie das Errechnen der Verbindungsgebühr beim Trennen der Verbindung müssen wohl nicht mehr erwähnt werden, das sollte schon Standard sein.

Wenn Sie sich mit einem großen Mailboxnetz vernetzen wollen, sind Sie mit MagicBOX ST gut bedient. Ein großer Vorteil ist, daß über kurz oder lang auch mindestens fünf Mailbox in der Schweiz folgen werden, die über das Piconet Zugang zum MagicNET bekommen. Dadurch ist nicht zuletzt eine Verbindung zur Firma DTZ geschaffen worden. Einen Nachteil sollte man allerdings nicht unterschlagen: Die Befehlsvielfalt des Programms ist für DFÜ-Anfänger unüber-

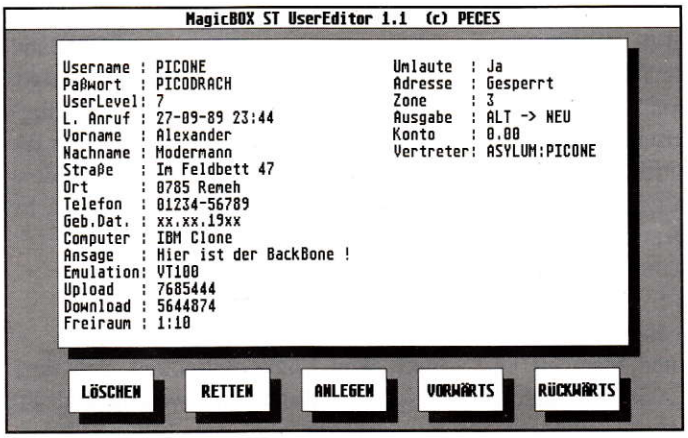


Bild 2: Der Usereditor von MagicBox ST

schaubar. Aus diesem Grunde sollten dem neuen User nur die wichtigsten Befehle gegeben werden. Wenn dem User alle Befehle bekannt sind, ist MagicBOX ST das Programm, bei dem der anrufende Benutzer die meisten Möglichkeiten besitzt. Etwas ungeschickt gelöst ist auch die "Zensur"-Möglichkeit im Netzbereich, bei der der Sysop jeden Tag alle "Zensurnachrichten" durchlesen muß. Möchten Sie Ihre Mailbox nur "nebenbei" betreiben und nicht länger als 30 Minuten täglich mit der Systemwartung verbringen, entscheiden Sie sich lieber für ein anderes Programm. Bedenken Sie dabei aber, daß auch die User Ihrer Mailbox komfortabel arbeiten möchten. MagicBOX ST ist in zwei unterschiedlichen Versionen erhältlich: Die Vollversion, die oben beschrieben wurde, kostet DM 349,-. Alle wichtigen Updates sind kostenlos. Hiermit stehen alle Möglichkeiten zur Verfügung. Die zweite Möglichkeit ist, eine "Point-Version" zu erwerben. Damit haben Sie die Möglichkeit, alle neuen Nachrichten des MagicNETs, wie bei einer vollwertigen Mailbox, automatisch zu sich nach Hause zu holen und auch zurückzusenden. Lediglich andere Benutzer können nicht bei Ihnen anrufen, ansonsten ist es das gleiche Programm. Die Point-Version kostet DM 29,-.

MP/HE

Alle Programmversionen sind exklusiv erhältlich bei:

BIELING Computersysteme
Spitzwegstraße 11
4350 Recklinghausen
Tel. (02361) 181485
Mailbox: (069) 683584

MAGICBOX ST MAILBOX

+ innovative Sysopoberfläche
+ komfortable Editoren
+ großer Netzverbund
+ 9600 Baud möglich
+ mannigfaltige Befehle
+ Z-Modem-Protokoll
+ Brettschutzmöglichkeiten
+ in verschiedenen Sprachen erhältlich

- Y-Modem-Protokoll fehlt (noch)
- sehr langsam von Diskette

Weide Elektronik GmbH
Regenstraße 34
D-4010 Hilden
Telefon 0211/341226

Ladenlokal: Gustav-Mahler-Straße im Einkaufszentrum
Telefon Ladenlokal 0211/03 18 80 Telefax 0211/03 1 82
viele Parkplätze direkt am Laden

ATARI System-fachhändler
Leasing für Komplettsysteme möglich
Atari + PC XT/AT

Schweiz
DTZ Data Trade AG
Langstrasse 94
CH-8021 Zürich
Tel. (01) 2428088
Fax (01) 2 91 05 07

Österreich
Reinhart Temmel GmbH
Markt 109
A-5440 Golling
Tel. (06244) 7081-0
Telefax 71883

SOFTWARE

WE SOFT
SINUS 79,-
Mathem. Software für Schule und Labor
UNISHELL 98,-
GEM-Shell
CRYPT 79,-
sehr schnelles Verschlüsselungsprogramm als .ACC-, TTP-, PRG
68881 + WCL 898,-
Ein starkes GESPANN
Mit WCL erhalten Sie mit 68881 auf ATARI ST/68000 Systemen die höchstmögliche Rechengeschwindigkeit. INFO anfordern!

Programmiersprachen
Schriftenerkennung a.A.
AUGUR 1.4 99,-
Omicron Assembler 149,-
GFA Assembler 179,-
Turbo C Assembler/Debugger 189,-
Omicron Basic Compiler 229,-
Omicron Basic 68881 Compiler
Omicron Basic Libraries (Midi, Statistik, Numerik, Complex, Easygem) ab 79,-
Maskeneditor, Link it, Draw je 79,-
GFA Basic interpreter 3.0 189,-
GFA Basic 68881 int. + Comp. 298,-
Aztek C prof. 398,-
Aztek C developer 598,-
Aztek C S. L. Debugger 148,-
Laser C 398,-
Lattice C 248,-
Mark Williams C 298,-
Prospero C 248,-
Turbo C 229,-
Turbo C + Assemb./Debugger 348,-
Prospero Fortran 378,-
SPC Modula 2 348,-
Melacomco Pascal 298,-
Pro PASCAL 248,-
OS-9 a.A.

Textverarbeitung
Tempus 2.0 129,-
1st Word plus 189,-
1st Extra 89,-
1st Proportional V 3.0 139,-
1st Adress 145,-
Daily Mail 179,-
Signum 398,-

Utilities
Crunch 98,-
(Opt. Harddiskzugriff, Backup)
UNISHELL 98,-
NEO Desk 89,-
Interlink ST – DFÜ d. Spitzenkl. 79,-
BTX-Manager 398,-
Turbo ST – der Softwareblitter 79,-
Revolver ST 79,-
– mehrere Programme auf Knopfdruck –
Desk Assist 4+ – Managerpr. 189,-
Hardisk Sentry a.A.
– täumt Ihre Platte auf –
Boot It – auch für 139,-
MATRIX Großbildschirm – 69,-
unterschiedl. Systeminstallationen a. Knopfdr.

Kaufm. Software
BS Handel 498,-
BS Fibu 548,-
BSS PLUS a.A.

CAD + Grafik
TechnoCAD System a.A.
CADJA 698,-

Plotter von A0-A3 preisgünstig
Arabesque 278,-
Mega Paint II 398,-
STAD 169,-
imagic 498,-
Creator 229,-

Tabellenkalkulation
LOGISTIX 398,-
LDW PowerCalc 248,-
BASICALC 59,-
Querdruck f. Tabellenkalk. 59,-

DTP
Calamus, neu, viel schneller 798,-
Outline Art 348,-
Fonteditor 198,-
Vektor Art Graf. Bibloth. je 198,-
CG Fonts a.A.
MegaFonts a.A.
Scan-Service a.A.
Scanner, versch. Typen
200-400 DPI a.A.

Emulatoren
PC DITTO V 3.96 198,-
PC-DITTO II a.A.
Aladin V.3.0 mit ROMs 598,-
ohne ROMs 398,-
(unterst. Harddisk + ATARI Laserdr.)
Spektre 128 a.A.

Bücher
ATARI ST Profibuch Sybex, 988 S. 69,-
Atari ST Hardware-Handbuch 69,-
Motorola 68000 59,-
Motorola 68030 98,-
Motorola 68881/82 98,-

SPEICHERKARTEN steckb.
auf 1 MB für 260/520 STM 348,-
auf 2 MB für 520 ST+/1040 a.A.
auf 2,5/4 MB a.A.
Jede Erweiterung einzeln im Rechner getestet! Sehr einfacher Einbau ohne Löten. Gut beschriebene Einbauanleitung. Verschiedene Modelle für Mega ST, 520 ST+, 1040 ST, Mega STE, 1040 STE, MMU-Sockels. Kein Bildschirmflimmern, keine zus. Software, keine zus. Stromversorgung!

Kompletter DTP-System bestehend aus Computer, Festplatte, Streamer, Monitor (oder Ganzseitenbildschirm) Software und evtl. Scanner a. A.

Komplette EDV-Auftragsabwicklung a. A.
Komplette EDV-Buchhaltung a. A. Vorführung nach Absprache! Sonderkonditionen für Wiederverkäufer.

HARDWARE

PROFESSIONAL SCANNER
CANON IX12F 3998,-
Technische Daten:
CCD Zeilensensor
Scannerelement: Flachbett mit innenlaufendem Schlitten
Scannertyp: Halogenlampe 50 V/42.5 W
Beleuchtungslampe: Bücher, Blätter, Objekte
Zulässige Originale: 216 x 297 mm (DIN A4)
Abtastfläche: 300 x 300 Punkte/Zoll (300 DPI)
Auflösung: 300/200/150/75 DPI
Scanoptionen: 32 Graustufen, softwaremäßig erhöhbar
Grauwerte: binär/dithered
Dichteeinstufung: 4 Sekunden pro DIN A4-Seite
Moduswahl: 368 (B), 570 (T), 83 (H) (mm)
Scangeschwindigkeit: 114 (K)
Abmessungen: 100-115 V/200 - 240 V
Gewicht: IMG, Postscript, Degas
Spannungsversorgung: Scansoftware, Retuschieprogramm
Bildformate:
Software:

ECHTZEITUHR PLUS 129,-
Keine Software nötig bei Betrieb mit Blitter TOS. Jede Uhr im Rechner getestet und gestellt. Interner Einbau ohne Löten. Dadurch freier ROM Port. Hohe Genauigkeit. Schaltjahrerkennung. Datum, Uhrzeit

ICD HARDDISKS
herausgeführter SCSI Port zum Anschluß von Standard PC Peripherie, unterstützt bis zu 8 SCSI-Geräte. DMA Eingang, DMA Ausgang 100 % kompatibel zu ATARI Harddisks
– sehr leiser Lüfter
alle Harddisks laufen auch mit TURBO DOS
eingebaute ECHTZEIT-UHR
Treiber mit oder ohne Verity
extrem leistungsstarke Software

85 MB form. 2498,-
50 MB Harddisk Plus 2198,-
Weide Harddisks
50 MB m. ICD Hostadapter 1598,-
85 MB m. ICD Hostadapter 1898,-
größere Platten a.A.

STREAMER 2498,-
Übertragungsrate 6,5 MB/Minute!
Komfortable Software Subsystem aus 50 MB
50 MB Harddisk + 155 MB Streamer 3798,-

ST HOST ADAPTER
Anschluß von IBM komp. Harddisks an Ihren ATARI ST unterstützt bis zu 8 SCSI Geräte, DMA-Eingang, DMA-Ausgang, SCSI Port integr. ECHTZEIT-UHR
Softwareunterstützung für über 100 versch. Harddisks und Controller
TOP Software
ST HOST ADAPTER 348,-

FLOPPYLAUFWERKE
3,5"-Einzellaufwerk 289,-
5 1/4"-Einzellaufwerk 448,-

COPROCESSOR 68881 498,-
Für Mega ST inkl. Softwareibraries für Megamax C, Mark Williams C, DRI C, Lattice C, Profortran, 77 AC, Fortran, Modula 2 usw. Einfacher Einbau ohne Löten! INFO anfordern!

BLITTER Nachrüstsätze 89,-
TURBO-ZILOG 198,-
TURBO-DRIVER 348,-
JUNIOR PRÖMMER 189,-
EPROMKARTE 64 KB 19,80
Epromblank, 128 KB 39,-
Epromdisk 576 kB 139,-
Floppyswitchbox –3 Laufwerke am ST– 89,-
Floppyweiche 98,-
Ersatzkabel 198,-
Ersatzteile a. A.
Floppystecker, Floppybuchsen, Monitorstecker, Monitorbuchsen 8,90 29,90
MMU/GLUE/BLITTERSOCKEL

COMPUTER-PROJEKTIONS-PANEL
Für alle, die etwas präsentieren müssen!
Gestochen scharf, 8 Graustufen, hochmoderenes DST LCD, mit IR Fernbedienung!
Keine Spezialsoftware nötig.
Kompensiert für jeden Overhead Projektor.
Invertier-Funktion, Löschfunktion, auch an IBM PS/2. IBM PC/XT/AT mit CGA, EGA, VGA-Karte anschließbar!
Auflösung 640 x 480 Bildpunkte, nur 3 kg Gewicht! Preis 3798,-
Passender Metallkoffer 348,-

ADIMENS ST plus

Das "PLUS" macht den Unterschied

Gerne schmücken die Software-Häuser ihre neuen Produkte mit einem kleinen unscheinbaren Zeichen, dem "+". Andere schreiben das Wort sogar aus: "plus" (z.B. 1st_Wordplus, ST-PASCAL plus, 1st-Proportional plus usw.). Bei den alteingesessenen Software-Produkten zum ATARI ST jagt meist eine Version die andere und anhand der Nummer orientieren sich die potentiellen Kunden kaum noch. Dann müssen aussagekräftigere Bezeichnungen her, die dem Kunden sagen, hier ist jetzt soviel Neues drin (in dem Programm), daß sich ein Kauf unbedingt lohnt. Halten diese "PLUS", "SUPERPLUS" und "HYPERSUPERPLUS" wirklich, was sie versprechen? Die Firma ADI-Software aus Karlsruhe hat schon auf der Düsseldorfer ATARI-Messe '89 einen Einblick in die neuen Funktionen ihres Produktes "ADIMENS ST plus" (Version: 3.0) gegeben. Was bringt das "Plus" den Anwendern jetzt?

ADIMENS ST plus besteht aus jenen drei Einzelprogrammen, die es auch in der Version 2.3 schon gab:

- "INIT" als Vereinbarungsteil
- "EXEC" als Datenbankausführungsteil
- "REORG" zur internen Neuorganisation

Grundsteinlegung

Mit dem Vereinbarungsprogramm "INIT" wird die Struktur der späteren Datenbank festgelegt. Es war für mich persönlich immer sehr angenehm, mit den GEM-Icons zu hantieren und dabei zu erfahren, wie leicht es ist, den Aufbau einer Datenbank in kürzester Zeit abzuwickeln: Einfach ein Symbol aus dem Teilefenster entnehmen und auf das freie Definitionsfenster bewegen. In der gegenständlichen Welt wäre dies einem Karteikasten vergleichbar. Dann ein Doppelklick, und schon öffnet sich das, was später die Datenmaske wird (vergleichbar mit einer Karteikarte). Jetzt werden die Felder entsprechend ihres Typs (numerisch, alphabetisch oder Datum usw.) aus dem Teilefenster gegriffen, und alsbald ist eine Datenbank entstanden.

Drei wesentliche Neuerungen im Programmteil INIT gilt es zu erwähnen:

a) Verbunde anlegen. Verbunde sind Masken, die genauso definiert werden wie die "normalen" Karteikarten. Das neue Symbol mit der Doppelschublade zeigt diese Funktion an. In Verbundmasken lassen sich Daten aus verschiedenen Dateien (Karteikästen) zusammenfassen und zeitgleich bearbeiten (sogen. JOIN-Funktion). Des weiteren können dabei bestimmte Ausschnitte aus verschiedenen Dateien in einer einzigen Maske zusammengelegt werden (sogen. VIEW-Funktion). Verbunde behandelt der spätere EXEC-Programmteil wie normale Masken (sie sehen auch genauso aus) mit den üblichen Operationen wie Eingeben, Ändern, Löschen, Anzeigen usw.

b) Eindeutige Schlüssel. Oft kann es nötig sein, nur eindeutige Schlüssel als Feldeinträge zuzulassen. Das bedeutet, daß in keinem weiteren Datensatz ein gleichlautender Eintrag in dasselbe Feld (d.h. mit demselben Feldnamen) möglich sein soll. Sinnvoll wird das beispielsweise bei der Vergabe von Kundennummern. Sobald eine Kundennummer zweimal vergeben ist, kann nicht mehr von einer eindeutigen Zuordnung ausgegangen werden. Deswegen muß ausgeschlossen sein, daß dieselbe Kundennummer nocheinmal vergeben werden kann. Die Zuordnung "eindeutiger Schlüssel" löst im EXEC-Teil eine automatische Überprüfung aus und weist nötigenfalls auf Doppeltvergabe hin. Weniger sinnvoll wird aber ein eindeutiger Schlüssel in folgendem Fall: Feld "Kundenname" und dortiger Feldinhalt "Müller". Hier muß es später erlaubt sein, weitere Kunden mit dem Namen Müller einzutragen - also kein "eindeutiges Schlüsselfeld"!

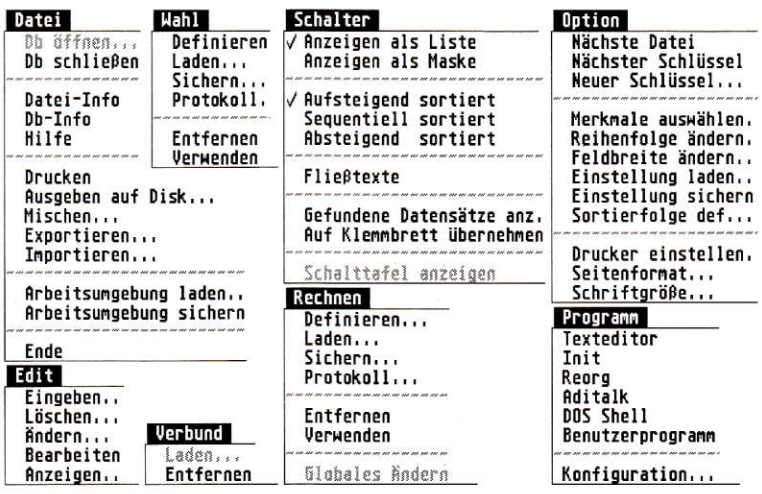


Bild 1: Die kompletten Pull-Down-Menüs des EXEC-Programms (Datenbankausführung)

c) Kommentare in Masken. Das war nun aber wirklich an der Zeit! Masken sollten möglichst wohlgeordnet, aussagekräftig und knapp bemessen sein. Langatmige Feldnamen möge der Datenbankdesigner bitte vermeiden. Was aber tun, wenn später ein(e) Nichtinformatiker/in im EXEC-Teil Dateneingaben machen soll und sich nicht auskennt? Da ist es doch ein Schritt hin zu mehr Bedienerfreundlichkeit, wenn an unklaren Positionen Hinweise, Kommentare oder Notizen stehen. Beispiel: Eine Auftragsnummer wird gerne benutzt, um die Interessenten und Käufe quasi durchzunumerieren. Da wäre es doch praktisch gleich dazuzuschreiben: "Auftragsnummer ist auch Rechnungsnummer!" Bediener oder Sekretärin wissen dann gleich Bescheid, längere

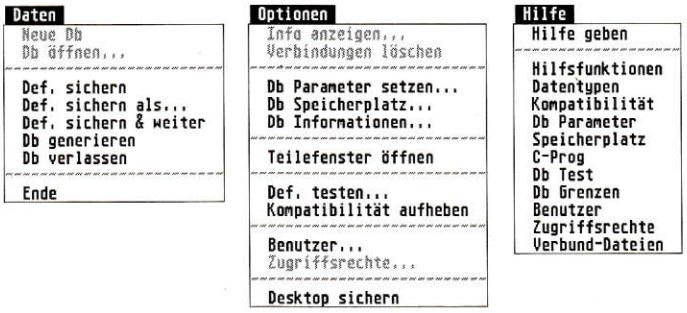


Bild 2: Die Pull-Down-Menüs des INIT-Programms (Datenbankeinrichtung)

schiedene Arbeitsoberflächen in beliebig vielen Dateien "*.INF" abgelegt werden. Jede "*.INF" kann ihre eigene "*.BAT" laden. Auch ein Wechsel der Anordnung der Oberfläche während des normalen Betriebes ist möglich.

fahren. Dies betrifft hauptsächlich die Menüpunkte "Datei", "Edit", "Option" und "Verbund". Die Zusammenstellung der Befehle wurde so umgestellt, daß sie übersichtlicher und komfortabler bedienbar sind. Befehle, welche auch im logischen Ablauf zusammengehören, stehen beieinander.

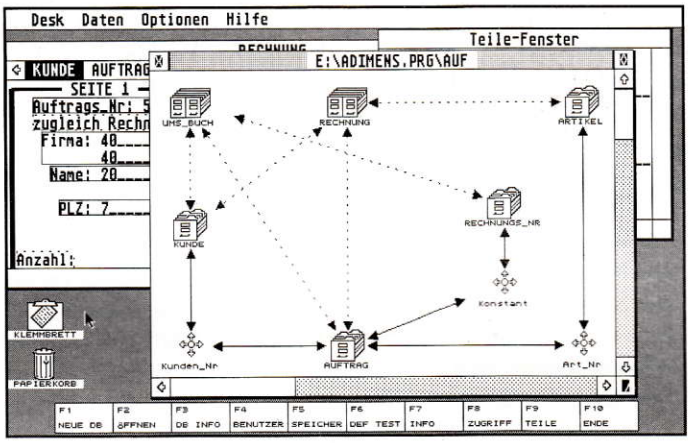


Bild 3: Im INIT-Programm sind die neuen Symbole "Doppelschublade" für Verbunddateien deutlich unterscheidbar. Gestrichelte Linien zeigen die Wege des Datenaustausches für Verbunde.

Rückfragen sind überflüssig. Mit Kommentarfeldern können nun Texte jeweils bis zu 21 Zeichen zusätzlich in die Maske eingebaut werden.

An die Arbeit

Wenn man sich die Arbeitsoberfläche des EXEC-Teils anschaut, werden einem gleich drei Änderungen bzw. Ergänzungen auffallen: 1. In der Pull-Down-Menüleiste hat es "Befehlszuwachs" gegeben (übrigens in den Pull-Downs selber noch viel mehr!), 2. eine "Schalttafel" ist hinzugekommen, und 3. ein Symbol "Sortierbrett" erscheint neu. Und das ist noch lange nicht alles. Viele Neuerungen wird man auf den ersten Blick nicht unmittelbar sehen können - aber der Reihe nach:

a) *Arbeitsumgebung*, soviel wie man will. Das normale ATARI-GEM speichert die Desktop-Ausgangslage der Symbole (Icons) und Fenster (Inhaltsverzeichnisse) üblicherweise in der Datei "DESKTOP.INF". Gleiches gab es auch für ADIMENS, aber auch nur einmal: die Datei "EXEC.INF". Jetzt können ver-

b) *RSC-Dateien*. Bislang wurden die Resourcedateien einmal ganz zu Anfang der Datenbankkonstruktion generiert. Jetzt erzeugt das neue ADIMENS ST plus alle benötigten Masken erst beim Öffnen der Datenbank. Das spart vor allem Speicherplatz auf dem externen Datenträger. Da dieses Verfahren völlig anders arbeitet, können RSC-Dateien aus den älteren Versionen nicht mehr benutzt werden.

c) *Neue Menüstruktur*. Die Menüleiste oben hat wesentliche Veränderungen er-

d) *Mehrfachsortierung*. Endlich! Jetzt besteht die Freiheit, die Daten einer Datei nach mehreren Schlüsselmerkmalen auszufiltern. In dem Menü "Optionen" wird eine Sortierfolge hierzu vereinbart. Wenn nun die betreffende Datei auf das Symbol "Sortierbrett" geführt wird, löst sie die Sortierfolge aus.

e) *Schalttafel*. Man muß nun nicht mehr rätseln, ob "Rechnen verwenden" oder "Wahl verwenden" aktiviert wurde. Auch ein ständiges Anwählen der entsprechenden Menüs ist nicht mehr nötig. Eine kleine Tafel zeigt an, welche Schalter zur angewählten Datei aktiv sind. Gerade hierdurch ist die Übersichtlichkeit wesentlich verbessert worden.

f) *Funktionstasten*. Natürlich gibt es immer Gegner der Maus. Man war es halt von älteren Rechnern gewohnt, ständig mit ALT- oder ESC-Sequenzen zu arbeiten. Bei ADIMENS ST plus werden parallel zu den GEM-Menüs die wichtigsten Funktionen dennoch per Funktionstasten erreichbar bleiben. Jetzt sind auch bei

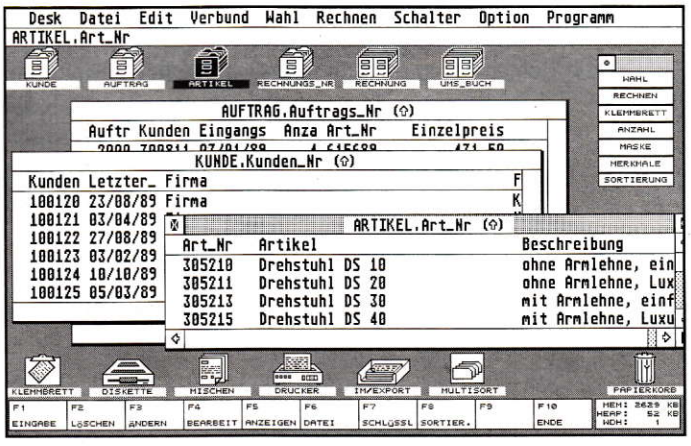


Bild 4: ADIMENS-Kenner können auf dem Desktop des EXEC-Programms einige neue Symbole entdecken: Die "Doppelschublade" für Verbunddateien, eine "Schalttafel" (rechts oben) zur Kontrolle der dateityp ischen Schalter und das Sortier(klemm)-brett "Multisort".

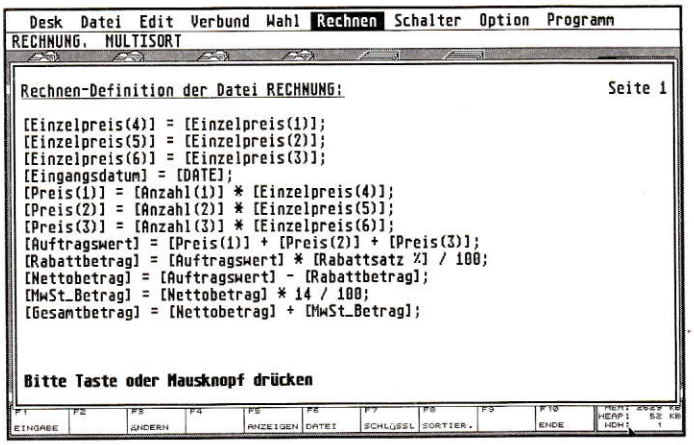


Bild 5: Auch bei den Rechenfunktionen hat sich einiges getan: Viele neue Rechenoperationen kamen hinzu.

geöffneten Dialogboxen die Funktionstasten ansteuerbar.

g) *Bearbeiten von Datensätzen.* Hinzugekommen ist der Menüpunkt "Bearbeiten", der es erlaubt, in der Maskenausgabe nach Datensätzen zu suchen. Anschließend kann für den angezeigten Datensatz sofort der nächste Arbeitsschritt direkt angewählt werden. Wenn die Listenausgabe voreingestellt ist, wird durch "Bearbeiten" noch schneller nach Datensätzen gesucht. Daß hierbei natürlich Jokerzeichen (Wildcards) wie "*" und "?" zugelassen sind, sei nur am Rande erwähnt.

h) *Wahl definieren.* Das mit den Mehrfachfeldern war schon immer schwierig zu verstehen. Die Einträge sind zwar unabhängig voneinander, dennoch aber in nur einem Feld. Diese Mehrfacheinträge mußten ganz verzwickt angesprochen werden. Jetzt braucht man nur noch einen Doppelklick, und ein Einzel(teil)feld ist spezifiziert.

i) *Rechnen definieren.* Gleiches zu den Mehrfachfeldern, wie eben bei Wahl definieren, gilt auch beim Rechnen. Zusätzlich können noch SYSDATE, SYSTIME und COUNT eingebunden werden. Gerade das Rechnen in EXEC hat viele Änderungen und Verbesserungen erfahren. So sind jetzt auch bedingte und logische Rechenanweisungen erlaubt wie: "IF-THEN-ELSE-ENDIF, größer als, kleiner als, gleich, AND, OR, NOT.

Beispiel:

```
IF >Bedingung<
THEN >Rechenprogramm<
```

Sogar Warnmeldungen (ALERT) kann man einbauen und sogar in eine IF-Anweisung.

Beispiel:

```
IF >Bedingung<
THEN ALERT >Zeichenkette<
     >Rechenprogramm<
```

Die Grenzen der EXEC-Rechenmaschine sind wesentlich erweitert worden:

- maximale Anzahl von Befehlen für eine Rechnung = 800 (bisher 200),
- der Stapel für numerische Konstanten und Variablen nimmt deren maximal 400 auf (bisher 100),
- der Stapel für Zeichenketten faßt nunmehr 4000 Zeichen (anstelle von 1000).

Die neue Variable SYSMODE

In SYSMODE wird eine Zahl als Information über jene Arbeitsumgebung festgehalten, von der aus das Rechnen gestartet wurde. Damit kann das Starten des Rechenvorgangs von der gerade aktiven Arbeitsumgebung abhängig gemacht werden.

Beispiel:

```
IF [SYSMODE] = 12
THEN [PLZ] = "D-" + [PLZ];
ENDIF;
```

Das heißt, nur wenn die Rechenfunktion aus der Arbeitsumgebung "globales Ändern" gestartet wurde, soll der Postleitzahl eine Zeichenkette "D-" vorangestellt werden. Die Zahl 12 für SYSMODE steht für "globales Ändern".

Der REORG-Teil bekam eine neue Oberfläche. Jetzt können über Dialogboxen die Datenbasis gewählt, die Kennwortangabe und verschiedene Einstellparameter eingegeben werden. UND: REORG ist erheblich schneller geworden!

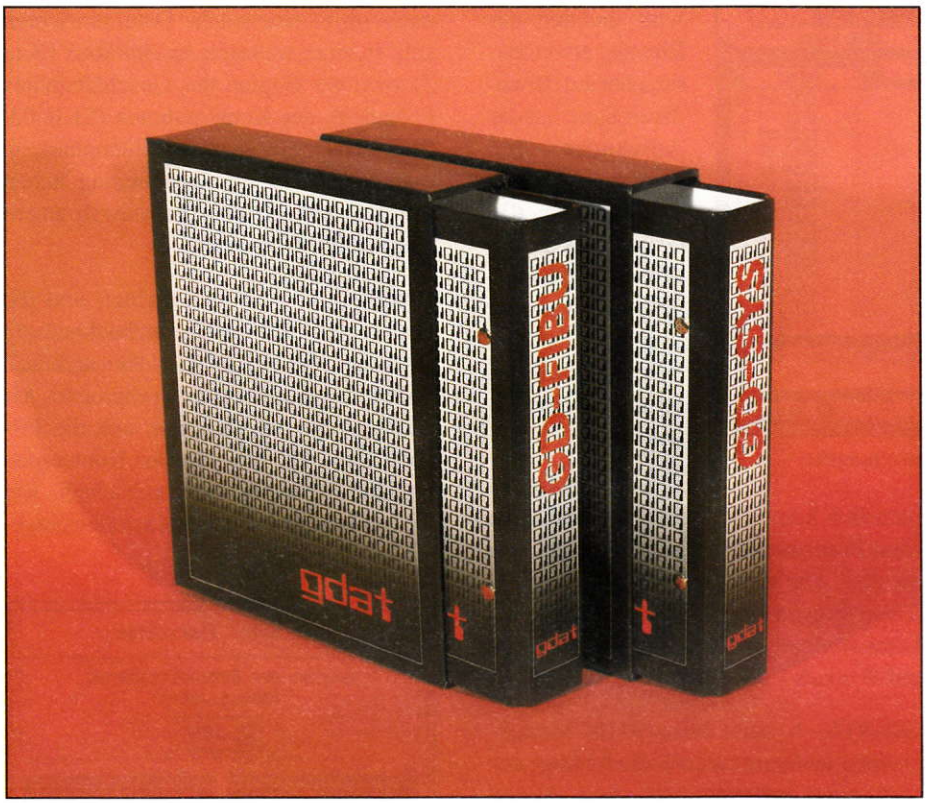
Darf's etwas mehr sein?

Das bisher schon als solide und ausgereift geltende Datenbanksystem ADIMENS ST hat durch die beschriebenen Veränderungen und Verbesserungen sicher an Interesse zugenommen. Bei einem Preis von DM 399,- bekommt der Anwender ein sehr brauchbares Datenbankprogramm. ADI-Software Karlsruhe zeigt durch diese Neuauflage, daß kleine Unzulänglichkeiten der früheren Versionen schnell aufgearbeitet und in sinnvolle Funktionen umgesetzt wurden. Als hervorhebenswert erscheinen mir die Verbundfunktion und das Mehrfachsortieren, sehr klug war auch das Einrichten der Schalttafel. Einen Schwachpunkt gibt es aber noch immer: Es sind einfach noch zu viele (auch oft benötigte) Funktionen in den Menüs versteckt, was mit der Maus länger dauert als mit Tastendrücken. UND: Wenn schon so oft die Maus beansprucht wird, warum ist die rechte Maustaste nicht für alternative Zwecke in Betrieb? Das aber sind nur kleine Wermutstropfen, die das Gesamtbild nicht trüben. UND: Welches Programm ist schon perfekt?

Dieter Kühner

Bezugsquelle:

ADI Software GmbH
Hardeckstraße 5
7500 Karlsruhe
Telefon: 0721/570000



GD-FIBU

Finanzbuchhaltung leicht gemacht?

Nach dem §238 des HGB ist jeder Kaufmann dazu verpflichtet, Bücher zu führen. Dabei muß die Buchhaltung so beschaffen sein, daß sich ein sachverständiger Dritter in angemessener Zeit einen Überblick über die Lage des Unternehmens verschaffen kann. Als Freiberufler, Klein- und Kleinstunternehmer muß man sich zumeist selber um die Buchhaltung und die Abrechnung gegenüber dem Finanzamt kümmern. Sicherlich kommt man um einen Steuerberater in der Regel nicht herum, aber mit ein paar Grundkenntnissen und dem richtigen Programm kann man sich schnell einen Überblick über die eigene wirtschaftliche Lage verschaffen. Besonders für diese Zielgruppe - kleine und mittlere Unternehmen - ist die Finanzbuchhaltung von gdat gedacht. Wird sie ihrem Anspruch auf leichte Bedien- und Erlernbarkeit gerecht, oder kommen auf den Anwender nur noch mehr durchwachte Nächte zwischen Hoffen und Bangen zu?

Die GD-FIBU ist bereits seit längerer Zeit im Einsatz. Die neue Version - die auf der ATARI-Messe vorgestellt wurde - ist völlig überarbeitet worden. Am Grundkonzept - möglichst viel Information übersichtlich am Bildschirm darzustellen - hat sich nichts geändert (Bild 1). Neu ist jetzt, daß man sich eine individuelle Finanzbuchhaltung aus mehreren Modulen zusammenstellen kann. Die Grundversion beginnt mit der einfachen Einnahme-Überschußrechnung. Mit den Modulen

- Bilanz
- Fremdwährung
  erleichtert die Abrechnung und verschaft den nötigen Überblick bei Fremdwährungen
- Offene Posten
- Kostenstellen
  nützlich, wenn der Betrieb in mehrere Abteilungen untergliedert ist
- Kontenplan-Verwaltung
  besonders interessant für Spezial-Kontenpläne im Mandantenbetrieb
- Abschluß-Formatierung
  kann man die Grundversion nach den eigenen Erfordernissen zu jedem beliebigen Zeitpunkt erweitern. Die Routinen für die Bildschirmausgabe sind alle in Assembler geschrieben, ein flüssiges Arbeiten ist dadurch gewährleistet.

Viele der neuen Features erkennt man gar nicht auf den ersten Blick, sie sind beim ATARI-ST einfach noch kein Standard. So zum Beispiel die Dateiauswahl mit assoziativer Namenssuche. Aber von diesen kleinen Details später mehr.

Was braucht man an Harter Ware?

Ein ATARI ST mit einem Megabyte Hauptspeicher reicht für den Anfang aus. Da die Daten von einem Abrechnungszeitraum alle im Speicher stehen, muß auch das TOS im ROM untergebracht sein. Als Massenspeicher genügt ein doppelseitiges Diskettenlaufwerk. Wie jede professionelle Software, so verwendet auch die GD-Fibu einen Monochrommonitor. Auf einem Großbildschirm läuft die Finanzbuchhaltung zwar, aber die volle Auflösung wird noch nicht unterstützt. Gdat arbeitet daran, daß durch ein Mehr an darstellbarer Information nicht die Übersichtlichkeit leidet. Ein Blitter wird empfohlen, aber es geht auch ohne so flott, daß man auf ihn verzichten könnte.

Dies ist wohl als Minimalkonfiguration zu verstehen. Für mich besteht eine komfortable Arbeitsumgebung aus einem MEGA-ST 2 oder MEGA-ST 4 und einer Festplatte, deren Größe sich nach dem Umfang der Buchhaltung und den sonstigen Anwendungen richtet. Auf eine doppelseitige Diskette passen ungefähr 10.000 Buchungen. Das ist bereits mehr, als viele Kleinunternehmen im ganzen Geschäftsjahr buchen. Sollte das nicht reichen, dann ist die GD-FIBU in der Lage, eine bis zu 12 MByte große Datei auf mehrere Disketten zu verteilen. Erst wenn die Datei noch umfangreicher wird, ist eine Festplatte nötig. Allein für die Finanzbuchhaltung und eventuell eine Textverarbeitung reicht deshalb eine kleine 20-30 MByte-Festplatte in den meisten Fällen aus.

Pro freiem MByte Hauptspeicher kann man 12.000 Buchungen in jedem Abrechnungszeitraum (monatlich, quartalsweise, jährlich oder wie Sie wollen) einge-

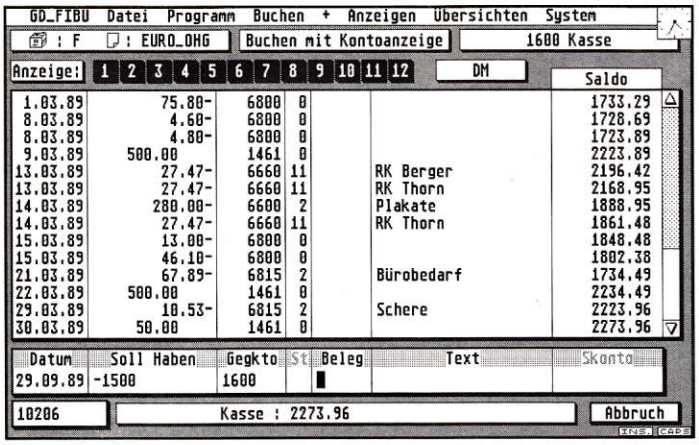


Bild 1: Viel Information übersichtlich dargestellt bei der GD-FIBU

ben. Bei 4 MByte RAM sind das über 40.000 pro Monat. Ich kenne keinen selbständigen Unternehmer, der mehr bräuchte.

Leider akzeptiert das Finanzamt noch keine Disketten mit der Abrechnung, und so ist auch ein Drucker nötig. Hier kann der Anwender aus einer breiten Palette wählen - zur Zeit stehen wohl an die 60 Druckertreiber zur Verfügung. Wer noch etwas mehr Geld los werden möchte, der besorgt sich extra Kontopapier, um dem Finanzamt eine Freude zu bereiten. Blanko tut's natürlich auch, es wird dann einfach das Formular mitgedruckt, was dann aber natürlich etwas länger dauert.

Starten und Installieren

Zum Starten der Finanzbuchhaltung wird ein ROM-Modul benötigt, welches man bei ausgeschaltetem Rechner einsteckt. In

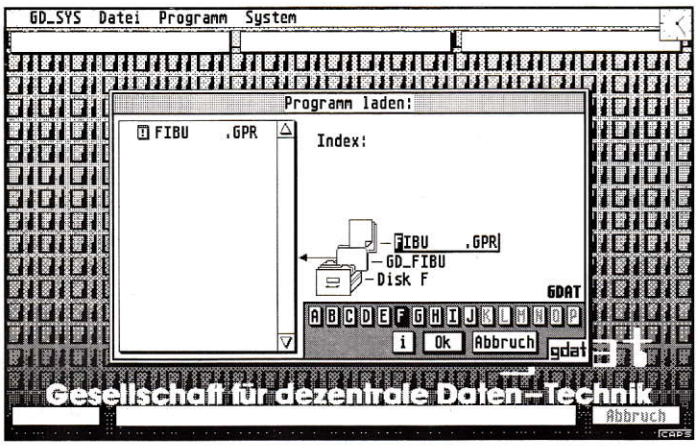


Bild 2: In GD-SYS wählen wir die Finanzbuchhaltung aus

diesem ROM-Modul sind die Grundroutinen, auf die alle Programme von gdat zugreifen. Zum Beispiel die Fileselectorbox, die kleine Analoguhr in der rechten obereren Ecke oder der kaufmänische Taschenrechner. Das ROM-Modul muß extra gekauft werden. Auf der einen Seite wird dadurch der Einstieg etwas teurer, aber auf der anderen Seite hat der Anwender den großen Vorteil, daß ihm die Grundfunktionen aller gdat-Programme vertraut sind und die Bedienung neuer Programme erleichtert wird.

Das Programm hat natürlich keinen Kopierschutz und kann ohne Problem auf der Festplatte installiert werden. Die einzelnen Pfadnamen und Einstellungen (Scroll-Geschwindigkeit, Dunkelschaltung,...) werden natürlich - auf Wunsch - dauerhaft übernommen.

GD-SYS - so heißt dieses ROM-Modul - ist unter anderem auch für die Abfrage des Paßwortes zuständig. Als erste Aktion muß dieses vom Benutzer eingegeben werden. Danach kann die Anwendung gestartet werden. Hier ist auch ein erster Ansatzpunkt für Verbesserungen. Es kann nur ein Paßwort vergeben werden, danach ist der Zugang zu allen Anwenderprogrammen frei. Sinnvoll wäre es, wenn man erstens für jedes Anwenderprogramm ein eigenes Paßwort definieren könnte, und zweitens, wenn es in der Anwendung selber mehrere abgegrenzte Bereiche gäbe. Dies ist zeckmäßig, wenn mehrere Personen mit dem Rechner arbeiten, aber nicht alle in der Bilanz der Firma stöbern sollen.

In GD-SYS wählen wir also als Anwenderprogramm die Finanzbuchhaltung (Bild 2) und nach einem mehr oder weniger langen Ladevorgang - abhängig davon, ob Sie mit einer Floppy oder einer Festplatte arbeiten - erscheint die Fileselectorbox, in der Sie Ihre gewünschte Datei anwählen können.

Ist noch keine eigene Datei vorhanden, gibt man einfach einen beliebigen Namen ein, das Programm erkennt das und fragt nach, ob es eine solche Datei einrichten soll. In einer weiteren Auswahlbox (Bild 3) wird der Beginn des Geschäftsjahres festgelegt, der Kontenrahmen (SKR 03, SKR04 oder ein Spezial-Kontenplan), ob man Soll- oder Ist-Versteuerung haben möchte und ob mit Fremdwährungen gearbeitet werden soll.

Die Datei ist nun eingerichtet, und die Bedienoberfläche der Finanzbuchhaltung erscheint wie in Bild 1, wobei natürlich noch keine Buchungen zu sehen sind. Als nächsten Schritt wird man nun die Firmendaten, den individuellen Kontenplan und die Steuerschlüssel einrichten. Bei Bedarf können auch Kostenstellen einge-

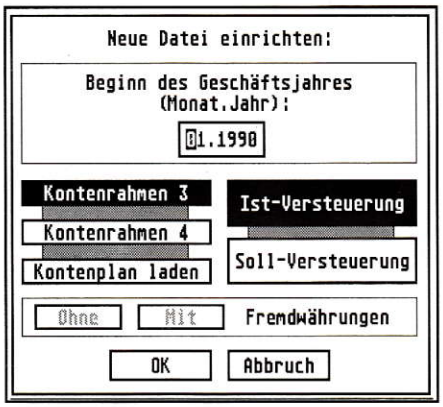


Bild 3: Neue Datei einrichten

geben werden. Alle diese Einstellungen finden sich unter dem Menüpunkt PROGRAMM und können der Reihe nach aufgerufen werden. Wird die aktuelle Datei zu einem späteren Zeitpunkt gesichert, werden diese Einstellungen selbstverständlich mit übernommen. Deshalb ist 'Mandantenfähigkeit' für die GD-FIBU kein Problem, jeder Mandant hat eine eigene Datei, in der alle Einstellungen gespeichert sind.

Die Firmendaten werden in einer Dialogbox - ähnlich wie beim Datei-einrichten in Bild 3 - eingegeben. Raffinierter ist da schon die Eingabe des Kontenplans. Nach dem Anwählen des entsprechenden Menüs erscheinen eine Dialogbox und ein Fenster, welches in beide Richtungen gescrollt werden kann (Bild 4). Man gibt nun einfach unter 'Nummer' die gewünschte Kontonummer ein und drückt <Return>. Sofort erscheint die Standardbezeichnung des gewählten Kontenrahmens (SKR 03 oder SKR 04) als Vorgabe, die man aber auch überschreiben kann. Bei 'St-Schl' hat man die Möglichkeit - sofern es für das gewählte Konto zulässig ist - einen der 98 möglichen Steuerschlüs-

sel zu wählen. Dabei müssen die Steuerschlüssel nicht unbedingt vorher unter dem Menüpunkt 'Steuerschlüssel' eigegeben worden sein. Man kann einen solchen auch beim Einrichten des Kontos definieren, wodurch lästiges Hin- und Herklicken eingespart wird.

Hat man das eine oder andere Konto vergessen und merkt dies erst später beim Buchen, ist das nicht weiter schlimm. Man gibt dann einfach die gewünschte, noch nicht existierende Kontonummer ein, und die GD-FIBU fragt nach, ob ein solches Konto eingerichtet werden soll. Solange ein Konto noch nicht bebucht worden ist, kann es auch wieder gelöscht werden.

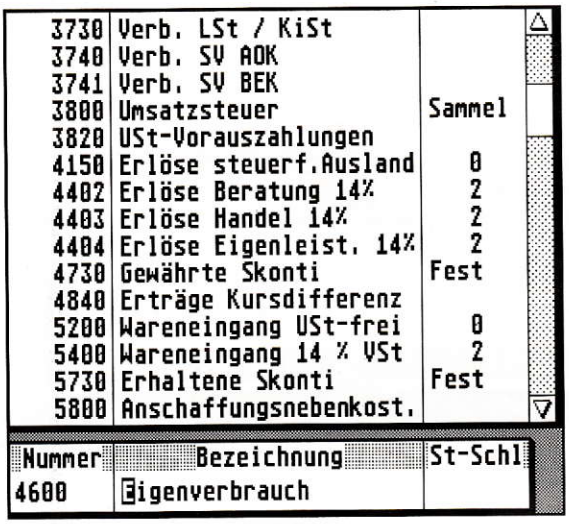


Bild 4: Der Kontenplan wird erstellt

Autobuchungen und eine Verwaltung der Abschreibungsgüter gibt es in der derzeitigen Version noch nicht. Ersteres wird es mit Sicherheit in der nächsten Version geben, erleichtert es doch das Buchen immer wiederkehrender Vorgänge (Miete, Bankgebühren,...). Bei der automatischen Verwaltung von Abschreibungsgütern dagegen ist man sich bei gdat noch nicht ganz einig. Vom Programmtechnischen gesehen gäbe es keine Schwierigkeiten, wurde mir versichert.

Der eigentliche Grund für das Fehlen liegt darin, daß bei kleinen Unternehmen - die Zielgruppe der GD-FIBU - das Jonglieren mit den Abschreibungen durch den Steuerberater ein mächtiges Werkzeug ist, um die Steuerlast zu mindern. Mit einer Automatik bestünde dann die Gefahr, daß man sich zuwenig Gedanken macht, in dem trügerischen Glauben, daß es die Finanzbuchhaltung schon richten wird. Dabei kann die Automatik nur nach einem Schema vorgehen und auf geänderte Rahmenbedingungen nicht reagieren. Es bleibt also abzuwarten, was sich gdat zu diesem Punkt einfallen läßt.

Buchen

Nach dieser Vorarbeit - die im übrigen leichter getan als beschrieben ist - kann man nun mit dem Buchen beginnen. Hier hat man drei Möglichkeiten, die sich jeweils durch die angezeigte Information unterscheiden.

- Buchen ohne Kontoanzeige
- Buchen mit Kontoanzeige
- Buchen mit Offener-Posten-Anzeige (sofern installiert)

Zunächst muß unter dem Menüpunkt 'Programm' das Erfassungsprogramm 'Buchungen' geladen werden. Sofort baut sich die Buchungsmaske auf, und das Programm befindet sich in der Option 'Buchen ohne Anzeige'.

An dieser Stelle ein kleiner Einschub: Der Programmspeicher wird vollkommen dynamisch verwaltet, das heißt, es wird nur soviel belegt, wie gebraucht wird. Die einzelnen Programmteile im Menü 'Programm' werden also solange wie möglich im Speicher gehalten und müssen nicht jedesmal neu nachgeladen werden - allerdings erst bei 2 MByte Hauptspeicher. Bei einer Festplatte fallen die Ladezeiten sowieso nicht groß ins Gewicht.

Bei 'Buchen ohne Anzeige' erscheinen die letzten 14 erfaßten Buchungen im Anzeigefeld, sofern überhaupt welche vorhanden sind. Diese Option ist vor allem dazu gedacht, die Buchungen schnell zu erfassen. In diesem Modus wird, anstelle des jeweiligen Saldos, das bebuchte Konto in der rechten Spalte des Rollfeldes angezeigt, um bereits abgelegte Buchungssätze auf ihre Richtigkeit hin überprüfen zu können. Um den aktuellen Saldo zu kontrollieren, wechselt man in das 'Anzeige'-Menü und läßt sich das Konto ausgeben.

Möchte man während des Buchens die volle Kontrolle über das bebuchte Konto, wählt man 'Buchen mit Kontoanzeige'. Hier wird nach jeder Buchung das bebuchte Konto, einschließlich der neuen Buchung, angezeigt (Bild 1). Diesmal erscheint in der rechten Spalte des Rollfeldes der aktuelle Saldo nach jeder Buchung.

'Buchen mit Offener-Posten-Anzeige' verwendet man, wenn man den sofortigen Überblick über Personenkonten benötigt. Es wird hier nicht der Mehrwertsteuerschlüssel angezeigt, sondern die offenen Posten. Ist der Saldo ausgeglichen, erscheint eine Fragebox, ob die entsprechenden Posten erledigt sind.

Der Buchungsvorgang

Die Art des Buchens ist gewählt, nun sollen die Daten folgen. Zunächst muß das Konto eingegeben werden. Das geschieht links unten, unter dem Feld für das Datum. Hier wartet der Cursor, als kleines Rechteck, auf die Eingabe (Bild 1). Jetzt hat man man mehrere Möglichkeiten:

1. Man gibt die gewünschte Kontonummer ein. Ist diese Nummer Bestandteil des eigenen Kontenplans, erscheint die dazugehörige Kontoanzeige rechts daneben. Ist das Konto jedoch nicht vorhanden, hat man die Möglichkeit, es einzurichten.

2. Man gibt einen Text ein. Wenn dieser Text Bestandteil einer Kontobezeichnung ist, erscheint wieder rechts daneben die gesamte Kontobezeichnung und im linken Feld die dazugehörige Kontonummer. Kommt der Text im Kontenplan mehrmals vor, wird der erste passende angezeigt. Mit den Cursor-Tasten 'aufwärts' und 'abwärts' kann man dann alle in Frage kommenden Konten anschauen. Sehr nützlich ist das bei den Kunden- und Lieferantenkonten, da man den Namen durch die Rechnung kennt, aber die Kontonummer meist nicht auswendig im Kopf hat.

3. Wenn einem gar keine Idee kommt, dann hilft vielleicht das '?' weiter. Nach der Eingabe erscheinen alle Sachkonten des eigenen Kontenplans. Das gewünschte Konto übernimmt man dann einfach mit Mausklick.

Bei '?K' werden alle Kundenkonten und bei '?L' alle Lieferantenkonten angezeigt. Die Auswahl erfolgt wieder mit der Maus.

4. Bei installierten offenen Posten kann man auch einen '*' und eine Belegnummer oder einen Belegtext eingeben. Jetzt durchsucht das Programm alle nicht komprimierten Buchungen. Ist die Suche erfolgreich, erscheint das Kunden- oder Lieferantenkonto im Kontofeld. Bei einer Zahlung muß man also nur die Rech-

nungsnummer wissen, das Konto wird vom Programm gesucht.

Als nächste Eingabe ist das Datum fällig. Es ist fast unnötig, darauf hinzuweisen, daß mehrere Schreibweisen als Eingabe zugelassen sind und die Eingabe auf Plausibilität hin geprüft wird.

Die Eingabe des Betrages erfolgt im Feld 'Soll Haben'. Ohne Vorzeichen wird der Buchungsbetrag im Soll (+) gebucht, mit einem Minuszeichen im Haben (-). Dieses Minuszeichen kann man auch durch zweimaliges Drücken der Return-Taste erzeugen. Bei installierten Fremdwährungen ist es möglich, hinter dem Betrag noch ein Kürzel für eine Fremdwährung anzugeben. Ist das eingegebene Kürzel noch nicht vorhanden, wird nachgefragt, ob eine neue Währung eingerichtet werden soll. Es erscheint eine Box, in der man den aktuellen Tageskurs dieser Währung eingeben kann. Zu einem Datum können verschiedene Kurse angegeben werden, etwa bei einem An- und Verkauf von Fremdwährungen an einem Tag. Nach der Eingabe des Kurses erscheint der umgerechnete DM-Betrag. Die Buchungen werden - je nach ausgewählter Option - in DM oder der Fremdwährung angezeigt. Dies ist besonders hilfreich, um zu erkennen, ob ein Konto mit Fremdwährungen ausgeglichen ist. Ohne diese Möglichkeit ist das nämlich gar nicht so einfach, da man Kursgewinne und Kursverluste berücksichtigen muß.

Im nächsten Eingabefeld muß das Gegenkonto eingegeben werden. Hier hat man dieselben Möglichkeiten wie bei der Eingabe des Kontos:

- eine Nummer
- einen Text
- einen Beleg
- ein Fragezeichen.

Mit der Auswahl von Konto und Gegenkonto ist bereits entschieden, ob Mehrwert- oder Vorsteuer zu buchen ist. Dementsprechend wird das nächste Feld angezeigt oder aber automatisch übersprungen. Ist die Eingabe eines Steuerschlüssels zulässig, bestehen wieder mehrere Möglichkeiten:

1. Automatische Errechnung der Mehrwertsteuer

Wie schon erwähnt, kann im Kontenplan-Editor einem Konto ein Steuerschlüssel zugewiesen werden. In diesem Fall wird das Feld übersprungen und die errechnete Steuer angezeigt. Mit den Cursortasten kommt man allerdings in das Feld zurück, um eventuell den vorgegebenen Steuerschlüssel zu ändern.

2. Buchung ohne Mehrwertsteuer

Ist keine Mehrwertsteuer zu buchen, gibt man einfach Return oder eine '0' ein, und der Cursor springt ins nächste Feld.

3. Buchen mit Steuerschlüssel

Eine Zahl zwischen 1 und 98 wird als Mehrwertsteuerschlüssel verstanden. Return beendet wieder die Eingabe, der errechnete Betrag steht neben der Kontonummer, und der Cursor springt in das nächste Feld.

Bei der Eingabe des Steuerschlüssels hat man noch ein paar weitere Möglichkeiten, die ich aber nicht alle aufzählen möchte. So signalisiert zum Beispiel ein 'i' inklusiv, der Buchungsbetrag wird nun als Bruttobetrag gewertet.

Im nächsten Eingabefeld kann eine Belegnummer oder eine Kombination aus Text und Zahlen eingegeben werden. Ist das Modul 'Offene Posten' installiert, hat man die zusätzliche Möglichkeit, die Buchhaltung nach allen zu dieser Nummer erfaßten Buchungen suchen zu lassen. Der ausstehende Betrag wird errechnet und angezeigt.

Im folgenden Textfeld kann man mit bis zu 20 Zeichen die Buchung kommentieren.

Bei installierten offenen Posten kann unter bestimmten Voraussetzungen automatisch ein Skonto gebucht werden. Den vom Programm gemachten Vorschlag kann man - wie immer - übernehmen oder verwerfen.

Buchungskomfort

Unter diesem Menüpunkt findet man zahlreiche Funktionen, die das schnelle Erfassen der Buchungen erleichtern (Bild 5). Da wären zum einen die Wiederholfunktionen. Ist das entsprechende Feld angeklickt (Konto, Datum, Betrag,...), braucht man bei der Eingabe einfach nur Return zu drücken, und es wird der Text der vorherigen Buchung übernommen. Im Falle der Belegnummer wird diese sogar um 1 hochgezählt, was besonders praktisch ist.

Unter der Rubrik 'Anzeige' wählt man aus, welche Informationen beim Buchen überhaupt zu sehen sind.

Korrekturen

Wie schon beschrieben, hat man bei der Eingabe der Buchung jederzeit die Möglichkeit, mit dem Cursor in das vorherige Feld zu gelangen, um etwas zu ändern. Was aber tun, wenn bereits gebucht ist, und es sich im Nachhinein herausstellt, daß korrigiert werden muß? Befindet man sich im Modus Kontoanzeige, Buchen mit Kontoanzeige oder Buchen mit OP-Anzeige, kann man ganz einfach mit Mausklick korrigieren. Der angeklickte Buchungssatz wird in die jeweiligen Eingabefelder übernommen und kann beliebig ediert werden. Mit dem letzten Return wird er wieder übernommen. Aber Achtung: Es wird automatisch eine Stornobuchung angelegt, um den gesetzlichen Richtlinien Genüge zu tun, wonach eine Aufzeichnung nicht dermaßen verändert werden darf, daß man den ursprünglichen Inhalt nicht mehr feststellen kann (HGB §239/3).

Allerdings sind korrigierte oder stornierte Buchungssätze nur in der Prima-Nota sichtbar. Auf den Kontoblättern, dem Journal, den Summen- und Saldenlisten sowie der betriebswirtschaftlichen Auswertung sind Stornobuchungen nicht zu sehen. Hier erscheinen die bereinigten Buchungen und Zahlenwerte, um deren Lesbarkeit zu erhöhen.

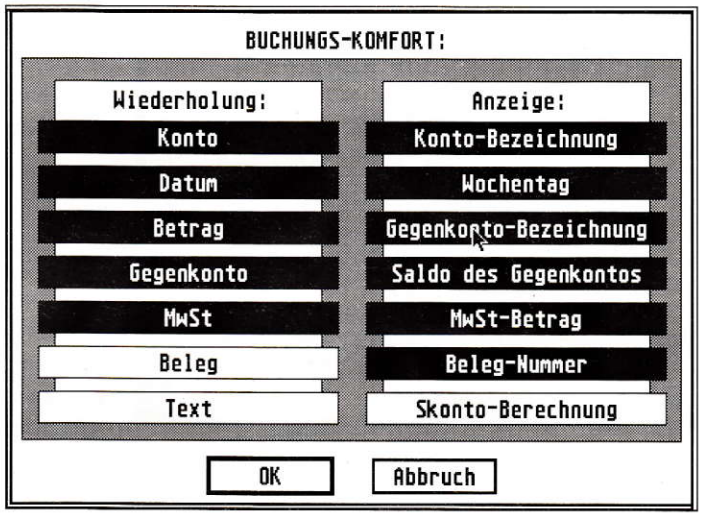


Bild 5: Einstellbarer Buchungskomfort

Keine Regel ohne Ausnahme: Wird eine Buchung am Tag ihrer Erfassung wieder korrigiert bzw. storniert, wird dieser Vorgang nicht in der Prima-Nota erfaßt. In diesem Fall kann man - wohl zu Recht - davon ausgehen, daß es sich um einen Tip- oder Flüchtigkeitsfehler handelt, der nicht zu einer schlechteren Lesbarkeit des Zahlenmaterials führen sollte.

Diese Vorgehensweise hat ihren Ursprung natürlich in der Praxis und ist durch die Steuerberater von gdat abgesegnet.

Sicherheit...

... hat bei der Buchhaltung einen sehr hohen Stellenwert. Die Aufbewahrungsfristen liegen zwischen 6 und 10 Jahren (HGB §257(4)). Nach §261 des HGB



Bild 6: Aussagekräftige Fehlermeldung

müssen in diesem Zeitraum Unterlagen, die auf Bild- oder Datenträgern vorliegen, auf eigene Kosten lesbar gemacht werden. Für den Anwender bedeutet dies die sorgfältige Auswahl der Datenträger, regelmäßige Sicherheitskopien und zum Schluß deren sorgfältige Aufbewahrung.

Vom Programm sollte man erwarten können, daß es einem bei dieser Aufgabe unterstützt. Die GD-FIBU läßt einen bei diesem wichtigen Punkt selbstverständlich nicht im Stich. Zu jedem Zeitpunkt hat man die Möglichkeit, eine Sicherungskopie anzulegen. Dabei ist man nicht auf die Kapazität einer Diskette beschränkt, sondern eine Datei kann bis zu einer maximalen Größe von 12 MByte auf mehrere Disketten verteilt werden. Sollte der Umfang im Einzelfall tatsächlich größer sein, empfielt sich sowieso ein leistungsfähigeres Backup-Medium wie Streamer oder Wechselplatte. Die eingegebenen Daten sind in dem Moment gesichert, wo mit 'Datei ablegen' der komplette Datensatz auf Massenspeicher geschrieben wird.

Sollte während des Abspeicherns ein Fehler auftreten, bleibt einem immer noch eine 'BAK'-Datei, und nur die frisch eingegebenen Daten sind verloren. Darüber hinaus kann man eine 'Datensicherungs-Automatik' aktivieren. Jetzt wird nach einer einstellbaren Anzahl von Buchungen automatisch die Datei gesichert. Diese Methode ist meiner Meinung nach besser als die Vorgabe eines Zeitintervalls, wenn der Rechner den ganzen Tag eingeschaltet bleibt. Entweder ist die Zeitspanne sehr lang, und im Falle eines Ausfalls die verlorenen Daten dementsprechend umfangreich. Oder aber das Intervall ist sehr kurz, und der Rechner ist ständig am Abspeichern.

Aber nicht nur im Umgang mit Dateien bietet die Finanzbuchhaltung von gdat wertvolle Hilfestellung. Viele Macken des Betriebssystems werden abgefangen: Ein eigener Critical-Error-Handler (also die Routine, die für das Abfangen von Bedienfehlern zuständig ist) sorgt dafür, daß in optisch ansprechender Weise (Bild 6) der Benutzer eine klare Aussage bekommt.

Auswertung

Unter dieser Rubrik möchte ich sowohl die Ausdrucke für das Finanzamt als auch die Ausdrucke für den eigenen Bedarf zusammenfassen. Bild 7 zeigt die Druckauswahl, aus der man beliebig auswählen

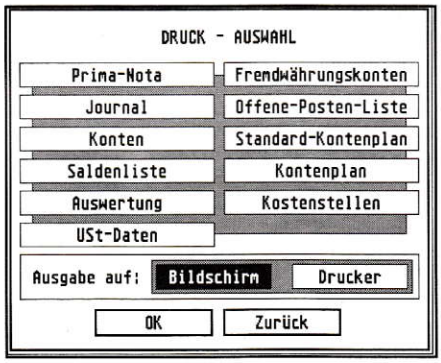


Bild 7: Alle wichtigen Daten im Zugriff

kann. Den Zeitraum, der ausgegeben werden soll, legt man einfach per Mausklick fest, desgleichen alle anderen einstellbaren Parameter. Möchte man Papier sparen, kann die Ausgabe auch auf den Bildschirm umgelenkt werden. Natürlich erscheinen die Daten dort genauso wie auf dem Drucker und können auf Wunsch auch noch seitenweise gedruckt werden. Extra aufgeführt unter 'Programm' sind die 'Abrechnung', die 'Einnahme-Überschuß-Rechnung' und die 'Bilanz'. Abrechnen bedeutet, daß die Prima-Nota, das Journal, die Saldenliste, die betriebswirtschaftliche Auswertung, alle Konten und die Werte für die Umsatzsteuervoranmeldung ausgedruckt werden. Wichtig dabei ist, daß abgerechnete Buchungen nicht mehr korrigiert werden können und die neuen Daten anstelle der alten abgelegt werden. 'Bilanz' macht das, was man erwartet, die Gliederung hält sich an die gesetzlichen Richtlinien des HGB. Gleiches gilt für die EÜ-Rechnung.

Die Druckqualität hängt natürlich in erster Linie vom eingesetzten Drucker ab. Für alle Ausdrucke kann Blanko-Papier verwendet werden, die entsprechenden Linien zur Unterteilung erzeugt das Programm. Für den Druck der einzelnen Konten kann man auf spezielles Kontopapier zurückgreifen. Das ist zwar teurer als einfaches weißes Papier, aber neben dem schöneren Aussehen (Bild 8) beschleunigt es auch den Ausdruck, da weniger Zeichen gedruckt werden müssen.

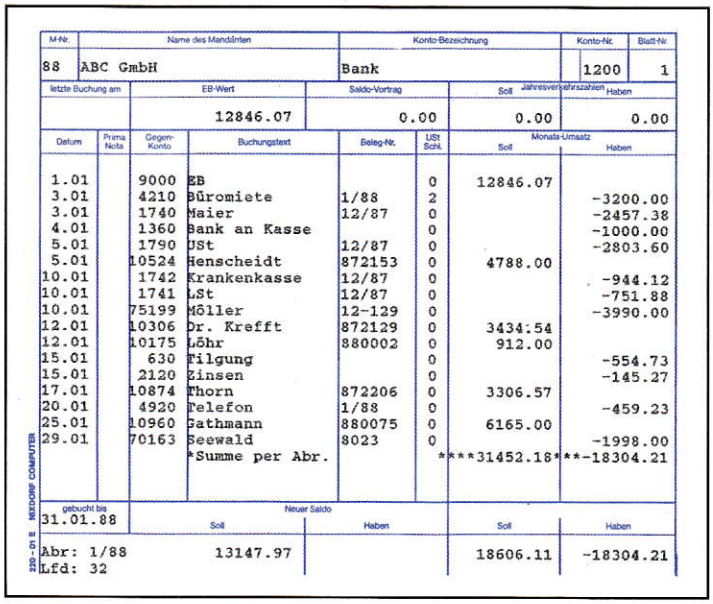
An dieser Stelle kann man die Druckausgabe noch perfektionieren. Von Haus aus bietet die GD-FIBU an die 60 verschiedene Druckertreiber, die man wahlweise auf die parallele Centronic und die serielle RS-232C- Schnittstelle loslassen kann. Allerdings ist nur ein Druckertreiber für eine der beiden Schnittstellen wählbar. Sehr angenehm wäre es, für beide Schnittstellen gleichzeitig je einen - auch unterschiedlichen - Treiber installieren zu können: den einen für die speziellen Kontoblätter (Matrixdrucker), den anderen für die 'schönen' Ausdrucke (Laserdrucker). Aber für die nächste Version der GD-FIBU muß ja auch noch etwas übrigbleiben. Man kann ja jetzt schon immerhin zwei Drucker anschließen und den jeweiligen Druckertreiber dann nachladen. Übrigens funktioniert der ATARI-Laserdrucker momentan mit der Epson-FX80-Emulation. Ein eigener Treiber ist aber bereits in Arbeit.

Mit den oben angeführten Auswertungsmöglichkeiten ist man in der Lage, jederzeit einen kompletten Überblick über die wirtschaftliche Leistungsfähigkeit seines Betriebes zu erhalten. Dabei ist man sehr flexibel und stößt auf keine Grenzen, die durch das Programm bedingt sind. Allenfalls durch den äußeren, gesetzlichen Rahmen ist nicht alles erlaubt, was programmtechnisch möglich wäre.

Service...

... wird bei gdat großgeschrieben. Man erhält alles, was im Zusammenhang mit der Buchhaltung nötig ist, angefangen bei den Kontoblättern von Bild 8 über die diversen Carbon-Farbbänder bis hin zur kompletten, quasi schlüsselfertigen Rechneranlage.

An Dienstleistungen hat gdat eine vierstündige Einführung, eine Einsatzbera-



M-Nr.	Name des Mandanten	Konto-Bezeichnung	Konto-Nr.	Blatt-Nr.
88	ABC GmbH	Bank	1200	1

letzte Buchung am	EB-Wert	Saldo-Vortrag	Jahresverkehrszahlen Soll	Haben
	12846.07	0.00	0.00	0.00

Datum	Prima Nota	Gegen-Konto	Buchungstext	Beleg-Nr.	USt Schl.	Monats-Umsatz Soll	Haben
1.01		9000	EB		0	12846.07	
3.01		4210	Büromiete	1/88	2		-3200.00
3.01		1740	Maier	12/87	0		-2457.38
4.01		1360	Bank an Kasse		0		-1000.00
5.01		1790	USt	12/87	0		-2803.60
5.01		10524	Henscheidt	872153	0	4788.00	
10.01		1742	Krankenkasse	12/87	0		-944.12
10.01		1741	LSt	12/87	0		-751.88
10.01		75199	Möller	12-129	0		-3990.00
12.01		10306	Dr. Krefft	872129	0	3434:54	
12.01		10175	Löhr	880002	0	912.00	
15.01		630	Tilgung		0		-554.73
15.01		2120	Zinsen		0		-145.27
17.01		10874	Thorn	872206	0	3306.57	
20.01		4920	Telefon	1/88	0		-459.23
25.01		10960	Gathmann	880075	0	6165.00	
29.01		70163	Seewald	8023	0		-1998.00
			*Summe per Abr.			***31452.18***	-18304.21

gebucht bis	Neuer Saldo Soll	Haben	Soll	Haben
31.01.88				
Abr: 1/88	13147.97		18606.11	-18304.21
Lfd: 32				

Bild 8: Mit Kontopapier ist es noch schöner.

tung (1.5 Stunden) und eine monatliche Software-Pflege im Programm. Die Einführung wird von Steuer- und Wirtschaftsberatern durchgeführt, die mit gdat zusammenarbeiten. Damit ist eine kompetente Beratung sichergestellt. Die Einsatzberatung findet in Bielefeld oder telefonisch ebenfalls durch Steuerberater statt.

Bliebe noch die Software-Pflege. Darunter ist zu verstehen, daß gdat für einen gewissen, monatlichen Betrag eine ständige Pflege der Software übernimmt. Hierunter fällt die telefonische Beratung von gdat beim Programmeinsatz, der schnelle Ersatz von beschädigten ROM-Modulen oder Disketten und, daß die Software automatisch immer auf dem neuesten Stand gehalten wird. Ausdrücklich betonen möchte ich, daß hierunter nicht der normale Update-Service und die akute Fehlerbeseitigung fallen. Bei neuen Programmversionen werden alle Anwender automatisch benachrichtigt, bei Fehlern erhalten ebenfalls alle eine korrigierte Version. Bei speziellen Kundenwünschen kann gdat auch eine individuelle Anpassung vornehmen. Aus einer solchen Anpassung ist zum Beispiel ein Programm entstanden, das es ermöglicht, Kostenstellen von verschiedenen Rechnern zu bebuchen. In der Hauptbuchhaltung wird dann alles zusammengefaßt. Nützlich ist das für räumlich getrennte Institutionen wie Vereine oder Behörden.

Die gdat-Bubbles

Hier wollte ich eigentlich die vielen Kleinigkeiten darstellen, die zeigen, mit wieviel Liebe zum Detail die GD-FIBU programmiert worden ist. Dazu gehört der kaufmännische Taschenrechner, den man jederzeit aufrufen kann, und der die errechneten Werte per Mausklick in das aktuelle Eingabefeld übernimmt. Auch der Kalender, die automatische Dunkelschaltung und die kleine Analoguhr im rechten oberen Eck gehören dazu. Stellvertretend für all diese gdat-Goodies möchte ich nur auf die Dateiauswahlbox eingehen (Bild 9).

Die Verschachtelungstiefe der Ordner wird grafisch übersichtlich angezeigt, die unterschiedlichen Dateien haben jeweils ein eigenes, grafisches Symbol (programmabhängig). Dank einer assoziativen Namenssuche werden auch fast nur richtig geschriebene Dateien gefunden. Vom Anwenderprogramm, welches die File-Selector-Box aufruft, kann auch die File-Kennung vorgegeben werden, also zum Beispiel *.BAK oder *.PRG. Andere File-Typen können dann gar nicht mehr geladen werden. Dadurch profitiert natürlich die GD-FIBU, die so gegen Fehlbedienungen gesichert ist.

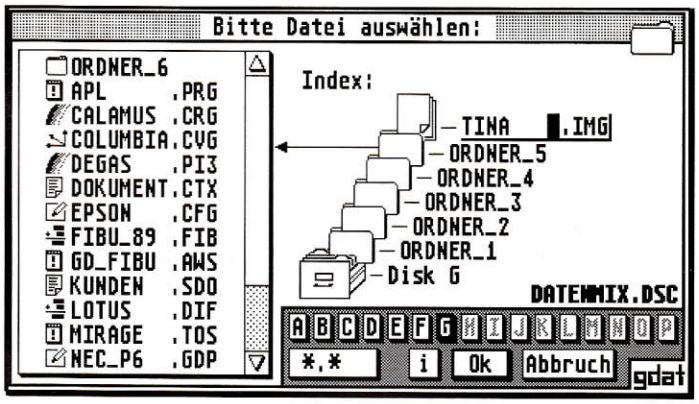


Bild 9: Die komfortabelste Dateiauswahl, die ich kenne.

Ausblicke?

Die GD-FIBU wird nicht alleine bleiben, eine Fakturierung ist bereits bei ausgewählten Firmen im Testeinsatz, eine Lohnbuchhaltung ist in der Entstehungsphase. Die einzelnen Programme werden natürlich die Daten soweit wie möglich austauschen können. An einer Netzwerklösung für die Finanzbuchhaltung wird ebenfalls gearbeitet. Eine DATEV-kompatible Abspeicherung der Daten ist programmtechnisch bereits realisiert und muß nur noch integriert werden. Gdat hat den Ergeiz, eine komplette Verwaltungs-Software auf die Beine zu stellen.

Und nun?

Die GD-FIBU besticht durch ihre Leistungsfähigkeit und die Benutzerfreundlichkeit. Die Einarbeitungszeit ist überaus kurz, alles ist gegen Fehlbedienung abgesichert. Die Handbücher kommen in einem stabilen Schuber daher, sind ordentlich aufgebaut und übersichtlich gegliedert. Hervorzuheben ist der Komplett-Service von gdat, der auch den Computer-Neuling nicht alleine läßt. Eine runde Sache also.

C.D. Ziegler

Preise

GD-SYS, Basissystem	*328.-*
GD-FIBU	*398.-*

Module:

Bilanz	*298.-*
Abschlußformatierung	*148.-*
Fremdwährung	*398.-*
Offene Posten	*148.-*
Kostenstellen	*198.-*
Kontenplanverwaltung	*148.-*

Demoversion *89.-*
Diskette mit allen Unterlagen wird beim nachfolgenden Kauf voll angerechnet

Demodiskette *15.-*



SCARABUS 2.0

Der Profi-Fonteditor für SIGNUM2! und SCRIPT

Zu Beginn dieses Testberichtes möchte ich der ST-Redaktion dafür danken, daß sie mir genügend Zeit zum Testen ließ. Entgegen der gängigen Praxis anderer Magazine, Vorversionen in Windeseile von hierzu kurzzeitig freigestellten Mitarbeitern testen zu lassen, durfte ich SCARABUS 2.0 zwei Monate lang auf Herz und Nieren prüfen. Anwendungsfeld des Testes waren die Verbesserungen und Erweiterungen meiner Fontdisketten Theolfont plus 2.0 und Arabicum 2.0 sowie der neuen Professional Fontdisk 3.0 [zu den Font(Zeichensatz)-Disketten Näheres am Schluß]. Erster Pluspunkt für SCARABUS 2.0 war seine Bombensicherheit. In 2 Monaten täglicher, oft mehr als 8 Stunden dauernder Arbeit ist es mir nicht gelungen, das Programm zum Absturz zu bringen.

Bei SCARABUS 2.0 handelt es sich um die neue Version des "erweiterten Fonteditors" für das Textverarbeitungsprogramm SIGNUM!, über das ich wegen seiner Beliebtheit auf dem ATARI ST keine Worte verlieren muß. Daß die SIGNUM!-Anwender sehr arbeitssam sind, was das Fonterstellen angeht, beweisen die inzwischen über 1000 SIGNUM!-Fonts. Das neue Textverarbeitungsprogramm SCRIPT derselben Softwarefirma Application Systems kann die SIGNUM!-Fonts ebenfalls verwenden (256! gleichzeitig), weshalb sich das Anwendungsgebiet von SCARABUS nun auch auf SCRIPT erstreckt.

Professioneller Editor

In dem Testbericht einer anderen ST-Zeitschrift hieß es zu SCARABUS 1.0, es handele sich nur um eine aufgepeppte Zusammenfassung der drei Original-SIGNUM!-Fonteditoren (im folgenden kurz: OFE). Entschuldigen Sie bitte die Schärfe! Zu solch einem Urteil kann m.E. nur ein Tester kommen, der ein wenig "getestet" hat, nicht jedoch professionell damit arbeiten mußte. Was verstehe ich nun unter professioneller Fonterstellung? Ein Profi-Fonteditor muß es mir erlauben, in angemessener Zeit kalligraphisch saubere, korrekt proportionierte Fonts zu erstellen.

Hierzu dient erstens die Möglichkeit, gescannte Vorlagen wie ein Alphabet als Bild einzulesen und buchstabenweise zu verarbeiten.

Zweitens benötigt man horizontale und vertikale Hilfslinien, die zwar nicht abgespeichert werden, aber die Orientierung erleichtern. Aus leidvoller Erfahrung mit dem OFE weiß ich, wie umständlich es vor zwei Jahren bei der Erstellung meiner Hebräischfonts zuging. Die hebräischen Vokale sitzen sublinear unter den Konsonanten. Ständig mußte abgezählt werden, damit alles stimmte (s. Bild 1).

Drittens wird der professionelle Anwender (z.B. Universitäten, Firmen) stets mehr als nur ein Druckerformat bearbeiten müssen: für 9- und 24-Nadler sowie Laserdrucker. Gegenüber den drei Programmen des OFE hat er es bei SCARABUS nur mit einem integrierten Programm zu tun. (Bei SCARABUS 2.0 ist übrigens der kleine Fehler der Version 1.0 behoben. Jetzt wird beim Umschalten von einem in das andere Format die Proportion gleich richtig angezeigt.) Unter diesem Punkt ist auch die Formatkonversion aufzuführen. Mit SCARABUS ist es möglich, aus jedem Format (P9/P24/L30) in eines der drei Formate zu konvertieren. Als Spezialanwendung gibt es die Konversion ohne Formatanpassung. Ich habe damit aus Times-11-italic-L30 (Laser) eine bessere Vorlage für Times-9-italic-P24 (24-Nadler) erzeugt, als wenn ich aus Times-9-italic L30 per Konversion mit Formatanpassung Times-9-italic-P24 erzeugt hätte.

Bearbeitungsmöglichkeiten

Viertens braucht der professionelle Anwender von einem bestimmten Font wie den TIMES-Fonts der Professional Fontdisk mehrere Größen (z.B. 13-Punkt für Überschriften, 11-Punkt für Haupt- und 9-Punkt für Fußnotentext) sowie Schriftarten (z.B. kursiv, fett, outlined, shado-

wed). Wie bei den meisten Computertextverarbeitungen kann man natürlich mit SIGNUM! und SCRIPT eine verwendete Schriftart mit den Optionen breit, fett, groß etc. kombinieren, doch das Ergebnis ist für ein geübtes Auge höchst unbefriedigend. Ein Ausweg wären hier Vektorfonts wie bei CALAMUS. Demgegenüber haben Sie aber mit SCARABUS (wie dem OFE) die Möglichkeit, die Fonts Punkt für Punkt ganz auf das jeweilige Druckermedium einzustellen. Ich selbst benutze dabei undokumentierte Tricks, die es mir erlauben, auch Feinheiten punktgenau zu zeichnen. Diese Feinheiten und Ebenmäßigkeit sind mit CALAMUS oder anderen Programmen, die auf Vektorfonts arbeiten, nicht möglich (sinnvolles Anwendungsgebiet für Vektorfonts sind sehr hohe Punktgrößen bzw. Lichtsatzmaschinen). Die Option kursiv können Sie bei SIGNUM! und SCRIPT zwar benutzen, doch wie auch sonst üblich, werden die vorhandenen Fonts hierbei nur geneigt. Abgesehen von den dabei auftretenden Rechenfehlern ist ein geneigter Font noch kein kursiver, der eben andere Formen hat (vgl. *a, e, f, ß*). Dieses Wissen müssen Sie schon selbst mitbringen. SCARABUS bietet aber die Möglichkeit (bei SCARABUS 2.0 endlich nicht nur für ein Zeichen, sondern gleich auf den ganzen Font anwendbar), Fonts gleichzeitig zu verbreitern und zu strekken (analog kürzer und schmäler), zu verdicken, helle, outlined (anstatt des üblichen Buchstabens nur die leichten Umrißlinien), schattierte (shadowed) und inverse Fonts zu erzeugen. Wer noch mehr Tatendrang hat, kann mit einem der 36 vorgegebenen Füllmuster seine Fonts bearbeiten (s. Bild 1).

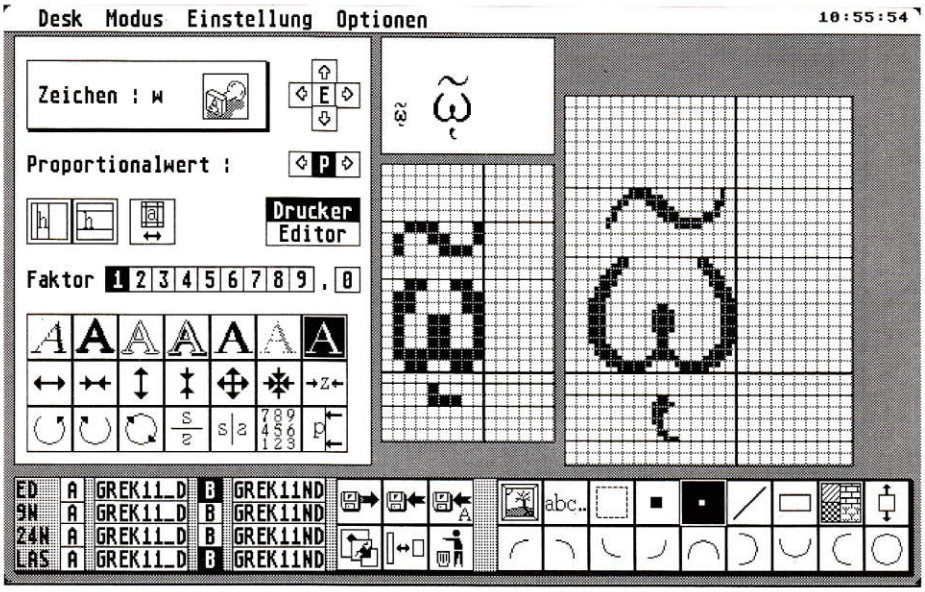


Bild 1: Hauptbedienungsoberfläche von SCARABUS

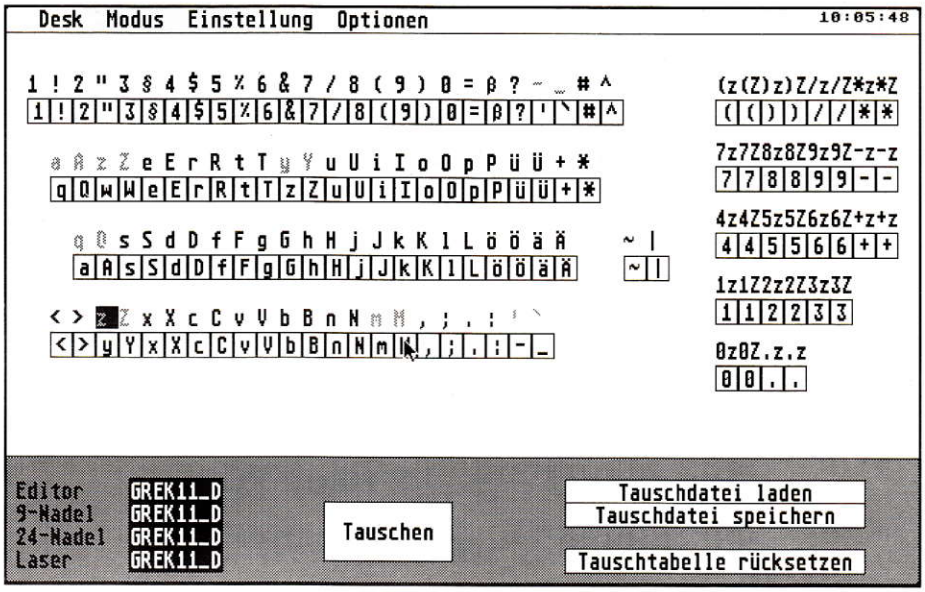


Bild 2: "Font tauschen" (helle Zeichen aus Font B)

Font tauschen

Fünftens ist es in den in Frage kommenden Anwenderkreisen manchmal nötig, individuelle Tastaturbelegungen für Spezialanwendungen zu erstellen. Das sind einerseits Publikationen. Für das Buch Diethelm Michel, Untersuchungen zur Eigenart des Buches Qohelet, Berlin-New York 1989 (BZAW 183) brauchte ich zwölf Fonts, die auf den sieben Positionen bei SIGNUM!2! unterzubringen waren. Andererseits müssen für die französischen, amerikanisch-englischen, dänisch-skandinavischen Versionen meiner Fontdisketten jeweils an die landesübliche ST-Tastatur angepaßte Belegungen erarbeitet werden. Für die 15 Fonts meiner *THEOLFONT plus* (in Wirklichkeit sind es 45, wenn man die drei Druckformate berücksichtigt) brauchte ich hierfür mit dem OFE mehrere Tage harter Arbeit - eingerechnet die vielen lauernden Fehlerquellen. Mit SCARABUS 2.0 geht dies unter dem Punkt "Font tauschen" in wenigen Stunden. Der eigentliche Konversionsvorgang mit SCARABUS dauert hierbei stets nur Sekunden. Die meiste Zeit vergeht mit der Vorarbeit, der Erstellung der neuen Tastaturbelegungspläne. Da stets alle drei Formate gleichzeitig getauscht werden können, fällt die Fehlerquelle weg, in einem Format einen Buchstaben vergessen zu haben (s. Bild 2 und 3). Alle genannten Optionen von SCARABUS 2.0 erleichtern dem professionellen wie dem passionierten, privaten Anwender die Fonterstellungsarbeit, die er so mit dem ansonsten guten SIGNUM!-OFE nicht bewerkstelligen kann.

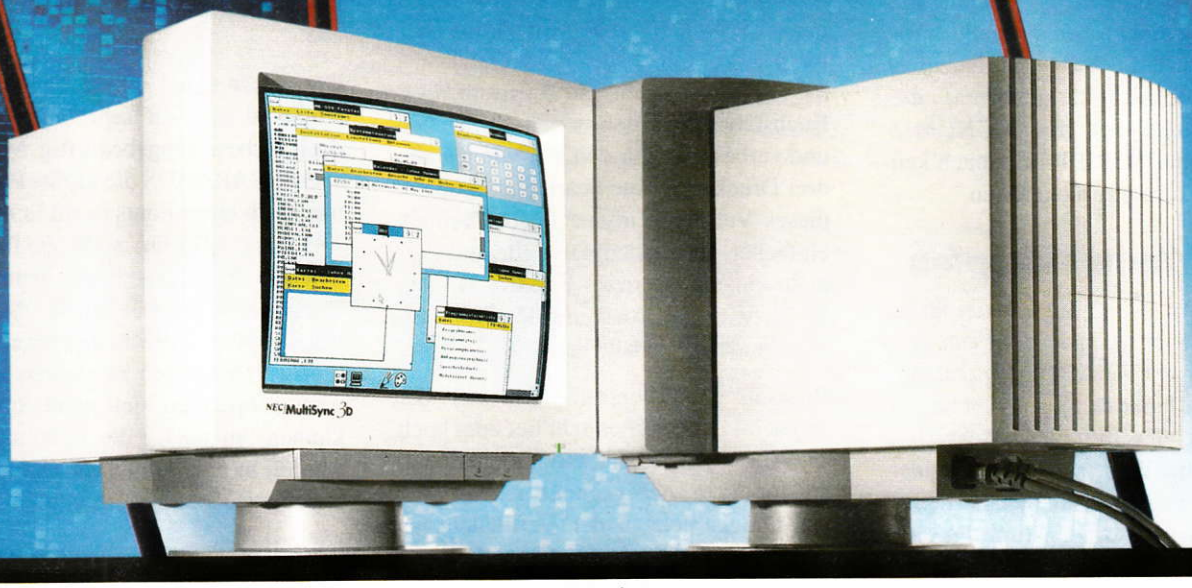
SCARABUS 2.0

Worin liegen nun die Unterschiede zwischen der Version 1.0 und 2.0? Die Version 2.0 bietet wie die Version 1.0 die Möglichkeit, das Programm alternativ über die Tastatur wie mit der Maus zu bedienen. Die Version 2.0 ordnet die mit der Maus auszuwählenden Optionen jedoch erheblich ergonomischer an, verzichtet auf unsinnige Tastaturbefehle und erleichtert das Pixeln.

Die alte Version glänzte beim Pixeln (Setzen und Löschen der einzelnen Punkte) gegenüber dem OFE mit einer unausstehlichen Lahmheit. Das Problem wurde dahingehend gelöst, daß nun nur noch die linke Maustaste aktiv ist. Gelöscht wird mit Shift-linke Maustaste. Obwohl ich seit 2 1/2 Jahren den OFE gewohnt bin

Gibt es einen Grund, sich erst für den Monitor und dann für den PC zu entscheiden?

Wir sagen ja.



Der NEC MultiSync 3D.

Das Allround-Talent.

Wenn Sie Monitore immer noch als notwendiges Anhängsel zum PC betrachten, dann sollten Sie jetzt schnell umdenken. Der MultiSync 3D von NEC wird Sie davon überzeugen, daß die Wahl des richtigen Monitors ebenso wichtig ist wie die Wahl des richtigen Computers.

Flexible Technik

Immerhin zeigt Ihr Monitor, was der PC Ihnen zu sagen hat. Da macht es schon einen Unterschied, ob Sie nur 16 oder 256 Farben darstellen können. Oder ob das Bild flimmert oder nicht.

Die Auflösung des NEC MultiSync 3D reicht bis zu 1024 x 768 Punkten (8514/A). Das heißt: Er ist kompatibel zu CGA, EGA, VGA. Natürlich auch zu SuperVGA, dem neuen internationalen Grafikstandard.

Besser arbeiten mit Digital Control.

Auch sonst hat der MultiSync 3D eine ganze Menge zu bieten. Einen entspiegelten, blendfreien 14 Zoll-Bildschirm. Natürliche Farben. Ein ergonomisches Design, das vom Haus Industrieform Essen ausgezeichnet wurde. Und das von NEC neuentwickelte Digital Control.

Dies ermöglicht dem MultiSync 3D, sich auf bis zu 15 unterschiedliche Grafikstandards einzustellen. Damit entfällt das mühsame Nachjustieren des Monitors.

Vergleichen Sie den **MultiSync 3D** ruhig mit anderen Monitoren.
Ihr NEC Fachhändler erwartet Sie.

Sag ja zu NEC.

Weitere Informationen erhalten Sie von:
NEC Deutschland GmbH · Klausenburger Straße 4 · 8000 München 80 · Telefon: 0 89/9 30 06-3 45
Fax: 0 89/93 77 76/8 · Telex: 5 218 073 und 5 218 074 necm d

NEC

(Setzen: linke Maustaste, Löschen: rechte), fällt mir das Umsteigen auf SCARABUS 2.0 relativ leicht. Bloß beim parallelen Anwenden von SCARABUS 2.0 und dem OFE innerhalb weniger Stunden (in außergewöhnlichen Fällen) ergeben sich leichte Umstellungsprobleme. Wie bereits oben erwähnt, sind jetzt einige Optionen auf den ganzen Font anwendbar, die früher nur für einzelne Zeichen aufrufbar waren.

