

PRISMA

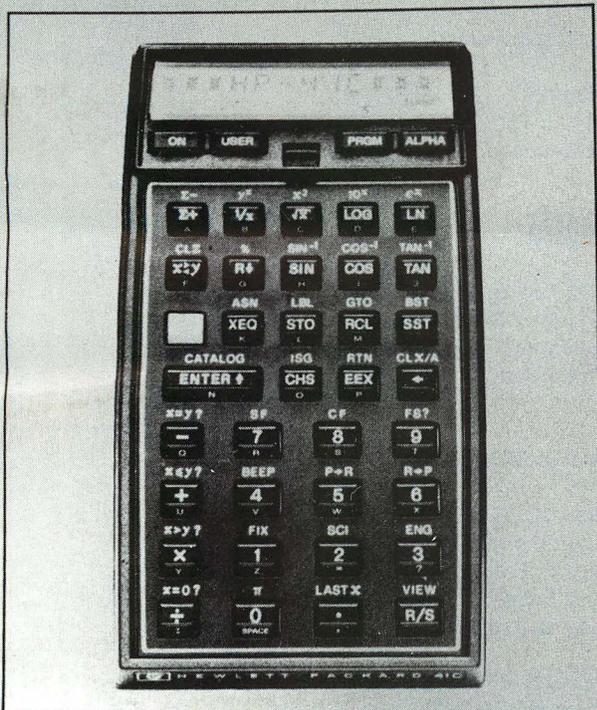
Computerclub Deutschland e.V. · Postfach 11 04 11 · Schwalbacher Straße 50 · D-6000 Frankfurt am Main 1

November/Dezember 1988 Nr. 6

D 2856 F

Best of

PRISMA



CCD - Computerclub Deutschland e.V.

Magazin

Clubbörse
Artikel selbst gemacht
Testbericht:
Zenith SupersPort 286

Praxis

Dateiverwaltung unter WORD 4.0

Serie 70

Untermenüs mit MENU
Forth Decompiler
LEXNEED
Lexfile HMSLEX
Schaltungsanalyse
QuietJet Steuerung
Lohnsteuer (Österr.)

Serie 40

Barcodes, was ist das?
Etiketten drucken
Romberg-Integration
Best of PRISMA
Konjugierte Gradienten
Zeitanzeige
Tips & Tricks
Zeilenlineal
Schreibschrittlineal
ASCII-Records
XROM (neue Variante)
Bessel und Neumann
Jahreskalender

CLUBBÖRSE

Suche **CCD-Modul** (Version B) für HP-41 CX, 110,- DM
Stefan Schneider, Wiesbaden, ☎ 06121/840983

HPIL Converter HP-82166 A	DM 280,-
HP-IL ThinkJet HP-2225B	DM 540,-
HP-IL/RS-232C Interface HP-82164A	DM 350,-
HP-IL Video-Interface HP-92198 (80 Col)	DM 480,-
Monitor 12" Highscreen amber	DM 120,-
für HP71:	
HPIL-Modul	DM 160,-
Dual-HPIL-Adapter	DM 140,-
G. Hebeckerl, ☎ 06103/81246	

HP-41 CV, Card Reader, Printer, Math Pac, viel Zubehör günstig abzugeben (auch einzeln),
Michael Kirchner (421), Goethestr. 29, 3200 Hildesheim, ☎ 05121/55398

Verkaufe HP-27S (180,-), für HP-71 64K RAM (Front Port; 600,- DM) 32K Ram (FP240,- DM), PAC-Screen Video-Interf. (V2.0 550,- DM).
Jochen Haas, Köln, ☎ 0221/519870

Für **separ. Berechn.** in organ. - chem. **Massenspektroskopie** (HA und NA) 'en bloc' abzugeben: **Software für HP 41** für umfangreiche Berechnungen (Präzisionsmasse, Intensitäten von Isotopen (≤41 amu), Simulation des Intensitätsverlaufs mit ≤15 superponierten Normalverteilungen, Summenformel, u.v.a.m.) und **Hardware:** HP 41 CV incl. Card Reader (82104A), X-Function-Modul (82180A), Math-I-Modul (14023) und Thermostreifendrucker (82143A) komplett mit Handbüchern, zugehör. Literatur! Preis nach VB. Anfragen schriftl. an Dr. E. Monter, Zeisigweg 6, 5653 Leichlingen - 1 (CCD 1833).

Suche: funktionstüchtigen HP-IL Konverter (HP 82166A) oder HP-IL/GPIO Interface (HP 82165)
Peter Jochen, ☎ 07121/630163

Verkaufe: „Tricks, Tips, und Routinen für Taschenrechner HP 41“ von Dalkowski, „Optimales Programmieren mit dem HP 41“ von Kruse Bücher im Bestzustand, je 10 DM.
Peter Jochen, ☎ 07121/630163

Verkaufe HP71B mit IL-Modul	700,00 DM
Forth/Assembler Modul	150,00 DM
Speichererweiterung 32KB, Port	200,00 DM
Diskettenlaufwerk 9114A mit Netzteil	1000,00 DM
ThinkJet 2225B mit Ständer, Druckkopf und Papier	500,00 DM
Mathe-Modul	150,00 DM
Dipl. Ing. Reinhard Schmidt (2053)	
Heinrich Sträter Str. 1, 4600 Dortmund 50	

HP 150 Software, Drawing Gallery Handbuch, Mouse usw. zu kaufen gesucht. ☎ 0201/72007-37 oder 0201/267049 Meichsner (1620).

Verkaufe wegen Systemaufgabe: HP-86 Computer 86 kB mit 256 kB Speichererweiterung, Monitor 82913A, Drucker 82905B, Floppy-Laufwerk 9121, Plotter 7475A für 2.250,- DM VB, I. Abidin ☎ 02234/74320

HP-75 C Computer komplett, sowie:
ROMs: I/O, Mathe, Text-Formatter
RAMs: 8 kB Extension Modul incl. ca. 500 kB Software (HP und eigen) für **Mitglieder zus. nur 650,- DM**
HP-9114B IL-Disketten-Laufwerk komplett, sowie:
2 Pack 3.5" Disketten (640 kB DS-Typ/orig. HP) für **Mitglieder zus. nur 700,- DM**
HP-2225B IL-ThinkJet-Drucker komplett mit Acrylständer für **Mitglieder zus. nur 650,- DM**
Alle Geräte sind in absolut neuwertigem Zustand!!
Peter Behrend (508), 2800 Bremen 1, ☎ 0421/505117

Verkaufe HP71 Version 1BBBB mit Manuals, DM 750,-; Math. Modul mit Manuals DM 150,-; FORTH-ASSEMBLER Modul mit Manual DM 150,-; Netzadapter DM 30,-; alles zusammen DM 1000,- statt DM 1080,-. Christine Gabor, ☎ 08231/7859 zw. 19 u. 23 h

Verkaufe:
HP ThinkJet (HP 2225B-IL) mit 1300 Blatt HP-Papier 500,- DM
HP-IL-Modul 125,- DM
Bar-Code-Leser 150,- DM
Standard-Modul 30,- DM
Statistik-Modul 30,- DM
Mathe/Statistik-Modul 50,- DM
Plotter-Modul 100,- DM
HP41 Akkusatz + Netzteil 60,- DM
Bücher:
HP41 Barcodes mit dem HP-IL-System 25,- DM
Wickes Synthetische Programmierung 10,- DM
Das HP-IL-System 15,- DM
Oliver Rieck, 5300 Bonn 2, ☎ 0228/354303 abends

CMT 64K-RAM-Modul für Kartenleserport sucht Manfred Hammer (2743) Oranienstraße 42, 6200 Wiesbaden, ☎ 06121/375294

Verkaufe HP 2225 B ThinkJet (IL) neuwertig, DM 650,-. Volker Lang (130) ☎ 07138/5349 nach 18 Uhr.

Suche Software, User und Hardware im Raum Köln/Düsseldorf zum APPLE IIc; IL-RS232C-/IL-Parallel-Interface. CCD 1092, R. Just, ☎ 01201/546842

Hewlett - Packard - Verkauf, Modell: HP 71B (neuwertig).

Gerät	Artikel-Nr.	Zusätzlich
Digital Cass. Drive	82161A	
HP/IL Schnittstelle	82401A	
8K Mem. Module	82420A	
HP-71/41 Translator	5061-7269	Handbuch
HP-71 Math	5061-7226	Handbuch
HP-71 Forth	5061-7234	Handbuch
HP-71		Benutzerhand
		Referenzhand
		Probleme der
		Statik
		Lehrbuch 1
		Lehrbuch 2

„Prgrm. in Forth“ Forth, Inc Lehrbuch 1
„In Forth denken“ Forth, Inc Lehrbuch 2
alles zusammen kostet ca. 1600 DM:
K. Maier ☎ 06032/35706

Suche für HP 41 Laufwerk 82161 A evtl. mit Kassetten, Portextender CT 4100, IL-Development-Modul, AEC-Rom, PPC-Rom, mit Handbüchern u. in funktionstüchtigem Zustand. ☎ 0261/63305 abends.

Regionalgruppe Köln

Der Ansprechpartner ist:

Frank Ortmann
Okerstr. 24
D-5090 Leverkusen

April 1989

Für den kommenden April sind bereits einige wichtige und interessante Clubaktivitäten vorgesehen, die am besten gleich im Terminkalender eingetragen werden sollten:

Mitgliederversammlung

Am Samstag den 22.04.1989 um 11:00 Uhr findet wieder die jährliche Mitgliederversammlung in Frankfurt statt. Die Einladung mit der Tagesordnung wird im nächsten PRISMA abgedruckt.

Erster MS-DOS Workshop

Thema: Microsoft WORD 4.0

Im Anschluß an die Mitgliederversammlung wird neben anderen Veranstaltungen auch der erste MS-DOS Workshop stattfinden. Dort können zusammen mit WORD-Experten Probleme der Textgestaltung u.a. durchgearbeitet werden.

Wenn Sie Interesse daran haben, senden Sie jetzt schon eine Postkarte oder einen Brief, in welchem kurz das WORD-Problem geschildert ist, an:

Redaktion PRISMA
Stichwort: Workshop
Postfach 11 04 11
D-6000 Frankfurt 1

(ant)

Regionalgruppe Hamburg

Die Ansprechpartner sind:

Alfred Czaya
An der Bahn 1
D-2061 Sülfeld
Tel. (040) 433668 Mo.-Do. abends

Horst Ziegler
Schüslerweg 18 b
D-2100 Hamburg 90
Tel. (040) 7905672

Regionalgruppe München

Der Ansprechpartner ist:

Victor Lecoq
Seumestr. 8
D-8000 München 70
Tel. (089) 789379

Regionalgruppe Rhein-Main

Der Ansprechpartner ist:

Andreas Eschmann
Lahnstr. 2
D-6096 Raunheim
Tel. (06142) 46642

Impressum

Titel:

PRISMA

Herausgeber:

CCD-Computerclub Deutschland e.V.
Postfach 11 04 11
Schwalbacher Straße 50
6000 Frankfurt am Main 1

Verantwortlicher Redakteur:

Alf-Norman Tietze (ant)

Redaktion:

Hans Jürgen Hübner (hjh)
Klaus Kaiser (kk)
Michael Krockner (mik)
Martin Meyer (mm)
Henry Schimmer (hs)
Dieter Wolf (dw)

Herstellung:

CCD e.V.

Manuskripte:

Manuskripte werden gerne von der Redaktion angenommen. Honorare werden in der Regel nicht gezahlt. Die Zustimmung des Verfassers zum Abdruck wird vorausgesetzt. Für alle Veröffentlichungen wird weder durch den Verein noch durch seine Mitglieder eine irgendwie geartete Garantie übernommen.

Druck und Weiterverarbeitung:

Reha Werkstatt Rödelheim
Biedenkopfer Weg 40 a, 6000 Frankfurt

Anzeigenpreise:

Es gilt unsere Anzeigenpreisliste 3 vom Juni 1987

Erscheinungsweise:

PRISMA erscheint jeden 2. Monat.

Auflage:

3000

Bezug:

PRISMA wird von allen Mitgliedern des CCD ohne Anforderung übersandt. Ein Anspruch auf eine Mindestzahl von Ausgaben besteht nicht. Der Bezugspreis ist im Mitgliedsbeitrag enthalten.

Urheberrecht:

Alle Rechte, auch Übersetzung, vorbehalten. Reproduktionen gleich welcher Art - auch auszugsweise - nur mit schriftlicher Genehmigung des CCD. Eine irgendwie geartete Gewährleistung kann nicht übernommen werden.

Inhalt

Magazin

Clubbörse	2
Telekommunikation	3
Artikel selbstgemacht	4
Testbericht: Zenith SupersPort 286	6

Praxis

Dateiverwaltung und Dateioorganisation unter Microsoft WORD 4.0	7
---	---

Serie 70

Untermenüs mit MENU	10
DECOMP: Forth Decompiler	11
LEXNEED	16
Lexfile: HMSLEX	20
Schaltungsanalyse	21
QuietJet-Steuerung	24
Lohnsteuer (Österr.)	26

Serie 40

Teil IV: Barcodes, was ist das ?	27
Etiketten drucken	28
Die Romberg-Integration	29
Best of PRISMA	31
Konjugierte Gradienten	31
Zeitanzeige auf PAC-Screen	34
Tips & Tricks	34
Zeilenlineal	35
Schreibschriftlineal	36
ASCII-Record Berechnungen	36
XROM - einen neue Variante	37
Bessel und Neumannfunktion	37
Jahreskalender	39
Weihnachtsgrüße	40

Barcodes	40
----------	----

Clubadressen	42
--------------	----

Werbung für den CCD

Liebe Clubmitglieder,

es ist immer wieder wichtig, für unseren Anwenderclub im Bekanntenkreis Werbung zu machen, damit die Clubleistungen des CCD durch eine breite Mitgliederzahl getragen werden.

Ob Familie, Schule, Universität, Arbeitsplatz oder Geschätskollegen - der CCD ist für ALLE interessant, die sich mit HP-Taschencomputern oder auch mit PC's unter MS-DOS befassen. Außerdem befindet sich bei uns im Club auch noch eine ATARI-Gruppe, die sich bereits aus dem "Test-Stadium" herausentwickelt hat und mittlerweile einen regulären Disketterservice unterhält.

Alf-Norman Tietze

Telekommunikation

In unserer heutigen kurzlebigen Informationsgesellschaft spielt die Telekommunikation eine wichtige Rolle. Das fängt beim Telefon bereits an und geht über Telex und Telefax bis hin zur Direktverbindung zwischen zwei Computersystemen. Die Telekommunikation ist also nicht mehr wegzudenken, denn alte Informationen sind "Schnee von gestern" und helfen heute nicht mehr ein Problem zu lösen. Sei es, weil die Lösung bereits gefunden, oder weil das alte Problem durch ein neues und komplizierteres abgelöst wurde.

Telefon

Zur Förderung der Kontakte unter uns Clubmitgliedern bietet sich das Telefon als Instrument der Telekommunikation direkt an. Für Mitglieder, die sich von den Regionalgruppen kennen, und für alle Aktiven, die im PRISMA unter der Rubrik "Clubadressen" zu finden sind ist das meistens selbstverständlich.

Anders sieht das aus, wenn Leser schnell einen Autor erreichen wollen. Häufig sind bei den Artikeln keine Telefonnummern abgedruckt, und die Kommunikation per Briefpost ist bekanntlich mit diversen Schwierigkeiten behaftet. Der Zeitbedarf ist unverhältnismäßig groß und auch der Ansprechpartner muß sich zu einem Antwortschreiben Zeit nehmen. Das schreckt meistens ab und es kommt keine Kommunikation mehr zustande - schade !

Weder der Redaktion noch der Clubverwaltung sind die Telefonnummern bekannt. Anfragen sind also zwecklos. Im PRISMA können wir nur Telefonnummern veröffentlichten, wenn sie von den Autoren freiwillig angegeben werden. Die Angabe der Telefonnummer hat aber manchmal auch negative Folgen. Dann nämlich, wenn nachts - Computerleute werden meistens erst nachts munter - ein Programm aus PRISMA probiert wird, dieses nicht so recht funktioniert und die einzige Abhilfe ein sofortiger Anruf beim Autor zu sein scheint. Dieser freut sich dann gar nicht darüber, daß er so sehr gefragt ist und wird beim nächsten Artikel seine Telefonnummer nicht mehr angeben. Das sind leider keine erfundenen Geschichten, sondern es ist Clubrealität, wie sie der Redaktion von einigen Autoren mitgeteilt wurde.

Auch wenn das Telefon eine tolle Erfindung ist, sollte sich trotzdem jeder kontrollieren, ob er möglicherweise zu einer

"unmöglichen" Zeit anruft und damit eine - unnötige - Belästigung auslöst.

Mailbox

Der CCD bietet nun schon seit einiger Zeit erfolgreich seinen Mitgliedern eine vergünstigte Teilnahme an der professionellen GeoNet Mailbox an. Diese elektronische Post (E-Mail) ist schnell und bietet sehr interessante Möglichkeiten für die Telekommunikation.

Die Benutzer von E-Mail haben es im Vergleich zu den Telefonanrufern direkt "idyllisch". Sie sind immer sicher, daß ihre Nachricht auch ankommt und können sogar kontrollieren, wann sie gelesen wurde. Eine Nachricht kann in der tiefsten Nacht versendet oder empfangen werden, ohne daß dabei irgendjemand gestört wird. Außerdem sind zur normalen Geschäftszeit die Gebühren für eine Nachricht günstiger als ein Ferngespräch mit dem Telefon - man bedenke, daß die CCD-Mitglieder ja in ganz Deutschland und im europäischen Ausland verteilt sind.

Darüber hinaus bietet die Mailbox noch viele andere Vorteile, wie z.B. der indirekte Zugriff auf die Telex-, Teletex-, Telefax- und BTX-Dienste der Deutschen Bundespost, ohne dafür weitere gebührenpflichtige Geräte zu besitzen. Mit der Mailbox läßt sich bei Bedarf sogar die "große weite Welt" der Datenbanken erschließen.

Als Hardware eignet sich jeder beliebige Personal Computer mit einem entsprechenden Terminal- oder Datenübertragungsprogramm. Die Verbindung zum Telefonnetz kann mit einem Akustikkoppler oder einem Modem erfolgen. Wie es aber bereits nicht nur Piloten aus unserem Club vorführen, muß man übrigens keinen Personal Computer haben, um an ein Mailboxfach zu kommen. Es eignet sich dazu auch der HP-71 mit Drucker und RS232-Schnittstelle.

Alle Unterlagen über die Voraussetzungen zu einer Teilnahme an der GeoNet Mailbox können beim ersten Vorsitzenden Wolfgang Fritz oder beim ersten Beisitzer Werner Dworak (siehe Clubadressen) per Postkarte oder Telefon angefordert werden - aber bitte nicht zu einer "unmöglichen" Zeit anrufen.

Alf-Norman Tietze
(Chefredakteur)

Artikel selbst gemacht

Ich möchte an dieser Stelle einmal wieder für alle Neuankömmlinge beim **CCD** und für die alten Hasen als Gedächtnisstütze die Vorgehensweisen beim Entstehen eines Artikels umreißen und den Ablauf schildern, den ein Manuskript in der Redaktion durchläuft, bis das Werk dann schließlich und endlich zur Veröffentlichung in der Clubzeitschrift **PRISMA** gelangt.

Uff, das war natürlich mal wieder ein herrliches Beispiel, wie man einleitende Worte **nicht** gestalten sollte, ein solches Monstrum von Wortgebilde erschlägt Jeden, der den Versuch macht einen Text zu lesen.

Ich glaube, es ist besser, wenn ich zu allererst einmal kurz einige Punkte zur allgemeinen Klarstellung aufzähle:

1. **Alle Clubmitglieder** können Artikel an die Redaktion einsenden.
2. Das **Manuskript kann**, wie sein Name ja eigentlich schon sagt, durchaus **mit der Hand geschrieben** sein, wenn dies ein normaler Sterblicher noch lesen kann, nicht jeder besitzt schließlich eine Schreibmaschine oder einen größeren Computer mit Drucker. Das Beste ist natürlich eine gedruckte Ausführung des Artikels, dazu aber später.
3. Ein **Artikel ist nicht nur ein Programm**, sondern jede irgendwie gearbete Sammlung von Informationen zu einer Problemstellung oder einem gemeinsamen Thema, von dem man annimmt, daß sie für mehr als ein Mitglied von noch so geringen Interesse sein kann.

Mit diesen Punkten möchte ich all die schlafenden Geister wecken, die meinen, "... meine Ideen interessieren ja sowieso niemanden, blubber, blubber ...".

So, kommen wir zu den Bestandteilen eines Artikels:

1. Der **Text**, d.h. die Beschreibung
 - a) des geschriebenen Programms
 - b) eines Problems, mit dem man sich an andere Mitglieder wenden möchte
 - c) einer Problemlösung aus egal welchem Wissensgebiet mit Hilfe des HPxx (41, 71, 75,80, 85, 86, 10C-16C, 20er und 30er und auch MARYs II, von dem hat man ja schon lange nichts mehr gehört, ich glaube man kann ihn durchaus zur Familie der Handhelds mit HP-Qualitäten zählen)
 - d) eines Gerätes, das gerade neu auf den Markt gekommen ist und in das Themengebiet von PRISMA paßt (Handhelds von HP, Peripherieprodukte zu diesen Geräten, ebenso Peripherieprodukte von HP, die man

an andere Rechner anschließen kann). Hier geht es in erster Linie um Entscheidungshilfen für andere Clubmitglieder bei einem anstehenden Kauf von neuen Geräten, warum soll man da eigene leidliche oder erfreuliche Erfahrungen für sich behalten?

- e) eines Fehlers eines Rechners (Bugs), dessen Ursache man entdeckt hat, und vor dem man die anderen Clubmitglieder warnen möchte
- f) und was Euch noch so alles einfällt, ich lasse hier Eurer Fantasie freien Lauf.

Denkt bei der Erstellung dieses Textes immer daran, daß ihn auch ein absoluter Laie verstehen können sollte, also jemand, der vielleicht erst mit dem Programmieren angefangen hat und natürlich nicht gleich von 0 auf 100 loslegen kann; bei einer Programmbeschreibung auch lieber etwas zu viel oder detailliert beschreiben, es könnte sonst ein vielleicht elementarer Hinweis verlorengelassen.

Man selbst geht immer davon aus, daß die Bedienung des eigenen Programms so furchtbar einfach ist, "... man hat es ja schließlich schon so oft gemacht, und nie ist etwas schief gegangen ...".

Ein Tip von mir: Gebt das Programm irgend jemand völlig unvorbelastetem in die Finger, Ihr werdet Euch wundern, was der alles anmeckert und falsch macht. ...

2. Das **Programmlisting** sollte auf schwarzes Papier gedruckt werden, hat jemand nur blaues, dann machen wir es in der Redaktion noch einmal neu, das blaue aber bitte als Gegenkontrolle trotzdem mitsenden.

Wer keinen Drucker besitzt, kann natürlich auch kein Listing anfertigen, auch klar. Solange nicht extrem seltene Einsteckmodule verwendet wurden können wir in der Redaktion das Listing jederzeit neu anfertigen.

Listings bis etwa 500 Zeilen im normalen zeilenweisen Ausdrucksmodus, längere Programme dann besser in kompakter Form, auch wenn sie schlechter zu lesen ist.

3. Die **Datenträger** können Magnetkarten, Digitalkassetten und Barcodelistsings sein, wobei letztere die sicherste Methode sind. Mit Magnetkarten gibt es doch öfter mal Schwierigkeiten mit dem Einlesen, jedes Staubkorn kann da schon tödlich sein.

Wir senden übrigens alle Datenträger außer Barcodes wieder zu Ihren recht-

mäßigen Besitzern zurück, das Zeug ist ja nicht gerade billig, das ist klar.

4. Die **Bilder oder Zeichnungen** (sehr erwünscht) sollten maximal so breit wie das äußere Spaltenmaß sein (linker Rand/linke Spalte bis rechter Rand/rechte Spalte), sonst müssen wir diese verkleinern. Ansonsten gibt es da keine besonderen Einschränkungen, was die Ausmaße betrifft, optimal, wenn die Bilder/Zeichnungen in die 2-spaltigen oder 3-spaltigen Spalten hineinpassen, die Maße einer Spalte betragen 87mm für 2-spaltigen Satz und 57 mm für 3-spaltigen Satz.

Sollte jemand Schwierigkeiten mit dem technischen Zeichnen haben, macht nichts, notfalls zeichnen wir das Ganze auf einem CAD-System neu, das sieht dann aus wie „geleckt“.

Gut, der Artikel ist nun fertig, man tütet ihn ein und trägt ihn zur Post, bitte nur an das Postfach schicken!

Dieter Wolf leert mindestens einmal die Woche dieses Postfach aus, bis dieser Artikel dann in die Redaktion kommt kann es also schon ein paar Tage dauern, dies gilt natürlich für alle Post dorthin.

Je nach Thema wird dann dieser Artikel vom entsprechenden Redakteur(en) bearbeitet, d.h. der Text wird auf Rechtschreibung durchgelesen und eventuell korrigiert, der Inhalt sollte nicht verändert werden. Eventuell müssen noch Bilder oder Zeichnungen überarbeitet werden, ein neues Listing angefertigt werden.

Der Artikel wird jetzt mit einem Formblatt mit den wichtigsten Daten versehen und in die Setzerei nach **Neu-Isenburg** gebracht. Hier wird der Text im Fotosatz eingegeben, d.h. in einen Computer eingetippt, der als Drucker auf Fotopapier belichtet, dies mit einer Auflösung von mehr als 1000 x 1000 Punkte pro Inch, ein Laser-Drucker schafft gerade 300 x 300!

Die **Setzer** sind **ganz normale Menschen**, die nicht die entfernteste Ahnung von einem HP41 oder HP71 haben, d.h. wenn sie etwas nicht lesen können, dann wird dies eben nicht gesetzt, fertig.

Ist der Text dann gesetzt, so muß man diesen abholen und auf Fehler durchlesen und korrigieren, die Setzer machen wegen der Unkenntnis der Materie natürlich auch einige Fehler, vor allem bei komplizierten mathematischen Formeln. Beim Korrekturlesen sieht man natürlich auch nicht alle Fehler, wir sind auch nur Menschen, scheut Euch aber nicht auf solche in Euren Artikeln hinzuweisen.

Jetzt wird die korrigierte Druckfahne wieder zurück in die Setzerei gebracht und korrigiert, erst danach steht sie für den „Umbruch“ zur Verfügung.

PRISMA entsteht dann folgendermaßen:
Zuerst werden die Druckfahnen kopiert und zur Probe zu einem Heft grob zusammengeklebt, dies dauert schon 1-2 Wochen, da wir alle noch einen Hauptjob haben und all dies abends in unserer Freizeit machen, da steht auch nicht jeder Abend zur Verfügung, ich selbst verwende meist die Hälfte meiner Wochenendfreizeit für Redaktionszwecke auf, Dieter Wolf geht es nicht besser, abgesehen von seiner Arbeit als „Finanzchef“ des Clubs.

Nach dem Probeumbruch steht dann etwa die Zahl der Artikel für diese Ausgabe fest, man kann ja immer nur Vielfache von 4 Seiten produzieren. Jetzt werden die Programme für den HP41 auf Datenträgern zusammen auf eine Kassette kopiert und

entweder bei ANT oder bei mir (Datenbarcodes) zu Barcodes gemacht. Diese sind aber noch zu groß und müssen noch fotografisch verkleinert werden.

Jetzt kann man auch den Platz hierfür. Die Montage, d.h. das Kleben der Originaldruckfahnen- und Zeichnungen auf Klarsichtfolien dauert dann noch einmal etwa 1 Woche. Erst jetzt kann das Machwerk in die Druckerei gebracht werden, hier werden die Druckvorlagen, d.h. das geklebte Original, gerastert und gedruckt, dies dauert auch einige Tage. Fotos im PRISMA müssen wegen der Rasterung schwarz/weiß sein, die Druckmaschine kennt keine Graustufen.

Nach dem Zusammenheften und Schneiden werden dann die Adressenaufkleber

auf die Hefte geklebt, die Hefte dann nach Postleitzahlengebieten sortiert zur Post gebracht, die Post verlangt dies für „Postvertriebsstücke“ oder Streifbandzeitungen, da kostet der Versand weniger. Der Briefträger hat jetzt drei Tage Zeit, diese Hefte an die Empfänger zu liefern, wegen des billigeren Versandweges ist die Streifbandzeitung also langsamer als Briefe, in den Großstädten kann sie also auch eher da sein als „auf dem Lande“.

So, jetzt habt Ihr wieder die neueste Ausgabe von PRISMA in den Händen, womit dieser Artikel ein natürliches Ende findet.

