MTX User-Club Deutschland

Info 22
01.09.1987

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
12. September bis 12. Oktober keine Sprechstunde

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

Entschuldigung

Leider konnten wir nicht alle Bestellungen von MTX-Staubschutzhauben zu DM 11.- erfüllen. Karl-Heinz Harter hatte nur noch neun Stück, die in der Reihenfolge des Bestellungseinganges vergeben wurden.

Preis für dieses Info: DM 9,50

Redaktionsschluß für Info 23: 30. Oktober 1987

Liebe Leserinnen, liebe Leser,

ab 12. September bis Mitte Oktober bin ich in **Urlaub**, und damit folglich nicht erreichbar. Fragen und Bestellungen an mich müssen daher leider etwas warten.

Die Software der **EDICTA**-Grafikkarte ist mittlerweile sogar um schnelle Umrechnungsroutinen für 3D-Grafiken erweitert worden. Einen Würfel in Echtzeit dreidimensional zu drehen ist kein Problem. Ich werde mir während meines Urlaubes Gedanken (evtl. praktischer Natur) machen, wie ich die Grafikkarte aufrüste, um auch Farben zu haben. In diesem Zusammenhang schauen wir uns auch nach einem geeigneten preiswerten Farbmonitor um.

Das nächste **Clubtreffen** soll an einem Samstag in der zweiten Novemberhälfte oder der ersten Dezemberhälfte dieses Jahres in Hannover stattfinden. **Jeder** von Euch, der Lust hat, an dem Clubtreffen teilzunehmen schicke bitte Christian Löhrmann eine **Postkarte** auf der folgende Informationen stehen: Absender, geeignete(r) Termin(e), Wünsche, Ideen, ... - naja, was halt wichtig ist. Und zwar bis spätestens zum 19. Oktober 1987. (Meine Karte geht diese Tage in die Post!) Christian wird das Clubtreffen vorbereiten, wenn sich mindestens 25 Teilnehmer bei ihm melden. Er schickt Euch dann als Antwort eine Postkarte mit den genauen Angaben über Termin und Ort.

Im nächsten Info erscheint wieder mal ein **Gesamt-Inhaltsverzeichnis**. Falls Ihr dazu noch offene Wünsche oder Anregungen habt, teilt sie bitte **mir** mit. Eine **Gesamt-Mitgliederliste** wird dann auch wieder dabei sein.

Mir wurde für meine Arbeit ein **Lotus**-Kurs beschert. NEIN, ich arbeite nicht so DEM Gewerbe! Lotus ist so etwas ähnliches wie unser SuperCalc, nur für IBM-PC's, jedoch etwas Leistungsfähiger als SC. Da ich im Rahmen dieses Kurses eine recht putzige Anwendung programmiert habe - die logischerweise leicht auf SuperCalc übertragbar war, habe ich meine SC-Diskette aus der Ecke gekramt, und übertragen. Meine Ergüsse stehen weiter unten.

Beim Hermes (präzise: Hermes Kreditversicherungs-AG) wo ich mittlerweile meine Brötchen verdiene und auch von Clubmitgliedern werde ich immer wieder gefragt, wie ich es aushalte, tagsüber mit einem Computer zu arbeiten, und mich dann auch noch privat mit so einem Gerät herumzuschlagen. Nun-ja ... Das ist leicht gesagt:
In der Firma ist die Arbeit am Computer nur ein Teil meiner Aufgaben. Ich arbeite im Bereich der Bürokommunikation und der PC's. Dabei fallen auch Computerferne Arbeiten - insbesondere im Bereich der Anwenderunterstützung an. Es sind immer wieder organisatorische und konzeptionelle Tätigkeiten gefragt. Zum anderen hat mein Memotech gegenüber dem IBM-Großrechner beim Hermes zwei Vorteile: Ich habe den Rechner für mich alleine, also kein Warten dank vieler Benutzer am System und den Memotech kann ich auch 'mit dem Lötkolben programmieren'.

Übrigens, wißt Ihr, was

Management by Corcodile

ist ?

Ganz einfach: Die Klappe weit aufreißen, auch wenn einem das Wasser bis zum Hals steht! Wer kennt noch mehr von denen ? Ich sammle solche Dinger - und gebe sie auch gerne in einem der nächsten Infos der öffent- und damit der Lächerlichkeit preis.

Herz-liche Grüße

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.
- SDX, ein Laufwerk, in Top-Zustand (Post-versand-fähig, d.h. rüttelfest) für DM 750.-
  Was ich weitergebe ist überprüft, und 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 Daenell, Luxemburger Str. 414, 5000 Köln 41:
80-Zeichen/RS232-Karte der SDX, d.h. passend in den MTX. Preis VHS. (VHS hat nichts mit Video zu tun, das heißt VerHandlungsSache!)

Karl-Heinz-Rößler, Habichtweg 3, 7920 Heidenheim 5, 07321-64426:
PASCAL-ROM mit Handbuch: DM 50.- (arbeitet nicht mit Floppy zusammen!)

C L U B: Redaktionelles / Fragen / Ankündigung

Korrektur & Nachtrag

Info 21, Seite 30 und **Info 21, Seite 32**

8 MHz: Leider war da ein kleiner Fehler im Schaltplan. Die Schaltung lieferte immer gerade die falsche Frequenz. Da ich aber gleich einige Tips aus meinen Erfahrungen mit dem ersten Umbau einen anderen Memotech berichten möchte lest bitte hierzui den Artikel weiter unten.

Info 21, Seite 37

Portdekodierung für Port 91H: Schalter 1,2,4 zu; Schalter 3 auf.

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.

F: (Peter Würfel, 7262)
Im Info 21 stand ja was zu MAKE.COM. Wie kann ich das denn nun wie dort erwähnt für Backups nutzen ?

F: (Peter Würfel, 7262)
Wer hat sein NW so gepatcht, daß es auf einem TA-alphatronic-Schrott-Computer läuft ?

Ankündigungen

(Herbert zur Nedden, 2000)

CLUB-PD: Nummer 23 und ggf. weitere erscheinen nach meinem Urlaub. U.a. wird darauf ein deutsches NewWord-Message-Overlay (d.h. deutsche Fehlermeldungen in NewWord) zu finden sein.

KLICKs: Olaf Krumnow sammelt diverses zum RAM4-KLICK. Wenn es seine Bundeswehr-Zeit zuläßt, wird es demnächst einige Disketten mit diversen Feinheiten hierzu geben.

dBASE - Tip

(Herbert zur Nedden, 2000)

Mit ACCEPT eine Zahl einlesen. Das Problem ist, daß nach dem ACCEPT die Variable als Textvariable genommen wird, weshalb folgendes nicht geht:

```
ACCEPT K
? $("1234567890",K,1)
```

Aber folgendes geht:

```
ACCEPT K
STORE VAL(K) TO L
? $("1234567890",L,1)
```

T E S T: dSM

Untersuchung des neuen MTX 512 S2 mit 256k auf der Hauptplatine

(Herbert zur Nedden, 2000)

Alois Binder ist stolzer Besitzer eines solchen Wundergerätes! Freundlicherweise schickte er es mir, damit ich mich mal mit der modernen Technik der Firma Memotech vertraut machen konnte.
Also frisch ans Werk mit dem Imbus-Schlüssel - und großes Erstaunen: Die Platine kenne ich doch. Wie haben die denn das geschafft. Keine fliegende Verdrahtung auf der Platine, aber die angekündigten 256k-RAMs. Meine Vorahnung hat sich bestätigt: unter der Platine sind einige Leitungen nachträglich verlegt. Naja, warum auch nicht, werdet Ihr nun einwenden - und ich gebe Euch auch Recht!
A-B-E-R-R-R-R: Memotech hat geschummelt! Der MTX 512 S2 hat nämlich einfach 4 Banks mit je 64kB, von denen je nachdem ob die ROM an oder aus sind die 16kB aller 4 Banks von 0000 bis 3FFF Hex ausgeblendet werden, und die 16kB des ungebankten Bereich von C000 bis FFFF Hex der Banks 1-3 werden auch einfach ausgeblendet.

Damit dies nun etwas klarer wird:
Teilen wir den Speicher von 265kB mal in 16kB Blöcke auf, die ich einfach mal bit B01, B02, .. B16 bezeichne.

Dann sollte die Speicheraufteilung im ROM-Betrieb wie folgt aussehen
So steht's MTX-Handbuch, und so tut's unsere Hauptplatinen-Aufrüstung

	0000 - 3FFF	4000 - 7FFF	8000 - BFFF	C000 - FFFF
Bank 0	ROM	B01	B02	B03
Bank 1	ROM	B04	B05	B03
Bank 2	ROM	B06	B07	B03
Bank 3	ROM	B08	B09	B03
Bank 4	ROM	B10	B11	B03
Bank 5	ROM	B12	B13	B03
Bank 6	ROM	B12	B13	B03
Bank 7	ROM	B14	B15	B03
Bank 8	ROM	B16		B03

So sieht die Speicheraufteilung im ROM-Betrieb aus

	0000 - 3FFF	4000 - 7FFF	8000 - BFFF	C000 - FFFF
Bank 0	ROM	B01	B02	B03
Bank 1	ROM	B04	B05	B03
Bank 2	ROM	B06	B07	B03
Bank 3	ROM	B08	B09	B03

A-L-S-O bietet der MTX 512 S2 im ROM-Betrieb nur popelige **144 kB**, und im RAM-Betrieb (CP/M-Modus, z.B. für RAM 4) popelige **208 kB**!

S u p e r C a l c: Amortisationsrechnung

(Herbert zur Nedden, 2000)

Ziel dieser Anwendung ist es, herauszufinden, wie lange ein Kredit abbezahlt werden muß, und was der Spaß insgesamt kostet.

Nehmen wir nun mal einen Kredit über die Summe von DM 150000,- zu 7% jährlichen Zinsen auf. Damit der Kredit auch mal weg ist, müssen wir den Kredit auch abbezahlen, d.t. tilgen. Also tilgen wir jedes Jahr 3,5% der Kreditsumme. Würden wir das nun hochmatematisch exakt so tun, dann würden wir im ersten Jahr 10,5% der Kreditsumme bezahlen, und dann im nächsten Jahr wieder 10,5% der restlichen Kreditsumme (schließlich haben wir schon was getilgt) zahlen. u.s.w. Aber dann würden wir ja nie fertig, da wir ja jedes Jahr 89,5% der Summe stehen lassen.
Nein ein halbwegs vernünftiger Mensch sollte einplanen jedes Jahr die gleiche Summe an den Kreditgeber zu geben, also jedes Jahr 10,5% der Gesamt-Kreditsumme von 150000,-, d.h. DM 15750,-.
Da wir aber jedes Jahr einen Teil der Kreditsumme abbezahlen, fallen auch immer weniger Zinsen an, d.h. wir zahlen immer mehr von Kredit zurück.