Handbuch

Das Handbuch ist übersichtlich und klar geschrieben. Den alten Hasen sei angeraten, es auch dann durchzulesen, wenn man mit dem leicht zu bedienenden Programm auch so schon zurecht kam. Es stecken nämlich einige hilfreiche Hinweise drin, die man sich so nur über lange Umwege zufällig erschließt. In Verbindung mit der Shift-Taste wirken z.B. bestimmte Funktionen kontinuierlich. Jetzt sind Verschiebearbeiten schnell und leicht zu erledigen. Wer spezielle Informationen zur Technik der einzelnen Drucker, dem Speicherformat etc. wünscht, sollte auf das SIGNUM!2-Handbuch (das SCRIPT-Handbuch kenne ich noch nicht) zurückgreifen.

Kleine Wermutstropfen

Anfangs verwies ich auf einen anderen SCARABUS-Testbericht. Diesem entgingen die in SCARABUS 1.0 enthaltenen Fehler. Damit ist niemand geholfen. Meines Erachtens hilft ein ehrlicher Test dem Programmierer wie dem Anwender. Wegen der nachfolgend beschriebenen, noch vorhandenen Unstimmigkeiten habe ich mich bereits mit dem Programmierer in Verbindung gesetzt. Er war für die Hinweise sehr dankbar. Für ein Update konnte er mir noch keinen genauen Termin angeben, doch er versprach, sich sobald wie möglich an die Behebung der Kritikpunkte zu machen.

Bezüglich der zu bemängelnden Programmpunkte muß fairerweise bemerkt werden, daß Sie nur bei Spezialanwendungen ins Gewicht fallen, während sie bei gängigen Anwendungen gar nicht benutzt werden.

Der störendste Fehler tritt beim Odern unter "Font spezial" auf. Mittels dieses Menüpunktes sollte es u.a. möglich sein, Umschrift- (Transkriptions-) und Eurofonts zu erstellen. Einziges Problem hierbei ist nur, daß SCARABUS 2.0 die Proportionen wahllos einstellt. Mal nimmt er

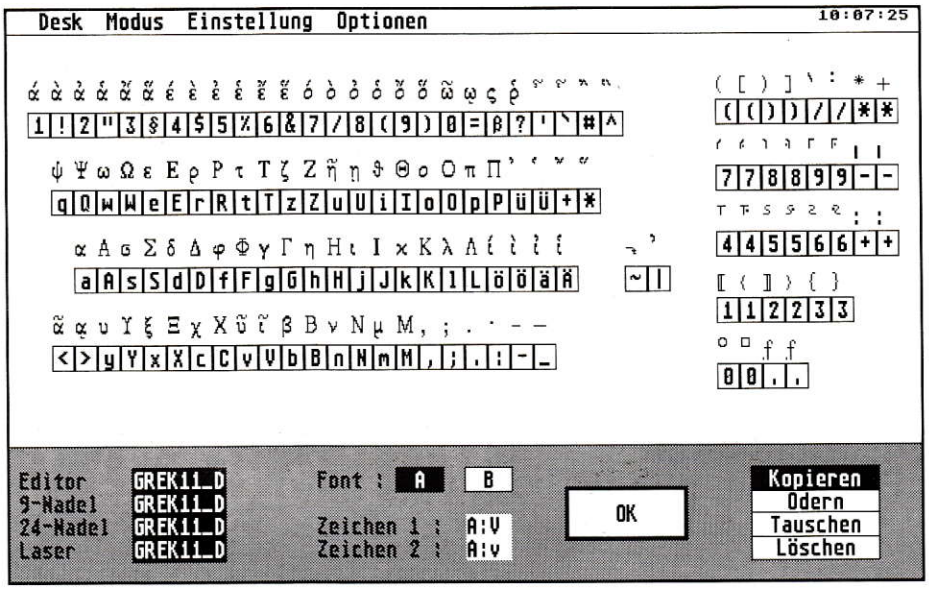


Bild 3: "Font spezial" - für buchstabenweises Bearbeiten

die des zu odernden, dann die des überschreibenden Zeichens, aber auch eigenständig gewählte Proportionen. Mein Tip: Bevor Sie odern, sollten Sie den betreffenden Font pur abspeichern und danach auf Position B laden und Buchstaben für Buchstaben die Proportionen vergleichen und verbessern. Für den, der wie ich alle drei Druckerformate bearbeiten muß, ist dieses Verfahren immer noch erheblich einfacher, als mit dem OFE alle drei Formate einzeln odern zu müssen. (Dieser Fehler wird laut Auskunft des Programmautors leicht zu beheben sein.)

Wer sehr große Fonts erstellen will oder wie beim Hebräischen sehr tief oder hoch sitzende Buchstaben und Zeichen hat, wird merken, daß gegenüber dem OFE das Druckernetz in der Vertikalen aus Platzmangel verkleinert ist. Der Hinweis, gegenüber dem OFE nach oben und unten z.T. ohne Verluste verschieben zu können, kann mich nicht befriedigen. Für diese Anwendungen wird ein neues Konzept erarbeitet werden müssen. Hierzu zählt auch das Problem mit dem Zahlenfeld des Menüpunktes "Font spezial": Alles was über eine bestimmte Höhe und Tiefe hinausgeht, wird nicht mehr dargestellt (bezieht sich nicht auf den wirklichen Editor- und Druckerfont), sondern nur auf die Monitordarstellung innerhalb dieses Menüpunktes.

Unter "Font tauschen" sollte die Alternative bestehen, einerseits in der momentan gewohnten Weise die Fonts darzustellen (optimal für lateinische Schriften), andererseits die wirklichen Editorfonts anzuzeigen. Wenn ich z.B. einen Arabischfont umbelegen muß, der sich an der arabischen Schreibmaschine orientierte, habe ich keine Assoziationsmöglichkeit mit den angezeigten lateinischen Buchstaben.

Unproportionaler Font: Dieser Menüpunkt ist an sich sehr begrüßenswert, doch verbesserungsbedürftig. Momentan sucht SCARABUS die größte Proportion innerhalb eines Fonts (wird in der Regel "W" oder "M" sein), stellt alle Proportionen auf diesen Wert und zentriert alle Buchstaben diesbezüglich. Sinnvoller wäre es, einen etwas kleineren Proportionswert vorgeben zu können. Die Proportion orientiert sich nämlich eher am kleinen "m"; d.h., die Großbuchstaben werden in einem unproportionalen Font etwas gestaucht werden müssen.

Ich selbst hatte auf einem Mega 4 [mit 2200 kB RAM-Disk (Flexdisk)] keine Speicherplatzprobleme. Auf einem 1040 soll es bei vielen Accessories schon mal eng werden können (wurde mir berichtet). Erstellen Sie sich dann für die Arbeit mit SCARABUS eine eigene Arbeitsdisk ohne Accessories. An dieser Stelle noch eine Merkwürdigkeit. SCARABUS selbst stürzte nie ab. Wenn ich aber direkt nach der Arbeit mit SCARABUS das Druckprogramm aufrief, stürzte dieses mit 2-3 Bomben manchmal ab. Liegt dies an der Speicherverwaltung, -freigebung durch SCARABUS?

Fazit

Wer als privater Anwender ausgefallene Fonts (outlined, shadowed etc.) erstellen möchte, sollte den Kauf von SCARABUS 2.0 in Erwägung ziehen. Der reine Hob-

byanwender, der nur mal einen schon vorhandenen Font bearbeiten möchte, ist mit den drei Original-SIGNUM-Fonteditoren bestens bedient - zumal dieser Anwender es nur mit einem Druckertyp zu tun hat. Für den professionellen Anwender gibt es keine Alternative zu SCARABUS 2.0. Allein die Zeitersparnis amortisiert schon den Kaufpreis von 100,- DM innerhalb kurzer Zeit.

Veit Brixius

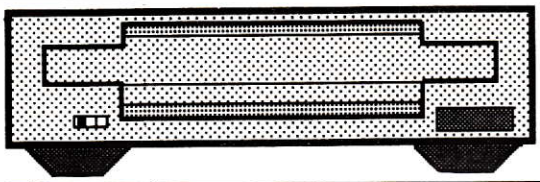
P.S.: Bezüglich der Fontdisketten Arabicum 2.0 (Arab.-Pers., Umschrift), Theolfont plus 2.0 (Hebr. Griech., Kopt., Jidd., Syr.) und Professional Fontdisk 3.0 (Times) dürfen Sie sich an mich wenden, wenn Sie Ihrer Anfrage mindestens 2,40 DM (Rückporto) in Briefmarken beilegen. Herzlichen Dank!

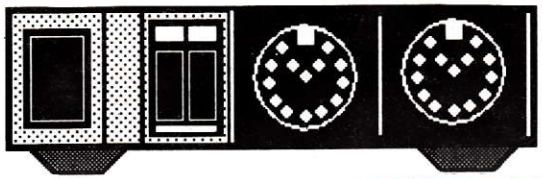
Veit Brixius
Römerstr. 48
D-6501 Budenheim

Bezugsadresse:

Application Systems /// Heidelberg
Englerstr. 3
6900 Heidelberg
Tel.: 06221/300002

PDD DISC DRIVES II Jetzt mit 1,2 MB





PDD-SERIE
DIES IST PDD-3

Grundausstattung: 100% ST-Compatibel

- ★ MIT NEC 1037a/1036a grau
- ★ MIT FLOPPYKABEL + NETZKABEL (> 1 m)
- ★ INTEGR. SPEZIALNETZTEIL + TRAFO (INTERN)
- ★ MIT STECKBAREN ANSCHLÜSSEN
- ★ 2. BUCHSE, AN/AUS-SCHALTER
- ★ 720 KB – 1,2 MB FORMATIERT
- ★ 2. SCHALTER FÜR A/B VERTAUSCH

PDD-3	EINZELSTATION (720 KB)	335,–
PDD-13	DOPPELSTATION (2 x 720 KB)	585,–
PDD-5	5 1/4 Zoll, Umschalter vorne! 40/80 Tracks + Software, IBM-Compatibel	389,–
PDD-16	3 1/2 Zoll plus 5 1/4 Zoll in einem Gehäuse, wie PDD-3 plus PDD 5 zusammen	689,–
PDD-18	2x3 1/2" + 5 1/4" in einem Gehäuse, A/B + 5/3-Schalter; wie PDD 5 + PDD 13 zusammen	898,–

WEITERE FLOPPYSTATIONEN (EINF. AUSFÜHRUNG) NEC 1037 A grau 169,–

ST-3	ST-13	ST-5	ST-16	Floppy Umbau
EINZELSTATION NEC 1037 A/1036 A KOMPL. ANSCHLUSSF. 229,–	DOPPELSTATION INTEGRIERTES NETZT. 2 x 720 KB 445,–	5 1/4 ZOLL, TEAK 55FR wesentlich verbesserte Version 40/80 TRACK 299,–	3 1/2 + 5 1/4 ZOLL NETZTEIL INTEGRIERT DOPPELSTATION 599,–	Wir bauen Ihr SF 354 um, mit NEC 1037 A. Für nur 219,– DM doppelseitig

PDD-4 498.- DM

wie PDD-3, jedoch mit 2 MB-Laufwerk (1,44 MB - über 2 MR formatierbar) Sub-Miniatur-D-Stecker (statt Buchsen; läuft am IBM, Atari ST, Atari II, Atari Workstation, etc. komplett im Metallgehäuse grau, eingebautes Spezialnetzteil, Netzschalter, Netzkabel, A/B Umschalter, 1/2 MB Umschalter (bei 2 MB Rechnern geht das automatisch); mit Atari Kabel für 2 Laufwerke, Bedienungsanleitung. Achtung: Bei allen Atari ST Modellen läuft dieses Laufwerk nur mit 1 MB (wegen Floppy-Controller), wie liefern jedoch bald Hardware-Zusatz, damit Ihr ST auch mit 2 MB läuft. Nähere Information: Bitte nachfragen! Kaufen Sie sich nicht für jeden neuen Rechner ein zusätzlich neues Laufwerk.

PC-Speed NEUESTE V 2.5

100% XT, ATARI kompatibel **IBM XT Emulator**
arbeitet 100% mit ATARI Programmen, keine Störungen dergleichen, wesentlich schneller als ein IBM XT, unterstützt hochauflösende Grafik Text Modus ebenso, 704 KB freier Speicher, hervorragendes Bildschirm-Bild; Festplatten plus Disketten plus Drucker kompatibel, keine Probleme!
Preis: 545.- **Express-Einbau + 75.-** **SOFORT LIEFERBAR**

FLAT LC DISPLAY

12" Bildschirm 640 x 440 Pixel, RGB analog, 8 Graustufen, s/w Bildschirm, flimmerfrei. Außerdem EGA/VGA/CGA/PS-2 mögl. 100% ST kompatibel; wird komplett geliefert mit Monitorstecker für ATARI, Kabel, Netzteil, Ständer. Aufstellen, einstecken, läuft...
Preis: 2.998,- Einfach professionell

Professionelle SCSI Festplatten für ATARI ST kompl. anschlußfertig

ATARI PC-Laufwerke
3 1/2", 5 1/4" Laufwerke, gemischt Stationen
Preis auf Anfrage

FISCHER-COMPUTER

Goethestraße 7 · 6101 Fränkisch-Crumbach · Telefon 06164-4601 + 3748 · abends zwischen 19.00 + 20.00 Uhr ebenso.



SPC-AdiProg

Daten bunkern mit Modula-2

Die Datenbank-Software ADIMENS ist für die verschiedensten Rechnersysteme erhältlich. Zur Entwicklung von Anwenderprogrammen gibt es zusätzlich eine Programmierschnittstelle namens AdiProg, mit der die Datenbankfunktionen in selbstprogrammierte Applikationen eingebunden werden können. Nach ST-Versionen für Mark-Williams-C und Turbo-C gibt es das System jetzt auch passend für Modula-Programmierer mit dem SPC-System von Advanced Applications Viczena (AAV).

Die ADIMENS-Implementierung für den ATARI ST ist schon eine Weile auf dem Markt und wurde in ST-Computer 1/87 vorgestellt. Einen Test der PROG-Schnittstelle für C finden Sie in Heft 11/88. ADIMENS ist eine relationale Datenbank, die Daten werden also in Tabellenform gespeichert, und einzelne Felder können als Schlüssel z.B. zum Sortieren verwendet werden. Zwischen mehreren Tabellen bzw. Dateien können Verbindungen über Felder existieren, in zwei oder mehr Dateien identisch vorkommen. Sind solche Verbindungen, die Relationen über mehrere Dateien, möglich, haben wir eine relationale Datenbank. Eine kleine Serie zur Programmierung solcher Datenbanken finden Sie in ST-Computer 4-6/88, allerdings dort auf dBASE-Kompatible bezogen.

Dieser Test soll sich nur mit der AAV-Implementierung beschäftigen und nicht in die Datenbanktheorie einsteigen. Am Anfang der Arbeit mit einem Software-Produkt steht immer die Installation. Und die gestaltet sich bei SPC-AdiProg einfach.

SPC-Modula wird durch eine - in der aktuellen Version grafische - Shell gesteuert, in der Suchpfade für das gesamte System verwaltet werden. Die Installation beschränkt sich also auf das Kopieren sämtlicher Dateien von der doppelseitigen Lieferdiskette und Setzen der Suchpfade mit dem SPC-Utility *PATHS*. Weitere Schritte sind nicht notwendig.

Zur Definition einer Datenbank wird das Originalprogramm *INIT* Version 2.3 von ADIMENS mitgeliefert. Es erlaubt auf komfortable Weise GEM-gestützt die Erstellung einer Datenbank mit Feld- und Maskendefinitionen. Auch Verzweigungen zwischen Dateien können ediert werden. Das zweite Originalprogramm, *REORG* (Version 2.23) muß von Zeit zu Zeit auf die Datenbank angesetzt werden, um auftretende Löcher in der Indexdatei zu entfernen.

Die Modulsammlung ist aufgeteilt in die eigentliche ADI-Schnittstelle samt Hilfsmodulen, einen Listen- und einen Maskengenerator sowie die Wordplus-Schnittstelle *WPProg*. Zur Vereinfachung der Programmierung gibt es zusätzlich das Modul *EasyPROG*.

Die ADI-Schnittstelle

Für den Programmierer laufen die eigentlichen Aufrufe der AdiProg-Funktionen über das Modul *AdiInterface*. Hier finden sich die Routinen z.B. zum Öffnen einer Datenbank oder zum Einlesen eines Datensatzes. Diese Funktionen sind der Kern des Datenbanksystems und erlauben für den Programmierer den Zugriff auf die Datenbank, die darin enthaltenen Dateien und die eigentlichen Datensätze und deren Felder. Mit *Find*, *Look* und *Next* kann der Dateizeiger entsprechend einem Suchmuster positioniert werden. Die Anzahl der Kernroutinen ist gering und bei allen AdiProg-Implementierungen identisch.

Von ADIMENS ist inzwischen eine netzwerkfähige Version erhältlich, die mit eLAN, Bionet und Rhonet arbeitet. AAV plant für die Netzwerkunterstützung wohl eine eigene Lösung. Diese Situation ist nicht weiter verwunderlich, da auch die Anbieter von Netzwerken momentan Software-Unterstützung zunächst für C-Systeme anbieten.

Zur Arbeit mit den eingelesenen Daten dient *AdiUtilities*. Die bereitgestellten Funktionen konvertieren z.B. Daten von Zeichenketten in die ADI-interne Darstellung und umgekehrt und bieten einige Routinen zur Manipulation des Datensatz-Puffers. Auch finden sich hier Routinen zur Verarbeitung der Mengen von Adimens (512 Elemente, 64 Bytes) und

die Funktionen zur Datumsdarstellung, wobei vier landesspezifische Parametersätze implementiert sind.

Ebenfalls in *AdiUtilities* sind einige Routinen für die interne Speicherverwaltung des Datenbankkerns vorhanden. Von ihrer Benutzung wird dringend abgeraten, vielleicht hätten sie in ein extra Modul gehört.

Der gesamte Kern von ADIMENS ist in C programmiert und liegt als ADI.PRG vor. Bei der Initialisierung der ADI-Schnittstelle aus dem Modula-Programm heraus wird dieser cirka 65 kByte große Kern von Disk oder Platte während der Laufzeit nachgeladen und dabei einmalig der Zugriffspfad in einer Environment-Variablen vermerkt. Zwei weitere interne Module übernehmen das Laden und Ermitteln der C-Routinen automatisch. Da der direkte Start von *ADI.PRG* vom Desktop keinen Sinn macht, wäre hier vielleicht ein Umbennen, z.B. *ADI.IMG*, wünschenswert.

Zur Abrundung der eigentlichen ADI-Schnittstelle sind im Modul *Utilities* weitere Hilfsroutinen vorhanden. Datensätze können in Strings für eine Listenausgabe oder in 24*79 Zeichen Bildschirm-Puffer konvertiert werden. Auch sind hier die Feldnamen abfragbar. Zur Verwendung der Struktur der Datenbank kann auf das nächste Schlüsselfeld geschaltet und die in *INIT* definierten Verzweigungen zwischen Dateien benutzt werden.

Dieser Kern der AdiProg-Routinen enthält gegenüber der C-Version nicht die Routinen, die beispielsweise das Datei-Handling, Directory-Operationen oder den Umgang mit Druckern bereitstellen. Dies ist auch nicht nötig, da solche Funktionen schon zum Standardumfang von SPC-Modula gehören.

Listengenerator

In der C-Version von AdiProg sind GEM-Routinen vorhanden, die eine Programmierung einer Benutzeroberfläche ähnlich dem original Abfrageprogramm *EXEC* ermöglichen (in der Library *GEM-PROG*). AAV hat diesen dort sehr umfangreichen Teil weggelassen und stattdessen zwei Module implementiert, die basierend auf dem SPC-Fenstersystem *SSWiS* Bildschirmdarstellungen unterstützen.

Der Listengenerator verwaltet die Ausgabe von Listen in einem Textfenster. Die Bestandteile der Listen sind natürlich die aus der Datenbank gewonnenen Felder. Das Modul *ListLib* kann Listenfenster öffnen und verwalten. Eine Redraw-Routine erhält zum Neuzeichnen des Fensterinhalts eine Anfangs- und Endzeile und kann die Daten mit der speziellen *WriteField*-Routine ausgeben.

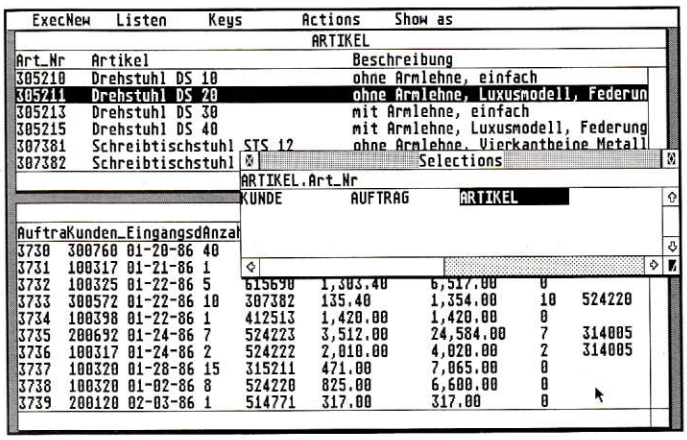


Bild 1: Die Bildschirmdarstellung mit dem Listengenerator

Der Clou ist, daß der Benutzer in diesen Fenstern einzelne Zeilen und Felder per Maus selektieren kann. Welche Listenteile ausgewählt wurden, kann mit Abfragerouti-nen einfach ermittelt werden. Was mit diesen Feldern geschieht, ist dann natürlich Sache der Applikation. Entsprechend der Konzeption von SSWiS erhält jedes Listenfenster eine eigene Menüzeile, die wiederum mit Funktionen aus der *ListLib* erstellt wird.

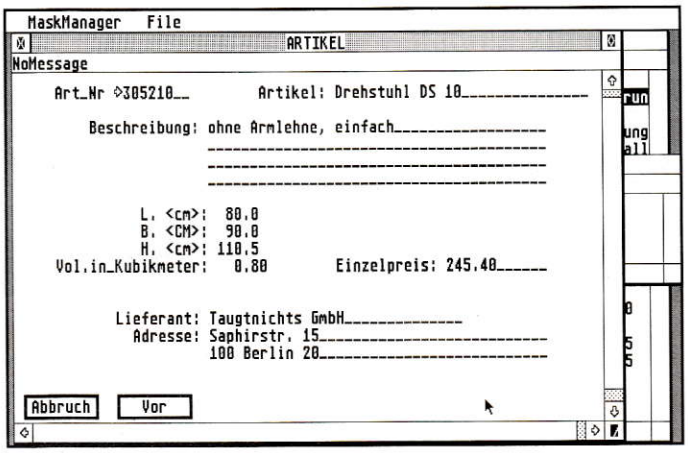


Bild 2: Eine Bildschirmmaske in einem SSWiS-Fenster

Bild 1 zeigt drei mit dem Listengenerator programmierte Listenfenster, die Daten einer Beispieldatenbank enthalten. Das oberste Fenster enthält die Namen der Dateien in einer horizontalen Liste, wobei das Anklicken einer Datei ein neues Listenfenster für den Inhalt der Datei öffnet. Im Hintergrund sehen Sie zwei solche Fenster.

Maskengenerator

Wie oben genannt, befindet sich in dem Modul *Utilities* eine Routine, die einen Datensatz nach der Maskendefinition in ein Feld aus Zeichenketten umwandeln kann. Jedoch geht diese Routine von einem herkömmlichen Textbildschirm aus. AAV hat einen Maskengenerator implementiert, der mit dem Fenstersystem SSWiS zusammenarbeitet und erheblich mehr GEM-Elemente erlaubt (Bild 2).

Zunächst müssen dazu die mit *INIT* erstellten Masken mit dem Modul *ReadDecl* eingelesen und danach mit *MaskHeap* Speicherplatz für die Masken angefordert werden. Zusätzlich wird man die Maskenbeschreibung aus *ReadDecl* umbauen müssen, da der Maskengenerator andere Datenstrukturen verwendet. Hier wäre vielleicht ein kleines Set an Konvertierungsroutinen angebracht.

Das Programm baut dann eine Maske durch mehrfachen Aufruf von *DefElem* auf. Für jeden Maskenbestandteil muß eine Beschreibung übergeben werden, in der z.B. die Bildschirmposition und der Text von Buttons steht. Mögliche Elementtypen sind *Title* für Texte, *Button* für GEM-Knöpfe, *Content* für die von GEM-Dialogen her bekannten edierbaren Felder und schließlich *UserDef* für selbstprogrammierte Objekttypen.

Für alle Elemente lassen sich Eigenschaften setzen, die auch vom GEM-Dialogmanager her bekannt sind. Teile können wählbar oder z.B. ausgeschaltet sein, Texte können formatiert werden, und Buttons sind mit verschieden starker Umrandung darstellbar. Bei edierbaren Feldern wird ein Maskenstring festgelegt, der bestimmt, welche Zeichen an welcher Stelle zur Eingabe erlaubt sind.

Über die Einführung von programmierbaren Objekten ist es z.B. auch möglich,

Grafiken darzustellen. Im Bild 3 (aus einem Beispielprogramm) ist dies natürlich eine Spielerei, die Darstellung eines gescannten Formulars im Hintergrund macht eine Maske allerdings spürbar anwenderfreundlicher.

Eine Reihe von Service-Funktionen erlaubt das Verändern der Attribute von Elementen, so z.B. das Setzen von Strings oder das Einfügen von einzelnen Zeichen in ein edierbares Maskenelement.

Um die Maske nun aktiv zu machen, wird ein SSWiS-Fenster eröffnet und darin die Maske dargestellt. Benutzereingaben werden dann mit der Routine *SSWiS.PollEvents* abgefragt und automatisch vom Fenstersystem an den für jede Maske zu programmierenden Eventhandler weitergeleitet. Dieser erhält dann z.B. die Mitteilung, daß eine Cursor-Taste gedrückt wurde, und muß dann mit der Routine *MoveCursor* des Maskenmanagers reagieren. Auch die Redraw-Events werden von SSWiS geliefert und können mit weiteren Service-Routinen in Funktionen von *MaskManager* befriedigt werden.

Der *MaskManager* ist also eine Sammlung von Funktionen, die die Maskenprogrammierung unterstützen. Vergleicht man die Masken mit den Standard-GEM-Formularen, so liegen seine Fähigkeiten etwa auf der Ebene des Form-Managers von GEM. Zur Verarbeitung von Benutzereingaben muß der Programmierer die SSWiS-Ebene benutzen und wird dabei sinnvoll von einigen Service-Routinen unterstützt. Für eine konkrete Anwendung wird man sich noch einige kleinere Funktionen hinzuprogrammieren müssen oder Prozedurrahmen aus den mitgelieferten Beispielprogrammen übernehmen.

xAllerdings bietet das Angebot aus *MaskManager* natürlich auch eine große Flexibilität: Während des Ausfüllens einer Maske könnten Felder aus der Datenbank nachgeladen werden. Bei einem Rechnungsformular könnte nach dem Ausfüllen des Namensfelds in der Datenbank nachgeschaut werden, ob die Adresse vielleicht schon vollständig in einer Kundendatei vorliegt. Wäre dies nicht der Fall, müßte der Benutzer weiter ausfüllen. Das Programm könnte aber sofort dann, wenn alle Adressenfelder ausgefüllt sind, die Kundendatei erweitern. Läge die Adresse vor, könnten die entsprechenden Felder schon während der Datenerfassung in der Maske ergänzt werden.

Easy-PROG

EasyPROG ist eine Erweiterung bzw. Vereinfachung der ADI-Schnittstelle. Die dortigen Routinen arbeiten mit Puffern, die im Speicher dynamisch angelegt werden müssen. Zum Bearbeiten eines Datenfeldes muß das Programm dessen Adresse im Puffer mit diversen Funktionen ermitteln. Weiterhin sind einige Parameter - z.B. die Dateinummer - bei jedem Funktionsaufruf zu übergeben, obwohl sie in einer realen Anwendung nicht ständig wechseln werden.

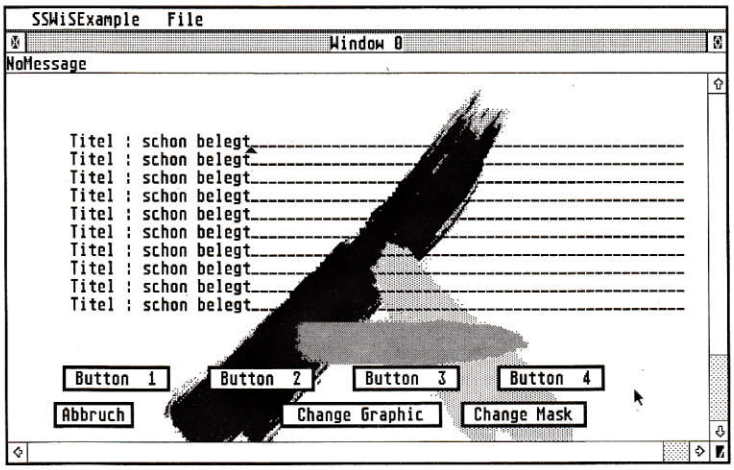


Bild 3: Eine Maske mit Grafik

EasyPROG liegt nun in der Modulhierarchie über der ADI-Schnittstelle und bietet deren Dienste in komfortablerer Weise an. Das zweite der genannten Probleme umgeht *EasyPROG* dadurch, daß nur einmal z.B. die Datei per *In(<Dateinummer>)* auszuwählen ist, die dann bei jedem Funktionsaufruf automatisch an die ADI-Schnittstelle weitergeleitet wird.

```
...
VAR name, vorname, strasse,
    ort:ARRAY[0..22] OF CHAR;
    i, nummer, plz : INTEGER;
...
  IF InitBase('E:\spc\user\','adr') THEN
    ConnectText(name,1);
    ConnectText(vorname,2);
    ConnectText(strasse,3);
    ConnectInt(nummer,4);
    ConnectInt(plz,5);
    ConnectText(ort,6);
    In(1);
    Reset;
    UseKey(1);
    SetTextKey('?');
    IF Find() THEN
      REPEAT
        Access;
        WriteString(name);      WriteLn;
        WriteString(vorname);   WriteLn;
        WriteString(strasse);   WriteLn;
        WriteInt(nummer,5);     WriteLn;
        WriteInt(plz,5);        WriteLn;
        WriteString(ort);       WriteLn;
      UNTIL ~Look();
    END;
  ExitBase;
...
```

Listing 1

Um das umständliche Hantieren mit dem Puffer zu vermeiden, gibt es die *Connect...*-Routinen. Mit ihnen kann ein Feld an eine Modula-Variable gebunden werden. Nach *ConnectInt(anzahl,1)* gilt das erste Feld eines Datensatzes in der aktuellen Datei als an die INTEGER-Variable *anzahl* gebunden. Beim Einlesen eines Feldes mit *Access* kopiert *EasyPROG* automatisch den Pufferinhalt in die Variable, mit der dann weitergearbeitet werden kann.

Listing 1 zeigt einen Programmausschnitt, in dem mit *EasyPROG* alle Datenfelder der Datenbank *adr* in der Datei 1 ausgegeben werden. Vorteil ist, daß das Speichermanagement für den Puffer entfällt und zudem das Anwendungsprogramm weniger mit Zeigervariablen umgehen muß. Insgesamt also ein Hilfsmodul, das die Arbeit deutlich vereinfacht und schließlich auch sicherer macht.

WPProg

Um die Verbindung zwischen Datenbank und Textverarbeitung zu ermöglichen, stellt SPC-AdiProg eine Schnittstelle zu Wordplus in dem Modul *WPProg* bereit. Es läßt sich übrigens auch unabhängig von den ADIMENS-Routinen benutzen. Vielleicht sollte AAV dieses Modul auch einzeln anbieten.

Sechs Module bieten auf verschiedenen Ebenen ihre Dienste an. *WPCode* enthält die diversen Typ-Definitionen - z.B. für Formatzeilen - und Grundfunktionen, um zwischen diesen Typen und der Wordplus-Darstellung in ASCII-Zeilen zu wandeln.

WPParam stellt Routinen zum Setzen der Parameter eines Wordplus-Dokuments bereit, also Dinge wie Seitenlänge, Fuß-

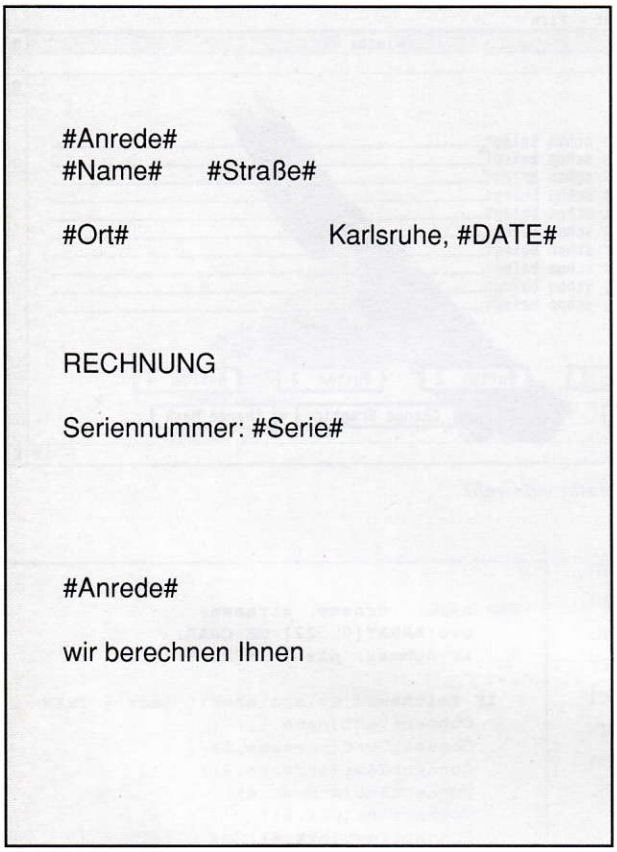


Bild 4: Eine Vorlage für Serienbriefe für WPMail

notenlinie oder die Lineallänge. Die Routinen erhalten als Parameter einen zu ändernden Record vom Typ *WPFormat*, in dem das Dokumentenformat gehalten wird, und überprüfen die Korrektheit der neuen Werte.

WPPrint erlaubt die Verarbeitung von Wordplus-Dokumenten auf dem Massenspeicher. Hier finden sich die Routinen zum Lesen und Schreiben einzelner Zeilen aus und in eine Datei. Es können bei der Ausgabe alle Wordplus-Bestandteile wie Seitenumbrüche oder Fußnoten erzeugt werden. Auch die verschiedenen Schriftarten lassen sich verwenden. Für das Programm werden alle Textzeilen als ARRAY OF CHAR behandelt; die verschiedenen Codes - z.B. für Fettschrift - hängen entsprechende Funktionen an. Übrigens lassen sich auch Bilder verarbeiten.

Die verschiedenen Formatierungsroutinen für Textzeilen sind in *WPFStr* zusammengefaßt. Hier werden die Zeichenfelder z.B. links- oder rechtsbündig formatiert und in einem weiteren String zurückgeliefert. Das Formatieren berücksichtigt natürlich die Wordplus-Steuerzeichen.

WPConv stellt Konvertierungsroutinen zwischen Wordplus-Dokumenten auf Disk bereit. Hier kann z.B. zwischen ASCII- und Wordplus oder zwischen der Wordplus-Version 2 und 3 gewandelt werden. Auch kann *WPConv* z.B. automatisch ein Inhaltsverzeichnis erstellen.

Das letzte Modul *MPMail* erlaubt es schließlich, Serienbriefe zu erstellen. Hier wird automatisch ein Wordplus-Dokument, das Platzhalter enthält (Beispiel in Bild 4), mit einer Ersetzungstabelle gemischt und schließlich der neue Text auf Disk geschrieben. Bei der Arbeit mit AdiProg extrahiert man dann die Ersetzungstexte natürlich jeweils aus der Datenbank.

00? - Lizenz zum Verkaufen

Als Vertreiber von mit AdiProg hergestellter Software hat man aufgrund der Lizenzbedingungen bestimmte Verpflichtungen gegenüber dem eigentlichen Hersteller ADI Software. Verlangt werden Belegexemplare, ein Hinweis auf ADI Software im Programm, und schließlich darf das Produkt nicht mit AdiProg oder ADIMENS konkurrieren. Es sind zwei Vertriebsformen gestattet, und zwar entweder ohne *INIT* und *REORG*, aber mit dem Hinweis, daß diese nachträglich erworben werden müssen, oder als Komplettpaket mit einer anteilig berechneten Lizenz durch ADI Software.

Die Höhe der Lizenz muß jeweils individuell vereinbart werden, von einem Festpreis, wie bei früheren AdiProg-Versionen, ist ADI Software inzwischen abgerückt.

Handbuch

Das Handbuch umfaßt fast 320 Seiten und wird in einem Ringordner geliefert. Der Update-Service von Advanced Applications für das Modula-System arbeitet sehr konsequent auch mit Handbuchergänzungen; die Ringordner-Lösung hat sich dabei sehr gut bewährt.

Das Manual besteht aus vier Teilen. Im ersten wird die Programmierung mit der Schnittstelle und den zusätzlichen Modulen in einer Art Tutorial sehr übersichtlich erläutert. Die abgedruckten Programmbeispiele geben einen Rahmen für wiederkehrende Aufgaben vor.

Der Referenzteil geht die einzelnen Module Funktion für Funktion durch. Am Seitenrand ist jeweils der Prozedurname aufgeführt, so daß ein Nachschlagen erleichtert wird. Die verwendeten, oftmals recht umfangreichen Datentypen finden sich in der Nähe der beschriebenen Funktion, so daß großes Blättern erspart bleibt. Erfreulich auch ein System von Querverweisen auf mit der dargestellten zusammenhängende Funktionen.

Die Beschreibung der Programme *INIT* und *REORG* wurde direkt aus dem AdiProg-C-Handbuch übernommen. Vielleicht hätten auch noch die einleitenden Kapitel über das Konzept der ADI-Datenbank aus dem Original übernommen werden sollen. Die bei Modula übliche Listung aller Definitionsmodule bildet den Abschluß des Handbuchs, das zwar noch ein Register verdient hätte, aber insgesamt gelungen ist.

Als Ergänzung zum Handbuch sind auch die mitgelieferten Beispielprogramme zu sehen. Ein Programm, *ExecNeu*, zeigt in der Praxis, wie man mit einer Datenbank beliebigen Formats umgeht. In einer konkreten Anwendung wird man wahrscheinlich Teile daraus in sein eigenes Programm übernehmen.

Fazit

SPC-AdiProg ist eine saubere Implementierung der ADIMENS-Schnittstelle. Sie fügt sich sehr gut in das Modula-Entwicklungspaket ein und enthält sinnvolle und leistungsfähige Erweiterungen. Die Funktionssammlung ist hervorragend strukturiert und eine konsequente Erweiterung des Produktangebots für SPC-Modula. SPC-AdiProg kostet DM 249,- und kann bezogen werden bei

Advanced Applications Viczena GmbH
Sperlingweg 19
7500 Karlsruhe 31
Tel.: 0721/700912

Unser ProFile-System – Ihr Vorteil.



Unser ProFile-System bietet ihnen die Zuverlässigkeit und die Flexibilität, die Sie als Atari-User heute brauchen. Spitzenentwickler waren an der Arbeit – das Ergebnis sind Geräte mit den besten Eigenschaften.

Festplatten von 20 bis 180 MB Speicherkapazität für eine schnelle – und leise – Verarbeitung und Verwaltung von Daten. Die DC-Serie ist zusätzlich mit einem integrierten 64-KB-Cachespeicher ausgerüstet, der die Zugriffszeit um bis zu 50 % verringert.

protar **– eine sichere Entscheidung für die Zukunft**

Bis zu 2 Jahre Garantie gibt die notwendige Sicherheit. Zu unserer Produktpalette gehören ebenso wiederbeschreibbare optische Speicher und Monochrome-Bildschirme sowie das ProFile R44 – ein Wechselplatten-System – mit dem Sie 44 MB schnell zur Hand haben.

Informationen zu protar-Produkten erhalten Sie bei Ihrem qualifizierten Fachhändler.

protar Elektronik GmbH · Puttkamer Straße 7 · D - 1000 Berlin 61 · Telefon 030 - 251 30 65 · Telefax 030 - 251 30 68
Vertretung in Österreich: Dipl.-Ing R. Temmel Ges. m.b.H. & Co. KG · Markt 109 · A - 5440 Golling · Telefon 06244 - 70 81 0 · Fax 06244 - 71 88 3
Vertretung in der Schweiz: DTZ DataTrade AG · Landstraße 1 · CH - 5415 Rieden/Baden · Telefon 056 - 82 18 80 · Telefax 056 - 82 18 84

TeX für den Alltag

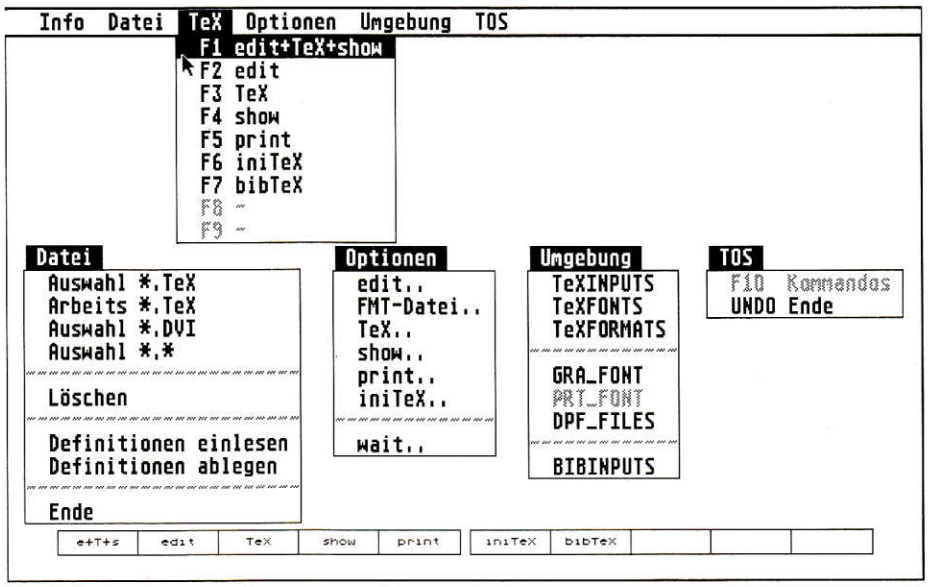


Bild 1: Die Shell mit allen Pull-Down-Menüs. Am unteren Bildschirmrand ist immer die Belegung der Funktionstasten zu sehen.

Das Satzsystem TeX zur Produktion erstklassiger Texte dürfte unseren Lesern ja mittlerweile zum Begriff geworden sein. Hier geht es nun um eine Implementierung dieses Systems auf dem ATARI ST, wie sie in dieser Vollständigkeit und benutzerfreundlichen Ausführung bisher noch nie geboten wurde. Es handelt sich um die unter dem Namen STTeX angebotene Implementierung der TH Darmstadt, die von der Firma "Detig Schrod TeXsys" vertrieben wird.

Das Satzsystem TeX (sprich: tech) wandelt mit Steueranweisungen versehene Texte in Druckausgabe beliebig hoher Qualität um. Der Text wird gemäß den Steueranweisungen und dem vorgewählten Format aufbereitet, also gesetzt, das Ergebnis dieses Arbeitsgangs wird dann mittels der verfügbaren Zeichensätze auf das gewünschte Ausgabegerät in der für dieses Gerät maximal möglichen Druckqualität ausgegeben. Im einfachsten Fall ist das Ausgabegerät der Bildschirm, daneben findet man im privaten Bereich häufig einen Star NL-10 oder einen anderen Epson-kompatiblen 9-Nadeldrucker. Wer etwas mehr anlegen kann, ist mit einem 24-Nadeldrucker gut beraten. Hier erreicht man 180 mal 180 dpi, bei versetztem Drucken bei manchen Geräten sogar bis zu 360 mal 360 dpi. In kleineren Sekretariaten kann die mit diesen Druckern erreichbare Qualität und Geschwindigkeit oft ausreichen. Im professionellen Einsatz haben sich Laserdrucker, hier insbesondere der HP-Laserjet und Kompatible, mit einer Auflösung von 300 mal 300 dpi gut bewährt. Für Veröffentlichungen höchster Qualität schließlich besteht die Möglichkeit der Ausgabe über eine Linotype, die nun wirklich keine Wünsche mehr offen läßt.

Ohne Fleiß kein Preis

Ein solches Super-System stellt aber auch Super-Anforderungen: an die Maschine, auf der ein solches Programm ablaufen soll, aber auch an den Benutzer, der fast schon eine Programmiersprache zu erlernen hat, bis er endlich durch einen Ausdruck in der gewünschten Form belohnt wird. Schon mehrfach hat sich die CT-Computer dieses Themas angenommen: In der ST 7/87 verglichen wir die beiden TeX-Implementierungen der Firmen TooLs und Kettler, in der ST 5/89 wurde ein kleiner Einblick in das Arbeiten mit TeX gegeben. An dieser Stelle allerdings gleich ein kleines Dementi: Die im zuletzt genannten Artikel vorgeschlagene Abtrennung des TeX-Logos durch das Sonderzeichen "~" sollte man natürlich tunlichst vermeiden. Dadurch wird nämlich nicht nur ein Zwischenraum erzwungen, sondern auch gleichzeitig ein Zeilenumbruch unterbunden. Richtig wäre die Abtrennung durch einen Backslash "\" mit nachgestellter Leerstelle oder durch eine leere Gruppe "{}", ebenfalls gefolgt von einer Leerstelle. Dies wird übrigens im TeXbook [1] auf den Seiten 8 und 19 ausführlich erläutert. Wie man sieht, ist die Sache doch nicht ganz so einfach.

Wenn nun schon der Umgang mit TeX an sich nicht ganz trivial ist, so sollte doch wenigstens die Bedienung der hierfür erforderlichen Programme einfach von der Hand gehen, damit dem Anwender der Kopf frei bleibt für das eigentliche Ziel: die Dressur von TeX. Dieser Anforderung wird das hier getestete Programmpaket in so vollem Umfang gerecht, daß man es mit gutem Gewissen für den Einsatz in ganz normalen Sekretariaten empfehlen kann, in denen die Schreibkräfte zwar eine gewisse Bereitschaft zum Erlernen des Umgangs mit dem Computer und zum Einarbeiten in TeX mitbringen, ohne jedoch schon Computerexperten zu sein.

Lieferumfang

Im Prinzip besteht das TeX-System aus zwei voneinander unabhängigen Teilen, die man auch einzeln bestellen kann. Da

ist zunächst einmal das Grundpaket mit dem eigentlichen TeX-Programm, einer umfangreichen Sammlung von Makro-Paketen, INITeX zur Generierung von Formatdateien und BibTeX zum Erstellen und Verwalten von Literaturlisten. Dazu gehört ein Preview, das ist ein Programm zur Ausgabe der von TeX erzeugten DVI-Dateien auf den Bildschirm. Vervollständigt wird das Ganze durch einen Druckertreiber, der die DVI-Dateien auf dem gewünschten Drucker zu Papier bringt. In der Regel wird man das Gesamtpaket mit TeX/Preview und dem Druckertreiber für das vorhandene Ausgabegerät erwerben. Bestandteil des Grundpakets ist weiter eine einfach zu bedienende graphische Benutzeroberfläche, heute heißt so etwas auch Shell, mit der die wichtigsten Programme des TeX-Systems in einfachster Weise per Tastendruck oder Mausklick, es ist beides möglich, bedient werden.

Das Grundpaket bekommt man auf sechs Disketten; davon sind drei für das Preview-Programm namens DVIGRA erforderlich, die fast ausschließlich für Zeichensätze benötigt werden. Der Platzbedarf für die Druckertreiber wird natürlich ebenfalls durch die Zeichensätze bestimmt. Wir testeten den DVIDOT mit einem Star NL-10 (6 Disketten) und den Lasertreiber DVIHP für den HP-Laserjet (9 Disketten).

Dokumentation

Selbst die allerbeste Software ist ohne gute Dokumentation nicht viel wert. Man kann sich nur wundern, wie oft und mit welcher Penetranz gegen diese eiserne Regel selbst von ansonsten exzellenten Programmierern verstoßen wird. Das vorliegende Paket ist ein äußerst gelungenes Beispiel dafür, wie gute Dokumentation aussehen kann. Und hier reden wir noch nicht einmal von der den umfangreichen Makro-Paketen beigegebenen Originaldokumentationen, sondern von der reinen Programmbeschreibung des vorliegenden Pakets. Diese Dokumentation setzt sich aus vier Handbüchern zusammen. Da gibt es zunächst unter dem Titel "STTeX" (25 Seiten) eine Beschreibung des Grundpakets, in der nach einer allgemeinen und knappen Einführung in die Arbeitsweise mit TeX ausführlich die Installation des Gesamtpakets auf den verschiedenen Gerätekonfigurationen erläutert wird. Ebenso wird hier der Umgang mit der Shell und diversen Hilfsprogrammen erkärt.Weiter findet man Tips, wie bei Platzproblemen vorzugehen ist. Teil 2 heißt "LaTeX auf dem ATARI ST" und enthält auf 13 Seiten den "local guide" für dieses Makro-Paket. Neben anderem gibt es hier knappe Hinweise zu den einzelnen vordefinierten Styles. Eine umfangreiche Liste mit Ergänzungen und Korrekturen zum LaTeX-Buch von Leslie Lamport [3] ist übrigens auf Diskette mitgegeben. Der dritte Teil (14 Seiten) heißt "TeX-Preview für ATARI ST Systeme" und erläutert die Handhabung des Bildschirmtreibers DVIGRA. In einem Anhang wird auf die Ausgabe von Bitmap-Graphiken eingegangen. Teil 4 (16 bis 18 Seiten) beschreibt den Umgang mit dem jeweiligen Druckertreiber. Hier wird unter anderem die separate Installation des Druckertreibers erklärt und nochmals ausführlich auf die Ausgabe von Graphiken eingegangen.

Installation

Bei der Installation, wie auch bei allen anderen Arbeitsschritten, erhält man zuverlässige und ausführliche Unterstützung durch die eben beschriebene Dokumentation. Schon bei einem einfachen 9-Nadeldrucker hat man es nach dem Obigen bereits mit 12 Disketten zu tun. Diese enthalten nun nicht einfach irgendwelche einzelne Dateien, sondern oft mit dem bekannten ARC-System gepackte Archive, um Platz zu sparen. Somit liegt es auf der Hand, daß das Einrichten eines gebrauchsfertigen Systems eine nichttriviale Angelegenheit ist. Weiter dürfte klar sein, daß man mit einem reinen Diskettensystem nicht weit kommt. Eine Festplatte ist für menschenwürdiges Arbeiten mit TeX unbedingte Voraussetzung.Mit einiger Sachkenntnis und erheblicher Selbstbeschränkung könnte man natürlich zur Not noch mit zwei Diskettenlaufwerken auskommen, die Anleitung zur Installation beschreibt nicht nur dies, sondern in geradezu rührender Gutmütigkeit auch noch die Installation auf einem System mit nur einem (!) Laufwerk. Ein solches Unterfangen ist jedoch nur Fanatikern zu empfehlen, die damit ins Guinness-Buch der Rekorde kommen wollen nach dem Motto: "Wer hält es am längsten mit TeX ohne Festplatte aus?" Der Betrieb mit Disketten ist somit nur eine Notlösung, auf die wir hier nicht weiter eingehen wollen.

TeX soll also auf einer Festplatte installiert werden. Diese ist in aller Regel zuvor gründlich aufzuräumen, damit genügend Platz ist. Eines wird der potentielle TeX-Benutzer nämlich schnell merken: TeX ist ein Speicherfresser, es emuliert sozusagen das Krümelmonster auf dem ATARI ST. Der Appetit erstreckt sich sowohl auf Haupt- als auch auf Plattenspeicher. Dies ist unter anderem der Grund, weshalb mit TeX auf einem IBM-PC nicht vernünftig zu arbeiten ist: Auf Maschinen mit weniger als einem Megabyte Hauptspeicher fristet TeX nur ein eher kümmerliches Schattendasein. Die Anleitung spricht von 6 Megabyte Plattenspeicherbedarf für STTeX, das umfaßt das TeX-Paket mit allen Makro-Paketen, der Dokumentation, allen Hilfsprogrammen und den Zeichensätzen des Bildschirm-Previews. Hinzu kommen noch einige Megabyte für die Zeichensätze des Druckertreibers, für den Laserdrucker sind dies nicht ganz 6 Megabyte, der NL-10 ist mit etwa 5 Megabyte zufrieden. Werden das TeX-System und der Druckertreiber gemeinsam installiert, so liegen die Zeichensätze für Bildschirm und Drucker in einer gemeinsamen Partition. Hier ist zu überlegen, ob man nicht den Druckertreiber separat installiert und dann dessen Zeichensätze in einer eigenen Partition unterbringt.

Für jedes Gerät liegen sämtliche Zeichensätze einer Vergrößerungsstufe jeweils in einem Ordner. Die Ordner mit den einzelnen Vergrößerungsstufen eines Ausgabegeräts kommen dann nochmals in einen gemeinsamen Ordner, damit sind die Zeichensätze zu verschiedenen Geräten besser auseinanderzuhalten. Hält man jedoch die Zeichensätze unterschiedlicher Geräte auch in unterschiedlichen Partitionen, so kann auf den letzten Ordner verzichtet werden, die Ordner mit den Vergrößerungsstufen liegen direkt im Hauptinhaltsverzeichnis der jeweiligen Partition und ermöglichen somit einen etwas schnelleren Zugriff. Da die Größe einer Partition der Größe der Zeichensätze ganz gut angepaßt werden kann, braucht man auch keine Probleme wegen mangelnder Übersicht mit den vielen Ordnern im Hauptinhaltsverzeichnis zu befürchten: Die Partition wird eben vollständig für Zeichensätze reserviert. Auf einer Festplatte mit 20 oder 30 Megabyte ist es ganz vernünftig, das TeX-System mit den Zeichensätzen für den Bildschirm auf einer und die Zeichensätze für den Drucker auf einer anderen Partition zu haben. Die zu bearbeitenden Texte werden dann am besten auf einer dritten Par-

tition gehalten, so daß man nicht versehentlich etwas von seiner Installation löscht oder überschreibt. Sind die Texte, die man in der Regel ja laufend ändert, auf diese Weise separiert, ist auch das Archivieren einfacher. Zieht man beispielsweise regelmäßig ein Backup, dann braucht man sich nur noch um eine Partition zu kümmern, da auf den anderen ja nichts verändert wird. Bei der Installation wird der gesamte im Handbuch angegebene Platz zunächst auch tatsächlich benötigt, auf die Tricks für Bastler (und bitte nur für solche!) kommen wir später.

Einfacher, als man denkt

Obwohl die umfangreiche Diskettenlieferung zunächst natürlich Respekt einflößt, ist die Installation tatsächlich ein Kinderspiel. Dies liegt daran, daß die Installation vollständig automatisiert ist und menügesteuert abläuft. Der Anwender muß sich nur zuvor überlegen, auf welchen Partitionen das Grundsystem und die Zeichensätze installiert werden sollen und diese freiräumen oder gegebenenfalls auch vergrößern oder verkleinern. Anschließend ist die Startdiskette einzulegen und das Programm INSTALL.TOS aufzurufen. Von nun an hat man nur noch auf einfache Fragen zu antworten (Wollen Sie die Standard-Installation übernehmen? Auf welcher Partition soll TeX installiert werden?) und Disketten zu wechseln. Bei allen Fragen werden Antworten zu einer Standard-Installation vorgegeben, die man einfach mit RETURN bestätigen kann. In der Regel wird man dies auch tun und allenfalls in der Wahl der Partitionen abweichen. Alle benötigten Ordner werden automatisch angelegt, zum Diskettenwechsel wird man unter Angabe des Namens der gewünschten Diskette jedesmal extra aufgefordert. Viel Zeit sollte man jedoch schon mitbringen: Die Installation des Grundpakets dauert etwa eineinhalb Stunden, für den Druckertreiber muß man nochmal mit der gleichen Zeit rechnen. Da nach dem anfänglichen Dialog aber nur noch Disketten zu wechseln sind, kann man getrost nebenbei Zeitung lesen. Im Hinblick darauf wäre es eine sinnvolle Verbesserung, die Aufforderung zum Diskettenwechsel mit einem akustischen Signal zu unterstützen.

Nach Abschluß der Installation findet man seine Festplatte wohlgefüllt. Den größten Platz beanspruchen natürlich die Zeichensätze, aber auch durch Anzahl und Umfang der mitgelieferten Makro-Pakete und Hilfsprogramme wird man angenehm überrascht.

Mother's little helpers

An Hilfsprogrammen gibt es zunächst INITeX, mit dessen Hilfe neue Makro-Pakete generiert und bestehende verändert werden können. Weiter gibt es BIB-TeX, das die komfortable Handhabung von Literaturzitaten in LaTeX unterstützt, einschließlich einer Sammlung bekannter und neuer Definitionen für die Form von Literaturlisten, selbstverständlich mit Originaldokumentation. Die vorliegende TeX-Implementierung versteht, wie sich das auf einem ATARI auch gehört, deutsche Umlaute, hier müssen also nicht irgendwelche Backslashes und Gänsefüßchen separat eingegeben werden, und einige auf dem ATARI verfügbare Sonderzeichen. Verwendet man diese in eigenen Texten, dann ist es zunächst einmal aus mit der Übertragbarkeit von Quelltexten auf andere Systeme. Hier hilft jedoch das ebenfalls mitgelieferte TRANSFER.TTP, welches in TeX-Quelldateien diese Sonderzeichen in die zugehörigen Standard-Befehlsfolgen von TeX umwandelt. Daneben gibt es noch Hilfsprogramme zum Einbinden von Graphiken, diese werden weiter unten noch ausführlicher beschrieben.

Das TeX-System alleine ist für einen normalen Anwender natürlich uninteressant. Erst eines der Makro-Pakete ermöglicht eine komfortable Nutzung dieses Systems. Mit Abstand am weitesten verbreitet ist das Makro-Paket LaTeX von Leslie Lamport [3]. Dieses lebt von der Auswahl der jeweils verfügbaren Styles, das sind Dateien, in denen verschiedene Layouts vordefiniert sind. Dabei wird nicht nur das Aussehen einer Seite festgelegt, sondern beispielsweise auch die Art, wie Kapitel, Unterabschnitte und Formeln automatisch numeriert werden, die Trennregeln und vieles mehr. Die mitgelieferte Styles-Sammlung zu LaTeX ist umfangreich, und zu jedem Style ist, soweit verfügbar, auch die Originaldokumentation (selbstverständlich als TeX-Quelltext auf Diskette!) dabei. Dies ist ungewöhnlich; normalerweise wird auf die einschlägige Literatur und auf Benutzervereinigungen verwiesen. Neben LaTeX bekommt man noch das AMS-Paket [4] von Michael Spivak, ebenfalls wohldokumentiert. Für die Puristen gibt es noch die Minimalausstattung PLAIN, wie sie im TeXbook [1] beschrieben ist.

Vorsicht, Falle!

Wer noch nicht allzuviel mit seiner Festplatte gearbeitet und auch noch nie von dem allseits beliebten 40-Ordner-Phäno-

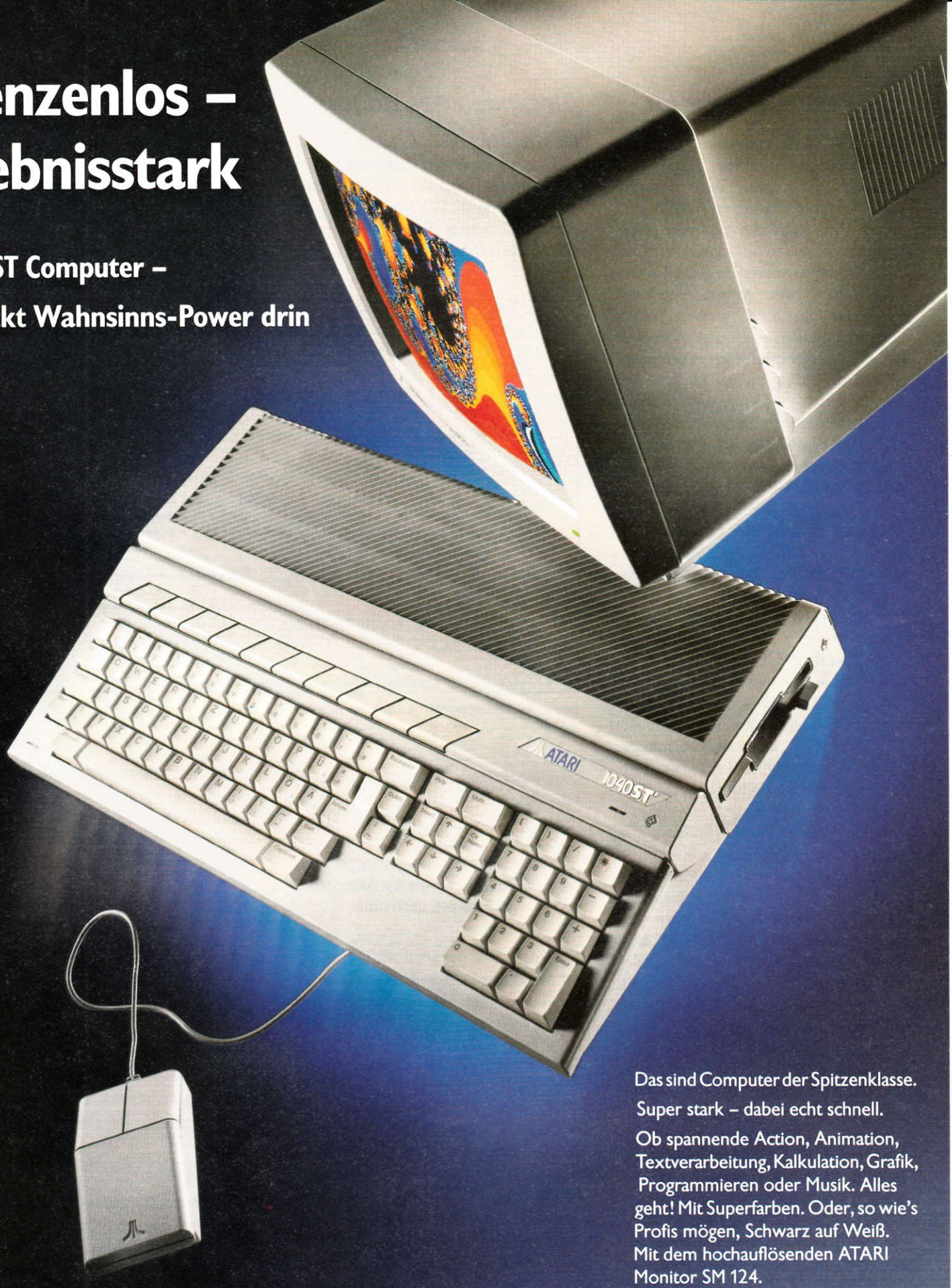
```
file=D:\NVR\
wait=ON
editor=C:\GEM\TEMPUS.PRG
editopt=
tex=E:\TEX.TTP
fmt=&lplaing
texopt=
TeXINPUTS=e:\latex;e:\initex
TeXFONTS=E:\tfm
TeXFORMATS=E:
initex=E:\BIN\INITEX.TTP
initexopt=-t50000 -s3000 -p36000
    -f19000
show=E:\BIN\DVIGRA.PRG
showopt=-X0 -Y-7
GRA_FONT=
DPF_FILES=
print=E:\BIN\DVIDOT.TTP
printopt=-y5mm
PRT_FONT=
bibtex=E:\BIN\BIBTEX.TTP
BIBINPUTS=E:\bibtex
F8=
F9=
command=
```

Bild 2: Beispiel einer Parameterdatei TEXMEN.DEF

men gehört hat, der wird nun wohl oder übel auf der Boot-Diskette zur Festplatte das Programm FOLDR100.PRG suchen müssen. Und das kommt so: Spätestens nach der Installation des TeX-Systems ist die im Betriebssystem vorgesehene maximale Anzahl von Ordnern, nämlich 40, mit Sicherheit überschritten. Was dann passiert, kann dem unbefangenen Benutzer oft den Eindruck vermitteln, daß es auf seinem Rechner spukt. Neuerdings heißt das wohl nicht mehr Spuk, sondern Virus. Seltsame Dinge haben wir da schon beobachtet. Im Desktop werden plötzlich nicht mehr alle Dateien angezeigt (nein, die nicht angezeigten Dateien sind nicht etwa verloren, oft erscheinen sie nach einem Reset sofort wieder); Dateien erscheinen zwar im Desktop, können aber partout nicht in der Auswahlbox beispielsweise eines Editors gefunden werden und vieles mehr. Beim TeX-System haben wir den Effekt erlebt, daß der Druckertreiber nach kurzer Anlaufzeit seinen Dienst unter Hinterlassung zweier niedlicher Bömbchen quittierte. Also: Nicht gleich auf TeX schimpfen, wenn solche sonderbaren Effekte auftreten, sondern FOLDR100.PRG starten, gegebenenfalls auch nach Umbenennen, wenn man den Puffer für die Ordner bereits erweitert hat. Das Handbuch zur Festplatte gibt hier genaue Auskunft.

Grenzenlos – Erlebnisstark

ATARI ST Computer –
da steckt Wahnsinns-Power drin



Das sind Computer der Spitzenklasse.
Super stark – dabei echt schnell.
Ob spannende Action, Animation, Textverarbeitung, Kalkulation, Grafik, Programmieren oder Musik. Alles geht! Mit Superfarben. Oder, so wie's Profis mögen, Schwarz auf Weiß. Mit dem hochauflösenden ATARI Monitor SM 124.

ATARI ST Computer –
Höchstleistung auf allen Gebieten.
2 x „Computer des Jahres".

Superprogramme Signum, Calamus, STAD, Lavadraw, PGRAPH, Imagic, Creator, CV Base, Twenty Four und viele andere gibts nur für ATARI ST Computer.

ATARI
...wir machen Spitzentechnologie preiswert.

Alles unter einem Hut

Bedingt durch das mehrstufige Arbeiten, also Quelltext eingeben, Text setzen, gesetzten Text ausgeben, hat der Anwender es schon mit mehreren Programmen zu tun. Hinzu kommen dann noch Hilfsprogramme, wie zum Beispiel BibTeX. Berücksichtigt man zudem, daß all diesen Programmen beim Aufruf auch noch Parameter mitgegeben werden können und im Normalfall auch mitgegeben werden müssen, dann gibt es schon eine ganze Menge zu tippen und zu klicken. Der übliche Ausweg aus einer solchen Situation ist natürlich eine Shell, und der wurde auch hier beschritten. Da wir auf dem ATARI sind, hatte es natürlich eine graphische, sprich GEM-Shell zu sein. Die erste Generation solcher Shells ist uns allen noch in schlechter Erinnerung: Es gab nur Abroll-Menüs, und für die simpelsten Funktionen waren umfangreiche Mauseleien erforderlich. Das andere Extrem sind Shells, die ausschließlich durch Tasten gesteuert werden. Glücklicherweise sind die Designer heute schon etwas schlauer, und auch die Leute von Detig Schrod TeXsys scheinen ihr Handwerk zu verstehen: Ihre Shell, sie heißt TEXMEN.PRG, kann sowohl mit der Maus als auch mit Funktionstasten gesteuert werden. Es sind sogar noch Funktionstasten frei, die man mit eigenen Programmen belegen kann. Die Zuordnung von Funktionstasten zu den damit aufrufbaren Programmen wird überdies auf dem Bildschirm äußerst gekonnt angezeigt, wie in Bild 1 schön zu sehen ist. In einer separaten Datei, TEXMEN.MDF, können Parameter definiert werden, die den Programmen beim Aufruf durch die Shell mitgegeben werden. Bild 2 zeigt ein Beispiel einer solchen Datei. Auf diesem Wege definiert sich das System selbst Umgebungsvariablen. Dies sind zum Beispiel Suchpfade für Eingabedateien und Zeichensätze, der Name des von TeX verwendeten Makro-Pakets oder auch der Name und Ort der aufzurufenden Programme. Beim Aufruf eines solchen Programms kann wahlweise auch der aktuelle Pfad umgesetzt werden, damit es etwa seine RSC-Datei findet. Diese Parameter kann man übrigens wahlweise mit einem Editor ändern oder in der Shell selbst über Abroll-Menüs verstellen. Selbstverständlich können Parameter in verschiedenen Dateien abgelegt und nach Belieben eingelesen werden. Überhaupt macht die ganze Shell einen sehr durchdachten und professionellen Eindruck.

Einbinden des Editors

Ein Editor ist beidiesem Paket natürlich nicht dabei, dafür werden beim Installieren Name und Pfad des Lieblingseditors abgefragt und in TEXMEN.MDF automatisch eingefügt. Diese Werte lassen sich selbstverständlich jederzeit, wie oben beschrieben, ändern. Sehr gute Erfahrungen haben wir mit Tempus2 gemacht, auch MicroEmacs ist für diesen Zweck gut geeignet. Für MicroEmacs gibt es übrigens von Anselm Lingnau eine spezielle Anpassung, die das Arbeiten mit LaTeX wirkungsvoll unterstützt.

Nach dem Start der Shell wird beim Aufruf des ersten Programms der Name der Datei abgefragt, die bearbeitet werden soll. Der dann eingegebene Name wird dann auch für die anderen Programme verwendet, wobei die Shell die jeweils benötigte Endung, wie 'TEX' oder 'DVI', selbst anpaßt. Natürlich kann ein einmal eingegebener Name, wieder über Abroll-Menü, nachträglich verändert werden. Besonders gelungen ist die Möglichkeit, für den Editor und TeX auf Wunsch unterschiedliche Namen angeben zu können. Wer viel mit '\include' und '\input' arbeitet, wird dies zu schätzen wissen.

Gefallen hat auch die Belegung der Funktionstaste F1, die zunächst den Editor, dann automatisch TeX und anschließend den Preview startet. So werden langwierige Entwicklungszyklen etwas erträglicher.

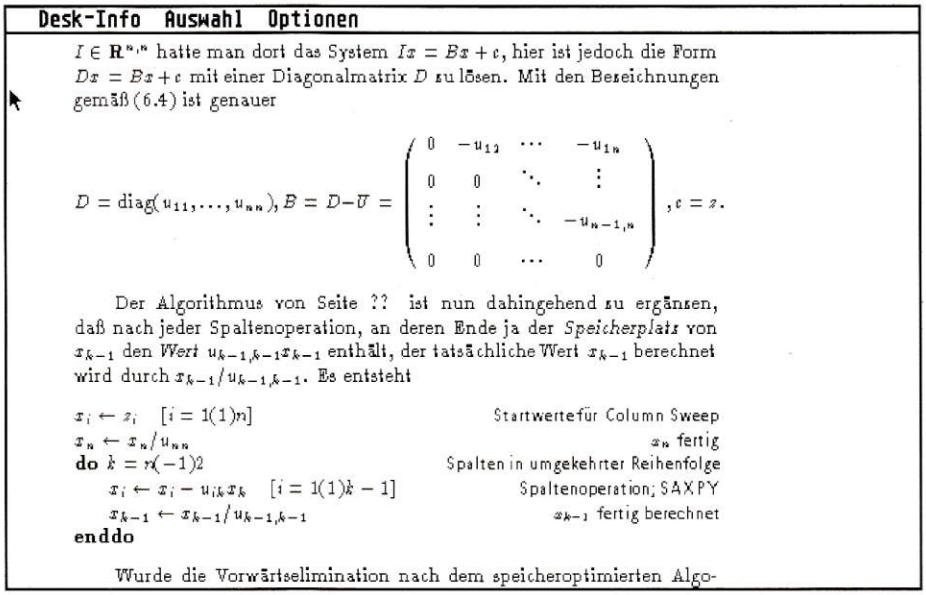


Bild 3: Bildschirmausgabe mit dem Preview in Originalgröße.

TeX

Das TeX trägt die Versionsbezeichnung 2.92, die LaTeX-Makros haben 2.09 und wurden im Mai 1988 zuletzt verändert. Die deutschen Anpassungen sind ebenfalls recht neu und scheinen den sich abzuzeichnenden Standard bereits zu enthalten. Ganz aktuell ist jedenfalls die umfangreiche Datei mit deutschen Trennmustern von Norbert Schwarz. Die ist übrigens, wie sämtliche Makros auch, Public Domain. Es ist allerdings eine besondere Stärke dieser Implementierung, auch die kostenfrei verfügbaren Teile des TeX-Systems zu enthalten und dafür nicht auf weitere Quellen zu verweisen. Im Menü kann eingestellt werden, daß der TeX-Bildschirm nach Beendigung des Programms so lange stehenbleibt, bis eine Taste gedrückt wird. So bekommt man einerseits auch noch alle Warnungen mit, andererseits wird dies auf die Dauer, besonders bei Verwendung der F1-Taste, doch recht lästig.

Preview

Das Preview-Programm DVIGRA.PRG ist eine besondere Stärke dieses Systems. Es ist einerseits so schnell, wie es auf dem ATARI eben geht, und da hat die MS-DOS-Welt gewaltig das Nachsehen, bietet andererseits aber auch noch eine ganze Menge Komfort. Hier ist ebenfalls wieder die Bedienung mit Tasten möglich, dies geht schneller, und man braucht sich nicht mit den Abroll-Menüs das Bild zu verdekken. Diese sind, etwa für Leute, die grund-

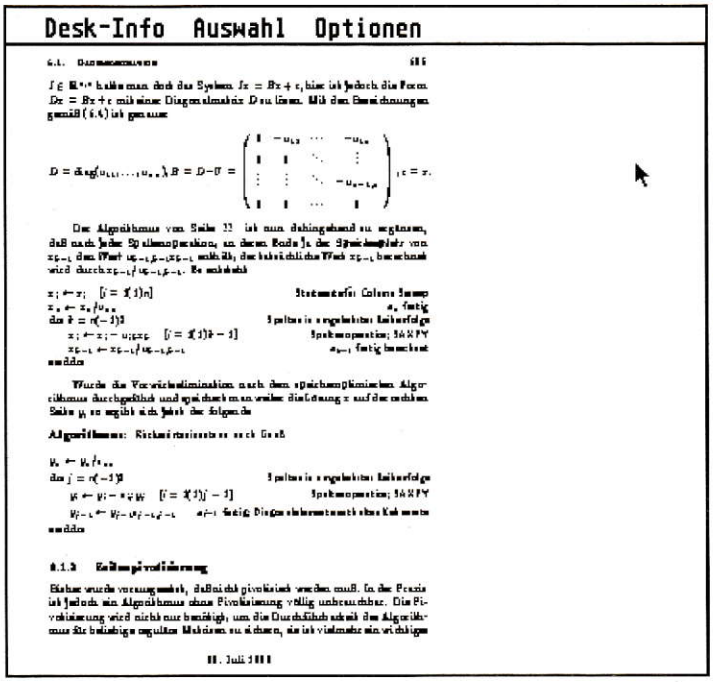


Bild 4: Die gleiche Seite wie in Bild 3 in vierfacher Verkleinerung

sätzlich keine Bedienungsanleitung lesen, noch zusätzlich vorhanden. Bild 3 zeigt eine solche Bildschirmausgabe mit Menüleiste.

So läßt sich etwa eine beliebige Seite, auch rückwärts, anspringen. Mit den Pfeiltasten wird der Bildschirmausschnitt bewegt, per Tastendruck kann eine Seite vor oder zurück geblättert werden. Weiter läßt sich per Mausklick der linke obere oder rechte untere Rand des anzuzeigenden Ausschnitts definieren.

Richtig begeistert waren wir von den Möglichkeiten des Verkleinerns: Auf Tastendruck kann die Ausgabe auf ein Viertel oder ein Neuntel verkleinert werden. Man kann dann zwar kaum noch etwas (bei einem Viertel) oder gar nichts mehr (bei einem Neuntel) entziffern, dafür paßt dann aber eine ganze Seite auf den Bildschirm und vermittelt so einen guten Eindruck vom Aufbau der Seite, wie in Bild 4 zu sehen ist.

Ist eine Seite erst einmal im Speicher aufbereitet, und das geht durch einen ausgefeilten Pufferungs- und Darstellungsalgorithmus recht flott, verlaufen alle graphischen Transformationen dieser Seite mit der vom GEM gewohnten Geschwindigkeit, also das Verschieben des angezeigten Ausschnitts und die eben beschriebenen Möglichkeiten des Verkleinerns, sowie deren Umkehr. Lediglich bei der expliziten Neudefinition der Vergrößerungsstufe müssen die Zeichensätze neu geladen werden. Die Behandlung fehlender Zeichensätze wird weiter unten beschrieben.

Was man schwarz auf weiß besitzt...

Die Druckertreiber haben sich in punkto Geschwindigkeit wohl am Preview orientiert. Bei den 9-Nadlern mit ihrem winzigen, wenn überhaupt vorhandenen, Pufferspeicher ist da natürlich nicht allzuviel herauszuholen. Bild 5 vermittelt einen Eindruck von der mit einem 9-Nadler erreichbaren Qualität. Zwischen 10 und 15 Minuten muß man in der Regel schon auf eine Seite warten. Bei einem Epson LQ geht die Sache schon etwas flotter, hier wird allerdings auch nur mit 180 mal 180 dpi gearbeitet. Der Treiber für den Laserdrucker schlägt jedoch alles, was wir bisher gesehen haben. Durch ein intelligentes Laden der Zeichen in den Druckerspeicher sind im Normalfall durchaus 5 bis 6 Seiten pro Minute möglich. Nur bei der Ausgabe von Bitmap-Graphiken geht das Programm natürlich deutlich in die Knie, hier läßt es sich eben nicht so gut mit programmierbaren Zeichensätzen tricksen. Getestet haben wir den Treiber mit einem original HP Laserjet Serie II und einem Sharp. Bild 6 zeigt einen solchen Ausdruck in einwandfreier Qualität.

Graphik

Als besonderer Knüller ermöglicht es diese TeX-Implementierung, Graphiken in den Text einzubinden. Damit ist nun nicht etwa die Möglichkeit gemeint, mit Hilfe der Picture-Umgebung einfachere Bilder zu erzeugen, die man in LaTex ja immer hat; vielmehr kann man die zuvor mit einem der üblichen Zeichenprogramme erstellten Bitmuster-Graphiken verwenden. Voraussetzung ist allerdings, daß das Zeichenprogramm Bilder im Degas-Format abspeichern kann. Dies dürfte aber wohl von jedem halbwegs anständigen Programm zu erwarten sein; eines der geeignetsten ist sicher STAD, und hier kann man selbstverständlich das Speicherformat wählen.

Ein im Degas-Format abgespeichertes Bild wird nun in einem separaten Arbeitsgang durch ein im Lieferumfang enthaltenes Programm in ein spezielles TeX-Format umgewandelt. Dabei wird die Auflösung des gewünschten Zielgerätes angegeben, und man kann gleich noch einige einfache Transformationen vornehmen, wie etwa die Verdoppelung der Pixel in horizontaler, vertikaler oder beide Richtungen. Durch einen einfachen TeX-Befehl, der als Parameter lediglich den Namen der Datei sowie die Höhe und Breite des für das Bild vorgesehenen Rahmens mitbekommt, wird eine solche Graphik nun in den Text eingefügt. Das Ausgabegerät gibt dann das Bild Pixel für Pixel in der dem Gerät eigenen Pixeldichte aus. Dies führt natürlich dazu, daß ein und dasselbe Bild auf verschiedenen Ausgabemedien unterschiedlich groß wird. Ein beispielsweise für den Laserdrucker aufbereitetes Bild kann nun wegen der unterschiedlichen Auflösung eben nicht auf dem Bildschirm angezeigt werden, in diesem Fall sieht man an Stelle des Bildes einfach einen leeren Rahmen und bekommt so wenigstens einen Eindruck von der Seitenaufteilung. Obwohl bei dieser Gelegenheit im Grunde von dem strengen Prinzip der Geräteunabhängigkeit der von TeX erzeugten Ausgabe abgewichen wird, ist diese Möglichkeit natürlich ein gewaltiger Fortschritt und eben immer noch viel besser als gar keine Graphikausgabe.

Literaturlisten...

...verarbeitet man unter LaTeX am besten mit Hilfe von BibTeX. Dieses Zusatzprogramm übernimmt die von LaTeX aufbereiteten Literaturzitate des Textes, sucht aus einer oder mehreren Literaturdateien die zugehörigen Verweise heraus und erstellt daraus eine Literaturliste, deren Form ebenfalls vorgegeben werden kann. BibTeX wird zu dem gerade bearbeiteten Text ebenfalls aus der Shell über eine Funktionstaste oder ein Abroll-Menü aufgerufen. Die Literaturdateien selbst werden wohl am besten mit einem Datenbankprogramm erzeugt, wir haben hier mit Adimens beste Erfahrungen gemacht.

INITeX

Die bei TeX unvermeidlichen Makro-Pakete können recht umfangreich werden und entsprechend viel Zeit beim Pro-

R. Schuster Computer

Computer-Hard- und Software

Atari ST Software

Auszug aus unserem Lieferprogramm

1943	51.90
A.P.B.	53.90
African Raiders 89	53.90
Afterburner	59.90
Altered Beast	58.90
Amazon	49.90
American Icehockey	69.90
Archipelagos	73.90
Arkanoid 2	54.90
Atax	41.90
Baal	34.90
Balance of Power	82.90
Balance of Power 1990	76.90
Ballistix	54.90
Bard's Tale 1	77.90
Batman	58.90
Battlehawks 1942	59.90
Battletech	77.90
Beam Volley	62.90
Beam	58.90
Bio Challenge	65.90
Bismarck	73.90
Blasteroids	58.90
Buffalo Bills W.W. Rodeo Games	76.90
Buggy Boy	57.90
California Games	54.90
Captain Blood	28.90
Carrier Command	73.90
Castle Warrior	69.90
Chariots of Wrath	77.90
Chicago 30	54.90
Chronoquest	69.90
Circus Attractions	57.90
Conflict Europe	77.90
Corruption	72.90
Cosmic Pirate	58.90
Crazy Cars	54.90
Crazy Cars 2	54.90
Custodian	58.90
Cybernoid 2	58.90
D. T. Olympic Challenge	58.90
Dakar 89	43.90
Darius	53.90
Dark Castle	66.90
Das Reich	57.90
Demons Winter	69.90
Dominator	58.90
Double Dragon	43.90
Dragon Ninja	58.90
Dragonscape	54.90
Dschungelbuch	43.90
Dungeon Master	76.90
Dungeonmaster Editor	29.90
Elemental	26.90
Eliminator	58.90
Elite	73.90
Emmanuelle	43.90
Epyx (The Worlds Greatest)	72.90
Espionage	58.90
Expansion Kit für Football Manager 2	39.90
Eye	21.90
F-16 Combat Pilot	69.90
F-16 Falcon	76.90
F-16 Falcon Mission Disk	63.90
F.O.F.T.	89.90
Final Assault	51.90
Firezone	77.90
Fish	77.90
Flight S. Disc 7 Florida	42.90
Flight S. Disc 9	42.90
Flight S. Disc 11 Michigan	42.90
Flight S. Disc Japan	42.90
Flight S. Western European	42.90
Flight Sim. 2	112.90
Football Manager 2	57.90
Forgotten Worlds	54.90
Fugger	53.90
Galactic Conqueror	54.90
Galdregons Domain	54.90
Gary Linekers Hotshots	59.90
Genius	53.90
Giants Compilation	77.90
Goldrush	76.90
Heroes of the Lance	69.90
Hostages	65.90
I Ludicrus	26.90
Incredible Shrinking Sphere	58.90
Indiana Jones T. Last Cruscade	57.90
International Karate Plus	58.90
Iron Tracker	53.90
Its a Kind of Magic	72.90
Jagd auf Roter Oktober	77.90
Jaws	54.90
Jeanne D'Arc	51.90
Jet	99.90
Jug	54.90
Kaiser	121.90
Kampf um die Krone	61.90
Kennedy Approach	73.90
Kick Off	45.90
Kings Quest 3er Pack	76.90
Kings Quest 4	89.90
Kult	58.90
Lancelot	51.90
Leaderboard Par 3	66.90
Leben und sterben lassen	43.90
Led Strom	51.90
Legend of Djel	53.90
Legend of the Sword	72.90
Leisure Suit Larry	58.90
Leisure Suit Larry 2	87.90
Leonardo	53.90
Lizenz zum Töten	51.90
Lombard Rac Ralley	73.90
Manhunter Ny	87.90
Manhunter San Fran.	91.90
Marble Madness	77.90
Mega Pack	77.90
Menace	51.90
Micky Mouse	54.90
Microprose Soccer	73.90
Millenium 2.2	76.90
Motor Massacre	58.90
Munsters	54.90
Murder in Venice	69.90
New Zealand Story	58.90
Night Raider	58.90
Off Shore Warrior	51.90
Oil Imperium	59.90
Operation Neptun	43.90
Out Run	28.90
Pacland	58.90
Pacmania	57.90
Personal Nightmare	89.90
Peter Pan	43.90
Phobia	62.90
Pink Panther	54.90
Pirates	72.90
Police Quest	58.90
Police Quest 2	76.90
Populous	73.90
Powerdrome	77.90
Precious Metal	62.90
President is Missing	69.90
Psion Chess	73.90
Purple Saturn Day	43.90
Quest for Time Bird	72.90
Questron 2	69.90
R-Type	58.90
Racing	57.90
Red Heat	63.90
Red Lightning	87.90
Return to Genesis	57.90
Rick Dangerous	72.90
Ringside	43.90
Roadblasters	54.90
Roadwar	57.90
Rockford	54.90
Run the Gauntlet	58.90
Running Man	73.90
Rückkehr der Jedi Ritter	43.90
RVF Honda	72.90

DMP 2160
Epson-kompatible 160 Z/Sec. nur **298.-**

Sargon 3	69.90
Savage	73.90
Shinobi	57.90
Shuttle 2	61.90
Silkworm	57.90
Skweek	51.90
Sky Chase	58.90
Skyrider	54.90
Slaygon Adventure	51.90
Sleeping Gods Lie	72.90
Sommer Olympiade 88	58.90
Space Quest 1	58.90
Space Quest 2	58.90
Space Quest 3	87.90
Spherical	59.90
Spitting Image	53.90
Stormtrooper	54.90
Stos Compiler	54.90
Stos Maestro	69.90
Stos Sprites	41.90
Stos the Game Creator	79.90
Super Hang On	58.90
Super Quintett	61.90
Tank Attack (CDS)	73.90
Technocop	58.90
Teenage Queen	43.90
Test Drive	77.90
The Deep	59.90
The Real Ghostbusters	58.90
Thunderbirds	76.90
Thunderblade	51.90
Tiger Road	54.90
Timescanner	58.90
Titan	54.90
Tom und Jerry	58.90
Tracksuit Manager	54.90
Trash Heap	54.90
Trivial Persuit	43.90
Turbo Cup	57.90
Ultima 4	69.90
Universal Military Scenery 1	39.90
Universal Military Simulator	73.90
Vindicators	54.90
Volleyball Simulator	51.90
Wall Street Wizzard	61.90
Wanted	57.90
War Hawk	28.90
War in Middle Earth	58.90
Waterloo	76.90
Wec Le Mans	58.90
Where time stood still	58.90
Whirligig	58.90
Wicked	58.90
Winter Edition	51.90
Winter Olympiad 88	58.90
Xybots	53.90
Yuppies Revenge	63.90
Zak McKracken	73.90

Hardware auf Anfrage.
Irrtümer und Preisänderungen vorbehalten. Bestellungen schriftlich oder telefonisch.

Reinhard Schuster Computer
Obere Münsterstr. 33-35 · Tel. 0 23 05/37 70 · 4620 Castrop-Rauxel

Bei allen Bestellungen unbedingt Computertyp angeben.
Geschäftszeiten: Montag – Freitag 9.00-13.00 und 14.00-18.00 Uhr, Samstag 9.00-13.00 Uhr.
Versand nur per NN zuzügl. 8,– DM Versandkosten oder Vorkasse auf Postgiro-Kto.-Nr. 69422-460 Postgiroamt Dortmund zuzüglich 6,00 DM. Ausland nur per Vorkasse zuzügl. 12,00 DM.
Neueste kplt. Softwareliste bei jeder Bestellung kostenlos oder gegen frankierten Rückumschlag.

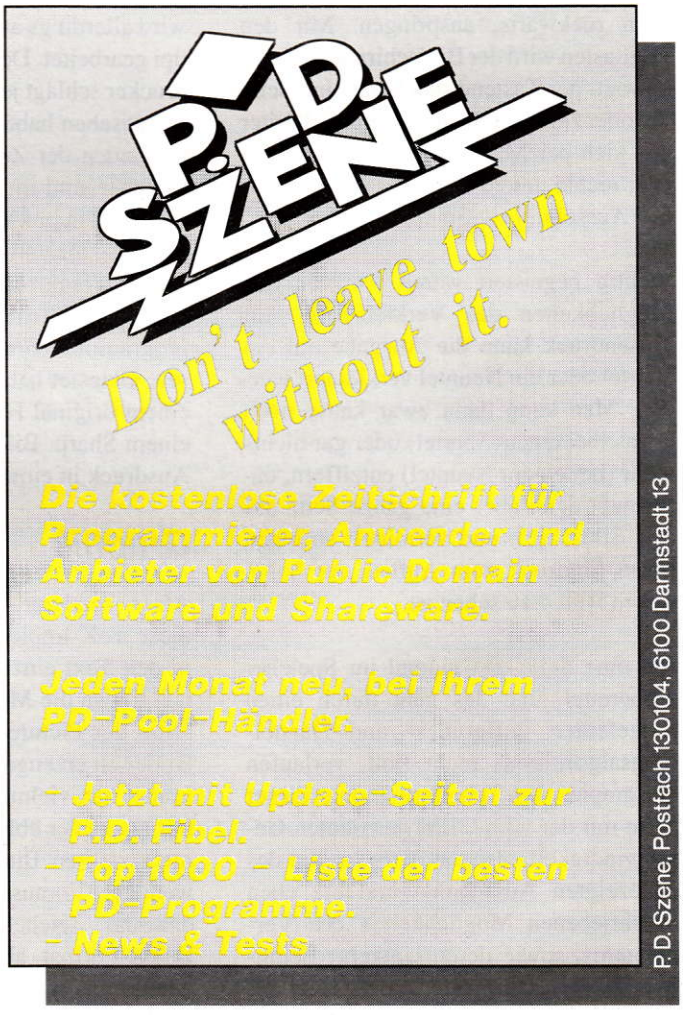
☆ ☆ ☆ ATARI ST ☆ ☆ ☆

Atari Mega ST 1, SM 124, Maus	1698.--
Atari Mega ST 2, SM 124, Maus	2298,--
Atari Mega ST 4, SM 124, Maus	3398,--
Atari Megafile 30 MB Festplatte	898,--
Vortex HDplus 30 MB Festplatte	1098,--
Vortex HDplus 40 MB Festplatte	1298,--
Vortex HDplus 60 MB Festplatte	1598,--
Star LC 10 9 Nadel Drucker	478,--
Star LC 24-10 24 Nadel Drucker	748,--
NEC P2200 24 Nadel Drucker	848,--
NEC P6 plus 24 Nadel Drucker	1398,--

Archipelagos (Deutsch)	79,--	**Kult** (Deutsch)	62,--
Bard's Tale (Deutsch)	79,--	**Leisure Suit Larry II**	93,--
Bolo (Deutsch)	62,--	**Microprose Soccer** (Deutsch)	76,--
California Games (Deutsch)	56,--	**Pirates** (Deutsch)	79,--
Chrono Quest (Deutsch)	79,--	**Police Quest II**	84,--
Corruption (Deutsch)	74,--	**Populous** (Deutsch)	79,--
Dungeon Master (Deutsch)	79,--	**Populous Lands** (Deutsch)	45,--
Elite (Deutsch)	79,--	**RVF Honda** (Deutsch)	79,--
F 16 Falcon (Deutsch)	79,--	**Shadowgate**	72,--
F 16 Falcon Mission Disk (Dtsch.)	65,--	**Space Quest III**	93,--
Flight Simulator II (Deutsch)	109,--	**Starglider II** (Deutsch)	76,--
Jeanne d'Arc (Deutsch)	54,--	**Star Trek** (Deutsch)	59,--
Kaiser (Deutsch)	109,--	**Sundog** (Deutsch)	49,--
Kings Quest IV	93,--	**Wall Street Wizard** (Deutsch)	64,--

☞ **Kostenlose Preisliste gegen 1,-- DM Rückporto anfordern!** ☜

Computer & Zubehör Versand G. und B. Waller GbR
Kieler Straße 623 · 2000 Hamburg 54
☎ 040/570 60 07 · Fax 040/570 29 92 · BTX 040 570 52 75



grammstart in Anspruch nehmen. Dies wird gewöhnlich dadurch gelindert, daß eine gewisse Grundmenge von Makros zunächst geladen und bearbeitet und anschließend in komprimierter Form abgespeichert wird, die danach beim erneuten Start schneller geladen werden kann. So entstehen die bekannten Formatdateien, auf dem ATARI durch den Typ FMT gekennzeichnet. Hierzu ist eine spezielle Version von TeX notwendig, die für den ATARI unter dem Namen INITeX mitgeliefert wird. Wegen des enormen Platzbedarfs ist die Generierung von Formatdateien auf einem ATARI mit nur einem Megabyte etwas schwierig, man muß mit den Größen für die einzelnen TeX-Speicherbereiche ein wenig herumrechnen. Auf jeden Fall wird jedoch das Arbeiten mit INITeX ebenfalls von der Shell, auch mit einer Funktionstaste, unterstützt; das Handbuch erläutert ausführlich die Vorgehensweise bei einem Megabyte Speicher. Somit wird der Benutzer auch bei INITeX sicher geführt. Die meisten Anwender werden sicherlich mit LaTeX deutsche Texte bearbeiten wollen. Im Hinblick darauf ist die zugehörige Makro-Datei bereits vorgefertigt und nach der Installation unter dem Namen LPLAIN.FMT vorhanden. Viele Anwender werden dadurch mit INITeX erst gar nicht in Berührung kommen.

