MTX User-Club Deutschland

Info 24
07.12.1987

Zweck: Zusammentragen und Austausch von Tips & Tricks u.s.w., Hilfestellung bei allen möglichen Problemen, Aufbau einer Programmbibliothek und Basteln von Hardware-Erweiterungen.

Programme (nur **Selbstgeschriebenes**): Tausch von kurzen und einfachen Routinen. Gute Programme (mit Dokumentation) können über den Club an alle Mitglieder verkauft werden. Wer solche Programme an uns schickt erhält ggf. Verbesserungshinweise und eine Besprechung im Info.

Mitglied kann jeder werden! Keine Beitragsgebühr! Anmeldung kostet DM 1.-.

Verpflichtungen: Einsendung unseres Anmeldeformulars.

Bitte: Einsendung von Tips & Tricks, Fragen, Antworten, kurzen Routinen, Programmen, Beiträgen zum Info, Hinweisen auf preiswerte Hard- und Software, und was noch so zusammenkommt und andere interessieren könnte.

Club-Info, unser Blatt, verschicken wir ca. 8-wöchentlich. Inhalt ist alles was uns über den MTX/FDX (ohne Copyright) in die Hände fällt. Es kostet nicht über DM 12.- (90 Seiten) je Exemplar. Jeder kann dazu Beiträge liefern und hier gratis Kleinanzeigen veröffentlichen.

Kosten: Wir berechnen ausschließlich Selbstkosten und verschicken **nichts**, wenn's Guthaben nicht reicht! (s.u.)
Schüler, Studenten, Auszubildende, W15-er, Rentner und Arbeitslose erhalten einen Nachlaß von 40% auf die zukünftigen Infos nach Einsendung einer entsprechenden Bescheinigung. Die Bescheinigung gilt nur für den auf ihr genannten Gültigkeitzeitraum.

Geld/Konto: Für jedes Mitglied führt Herbert zur Nedden ein Konto, von dem die jeweils enstehenden Kosten abgehen. Der Kontostand wird bei **jeder** Sendung mitgeteilt (**er steht über der Anschrift**), und kann selbstverständlich jederzeit erfragt werden! Wir verschicken nur gegen Vorkasse!

Einzahlungen bitte auf's Club-Konto: (oder V-Scheck)
(**Absender**! incl Name **und** Anschrift nicht vergessen!)
Postgiroamt Hamburg, BLZ 200 100 20,
Herbert zur Nedden, Sonderkonto C, Nr. 3480 00-200

Kontaktadressen: (nach PLZ geordnet)

Herbert zur Nedden	Christian Löhrmann	Thomas Wulf	Hans Gras
Sonnenau 2	Grevenbleck 24	Roritzer Str. 8	Statenhoek 49
2000 Hamburg 76	3005 Hemmingen 1	8500 Nürnberg 90	NL 1506 VM Zaandam
(040) 200 87 04	(0511) 41 78 77	(0911) 33 52 52	(0031-75) 17 49 91

Telefon-Sprechzeiten
Herbert zur Nedden: Do 18 - 22 Uhr, Sa 13 - 16 Uhr

Ein frohes Fest – schneeweiß mit Kerzen,
Tannenduft und lustig Scherzen,
Vieler Speisen Wohlgerüche,
Die dort dringen aus der Küche,
Wünschen wir von ganzem Herzen!

I n h a l t s v e r z e i c h n i s

Preis für dieses Info: DM 9,50

Redaktionsschluß für Info 25: 20. Januar 1988

MTX User-Club Deutschland Mitgliederliste (Korrektur zur letzten - dBASE-Fehler)

	Hollmann, Rüdiger	Cheruskerweg 48	4930 Detmold	05231 - 88216
	Braun, Peter	Veldastr. 23	5000 Köln 1	0221 - 373490
	Daenell, Wolfgang	Luxemburger Str. 414	5000 Köln 41	0221 - 462791
	Jaron, Marian	Salmstr. 99	5000 Köln 91	0221 - 835201
	Kempf, Eugen Karl	Karl-Schurz-Str. 8	5000 Köln 41	0221 - 436520
	Kochinka, Uwe	Lübeckerstr. 11	5000 Köln 40	0221 - 441186
*	Leinweber, Gerd	Neusser Str. 188	5000 Köln 60	0221 - 732000
	Lorenz, Iris	Altenburger Str. 37	5000 Köln 1	0221 - 311329
	Schmidt, Stephan	Zypressenstr. 12	5000 Köln 71	0221 - 798602
*	Schroeder, Michael	Philippstr. 62	5000 Köln 30	0221 - 512472
	Spruth, Jörg	Martin-Luther-Str. 3	5030 Hürth-Efferen	
	Niehörster, Christof	In der Auen 110	5060 Berg. Gladbach 1	02204 - 61209
	Waschke, Burkhard	Holzer Weg 38	5090 Leverkusen 3	02171 - 81601

Liebe Leserinnen, liebe Leser,

Ich wünsche Euch allen ein **frohes Weihnachtsfest** und einen **Guten Rutsch** sowie das Beste für Neunzehnhundertachtundachtzig!

Bitte entschuldigt, daß das letzte **Info** recht Mager war - und das tortz der knapp 50 Seiten. Leider hatte ich einfach nicht mehr Informationen, wollte Euch aber andererseits nicht noch 2-4 Wochen länger warten lassen.

KLICK.001, die erste Public-Domain zum Thema KLICK ist fertig! Was so drauf ist steht etwas weiter hinten unter 'Clubtreffen'. KLICK.002 ist auch schon in Arbeit, u.a. mit einem neuen HardCopy mit PullDown Menüs, unterstützung von mehr VS4, Edicta-Garfik, ... Auch die Assembler-Routinen zur einfachen Implementierung von PullDown-MenüOberflächen wird wohl mit drauf sein!

Dank einer **dBASE-Macke** fehlten im letzten Info in der Mitgliederliste einige Einträge, die ich auf der vorherigen Seite nachreiche. Peter Würfel hat eine Fehlerursache des dBASE entdeckt. Bei dem READ-Befehl wird auch die Index-Datei mitgepflegt, falls vorhanden - auch wenn sich der Schlüssel nicht ändert, oder das READ garnicht auf Einträge der Datei losgelassen wird. Um diesem **Murks** vorzubeugen sollte der READ-Befehl mit der Option NOUPDATE verwendet werden. Mehr zu dieser Problematik schreibt Peter für Info 25.

Klar, ich werde die **Fragebögen** auswerten, und das Resultat ins Info bringen, aber erst in das nächste (ja, schon wieder eine Vertröstung); schließlich ist die Frist noch nicht abgelaufen. Bislang sind stattliche 25 Exemplare eingetrudelt. Daher verlängere ich die Frist bis zum Freitag, den 08. Januar 1988.

Olaf und ich haben uns **Turbo-Modula** zugelegt. Daher steht zu befürchten, daß sich auch diese Sprache im Club breit machen wird. Der Hauptvorteil gegenüber Turbo-Pascal sind die Module: Es ist möglich Programmteile unabhängig voneinander zu Übersetzen, und später zusammenzubinden (Linken). Außerdem können Zeitunkritische Teile im kürzeren M-Code laufen gelassen werden.

Bernd Preusing hat wieder etwas für die RAM4-Umgebung programmiert. Dieses Programm ist jedoch auch für Nicht-RAM4ler geeignet. Es spart Platz auf der Diskette was Turbo-Pascal-.COM-Files angetrifft. Die Idee ist es, nur einmal die Laufzeitbibliothek als Ladeprogrmm zu haben. Dieses Ladeprogramm ruft das eigentliche Programm, welches als .CHN-Datei vorliegt auf. (Artikel s.u.)

Bitte habt dafür Verständnis, daß vom 22.12.1987 bis Anfang 1988 meine **Club-Sprechstunde** ausfällt.

Ich hoffe gesund!

Herbert z Nedde

Betrifft beiliegende Angebotsliste des MTX User-Club Deutschland

WENN (Du hast kein Angebot in Liste)
  THEN GOTO Seite 2
  ELSE WENN (Dein Angebot kann wegfallen)
    THEN GOTO Seite 2
    ELSE Schreibe Postkarte/Brief an Herbert zur Nedden
      mit Bestätigung Deines Angebotes.

WENN (Bestätigung nicht eingetroffen bis Montag, 15. Januar 1988)
  THEN Angebotsliste wird gekürtzt
  ELSE Angebotsliste wird ggf. entsprechend aktualisiert

C L U B: Wer tut Was / Ports

Wer tut Was

Allgemeines	H. zur Nedden
Info-Inhaltsverzeichnis	H. zur Nedden
(FDX-)BASIC	A. Viebke
BASICODE	H. Gras
CP/M System	H. zur Nedden
Assembler	H. Oppmann
NewWord	U. Grass, H. zur Nedden
Turbo-Pascal	O. Krumnow, T. Wulf
Modula	O. Krumnow, H. zur Nedden
SuperCalc	W. Gieger
Edicta-Grafik	H. zur Nedden, C. Löhrmann, C. Romanazzi
Hardware	H. zur Nedden, P. Kretschmar, U. Hönisch
Reparatur	U. Hönisch, H. zur Nedden, U. Grass

Wer sich auf dieser Liste fehl am Platz oder vermißt fühlt ... schreibe mir. (Bitte nur ernstgemeinte Zuschriften, d.h. Ihr solltet im genannten Bereich "firm" sein).

Ports (zur Nedden, 2000)

Bereich	Port	Verwendung
MTX	00 - 0F	Grudgerät
	10 - 14	SDX-Floppy-Controller!
	18 - 1B	8255-PIO-Box, H. zur Nedden
	1F	vorgesehen für Cassettenmotorsteuerung
FDX	30 - 33	80-Zeichen-Karte
	38 - 39	6845-Controller der 80-Zeichen-Karte
	40 - 47	FDX-Floppy-Controller
	50 - 5F	4x Memotech SiliconDisc
	70 - 73	EPROM/SRAM-Floppy von J. Marquart und F. Cröll
ECB	80 - 83	EDICTA Grafik-Karte
	88 - 8B	Reserviert für HardDisk
	98 - 9B	c't RAM-Floppy
	A0 - A3	EDICTA RAM-Floppy
	A4 - A7	c't EPROM-Floppy
	A8 - AB	c't SRAM-Floppy
	B8 - BB	Conitec-Floppy
	BC - BF	Conitec-Floppy
	C0 - C4	Reserviert für Testzwecke !!!!!!
	CC - CF	Janich & Klass Programmer
	F8 - FB	HD 64180 Sub-Prozessor-Karte

Falls jemand etwas bastelt, und dafür dann Ports belegen möchte, den bitte ich mir diese Pläne möglichst frühzeitig mitzuteilen, damit wir es vermeiden können, daß plötzlich zwei Dinge an der selben Adresse liegen, oder Ports aus einem falschen Bereich verwendet werden. Die adressierbaren Port-Bereiche sind:

MTX	00 - 1F
FDX	20 - 7F
ECB	80 - FF.

Dabei müßt Ihr natürlich beachten, daß in der Tabelle oben einige schon verwendete Port-Adresen genannt sind, die Ihr daher nicht nutzen solltet.

C L U B: Kleinanzeigen

K L E I N A N Z E I G E N

Herbert zur Nedden, Sonnenau 2, 2000 Hamburg 76, 040 - 2008704:
(Preise sind ohne Porto & Verpackung, ich gebe ggf. Mengenrabatt)
- Ich vermittle jederzeit gebrauchte/neue Geräte und Teile der selben. Außerdem weiß ich i.a. was es wo am billigsten gibt.
- SDX, ein Laufwerk, in Top-Zustand (Post-versand-fähig, d.h. rüttelfest, überprüft, und bootet einwandfrei) für DM 750.-
- SDX-Laufwerk, 80-Spur, mit Gehäuse und Trafo - Memotech Original, funktioniert einwandfrei für DM 450,-
- Taxan-Grün-Monitor für VS 4 und 80-Zeichen geeignet, Gehäuse schwarz: DM 70.-
- Interface für Olympia-Carrera, in eigenem Gehäuse, kann neben die Schreibmaschine gestellt werden. DMX 80-Kabel kann zum Anschluß verwendet werden. 100%-ig Centronics-Kompatibel, also auch für andere Computer geeignet: DM 100.-

Solange der Vorrat reicht:
- Platinenstecker für Erweiterungen links am MTX-Grundgerät. Natürlich mit dem Gegenstück zu der Kerbe an Pin 5. je DM 4.-
- Dynamische RAM's 32k x 1 Bit: 8 Stück DM 1.50
- Statische RAM's 2k x 8 Bit (6116): je DM 2.-
- TTL-IC's: 74LS175, 74LS368, 74LS21, 74LS173, 74LS158, 74LS258 je DM 0.50; 74LS10, 74LS11 je DM 0.30
- Z80-Chips: Z80B CPU DM 3.-, Z80 CPU DM 1.-
- Original-Memotech-Spielecassetten:
  Toado, Kilopede, Knuckles, Draughts, Reversi, Snappo, Blobbo, Utilities, Demo für je DM 4.-; StarCommand, Goldmine, Phaid und Flugsimulator (A. Viebke): je DM 6.- MENGENRABATT
  15 Leercassetten, mind C10 mit 2 Boxen für 5 Cass: Zusammen DM 15.-

V E R K A U F

Uwe Grass, Wachholtzstr. 8, 3300 Braunschweig, 0531-343167:
- Laufwerk zu verkaufen! TEAC 40-Spur, mit Headload-Zugmagneten, so gut wie neu, getestet. 200,-- DM, Einbau kann ich machen.
- 32k-Speichererweiterung zu verkaufen, auf Wunsch von Ulrich Hönisch aufgerüstet. 100,-- DM
- MTX und FDX zu verkaufen, Speichererweiterung, Netzteilumbau, Uhr, ein Laufwerk (wenig gelaufen). Preis Vhs.
- dBASE, fertig angepaßt: DM 100,-

Stefan Reher, Am Redder 4, 2000 Wedel:
- MTX 500 mit 5 Cassetten, 21 Infos: VB DM 400,-

Karl-Ludwig Wagner, Am Rosenhügel 1, 4100 Duisburg, 0203-82368:
- MTX 512, RS 232, FDX, 2 Lw (EPSON SD 521), TP 200, MTX in FDX eingebaut, Tastatur-Rundkabel, Alle Anschlüsse hinten an FDX, Netzteil seitlich angeblockt, neuer 80Z-Zeichensatz; folgende Mängel: seit Umbau VS4 nur noch wirre Bilder (Massefehler ?), nach langem Betrieb schwimmt 80Z-Bild etwas, wenn Netzteil ohne Gehäuse Bild besser. (Vermutlich Stormkabel zu lang, Kühlung schwach, Netzteil sollte nachgeregelt wertden): VB DM 700,-

Volker Griener, Sohienstr. 7, 8581 Donndorf, 0921-32427:
- MTX 512, FDX, 2 Lw (1x 77/80Spur), RS232, TP 200, Drucker Star Gemini 10x: DM 1500,-

K O N T A K T E

Jürgen Lindner, Althoffstr. 24, 4220 Dinslaken, 02134-15969:
- Lehrer für Physik und Chemie sucht Kollegen zwecks Austausch von Arbeitspapieren etc.

C L U B: Korrektur / Nachtrag / Clubtreffen

Korrektur / Nachtrag

Info 20, Seite 18: Schaltplan 512k-Karte (Herbert zur Nedden, 2000)
Leider habe ich in dem Schaltplan die Spannungsversorgung der RAM's verkehrtherum angegeben!
Richtig: 5V an Pin 8, 0 Volt (Masse) an Pin 16.