Und damit sieht die Chose so aus:

	A	B	C	D	E
1	A m o r t i s a t i o n s r e c h n u n g				
2					
3	Kredit		150000.00	DM	
4	Zins		7	%	
5	Tilgung		3.5	%	
6					
7	Jahressumme		15750.00	DM	
8					
9	Gesamtzinsen		105838.86	DM	
10	Gesamttilgung		150000.00	DM	
11	Laufzeit		16	Jahre	
12					

13 Jahr	Zins	Tilgung	Rest	Jahressumme
15 1	10500.00	5250.00	144750.00	15750.00
16 2	10132.50	5617.50	139132.50	15750.00
17 3	9739.28	6010.73	133121.78	15750.00
18 4	9318.52	6431.48	126690.30	15750.00
19 5	8868.32	6881.68	119808.62	15750.00
20 6	8386.60	7363.40	112445.22	15750.00
21 7	7871.17	7878.83	104566.39	15750.00
22 8	7319.65	8430.35	96136.04	15750.00
23 9	6729.52	9020.48	87115.56	15750.00
24 10	6098.09	9651.91	77463.65	15750.00
25 11	5422.46	10327.54	67136.10	15750.00
26 12	4699.53	11050.47	56085.63	15750.00
27 13	3925.99	11824.01	44261.62	15750.00
28 14	3098.31	12651.69	31609.94	15750.00
29 15	2212.70	13537.30	18072.63	15750.00
30 16	1265.08	14484.92	3587.72	15750.00
31 17	251.14	3587.72	.00	3838.86
32 0	.00	.00	.00	.00
33 0	.00	.00	.00	.00
u.s.w.				
39 0	.00	.00	.00	.00

S u p e r C a l c: Amortisationsrechnung

Faszinierend, Gell ? Nun zu der Auflösung:

	A	B	C	D	E
1	A m o r t i s a t i o n s r e c h n u n g				
2					
3	Kredit		150000	DM	
4	Zins		7	%	
5	Tilgung		3.5	%	
6					
7	Jahressumme		C3*(C4/100+C5/100)	DM	
8					
9	Gesamtzinsen		SUM(B15:B39)	DM	
10	Gesamttilgung		SUM(C15:C39)	DM	
11	Laufzeit		MAX(A15:A39)-1	Jahre	
12					
13	Jahr	Zins	Tilgung	R	Jahressu
14					
15	1	C3*C4/100	C7-B15	C3-C15	B15+C15
16	IF(B16>0,A15+1,0)	D15*C4/100	IF(D15-B16-C7>0,C7-B16,D15)	D15-C16	B16+C16
17	IF(B17>0,A16+1,0)	D16*C4/100	IF(D16-B17-C7>0,C7-B17,D16)	D16-C17	B17+C17
18	IF(B18>0,A17+1,0)	D17*C4/100	IF(D17-B18-C7>0,C7-B18,D17)	D17-C18	B18+C18
19	IF(B19>0,A18+1,0)	D18*C4/100	IF(D18-B19-C7>0,C7-B19,D18)	D18-C19	B19+C19
20	IF(B20>0,A19+1,0)	D19*C4/100	IF(D19-B20-C7>0,C7-B20,D19)	D19-C20	B20+C20
u.s.w.					
38	IF(B38>0,A37+1,0)	D37*C4/100	IF(D37-B38-C7>0,C7-B38,D37)	D37-C38	B38+C38
39	IF(B39>0,A38+1,0)	D38*C4/100	IF(D38-B39-C7>0,C7-B39,D38)	D38-C39	B39+C39

Die Zeilen 17 bis 39 können sehr leicht mit dem SuperCalc Befehl
/R,A16:E16,A17:a39,A
erzeugt werden. Das A am Ende heißt 'Ask for Adjust', d.h. Nachfragen, welche Zellenadressen entsprechend geändert werden sollen, und welche nicht. In Spalte B sollen offensichtlich die Zellenadressen D15, D16, ... sich ändern, das C4 hingegen immer C4 bleiben. Nach dem Abschicken des o.g. Befehls wird gefragt, welche Zellenadressen geändert (adjusted) werden sollen und welche nicht. Die Zellenadressen C4, und C7 dürfen nicht geändert werden, da diese sich auf feste Zellen beziehen, während die anderen Zellenadressen sich auf die aktuelle oder vorherige Zeile beziehen.

Nun zu einigen Erläuterungen der Formeln:

1. Jahr, d.h. Zeile 15:

B15: Zins = Kredit mal Zinssatz
C15: Tilgung = Kredit mal Tilgungsrate = Jahressumme - Zinszahlung
D15: Rest vom Kredit = Kredit minus Tilgung
E15: Jahressumme = Zinsen plus Tilgung

weitere Jahre:

A16:A39: Ein Jahr mehr.
Mit der IF-Abfrage wird hier eine Null eingetragen, wenn der Kredit abbezahlt ist, damit die Laufzeit mit der MAX-Funktion ermittelt werden kann.
B16:B39: Zins = Restkredit mal Zinssatz
C16:C39: Tilgung = Jahressumme minus Zinszahlung
D16:D39: Rest = alter Rest minus Tilgung
E16:E39: Jahressumme = Zinsen plus Tilgung

Viel Spaß

S u p e r C a l c: Patch / N e w W o r d: Tip

SuperCalc und ESC-Ä-T-3 (Peter Würfel, 7262)

Im Info 20 hat der Uwe aus Braunschweig gezeigt, wie man dBASE dazu bringt, beim Aufruf auf das gewünscht Funktionstastenmodul zu schalten. Um dies auch bei SC zu erreichen, habe ich folgende Patches durchgeführt (Ich hab dazu moni2 verwendet, und kann dieses Patch-Programm nur jedem empfehlen!):

Ab der Stelle 5D68 befindet sich in SC.COM der Text 'Enter "?" for HELP or "return" to start'. Das Wörtchen 'Enter' überschreibe ich (mit monis Hilfe) durch **1B 5B 54 33 20** (das entspricht ESC Ä T 3) und hab dadurch automatisch beim Starten von SuperCalc auf meine Funktionstastentabelle 3 umgeschaltet (daß auf dem Bildschirm das Wörtchen 'Enter' fehlt, stört mich nicht).
Um nach der Arbeit mit SC wieder auf der Funktionstastentabelle 1 zu sein, hab ich folgendes gepatcht:
Im Original-SC steht ab 1B34 Folgendes: 'Exit SuperCalc'. Hier setzt mein Patch ein: ich überschreibe ab 1B34: **1B 5B 54 31 45 85 49 54 20 53.** Damit wird mir in dem Moment, in dem ich den Befehl '/Q' eintippe automatisch zurück auf Tastaturtabelle 1 geschaltet. Auf dem Bildschirm erscheint mir die Anzeige (vergl. oben) 'EXIT SCalc'. Der Schönheitsfehler dieses Patches ist der: wenn ich nach der Bildschirmanzeige 'EXIT...' als Antworttaste 'n' drücke, also nicht aus SC aussteigen möchte, ist trotzdem auf die Funktionstastentabelle 1 (CP/M) zurückgeschaltet; aber in der Praxis: wenn ich '/Q' wünsche dann will ich aussteigen; wenn ich mirs dann doch anders überlege, muß ich halt über SHIFT ESC wieder die entsprechende Tastaturtabelle anwählen.

NewWord-Anwendung (Uwe Grass, 3300)

Neulich fragte jemand an, ob man im Header oder Footer eines Textes auch Fußnoten schreiben kann. Sie sollten nicht nur hoch- oder tiefgestellt werden, sondern auch noch eine geringer Höhe als normale Buchstaben haben. Da unser DMX80 solche Zeichen drucken kann, mußte es dafür auch eine Lösung geben. Leider unterstützt der Treiber des P1090 diese Möglichkeiten nicht, der Treiber DMX80 kann es, hat aber andere Macken (Hat jemand einen besseren Treiber produziert, z.B. mit microspacing, allen Möglichkeiten des DMX80 usw. Damit werde ich mich demnächst noch mal auseinandersetzen). Damit nicht jeder selbst daran rumknifteln muß, hier das Ergebnis der Nachforschungen. Die Fettgedruckten Zeichen sind Control-Codes. Hinter dem // sind Kommentare.

```
.XQ1B530                        // Druck in der oberen Hälfte der
.XW1B54                         // normalen Zeichenposition ein/aus
.PL25                           // Seitenlänge für Test
.he test^Qtest^Wtesttesttest        // Testtext
dies ist ein test
^A                              // Alternative Schrittweite
.cw5                            // auf cw5 eingestellt
^N                              // Wieder auf normale Schrittweite zurück
DIES IST EIN TEST ^A^Q DIES IST EIN TEST^W^N

TEST
.fo test ^ATEST ^QTEST^W^N TEST
.F2 TEST ^ATEST^N
```

Alles klar?! Wenn nicht, probiert es einfach mal aus, die Dot-Commands müssen ganz links beginnen.

UG

L e s e r b r i e f: Herbert Oppmann

Herbert Oppmann
Goethestr.19, Zi.27
8522 Herzogenaurach
Tel:09132/60103

Herzo, den 03.08.87

Hallo Herbert !

Nachdem es unter RAM 4.2 bzw. ZCPR2/P2DOS einige Befehle gibt, die mehr oder weniger von UNIX abgeschaut wurden, kam ich auf die Idee, daß es vielleicht ganz nett wäre, den UNIX-Befehl AT zu haben. Für nicht-UNIXer: z.B.

AT 2200 CC -C HALLO.C

heißt: heute Abend, um 22 Uhr (wenn niemand mehr am Rechner ist) starte das Programm CC mit den Argumenten -C HALLO.C (CC ist der C-Compiler, und der dauert immer so lang). Eine mögliche Anwendung am MTX wäre z.B.

AT 1600 ECHO Stop hacking, it's tea time !

Das dürfte nicht allzu schwer zu implementieren sein. Ich hab mir überlegt, daß man wie in UNIX dazu AT, CRON und CRONTAB braucht. CRONTAB ist eine Datei, in der in chronologischer Folge die Zeiten (incl. Datum!) und die auszuführenden Aktionen aufgelistet sind. Wenn man dafür sorgt, daß diese Datei immer auf einem physikalischen Laufwerk angelegt wird (sie wird ja nicht häufig benutzt werden), kann man auch sich selbst zum Geburtstag gratulieren oder irgendeinen Knalleffekt an Silvester loslassen, vor allem, wenn CRONTAB auf der Boot-Diskette abgelegt wird. AT muss das Benutzer-Interface stellen, also die Möglichkeit geben, alle bisherigen Einträge anzusehen, welche zu löschen und welche einzugeben. RAM 4.x muß eigentlich nur das aktuelle Datum/die aktuelle Zeit mit dem nächsten Zeitpunkt, an dem was zu tun ist vergleichen, und wenn gleich bzw. überschritten, an die multiple command line den Befehl CRON anhängen. Das ist alles. Das Datum/die Uhrzeit des nächsten Zeitpunktes muß irgendwo im RAM liegen und von AT bzw. CRON aktualisiert werden (die Frage ist, wie man das nach dem Booten hinkriegt). CRON muß, wenn es aufgerufen wird, CRONTAB suchen, und den nächsten Eintrag (sofern vorhanden) 'scharfmachen', nachdem es die auszuführenden Kommandos an die multiple command line angehängt hat. Natürlich ist das ein wenig Spielerei; eine ernsthafte Nutzanwendung fällt mir dafür nicht ein, aber lustig wäre das schon.

