MTX User-Club Deutschland

Info 21
14.07.1987

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

Inhaltsverzeichnis

Preis für dieses Info: DM 9,50

Redaktionsschluß für Info 22: 15. August 1987

Die meisten Menschen ängstigen sich vor dem, was ihnen unbekannt ist. Der Computer ist ein typisches Beispiel dafür. Welch ein Unsinn wird da oft erzählt: von der übermenschlichen Intelligenz des Computers — bis zu seiner angeblichen Fähigkeit, Menschen zu beherrschen!

Ein Computer ist nicht intelligent. Er ist eine Maschine, die vorprogrammierte Schritte auf der Basis von nur zwei Ziffern, nämlich 0 und 1, ausführen kann — und dies nur auf drei verschiedene Arten. Mit anderen Worten: Ein Computer ist ein Rechner, der nur ganz simple Rechenschritte ausführen kann, allerdings mit einer für Menschen unfaßbar hohen Geschwindigkeit. Deshalb ist der Rechner nichts anderes als ein Werkzeug, mit dem die Kapazität des menschlichen Gehirns millionenfach gesteigert wird. Dies spart den Anwendern, die sich des Rechners bedienen, Zeit und Energie: Während der Rechner die stupiden Rechnungsschritte absolviert, haben wir Zeit, kreativ zu sein. Wir können Denken und Ideen entwickeln — genau das, was der Rechner nie kann!

Liebe Leserinnen, liebe Leser,

bitte entschuldigt alle, aber ich habe nicht gemerkt, daß Info 21 mitten im Satz endet. Eigentlich sollte der halbe Satz einfach dem TipEx zum Opfer fallen, aber wie Murph so spielt.

Damit Ihr Euch schon jetzt darauf einstellen könnt: Ich werde vermutlich ab Mitte September für ca. 3 Wochen **Urlaub** machen, wobei noch nicht klar ist, ob einfach in/bei Hamburg, oder ob weiter weg. Jedenfalls werde ich unabhängig von meinem Urlaubsort während dieser Zeit keinerlei Dinge in Sachen Computer unternehmen. Auch meinen eigenen werde ich so lange abgeschaltet lassen, keine Telefonsprechstunden abhalten u.s.w. Bitte ruft mich in dieser Zeit nicht an. Das würde nichts nutzen, und da mein Bruder in dieser Zeit in unserer Wohnung hausen wird, kommt Ihr sogar durch, gebt Geld aus, und habt nichts davon! Wenn ich die Dinge richtig einschätze, dann brauche ich nach meinem Urlaub ca. 1 Woche, um die so anliegenden Dinge zu erledigen, bis ich wieder voll da bin.
In dieser Zeit bitte ich Euch Euch mit Euren Problemen mal an andere Mitgieder des Clubs zu wenden. Info-Nachbestellungen, Anmeldungen, ... müssen dann leider bis nach meinem Urlaub warten.

Auch in diesem Info sind hoffentlich wieder einige Kleinigkeiten, die für die eine oder den anderen interessant sind. Zum Beispiel güstige Altinfos, ein neuer Name in der Clubleitung (aber der selbe Mensch! - nämlich ICH), Grafik-Software, hohe Frequenzen, Proportionalschrift für die Olympia-Carrera ...

Jedesmal, wenn ich das Info in Druck gebe, stelle ich erst fest, wieviele Mitglieder das Info erhalten, d.h. genug Geld auf dem Konto haben, und versuche abzuschätzen, wieviele 'Nachzügler' und neue Mitglieder das Info später haben wollen. Zu meinem Pech habe ich so einige Restexemplare der Info's 7 und 11-20 hier liegen. Daher biete ich diese Altinfos solange der Vorrat reicht **günstig** an: siehe Angebotsliste.

Da ich regelmäßig bei **HW-Elektronik** in Hamburg einkaufen fahre - zumal es fast auf dem Weg zur Arbeit liegt, bin ich gerne bereit für Euch dort einzukaufen. Schickt mir Eure Einkaufsliste - und einen Scheck oder Geld.

Es ist mal wieder so weit. Eine **Bestellaktion** ist in diesem Info initiiert. Ich habe einige Dinge darin aufgenommen. Wenn Euch noch etwas einfällt, was nicht dort vermerkt ist - naja, dann nehmt bitte die Rückseite. Falls eine(r) von Euch Laufwerke o.ä. bei dieser Gelegenheit gebraucht, aber 100%-ig Funktionstüchtig abgeben möchte; warum nicht. Vielleicht sucht ein anderes Mitglied just dieses Teil, und ist bereit mehr für ein 15 Monate lang aktiv getestetes Gerät zu löhnen - oder wie ?

Nun da mein MTX/FDX-System mit etwas mehr als 4 MHz **Taktfrequenz** läuft, bin ich gere bereit auch anderen Memotechs zu Flügeln zu verhelfen. Garantieren, daß es so klappt wie bei mir kann ich beim besten Willen noch nicht - schließlich ist meine Kiste ein Prototyp dieser Leistungsklasse, und ich habe immer wieder von Problemen mit 6 MHz gehört. Außerdem mag nicht jede Hauptplatine 8 MHz ohne Wait! Aber 8 MHz mit Wait sind etwa 7 MHz, also doch recht zügig. Aber ich wollte 8, damit die Umschaltung jederzeit möglich ist.
Mehr dazu ab Seite 29.

Euer

Herbert zur Nedden

C L U B: Wer tut Was / Ports

Wer tut Was

Allgemeines	H. zur Nedden
Info-Inhaltsverzeichnis	U. Hönisch
(FDX-)BASIC	A. Viebke
CP/M System	B. Preusing, H. zur Nedden
Assembler	H. Oppmann
NewWord	U. Grass, H. zur Nedden
Turbo-Pascal	O. Krumnow, T. Wulf
SuperCalc	W. Gieger
Edicta-Grafik	H. zur Nedden, C. Löhrmann, C. Romanazzi
Was gibt's wo billig	H. zur Nedden
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
	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

Anzeigetexte und Absender bitte schriftich an Herbert zur Nedden!

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.
- Ich habe Apple-Communication-Software: Software für Rechnerkopplung Computer mit einem Apple. Das sind zwei Disketten (1x MTX, 1x Apple), die ich ggf. verleihe, da ich die Apple nicht kopieren kann.
- FDX, ein oder zwei Laufwerk(e), in Top-Zustand (Post-versand-fähig, d.h. rüttelfest) ab DM 900.-
- SDX, ein Laufwerk, in Top-Zustand (Post-versand-fähig, d.h. rüttelfest) für DM 750.-
  Was ich weitergebe ist überprüft, FDX bootet dann einwandfrei!
- Kleinen S/W-Monitor, als Zweitgerät für VS 4 geeignet: DM 60.-
- Einspaltige Etiketten, 1000 Stück, 8,9 cm x 3,6 cm: DM 16.-
- 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, 74LS139, 74LS258 je DM 0.50; 74LS10, 74LS11 je DM 0.30
- Z80-Chips: Z80A CPU DM 1.50, Z80A CTC DM 2.50, Z80A SIO DM 4.-

V E R K A U F

Uwe Grass, Wachholtzstr. 8, 3300 Braunschweig, 0531-343167:
MTX 500 mit 512k, RS232, ECB-Option, Netzteilumbau, verlötete Platinen, 5MHz umschaltbar (also allem was gut und teuer ist), FDX 2 Lw., Monitor DM 2200,--. Auf Wunsch wird auch noch die 80-Zeichenkarte umgebaut, Booteprom getauscht, SRAM-Floppy installiert.

Wolfgang Dexheimer, Gleiwitzer-Str.2, 6750 Kaiserslautern, 0631/75775:
Schaltuhr Grässlin: DM 10.-, TDA7000 neu DM 3.-, Elektor Platine neu Nr. EPS 84063 (Echolot) DM 5.-, Dimmer DM 5.-, Video-Cas. Beta-L500 neu DM 6.-

Michael Moritz, Niddablick 3, 6368 Bad Vilbel, 06101/44519:
MTX 500 (64kB), FDX, 2 Lw, TP 200, DMX 80, CBASIC VB DM 1100.-

Dietmar Gröning-Niehaus, Hustadtring 151, 4630 Bochum, 0234-702171:
MTX 500, 265k-Karte, def. RS232, Kompendium, Druckerkabel, Info 1-20 DM 450.-, TP200 DM 160.-

S U C H E

Thomas Völkel, Rahlstedter Weg 130, 2000 Hamburg 72, 040/6439735:
Suche billigen Akustikkoppler

C L U B: Redaktionelles / Fragen / Ankündigung

Kontostand (Herbert zur Nedden, 2000)

Nach dem letzen Info waren leider wieder einige von Euch über Euren Kontostand irritiert. Ich möchte daher etwas Licht in das Dunkel bringen:

Auf dem Info steht ein Datum. Dieses Datum ist der Tag, an dem ich die Infos von Euren Konten abbuche, und die Adreßaufkleber drucke. Dann setze ich mich - aber nicht immer am selben Tag - hin, und mache die Umschläge fertig. Normalerweise ist das Info an dem Tag schon in der Druckerei.
Dann dauert es logischerweise noch etwas, bis die Infos versandfertig eingetütet bei der Post angelangt sind, und danach auch noch mehr oder weniger lang, bir Ihr sie in Händen habt.
Wenn ich nun eine Überweisung oder einen Scheck nach dem Ausdrucken der Adreßaufkleber erhalte, so schreibe ich das Geld selbstverständlich sofort gut, aber ich gehe dann nicht mehr die Aufkleber, Umschläge oder gar eingetüteten Sendungen durch, um den auf dem Aufkleber genannten Kontostand zu korrigieren. Fazit: Der Kontostand auf dem Info-Umschlag ist veraltet.
Zu allem Überfluß kann eine Überweisung auf das Konto des MTX User-Club Deutschlands einige Zeit dauern, insbesondere, wenn von einer nicht-post Bank aus überwiesen wird. Zeiträume von 8 Werktagen, d.h. 10 Tagen kommen durchaus vor.

Betr. BASIC-Programmlistings (Karlheiz Rößler, 7920)

BASIC-Programme, die als Listing ins Info kommen sollten zuvor mit RENUMBER bearbeitet werden. Das Eintippen vereinfacht sich wesentlich, wenn man AUTO benutzen kann.

Anm.d.HzN: Ein richtiges RENUMBER gibt es bei Christian Löhrmann auf der BASIC-U.001-Pulic-Domain Diskette/Cassette.

Fragen

F: (Klaus-Peter Kielbassa, 4400)
An alle dataphon s21-23d Besitzer:
Ich suche eine Möglichkeit, das o.g. dataphon mit 1200/75 bzw. 75/1200 Baud zu betreiben - d.h. beide Frequenzen gleichzeitig! Die Verbindung soll mit einem anderen solchen Akustikkoppler hergestellt werden.

EDICTA-Grafik (Herbert zur Nedden, 2000)

Claudio Romanazzi hat es geschafft. Die Einbindung der EDICTA-Grafikkarte in TURBO-Pascal ist fertig, und von mir auch an alle mir bekannten Besitzer einer solchen Karte verschickt.
Die Routinen sind alle in Assembler geschrieben, und das Assembler-Modul ist derart geschrieben, daß es relativ leicht in andere Programmiersprachen eingebaut werden kann. Olaf Krumnow wird diese Routinen trotz seines Grundwehrdienstes für RAM 4.x-Besitzer in das KLICK einlagern. Damit können sie genutzt werden, ohne den kostbaren CP/M-Speicher zu verbrauchen.
Claudio ist jetzt dabei die Grafikroutinen zu erweitern, um alle möglichen Befehle des Prozessors ausnutzen zu können. Bislang sind nur die eingebaut, um vernünftig mit der Karte arbeiten zu können, als da wären PLOT, LINE, RECHTECK, KREIS, HARDCOPY, SAVE, LOAD, PRINT, LINIEN-MUSTER, u.e.a. Er sprach von Ende Juli - aber mit Vorbehalt, weil er dann in Urlaub fahren wird.

A S S E M B L E R: Kurs

Kommentare zum Assemblerkurs

(Kurt-Bernd Rohloff, 8000)

Zur Frage nach dem fiktiven Assembler in Info 20-4:

Anm.d.HzN: (d.h. Anm.d.Herbert zur Nedden)
Ich fragte im Info 20, warum Kurt-Bernd im Assembelrkurs nicht mit dem Z80-Assembler, sondern mit einem fiktiven Assembler arbeitet.

Dieser zugegebenermaßen hohe Aufwand scheint mir aus mehreren Gründen gerechtfertigt. Sie sind didaktischer Art, weshalb Du als anerkannter Experte vielleicht Schwierigkeiten haben wirst, sie nachzuvollziehen. Bedenke bitte, daß sich der Kurs an Leute richtet, die BASIC können - mehr nicht! Wenn ich bedenke, daß für viele bereits die englische Sprache eine Hürde darstellt, dann wird dies sicher erst recht auf das hexadezimale Zahlensystem zutreffen, daß Voraussetzung für die Verwendung des Panels ist, von den Panel-Befehlen einmal ganz abgesehen. Daher arbeitet die I2DITVM bewußt im Dezimalsystem. Außerdem braucht man, außer ihren Maschinenbefehlen, keine weiteren Steuerbefehle (für Anzeige Register, Speicher usw.), so daß diese Pseudo-Maschine schon in einem frühen Stadium eingesetzt werden kann. Daß kommt der Erfahrung entgegen, daß Lernen mit dem Computer weit mehr Spaß macht als mit Papier und Bleistift.