Edicta-Grafik-Software mit KLICK-Version
Siehe Seite 6

Clubtreffen vom 28.11.1987 (Herbert zur Nedden, 2000)

Das Clubtreffen machte viel Spaß und war sehr informativ. Da die Teilnehmerzahl mit 16 Mann recht klein war, haben wir im CLubraum eines Gasthofses getagt. Der Spaß kostete je Teilnehmer DM 30.- incl einem Mittagessen (samt Vor- & Nachspeise und einem Getränk) sowie 2 Tassen Kaffe je Person.

Hier nun die Highlights: (Reihelfolge rein zufällig)

1. Horst Kupka führte den Memotech als **Musikus** vor. Was man da zu hören bekam war sehr beeindruckend. Nach einigen kleinen Verbesserungen wird das Paket im Club angeboten (Public-Domain)
2. Wir werden einen **Grafik-Standard** festlegen.
   Näheres nächste Seite.
3. Wir wollen dem Memotech eine neue CPU verpassen:
   Die **Z280,** eine Z80-Aufwärtskompatible schnelle 16bit-CPU, die (ahhhh!) sich auf dem Bus (d.h. gegenüber dem Rest des Computers) ganz wie eine Z80 benimmt. Aber halt mit anderer interner Frequenz, Pipelining, Cache (was immer das sein mag ... macht die Sache jedenfalls schnell) richtig loslegt.
   Vermutlich wird diese CPU mit eigenem 1 MegaByte RAM auf den ECB-Bus kommen, aber so, daß sie natürlich unter RAM4 richtig tut. Warum auch nicht Disketten formatieren/kopieren und gleichzeitig ein Spiel auf dem Memotech spielen ??
4. Es gibt eine neue Public-Domain **KLICK.001:**
   Diverses zum KLICK, endlich! Die Scheibe, Format 09 (RAM4-Format) kostet DM 15.- und enthält einige KLICKer (u.a Taschenrechner), Spoolerinitialisierung auf eine beliebige Größe, oder auch Spooler-Inhalt auf die Diskette packen, diverse HELP-Dateien mit vielen Informationen, die Menügeführt angesehen werden können, einigen Z80-Libraries mit den für KLICKer und andere RAM4-Programme immer schon gebrauchten Funktionen, ...
   Der Preis für diese PD ist mit DM 15,- höher als für andere, da dieses Geld an Olaf Krumnow geht.
5. **FORMSTAR,** ein fast fertiges Formatierprogramm, welches alle möglichen und unmöglichen Formate formeiteren kann - also ein Vielfaches dessen was FORM4 kann. Es kann sogar im KLICK laufen. Will ich aus NewWord mal rasch sichern, die Scheibe ist voll, keine leere und formatierte zur Hand. NaUnd ? SHIFT-ESC ....
6. **BACKUP,** oder auch wie sichere ich schnell, bequem, und auch noch an den Datum/Zeit-Einträgen im Direktory orientiert ? (d.h. RAM4.x ist Voraussetzung). NaJa, man nehme BACKUP .... ist bald fertig, vermutlich DM 15.-.
7. Wir wollen versuchen eine **Tastatur** an die RS232 anzuschließen - natürlich RAM4-freundlich ... vermutlich müssen für SHIFT-ESC und SHIFT-BS Tasten am Rechner dazu.
8. Noch einiges, aber ich bin vergeßlich ..... sorry

G r a f i k: Standard

Wir werden einen **Grafik-Standard** festlegen.

Mit diesem soll eine eindeutige Schnittstelle zwischen dem Anwenderprogramm und dem eigentlichen Grafiktreiber festgelegt werden.

Die Sache wird dann so aussehen, daß ein Sprung festgelegt wird, und darüber die Grafik-Software änlich wie das BDOS aufgerufen wird: In einem Register die Funktionsnummer, und in einem GCB (Grafics-Control-Block) ähnlich dem FCB (File-Control-Block) für Dateizugriffe die Parameter übergeben werden.

Warum diese merkwürdige Mimik mit einem GCb werdet Ihr fragen.
Nun dafür gibt es zwei gute Gründe:

1. Turbo-Pascal schreibt die an eine Prozedur übergebenen Parameter auf den Stack, oder übergibt auf dem Stack die Adresse. Dies ist von der Art der Übergabe und der Art des Parameters abhängig.
   In Assembler pflegt man die Parameter ingendwo im RAM zu hinterlegen.
   Was BASIC, FORTRAN, Modula, ... so tun will ich hier garnicht genau erörtern - schließlich weiß ich nicht alles!

   Also muß eine universelle Mimik her, die aus allen Programmiersprachen leicht zu nutzen ist.

   Damit steht fest, daß die Parameter nicht auf den Stack kommen, zumal dies nur Zeit kostet!

   Bleibt nur noch die Möglichkeit die Parameter in Registern der CPU zu übergeben, oder sie im RAM an eine geeignete Stelle zu legen.

   Das Auslesen der Register in höheren Programmiersprachen ist bekanntlich nicht unbedingt leicht. Außerdem ist nicht immer sicher, daß ein Unterprogrammaufruf die Register unverändert läßt. Und ausserdem müssen die Parameter in die Register hinein.
   Damit bleibt aus diesen Überlegungen nur ein GCB.

2. Durch die Möglichkeit Assembler-Programme in den RAM4-KLICK-Heap einzulagern wurde der Aufbau des GCB bestimmt.

   Wenn ich die Parameter von einer auf einer anderen Bank laufenden Grafik-Software aus bearbeiten will, muß ich mir diese erst einmal in diese andere Bank kopieren, um den Zugriff nicht unnötig zu verkomplizieren.

   Da das übertragen von Daten von einer in eine andere Bank Zeit kostet, und außerdem ein langer komplizierter GCB Probleme aufwirft (insbesondere bei Erweiterungen), sollte der GCB möglichst kurz ausfallen.

   Wir haben festgestellt, daß alle Aufrufe an die Grafik-Software mit 8 Bytes (= 4 Worte) auskommen. Diese können z.B. zwei Koordinatenpaare (2 mal 2 Worte), für Kreise die Koordinaten des Mittelpunktes und den Radius (3 Worte, d.h. 2 Bytes sind über), oder auch 8 Bytes für ein Füllmuster.

G r a f i k: Standard / Korrekturen der Edicta-Software

Damit ist der GCB schon recht gut eingekreist:
Er ist 8 Bytes = 4 Worte lang.
Die Bedeutung der einzelnen Bytes hängt von der aufgerufenen Grafik-Funktion ab.

Das sind übrigens sogar Erfahrungswerte, da die neue Version der Edicta-Grafik-Software nach diesem Prinzip aufgebaut ist. Sie kann problemlos unter Turnbo-Pascal oder Assembler genutzt werden. Für FDX-BASIC ist die Software auch geeignet, und auch angepaßt. Nur die erforderlichen Programmteile, die die Parameter füllen und die Software aufrufen müssen noch geschrieben werden.

Diese Grafik-Software läuft übrigens sowohl im RAM4-KLICK-Heap, d.h. auf einer anderen Bank, und wird über einen Sysem-Einsprung angesprochen, oder als Overlay hinter dem Programm.

Wer die Edicta-Karte nicht hat, muß nur noch VS 4-Routinen, die den noch nicht fertigen Standard erfüllen schreiben, und schon sind die Programme austauschbar.

Da ich einen echten Flachbettplotter habe, werde ich auch eine Schnittstelle schreiben, die es ermöglicht ohne weiteres auch die Ausgabe auf den Plotter zu schicken. Für diese Routinen sollen genau dieselben Grafik-Funktionen entstehen, damit ich mein Grafik-Programm nicht umschreiben muß.

D.h. Es sollen die Grafik-Programme so geschrieben werden können, daß sie laufen, egal welcher Grafik-Treiber benutzt wird. Je nach vorhandener Hardware wird der entsprechende Treiber aktiviert.

Da der Standard noch nicht 100%-ig fertig ist, muß ich Euch leider auf Info 25 (welch schöne Jubiläums-Zahl) vertrösten.

Korrekturen zur Edicta-Grafik-Software (Herbert zur Nedden, 2000)
Leider sind mir in der Anpassung von Claudios HardCopy-Routine an den DMX80 und der Beschreibung dazu je ein Schnitzer unterlaufen:

1. CopyMod-Aufurf:
   Mode:=ord('D'); CopyMod;

2. Patches für einige Module:
   EDICTA.COM 45C4: 06 in 07
   EDITURBO.CMD 3FB4: 06 in 07
   EDIFDXB.CMD 3FB4: 06 in 07
   Dazu nimmt man am besten MONI, DDT, ...

Außerdem sollten 4 Procedures umbenannt werden:

CsrLeft in CrsLeft ebenso CsrRight, CsrUp, CsrDown.

Das geht mit ^Q^A: Csr in Crs umsetzen.

P a t c h e s: XDIR, DDIR und MENU

Patches (Uwe Grass, 3300)

Nun gibt es für die Flickwerker wieder etwas neues, die Vorraussetzung ist aber eine erweiterte 80-Zeichenkarte in der FDX. Die Patches erhöhen bei einigen Programmen die Anzahl der Zeilen in der Anzeige. So sind einige Directory-Programme etwas zu kurz geraten.
Die Patches im einzelnen:

PATCH in XDIR3.COM:

XDIR3 zeigt nur dann alle Userbereiche an, wenn im Aufruf zum Laufwerksbuchstaben ein ? miteingegeben wurde. Da andere Programme hier ein * benötigen, sind die entsprechenden Abfragen folgendermaßen geändert:

Reaktion auf XDIR3 d*:	445h	2Ah
	17ABh	2Ah
Fünf Zeilen mehr	C71h	42h
	C7Ah	42h.

Nun soll das Programm noch automatisch auf dunklen Schirm zurückschalten:

ab 8C0h bis 8C8h hex: 1B 5B 58 58 44 49 52 33 2C 20
ASCII: ESC [X X D I R 3 , SPACE.

PATCH in DDIR.COM:

28 ZEILEN	101h	1Ch

PATCH in MENU.COM:

Neuer Eintrag, erforderlich wenn das Datum der Erzeugung dieser Version (im Beispiel 1.4) festgehalten werden soll:

von 19Ch bis 7A7h (z.B.) hex: 31 2E 34 20 30 33 2E 30 39 2E 38 37
ASCII: 1 . 4 SPACE 0 3 . 0 9 . 8 7

Anzahl der angezeigten Zeilen auf 28 erhöhen:

31Fh: ändern von 17h auf 1Bh
569h: ändern von 17h auf 1Bh

Meldungen am unteren MENU-Rand kürzer:

388h: 3Fh
389h: 3Eh
von 38Ah bis 399h: 00
von 3B7h bis 3BAh: 00
500h: 00
von 5Efh bis 62Ch: 00

Alles klar?

UG

H a r d w a r e: 8 MHz

Meine Erfahrungen mit 8 MHz

(Dr. Holger Göbel, 8630)

Ja, ich habe es also geschafft: diesen Beitrag schreibe ich mit 8 Mhz, wobei ich nicht die Fingerfertigkeit, sondern die Prozessor-Taktrate meine.
Herbert hat ja in INFO 21-29 eine nahezu komplette Umbauanleitung angegeben. Ganz so einfach war es dann aber doch wieder nicht.
Dazu ist vorab noch etwas zu sagen: erst seit etwa 1/2 Jahr befasse ich mich intensiver mit Elektronik und habe zwar einen (unbedingt notwendigen) Oszillographen, aber eben auch noch meine Schwierigkeiten mit ersterer. Aus dieser Situation heraus folgende Gedanken:

1. Warum überhaupt den Computer schneller machen? Zum einen genügen ja für unsere private Nutzung die installierten 4 Mhz und zum anderen, was besonders Uli Hönisch zu bedenken gegeben hat, wird die Betriebssicherheit ab 6 MHz aufwärts fragwürdig.
   Hinter so einem Projekt steht sicherlich ein gehöriges Maß an Freude, herumzubasteln und das Maximale zu probieren, und Stolz, wenn es gelungen ist. Und schön ist es halt doch, wenn, v.a. in Verbindung mit EPROM- und CMOS-RAM-Floppy der Bildaufbau in NEWWORD nahezu verzögerungslos abläuft. Außerdem stehen wir ja ein wenig in Beweiszwang, daß unsere "veraltete" Z80-Kiste doch gar nicht so viel hinter den 68000-Porsches zurücksteht, auch wenn sich keine der großen Computerzeitungen mehr so recht um uns kümmert.
   Zur Betriebssicherheit: Weder Herbert noch ich (jedenfalls nach unten aufgeführten Modifikationen) haben damit auch nach stundenlanger Betriebsdauer Schwierigkeiten gehabt.
2. Wer sollte sich an den Umbau wagen? Der, der Freude daran und Durchhaltevermögen hat, der sorgfältig genug ist und der ausreichende elektronische Kenntnisse und Werkzeug besitzt (wobei ich, wie gesagt, kein Elektronik-Freak bin), denn mit einfachem Schaltplan-Abkupfern ist es hier nicht getan.
3. Jeder sollte seine Erfahrungen mitteilen. Dazu gehört, und das erscheint mir jetzt besonders wichtig, daß man auch sagt, welche Maßnahmen beim Herumprobieren nicht erfolgreich waren. Denn ohne Herumprobieren geht es nicht, schon wegen der fehlenden Spezial-Meßinstrumente. Weiterhin sollte man detailliert anführen, warum man dies oder das gemacht hat. Nur so spart man anderen Bastlern viel Zeit und kann ihnen entscheidend weiterhelfen. Manchmal liest man, daß dieser oder jene Hard- oder Software-Eingriff bei jemandem erfolgreich war. Wenn schon keine ausreichende Begründung mitgeliefert wird, dann sollte man wenigstens mitteilen, ob die Verbesserung auch reproduzierbar entscheidend war. D.h.: Verschlechtert sich die Situation wieder, wenn die Änderung rückgängig gemacht wird? Hat man nicht etwa zwei oder mehr Änderungen auf einmal vorgenommen und preist dann alle als nützlich an? Ich fand zum Beispiel den Artikel von Jan Bredereke (2000) im letzten INFO (23-17) gut, der von den Zeitkonstanten bei den Reset-Tasten handelte. Er wäre noch etwas besser gewesen, wenn er die Schaltung noch erklärt hätte, wenn er auch ausgeführt hätte, wie die Zeitkonstanten zu berechnen sind und welche Bauteile man gegebenenfalls ändern müßte, wenn man eine andere Verzögerungszeit bräuchte. Denn eines ist klar: Unsere Rechner unterscheiden sich alle ein wenig, weil die Toleranzen der Bauteile immens sind (z.B. reicht die Schaltzeit verschiedener 74LS00-Gatter von 5 bis 15 ns, das ist das dreifache!). Aus dem gleichen Grund wird auch nicht jeder Rechner mit 8 MHz laufen; vielleicht jedoch dann, wenn wirklich Meßdaten bekannt gegeben werden, z.B. die Zeit zwischen RAS- und CAS-Signal, damit man Anhaltspunkte hat, welche Bausteine nötigenfalls durch andere Typen zu ersetzen sind.

H a r d w a r e: 8 MHz

So, genug des Vorgeplänkels, Jan B. möge mir auch nicht böse sein.