Martin Meyer (1000)



Testbericht Zenith SupersPort 286

Vor kurzem konnte ich mir einen Zenith SupersPort 286 ausleihen, um eine Weile damit herumzuspielen. Gleich als erstes bekam ich einen großen Schreck und hatte schon Angst, etwas kaputt gemacht zu haben. Ich konnte nämlich den Rechner nicht aufklappen! Als ich ihn aber auf den Tisch gestellt hatte, ging es ganz leicht. Gegenprobe. In der Tat, senkrecht stehend läßt er sich nicht aufklappen. Ein guter Transportschutz, der Liebe zum Detail zeigt.

Dann eingeschaltet. Schnelles Aufstarten (neudeutsch booten) von der 20-MByte-Festplatte. Und eine Anzeige, die sich vor MDA oder Hercules nicht zu verstecken braucht. Dunkelblau auf Weiß. Einen Schritt nach rechts, einen Schritt nach links, in die Knie gegangen, an die Decke gesprungen, Anzeige immer noch fabelhaft zu lesen. Und der Schirm ist fast quadratisch wie eine Bildröhre. Ein Kreis ist auf dem LCD-Schirm wirklich ein Kreis und kein Oval.

Der Schirm kann (intern) 640x400 Punkte darstellen, die jedoch leider nur für die wirklich hervorragende Schrift genutzt werden. In Grafik-Modus ist nur CGA angesagt und kein Hercules oder gar VGA, ein Zugeständnis an den Stromverbrauch. Allerdings gibt es auch von der Konkurrenz nur ganz wenige Aktentaschenrechner (Laptops), die mehr als CGA können. Bei den „Farben“ lassen sich 4 Helligkeitsstufen unterscheiden. Nicht gerade ideal, aber bei fast allen Programmen läßt sich aus der Palette der Zuordnungen etwas Brauchbares finden. Und meistens kommt man mit ein und der selben Einstellung aus.

Eine Besonderheit hat der Bildschirm noch zu bieten. Wenn man den Rechner ortsfest einsetzen will, kann man die Anzeige flach nach hinten wegklappen, so daß sie unter einem Monitor-Untersatz verschwindet und nicht stört. Im Vergleich zu den Vorgängern ist der SupersPort 286 deutlich kleiner und leichter geworden, aber ich zähle ihn doch eher zu den „Schlepp-Top“, wie so schön über diese Rechnerklasse gelästert wird. Ohne Akkus bringt er 4,9 kg auf die Waage, mit Akku etwa 7,0 kg. Dazu kommt gegebenenfalls noch die Tragetasche und Zubehör. Auf kurzen Wegen zu Bahn, Flugzeug oder Auto kein Problem, aber zu einer Bergtour möchte ich ihn nicht mitnehmen müssen!

Der Akkukasten ist abnehmbar, und beim ortsfesten Betrieb muß man ihn sogar abnehmen, um an die Buchsen für externen Monitor und externes Disketten-Laufwerk heranzukommen. An die parallele Schnittstelle (Drucker) und an die serielle Schnittstelle (Modem, Maus) kommt man auch mit Akku ran. A propos Modem. Es ist ein Fach für ein eingebautes Modem vorgesehen, das allerdings (noch) nicht den Se-

gen der Bundespost hat, so daß zur Zeit dieses Fach in Deutschland leer bleibt. Das externe Netzteil ist als Schaltnetzteil angenehm leicht. Es ist ohne Umschaltung von 90 V bis 260 V benutzbar und kann den Rechner mit und ohne Akku betreiben. Der Akku wird auch bei eingeschaltetem Rechner geladen.

Und schnell ist der Henkelmann (ja, einen Griff zum Wegschmeißen hat er auch, unter der Tastatur ausklappbar). Ich habe mir die Werte der einzelnen Benchmarks nicht detailliert notiert, aber sie lagen deutlich überm Durchschnitt für 12-MHz-Geräte. Das kommt hauptsächlich daher, daß stets mit 0 Wartezyklen gearbeitet wird. Falls es nötig sein sollte (Spiele) oder um Strom zu sparen, kann man die Taktfrequenz auf 6 MHz herabsetzen. Auch die Festplatte läßt sich nicht lumpen. Mit 28 ms ist sie für einen batteriebetriebenen Rechner außerordentlich schnell. Das Diskettenlaufwerk im 3,5“-Format verdaut 1,44 MByte-Disketten, wie man es heute erwarten kann. Um die Batterie zu schonen, kann man die Festplatte nach einer einstellbaren Zeit ohne Aktivität stillsetzen lassen. Beim nächsten Zugriff wird sie selbsttätig wieder angeworfen. In gleicher Weise schaltet sich auch die Hintergrundbeleuchtung der Anzeige ab.

Der Rechner wird standardmäßig mit 1 MByte Hauptspeicher geliefert, der auf der Hauptplatine bis 2 MByte erweiterbar ist. 5 MByte sind in Vorbereitung. Wann zum Teufel gehen denn endlich mal die RAM-Preise herunter, damit man von üppigem Speicher nicht nur träumen kann. Geredet wird davon schon lange. Bloß gemerkt habe ich davon nicht viel. Den Speicher oberhalb 640 KByte kann der SupersPort 286 mit einem eingebauten EMS-Treiber LIM 4.0 verwalten. Etliche neuere Programme können auch mit Extended Memory (dem Speicher eines AT oberhalb 1 MByte) direkt etwas anfangen, viele Programme benötigen Expandet Memory (in ein Fenster zwischen 640 K und 1 M eingebundener Speicher bei XT und AT). Hier ist also beides möglich.

Recht geschickt ist die Tastatur des SupersPort ausgelegt. Wegen der gewünschten kleinen Abmessungen des Rechners kommt man ja mit einer gewöhnlichen Tastatur nicht hin. Das Haupt-Tastenfeld hat die normalen Abmessungen und es sitzen alle Tasten an der gewohnten Stelle in der üblichen Anordnung. Das gilt auch für die Tasten Shift, TAB, CTRL, ALT, <— und ENTER, die wie bei einer „großen“ Tastatur die doppelte und dreifache Breite haben. Darüber sind die Funktionstasten F1 bis F10 und einige Steuertasten angeordnet. Die Pfeiltasten sitzen als umgekehrtes T in der rechten unteren Ecke, sehr gut zu bedienen. Platz für ein getrenntes Numerikfeld ist natürlich keiner. Aber mit einer weiteren Umschalttaste namens „FKT“ läßt sich ein Bereich der Tastatur

als Numerik-Feld umdefinieren. Wer unbedingt will, kann extern ein numerisches Zehnerfeld anschließen. Über die FKT-Taste lassen sich noch weitere „fehlende“ Tasten erreichen und einige Sonderfunktionen bedienen. Das einzige, was mir mißfällt, sind die INS- und DEL-Taten im rechten oberen Eck. Ich bin sie halt unten gewohnt. Na ja, bei einer MF2-Tastatur sitzen sie beim Cursor-Feld auch weit oben.

Ganz hervorragend sind die mitgelieferten Handbücher. Es sind die gleichen wie bei den „ausgewachsenen“ Rechnern. Für ausgezeichnete Handbücher war Zenith ja schon immer bekannt. Ebenfalls schon Tradition sind Setup, Diagnose und Debugger im ROM. Da steht Zenith bisher einzigartig da, wenn auch andere Hersteller allmählich ebenfalls auf diesen Trichter kommen.

Ein Wermutstropfen ist natürlich der empfohlene Preis von ca. 9900 DM (mit Festplatte von 40 MByte ca. 11000 DM). Aber vergleichbare Rechner anderer Firmen kosten das gleiche oder mehr.

Um gerecht zu sein, muß man 4 Rechnerklassen unterscheiden. Einmal die Tischrechner (Desktop), die man allenfalls ein paar Schritte zum Auto schleppen kann. Dann die Koffergeräte (Portable) Sie sind einerseits deutlich transportfreundlicher und man kann sie auch ganz gut mit Bahn oder Flugzeug mitnehmen, andererseits erfüllen sie (fast) den vollen Standard eines Tischgeräts. Schließlich die wirklich tragbaren handlichen Aktentaschenrechner (Laptop), die man – sofern sie Batterien haben – auf dem Schoß betreiben kann. Und das ist der Unterschied der beiden Klassen: Betrieb nur am Netz oder auch überall unterwegs.

In dieser Reihenfolge steigt bei sonst vergleichbarer Ausstattung der Preis. Ein Koffergerät ist wenig teurer als ein Tischgerät. Es wird im Grunde mit der gleichen Technik gearbeitet, lediglich etwas robuster und enger. Ein netzabhängiger Aktentaschenrechner muß schon wesentlich aufwendiger sein. Es müssen bestimmte Bauteile und Baugruppen benutzt werden, damit man mit weniger Platz und Leistung auskommt. Denn diese Geräte sollen keinen Lüfter benötigen und dürfen deshalb nicht zu viel Wärme entwickeln. Richtig interessant wird es bei den batteriebetriebenen Geräten. Hier muß mit allen faulen und erlaubten Tricks gearbeitet werden, um Strom zu sparen und damit eine lange Betriebsdauer zu ermöglichen. Und das kostet. Beim Stromsparen hat sich Zenith einiges einfallen lassen, wie oben schon angedeutet. Und das macht sich bezahlt. Auch bei heftigem Betrieb lassen sich leicht über 4 Stunden Betriebszeit erreichen.

Werner Dworak (CCD 607)

Microsoft®

Microsoft Word Version 4.0

Dateiverwaltung und Dateioorganisation unter Microsoft WORD 4.0

von Alf-Norman Tietze

Die nachfolgende Ausarbeitung zur Dateiverwaltung unter Microsoft WORD 4.0 ist für allgemeine Belange zusammengestellt und kann – bei Übertragung des Prinzips – an beliebige Erfordernisse angepaßt werden. Der Artikel wendet sich an den bereits im Umgang mit WORD vertrauten Anwender. Die Erfahrungen mit dieser Verwaltungsmethode sind durchweg positiv. Alle Beteiligten (der Chef, die Schreibkräfte, die Sachbearbeiter) haben sich schnell damit zurechtgefunden.

Die Bezeichnungen der Unterverzeichnisse, die Kennbuchstaben in den einzelnen Dateinamen und die Schlüsselworte (Suchbegriffe) lassen sich auf individuelle Anforderungen anpassen. Begleitend zu dieser Dateioorganisation steuern spezielle Makro-„Programme“ per Tastendruck komplizierte Funktionen, wie z. B. eine graphische Übersicht der bereits vorhandenen Unterverzeichnisse oder die Suche eines bestimmten Begriffes aus dem Inhalt einer unbekannteten Textdatei „quer“ durch alle (!) WORD-Unterverzeichnisse etc.

Die Problematik

Alle Dateinamen können maximal 8 Zeichen lang sein und sollten keine Erweiterung haben, damit WORD automatisch die Namensweiterung „.TXT“ vergeben kann. Das begrenzt die Möglichkeiten für eine effiziente Dateioorganisation deutlich. Diese Begrenzung in der Namensvergabe der Dateinamen schafft ein ernstes Problem bei der Textverarbeitung. Es macht das „Wiederfinden“ von alten Dokumenten (Texten) oder deren Zuordnung zu bestimmten Themenbereichen anhand des Dateinamens fast unmöglich. Zur Abhilfe empfiehlt sich eine „dynamische“ Systematik – d.h. die Systematik muß flexibel genug sein, damit sie an zukünftige Erfordernisse angepaßt werden kann.

Als freiberuflicher EDV-Berater habe ich aus diesem Grunde für meine Kunden eine Systematik entwickelt, die die Eigenschaften

des Betriebssystems und des Word Datei-Managers miteinander kombiniert. Die Leistungsfähigkeit dieser Systematik ist von einer sorgfältig ausgedachten Verwaltungsstruktur abhängig.

Die Systematik

Zur Verwaltung der Textdokumente bietet sich folgende 3-gliedrige Systematik an:

Unterverzeichnisse Dateinamen Titel & Schlüsselworte

Unterverzeichnisse

Die Textdateien werden in – je nach Thematik verschiedenen – Unterverzeichnissen des WORD-Verzeichnisses abgelegt. Dadurch wird eine grobe Struktur geschaffen, die wesentlich zur Übersichtlichkeit beiträgt.

Dateinamen

Jede Datei innerhalb eines Verzeichnisses muß einen signifikanten Namen erhalten. Hier ist häufig ein Zahlenschlüssel günstig, der das Datum beinhaltet – zusammen mit einem oder zwei Kennbuchstaben. Der Dateiname hat somit nur noch eine technische Funktion, um die Datei von anderen zu unterscheiden. Die Identifikation erfolgt über Titel und Schlüsselworte.

Titel & Schlüsselworte

Mit Hilfe des WORD Datei-Managers erhält das einzelne Dokument einen „lesbaren“ Titel und es können Schlüsselworte (Suchworte) zugeordnet werden. Darunter fallen z.B. auch Autor und Bearbeiter des Textes. Außerdem kann der Datei-Manager

sowohl das Erstellungs- als auch das letzte Überarbeitungsdatum feststellen.

Die Anwendung

Voraussetzung für ein effizientes Arbeiten mit dieser Systematik ist die konsequente Verwendung des WORD Datei-Managers. Das WORD-Programm muß im WORD-Verzeichnis gestartet werden, damit alle Unterverzeichnisse im direkten Zugriff stehen. Die Dokumente sind thematisch in den entsprechenden Unterverzeichnissen vorgegliedert und werden nur (!) über den Datei-Manager gesucht und geladen. Auf diese Weise ist der umfangreiche (aussagekräftige) Titel des Dokumentes sichtbar, und das große Rätselraten um die Bedeutung eines Dateinamens entfällt. In den meisten Fällen wird ein Dokument anhand seines Titels zu identifizieren sein. Als zusätzliche Suchbegriffe stellt der Datei-Manager noch Schlüsselwörter, Autoren- und Bearbeiternamen, sowie Erstellungs- und letztes Überarbeitungsdatum zur Verfügung. Sollte sich auch damit ein bestimmtes Dokument nicht finden lassen, so kann direkt ein spezifischer Text in den vorhandenen (!) Dateien gesucht werden.

Die Verzeichnisse

Eine Verzeichnisstruktur kann z.B. wie folgt aussehen:

..\WORD\	Programm-Verzeichnis einschließlich aller Textbausteine und Druckformatvorlagen sowie Serienbrief-Steuerdateien
..\WORD\proj\	Projekt- oder Arbeitsverzeichnisse mit Projektbezeichnung (max. 8 Zeichen)
..\WORD\ANG _{jj} \	Angebote
..\WORD\BEST _{jj} \	Bestellungen/Reklamationen
..\WORD\BEW _{jj} \	Bewerbungen
..\WORD\BRIEF _{jj} \	Allgemeine Briefkorrespondenz
..\WORD\CCD\	Ausarbeitungen für PRISMA
..\WORD\ORI\	Originale von Serientexten u. a.
..\WORD\PERS _{jj} \	Personal und Mitarbeiter
..\WORD\PRIV\	Private Korrespondenz
..\WORD\RECH _{jj} \	Rechnungen
..\WORD\TEXT\	Sonstige Texte

jj = des Jahres "jj", z.B. "\ANG88\" für Angebote 1988

die Angabe der Jahreszeit ist optional.

Zugriff auf die Verzeichnisse beim Laden

a) Direkter Weg „Übertragen – Laden“

Mit **[Esc] [Ü] [L] verzeichnisname/ [F1]**, um ein Dokument zu laden,

oder

b) „Bibliothek – Datei-Manager – Suchen“

[Esc] [B] [D] [S] suchpfad [Return]. Um eine Übersicht einschließlich Autorennamen und Dokumententitel zu erhalten, muß die Anzeigeoption beim ersten Aufruf des Datei-Managers auf „Anzeige – Lang“ eingestellt werden. Mit dem Cursor kann dann das gewünschte Dokument ausgewählt und anschließend mit **[L] [Return]** geladen werden.

ACHTUNG: Der Datei-Manager verwendet immer den zuletzt definierten Suchpfad. Deshalb ist darauf zu achten, daß das gewünschte Verzeichnis auch im Suchpfad korrekt angegeben ist.

Der Datei-Manager kann nur zum Auffinden eines Dokumentes aber nicht zum Abspeichern benutzt werden. Zum Speichern muß der übliche Weg genommen werden.

Zugriff auf die Verzeichnisse beim Speichern

Mit **[Esc] [Ü] [S] verzeichnisname/dateiname [Return]**, wird eine Datei in das thematisch vordefinierte Unterverzeichnis gespeichert.

ACHTUNG: Dies ist der einzige Punkt, wo eine erhöhte „Benutzerdisziplin“ erforderlich ist. Wenn die Angabe des Unterverzeichnisses entfällt – weil sie z.B. vergessen wurde – speichert WORD die Datei im WORD-Verzeichnis ab. Die Bezeichnung des Unterverzeichnisses gehört somit also immer zur Benennung der Datei.

Die Dateinamen

WICHTIG: Der „lesbare“ Titel eines Dokumentes sollte **IMMER** deutlich im Datei-Manager angegeben werden. Dieser Titel sollte niemals mit dem Dateinamen identisch sein, denn das ist pure Verschwendung des Platzes für wichtige, „lesbare“ Zusatzinformationen.

Bei normalen Dokumenten darf – soweit möglich – jede beliebige Datei-Bezeichnung verwendet werden, solange kein Name innerhalb eines Unterverzeichnisses doppelt vergeben wird. Gültige Namen sind z.B.:

Meier-3, FAZ, CCD5000, schmitt7, EDV-Best usw.

Innerhalb eines Projekt- oder Arbeitsverzeichnisses ist jedoch strengstens darauf zu achten, daß immer unterschiedliche Namen für neue Dateien vergeben werden, da ein Projekt möglicherweise über viele Jahre bearbeitet wird. Ein Brief an den zuständigen Bearbeiter kommt dann nämlich mehrfach vor und ist über den Personen-Namen allein (z.B. MEIER) nicht mehr zu unterscheiden.

Deshalb ist hier als Dateiname das verschlüsselte Erstellungsdatum günstig – in der Form:

jjmmttxx.TXT (die Namenserweiterung ".TXT" wird von WORD automatisch vergeben)

jj = Jahreszahl, mm = Monatszahl, tt = Tageszahl und xx = beliebige Stellen (Ziffern oder Buchstaben) zur weiteren Unterscheidung der Dokumente vom gleichen Datum.

Folgende Vorzugsbuchstaben seien hier zur Verdeutlichung des Prinzips vorgeschlagen. Diese können mit einem zweiten Buchstaben oder einer Ziffer ergänzt werden.

- ! = Aktennote (Memorandum)
- A = Auftraggeber
- B = Benutzer
- E = Entwickler
- F = Familie
- G = Gutachter
- M = Mitglieder
- N = Notar, Anwalt, Gericht
- X = Sonstige

Der Kreativität sollte freien Lauf gegeben werden. So können die Buchstaben auch einzelnen Fachgebiete beschreiben etc.

Sonderfall: Rechnungen

Die Rechnungen für **ALLE** Projekte stehen zusammen in einem eigenen Verzeichnis "..\WORD\RECH_{jj}\ " für das jeweilige Kalenderjahr „jj“. Die Dateinamen sollten die Rechnungsnummer wiedergeben.

Z.B. "..\RECH88\R003.TXT" ist die Rechnungs-Nr. 003 für das Jahr 1988

Titel & Schlüsselworte

Der „lesbare“ Titel eines Dokumentes sollte **immer** deutlich im Datei-Manager angegeben werden. Dieser Titel sollte niemals mit dem Dateinamen identisch sein, denn das ist pure Verschwendung des Platzes für wichtige „lesbare“ Zusatzinformationen.

Beispiel:

Die Aktennote vom 9. Dezember 1988 über die zweite Mängelfeststellung vor Ort zum Projekt „Alte Mühle“ (Proj.-Nr. 8630) erhielt den Dateinamen „881209!1.TXT“

Es empfiehlt sich nicht als Titel „8630 Aktennote“ zu wählen, weil während des Projektzeitraumes zahlreiche Aktennoten geschrieben worden sind. Besser wäre folgender Titel (o. ä.) zu wählen:

„OT 2. Mängelfeststellung“ (OT = Ortstermin)

Das beschreibt den Textinhalt deutlich. Die Art des Schreibens und das Erstellungsdatum gehen bereits aus dem Dateinamen hervor:

881209!1.TXT für den 09.12.88 und die 1. Aktennote an diesem Tag.

Die Schlüsselworte haben eine wichtige Bedeutung bei der Zuordnung der Texte zu Projekten, Themengruppen, Art des Schreibens etc. – besonders dann, wenn die Texte nicht mehr im entsprechenden Projektverzeichnis gespeichert sind. Nach den Schlüsselworten kann im Datei-Manager gesucht werden. Groß- oder Kleinschreibung bleibt dabei unberücksichtigt.

0000 blabla...	Projekt-Nr. oder Bezeichnung
ak	Aktennote (Memorandum)
aufsatz	Aufsatz/Artikel für Presse

best
bew
büro
ccd
dbase
doku
edv
gericht
konzept
liste
lotus
notar
ori
personal
presse
prisma
privat
rechnung
reisen
serie
test
word

Bestellung
Bewerbung
Büroangelegenheit/-bedarf
Computerclub Deutschland e.V.
Betreff: dBASE
Anleitung/Dokumentation/Handbuch
EDV-Angelegenheit/-bedarf
Gericht-Korrespondenz
Konzept/Idee
Liste für Serienbrief (Steuerdatei)
Betreff: Lotus 1-2-3
Rechtsanwalt/Notar
Originalvorlage
Personalangelegenheit
Presse/Magazine/Zeitschriften
Artikel für PRISMA
private Korrespondenz
Rechnung oder Zahlung
Reisen oder Urlaub
Serienbrief
zu Testzwecken
Betreff: WORD

Weitere Ergänzungen

Noch leistungsfähiger wird der Umgang mit den Dateien, dem Datei-Manager und mit WORD überhaupt, wenn man sich die Möglichkeiten der Makroprogrammierung zu Nutze macht. Dann können per Tastendruck komplexe Befehlsfolgen automatisch ablaufen, um z.B. Umlaute für die Mailbox zu konvertieren, „Freitext“-Suche mit externen Hilfsprogrammen zu ermöglichen, den Datei-Manager zu konfigurieren und vieles andere mehr. Weiteres im Detail dazu wird im kommenden PRISMA-Heft zu lesen sein.

Alf-Norman Tietze
Thudichumstr. 14
6000 Frankfurt 90

Hardware · Software
Servicestation
Beratung · Zubehör

OSBORNE
Management by Computer.

WORDLORD · Textverarbeitung · CAD-Anwendungen · Komplettsysteme

PCE **PFORTNER GMBH**
Computer-Technik · Elektronik

Branchenlösung für Klein- u. Mittelbetriebe

Postfach 1220 · 4133 Neukirchen-Vluyn
Telefon 0 28 45/3 22 94



Sonderpreise für CCD-Mitglieder

Untermenüs mit MENU

"Ein Menüsystem für den HP-71"
PRISMA 5/88, Seite 5

Seit im letzten PRISMA von Matthias Rabe das Menüsystem für den HP-71 vorgestellt wurde bin ich überzeugter Anwender dieses kurzen, aber sehr nützlichen Hilfsprogrammes. Es ist tatsächlich so, daß man sich alle häufig genutzten Applikationen (Anwendungen) in einer menügerechten Auswahl individuell zusammenstellen kann. Und zwar nicht nur in einem Menü - nein, sondern in beliebig vielen Menüs.

Ein leicht zu übersehender Hinweis mit einem unscheinbaren Beispiel im letzten Artikel deutete diese interessante Fähigkeit an: "MNU kann sich auch rekursiv aufrufen". Da das eigentliche Menüprogramm MNU mit einem Textfile (besser: Menü-File) als Parameter aufgerufen wird, kann es sich natürlich auch selbst aufrufen - jedoch mit einem anderen Menü-File. Und dort kann es sich wieder mit einem weiteren Menü-File aufrufen und so weiter. Diese anderen Menü-Files sind dann sozusagen die Untermenüs, in denen jeweils die Anwendungsprogramme für thematische Bereiche zusammengefaßt sind.

Das klingt komplizierter als es tatsächlich ist. Ein kleines Beispiel soll hier für bessere Verständlichkeit sorgen.

Start mit CALL MENU, das automatisch MNU("MAINMNU") aufruft. Zur besseren Übersicht sind die Untermenüs und deren Aufrufe fett gedruckt.

Textfile **MAINMNU**
(Hauptmenü)

Datenbank @mnu("basemnu")
Texteditor @teddy
Mathematik @mnu("mathmnu")
Dual-Timer @timer
Hilfsprogramme @mnu("utilmnu")

Textfile **BASEMNU**
(Datenbank-Untermenü)

Termine @ccdbase("termine")
Adressen @ccdbase("adr")
Buchhaltung @mnu("buchmnu")
Neu/Sonstige @ccdbase("")
BASIC @system

Textfile **MATHMNU**
(Mathematik-Untermenü)

Integration @integ
Polynome @poly
Statistik @statpack
BASIC @system

Textfile **UTILMNU**
(Hilfsprogramm-Untermenü)

AutoLoop @autoloop("")
Druckereinstellung @prn
ListCopy @listcopy
DiskCopy @mcopy
Backup @backup
BASIC @system

Textfile **BUCHMNU**
(Buchhaltungs-Untermenü)

Reisekosten @ccdbase("reise")
Fahrtenbuch @ccdbase("km")
Bankkonten @ccdbase("konto")
BASIC @system

Alle Befehle hinter dem Klammeraffen '@' sind Programmnamen, die von MNU automatisch mit CALL aufgerufen werden. Normale BASIC-Befehle oder Funktionen können deshalb nicht direkt verwendet werden. Das ist aber nicht unbedingt von Nachteil, da man alle benötigten Befehle auch in ein Programm schreiben kann.

Mit dem Start von MENU wird nun automatisch MNU mit dem Menü-File MAINMNU aufgerufen. In diesem Menü stehen Datenbank, Texteditor, Mathematikprogramme u.a. zur Verfügung. Wenn nun "Datenbank", "Mathematik" oder "Hilfsprogramme" ausgewählt wird, dann wird MNU als Unterprogramm erneut aufgerufen, jedoch jeweils mit einem für die entsprechende Anwendung spezifischen Menü-File - BASEMNU, MATHMNU oder UTILMNU. Da sich die Datenbankanwendungen weiter untergliedern, gibt es ein zusätzliches Untermenü BUCHMNU zur Buchhaltung.

Auf diese Art und Weise lassen sich beliebige hierarchische Menü-Strukturen entwickeln, mit deren Hilfe man sich bequem durch das Angebot an eigenen Programmen tasten kann. Die gewünschte Applikation wird dann einfach mit [ENTER] gestartet.

Will man jedoch mal eben zwischendurch in die normale BASIC-Umgebung oder in den CALC-Modus, kann die ganze Sache lästig werden. Pro Menüebene darf dann einmal [ATTN] gedrückt werden bis endlich die BASIC-Ebene erreicht ist. Bei komplizierter Menüstruktur kann das schon drei bis fünfmal sein. Und anschließend darf man sich wieder auf die 5. Menüebene "herunter"-tippen. Auch dagegen gibt es Abhilfe, wie er durch den Menüpunkt

BASIC in den Untermenüs schon sichtbar wurde. Durch den Aufruf der nachfolgenden kleinen BASIC-Routine SYSTEM läßt sich eine einfache Unterbrechung erzeugen - und mit [RUN] geht es wieder weiter im Menü.

BASIC-File SYSTEM

```
10 CALL SYSTEM @ SUB SYSTEM
20 DEF KEY "#46","CONT"
30 U=FLAG(-9,1)
40 PUT "#43"
50 PAUSE
60 FLAG(-9,U)
70 DEF KEY "#46"
80 END SUB
```

Zeile 20: Die Taste [RUN] wird [CONT] belegt.

Zeile 30: U speichert den USER-Modus und USER wird angeschaltet.

Zeile 40: Aktiviert die Anzeige.

Zeile 50: Unterbricht das Programm.

Zeile 60: Der USER-Modus wird so wiederhergestellt, wie er vorgefunden wurde.

Zeile 70: Gibt die Taste [RUN] frei.

Alf-Norman Tietze
Thudichumstr. 14
6000 Frankfurt 90
Tel. (069) 789 3995



Decomp: Ein Forth-71 Decompiler

HP-71, FORTH

FORTH ist bekannt als Programmiersprache mit einem sehr steilen Lernpfad. Ist das vielleicht der Grund für die doch eher spärlichen FORTH-Beiträge in PRISMA? Für mich jedenfalls trafen die Konsequenzen dieser Ausgabe zu, dies umso mehr als das HP71-FORTH mit seiner 20-bit-(4 nibble)-Struktur des Adressraumes und seinen Text-Files anstelle von „Screens“ doch eher zu den Außenseitern gezählt werden muß. Andererseits ist dieses Forth mit seinen HP-41 ähnlichen Gleitkommafunktionen und den eigenständigen Stringmanipulationen doch als eine komplette Lösung anzusehen. Das Durchlaufen des Lernpfades habe ich noch nie beurt!