Als weiterer Vorteil wäre die ständige Visualisierung der gesamten Maschine zu nennen. Die Ausführung eines jeden Befehls kann sofort optisch nachvollzogen werden. Insbesondere der Stapelspeicher läßt sich so weit besser verstehen als mit dem Panel. Besonders gut scheint mit hier der optische Stapelzeiger zu sein, der den Begriff eines Zeigers (dem BASIC Programmierer ja völlig fremd) gut veranschaulicht.

Daß für die I2DITVM nicht einfach eine Teilmenge des Z80 Befehlssatzes genommen wurde, geschah deshalb, um den (intuitiv leicht entstehenden) Eindruck zu vermeiden, der Z80 Assembler sei nun der Assembler. Mit der I2DITVM sollte vielmehr die im Kurs bereits erwähnte Tatsache verdeutlicht werden, daß man sich bei einem anderen Prozessor auch mit einer anderen Assemblersprache abfinden muß. Wenn man immer nur von Z80 hört und mit Z80 arbeitet und keinen anderen Prozessor kennt, findet doch unwillkürlich eine geistige Einengung statt. Man kann sich dann ab einem Punkt einfach gar keine andere Assemblersprache mehr vorstellen. Dieser Irrmeinung sollte gleich von Anfang an entgegengewirkt werden.

Für die I2DITVM gibt es bewußt keinen Assembler. Dadurch, daß man nur die Maschinenbefehle eingeben kann, wird man nämlich gezwungen, die Übersetzung von Programmen in der Assemblersprache in die Maschinensprache selbst vorzunehmen. Nur so versteht man m. E. wirklich, was ein Assembler eigentlich tut. Dies ist umso wichtiger, als der BASIC Assembler ja völlig automatisch arbeitet, so daß dem Benutzer der Vorgang des Assemblierens, d. h. die Umsetzung in den Maschinencode, verborgen bleibt. Im Gegensatz zu den unter CP/M arbeitenden Assemblern "sieht" man von dem Maschinencode nichts. Das ist zwar sehr bequem, aber schwieriger zu begreifen. Gerade letzteres wollen wir durch den Kurs aber fördern!

A S S E M B L E R: Kurs

Natürlich spielt auch ein mehr psychologisches Argument mit hinein. Assembler wird von vielen als schwierig, ja geradezu esoterisch angesehen. Es wäre daher falsch, den Lernenden gleich ins kalte Wasser stürzen zu wollen. Gerade der Einstieg, der erste Kontakt mit einer neuen Materie, ist der kritische Punkt. Wenn hier der Lernende nicht sorgsam und feinfühlig an die Materie herangeführt (nicht gestoßen!) wird, werfen viele gleich die Flinte ins Korn. Hier soll die I2DITVM ansetzen und zeigen, daß es im Prinzip doch eigentlich ganz leicht ist. Ich bin sehr zuversichtlich, daß sich dieses Gefühl sowie, und das ist wohl ebenso wichtig, der Wunsch, nun doch weiter und tiefer in diese Materie einzudringen, beim Lernenden einstellt, wenn er den zweiten Teil mit der I2DITVM durchgearbeitet hat.

Schließlich ist auch noch die Sicherheit gegen Programmier- und Bedienungsfehler zu erwähnen. Auch im single-step Modus kann man im Panel schon einiges durcheinanderbringen. Diese sind dank der beiden schwarzen Tasten zwar nicht unbedingt tödlich, aber für den mit der Materie nicht vertrauten (und nur für die ist die I2DITVM gedacht) ist es zumindest sehr frustrierend, wenn der Rechner plötzlich hängt oder sonstwie spinnt.

Im übrigen habe ich das Konzept des zweistufigen Vorgehens (erst hypothetischer, dann realer Prozessor) dem Mikroprozessorkurs des DAG Technikums entlehnt, an dem ich im Winter 1982/83 teilnahm. Der Vorteil ist halt, kurz gesagt, der, daß man unabhängig vom realen Prozessor freier ist in der Wahl der didaktischen Vorgehensweise und auch Inhalte vermitteln kann, die der reale Prozessor gar nicht leistet (z. B. Multiplizieren). Zu Deiner Beruhigung: wir werden schon noch den Z80 Prozessor mit seinem Assembler besprechen, allerdings erst unter Punkt 3 und 4.

Ich hoffe, daß ich Dir meine (Hinter-)Gedanken einigermaßen überzeugend dargelegt habe.

B A S I C: DISC-Befehle

FDX-Disc-BASIC-Befehle: (Herbert Herberg, 2000)
Lange ist es her, daß ich in den Infos auf die DISC-Befehle des FDXB eingegangen bin. Also hier eine Übersicht.

Hinter den Befehl DISC können (siehe FDX-Handbuch) diverse Disketten-Befehle gesetzt werden, die aber nicht alle beschrieben worden sind:
CLOSE, DIR, EOF, ERA, INPUT, KILL, LINE INPUT, LOAD, OPEN, PRINT, QUIT, READ, REC, REN, SAVE, TYPE, WRITE.

```
DISC CLOSE #1            Schließt Datei von Kanal-Nr. 1
DISC DIR "files"         Listet Directory aller Dateien, passend zu files
DISC ERA "files"         Löscht files von Diskette
DISC INPUT #1,A$         Liest Daten von Kanal 1 nach A$
DISC KILL #1             Schließt und löscht Datei von Kanal 1
DISC LINE INPUT #1,A$ Liest eine Zeile von Kanal 1 nach A$
DISC LOAD "file"         Lädt BASIC-Programm file
DISC OPEN #1,"file","typ",reclen
                         öffnet file als Kanal 1 von angegebenen typ
                         (O,I,R) mit satzlänge reclen für typ R
DISC PRINT #1,A$         Schreibt A$ auf Kanal 1
DISC QUIT                Beendet FDX-BASIC (aber schlecht!)
DISC READ "file",loc     Liest file und speichert den Inhalt ab Adr. loc
DISC REC #1,nr           Positioniert Kanal 1 auf Satz Nummer  nr. Dafür
                         muß  der Kanal 1 mit typ R eröffnet werden.
                         O<= nr <= ?
DISC REN "fileneu"="filealt"
                         Benennt filealt in fileneu um
DISC SAVE "file"         Schreibt BASIC-Programm auf Diskette
DISC TYPE "file"         Listet file auf dem Bildschirm (ist nicht LIST!)
DISC WRITE "file",loc,len
                         Schreibt len Bytes aus dem RAM ab loc in die file
DISC EOF #1,lin          Wenn Ende der Datei von Kanal 1, dann GOTO Zeile
                         Nr. lin
```

Zum Verständnis:
Da es im BASIC nicht sinnvoll ist bei jeder Ein- und Ausgabe e.t.c von/auf Diskette den Dateinamen anzugeben, wird jeder Datei, mit der gearbeitet werden soll im DISC OPEN u.a. eine Kanalnummer zugeteilt, die dann an stelle des Dateinamen bei Zugriffen verwendet wird.

Da das BASIC.COM (von Andreas Viebke erweitertes/fehlerbereinigtes) FDXB.COM das FDXB.COM weitgehend vertrieben hat, auch die neuen Befehle:

BASIC.COM

```
DISC QUIT                herausgenommen
DISC LIST "file"         Liefert ein LIST in die Datei file
DISC NEW  drive          Meldet  das  Laufwerk  mit  der  Nummer  drive  an
                         (A:=O,  B:=1,  ...).  Ist  nach  Diskettenwechsel
                         notwendig!
RESTORE                  akzeptiert statt fester Zahl auch Variable
VAL                      Versteht auch Arithmetische Ausdrücke
                         z.B.: LET A=VAL(SIN(X)+X-3)
```

B A S I C: Tortengrafik

(Karlheinz Rößler, 7920)

Tortengrafik

```
10 REM Tortengrafik, Karlheinz Rößler, 23.06.87
20 INPUT "Titel: ";T$
30 INPUT "Anzahl der Posten: ";N
40 INPUT "Namenslänge der Posten: ";M
50 DIM X(N),Y(N),A(N),P$(N,M)
60 INPUT "Startwinkel in Grad: ";A
70 LET A=2*PI*A/360
80 LET Q=0
90 FOR I=1 TO N
100 PRINT "Name des";I;". Postens: ";: INPUT P$(I)
110 PRINT "Größe des";I;". Postens: ";: INPUT X(I)
120 LET Q=Q+X(I)
130 NEXT I
140 FOR I=1 TO N
150 LET Y(I)=X(I)/Q*PI
160 LET A(I)=A
170 FOR J=1 TO I
180 LET A(I)=A(I)+2*Y(J)
190 NEXT J
200 NEXT I
210 VS 4: COLOUR 0,15: COLOUR 2,15: CLS : COLOUR 3,1: COLOUR 1,1
220 PRINT "  ";T$
230 CIRCLE 104,96,60
240 ANGLE A
250 FOR I=1 TO N
260 COLOUR 3,1: PLOT 104,96: DRAW 60
270 READ C: IF C=14 THEN  RESTORE 380
280 PLOT 104,96: PHI Y(I): COLOUR 1,C: COLOUR 3,C: ATTR 2,1: ATTR 3,1: DRAW 30
290 ATTR 2,0: ATTR 3,0: DRAW 40
300 LET X=104+75*COS(A(I)-Y(I)): LET Y=96+75*SIN(A(I)-Y(I))
310 LET X=INT(X/8+.5): LET Y=23-INT(Y/8+.5): IF X<13 THEN  LET X=X-1: IF X<7 THEN  LET X=X-1
320 FOR J=0 TO LEN(P$(I))-1
330 CSR X+J,Y: IF SPK$<>" " THEN  LET Y=Y+1: LET J=LEN(P$(I))-1
340 NEXT J
350 CSR X,Y: PRINT P$(I)
360 ANGLE A(I)
370 NEXT I
380 DATA  3,4,8,7,10,12,13,14
390 COLOUR 3,1: LINE 215,0,215,191
400 RESTORE 380
410 FOR I=1 TO N
420 READ C: IF C=14 THEN  RESTORE 380
430 COLOUR 1,C
440 LET X$=STR$(X(I))
450 CSR 28,I: PRINT RIGHT$(X$,LEN(X$)-1)
460 NEXT I
470 GOTO 470
```

B A S I C: Daten aus dem Äther

Ich habe heute (10-6-1987) Info 20 bekommen. (Hans Gras, NL-1506)

Der **A**(ndreas) **V**(iebke) hat etwas geschrieben über **Videodat**.
Hierzu noch einige informationen für Info 21:

Videodat ist ein System um Daten ins Videobild zu übertragen.

Dazu brauchen wir ein kleines Interface und ein Rechner mit RS-232. Vom Video-Ausgang **AV**-Anschluss (ha, ha, ha) oder Scart vom Fernseher gehen wir ins Interface. Heraus komt ein Datenstrom (300 oder 600 Baud) welchen wir mit "**CONTACT**" einlesen (Mit M1 oder so etwas geht es nicht ohne Datenverlust wegen fehlendem XON/XOFF-Protokoll). **CONTACT** läuft sehr gut mit **RAM**! (RAM **1**). Ich habe ein Interface gekauft bei "CON DATA", Weilerstrasse 20, 5040 Brühl. Kosten: DM 68,- als Bausatz oder DM 108,- als Fertiggerät (incl. MWST). Es lief sofort.
In den **WDR**-Programmen "**COMPUTER-CLUB**" und "**WARUM? DARUM!**" werden ca. **20 DIN A4** Seiten ausgestraht (normalerwaise am letzen Sonntag im Monat um 17.30), ca. 30kB. Ein ganze Menge Informationen und wenig Programme. Diese ASCII Dokumente bearbeite ich mit NewWord

Nun zu Holland:

In Holland gibt es auch einen Sender, der häfig BASIC-Programme in den Äther strahlt. Diese sind in **BASICODE** geschrieben, einem speziell hierfür entwickelten BASIC-Dialekt.

Um ein so empfangenens Programm laufen zu lassen, müssen einige Rechner- und BASIC-Spezifische Unterprogramme im Programm stehen, das empfangene Programm dahinter gehängt werden, und ggf. einige Befehle, die das eigene BASIC nicht oder anders benötigt angepaßt werden. Hinzu kommt natürlich, daß die Programme im Klartext, d.h. wie bei LIST, gesendet werden, und nicht in der jedem BASIC eigenen internen Verschlüsselung. Also muß ein Programm her, welches diese Arbeit des Umsetzens, ... erledigt.

1. für MBASIC-User gibt es ein Übersetzprogramm um diese BASICODE-Programme zu übersetzen (**Public Domain**). Es heißt CONVERT.COM und braucht dazu noch einige Hilfsdatein mit Statements und Functions. Bei mir zu bekommen. Nur fur ASCII-Programmen auf Disc.