Mir ist zu RAM noch was eingefallen: ein Monitorschoner. Das erfordert einen Zähler in RAM, der bei einem Tastendruck zurückgesetzt wird (auch SHIFT alleine sollte in diesem Zusammenhang als Tastendruck gelten) und der durch einen regelmäßigen Interrupt (der läuft ja schon) hochgezählt wird. Wenn nach einiger Zeit ohne Tastendruck (ca. 4 Minuten) der Zähler einen gewissen Wert erreicht hat, wird der aktuelle Bildschirminhalt in den KLIX-HEAP oder sonstwohin gerettet und der Bildschirm leer, also schwarz gemacht. Das soll die Bildröhre schonen. Ein Tastendruck, vorzugsweise SHIFT, und der Inhalt wird wieder restauriert. Auch dieses Feature ist nicht unbedingt notwendig, aber dürfte ohne großen Aufwand einzubauen sein.

Tschüß! Herbert.

L e s e r b r i e f: Holger Hansen

Holger Hansen
Scharnhorststr. 2
3300 Braunschweig
Tel. 0531/795792

17.08.87

Hallo Herbert

Fürs Info 2 Dinge:

1. Ankündigung:Ich habe vor ein Backup Programm zu schreiben, das die Eigenschaften von Ram4 (Systemuhr und Zeiteinträge) ausnutzt, indem es nur dann kopiert, wenn die Quelldatei aktueller ist. Wer dazu Vorschläge oder Anregungen hat, soll sich ruhig bei mir melden.

2. Nachtrag zu Uwes Turbo Patch aus Info 20 Seite 22
   Sein Patch funktioniert nur dann korrekt, wenn die Overlays UND die COM Datei auf dem gleichen Laufwerk sind. Wenn man aber, wie ich, die COM Datei in der Statik-Ram Floppy, die Overlays in der Eprom Floppy hat und das Programm von der Ramfloppy A: startet, dann hängt sich Turbo nach Aufruf der X-option mit der Meldung "Insert TURBO.COM in Drive A:" auf, was bei einer Ramfloppy natürlich unmöglich und deshalb katastrophal ist.
   In Turbo 3.0 kann man sich wie folgt behelfen:

   Patch von TURBO.OVR mit z.B. DDT:

   An Adresse 033Dh steht ein BDOS Call der das Bezugslaufwerk festlegt. Dieser Aufruf wird geNOPt und in Register A wird das Laufwerk übergeben.(A: = 0,B: =1,...,I: =8)
   Nachteil: im Menü ist nun das eingeloggte Laufwerk I: statt A:.
   Man muß also mit Option L sein Laufwerk neu einloggen.

   Besitzer von Turbo 2.0 müssen sich den entsprechenden Aufruf leider selber suchen, da ich nicht über diese Version verfüge.

   Inhalt:
   ```
   33D  0E 19 CD 00 05 .... ALT
   33D  3E 08 00 00 00 .... NEU  (08 = LAUFWERK I:)
   ```

   (Natürlich NIE mit den Originalen arbeiten !!!!)

   ```
   DDT TURBO.OVR
   NEXT   PC
   0600  0100
   -s33d
   033D 0E 3E
   033E 19 08
   033F CD 00
   0340 00 00
   0341 00 .
  -G0
  A>SAVE 5 TURBO.OVR
   ```

Holger Hansen

T U R B O: Noch'n TURBO-Patch

Patches für TURBO.OVR (Vers.3.0)

Sofern man das X-Komanndo von TURBO überhaupt verwenden will, erweist sich Uwe Grass' Tip (Info 20 S.22) auch für diesen Fall als durchaus nützlich, jedoch hierfür als lediglich halbe Lösung:
Angenommen, auf H: befinde sich TURBO.COM sowie TURBO.OVR, aktives Laufwerk sei A:, ohne TURBO.COM. (Wozu auch, ist ja schließlich auf H:.) Nach Ausführung der von Uwe Grass vorgeschlagenen Änderungen läßt sich tatsächlich ein transientes Programm von TURBO aus starten. Probleme ergeben sich allerdings, wenn zu TURBO zurückgekehrt werden soll: Das Overlay sucht TURBO.COM dann nämlich auf dem Laufwerk, das bei Aufruf des X-Kommandos gerade aktiv war, im angenommenen Falle also auf A:, was zu der Fehlermeldung 'TURBO.COM not found' führt, verbunden mit der Aufforderung, man möge doch eine Diskette mit TURBO.COM darauf ins Laufwerk A: stecken, was ja nun leider bei einer Ram-Floppy nicht ganz so einfach ist... Zudem stellt sich das Overlay in seinem Verlangen nach TURBO.COM sehr stur und widersetzt sich jeglichen Unterbrechungsversuchen, so daß schließlich nur das Betätigen der Reset-Tasten verbleibt.
Eine Änderung, die TURBO.OVR dazu veranlaßt, TURBO.COM beispielsweise immer auf H: zu suchen, ist schnell durchgeführt:
(Adreß-Angaben beziehen sich im folgenden immer auf ein ab 100H (z.B. durch DDT) geladenes TURBO.OVR)

Adr	Mnemonics	Hex	Bemerkung
0328H	ld a,'H'	3E 48	ASCII-Zeichen für Laufwerk

Was aber, wenn TURBO.COM dann auf H: nicht zu finden ist ? Entweder...Siehe oben ... oder Rückkehr auf die ZCPR-Ebene dank folgender Änderung:

Adr	Mnemonics	Hex	Bemerkung
01E8	nop	00	Die nop's sind übrigens nicht zu-
01E9	nop	00	fällig hier hereingeraten, sondern
01EA	ld hl,0118H	21 18 01	so braucht die Tabelle mit der
01ED	push hl	E5	Verschiebeinfo für das Overlay
01EE	nop	00	nicht geändert zu werden.
01EF	ld de,019EH		Fehlermeldung ausgeben
01F2	ld c,09H		(Code wird unverändert
01F4	call 0005H		übernommen)
01F7	pop hl	E1	
01F8	ld a,C3H	3E C3	Schiebe dem Overlay einen Sprung
01FA	ld (hl),a	77	zur Warmstart-Adresse (jp 0000)
01FB	inc hl	23	an geeigneter Stelle unter, aber
01FC	xor a	AF	nicht ohne vorher einen Programm-
01FD	ld (hl),a	77	teil anzuspringen, der die BIOS-
01FE	inc hl	23	Einsprungs-Tabelle wiederher-
01FF	ld (hl),a	77	stellt.
0200	jr +10	18 0A	

Schließlich sollte man die Fehlermeldung auf den noch sinnvollen Teil beschränken:

Adr	Mnemonics	Hex	Bemerkung
02B6	'$'	24	

Wolfgang Tesch

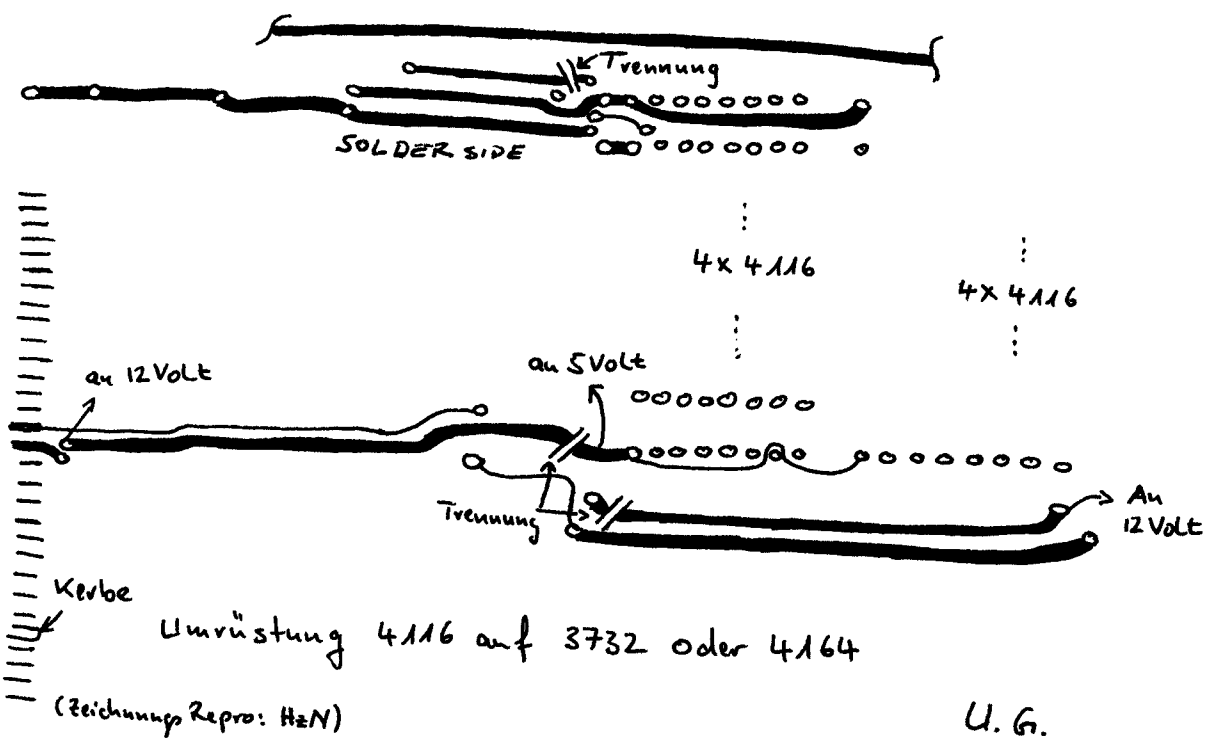
H a r d w a r e: VRAMs ersetzen

Ersetzen der VIDEO-RAMs 4116 (Uwe Grass, 3300)

Tja, nun war es passiert. Bei Basteleien an der Hauptplatine sind die Video-RAMs 4116 mit einem leisen "pitsch" gestorben. Die Spannungen waren nicht gleichzeitig an den Versorgungsspannungspins aufgetaucht. Warum "die Spannungen"? Die 4116 benötigen zum Betrieb +5V, -5V, +12V und natürlich 0V. So ein Mißgeschick kann also jedem passieren, der an seiner Spannungsversorgung des MTX herumbastelt, dabei aber eine Leitung nicht oder falsch anlötet, oder schlicht und ergreifend, so wie ich, den Stecker der Versorgungsspannung herrauszieht, ohne die Betriebsspannung des Netzteils abzuschalten.
Da ich nun ohnehin die RAMs austauschen mußte, wollte ich auch nicht wieder so empfindliche "Käfer" einsetzen. Also Handbücher heraus und das Pinout der verschiedenen möglichen ICs verglichen. Das Ergebnis dieser Tüftelei: man kann ohne großen Aufwand RAMs 4164 oder 3732 einbauen, die Geschwindigkeit sollte 200ns nicht unterschreiten. Die Pinbelegung der 3732 und 4164 sind gleich, Unterschiede zu den 4116 sind folgende:

IC/PIN	1	8	9	
4116	-5V	+12V	+5V	NC = not connected
3732/4164	NC	+5V	A7	= nicht angeschlossen

Die Unterschiede sind also gar nicht so gravierend, A7 wird nicht gebraucht, kann also ständig auf +5V liegen, PIN 1 ist nicht angeschlossen, darf also auch -5V bekommen. Lediglich PIN 8 muß von den 12V getrennt und mit 5V versorgt werden. Die 12V werden auf der Platine noch benötigt, ebenso die negative Spannung. Daraus resultiert die folgende Skizze für den Umbau:



Umrüstung 4116 auf 3732 oder 4164

(Zeichnung Repro: HzN) U. G.