Eine ausgezeichnete Übung, und insbesondere um der speziellen Forth-Implementierung auf die Spur zu kommen, besteht im Schreiben eines Forth-Decompilers. Er dient vor allem für die Analyse der im ROM vorgegebenen Worte, aber auch von beliebigem, kompilierten Benutzer-Code.

Der vorgestellte Decompiler ist für die Forth-Version 2A des HP41-Translator Packs geschrieben. Für die Version 1A des FORTH/Assembler ROMs sind die im Anhang 1 angegebenen absoluten Codeadressen der Version 1A einzusetzen. Das Programm ermöglicht das Decompilieren beliebiger höherwertiger Wörter (secondaries) mit und ohne Vorspann (Header) und erzeugt ein komplettes Abbild der in eine Doppelpunktdefinition hineincompilierten Codewörtern mit ihren Aufrufnamen. Besitzt der zu decompilierende Codeeintrag keinen Namen, d.h. besitzt er keinen Vorspann wird anstelle des Namens die Codeadresse in der Form :addr in Hex angezeigt.

Die Ausgabe erfolgt wählbar auf einen HPIL-kompatiblen Monitor oder Drucker. Für eine ausschließliche Anzeige auf der LCD-Anzeige sollte die in Anhang 2 angegebene Tastaturabfrage in das Wort UN:A eingebaut werden, mit der dann von Zeile zu Zeile durch Drücken von Endline weiterschaltet werden kann.

Das Listing und die nachfolgenden Kommentare verwenden die üblichen Stackkommentare, ergänzt durch das was Wort im Eingabe-Strom erwartet, z.B:

(name (addr n1 — n2)

bedeutet, daß das Wort im Eingabe-Strom 'name' und addr und n1 auf dem Stack erwartet. Das Wort gibt n2 auf dem TOS (top of stack) zurück.

UN: kann Objekte folgender Definitionen decodieren:

Konstanten, Gleitkomma-Konstanten, Stringvariable, String-Arrays
Vokabulare, Variable und CREATE...DOES>-Konstruktionen.

Die Inhalte der letzteren werden je nach Länge des Arrays entweder als 5-Nibble Zahlen, falls Länge ohne Rest durch 5 teilbar, als Gleitkomma-Zahlen, falls Länge ohne Rest durch 8 teilbar, oder als Text mit Array-Länge(Nibbles)/2 Charakter angezeigt.

Erläuterungen zu den wichtigsten Wörtern

UN:

Für UNcompile, erwartet entweder den Namen des zu decompilierenden Wortes oder die CFA in HEX eines vorspannlosen secondaries. Existiert das Wort im Vokabular, so wird sein Definitionstyp mit WDEF bestimmt und anschließend mit UN:A das Parameterfeld oder der DOES>-Teil einer CREATE...DOES>-Anweisung decompiliert. Existiert das Wort im Vokabular nicht, so nimmt UN: an, das ihm eine HEXadresse folgt. Diese Adresse wird mit NUMBER DROP in eine einfache-genaue Zahl gewandelt und zusammen mit -1 für das willkürlich angenommene Vorrangigkeitsbit (Precedence bit) auf den Stack für die weitere Verarbeitung an UN:A übergeben.

Zu beachten ist die Initialisierung mit 0 der Variablen BR.ADDR, die immer die letzte Verzweigungsadresse eines Decompilierlaufes enthält. Diese Variable wird bei jeder erkannten Vorwärtsverzweigung mit der errechneten Verzweigungsadresse aktualisiert. Bei einem Strichpunkt (SEMI), das im Normalfall das Ende eines Wortes und demnach das Ende eines Decompilierlaufes signalisiert, wird zusätzlich diese Variable überprüft. Enthält sie eine höhere Adresse als die Adresse des Semi, fährt UN:A bis zum nächsten Semicolon weiter. Solche dem Semi nachfolgende Codesequenzen sind im ROM an etlichen Stellen zu beobachten.

UN:A

(UNcompiliere ab Adresse) ist die eigentliche Decompilier-Schleife in der ab einer

Aufruf-Worte und deren Syntax:

UN: (Name (—))	— decompiliert Forth-Wort Name
UN: (hexaddr (—))	— decompiliert vorspannloses Wort an Hexadresse
UN: (:hexaddr (—))	— bis zum Wortende (Strichpunkt (Semi)).
UN:A (addr —)	— decompilierte ab Adresse bis Doppelpunkt.
PRINT (Wort (—))	— Resultat von Wort auf Drucker. Folgt PRINT kein Wortname so erscheint Fehlermeldung: no argument

Adresse auf dem TOS alle CFA bis zum endgültigen Strichpunktcode decompiliert wird.

WDEF

decodiert die CFA des zu decompilierenden Wortes. Ist das Wort maschinencodiert (Primitiv), so bricht WDEF mit einer entsprechenden Meldung ab. Andernfalls decodiert .HEAD den Vorspann unter Angabe ob es vorrangig ist (Immediate) oder nicht sowie der Linkadresse und dem an dieser Adresse liegenden Wortnamen. Anschließend wird in einer großen CASE-Anweisung der Definitionstyp bestimmt und die Definitionsdetails ausgegeben. Die einzelnen CASE-Marken sind aus einem mit DEFW zu adressierenden Laufzeitcode-Array, dessen Einträge während der Compilierung von DECOMP bestimmt werden, in die CASE-Anweisung hineincompiliert.

Entspricht die zu decodierende CFA keiner dieser Marken, so nimmt WDEF eine Sprungadresse zu einer DOES>-Codesequenz an und überprüft ob das dieser Adresse vorangehende Codewort das Ende des Compilierteils der CREATE...DOES> Anweisung markiert. Trifft dies zu, wird das Flag BODY? auf Wahr gesetzt für die nachfolgende Decompilierung dieser DOES>-Sequenz. Andernfalls ist der Definitionstyp unbekannt und die CASE-Anweisung wird verlassen.

.HEAD

überprüft zuerst die Existenz eines zulässigen Vorspanns, indem es das an der CFA und NFA vergleicht (HEADER?). Stimmt die Distanz mit dem Zählbyte überein, so existiert ein gültiger Vorspann und der Wortname sowie die Linkadresse zum vorangehenden Vokabulareintrag unter Angabe des Namens ausgegeben.

WNAME

vergleicht den zu decodierenden Wert mit den Laufzeitcodes der systemeigenen Definitionswörter. Bei Übereinstimmung gibt es den entsprechenden Namen aus und justiert den Adressenzeiger falls notwendig. So wird z.B. bei einem Inline-String der Stringausdruck mit .S+ ausgegeben und der Adressenzeiger um die Stringlänge in Nibbles erhöht.

Wird kein Laufzeitcode erkannt, so liegt die Adresse einer normalen Doppelpunktdefinition vor und WDEF überprüft wiederum die Existenz eines normalen Vorspanns. Ist ein Vorspann vorhanden, ermittelt .NAME den Wortnamen, ansonsten wird der Codewert in Hex mit vorangestellten Doppelpunkt als Hinweis für ein vorspannloses Wort ausgegeben.

Bei gewissen vorspannlosen Wörtern kann es vorkommen, daß der entsprechende Code den Adressenzeiger auf einen unbekanntem Wert erhöht. Dabei kann UN: aus seinem Tritt fallen, es erzeugt Unsinn oder bleibt irgendwo hängen. ATTN bricht den Decompilervorgang ab.

Mit der 5. und 6.CASE-Marke werden die Verzweigungsadressen der Programm-Strukturelemente bestimmt. Ist z.B. der Adressenoffset nach dem für IF, WHILE und UNTIL gültigen Verzweigungscode negativ, so muß ein Until vorliegen; ein positiver Offset signalisiert ein IF oder WHILE. IF/WHILE löst letztere Zweideutigkeit auf, indem es den Wert an der berechneten Sprungadresse und die dieser Adresse vorangehende CFA überprüft. Ist ersterer Wert negativ (→ Offset), und bedeutet die CFA ein unbedingter Sprungbefehl, so liegt ein REPEAT vor und die aufzulösende Verzweigung entspricht einem WHILE, ansonsten einem IF. In ähnlicher Weise wird nach ELSE und REPEAT aufgelöst.

'ENDA

bestimmt die Länge in Nibbles eines mit CREATE erzeugten Arrays. Ein solches Array schließt nicht mit einem Semikolon

sondern mit der Linkadresse des nachfolgenden Wörterbuch-Eintrages ab. Diese Linkadresse wird durch systematisches Durchlaufen der Wörterbuch-Kette, beginnend mit LATEST, in einer BEGIN...WHILE...REPEAT-Schleife bestimmt. Sobald die jeweilige Linkadresse auf dem Stack niedriger als die CFA des Arrays ist, wird die Schleife verlassen. Der auf dem Stack liegende Wert ist dann die gesuchte Linkadresse. Ein Sonderfall besteht, wenn das Array ohne einen nachfolgenden Vokabulareintrag in das Wörterbuch hineincompiliert worden ist, d.h. wenn das Array als letzter Eintrag im Vokabular steht. Dann zeigt offensichtlichcs HERE auf das Array-Ende und ein Suchvorgang erübrigt sich.

Hinweis: Mit 'ENDA läßt sich die Länge eines beliebigen compilierten Wortes inkl. Vorspann bestimmen: ' Wort >NAME 'ENDA - 5+

VOCA

decompiliert ein Vokabularwort mit Angabe seines letzten Worteintrages (Entsprechend LATEST bei aktiviertem Vokabular) sowie des Namens seines Eltern-Vokabulars in der Vokabular-Baumstruktur.

.NAME, NFA., NFA\$, >NAME

.NAME gibt den Wortnamen bei gegebener CFA aus, wobei >NAME (CFA — NFA) ein häufig anzutreffendes, aber nicht standardisiertes Wort ist. NFA\$ und NFA. ergeben sich aus einer für vorliegende Anwendung optimaler Code-Faktorisierung.

**Anhang1:
Adressenkonversion Forth 2A → Forth 1A**

Vorliegender Decompiler ist für Forth 2A geschrieben. Austausch der in nachstehender Tabelle aufgeführten Forth 2A-Adressen mit den äquivalenten für Forth 1A decompiliert Codesequenzen des FORTH/ASSEMBLER ROM.

Definition	FORTH2A	FORTH1A
DFLT	E0064	E004A
<COD	E68BE	E601C
COMP	E6873	E5FD1
BRNCH	E663B	E5D99
SEMI	E797E	E71E8
LITERAL	E2083	E1C54
FLITERAL	E2AB3	E22ED
+LOOP	E48B5	E3F81
LOOP	E4925	E3FF1
UNCOND.		
BRANCH	E6628	E5D86
"	E069B	E0640
."	E0FAF	E0EFA
ABORT"	E612D	E580E
abrt"	E4586	E3C86
SYS.ABORT	E4756	E317D
DO	E2096	E1C67
J	E01C3	E0168

**Anhang2:
Einzelschrittsteuerung bei ausschließlicher Verwendung der LCD-Anzeige**

Am besten wird die Tastaturabfrage STEP in UN:A zwischen WNAME und CR eingefügt. Drücken von ENDLINe schaltet das Decompilat jeweils zum nächsten Eintrag. Resultate die länger als die LCD-Anzeige sind, lassen sich mit ← bzw. → bequem nach links oder rechts durch die Anzeige schieben.

: STEP PAD EXPECT96 ;

```

DECOMP          TEXT      7186 07/02/88 17:14
18:12:15 02.07.88

( DECOMP: HP71-FORTH 2A DECOMPILER)
( E.GOSTELI Vers. 1.6 26.06.88)

BASE @ HEX
: >NAME ( CFA --NFA)      2- -1 TRAVERSE ;

: NFA$ ( NFA --NFA+2 cnt) COUNT 1F AND ;

: NFA. ( NFA+2 cnt --) ( gib Wortnamen-String aus)
  1- 2DUP TYPE 2* + C@ 7F AND EMIT ;

: .NAME ( NFA --) ( Textausgabe Wortname) NFA$ NFA. ;

( Erzeuge Feld mit Laufzeit-CFA's der Definitionswörter)
HERE 35 NALLOT
: DEFW ( n --adr)      5 * LITERAL + [ 0 ] ; DROP

0 CONSTANT CONST      0. FCONSTANT FCONST
VARIABLE VAR          1 STRING STR
                      1 1 STRING-ARRAY STRARY
    
```

```

VOCABULARY VOC
' DEFW @ 0 DEFW !
' CONST @ 1 DEFW ! ' FCONST @ 2 DEFW !
' VAR @ 3 DEFW ! ' STR @ 4 DEFW !
' STRARY @ 5 DEFW !
' VOC @ 6 DEFW !
FORGET CONST

( Konstanten)
E0064 CONSTANT DFLT ( Vorgabeadresse fuer ' bei keiner Eingabe)
E68BE CONSTANT <;COD ( Endmarke Compilezeitcode von ( CREATE..DOES>-Konstr)
E6873 CONSTANT COMP ( CFA von COMPILE)
E663B CONSTANT BRNCH ( CFA def unbedingten Sprunganweisung)
E797E CONSTANT SEMI ( CFA von Semikolon)

VARIABLE BR.ADDR ( laufende Verzweigadresse)

: ADDR. ( adr --) ( Adressausgabe in HEX) H. " : " ;

: HEADER? ( CFA --NFA+2 cnt ?) ( f:kein Vorspann, t:Vorspann)
DUP >NAME DUP NFA$ ( CFA NFA NFA+2 cnt)
2SWAP - 2/ 1- OVER = ;

: .HEAD ( CFA p --CFA) ( gib Wortvorspann aus falls vorhanden)
OVER HEADER? ( p:Vorrangigkeitsbit)
IF OVER 2- DUP 5- DUP
ADDR. ." Link " @ DUP ADDR. .NAME CR
ADDR. NFA.
0) IF ." /Immediate" THEN
ELSE 2DROP DROP THEN ;

: 'ENDA ( adr --)bodyend adr)
>R LATEST R@ U<
IF HERE
ELSE LATEST
BEGIN 5- DUP @ DUP R@ SWAP U< WHILE SWAP DROP REPEAT DROP
THEN R> ;

: IE. ( adr --) ( gib Inhalt der Zelle an adr aus)
@ DUP . ." / " H. ;

: VALUES ( 'end adr --) ( gib Werte zwischen adr und 'end aus)
2DUP - DUP ." VAR/ARRAY: " . ." nibs" CR
?DUP
IF DUP 5 MOD 0= IF DROP DO I ADDR. I IE. CR 5 +LOOP ELSE
DUP 10 MOD 0= IF DROP DO I ADDR. I RCL F. CR 10 +LOOP ELSE
2/ TYPE DROP THEN THEN
ELSE 2DROP THEN ;

: VOCA ( PFA-vocab.word --) ( gib letzter Vokab.Wortund Eltern-Vokab.aus)
2+ 2+ DUP @ DUP ." VOCABULARY" CR A SPACES
." last word: " ADDR. .NAME CR A SPACES 5+ @ DUP
." Parent Voc: " ADDR. 14 - ( CFA) >NAME .NAME ;

VARIABLE BODY? ( decompiliere Codekoerper?)
: WDEF ( CFA p --adr) ( gib Worttyp aus, p=Vorrangigkeitsbit)
( addr: CFA fall Doppelpkt.,sonst definierendes Wort)
( p=1: vorrangig, =0:nicht vorhanden)
OVER DUP @ ( p CFA [CFA]) SWAP >BODY ( PFA) = ( CFA p ?)
ABORT" Primitive"

.HEAD CR CR
DUP ADDR. DUP @ ( CFA CFA [CFA]) 0 BODY? !
CASE
[ 0 DEFW @ ] LITERAL OF ." : " CR -1 BODY? ! ENDOF
[ 1 DEFW @ ] LITERAL OF >BODY ." CONST: " IE. ENDOF
[ 2 DEFW @ ] LITERAL OF >BODY ." FCONST: " RCL F. ENDOF
[ 3 DEFW @ ] LITERAL OF >BODY 'ENDA VALUES ENDOF
[ 4 DEFW @ ] LITERAL OF >BODY C@ ." STRINGVAR of"
. ." CHAR max" ENDOF
[ 5 DEFW @ ] LITERAL OF >BODY DUP 2+ C@ ." STRING-ARRAY w/ "
. ." Elements of " C@
. ." CHARS max" ENDOF
[ 6 DEFW @ ] LITERAL OF >BODY VOCA ENDOF

```

```

DUP 5- @ <;COD <>      ( CFA von CREATE Compiler-Ende an [CFA]-5 ?)
IF  ." Cannot decompile"
ELSE OVER >BODY 'ENDA VALUES      ( decomp Codekoerper d.Def.wortes)
CR ." def at: " DUP ADDR.      ( CFA CFA does-addr)
'ENDA SWAP @ .NAME CR      ( finde Name des Definitionsw.)
SWAP DROP 2+ DUP -1 BODY? ! THEN ( decompiliere Laufzeitcode)
ENDCASE ;

: BRANCH ( addr+5 brnch-offset --addr+10) ( bestimme Verzweigadress)
OVER + DUP H.
BR.ADDR @ OVER U<
IF BR.ADDR ! ELSE DROP THEN 5+ ;

: IF/WHILE ( addr+5 brnch-offset --) ( bestimme if oder while)
+ 5- DUP @ 0< SWAP 5- @ BRNCH = AND
IF ." WHILE --> " ELSE ." IF --> " THEN ;

: .S+ ( adr --adr.of.string.end) ( gib string aus und aktualisiere adr)
COUNT 2DUP TYPE 22 EMIT 2* + ;

: -PRECOMP ( adr+5 --adr+5 ?) ( f:COMPILE vorgaengig Wort)
DUP 5- 5- @ COMP <> ; ( w:compiliertes Wort)

DECIMAL
: .MSG ( err# --) ( gib Fehler-Info aus) 47000 + " MSG$(FORTH1)" BASICX ;
HEX

: WNAME ( adr--[adr] next-adr) ( Wortname an adr)
DUP @ SWAP 5+ OVER ( [adr] adr+5 [adr])
CASE
E2083 OF ." (Lit):" -PRECOMP IF DUP I@. 5+ THEN ENDF
E2AB3 OF ." (Flit):" -PRECOMP IF DUP RCL F. 10 + THEN ENDF
E48B5 OF ." +LOOP" -PRECOMP IF ." --> " DUP @ BRANCH THEN ENDF
E4925 OF ." LOOP " -PRECOMP IF ." --> " DUP @ BRANCH THEN ENDF
E6628 OF -PRECOMP IF DUP @ DUP 0<
IF ." UNTIL --> " ELSE 2DUP IF/WHILE
THEN BRANCH
ELSE ." OBRANCH" THEN ENDF
BRNCH OF -PRECOMP IF DUP @ DUP 0<
IF ." REPEAT-->" ELSE ." ELSE --> "
THEN BRANCH
ELSE ." BRANCH" THEN ENDF
E069B OF 22 EMIT -PRECOMP IF 2+ .S+ THEN ENDF
E0F6F OF ." .' " -PRECOMP IF .S+ THEN ENDF
E612D OF ." ABORT' " -PRECOMP IF .S+ THEN ENDF
E4586 OF ." abrt': " DUP C@ .MSG 2+ ENDF
E4756 OF ." Sys.ABORT" 2+ ENDF
<;COD OF ." DOES>" 2+ 5+ ENDF
E2096 OF ." DO" ENDF
E01C3 OF ." J" ENDF
SEMI OF ." ;" ENDF
DUP HEADER? IF NFA. DROP ELSE 2DROP ." : " H. THEN DUP
ENDCASE ;

: UN:A ( adr --) ( decompiliere Codekoerper von adr bis zu letztem ;)
BEGIN
DUP ADDR.
WNAME CR ( [adr] naechste-adr)
SWAP SEMI = ( ;semi erreicht?)
OVER BR.ADDR @
SWAP U< AND ( Verzw.-adr nach ;semi?)
UNTIL DROP ;

: DISP " DISPLAY IS DISPLAY" BASICX ;
: PRINT ' DUP DFLT = ABORT" no argument"
" DISPLAY IS PRINTER" BASICX EXECUTE DISP ;

: UN: ( word ( --) ( decompiliere Wort an=adr)
( :HEX-addr ( --)
( HEX-addr ( --)
BL WORD FIND
DUP 0=
IF DROP BASE @ SWAP HEX NUMBER DROP SWAP BASE ! -1 THEN
WDEF 0 BR.ADDR !
BODY? @ IF >BODY UN:A THEN ;
BASE !

```

Decompiler Beispiele

```

: DEFER ( name ---) CREATE ['] NOP , ( vektorisiere name)
DOES> @ EXECUTE ;

: (IS) ( addr ---) R> DUP 5+ >R @ >BODY ! ; ( addr ist CFA d.vektori-)
( sierenden Wortes name)

UN: DEFER UN: (IS)
3312E : Link: 3308B : UN: 3312E : Link: 33133 : DEFER
33133 : DEFER 33178 : (IS)

3313F : : 33182 : :
33144 : CREATE 33187 : R>
33149 : (Lit):-125090 / E175E 3318C : DUP
33153 : , 33191 : 5+
33158 : DOES> 33196 : >R
33164 : @ 3319B : @
33169 : EXECUTE 331A0 : >BODY
3316E : ; 331A5 : !

: IS ( addr ---) STATE @
IF COMPILE (IS)
ELSE ' >BODY ! THEN ; IMMEDIATE

DEFER OUTP : PLOTTER ; ' PLOTTER IS OUTP

331AF : Link: 33178 : (IS) 331FB : Link: 331B4 : IS
331B4 : IS/Immediate 33200 : OUTP

331BA : : 3320A : VAR/ARRAY: 5 nibs
331BF : STATE 3320F : 209449/33229
331C4 : @
331C9 : IF --> 331E7 def at: 3315D : DEFER
331D3 : COMPILE 33164 : @
331D8 : (IS) 33169 : EXECUTE
331DD : ELSE --> 331F6 3316E : ;
331E7 : '
331EC : >BODY
331F1 : ! 33214 : Link: 33200 : OUTP
331F6 : ; 33169 : PLOTTER
33229 : :
3322E : ;

UN: E4310
E4310 : : E4324 : HERE
E4315 : 1 E432E : -
E431A : :E620C E4333 : ,
E431F : COMPILE E431A : ;
E4324 : BRANCH

```

Gewisse vorrangige Woerter (Immediates) compilieren mehrere Woerter in den Codekoerper. UN: decompiliert solche Woerter aufgeloeset in seine Komponenten, 'z.B OF in OVER = IF DROP, LEAVE als R> R> 2DROP ELSE.

E. Gosteli
 Chapfstraße 51
 8906 Bonstetten.

LEXNEED

157 Zeilen, 3524 Bytes, HP-71

Durch einigen Ärger und einige Enttäuschungen beim Weitergeben und Tauschen von BASIC-Programmen entnervt, entstand dieses Programm, das es ermöglicht, BASIC-Programme ohne viel Aufwand so zu dokumentieren, daß genau ersichtlich ist, welche Lexfiles darin verwendet werden.

Dies ist um so wichtiger, als heute schon eine ungeheure Anzahl von Lexfiles in Umlauf sind und viele sich daraus ihr eigenes, individuelles „Betriebssystem“ zusammengestellt haben. Schreibt man nun ein BASIC-Programm, so setzt man häufig bedenkenlos das eigene System voraus und verwendet dessen Funktionen. Beim Weitergeben eines solchen Programms an einen Freund z.B. kommt dann der Ärger: „XFN not found“ oder ähnliches ist dann das Ergebnis und das große Suchen in Tabellen beginnt, welche Lexfiles denn nun eigentlich fehlen.

LEXNEED erstellt nun von einem beliebigen BASIC-Programm eine Liste aller darin vorkommenden Lexfiles. So kann man dann leicht darauf achten, bei der Weitergabe auch alle nötigen „Zutaten“ nicht zu vergessen.

LEXNEED verwendet einige längere Textfiles. Daher ist es nötig, genügend Speicherplatz im Main-Memory bereit zu haben, etwa:

Size des BASIC-Programms * 1,9

Sonst ist noch zu beachten, daß sich das zu durchsuchende BASIC-Programm irgendwo im Speicher befinden muß, und dies möglichst im lauffähigen Zustand, d.h. alle darin vorkommenden Lexfiles sollten auch vorhanden sein. Ansonsten leidet darunter die Schönheit der Ausgabliste.

Das Programm ist mit RUN zu starten und anschließend auf die entsprechende Frage der Name des zu durchsuchenden BASIC-Programms einzugeben. Während des Programmlaufs (einige Minuten) huschen Zeichen über das Display, aus denen der Eingeweichte den Fortschritt des Ablaufs erkennen kann.

Am Schluß erscheint: „DRUCKER BEFREIT? TASTE.“ Drückt man nun eine Taste, so wird die Dokumentationsliste ausgegeben, entweder an den Drucker, falls dieser vorher mit PRINTER IS zugewiesen wurde, oder auf das Display. Hier muß nach Anzeige jeder Zeile eine Taste gedrückt werden.

Diese Liste bleibt auch nach Ende des Programms im Rechner erhalten, um sie beliebig oft auszugeben oder weiterverarbeiten zu können. Sie befindet sich im Textfile namens „XXXLIST“. Wenn sie

nicht mehr benötigt wird, kann sie gelöscht werden. Wenn nicht, so wird sie automatisch bei einem Neustart des Programms gelöscht.

LEXNEED benötigt, wie die Liste von LEXNEED, auf sich selbst angewendet, zeigt, die Lexfiles STRINGLX, DCMLX und STRUC2, alles Files, die bis heute in PRISMA veröffentlicht worden sein sollten. Falls nicht, sind sie bei mir schriftlich, bzw. in unserer Magnetkartenbibliothek auf magnetischen Medien erhältlich.

Weiters ist noch der Lexfile EDLEX vonnöten, von dem ich annehme, daß ihn so gut wie jeder HP-71B Benutzer besitzt, er findet sich unter anderem im HP-Texteditor, Forth-Assembler ROM und Translator ROM.

Für die Interessierten hier noch eine kurze Erklärung der Funktionsweise des Programms. Um das gewünschte Ziel zu erreichen (Lexfilebezüge in einem BASIC-Programm zu finden) fielen mir als BASIC-Programmierer eigentlich nur zwei mögliche Wege ein:

- 1.) Durchsuchen des Programms mit PEEK-Funktionen und überprüfen des tokenisierten Codes auf Lexfilebezüge.
- 2.) Kurzzeitiges Abschalten („disablen“) aller Lexfiles und Durchsuchen des Listings auf dann entstandene „XWORD“ und „XFN“-Nummern.

Der erste Weg hätte erfordert, einen kompletten Decompiler in BASIC zu schreiben, was bedeutet hätte, das Rad neu zu erfinden. Ein solcher Decompiler befindet sich ja bereits im Betriebssystem. Daher habe ich den zweiten Weg gewählt, der zwar etwas speicherplatzintensiver ist, aber nur beim Programmlauf. Anschließend werden die „fetten“ Textfiles ja wieder gelöscht.

Das Programm funktioniert nun so:

In einem ersten Durchgang des Blocks Zeile 230-330 werden alle Lexfileeinträge im sogenannten bLEX-Buffer mit der ID „BFC“ durch gezieltes Poken so verändert, daß die dazugehörigen Lexfile-Tabellen deaktiviert werden. Genauer gesagt, die Anfangs- und Endnummer der Tokens einer Tabelle werden auf 0 gesetzt. Eine Ausnahme dabei müssen die Lexfiles mit der ID# 01 und 00 sein, da es sich hier um Funktionen des Betriebssystems handelt. Der Rechner bliebe sonst sofort stehen, da die Funktionen des laufenden Programms deaktiviert würden. Aus diesem Grund ist dieser Block auch in „Minimal –“ BASIC geschrieben, ohne Lex – Funktionen.

Anschließend wird mit Hilfe des TRANSFORM-Befehls das Listing des BASIC-Programms in Text umgewandelt, wobei sich natürlich die abgeschalteten Lexfiles in Form von „XWORD“ und „XFN“-Meldungen im Listing bemerkbar machen. Nachdem der Textfile erzeugt ist, steht dem Wiedereinschalten aller Lexfiles nichts im Wege, schließlich wollen wir ja komfortabel weiterarbeiten.

Dies geschieht in Block Zeile 390-410, wo der Rechner kurz aus- und durch einen Timer wieder eingeschaltet wird. Dadurch wird ein neues Konfigurieren des HP-71B erzwungen, in dem der bLEX Buffer neu angelegt wird und alle Lexfiles wieder funktionsfähig werden.