Ich habe also alle Maßnahmen durchgeführt, die Herbert im INFO 21-30 ("Was ist zu tun:") vorgeschlagen hat. Die Gleichspannungen führe ich meinem Rechner schon seit längerem über ein externes Netzteil zu, allerdings kann ich auch auf Versorgung aus dem FDX-Netzteil umschalten (mit 78S05 und R60=2,2 Ohm, s. 17-82), was auch keine Betriebsunsicherheiten mit sich bringt (meine Hauptplatine nebst Erweiterungen sitzen weiterhin unter der Tastatur und am ECB-Bus hängen eine 256 k-EPROM und eine CMOS-RAM-Floppy).

Die beiden Schaltungen (WAIT und Frequenzumschaltung) habe ich auf einer Huckepack-Platine untergebracht, die im Sockel des ehemaligen Chips 9D steckt.

Die **Grundplatine** lief sofort anstandslos, auch im MTX-ROM-Modus.

Die **512-k-Karte** war viel widerspenstiger, RAM42 war zunächst nicht zum Laufen zu bringen (Herbert hatte mit ihr überhaupt keine Probleme). Um folgendes zu verstehen, wäre es gut, wenn ihr Euch den Schaltplan der 512-k-Karte (INFO 20-18) und den der Grundplatine (deutsches Handbuch S. 203) zur Hand nähmet; es geht nämlich um die Generierung des RAS- und CAS-Signals:
Dynamische RAMS legen ihr Datum dann auf den Datenbus, wenn sie von der CPU dazu aufgefordert werden, und wenn sie die zugehörige Adresse von ihr erhalten haben. Da wir einen 16-Bit-Adreßbus haben, müßte jeder RAM-Baustein 16 Adreß-Pins haben. Um aber Pins zu sparen, wird die Adresse in zwei Portionen eingelesen: zuerst die unteren 8, dann die oberen 8 Bits. Die unteren 8 Bits geben die "Reihe" ("row"), die oberen die "Spalte" ("column") der Speichermatrix im Inneren des RAM-Bausteins an. Damit nun der RAM-Baustein weiß, wann welche Adreß-Hälfte an seinen Pins anliegt, erhält er zwei Signale: das RAS-Signal ("row adress strobe") für die untere, das CAS-Signal ("column adress strobe") für die obere Adreßhälfte. Zwischen beiden Signalen besorgt ein Multiplexer (das ist der 74LS157) ein Umschalten der Adreß-Signale; dazu wird er über die MPX-("Multiplex-")Leitung aufgefordert. Das MPX-Signal wird also zwischen RAS- und CAS-Signal erzeugt. Natürlich braucht der RAM-Baustein eine gewisse Zeit für die Adreßaufnahme und -dekodierung. Deshalb müssen zwischen den drei Signalen immer einige Nanosekunden vergehen (typischerweise etwa 50-100 ns). Schnelle RAMs haben verkürzte sog. "Zugriffszeiten": Das ist die Zeit, die das RAM **nach dem CAS-Signal** braucht, bis es sein Datum auf den Datenbus legt.

Ein typischer Befehlszyklus des Z80-Prozessors sieht nun so aus:
Beim 1. Takt wird die Adresse des Befehlszählers auf den Adreßbus gelegt. Während des 2. Taktes wird der Befehlszähler um eins erhöht und im 3. Takt holt der Prozessor das Datum aus dem RAM ab (der Takt 4 dekodiert dann dieses Datum). Der RAM-Baustein hat also 2 Takte Zeit, sein Datum freizugeben. Das sind bei 4 MHz 500 ns, bei 8 MHz 250 ns. Wenn also RAS- und CAS-Signal etwa je 50 ns anliegen, dann bleiben als Zugriffszeit maximal noch 150 ns: deshalb also die schnellen Speicher, die dann auch noch in der Toleranz nach oben nicht mehr viel Luft haben.

H a r d w a r e: 8 MHz

Wie werden nun die Signale erzeugt? Schaut nun bitte auf S.203 im Handbuch: Das MREQ-Signal (das ist das Signal des Prozessors, das seine Absicht zum Speicherzugriff - "memory request" - bekannt gibt) wird über zwei Inverter geführt. Das verzögert zwar einige Nanosekunden, dafür hat man mit dieser Schaltung einen billigen Treiber, der das verrauschte CPU-Signal wieder auffrischt. Weil der im Handbuch angegebene 74LS14 zwar gute Signale liefert (er besitzt einen Schmitt-Trigger und eine Temperaturkompensation), aber zu langsam ist (etwa 15 ns Schaltzeit), ist bei uns der 74LS04 (10 ns) eingebaut. Das RAS-Signal ist also fertig.

Das MREQ-Signal wird "durch" das PAL 6A geführt und steht an Pin 14 zur Verfügung. Zwischen zwei Invertern seht ihr hier einen Widerstand und einen Kondensator. In dieser Schaltung wirken diese als Impulsverzögerer, die Verzögerungszeit berechnet sich aus t=R*C, hier also 680 Ohm mal 68 pF = 46 ns. Die beiden Inverter haben für sich eine Schaltzeit von etwa je 10 ns, so daß nach insgesamt etwa 66 ns das MPX-Signal für den Multiplexer zur Verfügung steht.
Nochmals wird jetzt verzögert (330 Ohm mal 47 pF = 16 ns) und nach NAND-Verknüpfung mit dem REFRESH-Signal kommt dann nach weiteren etwa 36 ns das CAS-Signal heraus.

Herbert hat jetzt, um Zeit zu sparen, die beiden Kondensatoren entfernt, so daß nur noch die Schaltzeiten der Gatter die Impulse verzögern. Das hat bei ihm und bei mir auf der Hauptplatine geklappt, vielleicht reagieren andere RAM-Typen empfindlicher, so daß durch Veränderung des Widerstandes oder des Kondensators evtl. andere Verzögerungszeiten ausprobiert werden müßten.

Weil auf der **512-k-Karte** 2*8 Speicherbausteine zur Verfügung stehen, muß noch eine Logik her, die entscheidet, welche Seite der RAMs angesprochen werden soll. Dafür stehen am PROM die Signale S0 und S1 zur Verfügung, die über ein UND-Gatter (74LS08) mit dem CAS-Signal verknüpft werden müssen. Das kostet natürlich wieder Zeit, und die fehlt dann den RAMs für die Zugriffszeit. Also habe ich auch hier die beiden Kondensatoren (C1 und C2) entfernt. Das funktionierte zunächst, aber nach einer Betriebsdauer von etwa 1/4 Stunde (nach zunehmender Erwärmung also) stürzte der Rechner regelmäßig unter RAM42 ab. Ich habe lange probieren müssen, bis ich die richtige Verzögerungskombination fand, die folgendermaßen aussah:
C1 wird entfernt, R1 und R2 bleiben, C2 wird 47 pF.

Anm.d.HzN.: Bei mir: Hauptplatine Kondensatoren weg
768k-Karte Original gelassen C1 47pF, C2 68 pF

Mit dem Oszilloskop habe ich folgende Zeiten zwischen den Signalen gemessen: MREQ-(20 ns)-RAS-(50 ns)-MPX-(70 ns)-CAS. Das funktionierte dann einwandfrei (mit den HM50256P-12 RAMs).

Was hat's dabei nicht bzw. nicht entscheidend gebracht?
Ich habe die rechte Reihe der RAMs von den Sockeln runtergeholt und direkt eingelötet (die andere Seite stak in gedrehten Sockeln, die habe ich belassen).
Das Austauschen der Gatter 74LS04 und 74LS08 gegen schnellere (S-Version) bzw. gegen solche mit saubereren Flanken (HC-Version) klappte zunächst, KLICK lief jedoch nicht (auch nach Rumprobieren mit den RC-Gliedern).

H a r d w a r e: 8 MHz

Der Austausch des 74LS04 gegen einen 74LS14, 74HC14 oder 74S14 brachte gar nichts, RAM42 erkannte die 512-k-Karte nicht.

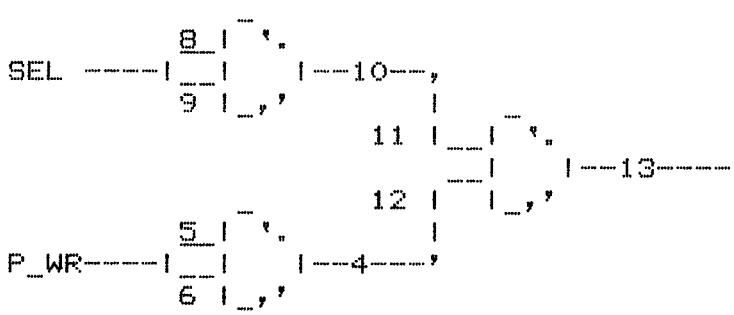
Um das CAS-Signal eher nach dem MPX-Signal folgen zu lassen, habe ich die Verbindung R1 und Pin 13 des 74LS04 gelöst und MREQ auch direkt auf Pin 13 geführt. Offensichtlich war das aber den RAMs zu schnell.
Ob das Zusammenlöten der Hauptplatine mit der Speichererweiterungs- und der RS232-Karte (s.INFO 22-19) entscheidend war, weiß ich nicht. Jedenfalls habe ich es gemacht, und alle befragten Profis (Herbert z.N., Uwe Grass, Ulrich Hönisch, Hagen Wenzek und Gerhard Witzel) meinen, daß es gut war (wegen der Wackelkontakte und der Kapazitäten an Steckverbindungen).

Wenn jemand trotzdem Schwierigkeiten hat, könnte er durch zusätzliches Einführen eines 74LS00 (was allerdings dann schon eine eingreifendere Maßnahme ist) zwischen MPX und CAS ein Gatter einsparen. Das hätte ich als nächstes probiert.

Die **ECB-Karten** brauchten deutliche Nachilfe.
Zuerst die
EPROM-Karte: Sie war mit 8 MHz gar nicht zum Laufen zu bringen, auch nicht, wenn ich WAITs einfügte. Da das aber bei Herbert nicht anders war, habe ich dann aufgegeben und schalte jetzt beim Zugriff auf die EPROM-Karte automatisch auf 4 MHz um (s.u.).
Die **CMOS-RAM-Karte** schien zunächst zu gehen. Als ich aber einen Schreibversuch machte, war ein File kaputt und das System meldete sich mit einem P2DOS-Error (14). Also schalte ich hier, nachdem andere Manipulationen erfolglos waren, beim Schreiben auf 4 MHz zurück (Lesen klappt mit 8 MHz). WAITen habe ich nicht probiert. Übrigens: Auch beim Directory-Lesen wird wohl anfänglich ein ganz kurzer Schreibimpuls erzeugt (?), denn die Taktfrequenz wird ganz kurz runtergeschaltet, was ich mir sonst nicht anders erklären kann (Herbert, Du? oder BP, der RAM-Autor?).

Anm.d.HzN.: Die EPROM-Karte mag auch bei mir keine 8 MHz, jedoch die CMOS-RAM-Karte läuft unter vollen 8 MHz. Beide Karten: c't/U.Grass.

Um diese beiden Konditionen, allgemeiner EPROM-Floppy-Zugriff und CMOS-RAM-Floppy-Schreibzugriff, zu berücksichtigen, führe ich von der EPROM-Floppy das Signal SEL (IC6 74LS688, Pin 19) und von der CMOS-RAM-Floppy das Signal PORTWR (IC8 74LS32, Pin 3) auf die MTX-ECB-Adapter-Platine, wo sie über ein 74LS02 (Huckepack auf den 04 gelötet, d.h. die beiden Pin 7 und die beiden Pin 14 zur Spannungsversorgung zusammenlöten, die übrigen Pins hochbiegen) miteinander verknpüpft werden. (Die beiden Signale sind übrigens LOW-aktiv):



(Das sollen übrigens NAND-Gatter (74LS00) mit Pinbelegung sein.)

Das Ausgangssignal kommt auf das LS02-Gatter (Pin 5 und 6) bei der Frequenzumschaltung, wobei ich die Schaltung von INFO 22-14 verwende!

H a r d w a r e: 8 MHz

Zwar machte bei mir das Beschreiben der Floppy-Controller-Register mit 8 MHz keine Schwierigkeiten, trotzdem habe ich (s. INFO 22-14) sowohl das SEL- als auch das I/O-Signal (IC D3 Pin 8 und IC D2 Pin 8 auf der Controller-Karte) wie angegeben verknüpft. Zu diesem Zweck habe ich einen LS02 Huckepack auf das IC D3 gelötet mit entsprechender Verdrahtung, d.h.: Pin 7 und 14 weden zur Stromversorgung mit den unteren gleichnamigen Pins verbunden, ebenso die beiden Pin 5 und die beiden Pin 8. Die anderen Pins werden hochgebogen. Pin 6 des Huckepack-IC wird an Pin 5 angelötet (rüberbiegen). Ein kurzer Draht verbindet Pin 4 und Pin 9. Pin 10, 11 und 12 werden miteinander verbunden, Pin 13 ist der Ausgang (er wird auf Pin 2 des LS02 der Frequenzumschaltung geführt). Zur Sicherheit kann man noch die beiden nicht benutzten Eingänge Pin 2 und 3 auf +5 V (Pin 14) legen.

Vielleicht fragt Ihr Euch noch, wie ich die verknüpften Signale von der Floppy-Controller-Karte und vom ECB-Bus in den MTX gebracht habe. Da ich keine neuen Kabel legen wollte, habe ich das Flachbandkabel verwendet, einige Signal liegen ja hier einfach auf Masse (z.B. 38 und 54). Auf jeder Steckkarte, die irgendwie mit dem Kabel Kontakt hat, müßt Ihr allerdings dann die Kontakte sicher von Masse trennen (was gar nicht so einfach ist, weil auf der CPM-Karte der Pfostenstecker die Leitungen verdeckt; hier habe ich den Pfostenstecker mit einer Laubsäge gespalten und P. 55-60 ausgelötet). Vielleicht gibt's hier bessere Lösungen.

Was hat's nicht entscheidend gebracht:
Ich habe alle 2*6 Bustreiber auf der RS232- und auf der CP/M-Karte (in der FDX die kleine Platine mit dem Kabelanschluß) ausgetauscht gegen ALS-, S- und HC-Typen. Mit allen Typen lief zwar der Rechner, aber die EPROM-Floppy trotzdem nicht mit 8 MHz. Also habe ich wieder die LS-Typen eingesetzt.
Das 60-adrige Kabel habe ich extrem kurz gemacht - ohne Leistungsverbesserung. Meine Kabellängen: zwischen MTX und FDX 35 cm, zwischen MTX und ECB-Karten 30 cm (demnächst probiere ich noch etwas längere Kabel aus). Der MTX muß bei mir zwischen FDX und ECB-Karten sitzen, sonst laufen die ECB-Karten nicht.

So, nun bin ich sehr ausführlich gewesen, wahrscheinlich wird sich trotzdem ein Hardware-Unbetuchter nicht an den Umbau wagen, wozu ich ihn auch nicht ermuntern möchte. Vielleicht hat er trotzdem etwas Lust bekommen, sich mit Elektronik zu befassen, wozu dich unser Rechner ja dank Standard-Technik im Gegensatz zu den "Porsches" als Versuchsobjekt noch gut eignet. Die Cracks haben natürlich alles schon gewußt, sie wollte ich aber eigentlich dazu anregen, mehr von ihren Aktivitäten und Versuchen preiszugeben, damit die nicht ganz versierten Interessierten, wozu noch eine ganze Menge gehören, mehr davon haben.

Vielleicht kann mir noch einer helfen: Ich boote ja mit Hilfe den neuen BOOT-EPROMs von der CMOS-RAM-Floppy. Seit einiger Zeit (auch unter 4 MHz und in der noch nicht umgabauten Version) kann nach dem Anschalten manchmal eine Diskette nicht gelesen werden (es erscheint eine Lesefehler-Meldung, die den physikalischen Sektor 0 betrifft). Es genügt dann auch nicht, daß ich mit den RESET-Tasten neu boote, sondern ich muß die FDX aus- und wieder einschalten.