2. Die meisten Programme laufen sofort (ca. 98%).

3. für MTX- und/oder FDX-BASIC-User gibt es mehrere Programme um es auch für diese Leute ans laufen zu bringen. dSM: Memotech hat ein wenig abweichende Basic. zB arrays mit fester Länge und sie fangen an bei (1,1) statt bei (0,0) usw. Aber alle Programme sind zum Laufen zu bringen. Viel Übung macht Spaß!!!

4. Diese Programme sind (ziemlich?) besser als AV's Programm. Es gibt auch Lösungen für z.B. PRINT"Hallo." --> PRINT "Hallo.". Unser Memotech braucht eine Leerstelle hinter einem Statement.

5. **BASICODE** Programmen werden über die holländischen Sender ausgestralht:
   - Mittwoch 19.00 uhr (MW 747kHz)
   - Sonntag 22.40 uhr (MW 1008kHz)
   Diese Sendungen müssen auch in Deutschland gut zu empfangen sein. (In Dänemark und Spanien sind diese Programmen zu hören und einzulesen!).

B A S I C: Daten aus dem Äther / D r u c k e r: Proportionalschrift

Bitte "**Downloaden**" auf Cassette, einlesen in mein Einlese-Programm, aufzeichnen auf Disc (notwendig fur MBASIC), überzetsen für MBASIC oder FDX-Basic. Für MTX aufzeichnen auf Cassette im MTX-Format (2400 Baud), und Übersetzen.
Diese Programme sind **nicht Public Domain.** Ich wollte gerne auch ein wenig Geld fur meine Mühe! Ich bastle seit Januar 1985!
Updates selbstverständlich lieferbar!

8. Bereits gesicherte Files können auch später bearbeitet werden!

9. An einer Ausgabe nur für Einlesen unter CP/M bastele ich gerade, aber läuft noch nicht gut wegen Interrupts von Cassetteport CTC-3. Es läuft mit FDX-BASIC einwandfrei.

Bis wiedersehen oder hören oder schreiben,

Proportionalschrift für DIABOLO-Kompatible oder **PROPRINT**

(Herbert zur Nedden, 2000)

Auf den nächsten zwei Seiten seht Ihr einen Ausdruck der Olympia Carrera, einer kleinen preiswerten Typenradschreibmaschine. Daran ist eigentlich nicht viel außergewöhnliches, außer der Tatsache, daß die Chose in Proportionalschrift gedruckt ist. Nun werden viele einwenden, das ist doch nichts dolles - zugegeben, wenn nicht trotz Proportionalschrift ein mehrspaltiger Blocksatz möglich wäre.
Ich habe von Andreas das Programm PROPRINT zusammen mit einem geeigneten Typenrad erstanden, PROPRINT aufgerufen, und es klappte nicht. Naja, wenn ich auch das falsche Typenrad verwende eigentlich kein Wunder. Also das mitgeschickte Typenrad geschnappt, eingesetzt, und wie nicht anders zu erwarten wurde ich mit einem beeindruckenden Ausdruck beschert.
Die Typenräder für Proportionalschirft unterscheiden sich von den anderen dadurch, daß Buchstaben wie das i nicht künstlich verbreitert, und solche wie das m und w nicht so eng sind. Dummerweise sind obendrein die Buchstaben auf dem Typenrad anders angeordnet, aber das Problem löst PROPRINT auch.
Die Dokumentation ist gut lesbar - 24 Seiten geschrieben mit Proportionalschrift. Also habe ich angefangen zu lesen, und mußte feststellen, daß die Handhabung verblüffend einfach ist, und auch so kleinigkeiten, wie das automatische Rausrücken an den rechten Rand, Seitennummer abwechselnd rechts und links mit in der Mitte sitzender Fußzeile, aber auch der mehrspaltige Blocksatz sind kein Problem!!! Und einige Überraschungen über die Möglichkeiten sind im Handbuch und in PROPRINT drin. Sogar drei umfangreiche Installationmenüs!

PROPORTIONALSCHRIFT FÜR DIE OLYMPIA CARRERA

Die Olympia Carrera ist ein feines Maschinchen. Deshalb bin ich auch umgestiegen: von meiner Olympia electronic compact 2 auf die Diablo-kompatible Carrera.

Zu Diablo kompatibel zu sein, heißt für die Carrera, horizontal 120 Schritte pro Zoll, vertikal 48 Schritte pro Zoll machen zu können. Nur leider hat sie keinen Umschalter für Proportionalschrift. Wer weiß, vielleicht könnte sie ja proportional, wenn man nur wüßte, wo man umschalten muß. Jedenfalls hat der Hersteller nichts dergleichen vorgesehen, ergo gibt's auch keine Typenräder mit Proportionalschrift für die Carrera. Das stimmt und stimmt auch wieder nicht, denn die Modelle mit Proportionalschrift verwenden Typenräder, die sich von den bei der Carrera verwendeten nur dadurch unterscheiden, daß sie mehr oder weniger fest in einer Plastik-Hülle sitzen. Die kann man aber mit sanfter Gewalt entfernen, so daß man das nackte Typenrad in der Hand hält. Und das sieht genauso aus wie ein Typenrad für die Carrera, paßt also! Nur schreiben kann man damit nicht, denn die Zeichen sind ganz anders angeordnet als bei Nicht-Proportionalschrift-Typenrädern. Man muß also umkodieren. Hat man das geschafft, muß man "nur" noch dafür sorgen, daß das 'i' weniger Platz bekommt als das 'e' und das wiederum weniger als das 'm'. Das reicht zwar noch nicht, aber ich hör' an dieser Stelle auf mit dem technischen Kram!

Hat man einmal angefangen, dann macht man auch was draus. So geht es mir zumindest meistens. Und so ist ein Programm entstanden, das **PROPRINT** heißt. Mit ihm ist dieser Text gedruckt worden. Und das Außergewöhnliche ist, daß **PROPRINT** Texte verarbeitet, die mit Newword erstellt worden sind (oder wohl auch mit Wordstar – ist Newword nicht vollkommen Wordstar-kompatibel?). Man muß sich nicht umgewöhnen, denn alles bleibt beim alten. Nur wird Newword nicht zum Drucken herangezogen; das macht jetzt **PROPRINT.** Natürlich nur, wenn man Diablo-kompatibel ist und Proportionalschrift-Typenräder verwendet.

Und weil man mit Proportionalschrift nicht so umgehen kann wie mit Nicht-Proportionalschrift, mußten zusätzliche Punkt- und Control-Kommandos eingeführt werden. Punkt-Kommandos, die Newword nicht kennt, werden ja nur mit einem '?' bedacht und ansonsten ignoriert. Rechts sind ein paar zu sehen.

Alle anderen und in Newword sowieso üblichen und beim Ausdruck relevanten Punkt-Kommandos werden verarbeitet.

.ml N: Linker Rand bei Kopf- und Fußzeilen ist N.

.mr N: Rechter Rand bei Kopf- und Fußzeilen ist N.

.ll gefolgt von einem oder mehreren Sonderzeichen ('$', '?', '<', '>') bewirkt, daß alle nachfolgenden Zeilen mit "Tabulatoren" versehen werden ('$'), so daß Spaltendruck möglich wird, oder zentriert ('?'), links oder rechts ('>') bündig gedruckt werden, letzteres auch abhängig von der Seitenzahl ('<').

.ba N: Bei Fettdruck soll der Zeichenabstand um N/120 Zoll erhöht werden.

.ul on/off: Durchgehendes Unterstreichen an/aus.

.sr N: Beim Hoch- und Tiefstellen N/48-Zoll-Schritte verwenden.

Die bei '.ll' genannten Funktionen können auch durch Control-Zeichen innerhalb von Zeilen eingeschaltet werden, gelten dann jedoch nur für die betreffende Zeile. Man muß also nicht jedes Mal ein Punkt-Kommando verwenden, wenn man mal eben eine Zeile zentrieren etc. will. Auch für Kopf- und Fußzeilen sind diese Control-Zeichen gedacht. Es ist auch möglich, die Control-Zeichen zu mischen. So kann z.B. ein Teil der Zeile zentriert, ein anderer Teil je nach Seitenzahl rechts- oder linksbündig gedruckt werden. (Wenn Herbert meine Seitenzahlen nicht weggetipext hat, dann müßte das unten auf der Seite zu sehen sein.) Als Control-Zeichen werden in Newword nicht oder selten benutzte Steuerzeichen verwendet: ^PX, ^PY, ^PU und ^PP. Ganz ehrlich: hat irgend jemand schon mal ^PX benutzt? ^PK existiert schon für den gleichen Zweck wie bei mir, nur kann man dieses Kommando auch im Text benutzen.

Also, ich fasse kurz zusammen: **PROPRINT** druckt Texte, die mit Newword erstellt wurden auf einem Diablo-kompatiblen Drucker mit Proportionalschrift-Typenrad aus. Es gibt eine wichtige und deshalb zu erwähnende Einschränkung, nämlich die, daß Merge-Printing nicht möglich ist. Dafür hat man einige andere Features, die Newword nicht hat, und Proportionalschrift. (Übrigens vermurksen ^PA-^PN-Sequenzen den Randausgleich **nicht**!

PROPRINT kann wie Newword installiert werden; man braucht aber kein extra Installationsprogramm, sondern ruft einen Installationsmodus auf. Die Anleitung wird so Mitte bis Ende Juli fertig sein, und **PROPRINT** wird für 45,- DM erhältlich sein.

Wer nicht genau weiß, ob sein Drucker Diablo-kompatibel ist, der möge das Vorhandensein folgender Steuersequenzen überprüfen: ESC LF = negativer Zeilenvorschub, ESC 1Eh n = vertikale Schrittweite ist n, ESC 1Fh n = horizontale Schrittweite ist n. Wenn jetzt noch Olympia-Typenräder 'reinpassen, dann kann proportional gedruckt werden! Bei Olympia gibt es übrigens und soweit ich weiß vier Proportionalschrift-Typenräder (das hier verwendete hat die Nummer **472** – die "Leitzahl" 4 ist wichtig), von denen eines häßlich ist (Juwel-Carré). Sie kosten pro Stück um die 50,- DM.

Andreas Viebke

C: Ein Taschenrechner

Wir bauen einen Taschenrechner (Michael Moewe,2000)

Im letzten Info war eine Anfrage, wer denn endlich einen Klick-Rechner schreibt. Da mich dieses Thema schon immer ein wenig interessiert hat und ich schon ein Taschenrechnerprogramm geschrieben habe -nicht klickfaehig-, moechte ich hier eine verbesserte Version eines Rechners zeigen, die jeden, der seine Programme klickfaehig machen kann, in die Lage versetzt sich seinen Taschenrechner selber zu 'schnitzen'.
Da es trivial ist einen Kalkulator zu schreiben, der 2 Operanden und einen Operator einliesst, und dann das Ergebnis berechnet und anzeigt, moechte ich einen Rechner vorstellen, der in der Lage ist 'beliebig' komplexe arithmetische Ausdruecke zu berechnen.

Um zuerst einmal zu klaeren, wie ein arithmetischer Ausdruck beliebiger Komplexitaet aussieht, moechte ich die GRAMMATIK (!) eines arithmetischen Ausdruckes aufschreiben. Dazu bediene ich mich der EBNF, die ich ein wenig erlaeutere.

EBNF ist eine Metasprache zur Beschreibung von Grammatiken.

arithmetischer Ausdruck (in EBNF):

```
<Ausdruck> ::= <Term> { (+|-) <Term> }
<Term>     ::= <Faktor> { (*|/) <Faktor> }
<Faktor>   ::= <Zahl> | '(' <Ausdruck> ')'
<Zahl>     ::= [-] [Praefix] <Digit> { <Digit> }
<Praefix>  ::= %|$|o
<Digit>    ::= 0|1|2|3|4|5|6|7|8|9|A|B|C|D|E|F
```

Die spitzen Klammern ('<', '>') um ein Wort bedeuten, dass dieses Wort im arithmetischen Ausdruck nicht vorkommt und nur der Beschreibung des Problem dient; z.B.: <Ausdruck> soll der arith. Ausdruck selber sein.
Ein Wort links von '::=' kann durch die Zeichen rechts von dem Zeichen '::=' ersetzt werden.
Das heisst, <Ausdruck> kann durch <Term> { (+|-) <Term> } ersetzt werden.
Ein arith. Ausdruck kann also sein: 1 + 2. (sic est)
Wobei 1 und 2 jeweils ein <Term> waeren.
Ein '|' beudtet, das alle Zeichen die durch ein '|' getrennt werden als Alternative benutzt werden koennen.
Das heisst, ein <Digit> kann jede Ziffer von '0' bis 'F' sein. Soweit alles klar ? ok.
Was zwischen '{' und '}' geklammert ist, kann beliebig oft (null bis unendlich mal) wiederholt werden.
{x} kann also sein: x, xx, xxx, ... u.s.w.
Was zwischen '[' und ']' geklammert ist, kann jedoch nur 0- oder 1-mal wiederholt werden.
[x] kann also sein: 'nichts' oder x.
Wer sich das hier ein- bis zweimal durchliesst, sollte eigentlich in der Lage sein, das wesentliche der EBNF-Darstellung eines arith. Ausdrukkes zu verstehen, wer das nicht kann, sollte nicht verzweifeln, es ist wirklich nicht sehr einfach, gerade auch wegen der gerafften Form.
Damit waeren wir also schon in der Lage, einen Kalkulator zu schreiben. Um ein wenig Hilfe zu geben, folgt (vielleicht) auf den naechsten Seiten ein kleines 'C'-Programm, das (fast) alles kann, was ich in der obigen EBNF dargestellt habe. Da durch die Grammatik der Vorrang Punkt- vor die Strichoperationen vorgegeben ist, brauche ich mir darum keine Sorgen zu machen.