Nun wird mit Hilfe von Textverarbeitungsfunktionen das Programmlisting nach „XWORD“ und „XFN“ Vorkommen durchsucht, die angegebenen ID#- und Tokennummern extrahiert und diese in einem Zielfile („XXXXTOKN“) sortiert gespeichert, sowie noch im Falle von „XWORD“ in einem Hilfsfile („XXXXEXPT“). Der Sinn dieser Aktion ist folgender:

Es gibt in BASIC-Programmen Fälle, wo innerhalb eines Statements auch eine Funktion vorkommt. Z.B.:

```
9000 OUTPUT :PRINTER ;REV$
(„ABCDE“)
```

Dabei ist OUTPUT ein Statement aus dem HPILROM-Lexfile und REV\$ eine Funktion, z.B. aus DCMLX. Wird nun sowohl HPILROM als auch DCMLX abgeschaltet, so zeigt das Listing lediglich:

```
9000 XWORD255019
```

Damit hat es aber die eingebettete XFN REV\$ versteckt. Aus diesem Grund wird der Block 230-330 ein zweites Mal durchlaufen, wobei aber diesmal alle Lexfiletabellen, die beim ersten Mal ein „XWORD“ lieferten, nicht abgeschaltet werden. Zeile 9000 würde dann im Listing nach dem 2. Durchlauf so aussehen:

```
9000 OUTPUT :PRINTER ;XFN144003
(„ABCDE“)
```

Nun ist es auch möglich, die eingebettete XFN auszuwerten.

Anschließend wird für jeden Token versucht, den dazugehörigen Lexfilenamen zu finden. Dies geschieht durch erneutes Durchsuchen des BFC-Buffers und Auswerten der dazugehörigen Namen. Diese Namen werden im Ausgabefile (XXXXLIST) abgespeichert mit ID-Nummer (hex) und Filegröße (dez). Dazu werden Zeile für Zeile die verwendeten Tokennummern angegeben, sowohl als Hex-

Zahl, als auch im Dezimalformat. Außerdem das entsprechende Schlüsselwort und die Parameterliste (dazu Erklärung im Artikel LEXCHECK2, PRISMA 3/88). Nach jedem Lexfile folgt eine Leerzeile.

Werden Tokennummern in der Liste nicht gefunden (z.B. Lexfile war nicht vorhanden, oder „private“), so wird der fehlende Name durch ???????? ersetzt.

Nun noch der Vollständigkeit halber eine Beschreibung der 4 Textfiles, mit denen LEXNEED arbeitet:

- 1.) XXXXTOKEN: Liste aller im Listing gefundener Token in Dezimalform. Filegröße: ca. 10 Bytes/Token. Wird am Programmende gelöscht.
- 2.) XXXXEXPT: File, in dem alle XWORD-produzierenden Token gespeichert werden. Diese werden im 2. Durchgang nicht mehr deaktiviert. Filegröße: kleiner als XXXXTOKEN. File wird am Programmende gelöscht.
- 3.) XXXXTEXT: File, der das transformierte BASIC-Listing enthält. Wird zwei mal erzeugt und gelöscht. Filegröße: ca. 1,5 mal Size des BASIC-Files. Ist am Programmende gelöscht.
- 4.) XXXXLIST: Ist der Ausgabefile, in den die Überschriften und Token ausgegeben werden. Filegröße: ca 42 Bytes/Zeile (z.B. etwa 1100 Bytes bei 5 verwendeten Lexfiles und 19 verwendeten Token). Dieser File bleibt am Ende des Programms erhalten.

Unnötig zu sagen, man sollte darauf achten, beim Betreiben von LEXNEED keine Filenamen im Rechner zu haben, die mit den 4 genannten identisch sind!

Hier noch die Bedeutung der Codes im Display während des Programmablaufes:

Im Block Zeile 230-330 werden jeweils die Codes aus Token-Range und ID#-Nummern (Hex) der bearbeiteten Lexfiletabellen aus dem BFC-Buffer angezeigt. Das

Ausschalten des Rechners in Zeile 390 ist natürlich auch deutlich zu erkennen. Nach dem Wiedereinschalten erscheint ein Doppelpunkt und dann die gefundenen Tokennummern (dez), wie sie in XXXXTOKEN eingeordnet werden. Dann folgt die Anzeige des zweiten Durchganges von 230-330. Kurz vor Programmende erscheinen dann noch einmal die Buffer-Codes und CAT-Einträge der Lexfiles, während nach den Filenamen gesucht wird.

Ausgabefile LEXNEED auf LEXNEED angewendet:

LEXNEED benötigt folgende Lexfiles:

STRINGLX	ID=52	SIZE=837	0C	REV\$	XFN	82012	(\$)
			10	SPAN	XFN	82016	(\$,\$,#)
DCMLX	ID=90	SIZE=121	02	MEM\$	XFN	144002	(\$,#)
STRUC2	ID=E1	SIZE=1541	42	END	Stmt	225066	
			43	WHILE	Stmt	225067	
			44	REPEAT	Stmt	225068	
			45	UNTIL	Stmt	225069	
			46	LEAVE	Stmt	225070	
			60	LOOP	Stmt	225096	
			63	IF	Stmt	225099	
			64	ELSE	Stmt	225100	
EDLEX	ID=F0	SIZE=2557	03	FILESZR	XFN	240003	(\$)
			04	INSERT	Stmt	240004	
			06	SEARCH	XFN	240006	(\$,#,#,#,#)

LISTING LEXNEED. HP-71B. 3524 Bytes

```

- ***LEXNEED***
  untersucht ein BASIC-Programm auf verwendete Lexfiles

  Setup:

50 DATA XXXXTOKN, XXXXEXPT, XXXXTEXT, XXXXLIST

60 DIM L$(127), F$(4) [8] @ DELAY 0
70 N9$='0123456789' @ M9=9999
80 SFLAG -1
90 FOR I=1 TO 4 @ READ F$(I) @ PURGE F$(I)
100 NEXT I @ CFLAG -1
110 CREATE TEXT F$(2)
120 ASSIGN #2 TO F$(2)
130 PRINT #2; '1001'
140 CREATE TEXT F$(1)
150 ASSIGN #1 TO F$(1)
160 CFLAG 1 @ Z=0
170 INPUT 'Basic-File ? ':N$
180 CAT N$
190 REPEAT @ Z=Z+1 @ CALL ADBUF(A$)
200 A$=DTH$(HTD(A$)+7) @ F=FILESZR(F$(2))
-
  Buffer durchsuchen und Lexfiles abschalten:
230 FOR J=1 TO M9 @ E$=PEEK$(A$,6)
240 FOR I=1 TO 6 @ E$[0,0]=E$[I*2,I*2] @ NEXT I
250 E$=E$[1,6] @ DISP E$
260 RESTORE #2
270 FOR I=1 TO F @ READ #2:X
280 IF FNM(X,E$) THEN I=998
    
```

```

290     NEXT I
300     IF I#999 THEN POKE DTH$(HTD(A$)+2),.0000'
310     A$=DTH$(HTD(A$)+11)
320     IF PEEK$(A$,2)='00' THEN J=INF
330     NEXT J
-
Listing mit XWORDS in Text transformieren:
360     TRANSFORM N$ INTO TEXT F$(3)
-
Rechner aus- und einschalten: neu konfigurieren (Lex-Buffer restored)
390     ON TIMER #1,1 GOTO 'ON' @ OFF

400     'ON': OFF TIMER #1 @ DISP ':'
410     ASSIGN #3 TO F$(3)
-
XWORD und XFN suchen und in Listen eintragen, sortieren:
440     S$='XWORD' @ GOSUB 'LOC'
450     S$='XFN' @ GOSUB 'LOC'
460     PURGE F$(3)
470     UNTIL FLAG(1.0)=0 OR Z=2
-
Lexfile - Namen suchen und Ausgabefile bereiten
500     CREATE TEXT F$(4) @ ASSIGN #4 TO F$(4)
510     PRINT #4;N$&' benoetigt folgende Lexfiles:' @ PRINT #4;'
520     E1$='000000' @ DELAY 0,8 @ yf='- '
530     FOR I=0 TO FILESZR(F$(1))-1
540         READ #1,I;T @ I$=DTH$(T DIV 1000)[4,5]
550         GOSUB 'SFN'
560         IF X$=Y$ AND I$=J$ THEN
570             L$=''
580         ELSE
590             PRINT #4;' ' @ L$=X$ @ L$[10]='ID='&I$
600             DISP CHR$(27)&'>';
610             IF FILESZR(X$)=-63 THEN CAT X$ @ Z=VAL(DISP$(18,22)) @
L$[16]='SIZE='&STR$(Z)
620         END IF
-
ID und Schlüsselwort holen
650     L$[27]=DTH$(MOD(T,1000))[4,5] @ R=0
660     IF POS(X$, '?')=0 THEN
670         D=MOD(T,1000)-HTD(E$[3,4])
680         S=HTD(REV$(PEEK$(DTH$(A-13),4)))+A-14
690         A1=A+9*D @ R=HTD(PEEK$(DTH$(A1+8),1))
700         X=S+HTD(REV$(PEEK$(DTH$(A1),3)))
710         L$[30]=MEM$(DTH$(X+1), (HTD(PEEK$(DTH$(X),1))+1)/2)
720         IF R#15 THEN L$[39]='Stmt' ELSE L$[39]='XFN'
730     ELSE
740         L$[30]=X$
750     END IF
-
Tokennummer ausgeben, Parameterliste erstellen
780     T$=STR$(T)
790     L$[50-LEN(T$)]=T$
800     IF R=15 THEN
810         L$[51]='('
820         X=A1+HTD(REV$(PEEK$(DTH$(A1+3),5)))
830         FOR J=1 TO HTD(PEEK$(DTH$(X+2),1))
840             IF J>HTD(PEEK$(DTH$(X+1),1)) THEN L$=L$&'[' @ SFLAG 0 ELSE CFLAG 0
850             IF J#1 THEN L$=L$&', '
860             R1=HTD(PEEK$(DTH$(X+1-J),1))
870             IF R1>11 THEN L$=L$&'?' ELSE IF R1>7 THEN L$=L$&'#' ELSE L$=L$&'$'
880             IF MOD(R1,4) THEN L$=L$&'()'
890             IF FLAG(0,0) THEN L$=L$&'|'
900             NEXT J @ IF J=1 THEN L$=L$&'-'
910             L$=L$&')'
920         END IF
930         PRINT #4;L$
940         Y$=X$ @ J$=I$
950     NEXT I @ DISP CHR$(27)&'<'
-
Nicht mehr benoetigte Files loeschen
980     SFLAG -1
990     FOR I=1 TO 3 @ PURGE F$(I)
1000    NEXT I
-
Liste ausgeben:
1030    IF FILESZR(F$(4))=2 THEN PRINT #4;'keine'
1040    DELAY 8,.1 @ DISP 'DRUCKER BEREIT? TASTE..'
1050    PLIST F$(4) @ PRINT @ PRINT @ DELAY 0 @ DESTROY F$ @ END
-

```

Unterroutine zum XWORD und XFN Finden:

```

1080 'LOC': S=-1
1090 LOOP
1100 S=SEARCH(S$,1,S+1,M9,3)
1110 IF S=0 THEN LEAVE
1120 READ #3,S:L$ @ L$=L$&' ' @ P=0
1130 LOOP
1140 P=POS(L$,S$,P+1) @ IF P=0 THEN LEAVE
1150 P1=P+LEN(S$) @ IF SPAN(L$,N9$,P1)#P1 THEN
1160 X=VAL(L${P1,P1+SPAN(L$,N9$,P1)})
1170 IF SEARCH(STR$(X),1,0,M9,1)=0 THEN
1180 DISP X
1190 I=0 @ F=FILESZR(F$(1))
1200 WHILE I<F
1210 READ #1,I;Y @ IF Y>X THEN LEAVE
1220 I=I+1
1230 END WHILE
1240 IF I=F THEN PRINT #1;X ELSE INSERT #1,I;STR$(X)
1250 IF S$='XWORD' THEN RESTORE #2,M9 @ PRINT #2;X @ SFLAG 1
1260 END IF
1270 END IF
1280 END LOOP
1290 END LOOP @ RETURN

```

DEF zum Ueberpruefen auf Token in Range

```
1320 DEF FNM(T,B$)=T>=FNT(B$,3) AND T<=FNT(B$,1)
```

DEF zum Umwandeln von Hex in Dez Tokenformat

```
1350 DEF FNT(E$,P)=HTD(E${5,6})*1000+HTD(E${P,P+1})
```

Routine zur Namenssuche mit Buffer-Methode

```

1380 'SFN': IF FNM(T,E1$) THEN X$=N1$ @ RETURN
1390 CALL ADBUF(A$)
1400 A$=DTH$(HTD(A$)+7)
1410 LOOP @ E$=REV$(PEEK$(A$,6))
1420 DISP E$
1430 IF E${5,6}='00' THEN E1$='000000' @ X$='????????' @ LEAVE
1440 A=HTD(REV$(PEEK$(A$,11)[7])) @ ON ERROR GOTO 'PR'
1450 IF PEEK$(DTH$(A-14),1)='F' THEN M=A-62 ELSE M=A-141
1460 OFF ERROR @ X$=MEM$(DTH$(M),8) @ GOTO 'NOPR'

```

```
1470 'PR': OFF ERROR @ X$='????????'
```

```

1480 'NOPR': IF FILESZR(X$)#-58 OR POS(X$, '?') THEN N1$=X$
1490 IF FNM(T,E$) THEN X$=N1$ @ E1$=E$ @ LEAVE
1500 A$=DTH$(HTD(A$)+11)
1510 END LOOP
1520 RETURN

```

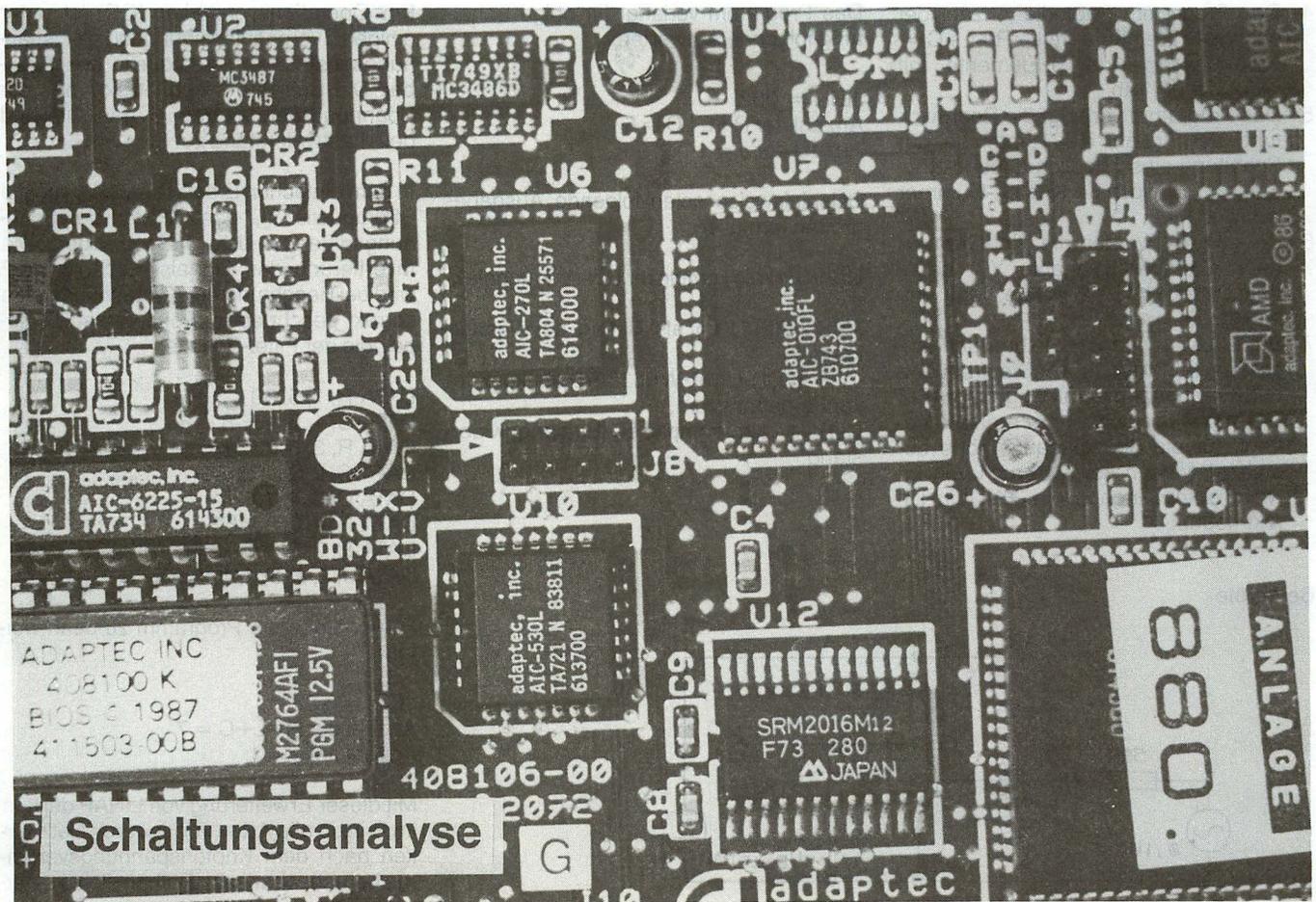
Unterprogramm zum Feststellen der Bufferadresse:

```

1550 SUB ADBUF(B$)
1560 B$=REV$(PEEK$('2F571',5))
1570 WHILE REV$(PEEK$(B$,7))[4,6]# 'BFC'
1580 B$=DTH$(HTD(B$)+7+HTD(REV$(PEEK$(B$,7))[1,3]))
1590 END WHILE
1600 END SUB

```

Michael Fiedler
Friedrichstraße 17
6070 Langen



HP-71, MATHE, IL, PRINTER

LINAP3 1008 Bytes
LINAP4 1267 Bytes

Schaltungsanalyse mit dem HP-71

2 Programme:
 LINAP3, 1008 Bytes, Mathe-Modul
 LINAP4, 1267 Bytes, Mathe-Modul, Drucker RX-80

Analyse linearer aktiver Netzwerke.

Die vorliegenden Programme sind Erweiterungen der in PRISMA 1/88 vorgestellten Netzwerkanalyse-Programme LINAP1 und LINAP2, die jetzt auch Frequenzgänge von Schaltungen mit aktiven Elementen, z.B. Transistoren oder Operationsverstärker berechnen können.

Die Liste der bisherigen Schaltelemente (R,L,C) wird um ein aktives Element erweitert, und zwar um eine spannungsgesteuerte Stromquelle J. Mit den neuen Programmen ist es prinzipiell möglich, alle Arten von Schaltungen (aktiv, passiv) in ihren komplexen Frequenzgängen (Eingangswiderstand, Übertragungsfaktor usw.) zu berechnen. Für die Topologie der passiven Elemente (R,L,C) gilt dieselbe Anordnung wie in PRISMA 1/88 beschrieben.

Spannungsgesteuerte Stromquellen.

Diese gehorchen der Beziehung

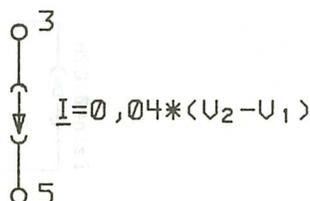
$$J = S \cdot U_{St}$$

Hierbei ist J der gesteuerte Wechselstrom und U_{St} die Steuerspannung. (J , U_{St} sind komplexe Größen). Die Steilheit S in [A/V] ist als reell angenommen. Die Steuerspannung U_{St} ist als Differenz $V_j - V_k$ zweier beliebiger Knotenpotentiale der Knoten j,k des Netzwerkes darstellbar.

Analog wie bei den passiven Analyse-Programmen bauen wir die Schaltungselemente J in Data-Zeilen in das Programm ein. Dabei wird folgendes Schema für die Reihenfolge der Knoten und die Steilheit festgelegt:

DATA J, #JL, #JE, #V+, #V-, S

Im Beispiel:
 DATA J, 3, 5, 2, 1, 0.04 bedeutet: der gesteuerte Strom J fließt von dem Knoten 3 (JL= J leaves) zum Knoten 5 (JE= J enters) und ist proportional der Spannungsdifferenz ($V_2 - V_1$).



Der Wert der Steilheit S ist 0.04 [A/V]. Bei Stromquellen ist – im Gegensatz zu passiven Elementen R,L,C – auf die Richtung des Stromes sowie auf die Richtung der steuernden Spannung zu achten. Wir werden dies an Beispielen veranschaulichen.

Da die gesteuerten Stromquellen unsymmetrisch in die Leitwertmatrix G (siehe hierzu Prisma 1/88) eingebaut werden, und der bewährte „schnelle“ Algorithmus des passiven Teils beibehalten werden soll, bedeutet dieses für die Reihenfolge der Data-Zeilen, das die J-Elemente am Schluß der Data-Liste stehen müssen. Ferner müssen wir noch dem Programm mitteilen, wieviel passive Elemente (R,L,C) und wieviel aktive Elemente J die Schaltung besitzt. Im übrigen gelten die Hinweise zur Schaltungseingabe der passiven Netzwerke aus PRISMA 1/88.

Benutzer-Hinweise:

1. Programm eingeben. LINAP4 erfordert zusätzlich das Printer-Plot-Programm PLOT2 aus PRISMA 1/88.
2. Data-Zeilen der elektrischen Elemente ab Zeile 10 in das Programm einbauen, wobei die J-Elemente nach den R,L,C-Elementen stehen müssen.
3. Ausgangsgrößen im Programm formulieren. Die zu plottende Funktion in LINAP4 wird der Variablen Y0 zugeordnet.

- Das Programm fragt die Anzahl der passiven Elemente mit „R+L+C=“ bzw. diejenige der J-Elemente mit „Anz. der J=“ ab.
- Das Programm verlangt die Beantwortung der Anzahl der Knoten in der Schaltung einschließlich Ground-Knoten, der die höchste Knoten-Nummer besitzen muß.
- Bei LINAP4 werden die Plot-Grenzen der Y-Richtung (Variable Y0) abgefragt, wobei Y1, Y2 = Min, Max entspricht.
- Bei LINAP4 wird der Reihe nach einzeln nach Startfrequenz, Stopfrequenz und Frequenz-Intervall abgefragt. Bei LINAP3 wird nach einer Frequenz gefragt, für die die Ausgangsgröße berechnet werden soll. Abbruch von LINAP3 durch Eingabe von „END“ im Direktmodus.

Das Programm verlangt für obige Schaltung folgende Antworten:

JEIN = .002
Anzahl der R+L+C = 6
Anzahl der J = 1

Beispielsweise seien folgende Ausgabe-größen formuliert:

$U(2)/U(1)$ als komplexer Übertragungsfaktor,
ABS ($U(2)/U(1)$) als Betrag der Verstärkung,
 $J1 = (U(1)-U(3))*W*.638E-6$ als komplexer Eingangsstrom,
 $U(1)/J1$ als komplexer Eingangswiderstand.

Ein Operationsverstärker (OP) ist unter einigen Vernachlässigungen als spannungsgesteuerte Stromquelle beschreibbar. Die Steilheit errechnet sich aus der Spannungsverstärkung V des OP und dem Lastwiderstand zu: $S = V/R_L$.

In dem einfachen Ersatzschaltbild wird die Stromquelle J_{OP} des OP von der Spannungsdifferenz $U(5)-U(4)$ gesteuert. Die Invertierung des Signals durch den OP wird bei Beachtung der Knotenreihenfolge korrekt beschrieben. Somit lautet die Topologie der Filter-Schalter:

DATA R, 1, 5, 50
DATA C, 1, 3, 1.E-6
DATA R, 3, 5, 7579
DATA C, 2, 3, 100 E-9
DATA C, 3, 4, 1.E-6
DATA R, 4, 2, 334E3
DATA R, 2, 5, 10
DATA J, 5, 2, 5, 4, 10000

Ferner sind dem Programm zu beantworten:

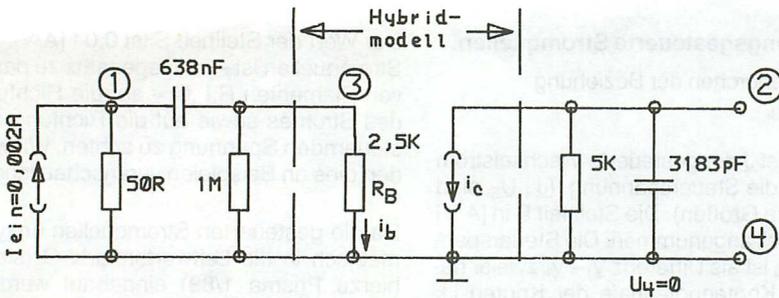
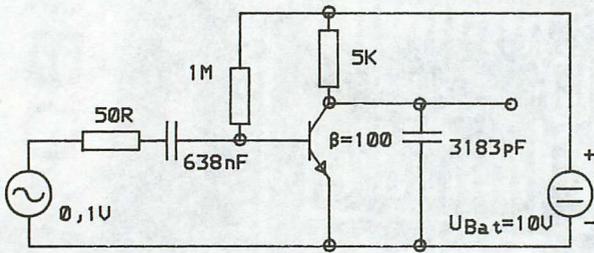
JEIN = 0.02
Anzahl der R+L+C = 7
Anzahl der J = 1

Mit dieser Erweiterung von LINAP dürften im Prinzip alle Arten von linearen Netzwerken nach dem Knotenspannungsverfahren berechenbar sein. Enthält die zu untersuchende Schaltung Spannungsquellen, so muß der Anwender diese zuvor in äquivalente Stromquellen von Hand umformen. Bei idealen Spannungsquellen ($R_i = 0$) kann man einen Serienwiderstand am Eingang der Schaltung zum „Innenwiderstand“ der Wechselspannungsquelle umfunktionieren, und diesen parallel zur Einströmung legen. Selbstverständlich kann man mit diesen beiden Programmen auch passive Vierpole untersuchen, wenn man die Anzahl der J mit „0“ beantwortet.