H a r d w a r e: 8 MHz / Neues

Kommentare zu 8 MHz (Herbert zur Nedden, 2000)
Hier einige Erfahrungen und Kommentare zu diesen Versuchen:

1. Auf der RS232-Karte und der Busadapter-Karte in der FDX habe ich ausschließlich 74LS-Typen als Treiberbausteine. Mit 74ALS-Typen lief der Rechner nicht. Jemand hat 74HC-typen eingesetzt, und der Rechner läuft mit 8 MHz einwandfrei: 74HC's liefern i.a sauberere Signale, sind aber nur für Leitungslängen von 10 cm konzipiert! Das kann also auch schief gehen.
   Auf jeden Fall sollten die Bustreiber auf den beiden Platinen nur aus einer 74xx-Serie stammen.
2. Der Floppy-Controller mag bekanntlich nur unter 4 MHz laufen. Diese Frequenz wird insbesondere für das RAM-Zugriffs-Timing des Controllers benötigt, da dieser für das Lesen und Schreiben von Sektoren selbst auf das RAM zugreift, und dabei die CPU vom Rechner vorübergehend abkoppelt.
   Peter Kretzschmar hatte die Idee, den Floppy-Controller immer mit 4 MHz zu versorgen, damit die CPU-Frequenz nicht für Diskettenzugriffe umgeschaltet werden muß - was macht es, wenn die CPU mit 8 MHz getaktet wird, während sie nichts tut ? Leider funktioniert dies nicht! Also muß bei Diskettenzugriffen die Frequenz gedrosselt werden.
3. Nicht alle RAM-Bausteine laufen mit 8 MHz - und das egal, ob sie gemäß Hersteller die gleichen Timing-Spezifikationen erfüllen. Mit TMM-RAMs lief mein MTX unter 8 MHz, diverse andere mochte er nicht. Claudio Romanazzi's MTX mochte übrigens nicht einmal mit TMM-RAMs mit vollen 8 MHz, wohl aber mit einem Wait.

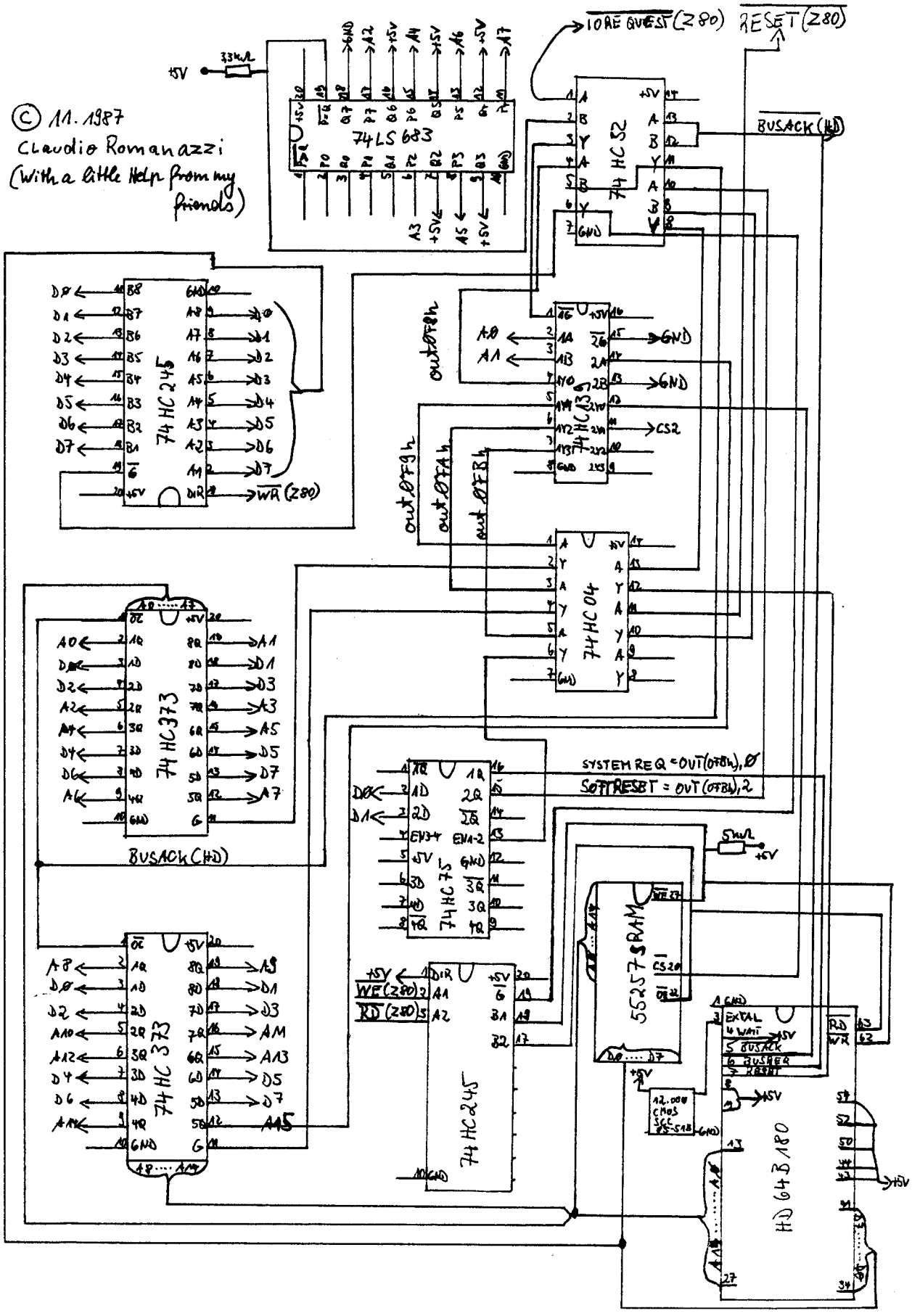
Speichermillion für die EPROM-Floppy auf dem ECB-Bus (Uwe Grass, 3300)
Die c't EPROM-Floppy ist erwachsen geworden. Sie kann von Stund' an auch 1 MegaByte (1048576 Byte) Programme aufnehmen. Das Beste daran ist, das auf der Platine außer der Jumperung nichts zu ändern ist. Es werden lediglich neue Eproms (27011) in die Sockel gesteckt. Zum Programmieren braucht man natürlich ein neues Programm. Dies hat Bernd Preusing aber bereits fertig und auch getestet, so daß auch dies keine Hürde darstellt. Die Programmierlogik ist ja sowieso schon auf der Platine vorhanden. Eine angepaßte RAM4-Version ist ebenfalls fertig.
Das einzige, das jetzt noch zu ergründen wäre, ist der Preis für diese Eproms. Wahrscheinlich werde ich ihn noch rechtzeitig für dieses Info bekommen, Herbert wird ihn (handschriftlich?) nachtragen müssen. Wenn nichts im Info steht, müßt ihr mich (bei Bedarf) per Telefon befragen. Aber weniger als 85,-- DM (offizieller Preis) werden es wohl werden.

UG

HD64180-CPU-Karte (Herbert zur Nedden, 2000)
Claudio Romanazzi hat lange getüftelt und gelötet. Jetzt ist seine ECB-Karte mit einer HD-CPU und eigenem RAM fertig. Er war so freundlich, mir seinen Schaltplan zu geben, der auf der nächsten Seite kommentarlos zu sehen ist. Just rief er an: Sie läuft unter CP/M. Ein Artikel dazu steht am Ende des Infos.

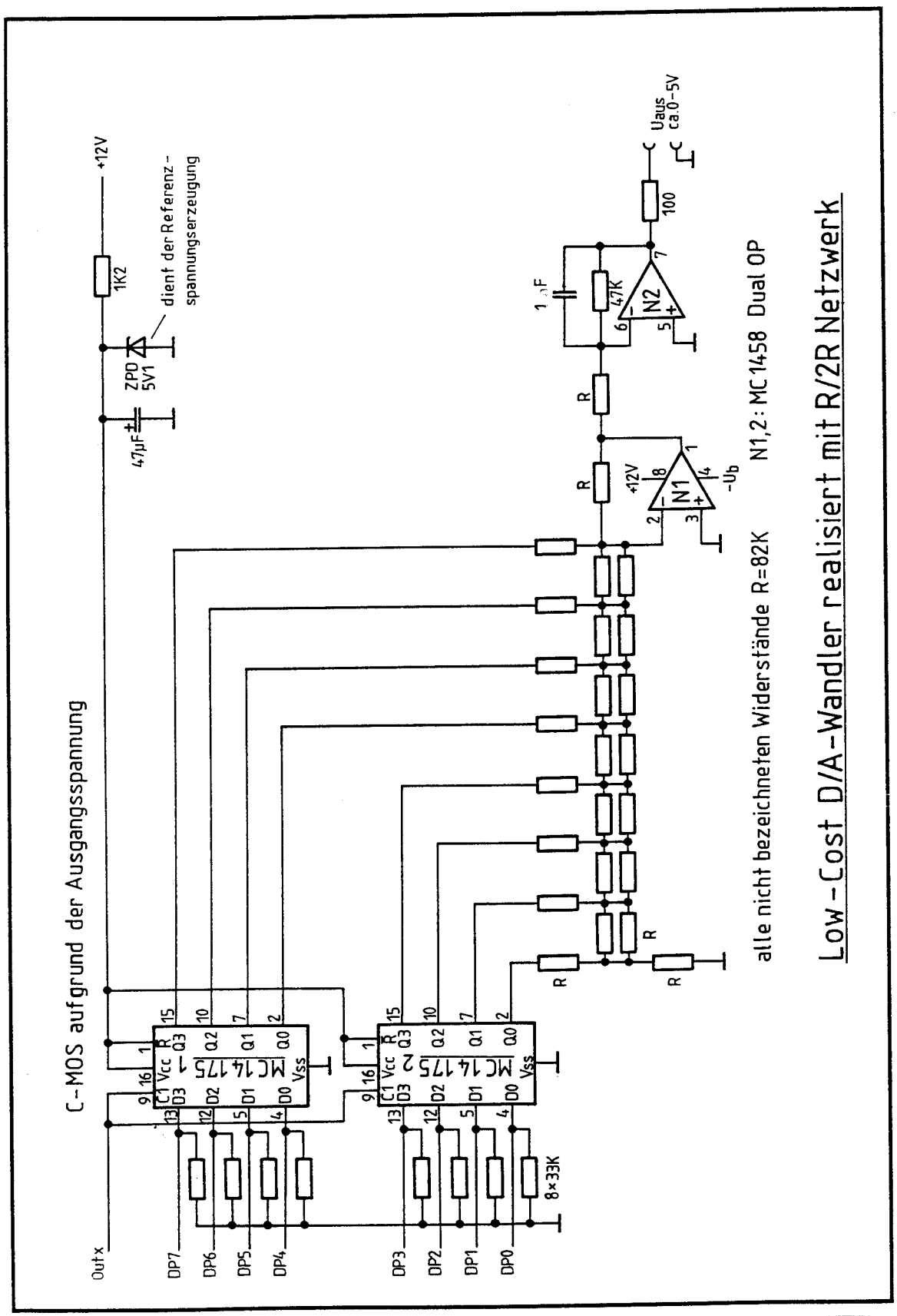
Low-Cost DA-Wandler (Herbert zur Nedden, 2000)
Horst Kupka hat nun ja das mit der Musik am laufen, und nun auch einen schönen Schaltplan dazu gezeichnet. Er folgt auf der übernächsten Seite.
Wer meint, die Schaltung sei aufwendig, hat recht. Aber dafür ist die Chose preiswert! - und funktioniert.

Hardware: HD64180-CPU-Karte



H a r d w a r e: Low-Cost DA-Wandler

(Horst Kupka, 4019)



F U N K: Programme aus dem Äther

Hans Gras, Holland hat schon einige Erfahrung mit Programmen, die verschiedene Radiosender über den Äther strahlen. Mit den zugehörigen Computer-Clubs hat er sich auch vertrauter gemacht.
Wer sich für dieses Thema interessiert möge sich bitte an Hans Gras wenden.

Hier das Protokoll einer Sitzung an der WDR-Compuerclub-Mailbox:

```
ATD 09,49221371076
CONNECT

Der  W D R  -  COMPUTERCLUB begruesst Sie mit einem
SIEMENS 9753 Transdata-Computer / 2 am Freitag, 20.02.1987 um 22:25:03

Bitte Namen eingeben: HANS GRAS

Sie sind der  97871. Anrufer !

Bedienungsanleitung mit '?'

Neu: Download mit XModem-Protokoll !!!
Naeheres im Redaktionsbriefkasten und unter Info's

An alle Mitblieder des WDR-Computerclubs und die die es gerne werden moechten:
Bitte AKTUELLES unter INFORMATIONEN oder COMPUTERCLUB-INFORMATIONEN im
Submenue COMPUTERCLUB lesen !!!

Da sich die gesammte Redaktionsmannschaft zur Hobbytronic in Dortmund aufhaelt,
wird diese Woche kei Update der Mailbox durchgefuehrt !!!

==>> Aktualisiert am: 16.02.1987

*** H A U P T M E N U ***

1   Informationen
2   Programm - Boerse
3   Briefkasten
4   Computerclub
5   Bild-Uebertragung
6   Musik-Uebertragung

( 844) Auswahl: 4

     Willkommen beim

*** C O M P U T E R C L U B ***

1   Club - Informationen
2   Basicode- und Videodat-Zeitung
3   BASICODE II - Informationen
4   BASICODE II Progr. DOWNLOAD
5   BASICODE II Progr.   UPLOAD
6   CAC - Mailboxnummern - Menu
7   Kopfnuesse

( 1135) Auswahl: 2
```

F U N K: Programme aus dem Äther

Bitte gewuenschte Ausgabe (z.B.: 1/86) eingeben :12/86

WDR-COMPUTERCLUB

7. Dezember 1986

Anschrift der Redaktion :
WDR-Computerclub
Postfach
5000 Koeln 100

Videodat-Inhaltsverzeichnis:
- Kryptologie
- Preisausschreiben
- Speedy Gonzales
- Beschreibung BASICODE-2 Programm "Sonne"
- Programm "Sonne" im BASICODE-2 Format
- Kalender 1987

Hinweise :

Die naechsten Sendetermine :

4.1.1987	"The best of Computerclub"
1.2.1987	17.30 bis 18.00 Uhr
1.3.1987	17.30 bis 18.00 Uhr
5.4.1987	17.30 bis 18.00 Uhr
3.5.1987	17.30 bis 18.00 Uhr
31.5.1987	17.30 bis 18.00 Uhr

Bei allen Sendungen werden VIDEODAT-Daten gesendet !

Kryptologie - Verschluesselungstaktik

Sie wissen ja, dort wo drei Menschen zusammenstehen, haben zwei ein Geheimnis vor dem dritten. Die Kryptologie ist die Wissenschaft der Geheimsprachen.

Das Geheimnisvolle, das den Chiffrier-methoden anhaengt, ist berechtigt. Es soll ja etwas geheim bleiben. Das nicht nur Nachrichtendienste und aehnlich merkwuerdige, aber leider notwendige Organisationen Informationen vor dem Zugriff Unberechtigter schuetzen muessen, liegt heute auf der Hand.

Die modernen Kommunikationsmethoden sind naemlichrecht anfaellig gegen gezieltes Mithoeren oder Mitlesen. Zum Beispiel lassen sich BTX-Nachrichten mit nur wenig Fachkenntnissen so abhoeren, dass man hoechstens zufaellig entdeckt wird.