H a r d w a r e: SDX

60-ploiger Bus an SDX (Joachim Keiser, 4925)

Im Info 17, Seite 42 steht die Schaltung für eine PIO-Box, die sich sicher leicht in eine FDX-Station ainbauen läßt, aber wohin damit bei den SDX-Stationen ?

Mit dieser Frage habe ich mich näher beschäftigt, und mit dazu den Controller für das Laufwerk einmal angesehen. Zwischen dem eigentlichen Controllerbaustein (der schwarze Käfer mit den meisten Beinen), sowie dem EPROM und den 60-poligen Stecker befinden sich genau 60 Lötpunkte. Diese Lötpunkte sind nun überweigend mit dem Stecker verbunden und ich habe die unten stehende Kontaktbelegung herausgefunden.

Mit einem feinen Lötkolben, Absuagpunpe und Entlötspitze habe ich diese Augen frei gelegt, und eine 60-polige Pfostenreihe eingelötet. Auf einem schmalen Streifen Lochrasterplatine kam eine 60-polige Buchsenleiste und von der Rückseite eine Stapelleiste (eine weitere Pfostenleiste reicht auch aus). An dieser Stapelleiste wurde die Platine mit dem PIO-Baustein und seiner Logik angelötet. Diese Platine kann nun auf die Controllerplatine gesteckt werden. Die Verlängerung der Steckverbindung ist wegen der Höhe des Controller-IC notwendig. An der Stirnwand habe ich eine Sub-D Buchse (25-ploig) eingebaut, an der die Leitungen PA0 bis PC7 des 8255 angeschlossen wurden. Die Stirnwand muß natürlich zum Einbau der Buchse von der Platine geschraubt werden. Alles zusammen paßt in das Controller-Gehäuse und auf der Rückseite stehen 3 mal 8-Bit breite Ports zur Verfügung. Verschiedene Versuchsschaltungen, darunter ein Epromer, habe ich erfolgreich darüber mit dem Rechner gekoppelt.

0 V	+60	59+	0 V	
0 V	+	+	Ncc	
0 V	+	+	0 V	
0 V	+	53+	Ncc	Diese beiden Anschlüsse wurden von den anderen
Ncc	+	51+	Ncc	Pins abgetrennt und an die 5 V-Versorgung mit-
Ncc	+	+	Ncc	tels eines Kurzen Drahtes angeschlossen
Ncc	+	+	Ncc	
PHI	+	+	0 V	
M1	+	+	WR	
RD	+	+	IORQ	
MREQ	+40	39+	RESET	
Ncc	+	+	Ncc	
Ncc	+	+	Ncc	
0 V	+	+	D7	
D6	+	+	D5	
D4	+	+	D3	
D2	+	+	D1	
D0	+	+	Ncc	
Ncc	+	+	Ncc	
Ncc	+	+	Ncc	
A15	+20	19+	A14	
A13	+	+	A12	
A11	+	+	A10	
A9	+	+	A8	
A7	+	+	A6	
A5	+	+	A4	
A3	+	+	A2	
A1	+	+	A0	
Ncc	+	+	Ncc	
Ncc	+ 2	1+	0 V	

H a r d w a r e: 8 Mega Hertz

Erste Erfahrungen und einige Tips (Herbert zur Nedden, 2000)

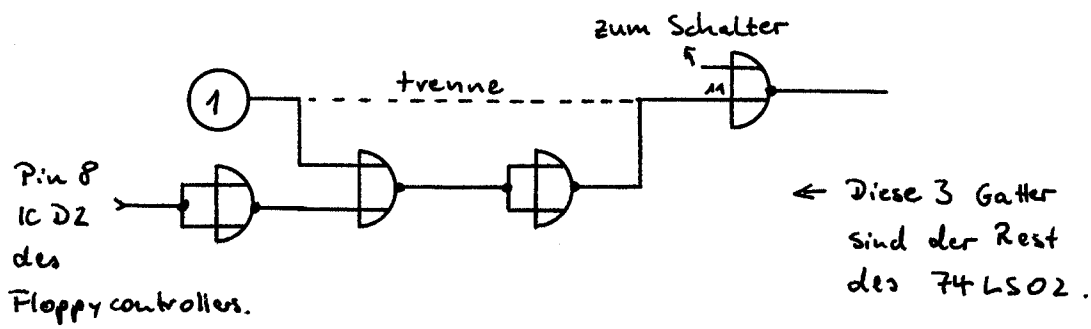
Gleich erst einmal die Korrekturen:
Info 21, Seite 32:
Streiche: Verbindung 74HC74, Pin 6 an 74HC157, Pin 1 (Sel)
Setze: 5
Info 21, Seite 30: Unten Punkt 5
Streiche: Pin 1, 74LS175
Setze: Plus 5 Volt.

Ich habe beim Aufbau der Frequenzumschaltung versucht den 75**HC**157 durch einen 74**LS**157 zu ersetzen, weil mir ein 74HC157 fehlte. An den Eingängen des IC's fand ich auch den 4 und den 8 MHz Takt sauber vor, aber am Ausgang hatte der jeweilige Takt nur noch die halbe Spannung!

Falls die Hauptplatine nicht mit 8 MHz ohne WAIT arbeitet, dann sollte nicht wie auf Seite 30, Info 21 unten unter Punkt 5 beschrieben, das NAND-Gatter des 74LS00 weggelassen, und der freigewordene Pin des FlipFlop (74LS74) an +5 V gelegt werden, sondern der Eingang des 74LS00 statt an das Signal RECPM vom PAL (Verbindung mit der Kennung 8), sondern an den Frquenzumschalter (d.h. Pin 12 des 74LS02) gelegt werden. Sonst gibt's auch bei 4 MHz Waits.

Auf der Busplatine in der FDX muß für das Booten mit 8 MHz (mit Wait) ein EPROM, das 150 ns schnell ist. Ein 200 ns-EPROM reicht evtl. nicht aus. Stattdessen könnt Ihr auch mit 4 MHz booten, d.h. das RECPM-Signal (Pin 13 des PAL) mit in die Frequenzumschaltung einbeziehen.

Da hing sich doch dieser bekn... Memotech beim Diskettenzugriff gerne auf. Naja, anscheinend mochte der Controller nicht einmal, wenn er mit 8 MHz die Controller-Register beschrieben bekam, d.h. noch lange vor dem eigentlichen Diskettenzugriff. Also mußte auch beim I/O-Zugriff auf den Floppy-Controller die 4 MHz eingeschaltet werden, d.h. die Umschaltung mußte ein Quentchen früher geschehen.
Das ging erfreulich leicht, wie folgender Schaltplan zeigt:



H a r d w a r e: Umbau der 80-Zeichenkarte ohne Patch

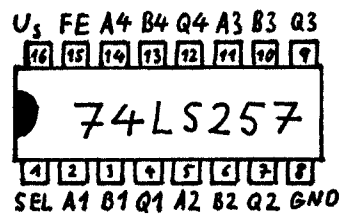
Umbau der 80-Zeichen-Karte ohne Patchen alter Software

(Jan Bredereke, 2000)

Da man ab RAM42 auch Bildschirmformate sinnvoll verwenden kann, die größer als 80*25 Zeichen sind, war es für mich attraktiv geworden, meine 80-Zeichen-Karte nach der Anleitung von Hagen Wenzek im Info 17-69 aufzurüsten.
Da man ja nie weiß, was noch alles passiert, fand ich es nachteilig, daß die Bildschirmtreiber der alten RAM- und CPM-Versionen gepatcht werden müssen, will man sie jemals wieder verwenden. Der Mini-Monitor im Bootstrap-PROM, den man vor dem Booten von Diskette durch Drücken der Returntaste aufrufen kann, ist aus verständlichen Gründen überhaupt nicht zu patchen.
Die einzige Ausnahme ist der BASIC-Bildschirmtreiber, der mit allen Bildschirmspeicherversionen läuft.
Langer Rede kurzer Sinn: Nach Lesen der Umbauanleitung stellte ich fest, daß sich ohne Mehraufwand ein Schalter einbauen läßt, der den Bildschirmspeicher wieder auf die alten 2*2K zurückschaltet.
Daher nun im Telegrammstil eine alternative Umbauanleitung:
Man nehme:

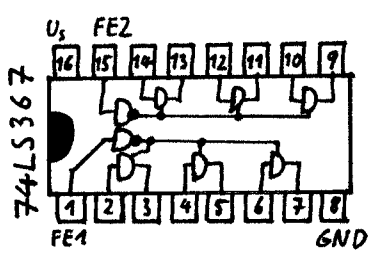
2 * 6264 LP15
1 * **74LS257**
1 * Kippschalter 1 * ein
einen 16-30 Watt Lötkolben,
Schaltdraht oder Fädeldraht
und Lötzinn (kein Lötfett!!!!)

FE	SEL	Q
H	X	Z
L	L	A
L	H	B



Auf das IC 3B, einen 367er, wird nachher Huckepack der 257er gelötet, deshalb nenne ich ihn IC 3BII.
Zuerst aber am 3BII: Pin 2, 3, 5, 6 an 8 (Masse) löten, 13, 10 auch an 8, Pin 4, 7 bleiben frei (falls niemandem mehr eine weitere Erweiterung einfällt).
Jetzt 3BII auf 3B löten: 16 an 16, 15 an 15, 8 an 8.
Nun nehmt die beiden alten 6116 aus den Fassungen.
Sie werden durch die beiden neuen 6264er ersetzt, von denen die Pins 1, 2, 28, 27 überhängen, außerdem werden die Pins 23 hochgebogen und bleiben draußen.
Mit den beiden 6264ern macht ihr jeweils folgendes:
Pin 26 an 28,
Platz unter Pin 23 an Pin 27,
Pin 2 an 4C-16, 23 an 4C-6
(d.h. Pin 6 vom IC auf Platz 4C).
Mit 3BII macht ihr:
9 an 4C-16, 12 an 4C-6.
Mit dem großen 6845er macht ihr:
15 an 3BII-14, 16 an 3BII-11.