C: Ein Taschenrechner

Die einzige Aufgabe besteht nur noch darin, die Grammatik in ein Programm umzusetzen.
Um noch ein kleines Problem offen zu lassen, habe ich die Verarbeitung verschiedener Zahlendarstellungen (HEX -sedezimal-,oktal ,binaer etc.), oben durch <Praefix> angedeutet, noch nicht implementiert. Die Nachruestung (?) dieses Features muesste in den Funktionen getsym() und toi() erfolgen, alle anderen Teile des Programmes bleiben davon verschont.
Um den Noerglern (oh Gott, ein Programm in **'C'**) von vorneherein den Wind aus den Segeln zu nehmen, ich haette das Programm auch in diesen Sprachen schreiben koennen: 8080-Assembler, Z80-Assembler, 6502-Assembler, 68000-Assembler, 8086-Assembler (mit 8087), Prolog, BASIC oder Turbo-Pascal; und, egal was ich genommen haette, irgendjemand waere sauer gewesen, dass ich nicht die Sprache gewaehlt habe, die **er/sie** versteht. Also habe ich die Sprache gewaehlt mit der ich mein Geld verdiene und die ich fuer sinnvoll hielt (BASIC waere wohl fehl am Platze gewesen).
Das Programm ist natuerlich nicht ohne weiteres klickfaehig, es muessten noch einige Veraenderungen vorgenommen werden; es ist aber auch mehr zur Demonstration gedacht, wie man 'so etwas' schreibt. Aber eine Umsetzung des Algorithmus' in Z80-Assembler sollte nicht unloesbar sein.
Falls ich Zeit haben sollte und jemand soviel Interesse hat, dass er/sie mir eine Diskette (#03) und fuer 1.10 DM Briefmarken schickt, schreibe ich das Ganze vielleicht noch in Turbo-Pascal um.
Entwickelt habe ich das ganze auf meinem AT-kompatiblen Rechner (10 MHz) mit Z80-Co-Rechner (6 MHz) und CP/M-Emulator. Und in Verbindung mit einer Winni (Festplatte) ist der Gedanke an eine RAM-Disk ueberfluessig, die paar Mikrosekunden, die ich da noch rausschinden koennte, verschenk' ich gerne.

```
/* Verschiedene Konstanten, die das Programm etwas lesbarer machen */
#define PLUS        0
#define MINUS       1
#define MAL         2
#define DURCH       3
#define KLA         4     /* runde KLammer Auf */
#define KLZ         5     /* runde KLammer Zu  */
#define ENDE        6     /* keine Zeichen mehr auszuwerten */
#define NO_SYM      7     /* kein gueltiges Symbol */
#define ZAHL        8     /* Zahl erkannt */
#define TRUE        0xFFFF
#define FALSE       0x0000

/* Globale Variable */
short fehler = FALSE;
short sym_wert = 0;
char *ptr,*altptr;
```

C: Ein Taschenrechner

```
/* verschiedene Testausdruecke */
char *test[] = { "1 + 2 * 3",
                 "(1 + 2) * 3",
                 "(7)",
                 "(1 + 2) * (3+4)",
                 "((((567))))",
                 "1+2+3-3-2-1",
                 "1 - 1 ++",
                 "(((1 +2) * 7)",
                 "1+2    - 4 X",
                 "(1+2) * (1+2 * 2)"
               };/* test */

/* Hautprogramm, hier nur zu Testzwecken, die eigentliche Anwendung
wird natuerlich anders aussehen. */
main()
{
  short i,lv;

  for (lv=0;lv<10;lv++)
    {
    fehler = FALSE;
    ptr = test[lv];
    printf("\n%s",ptr);
    i = ausdruck();
    if ((getsym() == ENDE) && (fehler != TRUE))
      printf(" =%d",i);
    else
      printf(" ist ein fehlerhafter Ausdruck !!!");
    }/* for (lv) */

  printf("\n");

}/* main() */

ausdruck()
{
  short tmp_ausdruck = 0;
  short next_sym,operator;

  tmp_ausdruck = term();
  if (fehler != TRUE)
    {
    do
      {
      operator = getsym();
      switch (operator)
        {
        case PLUS  : tmp_ausdruck += term();
                     break;
        case MINUS : tmp_ausdruck -= term();
                     break;
        }/* switch() */
      }
```

C: Ein Taschenrechner

```
    while ((operator == PLUS) ||
           (operator == MINUS));
    if (fehler != TRUE)
      putsym();/* letztes Symbol zurueecklegen */
    }/* if (fehler != TRUE) */

  return(tmp_ausdruck);

}/* ausdruck() */

term()
{
  short tmp_term = 0;
  short next_sym,operator;

  tmp_term = faktor();
  if (fehler != TRUE)
    {
    do
      {
      operator = getsym();
      switch (operator)
        {
        case MAL   : tmp_term *= faktor();
                     break;
        case DURCH : tmp_term /= faktor();
                     break;
        }/* switch() */
      }
    while ((operator == MAL) ||
           (operator == DURCH));
    if (fehler != TRUE)
      putsym();/* letztes Symbol zurueecklegen */
    }/* if (fehler != TRUE) */

  return(tmp_term);

}/* term() */

faktor()
{
  short tmp_faktor = 0;
  short next_sym;

  next_sym = getsym();
  if (next_sym == KLA)
    {
    tmp_faktor = ausdruck()
    if (getsym() != KLZ)
      fehler = TRUE;
    }
  else if (next_sym == ZAHL)
         tmp_faktor = sym_wert;
  else fehler = TRUE;

  return(tmp_faktor);

}/* faktor() */
```

C: Ein Taschenrechner

```
getsym()
{
  short ret = NO_SYM;

  altptr = ptr;
  while (*ptr == ' ')
    ptr++;
  if (*ptr == '\0')
    ret = ENDE;
  else if (*ptr == '*')
         {
         ptr++;
         ret = MAL;
         }
  else if (*ptr == '/')
         {
         ptr++;
         ret = DURCH;
         }
  else if (*ptr == '+')
         {
         ptr++;
         ret = PLUS;
         }
  else if (*ptr == '-')
         {
         ptr++;
         ret = MINUS;
         }
  else if (*ptr == '(')
         {
         ptr++;
         ret = KLA;
         }
  else if (*ptr == ')')
         {
         ptr++;
         ret = KLZ;
         }
  else if ((*ptr >= '0') &&
                    (*ptr <= '9'))
         {
         sym_wert = toi();
         ret = ZAHL;
         }
  else {
       ret = NO_SYM;
       fehler = TRUE;
       }

  return(ret);

}/* getsym() */
```

C: Ein Taschenrechner

```
putsym()
{
  ptr = altptr;
}/* putsym() */

/*  toi()  wandelt  den  naechsten  zu  analysierenden  Teil  der
Zeichenkette in eine Zahl um und gibt diesen Wert als Ergebnis an
den Aufrufer zurueck. Analysiert wird bis zum 1. Zeichen, das
keine Ziffer sein kann. */
toi()
{
  short i = 0;

  while ((*ptr >= '0') && (*ptr <= '9'))
    i = 10 * i + *ptr++ - '0';

  return(i);

}/* toi() */
```

zum Schluss noch eine kleine Erweiterung:

Hiermit koennen auch Potenzen berechnet werden.

```
<Ausdruck> ::= <Term> { (+|-) <Term> }
<Term>     ::= <Faktor> { (*|/) <Faktor> }
<Faktor>   ::= <Potenz> { '^' <Potenz> }
<Potenz>   ::= <Zahl> | '(' <Ausdruck> ')'
<Zahl>     ::= [-] [Praefix] <Digit> { <Digit> }
<Praefix>  ::= %|$|o
<Digit>    ::= 0|1|2|3|4|5|6|7|8|9|A|B|C|D|E|F
```

Wie man sieht, sind Erweiterungen in allen Variationen moeglich und -relativ- einfach machbar.

T U R B O: Messwertdarstellung

Numerische Darstellung von Messwerten (Kurt-Bernd Rohloff, 8000)

Gemessene Daten sind stets mit einem Fehler behaftet. Die Darstellung numerischer Messwerte sollte daher auf eine "sinnvolle" Stellenzahl beschränkt werden. Beispielsweise ist die Angabe 123.456789 unsinnig, wenn der Fehler (üblicherweise die Standardabweichung einer statistischen Auswertung) bereits 4.6 beträgt. Daher habe ich eine PASCAL Prozedur geschrieben, die den Messwert (x) immer auf zwei fehlerbehaftete Stellen genau ausgibt. Ebenso wird der Fehler (dx) auf zwei signifikante Stellen ausgegeben. Für obiges Beispiel würde meine Prozedur Show_x_dx
1.234 +/- 0.046 E 2
ausgeben. Zwei fehlerhafte Stellen erschienen mir aus 3 Gründen sinnvoll:

a) Man hat noch die Freiheit, die Rundung auf eine fehlerhafte Stelle selbst vorzunehmen (oder auch nicht).
b) Muß mit dem Ergebnis noch weitergerechnet werden, können durch zu frühzeitiges Runden unakzeptable Fehler im Endergebnis entstehen.
c) Häufig geht im Fall b auch die Fehlergröße dx noch in Folgerechnungen ein (Fehlerrechnung!). Dann wäre eine Stelle zu wenig.

Die Prozedur versagt, wenn einer der beiden Werte x, dx gleich Null ist. Dann wird auf das standardmäßige REAL Ausgabeformat von TURBO-PASCAL umgeschaltet. Die Ausgabe wird nicht mit einem Zeilenwechsel abgeschlossen, so daß gegebenenfalls noch eine Maßeinheit o. ä. angehängt werden kann.

Der dritte Parameter ist eine Dateivariable, die das Ausgabegerät bestimmt. Gedacht ist hier in erster Linie an Bildschirm und Drucker, jedoch wäre auch Ausgabe in eine Textdatei möglich. Im aufrufenden Programm muß dazu eine Dateivariable vom Typ TEXT vereinbart werden, der dann via ASSIGN ein logisches Gerät zuzuordnen ist. Ein kleines Beispielprogramm (am Schluß) verdeutlicht dir am besten, wie's geht.

```
PROCEDURE Show_x_dx(x, dx : REAL; VAR device : TEXT);
{ Anzeige von x (Messwert) und dx (sein Fehler) auf
  zwei fehlerhafte Stellen genau.
  Die Ausgabe erfolgt auf das der Dateivariablen device zugewiesene Geraet.
}
   VAR      a1, a2, tmp       : REAL;
            b1, b2, f         : INTEGER;
   { normalisierte Darstellung: x=a1*10**b1 mit 1 <= a1 < 10
     dx=a2'*10**b2 = a2*10**b1, wobei a2=dx/10**b1
     dargestellt wird a1 +/- a2 E b1 wobei a1,a2 mit f=b1-b2+1
     Nachkommastellen ausgegeben werden
   }
   FUNCTION Lg(x : REAL) : REAL;
      { Zehnerlogarithmus }
      BEGIN
      LG:=0.43429448 * LN(X);
      END;
```

T U R B O: Messwertdarstellung

```
FUNCTION zehn_hoch( loga : INTEGER) : REAL;
   VAR      h, faktor          : REAL;
            i                  : INTEGER;
   BEGIN
   h:=1.0;
   IF loga <> 0
      THEN BEGIN
           IF loga > 0
              THEN faktor:=10.0
              ELSE faktor:=0.1;
           FOR i:=1 TO Abs(loga)
           DO h:=h*faktor;
           END;
   Zehn_hoch:=h;
   END;

FUNCTION Zehner(x : REAL) : INTEGER;
   { Bestimmen des Zehnerexponenten der normalisierten Darstellung
   }
   CONST    diff               =1.0E-4;
   VAR      tmp, rund          : REAL;
            h                  : INTEGER;
   BEGIN
   tmp:=lg(Abs(x));
   { naechst kleinere ganze Zahl bestimmen }
   IF tmp > 0
      THEN rund:=INT(tmp + 0.5)
      ELSE rund:=INT(tmp - 0.5);
   IF Abs(tmp - rund) <= diff
      THEN { tmp liegt schon nahe einer ganzen Zahl, nimm diese }
           h:=Round(rund)
      ELSE BEGIN
           h:=Trunc(tmp);
           IF tmp < 0
              THEN h:=Pred(h);
           END;
    Zehner:=h;
    END;

BEGIN { show_x_dx }
IF (x <> 0) AND ( dx <> 0)
   THEN BEGIN
        b1:=Zehner(x);
        b2:=Zehner(dx);
        f:=b1 - b2 +1;
        IF f < 0
           THEN f:=0;
        tmp:=Zehn_hoch(-b1);
        a1:=x*tmp; a2:=dx*tmp;
        WRITE(device, a1:f+4:f, ' +/- ', a2:f+4:f);
        IF b1 <> 0
           THEN WRITE(device, ' E ', b1);
        END
   ELSE WRITE(device, x,' +/- ',dx);
END; { of show_x_dx }
```