Ein Brief kann nur mit viel Aufwand geoeffnet werden, ohne d

§§ Aborted §§

OK
ATH
OK

L e s e r b r i e f: Eugen Kaschubinsky

Haarlem, 28 september 1987

Geachte Memotech gebruiker,

wellicht kunt U zich nog voor de geest halen dat de HCC een oproep heeft gedaan aan alle Memotech gebruikers. Dit werd door het gebrek aan aanmeldingen c.q. interessenten geen succes. Het enige wat de HCC nog heeft gedaan, is aan alle inzenders een lijst sturen met de namen en adressen van de opgegeven gebruikers. Het behoeft niet te worden gezegd dat het aantal gebruikers in Nederland op zijn zachtst gezegd nihil is. Toch is er nog een lichtpunt! Als enthousiast Memotech gebruiker doe ik aan U de oproep om de Memotech stand (nr 5070) te bezoeken op de HCC beurs van 20 en 21 november 1987!

Ik heb op eigen houtje een stand gehuurd en zal met veel programmatuur en informatie trachten nog enig licht te werpen op onze hopeloze situatie. Ook heeft U de mogelijkheid om tegen dumpprijzen Memotech hardware aan te schaffen. Dit laatste is mogelijk gemaakt door de firma van WSUM de importeur van Memotech tegen wil en dank.

Een klein overzicht van de hardware

Memotech RS128	*HFL 450,-*
Memotech MTX500	*HFL 300,-*
SDX Disc system	*HFL 600,-!!*
80 kolomskaart voor MTX512	*HFL 350,- inc RS232!*
Monitor (Zenith amber)	*HFL 250,-*
Disc drives (QUME 500Kb)	*HFL 275,-*
NODE Roms (Local Netwerk)	*HFL 50,-!!*
ROM packs (eigenbouw)	*HFL 50,-*

Memotech User Manual

Dit boek bevat alle nuttige informatie met gratis demo programma's. Hierin vindt U alle technische informatie die in de Engelse handleiding niet voorkomt en veel veel meer. (Eigen uitgave, Engelstalig) HFL 49,-

Software

Communicatieprogramma's voor CP/M (ook VIDITEL)
Expanded BASIC (eigen programma) op ROM!!
Communicatieprogramnmma voor MTX op ROM!!
Language Learning Lab educatief programma (Mbasic)

U ziet, de Memotech is niet dood, alleen wat vergeten! Voor meer informatie en eventuele voorbestellingen kunt U contact opnemen met:

E.Kaschubinsky
A.C.Krusemanstraat 105
2032 HG Haarlem
Tel: 023-353299 (na 18:00 uur)

B A S I C: Digitaluhr

Aus Holland in MBASIC (Herbert zur Nedden, 2000)

Ich war so frei, den Source etwas umzuformatieren!

Schleifen habe ich z.T. eingerückt - BASIC macht das nicht!
Wenn eine Zeile keine Zeilennummer hat, so soll diese **direkt** hinter die vorherige ohne die Leerzeichen am Anfang der Zeile ohne Nummer gesetzt werden!

Die Zuweisung N=N AND 3 bedeutet: Nimm nur Bit 1 und 2 von N,
N=N AND 15 bedeutet: Nimm nur Bit 1 bis 3 von N.

```
1000 REM DIGITALE KLOK
1010 PRINT CHR$(12)+CHR$(27)+"XX"+CHR$(20)+CHR$(31)+CHR$(12)
1020 DEFINT A-Z: DIM A$(6),B$(9,6),CR$(7),M(9):
1030 E$=" "

1040 FOR I=1 TO 6: READ A$(I): NEXT I

1050 FOR I=0 TO 7
1060     READ R$:S$=""
1070     FOR J=1 TO LEN(R$)
1080         IF MID$(R$,J,1)="X" THEN S$=S$+CHR$(127) ELSE S$=S$+" "
1090     NEXT J
1100     CR$(I)=S$
1110 NEXT I

1120 FOR I=1 TO 9: READ M(I): NEXT I

1200 PRINT CHR$(12)+"        D I G I T A L E   K L O K"+CHR$(31)

1210 FOR J=1 TO 6
1220     A=6-J:   OUT(7),A
1230     A=A+128: OUT(7),A
1240     A=A-128: OUT(7),A
1250     A=A+16:  OUT(7),A
1260     A=A+64:  OUT(7),A
1270     N=INP(7): N=INP(7)
1280     IF J=1 THEN N=N AND 3 ELSE N=N AND 15
1290     OUT(7),N: OUT(7),16
1300     IF N<>0 OR J<>1 THEN
            FOR I=1 TO 5: B$(I,J)=CR$(VAL(MID$(A$(I),N+1,1))): NEXT I
1310 NEXT J
1320 PRINT CHR$(3)+CHR$(32)+CHR$(36);

1330 FOR I=1 TO 9
1340     J=M(I): K=I: IF I>5 THEN K=6
1350     PRINT E$+B$(J,1)+E$+B$(J,2)+E$+E$+E$+
              B$(J,3)+E$+B$(J,4)+E$+E$+E$+B$(K,5)+E$+B$(K,6)
1360 NEXT I
1370 IF INKEY$="" THEN 1210 ELSE PRINT CHR$(12)+CHR$(30): END

1400 DATA "7277577777","5211544155","5277777377","5241115151",
         "7277177177","0000000000"
1410 DATA "     ","    X","  X  ","   XX","X    ","X   X","x","XXXXX"
1420 DATA 1,2,2,2,3,4,4,4,5
```

Sorry, aber diese Seite musste ich entfernen, weil ich für eine Veröffentlichung außer in der ehemaligen, gedruckten Form keine Freigabe habe.

Sorry, aber diese Seite musste ich entfernen, weil ich für eine Veröffentlichung außer in der ehemaligen, gedruckten Form keine Freigabe habe.

Sorry, aber diese Seite musste ich entfernen, weil ich für eine Veröffentlichung außer in der ehemaligen, gedruckten Form keine Freigabe habe.

Sorry, aber diese Seite musste ich entfernen, weil ich für eine Veröffentlichung außer in der ehemaligen, gedruckten Form keine Freigabe habe.

Software-Coldboot von B: (Jan Bredereke, 2000)

Hans Gras fragte im Info 23-4, wie man aus dem ROM einen Coldboot machen kann, ohne die Meldung "Ramtest OK" zu bekommen. Da ich einige Tage später ein ähnliches Problem hatte, beschreibe ich hier meine Lösung dazu.
Im Info 17-49 wurden die ROM-IDs beschrieben, im Info 12-17 ist ein kurzes Programm angegeben, das einen völligen Coldboot macht. **(Vielen Dank für das Gesamt-Inhalts-verzeichnis!)** Also habe ich die ROMs ausgelesen und mit REZILOG disassembliert. Hier nun ein kleines Programm, das von B: einen Coldboot auf eine Nicht-RAMxx-Diskette macht, z.B. auf eine FDXB-Diskette:

```
        DI                  ;Interrupt aus
        LD HL,START         ;Programm ins Top-RAM kopieren, das beim
        LD DE,#C000         ;bank-switching nicht umgeschaltet wird.
        LD BC,#100          ;Länge ist mehr als genug.
        LDIR
        JP #C000            ;... und dort weitermachen.
START:  LD A,#40            ;ROMs einschalten, Bank 4 (DISC-ROM).
        OUT (0),A
-------------------------------------------------------------------
        LD HL,#2100         ;ROM ins Top-RAM kopieren, so wie es auch
        LD DE,#EA00         ;im ROM ab Adresse #2010 steht.
        LD BC,#1600
        LDIR
        LD A,#C3            ;die Kopie patchen:
        LD (#EACC),A        ;JP ...
        LD HL,#EBE8         ;... #EBE8 (keine Meldung ausgeben, nicht
        LD (#EACD),HL       ; von I:, sondern gleich von B: booten.)
        JP #EA62            ;und in die Kopie springen.
```

Dieses Programm läuft so nur mit dem neuen CP/M-EPROM, das der Reihenfolge nach von den SRAM-Disks H:, I: und dann von der Floppy B: zu booten versucht, wenn man es läßt.
Hat man noch das Original-(E)PROM, so muß man vermutlich den Teil unter dem Strich durch

```
        JP #2010
```

ersetzen, womit der Originaleinsprungpunkt benutzt wird. Mit dem neuen EPROM funktioniert das nicht, genauso wie DISC QUIT unter Basic zum Absturz führt.
Mit diesem Programm wird das Basic-ROM A nicht mehr angesprungen, das unter anderem auch prüft, ob etwa ein RING-ROM o.ä. da ist. (Hat das überhaupt irgendjemand?) Damit müßte auch der Ramfloppy-zerstörende Ramtest umgangen werden. Aber ich habe die Basic-ROMs gegen die neuen vom Club ausgetauscht, womit das Thema der zerstörten Ramfloppy für mich sowieso erledigt war. Die Meldung "Ramtest OK" müßte eigentlich aus dem CP/M-EPROM gekommen sein, aber wie gesagt, auch dieses ist bei mir schon ausgetauscht.
Warum ST.COM und RAMxx.COM ein mit dem obigen Programm gebootetes System nicht mögen, weiß ich nicht. Aber ich brauche es auch nicht, da ich es nur dazu benutze, um von meinem FDXB-40-Spielemenüprogramm aus ein DISC QUIT auszuführen, um automatisch ein Spiel aufzurufen, das unter CP/M läuft.
Das Spiel wird dann durch ein **$$$**.SUB-File gestartet, das ich vorher aus BASIC erzeugt habe.
Wer zu dem Thema noch Fragen hat oder sich für das kleine ROM-Ausleseprogramm interessiert, kann sich gerne an mich wenden. Im übrigen müßte Bernd Preusing auch zu den originalen CP/M-EPROMs etwas sagen können, da er schließlich die neuen geschrieben hat.

N e w W o r d: Variables Bildschirmformat / S o r t i e r e n

Variables Schirmformat für Newword (Jan Bredereke, 2000)

Seit RAM 4.2 von Bernd Preusing sind auf dem Memotech auch andere Bildschirmformate als 80*25 unterstützt, insbesondere größere, falls der Bildschirmspeicher erweitert wurde, und daher sollte man diese neuen Möglichkeiten auch mit Newword nutzen können. Denn auch Mtx-Edit von Herbert zur Nedden kann während des Editierens das Bildschirmformat verändern, um z.B auf einem 92*50-Schirm zwischendurch schnell einen Überblick über einen Text zu bekommen.
Diesen Komfort bietet jetzt auch Newword zusammen mit NwSchirm, einem von mir geschriebenen Programm.
NwSchirm ist ein Klix-Programm und patcht das laufende Newword so, daß Newword ein anderes Bildschirmformat einstellt und damit weiterarbeitet.

Aber:

Es funktioniert leider noch nicht richtig. Um es hinzukriegen, brauche ich noch einen Tip von jemandem, der Newword sehr gut von innen kennt.
Bisher patcht NwSchirm das laufende Newword 2.02 an vier Stellen:

HITE:	02E6H	(Zeichen pro Zeile)
WID:	02E7H	(Zeilenzahl)
ERASCR:	032AH	(Schirmlöschstring)

Damit während der Laufzeit von Newword auch die Spalte umgestellt wird, in der "<^.1?+" usw. am rechten Rand angezeigt werden, muß in die Speicherstelle 18B8H der Wert von HITE minus 1 gepatcht werden.
In den Schirmlöschstring wird
ESC Ä F <Spalten> <Zeilen>
gepatcht, so daß bei jedem Schirmlöschen, also auch nach Drücken der ESC-Taste, das Schirmformat aktualisiert wird. NwSchirm simuliert deshalb nach seiner Beendigung ein ESC für Newword.
Die Patches kann man ebenfalls, wenn auch viel weniger komfortabel, mit dem Klix-Monitor machen.
Dies funktioniert alles hervorragend, solange die Zahl der Zeilen auf dem Bildschirm nicht größer gepatcht wird, als sie in Newword installiert war. Wenn ich sie größer mache, wird beim seitenweisen Umblättern nicht mehr der ganze Inhalt des Schirmes gelöscht, sondern im unteren Teil bleiben noch einige alte Zeilen stehen, wenn sie nicht von neuen Zeilen überschrieben werden.
Mit REZILOG konnte ich dieses Problem nicht lösen, da fast das ganze Newword aus Overlays besteht. Durch Differenzbildung bin ich ja noch auf die Adresse 18B8H für den rechten Rand gekommen, aber bei der Zeilenzahl bin ich ratlos. Wenn mir jemand einen Tip geben kann, wie das Problem zu lösen ist, werde ich die korrekte Version dem Club als PD zur Verfügung stellen.

Zur Frage von Wolfgang Dexheimer in Info 23 nach Sortieralgorithmen
(Jan Bredereke, 2000)

Es ist wahr, daß viele Sortierprobleme mit völlig ungeeigneten Mitteln gelöst werden. Z.B. hat Bubblesort zwar einen schönen Namen, ist aber ansonsten praktisch immer der ineffizienteste Algorithmus von allen. Trotzdem wird es immer wieder verwendet.
Leider läßt sich das Thema nicht erschöpfend auf der Länge eines Artikels im Info abhandeln, da es sehr von der jeweiligen Aufgabe abhängt, welcher Algorithmus der beste ist. Trotzdem bringe ich hier einen kleinen Überblick über die Bewertung von Sortieralgorithmen.
Dabei richte ich mich grob nach dem 2. Kapitel von "Algorithmen und Datenstrukturen" von Niklaus Wirth, dem Standardwerk für alle Informatikstudenten. In Kapitel 2 werden auf 72 Seiten die Algorithmen gut verständlich erklärt und bewertet.
Dort kann man auch die Algorithmen im einzelnen nachlesen. Dieses Buch kann man sicherlich auch in größeren Bibliotheken ausleihen. Wirth ist übrigens der Erfinder von Pascal, und die Beispiele sind folglich auch in Pascal geschrieben. Es gibt auch eine neuere Ausgabe mit Modula-2, das eine Fortentwicklung Wirths von Pascal ist.

S o r t i e r e n

Nun aber zu den Bewertungen.
Ich beschränke mich hier nur auf das Sortieren von Arrays. Bei dem Sortieren von Files ist alles wieder ganz anders.
Bei dem Sortieren von Arrays sind folgende Faktoren wesentlich:
- der Aufwand für das Vergleichen der Schlüssel
- der Aufwand für das Umkopieren der Datenelemente

Man unterteilt die Sortierverfahren in die direkten und in die höheren Verfahren.

Direkte Verfahren
Sortieren durch direktes Einfügen (straight insertion)
Es zeigt natürliches Verhalten: Den größten Aufwand gibt es bei umgekehrt sortierem Array.
Das Verfahren ist stabil: Elemente mit gleichem Schlüssel bleiben in der gleichen Reihenfolge.
Bei fast oder ganz vorsortieren Arrays ist es ein wenig besser als das direkte Auswählen.
Modifikation: Direktes Einfügen mit binärem Suchen
Zeigt unnatürliches Verhalten durch die Binärsuche, ist also bei vorsortierten Arrays etwas schlechter.
Braucht im Normalfall zwar weniger Vergleiche, aber die gleiche Zahl von Kopiervorgängen, bringt also nicht viel mehr.
Sortieren durch direktes Auswählen (straight selection)
Ist im allgemeinen besser als direktes Einfügen.
Sortieren durch direktes Austauschen
Das Grundverfahren ist Bubblesort, Shakersort ist eine Weiterentwicklung.
Beide Verfahren sind schlecht.
Shakersort geht dann, wenn man sich ganz sicher ist, daß das Feld vorher schon fast sortiert ist.