Reduktion des Platzbedarfs

Die hier besprochene Implementierung ist einmalig in ihrer Vollständigkeit. Dies betrifft sowohl die Anzahl der mitgelieferten Zeichensätze als auch die Makro-Pakete und deren Dokumentation. Ein einzelner Anwender wird nun in der Regel nicht all dieses nutzen (es sei denn, er arbeitet als Tester für eine von Ihnen allen hochgeschätzte ST-Zeitschrift...) und daher schnell auf den Gedanken kommen, die Festplatte von unnötigem Ballast zu befreien. Wird das System jedoch von mehreren Anwendern genutzt, etwa in einem Sekretariat, so ist von einer solchen Löschaktion dringend abzuraten. Nach Murphy werden genau der Zeichensatz und der Style benötigt, den man gestern, nachdem bisher noch nie jemand darauf zugegriffen hatte, in das Write-Only-Memory unbegrenzter Kapazität, das man gewöhnlich links unten auf dem Desktop findet, verschoben hat.

Ein einzelner Benutzer sollte seinen Bedarf jedoch ganz gut abschätzen können; berücksichtigt er dann noch die Anforderungen von Dokumenten, die er von anderen übernimmt (Anleitungen!!!), dann kann er zur Tat schreiten. Zunächst werden nicht benötigte Makropakete entfernt. Danach druckt man die relevante Dokumentation und löscht die zugehörigen Dateien ebenfalls. Manch einer bearbeitet sowieso nur mit LaTeX deutsche Texte und verfügt auch nicht über die Sachkenntnis, um an den vorgefertigten Makropaketen etwas ändern zu können. Dann braucht er aber auch nicht INITeX: Weg damit! Die so erzielbare Ersparnis an Plattenplatz ist jedoch noch recht gering im Vergleich zu dem, was im nächsten Schritt anfällt.

Jetzt geht's erst richtig los

Hierzu ist etwas mehr Aufwand erforderlich. In längerem Probebetrieb mit typischen Dokumenten ist zunächst festzustellen, welche Zeichensätze in welchen Vergrößerungsstufen tatsächlich benötigt werden. Danach wird man zur Sicherheit alle Zeichensätze der Grundstufe des jeweiligen Geräts behalten, in den Vergrößerungsstufen jedoch alle nicht benötigten Zeichensätze löschen. Da die vorliegende TeX-Implementierung eine für Standardanwendungen vollständige Sammlung von Zeichensätzen beinhaltet, ergibt sich auf diese Weise eine drastische Einsparung. Im Alltagsbetrieb mit einem Star NL-10 konnte unser Testsystem einschließlich aller Zeichensätze für Bildschirm und Drucker auf ganze 2.5 Megabyte reduziert werden.

Das Löschen von Dateien ist im übrigen völlig gefahrlos: Sie können ja bei Bedarf jederzeit von den Originaldisketten nachgeladen werden. Da die archivierten Dateien mit dem weit verbreiteten ARC-System erstellt sind, kann man mit diesem PD-Programm (ist z.B. auf vielen PD-Disketten von MAXON als Zubehör dabei) deren Inhaltsverzeichnis ansehen und einzelne oder alle Dateien herausholen. Zur Not (Was? Sie kennen den PD-Versand der ST-Computer noch nicht?) geht das Auspacken auch mit dem beim TeX-System mitgelieferten Hilfspro-

Wie man leicht erkennt, sind die Spaltenoperationen bei der Berechnung der neuen Koeffizienten und rechten Seiten unabhängig voneinander und können daher parallel ausgeführt werden. Dies wird wieder durch den Zusatz parallel vor dem betreffenden do angezeigt. Im Hinblick auf die spätere Verwendung ist es noch günstig, jeweils im Hauptdiagonalelement a_{jj} an Stelle des Pivotelements $a_{jj}^{(j-1)}$ dessen Kehrwert r zu speichern. Insgesamt ergibt sich so der folgende speicheroptimierte Algorithmus.

Algorithmus: Vorwärtselimination nach Gauß

```
do j = 1(1)n - 1                                   Eliminationsstufen
   a_jj ← 1/a_jj
   a_ij ← a_jj a_ij   [i = j + 1(1)n]              Multiplikatoren ℓ_ij
   parallel do k = j + 1(1)n                       Restliche Spalten
      a_ik ← a_ik - a_jk a_ij   [i = j + 1(1)n]    k-te Spalte; SAXPY
   enddo
   parallel do k = 1(1)m                           Alle rechten Seiten
      y_ik ← y_ik - y_jk a_ij   [i = j + 1(1)n]    k-te rechte Seite; SAXPY
   enddo
enddo
```

6.1.2 Rückwärtseinsetzen

Grundlage hierfür ist der Column-Sweep-Algorithmus aus Abschnitt 5.1. Mit einer strikten oberen Dreiecksmatrix $B = (b_{ik})$ und der Einheitsmatrix

Bild 5: Probeseite mit dem STAR NL-10. Zum Korrekturlesen und für den persönlichen Gebrauch ausreichende Qualität.

gramm EXTRACT.TTP, dessen Handhabung das Handbuch selbstverständlich beschreibt. Benötigte Zeichensätze lassen sich natürlich ebenso nachladen, und hier gibt es ein ganz besonderes Bonbon: Wird bei der Ausgabe ein Zeichensatz in der benötigten Vergrößerungsstufe nicht gefunden, wird er automatisch durch den in der Grundstufe ersetzt. Dieses wird dann in einer Datei mitprotokolliert, auf die nach Beendigung der Ausgabe in einem solchen Fall besonders hingewiesen wird. So sieht man wenigstens den gewünschten Text, wenn auch einige Zeichen nicht in der verlangten Größe erscheinen, und man weiß genau, was gefehlt hat und nachzuladen ist.

Die Grundstufe steht für die Bildschirmausgabe übrigens im Ordner PXL0500, für den Laserdrucker in PXL1500 und für die 9-Nadler in PXL1200. Alle diese Angaben sind in der Dokumentation schnell und zuverlässig zu finden.

Dem Chaos keine Chance

Das gesamte TeX-System haben wir unter KAOS getestet. Dies ist eine korrigierte Version des aktuellen Betriebssystems, die neben wenigen Erweiterungen, wie etwa einer verbesserten Auswahlbox, lediglich einige Korrekturen der schlimmsten Fehler enthält und nicht zu verwechseln ist mit dem von ATARI an ausgewählte Kunden bereits gelieferten TOS 1.4. So ist es etwa unter KAOS nicht mehr möglich, eine Datei zum Lesen zu eröffnen und diese dann zu beschreiben. Dabei stellt es sich heraus, daß sogar professionelle und weit verbreitete Standardprodukte solche Schlampereien enthalten, die auf Grund des fehlerhaften Betriebssystems bisher nur nicht aufgefallen sind. Als Beispiele solcher Programme, die dann erst nach einem Patch wieder laufen, seien Adimens, Datamat und Beckertext genannt. Die Module von Adimens brechen gleich nach dem Laden ab, bei Datamat verschwindet der Cursor, und Beckertext druckt nicht mehr. Diese Liste läßt sich natürlich beliebig verlängern. Somit ist KAOS also ein harter Test für die Sorgfalt, mit der ein Programm implementiert wurde. Diesen Test hat STTeX mit Bravour bestanden. Während des gesamten Testbetriebs waren keine Unverträglichkeiten festzustellen.

6.1.2 Rückwärtseinsetzen

Grundlage hierfür ist der Column–Sweep–Algorithmus aus Abschnitt 5.1. Mit einer strikten oberen Dreiecksmatrix $B = (b_{ik})$ und der Einheitsmatrix $I \in \mathbf{R}^{n,n}$ hatte man dort das System $Ix = Bx + c$, hier ist jedoch die Form $Dx = Bx + c$ mit einer Diagonalmatrix D zu lösen. Mit den Bezeichnungen gemäß (6.3) ist genauer

$$D = \mathrm{diag}(u_{11}, \ldots, u_{nn}), B = D - U = \begin{pmatrix} 0 & -u_{12} & \cdots & -u_{1n} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \ddots & -u_{n-1,n} \\ 0 & 0 & \cdots & 0 \end{pmatrix}, c = z.$$

Der Algorithmus von Seite ?? ist nun dahingehend zu ergänzen, daß nach jeder Spaltenoperation, an deren Ende ja der *Speicherplatz* von x_{k-1} den *Wert* $u_{k-1,k-1}x_{k-1}$ enthält, der tatsächliche Wert x_{k-1} berechnet wird durch $x_{k-1}/u_{k-1,k-1}$. Es entsteht

```
x_i ← z_i   [i = 1(1)n]          Startwerte für Column Sweep
x_n ← x_n/u_nn                   x_n fertig
do k = n(−1)2                    Spalten in umgekehrter Reihenfolge
    x_i ← x_i − u_ik x_k  [i = 1(1)k − 1]   Spaltenoperation; SAXPY
    x_{k−1} ← x_{k−1}/u_{k−1,k−1}           x_{k−1} fertig berechnet
enddo
```

Bild 6: Probeseite mit dem HP-Laserjet. Einwandfreie Qualität.

Blow Up

Ein besonderes Schmankerl gibt es vom Preview noch zu berichten: Dieses Programm läuft nicht nur auf dem Originalmonitor SM124, sondern auch auf großen hochauflösenden Monitoren, die eine GEM-Einbindung haben. So wird zum Beispiel an der TH Darmstadt der 17"-Bildschirm von Matrix eingesetzt.

Die Wertung, bitte!

STTeX ist eine umfangreiche und in bezug auf LaTeX und dessen Zeichensätze weitgehend vollständige Implementierung auf dem ST. Noch nicht vollständig verfügbar ist zum Beispiel SliTeX, mit dem man bequem Folien erzeugen kann, aber das bekommt man für den ST auch nirgendwo sonst. AMS-TeX ist ebenfalls im Lieferumfang enthalten, eine andere Bezugsquelle hierfür ist uns nicht bekannt. Die Zeichensätze der verschiedenen Vergrößerungsstufen reichen für alle Standardanwendungen aus. Die Dokumentation ist umfangreich und auch gut gemacht. Im Grundpaket ist standardmäßig das LaTeX-Buch von Lamport [3] enthalten, man kann aber auch darauf verzichten und bekommt dafür 57 Mark erstattet.Die Installation ist menügesteuert und daher auch von Laien zu erledigen. Man sollte allerdings schon wissen, was eine Partition ist und wie der Editor heißt. Das System läuft stabil und muß nicht mit irgendwelchen Tricks erst überredet werden, seine Pflicht zu tun. Ist man erst einmal mit TeX vertraut, verläuft das Arbeiten problemloser als mit solchen "anwenderfreundlichen" Texteditoren wie 1st Wordplus oder Beckertext.

Berücksichtigt man den Lieferumfang mit den vielen Zeichensätzen, BibTeX und INITeX, was woanders ja noch zusätzlich kostet, und die bis ins Detail weitergedachte und ausgefeilte Implementierung, dann ist der Preis selbst für ATARI-Verhältnisse noch recht günstig. Zudem gibt es Sonderkonditionen für Universitäten und Campus-Lizenzen. Man erhält also eine arbeitsfähige und vollständige Version, die auch die verfügbaren Public Domain-Teile enthält, aufeinander abgestimmt und preiswert aus einer Hand.

Einsatzfeld

Wegen der Robustheit des Systems, der automatischen Installation, dem umfangreichen Satz von Makro-Paketen mit zugehöriger Beschreibung, dem vollstän-

digen System von Zeichensätzen in allen sinnvollen Vergrößerungsstufen und vor allem der einfachen Bedienung durch die graphische Benutzeroberfläche (Shell), in der die wichtigsten Parameter vordefinierbar sind, kann das System auch von jemandem installiert und betrieben werden, der kein ausgesprochener TeXniker und auch kein Betriebssystem-Profi ist. Grundkenntnisse im Umgang mit TeX und dem ATARI genügen. Das hier betrachtete System ist bereits mehrfach in Sekretariaten des Fachbereichs Mathematik der Johann Wolfgang Goethe-Universität, Frankfurt, installiert. Dabei hat es sich in der Praxis gezeigt, daß mit einem bereits installierten System auch jemand gut zurechtkommt, der sich erst in TeX einarbeitet. Besonders die Shell wird hier als große Hilfe und deutliche Verbesserung gegenüber anderen Implementierungen empfunden. Insgesamt handelt es sich hierbei um ein benutzerfreundliches System, das sich nicht nur auf dem Schreibtisch eines Testers, sondern auch in längerem Alltagseinsatz in der Praxis professionellen Schreibbetriebs, insbesondere in Verbindung mit [2], bestens bewährt hat.

With a little help...

Gerade der professionelle Anwender wird bald nicht mehr mit den aus der gängigen Literatur verfügbaren Informationen auskommen können und darüber hinaus auch an der Weiterentwicklung des Systems, insbesondere in den Bereichen Graphik und Eindeutschung, interessiert sein. Schon im TeXbook [1] wird auf die amerikanische Benutzervereinigung 'TUG' und deren Mitgliederzeitschrift TUGboat verwiesen. Seit einiger Zeit gibt es auch eine deutsche Vereinigung, der bereits der Status eines gemeinnützigen Vereins zuerkannt wurde.

Dr. Volker Kurz

Kontaktadresse TUG:

DANTE
Deutschsprachige Anwendervereinigung TeX e.V.
Rechenzentrum der Universität Heidelberg
z.Hd. Herrn J. Lammarsch
Im Neuenheimer Feld 293
6900 Heidelberg 1

Literatur:

[1] Knuth, D. E.: "The TeXbook", Addison Wesley, Reading-..., 1986.

[2] Kopka, H.: "LaTeX - Eine Einführung", Addison-Wesley, Bonn-..., 1988

[3] Lamport, L.: "LaTeX: A Document Preparation System", Addison Wesley, Reading-..., 1986.

[4] Spivak, M.: "The Joy of TeX. A Gourmet Guide to Typesetting with the AMS-TeX Macro Package", American Mathematical Society, Providence, 1986

[5] Wonneberger, R.: "Kompaktführer LaTeX", Addison-Wesley, Bonn-..., 1987

©

Bezugsadresse:

Firma Detig Schrod TeXsys
Kranichweg 1
6074 Rödermark-Urberach
(06074) 1617

STTeX (TeX-Grundsystem mit Shell, INITeX, BibTeX, den Makro-Paketen Plain, AMS-TeX und LaTeX, dem Preview DVIGRA mit Zeichensätzen und dem Buch [3] von Lamport): 305 Mark.

DVIDOT (Druckertreiber für Nadeldrucker mit Zeichensätzen, wahlweise für NEC P5/6/7, EPSON LQ oder 9-Nadler wie Epson FX-80, Star NL-10 und Kompatible oder Fujitsu DPL 24): 298 Mark.

DVIHP (Druckertreiber für HP Laserjet und Kompatible, mit Zeichensätzen): 378 Mark.

STTeX mit DVIDOT: 535 Mark.

STTeX mit DVIHP: 615 Mark.

STMF (Metafont mit INIMF): 198 Mark.

Voraussichtlich ab Oktober verfügbar: SliTeX-Fonts für Preview und Drucker: jeweils 98 Mark (dies ist eine Bearbeitungsgebühr und kein Kaufpreis: Die Zeichensätze dürfen beliebig weitergegeben werden).

Das Buch von Lamport kann bei STTeX auf Wunsch aber auch durch ein anderes der oben genannten Bücher ersetzt werden, oder es kann entfallen. Die Preise verringern sich dann jeweils um 57 Mark.

Verlangen Sie mehr!

Erscheint in Kürze

Dieter und Jürgen Geiß
Vom Anfänger zum Gem-Profi
Perfekte Programmierung auf Atari ST und IBM-PC
1989, ca. 300 S., kart., ca. DM 60,– incl. Diskette
ISBN 3-7785-1792-9
Dieses Buch beschreibt vollständig die Softwareentwicklung unter GEM inclusive Theorie der Benutzeroberflächen. Es entstand in Zusammenarbeit mit der Atari und Digital Research.



Hajo Lemcke, Volker Dittmar und Michael Sommer
Programmierlexikon für den Atari ST
2., neubearb. Aufl. 1989, 451 S., kart., DM 54,–
ISBN 3-7785-1671-X
Sie finden alles über GEM, VDI, AES, Chips, Schnittstellen, BIOS, XBIOS, GEMDOS, Systemvariablen und die Line-A Graphikbefehle. Jetzt mit Blitter-TOS und vielen neuen Erkenntnissen.

Christiane und Jürgen Kehrel
Omikron-BASIC
Befehle, Bibliotheken, Utilities
1989, ca. 400 S., kart., ca. DM 54,–
ISBN 3-7785-1662-0
Die umfassende Dokumentation zum Omikron-BASIC Interpreter und Compiler sowie zu den Bibliotheken (incl. MIDI und Statistik) und Hilfsprogrammen.

BESTELLCOUPON
einsenden an: Hüthig Buch Verlag GmbH
Im Weiher 10, 6900 Heidelberg

Titel
Name, Vorname
Straße, Nr.
PLZ, Ort
Datum, Unterschrift

Hüthig

OhstSoftware
Jutta Ohst
Nelkenstr. 2
4053 Jüchen 2

Public Domain je Markendisk ab: 5,50 DM

P.D.-Katalog → Über 90 Seiten gebunden. Nicht Quantität, sondern Qualität zeichnet die ausführlich erläuterten P.D.-Disks aus. Schutzgebühr 5,-DM in Briefmarken oder bar.
P.D.-Info → monatlich erscheinende Infoschrift über die neueste Public Domain.
Sonderinfos → Fast 100 Signum-PD-Zeichensätze. Jede Menge Grafik für STAD und Signum.
P.D.-Abo → Alle ST-Public-Domain kann bei uns bezogen bzw.abonniert werden.
Sämtliche P.D. wird ständig aktualisiert und auf Virenbefall überprüft.

Public Domain

TOP-SPIELE

Precious Metal *(4 Topspiele)*	59,- DM
Lizenz zum Töten - 007	59,- DM
Kick off	48,- DM
Blood Money *(Knüllerpreis)*	68,- DM
Buffalo Bill's Rodeo-Games	76,- DM
Kult *(Knüllerpreis)*	69,- DM
Triad (Menace,Tetris,Baal)	74,- DM
Xenon 2 - Megablast *(Knüller)*	79,- DM
Grand Monster Slam	74,- DM
Micropr.Soccer *(Knüllerpreis)*	69,- DM

TOP-ANWENDUNGEN

Signum!2	398,- DM	Script (brandneu)	188,- DM
Daily Mail	159,- DM	Tempus V2.0	119,- DM
STAD	159,- DM	Calamus (neue Vers.)	748,- DM
Megamax Laser C	368,- DM	Megamax Modula 2	358,- DM
Lattice C	288,- DM	GFA-Basic V3 + Comp.	188,- DM
GFA-Assembler	139,- DM	Anti Viren Kit 3	85,- DM
Adimens ST+(brandneu)	388,- DM	AdiPROG ST	249,- DM

Fibuman - Informationen, Installation, Preise auf Anfrage

Endloslabel für 3,5' Disks je 100 Stck.	9,- DM
Farbbänder - P6,P6+,LQ800,850 je	16,- DM
Speichererw. 512 KB	328,- DM
für 260 ST, 520 ST/STM. Neueste Version steckbar, kompakt, kein Einlöten, mit MBit Chips.	
Atari ST 1x1 Buch	49,- DM

Alle Produkte sofort lieferbar !
24-Std.-Telefonservice
persönliche Abholung möglich
02164/7898

SOFTWARE

Creator(Appl.)	219.-
Flexi-Disk Protos	je
Hd-Utitlity	64.-
Scarabus	90.-
Neo-Desk II	79.-
Turbo ST	79.-
Stad +	159.-
GFA 3.0	179.-

HARDWARE

3.5 Laufwerk	269.-
5.25 " "	339.-
HF-Modulator	189.-
3.5 Disketten 2DD 50 Stück	100.-
Monitorswitch	44.90
Scheibenkleister	79.-

diverse Spiele und Bücher

MIDI

Steinberg 12 99.-
Cubase 690.-
Songs und Pattern+Drum als PD-Soft je Disk 10.-

NEU
LDW Calc 229.-
Trackball Marconi 189.-
Speichererweiterung 1MB 298.-
Towergehäuse(Lighthouse) 398.-

Hard & Software
Werner Wohlfahrtstätter
Irenenstr. 76c
4000 Düsseldorf 30
0211/429876

Public Domain Service

1. PD-Katalog 70 Seiten Schutzgebühr 5.-
2. Alle ST-PD Disk können bei uns bezogen bzw. Abonniert werden.
3. Signum PD-Zeichensätze und Grafiken für Signum oder jedes Malprogramm

Btx/Vtx-Manager

Btx/Vtx: Nase vorn

in der Welt der Telekommunikation mit dem *Btx/Vtx-Manager V3.0.*
Sie wollen Ihr Konto verwalten, Bestellungen aufgeben, eine Urlaubsreise buchen ...
Entdecken Sie jetzt die neuen komfortablen Wege, die Ihnen der *Btx/Vtx-Manager* (als intelligente Komplettlösung) mit dem Abruf aktuellster Informationen und Daten rund um die Uhr liefert.
Ausführliche Informationen erhalten Sie bei Ihrem Atari-Fachhändler oder direkt von uns.

Atari ST Btx/Vtx-Manager V3.0 für DM 389,– *an Postmodem* bzw. DM 289,– *an Akustikkoppler/Hayes-Modem.*

Drews EDV + Btx GmbH
Bergheimerstraße 134 b
D-6900 Heidelberg
Telefon (0 62 21) 2 99 00
Fax (0 62 21) 16 33 23
Btx-Nummer 0622129900
Btx-Leitseite *2 99 00 #

Bildschirmtext

d Drews

Daten-Austausch

mit **TRANSFILE** z.B. für

CASIO SF-7000/7500/8000
PSION Organiser II
SHARP IQ-7000



Verbinden Sie mit dieser Rechnerkopplung Ihren Datenbankrechner mit Ihrem PC/XT/AT, Atari oder Amiga. Damit können Sie problemlos Ihre Daten aus Programmen wie Exel, Adimens und dBase sicher in beide Richtungen austauschen. Ihre aktuellen Daten von zu Hause haben Sie somit jederzeit unterwegs verfügbar. Natürlich können Sie Ihre unterwegs erfaßten Daten später im Büro wieder auf den PC, ST oder Amiga übertragen und mit der mitgelieferten Software bearbeiten.
Fordern Sie weitere Infos an!



Komplett mit Interface, Software und dt. Handbuch
DM 179,–

yellow C·O·M·P·U·T·I·N·G
Pf. 1136/32 · D-7107 Bad Friedrichshall
Telefon 07136/2 00 16 · Fax 07136/2 25 13

6520 Worms

orion
Computersysteme
GmbH

6520 Worms · Friedrichstraße 22

Telefon 0 62 41 / 67 57 - 58

6700 Ludwigshafen

MKV Computermarkt

Bismarck-Zentrum
6700 Ludwigshafen
Telefon 06 21 - 52 55 96

6720 Speyer

THEILLE
Computersysteme
Gilgenstraße 4 · 6720 Speyer
Telefon (0 62 32) 7 72 16

Etzkorn
DIE FACHLEUTE FÜR COMPUTER

AUTORISIERTER ATARI - VERTRAGSHÄNDLER
HARDWARE, SOFTWARE & ZUBEHÖR FÜR
• Apple • Acorn • Jonathan • MS/Dos •

6720 Speyer AUESTRAßE 20
Telefon 06232 /32428 oder 32435
Mo.-Fr. 9 - 12 u. 15 - 18.30, Sa. 9 - 14 Uhr

6750 Kaiserslautern

ATARI
wir machen Spitzentechnologie preiswert

INFODAT GmbH

Handels- und Entwicklungsgesellschaft
für Informatik und Datenverarbeitung
Schubertstr. 16, 6750 Kaiserslautern
Tel. (0631) 6 35 97 Fax (0631) 6 35 89
ATARI - VERTRAGSHÄNDLER

6800 Mannheim

GAUCH+STURM
Computersysteme + Textsysteme
6800 Mannheim 24
Casterfeldstraße 74-76
☎ (06 21) 85 00 40 · Teletex 6 211 912

Computer-Center
am Hauptbahnhof GmbH

L 14, 16-17
6800 Mannheim 1
Tel. (06 21) 2 09 83 / 84

6900 Heidelberg

JACOM FAMILA-CENTER
Hardware · Software
Schulung · Service

Hertzstraße 1 · 6900 Heidelberg 1
Telefon (0 62 21) 30 24 37

7000 Stuttgart

Walliser +Co. Personal Computer
Marktstr. 48, Tel. 07 11/56 71 43
7000 Stuttgart-Bad Cannstatt



7022 L-Echterdingen

Autorisierter ATARI-
System-Fachhändler
ATARI ST
Matrai computer
Matrai Computer GmbH
Bernhauser Str 8
7022 L-Echterdingen
☎ (07 11) 79 70 49

7030 Böblingen

Verkauf – Service – Software

Norbert Hlawinka
Sindelfinger Allee 1
7030 Böblingen
Tel. 0 70 31 / 22 60 15

mca
COMPUTER
CENTER

7047 Jettingen

Verkauf – Service – Software

Norbert Hlawinka
Heilbergstraße 3
Im Multi-Center
7047 Jettingen
Telefon (0 74 52) 7 76 15

mca
COMPUTER
SHOP

7100 Heilbronn

Walliser +Co. Personal Computer
Mönchseestraße 99, 7100 Heilbronn
Tel. 0 71 31/6 00 48



Computer-Welt

Seel's
Am Wollhaus 6
7100 Heilbronn
Tel. 0 71 31 - 6 84 01 - 02

7150 Backnang

Computer-Fans finden bei uns alles von:

Servicestation
Vertragshändler
Computer-Systeme
Software-Hardware

commodore
Schneider COMPUTER DIVISION
ATARI
Weeske
Das Elektrohaus am Nordring
Potsdamer Ring 10
7150 Backnang
Tel. 0 71 91 15 28

7312 Kirchheim/Teck

PD's Comp&Phone DTP
ATARI -Systemfachhandel
Hardware – Software – Service
Plochingerstr. 14 · 7312 Kirchheim/Teck
Tel. 07021/3949

7410 Reutlingen

MKV GMBH

Listplatz 2
7410 Reutlingen
Telefon 0 71 21 - 3 66 47

Werner Brock
Computer-Studio
Untere Gerberstr. 15 · 7410 Reutlingen
Tel. 0 71 21 - 3 42 87
Tx 1 72 414 024 RMI D box rmi.taisoft Fax 0 71 21 - 33 97 79
Autorisierter Systemfachhändler für:
ATARI, Schneider, Commodore, Panasonic,
Kaypro, Sharp, NEC, OKI, STAR,...

7475 Meßstetten

Ihr ATARI-Systemhändler im Zollern-Alb-Kreis
HEIM + PC-COMPUTERMARKT
HARDWARE · SOFTWARE · LITERATUR
SCHEURER
ATARI COMMODORE CUMANA DATA-BECKER
MULTITECH RITEMAN SCHNEIDER THOMSON
7475 Meßstetten 1 · Hauptstraße 10 · 0 74 31 / 6 12 80

7500 Karlsruhe

ERHARDT Am Ludwigsplatz
Am Ludwigsplatz · 7500 Karlsruhe 1 · Tel. (07 21) 16 08-0

MKV GMBH

Kriegsstraße 77
7500 Karlsruhe
Telefon (07 21) 8 46 13

7600 Offenburg

FRANK LEONHARDT ELECTRONIC
Ihr Fachgeschäft für Microcomputer · Hifi · Funk

In der Jeuch 3
7600 Offenburg
Telefon 07 81 / 5 79 74

7700 Singen

Udo Meier Computersysteme
Ringstraße 4
Telefon (0 77 31) 6 82 22

7730 VS-Schwenningen

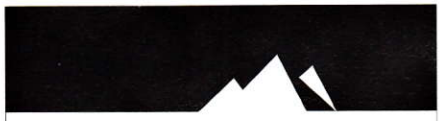
BUS BRAUCH & SAUTER COMPUTER TECHNIK

Villinger Straße 85
7730 VS-Schwenningen
Telefon 0 77 20 / 3 80 71-72

7750 Konstanz

ATARI ★ PC's ★ SCHNEIDER
computer - fachgeschäft
rösler
Rheingutstr. 1 · ☎ 0 75 31-2 18 32

7800 Freiburg



Kartäuserstraße 59
7800 Freiburg
Telefon: 0761/36870-70
Fax: 0761/25849
PYRAMID COMPUTER GmbH

7850 Lörrach



7890 Waldshut-Tiengen

hettler-data
service gmbh
Lenzburger Straße 4
7890 Waldshut-Tiengen
Telefon 0 77 51 / 30 94

7900 Ulm

EDV-Systeme
Software-erstellung
Schulung

Systemhaus:
Frauenstraße 28
7900 Ulm/Donau
Tel. (07 31) 2 80 76
Telex 7 12 973 csulm-d

COMPUTER STUDIO

7918 Illertissen

bidtech gmbh
technische Informationsysteme
Computerladen

Marktplatz 13
7918 Illertissen
0 73 03 / 50 45

8000 München

ludwig
COMPUTER + BÜROTECHNIK
COMPUTER · SOFTWARE · PERIPHERIE
BERATUNG · TECHN. KUNDENDIENST
INGOLSTÄDTER STRASSE 62L
EURO-INDUSTRIE-PARK · 8000 MÜNCHEN 45
TELEFON 089 / 3 11 30 66 · TELETEX 89 83 41

City Studio
Rindermarkt 6, 8000 München 2
Tel. 089 / 26 09 801 - 02

Ihr Spezialist für:
dBMAN - Komplettlösungen
vortex Massenspeicher
Vortex, dBMAN sind eingetragene Warenzeichen zugunsten Dritter
A B A C München
Kellerstraße 11, 8000 München 80
Tel. 089/ 448 99 88

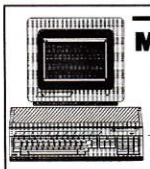
Michael Weichselgartner
MW
Ihr
Computerspezialist
Chiemgaustr. 152
8000 München 90
089/6804642

schulz computer
Schillerstraße 22
8000 München 2
Telefon (0 89) 59 73 39
Beratung · Verkauf · Kundendienst

8032 Gräfelfing

ProMarkt
Pasinger Straße 94 8032 Gräfelfing
Telefon 089 - 8548823
Fax 089 - 8541764

8150 Holzkirchen



MÜNZENLOHER GmbH
Tölzer Straße 5 * 8150 Holzkirchen
Tel.(08024)1814 * Fax.(08024)4879

ATARI-SCHNEIDER-NEC
PANASONIC-PHILIPS
TOSHIBA PORTABLE-LUCKY GOLDSTAR
Hard- und Software in großer Auswahl
Service und Beratung sind bei uns Inklusive

8200 Rosenheim

ludwig
COMPUTER + BÜROTECHNIK
Kufsteiner Str. 11, 8200 Rosenheim
Tel. (08031) 38 00 30
Fax (08031) 1 53 34

Autorisierter ATARI System-Fachhändler
Fischer & Bach
Computer GmbH
Münchner Straße 41 · 8200 Rosenheim
Tel.: 0 80 31 / 1 47 55 · Fax: 0 80 31 / 1 76 67
Hardware · Software
Schulung · Schnellreparatur

Bei uns werben bringt
GEWINN



Sprechen Sie mit uns.
Heim Verlag 0 61 51 / 56057
BUF

8400 Regensburg

Zimmermann elektroland

8400 Regensburg
Dr.-Gessler-Str. 8
☎ 09 41 / 9 50 85

8390 Passau
Kohlbruck 2a
☎ 08 51 / 5 20 07

8423 Abensberg

WITTICH COMPUTER GMBH
Tulpenstr. 16 · 8423 Abensberg
☎ 0 94 43 / 4 53





Programmier Praxis

INHALT

Tastenunterstützung in Dropdown-Menüs
C ..... Seite 80

Menüleisten total
GFA-BASIC ..... Seite 86

Eigener Desktop in Pascal plus
Pascal ..... Seite 90

Eingabe von Termen
GFA-BASIC ..... Seite 93

Tasten-unterstützung in Dropdown- Menüs

Urs Müller

Ein Grund für die hohe Benutzerfreundlichkeit von GEM-Programmen sind sicher die Dropdown-Menüs. Doch können sie nach einiger Zeit genauso benutzerunfreundlich werden, falls sie nicht konsequent durch Tastenkombinationen unterstützt werden. An diesem Punkt mangelt es leider bei vielen Programmen. Ein Grund dafür ist sicher, dass das AES solche Tastenkombinationen nicht direkt unterstützt. Ein weiterer Grund dürfte sein, dass das am weitesten verbreitete Programm, das Desktop, dies ebenfalls nicht tut. In diesem Beitrag wird eine Methode beschrieben, die mindestens den ersten Grund entfernt.

In der GEM-Dokumentation von Digital Research wir dem Programmierer empfohlen: Wenn der Benutzer eine Tastenkombination drückt, welche einem Menüeintrag zugeordnet ist, so soll das Programm zuerst den entsprechenden Titel mit der Funktion *menu_tnormal* invertieren. Danach kann die entsprechende Aktion durchgeführt werden, und schließlich muß durch einen erneuten Aufruf von *menu_tnormal* der Menütitel wieder normal dargestellt werden.

Jeder Programmierer, der obiges Rezept in seinen Programmen berücksichtigt, weiß, wie mühsam es unter Umständen sein kann, die Menüabfrage zweispurig fahren zu müssen. Einerseits wird ja auf die Meldung einer richtigen Menübetätigung gewartet, andererseits müssen die Tasteneingaben analysiert werden. Umstrukturierungen im Menü werden damit ebenfalls unnötig aufwendig.

Ein besseres Konzept

Nun ist es in den meisten Fällen für das Programm unwichtig zu wissen, ob ein Menüeintrag angeklickt oder ob die entsprechende Tastenkombination gedrückt worden ist. Für den Programmierer wäre es also angenehm, wenn er die Zuordnung der Menüeinträge zu den Tastenkombinationen bereits im Resource Construction Set vornehmen könnte, und wenn das AES danach für das Klikken auf einen Menüeintrag oder das Betätigen der entsprechenden Tastensequenz dem Programm dieselbe Meldung schicken würde.

Damit würde die oben erwähnte Zweispurigkeit verhindert. Es könnten jederzeit, ohne Programmänderung, die Tastenkombinationen anderen Menüeinträgen zugewiesen werden. Auch wäre die Neuzuweisung der Tastenkombinationen, welche bei der Übersetzung der Resource in eine andere Sprache eventuell notwendig ist, kein Problem mehr.

Meldungen unter GEM

Meldungen können nicht nur vom System an Programme geschickt werden, auch Programme selbst haben die Möglichkeit, Meldungen an andere Applikationen zu schicken. Dazu dient der AES-Befehl *appl_write*. So kann ein Programm beispielsweise mit einem Accessory "sprechen". Ein schönes Beispiel dafür ist der IDC-Standard [1]. Ebenfalls ist es für ein Programm nicht verboten, Meldungen an sich selbst zu verschicken.

Damit wird folgendes Vorgehen ermöglicht: Das Programm wartet im Event-Loop auf eine Meldung (*MU_MESSAG*) und auf einen Tastendruck (*MU_KEYBD*) und, falls nötig, auf weitere Ereignisse. Im Falle eines Tastendrucks wird dieser nicht auf die Zugehörigkeit zu einem Menüeintrag untersucht, sondern er wird einer Prozedur übergeben. Diese durchsucht die letzten Buchstaben aller Menüeinträge. Wird dabei zum Beispiel die Zeichenkette ^C gefunden, was heißt, daß die

entsprechende Funktion auch mit der Tastenkombination [Control] + 'C' ausgelöst werden kann, und ist diese Kombination auch gedrückt worden, wird eine entsprechende Meldung an die eigene Applikation geschickt. Danach wird die Subroutine beendet, und das Hauptprogramm schließt seinen Loop und wartet auf ein neues Ereignis. Dieses wird die zuvor abgeschickte Meldung sein, worauf das Programm glaubt, der entsprechende Menüeintrag sei mit der Maus angeklickt worden, und die richtige Funktion ausführt.

Das heißt, die Zuweisung der Tastenkombinationen zu den Menüeinträgen geschieht in den Menüeinträgen selbst, indem man dort die entsprechende Tastenkombination vermerkt.

Der Menübaum

Bevor wir den Menübaum nach Einträgen durchsuchen können, brauchen wir etwas Kenntnis über dessen Aufbau. Ein Menübaum unterscheidet sich prinzipiell nicht von anderen Objektbäumen. Die Objekte sind dieselben und werden durch die 24 Bytes lange Struktur (in pascal Record) *OBJECT* dargestellt. Für uns wichtig sind die drei Zeiger *ob_next*, *ob_head* und *ob_tail*, welche auf das nächste Objekt auf derselben Stufe, auf das erste und das letzte Objekt des jeweiligen Unterbaumes zeigen. Das letzte Objekt auf einer Stufe zeigt auf die "Mutter" zurück. Ein Zeiger ins "Nichts" wird durch -1 repräsentiert. Die Zeiger beinhalten nicht direkt eine Adresse, sondern besagen, welche Nummer das bezeichnende Objekt in der Objektliste hat.

Weiter werden uns *ob_type* und *ob_spec*, die Art des Objektes und ein Zeiger (Adresse) auf den String in den Menüeinträgen interessieren. Eine genaue Beschreibung der Objekte und der Objektbäume befindet sich in den meisten GEM-Büchern. Die Struktur eines Menübaumes ist in Abb. 1 dargestellt.

Eine mögliche Realisierung

Im folgenden wird eine mögliche Realisierung der obigen Idee beschrieben. Die Routine heißt *menu_search* und ist in der Programmiersprache C geschrieben. Sie kann zum Hauptprogramm gelinkt und von dort aufgerufen werden. Ein einfaches Beispielprogramm erläutert die Anwendung der Routine.

Folgende Tastenkombinationen werden unterstützt (X steht für ein beliebiges einzelnes Zeichen):

^X	[Control]+X
X	[Alternate]+X
F3	F1 bis F10
F5	[Shift]+F1...F10
F15	[Shift]+F1...F10
'X'	X (ohne zusätzliche Taste)

Accessory-Einträge sind von der Suche ausgeschlossen (der Eintrag "Über Programm..." wird aber berücksichtigt). Kommt dieselbe Tastenkombination in mehreren Menüeinträgen vor, wird immer der erste gefundene Eintrag genommen.

Wird bei der Suche durch den Menübaum auf einen Eintrag gestoßen, dessen Vermerk mit der übergebenen Tastenkombination (*kstate und key*) übereinstimmt, werden folgende Schritte unternommen:

1. Der entsprechende Menütitel wird invers dargestellt.

2. Es wird eine Meldung an die eigene Applikation abgeschickt. Der Message-Buffer sieht dabei folgendermaßen aus:

```
msg_buff[0] = 10 (MN_SELECTED)
msg_buff[1] = ap_id
msg_buff[2] = 0 (Länge der Meldung kleiner als 16 Bytes)
msg_buff[3] = Index des Menütitels
msg_buff[4] = Index des Menüeintrages
```

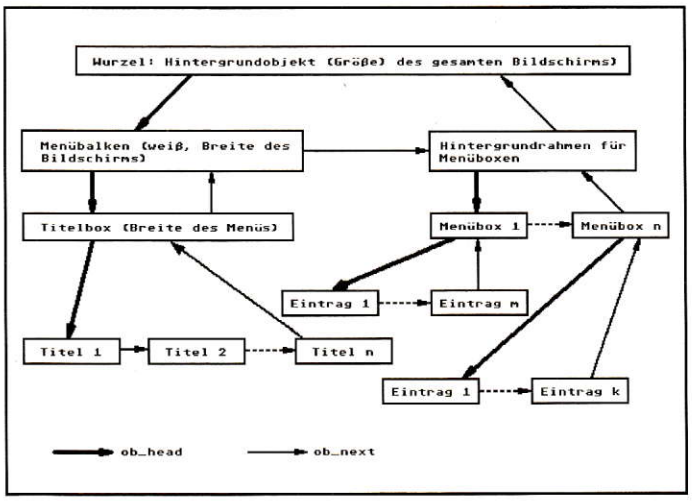


Abb.1: Struktur eines Menübaumes

Damit wird exakt die Situation simuliert, wie sie das Hauptprogramm antrifft, wenn derselbe Menüeintrag mit der Maus angeklickt worden wäre.

Ist ein Menüeintrag *disabled* (grau dargestellt), kann er mit der Maus nicht selektiert werden. Die Routine verschickt in einem solchen Fall ebenfalls keine Meldung an das Hauptprogramm. Ist ein Menütitel *disabled*, kann das entsprechende Menü gar nicht herunterklappen. In diesem Fall ist das ganze Menü von der Suche ausgeschlossen.

Folgende Parameter müssen beim Aufruf der Funktion übergeben werden:

int ap_id: Application Identification. Diese ist notwendig für die Suchroutine, um zu wissen, an wen die Meldungen verschickt werden müssen. Die *ap_id* wird von der Funktion *appl_init* zurückgegeben. Sie kann aber auch später noch aus dem AES-Global-Array gelesen werden.

long *m_tree*: Adresse des Menübaumes. Sie kann mit *rsrc_gaddr* nach dem *rsrc_load*-Aufruf erfragt werden und muß unter anderem auch der Funktion *menu_bar* übergeben werden.

int kstate: Status der Sondertasten ([Alternate], [Control], [Shift]). Dieser Wert wird von der Funktion *evnt_multi* zurückgegeben.

int key: Scan- und ASCII-Code der gedrückten Taste. Dieser Wert wird ebenfalls von der Funktion *evnt_multi* zurückgegeben.

Der Aufruf von C aus sieht also wie folgt aus:

menu_search(ap_id,m_tree, kstate,key);

Die Funktion kann aber auch vom ST Pascal Plus aus aufgerufen werden. Sie muß dann folgendermaßen importiert werden:

procedure menu_searc(ap_i: integer; m_tree: Menu_Ptr; kstate, key: integer);
c;

und kann danach wie gewohnt aufgerufen werden.

Auch von Assembler kann die Funktion aufgerufen werden:

```
move.w   key,-(sp)
move.w   kstate,-(sp)
move.l   m_tree,-(sp)
move.w   ap_id,-(sp)
jsr      menu_search
lea      10(sp),sp
```

Die Beschreibung der Routine

Die Routine besteht aus zwei Teilen: der Hauptroutine *menu_search*, welcheden gesamten Menübaum absucht, und der Hilfsroutine *test_entry*, welche die Menüeinträge auf zugewiesene Tastenkombinationen untersucht.

Ein erwähnenswertes Problem der Hauptroutine ist, festzustellen, welches Zeichen vom Benutzer gedrückt worden ist. Diese Information ist in den übergebenen Variablen *kstate* und *key* enthalten. Wenn zum Beispiel [Control] zusammen mit einer Taste gedrückt wird, erhält man nicht mehr den ASCII-Code des entsprechenden Zeichens. Der dann erhaltene ASCII-Code ist nicht einmal eindeutig. So liefern zum Beispiel die Kombinationen [Control]+'3 und [Control]+'S' denselben ASCII-Code!

Man ist also gezwungen, auf den Scancode zu achten. Dieser ist jedoch an die physikalische Position der Tasten gebunden. Gewisse Zeichen sind bei verschiedensprachigen Tastaturen an verschiedenen Orten angebracht, liefern also verschiedene Scancodes. Typisch ist das Beispiel von Programmen, welche in ihren Menüeinträgen die Kombination "^-" versprechen. Auf der deutschen Tastatur muß man aber [Shift][Control]'?' drükken.

Ein möglicher Ausweg besteht darin, die Tastaturtabelle des XBIOS zu verwenden. Sie ist bei jedem Betriebssystem an die jeweilige Tastatur angepaßt. Es sind insgesamt drei Tabellen für "Unshift", "Shift" und für "Caps Lock". Ein Zeiger auf ein Array mit den Zeigern auf die drei Tabellen kann mit der XBIOS-Funktion *keytbl* erfragt werden. Diese Tabellen werden mit dem Scancode als Index gelesen und enthalten den ASCII-Code der entsprechenden Taste.

Die Hilfsroutine *test_entry* durchsucht das Ende des übergebenen Strings nach möglichen Vermerken von Tastaturkombinationen. Diese Routine muß also abgeändert werden, falls andere als die hier implementierten Tastenkombinationen oder andere Notationsformen gewünscht sind.

Literatur:

[1] Obrero A./Waldvogel M.: "Ein Standard für Scannersoftware", ST-Magazin 1/89

```
/*********************************************/
/*                                           */
/* menu_search                               */
/*                                           */
/* Aufruf: menu_search(ap_id,m_tree,kstate,key)*/
/*                                           */
/* int    ap_id: ID der aufrufenden Applikation*/
/* OBJECT *m_tree: Zeiger auf den Menubaum     */
/* int    kstate: Keyboard state (liefert
                                  event_multi) */
/* int    key: ASCII & Scancode (liefert
                                  evnt_multi)  */
/*                                           */
/* Kompiliert mit Megamax Laser C, Version 1.2 */
/*                                           */
/*                                           */
/*  von Urs Mueller, Mai 1989                */
/*  (c) MAXON Computer GmbH                  */
/*********************************************/

/* durchsucht alle Menueintraege des
   uebergebenen Menubaumes nach den */
/* letzten Buchstaben. Wird bei einem Eintrag
   eine logische ueberein- */
/* stimmung mit den Werten von kstate und key
   gefunden, so wird eine  */
/* Meldung MN_SELECTED an die aufrufende
   Applikation geschickt.       */
/* Dies geschieht natuerlich nur dann, wenn der
   Menueintrag und der   */
/* entsprechende Menutitel nicht "disabled"
   sind.                    */

/* Moegliche Werte fuer die letzten Buchstaben
   in einem Menueintrag   */
/* sind:                  */
/* "F1" bis "F10": Funktionstasten             */
/* "F11" bis "F20": [Shift] Funktionstasten    */
/* [Shift]F1 bis [Shift]F10:                   */
/* ( [Shift] = ASCII(1) )
/* "^" und Buchstabe: [Control] Buchstabe      */
/* Fullbox-Symbol und Buchstabe  : [Alternate]
   Buchstabe              */
/* Hochkomma-Buchstabe-Hochkomma : Buchstabe
   ohne [Ctrl] oder [Alt]   */

/* Gesucht wird im Menueintrag von Rechts (wobei
   Spaces uebersprungen */
/* werden) nach einer der obigen Kombinationen.*/
/* Bei den Buchstaben wird zwischen klein- und
   Grossbuchstaben nicht  */
/* unterschieden
```

```


/********************/
/* Include-Dateien */
/********************/
#include "GEMDEFS.H"     /* common object
                         definitions and structures */
#include "OBDEFS.H"      /* common GEM
                            definitions */
#include "OSBIND.H"      /* gemdos, bios, xbios */




/**************/
/* Konstanten */
/**************/

#define TRUE 1
#define FALSE 0

#define F1  0x3b           /* Scancode von F1 */
#define F10 0x44           /* Scancode von F10 */
#define F11 0x54           /* Scancode von
                              [shift] F1 */
#define F20 0x5d           /* Scancode von
                              [shift] F10 */

#define M_NORMAL 1         /* Menutitel normal
                              dargestellt */
#define M_REVERSE 0        /* Menutitel Weiss auf
                              Schwarz */



/***************************/
/* Menueintrag untersuchen */
/***************************/

static int test_entry(str, chr, scan, state)
char  *str, chr;
int   scan, state;

{
char  *pchar, vchr;
int   ret, zahl;

ret = FALSE;                                    →
```

```
 88:  pchar = str;
 89:  while (*pchar) pchar++; /* pchar an den
                                Schluss des Strings */
 90:  while (*--pchar == ' ');/* von rechts
                              erstes Zeichen suchen */
 91:  vchr = *pchar;
 92:  if (vchr >= 'a' && vchr <= 'z') vchr = vchr -
              'a' + 'A';
 93:
 94:  if (vchr == chr)
 95:    {
 96:    pchar--;
 97:    if (*pchar=='^' && state==K_CTRL || *pchar==7
         && state==K_ALT)
 98:      ret = TRUE;
 99:    }
100:  else if (*pchar == '\'' && (state & (K_CTRL |
            K_ALT)) == 0)
101:    {
102:    pchar--;
103:    vchr = *pchar;
104:    if (vchr >= 'a' && vchr <= 'z') vchr = vchr -
         'a' + 'A';
105:    if (vchr == chr && *(pchar-1) == '\'')
106:      ret = TRUE;
107:    }
108:  else if (*pchar >= '0' && *pchar <= '9')
109:    {
110:    zahl = *pchar - '0';
111:    pchar--;
112:    if (*pchar >= '0' && *pchar <= '9')
113:      {
114:      zahl += (*pchar - '0') * 10;
115:      pchar--;
116:      }
117:    if (*pchar == 'F')
118:      {
119:      if (*(pchar-1) == 1)  /* [Shift]-Zeichen */
120:        zahl += 10;
121:      if ((zahl >= 1) && (zahl <= 10))
122:        if (zahl == scan - F1 + 1) ret = TRUE;
123:      if ((zahl >= 11) && (zahl <= 20))
124:        if (zahl == scan - F11 + 11) ret = TRUE;
125:      }
126:    }
127:  return(ret);
128:
129:  }   /* end of test_entry */
130:
131:
132:
133:
134:
135:  /*****************/
136:  /* Hauptprogramm */
137:  /*****************/
138:
139:  void menu_search(ap_id, m_tree, kstate, key)
140:  int    ap_id;
141:  OBJECT *m_tree;
142:  int    kstate;
143:  int    key;
144:
145:
146:  {
147:  int     msg_buff[8];   /* message buffer */
148:  int     do_quit;
149:  keytbl  *pkeytbl;
150:  char    *kbd_unshift;  /* TastTab. normal */
151:  char    *kbd_shift;    /* TastTab. shift */
152:  char    *kbd_caps;     /* TastTab. caps lock */
153:  char    chr;
154:  int     state, scan, desk;
155:  int     mother_title, child_title, mother_entry,
          child_entry;
156:
157:
158:  do_quit = FALSE;
159:  desk = TRUE;
160:
161:  pkeytbl = Keytbl(-1L, -1L, -1L);
```

```
162:  kbd_unshift = (*pkeytbl).unshift;  /* Zeiger auf
                                Tabelle 'normal' */
163:  kbd_shift   = (*pkeytbl).shift;    /* Zeiger auf
                                Tabelle 'shift' */
164:  kbd_caps    = (*pkeytbl).capslock; /* Zeiger auf
                                Tabelle 'caps lock' */
165:
166:  wind_update(BEG_UPDATE);        /* warten bis
                                   Drop-Downs.. */
167:  wind_update(END_UPDATE);        /* geschlossen */
168:
169:  scan = key/256 & 0xff;
170:  if ((kstate & (K_LSHIFT | K_RSHIFT)) == 0)
171:    chr = *(kbd_unshift + (scan));
172:  else
173:    chr = *(kbd_shift + (scan));
174:  if (chr >= 'a' && chr <= 'z') chr += 'A' - 'a';
                                   /* upcase */
175:
176:  if ((kstate&K_ALT) != 0 && (kstate&K_CTRL) == 0)
177:    state = K_ALT;      /* [Alt] gedrueckt */
178:  else if ((kstate&K_CTRL) != 0 && (kstate&K_ALT)
       == 0)
179:    state = K_CTRL;     /* [Shift] gedrueckt */
180:  else if ((kstate&K_ALT) == 0 && (kstate&K_CTRL)
       == 0)
181:    state = 0;          /* nichts gedrueckt */
182:  else do_quit = TRUE; /* [Alt] und [Shift] */
183:
184:  mother_title = (m_tree + m_tree->ob_head)-
                    >ob_head;
185:  child_title = (m_tree + mother_title)->ob_head;
186:  mother_entry = (m_tree + m_tree->ob_tail)-
                    >ob_head;
187:  child_entry = (m_tree + mother_entry)->ob_head;
188:
189:  while (!do_quit)
190:    {                          /* title loop */
191:    if (((m_tree + child_title)->ob_state &
           DISABLED) == 0)
192:      while(!do_quit && child_entry !=
             mother_entry && child_entry != -1)
193:        {
194:        if ((((m_tree + child_entry)->ob_state &
               DISABLED) == 0) &&
195:            ((m_tree + child_entry)->ob_type ==
              G_STRING ||
196:             (m_tree + child_entry)->ob_type ==
              G_BUTTON))
197:          do_quit = test_entry((char*)(m_tree +
                    child_entry)->ob_spec,
198:                    chr, scan, state);
199:        if (do_quit)
200:          {
201:          msg_buff[0] = MN_SELECTED;
202:          msg_buff[1] = ap_id;
203:          msg_buff[2] = 0;  /* Laenge Nachricht */
204:          msg_buff[3] = child_title; /* Titel */
205:          msg_buff[4] = child_entry; /* Eintrag */
206:          menu_tnormal(m_tree, child_title,
                      M_REVERSE);
207:          appl_write(ap_id, 16, msg_buff);
                     /* Message senden */
208:          }
209:        child_entry = (m_tree + child_entry)-
                       >ob_next;
210:        if (desk)
211:          {          /* Accessories nicht testen */
212:          child_entry = mother_entry;
213:          desk = FALSE;
214:          }
215:        }
216:    child_title = (m_tree + child_title)->ob_next;
217:    mother_entry = (m_tree + mother_entry)-
                     >ob_next;
218:    child_entry = (m_tree + mother_entry)->ob_head;
219:    if (child_title == mother_title) do_quit =
          TRUE;
220:    } /* while (!do_quit) */
221:
222:  }   /* end of menu_search */
```

→

```
#define MENU 0              /* TREE */
#define M_INFO 7            /* OBJECT in TREE #0 */
#define M_CTRL_R 16         /* OBJECT in TREE #0 */
#define M_ALT_E 17          /* OBJECT in TREE #0 */
#define M_QUEST 18          /* OBJECT in TREE #0 */
#define M_F1 19             /* OBJECT in TREE #0 */
#define M_F13 20            /* OBJECT in TREE #0 */
#define M_F15 21            /* OBJECT in TREE #0 */
#define M_QUIT 23           /* OBJECT in TREE #0 */
```

```
/*********************************************/
/* Beispiel zur Nutzung der Routine          */
/* menu_search                               */
/*                                           */
/* von Urs Müller, Mai 1989                  */
/* (c) MAXON Computer GmbH                   */
/*********************************************/



/*******************/
/* Include-Dateien */
/*******************/

#include "M_DEMO.H"        /* Includefile für
                              Resource */
#include "GEMDEFS.H"       /* common object
                         definitions and structures */
#include "OBDEFS.H"        /* common GEM
                              definitions */



/***************/
/* Konstanten */
/***************/

#define RSCNAME "M_DEMO.RSC" /* Name der
                                Resource-Datei */

#define TRUE 1
#define FALSE 0

/* Konstanten für menu_tnormal (AES 33) */

#define M_NORMAL 1         /* Menutitel
                              normal dargestellt */
#define M_REVERSE 0        /* Menutitel
                              Weiss auf Schwarz */



/**************/
/* Variablen */
/**************/

OBJECT        *menu_tree;     /* &Menu */
int           ap_id;          /* Application
                                 Identification */
int           dummy;



/***************/
/* Funktionen */
/***************/

static void do_info()
{
dummy = form_alert(1, "[0][ |Menueintrag Info ][
                              OK ]");
}

static void do_ctrl_r()
{
dummy = form_alert(1, "[0][ |Funktion 1 ][ OK
                              ]");
}

static void do_alt_e()
{
dummy = form_alert(1, "[0][ |Funktion 2 ][ OK
                              ]");
}

static void do_quest()
{
dummy = form_alert(1, "[0][ |Funktion 3 ][ OK
                              ]");
}

static void do_f1()
{
dummy = form_alert(1, "[0][ |Funktion 4 ][ OK
                              ]");
}

static void do_f13()
{
dummy = form_alert(1, "[0][ |Funktion 5 ][ OK
                              ]");
}

static void do_f15()
{
dummy = form_alert(1, "[0][ |Funktion 6 ][ OK
                              ]");
}



/**************/
/* event_loop */
/**************/

static void event_loop()

{
int     msg[8];            /* Message-Buffer */
int     event;             /* Event-Typ */
int     key, kstate;       /* Taste und Status
                              der Kontrolltasten */
int     do_quit;           /* Flag */


do_quit = FALSE;

do
  {
  event = evnt_multi(MU_KEYBD | MU_MESAG,
                     0, 0, 0, 0, 0, 0, 0, 0,
                     0, 0, 0, 0, 0,
                     msg,
                     0, 0, &dummy, &dummy,
                           &dummy,
                     &kstate,
                     &key,
                     &dummy);
  switch(event)
    {
    case MU_KEYBD: menu_search(ap_id, menu_tree,
                               kstate, key);
                   break;

    case MU_MESAG: if (msg[0] == MN_SELECTED)
                     {
                     switch(msg[4])
                       {
                       case M_INFO:   do_info();
                                      break;
                       case M_CTRL_R:do_ctrl_r();
                                     break;
```

→

```
                        case M_ALT_E: do_alt_e();
                                      break;
                        case M_QUEST: do_quest();
                                      break;
                        case M_F1:    do_f1();
                                      break;
                        case M_F13:   do_f13();
                                      break;
                        case M_F15:   do_f15();
                                      break;
                        case M_QUIT:  do_quit =
                                      TRUE; break;
                      }     /* switch(msg[4]) */
                    if (!do_quit)
                      menu_tnormal(menu_tree,
                             msg[3], M_NORMAL);
                  }
                break;
      }     /* switch(event) */
    }
while (!do_quit);
}     /* end of event_loop */


/*****************/
/* Haupt-Routine */
/*****************/
```

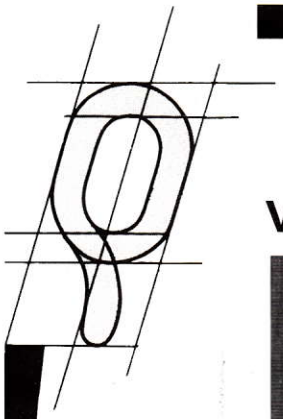
```

void main()

{
char    *rsc_name;

rsc_name = RSCNAME;  /* Name des Resourcefiles */

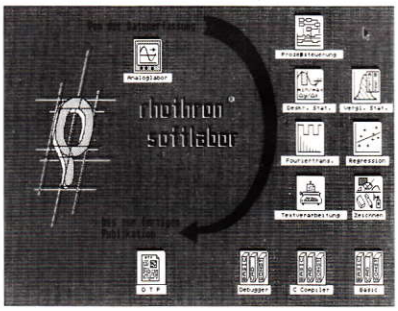
ap_id = appl_init();
if (ap_id >= 0)
  {
  graf_mouse(ARROW, 0L);              /*
   Mauszeiger = Pfeil */
  if (rsrc_load(rsc_name) == 0)
    dummy = form_alert(1, "[3][ |Resourcefile
                           not found! ][ OK ]");
  else
    {                 /* Resource wurde geladen */
    rsrc_gaddr(R_TREE, MENU, &menu_tree);
                      /* Menu */
    menu_bar(menu_tree, 1);   /* Menu zeichnen */
    event_loop();
    menu_bar(menu_tree, 0);  /* Menu entfernen */
    rsrc_free();
    }
  appl_exit();
  }
}
```

Entenmühlstraße 57
6650 Homburg/Saar
Telefon (06841) 64067
Telefax (06841) 2467

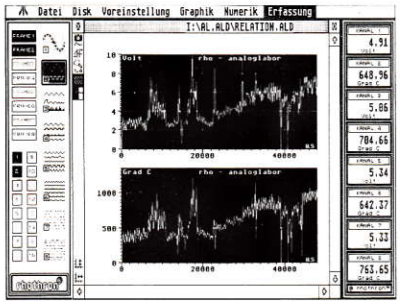
rhothron® GmbH

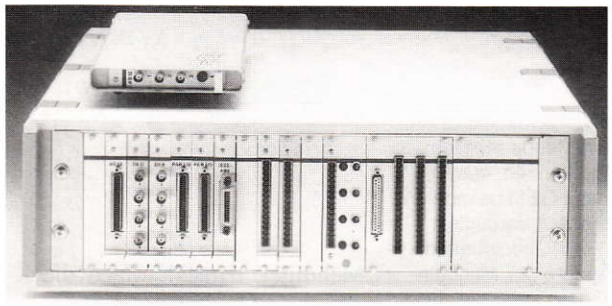
Von der Datenerfassung bis zur fertigen Publikation



- Messen
- Auswerten
- Dokumentieren

mit den
Hard- und Softwareprodukten
aus dem Hause rhothron





Computer Designed Instrumentation

für alle Wissenschaftler und Ingenieure

Menüleisten total

Matthias Andrä

Eine GEM-Menüleiste, die horizontal am oberen Bildschirmrand in einer vorgegebenen Schriftgrösse erscheint, ist zugegeben sehr praktisch und mit einigen GFA-BASIC-Befehlen auch leicht zu verwalten. Wie wäre es aber z.B., wenn man nicht nur eine Menüleiste, sondern beliebig viele darstellen könnte? Und wie wäre es dann, wenn man diese Menüleisten auf dem Bildschirm dort plazieren könnte, wo man dies gerade haben will, wenn man diese Menüleisten in vier verschieden Schriftgrössen und einer beliebigen Textfarbe zeichnen könnte, wenn man diese Menüleisten nicht nur horizontal sondern auch vertikal zeichnen könnte, wenn ..., wenn ..., wenn ..., das wäre doch nicht schlecht, oder?

Die hier vorgestellten Routinen ermöglichen es, einem Programm auf einfache Weise eine individuelle Note zu geben, ohne auf das bekannte Prinzip der GEM-Menüleisten verzichten zu müssen. Ein Nachteil dieser Routinen soll gleich zu Anfang genannt werden: Es ist nicht möglich, Accessories aus diesen Routinen heraus aufzurufen. Es ist daher sinnvoll, auch die normale Menüleiste zu verwenden, wenn man nicht auf Accessories verzichten will. Das können die Routinen:

- Unterstützung aller drei Auflösungen
- Verwaltung einer beliebigen Anzahl von Menüleisten
- horizontale oder vertikale Ausrichtung der Menüs
- wahlweise nach links, rechts, oben oder unten ausklappende Pull-Down-Menüs
- Menüleisten beliebig auf dem Bildschirm plazierbar
- 4 Schriftgrößen und frei definierbare Textfarbe
- versteckte Menüleisten möglich
- erst nach Mausklick ausklappende Pull-Down-Menüs möglich
- Menüeinträge können, zusätzlich zu den vom GEM bekannten Möglichkeiten, fett dargestellt werden (z.B. für Überschriften)
- einfaches Ändern der Menüeinträge (neuer Text kann beliebige Länge haben)

Der Menüaufbau

Die Data-Zeilen für eine Menüleiste sollten folgendermaßen aufgebaut sein: Der erste String in den Daten ist der erste Menütitel, danach folgen die einzelnen Menüeinträge, abgeschlossen wird das ganze mit einem Leerstring. Dann folgt der nächste Titel, usw., das Ende einer Menüleiste wird mit "***" angegeben. Daran anschließend können Daten für weitere Menüleisten nach dem selben Schema folgen. Wenn das erste Zeichen eines Menüeintrages ein "+" oder "-" ist, wird der Eintrag direkt unwählbar fett ("+"), oder unwählbar hell ("-") gemacht. Das "-" oder "+" erscheint nicht im Eintrag. Der Aufbau der Daten ist somit fast identisch mit dem von GFA-BASIC für GEM-Menüs verwendeten Prinzip. Die Menüdaten werden mit folgenden Befehlen eingelesen:

```
Restore >Menülabel<
Gosub Menues_berechnen(n)
Restore >Menülabel<
Gosub Menues_initialisieren
                (x_res,y_res)
```

n ist die Anzahl der Menüleisten, die am Stück eingelesen werden sollen. In *x_res* und *y_res* müssen die maximal möglichen Bildschirmkoordinaten angegeben werden (z.B. 639 und 399 für hohe Auflösung). Das ist schon alles, was zur Initialisierung notwendig ist, danach kann es losgehen.

Weitere Befehle

Gosub Menu_paint (m_nr,x1, y1,x2,y2,r1!,r2!,v!,k!) zeichnet und aktiviert eine Menüleiste.

m_nr entspricht der laufenden Nummer der Menüleiste (m_nr ≥ 0).

x1,y1,x2,y2 sind die Bildschirmkoordinaten der Menüleiste.

x1 und *y1* sind die Koordinaten der in Schreibrichtung linken unteren Ecke des Menüs.

x2 und *y2* geben dabei jeweils die Breite des Menüs an, wobei die Menüleiste immer mindestens so breit ist, wie es für den Menütext benötigt wird. Wird als Wert -1 angegeben, nimmt die Routine Standartwerte an.

r1! = Richtung 1
r1! = false => horizontal
r1! = true => vertikal
r2! = Richtung 2
r2! = false => Pull-Downs nach unten oder rechts
r2! = true => Pull-Downs nach oben oder links

v! = Versteckt-Flag
v! = false => Menü normal
v! = true => Menü versteckt
k! = Mausklick-Flag
k! = false => Pull-Downs normal
k! = true => Pull-Downs erscheinen erst nach Mausklick

*GosubOn_menu(*m_nr,*m_p)* fragt alle aktiven Menüleisten ab und gibt evtl. in *m_nr* die Nummer der Menüleiste, in *m_p* die Nummer des angewählten Eintrags zurück. Dieser Befehl sollte möglichst in eine Schleife gesetzt werden (z.B. DO..LOOP). Die Variablen müssen als Pointer eingesetzt werden, um eine Rückgabe zu ermöglichen!

Gosub Menu_image (m_nr, tg,f) setzt Textgröße und Farbe eines Menüs.

tg = Textgröße (0..3)
f = Textfarbe (0..15)

Gosub Menu_text (m_nr,m_p, a,t$) ändert Attribut und Text eines Eintrages.

m_p = Nummer des Menüeintrags
a = Attribut des Eintrags
a = -1=> keine Änderung,
a = 0 => normal ,
a = 1 => fett (unwählbar)
a = 2 => hell (unwählbar),
a = 3 => normal mit Häkchen
a ≥ 4 => hell mit Häkchen

t$ = Menüeintrag-String
wenn t$ = "", dann keine Änderung

Gosub Menu_kill(m_nr) löscht eine Menüleiste vom Bildschirm, um Änderungen in der Lage des Menüs oder in der Titelzeile zu ermöglichen.

Gosub Menu_lock (m_nr) sperrt die Bearbeitung eines Menüs, um z.B. einen Aufruf der in ihm enthaltenen Einträge vorübergehend zu vermeiden, oder in Verbindung mit *Gosub Menu_kill(m_nr)* ein Menü komplett zu entfernen. Die Daten des Menüs bleiben dabei erhalten und können für ein erneutes aktivieren der Menüleiste wieder genutzt werden. Alle anderen Prozeduren sollten nach Möglichkeit nicht genutzt werden, da das den normalen Ablauf mit hoher Wahrscheinlichkeit stören würde.

Die Variablen

Sämtliche von den Routinen benutzten externen Variablen haben ein *M_* vorangestellt. Daher dürfte es, wenn man im übrigen Programm Variablennamen mit einem solchen Anfang vermeidet, nicht zu ungewollten Effekten kommen. Im Allgemeinen sollte man darauf verzichten, auf Variablen, die von den Routinen benutzt werden, direkt zuzugreifen, "legale" Ausnahmen davon sind:

- M_hidden!(m_nr) => Flag für versteckte Menüleisten
- M_klick!(m_nr) => Maus klick-Flag für Pull-Down-Menüs

Die Variable *M_action!* kann dazu genutzt werden, um festzustellen, ob eine Menüfunktion aktiviert wurde oder nicht. Falls *M_action!=True* ist, sollte man nicht im Bildschirm zeichnen, da es dann z.B. möglich wäre, daß ein ausgeklapptes Pull-Down-Menü "im Weg" ist.

```
' ***********************************************
' *         Menü-Library von Matthias Andrä
' *             (c) MAXON Computer GmbH
' ***********************************************
'
Procedure Menues_berechnen(N%)
  ' Dimensionierungen der Arrays vornehmen
  ' und maximale AusmaPe der Menüs berechnen
  '
  Local I%,M%,T$,M_mpd%,M_mtit%
  '
  M_max%=Max(0,N%-1)
  Dim M_maxpulldown%(M_max%),M_maxtitel%(M_max%)
  Dim M_hidden!(M_max%),M_klick!(M_max%),
      M_back$(M_max%),M_back!(M_max%)
  Dim M_paint!(M_max%),M_aktiv!(M_max%),
      M_text_size%(M_max%)
  Dim M_th%(M_max%),M_tb%(M_max%),M_tof%(M_max%),
      M_color%(M_max%)
  Dim M_x%(M_max%),M_y%(M_max%),M_x1%(M_max%),
      M_y1%(M_max%)
  Dim M_direction1!(M_max%),M_direction2!(M_max%)
  '
  Clr M_mpd%,M_mtit%
  For M%=0 To M_max%
    M_maxpulldown%(M%)=0
    M_maxtitel%(M%)=0
    Clr I%
    M_titel!=True
    Repeat
      Read T$
      Exit If T$="***"
      If M_titel!
        Inc M_maxtitel%(M%)
        Clr I%,M_titel!
      Endif
      If T$=""
        M_titel!=True
        M_maxpulldown%(M%)=Max(M_maxpulldown%,I%)
      Endif
      Inc I%
    Until T$="***"
    Dec M_maxpulldown%(M%)
    Dec M_maxtitel%(M%)
    M_mpd%=Max(M_mpd%,M_maxpulldown%(M%))
    M_mtit%=Max(M_mtit%,M_maxtitel%(M%))
  Next M%
  '
  Dim Menu$(M_max%,M_mtit%,M_mpd%)
  Dim Menu_style%(M_max%,M_mtit%,M_mpd%)
Return
Procedure Menues_initialisieren(X_res%,Y_res%)
  ' Initialisiert die Daten der Menüleisten
  '
  Local A$
  M_xres%=X_res%
  M_yres%=Y_res%
  For M%=0 To M_max%
    For M_titel%=0 To M_maxtitel%(M%)
      For M_pulldown%=0 To M_maxpulldown%(M%)
        Let Menu$(M%,M_titel%,M_pulldown%)=""
      Next M_pulldown%
    Next M_titel%
    For M_titel%=0 To M_maxtitel%(M%)
      For M_pulldown%=0 To M_maxpulldown%(M%)+1
        Read T$
        Exit If T$=""
        If M_pulldown%=0
          Let Menu$(M%,M_titel%,M_pulldown%)=" "+
              T$+" "
        Else
          A$=Left$(T$)
          If A$="-" Or A$="+"
            @Menu_nr(M%,M_titel%,M_pulldown%,*
                     M_punkt%)
            @Menu_text(M%,M_punkt%,Asc(A$)-43-
                       (Asc(A$)=43),"")
            T$=Mid$(T$,2)
          Endif
          Let Menu$(M%,M_titel%,M_pulldown%)=T$
        Endif
      Next M_pulldown%
    Next M_titel%
    Read T$
```

→

```
78:     Next M%
79:     For M%=0 To M_max%
80:       If Xbios(4)=2
81:         Restore L_m2groesse
82:       Else
83:         Restore L_m1groesse
84:       Endif
85:       Read M_text_size%(M%),M_tb%(M%),M_th%(M%),
                        M_tof%(M%)
86:       M_color%(M%)=1
87:       M_hidden!(M%)=False
88:     Next M%
89:     M_tx%=-1
90:     M_ty%=-1
91:     M_pulldown%=-1
92:     M_pd%=-1
93:     Clr M_titel%
94:   Return
95:   Procedure Menu_image(M%,Groesse%,Farbe%)
96:     ' Setzen von Textgroesse und Farbe der
        Menüleiste
97:     '
98:     Local I%
99:     L_m_groesse:
100:    Data 4,6,6,0
101:    L_m1groesse:
102:    Data 6,8,8,0
103:    L_m2groesse:
104:    Data 13,8,16,2
105:    Data 32,16,32,4
106:    '
107:    If M%>=0 And M%<=M_max%
108:      If Groesse%>=0
109:        Restore L_m_groesse
110:        Groesse%=Min(3,Groesse%)
111:        For I%=0 To Groesse%
112:          Read M_text_size%(M%),M_tb%(M%),
                M_th%(M%),M_tof%(M%)
113:        Next I%
114:      Endif
115:      If Farbe%>=0
116:        M_color%(M%)=Min(15,Farbe%)
117:      Endif
118:    Endif
119:  Return
120:  Procedure Menu_paint(M%,X1%,Y1%,X2%,Y2%,
            Richtung1!,Richtung2!,Versteckt!,
            Klick!)
121:    ' Übernahme der Koordinaten für Menüleiste
122:    ' und Menüleiste zeichnen ( falls nicht
        versteckt ! )
123:    '
124:    If M%>=0 And M%<=M_max%
125:      M_direction1!(M%)=Richtung1!
126:      M_direction2!(M%)=Richtung2!
127:      M_hidden!(M%)=Versteckt!
128:      M_klick!(M%)=Klick!
129:      M_aktiv!(M%)=True
130:      If M_direction1!(M%)
131:        M_y%(M%)=M_yres%
132:        M_y1%(M%)=0
133:        If Y1%>=0
134:          M_y%(M%)=Min(M_yres%,Max(Y1%,M_tb%(M%)*
                    2))
135:        Endif
136:        If Y2%>=0
137:          M_y1%(M%)=Min(Y1%,Y2%)
138:        Endif
139:        M_x%(M%)=Max(M_th%(M%),Min(M_xres%-2,X1%))
140:        If Not M_hidden!(M%)
141:          @M_paint_ver
142:        Endif
143:      Else
144:        M_x%(M%)=0
145:        M_x1%(M%)=M_xres%
146:        If X1%>=0
147:          M_x%(M%)=Min(X1%,M_xres%-M_tb%(M%)*2)
148:        Endif
149:        If X2%>=0
150:          M_x1%(M%)=Min(Max(X1%,X2%),M_xres%)
151:        Endif
152:        M_y%(M%)=Max(M_th%(M%),Min(M_yres%-2,Y1%))
153:        If Not M_hidden!(M%)
154:          @M_paint_hor
155:        Endif
156:      Endif
```

```
157:    Endif
158:  Return
159:  Procedure On_menu(Adresse1%,Adresse2%)
160:    ' Fragt Menüleisten ab und gibt die Nummer des
        Menüs und
161:    ' eines evtl. angeklickten Menüpunktes zurück
162:    '
163:    Local Mn%,M%,Mx%,My%,Mk%,M_end!
164:    Mouse Mx%,My%,Mk%
165:    Clr M_end!,M_action!
166:    If Menu_nr%>-1 And (Not M_pulldown! And
           M_invt!)
167:      @Menu_off(Menu_nr%)
168:    Endif
169:    For M%=0 To M_max%
170:      If M_aktiv!(M%)
171:        If M_pd%=M% Or M_pd%=-1
172:          Mn%=-1
173:          If M_pulldown!
174:            M_action!=True
175:            If Mx%>M_pdx% And Mx%<M_pdx1% And
                   My%>M_pdy% And My%<M_pdy1%-1
176:              M_pnr%=(My%-M_pdy%-1)/M_th%(M%)+1
177:              If Menu_style%(M%,M_titel%,M_pnr%)=0
                     Or Menu_style%(M%,M_titel%,
                     M_pnr%)=3
178:                If M_nr%<>M_pnr%
179:                  If M_punkt!
180:                    Put Max(0,M_pdx%+1),Max(0,
                              M_py%),M_punkt$,10
181:                  Endif
182:                  M_py%=(M_pnr%-1)*M_th%(M%)+
                            M_pdy%+1
183:                  M_py1%=M_py%+M_th%(M%)
184:                  Get Max(0,M_pdx%+1),Max(0,M_py%),
                          Min(M_xres%,M_pdx1%-1),
                          Min(M_yres%,M_py1%),
                          M_punkt$
185:                  Put Max(0,M_pdx%+1),Max(0,M_py%),
                          M_punkt$,10
186:                  M_punkt%=M_pnr%
187:                  M_nr%=M_pnr%
188:                  M_punkt!=True
189:                Endif
190:                If Mk%=1
191:                  @Menu_nr(M%,M_titel%,M_punkt%,*
                            Mn%)
192:                  @Menu_pull_down_back
193:                  M_end!=True
194:                  Goto L_on_menu_end
195:                Endif
196:              Else
197:                Goto L_mpunkt_invert
198:              Endif
199:            Else
200:              L_mpunkt_invert:
201:              If M_punkt!
202:                Put Max(0,M_pdx%+1),Max(0,M_py%),
                          M_punkt$,10
203:                Clr M_punkt!,M_nr%,M_pnr%,M_punkt%
204:              Endif
205:            Endif
206:          Endif
207:          If M_direction1!(M%)
208:            If My%>=M_y1%(M%) And My%<=M_y%(M%)
                   And Mx%>=M_x%(M%)-M_th%(M%)
                   And Mx%<=M_x%(M%)+2
209:              M_action!=True
210:              If M_hidden!(M%) And Not M_back!(M%)
211:                @M_paint_ver
212:              Endif
213:              If M_pulldown! Or (Mk%=1 And
                     M_klick!(M%)) Or Not
                     M_klick!(M%)
214:                Y%=M_y%(M%)
215:                For M_t%=0 To M_maxtitel%(M%)
216:                  Sub Y%,Len(Menu$(M%,M_t%,0))*
                          M_tb%(M%)
217:                  Exit If Y%<My%
218:                Next M_t%
219:                Y%=Max(0,Y%)
220:                If Y%<My%
221:                  If Y%<>M_ty%
222:                    @Menu_inv(M%)
223:                    Get M_x%(M%)-M_th%(M%)-
                            M_direction2!(M%),Y%,  →
```

```
                        M_x%(M%)+1-
                        M_direction2!(M%),Y%+
                        Len(Menu$(M%,M_t%,0))*
                        M_tb%(M%),Inv_titel$
224:                  Put M_x%(M%)-M_th%(M%)-
                        M_direction2!(M%),Y%,
                        Inv_titel$,10
225:                  M_ty%=Y%
226:                  M_titel%=M_t%
227:                  M_invt!=True
228:                Endif
229:              Endif
230:            Endif
231:          Else
232:            Goto L_pulldown_delete
233:          Endif
234:        Else
235:          If My%>=M_y%(M%)-M_th%(M%) And
                My%<=M_y%(M%)+2 And
                Mx%>=M_x%(M%) And
                Mx%<=M_x1%(M%)
236:            M_action!=True
237:            If M_hidden!(M%) And Not M_back!(M%)
238:              @M_paint_hor
239:            Endif
240:            If M_pulldown! Or (Mk%=1 And
                  M_klick!(M%)) Or Not
                  M_klick!(M%)
241:              X%=M_x%(M%)
242:              For M_t%=0 To M_maxtitel%(M%)
243:                Add X%,Len(Menu$(M%,M_t%,0))*
                      M_tb%(M%)
244:                Exit If X%>Mx%
245:              Next M_t%
246:              If X%>Mx%
247:                Sub X%,Len(Menu$(M%,M_t%,0))*
                        M_tb%(M%)
248:                If X%<>M_tx%
249:                  @Menu_inv(M%)
250:                  Get X%,Max(0,M_y%(M%)-
                        M_th%(M%)-
                        M_direction2!(M%)),
                        Min(M_xres%,
                        Min(M_xres%,X%+
                        Len(Menu$(M%,M_t%,0))*
                        M_tb%(M%))),
                        Min(M_yres%,M_y%(M%)+1-
                        M_direction2!(M%)),
                        Inv_titel$
251:                  Put X%,Max(0,M_y%(M%)-
                        M_th%(M%)-
                        M_direction2!(M%)),
                        Inv_titel$,10
252:                  M_tx%=X%
253:                  M_titel%=M_t%
254:                  M_invt!=True
255:                Endif
256:              Endif
257:            Endif
258:          Else
259:            L_pulldown_delete:
260:            If M_paint!(M%) And ((M_pulldown!
                  And Mk%=1) Or Not
                  M_pulldown!)
261:              @Menu_pull_down_back
262:              @Menu_off(M%)
263:              Clr M_titel%,M_invt!
264:              M_back!(M%)=False
265:              M_ty%=-1
266:              M_tx%=-1
267:            Endif
268:          Endif
269:        Endif
270:        If M_invt! And M_titel%<>M_pulldown%
271:          @Menu_pull_down
272:        Endif
273:        L_on_menu_end:
274:        *Adresse1%=M%
275:        *Adresse2%=Mn%
276:      Endif
277:    Endif
278:    Exit If M_end!
279:  Next M%
280: Return
281: Procedure Menu_text(M%,M_n%,S%,T$)
282:   ' Ändert Menüeinträge
283:   '
284:   Local I%,N%,Mn%
285:   If M%>=0 And M%<=M_max%
286:     Clr Mn%
287:     For I%=0 To M_maxtitel%(M%)
288:       For N%=0 To M_maxpulldown%(M%)
289:         Exit If Mn%=M_n%
290:         Sub Mn%,Menu$(M%,I%,N%)<>""
291:       Next N%
292:       Exit If Mn%=M_n%
293:     Next I%
294:     If Mn%=M_n%
295:       If S%>=0
296:         Let Menu_style%(M%,I%,N%)=S%
297:       Endif
298:       If T$<>""
299:         Let Menu$(M%,I%,N%)=T$
300:       Endif
301:     Endif
302:   Endif
303: Return
304: Procedure Menu_kill(M%)
305:   ' Löscht angegebene Menüleiste
306:   '
307:   If M%>=0 And M%<=M_max%
308:     If M_direction1!(M%)
309:       Put M_x%(M%)-M_th%(M%),M_y1%(M%),
              M_back$(M%),3
310:     Else
311:       Put M_x%(M%),M_y%(M%)-M_th%(M%),
              M_back$(M%),3
312:     Endif
313:     Clr M_titel%,M_invt!
314:     M_paint!(M%)=False
315:     M_back!(M%)=False
316:   Endif
317: Return
318: Procedure Menu_lock(M%,Lock!)
319:   ' Sperrt Menübearbeitung, oder hebt Sperre auf
320:   '
321:   M_aktiv!(M%)=Not Lock!
322: Return
323: '
324: ' **** Interne Routinen für Menü-Library ****
325: '
326: Procedure Menu_inv(M%)
327:   If M_invt!
328:     If M_direction1!(M%)
329:       Put Max(0,M_x%(M%)-M_th%(M%)-
              M_direction2!(M%)),M_ty%,
              Inv_titel$,10
330:       M_ty%=-1
331:     Else
332:       Put M_tx%,Max(0,M_y%(M%)-M_th%(M%)-
              M_direction2!(M%)),Inv_titel$,10
333:       M_tx%=-1
334:     Endif
335:     Clr M_titel%,Inv_titel$,M_invt!
336:   Endif
337: Return
338: Procedure Menu_off(M%)
339:   @Menu_inv(M%)
340:   If M_hidden!(M%)
341:     @Menu_kill(M%)
342:   Endif
343: Return
344: Procedure M_leiste_vorbereiten
345:   Clr M_leiste$
346:   For M_titel%=0 To M_maxtitel%(M%)
347:     M_leiste$=M_leiste$+Menu$(M%,M_titel%,0)
348:   Next M_titel%
349:   M_paint!(M%)=True
350:   M_back!(M%)=True
351:   Graphmode 1
352:   Color 1
353: Return
354: Procedure M_paint_hor
355:   @M_leiste_vorbereiten
356:   M_x1%(M%)=Min(M_xres%,Max(M_x%(M%)+
            Len(M_leiste$)*M_tb%(M%),M_x1%(M%)))
357:   Get M_x%(M%),M_y%(M%)-M_th%(M%),M_x1%(M%),
          M_y%(M%)+2,M_back$(M%)
358:   Put M_x%(M%),M_y%(M%)-M_th%(M%),M_back$(M%),0
359:   Deftext M_color%(M%),0,0,M_text_size%(M%)
360:   Text M_x%(M%),M_y%(M%)-M_tof%(M%),M_leiste$
361:   If M_direction2!(M%)
```

→

```
362:        Line M_x%(M%),M_y%(M%)-M_th%(M%),M_x1%(M%),
                M_y%(M%)-M_th%(M%)
363:      Else
364:        Line M_x%(M%),M_y%(M%)+2,M_x1%(M%),M_y%(M%)+2
365:      Endif
366:    Return
367:    Procedure M_paint_ver
368:      @M_leiste_vorbereiten
369:      M_y1%(M%)=Max(0,Min(M_y%(M%)-Len(M_leiste$)*
                  M_tb%(M%),M_y1%(M%)))
370:      Get M_x%(M%)-M_th%(M%),M_y1%(M%),M_x%(M%)+2,
              M_y%(M%),M_back$(M%)
371:      Put M_x%(M%)-M_th%(M%),M_y1%(M%),M_back$(M%),0
372:      Deftext M_color%(M%),0,900,M_text_size%(M%)
373:      Text M_x%(M%)-M_tof%(M%),M_y%(M%),M_leiste$
374:      If M_direction2!(M%)
375:        Line M_x%(M%)-M_th%(M%),M_y%(M%),M_x%(M%)-
                M_th%(M%),M_y1%(M%)
376:      Else
377:        Line M_x%(M%)+2,M_y%(M%),M_x%(M%)+2,M_y1%(M%)
378:      Endif
379:    Return
380:    Procedure Menu_pull_down
381:      Local I%,Mn%
382:      @Menu_pull_down_back
383:      Clr I%,M_pdb%
384:      Repeat
385:        Inc I%
386:        M_pdb%=Max(M_pdb%,Len(Menu$(M%,M_titel%,I%))*
                  M_tb%(M%))
387:      Until I%>=M_maxpulldown%(M%) Or Menu$(M%,
                  M_titel%,I%)=""
388:      Inc M_pdb%
389:      Add I%,Menu$(M%,M_titel%,I%)=""
390:      M_pdl%=I%*M_th%(M%)+2
391:      If M_direction1!(M%)
392:        If M_direction2!(M%)
393:          M_pdx1%=M_x%(M%)-M_th%(M%)
394:          M_pdx%=M_pdx1%-M_pdb%
395:        Else
396:          M_pdx%=M_x%(M%)+2
397:          M_pdx1%=M_pdx%+M_pdb%
398:        Endif
399:        M_pdy1%=M_ty%+Len(Menu$(M%,M_titel%,0))*
                  M_tb%(M%)
400:        M_pdy%=M_pdy1%-M_pdl%
401:      Else
402:        M_pdx%=M_tx%
403:        M_pdx1%=M_tx%+M_pdb%
404:        If M_direction2!(M%)
405:          M_pdy1%=M_y%(M%)-M_th%(M%)
406:          M_pdy%=M_pdy1%-M_pdl%
407:        Else
408:          M_pdy%=M_y%(M%)+2
409:          M_pdy1%=M_pdy%+M_pdl%
410:        Endif
411:      Endif
412:      Y%=M_pdy%-M_tof%(M%)
413:      M_pulldown%=M_titel%
414:      M_pulldown!=True
```

```
415:      M_pd%=M%
416:      Graphmode 1
417:      Get Max(0,M_pdx%),Max(0,M_pdy%),Min(M_xres%,
              M_pdx1%),Min(M_yres%,M_pdy1%),
              M_pulldown$
418:      Deffill 1,0,0
419:      Pbox M_pdx%,M_pdy%,M_pdx1%,M_pdy1%
420:      For Mn%=1 To I%
421:        Add Y%,M_th%(M%)
422:        If Menu_style%(M%,M_titel%,Mn%)<3
423:          If Menu_style%(M%,M_titel%,Mn%)=2 And
                  M_text_size%(M%)=4
424:            Deftext M_color%(M%),3,0,M_text_size%(M%)
425:          Else
426:            Deftext M_color%(M%),Menu_style%(M%,
                      M_titel%,Mn%),0,M_text_size%(M%)
427:          Endif
428:          Text M_pdx%+1,Y%,Menu$(M%,M_titel%,Mn%)
429:        Else
430:          If Menu_style%(M%,M_titel%,Mn%)=3
431:            Deftext M_color%(M%),0,0,M_text_size%(M%)
432:          Else
433:            If M_text_size%(M%)=4
434:              Deftext M_color%(M%),3,0,
                        M_text_size%(M%)
435:            Else
436:              Deftext M_color%(M%),2,0,
                        M_text_size%(M%)
437:            Endif
438:          Endif
439:          Text M_pdx%+1,Y%,Menu$(M%,M_titel%,Mn%)
440:          Text M_pdx%+2,Y%,""
441:        Endif
442:      Next Mn%
443:    Return
444:    Procedure Menu_pull_down_back
445:      If M_pulldown!
446:        Put Max(0,M_pdx%),Max(0,M_pdy%),M_pulldown$,3
447:        M_pulldown%=-1
448:        M_pd%=-1
449:        Clr M_punkt!,M_pulldown!,M_pnr%,M_nr%
450:      Endif
451:    Return
452:    Procedure Menu_nr(M%,M_t%,M_n%,Adresse%)
453:      Local I%,N%,Nm%
454:      If M%>=0 And M%<=M_max%
455:        Clr Nm%
456:        For I%=0 To M_maxtitel%(M%)
457:          For N%=0 To M_maxpulldown%(M%)
458:            Exit If I%=M_t% And N%=M_n%
459:            Sub Nm%,Menu$(M%,I%,N%)<>""
460:          Next N%
461:          Exit If I%=M_t% And N%=M_n%
462:        Next I%
463:        If I%>M_maxtitel%(M%)
464:          Clr Nm%
465:        Endif
466:        *Adresse%=Nm%
467:      Endif
468:    Return
```


QUAZAR

Software Entwicklungs-gesellschaft mbH
Papenhuder Straße 2
2000 Hamburg 76
Telefon 040 / 22 77 66 2

DM 149,-
Atari ST GEM-Toolbox
Zeit sparen beim Softwareentwickeln. Benutzen Sie unsere Toolbox. Exceptionhandler, Rechenfunktionen, Uhrzeit / Datum, Fehlersuche, Disk I/O, Dialogtools, erweitertes 'form_do' Datenbankmasken, Tastatur, Bildschirm. Ca. 100 neue Funktionen.

DM 98,-
Atari Maskenlibrary
MS-DOS kompatibel Entwickeln Sie leicht portierbare MS-DOS Programme auf dem Atari ST. Alle Funktionen auf TOS-Ebene, IBM Rahmengrafik, Maskeneditor, Feldtypen.