T U R B O: Messwertdarstellung

Beispielprogramm:

```
Program TESTXDX;
{ testen von show_x_dx }
VAR      p, q                : REAL;
         device              : TEXT;
{$I SHXDX.INC }
{$I INKEY.INC }
BEGIN
WRITE(#6, #2, 'Ausgabe auf ', #6, #4, 'B', #6, #2, 'ildschirm oder ',
                              #6, #4, 'D', #6, #2, 'rucker? ');
CASE In_Key OF
     'b', 'B' : Assign(device, 'CON:');
     'd', 'D' : Assign(device, 'LST:');
     ELSE       WRITELN;
                WRITELN('Bildschirm wird genommen');
                Assign(device, 'CON:');
END; { case }
Rewrite(device);
REPEAT
   WRITELN;
   WRITE('Enter x dx  ');
   READLN(p,q);
   WRITE(device, 'Ergebnis: ');
   Show_x_dx(p,q, device); WRITELN(device);
   WRITE('Weiter (j/n)? ');
UNTIL UpCase(In_Key) <> 'J';
END.
```

Die Include-Datei INKEY.INC sieht wie folgt aus:

```
FUNCTION In_Key:Char;
{ liest ein Zeichen von der Tastatur mit Echo }
BEGIN
    In_Key:=Chr(BDOS(1));
END;
```

L E S E R B R I E F: Holger Hansen

Holger Hansen
Scharnhorststr. 2
3300 Braunschweig

Braunschweig,12.6.87

Hallo Herbert!!

Ich habe meinen Memotech zwar erst seit ein paar Wochen, aber ich konnte schon ein bißchen damit arbeiten (trotz diverser Diskprobleme). Also habe ich mich auf Bernds RAM 4.2 Disketten umgeschaut und das Programm MAKE getestet (bzw erstmal ausgetüftelt, wie es einzusetzen ist). Am Schluß habe ich noch einen Patch für dBase 2.40 (englische Version) aus meinen Schneider CPC Zeiten, einen für SuperCalc 1.12 (beide bewirken, daß Overlays auf anderen,als dem Defaultlaufwerk gefunden werden)und einen für STOP.COM (STOP für RETURN statt 6/DEL Taste) beigefügt.

Last but not least eine Frage
Wie erreiche ich über die Tastatur den ASCII Code #31 (=1Fh) ?
(SAG NICHT <CTRL>_ , DA LIEGT BEI MIR #13 (=ODh))

Bis zur nächsten BYTESHIFT

Holger

PS. Entschuldige meine "mangelhaften orthographischen Fähigkeiten"
(O-Ton: Langner, Deutsch Lehrerin)

PPS. Zu MAKE ist übrigens in der neuen C't (7.87) auf Seite 146 ein Artikel erschienen

R A M 4 . x: MAKE.COM

Wozu ist MAKE.COM ? (Holger Hansen, 3300)

Der erste Tip ist leider nur was für diejenigen im Lande, die sowohl einen MTX/FDX haben, als auch Bernds Ram Vier-Zwo. Ich gehöre nämlich zu dieser Minderheit. Das Stöbern in den 6 Disketten von Ram 4.2, brachte diesen Tip zu Tage.
Auf der Diskette 3 von RAM fand ich einige nette kleine Programme, die mich verdammt an einige gleichnamige UNIX bzw MSDOS Programme erinnerten. (z.B. PWD, CD, MKDIR und eben MAKE)
Nun steht im Ram-Handbuch, daß Bernd nicht weiß, was MAKE tut bzw tun soll, deshalb dachte ich mir diese Lücke müßte doch eigentlich zu schließen sein.

Also Start des Unternehmens **MAKE:**

Frage: Wozu dient MAKE ?

MAKE hat die Aufgabe in Abhängigkeit vom Zeiteintag Befehle auszuführen.
Im Klartext :

MAKE [-vz] [filename]

veranlaßt MAKE in der Steuerdatei **filename** (voreingestellt ist der Name MAKEFILE) nachzuschauen und die dortigen Befehle auszuführen. Das hört sich zwar nach einem ganz ordinären Submitprozessor an, ist aber ein bißchen cleverer.

Anm.d.HzN: Die eckigen Klammern liegen bekanntlich auf den Tasten:
Ä (auf) und Ü (zu).

Frage: Wie erstelle ich eine Steuerdatei ?

Die Steuerdatei hat folgenden Aufbau (Eingaben mit ; sind Kommentare, Erstellung mit Texteditor im NonDoc Modus)

```
;Steuerdatei

;Ziel              Quelle1 Quelle2 Quelle3 .. QuelleN
;                  Kommando
;-------------------------------------------------------------------------
MAIN.COM           MAIN.REL INCLUD1.REL
                   L80 MAIN,INCLUD1,LIBRARY,MAIN/N/E
MAIN.REL           MAIN.FOR
                   F80 =MAIN/z
INCLUD1.REL        INCLUD1.FOR
                   F80 =INCLUD1/z
;-------------------------------------------------------------------------
;Ende der Steuerdatei
```

Das Beispiel zeigt einen Übersetzungsvorgang für Fortan, bei dem aus den Dateien MAIN.FOR (Hauptprogramm) und INCLUD1.FOR (Unterprogramme), die ausführbare Datei MAIN.COM erzeugt wird. F80 ist der Compiler, L80 ist der Linker und Library ist die Fortranbibliothek.
MAKE vergleicht nun den Zeiteintrag von INCLUD1.REL mit dem von INCLUD1.FOR, wenn der Zeiteintrag von INCLUD1.FOR neuer ist, dann wird F80 =INCLUD1/z ausgeführt, wenn nicht wird der Befehl übersprungen und die nächste Zeile abgearbeitet.
(Achtung: Manche Programme werten Parameter hinter dem Namen aus, in solchen Fällen darf das Einrücken der Befehlszeile nur durch Tabs erfolgen!!)

R A M 4 . x: MAKE.COM und d B A S E: Patch

Nun werden MAIN.REL und MAIN.FOR entsprechend verglichen und zum Schluß MAIN.COM mit MAIN.REL & INCLUD1.REL, wenn eins von beiden neuer ist, dann wird entsprechen der Befehl L80 ... ausgeführt. Es wird also immer das Ziel mit der (den) Quelle(n) verglichen und bei einer aktuelleren Quelle der Befehl in der nächsten Zeile ausgeführt.
Dabei wird die Steuerdatei vom Ende zum Anfang hin abgearbeitet.
Bei den Optionen bin ich mir nicht ganz klar, was sie bedeuten, b.z.w. ich weiß nicht,ob sie nicht noch mehr Bedeutungen haben.
Die Option **-v** protokolliert die Ausführung auf dem Bildschirm, über die Funktion des Parameters **-z** von MAKE bin ich mir noch nicht sicher.

Frage: Was ist zu beachten ?

Natürlich funktioniert MAKE nur, wenn die Directories durch INITDIR mit Zeiteinträgen versorgt worden sind. (Was RAM 4.X voraussetzt)

Frage: Wozu kann man MAKE brauchen ?

Eine nützliche Anwendung, außer bei Compilierungen, wäre es, eine Steuerdatei zu schreiben, die jeweils nur die neusten Programmversionen von Ramdisk auf Diskette zurückkopiert.

Nun für alle FDX und dBase Besitzer: (Holger Hansen, 3300)

Ein Patch für dBase 2.40 engl.:

```
 (HEX)  4431:  00 44 42 41 53 45 4F 56 52 43 4F 4D
               ^
               ^--hier muß  1 für Laufwerk A
                            2 für Laufwerk B u.s.w. hin

        4452:  00 44 42 41 53 45 4F 56 52 43 4F 4D
               ^
               ^--wie oben
```

Mit diesen Patches sucht dBase seine Overlaydatei DBASEOVR.COM auf dem aktuellen Laufwerk (=0) oder entsprechend auf Laufwerk A-I (1..9).
Der Patch ist leicht mit DDT oder ähnlichen Tools durchzuführen:
(^m bedeutet RET Taste betätigen; im Bspl. wird Lfwk H: gewählt)
(Fettdruck bedeutet eigene Eingaben)

```
A>DDT DBASE.COM^M
NEXT PC 4D00 100
-s4331^m
 4331 00 08^m
 4332 44 .^m
-s4352^m
 4352 00 08^m
 4353 44 .^m
-g0^m
A>save 76 dbneu.com^m
A>
```

Die Änderung für die DBASEMSG.TXT Datei ist da schon aufwendiger.
Zuerst muß der Name um 2 Buchstaben verkürzt werden,(z.B. auf DBAMSG.TXT), weil in die Datei DBASEOVR.COM vor den Namen der Helpdatei noch Laufwerksbuchstabe und Doppelpunkt eingefügt werden müssen.
In diesem Beispiel Laufwerk H:.

d B A S E: Patch und S u p e r C a l c: Patch

Danach wird mit DDT oder ähnlichen Werkzeugen die Datei DBASEOVR.COM geladen.

```
        (HEX)  8969:       'D  B  A  S  E  M  S  G '
                      alt   44 42 41 53 45 4D 53 47
                      neu   42 3A 44 42 41 4D 53 47
                           'H  :  D  B  A  M  S  G '

               89E6:  alt     ---  wie oben  -----
                      neu     ---  wie oben  -----
```

```
A>DDT DBASEOVR.COM
NEXT PC 9E00 100
-s8969^m
 8969 44 48^m
 896A 42 3A^m
 896B 41 44^m
 896C 53 42^m
 896D 45 41^m
 896E 4D .^m
-s89E9^m
 89E9 44 48^m
 89EA 42 3A^m
 89EB 41 44^m
 89EC 53 42^m
 89ED 45 41^m
 89EE 4D .^m
-g0^m

A>save 157 dbaseovr.com^m
A>
```

Der Patch müßte analog auch bei allen anderen dBase Versionen funktionieren (bis auf eventuell unterschiedliche Adresslage).

Nun für fast alle FDX und SC Besitzer: (Holger Hansen, 3300)

Der nächste Patch ist für alle FDX Besitzer mit mindestens 2 Laufwerken (egal ob Ramdisk oder "echte" Laufwerke):

Patch für Supercalc 1.12:
veranlaßt, daß SC seine Overlays auf Laufwerk A:-I: (je nach Wahl) sucht. Nach bewährten Muster wird SC mit DDT, DDTZ etc behandelt.

```
A>DDT SC.COM^m
NEXT PC 6080 100
-s5B3E^m
 5B3E CD 00^m
 5B3F F6 3E^m
 5B40 51 08^m
 5B41 11 .^m
-g0^m

A>save 95 SC.COM^m
A>
```

S u p e r C a l c: Patch und R A M 4 . x: STOP.COM

Kurz erzählt was ich gemacht habe. An Adresse 5B3E(HEX) steht ein Unterprogrammaufruf, der über den BDOS Call 19(HEX) das Bezugslaufwerk ermittelt. Bdos 19 übergibt dabei im Akkumulator den Drivecode-1, sodaß dieser Wert für den FCB noch um eins erhöht werden muß. In dem Unterprogramm muß also ungefähr dies stehen

```
"LD C,19 / CALL 0005      / INC A              / RET"
 BDOS Nr   BDOS ausführen   A um eins erhöhen  zurück
```

also habe ich den Unterprogrammaufruf rausgenommen und die Anweisung LD A,8 eingefügt.(8, weil bei mir die Overlays in Laufwerk H: stehen A=1,B=2,C=3,..,I=9)
Wenn man den Namen der Overlays ändern will, dann muß man ihn ab Adresse 018F(HEX) eintragen (Großbuchstaben! max 8 Zeichen! mit Leerzeichen (20(hex)) auf 8 Buchstaben auffüllen). Der Filetyp steht ab Adresse 5C5D(HEX). Bei mir nacheinander "OVLHLP" für Overlay und Help Datei.

Schließlich für alle RAM 4.x Besitzer: (Holger Hansen, 3300)

Der letzte Patch ist wieder für Vier-Zwo Besitzer, die sich an die RET Taste gewöhnt haben:

Patch für STOP.COM

```
DDT STOP.COM^m
NEXT PC 200 100
-s101^m
 101 F7 80^m
 102 D3 .^m
-s106^m
 106 06 05^m
 107 FB FB^m
 108 47 77^m
 109    .^m
-g0^m

A>save 1 stop.com^m
A>
```

Ab jetzt reagiert STOP nicht mehr auf die 6/DEL Taste, sondern (wie es sich gehört) auf die RET Taste.