Höhere Verfahren
Sortieren durch Einfügen mit abnehmender Schrittweite (shellsort)
Es ist eine Verfeinerung des direkten Einfügens.
Die Theoretiker haben die beste Folge von Schrittweiten noch nicht herausbekommen.
Gut ist z.B. 1,4,13,40,121,...
Mit der Folge 1,3,7,15,31,... ist es von der Ordnung O(n^1.2), d.h. der Aufwand ist proportional zur 1.2-ten Potenz der Feldgröße.
Sortieren mit Bäumen
Wichtigstes Verfahren: Heapsort
Es ist bei wenigen Elementen schlecht, bei sehr vielen aber sogar besser als Shellsort.
Im schlimmsten Fall: O(n*log(n)).
Es zeigt unnatürliches Verhalten: Es ist am besten, wenn das Feld vorher fast umgekehrt sortiert war.
Sortieren durch Zerlegen (Quicksort)
Es beruht wie Bubblesort auf dem Prinzip des Austauschens, aber jetzt über große Distanzen.
Es ist bei kleiner Elementezahl relativ schwach.
Bei großen Zahlen ist es im Durchschnitt die beste bekannte Methode, kann aber in unglücklichen Ausnahmefällen extrem schlecht sein.
Es braucht im Durchschnitt log(n) Durchläufe, n*log(n) Vergleiche und n*log(n)/6 Austauschoperationen.
Es macht im schlimmsten Fall n Zerlegungen, was den Stack unter Umständen zum Überlaufen bringen kann, und braucht dabei den Aufwand O(n^2).
Man kann es so verbessern, daß die Stack-Länge maximal log(n) wird.
Ein Vorteil von Quicksort ist, daß sich kleine Teilzerlegungen einfach mit direkten Sortiermethoden sortieren lassen, denn diese Verfahren sind einfach in Quicksort einzubauen.

Allgemein sind die direkten Methoden von der Ordnung O(n^2) und die höheren von der Ordnung O(n*log(n)), also sind die höheren besser. Allerdings sind sie oft nur bei großen n gut, weil sie immer einen gewissen Grundaufwand treiben, der sich bei kleinen Arrays nicht auszahlt.
Wenn die zu bewegenden Datensätze groß sind, erzielt die direkte Auswahl noch bessere Ergebnisse unter den direkten Methoden, Bubblesort und Shakersort sind immer noch hinten und Quicksort wird von allen noch besser.
Im übrigen brauchen höhere, also kompliziertere Verfahren mehr Speicherplatz für den Programmcode.

Fazit:
Quicksort ist bei weitem die beste bekannte Methode für Arrays.
Die direkte Auswahl ist im allgemeinen die beste der direkten Methoden, und damit als einfaches Verfahren für kleine Datenmengen zu empfehlen. Es lohnt sich z.B. kaum, sich die Mühe zu machen, Quicksort zu implementieren, nur weil man ein Directory sortieren will. Da spart die direkte Auswahl Programmierzeit und Programmcode.
Aber wie schon gesagt, je nach den Umständen kann jeweils ein anderes Verfahren das beste sein.

Und noch eine Nachbemerkung: Es ist sehr aufwendig, größere Datensätze wie längere Strings usw. beim Sortieren umzukopieren, besser ist es, nur ein Feld von Zeigern auf diese Datensätze zu sortieren.

Zur Frage von Wolfgang Dexheimer in Info 23 nach LISP auf dem MTX
(Jan Bredereke, 2000)

Auf der PD-Diskette SIG/M 118 ist ein XLISP-Interpreter. XLISP ist eine objekt-orientierte Erweiterung von LISP. Das Programm läuft auch und läßt sich damit für Lernzwecke nutzen, auch wenn man zusätzlich ein Lehrbuch braucht, falls man noch nichts über LISP oder objektorientierte Programmierung weiß.
Allerdings ist der Interpreter konkurrenzlos langsam, und er braucht fast den ganzen Arbeitsspeicher für sich selbst. Ich habe einmal zwei Schildkröten programmiert, die mit jeweils einem Joystick steuerbar waren. (Generische Funktionen sind schon eine feine Sache.) Allerdings war damit der Speicher schon fast voll. Und die Schildkrötensymbole krochen wirklich über den Schirm, wobei sie jeweils nach drei Schritten noch eine Sekunde Pause einlegten, weil offenbar der Garbage-Collector wieder aufräumen mußte.

A s s e m b l e r : Kurs

ASSEMBLERKURS

Lösungen zu den Aufgaben im letzten Info:

Lösung zu 1.3.4-1 (Info 22-22):
Das Problem ist natürlich darin begründet, daß die I2DITVM nur bis 99 rechnen kann. Du erinnerst dich sicher noch an den Zahlenkreis. Wenn wir nun erst die Multiplikation durchführen, kann es sein, daß wir diese Grenze überschreiten (das tritt ab n=10 auf), obwohl das Endergebnis noch unter 100 liegt. Es kommt also darauf an, die Reihenfolge der Operationen so geschickt zu wählen, daß wir bis zu möglichst hohen Werten für n zuverlässig rechnen können. Dieses Problem tritt bei der Assemblerprogrammierung des öfteren auf, ebenso in TURBO-PASCAL bei Verwendung von INTEGER Variablen, in BASIC dagegen kaum, weil dort die Zahlen intern geschickter gespeichert werden. Die Lösung liegt natürlich darin, die Division vorzuziehen, damit es gar nicht erst zu so großen Zahlen kommt. Damit dadurch keine Rechenungenauigkeiten entstehen, sollten wir die gerade Zahl durch 2 dividieren. Da wir nicht im voraus wissen, ob n in Zelle 0 nun gerade oder ungerade ist, gebe ich zwei Varianten an:

```
n gerade:                    n ungerade:
LOAD (0)  hole »             LOAD (0)  hole »
DIV  02                      INC  A    ist n+1
INC  (0)  ist n+1            DIV  02
MUL  (0)                     MUL  (0)
EXIT                         EXIT
```

Lösung zu 1.3.4-2 (Info 22-22):
Mittelwert aus (1) und (2) bilden:

```
LOAD (0)
ADD  (1)
DIV  02
STO  (2)
EXIT
```

Auch hier kann man wieder Varianten bilden, die die Division so bald wie möglich durchführen. Dazu kann es u. U. vorteilhaft sein, zuerst die Zahl aus (1) zu laden, wenn diese gerade ist. Allerdings ist bei dieser Aufgabe nicht garantiert, daß eine der beiden Zahlen gerade ist.

Lösung zu 1.3.4-3 (Info 22-22):
Berechnung der Formel (2*4 + 4*6)/(3+13):

```
LOAD 02
MUL  04
STO  (0)  erstes Teilergebnis merken
LOAD 04
MUL  06
ADD  (0)  erstes Teilergebnis dazuaddieren = Zaehler
STO  (0)  wieder wegspeichern
LOAD 03
ADD  13   =Nenner
XCHG (0)  Akku mit Speicherzelle 0 vertauschen
DIV  (0)  Ergebnis im Akku
EXIT
```

A s s e m b l e r : Kurs

Lösung zu 2.1-1 (Info 22-28):
Umwandlung einer dezimalen Zahl in die binäre Darstellung:

```
100 REM Umwandlung dezimal --> binär
110 REM Lösung zu Aufgabe 2.1-1
120 INPUT "Gib dezimale Zahl ein -->";X
130 LET BIN$=""
140 LET REST=MOD(X,2)
150 LET BIN$=CHR$(REST+ASC("0"))+BIN$
160 LET X=(X-REST)/2
170 IF X>0 THEN GOTO 140
180 REM Zur Schönheit auf ganze Bytes auffüllen:
190 IF MOD(LEN(BIN$),8)=0 THEN GOTO 220
200 LET BIN$="0"+BIN$
210 GOTO 190
220 PRINT "Binärzahl ist ";BIN$
230 STOP
```

Lösung zu 2.1-1 (Info 22-29):
Umwandlung dezimaler Zahlen in die hexadezimale Darstellung:

```
100 REM Umwandlung dezimal --> hex
110 REM Lösung zu Aufgabe 2.2-1
120 INPUT "Gib dezimale Zahl ein -->";X
130 LET HEX$=""
140 LET REST=MOD(X,16)
150 IF REST<10 THEN LET HEX$=CHR$(REST+ASC("0"))+HEX$
       ELSE LET HEX$=CHR$(REST-10+ASC("A"))+HEX$
160 LET X=(X-REST)/16
170 IF X>0 THEN GOTO 140
180 REM Zur Schönheit auf ganze Bytes auffüllen:
190 IF MOD(LEN(HEX$),2)=0 THEN GOTO 220
200 LET HEX$="0"+HEX$
210 GOTO 190
220 PRINT "Hexzahl ist ";HEX$
230 STOP
```

Lösung zu 2.3-1 (Info 22-32):
Zeichen Ein- und Ausgabe:

```
10 REM
20 REM Übungsaufgabe 2.3-1
30 REM
100 LET A$=INKEY$: IF A$="" THEN GOTO 100
110 IF A$<" " THEN PRINT "^";CHR$(ASC(A$)+ASC("$")) ELSE PRINT A$
120 IF A$=CHR$(3) THEN PRINT : PRINT : PRINT : STOP
130 IF INKEY$<>"" THEN GOTO 130 ELSE GOTO 100
```

Lösung zu 2.3-2 (Info 22-32):
Klassifizierung von Zeichen:

```
10 REM
20 REM Übungsaufgabe 2.3-2
30 REM
100 LET A$=INKEY$: IF A$="" THEN GOTO 100
110 IF A$<" " THEN PRINT "Kontrollzeichen": GOTO 200
120 IF A$>="0" AND A$<="9" THEN PRINT "Ziffer": GOTO 200
130 IF A$>="A" AND A$<="Ü" THEN PRINT "Großbuchstabe": GOTO 200
140 IF A$>="a" AND A$<="ü" THEN PRINT "Kleinbuchstabe": GOTO 200
150 PRINT "Sonderzeichen"
200 IF A$=CHR$(3) THEN PRINT : PRINT : PRINT : STOP
```

<u>A s s e m b l e r : Kurs</u>

```
210 IF INKEY$<>"" THEN GOTO 210 ELSE GOTO 100
```

Lösung zu 2.3-3 (Info 22-34):
Zeichen in Hexcode ausgeben:

```
10 REM
20 REM Übungsaufgabe 2.3-3
30 REM
100 LET A$=INKEY$: IF A$="" THEN GOTO 100
110 LET B=ASC(A$)
120 GOSUB 1000
130 PRINT
140 IF A$=CHR$(3) THEN PRINT : PRINT : PRINT : STOP
150 IF INKEY$<>"" THEN GOTO 140 ELSE GOTO 100
1000 REM
1010 REM Ein Byte hexadezimal ausgeben
1020 REM Übergabe des Werts in Variable B
1030 REM
1040 LET N=INT(B/16)
1050 GOSUB 2000
1060 LET N=B-16*N
2000 REM
2010 REM Ein Nibble hexadezimal ausgeben
2020 REM Übergabe des Werts in Variable N
2030 REM
2040 IF N<10 THEN PRINT CHR$(N+ASC("0"));
     ELSE PRINT CHR$(N-10+ASC("A"));
2050 RETURN
```

Herbert hat hier einen Trick verwendet, der eigentlich kein guter Programmierstil ist, aber wegen seiner Effektivität gerade in der Assemblerprogrammierung häufig benutzt wird. Daher soll er hier auch erläutert werden, wobei es dir überlassen bleibt, ob du ihn später anwendest oder nicht (Stil ist schließlich jedem seine eigene Sache (deutsche Sprache auch)).

Worin besteht nun der Trick? Das Unterprogramm (kurz UP) 1000 ruft in Zeile 1050 ein weiteres UP, nämlich 2000, auf. Aber wo ist das UP 1000 eigentlich zu Ende? Wie du siehst, folgt nach Zeile 1060, der letzten Zeile von UP 1000, gleich die Zeile 2000, so daß diese und die folgenden Zeilen bis zum RETURN ja eigentlich noch zum UP 1000 zu zählen sind. Der Teil von 2000 bis 2050 wird also zweifach genutzt und auch zweimal durchlaufen: einmal als UP durch den GOSUB in Zeile 1050 und ein zweites Mal als Teil von UP 1000 selbst. Diese Technik nennt man "fall throught", da das UP 1000 gewissermaßen in das folgende UP hineinfällt. Sauberer wäre die Lösung

```
1050 GOSUB 2000
1060 LET N=B-16*N
1070 GOSUB 2000
1080 RETURN
```

aber das ist halt länger und damit auch langsamer. Übrigens hätte man in Zeile 2040 statt ASC("0") auch gleich 48 schreiben können. Ersteres entspricht mehr der Hochsprachendenkweise, letzteres mehr der Assemblerdenkweise. Auch wieder eine Frage des Stils!

Lösung zu 2.3-4 (Info 22-34):
Umkehrung der vorigen Aufgabe

```
10 REM
20 REM Übungsaufgabe 2.3-4
```

A s s e m b l e r : Kurs

```
30 REM
100 INPUT "Hexzahl eingeben: ";A$
110 IF LEN(A$)=0 OR LEN(A$)>2 THEN GOTO 500
120 LET A=0
130 FOR I=1 TO LEN(A$)
140 LET B$=A$(I)
150 GOSUB 1000
160 IF N=16 THEN GOTO 500
170 LET A=16*A+N
180 NEXT I
190 IF A<32 THEN PRINT "^";CHR$(A+64) ELSE PRINT CHR$(A)
200 GOTO 100
500 REM
510 REM Fehlerroutine
520 REM
530 PRINT "Eingabe fehlerhaft!"
540 GOTO 100
1000 REM
1010 REM Ein Nibble umwandeln
1020 REM Übergabe in B$, Rückgabe in N (16 = Fehler)
1030 REM
1040 IF B$>="a" THEN LET B$=CHR$(ASC(B$)-32)
1050 IF B$<"0" OR B$>"F" THEN LET N=16: RETURN
1060 IF B$>"9" AND B$<"A" THEN LET N=16: RETURN
1070 LET N=ASC(B$)-48
1080 IF N>10 THEN LET N=N-7
1090 RETURN
```

Jetzt geht es weiter im normalen Text:

3 Die Hardware aus der Sicht des Programmierers

Es ist wohl bei jeder Programmiersprache so (BASIC vielleicht mal ausgenommen), daß man zu Anfang erstmal über einen Berg von trockener Theorie hinweg muß, ehe die Sache beginnt, Spaß zu machen. Dieser Berg ist bei Assembler noch etwas höher als bei anderen (höheren!) Programmiersprachen, da man sich nicht nur mit der Sprache (Assembler), sondern auch noch ein bißchen mit der Maschine auskennen muß, auf der man seine Programme später laufen lassen will. Daher reiht man Assembler auch in die "maschinennahen" Sprachen ein. Wir kommen deshalb nicht umhin, nach den etwas abstrakten Ausführungen des letzten Kapitels noch ein relativ trockenes Kapitel über die verwendete Hardware folgen zu lassen. Wenn ihr dieses Kapitel hinter euch habt, seid ihr aber über den (anfangs erwähnten) Berg.