DM 1.498,-
PAM's Netzwerk Knoten
Atari-Netzwerk für alle Datenbankanwendungen, voller Multiuserbetrieb ohne Benutzerauflagen, Peripherie-Geräte gemeinsam nutzbar, mehrere Festplatten von allen Arbeitsplätzen ansprechbar. Wir sind Vertriebspartner von PAM Software, Mainz

DM 198,-
Midi Netzwerk
Netzwerk für Softwareentwickler. Der lästige Diskettentausch entfällt. Keine Hardware erforderlich, einfach Treiberprogramm als ACC installieren, Einbindung über Laufwerk 'N' auf dem Desktop, Ansprechbar wie eine RAM-Disk, Einfache Vernetzung über 5-Pol DIN Kabel.

DM 98,-
PAM's Net Testutility
Testen Sie Ihre Netzwerkinstallation. Manuelles File/Record Locking, Lock-Status Überwachung, Message-Test.

Wir entwickeln Software und beraten Softwareentwickler, bei der Erstellung netzwerkfähiger Programme auf dem Atari.

Eigener Desktop
in Pascal plus

Wolfram Püchert

Jeder kennt inzwischen das Einheitsgrau des Desktops von ATARI. Mit den zwei folgenden Prozeduren kann sich jeder PASCAL ST plus-Programmierer ein eigenes Desktop in seinen Programmen erstellen.

Mit dem Pascal-Compiler Pascal ST plus von CCD hat man ein gutes, einfach zu bedienendes Programmierwerkzeug zur Hand. Besonders die GEM-Bibliothek ist den Machern gut gelungen. Die einfache Handhabung macht das Umgehen mit GEM und mit seinen Oberflächen zum Kinderspiel. Doch wie jedes Ding hat Pascal ST plus auch seine Schattenseiten. Eine davon ist das Erzeugen und Verwalten eines eigenen DESKTOPs.

Installation des Desktops

Zuerst baue man einen Dialog auf (via *New_dialog*) und lege das Aussehen seiner Einträge fest (Zeile 86-94). Da die Menüzeile ausgespart wird, ist es ratsam, den Wert 1 als y-Koordinate der Prozedur New_dialog zu übergeben. Will man seinen DESKTOP mit einer Rasterung unterlegen, so addiere man als erstes einen Eintrag vom Typ G_BOX (hier: Unterlage Zeile 89) hinzu. Bei Unterlassung erhält man eine weiße Fläche als Hintergrund. Danach kann ganz normal der Dialog aufgebaut werden. Allerdings sollten nur folgende Typen verwendet werden: G_BOX, G_TEXT, G_BOXTEXT, G_IMAGE, G_USERDEF, G_STRING, G_ICON.

Ist der Aufbau erledigt, schlägt die Prozedur *EIGENES_DESKTOP* zu, der nur der Zeiger auf den Dialog übergeben werden muß (Zeile 20). GEM erkennt ein neues Desktop an der Kennung *DESKTOP=0* (Zeile 21) und an dem Parameter *WF_NEWDESKTOP=14*, die der Prozedur Wind_set übergeben werden. Diese Prozedur kann zum Glück verwendet werden. Die Compilerprogrammierer haben sie so übernommen, wie es im GEM vorgesehen ist, das heißt: Nach dem Parameter WF_NEWDESKTOP erwartet GEM eine Adresse (4 Byte) - aufgespalten in high und low (je 2 Bytes) - auf eine Objektstruktur. Da Pascal aber nur mit Zeigern arbeitet, muß man ein wenig tricksen, um an diese Adresse heranzukommen. Die varianten Records helfen dabei (Zeile 40). Man übergebe *adresse.z* den Wert aus *dia* - intern ist ein Zeiger nichts anderes als eine Adresse (=long_integer) - und in *adresse.l* steht nun die tatsächliche Adresse. Die Aufspaltung in low und high habe ich wieder über einen varianten Record vorgenommen (Achtung: In Version 2.0 muß die Compileroption I- gesetzt sein); es geht natürlich auch anders. Somit sind alle Werte für die *Wind_set*-Prozedur vorhanden. Nach dem Aufruf weiß GEM: 'Jetzt wird ein neues Desktop installiert'. Über die *FORM_DIAL*-Routine (AES-Nr. 51) wird dieses auf den Bildschirm gebracht (Zeile 42-45). Fertig ist unser Desktop. Die Verwaltung übernimmt jetzt GEM. Man kann sich mehrere Desktops erschaffen und diese abwechselnd auf die eben beschrieben Weise auf den Bildschirm bringen. Jeweils der letzte so übergebene Dialog wird als DESKTOP verwaltet.

Ändern

Manchmal ist es notwendig, Teile des Desktops zu verändern (Status, Text). Genau das macht die Prozedur *DESKTOP_AENDERN* (Zeile 47). Die Schwierigkeit dabei ist, daß auch zur Zeit durch Fenster etc. verdeckte Einträge geändert werden müssen, ohne daß diese Änderung sofort auf dem Bildschirm sichtbar ist. Doch dazu gleich. Erst zu den Übergabeparametern: Übergeben werden Desktopdialogzeiger; Kennung des zu ändernden Eintrags; der (evtl. leere) Text; Schriftart; Ausrichtung; ein bool'scher Wert, der angibt, ob Text geändert werden soll; Breite und Höhe des Eintrags in Pixel. Über *First_rect* und *Next_rect* werden die sichtbaren Rechtecke des Desktops (Parameter 0) ermittelt, über *Obj_offset* (AES-Nr. 44) die x-/y-Koordinaten des zu ändernden Eintrags, über *Rect_intersect* die Schnittmenge der

beiden Flächen (hier sind die Höhe und die Breite des Eintrags wichtig). Diese Schnittmenge wird als Ausgabebereich definiert [*Set_clip* (Zeile 76)], und dann wird mit Obj_draw (AES-Nr. 42) neu gezeichnet. Der *Set_clip*-Aufruf in Zeile 76 verhindert, daß ein Eintrag gezeichnet wird, der gerade verdeckt ist. Damit GEM aber von der Veränderung etwas mitbekommt, muß auf jeden fall der AES-Aufruf gemacht werden. Mit *Set_clip(0,0,640,400)* (Zeile 78) wird als Ausgabebereich der ganze Bildschirm wieder freigegegen. Jetzt steht dem eigenen Desktop hoffentlich nichts mehr im Wege.Viel Spaß bei der Programmiererei.

```
{-------------------------------------------------}
{--- Erstellen eines selbstdefinierten DESKTOP ---}
{-------------------------------------------------}
{----- Geschrieben von:                      -----}
{-----                                       -----}
{----- WOLFRAM PüCHERT                       -----}
{----- 6570 Kaiserslautern                   -----}
{----- Wilhelm - Raabe - StraPe 26           -----}
{-----                                       -----}
{----- 2.12.1987 ST Pascal plus 1.2/2.0      -----}
{----- (c) MAXON Computer GmbH               -----}
{-------------------------------------------------}

 1: Program Eigenes_Desktop_erstellen;
 2: CONST    {$I GEMCONST}
 3: TYPE     {$I GEMTYPE}
 4:          DESKITEM         = RECORD
 5:                unterlage : integer;
 6:                texte     : ARRAY[1..10] of integer;
 7:                kasten    : ARRAY[1..10] of integer;
 8:                            END;
 9: VAR      deskdia          : DIALOG_PTR;
10:          deskeintrag      : DESKITEM;
11:          i                : integer;
12: {$I GEMSUBS}
13: Procedure Status_aendern(VAR dia      : DIALOG_PTR;
14:                              eintrag  : TREE_INDEX;
15:                              status   : integer);
16: BEGIN
17: Obj_setstate(dia,eintrag,NORMAL,true);
18: Obj_setstate(dia,eintrag,status,true)
19: END;                        {Prozedur Status_aendern}
20: Procedure Eigenes_desktop(VAR dia : DIALOG_PTR);
21: CONST    DESKTOP       = 0;
22:          WF_NEWDESKTOP = 14;
23: TYPE     ZEIGER_ADRESSE= RECORD
24:                          CASE boolean OF
25:                            true : (z:^char);
26:                            false: (l:long_integer);
27:                            END;
28:          INTEGER_LONG  = RECORD
29:                          CASE boolean OF
30:                            false: (l:long_integer);
31:                            true : (i:PACKED ARRAY
                                     [1..2] of integer);
32:                            END;
33: VAR      adresse          : ZEIGER_ADRESSE;
34:          long_int         : INTEGER_LONG;
35:          int_in           : ARRAY[0..15] of integer;
36:          int_out          : ARRAY[0..45] of integer;
37:          addr_in          : ARRAY[0..1] of ^char;
38:          addr_out         : ARRAY[0..0] of ^char;
39: BEGIN
40: adresse.z := dia;  long_int.l := adresse.l;
                {Arbeitsbereich für neues Desktop}
```

```
41: wind_set(DESKTOP,WF_NEWDESKTOP,long_int.i[1],
             long_int.i[2],0,0);
42: int_in[0] := 3;
43: int_in[1] := 0;  int_in[2] := 0;
    int_in[3] := 640;  int_in[4] := 400;
44: int_in[5] := 0;  int_in[6] := 0;
    int_in[7] := 640;  int_in[8] := 400; {Form_dial}
45: Aes_call(51,int_in,int_out,addr_in,addr_out);
46: END;                        {Prozedur Eigenes_desktop}
47: Procedure Desktop_aendern(VAR dia   : DIALOG_PTR;
48:                           eintrag   : TREE_INDEX;
49:                           neuer_text: STRING;
50:                           schrift   : integer;
51:                           justierung: TE_JUST;
52:                           neuer_status: integer;
53:                           text_aendern: boolean;
54:                           ebreite, ehoehe : integer);
55: VAR      x_koor, y_koor, breite, hoehe : integer;
56:          int_in          : ARRAY[0..15] of integer;
57:          int_out         : ARRAY[0..45] of integer;
58:          addr_in         : ARRAY[0..1] of ^char;
59:          addr_out        : ARRAY[0..0] of ^char;
60:          bool            : boolean;
61: BEGIN
62: Hide_mouse;  Begin_update;
63: First_rect(0,x_koor,y_koor,breite,hoehe);
64: WHILE (breite<>0) AND (hoehe<>0) DO
65: BEGIN
66:   IF text_aendern THEN
67:     Set_dtext(dia,eintrag,neuer_text,schrift,
                  justierung);
68:   Status_aendern(dia,eintrag,neuer_status);
                    {Koordinaten holen}
69:   int_in[0] := eintrag;  addr_in[0] := dia;
70:   Aes_call(44,int_in,int_out,addr_in,addr_out);
71:   int_in[1] := 0;
72:   int_in[2] := int_out[1]; int_in[3]:=int_out[2];
73:   int_in[4] := ebreite;  int_in[5] := ehoehe;
                  {Rechteckausschnitt einschränken}
74:   bool:= Rect_intersect(x_koor,y_koor,breite,hoehe,
75:          int_in[2],int_in[3],int_in[4],int_in[5]);
76:   Set_clip(int_in[2],int_in[3],int_in[4],
              int_in[5]);
77:   Aes_call(42,int_in,int_out,addr_in,addr_out);
78:   Set_clip(0,0,640,400);
79:   Next_rect(0,x_koor,y_koor,breite,hoehe);
80: END;                        {Solange noch Rechtecke}
81: End_update;  Show_mouse;
82: END;                        {Desktop_aendern}
83: BEGIN
84: IF Init_gem>=0 THEN
85: BEGIN
86:   deskdia := New_dialog(21,0,1,80,24);
87:   WITH deskeintrag DO
88:   BEGIN
89:     unterlage := Add_ditem(deskdia,G_BOX,NONE,0,0,
                                80,24,-1,4305);
90:     texte[1] := Add_ditem(deskdia,G_BOXTEXT,NONE,
                              2,2,40,2,3,4480);
91:     kasten[1] := Add_ditem(deskdia,G_BOX,NONE,2,
                     10,10,3,-2,4257);
        {Es können weitere Einträge erfolgen}
92:     Set_dtext(deskdia,texte[1],' Desktop neu !!!',
                  SYSTEM_FONT,TE_CENTER);
93:     Obj_setstate(deskdia,kasten[1],SHADOWED,false);
94:   END;
95:   Eigenes_desktop(deskdia);
96:   i := Do_alert('[1][ Sieh es Dir an !! | |]
                     [ weiter ]',1);
97:   Desktop_aendern(deskdia,deskeintrag.texte[1],
                      'Neuer Text !!',
98:                    SYSTEM_FONT,TE_CENTER,NORMAL,
                       true,40*8,2*16);
99:   i := Do_alert('[1][ Sieh es Dir an !! | |]
                     [ weiter ]',1);
100:  Exit_gem;
101: END;
```

Eingabe von Termen

Philip S. Bauer

Dieses Programm ist in der Lage, einen von der Tastatur eingegebenen mathematischen Term zu analysieren und zu berechnen. Es beherrscht die Addition, die Subtraktion, die Klammerrechnung, die Potenzrechnung, die Multiplikation, die Division und verarbeitet einfache komplexe Zahlen. Weiter kann es trigonometrische Funktionen berechnen. Dieser Parser setzt die eingegebenen Komponenten sehr schnell zum gewünsch-

Das Programm hat viele Anwendungsmöglichkeiten. Es kann sowohl von Schülern, Studenten und weniger versierten Anwendern, als auch von Fachleuten in den unterschiedlichsten Branchen eingesetzt werden und unterstützt die verschiedensten Rechenprobleme des Alltags. Die Bedienung des Interpreters ist dem des im Direktmodus laufenden GFA-BASIC sehr ähnlich. Es können alle Rechenoperationen, die im Startup-Menue angegeben sind, verwendet werden, z.B *,-,+ usw. Häufig vorkommende Zahlen in der Mathematik sind ebenfalls integriert, wie z.B "pi" und "e". Die aufgeführten Sonderfunktionen "sin,cos,abs..." können wie in der linken Reihe angegeben benützt werden. Hier einige Beispiele.

Diese Terme können wie folgt eingegeben werden:

1.) OK->1+1-6 oder
OK->(1+1)-6 oder
OK->((1+1)-+6)

2.) OK->2/9+6 oder
OK->(2/9)+6 oder
OK->((2/9)+6)

3.) OK->2*9+4^5 oder
OK->(2*9)+(4^5) oder
OK->((2*9)+4^5)

Der Interpreter beherrscht auch ohne Klammerrechnung die Regeln der Mathematik. Natürlich kann man sie durch Setzen von Klammern umgehen.

Es werden weiterhin die Vorzeichen berücksichtigt, die im Term vorkommen z.B :

```
OK->--5+6 oder
OK->5+-8
```

u.s.w siehe Beispiel oben. Einige Beispiele zu den Sonderfunktionen. Der sin(45) soll berechnet werden:

1.) OK->s(45) oder
OK->s45 (Ergebnis in Grad)

Alle anderen Sonderfunktionen werden wie bei "sin" eingegeben, nur mit dem ersten kleinen Buchstaben.

2.) OK->c(80)OK->t(80)
OK->w(6) usw.

Sonderfunktionen kombinieren

Das ist aber noch nicht alles, was das Programm kann! Alle Sonderfunktionen können natürlich auch beliebig miteinander kombiniert werden:

z.B: OK->a(c(45)^2+s(45)^2)
usw.

Der Interpreter prüft, ob ein Term richtig eingegeben ist, indem er einen kompletten "Syntax-Check" durchführt. Auftretende Fehler werden größtenteils abgefangen.

z.B: OK->-*+ oder
OK->1/0 oder
OK->(((((((((1+6) usw.

Es können auch Variabeln definiert werden, und zwar alle GROSSBUCHSTABEN von A-Z, das sieht dann so aus :

```
OK->A=5 oder
OK->D=s(7)+5*9 oder
OK->L=d(7) usw.
```

Eine weitere Attraktion weist der Parser auf: Er ist in der Lage, einfache komplexe Zahlen zu verarbeiten. Es ist möglich, eine negative Wurzel oder einen negativen Logarithmus zu berechnen.

Beispiel: es soll die 2.te Wurzel aus (-8) berechnet werden

```
OK->w(-8) oder
OK->(-8)^(1/2) oder
OK->-8^(1/2)
```

Es erscheint eine komplexe Lösung, die aus einer imaginären und reellen Lösung besteht. Es soll eine 8. Wurzel aus einer negativen Zahl berechnet werden.

Das sieht dann so aus:

```
OK->(-5)^(1/8) usw.
```

Es können alle negativen Zahlen mit allen Elementen der natürlichen Zahlen potenziert werden: Vorsicht: den Exponenten nicht als Dezimalzahl schreiben, (-8)^(0.5) z.B. führt zum Überlauf!

```
OK->(-21)^(5/16) usw.
```

Eine Logarithmierung einer negativen Zahl ist ebenfalls möglich, das sieht dann so aus:

```
OK->l(-9) oder
OK-> d (-9) oder
OK->d-9.
```

Klammern sind in diesem Fall auch nicht nötig. Es erscheinen komplexe Lösungen von z1 - zn, die aus imaginären und reellen Lösungen bestehen. Ich hoffe, daß einige Anwender dieses Programm verwerten können. Mit einem kleinen Tip möchte ich diese Dokumentation zu meinem Programm schließen: Der Parser-Interpreter macht sich gut in einem Tabellenkalkulations-Programm!

```
' ------------------------------------------------
' Parse Interpreter written by P.S. Bauer,
  Romberstr.8 , 5000 Köln 41
' Version 4.00                    08.05.89
  (c) MAXON Computer GmbH
' ------------------------------------------------
DIM var$(60)
SETCOLOR 0,0
DO                          ! Menu Steuerung
  @text
  v=INP(2)                  ! Tastaturklick
  IF v=27                   ! ESC Taste
    SETCOLOR 0,1
    END
  ENDIF
  CLS                       ! Bild Löschen
  DO
    ADD pose,3              ! Icon Lage OK=>
    IF pose>=24             ! Letzte Zeile
      pose=3                ! 3 Addieren (Zeile+3)
      CLS                   ! Bild Löschen
      IF LEN(re$)>0
        PRINT AT(1,2+pose); ! Icon setzen
        PRINT "OK >"
        PRINT AT(5,2+pose);unt$      ! Term
        PRINT re$                    ! Ergebnis
                            falls schon vorhanden
        pose=pose+3
      ENDIF
    ENDIF
    @eingabe                ! Zur Hauptschleife
    EXIT IF ASC(re$)=27! Esc
  LOOP
  CLR re$                   ! Ergebnis Löschen
  pose=0                    ! zeile =0
  CLS
LOOP
PROCEDURE eingabe           ! Hauptprozedur
  GRAPHMODE 4
  DEFTEXT 1,0,0,13
  TEXT 5,15,"        Bitte Term Eingeben       "
  TEXT 5,31,"           mit Esc zurück         "
  GRAPHMODE 0
  ERASE sx$()
  DIM sx$(80)               !Feld vorbereiten
  nrt:
  DEFFILL 1,2,8             ! Daten für falsche Syntax
  tabe%=0                   ! Syntax
   Flag=0
  sqo:
  DATA "*)","(*","(/","(^","()","**","//","ss",
       "ww","cc","tt","^^","sw","tw","cw"
  DATA "/)","^)","/)","s)","w)","c)","t)","+)","-
       )","..",".)","(.",".(",")."
  DATA "-.","./","s.",".s",".w",".c",".t","w.",
       "c.","t.","+.","*.","/.","^.",".^"
  DATA ".-",".+",".*","./","a)","i)",".a",".i",
       ")a",")i","ia","is","ic","it","iw"
  DATA "aa","ii","sa","ca","ta","wa","i+","i-",i*
       ","i^","i/","a-","a+","a*","a/","a^"
  DATA "d*","d/","d)",".d","d.",")d","di","da",
       "ds","dc","dt","dw","d^"
  DATA "l+","l*","l/","l)",".l","l.",")l","li",
       "la","ls","lc","lt","lw"
  DATA "l^","ld","/*","/^"
  PRINT AT(1,2+pose);  ! Icon positonieren
  PRINT "OK >"
  PRINT AT(5,2+pose);
  FORM INPUT 40,re$         ! Term Eingeben
  unt$=re$                  ! Unt$=term
  IF INSTR(1,re$,"=")>0! Variablen Untersuchen
    IF LEN(re$)>=1 AND ASC(re$)<65 OR ASC(re$)>90
      tabe%=1
    ENDIF                   ! Var zuordnen?
    lio%=INSTR(1,re$,"=")+1
    IF INSTR(1,re$,"=")>0 AND ASC(re$)>=65 AND
       ASC(re$)<=90
    IF LEN(STR$(VAL(MID$(re$,lio%,
       LEN(re$)))))=LEN(MID$(re$,lio%,
       LEN(re$))) AND VAL(MID$(re$,lio%,
       LEN(re$)))>0
      @variable
      re$=MID$(re$,lio%,LEN(re$))  ! String
                                     erneuern
    ELSE
      varflag%=1            ! Variable Flag Setzen
      varterm$=MID$(re$,0,lio%-1)
                            ! Termvariable Bestimmen
      re$=MID$(re$,lio%,LEN(re$))  ! Term
                             Retten wenn Var
                             Vorhanden
    ENDIF
  ELSE
    tabe%=1                 ! Oh oh Fehler
  ENDIF
ENDIF
IF ASC(re$)<>27             ! ESC untersuchen
  IF LEN(re$)>0             ! Exp und E unterschieden
    CLR anzklam%,anzklam2%
    IF INSTR(0,re$,"E")>0
      exop$=MID$(re$,INSTR(0,re$,"E")-1,1)
    ENDIF
    FOR varwo%=1 TO LEN(re$)
      FOR wova%=ASC("A")-65 TO ASC("Z")-65
        obvar%=INSTR(varwo%,re$,CHR$(wova%+
               65))  !wo ist eine variable ?
        IF INSTR(0,re$,"E")>0 AND
           ASC(exop$)>ASC("0") AND
           ASC(exop$)<=ASC("9")
          obvar%=0
        ENDIF
        IF obvar%>0  ! Wert in Var einsetzen
          re$=MID$(re$,0,obvar%-1)+var$(wova%)+
              MID$(re$,obvar%+1,LEN(re$))
        ENDIF
      NEXT wova%
    NEXT varwo%
    CLR lop$
    pei%=INSTR(wobin%,re$,"pi") !wo ist pi ?
    IF pei%>0
      re$=MID$(re$,0,pei%-1)+STR$(PI)+MID$(re$,
          pei%+2,LEN(re$))
    ENDIF
    expo%=INSTR(wobin%,re$,"e") ! wo ist e
    IF expo%>0
      re$=MID$(re$,0,expo%-1)+STR$(EXP(1))+
          MID$(re$,expo%+1,LEN(re$))
    ENDIF
    IF INSTR(1,re$," ")>0 ! Leerstellen suchen
      FOR spce%=1 TO LEN(re$)
        ase$=MID$(re$,spce%,1)
        IF ase$<>CHR$(32) ! selbe entfernen
          f$=f$+ase$
        ENDIF
      NEXT spce%
      re$=f$
    ENDIF
    FOR synklam%=0 TO LEN(re$)
                ! Anzahl Klammern Überprüfen
      IF MID$(re$,synklam%+1,1)="("      !(
        ADD anzklam%,1     !( ist gefunden
      ENDIF
      IF MID$(re$,LEN(re$)-synklam%,1)=")"   !)
        ADD anzklam2%,1    !) ist gefunden
      ENDIF
    NEXT synklam%
    IF anzklam%<>anzklam2%! Klammer vergessen ?
      tabe%=1               ! Syntax Flag=1
    ENDIF
    IF LEN(re$)>0           ! Syntax prüfen
      FOR pol%=1 TO LEN(re$)
        polx%=ASC(MID$(re$,pol%,1))
```

→

```
                                ! String aufspalten
127:            IF polx%>57 OR polx%<=36 AND polx%<>94
                   AND polx%<>46 AND polx%<>115
                   AND polx%<>99 AND polx%<>116
                   AND polx%<>119 AND polx%<>105
                   AND polx%<>97 AND polx%<>100
                   AND polx%<>108 AND polx%<>69
128:              tabe%=1           ! Syntaxfehler Flag=1
129:            ENDIF
130:            EXIT IF polx%>57 OR polx%<=36 AND
                 polx%<>94 AND polx%<>46 AND
                 polx%<>115 AND polx%<>99 AND
                 polx%<>116 AND polx%<>119 AND
                 polx%<>105 AND polx%<>97 AND
                 polx%<>100 AND polx%<>108 AND
                 polx%<>69
131:          NEXT pol%
132:          RESTORE sqo
133:          FOR g=0 TO 107        ! Syntaxdata Einlesen
134:            READ syntax$
135:            IF INSTR(1,re$,syntax$)>0
136:              IF syntax$<>"/0"
137:                tabe%=1         ! Falls Fehler Syntax
                    Flag=1
138:              ENDIF
139:            ENDIF
140:            EXIT IF INSTR(1,re$,syntax$)>0
141:          NEXT g
142:          IF tabe%=1            ! Syntax Anzeigen
143:            PRINT "Syntax Fehler "
144:            ADD pose,3          ! nächste Position
145:            GOTO nrt            ! Neu eingeben
146:          ENDIF
147:          @fidel
148:          IF INSTR(1,re$,")")>0 AND INSTR(1,re$,
                 "(")>0             ! Klammer erkennen
149:            hint%=LEN(re$)
150:            c$=re$
151:            DO
152:              IF INSTR(1,c$,"(")>0 AND INSTR(1,c$,
                     ")")>0         ! Klammer
                   auswerten
153:                suche%=LEN(c$)
154:                DO              ! Klammer von
                                      Innen nach Aussen
155:                  SUB suche%,1
156:                  hint%=INSTR(suche%,c$,"(")
157:                  EXIT IF hint%>0
158:                LOOP
159:                CLR suche%
160:                tor%=INSTR(hint%+1,c$,")")
                                  ! ) Entfernen
161:                re$=MID$(c$,hint%+1,tor%)
162:                tor2%=INSTR(1,re$,")")
                                  ! ( Entfernen
163:                re$=MID$(re$,1,tor2%-1)
                                  ! String ohne ()
164:              ENDIF
165:              @fidel
166:              @num
167:              c$=MID$(c$,0,hint%-1)+
                     SPACE$(LEN(re$))+MID$(c$,
                     tor%+1,LEN(c$))
168:              MID$(c$,hint%,LEN(re$))=re$
                     ! String zusammenbauen
169:              EXIT IF INSTR(1,c$,")")=0 AND
                                INSTR(1,c$,"(")=0
170:            LOOP
171:            re$=c$        !String retten
172:          ENDIF
173:          @fidel
174:          @num
175:          DEFFILL 1,2,8
176:          IF varflag%=1         ! Termflag abfrage
177:            var$(ASC(varterm$)-65)=re$
                                  ! Termvar setzen
178:            varflag%=0          ! Flag auf Null
179:          ENDIF
180:          IF nuuld%=0 AND llog%=0
181:            PRINT re$           ! und ausgeben
182:          ENDIF
183:          CLR nuuld%,llog%
184:        ENDIF
185:      ENDIF
186:    ENDIF
187:    DEFFILL 1,2,8
```

```
188:    CLR sc$,rb$
189:    ERASE sx$()
190:  RETURN
191:  PROCEDURE num            ! Hier werden die
                                 Rechnungen ausgeführt
192:    tabe%=0
193:    DO
194:      EXIT IF INSTR(1,re$,"i")=0  ! Integer
195:      ADD w,1
196:      IF sx$(w)="i"         ! ist i dabei
197:        eg=LEN(sx$(w+1))
198:        isr=INSTR(1,re$,"i")  ! wo ist i
199:        we=1+LEN(sx$(w+1))
200:        sx$(w-1)=STR$(INT(VAL(sx$(w+1))))
                                  ! Ausführen
201:        @ordner
202:        @fidel
203:        CLR w
204:      ENDIF
205:    LOOP
206:    DO
207:      EXIT IF INSTR(1,re$,"a")=0   ! Absolut
208:      ADD w,1
209:      IF sx$(w)="a"                ! ist a dabei
210:        eg=LEN(sx$(w+1))
211:        isr=INSTR(1,re$,"a")       ! wo ist i
212:        we=1+LEN(sx$(w+1))
213:        sx$(w-1)=STR$(ABS(VAL(sx$(w+1))))
                                       ! Ausführen
214:        @ordner
215:        @fidel
216:        CLR w
217:      ENDIF
218:    LOOP
219:    DO
220:      EXIT IF INSTR(1,re$,"l")=0   ! Logarithmus e
221:      ADD w,1
222:      IF sx$(w)="l"
223:        eg=LEN(sx$(w+1))
224:        isr=INSTR(1,re$,"l")
225:        we=1+LEN(sx$(w+1))
226:        IF VAL(sx$(w+1))<=0
227:          nulllog%=1
228:          @komplex
229:        ELSE
230:          sx$(w-1)=STR$(LOG(VAL(sx$(w+1))))
231:          @ordner
232:          @fidel
233:        ENDIF
234:        CLR w
235:      ENDIF
236:      EXIT IF nulllog%=1
237:    LOOP
238:    CLR nulllog%
239:    DO
240:      EXIT IF INSTR(1,re$,"d")=0   ! Logaritmus 10
241:      ADD w,1
242:      IF sx$(w)="d"
243:        eg=LEN(sx$(w+1))
244:        isr=INSTR(1,re$,"d")
245:        we=1+LEN(sx$(w+1))
246:        IF VAL(sx$(w+1))<=0
247:          nulllog%=1
248:          @komplex
249:        ELSE
250:          sx$(w-1)=STR$(LOG10(VAL(sx$(w+1))))
251:          @ordner
252:          @fidel
253:        ENDIF
254:        CLR w
255:      ENDIF
256:      EXIT IF nulllog%=1
257:    LOOP
258:    CLR nulllog%
259:    DO
260:      EXIT IF INSTR(1,re$,"s")=0   ! Sinus
261:      ADD w,1
262:      IF sx$(w)="s"
263:        eg=LEN(sx$(w+1))
264:        isr=INSTR(1,re$,"s")
265:        we=1+LEN(sx$(w+1))
266:        sx$(w-1)=STR$(SIN(VAL(sx$(w+1))*PI/180))
267:        @ordner
268:        @fidel
269:        CLR w
270:      ENDIF
```

→

```
271:    LOOP
272:    DO
273:      EXIT IF INSTR(1,re$,"c")=0    ! Cosinus
274:      ADD w,1
275:      IF sx$(w)="c"
276:        eg=LEN(sx$(w+1))
277:        isr=INSTR(1,re$,"c")
278:        we=1+LEN(sx$(w+1))
279:        sx$(w-1)=STR$(COS(VAL(sx$(w+1))*PI/180))
280:        @ordner
281:        @fidel
282:        CLR w
283:      ENDIF
284:    LOOP
285:    DO
286:      EXIT IF INSTR(1,re$,"t")=0    ! Tangens
287:      ADD w,1
288:      IF sx$(w)="t"
289:        eg=LEN(sx$(w+1))
290:        isr=INSTR(1,re$,"t")
291:        we=1+LEN(sx$(w+1))
292:        sx$(w-1)=STR$(TAN(VAL(sx$(w+1))*PI/180))
293:        @ordner
294:        @fidel
295:        CLR w
296:      ENDIF
297:    LOOP
298:    DO
299:      EXIT IF INSTR(1,re$,"w")=0    ! Wurzelziehen
300:      ADD w,1
301:      IF sx$(w)="w"
302:        eg=LEN(sx$(w+1))
303:        isr=INSTR(1,re$,"w")
304:        we=1+LEN(sx$(w+1))
305:        IF VAL(sx$(w+1))<0
306:          nullsqr%=1
307:          wa%=2
308:          @komplex
309:        ELSE
310:          sx$(w-1)=STR$(SQR(VAL(sx$(w+1))))
311:          @ordner
312:          @fidel
313:        ENDIF
314:        CLR w
315:      ENDIF
316:      EXIT IF nullsqr%=1
317:    LOOP
318:    CLR nullsqr%
319:    DO
320:      EXIT IF INSTR(1,re$,"^")=0    ! Potenzieren
321:      ADD w,1
322:      IF sx$(w)="^"
323:        ad=LEN(sx$(w-1))
324:        eg=LEN(sx$(w+1))
325:        isr=INSTR(1,re$,"^")
326:        we=LEN(sx$(w-1))+1+LEN(sx$(w+1))
327:        minus%=0
328:        IF VAL(sx$(w-1))<0 AND INT(VAL(sx$(w+
                                  1)))<>VAL(sx$(w+1))
329:          nullsq2%=1
330:          wa%=anzzeig%
331:          @komplex
332:        ELSE
333:          IF VAL(sx$(w-1))<0 AND tor2%=0
                          ! negativ erkennen
334:            minus%=1
335:            sx$(w-1)=STR$(VAL(sx$(w-1))*-1)
336:          ENDIF
337:          sx$(w-1)=STR$(VAL(sx$(w-1))^(VAL(sx$(w+
                                          1))))
338:          IF minus%=1
339:            sx$(w-1)=STR$(VAL(sx$(w-1))*-1)
340:          ENDIF
341:          @ordner
342:          @fidel
343:        ENDIF
344:        CLR w
345:      ENDIF
346:      EXIT IF nullsq2%=1
347:    LOOP
348:    CLR nullsq2%
349:    DO
350:      EXIT IF INSTR(1,re$,"/")=0    ! Teilen
351:      ADD w,1
352:      IF sx$(w)="/"
353:        ad=LEN(sx$(w-1))
354:        eg=LEN(sx$(w+1))
355:        isr=INSTR(1,re$,"/")
356:        we=LEN(sx$(w-1))+1+LEN(sx$(w+1))
357:        IF VAL(sx$(w+1))<>0
358:          anzzeig%=VAL(sx$(w+1))
359:          sx$(w-1)=STR$(VAL(sx$(w-1))/(VAL(sx$(w+
                                          1))))
360:          @ordner
361:          @fidel
362:        ELSE
363:          GOSUB null
364:        ENDIF
365:        CLR w
366:      ENDIF
367:      EXIT IF nulldiv%=1
368:    LOOP
369:    CLR nulldiv%
370:    DO
371:      EXIT IF INSTR(1,re$,"*")=0    ! Multiplizieren
372:      ADD w,1
373:      IF sx$(w)="*"
374:        ad=LEN(sx$(w-1))
375:        eg=LEN(sx$(w+1))
376:        isr=INSTR(1,re$,"*")
377:        we=LEN(sx$(w-1))+1+LEN(sx$(w+1))
378:        sx$(w-1)=STR$(VAL(sx$(w-1))*(VAL(sx$(w+
                                          1))))
379:        @ordner
380:        @fidel
381:        CLR w
382:      ENDIF
383:    LOOP
384:    DO
385:      EXIT IF INSTR(1,re$,"-")=0    ! Subtrahieren
386:      EXIT IF INSTR(1,rb$,"-")=0    ! Negation ?
387:      ADD w,1
388:      IF sx$(w)="-"
389:        ad=LEN(sx$(w-1))
390:        eg=LEN(sx$(w+1))
391:        isr=INSTR(1,rb$,"-")
392:        IF VAL(sx$(w-1))<0
393:          ADD isr,1
394:        ENDIF
395:        we=LEN(sx$(w-1))+1+LEN(sx$(w+1))
396:        sx$(w-1)=STR$(VAL(sx$(w-1))-(VAL(sx$(w+
                                          1))))
397:        @ordner
398:        @fidel
399:        CLR w
400:      ENDIF
401:    LOOP
402:    DO
403:      EXIT IF INSTR(1,re$,"+")=0    ! Addieren
404:      EXIT IF INSTR(1,rb$,"+")=0    ! positiv ?
405:      ADD w,1
406:      IF sx$(w)="+"
407:        ad=LEN(sx$(w-1))
408:        eg=LEN(sx$(w+1))
409:        isr=INSTR(1,rb$,"+")
410:        IF VAL(sx$(w-1))<0
411:          ADD isr,1
412:        ENDIF
413:        we=LEN(sx$(w-1))+1+LEN(sx$(w+1))
414:        sx$(w-1)=STR$(VAL(sx$(w-1))+(VAL(sx$(w+
                                          1))))
415:        @ordner
416:        @fidel
417:        CLR w
418:      ENDIF
419:    LOOP
420:  RETURN
421:  PROCEDURE null
422:    nulldiv%=1
423:    PRINT "Division durch Null "
424:    nuuld%=1
425:  RETURN
426:  PROCEDURE ordner  ! String wieder zusammenbauen
427:    leh=LEN(sx$(w-1))
428:    re$=MID$(re$,0,(isr-ad-1))+SPACE$(leh)+
          MID$(re$,isr+eg+1,LEN(re$))
429:    MID$(re$,(isr-ad),leh)=sx$(w-1)
430:    CLR ad
431:  RETURN
432:  PROCEDURE fidel   ! Zahlen und Rechenoperationen
                          trennen
433:    CLR rb$
```

→

```
434:    DO
435:      ADD s,1
436:      anf$=MID$(re$,s,1)
437:      EXIT IF LEN(anf$)<=0
438:      ADD e,1
439:      syn%=ASC(MID$(re$,s,1))
440:      IF MID$(re$,s,1)="-"     ! Negation erkennen
441:        sn1%=ASC(MID$(re$,s-1,1))
442:        IF sn1%<=45 OR sn1%=47 OR sn1%=99 OR
              sn1%=116 OR sn1%=115 OR sn1%=119 OR
              sn1%=97 OR sn1%=105 OR sn1%=100 OR
              sn1%=108
443:          syn%=48              ! Programm Syntax ?
444:        ENDIF
445:      ENDIF
446:      IF syn%>47 AND syn%<>94 AND syn%<>46 AND
             syn%<>115 AND syn%<>99 AND syn%<>116 AND
             syn%<>119 AND syn%<>97 AND
             syn%<>105 AND syn%<>100 AND syn%<>108
447:        DO
448:          ADD s,1
449:          synt%=ASC(MID$(re$,s,1))
450:          IF MID$(re$,s,1)="--"  ! Negativ erkennen
451:            sn%=ASC(MID$(re$,s-1,1))
452:            IF sn%<=45 OR sn%=47
453:              synt%=48
454:            ENDIF
455:          ENDIF
456:          IF MID$(re$,s,2)="E+" OR MID$(re$,s,
               2)="E-"             ! Exponent erkennen
457:            expo%=1
458:            synt%=48
459:          ENDIF
460:          EXIT IF synt%<=45 OR synt%=47 OR
                      synt%=94 OR synt%=115 OR
                      synt%=99 OR synt%=116
461:          EXIT IF synt%=119 OR synt%=97 OR
                      synt%=105 OR synt%=100 OR
                      synt%=108
462:          IF expo%=1           ! Exponent setzten
463:            anf$=anf$+MID$(re$,s,2)
                                   ! mit Exponent
464:            ADD s,1
465:          ELSE
466:            anf$=anf$+MID$(re$,s,1)
                                   ! Ohne Exponent
467:          ENDIF
468:          expo%=0
469:        LOOP
470:        SUB s,1
471:      ENDIF
472:      sx$(e)=anf$              ! Aufspalten
473:      as%=INSTR(1,sx$(e),"E")
474:      wisch$=sx$(e)
475:      IF as%>0                 ! Exponent ?
476:        MID$(wisch$,as%+1,1)="0"
477:      ENDIF
478:      IF VAL(wisch$)>=0
479:        rb$=rb$+wisch$         ! Hilfstring bilden
480:      ELSE
481:        IF as%>0
482:          wisch$=STR$(ABS(VAL(wisch$)))
483:          MID$(wisch$,as%,1)="0"
484:          rb$=rb$+wisch$
485:        ELSE
486:          rb$=rb$+STR$(ABS(VAL(wisch$)))
                                   ! Hilfstring fertig
487:        ENDIF
488:      ENDIF
489:      CLR anf$
490:    LOOP
491:    CLR s,e
```

```
492:  RETURN                      ! Aufspaltung beendet
493:  PROCEDURE text              ! Info für Benutzer
494:    GRAPHMODE 4
495:    DEFTEXT 1,0,0,13
496:    TEXT 150,50,"   Parse-Interpreter written by
                       P.Bauer     "
497:    TEXT 150,66,"  Rombergstr.8, 5000 Köln 41.
                       Vers: 4.00  "
498:    GRAPHMODE 0
499:    TEXT 150,110," Folgende Rechenoperationen sind
                       möglich:"
500:    TEXT 148,140,"            Standardfunktionen :"
501:    TEXT 148,160,"     | + | - | / | * | ^ | pi |
                       () |"
502:    TEXT 148,180," Variablen = Alle GroPbuchstaben
                       von A->Z"
503:    TEXT 148,200,"             Sonderfunktionen :"
504:    TEXT 450,360," Taste Drücken !!"
505:    TEXT 450,376,"   Esc =End"
506:    RESTORE 44
507:    FOR bn%=1 TO 9
508:      READ hel$
509:      TEXT 280,210+(bn%*17),hel$
510:    NEXT bn%
511:    44:
512:    DATA s()=Sin(),c()=Cos(),t()=Tan(),d(+-)=Log(),
           l(+-)=Ln()
513:    DATA w(+-)=Sqr(),i()=Int(),a()=Abs(),e=Exp(1)
514:  RETURN
515:  PROCEDURE variable
516:    vai%=INSTR(1,re$,"=")-1        !wo ist = ?
517:    var$(ASC(MID$(re$,0,vai%))-65)=MID$(re$,vai%+2,
         LEN(re$))                 ! Zuweisungs var (z.B. A=)
518:  RETURN
519:  PROCEDURE komplex
520:    DIM z(100),h(100)
521:    DEFNUM 5
522:    IF nulllog%=1
523:      PRINT "z=ln(";(ABS(VAL(sx$(w+1))));")+i*
                 (";CHR$(227);"+2*k* ";CHR$(227);")
                 (für k ";CHR$(238);" G)"
524:      llog%=1
525:    ENDIF
526:    IF nullsqr%=1 OR nullsq2%=1
527:      nuuld%=1
528:      pose=pose+wa%
529:      FOR kom%=1 TO wa%
530:        IF nullsq2%=1
531:          a=(ABS(VAL(sx$(w-1))))^(1/wa%)
532:        ELSE
533:          a=(ABS(VAL(sx$(w+1))))^(1/wa%)
534:        ENDIF
535:        al=180/wa%
536:        umr=(al+(((kom%-1)*(360/wa%))))
537:        z(kom%)=a*(COS(umr*PI/180))
538:        h(kom%)=a*(SIN(umr*PI/180))
539:        IF ABS((z(kom%)))<1.0E-07
540:          z(kom%)=0
541:        ELSE
542:          IF ABS((h(kom%)))<1.0E-07
543:            h(kom%)=0
544:          ENDIF
545:        ENDIF
546:        ADD pos,1
547:        PRINT ;"z";kom%;"= ";z(kom%);" +
                 ";h(kom%);"*i"
548:      NEXT kom%
549:    ENDIF
550:    DEFNUM 12
551:    ERASE h()
552:    ERASE z()
553:  RETURN
```

PPP

Spaß muß sein !



Strategie

Balance of Power 1990	65,90
Waterloo	65,90

Simulation

Pirates	65,90
Stuntcar	65,90
F-16 Combat Pilot	65,90

Rollenspiele

Bloodwych	69,90
Kult	67,90

Action

Blood Money	65,90
Gemini Wing	54,90
Targhan	65,90
Red Heat	65,90
Phobia	59,90
Running Man (dt.)	63,90
Spherical	59,90
Indiana Jones - The last Crusade	54,90
Xenon II Megablast	69,90

Happy Games

Archipelagos	63,90
Rick Dangerous	65,90

Sport

Microprose Soccer	63,90
Buffalo Bill's Rodeo	65,90

Sampler

Triad II - Menace, Baal, Tetris	69,90
Giants	73,90

Jetzt wirds ernst !

SIERRA total

Goldrush	65,90
King's Quest 1/2/3	84,90
King's Quest 4	72,90
Larry 1	55,90
Larry 2	72,90
Manhunter	72,90
Police Quest 1	55,90
Police Quest 2	65,90
Space Quest 1	65,90
Space Quest 2	55,90
Space Quest 3	72,90

Wir haben auch sämtliche Lösungen zu diesen Sierra-Spielen: DM 12,- je Lösung, alle komplett, im Ringbuchordner DM 79,- ! Außerdem Lösung zu Kult - DM 12,-.

Data Becker

BECKERtext ST 2.0
die schöne Textverarbeitung, die Schlagzeilen macht.
298,00

BECKERpage ST 2.0
Desktop Publishing, in der neuen Version 2.0, mit noch mehr Komfort. 398,00

BECKERcalc/3 ST
Tabellenkalkulations-Programm 498,00

BECKERcad ST 1.2
Professionelles CAD-System
498,00

Bitte forderen Sie unseren kostenlosen Katalog (enthält auch PD-Software) unter Angabe Ihres Computers an.

Die Lieferung erfolgt per Nachnahme, zuzügl. DM 6,50 (Ausland DM 10,-) oder per Vorauskasse, zuzügl. DM 4,- (Ausland DM 6,-).

07252/86699

POWER PER POST

Bestellen können Sie bei:

Werner Rätz, Postfach 1640/ST, 7518 Bretten







Gehäuse 330×360×55 (Monitor-Untersatz, Mega-ST-Maße), für Hard- und/oderFloppydisks.	**79,–**
Gehäuse mit Netzteil	**179,–**
SCSI-Hostadapter mit Software (s. Spalte rechts)	**248,–**
Lüfter 60×60	**25,–**
HD-Netzteil 50 W	**99,–**
Wechselplatten	
Wechselplattenlaufwerk – 25ms – SCSI – ohne Medium	**2198,–**
Wechselplatte 44MB Medium	**275,-**
SD44E-S Subsystem mit herausgeführtem SCSI-Bus	**2498,-**

SD44E **2498,-**
Subsystem für ATARI ST – Interner SCSI-Adapter, gepufferter DMA out, super-Software (siehe Spalte rechts)

Fordern Sie unser Gratis-Info an, Händlerpreisliste gegen Gewerbenachweis.

ATARI-ST Festplatten Serie SD

Hardware:
SCSI-System (integrierter **Hostadapter**)
durchgeschliffener, gepufferter DMA,
Superleise **Lüftung** (Speziallüfter)

Autopark – Hardwaremäßig
Aufrüstfähig (bis 200 MB)
Thermo-geregelter Lüfter
SCSI-Output

Software:
Bis zu 14 Partitions installierbar

Wählbare Bootpartition (aus 16)
Abschaltbarer Schreibschutz
R-TOS/ALADIN/PC-DITTO-fähig
Passwort-Funktion (Datenschutz)
Abstellbares Datenverify

Lieferbar im nebenstehenden Gehäuse oder im 19" 2HE Rackgehäuse (+ 100,– DM)

Auch mit herausgeführter SCSI-Schnittstelle lieferbar (Subsystem für Applerechner, diverse Sampler etc.)

Kapazitäten: 32 – 202 MB
Preise: ab 1398,- DM
85MB 28ms 2098,- DM

32 MB / 28 ms: **1398,–**
48 MB / 28 ms: **1548,–**
85 MB / 28 ms: **1898,–**
Weitere Versionen lieferbar

LACOM

Ilias Lazaridis · Emscherstr. 45 · 4200 Oberhausen 1
Tel.: (0208) 655051 · Btx 0208654390 · FAX (0208) 654390

Komfortables Dialog-Handling in GFA-BASIC 3.0

Die AES-Funktion *FORM_DO* hat ihre Licht- und Schattenseiten. Einerseits erspart sie dem Programmierer eine Menge Arbeit, übernimmt sie doch die komplette Verwaltung einer Dialogbox, andererseits aber bietet sie dem Anwender wenig Komfort bei der Bedienung. Außerdem wird wohl so mancher nach einem FORM_DO-Aufruf verzweifelt zum Reset-Taster gegriffen haben, weil durch ein vergessenes Exit-Objekt oder einen falsch übergebenen Objektindex der Rechner mal wieder im Halbleiter-Nirwana verschwunden war.

Die Problemstellung war klar. Eine eigene FORM_DO-Funktion mit völlig neuen Eigenschaften mußte her. Sie sollte einfach anstelle der gleichnamigen AES-Routine einzusetzen sein und zusätzliche Funktionen, vor allem zur komfortablen Positionierung des Cursors, besitzen. Außerdem sollten sämtliche Objekte einer Dialogbox alternativ auch über die Tastatur bedient werden können - eine nützliche Hilfe für fortgeschrittene Anwender eines Programms, die sich bei Pulldown-Menüs bereits durchzusetzen beginnt.

Das Ergebnis dieser Bemühungen sind die Extended *FORM_DO*-Funktion und der KEY RESOURCE EDITOR, mit dessen Hilfe man die Tastenbelegungen der einzelnen Objekte einer Resource erzeugen kann.

Extended FORM_DO

Kommen wir zunächst zur Beschreibung der neuen FORM_DO-Routine. Sie besitzt sämtliche Eigenschaften der normalen FORM_DO-Funktion. Zusätzlich jedoch sind einige weitere Steuercodes standardmäßig eingebaut (Bild 1). Außerdem läßt sich der Cursor jetzt mit der Maus zeichengenau in Textfelder setzen und landet nicht wie gewohnt immer am rechten Ende. Auch Dialogboxen ohne Exit-Objekt werden verwaltet.

Die wohl interessanteste Eigenschaft ist aber die Möglichkeit, sämtliche Objekte einer Dialogbox wahlweise auch über die Tastatur bedienen zu können. Das Prinzip ist eigentlich recht simpel: Einem Objekt einer Dialogbox, beispielsweise einem Radiobutton, wird eine bestimmte Taste, z.B. <F1>, zugeordnet. Bei Druck auf <F1> verhält sich die Dialogbox genauso, als wäre der entsprechende Radiobutton angeklickt worden - er wird selektiert. Prinzipiell läßt sich dies für jedes beliebige Objekt realisieren. Eine Standardeinstellung wäre etwa die Belegung des *Cancel/Abbruch*-Buttons, den man in fast jeder Dialogbox findet, mit der Taste <Undo>. Eine solche, wenn auch die einzige, Tastaturbelegung ist ja bereits im AES implementiert, und zwar in Gestalt des Default-Objektes, das wahlweise mit der Maus oder der Return- bzw. Enter-Taste angewählt werden kann.

Wie funktioniert nun die Zuordnung einer Taste zu einem beliebigen Objekt? Dazu müßte man am besten den Code der Taste in der Objektstruktur des betreffenden Objekts innerhalb der Resource unterbringen, um unnötigen Speicherbedarf, etwa durch eine zusätzliche Datei, zu vermeiden. Betrachtet man die Objektstruktur näher, dann bleiben nur zwei Möglichkeiten, nämlich *OB_STATE* und *OB_FLAGS*, denn dies sind die einzigen Objektinformationen, die vom AES bitweise ausgewertet werden, und nur dort können wir den Tastaturcode "einpflanzen", ohne daß die interne Struktur der Resource oder das Objekt selbst verändert wird. Bei OB_STATE sind die Bits 6-13 unbelegt, bei OB_FLAGS die Bits 9-15. Laut Digital Research sind sie auch nicht reserviert, also gerade richtig für unsere Zwecke. Für den Code einer Taste benötigen wir acht Bits. Um die Routine möglichst flexibel zu halten, wird nicht der ASCII-, sondern der Scan-Code gespeichert und zusätzlich in drei Bits der Status der Umschalttasten Shift, Control und Alternate, wobei auf die Unterscheidung von rechter und linker Shift-Taste verzichtet wurde, um der völligen Verwirrung eines gestreßten Anwenders vorzubeugen. Die

Standard-Steuertasten von EXTENDED FORM_DO

SHIFT/CURSOR UP, HOME	Cursor in erstes Eingabefeld
SHIFT/CURSOR DOWN, SHIFT/HOME	Cursor in letztes Eingabefeld
SHIFT/CURSOR LEFT	Cursor an Anfang des Eingabefeldes
SHIFT/CURSOR RIGHT	Cursor ans Ende des Eingabefeldes

Bild 1

zusätzliche Berücksichtigung der Umschalttasten ist jedoch äußerst sinnvoll, vor allem bei Dialogboxen mit Texteingabefeldern, denn dort sind ja die meisten Tasten bereits für die Eingabe reserviert. Der Scan-Code der Taste wird in die freien Bits 8-15 von OB_STATE eingetragen, der zugehörige Status der Umschalttasten in die Bits 13-15 von OB_FLAGS (siehe Bild 2). Ich habe die beiden Werte mit Absicht an den "oberen Rand" gelegt, damit sie nicht mit möglichen Erweiterungen kollidieren, die ja beispielsweise vom Kuma-RCS unterstützt werden, wo man die Bits 6 und 7 von OB_STATE und die Bits 10 und 11 von OB_FLAGS manipulieren kann.

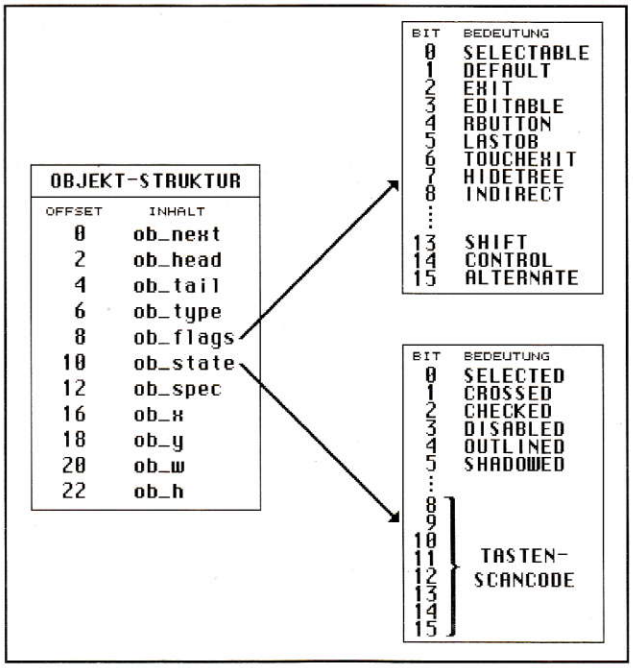


Bild 2: Die benötigten Tasten-Codes werden an unbenützte Stellen in ob_flags und ob_state eingetragen.

Eine Resource nach dem oben beschriebenen Verfahren von Hand zu verändern, wäre viel zu umständlich. Daher habe ich den Key Resource Editor geschrieben, der es auf einfache und komfortable Weise ermöglicht, Tastenbelegungen für Resource-Objekte zu erzeugen. Eine derart veränderte Resource bleibt natürlich auch weiterhin lauffähig mit den üblichen AES-Routinen. Weitere Einzelheiten über den Key Resource Editor finden Sie weiter hinten.

Format von OBJC_EDIT

Original GEM-Dokumentation von DR:
OBJC_EDIT(ob_edtree,ob_edobject,ob_edchar,ob_edidx,
ob_edkind,ob_newedidx)

Binding in den meisten (C-)Compilern:
OBJC_EDIT(ob_edtree,ob_edobject,ob_edchar,ob_edidx,
ob_edkind)
Hier wird für die bisherige Cursorposition (ob_edidx) und die zurückgelieferte neue Position (ob_newedidx) dieselbe Variable verwendet.

Bild 3

Im folgenden möchte ich die Funktionsweise der neuen FORM_DO-Routine erklären und dabei auch auf die scheinbar wenig beachteten AES-Funktionen *FORM_BUTTON*, *FORM_KEYBD* und *OBJC_EDIT* eingehen. Grundsätzlich muß die Funktion genau das leisten, was auch die Original-FORM_DO-Routine bietet, nämlich die Abfrage und Auswertung von Tastatur- und Mausknopfeingaben. Steuertasten wie Cursor up/down, ESC etc. müssen richtig interpretiert, eingegebene Buchstaben und Ziffern in Textfelder eingetragen und angeklickte Objekte eventuell selektiert werden. Nach Anwählen eines Objekts mit *Exit-* oder *Touchexit*-Status muß der Dialog korrekt beendet werden. Dazu kommen nun noch die bereits geschilderten Sonderfunktionen. Bereits in der ST Computer 8/87 wurde eine modifizierte Routine von Tim Oren, dem Autor des Digital Research RCS, vorgestellt, die die geschilderten grundsätzlichen Aufgaben übernimmt. Genau das gleiche Prinzip, wenn auch in stark abgewandelter Form, benutzt auch die hier gezeigte FORM_DO-Funktion. Als Parameter werden ihr wie bei der Originalfunktion die Baumadresse und das Startobjekt, das als erstes ediert werden soll, übergeben. Mit *WIND_UPDATE(2)* übernimmt die Applikation - das ist in diesem Fall unsere neue FORM_DO-Routine - die totale Kontrolle über die Maus. Das ist besonders deshalb wichtig, damit der Benutzer die Pull-Down-Menüs nicht anwählen kann. Dahinter finden wir schon den ersten entscheidenden Unterschied zur Originalroutine. Es wird geprüft, ob das übergebene Startobjekt tatsächlich edierbar ist. Wenn nicht, wird mittels der Suchfunktion *search_ob_flag* das erste edierbare Textfeld bestimmt. So kann es auch nicht passieren, daß eine versehentlich falsch übergebene Objektnummer den Cursor mitten in der Dialogbox erscheinen läßt und früher oder später zum Absturz führt. Die nun folgende WHILE...WEND-Schleife überwacht die Benutzereingaben über Maus und Tastatur. Zunächst wird überprüft, ob ein neues Textfeld für die nächste Eingabe bestimmt wurde. Das ist - unter der Voraussetzung, daß überhaupt Eingabefelder vorhanden sind - ganz zu Beginn der Fall und im folgenden immer dann, wenn der Cursor wegen eines Mausklicks oder einer Steuertaste in ein neues Feld bewegt werden muß.

Das Darstellen des Strichcursors übernimmt die Funktion OBJC_EDIT. Und genau diese Funktion hat wohl in der Vergangenheit immer wieder für Verwirrung gesorgt, denn die ursprüngliche DR-Dokumentation und die Bindings in verschiedenen Compilern (z.B. Megamax-C, DR-C) weichen voneinander ab. In GFA-Basic 3.0 ist diese Funktion löblicherweise nach der Originaldokumentation implementiert. Sollten Sie *Extended FORM_DO* auf eine andere Sprache umsetzen wollen, überprüfen Sie zunächst, in welchem Format OBJC_EDIT bei Ihnen eingebunden ist. Den Unterschied zeigt Bild 3. Nun zur Funktionsweise von OBJC_EDIT. Übergeben werden die Baumadresse, der Index des betreffenden Textobjekts, das eingegebene Zeichen als Wort (Hi-Byte: Scan-Code, Lo-Byte: ASCII-Code), die bisherige Position des Cursors, ein Flag zur Bestimmung der Funktion und als Rückgabewert die neue Position des Cursors. Wichtig ist vor allem der fünfte Parameter, nämlich das Flag *ob_edkind*. Dieses Flag kann die Werte 0-3 annehmen, wobei der Wert 0 *(ED_START)* keinerlei Funktion besitzt. Setzen wir für *ob_edkind* eine 1 *(ED_INIT)* ein, so wird aus der Textmaske und dem Text ein formatierter String berechnet und der Cursor eingeschaltet, einfach ausgedrückt: Der Cursor erscheint hinter dem letzten eingetragenen Zeichen in dem angegebenen Textobjekt. Der dritte und der vierte Parameter haben hier keine Bedeutung. Deshalb kann hier auch einfach eine 0 übergeben werden. Wichtig ist nur der zurückgelieferte Wert für die Variable *pos&*, welche nach dem Aufruf die Position des Cursors innerhalb des Eingabefeldes enthält.

Durch den nachfolgenden EVNT_MULTI-Aufruf wird auf einen Mausklick oder Tastendruck gewartet. War es ein Tastendruck, kommt die Funktion *process_key* zum Einsatz. Sie ist ein Er-

satz für die AES-Routine FORM_ KEYBD. Die Aufgabe von FORM_ KEYBD ist es, die Steuertasten Cursor up/down herauszufiltern und, falls eine der beiden Tasten gedrückt wurde, den Index des neuen Textobjekts zurückzuliefern. Außerdem ist sie dafür zuständig, bei Druck auf Return oder Enter zu überprüfen, ob ein Default-Objekt vorhanden ist, und gegebenfalls die Beendigung des Dialogs zurückzumelden. Trotz intensiver Bemühungen ist es mir jedoch nicht gelungen, diese AES-Funktion zum Laufen zu bringen (an der Implementierung in GFA-Basic kann's nicht liegen, denn selbst die Parameterübergabe "zu Fuß" durch Vorbesetzen der GEM-Arrays schaffte keine Abhilfe). Ich lasse mich natürlich gerne eines Besseren belehren. Allerdings ist das korrekte Funktionieren dieser Routine auch nicht so wichtig hier, denn durch die vielen neuen Steuertasten, die unser FORM_DO unterstützt, muß ohnenhin eine eigene FORM_KEYBD-Funktion vorhanden sein, die alle standardmäßig vorhandenen Steuercodes korrekt interpretiert und zusätzlich - was ja nicht vergessen werden darf - die Tastenbelegungen der einzelnen Objekte überprüft.

Parameter der Funktion process_key

Wertparameter:

tree%	Baumadresse
obj&	momentan ediertes Textobjekt (ed_obj& aus form_do)
kstate&	Status der Umschalttasten (SHIFT, CTRL, ALT)
ev_key&	Scan-Code und ASCII-Code der gedrückten Taste, wird von EVNT_MULTI geliefert (GEM-Format)

VAR-Parameter:

nxt_obj&	Index des neuen aktuellen Objekts. Das ist entweder ein neues Textobjekt oder ein Exit-/Touchexitobjekt oder Null
nxtchar&	Inhalt 0: Die gedrückte Taste wurde verwendet Inhalt=ev_key&: Die gedrückte Taste muß weiterverarbeitet werden.
pos&	aktuelle Position des Cursors im Eingabefeld

Bild 4

Genau das tut die Funktion *process_key*. Ich habe sie mit Absicht nicht form_keybd genannt, da die Parameterübergabe von der der normalen FORM_KEYBD-Routine abweicht. Die einzelnen Parameter und ihre Bedeutung sehen Sie in Bild 4. Entscheidend für uns ist vorerst nur, daß *process_key* für die Variable *ev_key&* eine Null zurückliefert, wenn es sich bei der gedrückten Taste um eine Steuertaste gehandelt hat, ansonsten wird der Tastencode unverändert zurückgegeben. Zusätzlich bestimmt der in *do_dial&* enthaltene Rückgabewert, ob der Dialog beendet werden muß, d.h. ob ein Exit- oder Touchexit-Objekt über die Tastatur angewählt wurde. Ist *ev_key&* nach dem Aufruf ungleich Null, dann wurde keine Steuertaste gedrückt, und wir müssen uns selbst darum kümmern, daß das eingegebene Zeichen in dem aktuellen Textobjekt verarbeitet wird. Das geschieht wiederum mit Hilfe von OBJC_EDIT, diesmal mit dem Flag 2. Dadurch wird uns eine Menge Arbeit abgenommen, denn wir müssen das eingebene Zeichen beispielsweise nicht auf seine Zulässigkeit an dieser Stelle überprüfen. Das alles erledigt OBJC_EDIT. Es ist übrigens nicht ganz richtig, wenn ich davon spreche, daß an OBJC_EDIT keine Steuertasten übergeben werden. Wenn Sie die Funktion *process_key* etwas näher betrachten, werden Sie feststellen, daß die Steuertasten Cursor links/rechts, Backspace, Delete und Escape dort gar nicht behandelt werden. Das hat auch seine Richtigkeit, denn alle Steuertasten, die sich auf das aktuelle Textobjekt beziehen und den Cursor nicht in ein anderes Eingabefeld bewegen, werden auch von OBJC_EDIT verwaltet. Das ist natürlich wahrer Luxus, denn wollte man all diese Funktionen selbst berücksichtigen, würde der Aufwand erheblich ansteigen.

Gehen wir nun weiter im Listing. Wurde also eine Taste gedrückt, ist diese korrekt verarbeitet worden. Im Falle eines Mausklicks wird mittels OBJC_FIND überprüft, ob sich unter dem Mauszeiger ein Objekt befindet. Ist die Maus außerhalb der Dialogbox, wird für *nxt_obj&* eine -1 zurückgeliefert, und wir quittieren das mit dem entsprechenden Klingelzeichen. Ansonsten muß bei Bedarf eine enstprechende Aktion, wie zum Beispiel das Selektieren eines Radiobuttons, ausgelöst werden. Glücklicherweise nimmt uns die Funktion FORM_BUTTON wiederum den größten Teil der Arbeit ab. Diese Funktion nämlich, die intern auch von FORM_DO benutzt wird, ermöglicht Mausknopfeingaben innerhalb einer Dialogbox. Übergeben werden Baumadresse, Index des angeklickten Objekts (hier ermittelt durch OBJC_FIND) und die Anzahl der Klicks. Genau wie bei *process_key* erhalten wir als Rückgabewert in *do_dial&* eine Null, falls ein Exit-/Touchexit-Objekt angeklickt wurde, und in *nxt_obj&* den Index des neuen aktuellen Objekts. Hier möchte ich nochmal einhaken, denn der Begriff "neues aktuelles Objekt" ist doch sehr schwammig. Ich muß zugeben, daß ich anfangs auch etwas Probleme mit dieser Formulierung hatte. Deshalb sei es hier nochmals klar und deutlich erklärt: *nxt_obj&* ist nach dem FORM_BUTTON-Aufruf nur dann ungleich Null, wenn entweder ein Exit- oder Touchexit-Objekt oder ein edierbares Textfeld angeklickt wurden. In diesen Fällen sind Reaktionen von unserer Seite nötig, nämlich die Beendigung des Dialogs unter Rückgabe des angeklickten Exit-Objekts bzw. die neue Positionierung des Cursors. Den Sinn der zusätzlichen Angabe der Klick-Anzahl finden wir bei den Touchexit-Objekten. Bei einem Doppelklick auf ein Touchexit-Objekt liefert FORM_ BUTTON den entsprechenden Objektindex mit gesetztem Bit 15 zurück, genauso wie bei der Original-FORM_DO-Funktion.

Nun wird es ein bißchen komplizierter, denn die neue FORM_DO-Routine soll ja auch die zeichengenaue Cursor-Positionierung mit Hilfe der Maus erlauben. Daher wird zunächst überprüft, ob es sich bei dem angeklickten Objekt um ein Texteingabefeld handelt. Wenn ja, dann werden zunächst die X-Koordinate des Textobjekts, die aktuelle Zeichenbreite und die Länge der Textmaske bestimmt. Die Textmaske bestimmt das Aussehen eines Textobjekts auf dem Bildschirm. Sie enthält den statischen Teil des Eingabefeldes (also z.B. die Bezeichnung) sowie für jedes edierbare Zeichen einen Unterstrich "_" als Platzhalter. Die Länge dieser Textmaske findet sich beim Offset 26 in der zugehörigen TEDINFO-Struktur, deren Adresse wir über *OB_SPEC(tree%,nxt_obj&)* erhalten. Nun wird abhängig von der Textausrichtung (*te_just*, TEDINFO Offset 16) die Position des Mauszeigers innerhalb der Textmaske bestimmt. Je nachdem, ob der Text linksbündig (0), rechtsbündig (1) oder zentriert (2) ist, verändert sich natürlich die horizontale Position der Maus relativ zur Textmaske. Ich hoffe, Sie haben es bis hierhin verstanden. Wenn nicht, dann wird ein intensiver Blick ins Listing bestimmt helfen. Noch sind wir aber nicht am Ziel angelangt. In der Variablen *mpos&* steht jetzt zwar, über dem wievielten Zeichen der Textmaske sich der Mauszeiger befindet, wir wissen jedoch immer noch nicht, welches edierbare Zeichen der Maus am nächsten ist, also wo der Cursor letztendlich erscheinen muß.

Das erledigt die nun folgende REPEAT...UNTIL-Schleife. Hier wird die Anzahl der vorkommenden Platzhalter "_" bis zur Mausposition gezählt. Die Variable *p&* schließlich enthält die neue aktuelle Cursorposition in dem angeklickten Textobjekt. Das scheint schon alles gewesen zu sein, jedoch gibt es da noch ein kleines Problem, nämlich: Wie positioniert man nun den Cursor beliebig innerhalb eines Eingabefeldes? Auf normalem Wege ist da nichts zu machen, denn der OBJC_EDIT-Aufruf zum Darstellen des Cursors (*ED_INIT*) läßt den Cursor grundsätzlich hinter dem letzten eingetragenen Zeichen erscheinen. Die Lösung dieses Problems findet sich in einem kleinen Trick: Man gaukelt der OBJC_EDIT-Routine einfach vor, daß die Anzahl der eingetragenen Zeichen gerade bis zur gewünschten Cursor-Position reicht. Um das zu erreichen, ersetzt man das entsprechende Zeichen, nachdem man es sich vorher gemerkt hat, durch ein Nullbyte. Dann ruft man OBJC_EDIT auf, was zur Folge hat, daß der Cursor an der angestrebten Position erscheint. Schließlich muß man das entsprechende Zeichen wieder an seiner alten Position eintragen. Bevor man jedoch den Cursor an seiner neuen Position erscheinen läßt, muß dieser erst an seiner alten gelöscht werden. Dies erledigt wiederum OBJC_EDIT mit *ob_edkind=3* (*ED_END*).

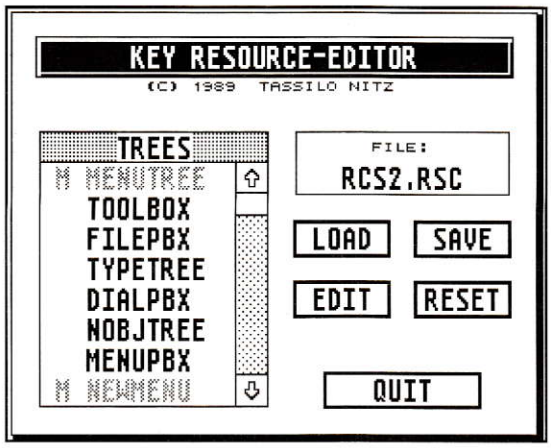


Bild 5: Der Key Resource Editor nach dem Start

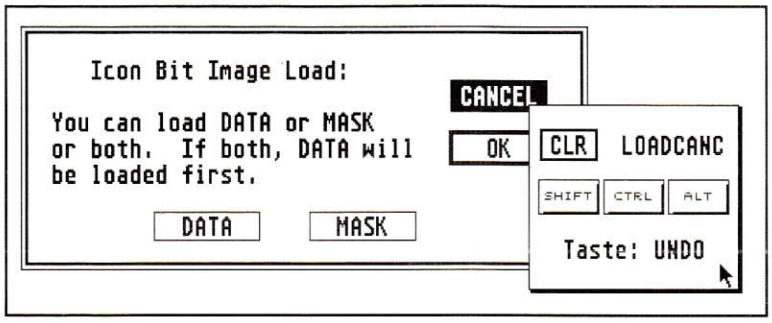


Bild 6: Tasten-Codes lassen sich den Buttons einfach zuordnen.

Wir sind nun fast am Ende von EXTENDED FORM_DO angelangt. Vor Abschluß der WHILE...WEND-Schleife wird noch für den Fall, daß der Dialog beendet werden muß oder ein neues Texteingabefeld angewählt wurde, der Cursor aus dem aktuellen Textobjekt entfernt. Zum Schluß wird die Mauskontrolle über *WIND_UPDATE(2)* wieder frei- und das Objekt, mit dem die Dialogbox verlassen wurde, zurückgegeben.

Noch ein paar Worte zur Funktion *process_key*. Bei Eingabe von Return oder Enter wird zunächst überprüft, ob ein Default-Objekt vorhanden ist. Ist dies nicht der Fall, wirken diese beiden Tasten genau wie Cursor down und bewegen den Cursor ins nächste Texteingabefeld. Eine Ausnahme jedoch bildet das letzte Eingabefeld. Befindet sich der Cursor im letzten Eingabefeld und existiert kein Default-Objekt, dann wird bei Druck auf Return/Enter geprüft, ob ein Exit-Objekt existiert. Wenn nicht, wird daraufhin der Dialog beendet. Das klappt natürlich auch bei Dialogboxen, die keine edierbaren Textobjekte besitzen. So gehören FORM_DO-Endlosschleifen aufgrund vergessener Exit-Objekte der Vergangenheit an. Wenn keine der im SELECT-Konstrukt aufgeführten Steuertasten gedrückt wurde, überprüft *search_key_obj*, ob einem Objekt der Dialogbox diese Taste bzw. Tastenkombination zugeordnet wurde. Wurde ein Objekt gefunden, wird mittels FORM_BUTTON die gleiche Aktion ausgelöst, als wäre das Objekt angeklickt worden.

Der Key Resource Editor

Soviel also zum Extended *FORM_DO*. Was wäre solch eine Erweiterung aber ohne die Möglichkeit, die notwendigen Tastaturbelegungen einfach, schnell und komfortabel in einer Resource zu erzeugen? Dazu habe ich den *Key Resource Editor* entwickelt. Er ermöglicht es auf einfache Weise, Resource-Files zu laden, diese zu edieren und wieder abzuspeichern. Der Key Resource Editor ist wegen seiner Listinglänge allerdings nur auf der Monatsdiskette zu finden. Wir bitten dafür um Verständnis. Ein großer Vorteil des Key Resource Editors ist, daß er Definitions-Dateien sowohl vom Digital Research RCS (Versionen 1 und 2) als auch vom Kuma-RCS lesen kann. Diese Fähigkeit ermöglicht es Ihnen, die einzelnen Objektbäume und Objekte namentlich anzusprechen, was eine große Arbeitserleichterung darstellt.

Nach dem Start des Programms erscheint die Auswahlbox (Bild 5). Von hier aus können Sie durch Anklicken von LOAD ein Resource-File einladen. Das Programm lädt das Resource-File und - falls vorhanden - das zugehörige Definitionsfile. Daraufhin werden alle Objektbäume der Resource in dem Fenster *TREES* dargestellt. Angewählt werden können nur Formularobjekte, das heißt also *Forms* (Kuma) bzw. *Dialogs, Panels* (DR). Alle übrigen Strukturen wie *Menüs* (M), *Alerts* (A), *Free Strings* (S) und *Free Images* (I) sind in heller Schrift dargestellt. Die jeweiligen Großbuchstaben davor geben Auskunft über die Art der Struktur. Ein Häkchen vor einem Objektbaum bedeutet, daß in dieser Dialogbox Tastenbelegungen eingetragen sind. Mit den Randobjekten des Fensters können Sie bei mehr als acht Strukturen den gezeigten Ausschnitt verändern, genau wie bei der Fileselektorbox. Um neue Tastenbelegungen zu speichern oder bereits bestehende zu ändern, klicken Sie auf den entsprechenden Objektbaum und danach auf EDIT. Das gleiche erreichen Sie durch Doppelklick auf den Baumnamen. Die Dialogbox wird daraufhin gezeichnet. Nun klicken Sie das erste Objekt an, dem Sie eine Taste zuordnen wollen. Es wird dann selektiert, und an der gleichen Stelle erscheint eine kleine Infobox (Bild 6). Sie gibt Auskunft über den Namen des Objekts sowie die momentane Tastenbelegung. Drücken Sie nun die Taste, die Sie diesem Objekt zuordnen möchten. Sie erscheint daraufhin in der Infobox. Denken Sie daran, daß auch Tasten in Verbindung mit Shift, Control und Alternate sowie beliebige Kombinationen derselben erlaubt sind. Solange die Infobox auf dem Bildschirm steht, können Sie die Belegung des Objekts durch erneutes Drücken einer Taste beliebig oft verändern. Um die Tastenbelegung für das angewählte Objekt ganz zu löschen, klicken Sie auf den Button CLR links oben in der Infobox. Wenn Sie das Edieren des Ob-

jekts beenden wollen, klicken Sie einfach mit der Maus auf den Bereich außerhalb der Infobox. Sie können jetzt weitere Objekte edieren oder sich auch nur deren momentane Belegung anschauen. Um das Edieren der Dialogbox zu beenden, drücken Sie entweder beide Maustasten gleichzeitig oder die linke Maustaste in Verbindung mit Shift, Control oder Alternate. Sie sehen nun wieder die Auswahlbox und können weitere Dialogboxen mit Tastenbelegungen versehen. Der RESET-Button in der Auswahlbox dient zum kompletten Löschen aller Tastenbelegungen eines Objektbaums. Dazu klicken Sie zunächst den entsprechenden Baumnamen an und danach RESET. Das Häkchen vor dem Namen verschwindet jetzt - die Dialogbox ist wieder "sauber".

Um die Tastenbelegungen sämtlicher Bäume auf einmal zu löschen, klicken Sie auf den RESET-Button, ohne vorher einen bestimmten Baum auszuwählen. Es erscheint daraufhin die Sicherheitsabfrage *RESET ALL TREES?*, die Ihnen die Möglichkeit gibt, Ihre Entscheidung zu revidieren. Sie können so sehr einfach eine komplette Resource von den Tastenbelegungen befreien. Sind Sie fertig mit dem Edieren der Resource, können Sie Ihr Werk über SAVE abspeichern. Wird die Resource unter gleichem Namen abgespeichert, so finden Sie die ursprüngliche Version nachher als Backup-File mit der Extension *.RSB*.

Soviel zu der Bedienung des Key Resource Editors. Da das Programm ausführlich dokumentiert ist, möchte ich nicht auf alle Einzelheiten eingehen und nur einige Teile des Listings ansprechen. Bild 7 dokumentiert übersichtlich das Format der Definitionsdateien der verschiedenen Resource Construction Sets. Ich hatte bei der Entwicklung nur die Möglichkeit, die Formate der jeweils angegebenen Versionen zu testen. Sollten Sie ausgerechnet ein RCS besitzen, dessen Definitionsdateien vom Key Resource Editor nicht verarbeitet werden, müssen Sie versuchen, herauszufinden, inwieweit das Format von den hier dargestellten abweicht. Sollten alle Stricke reißen, bin ich gerne bereit zu helfen.

Definitions-Dateien der verschiedenen Resource Construction Sets

Digital Research RCS V1.4 (Extension .DEF)

Bytes 0,1 Anzahl der Objektnamen im Motorola-Format

danach in jeweils 16 Bytes die einzelnen Einträge:

BYTE	INHALT
0,1	0
2	Baum-Index bei untergeordneten Objekten
3	Objekt-Index, Baum-Index,...
4	0 bei Bäumen, 1 bei Objekten
5	Baumtyp: 0: Unbekannt, 1: Free-Tree (=Panel), 2: Menübaum, 3: Dialogbaum, 4: Alert
6-13	Name des Baums/Objekts
14,15	0

Bei diesem RCS werden Free Strings/Free Images nicht verarbeitet

Digital Research RCS V2.1 (Extension .DFN)

Bytes 0,1 Anzahl der Objektnamen im Intel-Format (Lo-Byte/Hi-Byte)

danach in jeweils 14 Bytes die einzelnen Einträge:

BYTE	INHALT
0	Objekt-Index, Baum-Index, Free String-Index,...
1	Baum-Index bei untergeordneten Objekten
2	Baumtyp: 0: Unbekannt, 1: Panel, 2: Menübaum 3: Dialogbaum, 4: Alert
3	0 bei Bäumen, 1 bei Objekten
4-11	Name des Baums/Objekts
12,13	0

bei Free Strings: Byte 2: 1, Byte 3: 1
bei Free Images: Byte 2: 2, Byte 3: 1

Mit dem neuen Format lassen sich nun auch die Definitions-Dateien zwischen PC-GEM und Atari ST-GEM austauschen.

Kuma RCS V1.0 (Extension .RSD)

Jeweils 16 Bytes pro Eintrag. Im ersten Eintrag haben die Bytes 0 - 3 eine besondere Bedeutung:

Bytes 0,1	Anzahl der Objektnamen im Motorola-Format
Byte 2	0 (?)
Byte 3	Kennung für erzeugte Namensdatei: Bit 0: C, Bit 1: Pascal, Bit 2: Modula, Bit 3: Fortran

BYTE	INHALT
0-3	immer 0 bis auf ersten Eintrag (s.o.)
4	Baum-Index bei untergeordneten Objekten
5	Objekt-Index, Baum-Index, Free String-Index,...
6	0 bei Bäumen, 1 bei Objekten
7	Baumtyp: 2: Menübaum, 3: Dialogbaum, 4: Alert, 5: Free String, 6: Free Image
8-15	Name des Baums/Objekts

Wie man an dem Format der drei Dateien erkennen kann, werden bei allen Resource Construction Sets die Namen von Objekten, deren Index größer als 255 ist, nicht korrekt verwaltet.

Bild 7

In dem Listing finden sich zwei Dinge, die ich gesondert hervorheben möchte. Das erste ist die Lösung eines Problems, das wohl weit verbreitet zu sein scheint, nämlich die Verwaltung von mehreren Resource-Files in einem Programm, ist doch allzu schnell die magische Grenze von 32 kByte überschritten. Prinzipiell ist es ja gar kein Problem, mehrere Resource-Files mittels RSRC_LOAD zu laden.

Speichert man beispielsweise nach dem Laden des ersten RSC-Files alle Baumadressen in Variablen, so benötigt man nach Laden des zweiten RSC-Files den RSRC_GADDR-Aufruf für die Bäume der ersten Resource nicht mehr. Dieses Prinzip funktionierte auch hundertprozentig, wäre da nicht die Freigabe des Speichers bei Beendigung des Programms mit RSRC_FREE. Damit kann man zum Abschluß des Programms nur noch den Speicher für die zuletzt geladene Resource freigeben. Programmiert man in GFA-BASIC und möchte schließlich den mit RESERVE freigegebenen Speicher wieder an das BASIC zurückgeben, folgt unweigerlich ein "Fehler bei RESERVE". Abhilfe schafft hier eine kaum dokumentierte Adresse. Diese befindet sich im GEM-internen GLOBAL-Feld. Die Adresse des GLOBAL-Feldes steht im zweiten Langwort des AES-Parameter-Feldes. In GFA-BASIC erreicht man es über GB+4. In den Elementen 7 und 8 (Offset 14) des GLOBAL-Feldes findet man nach einem RSRC_LOAD die Adresse der Resource im Speicher und im Element 9 (Offset 18) die Länge der geladenen Resource. Auf diese Einträge greifen auch die AES-Funktionen zu. Der Rest ist dann recht einfach. Man muß nur nach dem Laden der ersten Resource deren Adresse und Länge zwischenspeichern, bevor man die zweite Resource lädt. Bei Beendigung des Programms wird mit RSRC_FREE zunächst der Speicher für die zweite Resource freigegeben. Daraufhin trägt man im GLOBAL-Feld Adresse und Länge der ersten Resource ein und ruft RSRC_FREE erneut auf. So wird der belegte Speicher wieder korrekt freigegeben. Prinzipiell läßt sich dieser Vorgang

auch mit mehr als zwei RSC-Files realisieren. Weitere Einzelheiten entnehmen Sie bitte dem Listing. Die hier geschilderten Eigenschaften haben meines Wissens Gültigkeit für alle TOS-Versionen bis einschließlich Blitter-TOS, so daß man wohl davon ausgehen kann, daß es auch auf dem neuen TOS 1.4 und späteren Versionen funktioniert.

Bei näherem Betrachten des Listings wird Ihnen vielleicht auffallen, daß die zu bearbeitende Resource zweimal geladen wird. Das erste Mal mit RSRC_LOAD und das zweite Mal mittels BGET. Der Grund dafür ist folgender: Die Resource, wie sie von einem RCS erstellt wurde, liegt in einem adreß- und auflösungsunabhängigen Format vor. Sämtliche absoluten Adressen, Objektkoordinaten und -maße werden von der Funktion RSRC_LOAD nach dem Laden automatisch berechnet und eingetragen. Um die Resource abzuspeichern, müßte man sie erst wieder in ihr ursprüngliches Format zurückverwandeln. Wenn der Benutzer nun nach dem Abspeichern aber weiter an der Resource arbeiten wollte, müßte sie wiederum umgewandelt oder gar ganz neu geladen werden. Das wäre zuviel Aufwand. Stattdessen wird die mit RSRC_LOAD geladene Resource ausschließlich benutzt, um die Dialogboxen auf dem Bildschirm zu zeigen, während die vorgenommenen Tastenbelegungen in das "Duplikat" eingetragen werden. Dieses Duplikat wird auch hinterher wieder abgespeichert. Dadurch erspart man sich einiges an Programmieraufwand, und an den maximal 32 kByte zusätzlich benötigtem Speicher soll es ja wohl nicht scheitern.

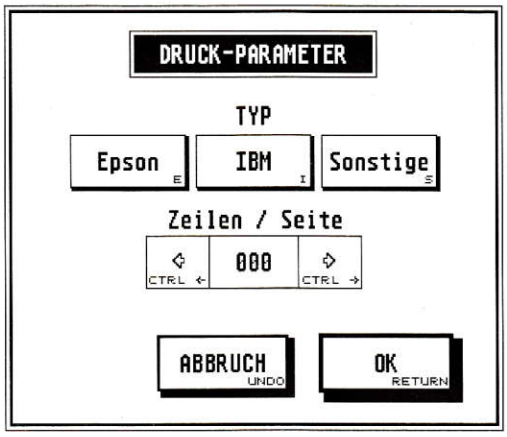


Bild 8: Beispiel für eine mit dem Key Resource Editor erstellte Dialogbox.

Abschließend noch ein paar Tips und Ratschläge. Man könnte die neue FORM_DO-Funktion dahingehend erweitern, daß sie bei Beendigung des Dialogs auch noch den Index des zuletzt edierten Textobjekts zurückliefert. So könnte der Cursor bei einem erneuten Aufruf wieder in dem zuletzt bearbeiteten Eingabefeld stehen. Das ist gerade bei Dialogboxen, die nach dem Anwählen eines Touchexit-Objekts auf dem Bildschirm stehen bleiben, sehr nützlich. Die Realisierung ist einfach. Dazu muß man lediglich die Variable *startobj&* durch *ed_obj&* ersetzen und in der Parameterliste der Funktion *ed_obj&* als VAR-Parameter deklarieren. Allerdings kann dann beim Aufruf für das Startobjekt keine Konstante mehr übergeben werden. Ich habe diese Erweiterung nicht schon von vornherein eingebaut, um Kompatibilität zur normalen FORM_DO-Routine zu gewährleisten. Denkbar wäre auch die zusätzliche Abfrage von Messages über EVNT_MULTI, damit der Benutzer auch während der Eingabe in eine Dialogbox die Möglichkeit hat, Menüeinträge anzuwählen. Der Sinn einer solchen Möglichkeit ist natürlich stark von der jeweiligen Anwendung abhängig.

Was die Gestaltung von Dialogboxen mit zusätzlichen Tastenbelegungen angeht, kann ich jedem nur raten, es nicht zu übertreiben. Immerhin soll diese Funktion eine Erleichterung für den Anwender darstellen. Die Belegung der einzelnen Objekte sollte nach bestimmten Prinzipien erfolgen und leicht einzuprägen sein. In Bild 8 sehen Sie ein kleines Beispiel einer auch für den Benutzer sichtbaren Tastenbelegung einzelner Objekte.

Tassilo Nitz

Literatur:
Atari ST Profibuch, Sybex-Verlag
ST Computer 8/87 S.20 ff

```
' ************************************
' *         EXTENDED FORM_DO         *
' *     (c) MAXON Computer GmbH      *
' * I: 08.06.89      Tassilo Nitz    *
' * U: 09.06.89      Krablerstr. 24  *
' *                  4300 Essen 12   *
' *                  0201 / 34 71 49 *
' ************************************
'
FUNCTION form_do(tree%,startobj&)
  '
  LOCAL ed_obj&,nxt_obj&,mes&,pos&,do_dial&,ox&,
        mpos&,p&,i&,buf|
  LOCAL ev_mx&,ev_my&,ev_mb&,ev_ks&,ev_key&,
        ev_clicks&
  '
  ~WIND_UPDATE(3)     !BEG_MCTRL
  '
  IF NOT BTST(OB_FLAGS(tree%,startobj&),3)
    !startobj& editable?
    nxt_obj&=@search_ob_flag(tree%,0,3,1)
  ELSE
    nxt_obj&=startobj&
  ENDIF
  ed_obj&=0       !Index des aktuellen Textobjekts
  do_dial&=TRUE
  '
  WHILE do_dial&
  '
  IF ed_obj&<>nxt_obj& AND nxt_obj&<>0   !neues
                                        Textobjekt?
    ed_obj&=nxt_obj&
    nxt_obj&=0
    ~OBJC_EDIT(tree%,ed_obj&,0,0,1,pos&) !ja,
                                  Cursor darstellen
  ENDIF
  '
  mes&=EVNT_MULTI(&X11,2,1,1,0,0,0,0,0,0,0,0,0,
                  0,0,0,ev_mx&,ev_my&,ev_mb&,ev_ks&,
                  ev_key&,ev_clicks&)
  '
  IF BTST(mes&,0)              !Tastendruck
    '
    do_dial&=@process_key(tree%,ed_obj&,ev_ks&,
                          ev_key&,nxt_obj&,ev_key&)
    '
    IF ev_key& !keine Steuertaste
      ~OBJC_EDIT(tree%,ed_obj&,ev_key&,pos&,2,
                 pos&)
                 !in Textobjekt einfügen
    ENDIF
    '
  ELSE IF BTST(mes&,1)         !Maustaste
    nxt_obj&=OBJC_FIND(tree%,0,8,ev_mx&,ev_my&)
                 !Objekt unter Mauszeiger
    '                                          →
```

```
46:         IF nxt_obj&=-1   !Maus auPerhalb der
              Dialogbox
47:           OUT 2,7
48:           nxt_obj&=0
49:         ELSE
50:           IF BTST(OB_FLAGS(tree%,nxt_obj&),3)
                    !Editable?
51:             ~OBJC_OFFSET(tree%,nxt_obj&,ox&,dummy&)
                    !X-Koord. Textobjekt
52:             ~GRAF_HANDLE(charw&,dummy&,dummy&,
                          dummy&)
                    !Breite eines Zeichens
53:             mlen&=CARD{OB_SPEC(tree%,nxt_obj&)+26}
                    !Länge der Textmaske
54:             SELECT CARD{OB_SPEC(tree%,nxt_obj&)+16}
                    !Textausrichtung
55:             CASE 0 !linksbündig
56:               mpos&=TRUNC((ev_mx&-ox&)/charw&+0.5)
57:               ' rechtsbündig: nur der
                    Vollständigkeit halber,
                    werden vom AES
58:             CASE 1    !sowieso nicht korrekt
                          verwaltet
59:               mpos&=TRUNC((ev_mx&-ox&-OB_W(tree%,
                      nxt_obj&)+mlen&*charw&)/
                    charw&-0.5)
60:             CASE 2      !zentriert
61:               mpos&=TRUNC((ev_mx&-ox&-(OB_W(tree%,
                      nxt_obj&)-mlen&*charw&)/2)/
                    charw&)
62:             ENDSELECT
63:             IF mpos&>0
64:               CLR p&,i&
65:               REPEAT    !Zeichenposition bestimmen
66:                 IF BYTE{{OB_SPEC(tree%,nxt_obj&)+4}+
                          i&}=95   !"_"
67:                   INC p&
68:                 ENDIF
69:                 INC i&
70:               UNTIL i&=mpos& OR i&=mlen&
71:             ELSE
72:               p&=0
73:             ENDIF
74:             ~OBJC_EDIT(tree%,ed_obj&,0,pos&,3,pos&)
                  !Positionierung des
75:             buf|=BYTE{{OB_SPEC(tree%,nxt_obj&)}+p&}
                  !Cursors mit einem
76:             BYTE{{OB_SPEC(tree%,nxt_obj&)}+p&}=0
                  !kleinen Trick
77:             ~OBJC_EDIT(tree%,nxt_obj&,0,pos&,1,pos&)
78:             BYTE{{OB_SPEC(tree%,nxt_obj&)}+p&}=buf|
79:                   ed_obj&=nxt_obj&       !aktuelles
                                Textobjekt neu setzen
80:           ENDIF
81:           do_dial&=FORM_BUTTON(tree%,nxt_obj&,
                                ev_clicks&,nxt_obj&)
82:         ENDIF
83:         '
84:       ENDIF
85:       '
86:       IF do_dial&=0 OR (nxt_obj&<>0 AND
            nxt_obj&<>ed_obj&) !wenn neues
                              Textobjekt
87:         ~OBJC_EDIT(tree%,ed_obj&,0,pos&,3,pos&)
                              !oder Ende des Dialogs
88:       ENDIF
89:       '
90:     WEND
91:     '
92:     ~WIND_UPDATE(2)     !END_MCTRL
93:     RETURN nxt_obj&
94:     '
95:   ENDFUNC
96:   '
97:   FUNCTION process_key(tree%,obj&,kstate&,ev_key&,
                        VAR nxt_obj&,nxtchar&)
98:     LOCAL key_obj&
99:     '
100:    IF BTST(kstate&,0)
101:      kstate&=BSET(kstate&,1)
102:    ENDIF
103:    kstate&=SHR&(kstate&,1)
104:    '
105:    nxtchar&=0
106:    SELECT ev_key&
107:    CASE &H4D36   !SHIFT/Cursor right
108:      ~OBJC_EDIT(tree%,ed_obj&,0,pos&,3,pos&)
109:      ~OBJC_EDIT(tree%,ed_obj&,0,pos&,1,pos&)
110:      '
111:    CASE &H4B34   !SHIFT/Cursor left
112:      ~OBJC_EDIT(tree%,ed_obj&,0,pos&,3,pos&)
113:      buf|=BYTE{{OB_SPEC(tree%,ed_obj&)}}
114:      BYTE{{OB_SPEC(tree%,ed_obj&)}}=0
115:      ~OBJC_EDIT(tree%,ed_obj&,0,pos&,1,pos&)
116:      BYTE{{OB_SPEC(tree%,ed_obj&)}}=buf|
117:      '
118:    CASE &H5000    !Cursor down
119:      nxt_obj&=@search_ob_flag(tree%,obj&,3,1)
120:      '
121:    CASE &H4800    !Cursor up
122:      nxt_obj&=@search_ob_flag(tree%,obj&,3,-1)
123:      '
124:    CASE &H5032,&H4737    !SHIFT/Cursor down  SHIFT/
                              HOME
125:      REPEAT
126:        nxt_obj&=obj&
127:        obj&=@search_ob_flag(tree%,obj&,3,1)
128:      UNTIL obj&=nxt_obj&
129:      '
130:    CASE &H4838,&H4700    !SHIFT/Cursor up / HOME
131:      nxt_obj&=@search_ob_flag(tree%,0,3,1)
132:      '
133:    CASE &H1C0D,&H720D    !RETURN/ENTER
134:      nxt_obj&=@search_ob_flag(tree%,0,1,1)
            !Default-Objekt vorhanden?
135:      IF nxt_obj&
            !ja, dann Mausklick simulieren
136:        RETURN FORM_BUTTON(tree%,nxt_obj&,1,
                              nxt_obj&)
137:      ELSE       !kein Default-Objekt
138:        nxt_obj&=@search_ob_flag(tree%,obj&,3,1)
            !nächstes Textobjekt suchen
139:        IF nxt_obj&=obj& AND @search_ob_flag(tree%,
                                              0,2,1)=0
140:          RETURN FALSE          !kein Exit-Obj und
                      Cursor im letzten Textobjekt
141:        ENDIF
142:      ENDIF
143:      '
144:    DEFAULT   !keine Steuertaste => Objekt mit
                    passender Tastenbelegung suchen
145:      key_obj&=@search_key_obj(tree%,kstate&,
            ev_key&)
146:      IF key_obj&
            !Objekt gefunden
147:        RETURN FORM_BUTTON(tree%,key_obj&,1,
              nxt_obj&) !Mausklick simulieren
148:      ENDIF
149:      nxt_obj&=obj&
150:      nxtchar&=ev_key&
151:    ENDSELECT
152:    '
153:    RETURN TRUE
154:    '
155:  ENDFUNC
156:  FUNCTION search_key_obj(tree%,kstate&,ev_key&)
157:    LOCAL key|,obj&
158:    key|=SHR&(ev_key&,8)        !nur SCAN-Code
159:    '
160:    obj&=0
161:    REPEAT
162:      INC obj&
163:      IF key|=SHR&(OB STATE(tree%,obj&),8) AND
            kstate&=SHR&(OB_FLAGS(tree%,obj&),13)
164:        RETURN obj&
165:      ENDIF
166:    UNTIL BTST(OB_FLAGS(tree%,obj&),5)
167:    '
168:    RETURN 0
169:  ENDFUNC
170:  FUNCTION search_ob_flag(tree%,startobj&,bitnr|,
                            direction&)
171:    LOCAL obj&
172:    obj&=startobj&
173:    REPEAT
174:      ADD obj&,direction&
175:      IF BTST(OB_FLAGS(tree%,obj&),bitnr|)
176:        RETURN obj&
177:      ENDIF
178:    UNTIL (obj&=0) OR BTST(OB_FLAGS(tree%,obj&),5)
179:    RETURN startobj&
180:  ENDFUNC
```



MODULA-2 KURS

Willkommen zum zehnten Teil des Modula-Kurses. In dieser Folge geht es um eine Eigenschaft von Modula, die andere auf Mikros verbreitete Sprachen wie Pascal oder C nicht kennen: die Möglichkeit zur nebenläufigen Programmierung. Doch zunächst noch ein anderes Thema.