FE	E	Q
H	X	Z
L	L	L
L	H	H



Schließlich verbindet ihr noch 3BII-1 mit dem einen Pol des Schalters, dessen zweiter Pol an Masse kommt, z. B. an 3BII-8.
Ist der Schalter geschlossen, sind die vollen 2*8K verfügbar, sonst die 2*2K.
Den Schalter habe ich in einer Bohrung an der Rückseite des FDX-Gehäuses untergebracht. Auf seiner Zuleitung liegt übrigens nur Gleichspannung, so daß die Leitung beliebig lang sein darf.

Was ist nun passiert? Die 16K SRAMs sind wie bei Hagen durch 64K SRAMs ersetzt worden, aber die beiden Multiplexadressen MA11 und MA12 werden nicht einfach durch Tri-State-Treiber geschickt, sondern durch einen 2-nach-1-Tri-State-Multiplexer, so daß durch den Schalter gesteuert entweder die beiden Adreßbits oder zweimal 0 Volt weitergegeben werden.

Nachdem ich den Umbau auf diese Weise gemacht habe, gibt es ohne irgendwelche Patcharbeiten auch bei den alten CPM- und RAM-Versionen keinerlei Bildschirmmüll, es sei denn, man legt mal eben den Schalter um. Schaltet man wieder zurück, ist alles wieder in Ordnung. Nur eines geht nicht ganz richtig: Wenn man unter RAM41 oder früher Klick aufruft, wird der Schirm nach Verlassen von Klick nicht restauriert, sondern das Klick-Fenster bleibt einfach stehen. Wen dies stört, der kann sich ja immer noch RAM41, wie im Info 18-8 beschrieben, patchen.

H a r d w a r e: Mus-Gaudi-D/A-Wandler

Die alten RAM-Versionen benutze ich z.B. noch, um meine selbstgeschriebenen Klick-C-Programme auch mit den alten Versionen zu testen.

Bei dem Umbau meiner 80-Zeichen-Karte habe ich übrigens noch eine **dSV** (dunkle Stunde von Vobis) entdeckt: Beim Einbau des zweiten Laufwerks wurde die Busterminierung im ersten Laufwerk nicht entfernt. Das ist bei mir der weiße, IC-artige Käfer, auf dem BECKMAN .. R150.. steht, und der gleich bei dem Busstecker auf der Platine des Laufwerks sitzt. Dieser Käfer darf nur bei dem letzten Laufwerk am Bus drinbleiben. Bisher lief bei mir trotzdem alles einwandfrei, aber wie lange und wie zuverlässig der Bustreiber noch mitgespielt hätte, weiß ich natürlich nicht.

MUSIK SIG/M 056-058

Hallo Herbert !

Das Musikprogramm gibt nun tatsächlich einige Töne von sich und spielt auch Lieder, die auf der Public-Domain-Diskette Nr. 57 als *.SNG-Files enthalten sind.
Obwohl mir das ganze Programmpaket noch einige Rätsel aufgbit, wie Fehler in Source-Listings, Fehlen einiger Sourcefiles, Bedeutung der Perkussion im Programm usw. usw., möchte ich doch ganz gern mal sagen was zu tun ist.
1.) Man braucht ein Ausgangsport, an dem ein DAC (Digital-Analog-Wandler) angeschlosen wird. Hier kann auch der Druckerport, wie in einem unserer Infos beschrieben, genommen werden. Ich habe Port 145 dez (91H) genommen. Portadressierung siehe später.
2.) Das gesamte Programm war wohl für den Tandy-TRS80 geschrieben. Die anderen Routinen für CP/M sind meiner Ansicht nach erst später hinzugekommen. Um das Programm laufen zu lassen, muß erst Initialisiert werden. Hierzu startet man SETUP. Die Fragen, leider nur in Englisch, müssen mit Y oder N beantwortet werden. Da ich faul bin, habe ich den Bildschirm als ADM 3A angenommen, aber Ram 4x nicht umgeschaltet. Danach kommt die Frage nach dem Keybord (Ausgangsgerät), hier ist nicht die Tastatur, sondern der DA-Wandler gefragt. Diese Frage wird mit NEIN beantwortet. Postwendent wird nach der Ausgangsadresse des DAC's gefragt, und nun muß die Adresse (in DEZIMAL) angegeben werden, an der mein DAC angeschlossen ist. Die Rhythmusbox ist nicht vorhanden, also mit NEIN beantworten. (Hier könnte es sich um die Percussion handeln). Den Drucker habe ich nicht initialisiert. Ach, die Taktfrequenz des Prozessors und der Type wird noch erfragt. Bitte mit 4000000 und mit Z80 antworten.
Bei der Cursoränderung Aufpassen, HOME darf nicht mit der Hometaste beantwortet werden, warum kann ich nicht sagen. Die Pfeile für hoch, runter usw. dürfen benutzt werden.

Die initialisierten Daten wurden nun auf die Diskette geschrieben. Ich nehme an unter dem Namen SONG.CNF.
Nun erscheint der CP/M-Promt. Das Programm mit MUSIC aufrufen, und ein Menue erscheint. Halt, jetzt nicht P für Play eingeben. Bevor ein Lied gespielt werden kann, muß es erst compiliert werden. Also erst C eingeben. Jetzt erscheint die Frage, welches Lied. Nun kann z.B. B:Birthday oder B:FLS eingegeben werden. Der Compiler wird aktiv und es wird ein File vom Typ SCG erzeugt, danach erscheint wieder das Menu. Endlich kann nun durch P das Lied gespielt werden. Es erscheint aber noch die Frage nach der WAVE-Form. Was ist das schon wieder. Das sind Files in denen die charakteristische Klangfarbe eines Instrumentes enthalten ist. Es gibt 13 solcher Files, gekennzeichnet mit *.WAV. Diese Files können auch selbst erstellt werden und zwar mit Hilfe der Fourie-Analyse. Es stehen max. 16 Harmonische zur Verfügung, die in Betrag und Phasenlage gemischt werden.

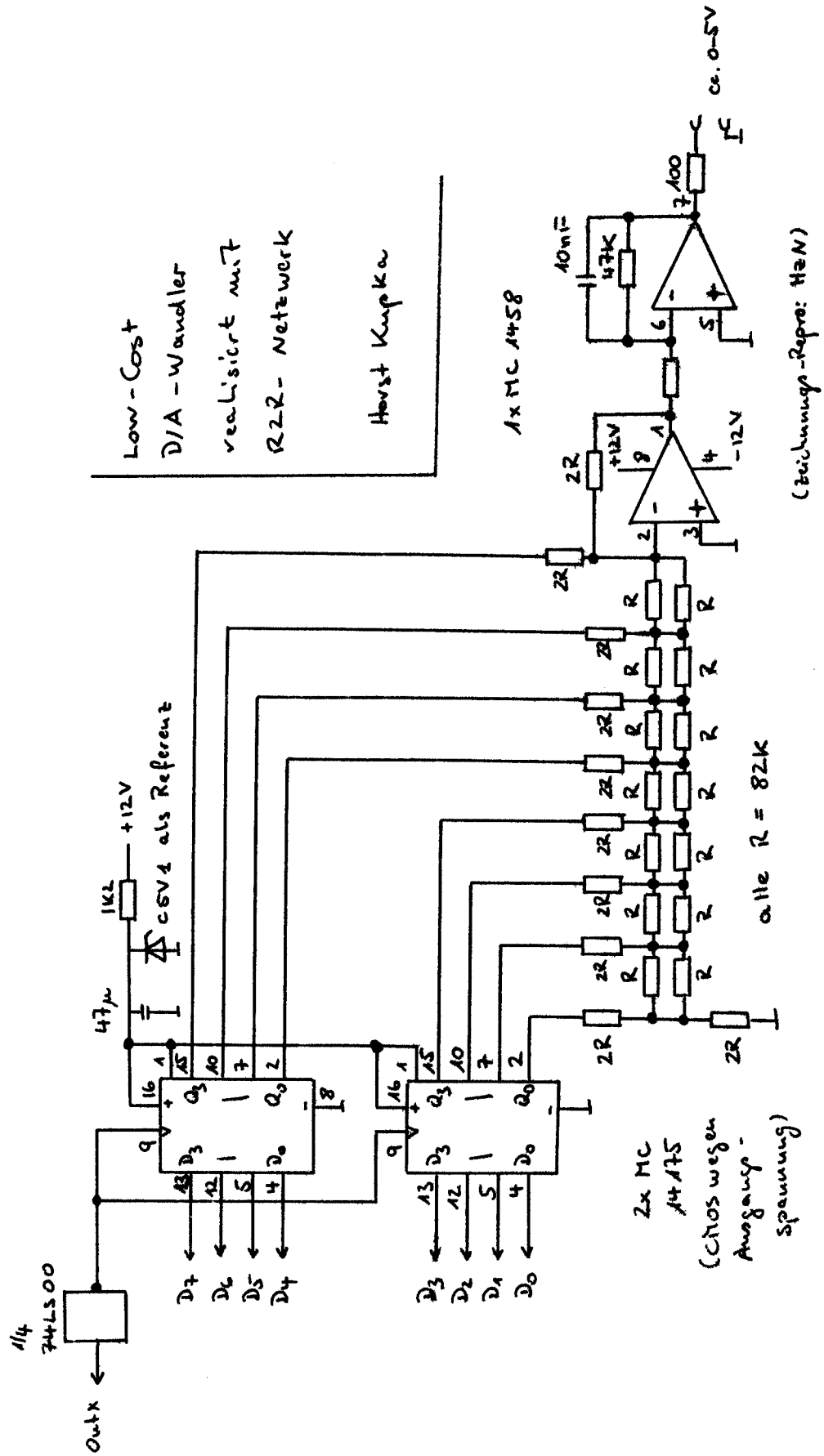
Wird also bei der Wave-Frage mit "=" geantwortet, so wird die Tabelle eingehalten, die programiert wurde, sonst müssen die Klangfarben eingegeben werden. Endlich aus den Lautsprecher, der mittels Verstärker am DAC angeschlossen ist ertönt ein Lied.
Ist das Lied beendet, erscheint wieder ein Menu.
Vor einen Buchstaben im Hauptmenu möchte ich warnen. Es ist das U und bedeutet JUKEBOX oder Utility. Diese Programm benutzt Adressen und Routinen, die auf unserem Rechner nicht laufen. Hier müssen erst sofrwaremäßig einige Änderungen vorgenommen werden.
Mit dem E>ditor können vorhandene Lieder vom Typ *.SNG geändert werden.
So das reicht einmal fürs erste. Du hörst weiteres von mir.
Ich meine es in Bezug auf Hardware und Software. Wenn es aber eilig ist, so kann man sich die Dokumentationen ausdrucken lassen und durch den amerikanischen Text beißen.
Zum Schluß lege ich Dir noch eine Schaltung zur Adresskodierung bei. Diese Art kann von 80H an aufwärts eingestellt werden und liefert gleich 8 Enable für Portadressen.
Bei mir läuft diese Dekodierung wunderbar. Sie ist am Extendet-Stecker links am Gerät angebracht und macht keine Schwierigkeiten.
Die Zahlen in den Klammern sind die Steckerpunkte von der Grundplatine. Die zweite Schaltung ist ein herkömmlich aufgebauter Digital-Analog-Wandler. Das Prinzip ist das sogenannte R2R-Leiterverfahren. Ich hoffe diese Bauteile hat wohl jeder in seiner Bastelschublade. Einfacher ist ein fertiges Chip der Firma Ferranti. Es heißt ZN426 E (oder ein zweiter Typ heißt ZN429 E) und kostet etwa 10 DM. Hier aber ein Speicher vorgeschaltet werden. Ich hoffe das Datenblatt kannst Du besorgen.

bis bald oder ruf doch mal an
Tel. 02173/50679

Horst Kupka

Anm.d.HzN: Horst ist dabei, die Software zu verbessern, und die Hardware zu vereinfachen.
Wenn die Sache weiter gediehn ist, dann werden wir eine PD mit einem angepaßten und lauffähigen MUSI-CRAFT herausbringen.