3.1 Die Z80-CPU (MiMö aus Hamburg)

Was eine CPU ist und kann ist durch die I2DITVM (hoffentlich) schon hinreichend geklärt. Der nächste Schritt soll nun sein, einen konkreten Mikroprozessor -die Z80 CPU- näher zu erläutern. Links ist das Programmiermodell (im folgenden PM genannt) der Z80 zu sehen, das für den Programmierer wichtigste 'Abbild' der CPU. Aus dem PM einer CPU kann der Programmierer ersehen, welche Register ihm zur Verfügung stehen. Register kann man sich vorstellen, wie Speicherstellen, die nicht im RAM, sondern in der CPU selber liegen und dadurch auch schneller 'ansprechbar' sind.

Die Buchstaben in den Kästchen sind die 'Namen' der Register der Z80-CPU, unter denen sie 'ansprechbar' sind. Die kleinen Kästchen kennzeichnen 8-Bit-Register, die großen Kästchen kennzeichnen 16-Bit-Register (jaaa, so etwas hat unser Mikroprofessor auch).

Assembler: Kurs

```
+-----+-----+
!  A  !  F  !
+-----+-----+
!  B  !  C  !
+-----+-----+
!  D  !  E  !
+-----+-----+
!  H  !  L  !
+-----+-----+
!    P C    !
+-----+-----+
!    S P    !
+-----+-----+
!    I X    !
+-----+-----+
!    I Y    !
+-----+-----+
!  I  !  R  !
+-----+-----+
```

Wir werden nur einige der Register etwas näher betrachten, da nicht alle die gleiche Wichtigkeit für das Funktionieren der CPU besitzen.

Als erstes sehen wir uns das Register mit dem Namen 'PC' an. PC ist die Abkuerzung Program-Counter (zu deutsch: Befehlszähler). Dieses Register sorgt dafür, dass das Programm (in Maschinensprache) in der richtigen Reihenfolge abgearbeitet wird.

Ein Programm in Maschinensprache wird, ähnlich wie ein BASIC-Programm von niedrigeren Speicherstellen (in BASIC: Zeilennummer) zu höheren Speicherstellen abgearbeitet, und damit die CPU dabei nicht in's Schleudern gerät, merkt sie sich mit dem PC, an welcher Speicherstelle der nächste Befehl steht. Das heisst, wenn -zum Beispiel nach einem Reset- der PC auf die Speicherstelle 0000H zeigt, dann wird der Befehl von dieser Speicherstelle gelesen und der PC zeigt auf die nächste Speicherstelle, an der ein Befehl steht. Das PC-Register ist fuer den Programmierer nur bedingt 'sichtbar', es wird nur durch Programm-Sprünge oder durch Unterprogrammaufrufe vom Programmierer beeinflusst. Normalerweise sorgt die CPU selber dafür, dass der PC auf den nächsten Befehl, der abzuarbeiten ist, zeigt.

Das nächste wichtige Register ist das 'A'-Register, 'A' steht für Akkumulator, was soviel wie Sammler heisst und in diesem Falle bedeutet, dass alle Zwischenergebnisse -aus Berechnungen- in diesem Register gesammelt werden; hierauf wurde ja schon in Kapitel 2 eingegangen. Das A-Register ist insofern eine Besonderheit gegenueber anderen Registern, als man NUR mit dem Akku rechnen kann.

Was passiert nun aber, wenn wir die Zahlen 255 und 7 addieren -man beachte, dass der Akku 8 Bit breit ist ? Logisch, Ihr habt es gleich erkannt, das Ergebnis ist 6 (!?!)... hmpf, da ist was falsch, es ist ein Ueberlauf entstanden, mit 8 Bit kann man die 262 nicht mehr darstellen. Um dem Programmierer nun mitzuteilen, dass dieses Malheur passiert ist, gibt es das F-Register, dessen einzelne Bits bestimmte Bedeutungen haben und in Abhängigkeit vom letzten Rechenergebnis gesetzt werden. Solch ein Bit wird im englischen als Flag (wörtlich: Flagge, gemeint ist damit ein Anzeiger) bezeichnet, woher auch das 'F' kommt. In unserem Fall bedeutet das, dass das Ueberlaufbit gesetzt ist vom Programmierer abgefragt werden kann. Auf diese Art und Weise kann der Programmierer feststellen, ob alles mit rechten Dingen zu-

ging. Das F-Register wird von der CPU selbstständig verwaltet. Der Programmierer kann nicht beliebige Werte in dieses Register hineinschreiben (außer durch Tricks). Die Bedeutung der übrigen 7 Bits im F-Register werden im weiteren Verlauf des Kurses noch erläutert. Dieses soll als Einleitung zum F-Register genuegen.

Die Register B, C, D, E, H und L sind Allzweckregister, in denen man beliebige 8-Bit-Werte zwischenspeichern kann. Da Register aus der Sicht des Programmierers so etwas wie 'schnelle Speicher' sind, kann man hier Werte ablegen, die öfters benötigt werden. Wer genau aufgepasst hat, dem ist aufgefallen, dass im PM die Register B und C nebeneinander angeordnet sind (wie auch D, E und H, L). Damit soll angedeutet werden, dass diese Register zu 'Paaren' zusammengeordnet werden können und dann als 16-Bit Registerpaar fungieren. Es kann aber nur das Registerpaar BC angesprochen werden, ein Registerpaar CB gibt es nicht. Das heisst, das B-Register hält das höherwertige, das C-Register das niederwertige Byte. Das gleiche gilt auch für die Registerpaare DE und HL. In diesem Zusammenhang muss natürlich noch gesagt werden, dass das HL-Registerpaar fuer 16-Bit-Werte als 'quasi-AKKU' angesehen werden kann, da man mit diesem Registerpaar -beschränkt- arithmetische Operationen durchführen kann. Die Register A bis L sind übrigens doppelt vorhanden (A', F' ... L'), wobei man aber zu einem Zeitpunkt immer nur einen Registersatz zur Verfügung hat. Es gibt jedoch Befehle, um auf die gestrichenen Register (Zweitregister, manchmal auch als Schattenregister bezeichnet) umzuschalten.

Nun kommt etwas bekanntes, der Stapelzeiger (engl. Stackpointer), das SP-Register. Die Funktion des SP-Registers ist ja im letzten Teil des Kurses schon erläutert worden, d.h.: man kann mit einem PUSH-Befehl einen Wert auf den Stack legen und mit einem POP-Befehl einen Wert vom Stack holen. Das SP-Register hat eine wichtige Funktion beim Aufruf von Unterprogrammen (in BASIC wäre das ein GOSUB <Zeilennummer>); ein (Assembler-)Unterprogramm wird durch einen CALL-Befehl aufgerufen, z.B.: CALL 100H. Der Stackpointer wird sowohl vom Programmierer als auch von der CPU verändert. Der Programmierer kann den Stackpointer mit einem beliebigen Wert laden und ihn (auf Umwegen) wieder auslesen. Bei den Befehlen PUSH, POP und CALL wird der SP jedoch automatisch von der CPU verändert.

Um es in diesem Anlauf noch nicht zu schwierig werden zu lassen, bin ich auf Interrupt-Abarbeitung, die Funktion des R-Registers (Speicher-Refresh) und die Möglichkeiten der IX- und IY-Register nicht eingegangen. Bezüglich der IX- und IY-Register möchte ich noch auf einen Beitrag von mir in Info 13-42 (relative Adressierung bezüglich des Stack-Top) verweisen.

Daß es daneben auf dem CPU Chip noch ein Rechenwerk und eine Einheit zur Befehlsdekodierung und -ausführung gibt, ist eigentlich selbstverständlich und braucht hier nur am Rande erwähnt zu werden, da der Programmierer damit nicht unmittelbar in Berührung kommt. Wichtiger sind da schon die 16 Adressleitungen und die 8 Datenleitungen, die die CPU mit dem Hauptspeicher verbinden. Denn dort steht ja schließlich das Programm, das das kleine Kerlchen bearbeiten soll. Deshalb wird im nächsten Abschnitt erklärt, wie es in unserem Hauptspeicher aussieht.

3.2. Speicher (Petra Jochem, 8012)

Dieser Abschnitt beschäftigt sich nur mit dem internen Hauptspeicher, nicht dagegen mit den Massenspeichern wie Diskettenlaufwerk etc..

Der Arbeitsspeicher setzt sich aus kleinsten Einheiten, den Speicher-

zellen, zusammen, die in Systemen mit Z80-CPU jeweils 1 Byte = 8 Bit an Information aufnehmen können. Die Speicherzellen sind in aufsteigender Folge durchnummeriert, und können über ihre Nummer, die sogenannte Adresse, angesprochen werden. Zwar sind die Speicherzellen in Wirklichkeit nicht linear angeordnet, aber der Programmierer muß sich darum nicht kümmern, weil die im Speicher integrierte Dekodierlogik der Adresse den physikalischen Ort der Information in den ROM- bzw. RAM-Bausteinen zuordnet. Der Z80 benutzt zur Adressierung 16-Bit-Worte; deshalb können maximal 65536 Speicherzellen direkt angesprochen werden.

Das in einer Speicherzelle enthaltene Bitmuster kann sowohl einen Programmbefehl als auch ein Datum (Singular v. Daten) repräsentieren. Das Register PC der CPU legt fest, in welcher Speicherzelle der nächste auszuführende Befehl enthalten ist. Der Programmierer muß also dafür sorgen, daß PC nicht auf eine Speicherzelle zeigt, deren Inhalt als Datenbyte gedacht ist.

An der Kommunikation zwischen CPU und Speicher sind 3 Busse (vom engl. bus, meint in diesem Zusammenhang Leitungsbündel) beteiligt:

- der 16 Bit breite unidirektionale Adressbus überträgt eine von der CPU erzeugte 16-bit-Adresse zum Speicher. Unidirektional heißt soviel wie "nur in einer Richtung arbeitend", d. h. die Signale auf diesem Bus (die Adressen) kommen immer von der CPU und wenden sich an den Speicher, nie umgekehrt.
- der 8 Bit breite bidirektionale Datenbus transportiert Daten von der CPU zur ausgewählten Speicherzelle und umgekehrt (daher bi!).
- der Kontrollbus übermittelt Signale zur Synchronisation der verschiedenen Bausteine. Für die Kommunikation mit dem Speicher sind die folgenden Leitungen des Kontrollbusses besonders wichtig:

  - $\overline{\text{MEMREQ}}$ (memory request, d. h. Speicherzugriff) wird aktiv (=low), wenn eine Speicheroperation (im Gegensatz zu einer I/O Operation, siehe nächster Abschnitt) ausgeführt werden soll

  - $\overline{\text{RD}}$ (read, d. h. lesen) wird aktiv, wenn Daten zur CPU fließen sollen

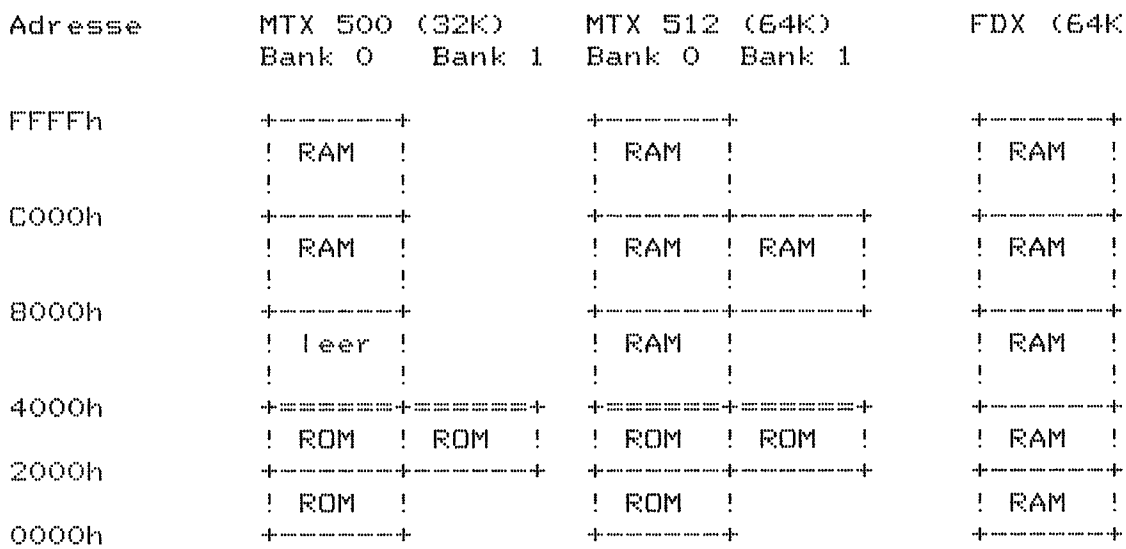
  - $\overline{\text{WR}}$ (write = schreiben) wird aktiv, wenn Daten von der CPU zum Speicher (oder zur Peripherie, siehe Abschn. 3.3) fließen sollen.

  (Daß der aktive Pegel low ist, wird dadurch gekennzeichnet, daß man einen Strich über dem symbolischen Namen des Signals setzt.)

Nach diesen etwas theoretischen Ausführungen nun zur Speicheraufteilung im konkreten Fall des MTX/FDX-Systems. Wie gesagt kann die CPU 65536 Byte, d.h. 64 KByte (Erläuterung im Vertiefungsstoff am Ende des Kapitels), direkt adressieren. Dies würde ausreichen, um die 24K ROM (read only memory = Nur-Lese Speicher) und 32K RAM (random access memory = Speicher mit wahlfreiem Zugriff (lesen und schreiben))im MTX 500 anzusprechen. Wie können aber die 24K ROM und 64K RAM im MTX 512 verwaltet werden? Hierzu werden einige Speicherabschnitte parallelgeschaltet, so daß ihnen dieselben Adressbereiche zugeordnet sind. Diese parallelgeschalteten Bereiche, genannt Banks, sind jedoch nie gleichzeitig zugänglich. Die Umschaltung erfolgt über das sogenannte Bankport-Register. Einzelheiten zu diesem Verfahren können dem MTX-Handbuch entnommen werden.

A s s e m b l e r : Kurs

Einen Überblick über die Speicheraufteilung der verschiedenen MTX-Systeme gibt Abb. 3.2-1:



Welcher Bereich steht uns für Assemblerprogramme und die dazu gehörigen Daten zur Verfügung? (Das folgende bezieht sich auf den Basic-Assembler.)
Allgemein gilt, daß der Speicherplatz ab FA52h von den Systemvariablen belegt wird. Dieser Bereich ist also tabu!!! Weiterhin reserviert sind die Bereiche, die von ROMs bzw. vom FDX-Basic beansprucht werden. Es bleiben also beim MTX 500 und beim FDX nicht ganz 32K von 8000h bis FA51h, beim MTX 512 knappe 48K von 4000h bis FA51H. Da das beim MTX 512 in Bank1 gelegene RAM nur durch Umschalten erreichbar ist, sollte man es nur in besonderen Fällen benutzen.

Vertiefungsstoff
Die Kapazität von Speichern wird meistens nicht in Byte, sondern in einer größeren Einheit, dem KByte (abgekürzt KB, sprich Kilobeit, oder noch kürzer 'ka') angegeben. Ein KB sind nicht, wie man vielleicht meinen könnte, 1000 Bytes, sondern 1024! Warum diese krumme Zahl? Nun, im Binärsystem ist es eine gerade Zahl, denn $1024 = 2^{10}$. Weil sie so nahe bei 1000 liegt, hat man die Vorsilbe 'Kilo' beibehalten. Die nächst größere Einheit ist ein MByte (MB, sprich Megabeit oder kurz 'embe' (hier wird witzigerweise das 'B' mitgesprochen, bei KB meist nicht)). Dies sind wiederum nicht 1 000 000 Bytes (wie man leider oft liest), sondern 1 KB * 1 KB = 1048576 Bytes.
Ende Vertiefungsstoff

3.3. Ports

Als Ports (zu deutsch etwa Tore) werden die Ein- und Ausgabekanäle bezeichnet, über die die CPU Daten mit der Peripherie austauschen kann. Unter den Begriff Peripherie fallen hier nicht nur externe Geräte wie Drucker etc., sondern auch Tastatur, Bildschirm, Tongeneratoren usw., also alles außer CPU und Hauptspeicher.