Prozedurvariablen

Noch immer ist die Auflistung der Datentypen in Modula-2 nicht komplett. Für das heutige Hauptthema benötigen wir eine neue Art von Variablen, die Prozedurvariablen. Sie erhalten nicht einen "normalen" Wert, sondern können eine ganze Prozedur speichern. Übrigens waren Prozedurvariablen auch schon in Pascal vorhanden, wurden jedoch kaum auf Mikros implementiert.

Wie war das? Eine ganze Prozedur in einer Variablen ablegen? Nun, in einer Modula-Implementierung wird sicherlich keine ganze Prozedur abgespeichert, lediglich ein Verweis auf eine Prozedur bildet eine Prozedurvariable.

Der Umgang mit Prozedurvariablen unterscheidet sich zunächst nicht von den bisher bekannten. Sie müssen deklariert werden, es gibt Zuweisungen, und sie können in Ausdrücken verwendet werden.

Welchen Typ hat nun eine Prozedur? Sie besteht aus einer Reihe von Anweisungen, lokalen Variablen und Parametern. Die internen Vorgänge interessieren nicht nach außen, lediglich die Parametertypen und ihre Reihenfolge sind bekannt. Und genau sie charakterisieren den Typ einer Prozedur.

Eine Prozedurvariable wird durch das Schlüsselwort *PROCEDURE* und eine Liste der Parametertypen deklariert:

```
VAR p,q:PROCEDURE(INTEGER, INTEGER, REAL);
```

p und q sind nun Variablen, die jede Prozedur aufnehmen können, die als Parameter zwei INTEGERs und ein REAL verlangt. Zum Speichern von Funktionen muß auch der Ergebnistyp in gewohnter Weise notiert werden:

```
VAR f,g:PROCEDURE(REAL):REAL;
```

Hier kann jede Prozedur aufgenommen werden, die ein REAL als Parameter erhält und ein REAL als Ergebnis abliefert.

Wie funktioniert eine Zuweisung? Um eine Prozedur zuzuweisen, muß sie natürlich zunächst deklariert werden, denn es gibt ja keine vordefinierten Werte wie z.B. bei den Zahlen:

```
PROCEDURE a(x,y:INTEGER; z:REAL);
BEGIN
 ...
END a;
```

und für das Beispiel mit Funktionen:

```
PROCEDURE double(x:REAL):REAL;
BEGIN
  RETURN 2.0*x;
END double;

PROCEDURE triplicate(x:REAL):REAL;
BEGIN
  RETURN 3.0*x;
END triplicate;
```

Nun ist die Zuweisung *p:=a;* legal. *a* ist eine Prozedur mit zwei INTEGER- und einem REAL-Parameter. *p* hat den ent-

sprechenden Typ, und die Zuweisung kann durchgeführt werden.

Ebenso kann man schreiben: *q:=p;*. Beide Variablen haben den gleichen Typ, und *q* erhält den Inhalt von *p*. Die Behandlung von Funktionen läuft ebenso ab.

Nun stellt sich die Frage, was das alles bringt. Bisher sind nur Prozeduren - oder Referenzen auf sie - in Variablen abgespeichert. Wir können damit Zuweisungen ausführen, mehr aber auch nicht, vom Rechnen ganz zu schweigen.

Aber man kann mit Prozedurvariablen das machen, was man auch mit Prozeduren macht: sie aufrufen. Und da sie variabel sind, kann ihr Inhalt während der Laufzeit des Programms bestimmt werden. Die Aufrufe von Prozeduren in einem Programm sind einerseits nicht mehr vorhersehbar, andererseits erhalten wir damit eine ungeheure Flexibilität.

Ist wie oben irgendwann der Prozedurvariablen *p* die Prozedur a zugewiesen, kann die Variable in der Art einer Anweisung notiert und damit der Aufruf von *a* ausgelöst werden:

```
...
p(1,2,42.3);
...
```

Wird der aktuelle Wert von *p* eingesetzt, entspricht das dem Aufruf *a(1,2,42.3);*. Funktionsaufrufe werden immer in Ausdrücken eingesetzt und liefern einen entsprechenden Wert ab. Prozedurvariablen, die Funktionen als Werte erhalten, müssen ebenso behandelt werden:

```
...
f:=double;
x:=f(1.0);
g:=triplicate;
x:=f(1.0)+g(3.0);
...
```

Setzt man die Werte der Prozedurvariablen ein, erhielte *x* zunächst 2.0 und danach 2.0+9.0=11.0. Es ist natürlich auch möglich, die Prozedurvariablen mit Funktionen zu schachteln:

```
...
x:=f(g(2.0));
...
```

würde *x* den Wert 12.0 zuweisen.

Weiterhin lassen sich Prozedurvariablen auch als Parameter für Prozeduren verwenden. Sie werden ganz normal in der Parameterliste notiert:

```
TYPE IntProz =PROCEDURE(INTEGER);
...
PROCEDURE Count(a:INTEGER;
p:IntProz);
...
```

Anworten von Teil IX

1. Die geforderte Prozedur muß so aussehen:

```
MODULE A1;

FROM SYSTEM  IMPORT TSIZE;
FROM Storage IMPORT ALLOCATE,
DEALLOCATE;

TYPE INTPOINTER = POINTER TO
                  INTEGER;

PROCEDURE NewInt(wert:INTEGER;
           VAR zeiger:INTPOINTER);
BEGIN
  ALLOCATE(zeiger,TSIZE(INTEGER));
  zeiger^:=wert;
END NewInt;
```

Der zweite Parameter von *NewInt* - der Zeiger - muß mit einem eigenen Typen versehen werden, da es sich bei POINTER TO INTEGER nicht um einen einfachen Typen handelt. *NewInt* richtet mit ALLOCATE die dynamische INTEGER-Variable ein und weist ihr den übergebenen Wert per Dereferenzierung zu.

2. Ampeln sind entsprechend dem Konzept von abstrakten Datentypen implementiert: Im Definitions-Modul werden lediglich der Typ *Ampel* und die gewünschten und notwendigen Prozeduren zur Arbeit damit bereitgestellt. Zur Beschreibung von Werten, die Ampeln annehmen können, gibt es schließlich noch die Typen *AmpelFarbe* und AmpelFarben. Damit ist es möglich, Ampelfarben als Konstanten im Programm zu notieren. Man könnte auch noch weitere Routinen SetRot, SetGruen etc. einführen und dann auf diese Typen verzichten.

```
DEFINITION MODULE Ampeln;
(* R.Tolksdorf/ST-Computer
   Version*: 14.09.8915:1 *)

TYPE Ampel;
     AmpelFarben = (Rot, Gelb,
                    Gruen);
     AmpelFarbe  = SET OF
                   AmpelFarben;

PROCEDURE MakeAmpel(VAR a:Ampel);

PROCEDURE KillAmpel(VAR a:Ampel);

PROCEDURE SetAmpel(a:Ampel;
                   f:AmpelFarbe);

PROCEDURE
GetAmpel(a:Ampel):AmpelFarbe;

PROCEDURE SwitchAmpel(a:Ampel);

END Ampeln.
```

Das Implementationsmodul realisiert *Ampel* als Zeiger auf eine Menge vom Typ *AmpelFarbe*, in der der Zustand der Ampel gehalten wird. *MakeAmpel* und *KillAmpel* richten die dynamische Variable per *ALLOCATE* und *DEALLOCATE* ein. Sicherheitshalber wird beim Löschen der Ampel der Zeiger auf NIL gesetzt.

GetAmpel und *SetAmpel* benutzen wie gewohnt eine dynamische Variable per Dereferenzierung. *SwitchAmpel* ist schon aus der zweiten Aufgabe im achten Teil dieser Serie bekannt, arbeitet hier aber auf einer dynamischen Variablen.

```
IMPLEMENTATION MODULE Ampeln;
(* R. Tolksdorf/ST-Computer
   Version*: 14.09.89   15:19 *)
FROM SYSTEM  IMPORT TSIZE;
FROM Storage IMPORT ALLOCATE,
DEALLOCATE ;

TYPE Ampel = POINTER TO
             AmpelFarbe;
(*von  AmpelFarben =(Rot, Gelb,
                     Gruen);
 .DEF  AmpelFarbe  =SET OF
                    AmpelFarben;*)

PROCEDURE MakeAmpel(VAR a:Ampel);
BEGIN
  ALLOCATE(a,TSIZE(AmpelFarbe));
END MakeAmpel;

PROCEDURE KillAmpel(VAR a:Ampel);
BEGIN
  DEALLOCATE(a);
  a:=NIL;
END KillAmpel;

PROCEDURE SetAmpel(a:Ampel;
                   f:AmpelFarbe);
BEGIN
  a^:=f;
END SetAmpel;

PROCEDURE
GetAmpel(a:Ampel):AmpelFarbe;
BEGIN
  RETURN a^;
END GetAmpel;

(* Entspricht der Antwort zur
   2.Hausaufgabe aus Teil 8  *)

PROCEDURE SwitchAmpel(a:Ampel);
BEGIN
  IF    (Gruen IN a^) THEN
    a^:=AmpelFarbe{Gelb}
  ELSIF ((Rot IN a^) AND
         (Gelb IN a^)) THEN
    a^:=AmpelFarbe{Gruen}
  ELSIF (Gelb IN a^) THEN
    a^:=AmpelFarbe{Rot}
  ELSE
    a^:=AmpelFarbe{Rot,Gelb}
  END;
END SwitchAmpel;

END Ampeln.
```

→

Beim Aufruf von *Count* muß eine Prozedurvariable des Typs *IntProz* übergeben werden.

Die Flexibilität in diesen Beispielen ist noch nicht ganz offensichtlich. Sie könnten aber ein Programm schreiben, in dem der Benutzer zur Laufzeit zwischen verschiedenen Funktionen auswählt, auf die Routinen angewendet werden können. Denken Sie nur an einen Funktions-Plotter. In einer CASE-Anweisung würde dann einer Variablen *f* die ausgewählte Funktion zugewiesen. Die restlichen Teile des Programms würden nur noch mit der Variablen *f* arbeiten und sind damit unabhängig von der statischen Notierung eines bestimmten Funktionsaufrufs.

Abschließend bleibt noch die Frage, welchen Typ parameterlose Prozeduren haben. Bei ihnen ist die Liste der Parametertypen leer, jedoch muß man - wie bei Funktionen - die Klammern notieren:

```
TYPE ParamloseProz = PROCEDURE();
```

Für *PROCEDURE()* ist die Abkürzung *PROC* vordefiniert.

Prozedurvariablen haben also einen Typ, der sich aus den Parametertypen und deren Reihenfolge ergibt. Sie können in Zuweisungen verwendet werden, wobei es allerdings keinerlei Rechenoperationen gibt. Eine Prozedurvariable kann als Anweisung verwendet werden, was einem Aufruf der Prozedur entspricht, die sie als Wert hat.

Nebenläufigkeit

Üblicherweise hört man viel von Multitasking-Betriebssystemen, z.B. beim AMIGA oder dem erheblich professionelleren UNIX, das es in unzähligen Versionen auch z.B. als XENIX oder SINIX gibt. Auf dem ATARI ST gibt es mehrere Multitasking-Betriebssysteme wie OS-9 oder MINIX. Modula-2 ermöglicht es, nebenläufige Programme unabhängig vom Betriebssystem zu schreiben.

Aber um nicht zuviele Hoffnungen zu wecken: Mit den Konstrukten von Modula direkt kann man nicht programmieren, daß zwei Zuweisungen gleichzeitig ausgeführt werden sollen. Jedoch lassen sich über Standard-Features von Modula Module programmieren, die eine solche parallele Ausführung unterstützen.

Doch zunächst ein paar theoretische Überlegungen. Wir wollen, daß der Rechner mehrere Dinge gleichzeitig erledigt, also parallel. Ein Mikro-Computer hat aber typischerweise nur einen Prozessor, der immer nur einen Rechenschritt gleichzeitig ausführen kann. Daher arbeitet er immer sequentiell, d.h. die Anweisungen werden Schritt für Schritt nacheinander ausgeführt.

Selbst in der Hochsprache Modula-2 findet sich diese Sequentialität. Das Semikolon ";" trennt nach allem, was Sie bisher gelernt haben, nur die einzelnen Statements, so daß der Compiler das Ende einer Anweisung erkennen kann. Man kann es aber auch so interpretieren, daß ";" bedeutet, "führe das Statement vor dem ; aus und *danach* das nächste". Damit sequentialisiert das Semikolon den Programmtext.

Da immer nur ein Statement ausgeführt wird, weiß man genau, welchen Zustand die Variablen vor und nach einem Statement haben. Hätten wir eine parallele Programmierung, bei der zwei Statements gleichzeitig zu zwei anderen ausgeführt würden, könnte man nicht genau sagen, welches Statement wann beendet ist. Damit wäre auch der Variablenzustand während der Ausführung nicht feststellbar. Es tun sich also Probleme auf, Zustände zu bestimmten Zeitpunkten vorherzusagen.

Was für eine Maschine bräuchte man, um zwei Anweisungen gleichzeitig auszuführen? Einen Rechner mit zwei Prozessoren natürlich. Die Anweisungen müß-

Anworten von Teil IX

Ihre Implementierung müßte die gleiche Struktur im Definitionsmodul aufweisen und auch beim Umgang mit den dynamischen Variablen der hier abgedruckten Lösung sehr ähnlich sein.

3. Das Programm zum Aufsummieren beliebig vieler Werte und zur Berechnung von deren Mittelwert lautet:

```
MODULE A3;
FROM INTLists IMPORT List,MakeList,
KillList, First, AtLast,
                    Next,SetValue,
GetValue, AppendElement,
                    AtFirst;
FROM InOut    IMPORT WriteString,
WriteLn, WriteInt, ReadInt;
VAR L:List;
    i, WerteAnzahl, Wert, Summe :
INTEGER;
BEGIN
 WriteString('Wieviele Werte? ');
 ReadInt(WerteAnzahl); WriteLn;
 WriteString('Werte eingeben');
 WriteLn;
 MakeList(L);
 FOR i:=1 TO WerteAnzahl DO
   ReadInt(Wert); WriteLn;
   AppendElement(L);
   SetValue(L,Wert);
 END;
 Summe:=0;
 First(L);
 REPEAT
   Summe:=Summe+GetValue(L);
   Next(L);
 UNTIL AtLast(L);
 IF ~AtFirst(L) THEN
   Summe:=Summe+GetValue(L);
 END;
 WriteString('Summe aller Werte:');
 WriteInt(Summe,5); WriteLn;
 WriteString('Mittelwert:');
 WriteInt(Summe DIV WerteAnzahl,5);
WriteLn;
 KillList(L);
END A3.
```

Nach dem Abfragen der Anzahl der zu verarbeitenden Werte wird per *MakeList* die Liste eingerichtet. Die einzelnen Zahlen fragt das Programm in der FOR-Schleife ab. In jedem Durchlauf richtet es ein neues Listenelement per *AppendElement* ein und schreibt den eingegebenen Wert mit *SetValue* in die erzeugte dynamische Variable.

Im Verarbeitungsteil muß die Summe aller Werte ermittelt und in der Variablen *Summe* vermerkt werden. Dazu setzt *First* den aktuellen Listenzeiger an den Anfang. Mit *GetValue* wird der aktuelle Werte Summe hinzuaddiert und danach der Listenzeiger mit *Next* ein Element weitergesetzt. Die Schleife endet, wenn das letzte Element erreicht ist.

Dieses muß nun noch hinzuaddiert werden. Es gibt dabei eine Ausnahme, nämlich die Liste mit nur einem Element. Dann liefern *AtFirst* und *AtLast* TRUE, und der Wert ist schon in der Schleife verarbeitet worden.

Der Rest des Programms gibt einfach die Summe und den errechneten Mittelwert aus. Abschließend säubert der Aufruf von *KillList* den Speicher.

ten jeweils einem Prozessor zugewiesen werden, der sie dann ausführt. Aber was wäre, wenn wir jetzt drei Anweisungen gleichzeitig rechnen lassen wollten, oder fünf oder ...?

Da man nicht jedesmal einen neuen Rechner bauen kann, muß das Problem per Software gelöst werden. Diese Software hat die Aufgabe, die vorhandenen Prozessoren so zu steuern, daß sie ein Stück an dieser, ein Stück an jener Anweisung arbeiten. Ist diese Steuerung einigermaßen gerecht, werden alle Anweisungen irgendwann abgearbeitet sein.

Aber ist das noch wirklich parallel? Nein, denn die vorhandenen Prozessoren arbeiten *nacheinander* einmal an dieser, einmal an jener Aufgabe. Richtigerweise muß man ein solches Verhalten "quasi-parallel" nennen.

Aber wir schweifen bei der Betrachtung des Problems etwas ab. Wir wollen lediglich mehrere Anweisungen quasi-parallel auf einem Prozessor ablaufen lassen. D.h. es gibt nur einen Prozessor, der auf die verschiedenen Aufgaben möglichst gerecht angesetzt werden muß. Genau das läßt sich in Modula unabhängig vom Betriebssystem beschreiben.

Genauer betrachtet

Der Begriff "verschiedene Aufgaben" ist vielleicht etwas zu schwammig. Eine Reihe von Statements, die zusammen ein Problem bilden, an dem der Prozessor gleichzeitig zu einem anderen arbeiten soll, nennen wir entsprechend den Bezeichungen für Modula-2 "Co-Routine". Stellen Sie sich unter einer Co-Routine eine Art Prozedur vor, die quasi-parallel zu einer anderen ausgeführt werden soll.

Eine solche Co-Routine hat einen bestimmten Zustand. Dieser besteht einerseits - wie bei einer Prozedur - aus einer Reihe von lokalen Variablen, die im Rechner auf einem Stapelspeicher abgelegt sind. Für jede Co-Routine ist ein Speicherbereich nötig, in dem diese Variablen liegen. Da der Prozessor "immer ein Stück" in einer Co-Routine arbeiten und dann zu einer anderen wechseln soll, gehört zum Zustand einer Co-Routine auch noch die Information darüber, an welcher Stelle der Prozessor weiterarbeiten muß, wenn er "ein weiteres Stück" darin rechnen soll.

Was geschieht, wenn der Prozessor veranlaßt wird, in einer anderen Co-Routine zu arbeiten? Sämtliche Zustandsinformationen der aktuellen Co-Routine müssen gerettet werden, eine neue Co-Routine ist auszuwählen und deren Zustand muß wiederhergestellt werden.

Dieses Umschalten in eine andere Co-Routine nennt man Kontextwechsel. Es handelt sich um die grundlegenste Operation, die man für parallele Routinen benötigt.

```
MODULE CoDem;
FROM SYSTEM    IMPORT ADDRESS, ADR ;
FROM Process   IMPORT PROCESS, NEWPROCESS, TRANSFER ;

VAR Ko1, Ko2 : PROCESS;
    Zu1, Zu2 : ARRAY [0..1000] OF INTEGER;

PROCEDURE K1 ...
PROCEDURE K2 ...

BEGIN
  NEWPROCESS(K1,ADR(Zu1),SIZE(Zu1),Ko1);
  NEWPROCESS(K2,ADR(Zu2),SIZE(Zu2),Ko2);
  ...
```

Listing 1

Jetzt in Modula

In einem ersten Beispiel sollen zwei Co-Routinen arbeiten, von denen eine von 0 an einen INTEGER hochzählt und die Werte ausgibt. Die zweite Co-Routine arbeitet analog, nur daß von 1000 herabgezählt werden soll. Die entsprechenden Anweisungen sind:

```
PROCEDURE K1;
VAR i:INTEGER;
BEGIN
 i:=-1;
 LOOP
  i:=i+1;
  WriteInt(i,5); WriteLn;
 END;
END K1;
```

und

```
PROCEDURE K2;
VAR i:INTEGER;
BEGIN
 i:=1001;
 LOOP
  i:=i-1;
  WriteInt(i,5); WriteLn;
 END;
END K2;
```

Die Co-Routinen sind zunächst mit unbedingten Schleifen versehen; eine Abfrage auf einen Endwert wird später eingebaut.

Nun sind Co-Routinen aber nicht einfach Prozeduren. In einem normalen Programm gibt es einen einzigen Zustand und einen Punkt, an dem momentan gerechnet wird. Bei Co-Routinen benötigt jede einen eigenen Zustand. In Modula muß dieser zunächst eingerichtet und die Co-Routine angemeldet werden.

Alle Prozeduren für Co-Routinen sind im Modul *Coroutines* oder *Process* zusammengefaßt (in älteren Systemen können sie auch in SYSTEM stehen). In LPR-Modula müssen Sie *Process* benutzen.

Die Prozedur zum Einrichten einer Co-Routine heißt *NEWPROCESS*. Sie erhält vier Parameter. Der erste ist eine parameterlose Prozedur, die die Co-Routine beschreibt, also im Beispiel *K1* bzw. *K2*.

Die nächsten beiden Parameter beschreiben einen Speicherbereich, in dem der Zustand der Co-Routine gehalten wird, also der Stack und weitere Informationen. Benötigt wird die Angabe eines Speicherbereiches und dessen Größe. Schließlich setzt *NEWPROCESS* eine Co-Routinen-Variable, über die die installierte Co-Routine angesprochen werden kann. Sie hat hier den Typ *PROCESS* aus *Process*. Dies ist allerdings nur wieder speziell für den LPR-Compiler nötig. Es kann sein, daß in Ihrem System ADDRESS verwendet wird. *PROCESS* ist zwar als ein *POINTER* implementiert und eigentlich ist ja ein Zeiger kompatibel zu ADDRESS. Wegen der Typenstriktheit bemängelt LPR jedoch einen Fehler.

Im Beispiel müßte das Rahmenprogramm mit dem Anmelden der Co-Routinen wie in Listing 1 aussehen. Es werden zwei Co-Routinen angemeldet. Die Prozedur *K1* beschreibt die Co-Routine *Ko1* und erhält den Speicher ab *ADR(Zu1)* mit der Größe *SIZE(Zu1)*. Analog wird *Ko2* eingerichtet. NEWPROCESS erzeugt die Co-Routinen, indem der Code der Prozeduren kopiert und ihnen der Speicher nach den übergebenen Parametern zugewiesen wird. Schließlich setzt die Routine die Co-Routinen-Variablen.

Oben wurde beschrieben, daß der Wechsel zwischen zwei Co-Routinen die grundlegende Operation für nebenläufige Programme ist. In Modula übernimmt diese Aufgabe die Prozedur *TRANSFER*.

Sie hat zwei Parameter, nämlich zwei Co-Routinen-Variablen. Das ergibt sich aus den Aufgaben von TRANSFER. Die Operation muß den aktuellen Zustand abspeichern und einen anderen wiederherstellen. Zu beidem wird jeweils eine Co-Routinen-Variable benötigt.

Erweitern wir also die beiden Co-Routi-

nen so, daß sie jeweils einen TRANSFER durchführen.

```
PROCEDURE K1;
VAR i:INTEGER;
BEGIN
  i:=-1;
  LOOP
    i:=i+1;
    WriteInt(i,5); WriteLn;
    TRANSFER(Ko1,Ko2);
  END;
END K1;
```

und

```
PROCEDURE K2;
VAR i:INTEGER;
BEGIN
  i:=1001;
  LOOP
    i:=i-1;
    WriteInt(i,5); WriteLn;
    TRANSFER(Ko2,Ko1);
  END;
END K2;
```

Was geschieht nun, wenn die beiden Co-Routinen ausgeführt werden? Nehmen wir an, momentan ist *K1* in der Schleife bei der Arbeit. Es werden die Operationen *i:=i+1;* und *Write...;* ausgeführt. Sie verändern den Zustand von *K1*, nämlich durch das Schreiben in *i*. Als nächste Anweisung folgt der TRANSFER. Er soll von der Co-Routine, die durch *Ko1* bezeichnet wird, in die umschalten, die unter *Ko2* angemeldet wurde. Im Bild 1 sehen Sie diesen und die folgenden Schritte dargestellt. Links steht *Ko1*, rechts *Ko2*, jeweils einschließlich ihres Zustands, also dem Programmzähler PC und der lokalen Variablen. Die jeweils schwarz dargestellte Co-Routine ist aktiv. Momentan sieht die Situation wie in A) aus.

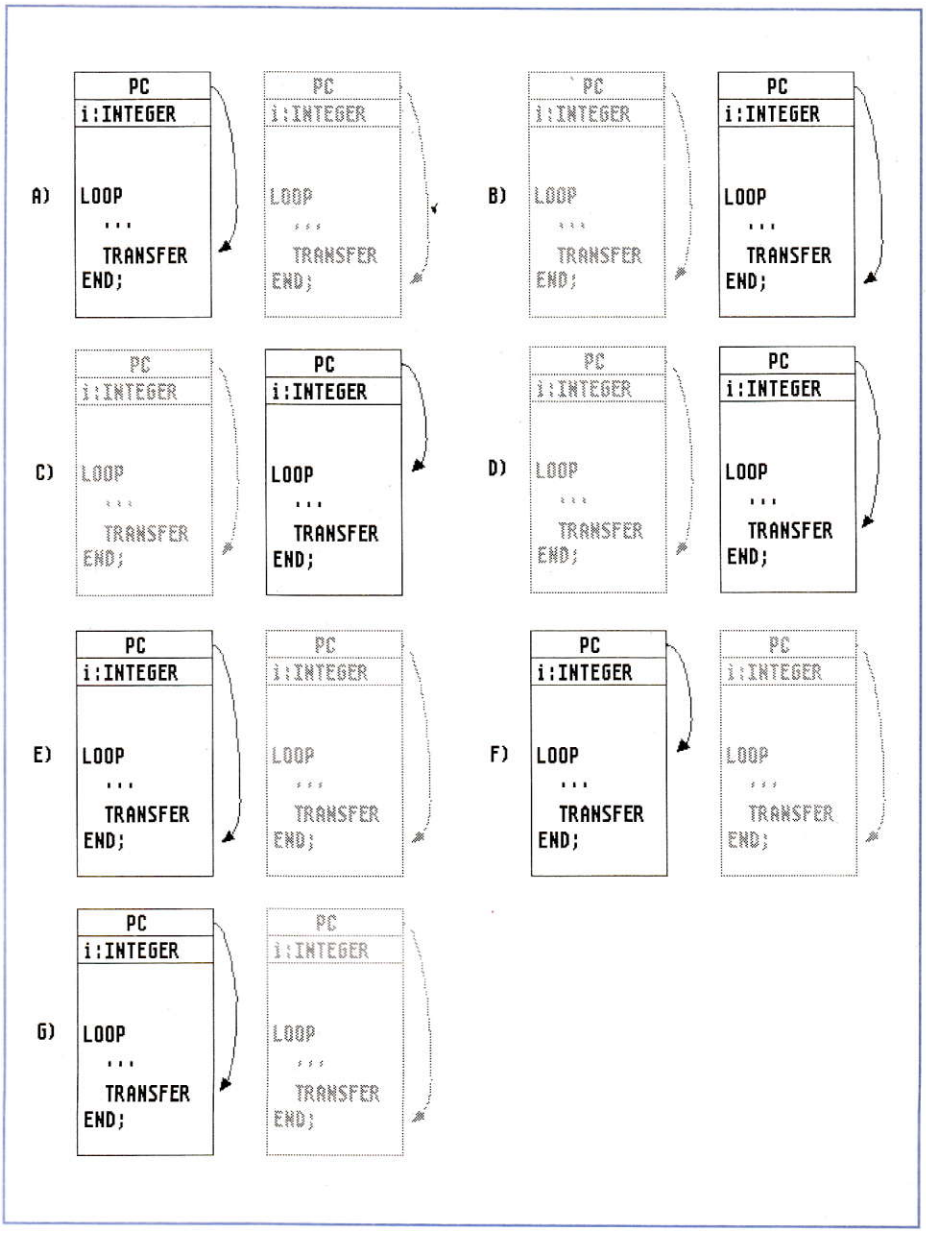


Bild 1: Die verschiedenen Zustände der Koroutinen bei den TRANSFERs

Was passiert? Der Zustand von *K1* wird in der Co-Routinen-Variablen *Ko1* abgelegt. Sie enthält insbesondere den Wert des Programmzählers, der besagt, daß als nächstes der Schleifensprung von LOOP...END ausgeführt werden soll.

Nehmen wir an, auch *K2* war vorher genau an dieser Stelle. In *Ko2* ist also der Zustand so vermerkt, daß die nächste auszuführende Anweisung die nach dem TRANSFER ist. Bei TRANSFER wird der Zustand wieder restauriert (Schritt B). Wir befinden und nun in *Ko2*. Vermerkt war dort der Programmzähler, also führt der Rechner den Sprung in *K2* aus (Schritt C). Die folgenden Anweisungen sind *i:=i-1;* und *Write...;*.

Nach diesen Anweisungen folgt in *K2* wiederum ein TRANSFER, und zwar von *Ko2* nach *Ko1* (Schritt D). Wie sieht die Situation nun aus? Gespeichert wird zunächst in *Ko2* der aktuelle Zustand, d.h. der Zustand der Variablen, also *i* und der Programmzähler, der auf das Statement nach dem TRANSFER in *K2* verweist.

Im zweiten Schritt kann *Ko1* restauriert werden. Wie oben beschrieben, sind dort die Variable *i* und der Programmzähler vermerkt. Dieser zeigt auf das Statement nach dem TRANSFER in *K1* (Schritt E). Damit beginnt das Spielchen von neuem (Schritt F und G).

Aber ganz realistisch ist diese Beschreibung noch nicht. Zunächst muß der ganze Ablauf des Hin- und Herschaltens zwischen den Co-Routinen angeworfen werden. Und er muß irgendwann zu einem Ende kommen.

Zum Starten der Co-Routinen muß irgendwann die Kontrolle vom Hauptprogramm - damit sind die "normalen" Statements zwischen BEGIN und END eines Moduls gemeint - auf eine der Co-Routinen übergehen. Sehen wir das Hauptprogramm als Co-Routine an, für die schon Arbeitsspeicher eingerichtet wurde, so reicht ein einfaches TRANSFER, um die Co-Routinen in Gang zu bringen. Dazu benötigt wird natürlich eine dritte Co-Routinen-Variable, *Main*. Der TRANSFER speichert in ihr den Zustand des Hauptprogramms, den Programmzähler.

Und wie enden Co-Routinen? Im Prinzip nie, ein reales Programm muß jedoch irgendwann zum Schluß kommen. Daher wird in einer Co-Routine eine Abfrage eingebaut, die eventuell mit einem abschließendem TRANSFER zum Hauptprogramm zurückschaltet. Dort kann es dann wie in einem normalen Programm

```
MODULE CoDem;
FROM SYSTEM     IMPORT ADDRESS, ADR ;
FROM Process    IMPORT PROCESS, NEWPROCESS, TRANSFER ;
FROM InOut      IMPORT WriteInt, WriteLn;

VAR Ko1, Ko2, Main : PROCESS;
    Zu1, Zu2 : ARRAY [0..1000] OF INTEGER;

PROCEDURE K1;
VAR i:INTEGER;
BEGIN
  i:=-1;
  LOOP
    i:=i+1;
    IF i>100 THEN
      TRANSFER(Ko1,Main);
    END;
    WriteInt(i,5); WriteLn;
    TRANSFER(Ko1,Ko2);
  END;
END K1;

PROCEDURE K2;
VAR i:INTEGER;
BEGIN
  i:=1001;
  LOOP
    i:=i-1;
    WriteInt(i,5); WriteLn;
    TRANSFER(Ko2,Ko1);
  END;
END K2;

BEGIN
  NEWPROCESS(K1,ADR(Zu1),SIZE(Zu1),Ko1);
  NEWPROCESS(K2,ADR(Zu2),SIZE(Zu2),Ko2);
  TRANSFER(Main,Ko1);
END CoDem.
```

Listing 2

bis zum END. weitergehen. Das Ziel des TRANSFERs ist in *Main* enthalten, so daß das jetzt vollständige Programm wie in Listing 2 aussieht. Führen Sie das Programm einmal aus und rekonstruieren Sie anhand der Bildschirmausgaben den Kontrollfluß!

Synchronisation

Nun ist diese direkte Programmierung mit TRANSFERs nicht sonderlich aufregend, man könnte den Kontrollfluß auch auf herkömmliche Weise steuern. Zudem hat ein solches TRANSFER einiges vom zurecht geschmähten GOTO, das es in Modula übrigens glücklicherweise nicht mehr gibt.

In der Praxis benutzt man die TRANSFERs, um eine Reihe von Funktionen zu implementieren, die ein bestimmtes Synchronisationsmodell beschreiben. Was ist das? Stellen Sie sich vor, daß verschiedene Co-Routinen auf einem Rechner quasi-parallel arbeiten und ihre jeweilige Rechenzeit nicht vorhersagbar ist. Solche Co-Routinen nennt man Prozesse. Dazu bräuchte man natürlich eine Art von Multitasking, bei dem z.B. ein Systemprogramm dafür sorgt, daß jeder angemeldete Prozeß eine bestimmte Zeit lang rechnen darf, dann "schlafengelegt" wird und ein anderer Prozeß drankommt. Dieses wäre ein "Zeitscheiben"-Verfahren. Es könnte auch möglich sein, daß jeder Prozeß auf einem eigenen Prozessor arbeitet (denken Sie an in OCCAM programmierte Transputer).
In einer solchen Situation kann es zu Problemen kommen, wenn zwei Prozesse das gleiche tun oder sich untereinander verständigen wollen. So könnten gleichzeitig zwei Prozesse auf denselben Block der Festplatte schreiben wollen, oder ein Sortierprozeß einem Druckprozeß mitteilen, daß z.B. ein Feld fertig sortiert ist. Damit keine Konflikte entstehen und die Prozesse wie gewünscht zusammenarbeiten, müssen sie sich synchronisieren. Bei korrekter Programmierung dürfen Sie dazu ausschließlich die angebotenen Synchronisationsfunktionen benutzen.

Diese Synchronisationsfunktionen sorgen dafür, daß der Kontrollfluß geregelt wird. Nehmen wir das Beispiel mit dem gleichzeitigen Versuch, die Festplatte zu beschreiben. In dem Synchronisationsmodell, das hier beschrieben wird, müssen beide Prozesse mit einem Funktionsaufruf bekanntgeben, daß sie jetzt diese Operation vornehmen wollen. Nennen wir die Funktion P, dann hieße der Aufruf jeweils *P(festplatte)*. Da die Prozesse korrekt programmiert sind, geben sie nach Ausführung des Plattenzugriffs bekannt, daß sie aus diesem Programmabschnitt heraus sind. Nennen wir diese Funktion V, so hätten wir das Statement *V(festplatte)*.

Die Routinen P und V enthalten nun die Steuerung zum Kontrollfluß per TRANSFER. Was muß geschehen? Bei P ist zu prüfen, ob vielleicht ein anderer Prozeß *P(festplatte)* aufgerufen hatte und noch nicht mit *V(festplatte)* die Beendigung der Arbeit bekanntgegeben hat. Ist dies der Fall, wird der Prozeß in einer Liste vermerkt und braucht auch nicht mehr bei dem oben genannten automatischen Zeitscheiben-Verfahren berücksichtigt zu werden. In dieser Liste finden sich also geordnet alle Prozesse, die auf die Freigabe warten. Ist die Plattenbenutzung unkritisch, wird lediglich vermerkt, daß ein Prozeß momentan an der Harddisk arbeitet.

Bei V gibt es wiederum zwei Möglichkeiten. Befinden sich in der angesprochenen Liste Prozesse, die bei einem P aufgelaufen sind, kann der erste in der Liste aus ihr entfernt und aktiviert werden. Dazu würde man in Modula selbstverständlich ein TRANSFER verwenden. Die anderen müssen weiterhin warten, und nach wie vor arbeitet ein Prozeß mit der Platte. Gibt es keine solchen Prozesse, ist die Arbeit mit der Festplatte unkritisch, und das kann vermerkt werden. Dann würde der nächste Prozeß, der ein P aufruft, sofort drankommen.

P und V sorgen also dafür, daß immer nur ein Prozeß in einem geschützten Abschnitt arbeiten kann. Sie gewährleisten auch, daß alle Prozesse, die einmal ein P aufgerufen haben, irgendwann bei einem V aktiviert werden. Für die Aktivierung sorgt die TRANSFER-Routine, die den Zustand der wartenden Prozesse einfriert. Informatiker wissen natürlich längst, wie dieses Synchronisationsmodell genannt wird: "Semaphore". Man geht dabei von einem "kritischen Abschnitt" aus, einem Programmteil, bei dem ein Prozeß nicht von anderen gestört werden darf. Der Name der P-Operation stammt von "Passieren" - der Prozeß will quasi die Schranke zum kritischen Abschnitt passieren. Entsprechend kommt V von "Verlassen" - der Prozeß verläßt den kritischen Abschnitt. Der Schöpfer dieses ganzen Modells ist einer der bekanntesten Informatiker, E.W. Dijkstra.

Sie ersehen aus dieser langen Beschreibung, daß die Implementierung der Funktionen nicht unbedingt einfach ist. Bei der Arbeit zu dieser Serie hat sich herausgestellt, daß sie den Umfang einer Folge überschreiten würde. Zudem haben einige Modula-Systeme ihre Besonderheiten in diesem Bereich, dieser Teil wird von Modula für den kommenden DIN-Entwurf überarbeitet; und schließlich bietet das GEMDOS leider so wenig Unterstützung für ein Zeitscheibenverfahren, daß ich auf eine vollständige Darstellung hier verzichten muß.

Es gibt weitere Synchronisationsmodelle, die bekanntesten sind die Monitore und Signale. Schauen Sie doch einmal in Ihrem Modula-System nach, ob vielleicht in Modulen wie *Process* oder *Signals* entsprechende Routinen zu finden sind.

Ausblick

Damit sind wir zusammen am Ende dieser Einführung in Modula-2 angelangt. In der nächsten, abschließenden Folge werden, wie angekündigt, noch einzelne weitere Aspekte und einige Leseranfragen geklärt.

Sie haben nun einen Überblick und vielleicht auch einen Einblick in die wichtigsten Features von Modula-2 gewonnen. Sicher sind viele kleinere Tricks und Programmiermöglichkeiten offengeblieben. Dennoch glaube ich, daß Sie nun - insbesondere nach Beschäftigung mit den Aufgaben - in der Lage sein müßten, auch etwas anspruchsvollere Projekte in Modula-2 zu verwirklichen.

Vielleicht haben Sie auch erkannt, daß die gesamte Serie einen Programmierstil andeutet, der sehr konservativ ist und auf implementationsnahe Tricks verzichtet. So habe ich beispielsweise bewußt nicht mit den verfügbaren Modulen zur Benutzung von GEMDOS gearbeitet und dafür den Standardmodulen den Vorzug gegeben. Vielleicht haben Sie gesehen, daß Modula-2 diesen Programmierstil durch die hohen Anforderungen - z.B. bezüglich der Typenstriktheit - fördert.

Zum Schluß des in sich geschlossenen Teils der Serie noch eine Bitte: Als Autor wäre ich für jede Zuschrift dankbar, die den Kurs, der Ihnen nun komplett vorliegt, bewertet. Welche Teile fanden Sie leicht, welche schwer verständlich? Wo würden Sie mehr Informationen und Beispiele erwarten? Was hat Ihnen gefehlt, welche Teile fanden Sie überflüssig? Zuschriften wie gewohnt an ST-Computer, z. Hd. R.Tolksdorf. Damit verabschiede ich Sie zum vorletzten Mal bis in vier Wochen, dann mit einem Frage-Antwort-System zur Klärung noch offener Fragen und einem besonderen Bonbon.

RT

Hausaufgaben

1. Was macht folgendes Statement, und welche Variablen müßten dazu wie deklariert sein?

```
...
Funktion^(10.0);
...
```

2. Schreiben Sie unter Benutzung des Moduls INTLists eine Prozedur *ProcessList*, die eine beliebige Prozedur auf alle Listenelemente anwendet. Ihr soll jeweils der Wert eines Listenelements als Parameter übergeben werden.

3. Es gibt auch ein Modell mit Semaphoren, bei dem mehrere Prozesse in einem kritischen Abschnitt arbeiten können. Beschreiben Sie, wie der geschilderte Ablauf von P und V dafür geändert werden müßte.

Lohn & Gehalt ST
Die Lohn- und Gehaltsabrechnung für mittlere Betriebe



Leistungsmerkmale

- Einfach zu bedienende, praxiserprobte Abrechnung
- Brutto-, Nettolohnabrechnung, Pauschalierung
- Unbegrenzte Mitarbeiteranzahl
- Keine Lohnarten- Einschränkung
- Individuelle Formulare
- Lohn- und Gehaltsabrechnung, Lohnjournal, Lohnkonto, Krankenkassenliste, Zahlungsaufträge etc.
- Aufteilung auf Kostenstellen möglich
- Durchschnittslohn
- Einmalzahlungen, auch unter Berücksichtigung der Märzklausel
- Berlinfähige Abrechnung
- Einstieg während des laufenden Geschäftsjahres möglich

DM COMPUTER
7530 Pforzheim • Kaiser-Friedrich-Straße 8 • Telefon 0 72 31/2 60 91

gdat
● Finanzbuchhaltung
► Test in diesem Heft
● Mirage
● APL
und viele neue Programme, Tools, Hardware, Literatur,...
Der neue Katalog ist da! KOSTENLOS ANFORDERN!
gdat Gesellschaft für dezentrale Daten–Technik mbH
Stapelbrede 39 • 4800 Bielefeld 1 • Telefon 0521/875 888
da vinci design

Numerische Mathematik

Teil 3
Integration und Differentiation

In dieser Folge befassen wir uns mit der Differential- und Integralrechnung. Dazu werden einige Begriffe erklärt und Beispielprogramme zu grundlegenden Algorithmen aufgeführt.

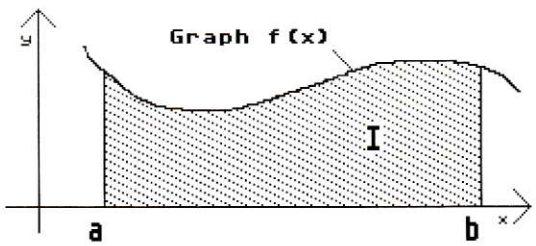
Was eine Funktion ist, wissen wir bereits aus der ersten Folge. Funktionen haben spezielle Eigenschaften. Stetige Funktionen haben einen Graphen, den man quasi mit einem Linienzug zeichnen kann, d.h. wenn man ihn zeichnen wollte, bräuchte man den Stift nicht abzusetzen. Ist eine Funktion stetig differenzierbar, verläuft der Graph so schön glatt, daß keine "Knicke" auftauchen. Wird eine Funktion *f* differenziert, erhält man die erste Ableitung dieser Funktion, genannt *f'*. Auch die Ableitung *f'* kann über die Eigenschaften Stetigkeit und Differenzierbarkeit verfügen. Ist *f'* differenzierbar, kommt man zur zweiten Ableitung *f''*. Sofern also Ableitungen nullten, ersten, zweiten, ... , n-ten Grads differenzierbar sind, können wieder Ableitungen einer höheren Stufe gebildet werden.

Es gibt Funktionen, die einmal, aber nicht zweimal stetig differenzierbar sind, auch welche, die nicht einmal stetig sind. Nehmen wir als Beispiel die Betragsfunktion $f(x)=|x|$. Die Funktion ist bei x=0 nicht differenzierbar, da der zugehörige Graph dort einen "Knick" hat.

Ist eine Funktion *f* stetig, hat sie eine Stammfunktion *F*, d. h. die Ableitung von *F* ist wieder *f*. Auch wenn die Existenz einer Stammfunktion bekannt ist, kann es unter Umständen sehr schwierig sein, diese zu bestimmen. Aber mit einer Stammfunktion *F* von *f* kann man eine Funktion *f* integrieren, d. h. den Wert des Integrals *I* von *f* über einem Intervall *[a;b]* angeben. Zudem gibt es auch noch nichtstetige Funktionen, die trotzdem integrierbar sind. Da für viele Probleme nur ein Näherungswert eines Integrals benötigt wird, greifen in diesen Fällen unsere Methoden der Numerischen Mathematik!

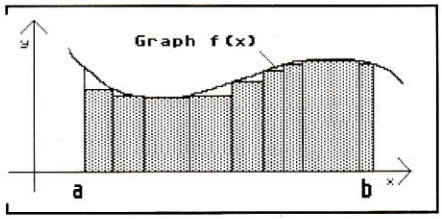
$$f'(x_0) = \lim_{x \to x_0} \frac{f(x)-f(x_0)}{x - x_0}$$

$$\int_a^b f(x)\,dx = F(b) - F(a) = I$$



Ableitungen und Integrale haben neben theoretischen auch anschauliche Bedeutungen. Ein Integral von *i*über einem Intervall gibt die Fläche an, die der Graph von *f* in diesem Intervall und die (x-)Koordinatenachse umspannen. Dabei werden Flächen unterhalb der Koordinatenachse negativ, die Flächen oberhalb positiv gezählt. Die erste Ableitung einer Funktion in einem Punkt x gibt die Steigung des Graphen der Funktion in diesem Punkt an, die zweite sagt etwas über das Krümmungsverhalten aus.

Der Integralbegriff, der hier zugrunde liegt, stammt von Bernhard Riemann (geb. 17. 9. 1826 in Breselenz, gest. 20. 7. 1866 in Salesca). Er wird in der Regel über Treppenfunktionen eingeführt. Hierzu werden die Koordinatenachse in kleine Teilintervalle unterteilt und dann die Flächen von passenden Rechtecken zwischen Graph und x-Achse berechnet

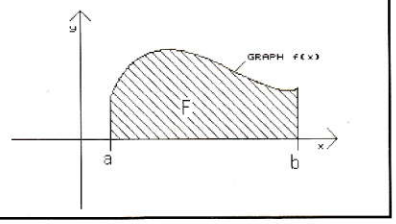


(siehe Bild 2). Näherungswerte für den korrekten Flächeninhalt erhält man damit durch geeignete Verkleinerung der Intervallängen zwischen den Intervallgrenzen. Aber wir wollen die Theorie ja nicht zu weit treiben, kommen wir zur Numerik.

In der letzten Folge haben wir Interpolationsformeln kennengelernt. Darunter war auch die Lagrangesche. Integriert man diese explizit, so erhält man einen Satz von sehr bekannten Näherungsformeln für den Integralwert. Die explizite Integration bereitet keine größeren Schwierigkeiten, da die Lagrangesche Interpolationsformel ein Polynom ist. Die entstehenden Formeln heißen Newton-Cotes-Formeln.

Problemstellung:

$$F = \int_a^b f(x)\,dx = ?$$



Die einzelnen Formeln, die sich nach der Anzahl der Stützstellen des Interpolationsproblems richten, haben wiederum Namen. So entsteht aus nur einer Stützstelle die bekannte Trapezregel, aus zwei Stützstellen die Simpson-Regel (nach Thomas Simpson, geb. 20. 8. 1710 in Market-Bosworth und gest. 14. 5. 1761 ebenda), aus drei Stützstellen die 3/8-Regel, aus vier Stützstellen die Milne-Regel und aus sechs Stützstellen die Weddle-Regel.

Simpsonsche Formel:

$$F = \frac{b-a}{3}\left(f(a) + 4\,f\left(\frac{a+b}{2}\right) + f(b)\right) + \mathit{Fehler}$$

Bei den Formeln von Newton-Cotes gelten die Intervallgrenzen immer als Stützstellen. Bei einer anderen Gruppe von Integrationsformeln wie z. B. den Formeln von Euler und Maclaurin ist dies nicht der Fall.

Als Beispiel zu den Newton-Cotes-Formeln wurde für das Listing 1 die Simpson-Regel benutzt. Allerdings wurde das Intervall weiter unterteilt, um eine höhere Genauigkeit zu erreichen. Damit haben wir die sogenannte zusammengesetzte Simpson-Regel.

Berechnet man nicht wie bei den Newton-Cotes-Formeln die Stützstellen und damit die Gewichte, sondern wählt Stützstellen und Gewichte so, daß eine maximale Genauigkeit erreicht wird, kommt man zu den Integrationsformeln von Gauß.

Gaußsche Formeln:

$$h = \frac{b-a}{2}$$

1 Stützstelle: $F \approx 2h\,f\left(\frac{a+b}{2}\right)$

2 Stützstellen: $F \approx h\left(f\left(\frac{a+b}{2} - \sqrt{3}^{-1}h\right) + f\left(\frac{a+b}{2} + \sqrt{3}^{-1}h\right)\right)$

3 Stützstellen: $F \approx h\left(\frac{5}{9}\left(f\left(\frac{a+b}{2} - \sqrt{\frac{2}{5}}h\right) + f\left(\frac{a+b}{2} + \sqrt{\frac{2}{5}}h\right)\right) + \frac{8}{9}f\left(\frac{a+b}{2}\right)\right)$

Als Stützstellen ergeben sich dann gerade die Nullstellen eines passenden Legendre-Polynoms. Auf diese Berechnung wollen wir hier jedoch nicht näher eingehen.

Die Integrationsformeln von Gauß sind also genauer als die Formeln von Newton-Cotes. Sie haben aber einen Nachteil, der gerade noch nicht klar wurde: Die Berechnung der Stützstellen ist aufwendiger. Außerdem erhält man nicht ganz so "schöne" Werte für die Koeffizienten, wie es z.B. bei den Newton-Cotes-Formeln der Fall ist.

Das zweite Beispielprogramm zeigt die Implementation dreier Gaußscher Integrationsformeln. An den Ergebnissen erkennt man jedoch, daß auch in unserem Beispiel die Formeln von Gauß genauer sind als die von Newton-Cotes.

Geht man den anderen Weg und differenziert das Interpolationspolynom von Lagrange, erhält man Formeln für die numerische Differentiation. Das Interpolationspolynom kann man theoretisch beliebig oft ableiten und erhält dann auch Formeln für höhere Ableitungen, die jedoch mit äußerster Vorsicht zu genießen sind, da sie meistens nicht genauer werden.

Man überlege sich dazu nur, daß man die Funktion *sin(x)* mit einem Interpolationspolynom lokal nähert. Beliebig hohe Ableitungen werden nie konstant gleich 0, aber wenn man das Interpolationspolynom ausreichend oft differenziert, wird es irgendwann konstant gleich 0.

Trotzdem erhält man zumindest für die erste und zweite Ableitung interessante Formeln. Sie heißen je nach Anzahl ihrer Stützstellen 3-, 4-, 5-Punkte-Formel, ... (Die Formeln finden sich in Listing 3.)

Das dritte Beispielprogramm wertet drei Formeln dieser Art aus. Im Vergleich mit dem im folgenden beschriebenen Romberg-Verfahren erhält man jedoch meistens schlechtere Werte. Das Romberg-Verfahren hat einen Vorteil, der hier vorweggenommen werden soll: Die Genauigkeit ist nicht nur höher, sie kann vorher bestimmt werden.

Ähnlich den Interpolationsverfahren von Newton und Neville aus der letzten Folge unserer Serie kann man auch beim Romberg-Verfahren ein Rechenschema angeben. Es ist in Bild 7 zu sehen. Der Index, der oben rechts am D steht, bezeichnet

Differentiation nach Romberg:

Startwerte mit $h_0 = h$ (Schrittweite):

$$D_j^{(0)}(f(x_0)) = \frac{f(x_0 + h_j) - f(x_0 - h_j)}{2h}$$

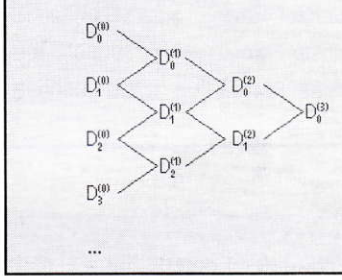
weitere Werte mit $h_{j+1} = h_j / 2$:

$$D_j^{(k)}(f(x_0)) = \frac{1}{2^{2k}-1}\left(2^{2k} D_{j+1}^{(k-1)}(f(x_0)) - D_j^{(k-1)}(f(x_0))\right)$$

den Schritt, der Index unten rechts, der wievielte Wert des passenden Schritts vorliegt. Ausgehend von der Anzahl der Werte pro Schritt wird die Schrittweite *hj* gewählt. Sie wird für jeden folgenden Wert eines Schritts halbiert.

Das Rechenverfahren wird solange fortgesetzt, bis die gewünschte Genauigkeit erreicht ist. Das ist genau dann der Fall,

Das Romberg-Schema:



wenn - wie in Listing 4 zu sehen ist - abs(d[1]-d[0]) unterhalb einer vorgegebenen Obergrenze liegt. Allerdings sollte man eine Maximalzahl an Schritten vorgeben, denn auch beim Romberg-Verfahren kann es passieren, daß Konvergenz nicht erreichbar ist.

Der einzige Nachteil des Verfahrens: Die Programmierung, der Speicherbedarf

und die Rechenzeiten sind größer als bei den einfachen n-Punkte-Formeln.

Nun sind wir schon am Ende der dritten Folge angelangt. In der nächsten wenden wir uns einem speziellen Integrationsproblem zu: der Berechnung einer Lösung einer gewöhnlichen Differentialgleichung erster Ordnung.

Auf bald...

Dipl.-Math. Dietmar Rabich

Literatur:

[1] Einführung in die Numerische Mathematik I, J. Stoer, Springer Berlin/ Heidelberg/ New York/ Tokyo, 4. Aufl. 1983, S. 106ff

[2] Formelsammlung zur Numerischen Mathematik mit BASIC-Programmen, G. Engeln-Müllges/F. Reutter, Bibliographisches Institut Mannheim/Wien/Zürich, 1. Aufl. 1983, S. 186ff

[3] Numerische Mathematik, H. R. Schwarz, Teubner Stuttgart, 1. Aufl. 1986, S. 319ff

[4] Numerische Methoden, Å. Björck/ G. Dahlquist, Oldenbourg München/ Wien, 2. Aufl. 1979, S. 213ff

[5] Methode der Numerischen Mathematik, W. Böhm/ G. Gose/ J. Kahmann, Vieweg Braunschweig/ Wiesbaden, 1. Aufl. 1985, S. 122ff

[6] Erfolgreich programmieren mit C, J. A. Illik, Sybex Düsseldorf/ San Francisco/ Paris/ London, 4. Aufl. 1987

[7] Programmieren in C, B. W. Kernighan/ D. M. Ritchie, Hanser München/Wien, 1. Aufl. 1983

[8] PASCAL für Anfänger, H. Schauer, Oldenbourg Wien/München, 4. Aufl. 1982

[10] PASCAL für Fortgeschrittene, H. Schauer, Oldenbourg Wien/ München, 2. Aufl. 1983

Ergebnisse der Programme:

Integration:

tatsächlicher Wert: 0.3333333333
ermittelt durch

a)	Simpson-Regel (5 Unterteilungen)	: 0.3666666102
b)	Simpson-Regel (10 Unterteilungen)	: 0.3499998976
c)	Simpson-Regel (15 Unterteilungen)	: 0.3444443892
d)	Simpson-Regel (20 Unterteilungen)	: 0.3416663722
e)	Simpson-Regel (25 Unterteilungen)	: 0.3399997366
f)	Simpson-Regel (30 Unterteilungen)	: 0.3388886482
g)	Simpson-Regel (35 Unterteilungen)	: 0.3380951832
h)	Simpson-Regel (40 Unterteilungen)	: 0.3374995357
i)	Simpson-Regel (45 Unterteilung	: 0.3370369821
j)	Simpson-Regel (50 Unterteilungen)	: 0.3366662074
k)	Gauß-Integration (1 Stützstelle)	: 2.5000000000E-01
l)	Gauß-Integration (2 Stützstellen)	: 3.3333333310E-01
m)	Gauß-Integration (3 STützstellen)	: 3.3333333333E-01

Differentiation:

tatsächlicher Wert: 1.75
ermittelt durch

a)	Romberg-Verfahren	: 1.7500000000E+00
b)	3-Punkte-Formel	: 2.7500000000E+00
c)	5-Punkte-Formel	: 1.7500000000E+00
d)	7-Punkte-Formel	: 1.7500000000E+00

```
 1: /***********************************************/
 2: /* Beispielprogramm zur Integration mit der    */
 3: /* summierten Simpsonregel                     */
 4: /* ------------------------------------------- */
 5: /* Entwickelt mit Turbo C.   15.02.1989        */
 6: /***********************************************/
 7:
 8: /* --------- + ---------------- */
 9: /* Listing 1 /  by D. Rabich    */
10: /*   (c) MAXON Computer GmbH    */
11:
12: /* Ein-/Ausgabe-Routinen importieren */
13: # include <stdio.h>
14:
15: /* Grenze zur Berücksichtigung eventueller
       Ungenauigkeiten */
```

```
16:  # define FLOATFEHLER 1.0E-6
17:
18:  /* Funktion f */
19:  float f(float *x)
20:
21:   {
22:    return((*x)*(*x));
23:   }
24:
25:  /* Berechnet Integral der Funktion f zwischen a
        und b */
26:  float simpson(float (*f)(), float a, float b,
                    int n)
27:
28:   {
29:    float x,
30:          integ,
31:          h;
32:
33:    integ=0;
34:
35:    /* Schrittweite */
36:    h=(b-a)/n;
37:
38:    /* Integral berechnen */
39:    for(x=a+h/2;x<b-h/2+FLOATFEHLER;x+=h)
40:     integ+=(*f)(&x);
41:
42:    integ*=2;
43:
44:    for(x=a;x<=b+FLOATFEHLER;x+=h)
45:     integ+=(*f)(&x);
46:
47:    return(integ*h/3);
48:   }
49:
50:  /* Hauptprogramm */
51:  void main(void)
52:
53:   {
54:    int n;
55:
56:    /* Ausgabe bei verschiedenen Unterteilungen */
57:    for(n=5;n<=50;n+=5)
58:     printf("Integral bei %2d Unterteilungen:
                %12.10g\n",
59:            n,simpson(f,0,1,n));
60:
61:    /* Auf Taste warten... */
62:    getchar();
63:   }
```

```
 1: (***********************************************)
 2: (* Beispielprogramm zur GauP-Integration.      *)
 3: (* ------------------------------------------- *)
 4: (* Entwickelt mit ST Pascal Plus. 14.02.1989   *)
 5: (***********************************************)
 6:
 7: (* --------- + ---------------- *)
 8: (* Listing 2 / by D. Rabich     *)
 9: (*  (c) MAXON Computer GmbH     *)
10:
11:  program gauss_intgration;                    →
```

```
12:
13:  (* Typ *)
14:  type poss_n = 0..2;
15:
16:  (* Variablen *)
17:  var i : poss_n;
18:
19:  (* Funktion f *)
20:  function f (x : real) : real;
21:
22:   begin
23:    f:=x*x
24:   end;
25:
26:  (* Berechnet Integral von Funktion f zwischen a
       und b *)
27:  function gauss (function f (x : real) : real;
28:                                  a,b : real;
29:                                    n : poss_n)
                                        : real;
30:
31:   var fakt,h : real;
32:
33:   begin
34:    h:=(b-a)/2;
35:    case n of
36:
37:     (* Formel für eine Stützstelle *)
38:     0 : gauss:=2*h*f(a+h);
39:
40:     (* Formel für zwei Stützstellen *)
41:     1 : begin
42:           fakt :=1.0/sqrt(3.0);
43:           gauss:=h*(f(a+h*(1.0-fakt))+
                    f(a+h*(1.0+fakt)))
44:         end;
45:
46:     (* Formel für drei Stützstellen *)
47:     2 : begin
48:           fakt :=sqrt(0.6);
49:           gauss:=h*(5.0*(f(a+h*(1.0-fakt))
                      +f(a+h*(1.0+fakt)))
50:                     +8.0*f(a+h))/9.0
51:         end
52:
53:    end
54:   end;
55:
56:  (* Hauptprogramm *)
57:  begin
58:
59:   for i:=0 to 2 do
60:    writeln(gauss(f,0.0,1.0,i));
61:
62:   repeat
63:   until keypress
64:
65:  end.
```

```
 1: (*********************************************)
 2: (* Beispielprogramm zur numerischen
       Differentiation mittels                   *)
 3: (* Differentiation des Interpolationspolynoms.  *)
 4: (* ----------------------------------------- *)
 5: (* Entwickelt mit ST Pascal Plus.    14.02.1989 *)
 6: (*********************************************)
 7:
 8: (* --------- + ---------------- *)
 9: (* Listing 3 /  by D. Rabich    *)
10: (*    (c) MAXON Computer GmbH    *)
11:
12:  program intpol_differentiation;
13:
14:  (* Typ *)
15:  type poss_n      = (drei,fuenf,sieben);
16:
17:  (* Variable *)
18:  var i : poss_n;
19:
20:  (* Funktion f *)
21:  function f (x : real) : real;
22:
23:   begin
```

```
24:     f:=x*x+x*x*x
25:    end;
26:
27:   (* Berechnet Ableitung der Funktion f an der
        Stelle x *)
28:   function diff_intpol
                 (function f (x : real) : real;
29:                                   x,h : real;
30:                                     n : poss_n)
                                          : real;
31:
32:    begin
33:
34:     case n of
35:
36:      (* 3 Stützstellen *)
37:      drei   : diff_intpol:=(-   f(x-  h)
38:                             +   f(x+  h)) /(2*h);
39:
40:      (* 5 Stuetzstellen *)
41:      fuenf  : diff_intpol:=(    f(x-2*h)
42:                             - 8*f(x-  h)
43:                             + 8*f(x+  h)
44:                             -   f(x+2*h)) /(12*h);
45:
46:      (* 7 Stützstellen *)
47:      sieben : diff_intpol:=(-   f(x-3*h)
48:                             + 9*f(x-2*h)
49:                             -45*f(x-  h)
50:                             +45*f(x+  h)
51:                             - 9*f(x+2*h)
52:                             +   f(x+3*h)) /(60*h)
53:
54:     end
55:
56:    end;
57:
58:   (* Hauptprogramm *)
59:   begin
60:
61:    for i:=drei to sieben do
62:     writeln(diff_intpol(f,0.5,1,i));
63:
64:    repeat
65:    until keypress
66:
67:   end.
```

```
 1: (*********************************************)
 2: (* Beispielprogramm zur numerischen
       Differentiation nach dem                  *)
 3: (* Verfahren von Romberg.                     *)
 4: (* ----------------------------------------- *)
 5: (* Entwickelt mit ST Pascal Plus.    14.02.1989 *)
 6: (*********************************************)
 7:
 8: (* --------- + ------------ *)
 9: (* Listing 4 / by D. Rabich *)
10: (*  (c) MAXON Computer GmbH *)
11:
12:  program romberg_differentiation;
13:
14:  (* Konstante *)
15:  const real_fehler = 1.0E-8;
16:        max_ber     = 50;
17:
18:  (* Funktion f *)
19:  function f (x : real) : real;
20:
21:   begin
22:    f:=x*x+x*x*x
23:   end;
24:
25:  (* Berechnet Ableitung der Funktion f
       an der Stelle x *)
26:  function romberg (function f (x : real) : real;
27:                                   x,h : real)
                                          : real;
28:
29:   var d     : array [0..max_ber] of real;
30:       i,j,p : short_integer;
31:
32:   begin                                    →
```

```
33:
34:     (* Romberg-Verfahren *)
35:     d[0]:=(f(x+h)-f(x-h))/(2*h);
36:     i:=1;
37:     repeat
38:      h:=h/2;
39:      d[i]:=(f(x+h)-f(x-h))/(2*h);
40:      p:=4;
41:      for j:=i-1 downto 0 do
42:       begin
43:        d[j]:=(p*d[j+1]-d[j])/(p-1);
44:        p:=p*4
45:       end;
46:      i:=i+1
47:     until (abs(d[1]-d[0])<real_fehler)
              or (i>max_ber);
48:
```

```
49:     (* Fehlermeldung, falls Genauigkeit nicht
           erreicht *)
50:     if abs(d[1]-d[0])>=real_fehler then
51:      writeln('*** Genauigkeit nicht erreicht ***');
52:
53:     romberg:=d[0]
54:    end;
55:
56:   (* Hauptprogramm *)
57:   begin
58:
59:    writeln(romberg(f,0.5,1));
60:
61:    repeat
62:    until keypress
63:
64:   end.
```


DIGITAL DATA DEICKE, NORDRING 9, 3000 HANNOVER 1, TEL: 0511/6370 -54, -55

Wir verbessern das Gute und erhalten ...

Die beste Festplatte zum ATARI

(Das ist unsere Meinung. Aber entscheiden Sie selbst)

- Original ATARI ACSI-Controller, deshalb
- voll soft- und hardwaremäßig kompatibel, autobootfähig
- natürlich lauffähig mit PC-Ditto, PC-Speed, Aladin usw.
- Original Megafile Gehäuse
- neue schnelle SEAGATE Laufwerke (s.u.)
- Lüfter unhörbar
- bis zu 14 Partitionen; DMA In und Out, gepuffert
- komplett, voll eingerichtet und betriebsfertig geliefert
- trotzdem unglaublich preiswert:

INFO ANFORDERN

ddd-HD 32 DM 924,- mit Park-Accessorie 65 ms

ddd-HD 48 DM 1294,- mit Hardware-Autopark 28 ... 40 ms

ddd-HD 64 DM 1444,- mit Hardware-Autopark 28 ms

Versand per NN, europaweit und Direktverkauf in Hannover

(Demnächst auch größere Festplatten lieferbar, dann allerdings lediglich mit SCSI-Controller: z.B. 85 MB für DM 1594,-)
(Wechselplatte für ATARI siehe unten)

INFO ANFORDERN

ddd-LASER/8: DM 2498,-

100% kompatibel zum ATARI Laser, CANON-Druckwerk, Standby Schalter

Aber auch sonst ...

NEC P6+	DM 1344,-	NEC P7+	DM 1694,-
STAR LC 24-10	DM 694,-	EPSON LQ 400	DM 694,-
SPAT Scanner	DM 888,-	Profi-Scanner	DM 2777,-
PC-Speed	DM 555,-	Einbau	DM 30,-

Wechselplatte WP 44MB für ATARI ST DM 1998,-
Medium dazu DM 250,- INFOS ANFORDERN !!!

Komplettsystem zum erstellen einer Anzeige wie dieser (Rechner, Monitor, Laserdrucker, Software) bei uns schon für unter DM 5500,-. Fragen Sie doch mal nach !

Öffnungszeiten: MO. - FR. von 10 - 18 Uhr durchgehend
aber auch außerhalb dieser Zeiten immer für Sie da !

Rufen sie doch mal an

Lovely Helper

Ein Desk-Accessory 7



Der "Taschenrechner"

Heute werde ich mit der Programmierung des Rechnerbestandteils unseres Lovely Helpers beginnen. Leider auch wieder ein recht umfangreiches Unterfangen. Aus organisatorischen Gründen wird deshalb in dieser Folge zwar das komplette Listing abgedruckt, aber nur etwa bis zur Hälfte kommentiert. Der Rest folgt dann beim nächsten Mal. Heute werden wir uns mit dem Resource und seiner Verwaltung beschäftigen. Da die Wechselwirkung zwischen Resource und Programm dabei etwas umfangreicher sein wird, reicht das als Programmpunkt für heute vollständig aus.

Resource

Einziger Bestandteil des Resources ist heute der Dialog *RECHNER*. Er ist hoffentlich in ausreichender Weise einem Taschenrechner nachempfunden (s. Abb. 24). Die Eingabe erfolgt über einen Satz von insgesamt 35 Feldtasten. Die Ausgabe - einschließlich diverser Statusanzeigen - erfolgt über insgesamt 10 Texte. Da die Feldtasten heute alle gleichartig sind - Flags *Selectable & Exit* - werde ich sie nicht, wie bisher, einzeln aufführen. Es sei auf die Abbildung verwiesen. Die Definitionen und die Semantik der verbleibenden Texte kann man aus Tabelle 1 entnehmen.

Aufgabenteilung

Bevor es an die Implementierung des Taschenrechners geht, möchte ich zunächst eine Aufgabenteilung vornehmen und ein wenig auf die zu erwartenden Probleme hinweisen. Dazu ist zu sagen, daß die Programmierung eines normalen Taschenrechners - "normal" hier im Sinne von Punkt-vor-Strich-Rechnung, Klammerung etc. - eine ganze Reihe von kleineren Tricks erforderlich macht.

Wie wir in der nächsten Folge sehen werden, kommen wir dabei nicht um die Benutzung von Stacks zur Berücksichtigung der Prioritäten (Klammern, Punkt-vor-Strich) herum. Eine weitere Problematik liegt darin begründet, daß beim Betrieb eines (guten) Taschenrechners eine ganze Reihe von Konventionen einzuhalten ist. Als Beispiel möchte ich die kurze Kommandosequenz *2 + =* nennen. Von ihr würde man etwa erwarten, daß *2 + 2 =* berechnet wird, also der zweite (nicht eingegebene) Operand aus dem Kontext gefolgert wird. Auch dies sollten wir bei der Konstruktion unseres Rechners berücksichtigen.

Wegen der Vielschichtigkeit dieser Probleme werden wir die Verarbeitung der von der Tastatur (Dialog) kommenden Befehle aufteilen. Und zwar in Editbefehle - sie beziehen sich ausschließlich auf das Edieren des momentan aktuellen Wertes - und Operatorbefehle - sie dienen der Eingabe von Operationen unter Einhaltung der Prioritäten.

Zum reinen Edieren dienen dabei die zehn Ziffern *(BNULL - BNEUN)*, der Dezimalpunkt *(BKOMMA)*, die Taste zum Vorzeichenwechsel *(BVZW)*, die Exponentialtaste *(BEXP)*, die Löschtaste *(BC)*, die Taste zur Wahl des Berechnungsmodus' der trigonometrischen Funktionen *(BMOD)* sowie die Taste zur Wahl der inversen Funktionen *(BINV)*.

Sämtliche übrigen Tasten - die Funktionstasten, die Klammern und die Grundrechenarten - sind Operatorbefehle. Demzufolge läßt sich die Arbeit des Taschenrechners ebenfalls unterteilen in einerseits das Edieren von Zahlen mit abschließender Operatoreingabe und andererseits in die Abarbeitung von Sequenzen derartiger Wert-/Operatorpaare. An der Schnittstelle, zwischen diesen beiden Teilen müssen sowohl Werte als auch Operatorsymbole übergeben werden.

Beispiel: Die Berechnung von *(1 + 2) * 3 =* würde in folgenden Portionen erfolgen:

Nr.	Wert	Operator
1.	_	(
2.	1	+
3.	2	)
4.	_	*
5.	3	=

Das Zeichen '_' steht dabei für einen leeren Wert. Als zusätzlicher Parameter ist also auch noch ein Wahrheitswert zu übergeben, der die Gültigkeit des Operators angibt. Wenn wir uns dem Listing 14 zuwenden, lassen sich diese Parameter in der Prozedur *do_edit* (Zeilen 131-499) wiederfinden. Wert und Operator werden dabei direkt über die Parameterliste übergeben, der Wahrheitswert über die Variable *gueltig* (ein Seiteneffekt von *do_edit*).

Das Edieren

Mit dem Inneren von *do_edit* sowie den Variablen und Operationen, auf die sich besagte Prozedur abstützt, wollen wir uns nun beschäftigen. Die ersten Deklarationen von Interesse, sind damit heute die Konstanten des zweiten Konstantenblocks (Zeilen 14-19).

Die Bedeutung von pi dürfte wohl klar sein. Die nächsten drei Konstanten stehen für die drei Maßeinheiten, mit denen in

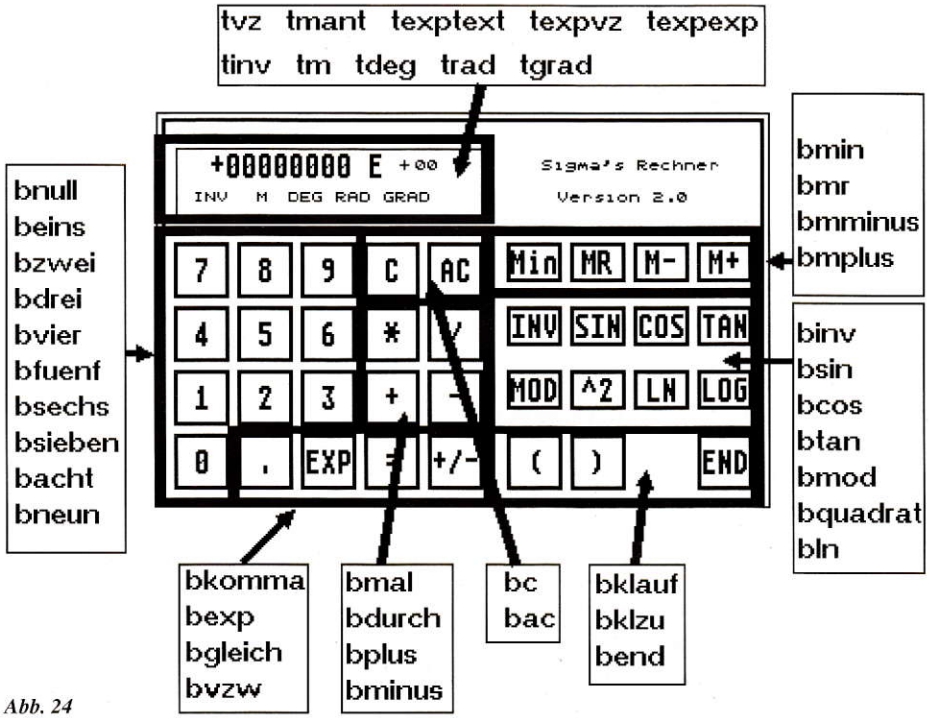


Abb. 24

Objekt	Objekttyp	Länge	Semantik
TVZ	TEXT	1	Vorzeichen der Mantisse
TMANT	TEXT	8	Mantisse
TEXPTEXT	TEXT	1	Anzeige des Exponentialmodus
TEXPVZ	TEXT	1	Vorzeichen des Exponenten
TEXPEXP	TEXT	2	Exponent
TINV	TEXT	3	Statusanzeige Invers
TM	TEXT	1	Speicher enthält Wert
TDEG	TEXT	3	Winkelfunktionen in Grad
TRAD	TEXT	3	Winkelfunktionen in Bogenmaß
TGRAD	TEXT	4	Winkelfunktionen in Altgrad (in USA üblich)

Tabelle 1: Die Object-Daten für den Dialog RECHNER (Abb. 24)

den trigonometrischen Funktionen gerechnet werden kann (deg, rad, grad). Die beiden letzten Konstanten kennzeichnen den Status der Inverstaste (an, aus).

Der einzige Typ von Interesse ist heute *op_type*. Er beschreibt das Vokabular, das von der abstrakten Maschine, der nächsten Folge, verstanden werden muß, um die Wert-/Operatorpaare interpretieren zu können. An der Schnittstelle von *do_edit* haben wir deshalb für die Übergabe der Operatoren in dieser Form zu sorgen.

Bei den Variablen interessieren uns zunächst die Statusvariablen des Taschenrechners (Zeilen 43-46). Sie besitzen folgende Bedeutung:

error kennzeichnet den Fehlerstatus. Aus der Tabelle 2 können Sie entnehmen, welche möglichen Belegungen für *error* existieren.

inv_modus und *tri_modus* nehmen den jeweiligen Status der Invers- und der Modustaste auf. In Abhängigkeit vom *tri_modus* schwankt auch der Wert von *t_faktor*. Diese Variable wird nämlich benutzt, um die trigonometrischen Funktionen allgemeingültig zu formulieren.

Die nächste Rechnerstatusvariable ist *speicher*. In ihr wird die eine (!) Speicherstelle unseres Taschenrechners aufbewahrt. Dabei wird die Konvention getroffen, daß der Speicher als leer gilt, wenn sein Wert Null beträgt.

first_rechner ist eine Variable, die angibt, ob unser Rechnerdialog neu auf dem Bildschirm ist oder sich noch hier befindet. In Abhängigkeit hiervon wird nämlich entschieden, ob der Dialog nur wieder ausgeführt werden muß (*redo_dialog*) oder noch komplett zu zeichnen ist (*do_dialog*).

gueltig ist der bereits besprochene Wahrheitswert, der angibt, ob eine edierte Zahl auch in die Berechnung eingeht.

Der nächste Variablenblock (Zeilen 51-60) ist schnell erklärt. Er beinhaltet die Strings, die später die Texte für das Ausgabefeld aufnehmen. Die Zuordnung dürfte mnemotechnisch klar sein.

Kommen wir zum Operationendeklarationsteil von *do_rechner*. Die erste hier angesiedelte Prozedur, *setze_texte* (Zeilen 62-82), dient der bereits hinlänglich behandelten Aufgabe, einen Dialog zu initialisieren. Dazu werden die gerade besprochenen Stringvariablen des letzten Variablenblocks in unseren heutigen Dialog eingetragen.

Die nächste Prozedur, *setze_redraw* (Zeilen 84-97), hat nun dafür zu sorgen, daß die redraw-Bits sämtlicher Textfelder gesetzt werden. Damit wird GEM bei der wiederholten Ausführung des Rechnerdialogs (*redo_dialog*) gezwungen, die Textfelder neu zu zeichnen. Leider geschieht diese Neuzeichnung **nicht** automatisch, wenn nur der Textstring geändert wird. Es **muß** - wie im Listing - manuell nachgeholfen werden.

Die letzte, kleinere Prozedur vor *do_edit* ist *do_error* (Zeilen 99-129). Sie hat die Aufgabe, gemäß dem Fehlerstatus die entsprechende Fehlerkennung in die Dialogtexte einzutragen und den Dialog einmal auszuführen, um für die Fehlerausgabe anzuhalten. Dabei wird die Bedeutung von *first_rechner* ersichtlich (siehe auch oben). Es dient dazu, daß der Rechnerdialog nur beim ersten Mal komplett gezeichnet wird, bei weiteren Aufrufen erfolgt lediglich eine Neuzeichnung der veränderten Dialogeinträge. Der Ratschlag des ST Pascal-Handbuches, die redraw-Anweisung bei *obj_setstate* betreffend, ist sehr wörtlich zu nehmen! Das redraw-Bit darf nur gesetzt werden, wenn der Dialog schon auf dem Bildschirm vorhanden ist. Wird es bei nichtvorhandenem Dialog gesetzt, hat das zur Folge, daß bei jedem *redo_dialog* trotz gegensätzlicher Angaben der komplette Dialog neu gezeichnet wird.

Kommen wir nun zu unserer eigentlichen heutigen Aufgabe, der Prozedur *do_edit*. Über die Bedeutung ihrer Parameter haben wir ja bereits gesprochen. Nun geht es ans Eingemachte.

do_edit hat vier Konstanten, die angeben, welchen Status die im Moment edierte Zahl besitzt. Zur Auswahl stehen: *fertig*, *vorkomma*, *nachkomma* und *exponent*. Die lokale Variable *status* nimmt genau einen dieser Werte auf.

Zwei Kernstücke von *do_edit* möchte ich Ihnen etwas detaillierter vorstellen. Es

handelt sich dabei um die beiden Operationen *setze_zahl* (Zeilen 142-298) und *get_zahl* (Zeilen 300-349).

Die erste dieser beiden Operationen - *setze_zahl* - initialisiert die Ausgabetexte, entsprechend der edierten Zahl, also eine Umwandlung von real in string. Da wir dabei ein ziemlich "verteiltes" Stringformat haben, scheiden leider die eingebauten ST Pascal-Operationen für diese Aufgabe aus. Also liegt die komplette Umwandlung in unserer Hand. *setze_zahl* unterscheidet dazu zunächst zwischen zwei Formaten: ganzzahlig und exponential.

Die ganzzahlige Darstellung wird dabei automatisch für ganze Zahlen unterhalb einer Million gewählt. Alle übrigen Zahlen werden exponential dargestellt. Unabhängig vom Format kann jedoch das Vorzeichen bestimmt werden (Zeilen 191-195). Die ganzzahlige Darstellung erfolgt, sind die notwendigen Bedingungen erfüllt (Zeile 196-197), in den Zeilen 199-214. Für die exponentiale Darstellung wird dagegen die Prozedur *exp_darstellung* (Zeilen 147-188) benutzt. Sie trennt Mantisse und Exponent und formt beide separat in Strings um. Die Details dieser beiden Transformationen seien dabei dem interessierten Leser überlassen.

Wir wollen nun noch die korrespondierende Funktion - *get_zahl*, die Umwandlung von string nach real - betrachten. Sie besitzt einen einfacheren Aufbau als *setze_zahl*. So erkennt man beispielsweise leicht, daß die Zeilen 315-321 den Vorkommateil der Zahl berechnen, die Zeilen 322-330 den Nachkommateil.
In 332 und 333 wird dann noch das Vorzeichen zugegeben.

Auch der Exponent berechnet sich ähnlich. Man muß hier allerdings darauf achten, daß kein Überlauf stattfindet, denn der Typ real nimmt unter Pascal+ maximal Werte bis zu 1e38 auf. Die Bedingung für die Abschätzung eines Überlaufs ist recht interessant (Zeile 344). Sie wird in ähnlicher Form sehr häufig beim nächsten Mal benutzt werden.

Die weiteren lokalen Operationen *do_edits* sind weniger aufwendig und schnell erklärt:

rsc_char berechnet mit einem CASE die einem Resourceangelpunkt zugeordnete Ziffer in Form eines Charakters.

get_operator hingegen wandelt die Angelpunkte in die schon besprochenen Operatorsymbole um.

Kommen wir nun zum Anweisungsteil von *do_edit* (Zeilen 351-499). Hier erfolgen zuerst einige Initialisierungen. Allen voran die des Gültigkeitbits für die Wertübergabe (Zeile 352). Es ist auf false zu setzen, denn ediert haben wir ja noch nichts. Der *status* bekommt als Initialwert "fertig" (Zeile 353). Erst wenn eine Eingabe getätigt wurde, kann in eines der drei anderen Stadien übergegangen werden. Als letzter Schritt der Initialisierung wird die übergebene Zahl in die Textstrings eingetragen (Zeile 354). Der Rechner tritt daraufhin in eine REPEAT-Schleife (Zeilen 355-496) ein, die nur durch Selektion eines Operatorsymbols wieder verlassen werden kann. In dieser Schleife erfolgt zunächst eine Eingabe mittels des Dialoges. (Dabei ist wieder das *first_rechner*-Bit zu beachten.)

Anschließend erfolgt die Auswertung mit einem sehr umfangreichen CASE-Statement. Im Verlauf dieses CASE-Statements sieht man, wie der Rechner auf die unterschiedlichen Eingaben reagiert:

1. Selektion von *BMOD*
(Zeilen 371-390):

Der *tri_modus* wird um eine Position weitergeshiftet. In Abhängigkeit von diesem Ergebnis werden die entsprechenden Texte neu gesetzt sowie der *t_faktor* neu berechnet.

2. Selektion von *BINV*
(Zeilen 391-397):

Der *inv_modus* wird neu gesetzt und der entsprechende Text umgesetzt.

3. Selektion einer Ziffer
(Zeilen 398-438):

Bei der Selektion einer Ziffer gibt es, entsprechend den vier Stadien des Editvorganges, vier Möglichkeiten der Reaktion.

War der vormalige Status fertig, werden nun sämtliche Ausgabestrings initialisiert, und dem ersten Wert der Mantisse wird der Wert der ausgewählten Feldtaste zugewiesen. Der Status wechselt dabei nach *vorkomma*.

War der Status *vorkomma*, ist zunächst zu überprüfen, ob noch Platz für weitere Eingaben vorhanden ist. Wenn ja, wird die eingegebene Ziffer von rechts mit der Mantisse verschmolzen.

Beim *nachkomma*-Status ist auch als erstes zu überprüfen, ob noch genügend Raum vorhanden ist. Wenn ja, wird die Ziffer ebenfalls an den Wert angehängt.

Im Status *exponent* ist lediglich die Fallunterscheidung zu treffen, ob der Exponent bisher leer war. Ist dies der Fall, wird die erste Ziffer des Exponential-Strings neu belegt, andernfalls die zweite.

4. Selektion von *BKOMMA*
(Zeilen 439-455):

BKOMMA hat nur im *vorkomma*- und im *fertig*-Modus eine Wirkung. Im *vorkomma*-Modus wird, falls Platz vorhanden ist, bewirkt, daß ein Punkt an die Mantisse angehängt wird. Im *fertig*-Modus wird die Mantisse mit dem Wert Null gefüllt und sämtliche anderen Strings initialisiert. Bei beiden Modi ist es jedoch erforderlich, mit dem Status nach *nachkomma* zu wechseln.

Nr.	Fehlerart
0	kein Fehler
1	Rechnerüberlauf
2	Division durch 0
3	Definitionsbereich verletzt
4	Klammerfehler

Tabelle 2: Statusmöglichkeiten für die Variable error

5. Selektion von *BVZW*
(Zeilen 456-470):

Ein Vorzeichenwechsel muß, je nach Status, für den Exponenten (*exponent*), oder für die Mantisse (*fertig, vorkomma, nachkomma*) vorgenommen werden.

6. Selektion von *BEXP* (Zeilen 471-482):

Allen Modi ist bei der Betätigung von *BEXP* gleich, daß in den *exponent*-Modus übergewechselt wird und die entsprechenden Strings initialisiert werden. Beim Status fertig wird zusätzlich davon ausgegangen, daß eine Mantisse von Eins erwünscht ist, da die Mantisse Null keinen Sinn macht - wieder eine der Konventionen zur Verbesserung der Bedienbarkeit.

7. Selektion von *BC* (Zeilen 483-491):

Bei der Selektion von *BC* werden sämtliche Ausgabetexte zurückgesetzt auf den Wert Null.

8. Selektion einer anderen Taste:

In diesem Fall wird in den *fertig*-Modus gewechselt.

Sind alle diese Fallunterscheidungen abgearbeitet, so ist im Falle fertig die Schleife zu verlassen. Einzige Ausnahme

SCANNER SPAT

982.-

VOLLSERVICE

MONITOR

14 ZOLL
3 Auflösungen

MAGIC SCREEN

MIT TON

492.-

H & T

HARDWARE & TECHNIK
RAINER MERTENS HANSJOACHIM ZEGEL
1000 BERLIN 10 MIERENDORFFPLATZ 6
TEL: 030-345 30 61 FAX: 030-345 81 72

VIDEOTEXT BOX

SOFTWARE
Speichern als Degas
Speichern als ASCII
Speichern Parameter
Textspeicher sichern
TOP Decoder
Seitenmemory
Mono in Graustufen
Colordarstellung

DRUCKER
Nec P6 P7
Panasonic
Epson
HP Desk-jet
HP Laserjet II

249.-

MONITOR

MIT TON

349.-

MAGIC SCREEN

12 ZOLL
3 Auflösungen

MONITOR
MONITOR
MONITOR
MONITOR
MONITOR
MONITOR
MONITOR
MONITOR

bilden dabei nur die vier Feldtasten: *BC*, *BMOD*, *BINV* und *BVZW*, bei denen in der Schleife verblieben wird. Hier liegt lediglich eine Statusänderung vor, die erst **nach Eingabe eines Kommandos** berücksichtigt werden muß.

Außerhalb der zentralen REPEAT-Schleife ist nur noch darauf zu achten, daß der Operator und der Operand mit den beiden dafür zuständigen Funktionen besorgt werden. Die Variable *gueltig* wurde bereits innerhalb der Schleife laufend korrigiert. Glücklicherweise haben wir es damit endlich geschafft. Unsere Schnittstelle ist mit den entsprechenden Parametern versorgt worden und wir können uns auf die nächste Folge vertagen.

Vorausschau

Abgedruckt finden Sie allerdings auch noch den Rest von *do_rechner* (Zeilen 501-893) und die Resource-Umgebung des Rechners (Listing 15). Wie bereits eingangs angekündigt, werden diese Bestandteile jedoch erst beim nächsten Mal näher erläutert. Es geht dabei im wesentlichen um die Abarbeitung der heute erhaltenen Wert-/Operatorsequenzen. Weiterer Programmpunkt der nächsten und letzten (!) Folge des Lovely Helpers ist das (lange ersehnte) Zusammenbinden der bisher erhaltenen einzelnen Bestandteile zu dem kompletten Accessory. Ich hoffe, Ihre Geduld reicht noch für einen Monat aus. Bis dahin!