Literatur:

H. Wupper: „Grundlagen elektronischer Schaltungen“, Hüthig-Verlag, Heidelberg 1983, ELTEX-Studientexte.

Beispiele

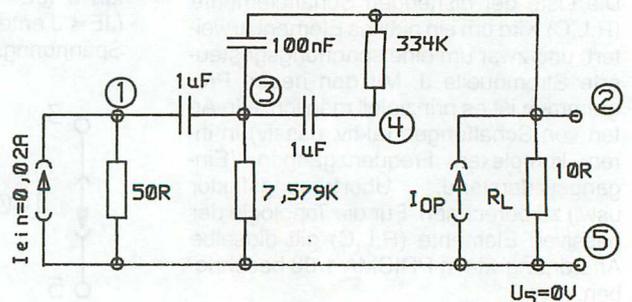
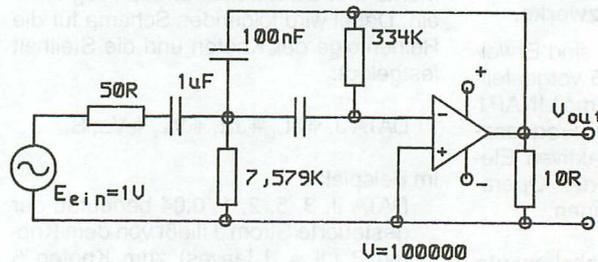


1. Transistor-NF-Verstärker

Ein Transistor mit $\beta = 100$ sei für einen Kollektorstrom von 1 mA durch sein Kleinsignal-AC-Hybrid-Modell dargestellt. Für diesen Arbeitspunkt ist sein differentieller Eingangswiderstand 2500 Ohm und seine Steilheit 40 mA/V. Die inneren Transistorkapazitäten seien vernachlässigt. Basis- und Kollektorwiderstand liegen wechselstrommäßig parallel zum Eingang bzw. Ausgang des Hybrid-Modelles. Mit den Knotenbezeichnungen des AC-Ersatzbildes erhält man folgende Topologie der Transistor-Stufe:

DATA R, 1, 4, 50
DATA R, 3, 4, 1 E6
DATA R, 3, 4, 2500
DATA C, 1, 3, .638 E-6
DATA R, 2, 4, 5000
DATA C, 2, 4, 3.183 E-9
DATA J, 2, 4, 3, 4, 0.04

2. Aktive Filter-Schaltung mit Operationsverstärker:



```

5 ! LINAP3 FUER AKTIVE VIERPOLE
6 OPTION BASE 1 @ DEGREES
7 DESTROY ALL
10 DATA R,1,4,50
11 DATA C,1,3,.638E-6
12 DATA R,3,4,2494
13 DATA R,2,4,5000
14 DATA C,2,4,3.183E-9
15 DATA J,2,4,3,4,.04
17 DATA
20 INPUT 'JEIN=';J1
25 INPUT 'ANZ.D:R+L+C=';NO
26 INPUT 'ANZ. DER J=';S1
27 NB=NO+S1
45 REAL F,W
50 DIM T$(NB)[1]
51 DIM N1(NB),N2(NB),V1(NB),N3(NB),N4(NB)
55 INPUT 'NODES INCL.GRND=';N9
61 RESTORE 10
65 FOR K=1 TO NO
70 READ T$(K),N1(K),N2(K),V1(K)
75 NEXT K
76 FOR K=NO+1 TO NB
77 READ T$(K),N1(K),N2(K),N3(K),N4(K),V1(K)
78 NEXT K
80 COMPLEX G(N9,N9),U(N9),I(N9),A

```

```

100 'START': INPUT 'FREQ=';F @ W=2*PI*F
105 MAT G=ZERO
110 FOR K=1 TO NO
115 IF T$(K)='R' THEN A=1/(V1(K),0)
120 IF T$(K)='L' THEN A=1/(0,W*V1(K))
125 IF T$(K)='C' THEN A=(0,W*V1(K))
130 G(N1(K),N1(K))=G(N1(K),N1(K))+A
135 G(N2(K),N2(K))=G(N2(K),N2(K))+A
140 G(N1(K),N2(K))=G(N1(K),N2(K))-A
145 G(N2(K),N1(K))=G(N1(K),N2(K))
150 NEXT K
151 FOR K=NO+1 TO NB
152 G(N1(K),N3(K))=G(N1(K),N3(K))+(V1(K),0)
153 G(N2(K),N3(K))=G(N2(K),N3(K))-(V1(K),0)
154 G(N2(K),N4(K))=G(N2(K),N4(K))+(V1(K),0)
155 G(N1(K),N4(K))=G(N1(K),N4(K))-(V1(K),0)
156 NEXT K
160 FOR K=1 TO N9
161 G(N9,K)=(0,0) @ G(K,N9)=(0,0)
162 NEXT K
163 MAT I=ZERO @ I(1)=(J1,0)
164 MAT U=SYS(G,I)
175 DISP '/VU/,PHI=';POLAR(U(2)/U(1))
180 PAUSE
185 GOTO 'START'
190 END

```

```

4 ! LINAP4 FUER AKTIVE VIERPOLE MIT PLOT2
5 ! FUER DRUCKER EPSON-RX 80
6 OPTION BASE 1 @ DEGREES @ PWIDTH 80
7 DESTROY ALL
10 DATA R,1,4,50
11 DATA C,1,3,.638E-6
12 DATA R,3,4,2494
13 DATA R,2,4,5000
14 DATA C,2,4,3.183E-9
15 DATA J,2,4,3,4,.04
17 DATA
20 INPUT 'JEIN=';J1
25 INPUT 'ANZ.D:R+L+C=';NO
26 INPUT 'ANZ. DER J=';S1
27 NB=NO+S1
45 REAL F,W
50 DIM T$(NB)[1]
51 DIM N1(NB),N2(NB),V1(NB),N3(NB),N4(NB)
55 INPUT 'NODES INCL.GRND=';N9
57 DISP 'GRENZEN FUER Y1,Y2=';
58 INPUT R1,R2 @ IF R2<=R1 THEN 58
59 CALL PLOTA(R1,R2) IN PLOT2
60 PRINT CHR$(27);"3";CHR$(18);
61 RESTORE 10
65 FOR K=1 TO NO
70 READ T$(K),N1(K),N2(K),V1(K)
75 NEXT K
76 FOR K=NO+1 TO NB
77 READ T$(K),N1(K),N2(K),N3(K),N4(K),V1(K)
78 NEXT K
80 COMPLEX G(N9,N9),U(N9),I(N9),A

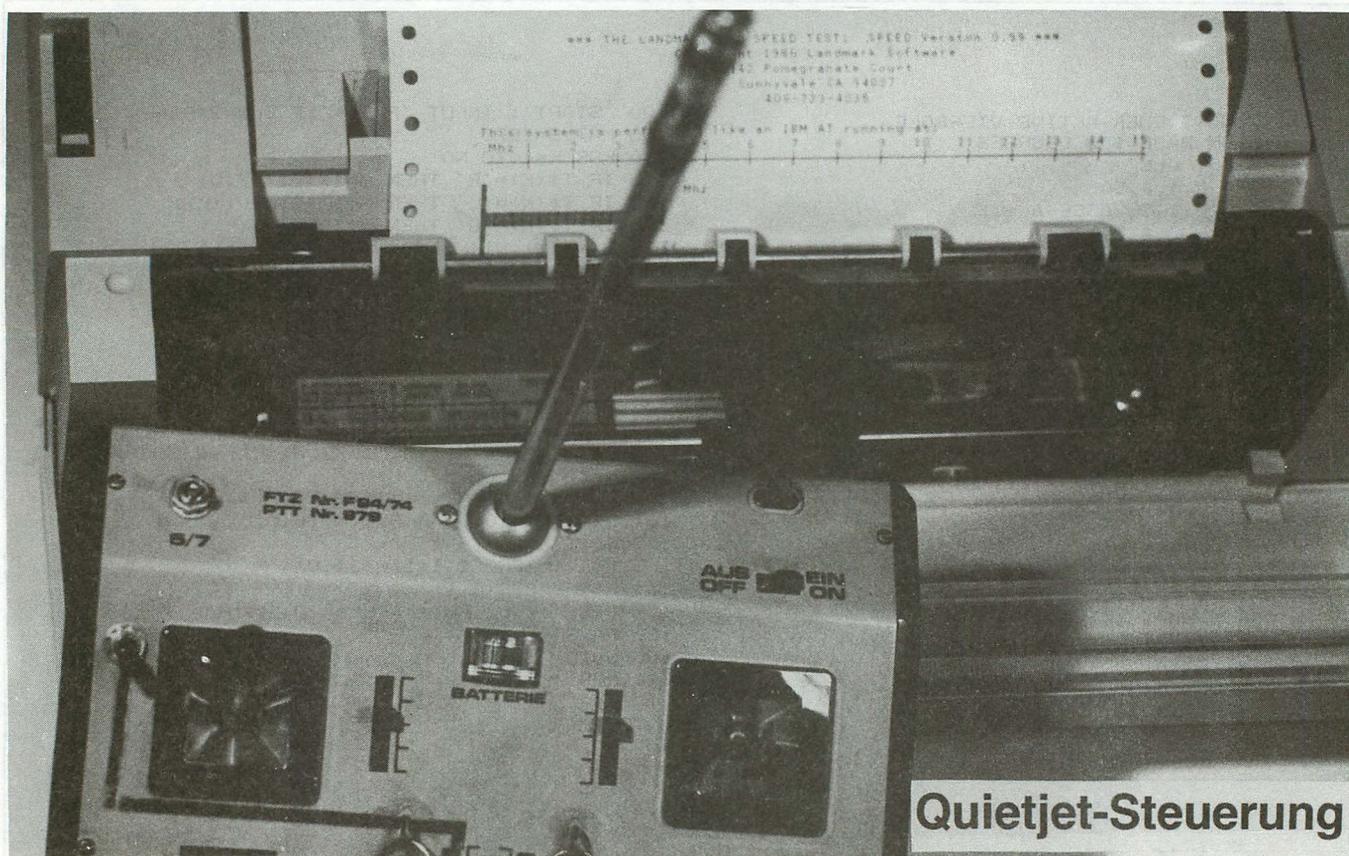
```

```

85 INPUT 'START-FREQ=';F1 @ W1=2*PI*F1
86 INPUT 'STOP-FREQ=';F2 @ W2=2*PI*F2
87 INPUT 'FREQ-STEP=';F3 @ W3=2*PI*F3
100 FOR W=W1 TO W2 STEP W3
101 DISP 'FREQ=';W/2/PI
105 MAT G=ZERO
110 FOR K=1 TO NO
115 IF T$(K)='R' THEN A=1/(V1(K),0)
120 IF T$(K)='L' THEN A=1/(0,W*V1(K))
125 IF T$(K)='C' THEN A=(0,W*V1(K))
130 G(N1(K),N1(K))=G(N1(K),N1(K))+A
135 G(N2(K),N2(K))=G(N2(K),N2(K))+A
140 G(N1(K),N2(K))=G(N1(K),N2(K))-A
145 G(N2(K),N1(K))=G(N1(K),N2(K))
150 NEXT K
151 FOR K=NO+1 TO NB
152 G(N1(K),N3(K))=G(N1(K),N3(K))+(V1(K),0)
153 G(N2(K),N3(K))=G(N2(K),N3(K))-(V1(K),0)
154 G(N2(K),N4(K))=G(N2(K),N4(K))+(V1(K),0)
155 G(N1(K),N4(K))=G(N1(K),N4(K))-(V1(K),0)
156 NEXT K
160 FOR K=1 TO N9
161 G(N9,K)=(0,0) @ G(K,N9)=(0,0)
162 NEXT K
163 MAT I=ZERO @ I(1)=(J1,0)
164 MAT U=SYS(G,I)
165 Y0=ABS(U(2)/U(1)) ! PLOT-VARIABLE
166 CALL PLOT2(Y0,R1,R2) IN PLOT2
167 NEXT W
168 PRINT CHR$(27);"2";
190 END

```

Peter Jochen
Heilbronner Str. 240
7410 Reutlingen



54 Zeilen, 2098 Bytes, HP-71, IL, Quietjet-Printer

Steuerbefehle zur Ansteuerung des Quietjet-Printers

Das Programm enthält eine Anzahl von Unterprogrammen zur Ansteuerung des Quiet-Jet-Printers. Alle Befehle müssen mit CALL 'Befehl' aufgerufen werden. Die Routine 'PINTIT' verwendet das Einheitswort 'INTRFCE' zur Identifikation des Druckers.

Allgemeines:

- PINIT: Printer initialisieren
Standartzeichensatz (A6-B1)
Zeilenlänge 86
12 Zeichen/Zoll
Schnelldruck
Perforationskip ON
HP Modus
- A4: A4-Format Seitenlänge 72
Textlänge 63
6 Zeilen/Zoll
- A4C: A4-Format Seitenlänge 96
Textlänge 84
8 Zeilen/Zoll
- D1: Display-Modus ON
- D0: Display-Modus OFF
- PNORM: PRINTER IS PRINTER, PWIDTH 96, ENDLINE CR, LF, LF

Schriftart, Zeilenlänge:

- N10(%): 10pitch normal, 10CPI, %Zeichen (0=80)
- EX10(%): 10pitch expanded 5CPI, %Zeichen (0=40)
- N12(%): 12pitch normal, 12CPI, %Zeichen (0=96)
- EX12(%): 12pitch expanded, 6CPI, %Zeichen (0=48)
- E(%): Engdruck, 21.3CPI, %Zeichen (0=48)
- EE(%): Engdruck expanded, 10.6CPI, %Zeichen (0=85)
- W1: automatischer Zeilenumbruch ON
- W0: automatischer Zeilenumbruch OFF
- KM: Korrespondenz-Druck
- SM: Schnell-Druck

Zeilenabstand, Seitenlänge:

- L(%): Zeilenabstand, %Zeichen/Zoll (0=6)
- T(%): Textlänge, %Zeichen, (0=63)
- P(%): Seitenlänge, %Zeichen, (0=72)
- S1: Perforationskip ON
- S0: Perforationskip OFF

Druckmodi:

- B1: Fettdruck ON
- B0: Fettdruck OFF
- U1: Unterstreichungen ON
- U0: Unterstreichungen OFF
- UD: unidirektionales Drucken
- BD: bidirektionales Drucken

Zeichenposition:

- FF: Formularvorschub
- BS: Rückwärtsschritt
- HS: Zeichen hochstellen
- TS: Zeichen tiefstellen
- NS: Zeichen Normalposition
- VP ("+-%"): vertikale Punktpos., %Anzahl Pkte. (1/192Zoll, 32Pkte.=1Zeile)
- HP ("+-%"): horizontale Punktpos., %Anzahl Pkte. (1/192Zoll)

Zeichensatz:

- CG: Zeichensatz deutsch
- CR: Zeichensatz Roman8
- CA: Zeichensatz ASCII
- CI8: Zeichensatz IBM8-US
- CI3: Zeichensatz IBM3-Europa
- CIS: Zeichensatz ISO IRV
- CS: Zeichensatz Strichsymbole

QUIETJET S BASIC 2098 09/07/88 20:11 0

```

10 SUB PINIT @ ENDLINE CHR$(13)&CHR$(13)&CHR$(10) @ PRINTER IS :INTRFCE @ PWIDTH 86
20 A$=CHR$(27)&"%A"&CHR$(27)&"(s12h0Q"&CHR$(27)&"(0@" @ CALL YY(A$)
30 SUB A4 @ A$=CHR$(27)&"&16d72p63F" @ CALL YY(A$)
40 SUB A4C @ A$=CHR$(27)&"&18d96p84F" @ CALL YY(A$)
50 SUB N12(L) @ A$=CHR$(27)&"(s12H" @ IF L=0 THEN L=96
60 PWIDTH L @ CALL YY(A$)
70 SUB EX12(L) @ A$=CHR$(27)&"(s6H" @ IF L=0 THEN L=48
80 PWIDTH L @ CALL YY(A$)
90 SUB N10(L) @ A$=CHR$(27)&"(s10H" @ IF L=0 THEN L=80
100 PWIDTH L @ CALL YY(A$)
110 SUB EX10(L) @ A$=CHR$(27)&"(s5H" @ IF L=0 THEN L=40
120 PWIDTH L @ CALL YY(A$)
130 SUB E(L) @ A$=CHR$(27)&"(s21,3H" @ IF L=0 THEN L=170
140 PWIDTH L @ CALL YY(A$)
150 SUB EE(L) @ A$=CHR$(27)&"(s10,6H" @ IF L=0 THEN L=85
160 PWIDTH L @ CALL YY(A$)
170 SUB B1 @ A$=CHR$(27)&"(s1B" @ CALL YY(A$)
180 SUB B0 @ A$=CHR$(27)&"(s0B" @ CALL YY(A$)
190 SUB U1 @ A$=CHR$(27)&"&d" @ CALL YY(A$)
200 SUB U0 @ A$=CHR$(27)&"&d@" @ CALL YY(A$)
210 SUB S1 @ A$=CHR$(27)&"&11L" @ CALL YY(A$)
220 SUB S0 @ A$=CHR$(27)&"&10L" @ CALL YY(A$)
230 SUB W1 @ A$=CHR$(27)&"&x0C" @ CALL YY(A$)
240 SUB W0 @ A$=CHR$(27)&"&s1C" @ CALL YY(A$)
250 SUB D1 @ A$=CHR$(27)&"Y" @ CALL YY(A$)
260 SUB D0 @ A$=CHR$(27)&"Z" @ CALL YY(A$)
270 SUB BS @ A$=CHR$(8) @ CALL YY(A$)
280 SUB UD @ A$=CHR$(27)&"&kow" @ CALL YY(A$)
290 SUB BD @ A$=CHR$(27)&"&k1W" @ CALL YY(A$)
300 SUB KM @ A$=CHR$(27)&"(s1Q" @ CALL YY(A$)
310 SUB SM @ A$=CHR$(27)&"(s0Q" @ CALL YY(A$)
315 SUB FF @ A$=CHR$(12) @ CALL YY(A$)
320 SUB L(L) @ IF L=0 THEN L=6
330 CALL XX("&1", "D", L)
340 SUB P(L) @ IF L=0 THEN L=72
350 CALL XX("&1", "P", L)
360 SUB T(L) @ IF L=0 THEN L=63
370 CALL XX("&1", "F", L)
380 SUB VP(P$) @ A$=CHR$(27)&"*p"&P$&"Y" @ CALL YY(A$)
390 SUB HP(P$) @ A$=CHR$(27)&"*p"&P$&"X" @ CALL YY(A$)
400 SUB CG @ A$=CHR$(27)&"(1G" @ CALL YY(A$)
410 SUB CR @ A$=CHR$(27)&"(8U" @ CALL YY(A$)
420 SUB CA @ A$=CHR$(27)&"(0U" @ CALL YY(A$)
430 SUB CI8 @ A$=CHR$(27)&"(10U" @ CALL YY(A$)
440 SUB CI3 @ A$=CHR$(27)&"(11U" @ CALL YY(A$)
450 SUB CIS @ A$=CHR$(27)&"(2U" @ CALL YY(A$)
460 SUB CS @ A$=CHR$(27)&"(0L" @ CALL YY(A$)
470 SUB HS @ A$=CHR$(27)&"(s1U" @ CALL YY(A$)
480 SUB TS @ A$=CHR$(27)&"(s-1U" @ CALL YY(A$)
490 SUB NS @ A$=CHR$(27)&"(s0U" @ CALL YY(A$)
500 SUB PNORM @ POKE "2F794", "1001F20" @ ENDLINE CHR$(13)&CHR$(10)&CHR$(10) @ PWIDTH 96
510 SUB XX(E1$, E0$, L) @ P$=PEEK$("2F6DC", 2) @ STD @ W$=PEEK$("2F956", 2)
520 PRINT USING "#,B,K,K,K"; 27, E1$, IP(L), E0$ @ POKE "2F6DC", P$ @ POKE "2F956", W$
530 SUB YY(A$) @ W$=PEEK$("2F956", 2) @ PRINT A$; @ POKE "2F956", W$
540 END SUB

```

Walter Kropf
Schoenowitzstraße 14
A-8700 Leoben

Lohnsteuer

61 Zeilen, 1508 Bytes, HP-71, IL, Printer

Berechnung der Lohnsteuer (Österr.)

Das Programm berechnet die Lohnsteuer aus der Bemessungsgrundlage für tägliche, wöchentliche, monatliche und jährliche Lohnzahlungszeiträume. Es kann mit oder ohne Drucker verwendet werden. Das Ausgabeformat ist auf den HP-IL-Printer abgestimmt. Der Hauptteil des Programms ist als Unterprogramm geschrieben. Es kann somit leicht von anderen Programmen aufgerufen werden.

Systemkonfiguration (.)-wahlweise:
 (Printer)
 (PNORM...Druckerinitialisierung)

Unterprogramm Steuerberechnung: CALL LST (T, G, B, S)

Übergabeparameter: T....Steuertage Woche = 6
 Monat = 26
 Jahr =312

Steuergruppe pa. kk
 p=1 bei Pensionszahlung
 p=0 bei aktiver Beschäftigung
 a=3 Alleinverdiener
 kk= Anzahl der Kinder

B....Bemessungsgrundlage
 S....Steuerbetrag

Programmablauf:

(1) Start mit RUN LOHNST
 Anzeige: Pens. <1>, Arbeitn. <0>? .0 oder 1 eingeben
 Alleinverd. <J/N>?mit J oder N beantworten
 #Kinder?Anzahl der Kinder eingeben
 Lohnzeitr.<TG WO MO JA>Lohnzahlungszeitraum eingeben
 TG für tägliche
 WO für wöchentliche
 MO für monatliche
 JA für jährliche Lohnfortzahlung

Bei angeschlossenem Drucker werden die Steuergruppe und der Lohnzahlungszeitraum ausgedruckt.

(2) Anzeige: Bemessg. Grdl.?.....Eingabe der Bemessungsgrundlage. Bei angeschlossenem Drucker wird die Bemessungsgrundlage und die Lohnsteuer ausgedruckt. Ohne Drucker wird die Lohnsteuer angezeigt. Programmfortsetzung mit ENDLINE. Sprung nach (2).

Programmbeendigung durch Eingabe von 0 als Bemessungsgrundlage oder mit GOSUB ES.

Variable: A\$ div.
 C\$ div.
 A div.
 B Bemessungsgrundlage
 G Steuergruppe
 S Steuer
 T Steuertage

im Unterprogramm:
 C Steuersatz in %
 C1 Progressionssprung
 J Jahresbemessungsgrundlage
 K1 Sonderausgabenpauschale
 K2 Werbungskostenpauschale
 K3 Alleinverdienerabsetzbetrag
 K4 allgemeiner Absetzbetrag
 K5 Arbeitnehmerabsetzbetrag
 K6 Pensionistenabsetzbetrag

Labels: EG ... Eingabe Bemessungsgrundlage
 ES Programmbeendigung
 im Unterprogramm:
 EE Programmintern
 BB Programmintern
 CC ... Programmintern
 DD ... Programmintern
 ES Ende Unterprogramm
 FNR(R). Rundung von R

Flag: 21 Printer vorhanden

```
490 J=J-C1
500 IF J<=0 THEN POP @ GOTO 'DD'
510 RETURN
520 'DD': S=S-K4-K5
530 IF G>10 THEN S=S-K6
540 IF IP(G)=3 THEN S=S-K3
550 IF S<1500 THEN S=S-(1500-S)
560 S=MAX(0,S)/312
570 RETURN
580 DEF FNR(R) ! Rundung
590 IF FP(R)<=.5 THEN FNR=IP(R) ELSE FNR=IP(R)+1
600 END DEF
610 'ES': END SUB
```

LOHNST BASIC 1508 09/07/88 21:08

```
10 DIM A$,C$,A,B,G,S,T
20 ON ERROR GOTO 50 @ A=DEVADDR("PRINTER")
30 IF A#-1 THEN SFLAG 21 ELSE CFLAG 21
40 IF FLAG(21) THEN CALL PNORM
50 IF FLAG(21) THEN PRINTER IS :PRINTER
60 OFF ERROR @ INPUT "Pens.<1>,Arbeitn.<0>?";G @ G=G+100
70 INPUT "Alleinverd.<J/N>?";AS @ IF AS="N" THEN G=G+2 ELSE G=G+3
80 INPUT "#Kinder?","0";A @ G=G+A/100
90 INPUT "Lohnzeitr.<TG WO MO JA>";AS
100 C$="TG1WO6MO26JA312" @ T=VAL(C$[POS(C$,AS)+2])
110 IF T=0 THEN 90
120 IF FLAG(21) THEN FIX 2 @ PRINT "Steuergr. ";G @ STD @ PRINT "#Tage ";T
130 'EG': INPUT "Bemessg.Grdl.?", "0";B @ IF B=0 THEN 'ES'
140 CALL LST(T,G,B,S)
150 IF FLAG(21) THEN PRINT USING "2(2A,XDP3DP3D.2D,/);";"BM",B,"ST",S
160 IF NOT FLAG(21) THEN DISP "LST";S; @ INPUT "" ;C$
170 GOTO 'EG'
180 'ES': DESTROY ALL @ CFLAG 21 @ END
190 SUB LST(T,G,B,S)
200 GOSUB 'BB'
210 IF T#1 THEN 'EE'
220 S=S*100 @ S=FNR(S)
230 S=S/100
240 GOTO 'ES'
250 'EE': S=S*T*10 @ S=FNR(S)
260 S=S/10
270 GOTO 'ES'
280 'BB':
290 K1=3276 ! Sonderausgabenpauschale
300 K2=4914 ! Werbungskostenpauschale
310 K3=3900+600*FP(G)*100 ! Alleinverdienerabsetzbetrag
320 K4=8460 ! allgemeiner Absetzbetrag
330 K5=4000 ! Arbeitnehmerabsetzbetrag
340 K6=2640 ! Pensionistenabsetzbetrag
350 J=B*(312/T)-K1 @ IF G<10 THEN J=J-K2
360 J=J/100 @ J=FNR(J)
370 J=J*100 @ S=0
380 C=21 @ C1=50000 @ GOSUB 'CC'
390 C=27 @ C1=50000 @ GOSUB 'CC'
400 C=33 @ C1=50000 @ GOSUB 'CC'
410 C=39 @ C1=50000 @ GOSUB 'CC'
420 C=45 @ C1=50000 @ GOSUB 'CC'
430 C=51 @ C1=50000 @ GOSUB 'CC'
440 C=56 @ C1=200000 @ GOSUB 'CC'
450 C=58 @ C1=500000 @ GOSUB 'CC'
460 C=60 @ C1=500000 @ GOSUB 'CC'
470 C=62 @ C1=INF @ GOSUB 'CC'
480 'CC': IF J>C1 THEN S=S+C1*C ELSE S=S+J*C
```

Walter Kropf
 Schoenowitzstraße 14
 A-8700 Leoben

Barcodes, was ist das?

Barcodes, was ist das ?

– 4. Teil –

Dies soll nun der letzte Teil dieser Serie werden; „wurde auch langsam Zeit“, werden sich manche jetzt denken, dachte ich mir übrigens auch. . . Vor allem hatte ich mir vorgenommen, diese Artikelserie noch im Jahr 1988 zu Ende zu bringen, damit ich mit guten Vorsätzen in's neue Jahr starten kann. . .

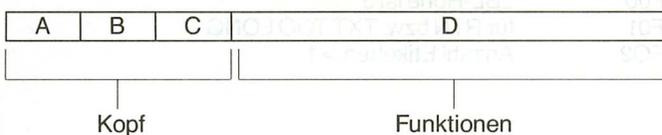
Ich möchte mich nun mit der letzten noch übrig gebliebenen Variante des HP41 Barcodes befassen, dem sogenannten „Direct Execution Bar Code“ oder zu gut deutsch dem Barcode für sofort auszuführende Funktionen.

Richtig, mit dieser Art Barcode kann man nicht nur jede einzelne auf dem HP41 zu findende Funktion ausführen, ebenso die Funktionen aller Einsteckmodule (über den Umweg mit XROM). Man kann sich also richtige Tastenfelder aus lauter Barcodes kreieren, auf denen man die am häufigsten z.B. beim Programmieren verwendeten Funktionen erreichen kann, ohne dafür ständig neue Tasten belegen zu müssen (USER Modus). Dies spart nicht nur Zeit sondern auch eventuell wertvollen Speicherplatz.

Das ist aber noch nicht alles, die eigentliche Anwendung dieses Barcode-types liegt im Prinzip in seiner Fähigkeit, mehrere Funktionen oder Eingaben mit einem einzigen Einlesevorgang zu erledigen, wie z.B. die Eingabe in's X-Register und das sofortige Starten des Programms, das diese Zahl verarbeiten soll. Man spart dadurch sehr viele Tastendrucke und kann damit ein Programm auch für absolute Laien zugänglich machen, denen man sonst erst einmal beibringen müßte, wie man XEQ ALPHA Programmname ALPHA ausführt oder wie man ein Programm von der Kassette lädt und startet, völlig unwichtig für viele Anwender, ohne sie dabei abwerten zu wollen, es soll ja nur eine Arbeitsentlastung sein, keine „Volksverdummung“, damit wir uns richtig verstehen!

Gut, nach diesen einleitenden Worten wenden wir uns gleich den technischen Details zu, damit niemandem langweilig wird.

Direkt auszuführender Funktionsbarcode



Kopf = 16 Bit = 2 Byte:

A 8 Bit Prüfsumme aus B+C+D mit jeweils dazuaddiertem Überlauf

B 4 Bit Typ-Indikator: $0100_2 = 4$

C 4 Bit unbenutzt, sollten 0 gesetzt werden

D bis zu 12 Byte für Funktionen etc.

Wir schauen uns nun am besten wieder ein Beispiel dazu an, daran kann man die Details besser erklären:

XEQ BEISP1



Beispiel 1

Hier sieht man schon, wie einfach der Aufbau dieses Barcodes ist, das dritte Byte ist die Funktion XEQ aus der 1. Reihe, 14. Spalte der Funktionstabelle (siehe PRISMA 1/88, Seite 24), hier ist das XEQ für ALPHA-Labels gemeint. Dieses Byte ist somit $1E_{16} (14_{10})$.

Was jetzt folgt ist ebenso trivial wie einsichtig, es folgt Byte für Byte das Argument dieser Funktion, nämlich die Buchstaben des gewünschten Labels, diese kann man in jeder ASCII-Tabelle oder im Druckerhandbuch nachschauen.

So, damit hätten wir schon das gesamte Prinzip dieses Barcode-types erschlagen oder abgehakt, wie man es auch sehen möchte. Gleich eine sehr wichtige Anmerkung:

Man kann immer nur eine Funktion pro Barcodezeile ausführen! Wenn mehr als eine Funktion im Barcode vorkommt, so wird nur die erste ausgeführt, alle weiteren werden ignoriert.

Dazu hier gleich ein Beispiel:

XEQ ABCDEFG, R/S



Beispiel 2

Der Befehl GTO „BEISP2“ wird ausgeführt, natürlich nur, wenn ein entsprechender Label im Speicher des HP41 steht. Schaltet man dann in den PRGM-Modus, dann sieht man den Rechner an der entsprechenden Stelle stehen, der Befehl R/S, in diesem Fall würde er dem Drücken dieser Taste entsprechen, wurde geflissentlich ignoriert, sonst wäre der Rechner ja „losgerannt“ und würde nicht auf den Label „ABCDEFG“ zeigen.