Zum Schluß kann ich jedem nur empfehlen sich RAM 4.2 zuzulegen. Durch dieses Programmpaket wird der MTX/FDX so komfortabel, daß er jeden MSDOS Rechner (in Sachen Komfort) vor Scham erröten läßt.

Weiterhin frohes Schaffen

Holger

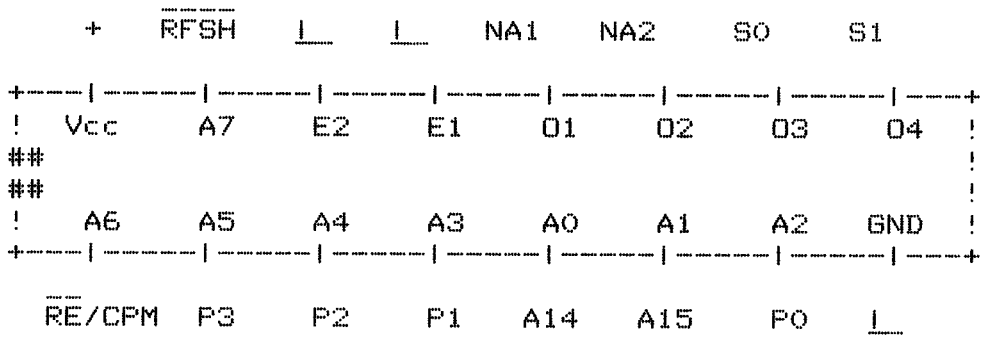
H a r d w a r e: PROM-Inhalt der MTX-RAM-Erweiterung

Prom auf Memotech-Speicherkarte (Herbert Oppmann, 8522)

Anm.d.HzN: Dieser längst versprochene Artikel liegt schon einige Zeit bei mir, und erscheint nun auch endlich in einer von mir überarbeiteten Version.

Prom-Typ: Texas TBP24S10 oder Valvo 82S129 ... 256 * 4 Bit

Belegung:



Inputs:

P0 - P3	:	Auswahlleitungen für Ram-Banks (16)
RE/CPM	:	1 bei CP/M-Betrieb (nur Ram), 0 bei Rom enabled
RFSH	:	CPU-Refresh über zwei Inverter
A14,A15	:	Die beiden höchstwertigen Adressleitungen

Outputs:

NA1,NA2	:	Modifizierte A14,A15 für Rams
S0	:	Select für linke Rambank (IC 7 - 14)
S1	:	Select für rechte Rambank (IC 15 - 22)

Aufbau der Tabelle:

Die erste Hälfte des PROM-Inhalts besteht nur aus 0FH, denn wenn RFSH Low (aktiv) ist, müssen beide Rambanks refreshed werden. Daher ist die erste Hälfte weggelassen. Außerdem wird jeweils nur das untere Nibble angegeben, da nur 4 Bit gespeichert werden. Die zwei Ziffern ganz links geben die Adresse des ersten Nibbles der jeweiligen Zeile an.
Da auf der Karte maximal 128kB RAM untergebracht werden können (512kB werden mit einer geänderten Schaltung verarbeitet), sind alle Speicherstellen von A0 - BF und E0 - FF Null. Daher werden auch diese zwei Zeilen unten nicht mit angegeben.
Jedes Nibble bestimmt für ein Segment, ob und welcher Speicherbereich des Zusatzspeichers aktiv werden soll. Also sind je vier Nibbles für eine Bank zuständig (daher der Zwischenraum).

H a r d w a r e: PROM-Inhalt der MTX-RAM-Erweiterung

MTX500, 2 Banks mit 3732L, d.h. 64kB-Karte

Inhalt:	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	
80	0 5 0 0	0 8 4 0	0 0 9 0	0 0 0 0	$\overline{RE}$/CPM = 0
90	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(ROM-Betrieb)
C0	4 5 0 0	8 9 0 0	0 0 0 0	0 0 0 0	$\overline{RE}$/CPM = 1
D0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(CPM-Betrieb)

MTX500, für linke Bank mit 3732L und rechte Bank mit 3732H: 64kB

Inhalt:	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	
80	0 5 0 0	0 A 4 0	0 0 B 0	0 0 0 0	$\overline{RE}$/CPM = 0
90	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(ROM-Betrieb)
C0	4 5 0 0	A B 0 0	0 0 0 0	0 0 0 0	$\overline{RE}$/CPM = 1
D0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(CPM-Betrieb)

MTX500, für 2 Banks mit 4164: 128kB

Inhalt:	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	
80	0 5 0 0	0 7 4 0	0 9 6 0	0 B 8 0	$\overline{RE}$/CPM = 0
90	0 0 A 0	0 0 0 0	0 0 0 0	0 0 0 0	(ROM-Betrieb)
C0	4 5 0 0	6 7 8 0	9 A B 0	0 0 0 0	$\overline{RE}$/CPM = 1
D0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(CPM-Betrieb)

MTX512, 2 Banks mit 3732L, d.h. 64kB-Karte

Inhalt:	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	
80	0 0 0 0	0 5 0 0	0 7 4 0	0 0 6 0	$\overline{RE}$/CPM = 0
90	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(ROM-Betrieb)
C0	0 0 0 0	4 5 6 0	7 0 0 0	0 0 0 0	$\overline{RE}$/CPM = 1
D0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(CPM-Betrieb)

MTX512, für 2 Banks mit 4164: 128kB

Inhalt:	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	
80	0 0 0 0	0 5 0 0	0 7 4 0	0 9 6 0	$\overline{RE}$/CPM = 0
90	0 B 8 0	0 0 A 0	0 0 0 0	0 0 0 0	(ROM-Betrieb)
C0	0 0 0 0	4 5 6 0	7 8 9 0	A B 0 0	$\overline{RE}$/CPM = 1
D0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	(CPM-Betrieb)

H a r d w a r e: 8 MegaHertz

Warum 8 MHz ? ... Warum eigentlich nicht ? (Herbert zur Nedden, 2000)

Kurz vorab:
Mein MTX/FDX-System läuft (außer bei Diskettenzugriffen und beim Booten) mit völlig ungebremsten Wait-freien **ACHT MHz**, auch im KLICK - und ich kann bei laufendem Computer jederzeit zwischen 4 und 8 MHz umschalten. Dabei ist die Hauptplatine und ein kräftiges Netzteil in der FDX, der ECB-Bus allerdings ist nicht in der FDX.

Nun die Knackpunkte:
Nach einigem Sträuben seitens meines MTX war er bereit im Grundgerätemodus mit 8 MHz zu laufen. Dazu mußte ich 'nur' das RAM-Timing ändern, 150 ns RAMs und 250 ns EPROMs einsetzen. Obendrein mußte als Taktgenerator-IC ein 74HC04 (kein 74S04) verwendet werden.
Leider habe ich feststellen müssen, daß es eine Gnade ist, wenn eine Hauptplatine 8 MHz ohne WAIT verkraftet - ich habe 3 Stück getestet, und nur eine wollte. Mit WAIT verbleiben aber immernoch ca. 7 MHz.

Die 80-Zeichenkarte mußte geringfügig geändert werden, um ein Flip-Flop, welches den Zugriff auf den Controller bremste zu überbrücken.

Der ECB-Bus mit allen Karten (RAM-Floppies ...) machte keine Schwierigkeiten - oh Wunder.

Der Floppycontroller war auch nach langem Versuchen (Ändern des Timings durch Umbrennen des PROMs) nicht bereit sicher mit 8 MHz zu arbeiten.

Lösung:
Das Booten mit 8 MHz kann auch nicht funktionieren, da das Boot-EPROM auf der Buskarte der FDX sitzt! Aber mit einem Wait beim Befehls-Zyklus der Z80 genügte, um zusammen mit einem 200 ns Boot-EPROM zu arbeiten.

Tja, bis auf den Floppycontroller klappte alles.

Da ich aber unbedingt auch mal auf 4 MHz herunterschalten können wollte - sonst will der VS 4, also einige Spiele und mein (E)PROM/PAL-Programmiergerät nicht so recht lag es nahe, bei Diskettenzugriffen auch mit 4 MHz zu arbeiten. Das ist auch nicht weiter tragisch, da die Verteilung der Sektoren auf der Diskette für 4 MHz ausgelegt ist, also mit 8 MHz fast garnicht schneller ist.

Und das war auch schon alles!!!!

Erläuterung der WAIT-Generator-Schaltung:

1. Die eigentliche Schaltung ist aus einem Z80-Handbuch übernommen, und funktioniert sogar. Der Z80-Befehlszyklus ist dann aktiv, wenn das sog. M1-Signal auf Low geht.

2. Die Abhängigkeit von Booten wird einfach durch das Memotech-Signal RE/CPM festgestellt, welches im ROM-Betrieb immer auf 0 Volt ist. Ist RE/CPM auf 5 Volt, so werden keine WAITs generiert.

3. Genauer will ich auf diese Schaltung nicht eingehen.

H a r d w a r e: 8 MegaHertz

Erläuterung der Frequenzumschalt-Schaltung:

1. Mit einem FlipFlop (74HC74) teile ich den 8 MHz-Mastertaktauf 4 MHz, um auch diese Frequenz zu haben. Die 8 MHz verzögere ich über zwei Gatter (74HC00), damit die Takt-Signale phasengleich sind, d.h. sauber ineinander leigen.

2. Die beiden Frequenzen gebe ich auf einen Multiplexer (74HC157), über dessen Select-Eingang angesteuert wird, welche Frequenz die CPU erreicht.

3. Um nicht im falschen Moment umzuschalten, schalte ich die Frequenz nur dann tatsächlich um, wenn die 4 und die 8 MHz beide auf 5 Volt liegen. Also wird dieser Vergleich mit einem Nand-Gatter (74HC00) erledigt, über ein weiteres sicherheitshalber einige Nanosekunden verzögert. Das so erhaltene Umschalt-Enable-Signal gebe ich auf ein FlipFlop (74HC74) als Umschalttakt, und dieses FlipFlop steuert den o.g. Multiplexer.

4. Fehlt also nur noch die Schaltung, die entscheidet, welche Frequenz tatsächlich ausgewählt werden soll. Dafür benötige ich zum einen meinen Schalter, um von Hand 4 MHz zu erzwingen, und zum anderen ein Signal, welches mir sagt, daß ein Laufwerk angesprochen wird. Als letzteres nehme ich das Signal, welches auf dem Floppy-Controller das Select für die Laufwerke freigibt.
   Damit die Chose nicht prellt, habe ich ein MonoFlop (72LS123), d.h. einem FlipFlop, welches nach einer bestimmten Zeit von alleine in den Ruhezustand zurückschaltet eingebaut. Dieses MonoFlop hat als Dateneingang die beiden Frequenzauswahl-Signale (Schalter und Diskettenzugriff, die über ein Oder-Gatter (74LS02) zusammengeführt werden). Damit das MonoFlop regelmäßig seinen Dateneingang untersucht wird es mit 8 MHz getaktet.

Was ist zu tun:

Auf der MTX-Hauptplatine:

1. IC 9D muß ein 74HC04 sein. R35 und R36 müssen fehlen, und R32, R33 müssen je 3,3 kOhm haben.
2. IC 9D Pin 6 und IC 9E (74LS193) Pin 4 müssen über der Platine abgekniffen und hochgebogen werden.
3. XTAL1 muß ein 8 MHz-Quartz sein
4. Kondensatoren C5 und C6 entfallen
5. Die acht ICs 2D-5D, 2C-5C müssen für MTX 500 3732, und für MTX 512 4164 mit 150 ns Zugriffszeit sein - oder der WAIT muß immer bei 8 MHz generiert werden, d.h. der 74LS00 entfällt, und der Pin 1 des zugehörigen 74LS74, muß mit Pin 1 des 74LS175 verbunden werden.
6. Die BASIC-EPROMs müssen 350 ns schnell sein. Das ist i.a. der Fall.
7. Z80-Bausteine: Z80 H CPU, B CTC, B DART

Auf der 80-Zeichenkarte

1. IC 1C (74LS74) Pin 5 über der Platine abkneifen, und unter der Platine Pin 5 mit Pin 8 dieses IC verbinden.

Netzteil

1. kräftig genug ? Evtl. verstärken oder gar wie ich ersetzen.