H a r d w a r e: Mus-Gaudi-D/A-Wandler



H a r d w a r e: Der Anti-Wackelkontakt-Tip

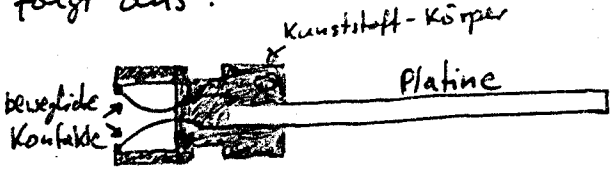
DER TIP

Wer hatte nicht schonmal Kontaktschwierigkeiten im Grundgerät, weil da die Busverbindungen teils aus Gold, teils aus Zinn sind? Wer wollte daran/hin nicht schonmal alles fest zusammenlöten, hat sich aber vor dem Aufwand gescheut?

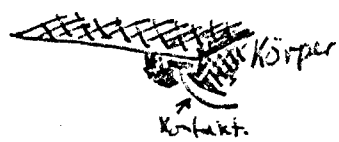
Hier ist DIE Lösung:

1) Man zerlege sein Grundgerät in die drei Bestandteile Motherboard, Speicherkarte, RS232-Karte.

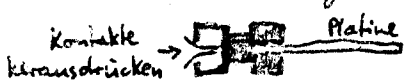
2) Nun schaue man sich die Buchse mal genau an. Sie sieht wie folgt aus:



Die Kontakte rasten vorne im Kunststoff-Körper ein: Der Kontakt ist vorne T-förmig



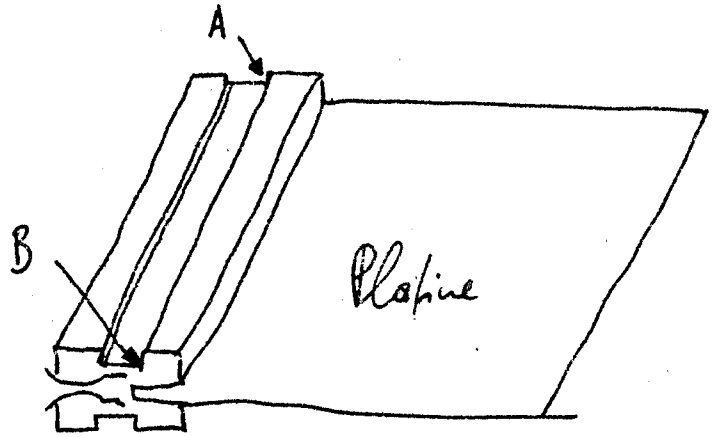
3) Mit Hilfe eines spitzen Gegenstands werden nun die Kontakte einzeln aus ihren Halterungen nach vorne herausgedrückt.



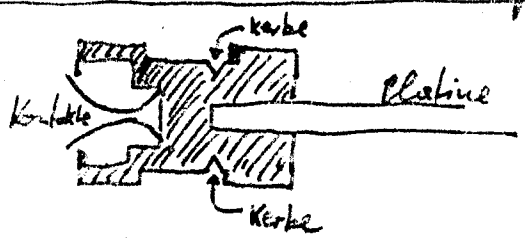
4) Man legt nun den Kunststoffkörper der Buchse auf eine feste Unterlage und nimmt ein Teppichmesser o.ä. zur Hand.

H a r d w a r e: Der Anti-Wackelkontakt-Tip

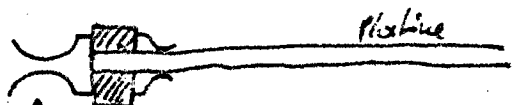
Dann macht man folgende Einkerbungen:



Eine ca 1mm tiefe Kerbe entlang der Verbindung A B, also über die ganze Länge des Buchsen Körpers, gleiches tut man auch auf der Unterseite des Buchsenkörpers.



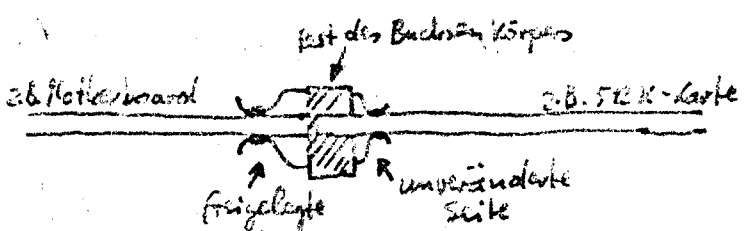
5) Nun kann man mit einer Flachzange den vorderen Teil des Kunststoffkörpers oder Buchse leicht nach oben abbrechen. (Unterseite analog)



vorderer Teil des Kunststoffkörpers abgebrochen

Die Kontakte bleiben hierbei unversehrt und auch nach wie vor funktionsfähig.

6) Man schiebt nun die beiden Platinen wie gewohnt zusammen und lötet dann die freigelegten Kontakte fest.



Das Ganze dauert insgesamt keine halbe Stunde !!!!

A s s e m b l e r: Kurs

ASSEMBLERKURS

Lösung zu Aufgabe 1.3.2-1:

Ich gebe hier zwei Lösungen an. Die erste ist "geradeaus" und berechnet erst die Fläche des großen Rechtecks, dann die des kleinen. Die Schwierigkeit ist hierbei, daß dann für die Differenzbildung die "falsche" Zahl (nämlich der Subtrahend) im Akku steht. Daher muß vor der Subtraktion der Akkuinhalt gegen den Inhalt von Zelle 0 ausgetauscht werden. Das geschieht mit dem XCHG Befehl. Die zweite Lösung vermeidet dieses Problem, indem erst das kleine und dann das große Rechteck berechnet wird.

Lösung 1

```
8011 LOAD 11   grosses R.
6008 MUL  08
910  STO  (0) zwischenspeichern
8004 LOAD 04   kleines R.
6003 MUL  03
310  XCHG (0) vertauschen
510  SUB  (0)
991  EXIT
```

Lösung 2

```
8004 LOAD 04   kleines R.
6003 MUL  03
910  STO  (0) zwischenspeichern
8011 LOAD 11   grosses R.
6008 MUL  08
510  SUB  (0)
991  EXIT
```

Jetzt geht es weiter im normalen Text:

1.3.4 Weitere Beispiele

Da Übung bekanntlich den Meister macht, werden wir hier noch einige Beispielprogramme angeben, ohne allerdings den Ehrgeiz zu haben, dadurch alle Befehle abzudecken.

Zunächst wollen wir ein Programm schreiben, das die Summe der Zahlen von 1 bis zu einer gewissen Obergrenze n berechnet. Der alte Gauss fand dafür die Formel Summe= n*(n+1)/2 heraus. Die Zahl n möge bereits in Zelle 0 gespeichert sein, d. h. das Programm kann einfach annehmen, daß es diesen Wert dort vorfindet. Bevor du weiterliest, versuche mit Hilfe der Befehlstabelle ein Assemblerprogramm für die Summationsaufgabe zu entwerfen.

Zunächst werden wir die Zahl n in den Akku laden müssen. Der Befehl dazu ist LOAD (0). (Wir erinnern uns daran, daß mit (0) der Inhalt der Speicherzelle 0 gemeint ist.) Um mit n+1 zu multiplizieren, könnte man mit ADD 01 die soeben geladene Zahl n um 1 erhöhen. Da dieser Fall jedoch relativ häufig vorkommt, gibt es einen eigenen Befehl dafür: INC A (Maschinenbefehl 10) erhöht (inkrementiert) den Akku gerade um 1. Auch hier gibt es wieder eine Variante, die auf eine Speicherzelle wirkt: INC (x). Das Gegenstück dazu ist der DEC Befehl, der um 1 erniedrigt (dekrementiert). Damit sieht das Programm dann so aus:

Programm 1.3.4-1

```
LOAD (0)      hole n aus Zelle 0 in den Akku
INC A         gibt n+1
MUL (0)       n*(n+1)
DIV 02        Ergebnis
EXIT
```

Bevor du das Programm laufen läßt, mußt du Zelle 0 mit einer (kleinen) Zahl "von Hand" laden. Die Befehle dazu kannst du dem Programm 1.3.2-2 entnehmen (bis n=9 müßte es problemlos klappen, aber was passiert bei größeren Werten?).

Übungsaufgabe 1.3.4-1:
Versuche, das Programm so zu verbessern, daß es auch noch für größere n richtig rechnet. Vielleicht wird es dazu zweckmäßig sein, für gerade und ungerade n eine eigene Variante zu entwickeln.

Vertiefungsstoff
Es scheint so, daß der Befehl INC A einen Operanden A benötigen würde. Wenn du dir aber den Maschinenbefehl dazu ansiehst, kommt da keiner vor. Daß sich der Befehl auf den Akku bezieht, ist bereits implizit im Operationscode enthalten. (Der Operationscode (Opcode) ist der Teil des Maschinenbefehls, der angibt, was getan werden soll. Bei der I2DITVM sind es immer die ersten beiden Ziffern. Der Operandenteil legt fest, womit dies getan werden soll (dritte und gegebenenfalls vierte Ziffer).) Diese Art, den Operanden schon im Operationscode festzulegen, bezeichnet man daher als implizite Adressierung.
Ende Vertiefungsstoff

Übungsaufgabe 1.3.4-2:
Das nächste Programm soll den Mittelwert der beiden Zahlen in den Speicherzellen 0 und 1 berechnen und das Ergebnis in Zelle 2 ablegen. Probiere es mit folgenden Zahlen aus: (9;11), (9;12), (9;13), (9;14). Da die I2DITVM keine Dezimalbrüche darstellen kann, wird nach einer Division mit dem gerundeten Ergebnis weitergerechnet.