Dieses Beispiel funktionierte übrigens nur deshalb, weil die maximale Länge eines Labels ausgenutzt worden war. Wäre der Label z.B. nur 6 Buchstaben lang gewesen, so wäre die Funktion RUN (=Taste R/S) noch als 7. Buchstabe interpretiert worden! Das wird einem klar, wenn man bedenkt, daß es keine Information über die Länge des GTO-Zieles gibt, der Barcodeleser erkennt also nur an den Stopbalken oder der Anzahl Buchstaben <7, das die Funktion GTO zu Ende ist. Liest man die nachfolgende Barcodezeile im PRGM-Modus ein, so sieht man dies ganz deutlich, die Funktion wird dann ja programmiert:

GTO BEISP3, R/S



Beispiel 3

Um das Prinzip noch einmal zu verdeutlichen, gleich ein Beispiel aus der Reihe der nichtprogrammierbaren Funktionen:

CAT 1



Beispiel 4

1. Man nimmt also als erstes in diesem Barcode den Wert der gewünschten Funktion aus der Bytetabelle des HP41.
2. Das Argument der Funktion besteht ganz einfach aus den ASCII-Zeichen, die man ja in jedem Druckerhandbuch nachsehen kann, welche man für diesen Befehl über die Tastatur eingibt.
3. Zahlen als Argument werden ebenso als ASCII-Zeichen kodiert, hier gibt es nichts Spezielles.

4. Mehr als 12 Byte sind für den Abschnitt D des Barcodes nicht erlaubt, fragt bitte nicht, warum, ich weiß es auch nicht, 14 wären ja rein rechnerisch möglich, wenn man von 16 möglichen Bytes einer Barcodezeile maximaler Länge die 2 Byte Kopf abzieht.

Ich glaube, daß dieser Barcodetyp in erster Linie auch zum Experimentieren reizt, dann nichts wie ran, die Weihnachtszeit sollte genügend Zeit dafür erübrigen.

Martin Meyer (1000)

Etiketten drucken

249 Zeilen, 506 Bytes, 73 Regs., SIZE 007 HP-41CX, THINK-JET, Z-Fold Jetlabels

Das Programm dient zum Ausdrucken von umrahmten Etiketten mit einzeiligem Text in einer der vier Schriftgrößen des ThinkJet-Druckers.

Das Programm ist relativ umständlich, da viele Sicherungen eingebaut sind:

Ungerade Labelbreiten werden nicht genommen. Texte länger als Labelbreite - 2 werden mit TXT TOO LONG zurückgewiesen. Mit R/S wird dann der neue, kürzere Text eingegeben.

Ungerade Textlängen erhalten ein Leerzeichen angehängt. Im print pitch „compressed“ muß der Text folgender Beziehung genügen: Textlänge ≥ Labelbreite - 50. (Um den Text in die Mitte des Labels zu setzen, werden links und rechts Leerzeichen angehängt, dies aber nur bis zu einer Höchstzahl von 24. Da das ALPHA-Register hiermit voll ist, gibt es einen Signalton. Weiter angehängte Zeichen gehen verloren!).

Es kann für **einen** Programmdurchlauf nur eine Schriftart gewählt werden.

Es ist keine Kontrolle vorgesehen, ob der Text richtig eingegeben wurde.

"LBLHOEHE? 3:5" bedeutet: Soll der Aufkleber inklusive Rahmen drei- oder fünfzeilig gedruckt werden?

Ist der Text 24 Zeichen lang, so wird nach "FORTS=" gefragt. Wenn keine, dann R/S.

Ist die letzte Eingabe beendet, dann nach der Frage "N= ..." R/S.

Will man inmitten des Programmes ganz von vorne anfangen, dann XEQ 19. VORSICHT, LBL 19 nicht im RUN-Modus durchtasten, da sonst MEMORY LOST!

Will man statt ***** als Rand xxxxxxx wählen, dann

Zeile	Ersatz													
44	0,12													
47	2,12													
127 } 178 }	*													
R00	<table border="0"> <tr> <td rowspan="5" style="font-size: 3em; vertical-align: middle;">}</td> <td>normal</td> <td>36</td> <td rowspan="5" style="font-size: 3em; vertical-align: middle;">}</td> <td rowspan="5">Zeichen maximal</td> </tr> <tr> <td>expanded</td> <td>18</td> </tr> <tr> <td>compressed</td> <td>64</td> </tr> <tr> <td>expanded-compressed</td> <td>32</td> </tr> <tr> <td colspan="2">(LBL-Breite-2),032</td> </tr> </table>	}	normal	36	}	Zeichen maximal	expanded	18	compressed	64	expanded-compressed	32	(LBL-Breite-2),032	
}	normal		36	}			Zeichen maximal							
	expanded		18											
	compressed		64											
	expanded-compressed		32											
	(LBL-Breite-2),032													
R01	LBL-Breite													
R02	-"- -2													
R03	-"- ,042													
R04 } R05 }	Je nach Höhe: 1 oder 3													
R06	Summe ALENG des Textes, später im Programm Anzahl der Etiketten													
F00	LBL-Höhe ist 3													
F01	für RTN bzw. TXT TOO LONG													
F02	Anzahl Etiketten > 1													

```

*****
# HP-41 CX #
*****

01*LBL "LABELS" 19 "125" 37 MOD 55 3
02 FIX 0 20 64 38 X#0? 56 STO 04
03 CF 17 21 GTO 00 39 GTO 05 57*LBL 06
04 "? NECX" 22*LBL 04 40 X<>Y 58 CF 22
05 PMTK 23 "135" 41 STO 01 59 "N="
06 "rEtK" 24 32 42 ,042 60 PROMPT
07 EMROOM 25*LBL 00 43 + 61 FC? 22
08 CRFLAS 26 STO 00 44 STO 03 62 GTO 08
09 GTO IND Y 27 ACA 45 2,042 63 CLA
10*LBL 01 28*LBL 05 46 - 64 ARCL X
11 "10S" 29 RCL 00 47 STO 02 65 APPREC
12 36 30 "LBL.BREITE=" 48 E 66*LBL 07
13 GTO 00 31 PROMPT 49 "LBL.HOEHE? 35" 67 SF 01
14*LBL 02 32 X>Y? 50 PMTK 68 2
15 "11S" 33 GTO 05 51 X#Y? 69 CLA
16 18 34 ENTER↑ 52 SF 00 70 PMTA
17 GTO 00 35 ENTER↑ 53 1 71 ALENG
18*LBL 03 36 2 54 FS? 00 72 X>NN?
    
```

73 GTO 18	105 2	137 CLX	169 "ΔE="	201 GTO 19	233 "TXT TOO LONG"
74 STO 06	106 /	138 SEEKPT	170 OUTA	202 GTO 12	234 PROMPT
75 24	107 X=0?	139*LBL 12	171 RCL 04	203*LBL 16	235 FC?C 01
76 X=Y?	108 GTO 11	140 RCL 04	172 STO 05	204 CF 23	236 DELREC
77 GTO 16	109 CLA	141 STO 05	173 SF 01	205 FS?C 01	237 GTO 07
78 X<>Y	110 32	142 RCLPT	174 RCL 00	206 APPREC	238*LBL 19
79 2	111*LBL 10	143 INT	175*LBL 15	207 "FORTS="	239 CF 17
80 MOD	112 XTOA	144 SEEKPT	176 "*"	208 PMTA	240 "E&k0S"
81 X#0?	113 DSE Y	145 FC? 02	177 OUTA	209 FC? 23	241 ACA
82 "+ "	114 GTO 10	146 GETREC	178 ACLX	210 GTO 17	242 CLX
83 APPREC	115 APPCHR	147 ANUM	179 "+ΔE="	211 ALENG	243 X<>F
84 GTO 06	116 RCL Z	148 FC? 02	180 OUTA	212 ENTER↑	244 X<> c
85*LBL 08	117 INT	149 STO 06	181 DSE 05	213 ST+ 06	245 STO 64
86 CF 01	118 SEEKPT	150*LBL 13	182 GTO 15	214 CLX	246 X<> c
87 E	119 INSCHR	151 RCL 03	183 FC? 01	215 ENTER↑	247 .END.
88 SEEKPT	120*LBL 11	152 ACLX	184 RTN	216 RCL 06	
89*LBL 09	121 "*"	153 "ΔE="	185 XEQ 13	217 X>NN?	
90 RCL 02	122 APPCHR	154 SF 17	186 "E="	218 GTO 18	
91 "#"	123 RCL T	155 OUTA	187 FC? 00	219 APPCHR	
92 APPCHR	124 INT	156 FS?C 01	188 "+ "	220 24	
93 RCLPT	125 SEEKPT	157 RTN	189 OUTA	221 RCL T	
94 E-3	126 INSCHR	158 RCL 00	190 SF 02	222 X=Y?	
95 -	127 2	159 XEQ 15	191 DSE 06	223 GTO 16	
96 SEEKPT	128 +	160 CF 17	192 GTO 12	224*LBL 17	
97 E	129 SF 25	161*LBL 14	193 CF 02	225 RCLPT	
98 DELCHR	130 SEEKPT	162 GETREC	194 RCLPT	226 2 E-3	
99 RDN	131 FS? 25	163 FS? 17	195 INT	227 MOD	
100 STO Z	132 GTO 09	164 OUTA	196 E	228 " "	
101 FRC	133 RCL 01	165 FS? 17	197 +	229 X#0?	
102 E3	134 1,968	166 GTO 14	198 SF 25	230 APPCHR	
103 *	135 -	167 SF 17	199 SEEKPT	231 GTO 06	
104 -	136 STO 00	168 OUTA	200 FC?C 25	232*LBL 18	

LABELS

006:F4 0E 1B 26 6B
153:F3 0D 1B 3D
169:F3 0D 1B 3D
179:F4 7F 0D 1B 3D
186:F3 0A 1B 3D
188:F3 7F 0A 0A
240:F5 1B 26 6B 30 53

Dr. Martin Hochenegger
Heidelberger Landstraße 97
6100 Darmstadt 13

Die ROMBERG-Integration

178 Zeilen, 340 Bytes, 49 Regs., SIZE ≥050, HP-41CX

Grundlagen:

Das Verfahren basiert auf der Verknüpfung der SIMPSON-Integration mit der RICHARDSON-Extrapolation. Romberg baute mit ihrer Hilfe eine Matrix T=(N,M) (N=0,1,2,...;M=1,2,...), auch Romberg-Tabelle genannt, auf. Die*erste Spalte besteht aus Trapezintegralen T_{i,1} mit 2ⁱ Teilintervallen (i N U (0)). Alle anderen Elemente werden mit wiederholter Richardson-Extrapolation berechnet.

Es gilt also:

$$\frac{b-a}{2n} < f(a) + f(b) + 2^* \sum_{j=1}^{n-1} f(x_j) >; n=2N, M=1, x_j=a+j^* \frac{b-a}{n}$$

$$\frac{4^{M-1} * T_{N,M-1} - T_{N-1,M-1}}{4^{M-1} - 1} ; M>1$$

Das Schema hat die Form:

T _{0,1}			
T _{1,1}	T _{1,2}		
...			
T _{N-1,1}	T _{N-1,2}	... T _{N-1,M-1}	
T _{N,1}	T _{N,2}	... T _{N,M-1}	T _{N,M}

An dieser Stelle bedanke ich mich bei Herrn Prof. Dr. Becker von der FH-Niederrhein, aus dessen Vorlesung der Text bis hierher im wesentlichen stammt.

Nun zum Programm:

Bedienung:
Funktion unter LBLFF <x> programmieren
XEQ ROMB

	Stellenzahl eingeben	R/S
gewünschte Genauigkeit	"	"
untere Grenze	"	"
obere Grenze	"	"

Das Ergebnis liegt nachher in einem ASCII-File mit dem Namen *ROMB.

Erklärung des Programms:

Die äußere Schleife läuft unter LBL 06. Zuerst wird getestet, ob die Datenregisterzahl für einen weiteren Iterationsschritt ausreicht. (Das heißt: die nach oben hin benötigte Registerzahl ist grundsätzlich unbekannt; wer aber das Verfahren nach wenigen Minuten abbricht, oder die gewünschte Genauigkeit schon vorher erreicht, wird wohl kaum über die Registernummer 50 hinauskommen.) Nun wird die Anzahl der benötigten Stützstellenwerte berechnet und in R23 abgelegt. Es folgt (b-a)/n Δ (R22-R21)/(INT(R23)+1).

In der Schleife unter LBL 07 werden die erforderlichen Stützstellen berechnet und aufsummiert. Dabei werden die Stützstellen aus den vorhergehenden Iterationen wieder verwendet und nur die dazwischenliegenden Stützstellen neu berechnet.

Dann wird gemäß der ersten Formel T_{N,1} berechnet.

SERIE 40

R23 dient jetzt als Schleifenzähler zur Berechnung der Werte $T_{N,M}$ für $M = \langle 2, \dots, N+1 \rangle$. Die Schleife wird unter LBL 08 ausgeführt und benützt die zweite Formel.

Dann wird das Ergebnis angezeigt und in einem ASCII-File abgespeichert. (ASCII-File deswegen, weil man mit dem CX das Ergebnis sehr bequem über den Texteditor abfragen kann.)

Es folgt die Fehlerabschätzung, es wird einfach die Differenz der letzten und vorletzten Näherung gebildet und mit der gewünschten Genauigkeit verglichen: ist sie erreicht, wird das Endergebnis angezeigt, wenn nicht, rückt die zuletzt berechnete Zeile aus der Matrix auf, sodaß das erste Element in R32 liegt und die nächste Iteration beginnt bei LBL 06.

Beim Schreiben ist mir erst bewußt geworden, daß ich den Anfang des Programms unterschlagen habe, deswegen noch kurz folgender Nachtrag zur Beschreibung:

Es wird der File zur Speicherung des Ergebnisses initialisiert. Dann folgt die Eingabe. Die Stützstellen $F(a)$ und $F(b)$ werden

berechnet und als Summe in R26 abgelegt. Nach R32 kommt die 0-te Näherung ($N=0$).

Registerbelegung:

R20 : Iterationsschrittzähler und Anzahl der Näherungswerte in der N-ten Zeile

R21 : untere Integrationsgrenze a

R22 : ober Integrationsgrenze b

R23 : Zähler für die aufzusummierenden Funktionswerte $n=2^N-1$

R24 : N

R25 : M

R26 : $F(a) + F(b)$

R27 : $(b-a)/n$

R28 : Aufsummieren der Stützstellenwerte nach angegebener Formel

R29 : verlangte Genauigkeit

R30 : Stellenzahl (für FIX-Format)

R31 : Zwischenergebnis

R32 : Anfang der Näherungswerte $T_{N,M}$

01*LBL 06	44 RCL 26	87 RDN	130 ARCL X	173 /
02 E	45 +	88 STO IND 31	131 APPREC	174 STO 32
03 ST+ 20	46 RCL 27	89 DSE 23	132 AVIEW	175 CLX
04 ST+ 24	47 *	90 GTO 08	133 RTN	176 STO 28
05 STO 25	48 2	91 CLX	134*LBL "ROMB"	177 GTO 06
06 SIZE?	49 /	92 SEEKPT	135 "*ROMB"	178 END
07 RCL 20	50 RCL 25	93 SF 25	136 SF 25	
08 ST+ X	51 30	94 DELREC	137 PURFL	
09 31	52 +	95 CF 25	138 CF 25	
10 +	53 RCL 20	96 RCL IND 31	139 3	
11 X>Y?	54 +	97 CLA	140 CRFLAS	
12 GTO 09	55 STO 31	98 ARCL X	141 CLX	
13 2	56 RDN	99 APPREC	142 STO 23	
14 RCL 24	57 STO IND 31	100 VIEW X	143 STO 24	
15 Y↑X	58 RCL 20	101 RCL 31	144 E	
16 E	59 E	102 RCL 20	145 STO 20	
17 -	60 -	103 -	146 STO 25	
18 E-5	61 STO 23	104 X<>Y	147 STO 28	
19 ST+ X	62*LBL 08	105 RCL IND Y	148 "STELLENZAHL?"	
20 +	63 4	106 -	149 PROMPT	
21 STO 23	64 RCL 25	107 ABS	150 STO 30	
22 INT	65 Y↑X	108 RCL 29	151 "EPSILON?"	
23 E	66 RCL IND 31	109 X>Y?	152 PROMPT	
24 +	67 *	110 GTO 09	153 STO 29	
25 RCL 22	68 RCL 31	111 RCL 25	154 "FU?"	
26 RCL 21	69 RCL 20	112 31,032	155 PROMPT	
27 -	70 -	113 +	156 STO 21	
28 X<>Y	71 E	114 RCL 20	157 "FO?"	
29 /	72 +	115 E6	158 PROMPT	
30 STO 27	73 X<>Y	116 /	159 STO 22	
31*LBL 07	74 RCL IND Y	117 +	160 X<>Y	
32 RCL 23	75 -	118 REMOVE	161 -	
33 INT	76 4	119 GTO 06	162 STO 27	
34 RCL 27	77 RCL 25	120*LBL 09	163 RCL 21	
35 *	78 Y↑X	121 TONE 7	164 XEQ "F<X>"	
36 RCL 21	79 E	122 DELREC	165 STO 26	
37 +	80 -	123 "I("	166 RCL 22	
38 XEQ "F<X>"	81 /	124 FIX 0	167 XEQ "F<X>"	
39 ST+ 20	82 RCL 25	125 DSE 20	168 ST+ 26	
40 DSE 23	83 E	126 ARCL 20	169 RCL 26	
41 GTO 07	84 ST+ 31	127 "t)="	170 RCL 27	
42 RCL 28	85 +	128 RCL IND 31	171 *	
43 ST+ X	86 STO 25	129 FIX IND 30	172 2	

Klaus Huppertz
Nieselsteinstr. 30
4050 Mönchengladbach 3

Best of Prisma

Ergänzungen zum hervorragenden Best of Prisma

- Seite 3: **Letztes RTN.**
RTN und END verhalten sich auch über die Tastatur ausgeführt gleich.
- Seite 5: **256 Zeichen mit BLDSPEC.**
Für die Zeichen 0-127 kürzer xyz Enter↑
BLDSPEC, für die Zeichen 128-225 :1 ENTER↑
BLDSPEC abc BLDSPEC. War eine Zahl im X-Register, ist ENTER↑ überflüssig.
- Seite 7: **Bytesparen mit Zifferneingaben.**
3. Statt 1643,8766 E9 kürzer 16438766 E5.
477779213 E9 funktioniert nicht.
- Seite 8: **Unterprogrammrückspungadressen.**
...werden auch von END über die Tastatur und CLP gelöscht.

Ersatz.
ACOS SIN = ASIN COS, je nach Argument genauer oder ungenauer.
- Seite 12: Nicht SORT, sondern SQRT.
- Seite 38: Wenn man PRP programmiert, wird nicht immer NONEXISTENT angezeigt. Ist im Register Q ein existierendes Label in umgekehrter Reihenfolge abgespeichert, kann man auch andere Reaktionen mit dem folgenden Programm hervorkitzeln, siehe Tabelle. Es muß in diesem Fall irgendwo LBL "ABC" existieren.
01 "CBA"
02 RCL M
03 STO Q
04 PRP

Tabelle

Flag 21	Flag 55	Drucker	Wirkung von PRP im Programm
aus	aus	aus	GTO "ABC"
aus	aus	an	GTO "ABC"
aus	an	aus	GTO "ABC"
aus	an	an	NONEXISTENT
an	aus	aus	Sprung zum END Printer Off
an	aus	an	Sprung zum END
an	an	aus	Sprung zum END Printer Off
an	an	an	NONEXISTENT

- Seite 85: Programmzeile 212=F8 2A 10 2A 00 00 2A 2A F0
- Seite 102: **Kürzen von Brüchen**
Das Programm läuft nicht, siehe Seite 125.

Torsten Römer
Kollwitzstraße 25
3303 Vechede



Konjugierte Gradienten

433 Zeilen, 742 Bytes, 106 Regs., SIZE 021 min., HP-41CV

Das Programm optimiert (minimiert) Funktionen von mehreren Variablen mit einem Algorithmus nach Fletcher-Reeves, der „konjugierte Gradienten“ verwendet. Es handelt sich dabei um globale Optimierung von nichtlinearen Funktionen ohne Nebenbedingungen (unconstrained). Die benötigten Gradienten werden durch Differenzquotienten angenähert. Bei jeder Iteration wird die Schrittweite durch quadratische Interpolation approximiert.

Die Funktionen müssen unter einem globalen Label programmiert werden, wobei sich die „Design“-Variablen in den Registern R21 und aufwärts befinden. Die Anzahl der Variablen ist nur durch den Speicher begrenzt. Das Programm ist sehr anwenderfreundlich und fragt alle notwendigen Daten ab. Außerdem wird geprüft, ob die momentane Registerallokierung ausreichend ist.

Die Methode ist „quadratisch konvergent“ in dem Sinn, daß quadratische Funktionen bei exakter Schrittweitenbestimmung nach maximal n (n= Anzahl Variablen) Iterationen konvergiert. Das Programm endet, wenn die euklidische Norm des Gradienten auf 0 gerundet wird oder wenn die vorgegebene Anzahl Iterationen überschritten wird.

Ich möchte betonen, daß die Wahl der Parameter entscheidend (wie immer bei numerischen Problemen) das Konvergenzverhalten beeinflusst. Bei „richtigen“ Parametern ist die Routine ziemlich stabil und konvergiert auch bei „extremen“ Funktionen wie z.B. bei den folgenden:

$$F_1(x_1, x_2) = 100 \cdot (x_2 - x_1^2)^2 + (1 - x_1)^2$$

$$F_2(x_1, x_2, x_3) = 100 \cdot (x_3 - 10 \cdot G(x_1, x_2))^2 + (\sqrt{x_1^2 + x_2^2} - 1)_2 + x_3^2$$

$$\text{wobei } G(x_1, x_2) = \begin{cases} \frac{x_2}{\sqrt{x_1^2 + x_2^2}} \text{ wenn } x_1=0 \\ \frac{x_1}{\sqrt{x_1^2 + x_2^2}} \text{ wenn } x_2=0 \end{cases}$$

Ein weiteres mehr realistisches Beispiel F3:

Betrachte die folgende nichtlineare funktionale Beziehung:

$$Y = a - b \cdot c^x \quad \text{wobei } 0 < c < 1$$

Folgende Werte sind gegeben:

x	y
0	44.4
1	54.6
2	63.8
3	65.7
4	68.9

Durch die Minimierung der „Kleinsten-Quadrat“-Funktionen:

$$F(a, b, c) = (44.4 - a + b)^2 + (54.6 - a + b \cdot c)^2 + (63.8 - a + b \cdot c^2)^2 + (65.7 - a + b \cdot c^3)^2 + (68.9 - a + b \cdot c^4)^2$$

werden Schätzungen von a, b und c berechnet.

Wie gewöhnlich wird das Programm durch XEQ "CG" gestartet:

Display:	Action
NR. OF VAR?	Anzahl der Variablen
NR. OF REG (0) R/S	Anzahl Register, nur R/S ergibt Defaultwert. (Wenn man nämlich mehrere Register braucht, um die Funktionen zu definieren, so werden die zur Verfügung stehenden Register angezeigt und die Konstanten abgefragt).
DEFUNCTION	R/S, wenn schon programmiert, sonst GTO...
NAME?	Funktionsname
STEP (0.2) R/S	Anfangsschrittweite
H (E -6) R/S	Parameter H für den Differenzquotienten.
REFITS (6) R/S	Maximale Anzahl von „Wiederanpassungen“ des Interpolationspolynomes.
MAXIT (50) R/S	Maximale Anzahl von Iterationen.
TOL (E -2) R/S	Genauigkeit, gesteuert durch das Display.
X1=?	Startwert der ersten Variable
Xn=?	Startwert der letzten Variable
F(x1, xn)	D, um das Programm zu starten
F(x1, xn)	BEEP und optimierter Funktionswert. E ergibt die Variabelwerte.

Das Display zeigt dauernd das gerade erreichte Minimum an.

Beispiele:

Mit F1 und Startvektor (-1.2,1) konvergiert CG nach ca. 7 Minuten unter Verwendung der Defaultparameter zum Minimum (1,1) und benötigt dazu 180 Funktionsberechnungen inklusive denen, die zur Approximation des Gradienten gebraucht werden. (Zum

Vergleich: Auf dem Atari 1040ST als kompiliertes Omikronbasic-Programm mit den gleichen Parametern dauerte es 0.95 Sek. mit Coprocessor 68881 ca. 0.3 Sek.).

F2 repräsentiert ein spiralenförmiges 3-dimensionales „Tal“ und ist ein schweres Problem für jede Optimierungsmethode. Ausgehend von (-1,0,0) mit H=.0001 ansonsten Defaultwerten konvergiert das Programm zum Minimum (1,0,0) und braucht dazu ca. 23 Minuten und insgesamt 328 Funktionsberechnungen.

Bei F3 gibt man bei Nr. OF REG(0)? R/S 7 ein und belegt die Register (in Reihenfolge) mit den 5 Y-Werten, bei den beiden letzten Registern drückt man nur R/S da diese Werte von F3 aus immer neu geladen werden. Für STEP werden 1, für MAXIT 10 ansonsten die Defaultwerte genommen. Mit Startvektor (70,30,0.5) wird nach ca. 20 Minuten, 193 Funktionsberechnungen und Überschreitung von MAXIT als Minimum 3.98 ausgegeben.

F2 und F3 gibt es als extra Barcode.

Um die „Conjugate gradient“-Methode nach Fletcher-Reeves im Detail zu beschreiben, müßte ich auf gewisse mathematische Grundlagen eingehen. Ich verweise da lieber auf einschlägige Literatur über numerische Methoden. Da mir kein deutsches Fachbuch bekannt ist möchte ich Rao's „Optimization, Theory and Applications“ (Wiley Eastern Limited) empfehlen.

Das Programm ist 433 Zeilen oder 106 Register lang und benötigt keine speziellen Konfigurationen, um laufen zu können.

Kommentar: Es dürfte ziemlich einfach sein „CG“ mit Hilfe der Feldbefehle des Advantage-Roms umzuschreiben und dadurch eine bedeutende Geschwindigkeitssteigerung, speziell bei einer größeren Anzahl von Variablen zu erzielen.