D.Brockhaus

```
(* resource set indicies for RECHNER *)

CONST
  rechner  = 0;   (* form/dialog *)
  tmant    = 2;   (* TEXT in tree RECHNER *)
  texpexp  = 3;   (* TEXT in tree RECHNER *)
  tvz      = 4;   (* TEXT in tree RECHNER *)
  texpvz   = 5;   (* TEXT in tree RECHNER *)
  texptext = 6;   (* TEXT in tree RECHNER *)
  tinv     = 7;   (* TEXT in tree RECHNER *)
  tm       = 8;   (* TEXT in tree RECHNER *)
  tdeg     = 9;   (* TEXT in tree RECHNER *)
  trad     = 10;  (* TEXT in tree RECHNER *)
  tgrad    = 11;  (* TEXT in tree RECHNER *)
  bsieben  = 14;  (* BUTTON in tree RECHNER *)
  bacht    = 15;  (* BUTTON in tree RECHNER *)
  bneun    = 16;  (* BUTTON in tree RECHNER *)
  bc       = 17;  (* BUTTON in tree RECHNER *)
  bac      = 18;  (* BUTTON in tree RECHNER *)
  bmin     = 19;  (* BUTTON in tree RECHNER *)
  bmr      = 20;  (* BUTTON in tree RECHNER *)
  bmminus  = 21;  (* BUTTON in tree RECHNER *)
  bmplus   = 22;  (* BUTTON in tree RECHNER *)
  bvier    = 23;  (* BUTTON in tree RECHNER *)
  bfuenf   = 24;  (* BUTTON in tree RECHNER *)
  bsechs   = 25;  (* BUTTON in tree RECHNER *)
  bmal     = 26;  (* BUTTON in tree RECHNER *)
  bdurch   = 27;  (* BUTTON in tree RECHNER *)
  binv     = 28;  (* BUTTON in tree RECHNER *)
  bsin     = 29;  (* BUTTON in tree RECHNER *)
  bcos     = 30;  (* BUTTON in tree RECHNER *)
  btan     = 31;  (* BUTTON in tree RECHNER *)
  beins    = 32;  (* BUTTON in tree RECHNER *)
  bzwei    = 33;  (* BUTTON in tree RECHNER *)
  bdrei    = 34;  (* BUTTON in tree RECHNER *)
  bplus    = 35;  (* BUTTON in tree RECHNER *)
  bminus   = 36;  (* BUTTON in tree RECHNER *)
  bmod     = 37;  (* BUTTON in tree RECHNER *)
  bquadrat = 38;  (* BUTTON in tree RECHNER *)
  bln      = 39;  (* BUTTON in tree RECHNER *)
  blog     = 40;  (* BUTTON in tree RECHNER *)
  bnull    = 41;  (* BUTTON in tree RECHNER *)
  bkomma   = 42;  (* BUTTON in tree RECHNER *)
  bexp     = 43;  (* BUTTON in tree RECHNER *)
  bgleich  = 44;  (* BUTTON in tree RECHNER *)
  bvzw     = 45;  (* BUTTON in tree RECHNER *)
  bklauf   = 46;  (* BUTTON in tree RECHNER *)
  bklzu    = 47;  (* BUTTON in tree RECHNER *)
  bend     = 48;  (* BUTTON in tree RECHNER *)
```

```
{*****************************************}
{* Listing 14  : Ein naturwissenschaftlicher
               Taschenrechner              *}
{*              (c) MAXON Computer GmbH    *}
{* Datei       : RECHNER1.PAS              *}
{* last update : 19.5.1988                *}
{*****************************************}

PROCEDURE do_rechner;

  CONST leerer_stack   = 0;
        max_op_stack   = 60;
        max_real_stack = 50;

        pi             = 3.1415926;
        deg            = 0;
        rad            = 1;
        grad           = 2;
        an             = 0;
        aus            = 1;

  TYPE  op_type = (f_klammer_auf, f_klammer_zu,
                   f_gleich,
                   f_ac, f_end, f_min, f_mr,
                   f_mminus, f_mplus,
                   f_mult, f_div, f_add, f_sub,
                   f_sin, f_cos, f_tan,
                   f_inv_sin,
                   f_inv_cos,
                  f_inv_tan, f_quadrat, f_wurzel,
                  f_ln, f_log,
                  f_exp, f_exp10);

       real_stack = RECORD
                      element :
                       ARRAY [1..max_real_stack]
                              OF real;
                      ptr     : integer;
                    END;

        op_stack = RECORD
                     element :
                       ARRAY [1..max_op_stack]
                              OF op_type;
                     ptr     : integer;
                   END;

  VAR stack_op      : op_stack;
      operator      : op_type;
      stack_val     : real_stack;
      value         : real;

      error         ,
      inv_modus     ,
      tri_modus     : integer;
      t_faktor      ,
      speicher      : real;
      first_rechner ,
      gueltig       : boolean;

      mant_vz       ,
      mant_wert     ,
      exp_text      ,
      exp_vz        ,
      exp_wert      ,
      inv_text      ,
      mem_text      ,
      deg_text      ,
      rad_text      ,
      gra_text      : str255;

  PROCEDURE setze_texte;

    VAR str : str255;
        i   : integer;
```

→

```
67:       BEGIN
68:         set_dtext(rechner_dialog,tvz,mant_vz,
                      system_font,te_left);
69:         str:=mant_wert;
70:         WHILE length(str)<8 DO
71:           str:=concat(str,' ');
72:        set_dtext(rechner_dialog,tmant,str,
                      system_font,te_left);
73:        set_dtext(rechner_dialog,texpvz,exp_vz,
                      small_font,te_left);
74:        set_dtext(rechner_dialog,texpexp,exp_wert,
                      small_font,te_left);
75:        set_dtext(rechner_dialog,texptext,exp_text,
76:                  system_font,te_left);
77:        set_dtext(rechner_dialog,tinv,inv_text,
                      small_font,te_left);
78:        set_dtext(rechner_dialog,tm,mem_text,
                      small_font,te_left);
79:        set_dtext(rechner_dialog,tdeg,deg_text,
                      small_font,te_left);
80:        set_dtext(rechner_dialog,trad,rad_text,
                      small_font,te_left);
81:        set_dtext(rechner_dialog,tgrad,gra_text,
                      small_font,te_left);
82:       END;
83:
84:     PROCEDURE setze_redraw;
85:
86:       BEGIN
87:         obj_redraw(rechner_dialog,tvz);
88:         obj_redraw(rechner_dialog,tmant);
89:         obj_redraw(rechner_dialog,texpvz);
90:         obj_redraw(rechner_dialog,texpexp);
91:         obj_redraw(rechner_dialog,texptext);
92:         obj_redraw(rechner_dialog,tinv);
93:         obj_redraw(rechner_dialog,tm);
94:         obj_redraw(rechner_dialog,tdeg);
95:         obj_redraw(rechner_dialog,trad);
96:         obj_redraw(rechner_dialog,tgrad);
97:       END;
98:
99:     PROCEDURE do_error;
100:
101:      VAR button : integer;
102:
103:      BEGIN
104:        CASE error OF
105:          1 : mant_wert:='Error ^';
106:          2 : mant_wert:='Error 0';
107:          3 : mant_wert:='Error U';
108:          4 : mant_wert:='Error ()';
109:          5 : mant_wert:='Error M';
110:        END;
111:        mant_vz:=' ';
112:        exp_vz:=' ';
113:        exp_wert:='  ';
114:        exp_text:=' ';
115:        inv_text:='   ';
116:        setze_texte;
117:        IF first_rechner THEN
118:          BEGIN
119:            button:=do_dialog(rechner_dialog,0);
120:            obj_setstate(rechner_dialog,button,
                             normal,true);
121:            first_rechner:=false;
122:          END
123:        ELSE
124:          BEGIN
125:            setze_redraw;
126:            button:=redo_dialog(rechner_dialog,0);
127:            obj_setstate(rechner_dialog,button,
                             normal,true);
128:          END;
129:      END;
130:
131:    PROCEDURE do_edit(VAR zahl     : real;
132:                      VAR operator : op_type);
133:
134:      CONST fertig    = 0;
135:            vorkomma  = 1;
136:            nachkomma = 2;
137:            exponent  = 3;
138:
139:      VAR status ,
140:          button : integer;
141:
```

```
142:      PROCEDURE setze_zahl(zahl : real);
143:
144:        VAR help : long_integer;
145:            i    : integer;
146:
147:        PROCEDURE exp_darstellung;
148:
149:          VAR log_help  ,
150:              mant_real : real;
151:              i         ,
152:              exp_int   : integer;
153:
154:          BEGIN
155:            log_help:=log(zahl);
156:            exp_int:=trunc(log_help);
157:            mant_real:=exp10(log_help-exp_int);
158:            mant_wert:=' .         ';
159:            mant_wert[1]:=
                        charakter(trunc(mant_real));
160:            mant_real:=mant_real-trunc(mant_real);
161:            FOR i:=3 TO 8 DO
162:              BEGIN
163:                mant_real:=mant_real*10;
164:                mant_wert[i]:=
                          charakter(trunc(mant_real));
165:                mant_real:=mant_real
                              -trunc(mant_real);
166:              END;
167:            exp_wert:='  ';
168:            exp_text:=' ';
169:            exp_vz:=' ';
170:            IF exp_int<>0 THEN
171:              IF abs(exp_int)=38 THEN
172:                BEGIN
173:                  IF exp_int<0 THEN
174:                    exp_vz:='-';
175:                  mant_wert:='9.999999';
176:                  exp_wert:='37';
177:                  exp_text:='E';
178:                END
179:              ELSE
180:                BEGIN
181:                  IF exp_int<0 THEN
182:                    exp_vz:='-';
183:                  exp_int:=abs(exp_int);
184:                  exp_wert[1]:=
                            charakter(exp_int DIV 10);
185:                  exp_wert[2]:=
                            charakter(exp_int MOD 10);
186:                  exp_text:='E';
187:                END;
188:            END;
189:
190:          BEGIN
191:            IF zahl>=0 THEN
192:              mant_vz:=' '
193:            ELSE
194:              mant_vz:='-';
195:            zahl:=abs(zahl);
196:            IF zahl<9999999 THEN
197:              IF zahl=long_trunc(zahl) THEN
198:                BEGIN
199:                  help:=long_trunc(zahl);
200:                  mant_wert:='      0. ';
201:                  FOR i:= 7 DOWNTO 1 DO
202:                    IF help>0 THEN
203:                      BEGIN
204:                        mant_wert[i]:=
                                  charakter(help MOD 10);
205:                        help:=help DIV 10;
206:                      END;
207:                  WHILE mant_wert[1]=' ' DO
208:                    BEGIN
209:                      delete(mant_wert,1,1);
210:                      insert(' ',mant_wert,8);
211:                    END;
212:                  exp_vz:=' ';
213:                  exp_wert:='  ';
214:                  exp_text:=' ';
215:                END
216:              ELSE
217:                exp_darstellung
218:            ELSE
219:              exp_darstellung;
220:            deg_text:='   ';
221:            rad_text:='   ';
```

→

```
222:          gra_text:='    ';
223:          CASE tri_modus OF
224:            deg  : deg_text:='DEG';
225:            rad  : rad_text:='RAD';
226:            grad : gra_text:='GRAD';
227:          END;
228:          IF inv_modus=an THEN
229:            inv_text:='INV'
230:          ELSE
231:            inv_text:='   ';
232:          IF speicher=0 THEN
233:            mem_text:=' '
234:          ELSE
235:            mem_text:='M';
236:        END;
237:
238:      FUNCTION rsc_char : char;
239:
240:        BEGIN
241:          CASE button OF
242:            bnull  : rsc_char:='0';
243:            beins  : rsc_char:='1';
244:            bzwei  : rsc_char:='2';
245:            bdrei  : rsc_char:='3';
246:            bvier  : rsc_char:='4';
247:            bfuenf : rsc_char:='5';
248:            bsechs : rsc_char:='6';
249:            bsieben: rsc_char:='7';
250:            bacht  : rsc_char:='8';
251:            bneun  : rsc_char:='9';
252:          END;
253:        END;
254:
255:      FUNCTION get_operator : op_type;
256:
257:        BEGIN
258:          CASE button OF
259:           bgleich  : get_operator:=f_gleich;
260:           bmal     : get_operator:=f_mult;
261:           bdurch   : get_operator:=f_div;
262:           bplus    : get_operator:=f_add;
263:           bminus   : get_operator:=f_sub;
264:           bac      : get_operator:=f_ac;
265:           bklauf   : get_operator:=f_klammer_auf;
266:           bklzu    : get_operator:=f_klammer_zu;
267:           bend     : get_operator:=f_end;
268:           bmin     : get_operator:=f_min;
269:           bmr      : get_operator:=f_mr;
270:           bmminus  : get_operator:=f_mminus;
271:           bmplus   : get_operator:=f_mplus;
272:           bsin     : IF inv_modus=aus THEN
273:                        get_operator:=f_sin
274:                      ELSE
275:                        get_operator:=f_inv_sin;
276:           bcos     : IF inv_modus=aus THEN
277:                        get_operator:=f_cos
278:                      ELSE
279:                        get_operator:=f_inv_cos;
280:           btan     : IF inv_modus=aus THEN
281:                        get_operator:=f_tan
282:                       ELSE
283:                        get_operator:=f_inv_tan;
284:           bquadrat : IF inv_modus=aus THEN
285:                        get_operator:=f_quadrat
286:                      ELSE
287:                        get_operator:=f_wurzel;
288:           bln      : IF inv_modus=aus THEN
289:                        get_operator:=f_ln
290:                      ELSE
291:                        get_operator:=f_exp;
292:           blog     : IF inv_modus=aus THEN
293:                        get_operator:=f_log
294:                      ELSE
295:                        get_operator:=f_exp10;
296:          END;
297:          inv_modus:=aus;
298:        END;
299:
300:      FUNCTION get_zahl : real;
301:
302:        VAR vor   ,
303:            nach  ,
304:            mant  ,
305:            expo  ,
306:            d_fak : real;
307:            str   : str255;
```

```
308:
309:        BEGIN
310:          mant:=0;
311:          vor:=0;
312:          nach:=0;
313:          expo:=0;
314:          d_fak:=1;
315:          str:=mant_wert;
316:          WHILE (str[1]<>'.') AND (length(str)>0) DO
317:              BEGIN
318:                IF str[1] in ['0'..'9'] THEN
319:                  vor:=vor*10+digit(str[1]);
320:                delete(str,1,1);
321:              END;
322:            WHILE length(str)>0 DO
323:              BEGIN
324:                IF str[1] in ['0'..'9'] THEN
325:                  BEGIN
326:                    nach:=10*nach+digit(str[1]);
327:                    d_fak:=d_fak*10;
328:                  END;
329:                delete(str,1,1);
330:              END;
331:            mant:=vor+nach/d_fak;
332:            IF mant_vz='-' THEN
333:              mant:=-mant;
334:            expo:=0;
335:            str:=exp_wert;
336:            WHILE length(str)>0 DO
337:              BEGIN
338:                IF str[1] in ['0'..'9'] THEN
339:                  expo:=10*expo+digit(str[1]);
340:                delete(str,1,1);
341:              END;
342:            IF exp_vz='-' THEN
343:              expo:=-expo;
344:            IF (expo>0)
                  AND (trunc(log(abs(mant)+1e-31)+1)+
345:                abs(expo)>38) THEN
346:              error:=1
347:            ELSE
348:              get_zahl:=mant*exp10(expo);
349:          END;
350:
351:        BEGIN
352:          gueltig:=false;
353:          status:=fertig;
354:          setze_zahl(zahl);
355:          REPEAT
356:            IF first_rechner THEN
357:              BEGIN
358:                setze_texte;
359:                first_rechner:=false;
360:                button:=do_dialog(rechner_dialog,0);
361:                obj_setstate(rechner_dialog,button,
                                 normal,true);
362:              END
363:            ELSE
364:              BEGIN
365:                setze_texte;
366:                setze_redraw;
367:              button:=redo_dialog(rechner_dialog,0);
368:                obj_setstate(rechner_dialog,button,
                                 normal,true);
369:              END;
370:            CASE button OF
371:              bmod     : BEGIN
372:                           tri_modus:=
                                  (tri_modus+1) MOD3;
373:                         deg_text:='    ';
374:                         rad_text:='    ';
375:                         gra_text:='    ';
376:                         CASE tri_modus OF
377:                          deg  : BEGIN
378:                                   t_faktor:=pi180;
379:                                   deg_text:='DEG';
380:                                 END;
381:                          rad  : BEGIN
382:                                   t_faktor:=1;
383:                                   rad_text:='RAD';
384:                                 END;
385:                          grad : BEGIN
386:                                 t_faktor:=pi/200;
387:                                 gra_text:='GRAD';
388:                                  END;
```

→

```
389:                        END;
390:                      END;
391:          binv      : BEGIN
392:                        inv_modus:=
                                 (inv_modus+1)MOD2;
393:                      IF inv_modus=an THEN
394:                        inv_text:='INV'
395:                      ELSE
396:                        inv_text:='   ';
397:                    END;
398:          bnull     ,
399:          beins     ,
400:          bzwei     ,
401:          bdrei     ,
402:          bvier     ,
403:          bfuenf    ,
404:          bsechs    ,
405:          bsieben   ,
406:          bacht     ,
407:          bneun     : BEGIN
408:                        gueltig:=true;
409:                        CASE status OF
410:                          fertig    : BEGIN
411:                                mant_wert:=' ';
412:                                mant_vz:=' ';
413:                                exp_text:=' ';
414:                                exp_wert:=' ';
415:                                exp_vz:=' ';
416:                                mant_wert[1]:=
                                           rsc_char;
417:                                status:=vorkomma;
418:                                END;
419:                          vorkomma  :
                               IF length(mant_wert)<7
                               THEN
420:                           BEGIN
421:                            mant_wert:=
422:                            concat(mant_wert,'');
423:                             mant_wert[length(
424:                            mant_wert)]:=rsc_char;
425:                           END;
426:                          nachkomma :
                               IF length(mant_wert)<8
                               THEN
427:                          BEGIN
428:                           mant_wert:=
429:                           concat(mant_wert,' ');
430:                           mant_wert[length(
431:                           mant_wert)]:=rsc_char;
432:                          END;
433:                          exponent  :
                               IF exp_wert=' ' THEN
434:                           exp_wert[1]:=rsc_char
435:                          ELSE
436:                           exp_wert[2]:=rsc_char;
437:                        END;
438:                      END;
439:          bkomma    : IF (status=vorkomma) AND
440:                        (length(mant_wert)<7) THEN
441:                        BEGIN
442:                          gueltig:=true;
443:                          status:=nachkomma;
444:                          mant_wert:=
                                 concat(mant_wert,'.');
445:                        END
446:                      ELSE
447:                        IF status=fertig THEN
448:                          BEGIN
449:                            gueltig:=true;
450:                            exp_text:=' ';
451:                            exp_vz:=' ';
452:                            exp_wert:='  ';
453:                            status:=nachkomma;
454:                            mant_wert:='0.';
455:                          END;
456:           bvzw     : BEGIN
457:                        gueltig:=true;
458:                        CASE status OF
459:                          fertig    ,
460:                          vorkomma  ,
461:                          nachkomma :
                               IF mant_vz=' ' THEN
462:                              mant_vz:='-'   '
463:                           ELSE
464:                              mant_vz:=' ';
465:                        exponent  : IF exp_vz=' '
```

```
                                    THEN
466:                                exp_vz:='-'
467:                               ELSE
468:                                exp_vz:=' ';
469:                          END;
470:                        END;
471:          bexp      : BEGIN
472:                        gueltig:=true;
473:                        exp_text:='E';
474:                        exp_vz:=' ';
475:                        exp_wert:='  ';
476:                        IF status=fertig THEN
477:                          BEGIN
478:                            mant_vz:=' ';
479:                            mant_wert:='1';
480:                          END;
481:                        status:=exponent;
482:                      END;
483:          bc        : BEGIN
484:                        gueltig:=true;
485:                        mant_wert:='0.      ';
486:                        mant_vz:=' ';
487:                        exp_wert:='  ';
488:                        exp_vz:=' ';
489:                        exp_text:=' ';
490:                        status:=fertig;
491:                      END;
492:           OTHERWISE: status:=fertig;
493:         END;
494:       UNTIL (status=fertig) AND (button<>bc) AND
495:             (button<>bmod) AND (button<>binv) AND
496:              (button<>bvzw);
497:       operator:=get_operator;
498:       zahl:=get_zahl;
499:     END;
500:
501:   PROCEDURE create_op(VAR x : op_stack);
502:
503:     BEGIN
504:       x.ptr:=leerer_stack;
505:     END;
506:
507:   FUNCTION is_empty_op(x : op_stack) : boolean;
508:
509:     BEGIN
510:       is_empty_op:=x.ptr=leerer_stack;
511:     END;
512:
513:   FUNCTION pop_op(VAR x : op_stack) : boolean;
514:
515:     VAR help : boolean;
516:
517:     BEGIN
518:       help:=NOT is_empty_op(x);
519:       IF help THEN
520:         x.ptr:=x.ptr-1;
521:       pop_op:=help;
522:     END;
523:
524:   FUNCTION push_op(VAR x : op_stack;
525:                        a : op_type) : boolean;
526:
527:     VAR help : boolean;
528:
529:     BEGIN
530:       WITH x DO
531:         BEGIN
532:           help:=ptr<max_op_stack;
533:           IF help THEN
534:             BEGIN
535:               ptr:=ptr+1;
536:               element[ptr]:=a;
537:             END;
538:         END;
539:       push_op:=help;
540:     END;
541:
542:   FUNCTION top_op(x : op_stack) : op_type;
543:
544:     BEGIN
545:       IF NOT is_empty_op(x) THEN
546:         top_op:=x.element[x.ptr];
547:     END;
548:
549:   FUNCTION anz_grund_op(x : op_stack) : integer;
550:                                                  →
```

```
551:      VAR count ,
552:          i      : integer;
553:
554:      BEGIN
555:        count:=0;
556:        FOR i:=1 TO x.ptr DO
557:          IF x.element[i] in [f_add,f_sub,f_div,
                                  f_mult] THEN
558:            count:=count+1;
559:        anz_grund_op:=count;
560:      END;
561:
562:    PROCEDURE create_real(VAR x : real_stack);
563:
564:      BEGIN
565:        x.ptr:=leerer_stack;
566:      END;
567:
568:    FUNCTION is_empty_real(x: real_stack): boolean;
569:
570:      BEGIN
571:        is_empty_real:=x.ptr=leerer_stack;
572:      END;
573:
574:    FUNCTION push_real(VAR x : real_stack;
575:                           a : real) : boolean;
576:
577:      VAR help : boolean;
578:
579:      BEGIN
580:        WITH x DO
581:          BEGIN
582:            help:=ptr<max_real_stack;
583:            IF help THEN
584:              BEGIN
585:                ptr:=ptr+1;
586:                element[ptr]:=a;
587:              END;
588:          END;
589:        push_real:=help;
590:      END;
591:
592:    FUNCTION pop_real(VAR x : real_stack): boolean;
593:
594:      VAR help : boolean;
595:
596:      BEGIN
597:        help:=NOT is_empty_real(x);
598:        IF help THEN
599:          x.ptr:=x.ptr-1;
600:        pop_real:=help;
601:      END;
602:
603:    FUNCTION top_real(x : real_stack) : real;
604:
605:      BEGIN
606:        IF NOT is_empty_real(x) THEN
607:          top_real:=x.element[x.ptr];
608:      END;
609:
610:    FUNCTION depth_real(VAR x:real_stack): integer;
611:
612:      BEGIN
613:        depth_real:=x.ptr;
614:      END;
615:
616:    PROCEDURE reset_rechner;
617:
618:      BEGIN
619:        inv_modus:=aus;
620:        tri_modus:=deg;
621:        t_faktor:=pi/180;
622:        create_op(stack_op);
623:        create_real(stack_val);
624:        operator:=f_gleich;
625:      END;
626:
627:    FUNCTION prior(operator : op_type) : integer;
628:
629:      BEGIN
630:        CASE operator OF
631:          f_klammer_auf : prior:=0;
632:          f_add         ,
633:          f_sub         : prior:=1;
634:          f_div         ,
635:          f_mult        : prior:=2;
```

```
636:        END;
637:      END;
638:
639:    PROCEDURE do_operator;
640:
641:      VAR operator : op_type;
642:          zahl1    ,
643:          zahl2    : real;
644:
645:      FUNCTION anz_operanden(operator : op_type) :
                                          integer;
646:
647:        BEGIN
648:          CASE operator OF
649:            f_mult     ,
650:            f_div      ,
651:            f_add      ,
652:            f_sub      : anz_operanden:=2;
653:            OTHERWISE  : anz_operanden:=1;
654:          END;
655:        END;
656:
657:      FUNCTION vorz_plus(op1 ,
658:                         op2 : real) : boolean;
659:
660:        VAR vorz : integer;
661:
662:        BEGIN
663:          vorz:=1;
664:          IF op1<0 THEN
665:            vorz:=-1;
666:          IF op2<0 THEN
667:            vorz:=-vorz;
668:          vorz_plus:=vorz=1;
669:        END;
670:
671:      BEGIN
672:        operator:=top_op(stack_op);
673:        IF NOT pop_op(stack_op) THEN
674:          error:=4;
675:        IF anz_operanden(operator)=2 THEN
676:          BEGIN
677:            zahl2:=top_real(stack_val);
678:            IF NOT pop_real(stack_val) THEN
679:              error:=4;
680:            zahl1:=top_real(stack_val);
681:            IF NOT pop_real(stack_val) THEN
682:              error:=4;
683:          END
684:        ELSE
685:          BEGIN
686:            zahl1:=top_real(stack_val);
687:            IF NOT pop_real(stack_val) THEN
688:              error:=4;
689:          END;
690:        IF error=0 THEN
691:          CASE operator OF
692:            f_log      : IF zahl1>0 THEN
693:                           zahl1:=log(zahl1)
694:                         ELSE
695:                           error:=3;
696:            f_exp10    : IF zahl1<38 THEN
697:                           zahl1:=exp10(zahl1)
698:                         ELSE
699:                           error:=1;
700:            f_ln       : IF zahl1>0 THEN
701:                           zahl1:=ln(zahl1)
702:                         ELSE
703:                           error:=3;
704:            f_exp      : IF zahl1<=87.49823 THEN
705:                           zahl1:=exp(zahl1)
706:                         ELSE
707:                           error:=1;
708:            f_sin      : IF abs(zahl1*t_faktor)<=30
                             THEN
709:                         zahl1:=sin(zahl1*t_faktor)
710:                         ELSE
711:                           error:=1;
712:            f_inv_sin  : IF abs(zahl1)<=1 THEN
713:                           IF abs(zahl1)=1 THEN
714:                            zahl1:=
                                   zahl1*pi/2/t_faktor
715:                         ELSE
716:                         zahl1:=arctan(zahl1/sqrt
                                   (-zahl1*zahl1+1))
                                                  →
```

```
717:                                    /t_faktor
718:                          ELSE
719:                            error:=3;
720:          f_cos      : IF abs(zahl1*t_faktor)<=30
                          THEN
721:                          zahl1:=cos(zahl1*t_faktor)
722:                          ELSE
723:                            error:=1;
724:          f_inv_cos  : IF abs(zahl1)<=1 THEN
725:                            IF abs(zahl1)=1 THEN
726:                             zahl1:=
                                (1-zahl1)*pi/2/t_faktor
727:                            ELSE
728:                             zahl1:=(pi/2-
                               arctan(zahl1/sqrt(-zahl1*
                                zahl1+1)))/t_faktor
729:                          ELSE
730:                            error:=3;
731:          f_tan      : IF abs(zahl1*t_faktor)<=30
                          THEN
732:                           IF cos(zahl1*t_faktor)<>0
                                 THEN
733:                             zahl1:=sin(zahl1*
                                        t_faktor)/
734:                                    cos(zahl1*
                                        t_faktor)
735:                          ELSE
736:                            error:=3
737:                          ELSE
738:                            error:=1;
739:          f_inv_tan  : zahl1:=arctan(zahl1)/
740:                               t_faktor;
741:          f_quadrat  : IF abs(zahl1)<=9.999999e18
                          THEN
742:                          zahl1:=sqr(zahl1)
743:                          ELSE
744:                            error:=1;
745:          f_wurzel   : IF zahl1>0 THEN
746:                          zahl1:=sqrt(zahl1)
747:                          ELSE
748:                            error:=3;
749:          f_min      : speicher:=zahl1;
750:          f_mr       : zahl1:=speicher;
751:          f_mminus   : IF ((abs(speicher)<=
                            4.999999e37) AND
752:                          (abs(zahl1)<=
                            4.999999e37)) OR
753:                         vorz_plus(speicher,zahl1)
                            THEN
754:                          speicher:=speicher-zahl1
755:                          ELSE
756:                            error:=5;
757:          f_mplus    : IF ((abs(speicher)<=
                            4.999999e37) AND
758:                         (abs(zahl1)<4.999999e37))
                            OR NOT
759:                         vorz_plus(speicher,zahl1)
                            THEN
760:                          speicher:=speicher+zahl1
761:                          ELSE
762:                            error:=5;
763:          f_add      : IF ((abs(zahl1)<=
                            4.999999e37) AND
764:                          (abs(zahl2)<=
                            4.999999e37)) OR NOT
765:                          vorz_plus(zahl1,zahl2)
                            THEN
766:                          zahl1:=zahl1+zahl2
767:                          ELSE
768:                            error:=1;
769:          f_sub      : IF ((abs(zahl1)<=
                            4.999999e37) AND
770:                          (abs(zahl2)<=
                            4.999999e37)) OR
771:                          vorz_plus(zahl1,zahl2)
                            THEN
772:                          zahl1:=zahl1-zahl2
773:                          ELSE
774:                            error:=1;
775:          f_mult     :IF (zahl1<1) OR (zahl2<1) OR
776:                         (log(abs(zahl1)+1e-7)+
777:                          log(abs(zahl2)+1e-7)<38)
                          THEN
778:                          zahl1:=zahl1*zahl2
779:                          ELSE
780:                            error:=1;
```

```
781:            f_div      : IF zahl2<>0 THEN
782:                           IF (zahl1<1) OR (zahl2>1)
                                  OR
783:                              (log(abs(zahl1)+1e-7)-
784:                               log(abs(zahl2))<38)
                               THEN
785:                             zahl1:=zahl1/zahl2
786:                           ELSE
787:                             error:=1
788:                         ELSE
789:                           error:=2;
790:          END;
791:        IF NOT push_real(stack_val,zahl1) THEN
792:          error:=4;
793:      END;
794:
795:    BEGIN
796:      first_rechner:=true;
797:      speicher:=0;
798:      value:=0;
799:      reset_rechner;
800:      begin_update;
801:      REPEAT
802:        error:=0;
803:        do_edit(value,operator);
804:        IF is_empty_op(stack_op) THEN
805:          BEGIN
806:            create_real(stack_val);
807:            IF push_real(stack_val,value) THEN
808:              ;
809:            END
810:        ELSE
811:          IF gueltig THEN
812:           BEGIN
813:            IF NOT push_real(stack_val,value) THEN
814:                error:=4;
815:           END
816:          ELSE
817:            IF (operator<>f_klammer_auf) AND
818:               (anz_grund_op(stack_op)=
                          depth_real(stack_val)) THEN
819:              BEGIN
820:                IF NOT push_real(stack_val,
                            top_real(stack_val)) THEN
821:                  error:=4;
822:              END;
823:        IF error=0 THEN
824:          CASE operator OF
825:            f_min      ,
826:            f_mr       ,
827:            f_mminus   ,
828:            f_mplus    ,
829:            f_sin      ,
830:            f_cos      ,
831:            f_tan      ,
832:            f_inv_sin  ,
833:            f_inv_cos  ,
834:            f_inv_tan  ,
835:            f_quadrat  ,
836:            f_wurzel   ,
837:            f_ln       ,
838:            f_log      ,
839:            f_exp      ,
840:            f_exp10    : IF NOT push_op(stack_op,
                                    operator) THEN
841:                           error:=4
842:                         ELSE
843:                           do_operator;
844:            f_mult     ,
845:            f_div      ,
846:            f_add      ,
847:            f_sub      : BEGIN
848:                           WHILE (prior(top_op
                                  (stack_op))>=
849:                                  prior(operator))
850:                                 AND NOT is_empty_op
                                          (stack_op)
851:                                 AND (error=0) DO
852:                             do_operator;
853:                           IF NOT push_op(stack_op,
                                      operator) THEN
854:                             error:=4;
855:                         END;
856:            f_klammer_auf :
857:                           IF NOT gueltig THEN
                                                    →
```

```
858:                          IF NOT push_op(stack_op,
                                        operator) THEN
859:                            error:=4;
860:          f_klammer_zu:
861:                        BEGIN
862:                         WHILE (top_op(stack_op)<>
                                  f_klammer_auf)
863:                           AND NOT is_empty_op
                                        (stack_op)
864:                            AND (error=0) DO
865:                          do_operator;
866:                         IF top_op(stack_op)=
                                f_klammer_auf THEN
867:                          IF NOT pop_op(stack_op)
                              THEN
868:                            error:=4;
869:                        END;
870:          f_gleich   : BEGIN
871:                         WHILE NOT is_empty_op
                                    (stack_op) DO
872:                          do_operator;
873:                         value:=top_real
                                    (stack_val);
874:                         reset_rechner;
875:                         IF NOT push_real
                              (stack_val,value) THEN
876:                          ;
877:                        END;
878:          f_ac       : reset_rechner;
879:        END;
880:      IF error<>0 THEN
881:        BEGIN
882:          do_error;
883:          reset_rechner;
884:          error:=0;
885:        END;
886:      IF is_empty_real(stack_val) THEN
887:        value:=0
888:      ELSE
889:        value:=top_real(stack_val);
890:    UNTIL operator=f_end;
891:    end_dialog(rechner_dialog);
892:    end_update;
893:  END;
```

```
 1:  {****************************************}
 2:  {* Listing 15  : Resource-Handling für den
                       Taschenrechner          *}
 3:  {*               (c) MAXON Computer GmbH  *}
 4:  {* Datei       : RECHNER.PAS              *}
 5:  {* last update : 19.5.1988                *}
 6:  {****************************************}
 7:
 8:  {$s10}
 9:
10:  PROGRAM rechner (input,output);
11:
12:  CONST {$i gemconst.pas}
13:        {$i trixcons.pas}
14:        {$i rechner.i}
15:
16:  TYPE  {$i gemtype.pas}
17:        {$i trixtype.pas}
18:
19:  VAR   msg             : message_buffer;
20:        apl_name        : str255;
21:        apl_nr          ,
22:        menu_nr         ,
23:        event           ,
24:        dummy           : integer;
25:
26:        rechner_dialog  : dialog_ptr;
27:
28:  {$i gemsubs.pas}
29:  {$i trixsubs.pas}
30:  {$i hilf.pas}
31:  {$i rechner1.pas}
32:
33:  FUNCTION initialisieren : boolean;
34:
35:    VAR ok : boolean;
36:
37:    BEGIN
38:      ok:=load_resource('A:\RECHNER.RSC');
39:      IF ok THEN
40:        BEGIN
41:          apl_name:='  Taschenrechner';
42:          menu_nr:=menu_register(apl_nr,apl_name);
43:          find_dialog(rechner,rechner_dialog);
44:          center_dialog(rechner_dialog);
45:        END;
46:      initialisieren:=ok;
47:    END;
48:
49:  BEGIN
50:    apl_nr:=init_gem;
51:    IF apl_nr>=0 THEN
52:      IF initialisieren THEN
53:        WHILE true DO
54:          BEGIN
55:          event:=get_event(e_message,0,0,0,0,true,
56:                                0,0,0,0,true,0,0,0,0,
                                msg,dummy,dummy,dummy,
57:                                dummy,dummy,dummy);
58:            IF msg[0]=ac_open THEN
59:              do_rechner;
60:          END;
61:  END.
```


Multi I/O – Karte und AD – Wandler

für Ihren Atari ST 260 / 520 / 1040

- 15 Ein / 16 Ausgänge mit TTL – Pegel (frei programmierbar)
- 8 Bit AD – Wandler (max. 100 KHz Abtastfreq.)
- erweiterbar auf 8 – Kanalbetrieb
- triggerbar auf Wort und Bit
- Romport – Anschluss
- Eingangsspg. 2,5 V

für 125.00 DM

plus Porto u. Verpackung

A. Esch
Hauptstr. 50
5405 Ochtendung
Tel.: 02625 / 1231

Hardware – Uhr

- Schaltjahrerkennung
- Sommer – Winterzeit – Kennung
- autom. Jahresfortschaltung
- Langzeit akkugepuffert
- keine Änderungen im Ab – schirmblech notwendig

DM 65.00

plus Porto u. Verpackung

ST-Floppy-Stationen:

anschlußfertig doppelseitig garantiert kompatibel mit formschönem, hochwertigem Metallgehäuse mit der einzigartigen automatischen Netzanschaltung

ESN:	3,5"- Einzelstation, 42 * 108 * 230	**249,-- DM**
ESN/A:	dto mit Ausgang für Laufwerk B	**268,-- DM**
DSN:	3,5"- Doppelstation, 75 * 106 * 230	**398,-- DM**
GSN/3:	5,25" - Einzelstation, Ausgang für 3. Laufwerk, 40/80 Spuren, 50 * 152 * 290 mm	**368,-- DM**

Dipl.Ing. Gerhard Trumpp
Mitterlängstrasse 7
8039 Puchheim - Ort

Tel. 089 / 80 68 23
ab 17 Uhr

Programme unter GEM Teil 2

Der letzte Artikel war mehr theoretisch, heute schreiten wir zur Praxis. Ein Beispielprogramm zeigt uns, wie man sich ein Grundgerüst für ein GEM-Programm verschaffen kann. Das Rahmenprogramm ist in der Lage, einen eigenen Desktop (mit einem Icon) anzumelden, ein Menü darzustellen und natürlich auch, den Desktop und das Menü zu verwalten. Unser Desktop beinhaltet lediglich ein Icon, damit das Prinzip der Programmierung klar wird. Möchte man jedoch mehrere Icons haben, so müssen diese entweder vorher (beim Entwurf mit dem Resource-Construction-Programm) deklariert werden oder der Objekt-Baum (ein Array von Objekten, die miteinander verknüpft sind) muß erweitert werden. Die Erweiterung scheitert jedoch daran, daß ein Feld nicht so einfach vergrößert werden kann. Also muß der gesamte Objekt-Baum (das Array, nicht jedoch Texte etc.) an eine andere Stelle kopiert werden, die genügend Platz bietet.

Sollen eigene Dinge auf das Desktop gebracht werden, so darf dies niemals direkt geschehen. Werden Ausgaben direkt auf das Desktop gemacht und wird danach darüber ein Fenster gebracht, so ist die Ausgabe verschwunden, wenn das Fenster geschlossen wird. Für die eigenen Ausgaben findet die *Application-Block-Struktur* (USERBLK) Verwendung. Somit erhält das Programm immer dann eine Meldung (über USERBLK), wenn das Desktop zu erneuern ist. Das Menü sollte immer die Menüeinträge sperren, die nicht anwählbar sein sollen. Dies geschieht mit *menu_ieanble*, indem die Adresse des Menüs, die Nummer des Eintrags und eine 0 (für desaktivieren) oder eine 1 (für aktivieren) übergeben wird. Das Beispielprogramm macht das nicht, da gar keine Lade- oder Speicher-Routinen implementiert wurden. Das Tastenkürzel steht jeweils am rechten Rand des Menüeintrags. Das entsprechende Tastaturereignis wird von dem Beispielprogramm ausgewertet. Soll das Programm noch unabhängiger gestaltet werden, so besteht die Möglichkeit, den Text des Menüeintrags nach Programmstart zu untersuchen. Dazu wird der Menü-Objektbaum nach den betreffenden Einträgen abgesucht und das Tastenkürzel ausgewertet. Somit wird auch bei einer Veränderung des Resource-Files der Tastaturaufruf geändert. Das Bild 1 zeigt das Beispiel-Programm in Aktion. Benutzt wurde GEM der Version 2.x, weshalb auch das "Desk"-Menü am rechten Rand zu finden ist. Eine Alertbox ist auf Bild 2 zu sehen und auf dem dritten Bild eines der beiden Drop-Down-Menüs.

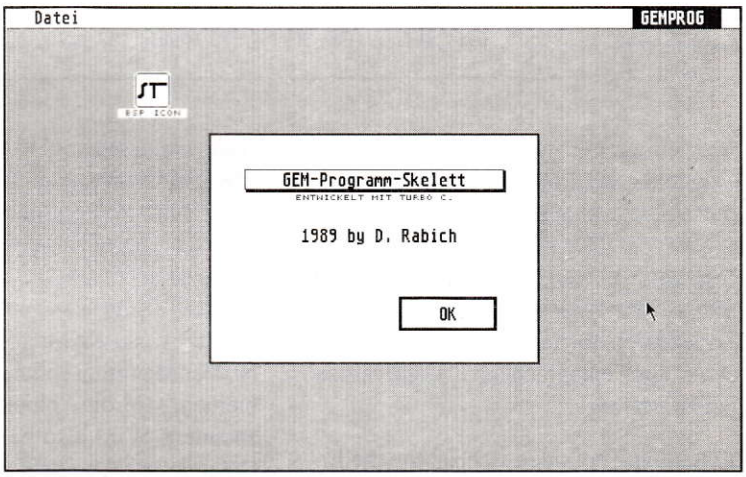


Bild 1: Das Beispielprogramm in Aktion



Bild 2: Eine der Alertboxen



Bild 3: Ein heruntergeklapptes Menü

Größe angepaßt und ausgegeben. Dann wird der Scrap-Pfad gesetzt, falls er noch nicht existiert. Schließlich erfolgt die Ausgabe des Menüs und die Maus wird wieder ein Pfeil. Eine Parameterdatei könnte in dieser Routine auch eingelesen und ausgewertet werden. Die Routine *exit_prg* ist vergleichsweise kurz. Sie meldet das Menü und das Desktop ab, gibt den Resource-Speicher wieder frei und meldet das Programm ab.

Die Handle-Routine *hdle_mouse* ermittelt zuerst das Objekt, welches sich an der Mausposition befindet. Dann wird zwischen einem Doppel- und einem Einfachklick unterschieden. Bei einem Doppelklick wird das entsprechende Icon invertiert, eine Alertbox ausgegeben und danach die Invertierung wieder aufgehoben. Wurde nur einmal geklickt, ist noch zu unterscheiden, ob die Maustaste mittlerweile losgelassen wurde oder nicht. Ist die linke Maustaste noch gedrückt, wird das Icon bewegt. Ist sie nicht gedrückt, wird das Icon invertiert, wenn sich der Mauszeiger darüber befindet, bzw. die Invertierung aufgehoben, wenn außerhalb des Icons geklickt wurde.

In der Maus-Handling-Routine werden zwei neue Routinen aufgerufen. Dies ist zum einen eine Ausgabe-Routine und zum anderen eine Routine, die die Icon-Bewegung übernimmt. Die Ausgabe-Routine *draw_objc* gibt ein Objekt unter Berücksichtigung der Rechteckliste aus. Dies ist notwendig, da nicht immer ein komplettes Objekt neu gezeichnet werden muß. Denken Sie nur daran, wenn ein Fenster ein Icon halb überdeckt!

Move_objc versteckt zuerst das Objekt und gibt es über *draw2_objc* aus. Die zweite Ausgabe-Routine ist erforderlich, da mit HIDETREE versteckte Objekte nicht direkt angesprochen werden können. Dann wird der Arbeitsbereich - der Bereich, in dem das Objekt bewegt werden darf - ermittelt, die Maus als flache Hand dargestellt und das Objekt via *graf_dragbox* verschoben. Anschließend stehen die neuen Koordinaten des Icons fest. Wenn man nur bestimmte Positionen zuläßt, so ist an dieser Stelle eine Anpassung - beispielsweise auf Byte-Grenze - vorzunehmen. Letztendlich wird die Maus wieder als Pfeil dargestellt und das Icon ausgegeben. Damit seien die Erklärungen beendet. Im Listing 2 finden sich noch weitere Erklärungen und Anregungen. Ich hoffe, daß Ihnen die kleine Einführung in die Benutzung des GEM gefallen hat und sie als nächstes Programm ein wunderschönes GEM-Programm schreiben werden.

Dietmar Rabich

```
➡ APPL-Funktionen          ➡ MENU-Funktionen          ➡ SCRP-Funktionen
      appl_init                  menu_bar                   scrp_read
      appl_exit                  menu_tnormal               scrp_write
➡ EVNT-Funktionen          ➡ OBJC-Funktionen          ➡ GRAF-Funktionen
      evnt_multi                 objc_draw                  graf_dragbox
                                 objc_find                  graf_mouse
                                 objc_offset                graf_mkstate
➡ RSRC-Funktionen
      rsrc_load
      rsrc_free
      rsrc_gaddr (R_TREE,R_STRING)
➡ FORM-Funktionen
      form_do
      form_dial (FMD_START,FMD_GROW,FMD_SHRINK,FMD_FINISH)
      form_alert
      form_center
➡ WIND-Funktionen
      wind_get (WF_WORKXYWH,WF_FIRSTXYWH,WF_NEXTXYWH)
      wind_set (WF_NEWDESK)
      wind_update (END_UPDATE,BEG_UPDATE,END_MCTRL,BEG_MCTRL)
```

Bild 4: Im Beispielprogramm benutzte AES-Funktionen

Wenden wir uns nun dem Programm zu. Es wurde mit Turbo C geschrieben, sollte aber auch mit anderen C-Compilern übersetzbar sein. Im Listing 1 ist die für unser Programm notwendige Projekt-Datei aufgeführt. Die Projekt-Dateien sind Turbo C-spezifisch, für andere Entwicklungssysteme sind entsprechende Vorbereitungen zu treffen.

Im Listing 2 befinden sich zahlreiche Erklärungen zu den einzelnen Schritten, weshalb diese hier nicht ausführlich erklärt werden sollen. Das Hauptprogramm *main* ist für den groben Rahmen zuständig. Dazu gehört die Initialisierung *init_prg* und das Abmelden *exit_prg*. In der while-Schleife wird solange gewartet, bis die Variable *finish* den Wert 1 (analog zur TRUE) erhält. Für den Fall, daß ein Menüeintrag angeklickt wurde, wird die Routine *hdle_menu* aufgerufen, bei einem Mausklick *hdle_mouse* und bei einem Tastaturereignis *hdle_key*. Die Initialisierungsroutine *init_prg* meldet das Programm zuerst beim AES an, stellt dann die Maus als Sanduhr (GEM 2.x) bzw. als Biene (GEM 1.x) dar, weil die folgenden Aktionen doch etwas mehr Zeit erfordern. Dann wird das Resource-File geladen und ausgewertet. Direkt nach der Auswertung wird das Desktop in der

Hdle_menu ist wohl die einfachste unter den Handle-Routinen. Entsprechend dem Menüeintrag wird eine Aktion durchgeführt und schließlich der Menütitel wieder normal dargestellt. Die normale Darstellung ist übrigens spätestens dann zu wählen, wenn das Menü wieder verfügbar ist. Der Menütitel sollte solange invertiert bleiben, wie die "blockierende" Aktion andauert. Wird also beispielsweise eine Datei geladen, so hebt man die Invertierung auf, sobald diese geladen ist. Von *hdle_menu* aus wird auch die Informationsdialogbox über *do_info* aufgerufen. In *do_info* wird ein vollständiger Dialog durchgeführt.

Ruft ein Programm mehrere Dialogboxen auf, so empfiehlt es sich, den Vor- und den Nachbereitungsteil in getrennten Funktionen zu deklarieren, denn diese unterscheiden sich in der Regel nicht. Der Durchführungsteil kann sich erheblich unterscheiden, beispielsweise können wie bei einem Fileselector Dateinamen gescrollt werden. Die dritte Handle-Routine - *hdle_key* - ist schon etwas komplizierter. Dort werden nicht nur die oberen 8 Bit des Tastencodes ausgewertet, sondern es wird auch entsprechend der gedrückten Taste der Menütitel invertiert und dann wieder normal dargestellt.

Literatur:

[1] Atari ST Profibuch, H.-D. Jankowski/ J. F. Reschke/ D. Rabich, Sybex 1987/88/89
[2] GEM Programmier-Handbuch, P. Balma/ W. Fitler, Sybex 1987/88
[3] Professionel GEM, T. Oren, ANTIC 1985/86

```
                ; GEMPROG.PRJ
                ; -----------
*               ; name of executable pgm is
                  topmost window
=               ; list of modules follows...
tcstart.o       ; startup code

gemprog (gemprog.h, interna.h, scankey.h) ;
 Hauptmodul
interna (interna.h) ; ST-interne Funktionen

tcstdlib.lib    ; Standard-Library
tctoslib.lib    ; TOS-Library
tcgemlib.lib    ; GEM-Library
```

```
/*********************************************/
/* GEM-Rahmen-Programm    Vers. 1.00c        */
/*                                           */
/* Datum: 13. Mai 1989                       */
/* Letztes Edier-Datum: 1. September 1989    */
/* Autor: D. Rabich     Entwickelt mit Turbo C */
/*                                           */
/* Quelldatei: GEMPROG.C                     */
/*********************************************/


/* Include-Dateien */
# include "gemprog.h"    /* Resource-Datei */
# include "interna.h"    /* ST-Interna     */
# include "scankey.h"    /* Scancodes      */
# include <aes.h>        /* AES-Routinen   */


/* Definition zweier Funktionen */
# define min(a,b)  ((a) < (b) ? (a) : (b))
                                  /* Minimum */
# define max(a,b)  ((a) > (b) ? (a) : (b))
                                  /* Maximum */


/* Handle des Desktops definieren */
# define DESK 0


/* Definitionen */
# define MAXSTRING 80

# define INIT_OK     0
# define NO_INIT     1
# define NO_RESOURCE 2


/* Strings */
char no_resource[] = "[3][ |  GEMPROG.RSC|
                     nicht gefunden.  | ][ OK ]";
char no_init[]     = "[3][ |  Initialisierung |
                      erfolglos.| ][ OK ]";
char rsc_name[]    = "GEMPROG.RSC";
char std_scrap[]   = "x:\\CLIPBRD";
/* hier auch Parameter-Datei angeben! */
/*  Format  : <Programm-Name>.INF     */
/*  Beispiel: GEMPROG.INF             */


/* Struktur(en) */
typedef struct { int x,y; /* kennzeichnet
                             einen Punkt  */
                 } POINT;


/* Prototypen */
static int  init_prg(void);
static void exit_prg(int);
static void draw_objc(int, OBJECT*, int);
static void clip(GRECT*, GRECT*);
static void draw2_objc(int, OBJECT*, int);
static void move_objc(int, OBJECT*, int);
static void do_info(void);
static void hdle_menu(int, int);
static void hdle_mouse(int, POINT*);
static void hdle_key(int, int);
```

```

/* Variablen */
int       apl_id,     /* Applikations-ID */
          finish = 0; /* Flag für Ende */
OBJECT    *menu,      /* Menü */
          *desktop,   /* Desktop */
          *infodial,  /* Dialogbox */
          *aktion,    /* Alert-Boxen */
          *laden,     /* Alert-Boxen */
          *speichern, /* Alert-Boxen */
          *ausgabe;   /* Alert-Boxen */
GRECT     desksize;   /* Desk-Grösse */


/*********************************************/
/* Anmeldung des Programms                   */
/* Aufgabe: Anmeldung beim AES, laden und
            auswerten der Resource-Datei,    */
/*          Scrap-Pfad setzen, Menü ausgeben,
            Desktop setzen,                  */
/*          Mauszeiger auf Pfeil setzen      */
/*********************************************/

static int init_prg (void)

{ char scrp_path[MAXSTRING];

  /* anmelden */
  apl_id = appl_init();
  if (apl_id < 0)           /* korrekte ID? */
    return(NO_INIT);

  /* Maus als Biene/Sanduhr */
  graf_mouse(HOURGLASS,0L);

  /* Resource-Datei laden */
  if (!rsrc_load(rsc_name)) /* nicht
                               gefunden? */
    { form_alert(1,no_resource);
      return(NO_RESOURCE);
    }
  /* Accessories, die unter GEM 2.x laufen
     sollen, */
  /* dürfen keine Resource-Dateien laden. */

  /* Anfangsadressen ermitteln */
  rsrc_gaddr(R_TREE,  MENUE,    &menu); /* Menü */
  rsrc_gaddr(R_TREE,  DESKTOP, &desktop);
                               /* Desktop */
  rsrc_gaddr(R_TREE,  INFODIAL,&infodial);
                               /* Dialog  */

  rsrc_gaddr(R_STRING,DOPPEL,  &aktion);
                               /* Alert-Boxen */
  rsrc_gaddr(R_STRING,LADEN,   &laden);
  rsrc_gaddr(R_STRING,SPEICHER,&speichern);
  rsrc_gaddr(R_STRING,AUSGABE, &ausgabe);

  /* Grösse des Desktops ermitteln */
  wind_get(DESK,WF_WORKXYWH,
           &desksize.g_x,&desksize.g_y,
           &desksize.g_w,&desksize.g_h);

  /* Object-Struktur anpassen */
  desktop[ROOT].ob_x      = desksize.g_x;
  desktop[ROOT].ob_y      = desksize.g_y;
  desktop[ROOT].ob_width  = desksize.g_w;
  desktop[ROOT].ob_height = desksize.g_h;

  /* Default-Objektbaum setzen (Desktop) */
  objc_draw(desktop,ROOT,MAX_DEPTH,
            desksize.g_x,desksize.g_y,
            desksize.g_w,desksize.g_h);
  wind_set(DESK,WF_NEWDESK,(long) desktop,0);

  /* Scrap-Pfad setzen */
  scrp_read(scrp_path);
  if (scrp_path[0] == 0)
    { std_scrap[0] = 'A' + (char) boot_dev();
      scrp_write(std_scrap);
    }

  /* hier kann die INF-Datei geladen werden! */
  /* Der Pfad wird mit shel_find ermittelt.  */

  /* Menü ausgeben */                    →
```

```
142:    menu_bar(menu,1);
143:
144:    /* Maus als Pfeil */
145:    graf_mouse(ARROW,0L);
146:
147:    return(INIT_OK);
148:  }
149:
150:
151:  /*********************************************/
152:  /* Abmeldung des Programms                   */
153:  /* Aufgabe: Menü abmelden, Desktop
                              zurücksetzen,       */
154:  /*          Resource-Speicher freigeben,
              Abmeldung beim AES                  */
155:  /*********************************************/
156:
157:  static void exit_prg (int rc)
158:
159:  {
160:
161:    if (rc == INIT_OK)
162:      { /* Menü abmelden */
163:        menu_bar(menu,0);
164:
165:        /* Leerer Default-Objektbaum */
166:        wind_set(DESK,WF_NEWDESK,0L,0);
167:
168:        /* Resource-Speicher freigeben */
169:        rsrc_free();
170:      }
171:
172:    if ((rc == INIT_OK) || (rc == NO_RESOURCE))
173:      /* abmelden */
174:      appl_exit();
175:  }
176:
177:
178:  /*********************************************/
179:  /* Objekt ausgeben                           */
180:  /* Aufgabe: Objekt unter Berücksichtigung der
              Rechteckliste ausgeben              */
181:  /*********************************************/
182:
183:  static void draw_objc (int handle, OBJECT *tree,
                           int child)
184:
185:  { int wi_gw1,wi_gw2,wi_gw3,wi_gw4;
186:
187:    wind_update(BEG_UPDATE); /* Fenster wird
                                  erneuert */
188:
189:    /* Rechteckliste abarbeiten */
190:    wind_get(handle,WF_FIRSTXYWH,&wi_gw1,&wi_gw2,
191:                                 &wi_gw3,&wi_gw4);
                    /* erster Rechteck der Liste */
192:    while (wi_gw3 && wi_gw4)
193:      { objc_draw(tree,child,MAX_DEPTH,wi_gw1,
194:                           wi_gw2,wi_gw3,wi_gw4);
                              /* Objekt ausgeben */
195:        wind_get(handle,WF_NEXTXYWH,&wi_gw1,
196:                           &wi_gw2,&wi_gw3,&wi_gw4);
                              /* nächstes Rechteck */
197:      }
198:
199:    wind_update(END_UPDATE); /* Erneuerung
                                  beendet */
200:  }
201:
202:
203:  /*********************************************/
204:  /* Clipping                                  */
205:  /* Aufgabe: Clippt das Rechteck a unter
              Berücksichtigung von b              */
206:  /*********************************************/
207:
208:  static void clip (GRECT *a, GRECT *b)
209:
210:  { a->g_x = max(a->g_x,b->g_x); /* linke obere
                                      Ecke */
211:    a->g_y = max(a->g_y,b->g_y);
212:    a->g_w = min(a->g_x + a->g_w, b->g_x +
              b->g_w);  /* rechte untere Ecke */
213:    a->g_h = min(a->g_y + a->g_h, b->g_y + b->g_h);
214:
215:    a->g_w -= a->g_x; /* Breite */
216:    a->g_h -= a->g_y; /* Höhe */
217:  }
218:
219:
220:  /*********************************************/
221:  /* Object ausgeben                           */
222:  /* Aufgabe: Object auch unter Berücksichtigung
                von HIDETREE ausgeben             */
223:  /*********************************************/
224:
225:  static void draw2_objc (int handle, OBJECT *tree,
                            int child)
226:
227:  { GRECT a, b;
228:
229:    wind_update(BEG_UPDATE); /* Fenster erneuern */
230:
231:    /* Rechteckliste abarbeiten */
232:    wind_get(handle,WF_FIRSTXYWH,
233:             &a.g_x,&a.g_y,&a.g_w,&a.g_h);
                       /* erstes Rechteck der Liste */
234:    while (a.g_w && a.g_h)
235:      { objc_offset(tree,child,&b.g_x,&b.g_y);
                          /* Position des Objekts */
236:        b.g_w = tree[child].ob_width;
237:        b.g_h = tree[child].ob_height;
238:        clip(&a,&b); /* Clipping */
239:        objc_draw(tree,ROOT,MAX_DEPTH,
240:                  a.g_x,a.g_y,a.g_w,a.g_h);
                       /* Objekt ausgeben */
241:        wind_get(handle,WF_NEXTXYWH,
242:                 &a.g_x,&a.g_y,&a.g_w,&a.g_h);
                       /* nächstes Rechteck */
243:      }
244:
245:    wind_update(END_UPDATE); /* Erneuerung
                                  beendet */
246:
247:    /* In dieser Funktion könnte auch
       form_dial(FMD_FINISH,...) Verwendung */
248:    /*  finden. */
249:  }
250:
251:
252:  /*********************************************/
253:  /* Objekt bewegen                            */
254:  /* Aufgabe: Objekt verstecken, bewegen und
              wieder darstellen                   */
255:  /*********************************************/
256:
257:  static void move_objc (int handle, OBJECT *tree,
                           int child)
258:
259:  { int x,y,ox,oy,nx,ny;
260:
261:    /* Objekt verstecken */
262:    tree[child].ob_flags |= HIDETREE;
263:    draw2_objc(handle,tree,child);
264:
265:    /* Objekt child innerhalb des Objekts ROOT
       bewegen */
266:    objc_offset(tree,ROOT,&x,&y); /* Offset
                                     berechnen */
267:    objc_offset(tree,child,&ox,&oy);
268:    wind_update(BEG_UPDATE);
                          /* Fenstererneuerung */
269:    wind_update(BEG_MCTRL);  /* Mauskontrolle */
270:    graf_mouse(FLAT_HAND,0); /* Maus auf
                                  "flache Hand" */
271:    graf_dragbox(tree[child].ob_width,
        tree[child].ob_height, /* bewegen */
272:                 ox,                  oy,
273:                 x,                   y,
274:                 tree[ROOT].ob_width,
                     tree[ROOT].ob_height,
275:                 &nx,                 &ny);
276:    graf_mouse(ARROW,0); /* Maus wieder Pfeil */
277:    wind_update(END_MCTRL); /* keine
                               Mauskontrolle */
278:    wind_update(END_UPDATE); /* Erneuerung
                                  beendet */
279:    tree[child].ob_x += nx - ox; /* Verschub
                                    addieren */
280:    tree[child].ob_y += ny - oy;
281:
282:    /* Objekt darstellen */                    →
```

```
283:    tree[child].ob_flags &= ~HIDETREE;
284:    draw_objc(handle,tree,child);
285:  }
286:
287:
288:  /*********************************************/
289:  /* Information ausgeben                      */
290:  /* Aufgabe: Dialogbox ausgeben               */
291:  /*********************************************/
292:
293:  static void do_info (void)
294:
295:  { GRECT a, b;
296:    int   ret;
297:
298:    /* Position des Menü-Titels berechnen */
299:    objc_offset(menu,MNINFO,&b.g_x,&b.g_y);
300:    b.g_w = menu[MNINFO].ob_width;
301:    b.g_h = menu[MNINFO].ob_height;
302:
303:    /* Vorbereitung */
304:    wind_update(BEG_UPDATE);
                          /* Fenstererneuerung */
305:    form_center(infodial,
306:                &a.g_x,&a.g_y,&a.g_w,&a.g_h);
                    /* Dialogbox zentrieren */
307:    form_dial(FMD_START,
308:              a.g_x,a.g_y,a.g_w,a.g_h,
309:              a.g_x,a.g_y,a.g_w,a.g_h);
                  /* Hintergrund reservieren */
310:    form_dial(FMD_GROW,
311:              b.g_x,b.g_y,b.g_w,b.g_h,
312:              a.g_x,a.g_y,a.g_w,a.g_h);
                  /* vergrösserndes Rechteck */
313:    objc_draw(infodial,ROOT,MAX_DEPTH,
314:              a.g_x,a.g_y,a.g_w,a.g_h);
                  /* Dialogbox ausgeben */
315:
316:    /* Durchführung */
317:    ret=form_do(infodial,0); /* Dialog
                                  durchführen */
318:    infodial[ret].ob_state &= ~SELECTED;
                    /* SELECTED zurücksetzen */
319:
320:    /* Nachbereitung */
321:    form_dial(FMD_SHRINK,
322:              b.g_x,b.g_y,b.g_w,b.g_h,
323:              a.g_x,a.g_y,a.g_w,a.g_h);
                  /* verkleinerndes Rechteck */
324:    form_dial(FMD_FINISH,
325:              a.g_x,a.g_y,a.g_w,a.g_h,
326:              a.g_x,a.g_y,a.g_w,a.g_h);
                  /* Hintergrund freigeben */
327:    wind_update(END_UPDATE); /* Erneuerung
                                  beendet */
328:
329:    /* Vorbereitung und Nachbereitung können auch
          in eigene Routinen  */
330:    /* ausgelagert werden. */
331:  }
332:
333:
334:  /*********************************************/
335:  /* Menü-Handler                              */
336:  /* Aufgabe: Auswertung des ausgewählten Menü-
              Eintrags                            */
337:  /*********************************************/
338:
339:  static void hdle_menu (int title, int item)
340:
341:  {
342:    /* Welches Drop-Down-Menü? */
343:    switch (title)
344:      {
345:        /* Info-Menü? */
346:        case MNINFO  :
347:        /* Hier gibt es nur einen Eintrag */
348:                      if (item == ITINFO)
349:                        do_info();
350:
351:                      break;
352:
353:        /* Datei-Menü? */
354:        case MNDATEI :
355:                      /* Welcher Eintrag? */
356:                      switch (item)
357:                      {
358:                        /* Datei neu anlegen?
                            (nicht laden!) */
359:                        case ITNEU    :
                          /* Eigene Routine... */
360:                         form_alert(1,
                                 (char*) laden);
361:                                 break;
362:
363:                        /* Datei öffnen? */
364:                        case ITOEFFNE :
                          /* Eigene Routine... */
365:
                           form_alert(1,
                                 (char*) laden);
366:                                 break;
367:
368:                        /* Datei sichern? */
369:                        case ITSICHER :
                          /* Eigene Routine... */
370:
                           form_alert(1,
                              (char*) speichern);
371:                                 break;
372:
373:                        /* Datei unter neuem
                            Namen sichern? */
374:                        case ITSICALS :
                          /* Eigene Routine... */
375:
                           form_alert(1,
                              (char*) speichern);
376:                                 break;
377:
378:                        /* Ausgabe-Programm
                              starten? */
379:                        case ITAUSGAB :
                          /* Eigene Routine... */
380:
                           form_alert(1,
                                (char*) ausgabe);
381:                                 break;
382:
383:                        /* Programm-Ende? */
384:                        case ITENDE   :
                                      finish = 1;
385:                                 break;
386:
387:                      }
388:      }
389:    menu_tnormal(menu,title,1);  /* Invertierung
                                      aufheben */
390:  }
391:
392:
393:  /*********************************************/
394:  /* Maus-Handler                              */
395:  /* Aufgabe: Auswertung des Mausklicks (linke
       Taste)                                     */
396:  /*********************************************/
397:
398:  static void hdle_mouse (int clicks, POINT *where)
399:
400:  { int obj,
401:        dummy,
402:        but_state;
403:
404:    /* Welches Object? */
405:    obj = objc_find(desktop,ROOT,MAX_DEPTH,
                        where->x,where->y);
406:
407:    /* Doppelklick? */
408:    if (clicks == 2)
409:      {
410:        /* Welches Objekt? */
411:        switch (obj)
412:          { case BSPICON : /* invers darstellen */
413:                           if (!(SELECTED &
                               desktop[obj].ob_state))
                                 {
414:                           desktop[obj].ob_state |=
                                       SELECTED;
415:                             draw_objc(DESK,
                                       desktop,
                                       obj);
416:                           }                    →
```

```
417:
418:                            /* Hier können eigene
                                 Routinen */
419:                            /* aufgerufen werden...
                                */
420:                            form_alert(1,(char*)
                                          aktion);
421:
422:                            /* normal darstellen */
423:                            desktop[obj].ob_state
                                          &= ~SELECTED;
424:                            draw_objc(DESK,desktop,
                                          obj);
425:                            /* Es könnte auch
                                 objc_change
                                 benutzt werden! */
426:
427:                            break;
428:
429:  /* Platz für weitere Auwertungen und Icons... */
430:
431:          }
432:      }
433:
434:    /* Einfachklick? */
435:    else if (clicks == 1)
436:      {
437:        /* Status der Maustasten abfragen */
438:        graf_mkstate(&dummy,&dummy,&but_state,
                        &dummy);
439:
440:        /* linke Maustaste festgehalten? */
441:        if (but_state & 0x01)
442:          {
443:            /* Welches Object? */
444:            switch (obj)
445:              {
446:                /* Objekt bewegen */
447:                case BSPICON : move_objc(DESK,
                                          desktop,obj);
448:                               break;
449:
450:                /* Für weitere Icons... */
451:
452:              }
453:          }
454:
455:        /* einfach geklickt? */
456:        else
457:          {
458:            /* Welches Object? */
459:            switch (obj)
460:              {
461:                /* Objekt invertieren */
462:                case BSPICON : if (!(SELECTED &
                                  desktop[obj].ob_state))
463:                               {
                                  desktop[obj].ob_state |= SELECTED;
464:                                draw_objc(DESK,
                                   desktop, obj);
465:                               }
466:                               break;
467:
468:                /* Platz für weitere Aktionen... */
469:
470:                /* Invertierungen aufheben */
471:                default      : if (SELECTED &
                                  desktop[BSPICON].ob_state)
472:                               {
                                  desktop[BSPICON].ob_state &= ~SELECTED;
473:                                draw_objc(DESK,
                                   desktop, BSPICON);
474:                               }
475:                               break;
476:
477:              }
478:
479:          }
480:      }
481:  }
482:
483:
484:  /*********************************************/
485:  /* Key-Handler                               */
486:  /* Aufgabe: Auswertung des Tastendrucks      */
487:  /*********************************************/
```

```
488:
489:  static void hdle_key (int key, int special_key)
490:
491:  {
492:    /* Control gedrückt? */
493:    if (K_CTRL & special_key)
494:
495:      {
496:        /* Welche Taste? */
497:        switch(key >> 8)    /* Scancode in den
                                   oberen 8 Bit */
498:          {
499:            /* N gedrückt? */
500:            case K_N : menu_tnormal(menu,MNDATEI,
                           0); /* Titel invertieren */
501:
502:                       /* Eigene Routine... */
503:                       form_alert(1,(char*) laden);
504:
505:                       menu_tnormal(menu,MNDATEI,
                                  1); /* Titel normal
                                         darstellen */
506:
507:                       break;
508:
509:            /* O gedrückt? */
510:            case K_O : menu_tnormal(menu,MNDATEI,
                            0); /* Titel invertieren */
511:
512:                       /* Eigene Routine... */
513:                       form_alert(1,(char*) laden);
514:
515:                       menu_tnormal(menu,MNDATEI,
                                  1); /* Titel normal
                                         darstellen */
516:
517:                       break;
518:
519:            /* S gedrückt? */
520:            case K_S : menu_tnormal(menu,MNDATEI,
                            0); /* Titel invertieren */
521:
522:                       /* Eigene Routine... */
523:                       form_alert(1,
                                    (char*)speichern);
524:
525:                       menu_tnormal(menu,MNDATEI,
                                  1); /* Titel normal
                                         darstellen */
526:
527:                       break;
528:
529:            /* A gedrückt? */
530:            case K_A : menu_tnormal(menu,MNDATEI,
                            0); /* Titel invertieren */
531:
532:                       /* Eigene Routine... */
533:                       form_alert(1,
                                    (char*)speichern);
534:
535:                       menu_tnormal(menu,MNDATEI,
                                  1); /* Titel normal
                                         darstellen */
536:
537:                       break;
538:
539:            /* Q gedrückt? */
540:            case K_Q : finish = 1;
541:                       break;
542:          }
543:      }
544:  }
545:
546:
547:  /*********************************************/
548:  /* Hauptprogramm                             */
549:  /* Aufgabe: Initialisierung, Auswertung der
                 Events, Abmeldung                */
550:  /*********************************************/
551:
552:  int main (void)
553:
554:  { int     events,         /* für Multi-Event */
555:            mouse_button,   /* für Multi-Event */
556:            sp_key,key,clicks, /* f.Multi-Event */
                                                  →
```

```
557:            msg_buffer[8]; /* für Multi-Event */
558:    POINT   mouse_pos;     /* für Multi-Event */
559:    int     ret_code;    /* aus Initialisierung */
560:
561:    if ((ret_code = init_prg()) == INIT_OK)
562:
563:      { while (!finish)
564:
565:          { events=evnt_multi(
566:            /* Message-, Maustasten- und
                  Tastatatur-Ereignis  */
567:                    MU_MESAG|MU_BUTTON|MU_KEYBD,
568:
569:            /* Doppelklick und linke Maustaste */
570:                    0x02,0x01,0x01,
571:
572:            /* Keine Überwachung von Rechtecken */
573:                    0,0,0,0,0,0,0,0,0,0,
574:
575:                    /* Message-Buffer */
576:                    msg_buffer,
577:
578:                    /* Kein Timer */
579:                    0,0,
580:
581:                  /* Maus-Position und -Taste */
582:                    &mouse_pos.x,
                        &mouse_pos.y,
                        &mouse_button,
583:
584:                  /* Sondertaste und Taste */
585:                    &sp_key,&key,
586:
587:                  /* Anzahl der Mausklicks */
588:                    &clicks);
589:
590:            /* Message-Event? */
591:            if (events & MU_MESAG)
592:
```

```
593:              { switch (msg_buffer[0])
594:                  {
595:                    /* Menü angeklickt? */
596:                    case MN_SELECTED :
                    hdle_menu(msg_buffer[3],msg_buffer[4]);
597:                                        break;
598:
599:  /* Platz für weitere Message-Auswertungen...   */
600:  /* Z.B. für Fenster: WM_REDRAW, WM_TOPPED, ... */
601:
602:                  }
603:              }
604:
605:            else
606:
607:              {
608:                /* Maus-Button-Event? */
609:                if (events & MU_BUTTON)
610:
611:                  hdle_mouse(clicks,&mouse_pos);
612:
613:                /* Tastatur-Event? */
614:                if (events & MU_KEYBD)
615:
616:                  hdle_key(key,sp_key);
617:
618:              }
619:          }
620:      }
621:
622:    else
623:      form_alert(1,no_init); /* Fehlermeldung */
624:
625:   exit_prg(ret_code);
626:
627:   return(0);
628:  }
629:  /*********************************************/
```


STEIGERN SIE IHRE ANSPRÜCHE

Arabesque

Anspruchsvolle Aufgaben erfordern entsprechende Werkzeuge, die Ihre Kreativität fördern, statt Sie einzuengen. Arabesque bietet Ihnen die Möglichkeit, sowohl mit Raster- als auch mit Vektorgrafiken zu arbeiten. Oder beides zu vereinen.

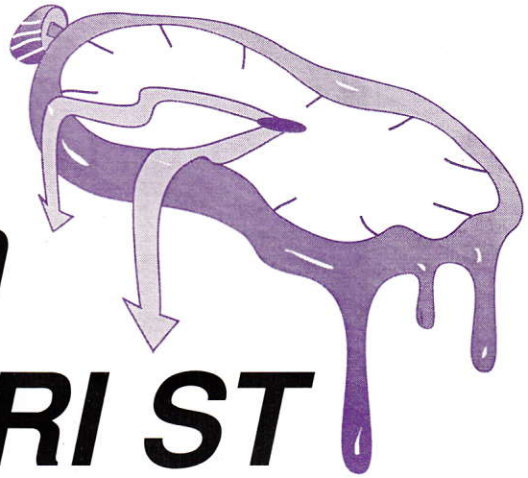
Arabesque ist die professionelle Lösung für den Atari ST. Einfach zu bedienen – und dennoch mächtig. Die richtige Software für anspruchsvolle Gestaltungsaufgaben – zu einem fairen Preis. Überzeugen Sie sich bei Ihrem Fachhändler, rufen Sie uns an, oder schreiben Sie uns. Wir informieren Sie gerne.

Nebenbei… Sollten Sie zum Kreis der Grafiker, Textverarbeiter und Schreibtisch-Publizisten gehören, wird es Sie interessieren, daß Arabesque alle wichtigen Grafikformate unterstützt. Es ermöglicht sogar Vektorgrafiken in Programmen wie 1ST Word Plus und Signum! Zwei.

Arabesque wird mit einem leichtverständlichen Handbuch im stabilen Schuber geliefert und kostet (unverb. Preisempfehlung) 278,– DM. Service inclusive.

shift

SHIFT SONNENSCHEIN & HANSEN · UNTERER LAUTRUPWEG 8 · D-2390 FLENSBURG · TELEFON (0461) 2 28 28
SCHWEIZ: EDV-DIENSTLEISTUNGEN · STIFTUNG GRÜNAU · ERLENSTRASSE 73 · CH-8805 RICHTERSWIL · TELEFON (01) 784 89 47
Grafik: *Arabesque* ★ Satz und Film: Calamus



Wartezyklen beim ATARI ST

In einigen Zeitschriften konnte man über den ST lesen, daß der 68000-Prozessor (CPU) beim Zugriff auf den Speicher nicht durch *WAIT*-Zyklen gebremst wird und die Buszyklen mit voller Geschwindigkeit durchgeführt werden. Dies ist aber leider nicht ganz richtig. Es wird zwar nicht jeder Buszyklus mit *WAITs* verlangsamt, und der eigentliche Zugriff auf den Bus dauert nur vier Taktzyklen, aber bei einigen Buszyklen wird der Zugriff auf den Bus durch zwei *WAITs* verzögert.

Ursprünglich wollte ich ein kleines Programm zur Messung der Laufzeiten einzelner Befehle schreiben, um die in verschiedenen Handbüchern zum 68000 angegebenen Zeiten zu überprüfen. Denn erstens gaben verschiedene Bücher verschiedene Zeiten für gleiche Befehle an, und zweitens schienen viele Zeiten fehlerhaft und nicht plausibel. Also schrieb ich ein kurzes C-Programm mit Inline-Assembly, das einen Timer startet, dann 50mal den zu testenden Befehl ausführt und den Timer ausliest.

Ein kurzer Test mit Befehlsfolgen, die aus einem bis fünf *NOPs* bestanden, zeigte, daß das Programm richtig arbeitete. Beim Testen von etwa 15-20 weiteren Befehlen (hauptsächlich *MOVE*- und *CLR*-Befehle) stimmten zu meiner Verwunderung die meisten gemessenen Zeiten nicht mehr mit denen in den Dokumentationen überein. Trotzdem überprüfte ich weitere Befehle und stieß auf immer merkwürdigere Ergebnisse, die sich immer schwerer erklären ließen.

Bei den Messungen zeigte sich nämlich, daß Befehlsfolgen, in denen sich nur die Reihenfolge der Befehle unterschied, nicht die gleiche Ausführungszeit brauchen. Beispielsweise braucht die Schleife

```
loop: nop
      clr.l   D0
      dbra    D7,loop
```

nur 20 Taktzyklen pro Durchlauf,

```
loop: clr.l   D0
      nop
      dbra    D7,loop
```

dagegen benötigt 24 Takte.

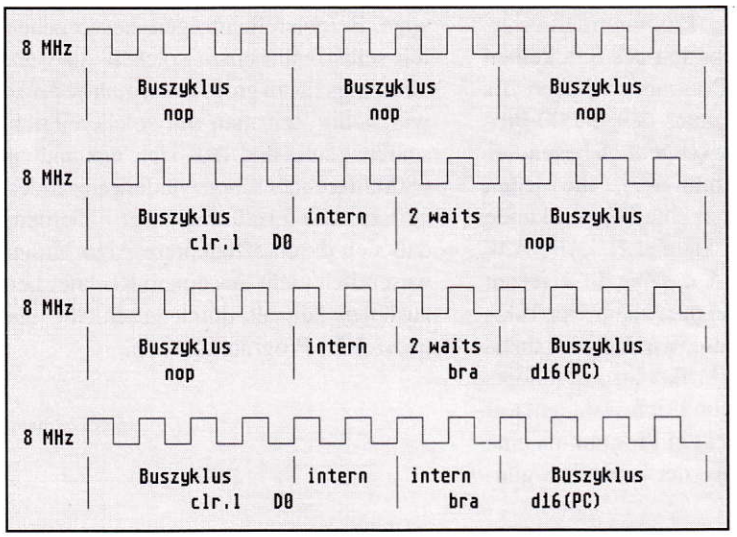
Zuerst überlegte ich mir verschiedene Modelle, mit denen ich dieses Verhalten der CPU zu erklären versuchte. Aber erstens gefiel mir keines dieser Modelle so richtig, und zweitens konnte keines die gemessenen Zeiten vollständig erklären.

Obwohl es laut Motorola viele Befehle gibt, die in 6,10,14... Takten ausgeführt werden, z.B. *clr.l Dn* (6 Takte) und *dbra* (10 bzw. 14 Takte), sind die Ausführungszeiten aller Befehlsfolgen, die das Programm ausgibt, stets Vielfache von vier Taktzyklen.

Das brachte mich auf die Idee, daß die Buszugriffe der CPU möglicherweise mit einem 2-MHz-Takt, also einem Viertel des 8-MHz-Taktes der CPU, synchronisiert werden, und daß daher alle Zeiten Vielfache von vier CPU-Takten sind. Der Grund für solche Wartezyklen ist bei vielen Computern (z.B. auch beim C64), daß das RAM, in dem der Bildschirminhalt gespeichert wird, nicht vom Speicher des Prozessors getrennt ist. Daher müssen sich der Prozessor und der Video-Chip das RAM teilen und können nicht gleichzeitig den Inhalt auslesen oder beschreiben. Aus der Bildfrequenz von 71Hz im Monochrommodus und 32 kByte Bildschirmspeicher kann man abschätzen, wenn man die Zeit für den horizontalen und vertikalen Strahlrücklauf sowie die schwarzen Ränder des Bildes berücksichtigt, daß der Video-Shifter des ST tatsächlich mit ungefähr 2 MHz auf das RAM zugreifen - und damit die CPU bremsen - muß, um das Bild darzustellen. Also versprach dieser Ansatz eine Erklärung der erwähnten Effekte.

Numeriert man die Takte mit 0 beginnend, so bedeutet das, daß die Buszugriffe der CPU nur bei Taktnummern beginnen können, die ein Vielfaches von vier sind. Will die CPU aber einen Zugriff im Takt 4n+2, n>=0 durchführen, muß sie zwei Takte warten und kann den Bus erst im Takt 4n+4 benutzen.

Auf diese Weise läßt sich dann auch erklären, warum die Ausführungszeiten einiger Befehlsfolgen von der Reihenfolge der Befehle abhängen. Es gibt nämlich Befehle, die nach ihrem letzten Buszyklus noch 2 Takte intern arbeiten (z.B. *clr.l Dn*). Der darauffolgende Befehl wird also in einem Takt mit der Nummer 4n+2 begonnen. Wenn dieser gleich mit einem Buszyklus anfängt (z.B. nop), muß die CPU erst zwei Takte warten, bevor sie auf den Bus zugreifen darf, weil sie sonst den Video-Shifter beim Bildaufbau stören würde. Wenn aber statt eines nop ein Befehl folgt, der vor seinem ersten Bus



zugriff zwei Takte verstreichen läßt (z.B. *dbra* d_{16} oder *bra* d_{16}, also Branch-Befehle mit 16-Bit-Distanz), fällt dieser Zugriff auf den Takt 4n+4 (also ein Vielfaches von vier), und die CPU muß keine WAIT-Zyklen einfügen. Das Bild zeigt das Timing noch einmal grafisch auf. Aus Gründen der Übersichtlichkeit habe ich die WAIT-Zyklen immer vor dem Buszyklus, in dem sie auftreten, eingezeichnet. In Wirklichkeit werden diese Zyklen in den Buszyklus eingebaut, aber noch vor dem eigentlichen Zugriff auf den Bus.

In Listing 1 ist ein kurzes Programm in Megamax-C angegeben, mit dem einzelne Befehle schnell auf ihr Timing untersucht werden können. Der zu testende Befehl wird in drei verschiedenen Schleifen ausgeführt. In der ersten Schleife folgt der Befehl immer einem nop, beginnt also synchron zum 2-MHz-Takt (also mit einer Taktnummer 4n), denn der *nop*-Befehl endet sofort mit dem Abschluß des Buszyklus'. In der zweiten Schleife steht an Stelle des nop ein *clr.l D0*, der zwei Takte länger dauert als ein *nop*. Der zu testende Befehl beginnt also in der Mitte eines 2-MHz-Taktes (Taktnummer 4n+2). Wenn der zu testende Befehl sofort mit einem Buszyklus beginnt, kann dieser in der ersten Schleife sofort ausgeführt werden, während in der zweiten Schleife zwei zusätzliche *WAITs* eingefügt werden, so daß der Schleifendurchlauf im zweiten Fall vier Takte länger dauert. Bei Testbefehlen, die ihren ersten Buszyklus erst nach zwei Takten beginnen, sind die Ausführungszeiten der beiden Schleifen gleich. In der ersten Schleife müssen nach diesen ersten zwei Takten noch zwei zusätzliche *WAITs* eingefügt werden, und der Zugriff beginnt im Takt 4n+4. In der zweiten Schleife beginnt der Befehl erst im Takt 4n+2 und arbeitet zuerst zwei Takte intern. Danach kann der Buszyklus sofort im Takt 4n+4 ausgeführt werden.

Entsprechend kann man an der Differenz der Laufzeiten für die erste und die dritte Schleife ablesen, ob die CPU nach dem letzten Buszyklus noch 2,6,10,... oder 0,4,8,... Takte intern arbeitet, bevor der Befehl beendet und der nächste begonnen wird.

Bevor das Assemblerprogramm zur eigentlichen Zeitmessung aufgerufen wird, übergibt das Hauptprogramm in den globalen Variablen *ptr* und *iter* einen Zeiger auf einen 8 kByte großen Speicherblock und die Anzahl der Schleifendurchläufe. Der Speicher wird nur gebraucht, falls man Befehle wie *move ea,-(A0)* oder *move ea,(A0)+* testen will, denn irgendwohin muß das Adreßregister ja zeigen. Zuerst zeigt das Adreßregister in die Mitte dieses Speichers und kann dann bei jedem Schleifendurchlauf erhöht oder erniedrigt werden. So wird verhindert, daß es einen zufälligen Wert enthält und beim Schreiben der Rechner abstürzt. Die gemessenen Zeiten werden in dem globalen Array *ticks[3]* zurückgegeben und in die Anzahl der Takte umgerechnet. Dazu dient die Konstante *CLKF*. Sie setzt sich zusammen aus den 8 MHz des CPU-Taktes und den 2.4576 MHz, die am MFP 68901 anliegen und vom Vorteiler des Timers noch durch 4 geteilt werden. Für eine leere Schleife, die nur aus dem dbra-Befehl besteht, mißt das Programm 11.98 Takte, die deshalb noch subtrahiert werden. Warum die leere Schleife 12 Takte und nicht, wie es laut 68000-Dokumentation sein müßte, 10 Takte dauert, kann sich nun jeder selbst überlegen. Einige interessante Ergebnisse kann man Tabelle 1 entnehmen.

Will man also die Laufzeit eines Programms auf dem ST berechnen, muß man nicht nur wissen, wieviele Takte jeder einzelne Befehl dauert, sondern auch, wie die Buszyklen innerhalb des Befehls verteilt sind und wo *WAITs* auftreten. Dabei ist nicht nur interessant, wieviele Takte die CPU vor dem ersten bzw. nach dem letzten Buszugriff eines Befehls nur intern arbeitet. Bei *MOVE (A0),d8(A1,A2)* braucht die CPU zwischen zwei Buszugriffen zwei Takte, um die zweite, relativ komplizierte Adresse zu berechnen. Diese zwei Takte werden auf dem ST aber aus den genannten Gründen immer auf

nop		(0,4,0)	
clr.w	Dn	(0,4,0)	
clr.l	Dn	(0,4,2)	
scc	Dn	(0,4,2)	falls Bedingung erfüllt
scc	Dn	(0,4,0)	falls Bedingung nicht erfüllt
bcc	d_{16}	(2,8,0)	falls Bedingung erfüllt
bcc.s	d_8	(2,8,0)	falls Bedingung erfüllt
bcc	d_{16}	(4,8,0)	falls Bedingung nicht erfüllt
bcc.s	d_8	(4,4,0)	falls Bedingung nicht erfüllt
dbcc	Dn,d_{16}	(4,8,0)	falls Bedingung erfüllt
dbcc	Dn,d_{16}	(6,8,0)	falls Sprung nicht ausgeführt
dbcc	Dn,d_{16}	(2,8,0)	falls Sprung ausgeführt
bsr	d_{16}	(2,16,0)	
rts		(0,16,0)	
asr.w	#m,Dn	(0,4,2+2m)	
asr.l	#m,Dn	(0,4,4+2m)	
move.x	-(An),ea	(2,x,0)	hängt von ea ab
move.x	Dn,d_8(An,Rm)	(2,x,0)	
nop		(0,4,0)	
clr.w	Dn	(0,4,0)	
clr.l	Dn	(0,4,2)	
scc	Dn	(0,4,2)	falls Bedingung erfüllt
scc	Dn	(0,4,0)	falls Bedingung nicht erfüllt
bcc	d_{16}	(2,8,0)	falls Bedingung erfüllt
bcc.s	d_8	(2,8,0)	falls Bedingung erfüllt
bcc	d_{16}	(4,8,0)	falls Bedingung nicht erfüllt
bcc.s	d_8	(4,4,0)	falls Bedingung nicht erfüllt
dbcc	Dn,d_{16}	(4,8,0)	falls Bedingung erfüllt
dbcc	Dn,d_{16}	(6,8,0)	falls Sprung nicht ausgeführt
dbcc	Dn,d_{16}	(2,8,0)	falls Sprung ausgeführt
bsr	d_{16}	(2,16,0)	
rts		(0,16,0)	
asr.w	#m,Dn	(0,4,2+2m)	
asr.l	#m,Dn	(0,4,4+2m)	
move.x	-(An),ea	(2,x,0)	hängt von ea ab
move.x	Dn,d_8(An,Rm)	(2,x,0)	

Tabelle 1: Einige interessante Erkenntnisse über Buszyklen. Die drei Zahlen in den Klammern bedeuten jeweils die Anzahl der Takte vor dem ersten Buszyklus, vom Beginn des ersten bis zum Ende des letzten Buszyklus' und die Anzahl der Takte nach dem letzten Buszyklus.

vier Takte ausgedehnt, gleichgültig welche Befehle vor oder nach dem *MOVE* stehen.

Hier sei noch erwähnt, daß alles, was ich über das Timing geschrieben habe, erstmal nur eine Hypothese ist, die nur auf den Ergebnissen des vorgestellten Programms basiert. Leider habe ich kein Oszilloskop, um das Timing genauer zu untersuchen und diese Hypothese zu stützen. Es lassen sich damit jedoch die Ausführungszeiten aller Befehle, die ich bisher getestet habe, so gut erklären, daß sie mit ziemlicher Wahrscheinlichkeit richtig ist.

Für viele Assembler-Programmierer werden diese Eigenschaften des STs keinen Einfluß auf ihre Programme haben. Es gibt jedoch auch unter den 68000-Programmierern Leute (angeblich besonders die Grafikprogrammierer), die jeden Trick anwenden, um eine Mikrosekunde zu sparen. Man kann z.B. *ADDA.W* $\#d_{16}$,*An* durch *LEA* d_{16}(*An*),*An* ersetzen und spart dann zwei oder sogar vier Taktzyklen. Solche Geschwindigkeitsfanatiker sollten diesen Artikel also besonders aufmerksam lesen und sich vielleicht mit Hilfe des abgedruckten Programms eine Tabelle anlegen, aus der besonders günstige Befehlsreihenfolgen hervorgehen. Ich selbst halte einen solchen Aufwand allerdings für zu groß, wenn man bedenkt, wie wenig Zeit man mit solchen Tricks sparen kann, und daß sich ein anderer 68000-Rechner schon wieder ganz anders verhält. Die Erfahrung zeigt außerdem, daß sich durch effizientere Algorhitmen wesentlich mehr aus einem Rechner herausholen läßt als durch geschickte und trickreiche Programmierung.

Urs Thürmann

```
 1:  #include <stdio.h>
 2:  #include <osbind.h>
 3:
 4:  #define CLKF (8e6/(2.4576e6/4.0))
 5:  #define SIZE 8192L
 6:
 7:  #define INSTR clr.l D0    /* Hier steht der zu
                                  testende Befehl  */
 8:
 9:  extern gettim();
10:  char *ptr;                 /* Zeiger auf 8kB RAM */
11:  int iter;                  /* Anzahl der
                                  Schleifendurchläufe */
12:  int ticks[3];              /* gemessene
                                  Ausführungszeiten */
13:                            /* in MFP-Takten
                                  (0.6144 MHz) */
14:  main()
15:  {
16:       double time[3];      /* Ausführungszeiten
                                  in Sekunden */
17:       int i;
18:
19:       ptr = (char*)Malloc(SIZE);  /* pointer auf 8k
                                         RAM */
20:       iter = 50;        /* 50 Schleifendurchläufe */
21:       Supexec(gettim); /* Routine muP im
                              Supervisor-Mode laufen */
22:       printf("Die Anzahl der Takte pro
                Schleifendurchlauf war\n");
23:       for(i = 0; i < 3; i++)
24:       {    time[i] = ticks[i] * CLKF;    /* Zeit in
                                                 Sekunden berechnen */
25:            printf("in der %d. Schleife: %.01f\n",i,
                      time[i] / iter - 11.98);
26:       }
27:       Cnecin();              /* auf Taste warten */
28:       Mfree(ptr);
29:  }
30:
31:  asm
32:  {
33:  gettim:
34:          move    SR,D6
35:          ori     #0x0F00,SR  ; Interrupts sperren
36:          move.b  #0,0xFA19   ; Timer initialisieren
37:
38:          bsr     getpar
39:          move.b  #255,0xFA1F  ; Startwert 255
40:          move.b  #1,0xFA19    ; und Countdown
                                    starten
41:
42:  loop1:
43:          nop
44:
45:          INSTR               ; der zu testende Befehl
46:
47:          nop
48:          nop
49:          dbf     D7,loop1
50:
51:          move.b  #0,0xFA19        ; Timer anhalten
52:          move.b  0xFA1F,D1
53:          move.b  #255,D0
54:          sub.b   D1,D0
55:          and.w   #0xFF,D0
56:          move.w  D0,ticks(A4)
57:
58:          bsr     getpar
59:          move.b  #255,0xFA1F      ; Startwert 255
60:          move.b  #1,0xFA19        ; und Countdown
                                        starten
61:
62:  loop2:
63:          clr.l   D0
64:
65:          INSTR               ; der zu testende Befehl
66:
67:          nop
68:          nop
69:          dbf     D7,loop2
70:
71:          move.b  #0,0xFA19        ; Timer anhalten
72:          move.b  0xFA1F,D1
73:          move.b  #255,D0
74:          sub.b   D1,D0
75:          and.w   #0xFF,D0
76:          move.w  D0,ticks+2(A4)
77:
78:          bsr     getpar
79:          move.b  #255,0xFA1F      ; Startwert 255
80:          move.b  #1,0xFA19        ; und Countdown
                                        starten
81:
82:  loop3:
83:          nop
84:
85:          INSTR               ; der zu testende Befehl
86:
87:          bra     lab
88:          nop
89:  lab:    dbf     D7,loop3
90:
91:          move.b  #0,0xFA19        ; Timer anhalten
92:          move.b  0xFA1F,D1
93:          move.b  #255,D0
94:          sub.b   D1,D0
95:          and.w   #0xFF,D0
96:          move.w  D0,ticks+4(A4)
97:          move    D6,SR
98:
99:          rts
100:
101: getpar:
102:         move.l  ptr(A4),A0   ; Zeiger auf den 8
                                    kByte-Block nach A0
103:         adda    #SIZE/2,A0   ; für   move ea,-(An)
104:         move.l  A0,A1        ; oder  move ea,(An)+
                                          Befehle u.ä.
105:         move.w  iter(A4),D7  ; Anzahl der
                                    Schleifendurchläufe
106:         subq.w  #1,D7        ; nach D7
107:         rts
108: }
```

Somewhere over the Rainbow

Intimes zu TOS 1.4



Nun ist es endlich soweit - die von vielen lang erwartete neue Version unseres Lieblings-Betriebssystems TOS ist fertig. Das neue TOS 1.4 wurde auf der Düsseldorfer ATARI-Messe offiziell vorgestellt und endgültig freigegeben. Dies ist Grund genug, es auch unseren Lesern vorzustellen.