Übungsaufgabe 1.3.4-3:
Schreibe ein Programm zur Berechnung der Formel (2*4 + 4*6)/(3+13). Versuche dabei, mit nur einer Speicherzelle für Zwischenergebnisse auszukommen! Der Befehl XCHG (x), der den Inhalt von Akku und Zelle x vertauscht, dürfte sich dabei als nützlich erweisen.

1.3.5 Die merkwürdige Arithmetik der I2DITVM

Wir hatten bereits gesehen, daß die I2DITVM manchmal etwas merkwürdig rechnet. Bei einer Division, bei der das Ergebnis nicht ganzzahlig ist, wird eine Rundung vorgenommen. Aber auch bei einer Addition können eigenartige Dinge passieren, wie du an folgendem Programm siehst:

```
LOAD 99          lade Akku mit 99
INC A            sollte 100 sein ???
EXIT
```

Da die I2DITVM immer nur mit 2 dits (decimal digits = Dezimalziffern) rechnet, kann die Zahl 100 nicht mehr dargestellt werden. Statt dessen wird die 1 einfach weggeworfen und übrig bleibt 00. Man kann sich das etwa folgendermaßen vorstellen: Normalerweise kann man sich ja die Zahlen auf einer unendlichen Zahlengeraden angeordnet vorstellen. Die I2DITVM verfährt nun so, als ob diese Zahlengerade bei 0 und 100 abgeschnitten wird und das verbliebene Stück in der Mitte so zu einem Kreis zusammengebogen wird, daß die 100 auf der 0 zu liegen kommt, wie in Abb. 1.3.5-1 gezeigt.

A s s e m b l e r: Kurs

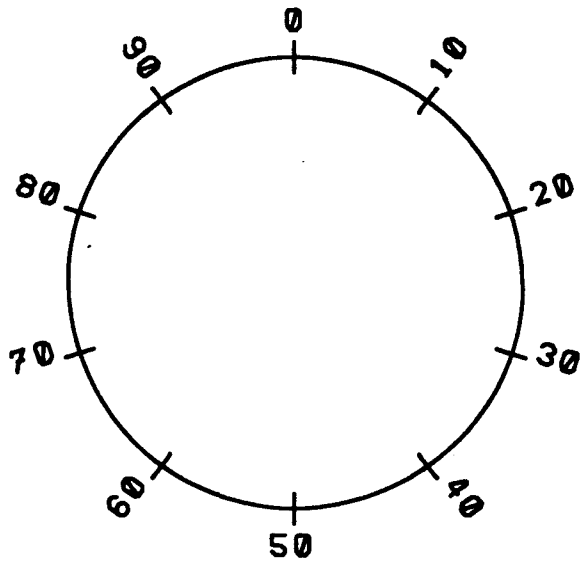


Abb. 1.3.5-1 Der Zahlenkreis

Vertiefungsstoff
1.3.6 Der Stack

Der Stack ist eigentlich Bestandteil des Hauptspeichers. In einem realem Computer wird er aus den gleichen Speicherbausteinen aufgebaut wie der Hauptspeicher auch. Warum er gerade Stack (Stapel, manchmal auch Kellerspeicher genannt) heißt, liegt an der besonderen Art, wie seine Zellen verwaltet werden. Wir werden dies an einem Analogon zeigen. Stelle dir einen Stapel Teller vor (aber vorsichtig, damit sie nicht zerbrechen!). Wir legen nun drei Teller obendrauf. Der jeweils oberste Teller wird dabei markiert. Wenn wir jetzt einen Teller abnehmen, erhalten wir den, der als letzter draufgelegt wurde. Diese Art, einen Stapel zu verwalten, nennt man LIFO (von last in, first out). Das ist eigentlich ganz normal. In Abb. 1.3.6-1 ist dieser Vorgang nochmal illustriert.

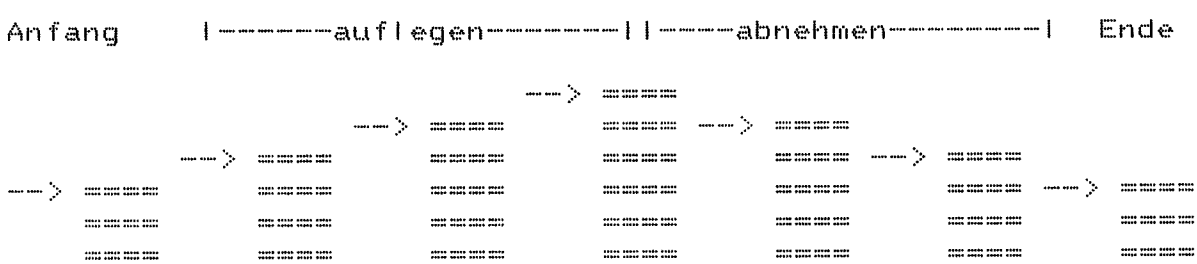


Abb. 1.3.6-1 Tellerstapel

Erst wenn wir alle drei Teller wieder abgeräumt haben, kommt derjenige wieder zum Vorschein, der zuerst obenauf lag. Das ist nicht schwer zu verstehen.

Genauso wie dieser Tellerstapel wird auch der Stack verwaltet, nur daß er nach unten wächst. (Die Teller hängen jetzt gewissermaßen an der Decke.) Die jeweilige Spitze (das unterste Ende!) des Stapels wird in einem besonderen Register vermerkt. Dieses Register heißt Stackpointer (SP, zu deutsch: Stapelzeiger). In der I2DITVM wird der SP durch das kleine Zirkumflex (^) angezeigt. "Vermerkt" bedeutet genauer, daß im SP [illegible] gespeichert

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

ist. Man sagt in einem solchem Fall auch, der SP "zeigt" auf die Spitze des Stacks. Immer, wenn ein Wert auf dem Stack abgelegt werden soll, wird zunächst der Stackpointer dekrementiert (um 1 erniedrigt). Er zeigt dann auf einen freien Platz und dort wird der Wert abgespeichert. Der Befehl dazu heißt PUSH (wegdrücken). Wenn wir dagegen einen Wert vom Stack in den Akku zurückholen wollen, wird der Wert, auf den der SP gerade zeigt, in den Akku geholt und der SP anschließend inkrementiert (um 1 erhöht). Der Befehl dazu ist POP. Alles, was sich unterhalb des TOS befindet, ist quasi weg. Es ist nicht mehr reserviert und kann beim nächsten PUSH einfach überschrieben werden. Abb. 1.3.6-2 veranschaulicht den Stack mit willkürlichen Zahlen. Zu Beginn steht der SP "über" dem Stack, da er ja beim ersten PUSH zuerst dekrementiert wird. Der Stackpointer ist durch "-->" dargestellt.

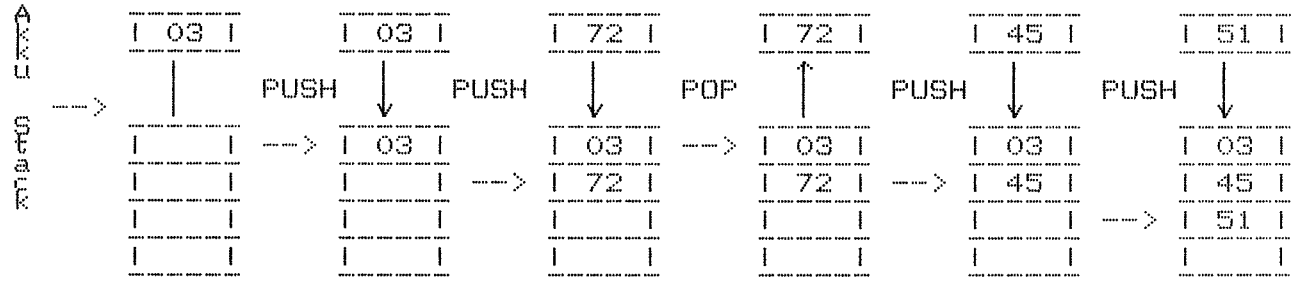


Abb. 1.3.6-2 Der Stack der I2DITVM (auf 4 Zellen verkürzt)

Bedenke bitte, daß in der I2DITVM niedrige Adressen links sind, d. h. "unten" ist dort links. Bedenke ferner, daß sich der Akku zwischen den Spalten 3 und 4 noch durch andere Befehle, die in der Abbildung nicht dargestellt sind, verändert haben könnte. Nach einem POP übernimmt er aber auf alle Fälle den Wert vom TOS. In der Spalte 4 steht die Zahl 72 zwar noch im Stack, aber unterhalb des Stackpointers. Sie wird daher beim nächsten PUSH gnadenlos überschrieben.

Wir wollen nun aml sehen, wie sich der Stack verwenden läßt. Wir nehmen uns nochmal Programm 1.3.2-2 vor und speichern das Zwischenergebnis nun statt in Zelle 0 auf dem Stack.

Programm 1.3.6-1

```
LOAD 06
PUSH            Zwischenergebnis auf den Stack
LOAD 03
ADD 07
MUL (SP)        multipliziere mit TOS
EXIT
```

Der Befehl MUL (SP) multipliziert dabei den Akku mit der Zahl, auf die der Stackpointer gerade zeigt. Diesen Wert werden wir TOS Element oder nur kurz TOS nennen. Auch für die anderen arithmetischen Befehle gibt es eine Variante, die den Akku mit dem TOS Element verknüpfen.

Zum Schluß wollen wir noch ein Programm schreiben, das die Wurzel aus einer Zahl xy zieht. Der Algorithmus dazu beruht auf der Tatsache, daß sich jede Quadratzahl als die Summe der ungeraden Zahlen von 1 bis zu einer gewissen Obergrenze n darstellen läßt. Die Anzahl der nötigen Summanden ist dann die Wurzel der Quadratzahl. Z. B. ist 25=1+3+5+7+9, das sind gerade 5 Summanden. Statt die Summanden zu zählen, kann man auch den letzten Summanden (9) ermitteln, denn er ist gerade um 1 kleiner als das doppelte der gesuchten Wurzel (9+1=2*5).

A s s e m b l e r: Kurs

Programm 1.3.6-2

```
                LOAD 01         Anfangswert fuer ungerade Zahlen
                PUSH            weg damit
                LOAD xy         woraus die Wurzel gezogen werden soll
Wiederhole      Pruefe, ob das TOS Element kleiner als der Akku ist.
                Wenn ja, fuehre den Teil von LOOP bis POOL aus, sonst
                fahre hinter POOL fort.
    LOOP:       SUB (SP)        subtrahiere ungerade Zahl
                INC (SP)        ermittle naechste ungerade Zahl
    POOL:       INC (SP)
                                das TOS Element ist jetzt der letzte
                                Summand
                POP             hole letzten Summand in den Akku
                INC A           ergibt das Doppelte der Wurzel
                DIV 02          durch zwei dividieren
                EXIT            fertig
```

Der Vorteil an der Benutzung eines Stacks ist, daß man sich nicht mehr um Adressen zu kümmern braucht. Wir sagen einfach PUSH und die I2DITVM speichert den Akku automatisch ab. Allerdings dürfen wir auch nicht zu viel pushen, da der Stack begrenzt ist. Wenn du ganz unten (d. h. links) angekommen bist und nochmal pushen willst, liefert die I2DITVM eine Fehlermeldung (ebenso wenn du gar nichts gepusht hast, aber etwas poppen willst). Ein wirklicher Mikroprozessor, insbesondere der Z80, kümmert sich jedoch nicht um solche "Kleinigkeiten", sondern pusht einfach dorthin, wo der Stackpointer gerade hinzeigt, auch wenn du damit dein Programm überschreibtst! Der Stack hat darüberhinaus noch einen anderen Vorteil, der aber erst erklärt werden kann, wenn wir über Unterprogramme sprechen (in BASIC: GOSUB).
Ende Vertiefungsstoff

2. Binäre Informationsdarstellung (Jürgen Freimuth 8000)

In diesem Kapitel geht es um die Frage, wie all die verschiedenen Arten von Informationen, die ein Computer verarbeiten kann, innerhalb des Computers dargestellt werden. Zu diesen Informationsarten gehören zum Beispiel Programme, Texte, Grafiken, Zahlen u.s.w.