01*LBL "CG"	31 ST+ 02	61 ARCL X	91 E-6
02*LBL A	32 .9	62 AVIEW	92 "H(E-6)R/S"
03 "NR. OF VAR?"	33 ST+ 03	63 PSE	93 PROMPT
04 PROMPT	34 RCL 03	64 RCL 10	94 STO 05
05 STO 00	35 RCL 00	65 RCL 11	95 6
06 CLX	36 +	66 E-3	96 "REFITS(6)R/S"
07 "NR. REG.(0)R/S"	37 INT	67 *	97 PROMPT
08 PROMPT	38 FIX 0	68 +	98 STO 04
09 STO 06	39 "SIZE "	69 STO 10	99 50
10 RCL 00	40 ARCL X	70*LBL 07	100 "MAXIT(50)R/S"
11 20	41 SF 25	71 "R"	101 PROMPT
12 +	42 RCL IND X	72 RCL 10	102 E3
13 E3	43 FC?C 25	73 ARCL X	103 /
14 /	44 PROMPT	74 "t=?"	104 STO 20
15 21	45 RCL 06	75 PROMPT	105 E-2
16 STO 02	46 X=0?	76 STO IND 10	106 "TOL(E-2)R/S"
17 STO 03	47 GTO B	77 ISG 10	107 PROMPT
18 +	48 FIX 0	78 GTO 07	108 LOG
19 STO 01	49 RCL 00	79*LBL B	109 ABS
20 RCL 06	50 21	80 "DEF. FUNCTION"	110 9
21 RCL 00	51 +	81 PROMPT	111 X>Y?
22 +	52 STO 10	82 "NAME?"	112 X<Y
23 ST+ 02	53 "R"	83 AON	113 STO 19
24 ST+ 03	54 ARCL X	84 PROMPT	114 GTO C
25 LASTX	55 "t-R"	85 ASTO 16	115*LBL E
26 ST+ 03	56 RCL 06	86 AOFF	116 SF 10
27 RCL 03	57 +	87 .2	117*LBL C
28 DSE X	58 1	88 "STEP(0.2)R/S"	118 RCL 01
29 E3	59 -	89 PROMPT	119 STO 06
30 /	60 STO 11	90 STO 09	120*LBL 08

121 FIX 0	183 STO 1	245 /	307 RCL 13	369 GTO 03	421 -
122 CLA	184 XEQ 20	246 ST* 00	308 *	370 DSE 18	422 RCL 15
123 RCL 06	185 RCL 11	247+LBL 33	309 +	371 GTO 41	423 /
124 INT	186 X<>Y	248 RCL 00	310 RCL 10	372 RCL 00	424 STO IND 13
125 20	187 X<Y?	249 STO [311 ST* X	373 ST+ 20	425 X↑2
126 -	188 GTO 12	250 XEQ 20	312 RCL 12	374 ISG 20	426 ST+ 07
127 "X"	189 10	251 STO 17	313 ST* X	375 GTO D	427 RCL 14
128 ARCL X	190 STO 17	252 RCL 00	314 -	376+LBL 60	428 STO IND 12
129 "I="	191+LBL 11	253 RCL 12	315 RCL 15	377 RCL 11	429 ISG 13
130 FS? 10	192 STO 15	254 X<Y?	316 *	378 BEEP	430 RAD
131 GTO 09	193 2	255 GTO 15	317 +	379 RTN	431 ISG 12
132 "I?"	194 ST/ 00	256 RCL 17	318 RCL 12	380+LBL 20	432 GTO 01
133 PROMPT	195 XEQ 20	257 RCL 13	319 RCL 14	381 RCL 00	433 END
134 STO IND 06	196 RCL 11	258 X>Y?	320 -	382 RCL 1	
135 GTO 06	197 X<>Y	259 GTO 16	321 RCL 11	383 -	
136+LBL 09	198 X<Y?	260 RCL 00	322 *	384 STO 1	
137 FIX IND 19	199 GTO 05	261 STO 10	323 RCL 14	385 RCL 01	
138 ARCL IND 06	200 DSE 17	262 RCL 17	324 RCL 10	386 RCL 03	
139 AVIEW	201 GTO 11	263 STO 11	325 -	387 ENTER↑	
140 PSE	202+LBL 05	264 GTO 21	326 RCL 13	388+LBL 04	
141+LBL 06	203 STO 13	265+LBL 15	327 *	389 CLX	
142 ISG 06	204 RCL 00	266 RCL 17	328 +	390 RCL IND Y	
143 GTO 08	205 STO 12	267 RCL 13	329 RCL 10	391 RCL 1	
144 FS?C 10	206 GTO 13	268 X>Y?	330 RCL 12	392 *	
145 RTN	207+LBL 12	269 GTO 17	331 -	393 ST+ IND Z	
146 FIX IND 19	208 STO 13	270 RCL 00	332 RCL 15	394 ISG Y	
147 XEQ IND 16	209 RCL 00	271 STO 14	333 *	395 ISG Z	
148 RTN	210 STO 12	272 RCL 17	334 +	396 GTO 04	
149+LBL D	211 ST+ X	273 STO 15	335 ST+ X	397 RCL 00	
150 RCL 00	212 STO 00	274 GTO 21	336 X=0?	398 STO 1	
151 1	213 XEQ 20	275+LBL 16	337 GTO 40	399 XEQ IND 16	
152 +	214 RCL 13	276 RCL 12	338 /	400 RTN	
153 STO 18	215 X<>Y	277 STO 14	339 STO 00	401+LBL 30	
154 XEQ 30	216 X>Y?	278 RCL 13	340 RND	402 CLX	
155 RCL 07	217 GTO 14	279 STO 15	341 RCL [403 STO 07	
156 STO 06	218 DSE 17	280 RCL 00	342 RND	404 XEQ IND 16	
157 RCL 02	219 GTO 12	281 STO 12	343 X=Y?	405 STO 11	
158 RCL 03	220+LBL 14	282 RCL 17	344 GTO 00	406 RCL 01	
159 ENTER↑	221 STO 15	283 STO 13	345 DSE \	407 STO 12	
160+LBL 02	222 RCL 00	284 GTO 21	346 GTO 33	408 RCL 02	
161 CLX	223 STO 14	285+LBL 17	347+LBL 00	409 STO 13	
162 RCL IND Z	224 2	286 RCL 13	348 XEQ 20	410+LBL 01	
163 CHS	225 ST/ 00	287 STO 11	349 RCL 07	411 RCL IND 12	
164 STO IND Y	226+LBL 13	288 RCL 12	350 STO 06	412 STO 14	
165 ISG Y	227 RCL 13	289 STO 10	351 XEQ 30	413 RCL 05	
166 ISG Z	228 4	290 RCL 00	352 RCL 07	414 *	
167 GTO 02	229 *	291 STO 12	353 RCL 06	415 X=0?	
168+LBL 41	230 STO Y	292 RCL 17	354 /	416 LASTX	
169 VIEW 11	231 RCL 11	293 STO 13	355 STO 10	417 ST+ IND 12	
170 RCL 07	232 3	294+LBL 21	356 RCL 02	418 STO 15	
171 SORT	233 *	295 RCL 12	357 RCL 03	419 XEQ IND 16	
172 RND	234 -	296 ST* X	358 ENTER↑	420 RCL 11	
173 X=0?	235 RCL 15	297 RCL 14	359+LBL 03		
174 GTO 60	236 -	298 ST* X	360 CLX		
175 RCL 09	237 X<>Y	299 -	361 RCL IND Y		
176 STO 00	238 LASTX	300 RCL 11	362 RCL 10		
177 RCL 04	239 RCL 11	301 *	363 *		
178 STO \	240 +	302 RCL 14	364 RCL IND Z		
179 2	241 ST+ X	303 ST* X	365 -		
180 STO 17	242 -	304 RCL 10	366 STO IND Y		
181 CLX	243 X=0?	305 ST* X	367 ISG Y		
182 STO 10	244 GTO 40	306 -	368 ISG Z		

Guido Petz
Ringansvägen 5 b
S 12633 Hågersten

Zeitanzeige auf PAC-SCREEN

84 Zeilen, 254 Bytes, 37 Regs., SIZE 000, HP-41CV, X-F PAC-SCREEN

Wochentag, Tag und laufende Uhrzeit
→ PAC-SCREEN

Mittels PRP erhält man auf dem IL-Drukker einen Ausdruck, mit dem man nichts anfangen kann. Durch die Bytes der inversen Darstellung wird der Ausdruck „chinesisch“.

Der richtige Ausdruck erfolgte durch Einkleben der Bytes der synthetischen Zeilen.

Am ThinkJet-Drucker ist die Programmausgabe wesentlich einfacher.

Durch R/S wird das Programm angehalten.

Falls es über Mitternacht läuft, bleiben der alte Wochentag und das alte Datum bestehen.

Nach Abschluß der Ausgabe ist F17 zu löschen.

```

01*LBL "UHR"
02 FIX 4
03 CF 17
004:F2 1B 36
05 ACA
06 3,032
07 ACLX
08 34,041
09 ACLX
010:F4 1B 25 03 10
11 OUTA
12 X<>Y
13 ACLX
14 X<>Y
15 ACLX
016:F4 1B 25 03 04
17 ACA
018:FA C7 D5 D4 C5 CE A0
D4 C1 C7 A1
19 OUTA
020:F4 1B 25 03 07
21 ACA
022:FC 0A 0A C8 C5 D5 D4
C5 A0 C9 D3 D4 A0
23 SF 17
24 OUTA
25 DATE
26 DOW
27 GTO IND X
28*LBL 00
029:F4 D3 CF CE CE
30 GTO 07
31*LBL 01

```

```

032:F3 CD CF CE
33 GTO 07
34*LBL 02
035:F5 C4 C9 C5 CE D3
36 GTO 07
37*LBL 03
038:F8 CD C9 D4 D4 D7 CF
C3 C8
39 SF 00
40 GTO 07
41*LBL 04
042:F7 C4 CF CE CE C5 D2
D3
43 GTO 07
44*LBL 05
045:F4 C6 D2 C5 C9
46 GTO 07
47*LBL 06
048:F4 D3 C1 CD D3
49*LBL 07
50 FC?C 00
051:F4 7F D4 C1 C7
52 OUTA
053:F6 AC A0 C4 C5 D2 A0
54 OUTA
55 DATE
56 CLA
57 ADATE
58 8
59 STO 00
60*LBL 08
61 ATOXL
62 128
63 +
64 XTOAR
65 DSE 00
66 GTO 08
67 CF 17
68 OUTA
69 ADV
70 ADV
71 14,032
72 ACLX
73 "ES IST JETZT"
74 OUTA
75*LBL 09
76 SF 17
077:F5 1B 25 0D 0C AA
78 TIME
79 ATIME
80 OUTA
081:F6 7F 20 55 68 72 AA
82 OUTA
83 GTO 09
84 END

```

Tips & Tricks

Anzeigeformate, Tastenzuweisungen
24 Zeilen, 50 Bytes, SIZE 000, Synthe-
tik, HP-41, X-Functions

Noch ein KA-Programm, aber eine Sparversion mit Haken und Ösen, denn die KA-Register müssen vor der Ausführung vollständig gepackt sein, d.h. die nächste ungerade Zuordnung muß ein Register anbrechen. Soll eine Zuordnung überschrieben werden, dann nur die zuletzt gemachte. Ansonsten kann man mit dem Programm beliebig viele Zuweisungen hintereinander machen. Liegt auf der gewünschten Taste plötzlich %, so ist etwas schiefgegangen, die gewollte Zuweisung liegt dann woanders. Aber keine Panik, Abstürze gibt es nicht. Außerdem ändert das Programm nur das niedrigste KA-Register (0C0).

Bedienung:
Präfix
ENTER↑
Postfix
ENTER↑
Tastencode
XEQ "TZ"

Der Programmlauf darf nicht unterbrochen werden, da man sonst eine ziemlich LOSTige Anzeige erhält!

Als Test solltet Ihr mal 4 ENTER↑ 32 ENTER↑ 15 XEQ "TZ" ausführen. Wenn Ihr dann im USER-Modus die LN-Taste drückt, landet Ihr immer (?) im ROM. Ich nenne die Funktion hier ROM Springer (RS). Mir ist es bei meinem älteren HP-41CX zweimal passiert, daß ich nach Betätigung des RS im Programmmodus nach weiteren wahllosen Tasten ein MEMORY LOST hatte, während PRIVATE angezeigt wurde. Diese Funktion scheint die ROM-Adresse anzuspringen, die man vorher mit STO b im RAM geladen hatte.

Tipt man ein beliebiges Zeichen ins ALPHA-Register und tastet dann RS, erscheint CAT, drücken der Korrekturtaste und erneutes RS bewirkt CAT 3. Jetzt noch etwas Komisches: die Tastenfolge STO 00 RS ALPHA oder R/S ergibt zwei Prompts hinter dem X-Register, eine Taste z.B. aus der oberen Reihe einen Absturz (Abhilfe: gleichzeitig ENTER↑ ON oder bei altem HP-41 Batterie kurz herausnehmen), wenn 10 Ziffern im X-Register waren.

Drückt man nach STO eine der oberen Tasten und dann RS, so werden die Flags verändert oder auch die umgeschalteten Tastenbelegungen. Die ursprünglichen Tastenzuweisungen lassen sich mit (GTO.. CLA) PCLPS retten.

Neuste Meldung: ein neuerer HP-41CV mit abgerundeter Anzeige liefert immer MEMORY LOST.

Dr. Martin Hochenegger
Heidelberger Landstraße 97
6100 Darmstadt 13

```
01*LBL "TZ"
02 "% "
03 PASH
04 RCL c
05 "*v="
06 ASTO c
07 ARCL 00
08 ASHF
09 R↑
10 XTOA
11 R↑
12 XTOA
13 SIGN
14 AROT
15 XTOA
16 RCL c
17 STO \
18 ASTO X
19 ARCL X
20 X(>) \
21 STO 00
22 R↑
23 STO c
24 END
```

Synthetische Textzeile:
Z05 F4 2A 0C 01 F0

Anzeigeformate

Da ich in meinen Prismaheften nichts über die Wirkung synthetischer Anzeigeformate finden konnte, habe ich eigene Experimente unternommen. Hier mein Ergebnis für die Nachsilben 0...127. Im folgenden sind diese acht Bits auch mit 0abcdefg bezeichnet. Wäre Bit 7 nicht 0, sondern 1, hätten wir die gewohnte indirekte Adressierung.

```
FIX = SF 40 CF 41
SCI = CF 40 CF 41
ENG = CF 40 SF 41
```

Die Nachkommastellen werden durch defg eingestellt. Das ist alles bekannt. Jetzt muß man sich alle Flags als eine 56 Bit Zahl vorstellen, aber mit Flag 43 als niederwertigstem und Flag 44 als höchstwertigstem Bit (siehe Abbildung). Zu dieser Zahl wird nun abc binär addiert. Das Ergebnis ist der neue Flagstatus. Das Programm FIX SCI ENG setzt alle Flags außer Flag 53 und 54, dann löscht FIX 95 (=FIX 01011111) Flag 41 und addiert danach 101 zum Flagstatus, der alte Status wird wieder hergestellt und das Ergebnis OFFA ins Alpha Register gelegt. Viel Spaß beim Nachrechnen!

Nützlich sind woll nur FIX 64...73 für FIX/ENG 0...9

Achtung, zumindest für meinen HP-41CX gilt:
F53 gesetzt, dann F53 gelöscht
F53, F54 gesetzt, dann F53, F54 gelöscht
F45, F53 gesetzt, dann F30, F45, F47, F50, F53, F54 gelöscht

```
01*LBL "FIX SCI ENG"
02 ""
03 RCL I
04 X(>) d
05 FIX 5
06 X(>) d
07 STO I
08 END
```

Synthetische Zeilen:
Z02 F7 FF FF FF FF FF FF F9
Z05 9C 5F

Flags als 56 Bit Zahl

```

          >
44 45 46...53 54 55 0 1 2 ...40 41 42 43
+                               a b c
```

Tips und Tricks

Wenn man sich nicht sicher ist, ob die Batterie noch für den Kartenleser reicht, so sollte man probierhalber eine Magnetkarte bei ausgeschaltetem Rechner durchschieben, der Schaden ist dann geringer. Liest man eine Magnetkarte ein, wenn die Batterie schon schwach ist, so bleibt die Karte meistens stecken. Ärgerlich, da dann bei eingeschaltetem Rechner nach einem Batteriewechsel oft die Speicherorganisation durcheinandergewürfelt und die Uhr zurückgesetzt ist. Ausschalten lassen sich die meisten Rechner vor dem Batteriewechsel so: ENTER↑ ON gleichzeitig drücken, jetzt ENTER↑, dann ON loslassen.

Besonders in Astronomieprogrammen wird oft nach einer Division die Funktion ATAN benutzt, um einen bestimmten Winkel zu erhalten. Es sind jedoch ziemliche Klimmzüge nötig, um jetzt noch im richtigen Quadranten zu landen.

Beispiel: -1 jeweils im X- und im Y-Register muß als Winkel -135 Grad oder noch besser -135+360=225 Grad ergeben. Die Tastenfolge / ATAN ergibt aber 45 Grad. Abhilfe: R-P CLX 360 MOD

Bugs, Kartenleser

Der folgende Fehler tritt wahrscheinlich bei allen Kartenlesern bis zur Version CARD RDR 1E auf. Gibt man eine Funktion ein, die drei Ziffern erwartet, wie z.B. SIZE, GTO. oder LIST, tippt dann genau zwei Ziffern ein und nimmt sie mit der Korrekturtaste zurück, so rotiert der Anzeigeinhalt um etwa vier Zeichen je nach Funktion. Der HP-41 läßt sich mit allen Tasten außer der Korrekturtaste wiederbeleben. Wird die Taste nur ganz kurz angetippt, dann schalten sich manche Rechner ab.

Bei Kartenlesern bis CRD RDR 1F wird bei Ausführung von VER das letzte Register der soeben geprüften Spur in das Register 3EF des HP-41 geschrieben. Das ist der bekannte Manipulator für den X-Memory in Steckplatz 2.

7CLREG jedes Kartenlesers löscht die Speicher 00...09 und 20...25, auch wenn SIZE<026 und damit diese Speicher oberhalb des RAM liegen. Peinlich wird es, wenn man einen Rechner mit 319 Registern und einen X-Memory in Steckplatz 1 oder 3 hat, denn dann liegen diese Speicher ganz (bei SIZE 000) oder teilweise (bei SIZE<025) in diesem X-Memory.

Beim Benutzen von EMDIRX beim HP-41CX kann es passieren, daß im Alpha Register nicht der erwartete Inhalt erscheint. Da Filenamen im X-Memory mit Leerzeichen auf sieben Zeichen aufgefüllt werden, stehen diese nach EMDIRX auch im Alpha Register. Die Tastenfolge:

```
"TEST"
SAVEP
n (TEST ist n-tes File)
EMDIRX
ASTO 00
GTO IND 00
liefert NONEXISTENT
```

Torsten Römer
Kollwitzer Straße 25
3303 Vechele

Zeilenlineal für den THINK-JET

20 Zeilen, 48 Bytes, 7 Regs., SIZE 000, HP-41C, IL, PRINTER

In Ergänzung zu meinem Programm „Scheibschrift-Linial“, erschienen in Prisma 5/88, das nur in horizontaler Richtung gilt, noch ein Zeilenlineal für die vertikale Richtung.

```
01*LBL "ZL"
02 CF 28
03 CF 29
04 FIX 0
05 "E&k1S-r"
06 ACA
07 0.059
08 ENTER↑
09 4
10*LBL 00
11 RCL Y
12 INT
13 CLA
14 ACAXY
15 "--"
16 ACA
17 PRBUF
18 ISG Y
19 GTO 00
20 END
```

ZL

```
-----
005:F6 1B 26 6B 31 53 0E
```

Schreibschrittlineal für IL-Drucker

```

01+LBL "ILPL2"
02 "αδeLe"
03 ASTO 00
04 "pαΓα"
05 ASTO 01
06 "0CαDe"
07 ASTO 02
08 "αδα↓"
09 ASTO 03
10 "↓++"
11 ASTO 04
12+LBL 01
13 CLRDEV
14 "LLLLLLLLLLLLLLLL"
15 "↑LLLLLLLLLLLL"
16 OUTA
17 STOP
18 4.032
19 ACLX
20 RCL 04

                ILPL2
-----
21
22 002:F6 71 12 65 4C 91 1C
23
24 004:F6 70 40 81 02 06 08
25
26 006:F6 F8 10 43 08 11 1C
27
28 008:F6 71 12 04 07 81 3E
29
30 010:F4 0F E0 00 00
31 014:FF 7C 7C 7C 7C 7C 7C
32 7C 7C 7C 7C 7C 7C 7C
33 7C
34 015:FA 7F 7C 7C 7C 7C 7C
35 7C 7C 7C 7C
36
37
38
39
40 01+LBL "ILPL1"
41 02 CLRDEV
42 03 "LLLLLLLLLLLLLLLL"
43 04 "↑LLLLLLLLLLLL"
44 05 OUTA
45 06 STOP
46 07 "123456789012345"
47 08 "↑678901234"
48 09 OUTA
49 10 END

                ILPL1
-----
50
51
52 003:FF 7C 7C 7C 7C 7C 7C
53 7C 7C 7C 7C 7C 7C 7C
54 7C
55 004:FA 7F 7C 7C 7C 7C 7C
56 7C 7C 7C 7C
57
58
59

```

```

21 ACSPEC
22 X<>Y
23 ACLX
24 RCL 01
25 ACSPEC
26 X<>Y
27 ACLX
28 RCL 04
29 ACSPEC
30 X<>Y
31 ACLX
32 RCL 02
33 ACSPEC
34 PRBUF
35 STOP
36 RCL 01
37 ACSPEC
38 3.032
39 ACLX
40 RCL 03
41 ACSPEC

```

```

42 4.032
43 ACLX
44 RCL 00
45 ACSPEC
46 X<>Y
47 ACLX
48 RCL 03
49 ACSPEC
50 X<>Y
51 ACLX
52 RCL 00
53 ACSPEC
54 PRBUF
55 CLX
56 .END.

```

	Zeilen	Bytes	Regs.	SIZE
ILPL1:	10	73	11	000
ILPL2:	56	158	23	005

LBL „ILPL1“ IL-Printer-Pitch-Lineal

IL-Drucker
EXT-i/O
(Falls Extended I/O nicht vorhanden,
Drucker vor Start aus- und wieder ein-
schalten)

Nach dem Zwischenstopp den Druckstrei-
fen ein wenig zurückziehen.

LBL „ILPL2“
Zu den oben angeführten Geräten wird
noch zusätzlich das CCD-Modul benötigt.

Auch hier nach jedem Zwischenstopp den
Druckstreifen ein wenig zurückziehen.

Für eine Wiederholung XEQ 01.

Den Ausdruck kopieren und dann auf ei-
nen Träger kleben.

ASCII-Record Berechnungen

**Recordlänge und letzte Recordposi-
tion RECLENG/RECMAX**
14/7 Zeilen, 34/22 Bytes, 5/4 Regs.,
SIZE 000, HP-41C, X-F

In Zeile 02 der Programme kann jedes be-
liebige ASCII-Zeichen Verwendung fin-
den.

1) File einstellen.

RECLENG wurde in meinem Programm
LABELS (s. dieses Heft) verwendet.

2) XEQ RECMAX ergibt im X-Register die
Nummer des letzten Records.

RECMAX benötigte ich einmal, um in ei-
nem nach oben offenen Adreßfile den letz-
ten Record zu finden.

3) Vor XEQ RECLENG den entsprechen-
den Record einstellen. Das Ergebnis er-
scheint ebenfalls im X-Register.

```

01+LBL "RECLENG"
02 "A"
03 APPCHR
04 RCLPT
05 E-3
06 -
07 SEEKPT
08 E
09 DELCHR
10 X<>Y
11 FRC
12 E3
13 *
14 END

```

```

01+LBL "RECMAX"
02 "A"
03 APPREC
04 RCLPT
05 DELREC
06 INT
07 END

```

Dr. Martin Hochenegger
Heidelberger Landstraße 97
6100 Darmstadt 13

XROM eine neue Variante

78 Zeilen, 133 Bytes, 19 Regs., SIZE 002, HP-41C

Ich konnte es nicht lassen, eine weitere Version der Problemlösung, XROMs in fremden Programmen aufzulisten, zu präsentieren.

Es ist in erster Linie schneller als die beiden zuerst veröffentlichten Programme. Abgesehen davon, daß diese Aufgabe besser auf einem PC gelöst wird (ich denke hierbei an die Redaktion), halte ich das Konzept von Walter Lutz, nur die einzelnen Module, nicht aber jede Funktion, auszugeben, für sinnvoller. Wer Namen haben möchte, kann nach Zeile 59 die Befehle 10, +, SF25, XEQ IND X anfügen, sowie die entsprechenden Labels, aber meines Erachtens erfüllt eine einmal ausgedruckte Liste, die an der Wand hängt, den Zweck besser.

Das Programm von Armin Jakob aus 4/88 ist zwar fein, aber in der Dekodierung der einzelnen Befehle zu umständlich bzw. zu gradlinig.

Bei Walter Lutz ist das Konzept zwar in Ordnung, aber die Dekodierung der Befehle nicht so sehr gelungen, da nicht vorhanden. Das Programm bringt falsche Aussagen, falls ein Adressbyte bzw. allgemein ein Folgebyte eines Befehls zufällig einen Wert zwischen 160 und 167 hat.

```

01*LBL "XROM"
02 32
03 WSIZE
04 "NAME ? "
05 PMTA
06 AVIEW
07 PPLNG
08 STO 00
09 CLX
10 STO 01
11 PHD
12*LBL 00
13 PEEKB
14 144
15 -
16 X<0?
17 GTO 09
18 16
19 /
20 GTO IND X
21*LBL 00
22*LBL 02
23*LBL 03
24 SIGN
25 GTO 08
26*LBL 04
27*LBL 05
    
```

```

28 SIGN
29 ST+ X
30 GTO 08
31*LBL 06
32 LASTX
33 X<>Y
34 FRC
35 *
36 GTO 08
37*LBL 01
38 FRC
39 ,5
40 X<=Y?
41 GTO 07
42 RCL Z
43 A-
44 PEEKB
45 R↑
46 64
47 ST/ Z
48 *
49 +
50 RCL 01
51 X<>Y
52 BS?
53 GTO 00
54 LASTX
55 Sb
56 STO 01
57 X<> L
58 "XROM "
59 ARCLI
60 AVIEW
61*LBL 00
62 DSE 00
63 GTO 09
64*LBL 07
65 RDN
66 SIGN
67*LBL 08
68 ST- 00
69 CHS
70 A+B
71 ENTER↑
72*LBL 09
73 RDN
74 A-
75 DSE 00
76 GTO 00
77 CLD
78 END
    
```

* Beispiel *

```

XEQ "XROM"
XROM
XROM
XROM 9
XROM 11
    
```

Andreas Marktscheffel
Gnessener Straße 63
7000 Stuttgart 50

Bessel und Neumannfunktion

129 Zeilen, 193 Bytes, 28 Regs., SIZE 008, HP-41C

Bei vielen Problemen der theoretischen Physik treten immer wieder sphärische Bessel- und Neumannfunktionen auf. So zum Beispiel in der Quantenmechanik als Lösung der freien radialen Schrödingergleichung /1/ oder in der Elektrodynamik bei Multipolentwicklungen /2/.

Die mathematische Definition lautet:

$$j_n(z) = z^n \cdot \left(-\frac{1}{z} \cdot \frac{d}{dz} \right)^n \left(\frac{\sin z}{z} \right) \text{ für Besselfunktionen der Ordnung } n$$

$$n_n(z) = -z^n \cdot \left(-\frac{1}{z} \cdot \frac{d}{dz} \right)^n \left(\frac{\cos z}{z} \right) \text{ für Neumannfunktionen der Ordnung } n$$

Beide Funktionen genügen der Rekursionsformel:

$$f_{n+1}(z) = \frac{2n+1}{z} \cdot f_n(z) - f_{n-1}(z) \quad n = 1, 2, 3, \dots$$

Genauer über Bessel- und Neumannfunktionen, insbesondere graphische Darstellungen, findet sich in /3/.

Der Algorithmus beruht im wesentlichen auf obiger Rekursionsformel, die für Neumannfunktionen in Vorwärts- und Besselfunktionen zusammen mit der Normierungsbedingung

$$\sum_{n=0}^{\infty} (2n+1) \cdot j_n^2(z) = 1$$

in Rückwärtsrichtung verwendet wird. Dadurch kann eine Genauigkeit von 9-10 Dezimalen erzielt werden! Nähere Ausführungen zur Idee des Algorithmus finden sich in /4/.

Programmbeschreibung:

- 1) Aufruf mit XEQ "SBNF"
- 2) "Z?": Eingabe des Arguments, R/S
- 3) "N?": Eingabe der Ordnung der Funktion, R/S

Es erscheint das Menü "BES.....NEU"

- 4a) XEQ A: Berechnung der Besselfunktion $j_n(z)$
- 4b) XEQ B: Berechnung der Neumannfunktion $n_n(z)$

Die Anzeige erfolgt im SCI-4 Format. CLX bewirkt jeweils Anzeige des entsprechenden Wertes im FIX-9 Format.