H a r d w a r e: 8 MegaHertz

Fliegende Verdrahtung:

1. Schaltungen der beiden folgenden Seiten.
   Die Stromversorgung der IC's:
   14-polige: Pin 7 = Masse, Pin 14 = 5 Volt,
   16-polige: Pin 8 = Masse, Pin 16 = 5 Volt,
   Zusätzlich sollten noch für die beiden Schaltungen jeweils ein 100 nF Kondensator zwischen 5 Volt und Masse direkt bei den ICs gelötet werden.
2. Die acht in Kreisen angegebenen Anschlüsse sind wie folgt anzuschließen:

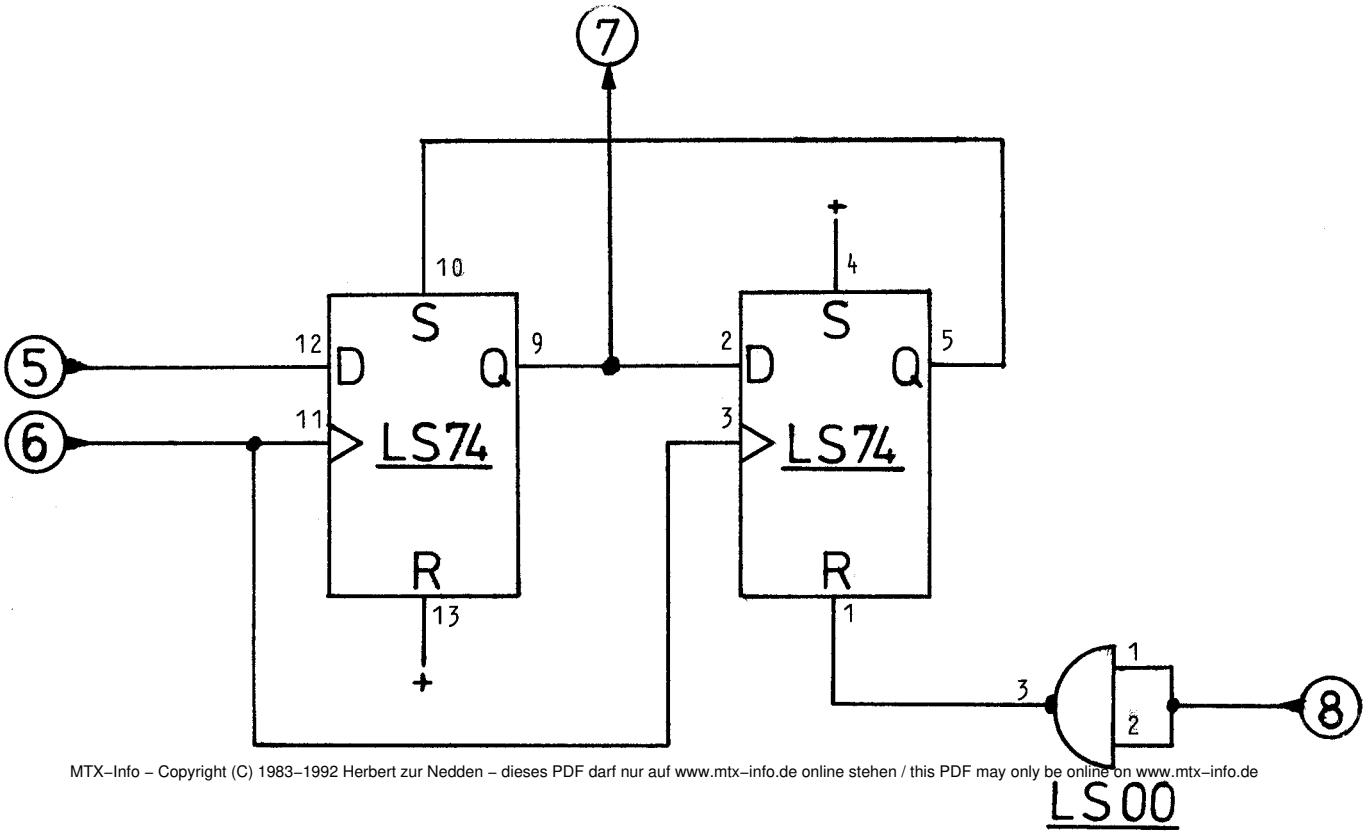
1 an Pin 8,	IC D3 auf dem Floppycontroller	(74LS32)
2 an Pin 6,	IC 9D auf der Hauptplatine	(74HC04)
3 an Pin 4,	IC 9E auf der Hauptplatine	(74LS193)
4 an Kontakt unter Pin 6,	IC 9D auf der Hauptplatine	(74HC04)
5 an Pin 27,	IC 10A auf der Hauptplatine	(Z80 CPU)
6 an Pin 6,	IC 10A auf der Hauptplatine	(Z80 CPU)
7 an Pin 24,	IC 10A auf der Hauptplatine	(Z80 CPU)
8 an Pin 13,	IC 6A auf der Hauptplatine	(PAL 14L4)

3. Den WAIT-Generator kann man am besten auf die CPU, die Frequenzumschaltung über dem HC04 anbringen.

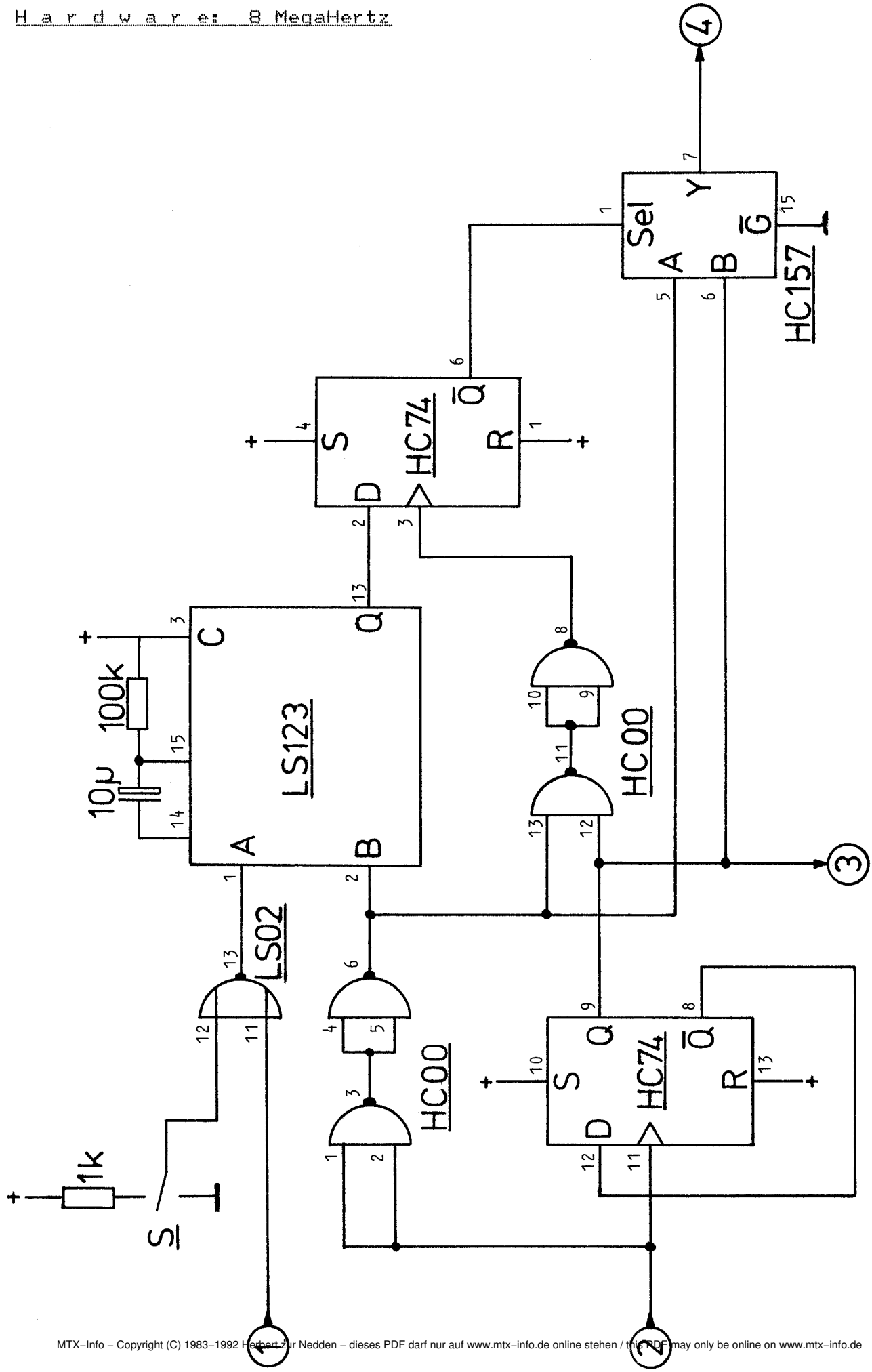
Übrigens:

1. Die RS232 wird mit 4 MHz vorgetaktet (über den CTC), damit die Initialisierung der RS232 nicht geändert werden muß!!!
2. Die als 74HCxx genannten ICs sollten auf jeden Fall auch solche sein, damit die Flanken besser sind!
3. Der 100k Widerstand ist gemäß Texas-Instruments-Applikations-Datenbuch zu groß, da max. 40k erlaubt sind. Es läuft trotzdem!

A-B-E-R
Es ist nicht gesagt, daß diese Lösung so einfach, und ohne WAIT (außer beim Booten) auf anderen Memotechs funktioniert!!!!!!!



Hardware: 8 MegaHertz



H a r d w a r e: Umbaustory

Anm.d.HzN: Bitte entschludigt ewatige Schreibfehler in dem nun folgenden Text. Ich hatte einfach Michaels Ausdruck in das Info mittels Schrere und Klebstoff übernommen, aber gerade (17.20 Uhr) in der Druckerei erfahren, daß der Kontrast schon fast zu schwach ist. Da aber zusätzlich die Farbe innerhalb einzelner Zeichen sehr start schwankt, d.h. oben ist der Buchstabe halbdunkel, und unten hell konnte von der Vorlage keine Druckplatte gemacht werden, und eine Fotokopie wäre wegen des schlechten Kontrastes ein graues Blatt.
Fazit: Ich habe den Text rasch abgeschrieben, ausgedruckt, und bis 18.00 Uhr in die Druckerei gebracht.

Bitte schickt mir, wenn Euer Druckerfarbband zu schwach auf der Brust ist einfach eine Diskette.

Umbaustory (Michael Kessler, 5600)
Ich hatte mich vor einiger Zeit entschlossen, die Spannungsrtregler-Bräter von der Hautplatine runterzuwerfen und die Platine stattdessen vom FDX-Schaltnetzteil mitversorgen zu lassen. Wo ich denn eh gerade dabei war, habe ich kurzentschlossen die ganze Kiste umgekrempelt. Ergebnis: Hauptplatine in der FDX, RS232 und 512k sind drin, zwei Laufwerke in der FDX, Original-Netzteil in der FDX + zweites Netzteil, alle Karten sind auch noch da, ebenso der Ventilator, alle Ein- und Ausgänge in der FDX, ein kleiner Lautsprecher für den Systempieps und zusätzlich noch freier Platz für ECB-Option mit mindestens 4 (freien) Slots oder sonstige Einbauten wie z.B. zwei weitere Laufwerke oder Festplatte. Allerdings muß man ein wenig mit Säge, Feile und Bohrmaschine umgehen können, um den Umbau durchzuführen. Die beiden Skizzen, die ich meinem Schreiben beilege, zeigen einmal die FDX mit abgenommener Frontpartie, zum anderen die Rückansicht mit den Änderungen am hinteren Gerätedeckel.

Kurze Erläuterung zu meinem Umbau:

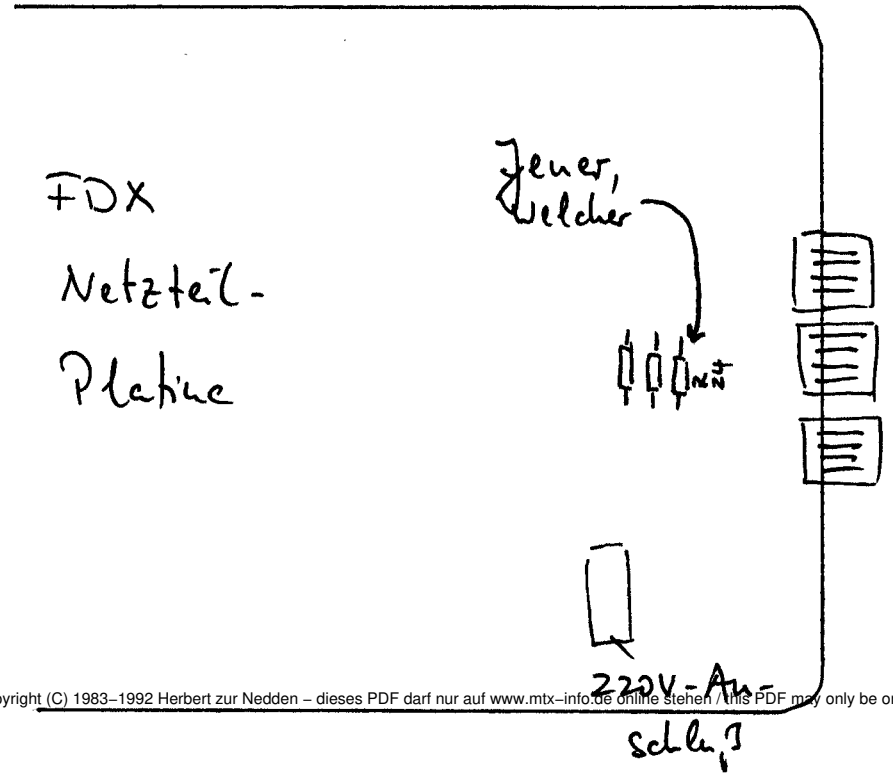
1. der serienmäßige Laufwerksträger wurde entfernt und durch eine neue Konstruktion ersetzt (gebogen aus ca. 1,5 mm feuerverzinktem Blech).
2. das serienmäßige Schaltnetzteil wurde mit seiner Längsseite unmittelbar neben der rechten Seitenwand montiert.
3. der Kartenkäfig wurde bis auf die drei unteren Slot abgesägt und sitzt nun links neben dem Schaltnetzteil unterhalb des linken Laufwerks.
4. Die Hauptplatine mit 512k-Karte und RS232-Karte wurde mit der Bestückungsseite nach unten unterhalb des Gehäusedeckels angebracht. Hierzu mußte der Deckel nicht zu Befestigungszwecken durchbohrt werden, vielmehr habe ich in die U-förmigen Längsverstrebungen des Deckels mittels eines Gewindeschneiders die zur Befestigung nötigen Gewinde geschnitten.
5. Da die Hautplatine nunmehr durch das FDX-Netzteil versorgt wird, wurde der Strom der FDX-Karten etwas kanpp (manchmal Bootprobleme). Deshalb habe ich links neben dem Laufwerksträger ein zweites Schaltnetzteil hochkant montiert. Dieses kann aber entfallen, wenn man statt des Original FDX-Netzteils ein stärkeres, aber etwa gleich großes Netzteil auftreiben kann.
6. In die Rückwand der FDX habe ich folgende Buchsen montiert: monochrom Monitor, Farbmonitor, VS4, HiFi, Centronics (D 25), Port 7 (D 25), SIO1 (D 25), SIO 2 (D 25), Joystick1, Joystick2, 8"-Drive-Bus, Tastatur (D 25).