Ähnlich wie bei Schreib- oder Leseoperationen im Speicher wird auch über die Ports jeweils ein Datenbyte übertragen; der Transfer erfolgt über die 8 Leitungen des Datenbusses. Zur Auswahl eines bestimmten Ports legt die CPU eine 8-Bit-Adresse auf die 8 niederwertigen Leitungen des Adressbusses, die 8 höherwertigen Leitungen sind irrelevant. Es können also maximal 256 verschiedene Ports adressiert werden

A s s e m b l e r : Kurs

(zur Erinnerung: bei Speicheroperationen sind alle 16 Adressleitungen relevant, 65536 Speicherzellen können angesprochen werden).

Außer Daten- und Adressbus sind auch Steuerleitungen am Datentransfer beteiligt. Die Leitung $\overline{IOREQ}$ wird bei allen Ein- und Ausgabeoperationen aktiv, die $\overline{RD}$ - Leitung beim Lesen von Eingabeports (z.B. durch X=INP(n) in Basic), die $\overline{WR}$ - Leitung beim Schreiben auf Ausgabeports (z.B. durch OUT n,x in Basic). Diese Leitungen (übrigens alle aktiv low) werden von speziellen Bausteinen (z.b. PIO) ausgewertet, die dann den eigentlichen Datentransfer autonom abwickeln.

Zu beachten ist, daß Ein- und Ausgabeports strikt getrennt zu sehen sind, auch wenn sie dieselbe Adresse haben. Wird zum Beispiel ein Byte auf Port 4 ausgegeben und anschließend Port 4 gelesen, so muß der erhaltene Wert nicht mit dem ausgegebenen übereinstimmen.

Wie oben beschrieben, kann der Z80 je 256 Ein- und Ausgabekanäle ansprechen. Beim MTX ist nur ein kleiner Teil dieser Anschlüsse belegt, der Rest ist für Erweiterungen frei verfügbar. Eine Übersicht über die zu den verschiedenen Peripheriegeräten gehörigen Portadressen ist im Handbuch zu MTX bzw. FDX und im Info zu finden.

Wichtig ist hierbei, daß in der Regel die Relation 1 Port - 1 Peripheriegerät nicht stimmt, es können also mehrere Ports nötig sein, um gültige Daten zu/von einem Peripheriegerät zu übertragen. Ein Beispiel: an der Ausgabe eines Zeichens an die Drucker-Parallelschnittstelle sind 2 Eingabe- und ein Ausgabeport beteiligt. Am Ausgabeport 4 wird das Datenbyte bereitgestellt, durch Lesen von Eingabeport 4 wird der Zustand des Druckers ermittelt, und durch Lesen von Eingabeport 0 wird die Druckerleitung STROBE auf 0 gesetzt, d.h. dem Drucker wird mitgeteilt, daß die an Port 4 anstehenden Daten gültig sind und übernommen werden können.

H a r d w a r e: Druckerreset / Hitze

Hardware printer reset für Centronics parallel printer

(Jürgen Lindner, 4220)

Habe mir einen kleinen Taster eingebaut, der bei Betätigung den Drucker zurücksetzt. Das ist störungsfrei und auch bequemer als durch Aus- und Einschalten des Druckers oder durch software reset. Dieser Taster verbindet das PRIME-Signal auf Leitung Nr. 31 mit der Leitung Nr. 30. Man kann ihn auch am Drucker selbst einbauen, oder am Verbindungskabel anbringen. Ich habe ihn in den Computer eingebaut, und zwar am Stecker J6 (Handbuch Seite 199) auf der Hauptplatine (bei D/E1). Die Anschlüsse vorne, die leicht zugänglich sind, sind von rechts nach links die Leitungen 19-35. Wenn man hier die Leitungen 30 und 31 verbindet, wird der Drucker zurückgesetzt, was sich durch eine kurze Bewegung des Druckkopfes und durhc ein Piepsen bemerkbar macht. Gleichzeitig wird er online geschaltet, wenn er offline war und wenn Papier eingelegt ist.

MTX-Bräter (Herbert zur Nedden, 2000)

Wen hat die schlechte Kühlung im MTX nicht schon gestört. Aber da ist doch die Schraube, mit der der Kühlkörper an das Bodenblech gedrückt wird, aber wenn man fest zuzieht erntet man nur eine gebogene Platine und sonst nichts.

Die Lösung, bzw. eigentlich eine nur Verbesserung ist richtig einfach:

Man nehme: Schraubenzieher
Imbus-Schlüssel
Feine Flachzange
Seitenschneider
Bohrmaschine mit ca. 3,5mm - 4mm Bohrspitze (für Metall)
Kleine Schraube mit Mutter (3 mm Durchmesser, kurz)

1. Mit dem Imbus werden die Seitenwände der MTX entfernt. Dann das Tastaturkabel von der Hauptplatine (neudeutsch: Motherboard = Mutterbrett) abziehen, und Tastatur entfernen (aber aufheben!).
2. Mit dem Seitenschneider das Plastik der MTX-Rückwand oberhalb des Kühlkörpers (das ist das überstehende Metallstück auf der Hauptplatine) zum Teil entfernen (und wegwerfen).
3. Nun mit der Bohrmaschine durch das überstehende Kühlblech und die darunterliegende Bodenplatte des MTX ein Loch bohren - nein nicht vorher Hauptplatine ausbauen, Schraube lösen o.ä.!! Aber dabei natürlich nicht durch die Hauptplatine bohren.
4. So nun die Schraube, die durch die MTX-Bodenplatte, Hauptplatine und das Kühlblech geht herausschrauben (Schraubenzieher und Flachzange), und die Hauptplatine aus dem MTX herausziehen.
5. Jetzt die Metallspäne vom Bohren entfernen, und durch das freigewordene alte Loch Hauptplatine/Kühlkörper/Spannungsregler eine neue kurze Schraube setzen und festzeihen.
6. Jetzt die Hauptplatine wieder einbauen, und die Schraube, die vorher durch die Bodenplatte des MTX und das Kühlblech ging durch das eben gebohrte Loch setzen und richtig fest anziehen - das alte Loch ist ja schließlich auch durch die neue Schraube blockiert.
7. Wer möchte kann auch noch die Bodenplatte dort abschleifen, wo der Kühlkörper aufliegt, und auch mit Wärmeleitpaste (sieht aus wie Zahnpasta, ist nur schmieriger) einreiben.

P A S C A L: Platzsparen

CMDRUN oder der CHaiNer (Herbert zur Nedden, 2000)

Unter dem RAM4-CP/M, genauer gesagt unter ZCPR2/P2DOS gibt es die Option, ein Programm CMDRUN.COM einzurichten. Dieses hat folgende hübsche Funktion:

Wenn ich ein Kommando eingebe, welches durch ein .COM-File auszuführen ist, dann sucht das System erst auf dem eingeloggten Laufwerk, und dann falls nicht gefunden entlag dem Suchpfad, einer Laufwerksliste, die mit dem Programm PATH.COM gesetzt werden kann, ob dieses .COM nicht irgendwo auf dem Suchpfad (Path = Pfad) zu findes ist. Wenn ja, wird dieses geladen und los gehts.

Nun wenn das .COM nicht gefunden wurde, wird entlang des Suchpfades das Programm CMDRUN.COM gesucht, und aufgerufen. Dabei wird die alte CP/M-Kommandozeile als Parameter mitgegeben. Wenn SUB.COM in CMDRUN.COM umbenamst wird, so bedeutet das, daß der Aufruf

```
A>SUB TEST
```

auch als

```
A>TEST
```

eingegeben werden kann (falls kein TEST.COM gefunden wird).

Und genauso sollen nun die PASCAL-.CHN-Dateien gestartet werden. Die .CHN-Dateien unterscheiden sich von Turbo-Pascal-.COM-Dateien darin, daß das Laufzeitsystem nicht mit drin ist. Also kann ein .CHN-File nur von einem laufenden Turbo-Pascal-Programm aus gestartet werden. Dafür sind .CHN-Dateien von Turbo-Pascal deutlich kürzer als .COMs.

Genau das macht PASCAL.COM! - von Bernd Preusing

Wird PASCAL als .COM übersetzt und in CMDRUN umbenamst, dann bewirkt ein nicht gefundenes .COM,daß vom System vor die Kommandozeile CMDRUN gesetzt wird, also PASCAL den Salat erhält. Nun sucht PASCAL (alias CMDRUN) nach einem .CHN-File, welches als Namen den Namen des gesuchten .COM-File hat.
In Obigen Beispiel:

```
A>TEST
```

ohne Findet von TEST.COM bewirkt

```
A>CMDRUN TEST
```

worauf PASCAL (ist ja in CMDRUM umbenamst) nach TEST.CHN entlang des Suchpfades sucht, und falls gefunden auch aufruft.

Werden übrigens auch weitere Parameter der CP/M-Kommandozeile weitergereicht!

Der Suchpfad liegt ab Adresse Hex E96E. Jeder Eintrag besteht aus zwei Bytes, dem Laufwerk (A=1, B=1, . ..) und dem User-Bereich.

Nun zu PASCAL.PAS:
Auf der folgenden Seite ist das Listing. (TURBO-PASCAL 3.0)

Die Zeilen vom Kommentar (* ***** *) bis zum Kommentar (* ***** *) müssen für nicht-RAM4-Systeme entfallen, da diese keinen Path haben. Ebenso kann die VARiable Path in diesem Fall auch weg.

P A S C A L: Platzsparen

```
PROGRAM Pascal;
{ Chain als CMDRUN unter ZCPR2/RAM42 von Bernd Preusing }

TYPE
 Str14 = String[14];
VAR
 Path: Array[1..8] of Record                    (* Suchpfad    *)
                         Drive, User: Byte      (* nur RAM 4.x *)
                       End absolute $E96E;

 CmdLin: String[80] ABSOLUTE $0080;             (* CP/M-Kommandozeile *)
 NewLin: String[80];
 OldUser, OldDrive, i: Integer;
 f:File;
 Name: Str14;

PROCEDURE SetUser(u:Integer);
          BEGIN
                BDOS(32,u)
          END;

FUNCTION FileExists(VAR Name:Str14; Drive, User:Byte):Boolean;
         VAR nn:String[20];
         BEGIN
               nn:=Chr(Drive+$40)+':'+Name+'.CHN';
               SetUser(User);
               Assign(f,nn);
               {$I-} Reset(f); {$I+}
               FileExists:=(IOResult=0);
               {SetUser(OldUser) darf leider nicht wg. chain }
         END {FileExists};

BEGIN
      OldUser:=BDOS(32,$FF);
      OldDrive:=BDOS(25);
      IF ParamCount>0
      THEN BEGIN
           Name:=ParamStr(1);
           NewLin:='';
           IF ParamCount>1
           THEN FOR i:=2 TO ParamCount DO NewLin:=NewLin+' '+ParamStr(i)
           ELSE NewLin:=' ';
           CmdLin:=NewLin;
           IF FileExists(Name,OldDrive+1,OldUser)
           THEN Chain(f)
           ELSE BEGIN                           (* ****** *)
                i:=1;
                WHILE Path[i].Drive<>0 DO BEGIN
                      IF FileExists(Name,Path[i].Drive,Path[i].User)
                      THEN Chain(f);
                      i:=Succ(i);
                END; { while }
                WriteLn('CMDRUN: File nicht gefunden');
                END                             (* ****** *)
           END
      ELSE WriteLn('CMDRUN: Kein Name angegeben!');
END.
```

H a r d w a r e: HD64180-Karte

HD64180 ECB-Karte fertig (Claudio Romanazzi, 3070)

Schon lange hat mich der 1984 auf dem Markt erschienene Mikroprozessor HD64180 interessiert. Er ist vollständig kompatibel zum legalen Befehlssatz des Z80 und hat einige interessante zusätzliche Befehle. Am interessantesten ist für mich der Multiplikationsbefehl. Die Einzelregister eines Doppelregisters werden mit den Faktoren geladen, dann miteinander multipliziert und als 16-Bit-Wert im selben Doppelregister wieder ausgegeben (**MLT**).

Neu ist der **SLEEP**-Befehl. Er versetzt die MPU (Micro Processing Unit) in einen stromsparenden Schlafmodus.

IN0 g,(m) und **OUT0 (m),g** versorgen die internen I/O-Ports. Und mit den Spezialbefehlen **OTIM, OTIMR, OTDM** und **OTDMR** kann man die gamze Latte der Ports per Blockbefehl initialisieren. Dabei wird automatisch das höherwertige Portbyte auf 0 gesetzt, weil das B-Register als Zähler mißbraucht wird. Wenn das B-Register um 1 decrementiert wird, wird das c-Register je nach Bedarf um 1 de- oder incrementiert. Dadurch entfällt die Aktualisierung des Portregisters.

Neu sind ebenfalls die Testbefehle. Das Ergebnis verändert nur das Flagregister:

TSTIO m ANDet I/O-Port und Accu,
TST g ANDet Register und Accu,
TST m ANDet Zahl und Accu und
TST (HL) ANDet (HL) und Accu.

Soweit die neuen Befehle. Nun zur Karte: Sie besteht aus nur 12 Bausteinen, bei 64K Ram aus 13. Die Schaltungszeichnung hat Herbert wohl irgendwo im Info beigelegt.

So wie die Karte jetzt ist könnte sie 128K Ram adressieren, mehr ist ohne weiteres möglich. Von den vielen Möglichkeiten, die der Prozessor bietet ist im Moment nichts weiter angeschlossen. Möglich wären:

Memory Management Unit (MMU) für bis zu 512K Ram und 64 K I/O,
2-Kanal DMA mit den Möglichkeiten Memory <-> Memory, Memory <-> I/O und Memory <-> Memory Mapped (zur Zeit nur M<->M möglich)
Programmierbarer Refreshgenerator,
2-Kanal, vollduplex, asynchrone serielle Schnittstelle mit programmierbarem Baudratengenerator und Handshakesignelen für die Modemkontrolle,
getakteter, serieller I/O-Port, sehr schnell mit 200k Bits/Sekunde,
2-Kanal 16-Bit programmierbarer Timer
vielseitiger Interruptkontroller für die vier externen und acht internen Interruptmöglichkeiten.

Desweiteren können die internen Ports verschoben werden, damit diese nicht mit anderen in Konflikt geraten.

Zur Zeit bin ich gerade daran, ein Bios zu schreiben, damit man die Karte richtig ansprechen kann. Unter CPM funktioniert das schon ganz gut, mit RAM4 habe ich noch Schwierigkeiten. Das ganze soll dann so funktionieren, als ob die HD-Karte der Hauptprozessor wäre. Der Z80 führt alle Biosroutinen aus und übergibt, soweit nötig die Parameter ins HD-Ram, wo dann völlig separat weitergerechnet wird. Um zu 'wissen', ob was anliegt, unterbricht der Z80 die Karte etwa alle Millisekunde. Das reicht völlig aus, um zum Beispiel die Tastatur zu bedienen. Mehr beim nächsten mal.

Ach ja, lt. C't kann man bis 36 mHz takten. Das entspricht einem mit 22,5 mHz getakteten Z80. IST DAS NICHTS??

Claudio