Literatur:

- /1/ *Messiah* Quantenmechanik Band 1, Seite 317 ff.
- /2/ *Jackson* Klassische Elektrodynamik, Seite 884 ff.
- /3/ *Abramowitz, Stegun* Pocketbook of Mathematical Functions, Chapter 10.1
- /4/ *Koonin* Computational Physics, Chapter 4.1

SERIE 40

01*LBL "SBNF"
 02 CF 01
 03 SF 27
 04 CF 29
 05 RAD
 06 "Z?"
 07 PROMPT
 08 STO 01
 09 "N?"
 10 PROMPT
 11 STO 00
 12 *BES NEU"
 13 PROMPT
 14*LBL E
 15 RCL 01
 16 COS
 17 LASTX
 18 /
 19 CHS
 20 STO 05
 21 LASTX
 22 /
 23 LASTX
 24 SIN
 25 LASTX
 26 /
 27 -

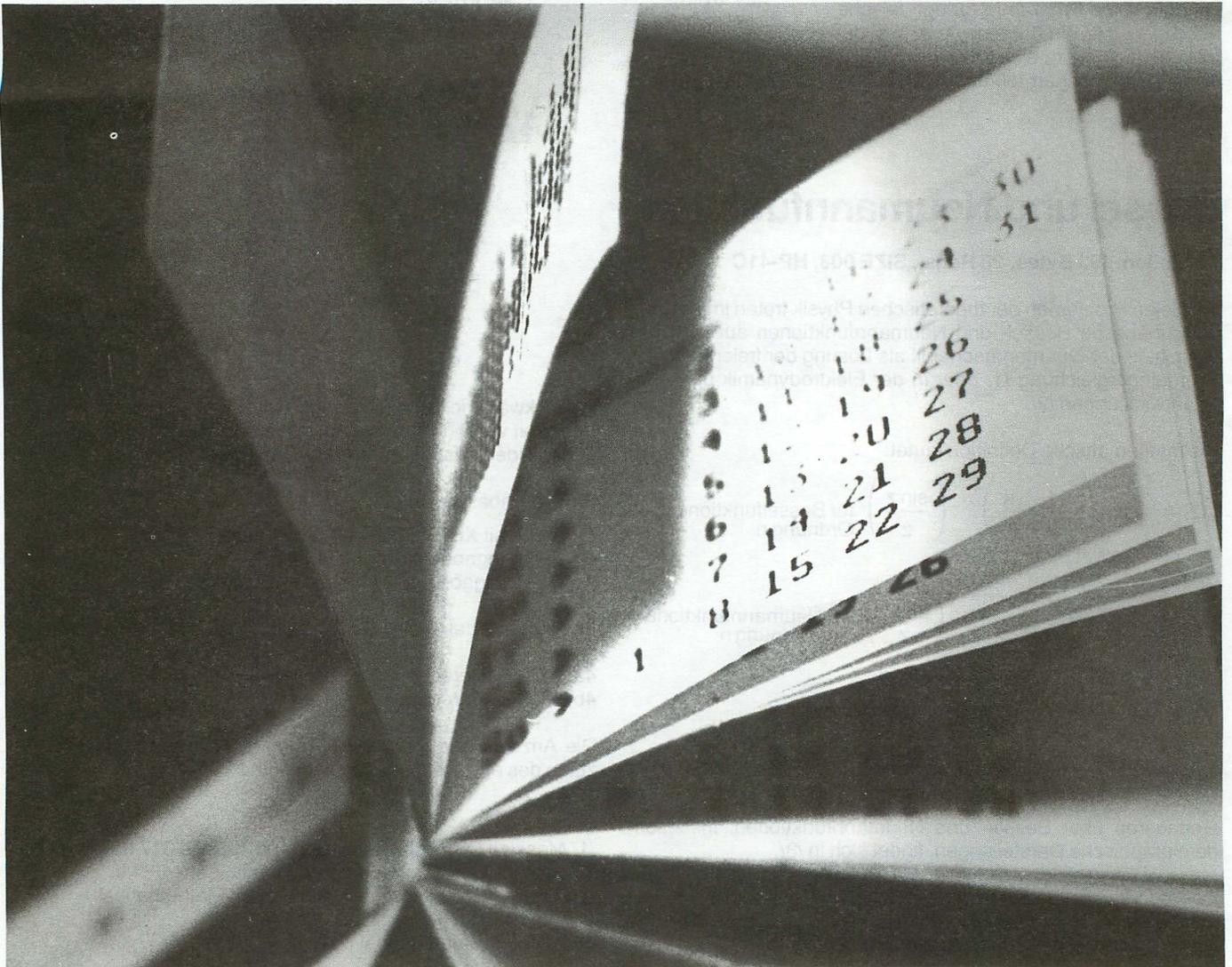
28 STO 04
 29 RCL 00
 30 X#0?
 31 GTO 02
 32 RCL 05
 33 STO 04
 34 GTO 10
 35*LBL 02
 36 E
 37 X<>Y
 38 X=Y?
 39 GTO 10
 40 E3
 41 /
 42 2
 43 +
 44 STO 02
 45*LBL 03
 46 XEQ 05
 47 ISG 02
 48 GTO 03
 49 GTO 10
 50*LBL A
 51 .
 52 STO 07
 53 SF 01
 54 RCL 01

55 INT
 56 ABS
 57 RCL 00
 58 X<Y?
 59 RDH
 60 6
 61 +
 62 E-3
 63 +
 64 STO 02
 65 .
 66 STO 05
 67 E-20
 68 STO 04
 69*LBL 04
 70 XEQ 05
 71 RCL 02
 72 INT
 73 2
 74 -
 75 RCL 00
 76 X#Y?
 77 GTO 09
 78 RCL 04
 79 STO 06
 80*LBL 09
 81 RCL 04

82 X+2
 83 RCL Z
 84 2
 85 *
 86 E
 87 +
 88 *
 89 ST+ 07
 90 DSE 02
 91 GTO 04
 92 RCL 06
 93 RCL 07
 94 SORT
 95 /
 96 STO 04
 97 GTO 10
 98*LBL 05
 99 RCL 05
 100 CHS
 101 RCL 02
 102 INT
 103 E
 104 -
 105 2
 106 *
 107 E
 108 +

109 RCL 01
 110 /
 111 RCL 04
 112 STO 05
 113 *
 114 +
 115 STO 04
 116 RTN
 117*LBL 10
 118 FS? 01
 119 "J"
 120 FC? 01
 121 "N"
 122 FIX 0
 123 ARCL 00
 124 "I="

125 SCI 4
 126 ARCL 04
 127 FIX 9
 128 PROMPT
 129 END
 Werner Moßner
 Kernriedstr. 18
 8900 Augsburg 1



Jahreskalender 1989

– zum Selbermachen –

165 Zeilen, 401 Bytes, 58 Regs., SIZE 004, HP-41CX, IL, PRINTER

Nochmals ein Kalenderprogramm, diesmal für ein ganzes Jahr, mit einer Anleitung, daraus einen handlichen Taschenkalender zu fertigen.

Diesem Programm liegt der Monatskalender von O. Lohkamp, Prisma 87.2.21 zugrunde.

In Prisma 6/87, S.30 ist für die Serie 71 ein Jahreskalender erschienen. Da ich als 41-er damit nichts anfangen kann, konnte mir dieses Programm keine Hilfe geben.

Es werden drei Blätter in den ThinkJet-Drucker eingelegt. Nach erfolgtem Ausdruck kann man die Blätter umwenden und den Kalender nochmals ausdrucken lassen.

Wenn man den Ausdruck nach dem Schema ausschneidet, ihn mit zwei Deckeln versieht, am linken Rande zusammenklebt (lumbekkt), ein Rückenband aufklebt, die Ränder beschneidet, dann hat man einen Taschenkalender. Die Jahreszahl wird vorn aufgeklebt.

Will man nach einem manuellen Programm-STOP von vorne anfangen, dann ist Flag 17 zu löschen.

KALENDR

```

009:FA 1B 26 6B 31 53 1B
    26 64 44 20
041:F5 7F 20 0A 0A 0A
043:FD 1B 26 6C 31 6C 34
    36 46 1B 26 6B 33 53
044:F5 7F 1B 26 64 40
049:F2 7F 0D
066:F6 4A 61 6E 75 61 72
069:F7 46 65 62 72 75 61
    72
072:F4 4D CC 72 7A
075:F5 41 70 72 69 6C
078:F3 4D 61 69
081:F4 4A 75 6E 69
084:F4 4A 75 6C 69
087:F6 41 75 67 75 73 74
090:F9 53 65 70 74 65 6D
    62 65 72
093:F7 4F 6B 74 6F 62 65
    72
096:F8 4E 6F 76 65 6D 62
    65 72
099:F8 44 65 7A 65 6D 62
    65 72
112:F2 4D 6F
114:F2 44 69
116:F2 4D 69
118:F2 44 6F
120:F2 46 72
122:F2 53 61
124:F2 53 6F
128:F3 0A 0A 0A
135:F3 7F 20 FD
    
```

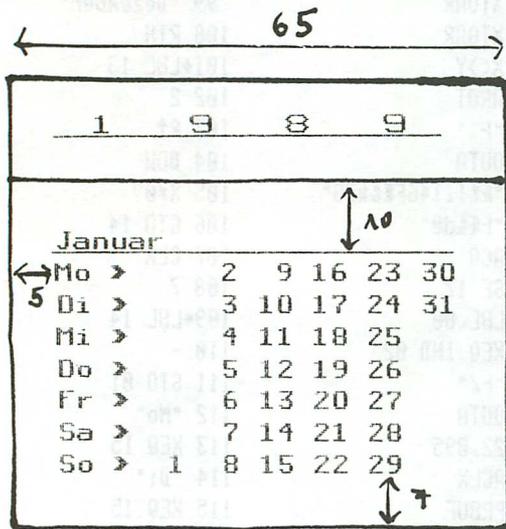
```

01*LBL "KALENDR"
02 DMY
03 1,012
04 STO 02
05 FIX 0
06 SF 21
07 SF 25
08 CF 29
09 "E&k1SE&dD "
10 ACA
11 "JAHR="
12 PROMPT
13 ENTER↑
14 ENTER↑
15 ENTER↑
16 E6
17 /
18 1,01
19 +
20 STO 03
21 X<>Y
22 CLA
23 ARCL X
24 E
25 AROT
26 32
27 XTOAR
28 XTOAR
29 X<>Y
30 AROT
31 X<>Y
32 XTOAR
33 XTOAR
34 X<>Y
35 AROT
    
```

```

36 X<>Y
37 XTOAR
38 XTOAR
39 X<>Y
40 AROT
41 "+ "
42 OUTA
43 "E&11146FE&k3S"
44 "+E&d0"
45 ACA
46 SF 17
47*LBL 00
48 XEQ IND 02
49 "+L"
50 OUTA
51 22,095
52 ACLX
53 PRBUF
54 RCL 03
55 ENTER↑
56 ENTER↑
57 ENTER↑
58 E-2
59 +
60 DDAYS
61 FC? 25
62 31
63 STO 00
64 GTO 13
65*LBL 01
66 "Januar"
67 RTN
68*LBL 02
69 "Februar"
70 RTN
71*LBL 03
72 "Mrz"
73 RTN
74*LBL 04
75 "April"
76 RTN
77*LBL 05
78 "Mai"
79 RTN
80*LBL 06
81 "Juni"
82 RTN
83*LBL 07
84 "Juli"
85 RTN
86*LBL 08
87 "August"
88 RTN
89*LBL 09
90 "September"
91 RTN
92*LBL 10
93 "Oktober"
94 RTN
95*LBL 11
96 "November"
97 RTN
98*LBL 12
99 "Dezember"
100 RTN
101*LBL 13
102 2
103 R↑
104 DOW
105 X#0?
106 GTO 14
107 CLX
108 7
109*LBL 14
110 -
111 STO 01
112 "Mo"
113 XEQ 15
114 "Di"
115 XEQ 15
116 "Mi"
117 XEQ 15
118 "Do"
119 XEQ 15
120 "Fr"
121 XEQ 15
122 "Sa"
123 XEQ 15
124 "So"
125 XEQ 15
126 E-?
127 ST+ 03
128 ""
129 ACA
130 ISG 02
131 GTO 00
132 CF 17
133 STOP
134*LBL 15
135 "E "
136 OUTA
137 CLA
138 6
139 RCL 01
140*LBL 16
141 " "
142 RCL 00
143 X<>Y?
144 GTO 17
145 CLX
146 E
147 X>Y?
148 GTO 17
149 CLX
150 E1
151 X>Y?
152 " "
153 ARCL Y
154 GTO 18
155*LBL 17
156 " "
157*LBL 18
158 CLX
159 7
160 +
161 DSE Y
162 GTO 16
163 PRA
164 ISG 01
165 END
    
```

Schema zum Ausschneiden, Maße in mm



Weihnachtsgrüße

112 Zeilen, 296 Bytes, 43 Regs., SIZE 004, HP-41CX, IL, PRINTER

Einer unserer eifrigsten Programmautoren dieses Jahres möchte seine Arbeit für dieses Jahr mit einem Gruß an alle Mitglieder und Leser von PRISMA abschließen.

Damit auch andere an diesem Gruß teilhaben können, dieses nun folgende Programm; den Ausdruck kann man ja auf Weihnachtskarten kleben, dienicht in sonnige Gefilde verschickt werden, bis zum Frühling ist der Tannenbaum dann sowieso bei allen verschwunden (ausbleicht).

Durch die Veränderung ab Zeile 075 können persönliche Weihnachts- und Neujahrswünsche gestaltet werden, es soll ja Leute geben, die unbedingt noch Kerzen haben wollen!

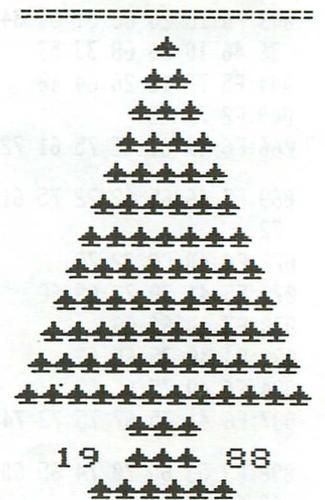
```
01+LBL "WEIH"
02 FIX 0
03 CF 29
04 "08.0"
05 RCL [
06 STO 00
07 E
08 STO 01
09 STO 03
10 12
11 STO 02
12 24,061
13 ACLX
14 PRBUF
15 SF 12
```

```
16 RCL 00
17+LBL 00
18 ACSPEC
19 DSE 01
20 GTO 00
21 ISG 03
22 " "
23 RCL 03
24 STO 01
25 FMT
26 PRBUF
27 X<Y
28 DSE 02
29 GTO 00
30+LBL 01
31 RCL 00
32 ACSPEC
33 ACSPEC
34 ACSPEC
35 FS?C 00
36 RTN
37 FMT
38 PRBUF
39 DATE
40 E2
41 *
42 FRC
43 E4
44 *
45 ENTER↑
46 ENTER↑
47 E2
48 /
49 INT
50 CLA
51 ARCL X
52 "† "
53 ACA
54 SF 00
55 XEQ 01
56 RCL Z
57 E2
58 /
59 FRC
60 E2
61 *
62 " "
63 ARCL X
64 XEQ 02
65 SF 00
66 XEQ 01
67 ACSPEC
68 ACSPEC
69 ACSPEC
70 FMT
71 PRBUF
72 CF 12
73 PRL
74 SF 12
75 "C C D"
76 XEQ 02
77 CF 12
```

```
78 "wünscht allen"
79 XEQ 02
80 SF 12
81 "F R O H E"
82 XEQ 02
83 "WEIHNACHTEN"
84 XEQ 02
85 CF 12
86 "und ein gutes"
87 XEQ 02
88 SF 12
89 "NEUES JAHR"
90 XEQ 02
91 "****"
92 DATE
93 E2
94 *
95 FRC
96 E4
97 *
98 E
99 +
100 ARCL X
101 "†****"
102 XEQ 02
103 CF 12
104 24,061
105 ACLX
106 PRBUF
107 STOP
108+LBL 02
109 ACA
110 FMT
111 PRBUF
112 .END.
```

WEIH

```
004:F8 F7 10 42 C5 CF F7
2C 10
078:FD 77 1A 6E 73 63 68
74 20 61 6C 6C 65 6E
086:FD 75 6E 64 20 65 69
6E 20 67 75 74 65 73
```



```
***C C D***
wünscht allen
F R O H E
WEIHNACHTEN
und ein gutes
NEUES JAHR
***1989***
```

Dr. Martin Hochenegger
Heidelberger Landstraße 97
6100 Darmstadt 13

CG (Petz) 1628

Computerclub Deutschland e.V.

Zeile 1 (1- 3) CCD-Barcodes
 Zeile 2 (3- 7) CCD-Barcodes
 Zeile 3 (7- 8) CCD-Barcodes
 Zeile 4 (8- 18) CCD-Barcodes
 Zeile 5 (18- 28) CCD-Barcodes
 Zeile 6 (28- 36) CCD-Barcodes
 Zeile 7 (36- 41) CCD-Barcodes
 Zeile 8 (41- 49) CCD-Barcodes
 Zeile 9 (49- 56) CCD-Barcodes
 Zeile 10 (56- 66) CCD-Barcodes
 Zeile 11 (66- 74) CCD-Barcodes
 Zeile 12 (74- 80) CCD-Barcodes
 Zeile 13 (80- 82) CCD-Barcodes
 Zeile 14 (82- 88) CCD-Barcodes
 Zeile 15 (88- 90) CCD-Barcodes
 Zeile 16 (90- 92) CCD-Barcodes
 Zeile 17 (92- 96) CCD-Barcodes
 Zeile 18 (96- 100) CCD-Barcodes
 Zeile 19 (100- 104) CCD-Barcodes
 Zeile 20 (104- 106) CCD-Barcodes
 Zeile 21 (106- 114) CCD-Barcodes
 Zeile 22 (114- 123) CCD-Barcodes
 Zeile 23 (123- 130) CCD-Barcodes
 Zeile 24 (130- 137) CCD-Barcodes
 Zeile 25 (137- 146) CCD-Barcodes
 Zeile 26 (146- 154) CCD-Barcodes
 Zeile 27 (154- 164) CCD-Barcodes
 Zeile 28 (164- 172) CCD-Barcodes

Zeile 29 (172- 181) CCD-Barcodes
 Zeile 30 (181- 189) CCD-Barcodes
 Zeile 31 (189- 198) CCD-Barcodes
 Zeile 32 (198- 207) CCD-Barcodes
 Zeile 33 (207- 217) CCD-Barcodes
 Zeile 34 (217- 226) CCD-Barcodes
 Zeile 35 (226- 238) CCD-Barcodes
 Zeile 36 (238- 247) CCD-Barcodes
 Zeile 37 (247- 255) CCD-Barcodes
 Zeile 38 (255- 262) CCD-Barcodes
 Zeile 39 (262- 269) CCD-Barcodes
 Zeile 40 (269- 278) CCD-Barcodes
 Zeile 41 (278- 287) CCD-Barcodes
 Zeile 42 (287- 297) CCD-Barcodes
 Zeile 43 (297- 307) CCD-Barcodes
 Zeile 44 (307- 318) CCD-Barcodes
 Zeile 45 (318- 331) CCD-Barcodes
 Zeile 46 (331- 341) CCD-Barcodes
 Zeile 47 (341- 348) CCD-Barcodes
 Zeile 48 (348- 358) CCD-Barcodes
 Zeile 49 (358- 367) CCD-Barcodes
 Zeile 50 (367- 374) CCD-Barcodes
 Zeile 51 (374- 382) CCD-Barcodes
 Zeile 52 (382- 391) CCD-Barcodes
 Zeile 53 (391- 399) CCD-Barcodes
 Zeile 54 (399- 409) CCD-Barcodes
 Zeile 55 (409- 419) CCD-Barcodes
 Zeile 56 (419- 429) CCD-Barcodes
 Zeile 57 (429- 433) CCD-Barcodes

LABELS (Hochenegger) 1634

Computerclub Deutschland e.V.

Zeile 1 (1- 3) CCD-Barcodes



Zeile 2 (3- 6) CCD-Barcodes



Zeile 3 (6- 11) CCD-Barcodes



Zeile 4 (11- 17) CCD-Barcodes



Zeile 5 (17- 23) CCD-Barcodes



Zeile 6 (23- 30) CCD-Barcodes



Zeile 7 (30- 34) CCD-Barcodes



Zeile 8 (34- 43) CCD-Barcodes



Zeile 9 (43- 49) CCD-Barcodes



Zeile 10 (49- 51) CCD-Barcodes



Zeile 11 (51- 59) CCD-Barcodes



Zeile 12 (59- 67) CCD-Barcodes



Zeile 13 (67- 75) CCD-Barcodes



Zeile 14 (75- 83) CCD-Barcodes



Zeile 15 (83- 91) CCD-Barcodes



Zeile 16 (91- 98) CCD-Barcodes



Zeile 17 (98- 108) CCD-Barcodes



Zeile 18 (108- 116) CCD-Barcodes



Zeile 19 (116- 123) CCD-Barcodes



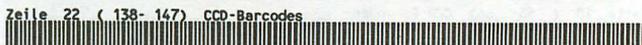
Zeile 20 (123- 131) CCD-Barcodes



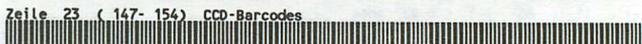
Zeile 21 (131- 138) CCD-Barcodes



Zeile 22 (138- 147) CCD-Barcodes



Zeile 23 (147- 154) CCD-Barcodes



Clubadressen:

1. Vorsitzender

Prof. Dr. Wolfgang Fritz (125)
Kronenstraße 34
7500 Karlsruhe
GEO1: W.FRITZ

2. Vorsitzender

Erich H. Klee (1170)
Ruhrallee 8
4300 Essen 1
GEO1: E.H.KLEE

Schatzmeister Mitgliederverwaltung

Dieter Wolf (1734)
Pützerstraße 29
6000 Frankfurt 90
☎ 069 / 765912
GEO1: D.WOLF

1. Beisitzer

Werner Dworak (607)
Allewind 51
7900 Ulm
☎ 07304 / 3274
GEO1: W.DWORAK

2. Beisitzer

Geowissenschaften
Alf-Norman Tietze (1909)
Thudichumstraße 14
6000 Frankfurt 90
☎ 069 / 7893995
GEO1: A.N.TIETZE

PRISMA-Nachsendedienst

CCD e.V.
Postfach 11 04 11
6000 Frankfurt 1
☎ 069 / 765912

Programm-Bibliothek HP-41

Beirat

Martin Meyer (1000)
Robert-Stolz-Straße 5
6232 Bad Soden 1

Programm-Bibliothek HP-71

Henry Schimmer (786)
Homburger Landstraße 63
6000 Frankfurt 50

Serie 80 Service

Klaus Kaiser (1661)
Mainzer Landstraße 561
6230 Frankfurt am Main 80
☎ 069 / 397852

Beirat

MS-DOS Service

Alexander Wolf (3303)
Pützerstraße 29
6000 Frankfurt 90
☎ 069 / 765912

Hardware 41

Winfried Maschke (413)
Ursulakloster 4
5000 Köln 1
☎ 0221 / 131297

Grabau GR7 Interface

Holger von Stillfried (2641)
Alsterkrughaussee 212
2000 Hamburg 60
☎ 040 / 5116346

CP/M-80 Service

Peter-C. Spaeth
Michaeliburgstraße 4
8000 München 80

E-Technik

Werner Meschede (2670)
Sorpestraße 4
5788 Siedlingshausen

Mathematik

Andreas Wolpers (349)
Steinstraße 15
7500 Karlsruhe

Vermessungswesen

Ulrich Kulle (2719)
Schnuckentritt 14
3000 Hannover 51
☎ 0511 / 6042728

Regionalgruppe Berlin

Jörg Warmuth (79)
Wartburgstraße 17
1000 Berlin 62

Regionalgruppe Hamburg

Alfred Czaya (2225)
An der Bahn 1
2061 Sülfeld
☎ 040 / 433668 (Mo.-Do. abends)

Horst Ziegler (1361)
Schüslerweg 18 b
2100 Hamburg 90
☎ 040 / 7905672

Beirat

Regionalgruppe Karlsruhe

Stefan Schwall (1695)
Rapenwörtstraße 42
7500 Karlsruhe 21
☎ 0721 / 576756
GEO1:S.SCHWALL

Regionalgruppe Köln

Frank Ortmann (1089)
Okerstraße 24
5090 Leverkusen 1

Regionalgruppe München

Victor Lecoq (2246)
Seumestraße 8
8000 München 70
☎ 089 / 789379

Regionalgruppe Rhein-Main

Andreas Eschmann (2289)
Lahnstraße 2
6096 Raunheim
☎ 61442 / 46642

Beirat

Peter Kemmerling (2466)
Danzigerstraße 17
4030 Ratingen

Beirat

Ulrich Schwaderlap (438)
An den Berken 34
5840 Schwerte 6

Beirat

Günther Schwarz (2658)
Bodelschwinghstraße 34
3408 Duderstadt 1

Atari Service

Werner Müller
Classen-Kappelmanstr. 30 a
5000 Köln 41

Zeile 24 (154- 161) CCD-Barcodes



Zeile 25 (161- 168) CCD-Barcodes



Zeile 26 (168- 175) CCD-Barcodes



Zeile 27 (175- 180) CCD-Barcodes



Zeile 28 (180- 186) CCD-Barcodes



Zeile 29 (186- 190) CCD-Barcodes



Zeile 30 (190- 198) CCD-Barcodes



Zeile 31 (198- 204) CCD-Barcodes



Zeile 32 (204- 208) CCD-Barcodes



Zeile 33 (208- 216) CCD-Barcodes



Zeile 34 (216- 223) CCD-Barcodes



Zeile 35 (223- 228) CCD-Barcodes



Zeile 36 (228- 233) CCD-Barcodes



Zeile 37 (233- 237) CCD-Barcodes



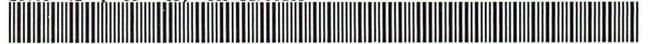
Zeile 38 (237- 241) CCD-Barcodes



Zeile 39 (241- 247) CCD-Barcodes



Zeile 12 (55- 63) CCD-Barcodes



Zeile 13 (63- 69) CCD-Barcodes



Zeile 14 (69- 72) CCD-Barcodes



Zeile 15 (72- 78) CCD-Barcodes



Zeile 16 (78- 84) CCD-Barcodes



Zeile 17 (84- 88) CCD-Barcodes



Zeile 18 (88- 92) CCD-Barcodes



Zeile 19 (92- 96) CCD-Barcodes



Zeile 20 (96- 99) CCD-Barcodes



Zeile 21 (99- 106) CCD-Barcodes



Zeile 22 (106- 114) CCD-Barcodes



Zeile 23 (114- 118) CCD-Barcodes



Zeile 24 (118- 123) CCD-Barcodes



Zeile 25 (123- 127) CCD-Barcodes



Zeile 26 (127- 133) CCD-Barcodes



Zeile 27 (133- 140) CCD-Barcodes



Zeile 28 (140- 148) CCD-Barcodes



Zeile 29 (148- 154) CCD-Barcodes



Zeile 30 (154- 161) CCD-Barcodes



Zeile 31 (161- 165) CCD-Barcodes



KALENDR (Hochenegger) 1639

Computerclub Deutschland e.V.

Zeile 1 (1- 2) CCD-Barcodes



Zeile 2 (2- 8) CCD-Barcodes



Zeile 3 (8- 10) CCD-Barcodes



Zeile 4 (10- 16) CCD-Barcodes



Zeile 5 (16- 25) CCD-Barcodes



Zeile 6 (25- 32) CCD-Barcodes



Zeile 7 (32- 40) CCD-Barcodes



Zeile 8 (40- 43) CCD-Barcodes



Zeile 9 (43- 44) CCD-Barcodes



Zeile 10 (44- 50) CCD-Barcodes



Zeile 11 (50- 55) CCD-Barcodes



UHR (Hochenegger) 1638

Computerclub Deutschland e.V.

Zeile 1 (1- 4) CCD-Barcodes



Zeile 2 (4- 8) CCD-Barcodes



Zeile 3 (8- 12) CCD-Barcodes



Zeile 4 (12- 18) CCD-Barcodes



Zeile 5 (18- 20) CCD-Barcodes



Zeile 6 (20- 22) CCD-Barcodes



Zeile 7 (22- 26) CCD-Barcodes



Postvertriebsstück
Gebühr bezahlt

D 2856 F

CCD – Computerclub Deutschland e.V.
Schwalbacherstraße 50
D-6000 Frankfurt am Main 1

CCD

SN 0176-8735

PRISMA

September 1988 Nr.6

Zeile 8 (26- 32) CCD-Barcodes



Zeile 9 (32- 36) CCD-Barcodes



Zeile 10 (36- 39) CCD-Barcodes



Zeile 11 (39- 43) CCD-Barcodes



Zeile 12 (43- 48) CCD-Barcodes



Zeile 13 (48- 53) CCD-Barcodes



Zeile 14 (53- 58) CCD-Barcodes



Zeile 15 (58- 66) CCD-Barcodes



Zeile 16 (66- 71) CCD-Barcodes



Zeile 17 (71- 73) CCD-Barcodes



Zeile 18 (73- 77) CCD-Barcodes



Zeile 19 (77- 81) CCD-Barcodes



Zeile 20 (81- 84) CCD-Barcodes



ROMB (Huppertz) 1642

Computerclub Deutschland e.V.

Zeile 1 (1- 8) CCD-Barcodes



Zeile 2 (8- 17) CCD-Barcodes



Zeile 3 (17- 25) CCD-Barcodes



Zeile 4 (25- 34) CCD-Barcodes



Zeile 5 (34- 40) CCD-Barcodes



Zeile 6 (40- 47) CCD-Barcodes



Zeile 7 (47- 56) CCD-Barcodes



Zeile 8 (56- 65) CCD-Barcodes



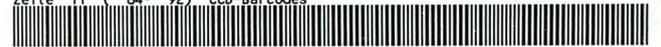
Zeile 9 (65- 74) CCD-Barcodes



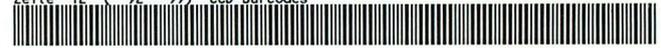
Zeile 10 (74- 84) CCD-Barcodes



Zeile 11 (84- 92) CCD-Barcodes



Zeile 12 (92- 99) CCD-Barcodes



Zeile 13 (99- 108) CCD-Barcodes



Zeile 14 (108- 113) CCD-Barcodes



Zeile 15 (113- 121) CCD-Barcodes



Zeile 16 (121- 127) CCD-Barcodes



Zeile 17 (127- 134) CCD-Barcodes



Zeile 18 (134- 135) CCD-Barcodes



Zeile 19 (135- 143) CCD-Barcodes



Zeile 20 (143- 148) CCD-Barcodes



Zeile 21 (148- 151) CCD-Barcodes



Zeile 22 (151- 154) CCD-Barcodes



Zeile 23 (154- 159) CCD-Barcodes



Zeile 24 (159- 165) CCD-Barcodes



Zeile 25 (165- 169) CCD-Barcodes



Zeile 26 (169- 178) CCD-Barcodes



Zeile 27 (178- 178) CCD-Barcodes



ZL (Hochenegger) 1637

Computerclub Deutschland e.V.

Zeile 1 (1- 5) CCD-Barcodes



Zeile 2 (5- 7) CCD-Barcodes



Zeile 3 (7- 16) CCD-Barcodes



Zeile 4 (16- 20) CCD-Barcodes