TOS 1.4 hört auch auf den Decknamen "Rainbow TOS", da das ATARI-Zeichen in der "Desktop info"-Dialogbox in einer farbigen Animation (nur im Farbmodus) zu sehen ist, die wohl an einen Regenbogen erinnern soll (engl. "rainbow" = Regenbogen). Diese offizielle Version datiert vom 6.4.1989 und ist damit zu unterscheiden von den "Beta-Versionen" des Jahres 1988 und dem "Messe-TOS" vom 22.2.1989, das auf der letzten CeBIT zu sehen war.

Erfreulicherweise gibt es für Software-Entwickler auch eine Dokumentation, in der Änderungen, Fehler und besondere Programmierhinweise aufgeführt sind. Diese Doku ist zwar nicht ganz vollständig und fehlerfrei, aber bietet wesentlich mehr, als man sonst von ATARI gewohnt ist. Überhaupt ist die Unterstützung der Software-Entwickler besser geworden, als sie es früher einmal war. Trotzdem konnte ich es nicht lassen, mir das TOS (speziell BIOS und GEMDOS) ein wenig unter der Lupe anzusehen, sprich zu disassemblieren und recompilieren.

Heute möchte ich Ihnen die Ergebnisse bezüglich des BIOS mitteilen, und zwar in drei Stufen: für Anwender, Programmierer und die TOS-Spezialisten, die ihren Computer bis in die letzten Eingeweide kennen möchten. In den nächsten Ausgaben der ST Computer beschäftige ich mich dann intensiv mit den Neuerungen beim GEMDOS, in Anlehnung und als Ergänzung zur GEMDOS-Serie "Auf der Schwelle zum Licht" (ST 1/88-11/88). Auf die für Nicht-Programmierer interessanteren Verbesserungen am Desktop wird in dieser Zeitschrift an anderer Stelle eingegangen.

Für den Anwender

Kommen wir also zunächst zu den Änderungen, die für den Anwender interessant sind. Bekanntlich gibt es die sogenannten boot-fähigen Disketten, d.h. Disketten, von denen beim Reset automatisch ein kleines Programm geladen wird, das allerlei nützliche Dinge tun kann. Dazu zählen z.B. der "TOS-Loader" von ATARI, mit dem ein RAM-TOS von Diskette gebootet werden kann, Programme, die einem das Einstellen der Systemuhr erlauben oder Hinweise über die Diskette ausgeben (Aladin). Es gibt vor allem bei Spielen sogar Boot-Sektor-Programme, die gleich das eigentliche Programm starten (Auto-Start-Disketten). Am verbreitetsten unter den Boot-Programmen sind aber wohl die "Boot-Sektor-Viren", über die wohl schon genug geschrieben wurde.

Das Laden und Starten dieser Boot-Programme geschah aber bisher nur beim Kaltstart (d.h. nach dem Einschalten des Rechners) oder wenn keine weiteren boot-fähigen Laufwerke vorhanden waren (Harddisk, EPROM-Disk, reset-residente RAM-Disk). Dies ist nun anders, da jetzt bei jedem Reset auf jeden Fall versucht wird, ein Boot-Sektor-Programm von Laufwerk A: zu laden. Dies vereinfacht die Benutzung der oben erwähnten Auto-Start-Disketten, erhöht aber auch die Virengefahr. Ferner sollte man nun immer eine Diskette in A: haben, wenn man nicht von Diskette bootet, da der Boot-Vorgang sonst unnötig lange dauert (wegen des versuchten Zugriffs auf eine nicht vorhandene Diskette).

Beim Booten von Harddisk (oder anderen DMA-Geräten) wird nun beim Lesen des Boot-Sektors ein zweiter Versuch gemacht, falls beim ersten ein Fehler auftrat. Ausgenommen ist der Fall, daß keine Harddisk angeschlossen ist; dann wird weiterhin nur ein Leseversuch gemacht.

Eine wichtige und schwierige Aufgabe des TOS ist die sichere Erkennung von Diskettenwechseln, da sonst sehr leicht Daten auf Disketten zerstört werden können. Bei Benutzung eines physikalischen Laufwerks als zwei logische Laufwerke, was zur bekannten häufigen Aufforderung "Bitte Disk X: in Floppy A: einlegen" führt, konnte es passieren, daß ein Diskettenwechsel nicht erkannt wurde, wenn er und die Bestätigung dieser Aufforderung sehr schnell (unter 1,5 Sekun-

den) erfolgten. Dieser Mangel ist nun behoben.

Auch an der Tastaturbehandlung wurde einiges geändert. So kann ein Reset nun endlich durch Tastendruck, nämlich durch CONTROL-ALTERNATE-DELETE, ausgelöst werden. Dabei wird ein Warmstart durchgeführt, d.h. reset-residente RAM-Disks usw. bleiben erhalten. Auch ein Kaltstart ist möglich, wenn zusätzlich noch die *rechte* SHIFT-Taste gedrückt wird. Dabei wird fast der gesamte Speicher gelöscht, so daß dies vorher gut überlegt sein will.

Die Tastenwiederholung wurde leicht modifiziert. Wenn man eine Taste gedrückt hielt (z.B. 'a'), bis die Tastenwiederholung ansprach, und dann eine weitere betätigte (z.B. 's'), erschien nur ein 's', und die Tastenwiederholung blokkierte, bis mindestens eine der Tasten losgelassen und eine weitere gedrückt wurde. Nun wird auch das 's' automatisch wiederholt, bis eine weitere Taste gedrückt wird, usw.

Für den Programmierer

Auch für den Programmierer gibt es einige Neuigkeiten. Da wären zunächst einmal eine geänderte und eine neue XBIOS-Funktion. Die neue XBIOS-Funktion *Floprate* (Nummer 41) erlaubt das Ermitteln und Verstellen der Floppy-Seek-Rate (Spurwechselzeit), und zwar für jedes Laufwerk einzeln. Dies ging bisher nur mit bestenfalls halblegalen Methoden (s. [1], Kapitel 8.2.). Warum ATARI die XBIOS-Nummer 40 unbenutzt gelassen hat, bleibt wohl ein Geheimnis. Die Deklaration in C sieht so aus:

```
long Floprate(int drive, int rate)
```

Für die Nicht-C-Programmierer: Die Funktion hat zwei 16-Bit-Integer-Parameter und liefert einen 32-Bit-Integer-Wert zurück. Der erste Parameter *drive* gibt das Laufwerk an (0 oder 1, andere Werte sind zu vermeiden), der zweite die neue Seek-Rate. *rate* ist dabei genauso wie die Systemvariable i ($440) kodiert. Werte von 0 bis 3 stehen dabei für Seek-Raten von 6, 12, 2 bzw. 3 Millisekunden.

Die XBIOS-Funktion *Rsconf* wurde erweitert und besser dokumentiert. Übergibt man als ersten Parameter (Baud-Rate) den Wert -2, liefert die Funktion die mit *Rsconf* eingestellte Baud-Rate zurück. Wenn der MFP-Chip durch direkten Zugriff programmiert wird, bemerkt dies

Übersicht über das BIOS des TOS 1.4

```
; L=siehe Listing, T=siehe Text
; sonst inhaltlich unverändert

fc0000 L system header block
fc0030   reset, Kaltstart
fc01f2 L Warmstart
fc0530   von Floppy booten
fc054c   von DMA booten
fc0578 L Boot-Sektor von DMA-Bus laden
fc060e   DMA-Kommandobyte ausgeben
fc0628   ROM-Modul testen und init.
fc0652   Dummy-RTS-Routine
fc0654   RAM-Lesetest
fc066a   Speicherkonfigurationstest
fc068a   Standard-BIOS-Farbpalette
fc06aa   HBL-Handler
fc06c0 L VBL-Handler
fc07a2   XBIOS-Vsync
fc07ba   BIOS-Critical-Error-Handler
fc07c4   TRAP #14/TRAP #13-Handler
fc0950   XBIOS-Supexec
fc0956   BIOS-Bcon...
fc09f8   BIOS-Drvmap
fc09fe   BIOS-Kbshift
fc0a10   BIOS-Getmpb
fc0a3c   BIOS-Setexc
fc0a54   BIOS-Tickcal
fc0a5c   XBIOS-Physbase/Logbase/Getrez/Setscreen
fc0ac4   XBIOS-Setpalette/Setcolor
fc0af0   XBIOS-Puntaes
fc0b0a   Bömbchen-Routinen
fc0bb6   Sektor kopieren
fc0bd8   Laufwerke init. (Sprung über hdv_init)
fc0bf4   Auto-Ordner-Programme ausführen
fc0efa   XBIOS-Scrdmp
fc0d0c   über 'dumpvec' aufgerufene Hardcopy-Routine
fc0dc2   Timer-B-Zählroutine
fc0dfe   reset-residente "Päckchen"-Programme suchen und ausführen
fc0e3e   XBIOS-Gettime/Settime
fc0e62   Kopie des System-header-blocks anlegen
fc0e9e   XBIOS-Blitmode
fc0ec6   Test, ob Blitter vorhanden


Floppy-Routinen
fc0ef0   Laufwerk init. (flopini)
fc0f38   XBIOS-Floprd
fc0fe8   Fehlernummer aus FDC-Fehler bestimmen
fc100a   XBIOS-Flopwr
fc10c6   XBIOS-Flopfmt
fc1286   XBIOS-Flopver
fc1360   Floppy-VBL-Handler
fc13dc   floplock
fc145c   flopfail
fc146a   flopok
fc14b0   Spur ansteuern
fc14ce   Restore und Seek
fc1502   Spur suchen (go2track)
fc1528   Restore
fc153e   FDC-Kommando schicken
fc1582   FDC-Reset
fc159c   Laufwerk und Seite selektieren (2 Routinen)
fc16d8   Kommando-Bytes an FDC schicken
fc1640 L Test auf Diskettenwechsel für 'Bitte Disk einlegen...'
fc1682   DSB neu setzen
fc1692 L *** XBIOS-Floprate: Seek-Rate ermitteln/neu setzen ***
fc16ba T Floppy-hdv_init
fc1732   BIOS-Getdsb
fc173c   BIOS-Getbpb
fc18ec T BIOS-Mediach
fc1956   interne Diskettenwechsel-Routine
fc1a24   BIOS-Rwabs
fc1aac   floprw
fc1c76   XBIOS-Random
fc1cc6 L Floppy-hdv_boot
fc1d42 T XBIOS-Protobt
fc1e2e   Checksumme für Boot-Sektor berechnen
fc1e5e   Intel- nach Motorola-Format wandeln
```

→

Rsconf nicht und liefert einen falschen Wert. Die restlichen Parameter sind bei diesem Aufruf beliebig.

Rsconf hat laut ATARI-XBIOS-Dokumentation keinen Rückgabewert, aber schon immer wurden die MFP-Register UCR, TSR, RSR und UDR vor dem *Rsconf* zurückgeliefert (außer bei einer "Baud-Rate" von "-2"). Dabei sind alle vier Register in einem Langwort zusammengepackt (UCR im höchstwertigen Byte, usw.). Die Rainbow-TOS-Dokumentation segnet diese nun ab, doch spricht sie dabei von den Registern SCR, UCR, RSR und TSR. Diese Bezeichnungen wären auch sinnvoller, da dies die Register sind, die man mit *Rsconf* auch setzen kann.

Protobt soll "IBM-kompatiblere" Boot-Sektoren erzeugen. Die einzige Änderung, die ich feststellen konnte, ist ein verändertes *Media-Byte* im Boot-Sektor bei einseitigen 80-Track-Disketten. Statt $F8 steht hier nunmehr $F9, obwohl eigentlich unter PC-DOS $F8 vorgesehen ist ([1]). Da ich zu wenig von PC-DOS verstehe, kann ich nicht sagen, ob das ein neuer Fehler oder tatsächlich eine Verbesserung ist. Um volle MS-DOS-Kompatibilität zu erreichen, sollen die ersten drei Bytes im Boot-Sektor $E9,$00,$4E sein. Der eigentliche Fortschritt besteht aber in einer verbesserten Formatierroutine des Desktops, die das Media-Byte in der FAT und die ersten drei Bytes im Boot-Sektor richtig setzt.

Die "Bömbchen"-Routine, die unerwartete Exceptions "behandelt", terminiert das aktive Programm nun mit *Pterm(-1)* statt mit *Pterm(0)*, so daß das übergeordnete Programm 0x0000ffff von *Pexec* zurückbekommt.

Eigene Tastatur-Interrupt-Routinen standen immer vor dem Problem, "normale" Tastendrücke von aus mehreren Bytes bestehenden "Paketen" (z.B. Maus-Bewegungs-Paket) unterscheiden zu müssen. Am Offset $24 ab dem von *Kbdvbase* zurückgelieferten Zeiger fand sich in allen TOS-Versionen ein Byte-Flag, das anzeigt, ob gerade ein aus mehreren Zeichen bestehendes "Paket" vom Tastaturprozessor erwartet wird. Ab nun ist seine Lage (dieser Offset) offiziell dokumentiert.

Eigene Tastatur-Interrupt-Routinen, die sich in den *ikbdsys*-Vektor einklinken und Tastendrücke einer "Sonderbehandlung" unterziehen wollen, sollten Bytes nur berücksichtigen, wenn das Flag Null ist.

```
Diverses
fc1e84    Hardcopy-Ausgaberoutinen (4 Stück)
fc1ee0    superschnelle Löschroutine
fc1f4c    MEGA-Uhr nach GEMDOS-Uhr
fc1f70 L  Test, ob MEGA-Uhr vorhanden
fc1fc2    Gettime für MEGA-Uhr
fc2080    Settime für MEGA-Uhr
fc215c    Hardcopy


IKBD, RS232, Centronics, MFP
fc315e    Uhrzeit-Paket in BIOS-Format umrechnen
fc31a8 L  IKBD-Gettime
fc31d2    IKBD-Settime
fc3240    Byte -> BCD
fc3254    BCD -> Byte
fc326a    MIDI-I/O-Routinen
fc32f6 L  Centronics-I/O-Routinen
fc33a6    RS232-Routinen (Ein-/Ausgabe und Status)
fc344a    IKBD-Routinen (Ausgabe und Status)
fc3480    XBIOS-Ikbdws
fc3494    Tastatur-Routinen (Eingabe, Status, Ausgabestatus)
fc34e4    "Glocke" starten
fc34fc    MFP init.
fc365e    Default-IOREC-Daten und RS232-Interrupt-Vektoren
fc36ac    MFP-Timer init.
fc3754    XBIOS-Mfpint
fc377e    XBIOS-Jdisint/Jenabint
fc37f2    RS232-Interrupt-Routinen (4 Stück)
fc3932    weitere RS232-Routinen
fc39fe    XBIOS-Iorec
fc3a16 L  XBIOS-Rsconf
fc3aec T  IKBD/MIDI-Interrupt
fc3c70 L  Keyboard-Interrupt
fc3fbe    MIDI-Eingabe-Interrupt
fc3fec    Maus-Emulator
fc4034 L  *** Shift-Ctrl-Alt-Routine ***
fc407e    XBIOS-Giaccess/Ongibit/Offgibit
fc4102    XBIOS-Initmous
fc41cc    XBIOS-Xbtimer
fc4206    XBIOS-Keytbl/Bioskeys
fc424c    XBIOS-Dosound
fc4260    XBIOS-Setprt
fc4272    XBIOS-Kbrate/Kbdvbase
fc429c    200 Hz-Interrupt-Handler
fc42fe    Sound-Interrupt-Routine (ohne Sound-Daten)

Ende des BIOS
fc4388    GEMDOS und BIOS-Routinen für GEMDOS


BIOS-Bildschirmtreiber
fca2ec    VDI-ESC-Handler
fca342    VDI-ESC 101
fca358    ASCII-Out/Conout-Einsprung
fca38e    CTRL-Kodes
fca3b4    TAB
fca3c6 T  ESC-Behandlung
fca45c    ESC-Verteiler
fca4b6    VDI ESC 1
fca4d8    VDI ESC 17,3,2, ESC E,A,C,D,H
fca540    Rest-Bildschirm, -Zeile löschen
fca564    VDI ESC 11,13,14,15,16,18,19
fca5e8    ESC I,L,M
fca636    Cursor ein, ESC e,f,j,k,l
fca6b6    Bildschirmbereich löschen, LF
fca720    XBIOS-Cursconf
fca778    ASCII-Zeichen ausgeben
fca846 T  Blitter-Routine #0
fca8d4    ab Cursor nach oben scrollen
fca8da T  Blitter-Routine #1
fca936    ab Cursor nach unten scrollen
fca93c T  Blitter-Routine #2
fca9b4    Bildschirmbereich löschen
fcaa46    Cursor positionieren
fcab20    ...
fcb4fc    TOS-Bildschirmausgabe je nach Auflösung init.
fcb586    Blitter-Vektoren installieren
fcb5a0    ...
fcb5de    Blitter-Vektor-Tabelle laden fcb65e   ...
```

→

Es wird vom BIOS auf ungleich Null gesetzt, wenn ein Header-Byte eines "Paketes" empfangen wurde, und ist wieder Null, wenn der Empfang des "Pakets" beendet ist. Wenn man das Flag abfragen möchte, bevor man seine Interrupt-Routine installiert, sollte man zuerst solange warten, bis es Null wird, dann das Interrupt-Level (IPL) der CPU auf 7 setzen, und, falls das Flag nun schon wieder gesetzt ist, den IPL wieder zurücksetzen und erneut warten. Ansonsten kann *ikbsys* neu gesetzt und der IPL restauriert werden. So schlägt es ATARI vor. Ich würde zusätzlich noch eine Timeout-Abfrage einbauen, um eine Endlosschleife zu verhindern, falls das "Paket" aus irgendwelchen Gründen nicht vollständig gesendet wird bzw. ankommt.

Hier noch einige Erklärungen zur verbesserten Diskettenwechselerkennung, die oben schon erwähnt wurde. Bei jedem Diskettenzugriff merkt sich das BIOS die aktuelle Systemzeit. Die *Mediach*-Routine meldet nun, wenn es eigentlich einen "unsicheren" Wechselstatus festgestellt hat, trotzdem ein "kein Wechsel", wenn seit dem letzten Zugriff nicht mehr als eine bestimmte maximale Zugriffszeit vergangen ist. Es geht also davon aus, daß "schnelle" Diskettenwechsel (kürzer als die maximale Zugriffszeit) nicht möglich sind. Dies spart das zu häufige Lesen des Boot-Sektors bei schreibgeschützten Disketten (wo ein unsicherer Status ständig auftritt).

Bisher wurde zur Zeitmessung der 200-Hz-Zähler (*_hz_200*) benutzt, nun wird *frclock* genommen (warum nur?). Daher ist die maximale Zugriffszeit jetzt von der Bildwiederholfrequenz abhängig. Sie liegt bei 1,2 s, 1,37 s bzw. 1,64 s für 71 Hz, 60 Hz bzw. 50 Hz Bildwiederholfrequenz (bisher immer 1,5 s). Daher wird in *hdv_init* die interne Variable für die maximale Zugriffszeit auf 82 statt bisher 300 gesetzt. Die eigentliche Verbesserung besteht nun darin, daß die gemerkte letzte Zugriffszeit beim Wechsel zwischen den logischen Laufwerken A: und B: nach der Aufforderung "Bitte Disk X: in Floppy A: einlegen" auf Null gesetzt wird. Damit liefert *Mediach* nach einem solchen Wechsel nun auf jeden Fall den wahren Diskettenwechselstatus zurück; bisher wurde also bei einer Wechselzeit unter 1,5 s der Wechsel von *Mediach* nicht gemeldet.

Die folgenden Änderungen sind von ATARI *nicht* dokumentiert worden, aber trotzdem wahr.

BIOS-Listing wesentlich geänderter Routinen

```
;OS Header
FC0000 BRA      $FC0030
FC0002 dc.b     1,4                     ;Version 1.4
FC0004 dc.l     $FC0030                 ;Reset-Adresse
FC0008 dc.l     $FC0000                 ;Beginn TOS
FC000C dc.l     $00611c                 ;Beginn freies RAM
FC0010 dc.l     $FC0030
FC0014 dc.l     $FE8230                 ;Adresse GEM-MAGIC
FC0018 dc.l     $04061989               ;Default-Systemdatum 22.2.1989
FC001C dc.w     3                       ;Land: FRG, PAL-Norm
FC001E dc.w     $1286                   ;Default-Systemdatum im GEMDOS-
                                         Format
FC0020 dc.l     $00378c                 ;Adresse GEMDOS-Variable 'mifl'
FC0024 dc.l     $000E7d                 ;Adresse BIOS-Tastaturvariablen
FC0028 dc.l     $005622                 ;Adresse GEMDOS-Variable 'act_pd'
FC002C dc.l     0
```

Listings wesentlich geänderter Routinen

```
;Warmstart
...
FC0432 JSR      $FC95C8                 ;GEMDOS init.
FC0438 MOVE.W   $FC001E,$0060BE         ;Default-Datum setzen
FC0442 JSR      $FC1F4C                 ;MEGA-Uhr übernehmen,
                                              falls vorhanden
FC0448 BCC      $FC0462                 ;*** -> MEGA-Uhr da       ***
FC044A BSR      $FC31A8                 ;*** IKBD-Uhr auslesen    ***
FC044E SWAP     D0                      ;*** Datum holen          ***
FC0450 TST.B    D0                      ;***                      ***
FC0452 BEQ      $FC0462                 ;*** -> "Zeit ungültig"   ***
FC0454 MOVE.W   D0,$0060BE              ;*** Datum übernehmen     ***
FC045A SWAP     D0                      ;***                      ***
FC045C MOVE.W   D0,$00378A              ;*** Zeit übernehmen      ***
FC0462 BSR      $FC0530                 ;Floppy-Boot versuchen
...


;Boot-Sektor vom DMA-Bus laden
;A4 von DMA-Boot-Routine gelöscht
FC0578 MOVEQ    #$01,D5                 ;*** 2 Versuche ***
FC057A LEA      $8606(A4),A6            ;alles wie bisher...
FC057E LEA      $8604(A4),A5
FC0582 ST       $043E(A4)
FC0586 MOVE.L   $04C6(A4),-(A7)
FC058A MOVE.B   $0003(A7),$860D(A4)
FC0590 MOVE.B   $0002(A7),$860B(A4)
FC0596 MOVE.B   $0001(A7),$8609(A4)
FC059C ADDQ.W   #4,A7
FC059E MOVE.W   #$0098,(A6)
FC05A2 MOVE.W   #$0198,(A6)
FC05A6 MOVE.W   #$0098,(A6)
FC05AA MOVE.W   #$0001,(A5)
FC05AE MOVE.W   #$0088,(A6)
FC05B2 MOVE.B   D7,D0
FC05B4 OR.B     #$08,D0
FC05B8 SWAP     D0
FC05BA MOVE.W   #$008A,D0
FC05BE BSR      $FC060E
FC05C0 BNE      $FC05F0                 ;-> Timeout bei Kommando-Byte:
                                                   Abbruch
FC05C2 MOVEQ    #$03,D6
FC05C4 LEA      $FC05FE(PC),A0
FC05C8 MOVE.L   (A0)+,D0
FC05CA BSR      $FC060E
FC05CC BNE      $FC05F0                 ;-> Timeout bei Kommando-Byte:
                                                   Abbruch
FC05CE DBF      D6,$FC05C8
FC05D2 MOVE.L   #$0000000A,(A5)
FC05D8 MOVE.W   #$0190,D1
FC05DC BSR      $FC0612
FC05DE BNE      $FC05F0                 ;-> Timeout bei Kommando-Byte:
                                                   Abbruch
FC05E0 MOVE.W   #$008A,(A6)
FC05E4 MOVE.W   (A5),D0                 ;DMA-Statusregister
FC05E6 AND.W    #$00FF,D0
FC05EA BEQ      $FC05F2                 ;-> ok
FC05EC DBF      D5,$FC057A              ;*** DMA-Fehler: neuer Versuch ***
FC05F0 MOVEQ    #$FF,D0                 ;Fehlermeldung
```

→

Die *ESC Y*-Funktion des VT52-Emulators zum Setzen der Cursor-Position kontrolliert nun den Spalten- und Zeilenparameter. Bei Werten größer als die maximale Spalte bzw. Zeile wird der Maximalwert benutzt. Bisher wurden beliebige Koordinaten "geschluckt". Negative Werte werden weiterhin nicht abgefangen.

Die XBIOS-Funktion *Gettime* blieb bisher in einer Endlosschleife hängen, wenn das vom Tastaturprozessor angeforderte Uhrzeit-Paket nicht ankam. Nun erfolgt nach einer Sekunde ein Timeout, und *Gettime* liefert dann Null zurück.

Die über die *IOREC*-Strukturen verwalteten I/O-Puffer für Tastatur, RS232 und Midi durften nur maximal 32 kB groß sein. Jetzt sind sogar 64 kB erlaubt.

Für Leute, die eigene Interrupt-Routinen über die *Kbdvbase*-Vektoren *midisys* und *ikbdsys* einhängen, hat TOS 1.4 noch eine Überraschung parat. Vom BIOS werden die Register D0-D3, A0-A3 und A5 gerettet, so daß man annehmen konnte, daß man diese Register selbst nicht retten muß (dokumentiert ist die Programmierung solcher Interrupt-Routinen ja nirgends). Wer nun A5 verändert hat, guckt ab TOS 1.4 in die Röhre, da A5 nach Beendigung der eigenen Routinen immer Null sein muß. Dies kommt durch die Code-Optimierungen zustande, die ATARI vorgenommen hat.

Neue garantierte Systemvariablen gibt es eigentlich keine. "Eigentlich" heißt, daß die in früheren TOS-Versionen unbenutzte Systemvariable $59E (Wort) beim Reset zwar gelöscht, aber sonst nirgendwo im TOS angesprochen wird. Noch merkwürdiger ist, daß bei der Beta-Version an dieser Stelle stattdessen noch die Systemvariable *_bootdev* ($446) gelöscht wurde, wozu das auch immer gut sein sollte. Falls dies nicht einfach Unsinn ist, könnte es höchstens sein, daß $59E von einem anderen ATARI-Programm noch benötigt wird (ähnlich dem *punptr* des Harddisk-Treibers), wer weiß.

Der *system header block*, der am Anfang des TOS zu finden ist (die Systemvariable *_sysbase* ($4f2) zeigt darauf), ist nun offiziell dokumentiert (s. Abb. 1). Eine genauere Erläuterung findet sich in [4].

Neu ist allerdings die Dokumentation zu den *Konfigurations-Bits*. Bit 0 steht hier für NTSC- bzw. PAL-Fernsehnorm, Bits 1 bis 15 beinhalten eine Länderkennung (s. Abb. 2).

```
FC05F2 MOVE.W  #$0080,(A6)             ;Rest wie bisher...
FC05F6 TST.B   D0
FC05F8 SF      $043E(A4)
FC05FC RTS


;VBL-Routine
FC06C0 ADDQ.L  #1,$000466              ;zunächst wie bisher
FC06C6 SUBQ.W  #1,$000452
FC06CC BMI     $FC079A
FC06D0 MOVEM.L D0-A6,-(A7)
FC06D4 ADDQ.L  #1,$000462
FC06DA SUBA.L  A5,A5
FC06DC MOVE.B  $FA01(A5),D1       ;*** Monochrom-Bit nur einmal lesen ***
FC06E0 MOVE.B  $8260(A5),D0
FC06E4 AND.B   #$03,D0
FC06E8 CMP.B   #$02,D0
FC06EC BGE     $FC0702
FC06EE BTST    #$07,D1            ;*** Abfrage im Register            ***
FC06F2 BNE     $FC0722
FC06F4 MOVE.W  #$07D0,D0
FC06F8 DBF     D0,$FC06F8
FC06FC MOVE.B  #$02,D0
FC0700 BRA     $FC0714
FC0702 BTST    #$07,D1            ;*** Abfrage im Register            ***
FC0706 BEQ     $FC0722
...                               ;ab hier weiter wie bisher


;Diskettenwechsel-Aufforderung
FC1640 CMPI.W  #$0001,$0004A6          ;zunächst nichts Neues
FC1648 BNE     $FC1680
FC164A MOVE.W  $0010(A7),D0
FC164E CMP.W   $000ED4,D0
FC1654 BEQ     $FC167C
FC1656 MOVE.W  D0,-(A7)
FC1658 MOVE.W  #$FFEF,-(A7)
FC165C BSR     $FC07BA
FC1660 ADDQ.W  #4,A7
FC1662 MOVE.W  #$FFFF,$0009FA          ;Schreibschutzstatus A,B unsicher
FC166A LEA     $0009FC,A0              ;*** letzte Zugriffszeit  ***
FC1670 CLR.L   (A0)+                   ;*** rücksetzen für A:    ***
FC1672 CLR.L   (A0)                    ;*** und B                ***
FC1674 MOVE.W  $0010(A7),$000ED4       ;akt. Floppy neu setzen
FC167C CLR.W   $0010(A7)
FC1680 RTS


;XBIOS #41: Seek-Rate ermitteln/neu setzen
FC1692 LEA     $000A4C,A0          ;DSB Floppy A
FC1698 TST.W   $0004(A7)           ;Laufwerk
FC169C BEQ     $FC16A4
FC169E LEA     $000A50,A0          ;DSB Floppy B
FC16A4 MOVE.W  $0002(A0),D0
FC16A8 MOVE.W  $0006(A7),D1        ;neue Seek-Rate
FC16AC CMP.W   #$FFFF,D1
FC16B0 BEQ     $FC16B6             ;-> nein: nur lesen
FC16B2 MOVE.W  D1,$0002(A0)        ;neue Seek-Rate setzen
FC16B6 EXT.L   D0                  ;akt. Seek-Rate zurück
FC16B8 RTS


;Floppy-hdv_boot
FC1CC6 LINK    A6,#$0000
FC1CCA MOVEM.L D6-D7,-(A7)
FC1CCE JSR     $FC0BD8             ;hdv_init aufrufen
FC1CD4 TST.W   $0004A6             ;nflops
FC1CDA BEQ     $FC1D12             ;-> keine Floppy: Abbruch mit 1
FC1CDC MOVEQ   #$02,D7
FC1CDE MOVE.W  #$0001,(A7)         ;1 Sektor
FC1CE2 CLR.L   -(A7)               ;Seite 0 und Spur 0 (*** optimiert! ***)
FC1CE4 MOVE.W  #$0001,-(A7)        ;Sektor 1
FC1CE8 CLR.W   -(A7)               ;*** immer Floppy A: statt 'bootdev' ***
                                   ;(Abbruch bei 'bootdev' >=2 fehlt jetzt)
FC1CEA CLR.L   -(A7)                 ;Filler
FC1CEC MOVE.L  #$0000181C,-(A7)      ;BIOS-Sektorpuffer
FC1CF2 JSR     $FC0F38               ;Floprd
FC1CF8 ADDA.W  #$0010,A7
FC1CFC TST.L   D0
FC1CFE BNE     $FC1D04               ;-> Sektor nicht lesbar: Abbruch mit 2
FC1D00 CLR.W   D7                    ;Boot-Sektor gelesen: kein Fehler
FC1D02 BRA     $FC1D10
FC1D04 TST.B   $0009F8               ;WP-Status von A: (früher falsch bei B:)
FC1D0A BNE     $FC1D10               ;-> nein: Abbruch mit 2
FC1D0C MOVEQ   #$03,D0               ;ja: Abbruch mit 3
FC1D0E BRA     $FC1D38
```

→

Für die Benutzung von *p_run* aus dem OS-Header ist es wichtig, die Adresse unter TOS 1.0 zu kennen. Bei allen TOS-Versionen außer der spanischen (!) lautet sie $602C, bei der spanischen $873C.

Alte und neue Fehler

Leider hat ATARI einige Fehler im BIOS immer noch nicht beseitigt. BIOS/XBIOS-Aufrufe aus Interrupt-Routinen heraus sind immer noch gefährlich, da die Interrupts während des Rettens der Register in die *BIOS save area* weiterhin nicht gesperrt werden (s. hierzu [2]).

Der *Boot Device*-Fehler besteht noch immer. Beim Erstellen der Ur-Environment-Strings wird das Boot-Laufwerk fälschlich mit 'move.b $446,d0' statt 'move.w $446,d0' ermittelt. Beim Laden der Auto-Ordner-Programme ist an einer Stelle eine Stack-Korrektur immer noch falsch (addq #6 statt add #12), was in ungünstigen Fällen zum Absturz führen kann.

Einige Fehler wurden nun zum Feature deklariert, d.h. man muß mit ihnen leben bzw. sie umgehen, wobei auch hier ATARI hilfreich zur Seite steht:

Die Ausgabe-Status-Routinen (*Bcostat*) für Tastaturprozessor und MIDI-Schnittstelle waren und bleiben vertauscht. Bcostat(4) liefert also nun offiziell den MIDI- und Bcostat(3) den IKBD-Status. Witzigerweise ist dies in der TOS-Doku genau verkehrt herum, also so wie es eigentlich sein sollte, angegeben!

screenpt wird nach dem Setzen der neuen Bildschirmadresse in der VBL-Routine nicht gelöscht. Dies führt zu Komplikationen, wenn parallel mit *Setscreen* gearbeitet wird (s. [4]). ATARI empfiehlt hier, entweder nur jeweils eine Möglichkeit zu benutzen, oder vor dem Aufruf von *Setscreen screenpt* auf Null zu setzen. Da aber ein vorher gelaufenes Programm schon screenpt benutzt haben könnte, bleibt einem wohl nichts anderes übrig, als *screenpt* bei jedem *Setscreen* zu löschen!

Programme, die sich in den *resvector* eingehängt haben, können zwar mit 'jmp (a6)' ins BIOS zurückspringen, doch müssen sie sich dann vorher aus *resvector* aushängen, da sie sonst erneut aufgerufen werden. Daher hatte ATARI in der Dokumentation zur Beta-Test-Version des TOS 1.4 garantiert, daß 'jmp $24(a6)' zum Sprung ins BIOS nach der *resvector*-Abfrage führt (siehe [4]). Dies ist in der TOS 1.4-Doku wieder verschwunden, vermutlich weil sich damit nur jeweils ein Programm in *resvector* installieren kann. Nun wird eine andere Methode vorgeschrieben, nach der sich jedes Programm bei der Installation seiner Reset-Routine *resvector* und *resvalid* merken soll, um diese Werte, wenn es über resvector aufgerufen wird, wieder zurückzusetzen. Dadurch können mehrere Programme auf diese Weise resident gemacht werden, allerdings müssen sie sich nun auch beim Reset aushängen!

TOS 1.4 sollte dafür sorgen, daß Zeit und Datum zumindest beim Reset nicht verlorengehen, indem es ausnutzt, daß der Tastaturprozessor die Zeit beim Reset nicht vergißt. Für ST-Benutzer, die über keine eingebaute Uhr verfügen, wäre dies eine Erleichterung, da sie die Uhr nur einmal nach dem Einschalten stellen müßten (z.B. mit dem Kontrollfeld-Accessory). Nur wenn die Uhr beim Warmstart nicht sinnvoll eingestellt ist, würde sie auf den bekannten Default-Wert (hier 6.4.1989) gesetzt. Beim Warmstart wird also zunächst geprüft, ob eine eingebaute MEGA-Uhr vorhanden ist, um ihr gegebenenfalls die Priorität zu geben. Diese Abfrage ist falsch programmiert, so daß stets angenommen wird, daß eine MEGA-Uhr vorhanden ist, und somit niemals die Zeit aus der Tastaturprozessor-Uhr übernommen wird. Dokumentiert ist dieses offensichtlich geplante Feature übrigens auch nicht. Vielleicht deshalb, weil man bei ATARI den Fehler nicht finden konnte?!

Offset	Name	C-Datentyp	Bedeutung
$00	os_entry	unsigned int	Sprungbefehl auf Reset-Handler
$02	os_version	unsigned int	TOS-Versionsnummer
$04	reseth	char *	Zeiger auf Reset-Handler
$08	os_beg	OSHEADER *	Zeiger auf diesen Header
$0C	os_end	char *	Zeiger auf Ende des von BIOS/GEMDOS/VDI benutzten RAMs
$10	os_rsv1	char *	unbenutzt, reserviert
$14	os_magic	char *	Zeiger auf Parameterblock für Speicherbedarf des GEM
$18	os_date	long	Datum der TOS-Erstellung (Format $JJJJMMDD)
$1C	os_conf	unsigned int	Konfigurations-Bits (s. Text)
$1E	os_dosdate	unsigned int	Datum der TOS-Erstellung (GEMDOS-Format)
;ab hier erst ab TOS 1.2			
$20	p_root	char **	Zeiger auf 'mifl'-Listen
$24	pkbshift	char **	Zeiger auf Keyboard-Shift-Variable
$28	p_run	char **	Zeiger auf Zeiger des aktuellen GEMDOS-Prozeßdescriptors
$2C	p_rsv2	char *	unbenutzt, reserviert

Abb. 1: Der "OS-header" (OSHEADER)

0	USA	USA
1	FRG	BR Deutschland
2	FRA	Frankreich
3	UK	Großbritannien
4	SPA	Spanien
5	ITA	Italien
6	SWE	Schweden
7	SWF	Schweiz (Französisch)
8	SWG	Schweiz (Deutsch)
9	TUR	Türkei
10	FIN	Finnland
11	NOR	Norwegen
12	DEN	Dänemark
13	SAU	Saudi-Arabien
14	HOL	Niederlande

Abb. 2: Länderkennungen

Beim Überarbeiten der *Rsconf*-Funktion des XBIOS schlich sich leider ein Fehler ein, der dazu führt, daß der Handshake-Modus sich nicht richtig setzen läßt. Auch ATARI hat dies erkannt und bietet daher ein Patch-Programm, das diesen und einen weiteren Fehler in *shel_find* korrigiert. Dieser Patch liegt reset-resident von $600-$800 und wird von einem AUTO-Ordner-Programm installiert.

Die TOS-Dokumentation behauptet, das Setzen von 'RTS/CTS' würde zum Abschalten jeden Handshakes führen. Für mich sieht das eher so aus, als würde das Setzen von 'XON/XOFF' zum Abschalten und das Setzen von 'RTS/CTS' zum Aktivieren von 'XON/XOFF' führen, ich konnte dies aber nicht genau verifizieren. Auch das Patch-Programm liegt mir zur Zeit nicht vor, so daß ich über dessen Funktionsfähigkeit nichts sagen kann.

SCSI Speed Drive Festplatten



- *SCSI Speed Drive Festplatten — eine der schnellsten und leisesten Festplatten für den Atari ST.*
  1 Jahr Garantie,
  7 Tage Rückgaberecht,
  49 MB 28 ms und 85 MB 28 ms.
- *Neu: Ultra Speed Drive 42 MB, 19 ms, 64 KB Cache*
- *Neu: 155 MB SCSI Speed Drive Streamer, Übertragungsrate 6,5 MB/Minute*
- *Nicht nur Bestellungen werden zu 95 % innerhalb von 24 Stunden ausgeliefert, auch technische Überprüfungen, Anpassungen und Reparaturen brauchen selten länger. Wer sonst bietet das?*
  Info-Telefon (0 23 05) 1 20 22

Leistungsdaten: Die Verbindung eines reinen SCSI-Hochgeschwindigkeits-Hostadapters und die Verwendung von SCSI-Festplatten ermöglichen Geschwindigkeiten, die bisher noch nicht erreicht wurden. In der Praxis ergeben sich Geschwindigkeitssteigerungen zwischen 30 — 60 %. Die Festplatte ist 100 % kompatibel zu den original Atari ST Festplatten. Das heißt: Sie können auch andere Harddisktreiber oder den original Atari Harddisktreiber benutzen. PC Speed, PC Ditto, Aladin usw. sind auf unserer Festplatte selbstverständlich lauffähig. Desweiteren ist in der Festplatte eine Echtzeituhr integriert. Die Festplatte wird mit einer sehr umfangreichen Software ausgeliefert.

DMA-Port: Der DMA-Port der Festplatte ist herausgeführt und komplett gepuffert. Das macht den Anschluß weiterer DMA-Geräte (Atari Laserdrucker, weitere Festplatten etc.) möglich.

Die Technik: Durch eine besondere Art der Luftzirkulation wird die Festplatte ohne störenden Lüfter betrieben und die Laufgeräusche der Festplatte optimal unterdrückt. Das macht die Festplatte zu eine der leisesten Festplatten für den Atari ST.
Das Gehäuse entspricht den Gehäuseabmessungen des Mega ST. Durch die robuste Ausführung kann es auch als Monitoruntersatz verwendet werden. Das Netzteil (VDE, GS) verfügt über 65 W und kann auch eine zweite interne Festplatte versorgen. Alle Festplatten verfügen über einen AUTO Park und sind mit einer speziellen Pufferung ausgestattet, die vor Schäden der Festplatte schützen, die durch kleine Stöße entstehen können.

Die Software: „SCSI TOOLS" ist ein bisher einzigartiges Softwarepaket, das in Leistung, Zuverlässigkeit und Geschwindigkeit neue Maßstäbe setzt. SCSI TOOLS ist die erste HD-Software, die zum neuen Atari-Standard (AHDI 3.0) kompatibel ist und die neuen Möglichkeiten von TOS 1.4 nutzt. Hochgeschwindigkeitstreiber voll AHDI 3.0 kompatibel, beliebig große Partitionen, Sektorgröße veränderbar, variabler GEM DOS Cache Buffer, Turbo DOS Kompatiblitätsmodus, besonders ausgeklügelter Softwareschreibschutz, Booten von allen Partition per Tastendruck, zusätzliche Datensicherheit durch Sicherheitskopie der Verwaltungsinformationen, Ausmappen von defekten Sektoren auf Controler und GEM DOS Ebene, komfortable graphische Benutzerführung mit Help Funktion, mit TOS 1.6 (1040 STE) lauffähig, Speed Cache, Treibersoftware für integrierte Echtzeituhr, außergewöhnliches Back Up Programm.

7 Tage Rückgaberecht

Garantie, Service: Auf unsere Festplattensysteme gewähren wir 1 ganzes Jahr Garantie. Sagt Ihnen die Festplatte trotz unserer Qualität nicht zu, gewähren wir Ihnen ein siebentägiges Rückgaberecht unter Übernahme der Porto- und Verpackungskosten Ihrerseits.

Preise: 49 MB
28 ms 1598,- DM,
85 MB 28 ms 1998,- DM,
155 MB SCSI
Streamer 1998,- DM

Hard & Soft A. Herberg

Bahnhofstr. 289 · 4620 Castrop-Rauxel · ☎ (0 23 05) 1 57 64 · Fax 1 20 22

Kommentiertes BIOS-Listing

Für die Spezialisten unter Ihnen möchte ich noch direkt auf die Änderungen im BIOS eingehen. Generell wurden einige Optimierungen vorgenommen. Der Zugriff auf Systemvariablen wird nun noch häufiger mit der Adressierungsart "Adreßregister indirekt mit 16-Bit-Offset" (bei gelöschtem Adreßregister) statt "absolut lang" vorgenommen. Diese Methode wird außerdem jetzt auch bei den I/O-Adressen ab $FF8000 benutzt. Daraus darf man wohl schließen, daß ATARI immer noch keinen Assembler verwendet, der die "absolut kurz"-Adressierung beherrscht. Weiterhin wurden jetzt einige (aber noch nicht alle) "MOVE.X #0,.."-Befehle durch "CLR.X .." ersetzt, und andere Kleinigkeiten wurden geschickter programmiert. Prinzipielle Optimierungen gibt es sonst keine.

Zur Analyse habe ich das BIOS bis auf die Hardcopy-Routine disassembliert und mehr oder weniger gründlich mit dem des TOS 1.2 verglichen. Im abgedruckten Listing finden sich alle Routinen, an denen inhaltliche Änderungen vorgenommen wurden (markiert durch ***). Ausgenommen sind nur einige Teile, bei denen lediglich Kleinigkeiten verändert wurden, die hier im Text erwähnt werden. Die Tabelle umfaßt die Anfangsadressen aller wichtigen Routinen, was - zusammen mit einem BIOS-Listing des TOS 1.2, z.B. [3] - eine gute Orientierungshilfe bietet.

Noch nicht erwähnt habe ich bisher einige Kleinigkeiten. In der VBL-Routine wurde zweimal die Monochrommonitor-Detect-Leitung direkt am MFP abgefragt. Da sie sich zwischen den beiden Abfragen geändert haben könnte, wird der MFP nur noch einmal angesprochen; das zweite Mal wird der beim ersten Mal gemerkte Zustand getestet.

Bei der Ausgabe eines Zeichens über die Centronics-Schnittstelle werden alle Interrupts gesperrt, während der Strobe gesendet wird. Damit wird verhindert, daß der Strobe wesentlich länger als üblich (zu lange?) dauern kann.

Beim Test, ob die Hardware-Uhr des MEGA ST vorhanden ist, wird nun auch ein Bus-Error abgefangen. Normalerweise gibt es zwar keinen Bus-Error, wenn auf den Adreßbereich der MEGA-Uhr zugegriffen wird, obwohl diese gar nicht vorhanden ist, aber vielleicht gibt es hier

```
FC1D10 BRA     $FC1D14
FC1D12 MOVEQ   #$01,D7            ;Abbruch mit 1
FC1D14 TST.W   D7
FC1D16 BEQ     $FC1D0C            ;-> bisher kein Fehler
FC1D18 MOVE.W  D7,D0              ;Abbruch mit D7
FC1D1A BRA     $FC1D38
FC1D1C MOVE.W  #$0100,(A7)        ;Checksumme berechnen
FC1D20 MOVE.L  #$0000181C,-(A7)
FC1D26 BSR     $FC1E2E
FC1D2A ADDQ.L  #4,A7
FC1D2C CMP.W   #$1234,D0
FC1D30 BNE     $FC1D36     ;-> Boot-Sektor nicht ausführbar: Abbruch mit 4
FC1D32 CLR.W   D0          ;Boot-Sektor ausführbar: Ende mit 0
FC1D34 BRA     $FC1D38
FC1D36 MOVEQ   #$04,D0             ;Abbruch mit 4
FC1D38 TST.L   (A7)+
FC1D3A MOVEM.L (A7)+,D7
FC1D3E UNLK    A6
FC1D40 RTS

;Test ob MEGA-Uhr vorhanden
FC1F70 SUBA.L  A1,A1               ;*** für optimierten Zugriff  ***
FC1F72 MOVE.W  #$FC20,A0           ;I/O-Basisadresse der MEGA-Uhr
FC1F76 MOVE.L  $0008(A1),D2        ;*** Bus-Error-Vektor merken  ***
FC1F7A MOVE.L  A7,A2               ;*** SP merken                ***
FC1F7C MOVE.L  #$FC1FB6,$0008(A1)  ;*** eigene Bus-Error-Routine ***
FC1F84 MOVE.B  #$09,$001B(A0)      ;Uhr an, Alarm aus, Bank 1 anwählen
FC1F8A MOVE.L  D2,$0008(A1)        ;*** alte Bus Error-Routine   ***
FC1F8E MOVE.W  #$0A05,D0
FC1F92 MOVEP.W D0,$0005(A0)        ;Alarmzeit-Minuten auf unsinnigen Wert
FC1F96 MOVEP.W $0005(A0),D1
FC1F9A AND.W   #$0F0F,D1           ;hat dies geklappt ?
FC1F9E CMP.W   D0,D1
FC1FA0 BNE     $FC1FBC             ;-> nein: MEGA-Uhr nicht vorhanden
FC1FA2 MOVE.B  #$01,$0001(A0)      ;CLKOUT auf 16384 Hz
FC1FA8 MOVE.B  #$08,$001B(A0)      ;Uhr an, Alarm aus, Bank 0 anwählen
FC1FAE MOVE.B  #$00,$001D(A0)      ;Testregister löschen, C=0 !
FC1FB4 RTS
;eigene Bus Error-Routine
FC1FB6 MOVE.L  A2,A7               ;*** SP restaurieren                ***
FC1FB8 MOVE.L  D2,$0008(A1)        ;*** Bus-Error-Vektor restaurieren ***
FC1FBC ORI.B   #$01,CCR               ;C=1: MEGA-Uhr nicht vorhanden
FC1FC0 RTS

;Gettime von IKBD
FC31A8 MOVE.B  #$FF,$0EAE(A5)      ;Flag setzen: Zeitpaket noch nicht da
FC31AE MOVE.B  #$1C,D1             ;Befehl 'interrogate time-of-day clock'
FC31B2 BSR     $FC3460             ;Zeitpaket vom IKBD anfordern
;*** auf Zeit-Paket warten mit Timeout ***
FC31B6 MOVE.L  $04BA(A5),A0
FC31BA ADDA.W  #$00C8,A0           ;hz_200 + 200 (Startzeit + 1s)
FC31BE MOVEQ   #$00,D0
FC31C0 CMPA.L  $04BA(A5),A0
FC31C4 BCS     $FC31D0             ;-> 1s überschritten: 0L zurückgeben
FC31C6 TST.B   $0EAE(A5)           ;Flag testen
FC31CA BNE     $FC31C0             ;-> Zeitpaket noch nicht empfangen
FC31CC MOVE.L  $0E6C(A5),D0        ;empfangenes Paket im XBIOS-Format
FC31D0 RTS

;Zeichen über Centronics ausgeben
FC3332 MOVE.W  SR,D3                  ;SR retten
FC3334 ORI.W   #$0700,SR              ;IPL 7
FC3338 MOVEQ   #$07,D1                ;Register 7 PSG lesen
FC333A BSR     $FC4086                ;(Giaccess)
FC333E ORI.B   #$80,D0                ;Port B auf Ausgabe schalten
FC3342 MOVEQ   #$87,D1
FC3344 BSR     $FC4086                ;(Giaccess)
FC3348 MOVE.W  D3,SR                  ;IPL restaurieren
FC334A MOVE.W  $0006(A7),D0           ;auszugebendes Zeichen
FC334E MOVEQ   #$8F,D1                ;nach Port B schreiben
FC3350 BSR     $FC4086                ;(Giaccess)
FC3354 MOVE.W  SR,-(A7)               ;*** SR retten        ***
FC3356 ORI.W   #$0700,SR              ;*** IPL 7            ***
FC335A BSR     $FC336C                ;Strobe low setzen
FC335C BSR     $FC336C                ;nochmal !? (= Pause)
FC335E BSR     $FC3366                ;Strobe high setzen
FC3360 MOVE.W  (A7)+,SR               ;*** IPL restaurieren ***
FC3362 MOVEQ   #$FF,D0                ;alles klar
FC3364 RTS
FC3366 MOVEQ   #$20,D2                ;Strobe (Bit 5)
FC3368 BRA     $FC40BC                ;mit 'ongibit' setzen
FC336C MOVEQ   #$DF,D2                ;Strobe (Bit 5)
FC336E BRA     $FC40E2                ;mit 'offgibit' löschen
```

→

Ausnahmen oder Fehlfunktionen, denen hiermit abgeholfen werden soll.

In den Blitter-Routinen (im Bildschirmtreiber die Nummer 0 bis 2) gibt es Schleifen, die warten, bis der Blitter seine Arbeit beendet hat. Bisher sahen die so aus:

```
  ...
  moveq #7,D4
loop:
  bset  d4,(a5)   ;Blitter Busy testen und
                  neu starten
  nop
  bne.s loop      ;-> Blitter noch beschäftigt
```

Nun wird folgendes gemacht:

```
loop:
  tas   (a5)      ;Blitter Busy testen und
                  neu starten
  nop
  bmi.s loop      ;-> Blitter noch beschäftigt
```

Der Unterschied besteht darin, daß bei *tas* das Testen und Setzen des Bits 7 des Zieloperanden in einem nicht-unterbrechbaren Schreib-Lese-Zugriff des 68000 erfolgt. Beim *bset* könnte zwischen dem Testen und Neustarten des Blitters z.B. der DMA-Chip die Kontrolle über den Prozessor-Bus übernehmen. Sicherer ist die Verwendung von *tas* auf jeden Fall; ob es auch praktische Konsequenzen hat, weiß ich nicht.

Ein letztes Wort

So das war's für heute. Auch wenn ich das BIOS relativ gründlich durchgearbeitet habe, ist dazu sicherlich noch nicht das letzte Wort gesprochen. Warten wir's also ab. Festzuhalten bleibt, daß einige wichtige Punkte geändert oder verbessert wurden, anderes unverständlicherweise beim alten geblieben ist. Auch die TOS-Dokumentation ist ganz brauchbar und informativ. Ich kann das "Rainbow-TOS" nur empfehlen, da es einige ganz entscheidende Vorteile bietet, die hier nicht deutlich wurden, da sie hauptsächlich GEMDOS und Desktop betreffen. Bei dieser Gelegenheit möchte ich noch erwähnen, daß ich seit einiger Zeit auch in der MAUS-Mailbox Aachen (0241/154949) zu erreichen bin (bzw. im angeschlossenen MAUS- und FIDO-Net), falls Sie noch Fragen zur GEMDOS-Serie oder dem TOS im allgemeinen haben sollten.

Alex Esser

Literatur:

[1] Brod/Stepper: "Scheibenkleister II", MAXON 1989

[2] Esser: "EXTKEY-Tastaturbelegung einmal anders", ST-Computer 4/89

[3] Brückmann/Englisch/Gerits: "ST intern", Data Becker 1987

[4] Esser: "Die Systemvariablen des TOS", ST-Computer 11-12/88


```
;Rsconf
FC3A16 CMPI.W  #$FFFE,$0004(A7)    ;"Baudrate" = -2 ? FC3A1C BNE
$FC3A26
FC3A1E MOVE.W  $000A6E,D0          ;ja: alte Baudrate zurück
FC3A24 RTS
FC3A26 ORI.W   #$0700,SR           ;IPL 7 bis zum Ende des TRAP-Aufrufs
FC3A2A LEA     $000C70,A0               ;Adresse IOREC für RS232-Eingabe
FC3A30 LEA     $FFFA01,A1          ;Adress MFP
FC3A36 MOVEP.L $0028(A1),D7        ;UCR, RSR, TSR und UDR merken
                                   ;statt SCR, UCR, RSR und TSR
FC3A3A MOVE.W  $0006(A7),D0        ;Handshake-Modus
FC3A3E CMP.W   #$FFFF,D0           ;*** Test auf '-1' statt '<0' !      ***
FC3A42 BEQ     $FC3A58             ;-> nicht ändern
FC3A44 MOVE.B  D0,$0020(A0)        ;in IOREC-Verlängerung merken
FC3A48 BEQ     $FC3A54             ;-> kein Handshake
FC3A4A AND.B   #$3E,D0             ;*** Bits 0,6,7 löschen statt Bit 1 ***
FC3A4E BEQ     $FC3A54             ;-> kein Handshake
FC3A50 MOVE.B  #$01,D0
FC3A54 MOVE.B  D0,$0020(A0)        ;Handshake-Flag in IOREC setzen
                                   ;*** jetzt immer nur 0 oder 1        ***
FC3A58 TST.W   $0004(A7)           ;Baudrate
FC3A5C BMI     $FC3A98             ;-> nicht ändern
FC3A5E MOVEQ   #$00,D0
FC3A60 MOVE.B  D0,$002A(A1)        ;RSR löschen
FC3A64 MOVE.B  D0,$002C(A1)        ;TSR löschen
FC3A68 MOVE.W  $0004(A7),D1        ;neue Baudrate
FC3A6C MOVE.W  D1,$000A6E          ;merken für Rsconf(-2)
FC3A72 LEA     $FC3ACC,A2
FC3A78 MOVE.B  $00(A2,D1.W),D0     ;Timer-Control-Wert
FC3A7C LEA     $FC3ADC,A2
FC3A82 MOVE.B  $00(A2,D1.W),D2     ;Timer-Data-Wert
FC3A86 MOVE.L  D0,D1
FC3A88 MOVEQ   #$03,D0             ;Timer D
FC3A8A BSR     $FC36AC             ;Timer setzen
FC3A8E MOVEQ   #$01,D0
FC3A90 MOVE.B  D0,$002A(A1)              ;RSR: Empfänger ein etc.
FC3A94 MOVE.B  D0,$002C(A1)              ;TSR: Sender ein etc.
FC3A98 TST.W   $0008(A7)                 ;UCR
FC3A9C BMI     $FC3AA4                   ;-> nicht ändern
FC3A9E MOVE.B  $0009(A7),$0028(A1)       ;UCR setzen
FC3AA4 TST.W   $000A(A7)                 ;RSR
FC3AA8 BMI     $FC3AB0                   ;-> nicht ändern
FC3AAA MOVE.B  $000B(A7),$002A(A1)       ;RSR setzen
FC3AB0 TST.W   $000C(A7)                 ;TSR
FC3AB4 BMI     $FC3ABC
FC3AB6 MOVE.B  $000D(A7),$002C(A1)       ;TSR setzen
FC3ABC TST.W   $000E(A7)                 ;SCR
FC3AC0 BMI     $FC3AC8                   ;-> nicht ändern
FC3AC2 MOVE.B  $000F(A7),$0026(A1)       ;SCR setzen
FC3AC8 MOVE.L  D7,D0
FC3ACA RTS

;aus Tastatur-Interrupt (nach Behandlung SHIFT/CTRL/ALT/CAPS)
;verbesserte Tastenwiederholung
FC3CFA BTST    #$07,D0
FC3D04 BNE     $FC3D16             ;-> Taste losgelassen
FC3D00 MOVE.B  D0,$0E9B(A5)        ;Scancode für Tastenwiederholung
FC3D04 MOVE.B  $000E9E,$0E9C(A5)   ;Zähler für Wiederholschwelle init.
FC3D0C MOVE.B  $000E9F,$0E9D(A5)   ;Zähler für Wiederholrate init.
FC3D14 BRA     $FC3D48             ;Taste in Tastaturpuffer übernehmen
FC3D16 MOVE.B  D0,D1               ;Taste losgelassen
FC3D18 BCLR    #$07,D1             ;reiner Scancode
FC3D1C CMP.B   $0E9B(A5),D1
FC3D20 BNE     $FC3D30             ;-> andere Taste als letzte gedrückte
FC3D22 MOVEQ   #$00,D1             ;Tastenwiederholung aus
FC3D24 MOVE.B  D1,$0E9B(A5)
FC3D28 MOVE.B  D1,$0E9C(A5)
FC3D2C MOVE.B  D1,$0E9D(A5)
FC3D30 CMPI.B  #$C7,D0             ;HOME losgelassen ?
FC3D34 BEQ     $FC3D3E
FC3D36 CMPI.B  #$D2,D0             ;INSERT losgelassen ?
FC3D3A BNE     $FC3FBC             ;-> weder HOME noch INSERT: fertig
FC3D3E BTST    #$03,$0E7D(A5)      ;ALT gedrückt ?
FC3D44 BEQ     $FC3FBC             ;-> nein: fertig
FC3D48 ;Tastencode soll nach Tastaturpuffer, dazu weiter untersuchen

;Ende des Tastatur-Interrupts: Zeichen nach Tastaturpuffer
;D0: ASCII-Kode in Bits 0-7, Scan-Kode in Bits 8-15
FC3F74 MOVE.L  $0000(A0),A2        ;Pufferadresse
FC3F78 SWAP    D0                  ;Scan- u. ASCII-Kode nach Bits 16-31
FC3F7A MOVE.W  #$0000,D0
FC3F7E MOVE.B  $0E7D(A5),D0        ;Kbshift-Status
FC3F82 SWAP    D0                  ;Status jetzt nach Bits 16-23
FC3F84 LSL.L   #8,D0
FC3F86 LSR.W   #8,D0               ;Bits 0-7: ASCII, 16-23: Scan, 24-31:
                                                              Kbshift →
```

```
FC3F88 MOVE.L  D0,D2
FC3F8A BCLR    #$1C,D2          ;CAPS ignorieren
FC3F8E SWAP    D2
FC3F90 CMP.W   #$0C53,D2        ;CTRL-ALT-DEL ?
FC3F94 BEQ     $FC0030          ;-> Warmstart
FC3F98 CMP.W   #$0D53,D2        ;CTRL-ALT-SHIFT
RECHTS-DEL ?
FC3F9C BEQ     $FC4034          ;-> Kaltstart
FC3FA0 BTST    #$03,$0484(A5)   ;conterm: Kbshift für
Bconin ?
FC3FA6 BNE     $FC3FAE
FC3FA8 ANDI.L  #$00FFFFFF,D0    ;nein: Kbshift löschen
FC3FAE AND.L   #$0000FFFF,D1
FC3FB4 MOVE.L  D0,$00(A2,D1.L)  ;Zeichen nach
                                 Tastaturpuffer
FC3FB8 MOVE.W  D1,$0008(A0)     ;neuer Pufferzeiger
FC3FBC RTS

;Kaltstart auslösen
;äußerst gründlich, da Hauptspeicher komplett gelöscht
;es fehlt höchstens noch das RAM im IKBD...
FC4034 MOVE.W  #$2700,SR        ;IPL 7
FC4038 LEA     $FC4050(PC),A0   ;Löschroutine ins RAM
                                 kopieren
FC403C MOVE.W  #$000F,D0        ;32 Worte (ein bißchen
                                 viel zwar...)
FC4040 MOVE.W  #$0008,A1        ;ab Adresse $8
                                 (Bus-Error-Vektor) !
FC4044 MOVE.L  (A0)+,(A1)+
FC4046 DBF     D0,$FC4044
```

```
FC404A SUBA.L  A5,A5
FC404C JMP     $0008(A5)        ;Löschroutine
                                 starten
;Speicher-Löschroutine für Kaltstart, ins RAM ab $8
kopiert
FC4050 LEA     $FC4076(PC),A0   ;eigener Bus-Error-
                                 Handler
FC4054 MOVE.L  A0,$0008(A5)     ;Anfang dieser
                                 Routine nun selbst
                                 zerstört !!!
FC4058 LEA     $FC4082(PC),A0   ;Beginn löschen nach
                                 dieser Routine
FC405C MOVEQ   #$00,D0
FC405E MOVE.L  D0,D1
FC4060 MOVE.L  D0,D2
FC4062 MOVE.L  D0,D3
FC4064 MOVE.L  D0,D4
FC4066 MOVE.L  D0,D5
FC4068 MOVE.L  D0,D6
FC406A MOVE.L  D0,D7
FC406C MOVEM.L D0-D7,(A0)       ;schnelles Löschen
FC4070 ADDA.W  #$0020,A0
FC4074 BRA     $FC4072          ;immer weiter... bis
                                 Bus-Error kommt
FC4076 MOVE.L  $0004(A5),A0     ;ROM-Reset-Adresse
FC407A JMP     (A0)             ;Reset (wirkt wie
                                 Kaltstart) durch
                                 ROM
       dc.w 0                   ;???
```


RTS - Elektronik

Die neue Flachtastatur

Wir stellen aus
Hobby + Elektronik
89 Stuttgart
Halle 10

ATARI
Baureihe ST+ MEGA ST



Komfortable und preisgünstige Umrüstung mit hohem Bedien-Komfort und optimalem Design

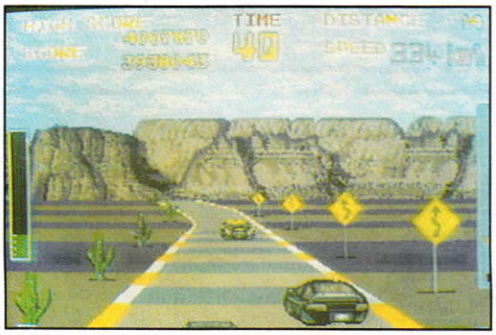
- Farblich abgesetzte Flachtastatur mit blendfreien Tastaturkappen
- Exakter Endanschlag durch Hubverkürzung **mit dem RTS-Anschlagsystem**
- Geräuscharme Betätigung durch Formgebung
- Sichere Dateneingabe durch große Tastenzwischenräume
- Gewohnte originale Tastenbedruckung
- Einfacher Einbau (alte Tastenkappe raus, neue rein)
- Klare Trennung der Funktions- und Schreibtastenblöcke
- Bedruckung: Deutsch, US-englisch, englisch, französisch, spanisch, VSM-Schweiz
- Verstärkung des Tastendruckes durch Federnsatz

Nr.	Artikel	Stück	Preis/DM
1	Tastensatz Farbe weiß Baureihe ST		99,–
2	Tastensatz Farbe weiß Baur. MEGA ST		105,–
3	Funktionstastensatz Farbe beige		25,–
4	Funktionstastensatz Farbe grau		25,–
5	Federnsatz für Baureihe ST		15,–

ATARI ist eingetragenes Warenzeichen der Atari-Cooperation

RTS - Elektronik Postfach 64 · 7533 Tiefenbronn ☎ (0 72 34) 69 15 + 52 32 Fax-Nr. 0 72 34 / 55 74





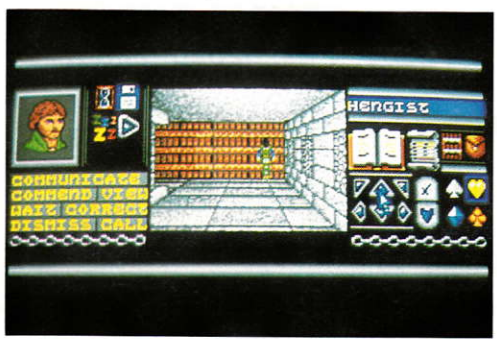
⇦ **Chase HQ**

Mitte September führte OCEAN in Manchester Mitarbeitern der Vertriebsfirma Ariolasoft sowie einigen Fachjournalisten Vorabversionen ihrer neusten Games vor: **"Adidas Golden Shoe"** ist ein Fußballspiel, bei dem ein oder zwei Spieler ähnlich wie bei "Microprose Soccer" und "Kick off" die Action aus der Vogelperspektive verfolgen. Apropos Vogel. In **"F29 Retaliator"** gehts in die Luft, um verschiedene militärische Ziele zu vernichten. Radar und Echtzeitanzeigen sollen bei diesem Flugsimulator mit ausgefüllter Vektorgrafik für realistisches Fluggefühl sorgen. Wem es in den Wolken zu luftig ist, kann in **"Chase HQ"** mit einem Porsche Turbo über die Straße brettern, um Gangsterautos zu rammen. Für Kämpfernaturen kommt **"The Untouchables"**, ein knallhartes Actionspiel, in dem der Spieler Elliot Ness und seine Anti-Mafia-Einheit durch sechs gefährliche Szenen steuert. Leute, denen diese bleihaltige Atmosphäre keinen Reiz bietet, kommen vielleicht mit **"Lost Patrol"** auf ihre Kosten. Eine Eliteeinheit von sieben stahlharten Soldaten ist mit einem Hubschrauber im vietnamesischen Dschungel abgestürzt. Keine Bange! Es handelt sich nicht um ein Metzel- und Meuchelspielchen. Strategie und Adventure-Elemente stehen im Vordergrund. Es gilt, durch feindliches Gebiet in ein amerikanisches Militärcanmp zu gelangen. Dabei müssen Sie die Truppe motivieren, für Proviant sorgen und sich mit den Eingeborenen verständigen. Ins dunkle Mittelalter führt **"Ivanhoe"**, ein horizontal scrollendes Actionspiel, in dem der Spieler den berühmten Ritter Ivanhoe steuert und mit Schwert und Schild gegen Drachen, Ritter, Kobolde und Geister kämpft.

Fans von Rollenspielen aufmerken: **"Image Works"** veröffentlichte kürzlich mit **"Bloodwych"** das erste Rollenspiel, in dem zwei Spieler gleichzeitig durch dunkle Dungeons laufen. Ein zweigeteilter Bildschirm macht`s möglich. Ziel des Spiels ist es, in einem unheimlichen Schloß vier Kristalle zu finden. Beide Spieler können gegeneinander oder im Teamwork die Aufgabe lösen. Wie in fast allen Rollenspielen ist die Fantasywelt mit miesen Monstern und hilfreichen Charakteren gefüllt. Wer keinen Mitspieler findet, kann sich auch alleine ins Abenteuer stürzen. Egal, ob Sie alleine oder zu zweit spielen, "Bloodwych" bleibt ein Rollenspielerlebnis.



⇦ **Ivanhoe**



⇦ **Bloodwych**

Garfield - Winter's Tail



Frech, fett und faul, das ist Garfield, wie er leibt und lebt. Nach einem ausgiebigen Schlemmermahl hat der beliebte Comic-Kater mächtig Kalorien in sich hineingestopft und liegt jetzt in seiner Kuschelkiste, um von noch mehr Schleckereien zu träumen. In seiner Fantasie ist er in der Nähe von Norditalien, wo es feine Lasagne gibt. Auch die Schweiz mit ihrer tollen Schokolade ist nicht weit weg. Keine Frage, der Spieler muß Garfield auf seiner erträumten Schlemmertour begleiten und aufpassen, daß er nicht dabei aufwacht. Per Joystick sucht man eines von drei Teilspielen aus: Der Abfahrtslauf hat's in sich. Baumstämmen, Ästen und Begrenzungssteinen sollte man ausweichen. Gelingt das nicht, vollführt Garfield eine unfreiwillige Akrobateneinlage im Pulverschnee. Stärkungen werden am Rand der Piste bereitgehalten. Auffuttern! Das bringt Energie. Nimmt Garfield auch die letzte Sprungschanze mit Bravour, kracht er durch das Dach einer Lasagne-Fabrik. Hier darf er sich solange den Bauch vollstopfen, bis Odie kommt und ein Kaminfeuer auspustet. Weiter geht es dann sogleich in einer Schokoladenfabrik, einem Labyrinth aus Plattformen, Aufzügen, Stangen und Fließbändern. Odie sollte man des öfteren einen Tritt in den Allerwertesten verpassen, sonst frißt er alles weg. In der dritten Actionszene verfolgt Garfield ein Huhn, das auf einem zugefrorenen See Schokoladeneier legt. Sie sehen, ein recht abwechslungsreiches Actionspiel. Freude bereitet bei diesem Spiel allerdings nur die samtweiche Animation des Titelhelden (teilweise echt trickfilmreif!). Ansonsten halte ich das neue Garfield Spiel für einen Langweiler. Bis auf die Grafiken ist das Programm auch technisch danebengegangen. Das Scrolling ruckelt, als erschüttere ein Erdbeben Garfields Traumlandschaft. "Garfield - Winter Tail" ist nur eingefleischten Fans des Comic-Katers zu empfehlen.

CBO



Hard'n Heavy!

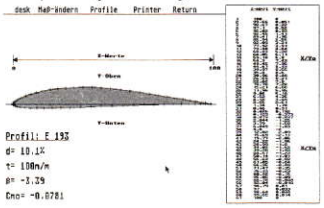


Fans von Plattformspielen aufgepaßt! "ReLine" hat ein starkes Spielchen auf der Pfanne, bei dem Schuß- und Sammelleidenschaften geweckt werden. In Gestalt eines kleinen Roboters flitzt man durch 24 verschiedene Spielstufen, möglichst bevor das Zeitlimit abläuft. Dabei versuchen Fantasymonster (herumpurzelnde Kugeln, schleimige Sterne und gnadenlose Gespenster), den Roboter zu haschen. Vorsicht! Jede Kollision kostet ein Bildschirmleben. Widersacher kann man überspringen oder mit einer kleinen Kugel aus dem Weg räumen. Dazu drückt man einfach den Feuerknopf - und schwupp, segnet das Monster das Zeitliche. Mit Extrawaffen geht's noch leichter! Eine Smart Bomb fegt alle Gegner vom Bildschirm. Trotz effektiver Waffen sind einige Spielstufen eine ganz schön harte Nuß. Da ist schnell ein Bildschirmleben verloren. Neue bekommt der springfreudige Roboter übrigens hinzu, wenn er von den Sternchen, die überall herumliegen, 100 Stück einsammelt. "Hard'n Heavy" steckt voller Überraschungen. In manchem Abgrund haben die Programmierer Schatzkammern versteckt, in denen sich nach Herzenslust Sternchen zusammenraffen lassen. Eine spaßige Angelegenheit! Nicht weniger unterhaltsam sind die verschiedenen Spielmodi. Entweder spielt man alleine, oder man macht Gebrauch von den beiden Zwei-Spieler-Modi. Da wechseln sich zwei Spieler ab, wenn einer von beiden ein Bildschirmleben verloren hat oder der Wechsel erfolgt automatisch nach einigen Sekunden. Ganz schön stressig, wenn Spieler 1 gerade in eine unangenehme Situation gelangt ist und Spieler 2 übernehmen muß. "Hard'n Heavy" macht gigantisch viel Spaß. Bei diesem Spiel stimmt einfach alles. Pfiffige Musik und die beiden Zwei-Spieler-Modi machen aus ReLines neuem Game das beste Hüpfspielchen seit "Great Giana Sisters". Der einzige Kritikpunkt gilt den einfallslosen Hintergrundgrafiken. Wenn man die langweiligen Muster überhaupt als solche bezeichnen kann. Doch dieses Manko tut dem Spielspaß keinen Abbruch.

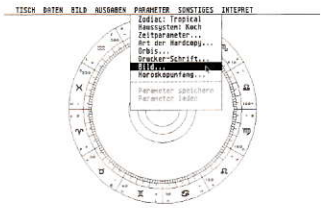
CBO



Gesamtmodells und Einstellwinkel-überprüfung am fertigen Modell (s/w).

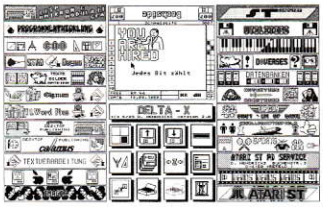


VOKAV plus 1.0: Programm zum Lernen und Archivieren von Vokabeln aller Sprachen. Die 'intelligente' Abfrage sorgt dafür, daß reine Tippfehler nicht als falsche Anwort gewertet werden, sofern die Eingabe im Kern richtig ist (s/w).

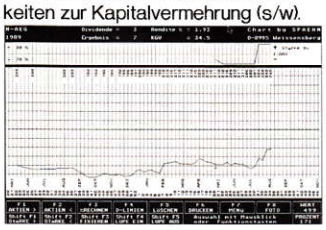


Astro: Grafisch sehr eindrucksvoll aufgemachtes Astrologieprogramm mit vielen Funktionen (s/w).
Lotto/Toto Systemwetten werden mit 'SYSTEM.PRG' auf einfache und komfortable Weise eingegeben, getestet und nach angenommenen Quoten ausgewertet (s/w).
Lineare Algebra und **Fakultät**: Nützliche Matheprogramme (s/w).

2059



Delta X 2.0: Neue Version des beliebten Etikettendruckers (s/w, S).
Inventur 0.9 hilft bei der Bestandsaufnahme. Die vorliegende Version verwaltet einen Geschäftsbereich mit fünf Firmen und fünf Sortimenten mit je fünf Artikeln (s/w, S).
SP-Börse - Charts*Analyse* Portfolio - ein Programm für spekulative Anleger. Börse verfügt über logisch nachvollziehbare Analysemöglichkeiten zur Kapitalvermehrung (s/w).



Lohn- & Einkommensteuer '88: Berechnungsprogramm (s/w, S).
Tiluie: Adressverwaltung (s/w).

2060

Super Boot 5.5: Neue Version des Bootprogramms, mit dem Auto-Programme und Accessories auf einfache Weise beim Systemstart geladen werden können (S).
Automat: Kombination aus ACC-Loader und Autobooter.
Bonus: GFA-Utility, zur Überprüfung der Grafikbefehle unter GFA-Basic.
Font Konverter für DEGAS- Fonts in Systemfonts und umgekehrt. Mit Fontedit (PD 189) erstellte Schriften können damit in DEGAS-Format übertragen und in STAD oder Tempus eingesetzt werden (s/w).
1200/75: Software-Support für den BTX-Modus.
Text-Add verbindet die vom BTX-Manager seitenweise abgelegten Textdateien zu einem Dokument.
Tiny Boot versüßt den Bootvorgang, indem es ein zufällig gewähltes Bild auf's Desktop projeziert.
Quick ST 1.0 beschleunigt die Textausgabe unter TOS 1.0 (80%), TOS 1.2 und 1.4 (25%).
HD-Shell: Neue Oberfläche für Festplattenbesitzer, von der aus bis zu 18 Programme automatisch aufgerufen werden können.
Encrypt schützt Dateien vor fremdem Zugriff (s/w, e).
Slow-Down stellt den ST auf Zeitlupentempo ein.

PD-Szene schon gelesen?

Jetzt mit Update-Seiten zur P.D. Fibel.

PD-Pool sucht noch einige gute Programme zur Veröffentlichung auf den Disketten 2061 - 2070. Die Vorstellung erfolgt gleichzeitig in mehreren großen ST- und PD-Zeitschriften.

Die vorgestellten Disketten erhalten Sie exklusiv bei folgenden PD-Anbietern:

HD Computertechnik
Pankstr. 42
1000 Berlin 65
030/4657028-29

trendDATA
Am Marstall 18-22
3000 Hannover 1
0511-166051

Intersoft
Nohlstr. 76
4200 Oberhausen 1
0208-809014

Computer Software Markert
Balbachtalstr. 71
6970 Lauda 9
09343/3854

=PD-EXPRESS= J. Rangnow
Ittlinger Straße 45
7519 Eppingen-Richen
07262/5131 (ab 17 Uhr)

V.U. - Volker Uecker
Hohenkamp 2
2308 Preetz
Tel: 04342-83842

H&S Wohlfahrtstätter
Irenenstr. 76c
4000 Düsseldorf 30
Tel: 0211-429876

Schreiber Software
Josefstr. 27
5120 Herzogenrath
02406-3223

Weeske Computer
Potsdamer Ring 10
7150 Backnang
07191/1528-29 od. 60076

Software-Service Duffner
Ritterstr. 6
7833 Endingen a. K.
07642-3875

ST Profi-Partner
Mönkhofer Weg 126
2400 Lübeck
Tel: 0451/505367

OHST-Software
Nelkenstr. 2
4053 Jüchen 2
Tel: 02164/7898

IDL Software
Lagerstraße 11
6100 Darmstadt 13
06151-58912

Gauger Software
Buhlstr. 16a
7505 Ettlingen
Tel: 07243 - 31828

LAUTERBACH Software
Josephsplatz 3
8000 München 40
089-2722377

T.U.M.-ST-Soft
Postfach 1105
2905 Edewecht
Tel: 04405/6809

Elektronikversand Michiels
Leloh 24
4056 Schwalmtal
Tel: 02163/4187

Computer Treff
Nettelbeckstr. 12
6200 Wiesbaden
06121-404302

iks
Schönblickstr. 7
7516 Karlsbad 4
ab 18.00 Uhr 07202/6793

Graf & Schick EDV
Hauptstraße 32a
8542 Roth
Tel: 09171/5058-59

Händleranfragen erwünscht: Wir suchen noch PD-Anbieter und Fachhändler für gemeinsame Projekte und Veranstaltungen im PD-Bereich.

☐ Scheck über DM liegt bei, ich erhalte die Ware verpackungs- und versandkostenfrei (Ausland: Bitte Euroscheck).

☐ Per Nachnahme. Nur Inland! (zuzüglich DM 6,- Nachnahmegebühr).

☐ 2001 ☐ 2011 ☐ 2021 ☐ 2031 ☐ 2041 ☐ 2044 ☐ 2047 ☐ 2052
☐ 2002 ☐ 2012 ☐ 2022 ☐ 2032 ☐ 2042 ☐ 2045 ☐ 2048 ☐ 2053
☐ 2003 ☐ 2013 ☐ 2023 ☐ 2033 ☐ 2043 ☐ 2046 ☐ 2054
☐ 2004 ☐ 2014 ☐ 2024 ☐ 2034 ☐ 2055
☐ 2005 ☐ 2015 ☐ 2025 ☐ 2035 ☐ 2056
☐ 2006 ☐ 2016 ☐ 2026 ☐ 2036 ☐ 2057
☐ 2007 ☐ 2017 ☐ 2027 ☐ 2037 ☐ 2058
☐ 2008 ☐ 2018 ☐ 2028 ☐ 2038 ☐ 2059
☐ 2009 ☐ 2019 ☐ 2029 ☐ 2039 ☐ 2060
☐ 2010 ☐ 2020 ☐ 2030 ☐ 2040

☐☐☐☐☐☐

Weitere PD-Disks bitte hier mit Nummer angeben.

TeX Paketangebot:
Programm + Druckertreiber.
Beschreibung in Heft 10/89, bzw. PD-Szene 2 und 3.

☐ TeX + C 16,-
☐ TeX + D 16,-
☐ TeX + E 20,-
☐ TeX + D 20,-
☐ TeX + G 20,-
☐ TeX + H 16,-
☐ TeX + J 16,-
☐ 2051 10,- TeX - Metafont

Diskpreis: 8,- DM

unverbindlich empfohlener Verkaufspreis

Lieferung an meine Adresse:

STC 11/89

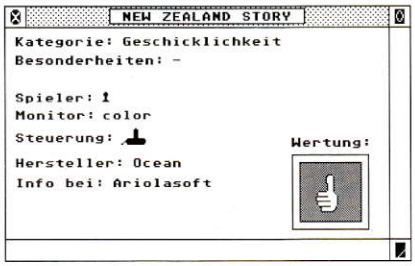
Gewünschte Disketten ankreuzen und Bestellschein an einen der oben angegebenen Anbieter einsenden.

New Zealand Story

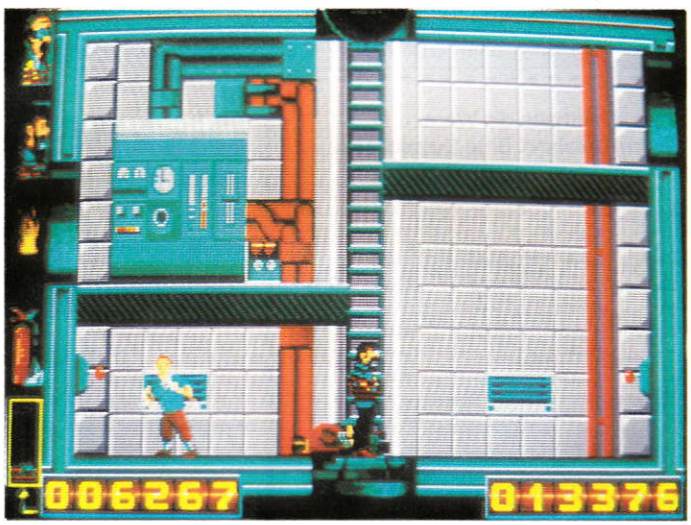


Ocean steht für knallharte Kriegsspiele, die in Deutschland schnell auf dem Index landen. Mit **"New Zealand Story"** schlägt der Hersteller jetzt in eine ganz andere Kerbe: Protagonist ist nicht ein meuchelnder Elitesoldat, sondern ein putziges Küken namens Tiki. Das niedliche Tierchen steckt in großen Schwierigkeiten: Ein biestiger Bösewicht hat Tikis Kameraden entführt und an 20 verschiedenen Orten im Auckland Zoo versteckt. Als geschickter Spieler müssen Sie Tiki durch 20 Spielabschnitte steuern und jedesmal ein Küken befreien. Das ist gar nicht leicht! In jedem Level wimmelt es nur so von Gestalten, die dem Helden ans Federkleid wollen. Da gibt es Ameisen mit Bumerang, feuerspeiende Vulkane, schießwütige braune Bären mit blauer Sonnenbrille und vieles mehr. Weichen Sie den Schüssen aus und vermeiden Sie Berührungen. Um den vielen Gegnern habhaft zu werden, ist Tiki mit Pfeil und Bogen bewaffnet. Trifft er damit einen Widersacher, bleiben Bonusfrüchte oder Extrawaffen auf dem Bildschirm zurück, die man einsammeln sollte. Mit Bomben oder Wasserpistole erhöht sich die Chance, auch einmal höhere Spielabschnitte kennenzulernen. Für Abwechslung ist gesorgt: "New Zealand Story" steckt voller Überraschungen. Mal springt der Held über Plattformen, dann fliegt er mit einem Ballon oder schwimmt im Taucheranzug. Ich habe viele Tage vor dem Monitor verbracht, um Tikis Freunde zu befreien. Oceans neues Spiel ist wie eine Droge. Man fängt einmal an und kommt schwer davon los. Das Spielprinzip macht total süchtig! "New Zealand Story" ist zusammen mit "Giana Sisters" das beste Geschicklichkeitsspiel, das es derzeit auf dem Markt gibt. Fans dieses Genres sollten unbedingt zugreifen! Es lohnt sich!

CBO



Tim und Struppi auf dem Mond

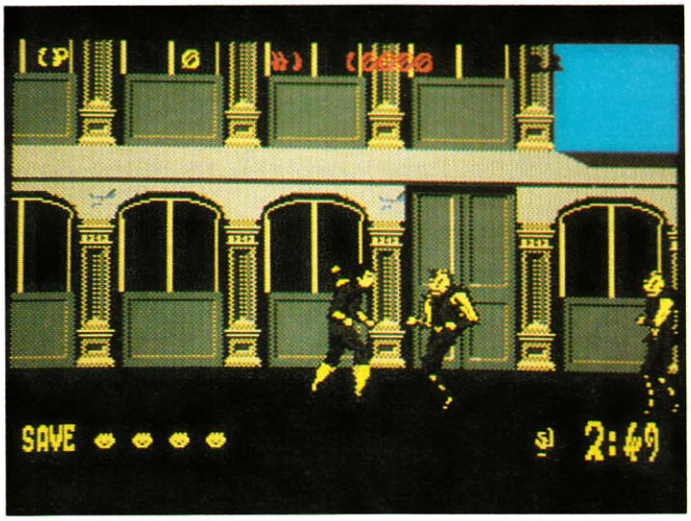


Tim und Struppi, die beiden beliebten Comic-Helden, erleben ihre Abenteuer jetzt auch auf dem Computer. Zu Beginn des Spiels zeigt der Computer einen hübschen Vorspann: Die Mondrakete steht an der Startrampe, Tanklastzüge schaffen Sprit heran, und die Techniker arbeiten emsig. Dann rast die Rakete gen Himmel. Im ersten Level steuern Sie die Mondrakete mit Maus oder Joystick durch den Weltraum. Riesige Asteroiden fliegen auf die Rakete zu. Da heißt es ausweichen. Neben Steinklumpen schwirren auch Bojen noch im All herum. Die gelben Bojen bringen zusätzliche Energie. Jetzt steuert man Tim durch die vielen Räume der Mondrakete. Für alle Probleme ist ein blinder Passagier verantwortlich. Dieser Gauner möchte aus irgendeinem Grund die Mondmission verhindern. Deshalb bringt er Sprengladungen an, legt kleine Feuerchen, schlägt Besatzungsmitglieder nieder und fesselt sie. Deshalb hat Tim alle Hände voll zu tun, die Rakete nach brennenden Dynamitstangen abzusuchen. Ist man in der Nähe einer Bombe, hört man zwar das Ticken einer Zeitzündung. Doch die Biester zu finden, ist ein Kunststück. Zwei- oder dreimal ist es mir gelungen, eine Sprengladung zu finden, als ich zufällig einen Raum betrat, in dem ein zusammengerolltes Dynamitbündel auf dem Boden lag. Abgesehen von der Tatsache, daß die Bomben schwierig zu finden sind, gibt es noch einen anderen merkwürdigen Umstand: Die Rakete besitzt viele Türen und Luken, die sich aber nicht öffnen lassen. Es gibt weder Schlüssel noch Schalter, mit denen man den einen oder anderen versperrten Zugang öffnen könnte. Neben dem Bombenentschärfen wollten Infogrames Programmierer den Spieler auch mit dem Löschen von kleinen Feuerchen unterhalten. Dazu findet Tim in der Rakete einen Feuerlöscher, mit dem er die Flammen ausblasen kann. Doch es kommen ständig neue nach. Die dritte spielerische Komponente liegt darin, gefesselte Besatzungsmitglieder durch einfaches Berühren zu befreien. Daß "Tim & Struppi" spielerisch nicht gerade Bieber vom Barhocker haut, dürfte klar geworden sein. Doch es gibt auch Positives zu vermelden: Die Grafik ist fein. Alle Comic-Stars sind einwandfrei zu erkennen.

CBO



SHINOBI



Endlich ist Shinobi erschienen. Um es kurz zu beschreiben: Sie, ein Ninja Warrior, einer der besten Schüler der geheimen Samurai-Schule, haben die Aufgabe, diverse Ninja-Schüler aus der Gewalt von Bwah Foo zu retten. Wie könnte es anders sein: Sie kämpfen sich in 5 Missionen durch verschiedenste Unholde, Schelme, Bösewichte und Schurken.