H a r d w a r e: Umbaustory

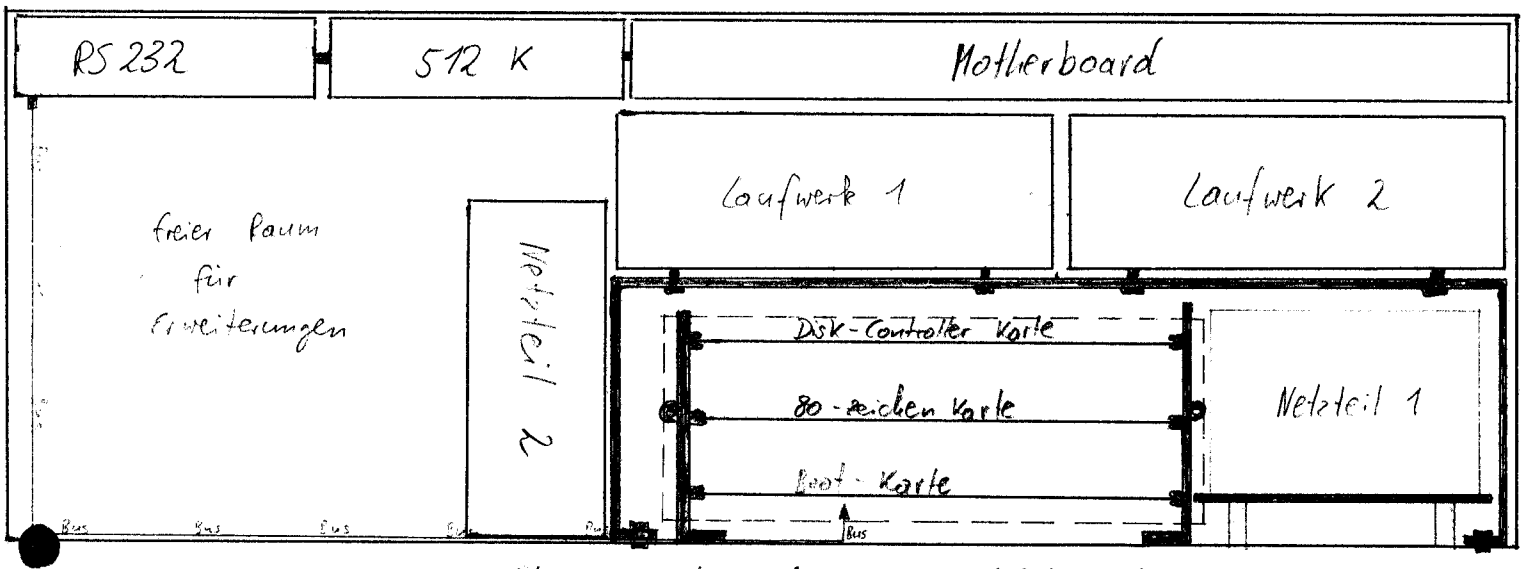
Fazit: Der Umbau stellt die bislang dichteste Packung dar. Trotz zusätzlicher Komponenten bleibt mindestens 1/4 der FDX völliog leerer Raum, hier kann eine ECB-Erweiterung mit 5-6 Slots installiert werden, wovon dann mind. 4 für Anwendungszwecke frei bleiben. Dank des verbliebenen Ventilators ist auch in dieser kompakten Anordnung die Kühling stets gewährleistet. Zusätzlich habe ich im Innenraum der FDX noch ein Bißchen aufgeräumt, die 110 Volt für den Ventilator werden jetzt übersichtlich an einer Klemmleiste angeliefert, der zugehörige Trafo sitzt jetzt hinter der frontseitigen Gehäuseschale. Dadurch wird wird hinter den Laufwerken/FDX-Karten genügend Raum zur Verlegung der dieversen Kabel frei.

Anm.d.HzN:

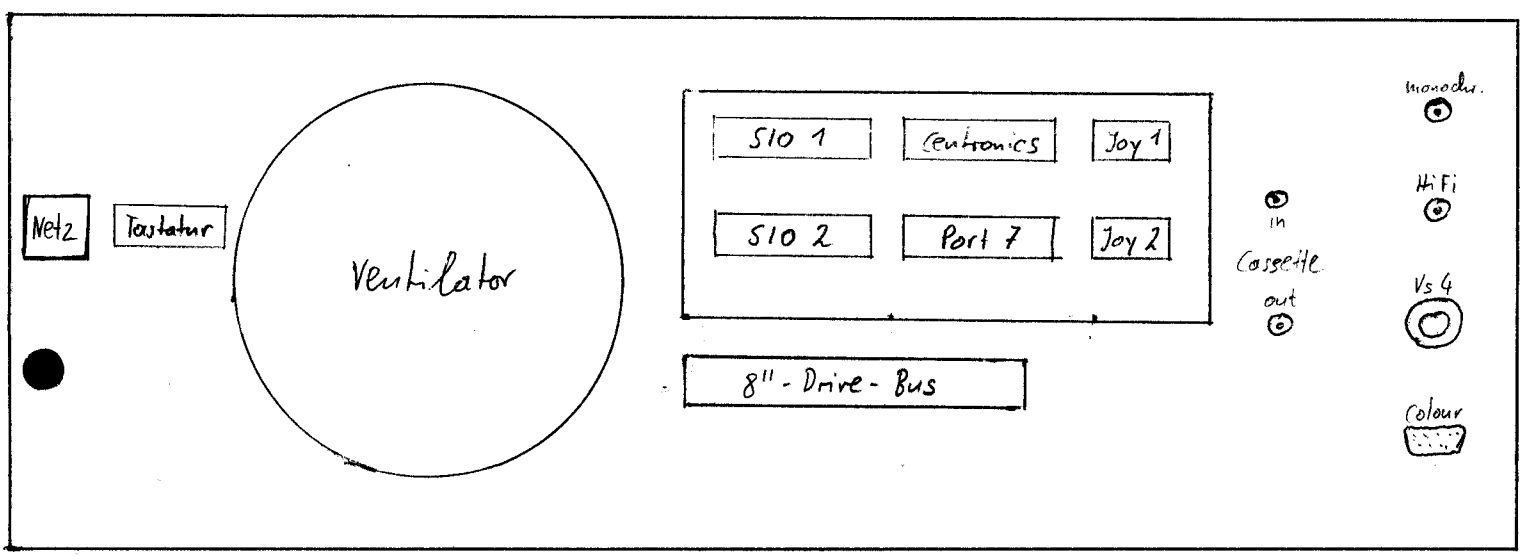
1. Die im Info angebotene ECB-Option ist nur ein kleines IC! Für den eigentlichen ECB-Bus wird zusätzlich der ECB-Adapter von Uwe Grass benötigt.
2. Ich frage mich, ob es nicht besser gewesen wäre, den FDX-Slot-Träger ganz an den rechten FDX-Rand zu setzen, damit das Schaltnetzteil dann besser im Luftstrom des Ventilators sitzt.
3. Man kann übrigens mit einem 10 kOhm Widerstand, und einem in Serie gelöteten 100 kOhm-Poti das FDX-Originalschaltnetzteil nachregeln!!!
   Dazu löte man die beiden o.g. Bauteile in Serie, d.h. hintereinander. So hat man also ein Poti von 10 - 110 kOhm. Dieses stelle man auf **110 kOhm** ein, und löte es an die beiden Enden des folgenden Widerstandes auf dem FDX-Netzteil.
   Bezeichnung **R24**; es ist der kleine Widerstand von dreien, die am dichtesten zu dem mittleren Ausgang des Schaltnetzteiles sitzen, der am nächsten zu diesem Anschluß ist. D.h. der der drei kleinen Widerstände, der am dichtesten am mittleren Ausgang des Netzteils sitzt.
   Nun (bei 110 kOhm-Stellung) alles an das Netzteil dranhängen, und Rechner anschalten. Mit einem Voltmeter die Spannung auf einer der FDX-Karten messen, und wenn sie unter 5Volt ist durch drehen des Poti versuchen hochzuregeln. **ACHTUNG: Hochspannung am Schaltnetzteil!**



Hardware: Umbaustory



so sieht meine FDX mit abgenommener Frontplatte aus !



Anschlüsse auf der Rückseite der FDX

H a r d w a r e: Tips

Joystick-Anschluß an Tastatur nach dem Umbau

(Herbert zur Nedden, 2000)

Bei den Joysticks ist es so, daß **eine** gemeinsame Leitung für alle Tasten, und fünf weitere je Taste vorhanden sind. Das selbe kannst Du bei der Tastatur feststellen, wenn Du die Tasten, die einem Joystick entsprechen verfolgst. Ein Joystick liegt auf den Pfeil- und der HOME-Taste, der andere auf YCBM und Space. Diese haben auch jeweils eine gemeinsame Leitung, die mit Pin 8 verbunden wird, und die anderen Kontakte der Tasten mit Pins 1-4, 6 der Joysticks.

Pin 8 = gemeinsam
Pin 6 = Feuer
Pins 1 - 4 = Richtungen

Meine FDX war so laut wie ein Computer (Herbert zur Nedden, 2000)

Tja, das leidige Übel. Was tun gegen den Krach. Uwe Grass hat da eine genial einfache und Pragmatische Lösung gefunden: Er hat den Ventilator einfach ausgebaut - aber auch das Netzteil an die frische Luft gesetzt. Das ist sicherlich nicht jedermanns Sache. Jedenfalls kann ich mich mit dieser Lösung irgendwie nicht anfreunden!
Bei Völkner gibt es für ca. dreißig deutsche Märker einen leiseren 12-Volt Lüfter, der 'Brushless', d.h. bürstenlos ist. Das hat den Geräuschpegel etwas gesenkt.
Aber die Idee von Michael Schlüter: Er hat die FDX-Seitenwände innen mit dünnen Schaumstoff wellig beklebt, und zwar bis zum Rand. Beim Zusammemschrauben wird dann der Schaumstoff zwischen den Seitenwänden der FDX und den anderen Blechen etwas eingeklemmt. Aber das hat den Geräuschpegel tatsächlich drastisch gesenkt. Die Seitenbleche sind durch ihre Dünne gute Schwingkörper!
Wenn Ihr jetzt noch zwischen die Deck-/Bodenplatten der FDX und die Vorder-/Rückwandschalen etwas Schaumstoff klemmt - ja, das ist etwas friemelig -, dann scheuern diese auch nicht mehr gegeneinander.

Portdekodierung (Herbert zur Nedden, 2000)

Horst Kupka (4019) ist dabei das Programm MUSICRAFT von den SIG/M-Public-Domain Disketten zum Laufen zu bringen, und kann schon einige Erfolge nachweisen. U.a. hat er mir im Zusammenhang damit vorab einen Schaltplan für eine flexible Port-Dekodierung mit Pufferung der Datensignale geschickt, der auf der nächsten Seite abgebildet ist.
Die anderen Dinge die er herausgefunden hat folgen.

Mit den vier Schaltern werden die Adreßbits A3-A6 festgelegt, und damit auch die dekodierten Portadressen. Adreßbit A7 muß High sein (daher I/O-Ports ab 80 Hex), und A0-A2 gehen auf Binärdekoder, die je acht Selektionsausgänge für IN und OUT liefern.
Also muß mit den Schaltern 1-4 'ADRESSE' nur festgelegt werden, welchen Bereich von Adressen man haben möchte.

Beispiel:
Wenn A4=Low, A3,A5,A6=High, dann werden die Adressen Hex 90-97 dekodiert.

Hardware: Portdekodierung