In der ersten Mission begegnen Sie Pistolettis und schwertwerfenden Böslingen. Sie starten, kärglich ausgestattet, mit Wurfsternen. Natürlich können Sie auch einfach drauf los prügeln, laufen dann aber Gefahr, in ein Schlachtermesser zu tapsen. Sie sehen eine Kanonenkugel auf sich zukommen? Ducken Sie sich, um auszuweichen. Die Kugel kommt von unten? Springen Sie 3 Meter hoch. In der ersten Mission müssen Sie 5 Schüler finden und das Ende des Levels erreichen. Dabei scrollt der Bildschirm horizontal von rechts nach links - erfreulicherweise, ohne groß zu ruckeln. Sollte die Situation einmal auswegslos sein, können Sie pro Level mit der Leertaste einen "Dschäi Aaahhhhr"-Schrei ausstoßen, dessen nachfolgender Wirbelsturm alle Schurken vom Bildschirm in die Jagdgründe befördert. Diesen Spell können Sie pro Level einmal benutzen. Wenn Sie einen Level gemeistert haben, erhalten Sie Bonuspunkte und werden zur nächsten Mission befördert, in der weitaus gefährlichere Unholde nur darauf warten, Sie gar grausig zu zerstückeln. Leider bleibt in allen Missionen die Spieltaktik immer gleich. Nachdem Sie den ersten Führer der Gegner, der mit Feuerbällen um sich wirft, besiegt haben, erwartet Sie eine Bonusrunde. Hier müssen Sie mit vielen Wurfsternen alle Gegner totwerfen, die waagerecht über den Bildschirm huschen (eine Mischung aus Space Invaders und Kegeln), um ein Bonusleben zu bekommen. In der dritten Mission begegnen Ihnen diverse Froschmänner (und Froschfrauen?), während Sie forellenartig über einen Wassergraben hüpfen. In einer anderen schwierigen Mission müssen Sie einen riesigen Helikopter zerstören (mit Wurfsternen...), der ziellos 2 Meter vor Ihnen herumfliegt. Vermeiden Sie dabei, in die Raketen zu laufen... Shinobi spielt man doch besser in der Spielhalle. Häßliche Farben, immer der gleiche Ablauf, und wenn man es einmal geschafft hat, verstaubt das Spiel in der Ecke.

ddf/MP



Rainbow Island



Es gibt Programme, die bringen einen ganz schön in Schwierigkeiten: Man läßt alle Termine sausen, geht nicht mehr ans Telefon und vergißt, seine Texte vor Redaktionsschluß abzugeben. Stattdessen sitzt man völlig fasziniert vor dem Monitor und ruckelt mit schweißgebadeten Fingern am Joystick. **"Rainbow Island"** ist so ein fatales Spiel. Sie steuern darin einen Jungen durch sieben Plattformwelten, die jeweils aus vier Spielabschnitten bestehen. Auf seinem Weg nach oben sammelt er schmackhafte Früchte, Diamanten und Extrawaffen. Manchmal sind zwei Plateaus zu weit auseinander. Dann setzt Bubby seine zauberhafte Geheimwaffe ein: den Magic Rainbow. Dummerweise verläuft der Weg über die Plattformen nicht ohne Komplikationen. Putzig gezeichnete Widersacher wollen Bubby an die Wäsche. Da gibt es grüne Raupen, Spinnen, herumballernde Marienkäfer, Flugzeuge mit Augen und Nase, die kleine Bomben abwerfen und vieles mehr. Kommt einer der zahlreichen Störenfriede zu nahe, kann man ihn mit dem magischen Regenbogen außer Gefecht setzen. Ist das Zeitlimit überschritten, erscheint die Meldung "Hurry up" auf dem Bildschirm. Dann sollte Bubby so schnell wie möglich die oberste Plattform erreichen, sonst ertrinkt er. Nach jeder vierten Spielstufe muß Bubby ein riesiges Monster mit seinen Regenbogengeschossen besiegen. Dann ist der Weg in die nächste Plattformwelt frei. Für lang anhaltenden Spielspaß ist gesorgt. Insgesamt sind auf den beiden ST-Disketten 28 unterschiedliche Spielstufen untergebracht. Spielerisch darf sich "Rainbow Island" zu den besten Geschicklichkeitsspielen zählen. Die Gegner sind nie link und lassen sich stets mit List, Tücke und einer Portion Spielpraxis beseitigen. Auch grafisch versteht "Rainbow Island" zu beeindrukken. Unzählige kunterbunte Sprites, die allesamt detailliert gezeichnet sind, verzücken das Auge. Meisterprogrammierer Andrew Braybrook ("Uridium") hat in "Rainbow Island" mit seinem Programmierteam "Craftgold" eine hervorragende Möglichkeit gefunden, sauberes, vertikales Scrolling auf dem ST zu realisieren. Dabei wird der Bildschirm bei jedem Sprung ohne Ruckeln nur ein wenig nach oben verschoben. Genial! So, mehr schreibe ich nicht. Ich will jetzt wieder "Rainbow Island" spielen.

CBO



Programmierte Logik

Teil 2

Einige Schaltungsbeispiele im GAL

Nachdem im letzten Teil der Artikelreihe die Grundlagen besprochen wurden, und man erahnen konnte, welche Möglichkeiten in diesen interessanten Bausteinen stecken, folgen diesmal einige Beispiele. Anhand von logischen Schaltungen werden die Möglichkeiten zu deren Implementierung in GALs aufgezeigt. Zuvor müssen wir uns aber noch ein wenig Theorie gönnen.

Wenn man ein GAL brennen will, muß man die entsprechende Fuse-Matrix in das Brennprogramm eingeben. Damit man nun beim späteren Brennen eines GALs mit demselben Inhalt nicht wieder die gesamte Matrix eintippen muß, wurde vom *Joint Electron Device Engineering Council* das JEDEC-Format als Standard bestimmt. Eine Fuse-Matrix, die sich in einer Datei mit diesem Format befindet, kann nun immer wieder zum Brennen eines Logikbausteins benutzt werden. Auch ist eine Weitergabe der Datei auf andere Rechner ohne Probleme möglich, da sie nur ASCII-Zeichen enthält. Ein weiterer Vorteil dieser Standardisierung ist, daß man solche Dateien auch mit anderer Brenn-Hard- und Software benutzen kann. Logik-Compiler (das sind Programme, die aus logischen Gleichungen oder Funktionstabellen Fuse-Matrixen erzeugen) benutzten dieses Format ebenfals, so daß man ausgegehend von den Logikgleichungen bis zum fertig gebrannten GAL jede beliebige Soft- und Hardware-Kombination (auch auf verschiedenen Rechnern) benutzen kann. Ich werde im weiteren nicht auf das komplette Format eingehen, da nur ein Teil zur Programmierung der GALs notwendig ist.

Das JEDEC-Format

Jeder Eintrag in einer JEDEC-Datei beginnt mit einem Steuerzeichen und endet mit dem Asterix (*). Als erstes steht in dem File eine **02** (Steuerzeichen: <STX>). Nun können beliebige Angaben folgen (z.B. Namen, Versionsnummern, Hinweise...), die später nicht ausgewertet werden (* am Ende nicht vergessen!). Die folgenden Datenfelder können in beliebiger Reihenfolge in der Datei stehen.

Fuse-Feld: Es besteht aus maximal drei Teilen. Wenn in der folgenden Fuse-Liste nicht für alle Sicherungen der jeweilige Zustand angegeben wird, muß der folgende für die weggelassenen aufgeführt werden. Hierzu werden der Kennbuchstaben **F** und die Ziffern **0** (Verbindung) oder **1** (Verbindung wird nicht hergestellt), gefolgt von einem Asterix, benutzt. Dies gibt dem Anwender die Möglichkeit, nicht alle Sicherungszustände in seiner Datei aufführen zu müssen. Genutzt wird dies dann, wenn man nicht alle Ausgänge eines GALs verwendet, oder aber, wenn bei einem Ausgang nicht alle acht Zeilen benötigt werden. Die ungenutzten Zeilen braucht man so nicht in das JEDEC-File zu schreiben.

```
Byte 00   7  6  5  4  3  2  1  0
Fuse Nr.

Byte 01  15 14 13 12 11 10  9  8
Fuse Nr.
  ⋮      ⋮  ⋮  ⋮  ⋮  ⋮  ⋮  ⋮  ⋮
Byte 12   -  -  -  -  99 98 97 96
Fuse Nr.

An die Stelle der Zahlen kommen
die entsprechenden 0er und 1er.
An die Stelle der - kommen 0er.
Die einzelnen Bytes werden zu
einem 16-bit Word aufaddiert.
```

Bild 1: Bildung der Fuse-List-Prüfsumme

Die eigentliche **Fuse-Liste** beginnt mit einem **L** und der Adresse der ersten Sicherung, für die folgende, fortlaufende Reihe gilt. Hier könnnen nun die Zustände für alle Sicherungen stehen, oder aber man teilt die Liste in mehrere Teile auf (jeder muß jedoch mit * enden und der nächste mit **L** und einer Adresse anfangen). Teilt man die Liste so auf, daß in jeder Reihe 32 (bzw. 40 beim GAL 20v8) Zustände stehen und sich die nächste Reihe bündig darunter befindet, so läßt sich leicht ersehen, welche Verbindungen hergestellt werden sollen. Man nimmt dazu die Bilder 2 und 3 aus der letzten Folge und vergleicht die Logikmatrix mit den Nullen aus der Liste. An diesen Stellen wird eine Verbindung zwischen den Zeilen und Spalten hergestellt.

Der dritte Teil ist eine Prüfsumme. Diese wird gebildet indem man die Fuse-Liste, wie in Bild 1 zu sehen, in Bytes aufteilt und diese aufaddiert (das letzte Byte wird, sofern die Anzahl der Sicherungen nicht durch acht teilbar ist, mit Nullen aufgefüllt). Das Ergebnis wird als 16-bit Zahl hinter den Buchstaben **C** notiert.

Sicherungsfeld: Eine weitere Angabe bezieht sich auf die **Security Cell**. Soll diese gebrannt werden (Fuse-Matrix des GALs ist danach nicht mehr auslesbar), schreibt man **G0***, im anderen Fall **G1***.

Kommentarfeld: Es besteht auch die Möglichkeit, Kommentare innerhalb der JEDEC-Datei unterzubringen. Der Kennbuchstabe hierfür ist **N**.

Am Schluß der Datei kommt noch eine Prüfsumme. Sie wird von dem Steuerzeichen <ETX> (Hex: **03**) eingeleitet und ist eine 16-Bit-Summe aller ASCII-Zeichen des Files einschließlich der Steuerzeichen <STX> und <ETX>. Da manche Computer jedoch am Ende jeder Zeile noch Zeichen einfügen (Return...), wird in vielen Fällen an dieser Stelle statt der Prüfsumme einfach nur **0000** geschrieben. Die Datei endet direkt hinter der Zahl; es folgt hier **kein** *!

Nachdem nun klar ist, wie das JEDEC-Format aussieht, taucht natürlich die Frage auf, welche Adresse die Sicherungen in den GALs haben. Die Adresse der Logikmatrix läßt sich sehr einfach anhand der Bilder 2 und 3 aus der letzten Ausgabe (S. 165 oben) errechnen. Man muß die Nummer des Produkt-Terms mal der Anzahl der Spalten des GALs (beim 16v8 32, beim 20v8 40) nehmen und die **Row-Adresse** der Zeile, in der sich die Sicherung befindet dazuaddieren. [An dieser Stelle sei auf eine Tatsache hingewiesen, die zu Unstimmigkeiten führen kann: In den Datenbüchern über GALs (auch bei Lattice) beginnen die Produkt-Terme beim GAL 20v8 bei acht. Da sich bei der von mir gewählten Numerierung jedoch die Adressen im JEDEC-File leichter errechnen lassen habe ich diese auch für den Artikel übernommen.] Die übrigen Adressen entnimmt man Bild 2.

Fuse-Adressen für GAL 16v8	Fuse-Adressen für GAL 20v8	Funktionen (Bez. der Sicherungen)
0000-2047	0000-2559	Fuse-Matrix (Produktterme)
2048-2055	2560-2567	XOR-Bits der Ausgangszellen
2056-2119	2568-2631	Signatur (Versionsnr. o.ä.)
2120-2127	2632-2639	AC1-Bits der Ausgangszellen
2128-2191	2640-2703	Produkttermfreigabe
2192	2704	SYN-Bit der Ausgangszellen
2193	2705	AC0-Bit der Ausgangszellen

Bild 2: Adressen der GALs in der Fuse-Liste des JEDEC-Formats

Bei den folgenden Beispielen werde ich jeweils nur die Fuse-Liste angeben [alle nicht aufgeführten Sicherungen werden gebrannt (0)].

Logische Gatter im GAL

Die Realisierung der logischen Grundgatter mittels eines GALs ist in Bild 3 zu sehen. Das NOT ergibt sich durch die Aufteilung der Eingänge in einen normalen und einen invertieren Eingang (3a). UND-verknüpft werden all die Eingänge, deren Sicherungen auf einer Zeile gebrannt wurden (3b). Alle Zeilen, die denselben Ausgangstreiber haben, werden untereinander ODER-verknüpft (3c). Die EXOR-Verknüpfung (exklusiv oder: der Ausgang wird nur dann Eins, wenn genau Einer der Eingänge eins ist) muß auf eine UND/ODER-Verknüpfung zurückgeführt werden (3d).

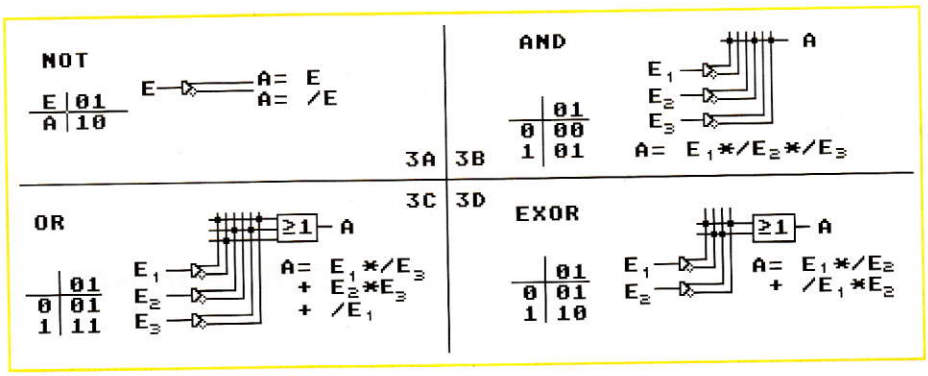


Bild 3: Logische Gatter im GAL

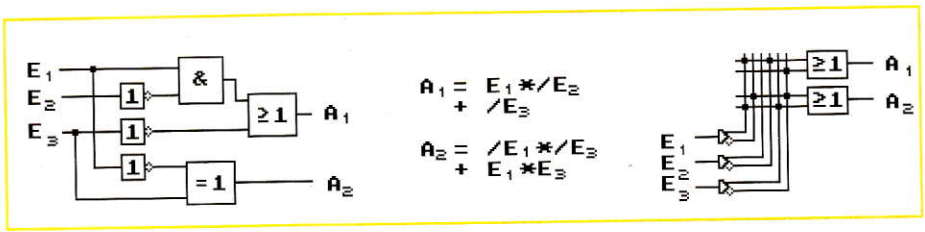


Bild 4: Verknüpfungsnetz mit TTL-Logik und im GAL

In Bild 4 ist die Übertragung eines logischen Verknüpfungsnetzes in ein GAL wiedergegeben. Zuerst werden anhand des Netzes die Logikgleichungen aufgestellt. Nun (sofern man nicht einen Logik-Compiler zur Hand hat) werden diese vereinfacht, was bei diesem Beispiel entfällt. Der nächste Schritt ist zu Überlegen, wie man diese Gleichungen am günstigsten in einem GAL unterbringt. In unserem Fall brauchen wir drei Eingänge und zwei Ausgänge. Da bei den Gleichungen maximal zwei ODER-Verknüpfungen vorkommen, benötigen wir Ausgänge, die mindestens zwei Zeilen verODERn. In diesem Fall haben wir beim GAL (jeder Ausgang hat acht Eingangszeilen) keine Schwierigkeiten. Hätten wir jetzt eine Gleichung mit mehr als acht ODER-Verknüpfungen (nach der Vereinfachung), hätten wir mit einer Rückführung arbeiten müssen. D.h. wir hätten einen Teil der Gleichung auf einem Ausgang realisieren, dieses Ergebnis auf den nächsten Ausgang zurückführen und dort die restliche Gleichung implementieren müssen. Dadurch wäre natürlich ein Ausgang verlorengegangen [es sei denn, wir hätten das Ergebnis der Zwischengleichung benötigt; z.B. wenn eine Gleichung vollständig in einer weiteren enthalten ist: A = X * Y und B = (X * Y) + Z. In diesem Fall hätte man A auf einem Ausgang realisiert und das Ergebnis auf die Zeilen vom Ausgang B zurückgeführt. Wobei es jedoch aus Geschwindigkeitsgründen auch sinnvoll sein kann, die gleichen Terme für jeden Ausgang neu zu realisieren, denn durch die Rückführung liegt am Ausgang B erst nach der doppelten Zeit wie am Ausgang A das richtige Signal an]. Jetzt müsen wir uns nur noch entscheiden, welche Ein- und Ausgänge wir am GAL belegen. Die Aufstellung der Fuse-Matrix ist bei diesem Beispiel sehr einfach. Sie wird anschließend in den Baustein gebrannt, und wir haben unsere Logik im GAL. Sollte sich nun nachträglich an der

Logik etwas ändern (z.B. ein ODER statt einem EXOR), ändern wir die entsprechenden Stellen in der Fuse-Matrix und brennnen das GAL neu. Es muß nicht ein neues Layout für die Platine realisiert werden.

Ein Zähler im GAL

Bei unserem nächsten Beispiel wird die Sache schon um einiges schwieriger. Wir wollen einen 4-Bit-Zähler realisieren, der nur aufwärts zählt, einen Takt, einen Rücksetz- (clear) sowie einen Setzeingang (preset) besitzt, mit dessen Hilfe man den Zähler auf einen bestimmten Anfangswert bringen kann (wird bei uns jedoch nicht verwendet, ist schmückendes Beiwerk).

```
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
```

Bild 5: So wird digital gezählt

Zuerst stellen wir die Logikgleichungen auf. Dazu überlegen wir uns, was ein Zähler eigentlich macht. Richtig, er zählt. Aber wie? Bild 5 gibt darüber Auskunft. Dort sind alle mit 4 Bit darstellbaren Zahlen der Größe nach sortiert (von oben nach unten) aufgeführt. Diesen Zyklus soll das GAL an vier seiner Ausgänge (A_3 bis A_0) wiedergeben, vorausgesetzt die Eingänge **clear** und **preset** sind beide inaktiv (low). Mittels dieser Informationen können wir nun die Gleichungen für die vier Ausgänge aufstellen. Zuerst für A_0. Der Ausgang wechselt jedesmal wenn ein Taktimpuls kommt, seinen Wert von 0 auf 1 und umgekehrt. Da nun die von den Logikzeilen erzeugten Signale erst beim nächsten Taktimpuls übernommen werden (wir arbeiten mit Registern in den **OLMCs**), können wir den neuen Zustand erzeugen sowie der vorhergehende übernommen wurde. Bei A_0 heißt dies, daß wir den Ausgang immer invertieren:

$$A_0 = /A_0$$

Wenn clear gesetzt wird, muß der Ausgang (wie alle anderen auch) auf 0 gesetzt werden (in unserem Fall erst beim nächsten Taktimpuls). Wir beachten dies, indem wir bei jedem UND-Term in unseren Gleichungen ein /clr schreiben:

$$A_0 = /clr * /A_0$$

Das gleiche gilt für preset, nur muß hier der Fall, daß der Ausgang gesetzt werden soll [es liegt am entsprechenden Eingang (E0 bis E3) eine 1], mitbeachtet werden. Bei uns ist das immer der letzte UND-Term jeder Gleichung. Dies ergibt für den ersten Ausgang die vollständige Logikgleichung:

$$\begin{aligned} A_0 &= /clr * /ps * /A_0 \\ &+ /clr * ps * E_0 \end{aligned}$$

Der zweite Ausgang wird bei den Übergängen 01 -> 10 und 10 -> 11 aktiv (1), so daß wir für diesen drei UND-Terme verODERn müssen:

$$\begin{aligned} A_1 &= /clr * /ps * /A_1 * A_0 \\ &+ /clr * /ps * A_1 * /A_0 \\ &+ /clr * ps * E_1 \end{aligned}$$

Beim nächsten Ausgang sind es bereits vier Übergänge: 011 -> 100, 100 -> 101, 101 -> 110 und 110 -> 111. Dies ergibt fünf VerODERungen:

$$\begin{aligned} A_2 &= /clr * /ps * /A_2 * A_1 * A_0 \\ &+ /clr * /ps * A_2 * /A_1 * /A_0 \quad (i) \\ &+ /clr * /ps * A_2 * /A_1 * A_0 \quad (i) \\ &+ /clr * /ps * A_2 * A_1 * /A_0 \\ &+ /clr * ps * E_2 \end{aligned}$$

Der letzte Ausgang beschert uns mit seinen acht Übergängen (0111 -> 1000, 1000 -> 1001, 1001 -> 1010, 1010 -> 1011, 1011 -> 1100, 1100 -> 1101, 1101 -> 1110 und 1110 -> 1111) neun UND-Terme:

$$\begin{aligned} A_3 &= /clr * /ps * /A_3 * A_2 * A_1 * A_0 \\ &+ /clr * /ps * A_3 * /A_2 * /A_1 * /A_0 \quad (ii) \\ &+ /clr * /ps * A_3 * /A_2 * /A_1 * A_0 \quad (ii) \\ &+ /clr * /ps * A_3 * /A_2 * A_1 * /A_0 \quad (iii) \\ &+ /clr * /ps * A_3 * /A_2 * A_1 * A_0 \quad (iii) \\ &+ /clr * /ps * A_3 * A_2 * /A_1 * /A_0 \quad (iv) \\ &+ /clr * /ps * A_3 * A_2 * /A_1 * A_0 \quad (iv) \\ &+ /clr * /ps * A_3 * A_2 * A_1 * /A_0 \\ &+ /clr * ps * E_3 \end{aligned}$$

Die Gleichung für den letzten Term bereitet uns bei der Realisierung im GAL Schwierigkeiten, da man für jeden Ausgang nur acht ODER-Verknüpfungen hat. Zum Glück kann man sich bei diesem Beispiel jedoch mittels Vereinfachungen der Geichungen helfen. Könnte man das nicht machen (z.B. bei einem 5-Bit-Zähler), müßte man einen weiteren Ausgang heranziehen und, wie oben bereits erwähnt, die Gleichungen auf die beiden Ausgänge verteilen und mit einer Rückführung arbeiten. Bei unseren Zählern kann man jedoch sowohl bei A_3 als auch bei A_2 die Gleichungen vereinfachen. Schaut man sich die Gleichungen von A_2 näher an, sieht man, daß sich die beiden UND-Terme vor der (i) nur minimal unterscheiden. Sie sind bis auf den Teil mit A_0 identisch. Da dies jedoch heißt, daß sowohl dann, wenn A_0 gesetzt ist, als auch, wenn der Ausgang nicht gesetzt ist, diese UND-Bedingung erfüllt werden soll, kann man die beiden Zeilen zusammenfassen, indem man den A_0-Teil wegläßt. Dies gibt eine Ersparnis von einem ODER:

$$\begin{aligned} A_2 &= /clr * /ps * /A_2 * A_1 * A_0 \\ &+ /clr * /ps * A_2 * /A_1 \quad (i) \\ &+ /clr * /ps * A_2 * A_1 * /A_0 \\ &+ /clr * ps * E_2 \end{aligned}$$

Beim letzten Ausgang ist die Ersparnis sogar noch größer. Man kann nach derselben Methode wie oben die Zeilen bei (ii), (iii) und (iv) zusammmenfassen. Des weiteren kann man danach noch einmal die Zeilen, welche man bei (ii) und (iii) erhalten hat, zusammenfassen, wobei dort A1 wegfällt. So hat man am Ende vier ODER-Verknüpfungen gespart und kann die Gleichung mit einem Ausgang realisieren:

$$\begin{aligned} A_3 &= /clr * /ps * /A_3 * A_2 * A_1 * A_0 \\ &+ /clr * /ps * A_3 * /A_2 \quad (ii \,\&\, iii) \\ &+ /clr * /ps * A_3 * A_2 * /A_1 \quad (iv) \\ &+ /clr * /ps * A_3 * A_2 * A_1 * /A_0 \\ &+ /clr * ps * E_3 \end{aligned}$$

Zusätzlich soll nun unser Zähler beim Erreichen der Zahl 11 (1011) wieder von 0 an anfangen. Dies ergibt als letzte Gleichung für clear (wobei clear_out der von außen kommende clear-Eingang sei):

$$\begin{aligned} clr &= clear_out \\ &+ A_3 * /A_2 * A_1 * A_0 \end{aligned}$$

In Bild 6 ist die Realisierung mittels TTL-Bausteinen wiedergegeben. Man braucht

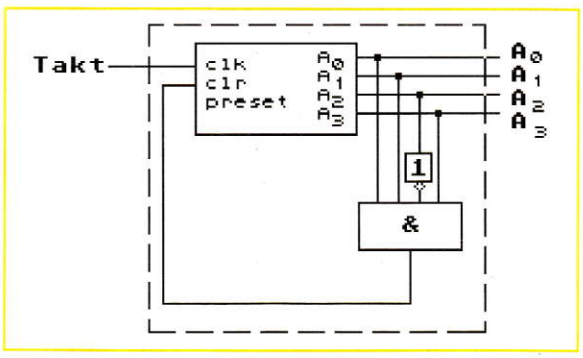


Bild 6: Der 4-Bit-Zähler in TTL

einen Zähler, einen Inverter und ein 4er UND. Der innerhalb der gestrichelten Linien befindliche Bereich ist nun die Logik, die wir in unser GAL brennen (Fuse-Liste siehe Bild 7). Man erkennt auch hier: Bei Änderungen der Zahl, ab der der Zähler wieder von vorne anfangen soll, braucht man nur das GAL neu zu programmieren, bei der TTL-Lösung bräuchte man für die Rückführung zum clear-Eingang eine neue Logik samt Verdrahtung.

Ein GAL als Ampel?

Als letztes Beispiel etwas zum Thema Zustandsautomaten. Wir wollen ein GAL zur Steuerung einer Verkehrsampel einsetzen. Wir gehen davon aus, daß der Takt zum Wechseln der Ampelphasen von außen (von einer Zeitgeberschaltung) gegeben wird. Als erstes haben wir uns zu fragen, welche Zustände es gibt. Klar, wie bei jeder Ampel: rot, rot-gelb, grün, gelb und wieder rot, wobei der letzte Zustand schon wieder der erste ist. Wir haben also vier Zustände. Zur Ansteuerung der einzelnen Ampellampen sehen wir jeweils einen Ausgang vor (rot, gelb und grün). Des weiteren numerieren wir die Ampelphasen von 0 bis 3 (binär) durch, wobei wir bei irgendeinem Zustand (hier: Ampel rot) beginnen. Wir erhalten dann folgende Tabelle , die zyklisch (d.h. immerfort) durchlaufen wird.

Zustands-nummer		Ampelfarben rot	gelb	grün
(0) 0	0	1	0	0
(1) 0	1	1	1	0
(2) 1	0	0	0	1
(3) 1	1	0	1	0
z_1	z_0	r	y	g

```
<STX>
Beispiel: 4-bit Zaehler im GAL 16v8
Die Ausgaenge haben Register.
Pinbelegung:

Eingaenge:  Ausgaenge:
1: Takt     19: interne Erzeugung von clear
2: clear    18: A3
3: preset   17: A2
4: E3       16: A1
5: E2       15: A0
6: E1
7: E0*

N Der clear-Ausgang:

  Die Belegung der Spalten (Achtung: die Rueck-
  fuehrungen der Ausgaenge von den Registern
  erfolgt invers!):
      cocl psa3 e3a2 e2a1 e1a0 e0-- ---- ----*
L0000 0111 1110 1101 1110 1110 1111 1111 1111
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000
      0000 0000 0000 0000 0000 0000 0000 0000*

N In den nichtbenutzen Zeilen werden alle
  Sicherungen gebrannt, da diese dann nicht
  beim ODER stoeren.

  Ausgang A3:*

L0256 1101 1001 1110 1110 1110 1111 1111 1111
      1101 1010 1101 1111 1111 1111 1111 1111
      1101 1010 1110 1101 1111 1111 1111 1111
      1101 1010 1110 1110 1101 1111 1111 1111
      1101 0111 0111 1111 1111 1111 1111 1111*

N Ab hier lasse ich die restlichen Zeilen die
  nur gebrannte Sicherungen enthalten weg

  Ausgang A2:*

L0512 1101 1011 1101 1110 1110 1111 1111 1111
      1101 1011 1110 1101 1111 1111 1111 1111
      1101 1011 1110 1110 1101 1111 1111 1111
      1101 0111 1111 0111 1111 1111 1111 1111*

N Ausgang A1:*

L0768 1101 1011 1111 1101 1110 1111 1111 1111
      1101 1011 1111 1110 1101 1111 1111 1111
      1101 0111 1111 1111 0111 1111 1111 1111*

N Ausgang A0:*

L1024 1101 1011 1111 1111 1101 1111 1111 1111
      1101 0111 1111 1111 1111 0111 1111 1111*

N ACW:*

L2048 11111111
      00000000000000000000000000000000
      00000000000000000000000000000000
      00000000
      11111111111111111111111111111111
      11111111111111111111111111111111
      0
      1*
<ETX>0000
```

Bild 7: Fuse-Liste des 4-Bit-Zählers

```
<STX>
Beispiel: Ampelsteuerung im GAL 16v8
Die Ausgaenge haben Register.
Pinbelegung:

Eingaenge:  Ausgaenge:
1: Takt     19: z1
            18: z0
            17: r
            16: y
            15: g*

N Zustandvariable z1

  Die Belegung der Spalten (Achtung: die Rueck-
  fuehrungen der Ausgaenge von den Registern
  erfolgt invers!):
      --z1 --z0 ---- ---- ---- ---- ---- ----*
L0000 1101 1110 1111 1111 1111 1111 1111 1111
      1110 1101 1111 1111 1111 1111 1111 1111*

N Zustandvariable z0*

L0256 1111 1101 1111 1111 1111 1111 1111 1111*

N Ampelfarbe rot:*

L0512 1101 1101 1111 1111 1111 1111 1111 1111
      1101 1110 1111 1111 1111 1111 1111 1111*

N Ampelfarbe gelb:*

L0768 1101 1110 1111 1111 1111 1111 1111 1111
      1110 1110 1111 1111 1111 1111 1111 1111*

N Ampelfarbe gruen:*

L1024 1110 1101 1111 1111 1111 1111 1111 1111*

N ACW:*

L2048 11111111
      00000000000000000000000000000000
      00000000000000000000000000000000
      00000000
      11111111111111111111111111111111
      11111111111111111111111111111111
      0
      1*
<ETX>0000
```

Bild 8: Ein GAL als Ampelsteuerung

Nun werden die Gleichungen für die beiden Zustandsvariablen aufgestellt [in diesem Fall, da keinerlei Einwirkungen von außen auf den Zyklus erfolgen können, hat man einen 2-Bit-Zähler. Wenn Einflüsse da sind, müssen die entsprechenden Eingänge bei der Aufstellung der Gleichungen berücksichtigt werden (siehe Beispiel im Handbuch zum GAL-Prommer)]:

$$z_0 = /z_0$$

$$z_1 = /z_1 * z_0 + z_1 * /z_0$$

Die Gleichungen für die Ausgänge dürfen nur von den Zustandsvariablen abhängen. In unserem Fall ergibt sich:

$$r = /z_1 * /z_0 + /z_1 * z_0$$

$$y = /z_1 * z_0 + z_1 * z_0$$

$$g = z_1 * /z_0$$

Die Realisierung dieser Gleichungen im GAL (an zwei Ausgängen erzeugt man die Werte, die Zustandsvariablen und koppelt diese zurück) ist im Bild 8 zu sehen.

Die Beispiele in dieser Folge sollten einige Möglichkeiten aufzeigen, was man mit GALs alles machen kann. Sicherlich konnten es nur Stichproben sein. Am besten ist, man stellt selbst Versuche mit diesen Bausteinen an. Zu diesem Zweck folgt in der nächsten Folge das Selbstbauprojekt für einen GAL-Prommer. Auch werde ich dort das Thema Emulation von PALs durch GALs, das ich ja bereits für diese Folge angekündigt hatte, nachholen.

Thomas Werner

Literatur:

Lattice: GAL Handbook
mc 1/88:
Programmierbare Logikbausteine
elrad 5/89:
Die Logik durchgebrannter Sicherungen
JEDEC FORMAT, Datenbuch

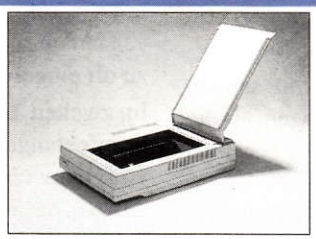
Nikolaistraße 2
8000 München 40
West-Germany

PRINT TECHNIK

Tel. 0049-89-368197
FAX: 0049-89-399770

Wir stellen aus: Systems München,





Neuer Superpreis: 998.-

zzgl. OCR-Schrifterkennung
Univ. Scanner, Drucker, Kopierer **1148.-**

Dieses mit 200 DPI arbeitende Bilderfassungsgerät ist die ideale Arbeitshilfe für alle Anwender, die über Geräte mit einem Mega-Speicher verfügen (1040, ein Mega, oder aufgerüstete Einheiten). Durch rationellste Produktionsmethoden und günstigen Einkauf des Thermo-Kopierers ist uns nochmals eine Preissenkung für dieses Gerät gelungen.

Alle Formate möglich / Calamus compatibel.

Ein absoluter Preishit für jeden ATARI-Nutzer.

***Professional Scanner* 2998.-**

mit OCR-Junior inkl. Ganzseitenmalprogramm ROGER PAINT OCR Junior selbstlernende Schrifterkennung PEGASUS + ST 1 Raster Vektor Konvertierungsprogramm
300 x 300, 300 x 600, 600 x 600 DPI-Auflösung und 64 Graustufen, einschl. Zeichenprogramm und OCR-Schrifterkennung.
Diese Scannerneuheit für den Industrie- und DTP-Bereich stellt einen absoluten Preishit dar. Mit ihm lassen sich sowohl Halbton als auch binäre Vorlagen scannen und ablegen und mit allen auf dem Markt befindlichen Programmen (auch Calamus) weiterverarbeiten.
Das mitgelieferte Schrifterkennungsprogramm erlaubt das Umsetzen von Text in ASCII-Zeichensatz und ist durch seine Lernfähigkeit von hoher Effektivität.

Videodigitizer PRO 8900 für ATARI
Der Videodigitizer PRO 8805 liefert die höchste Auflösung, die bei Verwendung einer normalen Videokamera möglich ist: 1024 Punkte in 512 Zeilen. Gleichzeitig digitalisiert er mit einer Genauigkeit von 7 Bit, was einer Anzahl von 128 Graustufen entspricht. Technische Daten des PRO 8805: Bildformate: Neochrome, IMG, Doodle, Spat. Ausdruck auf: NEC P6/P7, ATARI Laser. Auflösung: 320 x 200, 640 x 200, 640 x 400, 512 x 512, 1024 x 512. Graustufen: 128 (7 Bit). Anschluß: ROM-Port des ATARI ST. Eingangssignal: BAS oder FBAS. S/W und Farbmonitor.
Preis: DM 498,-

Neue Colorsoft von Imagic
16 Farben aus 4096/Zusatzsoft zum PRO 8900 **Preis: DM 98,-**

PRO 8900 mit RGB-Filter + Imagic Soft.
Der »Farb«-Digitizer **DM 748,-**

Realtizer für ATARI ST
Der REALTIZER ist ein in den ROM-Port einsteckbares Modul zur rasanten Digitalisierung von Videobändern aller Art. Die Auflösung beträgt 320 x 200 Punkte, wobei der Farb- und Monochrome-Modus (640 x 400) des ATARI ST unterstützt wird. Die Auflösung: 16 Graustufen. Pro Graustufe beträgt die Digitalisierungszeit 1/25 Sekunde.
Automatische Helligkeits- und Kontrastregelung. **Preis: DM 198,-**

OCR-Junior Schrifterkennung
Selbstlernende Schrifterkennung zu Universalscanner für ATARI ST.
Preis: DM 198,-

Romportstecker
Freier Druckerport beim Universalscanner. Ermöglicht Sofortausdruck z.B. mit NEC P6/P7. **Preis: DM 148,-**

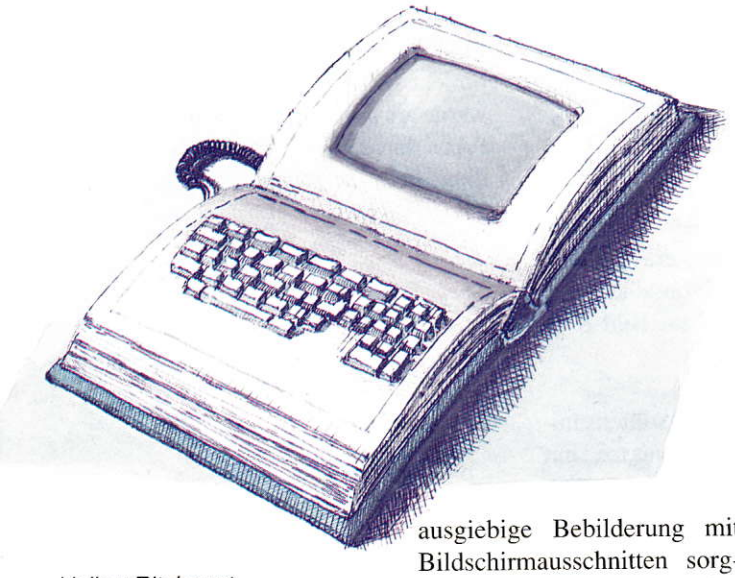
RGB Splitter
Der RGB-Splitter zerlegt jedes Farb-Videosignal in seine Grundfarben Rot, Grün und Blau. Mittels Drehschalter kann jede Grundfarbe und Schwarz/Weiß an einen Videoausgang geschaltet werden. Passend für alle Videodigitizer mit Farbdigitalisierungssoftware (z. B. PRO 8805).
Noch nie erreichte Farbbildqualität. **Preis: DM 248,-**

Videotext Dekoder
Zum Anschluß an den ROM-Port. Kann mit jedem Videosignal betrieben werden. Läuft auf Farb- oder S/W-Monitor. Seitenweises Aufrufen – Automatisches Blättern – Seiten halten – Speichern und Laden der empfangenen Seiten im Text- oder Bildformat – Textausdruckmöglichkeiten über beliebige Drucker. **Preis: DM 248,-**

Schweiz: Microtron 0041-32-872429 NL 010-4507696 VISA/Eurocard Accepted

VISA/Eurocard Accepted Austria: Print-Technik-Wien 0043-222-5973423

PRODUCTION – IMPORT – EXPORT – DISTRIBUTION



Volker Ritzhaupt

Das Atari-ST 1x1

Ein Leitfaden für Anfänger und Fortgeschrittene

Heidelberg 1989,
Application Systems
320 Seiten
DM 49,-
ISBN 3-9801834-3-2

Ritzhaupt will mit seinem Buch "mehr sagen, als die Bedienungsanleitung hergibt", wendet sich also an den Anfänger, der die ersten Schritte mit seinem ST machen will. Dementsprechend beschäftigt sich der erste größere Teil des Buchs mit dem Desktop, Fenstern und Mausfunktionen. Es geht weiter mit einer ausführlichen Einführung in die grundlegenden Funktionen wie Dateien kopieren, löschen und umbenennen sowie Formatieren.

Je ein spezielles Kapitel haben die Fileselectbox und die Installation und Benutzung der Festplatte erhalten. Die von ATARI mitgelieferten Systemprogramme, wie das Control-Accessory, werden anschließend beschrieben, worauf der erste Teil des Buches mit einigen kleinen Tips zur ST-Benutzung endet.

Diese cirka 200 Seiten können sehr gut die Original-Anleitung ersetzen. Sie sind für den Anfänger geschrieben, gehen auf mögliche Fehlerquellen ein und leiten den Anfänger durch ausgiebige Bebilderung mit Bildschirmausschnitten sorgfältig durch die ersten Stunden mit dem ST.

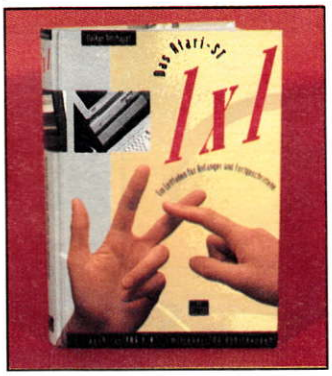
Unterstützt wird der Text durch Symbole am Rand, die z.B. darauf hinweisen, daß hier ein warnender Hinweis oder eine Tastatureingabe zu finden sind. Die Systematik der Symbole leidet etwas durch zu viele verschiedene Sinnbilder, die zu oft eingesetzt werden.

Im zweiten Teil gibt Ritzhaupt Empfehlungen für den Kauf von Anwenderprogrammen und will das vorhandene Angebot durch Empfehlungen für die verschiedenen Bereiche dem Einsteiger näherbringen. Der Anwender wird aufgeklärt über den Kauf von Programmen und den Support, der per Registrierungen geboten wird. Schließlich fehlt auch nicht die Anmahnung, Raubkopien nicht zu benutzen.

Die Auswahl der Anwendersoftware kann leider keine konkrete Hilfestellung bieten, dafür sind die Beschreibungen zu kurz und die Auswahl zu unvollständig. Auch kann nicht verleugnet werden, daß sich die Produktpalette von Application Systems fast komplett wiederfindet und bestimmte wichtige Konkurrenzprodukte nicht zu finden sind. Dieser cirka 30 Seiten starke Teil kann also keineswegs die Informationen aus Fachzeitschriften ersetzen und wird auch in absehbarer Zeit deutlich an Aktualität verloren haben.

Die restlichen 70 Seiten dienen als Nachschlageteil mit Glossar und Register. Im Glossar werden cirka 160 Begriffe in wechselndem Umfang dargestellt. Die Schlagworte reichen von Absturz bis Zugriffspfad und stellen eine annehmbare Auswahl dar, auch wenn sie lange nicht vollständig sind. Einige Bereiche sind inkonsequent abgedeckt, so erscheint zwar die eher unübliche Abkürzung "PCL" für die Seitenbeschreibungssprache von HP-Laserjet-Laserdruckern, "PostScript" wird jedoch nicht genannt.

Die Kurzerläuterungen enthalten das wesentliche zu einem Begriff und sollten dem Einsteiger ein gutes Grundwissen bereitstellen. Das Register ist mit acht Seiten umfangreich und brauchbar geraten.



Manchmal fallen im Buchtext Ungenauigkeiten auf: An einer Stelle ist eine Festplatte "um ein Vielfaches empfindlicher als eine Diskette", an einer anderen steht, daß "Festplatten ja zuverlässiger speichern als Disketten". Einige Passagen sind recht inhaltsleer geraten, so die Bitte, Briefe an Software-Häuser nicht zu "vernageln" oder die drei Seiten einer Auswahl von Public Domain-Programmen, deren Beschreibungen z.B. nicht über "diverse Hardcopyroutinen" hinausgehen.

Sprachlich ist das Buch in einem lockeren Umgangston gehalten und versucht sich von einer rein technischen Anleitung abzuheben. Insgesamt ist der Stil gelungen, schwankt allerdings an einigen Stellen doch zu sehr zwischen augenzwinkerndem Witz, flapsigen Einlagen und freundlichem, leichtverständlichem Stil für den Einsteiger, der lernen will. Eine Stilblüte soll an dieser Stelle nicht verschwiegen werden; sie ist aber keineswegs typisch: *"Klicken Sie keinesfalls auf OK, wenn die Meldung nicht lautet: von A: auf B:. Sonst kopieren Sie die leere auf die volle Diskette (das wäre schade)."*

Aber dennoch: "Das Atari-ST 1x1" ist ein Buch, das sich hervorragend für den Anfänger eignet, der ohne größere Computer-Kenntnisse vor seinem neuen ST sitzt. Mit der Einführung in die Desktop-Benutzung werden die Defizite des Original-Handbüchleins ausgeglichen, und mit dem Nachschlageteil sind wichtige Informationen einfach abzurufen. Eigentlich ein typisches Geschenkbuch, dem erfüllten Geburtstagswunsch beizulegen.

RT

GUTSCHEIN Nr. SI 11

Bitte senden Sie mir:

O Kostenlose Info zu PegaFAKT, dem Fakturierungsprogramm mit Lager- und Adressenverwaltung für ATARI ST's, das nur 99,- DM kostet.
(jetzt Version 2.0 mit vielen neuen Funktionen !)

O Die PegaFAKT-DEMO-Diskette für nur 20,- DM (wird beim Programmkauf zurückerstattet)

O Kostenlose Info zu PegaSTIC, dem universellen Etikettendruckprogramm für alle ATARI ST's, das nur 49,- DM kostet.

X Kostenlose Liste mit COMPUTERZUBEHÖR und PROGRAMMEN zu DISCOUNT-PREISEN für:
O C16/64/128 O AMIGA O CPC
O ATARI XL/XE O ATARI ST O PC

Einfach auf Postkarte kleben, den Absender nicht vergessen und ab die Post an:

PegaSoft R. GÄRTIG
SOFTWARE-ENTWICKLUNG & -VERTRIEB
Ringstr.4 7450 Hechingen-Beuren Tel: 07477/8158



MODEME

Kennzeichen:
An alle ATARI Rechner problemlos anschließbar
– Hayes-kompatibel

Anschluß:
Über RS-232 Schnittstelle/serielles Kabel

Lieferumfang:
Einheit, Modembox, modulares US-Telefonkabel, 220V-Netzteil, Handbuch

A. TAILYN 1200, extern

0 – 300/1200 bps für asynchrone Kommunikation (BTX mit 1200/1200 bps) CCITT V.21, V.22, BELL 103/212A – kompatibel Voll- oder Halbduplex; 8 LED Kontrollanzeigen, eingebauter Lautsprecher, Hayes-AT-Befehlssatz, automatisches Wählen und Antworten

DM **199,-**

B. DISCOVERY 1200CK, extern

0 – 300/1200 bps; sonst wie Tailyn 1200

DM **249,-**

C. LIGHTSPEED 2400, extern

1200/2400 bps; asynchron/synchron CCITT V.22, V.22bis, BELL 103/212A Automatische Wahl der Übertragungsgeschwindigkeit, wählt und antwortet automatisch; automat. Anruferkennung, 8 LED-Kontrollanzeigen, eingebauter Lautsprecher, erweiterter AT-Hayes Befehlssatz, Telefonnummern-Speicher, Speicherung des Konfigurationsprofiles im nichtflüchtigen Speicher

DM **389,-**

D. LIGHTSPEED 2400C, extern

300/1200/1200-75/2400 bps; synchr./asynchr., CCITT V.21, V.22, V.22bis, V23
sonst wie LIGHTSPEED 2400

DM **499,-**

E. SMARTLINK 2400S, extern

0-300/1200/2400 bps; asynchron/synchron CCITT V.21, V.22, V.22bis, BELL 103/212A – sonst wie Lightspeed 2400 –

DM **399,-**

F. SMARTLINK 2400M,
MNP Klasse 1 – 5

1200/2400 bps, asynchron/synchron, CCITT V.22, V.22bis, BELL 103/212A, MNP Klasse 1 – 5 Datenkompressionsprotokoll mit Fehlerkorrektur, Voice/Data Umschaltung
– sonst wie Lightspeed 2400 –

DM **749,-**

RS 232 C serielles Kabel für ATARI

DM **14,90**

14 Tage Rückgaberecht
Gewährleistung 1 Jahr
Alle Angebote freibleibend.
Lieferung ab Lager Hamburg.

Hinweis: Der Anschluß der Modeme an das öffentliche Postnetz ist verboten und unter Strafe gestellt.

Gleich bestellen (a. telefonisch) oder Info-Material anfordern!

CompuScience
Computerhandels GmbH
Steinwegpassage 5 · 2000 Hamburg 36
Tel. (040) 35 39 41 · Fax (040) 35 39 45

Wenn es um Beratung, Service und gute Preise für Sie geht :

z.B. Software

PHONE DM 79,-

Die bequeme Adressverwaltung um ständig auf dem laufenden zu sein

- als Accessory jederzeit aufrufbar
- schnell durch ausgereifte Programmiertechnik
- Datenaustausch mit anderen Programmen
- minimaler Speicherplatz

ST-DCL DM 298,-

Kommandozeileninterpreter+ Tools

ST-DCL erlaubt es, auf einem Atari unter einer Kommandoumgebung zu arbeiten, die der VAX /VMS-Digital Command Language (DCL) nachempfunden ist. Auch für MS-DOS lieferbar.

PRIVAT-LIQUIDATION

PRAXIS-BUCHHALTUNG

je DM 375,-

Arzt - Software

Aus der Praxis für die Praxis. Praxisliquidation und Praxisbuchhaltung. Info anfordern. Demo plus Handbuch für DM 40,-. (Wird bei Kauf angerechnet).

Sämtliche PD- und Standard-Software lieferbar: Application Systems, C.A.S.H., CCD, DMC, Drews EDV+BTX, G Data, GFA, Hagera, Hyper-Soft, Kniss-Soft, Kieckbusch, NovoPlan, Technobox u.v.m.

z.B. Hardware

vortex HD plus 20	**DM**	**998,-**
vortex HD plus 30	**DM**	**1198,-**
vortex HD plus 60	**DM**	**1798,-**
star LC 10	**DM**	**598,-**
NEC P6 plus	**DM**	**1698,-**

Wir sind ATARI-Systemhändler.

Alle Drucker mit deutschem Handbuch und Kabel. Sämtliche Hardware für Atari-Computer lieferbar. Alle Angebote freibleibend.

MAUSO ST DM 128,-

Die Mäuse sind los

RS 232 Maus für Atari lieferbar. Die alternative Maus. Inclusive Software.

Hardware
Software
Service

LOGITEAM

Computerhandelsgesellschaft mbH
Kölner Str. 132 - 5210 Troisdorf
Telefon (0 22 41) 7 18 97/98

DER ETWAS ANDERE VERSAND!

24-Stunden-Service!

Wir garantieren, das jede Bestellung spätestens 24 Stunden nach Eingang unser Haus verläßt, sofern verfügbar. Auf alle gekauften Artikel erhalten Sie natürlich volle Garantie. Wir führen jede verfügbare Hard- und Software für den Atari ST, sowie alle Bücher. Hier ein kleiner Auszug aus unserem reichhaltigen Programm:

SPIELESOFTWARE:

Afterburner	45,-
Archipelagos	80,-
Balance of Power 1990	75,-
Barbarian II (Palace)	60,-
Battletech	80,-
Bio Challenge	75,-
Bolo Werkstatt	55,-
Buffalo Bill's Rodeo	80,-
Daley Thompson	45,-
Das Reich Anno 1871	55,-
Dschungelbuch	60,-
Dungeon Master	75,-
Elite	65,-
Espionage	60,-
Eye	20,-
F-16 Falcon	80,-
F-16 Mission Disk 1	65,-
Fish	80,-
Flight Simulator II deutsch	95,-
jede Scenery Disc dazu	45,-
F.O.F.T.	60,-
Fugger	60,-
Gauntlet II	70,-
Grand Monsterslam	65,-
Growth	45,-
Hacker	45,-
Hostages	75,-
Jet	95,-
Joan of Arc	65,-
Kaiser	120,-
Kick off	45,-
Kult	80,-
Leaderboard Birdie	70,-
Legend of Djel	80,-
Leisure Suit Larry	60,-
Leisure Suit Larry II	85,-
Leonardo	60,-
Licence to Kill	60,-
Lombard RAC Rallye	80,-
Mega Pack Compilation	80,-
Millenium	80,-
Minigolf	55,-
Ol Imperium	65,-
Operation Neptun	65,-
Pacmania	60,-
Paper Boy	65,-
Pirates	80,-
Populous	75,-
Populous Scenery Disk I	35,-
Powerdrome	85,-
Psion Chess	65,-
Purple Saturn Day	75,-
Reisende im Wind II	70,-
RVF Honda	80,-
Skrull	60,-
Space Quest III	95,-
Speedball	80,-
Star Trek	65,-
Starglider II	65,-
Star Wars Trilogie	80,-
Superman	80,-
Technocop	60,-
Time Scanner	65,-
Time of Lore	85,-
Thunderblade	45,-
Trivial Pursuit	60,-
Ultima IV	80,-
Vectorball	45,-
Virus	65,-
Volleyball Simulator	60,-
Wallstreet Wizard	65,-
Wallstreet Wizard Editor	45,-
Waterloo	80,-
Zack Mc Kracken	75,-

ANWENDERSOFTWARE:

Adimens 2.3	195,-
Aladin MacIntosh Enhancer	595,-
Anti Virus Kit	85,-
Arabesque	275,-
Beckercad-Studentenversion	195,-
BS-Fibu	590,-
BS-Handel	490,-
BSS-Plus Module	auf Anfrage
BTX-Manager 3.02	385,-
CAD 3D Cyber Studio	175,-
CAD 3D Cyber Control	90,-
Convert	95,-
Copy Star 3.0	160,-
CADja	695,-
Disk Royal	85,-
Epsimenu	85,-
Fibu Man	ab 395,-
GFA-Chemgraf	75,-
GFA-Draft plus	340,-
Systembibliotheken dazu	je 145,-
Headline Signum Utility	95,-
Hotwire	75,-
IPA Degenis III	165,-
Logistix	390,-
LDW-Power Calc	245,-
Monostar/Plus	35,-
Multidesk	75,-
Neo Desk	85,-
Omikron Compiler	175,-
PC-Ditto	150,-
Redakteur	145,-
Retouche	395,-
Revolver	125,-
Spectre	ab 495,-
ST Pascal plus	240,-
Spectrum 512	140,-
Star-Writer Lasertreiber	90,-
Steuer Tax '88	90,-
Superbase Professional	590,-
Tempus 2.0	120,-
Tim II Fibu	590,-
Timeworks Publisher	230,-
Turbo C	ab 225,-
Turbo ST	75,-
Wordstar	190,-
1st Proportional	115,-
1st Adress	95,-

ZUBEHÖR:

Staubschutzhauben Kunstleder für:	
ATARI SM 124	25,-
ATARI 1040 o. Mega Tast.	je 18,-
ATARI 260/520 ST	15,-
Mega ST Set Monit. + Tast.	50,-
andere Monitore + Drucker	a. A.
Mausmatte	18,-
Media Box 3,5" f. 150 Disk's	40,-
Monitorumsch. ohne Reset	ab 50,-
Marconi Trackball	190,-
Handy Scanner inc. Texterk.	450,-
NEC P 6 +	1395,-
PC-Speed	595,-
SPAT Flachbettscanner	98[illegible]
ATARI Megafile 30	10[illegible]
Vortex HD 60	17[illegible]
3,5" NO NAME MF2DD	[illegible]
3,5" BOEDER 2DD farbig	

PUBLIC DOMAIN:

Wir haben über 2.000 Pro[illegible]
auf über 300 Disketten. N[illegible]
rung wie in ST-Computer
Außerdem über 10.000 [illegible]
auf 2.000 Disketten a[illegible]
JEDE DISKETTE
Auch Neuheiten AB

Unseren Gesamtkatalog erhalten Sie kostenlos.
Lieferung per NN zzgl. 7,- DM Versandkosten. Bei V[illegible]
3,- DM Versandkosten, ab 100,- DM Bestellwert versan[illegible]
Auslandsversand grundsätzlich zzgl. 15,- DM Versand

VISA/Eurocard Accepted

Der R[illegible]
Real[illegible]
Der R[illegible]
rasanten[illegible]
beträgt 32[illegible]
x 400) des [illegible]
Graustufe be[illegible]
Automatische[illegible]

COMPUTER-V[illegible]

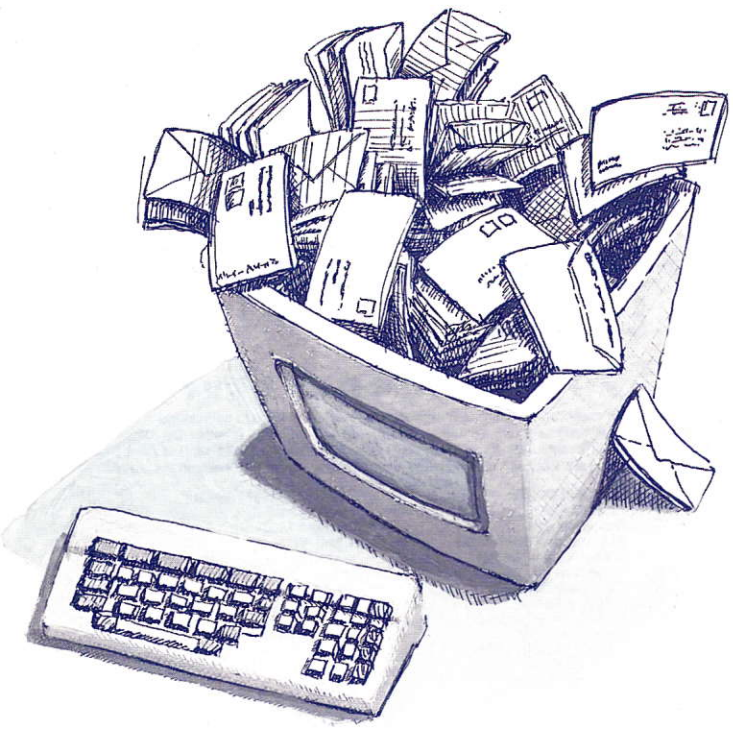
Schlich[illegible]

...der etwas andere Versand

ATARI-Fachmarkt · MS-DOS Fachmarkt · NEC-Fachhandel

Rund um die Uhr: ☎ 030 / 7 86 10 96

Postanschrift: Katzbachstraße 8 · D-1000 Berlin 61
Ladengeschäft: Katzbachstraße 6+8 · D-1000 Berlin 61
Fax: 030 / 786 19 04 · Händleranfragen erwünscht



Ein Wort in eigener Sache

In den Jahren, die unsere Zeitschrift existiert, haben wir immer wieder versucht, durch die Beantwortung der bei uns eingehenden Briefe ein wenig Licht in das Dunkel zu bringen, das bei der Arbeit mit dem ATARI ST schon so manch einen aus der Fassung bringen konnte - eine Tatsache, die nicht nur Ihnen, verehrter Leser, sondern auch uns oft genug zu schaffen machte. Nichtsdestotrotz haben wir uns bemüht, die Probleme zu lösen und diverse Leserbriefe zu veröffentlichen, da wir der Meinung waren, daß die jeweilige Thematik auch einen größeren Leserkreis interessieren könnte. Trotzdem gibt es immer wieder Briefe, die wir nicht beantworten können oder dürfen. Damit Sie nicht allzusehr enttäuscht zu sein brauchen oder keine Antwort erhalten, möchten wir Sie bitten, sich an folgende Spielregeln zu halten, die sich aus unserer Erfahrung ergeben haben. Fällt Ihr Brief nicht unter die folgenden Kriterien, hat er gute Chancen, positiv beantwortet oder wenigstens als Hilferuf an unsere Leserschaft gedruckt zu werden.

1. Leider gehen immer wieder Briefe mit dem Wunsch ein, ein Produkt für diesen oder jenen Anwendungsfall vorzuschlagen, verschiedene Produkte bezüglich der Vor- und Nachteile gegeneinander abzuwägen und zu bewerten. Es ist uns aus Wettbewerbsgründen nicht erlaubt, ein bestimmtes Produkt zu favorisieren, selbst wenn wir das eine oder andere in der Redaktion überzeugt einsetzen. Wir können Sie in diesem Fall ausschließlich auf die von uns möglichst objektiven Tests und eventuell anstehende Fachmessen hinweisen. Bedenken Sie bitte, daß auch wir nicht jede Textverarbeitung, jedes Malprogramm und so weiter kennen und bestimmte Produkte dadurch in das Abseits drängen würden.

2. Oft erreichen uns Briefe, die sich positiv oder auch negativ über bestimmte Händler, Softwarehäuser oder deren Produkte auslassen. Sicherlich interessieren uns solche Bemerkungen. Bitte haben Sie aber Verständnis, daß wir weder Lob noch Tadel abdrucken dürfen, da diese Aussagen meist subjektiv sind. Anders sieht die Sache beispielsweise bei Gerichtsurteilen aus, die Sie, verehrte(r) Leser(in), erfochten haben.

3. Aufgrund der Vielzahl an Briefen, die uns täglich erreichen, sind wir leider nicht in der Lage, Programmfehler anhand von Listings oder ähnlichem zu korrigieren. Dennoch sollte ein Problem möglichst detailliert beschrieben sein, denn Ferndiagnosen sind prinzipiell sehr schwer, jedoch mit genauerer Angabe der Symptome eventuell durchführbar.

4. Von Zeit zu Zeit erreichen uns Briefe mit der Bitte, die Adresse des Lesers zwecks allgemeiner Kontaktaufnahme zu veröffentlichen. Würden wir dies in die Tat umsetzen, würde sich der Umfang des anderen redaktionellen Teils beträchtlich verkleinern. Ausnahmen stellen Leser in fernen Ländern dar, für die eine Kontaktaufnahme im eigenen Land recht schierig ist.

Zum Schluß sollen ein paar Tips eventuell voreilig geschriebene Briefe verhindern.

1. Wenn Sie ein Problem bezüglich einer bestimmten Problematik haben oder an einem bestimmten Produkt interessiert sind, finden Sie interessante Artikel darüber eventuell in vorhergehenden Ausgaben userer Zeitschrift. Zur Auswahl eignet sich das Jahresinhaltsverzeichnis besonders gut, das immer am Jahresende in der ST Computer abgedruckt wird.

2. Sollten die Probleme mit der Handhabung eines Produktes zu tun haben, wenden Sie sich zunächst an Ihren Händler und über diesen an den Distributor beziehungsweise an das Software-Haus. Die Wahrscheinlichkeit, daß Ihnen das Software-Haus weiterhelfen kann, ist um ein Vielfaches höher als die, daß wir Ihnen helfen können.

3. Lesen Sie aufmerksam die Leserbrief-Seite. Viele Fragen wiederholen sich immer wieder, obwohl wir bestimmte Probleme schon mehrfach angesprochen haben.

Resource-Laden bei LovelyHelper

Als stolzer Besitzer Ihres "Lovely Helper"s stehe ich vor folgendem Problem: Wie kann ich das ACC von meiner Harddisk (Partition C) installieren, wenn ich weder ein PASCAL-Programmsystem zur Änderungen des mitgelieferten Quellcodes mein Eigen nennen kann, noch bei jedem Systemstart die Disk mit dem RCS-File im Laufwerk haben möchte? Vielleicht ist es möglich, den Programmcode dahingehend zu patchen, daß das RCS-File im gleichen Verzeichnis, in dem auch das ACC-File steht, gesucht wird und daher auch das ACC von beliebigen Partitionen aus installiert werden kann? Für eine Lösung meines kleinen Problems wäre ich Ihnen sehr dankbar.

C. Nachtwey, 2000 Hamburg

Red.: Die Lösung ist relativ einfach, vorausgesetzt, Sie haben zumindest einen Diskmonitor zur Hand. Am einfachsten ist es, wenn dieser in der Lage ist, innerhalb einer vorgegebenen Datei zu arbeiten. Ihre Aufgabe ist es nun, innerhalb der Accessory-Datei nach dem String "A:\HELPER.RSC" zu suchen, was sehr einfach ist, sollten Sie einen Diskmonitor besitzen, der innerhalb einer Datei suchen kann. Ansonsten würde ich Ihnen empfehlen, eine Diskette neu zu formatieren, darauf diese Datei zu kopieren und dann diese Diskette zu durchsuchen. Dadurch verhindern Sie, daß Sie diesen String in einer von Ihnen nicht gewünschten Datei finden. Danach überschreiben Sie den String mit "\HELPER.RSC" oder "HELPER.RSC", wobei die erste Version dafür sorgt, daß tatsächlich im Hauptverzeichnis des Boot-Laufwerks gesucht wird. Bitte fügen Sie aber unbedingt am Ende des Strings eine 0 an, was dafür sorgt, daß der String abgeschlossen wird. Sollte der obengenannte String mehrfach vorkommen, wird der Vorgang entsprechend wiederholt. Da Lovely-Helper auch auf eine Parameterdatei zugreift, sollte "A:\HELPER.PAR"in "\HELPER.PAR" geändert werden. Eine Bitte noch: Führen Sie den beschriebenen Vorgang auf jeden Fall nicht auf der Originaldiskette durch, sondern nur auf einer Sicherheitskopie.

*

QL-Emulator

Ich habe gerade in der Juli/August-Ausgabe Ihrer ST-Zeitschrift gelesen, daß Sie eine extra Rubrik für Emulatoren eingerichtet haben. Aus diesem Grunde möchte ich Sie bitten, über einen leider sehr unbekannten Emulator zu berichten: den QL-Emulator. Er emuliert den Sinclair QL (CM MC68008) perfekt und erreicht sogar eine wesentlich höhere Geschwindigkeit als das Original. Seit ich diesen Emulator habe, nutze ich meinen ST zu 99% als QL, da sein Betriebssystem QDOS wesentlich besser ist als das des ST und perfektes Multitasking bietet. Ich habe von anderen ST-Anwendern, die den Emulator auch gekauft haben, gehört, daß sie die gleichen guten Erfahrungen gemacht haben wie ich. Ich denke, wenn ein "Schrottbetriebssystem" wie MS-DOS soviel Beachtung findet, dann sollte dieser wenn auch exotische Emulator wenigstens einmal in Ihrer Zeitschrift erwähnt werden!

Tilman Kurz, 3500 Kassel

Red.: Leider tun Sie uns Unrecht mit ihrer kleinen Anklage, da wir diesem Emulator einen mehrseitigen Bericht gewidmet haben, der von einem eingefleischten QL-Anwender geschrieben wurde. Der Test ist in der ST 3/89 unter dem Namen 'Sinclair QL-Emulator - Ein neuer Job für den ST' erschienen. Diese Zeitschrift ist selbstverständlich noch erhältlich. Übrigens: Ob ein Betriebssystem besser

als das andere ist, hängt sicherlich von vielen Faktoren ab, so daß eine Abwertung des ST-Betriebssystems in dieser Weise nicht gerechtfertigt ist. Auch wenn es nicht multitaskingfähig ist, hat es andere Stärken, die QDOS nicht besitzt.

*

Dokumentationen zum ATARI ST

In vielen Artikeln über den ATARI ST liest man von 'offiziellen Dokumentationen' oder ähnlichem. Dabei meine ich Dokumentationen wie zum Beispiel "The Hitchhikers Guide to the BIOS" oder die ATARI-Infos von der ATARI-Corp. und Digital Research Inc. Mich interessiert brennend, wie auch ich davon profitieren kann, das heißt, wie ich an diese Schriften gelangen kann. Dazu muß ich sagen, daß ich nicht über ein Modem oder einen Akustikkoppler verfüge, mir also nicht zum Beispiel aus der ATARI-Mailbox downloaden kann.

Weiterhin interessieren mich das ATARI-Entwicklungspaket und GDOS. Auch hier die Frage, wie ich ohne DFÜ an die Programme kommen kann. Da dreimal "Bremer Recht" ist, nun zum dritten Punkt: Vor einigen Tagen habe ich mal wieder ein wenig mit meinem GFA-Assembler "im GEM rumprogrammiert". Als Hilfe benutzte ich dazu das sehr gute ST-Profibuch. Als ich die SHEL-Funktionen des AES ausprobieren wollte, stieß ich auf ein kleines Problem. Im PROFIBUCH ist zum Beispiel die Funktion SHEL_WRITE folgendermaßen deklariert:

```
sh_return = SHEL_WRITE
            (sh_wdoex, sh_wisgr,
             sh_ wiscr, sh_wpcmd,
             sh_wptail)
```

Beim Aufruf dieser Funktion fand ich jedoch heraus, daß sh_wpcmd und sh_wptail vertauscht werden müssen! Auch bei SHEL_READ müssen die beiden Parameter vertauscht werden. Fehler des Profibuchs oder hat Digital Research diese Funktion von Anfang an falsch dokumentiert?

Stefan Näwe, 2800 Bremen

Red.: Zunächst einmal sollte erwähnt werden, daß das sogenannte Entwicklungspaket aus einer Dokumentation und einer Software (C-Compiler, Assembler, RCS etc.) besteht. Daher kostet es relativ viel Geld, zumal die Dokumentation zum Großteil aus drei telefonbuchgroßen Büchern besteht. Des weiteren findet man auch die inzwischen veröffentlichten Informationen über das BIOS (Hitchhikers..), LINE-A (the long awaited...) und den Blitter. Ein richtiges GDOS-Dokument (abgesehen von dem, was sich auf den GDOS-Disks befindet) gibt es meines Wissens nicht, ich lasse mich aber gerne belehren. All diese Dokumente sind nach meiner Meinung aber nicht besonders vollständig und alles andere als didaktisch gut aufgebaut. Abgesehen davon sind 'Hitchhikers Guide to the BIOS' sowie 'The long awaited Line-A-Dokument' alles andere als neu, sprich, es gibt sie schon mehrere Jahre, und sie sind durch neuere Betriebssysteme nicht mehr aktuell. Stattdessen ist es sicherlich sinnvoll, mit Büchern wie dem von Ihnen erwähnten zu arbeiten, da man sich hier die Arbeit gemacht hat, alles Wichtige geordnet(!) zusammenzutragen. Natürlich kann man auch hier Fehler nicht ausschließen - es sind aber sicher weniger als in der ATARI-Original-Dokumentation - so schlimm es auch klingt. Ich denke, daß ATARI diesen Fehler nicht noch einmal bei dem neuen ATARI TT begehen sollte. Sollten Sie dennoch diese Informationen haben wollen, wenden Sie sich am besten an ATARI direkt, oder schauen Sie doch einmal, ob sich ein ATARI-User-Club bei Ihnen in der Nähe befindet, der diese Informationen besitzt.

Zu Ihrer Frage bezüglich der Shel-Funktionen kann ich sagen, daß Sie recht haben und die Funktionen falsch deklariert sind. Allerdings funktioniert es tatsächlich, wenn man die Variablen so füllt, wie sie im Profibuch beschrieben sind. Dort werden bei shel_write die Variable sh_wpcmd als Programmname und sh_wptail als Kommandozeile beschrieben. Füllt man diese Variablen so, funktioniert es auch, da das Binding entsprechend abgedruckt ist. Allerdings sind die Namen in dieser Form nicht gerade sinnig, da man in einer Variablen sh_wpcmd eine Kommandozeile und in einer Variablen sh_wptail einen Namen übergeben sollte. Deshalb sollte die Routine wie folgt umdefiniert werden, wobei wichtig ist, daß in der GEM-Variablen addr_in[0] der Programmname und in addr_in[1] die Kommandozeile zu finden ist.

```
int shel_write (sh_wdoex,
                sh_ wisgr,
                sh_wisover,
                progname, cmdline)
int sh_wdoex,   sh_wisgr,
                sh_wisover;
char *progname, *cmdline;
{
 int_in[0]    = sh_wdoex;
 int_in[1]    = sh_wisgr;
 int_in[2]    = sh_wisover;
 addr_in[0]   = (long)progname;
 addr_in[1]   = (long)cmdline;

  return(crys_if(121));
}
```

Bei shel_read müssen die Parameter in der Kommandozeile sowie in der Zuweisung innerhalb des Bindings vertauscht werden. Bei den Bindings der diversen Compilerssprachen ist dieser Fehler allerdings nicht gemacht worden, so daß das Aufrufen der Routinen ohne Probleme funktionieren müßte.

*

Anleitung für RCS 2.1 gefragt

Seit einiger Zeit benutze ich GFA-BASIC 3.0. Jetzt möchte ich mich zum ersten Mal an die Verwendung von Resource-Dateien heranwagen. Nun hat GFA dem Interpreter das RCS von Digital Research Version 2.1 beigegeben, jedoch mangelt es an einer Gebrauchsanweisung. Eine telefonische Anfrage bei GfA ergab, daß man von Digital Research zwar die Erlaubnis habe, das RCS weiterzugeben, jedoch keine Anleitung hierzu. Danach versuchte ich es bei mehreren Händlern, und ich stöberte in verschiedenen Büchern. Der Erfolg war leider gleich Null. Meine Frage lautet: Woher bekomme ich eine möglichst vollständige Anleitung zum RCS Version 2.1 von Digital Research? Es gibt doch mit Sicherheit noch mehr Programmierer, die noch nicht den vollen Durchblick haben und noch etwas lernen müßten oder wollen.

Kersten Ohland, Raunheim

Red.: Auch wenn Sie praktisch eine Handbreit entfernt von ATARI wohnen, so hätte es sicherlich keinen Zweck, zu ATARI zu gehen, da es es meines Wissens keine offizielle Anleitung für dieses Programm gibt. Allerdings gibt es einen kleinen Lichtblick: Die Version 1.4 des RCS darf (freundlicherweise von ATARI genehmigt) von MAXON als Sonderdisk vertrieben werden. Auch zu diesem RCS gibt es keine offizielle Anleitung, allerdings haben wir uns in unserem 2. Sonderheft die Mühe gemacht, eine zu schreiben: Dort finden Sie eine relativ gute Ein- sowie Anleitung zum RCS 1.4. Zusätzlich gibt es auch Beispiele, wie die Dialogboxen und Menüleisten des STs aufgebaut und angewendet werden. Anwendungen von Dialogboxen und Menüleisten finden Sie immer wieder in unserer Zeitschrift. Sollten Sie diese Artikel verpaßt haben, können Sie beispielsweise die Jahresinhaltsverzeichnisse 'durchforsten', die Sie in den Januar-Heften finden.

Immer up to date

Mit dieser Sparte wollen wir allen unseren Lesern die Möglichkeit geben, sich über die neuesten Programm-Versionen zu informieren. Angegeben werden die aktuelle Versionsnummer, ein eventueller Kopierschutz, die Bildschirmauflösungen und der Speicherbedarf. Softwarefirmen ist es somit möglich, die ST-Computer-Leser über ihre Updates zu informieren.

Programmname	Version	Daten
Adimens ST	3.0	N HM
Aditalk ST	2.3	N HM
Adress ST / Check ST	1.0	N H
Afusoft Morse-Tutor	2.0	N HML
Afusoft Radio-Writer	1.0	N HML
Afusoft Radiofax plus	1.0	N HML 1M
Aladin	3.0	J H
AnsiTerm	1.4	N
Assembler Tutorials	1.05	N
Banktransfer	1.0	N H
1st BASIC Tool	1.1	N HML
BTX/VTX-Manager	3.0	N H 1M
Calamus	1.09.2	N H 1M
Cashflow	1.0	N H 1M
CIS-L&G	1.01	
Creator	1.0	N H
dBMAN	5.10	N HML
fibuMAN	3.0	N H
fibuSTAT	2.3	N H
Flash-Cache/Flash-Bak	1.0	N HM
Flexdisk	1.3	N HML
FM-Meßtechnik	1.0.b	N HM
Gadget	1.2.5b	N H
GEMinterface ST	1.1.	N HML
GFA-Artist	1.0	N L
GFA-Assembler	1.2	N HML
GFA-BASIC 68881	1.3	N HML
GFA-BASIC-Compiler	3.02	N HML
GFA-BASIC-Interpreter	3.07D	N HML
GFA-Draft	2.1	N
GFA-Draft plus	3.0	N
GFA-Farb-Konverter	1.2	N H
GFA-Monochrom-Konverter	1.2	N ML
GFA-Objekt	1.2	N HM
GFA-Starter	1.1	N HML
GFA-Vektor	1.0	N
G+Plus	1.4	N HML
GrafStar	1.0	N H
Hänisch Modula-2	N.1	N HML
Hard Disk Accelerator	1.0	N HML
Hard Disk Sentry	1.10	
Hard Disk Toolkit	2.0	N HM
Harddisk Utility	2.2	N HM
Imagic	1.1	N HML
Intelligent Spooler	1.10	N HML
Interlink ST	1.89	N HM
K-Resource	2.0	N HM
Kleisterscheibe	2.2	N HM
Label ST	1.0	N HML
Laser C (Megamax)	2.1	N HML
1st_Lektor	1.2	HM
Lern ST	1.22	N HML
Link_it GFA	1.1	N HML
Link_it Omikron	2.0	N HML
MagicBOX ST	7.58	N H 1M
Mega Paint	2.20	N H 1M
Megamax Modula 2	3.5	N HM
Micro C-Shell	2.70	N HM
MT C-Shell	1.2	N HM 1M
Multidesk	1.82	N HML
Musix32	1.01	J H
NeoDesk	2.05	N HML
Omikron Assembler	1.86	N HML
Omikron BASIC Compiler	3.06	N HML
Omikron BASIC 68881-Compiler	3.06	N HML
Omikron BASIC Interpreter	3.03	N HML
Omikron DRAW! 3.0	3.01	N HML
Omikron EasyGEM-Lib	1.0	N HML
Omikron Maskeneditor	1.0	N HML
Omikron Midi-Lib	2.1	N HML
Omikron Numerik-Lib	1.2	N HML
Omikron Statistik-Lib	1.5	N HML
PAM's TERM/4014	3.012	N H
PAM's TurboDisk	1.7	N HML
PAM's NET	1.0	N HML
PCB-layout	1.19	N H
PC ditto Euroversion	3.96	N HML
PegaFakt	1.3	N H
phs-BTX-Box	6.0	N HML 1M
phs-ST-Box	1.2	N HM
phs-Boxtalk	1.0	N HM 1M
phs-Boxedi	1.0	N HML 1M
phs-Cheapnet	1.2	N HM
Pro Sound Designer	1.2	L
Pro Sprite Designer	1.0	L
Protos	1.1	N H 1M
Revolver	1.1	N HML 1M
Search!	2.0	N HM
Signum! zwei	1.0	N H
Soundmachine ST	1.0	N HM
SoundMerlin	1.0	N HM
SPC-Modula-2	1.42	N HML
Spectre 128	1.9	J HM
1st_Speeder 2	1.0	N HML 1M
STAD	1.3+	N H
Steuer-Tax 2.9	1.10	N HM
Steuer-Tax 3.9	1.10	N HM
STop	1.1	N HM
ST Pascal plus	2.06	N HM
SuperScore	1.4	J H 1M
Tempus	2.05	N HM
TIM	1.2	N H
TIM II	1.0	N H 1M
Transfile ST 1600	1.1	N HM
Transfile ST 850	1.1	N HM
Transfile ST plus	3.0	N HM
Turbo C	1.1	N HM
Turbo ST	1.6	N HML
UIS II + Hermes	2.5	
VSH Manager	1.0	N HML 1M

Irrtum vorbehalten!
Daten-Legende : N = kein Kopierschutz, J = Kopierschutz, H = hohe Auflösung, M = mittlere Auflösung, L = niedrige Auflösung, 1M = mindestens 1 Megabyte

INSERENTENVERZEICHNIS

ATARI	63	GENG TEC	182	LESCHNER	24,181	SCHNEIDER	170
AFM COMPUTER	185	GFA	196	LIGHTHOUSE	15	ST-PROFI-PARTNER	166,167,170
ADI SOFTWARE	9	GDAT	117	LACOM	98	SCHLICHTING	183
ABAC	109	GTI	31	MAYER-GÜRR	121	SCHLEICH	15
APPLICATION SYSTEMS	2	GAUGER	166,167	MAXON	13,23,48,54,103,134	SCILAB	182,185
ABAKUS	121	GRAF + SCHICK	166,167	MARKERT	166,167	SCHUSTER	66
AS DATENTECHNIK	109	HEBER-KNOBLOCH	121	MICHIELS	166,167	SCHEIDT	31
AB-COMPUTER	181	H + T	127	NEUBAUER	121	SCHÖN	24
BUSCH UND REMPE	31	HÜTHIG VERLAG	69	NEC	51	STIEHL	181
COMPUTER TREFF	166,167	HORN	24	NEERVOORT	14	SEH	172
COMPEDO	171	HOFMANN	171	NOVOPLAN	185	SCHREIBER	166,167
COMPUTEC VESL.	121	HAASE	170	OLUFS	24	TREND-DATA	166,167
CWTG	14	HOLM	170	OHST	70,166,167	TOMMY SOFTWARE	11,19
COMPU SCIENCE	183	HERBERG	158,159	PD-EXPRESS	166,167,185	TK-COMPUTER	139
COPYDATA	170	HEIM	143,106,99	PRINT TECHNIK	179	TRUMPP	136
DITTRICH	139	HD-COMPUTERTECHNIK	166,167	PROTAR	59	T.U.M.-ST-SOFT	166,167
DATA-BECKER	36,37	INTERSOFT	166,167,121	PETILLON	181	UECKER	166,167
DIGITAL DATA DEICKE	123	IKS	166,167,113	PORADA	171	VORTEX	155
DREWS	70	IDEE SOFT	109	PITZ	171	WITTICH	24
DM-COMPUTER	117	IDL	66,166,167	QUAZAR	90	WITASEK	121
DUFFNER	166,167	KLEEFELD + PARTNER	181	RHOTHRON	85	WEIDE	35,181
ESCH	136	KIECKBUSCH	163	RÜCKEMANN	192	WELLER-TOOLS	171
EICHHORN	24	KUSCHEK	109	ROSIN	109	WALLER	66
EURASYSTEMS	27	KNISS SOFT	113	RINGHUT	171	WEESKE	166,167,193
EICKMANN	15	KAROSOFT	171	RTS	162	WOHLFAHRTSTÄTTER	70,166,167
FSE	113	KURWIG	24	RÄTZ	98	YELLOW	70
FSKS	177	KRÜGER	181	SOFT 2000	181	ZELLER	193
FISCHER	53	LAUTERBACH	166,167	SHIFT	145		
GÄRTIG	180	LOGITEAM	183	SOMMERFELD	109		
G-DATA	45,195	LUDWIG	181	SENDER	111		
GMA-SOFT	171	LOGO VERLAG	131,139	SLAVNIC	181		

SUPER - STARDRIVER
SUPER NEC-DRIVER

NEUE DRUCKERTREIBER FÜR 1ST WORD PLUS

Nutzen Sie alle Möglichkeiten Ihres Druckers voll aus. Super-Driver erweitern die Textverarbeitung: 1st Word Plus wird zum DTP-System.

- Zeilenabstände beliebig verändern (z.B. 1 1/2-zeilig)
- Texte in 2-6 Spalten drucken (echter Mehrspaltendruck)
- alle Druckerschriftarten in einem Dokument einsetzen
- neue Zeichensätze laden und benutzen (z. Zt. 40 Fonts)
- Titel in doppelt und vierfach hohen Zeichen drucken
- gesperrt drucken, Viertelschritt- und Halbschrittaste
- bedienungsfreundliches Handbuch und Referenzkarte

Haben Sie das von 1st Word Plus erwartet?
Warum weniger Leistung, wenn es Superdriver gibt!
Erhältlich für STAR NL10/ LC10/LC24-10 und NEC P6/P7/P2200
Neu: Jetzt auch für Seikosha SL-80IP !

RR - Soft
Reinhard Rückemann
Grundstrasse 63
5600 Wuppertal 22
T.: 02 02 / 64 03 89

35.- DM
+ 5.- DM Versandkosten
Scheine, Scheck oder Nachnahme (+ 2.- NN-Geb.)

RSOFT

3rd Word
Synonymenlexikon
Formulierungshilfe
Rechtschreibhilfe

... denn oft trifft erst das dritte Wort den Sinn

Das Synonymenlexikon für den ATARI ST

Wörterbuch mit sinnverwandten Begriffen
Formulierungshilfe für alle Textarten

Grundversion enthält ca. 50 000 Begriffe
erweiterte Version mit ca. 65 000 Begriffen

sinnvolle Ergänzung zu Textverarbeitungen arbeitet mit 1st Word Plus, Calamus, Tempus etc. 3rd Word ist als Accessory jederzeit aufrufbereit keine Unterbrechung der Arbeit am Text nötig

RR - Soft
Reinhard Rückemann
Grundstrasse 63
5600 Wuppertal 22
T.: 02 02 / 64 03 89

RSOFT

Grundversion 69.- DM
Erweiterte Version 98.- DM
+ 5.- DM Versandkosten
Händleranfragen erwünscht.

Autoren-Collection

Johannes Uhrlau: ComponiST DM 49.-
Ein professionelles Programm zum Eingeben, Vorspielen und Drucken von Musikstücken. Dreistimmig komponieren und spielen. Abspeicherung des Soundstrings und Einbindung in andere Programme möglich.

M. Kemmer, F. Walter: ArCADe DM 79.-
Ein universelles CAD-Programm zum Entwerfen und Drucken von technischen Zeichnungen und anderen exakten Darstellungen. Ein optimales Preis - Leistungs - Verhältnis!

Franz J. Pentenrieder: MiniFAKT DM 95.-
Ein GEM-Programm zum Erstellen von Rechnungen. Viele Formate vorgegeben, individuelle Anpassung möglich. Schnittstellen zu 1st WordPlus und Adimens. 50-seitiges Handbuch. Demoprogramm mit Anleitungskopie 15.- DM.

Egbert Güssgen: Grafic-Collection DM 139.-
4000 Grafiken (IMG-Format, 1st Word Plus!) auf 8 Disketten mit ausführlichem Katalog. (Jedes Bild in Originalgröße) Programm zum Anzeigen und Formatändern mitgeliefert.

R. Brück, J.J. Pypers: Boot It DM 69.-
Die Festplattenutility zum automatischen Start Ihrer Programme. Freie Auswahl der Accessories und Autoordner - Programme.

RR - Soft
Reinhard Rückemann
Grundstrasse 63
5600 Wuppertal 22
T.: 02 02 / 64 03 89

RSOFT

Bitte fordern Sie unseren Softwarekatalog mit weiteren Programmen an!

In der nächsten ST-Computer lesen Sie unter anderem

Großer Weihnachtsbasar

Es ist mal wieder soweit: Die Weihnachtszeit naht, und viele überlegen, was man sich aus dem reichhaltigen Angebot von Computern und Zubehör alles wünschen und schenken lassen kann. Damit Sie nicht voller Streß durch verschiedene Geschäfte hetzen müssen, um die richtigen Geschenke zu finden, haben wir eine Auswahl der interessantesten Accessoires für Sie zusammengestellt.

Festplatten-Übersicht

Manch einer möchte sich schon lange eine Festplatte anschaffen, weil ihm die Arbeit mit Diskettenlaufwerk(en) zu langsam und unkomfortabel wird. Welche Festplatten gibt es? Wie schnell sind sie? Wie gut ist der Treiber? Sind sie ATARI-kompatibel? Was kosten sie? Diese und viele Fragen mehr werden wir in der nächsten Ausgabe beantworten.

Wir bauen uns einen Super-ST

Sind Sie unzufrieden mit Ihrem ST? Ist er Ihnen vielleicht zu langsam, zu unkomfortabel, stört Sie die Tastatur, möchten Sie gerne IBM-Programme benutzen, oder ist Ihnen vielleicht der Monitor zu klein? In der Originalverpackung von ATARI befindet sich ein eher kärglich ausgestatteter ST. Mit Hilfe von verschiedenen Erweiterungen, Accessoires und anderen nützlichen Dingen werden wir versuchen, uns einen Super-ST zu bauen, vor dem selbst eingefleischte IBM-Anwender Respekt zeigen dürften. Sie werden staunen, was mit dem ST alles zu realisieren ist!

Die nächste ST Computer erscheint am Fr., den 24.11.89

Fragen an die Redaktion

Ein Magazin wie die ST-Computer zu erstellen, kostet sehr viel Zeit und Mühe. Da wir ja weiterhin vorhaben, die Qualität zu steigern (ja, auch wenn das manchmal danebengeht), haben wir Redakteure ein großes Anliegen an Sie, liebe Leserinnen und Leser:

Bitte haben Sie Verständnis dafür, daß Fragen an die Redaktion nur Donnerstags von 14^{00}-17^{00} Uhr telefonisch beantwortet werden können.

Vielen Dank für Ihr Verständnis

Impressum ST Computer

Chefredakteur: Uwe Bärtels (UB)

Redaktion:
Uwe Bärtels (UB)
Harald Egel (HE)
Harald Schneider (HS)
Martin Pittelkow (MP)

Redaktionelle Mitarbeiter:
C.Borgmeier (CBO)
Fernando Brand (FB)
Claus Brod (CB)
Ingo Brümmer (IB)
Derek dela Fuente (ddF)
Stefan Höhn (SH)
Raymund Hofmann (RH)
Jürgen Leonhard (JL)
Claus P. Lippert (CPL)
Markus Nerding (MN)
Chr. Schormann (CS)
R.Tolksdorf (RT)

Autoren dieser Ausgabe:
M.Andrä
P.S.Bauer
V.Brixius
D.Brockhaus
A.Esser
D.Kühner
Dr.V.Kurz
U.Müller
T.Nitz
M.Pfahl
W.Püchert
D.Rabich
U.Thürmann
T.Werner
C.D.Ziegler

Auslandskorrespondenz:
C.P.Lippert (Leitung), D.Dela Fuente (UK)
L.Hennelly (Nordamerika)

Redaktion: MAXON Computer GmbH
Postfach 59 69
Industriestr. 26
6236 Eschborn
Tel.: 0 61 96/48 18 14, FAX : 0 61 96/4 11 37

Verlag: Heim Fachverlag
Heidelberger Landstr. 194
6100 Darmstadt 13
Tel.: 0 61 51/5 60 57, FAX : 0 61 51/5 56 89 + 5 60 59

Verlagsleitung:
H.J.Heim

Anzeigenverkaufsleitung:
U.Heim

Anzeigenverkauf:
K.Margaritis

Anzeigenpreise:
nach Preisliste Nr.3, gültig ab 1.1.88
ISSN 0932-0385

Grafische Gestaltung:
Gabriele Imhof

Layout:
Kerstin Feist, Gülsu Okay, Manfred Zimmermann

Titelgestaltung:
Gunter Wenzel (Tel.: 06172/37193)

Fotografie:
Gabriele Imhof, Archiv, dpa

Produktion:
K.H.Hoffmann, B.Failer, S.Failer

Druck:
Ferling Druck W.Frotscher GmbH

Lektorat:
V.Pfeiffer

Bezugsmöglichkeiten:
ATARI-Fachhandel, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag

ST Computer erscheint 11 x im Jahr
Einzelpreis: DM 7,-, ÖS 56,-, SFr 7,-
Jahresabonnement: DM 70,-
Europ. Ausland : DM 90,- Luftpost : DM 120,-
In den Preisen sind die gesetzliche MWSt. und die Zustellgebühren enthalten.

Manuskripteinsendungen:
Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit seiner Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern der MAXON Computer GmbH. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

Urheberrecht:
Alle in der ST-Computer erschienenen Beiträge sind urheberrechtlich geschützt. Reproduktionen gleich welcher Art, ob Übersetzung, Nachdruck, Vervielfältigung oder Erfassung in Datenverarbeitungsanlagen sind nur mit schriftlicher Genehmigung der MAXON Computer GmbH oder des Heim Verlags erlaubt.

Veröffentlichungen:
Sämtliche Veröffentlichungen in der ST-Computer erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes, auch werden Warennamen ohne Gewährleistung einer freien Verwendung benutzt.

Haftungsausschluß:
Für Fehler in Text, in Schaltbildern, Aufbauskizzen, Stücklisten usw., die zum Nichtfunktionieren oder evtl. zum Schadhaftwerden von Bauelementen führen, wird keine Haftung übernommen.

(c) Copyright 1989 by Heim Verlag

GFA für ATARI

GFA-BASIC weltweit über 100.000 mal im Einsatz!

GFA-BASIC 3.0 EWS ST

Hochgeschwindigkeitsinterpreter und integrativer Compiler als komplettes Entwicklungssystem. Einbindung von Assembler und C-Source-Codes in GFA-BASIC-Programme.

DM 198,–

GFA-BASIC 2.0 EWS ST

Das GFA-BASIC 2.0 Entwicklungssystem ST – Interpreter + Compiler – für Einsteiger.
(Upgrade-Möglichkeit zum GFA-BASIC 3.0 Entwicklungssystem ST DM 160,–)

DM 49,90

GFA-ASSEMBLER ST

Professioneller Makro-Assembler für 68000-Programmierer: Leistungsfähiger Editor mit integriertem Assembler und Linker. Nachladbarer Debugger.

DM 149,–

GFA-BASIC 3.0 ST Training

Der ideale Einstieg in die Version 3.0.
14 Themenschwerpunkte aus allen wichtigen Bereichen.
272 Seiten, Hardcover, ISBN 3-89317-005-7

DM 29,–

GFA-BASIC ST: Version 3.0

Das Umsteigerbuch für alle diejenigen, die bisher mit 2.0 gearbeitet haben und jetzt die phantastischen Möglichkeiten der „großen" Version 3.0 optimal nutzen wollen. 394 Seiten, Hardcover, inkl. Diskette, ISBN 3-89317-004-9

DM 59,–

neu

Das GFA-Anwenderbuch

Wann GFA-BASIC? Wann GFA-ASSEMBLER? – Die Antwort finden Sie in dem neuen GFA-Anwenderbuch (inkl. Programmiertricks und Anwendungsbeispielen zum neuen GFA-BASIC 3.0 Compiler).
Ca. 450 Seiten, Hardcover, inkl. Diskette, ISBN 3-89317-011-1

DM 59,–

GFA-DRAFT-plus ST

Leistungsfähiges, zweidimensionales CAD-Programm, seit Jahren bewährt, tausendfach im Einsatz.
(Symbolbibliotheken zu GFA-DRAFT-plus auf Anfrage)

DM 349,–

GFA-CASTELL – Architektensystem

Das Architektensystem für den ATARI-Mega ST4.
Mächtige Funktionen mit anwendergerechter Benutzerführung.
Fordern Sie unseren Sonderprospekt „GFA-CASTELL" an. **Preis auf Anfrage**

GFA-STATISTIK

Das professionelle Statistikpaket.
Über 70 Verfahren der beschreibenden und schließenden Statistik.
Umfangreiches Handbuch, Beschreibung jedes Verfahrens sowohl von der rein formalen als auch der Anwendungsseite. **DM 998,–**
Sonderprospekt GFA-STATISTIK anfordern.
Campus- und Studentenversion: **Preis auf Anfrage**

Anruf genügt
0211/5504-0

GFA Systemtechnik GmbH
Heerdter Sandberg 30-32
D-4000 Düsseldorf 11
Tel. 0211/5504-0 · Fax 0211/550444

GFA SYSTEMTECHNIK