Da der Computer ein elektronisches Gerät ist, müssen diese Informationen durch elektrische Größen dargestellt werden. Nun kann man versuchen, Zahlen einfach durch verschieden hohe Spannungen darzustellen. Eine Spannung von 1 V entspricht dann einer Eins, eine Spannung von 2 V einer Zwei u.s.w. In dieser Darstellung sind auch alle denkbaren Werte zwischen den ganzen Zahlen darstellbar. Man nennt diese Art der Informationsdarstellung analog. Diese Technik wurde in sogenannten Analogrechnern für ganz spezielle Aufgaben auch schon verwendet. Für den Bau eines "Mehrzweckcomputers" ist die Analogdarstellung jedoch weniger geeignet, da sich, aufgrund kleiner Ungenauigkeiten, die alle elektronischen Bauteile aufweisen, elektrische Spannungen nicht beliebig genau darstellen lassen, und mit der Ausgabe "Das Ergebnis ist ungefähr 3" wären wohl nur wenige zufrieden. Außerdem soll unser Rechner ja nicht nur Zahlen, sondern auch alle anderen Arten von Informationen verarbeiten können.

Deshalb wurden schon bei den ersten Computern sogenannte digitale Signale verwendet. D.h., eine Spannung soll nur bestimmte Werte annehmen. Bei heutigen Computern sind das im allgemeinen 0 Volt und 5 Volt. Wenn jetzt durch Ungenauigkeiten der Elektronik eine andere Spannung auftritt, behandelt der Computer alle auftretenden Spannungen unter 0,8 V wie eine Spannung von 0 Volt und alle Spannungen über

2,0 V wie eine Spannung von 5 V. In der Elektronik können also größere Spannungsabweichungen auftreten, ohne die dargestellte Information zu verfälschen. Nur wenn die Spannung zwischen 0,8 V und 2,0 V liegt ist ihre Bedeutung nicht definiert, und es treten Fehler auf. Diese Informationsdarstellung, bei der nur zwei (gültige) Zustände möglich sind, wird auch als binäre Darstellung bezeichnet.

Die beiden Zustände 0 V und 5 V werden auch als 0 und 1, Low und High, L und H oder Aus und Ein bezeichnet. Dabei sind 0 und 1 nicht unbedingt als Zahlen zu betrachten, sondern nur als Bezeichnungen für die unterschiedlichen Spannungspegel. Dabei ist es egal, welche Bezeichnung welcher Spannung zugeordnet wird. Im allgemeinen gilt jedoch: 0 entspricht 0 V und 1 entspricht 5 V.

Eine Informationseinheit dieser Art, die nur zwei Zustände einnehmen kann, bezeichnet man als ein Bit (englisch: **bi**nary dig**it**). Mit einem Bit lassen sich bereits Informationen darstellen und übertragen. Zum Beispiel kann der Mikroprozessor dem Speicher anzeigen, ob er in den Speicher schreiben will (1), oder aus ihm lesen will (0), oder der Drucker kann dem Computer melden, ob er gerade beschäftigt ist (1) oder ob er bereit ist Daten zu empfangen (0).

2.1 Zeichen

Mit einem Bit lässt sich ja noch nicht allzuviel anfangen, denn der Computer soll ja Daten aller Art verarbeiten, und dabei sind im allgemeinen mehr als nur zwei Zustände verlangt. Als Beispiel wollen wir versuchen, die 26 Buchstaben des Alphabets darzustellen. Die einfachste Lösung wäre: Wir verwenden einfach 26 Bits (zum Beispiel 26 Leitungen, deren Spannung wir zwischen 0 und 5 V umschalten). Wenn ein "A" dargestellt wird wird das erste Bit high, wenn ein "B" dargestellt wird das zweite u.s.w. Die anderen Bits sind jeweils Low. Das würde durchaus funktionieren, aber für jedes weitere Zeichen, das wir darstellen wollen (z. B. Punkt, Komma, Ziffern usw.), würden wir auch ein zusätzliches Bit benötigen. Und da jedes Bit Platz, Strom und Geld kostet, ist man in der EDV einen anderen Weg gegangen.

Mit einem Bit können wir ja nur zwei Zustände darstellen, aber wenn wir zwei Bits verwenden, dann erhalten wir vier mögliche Zustände (Low/Low, Low/High, High/Low und High/High). Und das tollste ist: Jedesmal, wenn wir ein Bit dazugeben, verdoppelt sich die Anzahl der möglichen Zustände. Mit drei Bits können wir 8 Zustände, mit 4 Bits 16 und mit 5 Bit 32 darstellen. Die Mathematiker sagen, mit n Bits lassen sich 2^n Zustände darstellen (2^n = 2*2*2*... mit insgesamt n Zweiern).

Um unsere 26 Buchstaben darzustellen, brauchen wir jetzt nur noch 5 Bits. Das ergibt 32 mögliche Zustände. Es bleiben also sogar noch 6 Zustände frei, die wir dazu benutzen können Leerzeichen, Punkte oder Kommas darzustellen. Welcher Bitkombination wir dabei welchen Buchstaben zuordnen, bleibt zunächst uns überlassen. Damit aber das Chaos in der Computerwelt nicht allzu groß wird, haben sich die meisten Hersteller von EDV-Geräten auf eine einheitliche Darstellung von Buchstaben, Ziffern, Sonderzeichen und einigem anderen geeinigt. Das Ergebnis ist der sog. ASCII Code. In ihm werden alle Zeichen durch 7 Bits dargestellt. Der Buchstabe "A" erhält dann die Codierung 1000001, "B" ist 1000010, "C" ist 1000011 usw. bis 1011010 für "Z". Näheres dazu erfahrt ihr in Abschnitt 2.3. Vorher geht es nochmal um die einzelnen Bits.

Die Elektronik eines Computers würde sehr langsam werden, wenn sie die Bits einzeln durch den Rechner schaufeln würde. Deshalb ist die Anzahl

der Bits, die auf einmal verarbeitet oder übertragen werden, in jeder CPU festgelegt. Wenn ein Programm für eine bestimmte Aufgabe weniger Bits benötigt, laufen die anderen leer mit. Die Z80-CPU im MTX verarbeitet 8 Bits auf einmal, sie gehört daher zu den 8-Bit-Prozessoren. Einen Block von 8 Bits nennt übrigens Byte. Warum das so ist, weiß heute niemand mehr genau, wahrscheinlich, weil Byte und Bit so ähnlich klingen.

2.2 Zahlen

Im vorigen Kapitel ging es unter anderem um die Darstellung von Buchstaben und Ziffern. Diese Ziffern sind nur eine besondere Art von Zeichen, während Zahlen einen Zahlenwert repräsentieren. So stellen 13 und XIIV die gleiche Zahl dar, obwohl sie aus unterschiedlichen Ziffern bestehen. Im ASCII Code wird die Ziffer "1" (als Zeichen, nicht als die Zahl 1) durch 00110001 dargestellt, die Ziffer "2" durch 00110010, "3" durch 00110011. (Dabei haben wir die Codes mit einer führenden Null auf 8 Bit aufgefüllt. Dies ist die übliche Darstellung von ASCII Codes in Bytes.) Die Zeichenfolge "13" wird dann durch die beiden Bytes 00110001 00110011 dargestellt. Da mit den Zahlen gerechnet werden soll, ist es im Gegensatz zu den Buchstaben und Ziffern nicht egal, mit welcher Bitkombination eine Zahl dargestellt wird. Um das System, das der Darstellung von Zahlen im Computer zugrundeliegt, zu verstehen, wollen wir zunächst unser bekanntes Zehner- oder Dezimalsystem genauer betrachten.

Zur Darstellung aller Zahlen haben wir nur zehn Ziffern zur Verfügung, deren Zahlenwert davon abhängt, an welcher Stelle sie stehen. In der Zahl 333 hat die erste Ziffer von rechts den Wert 3, die zweite den Wert 3*10 und die dritte den Wert 3*10*10. Den Wert der Zahl 333 erhalten wir dann einfach durch Addition dieser drei Teile. Das ist uns so geläufig, daß es lapidar klingt.

Computer verwenden das Zweier- oder Binärsystem zur Darstellung von Zahlen. Dieses Zahlensystem funktioniert wie das bekannte Dezimalsystem, nur ist hier nicht die 10, sondern die 2 Grundlage aller Regeln. Es gibt nur zwei Ziffern, die sich durch die 2 Zustände eines Bits darstellen lassen. Diese beiden Ziffern nennen wir 0 und 1. Auch im Binärsystem entspricht der Wert der ersten Ziffer von rechts dem Wert der Ziffer selbst. Der Wert der zweiten Ziffer wird aber nicht durch Multiplikation mit 10, sondern mit 2 berechnet; der Wert der dritten Ziffer demnach durch multiplizieren mit 2*2. Dazu einige Beispiele:

Binär	Umrechnung	Dezimal
0	**= 0**	**= 0**
1	**= 1**	**= 1**
11	**= 1*2 + 1**	**= 3**
1101	**= 1*2*2*2 + 1*2*2 + 0*2 + 1**	**= 13**

Wie man binäre Zahlen in dezimale übersetzt könnt Ihr in den Beispielen sehen. Recht praktisch ist auch die folgende Tabelle, die die Wertigkeiten der Ziffern in einem Byte darstellt:

A s s e m b l e r: Kurs

```
Bit    7     6     5     4     3     2     1     0
     +-----+-----+-----+-----+-----+-----+-----+-----+
     ! 128 !  64 !  32 !  16 !   8 !   4 !   2 !   1 !
     +-----+-----+-----+-----+-----+-----+-----+-----+
```

Z. B. ist der Dezimalwert von
0 1 0 0 0 0 1 0
1*64 + 1*2 = 66, also der ASCII Code von "B". Nun kommt die Umkehrung.

Die Umwandlung von dezimal in binär geht ganz einfach. Nehmen wir an, wir haben eine Dezimalzahl x. Damit verfahren wir wie folgt:

a) Wir teilen x durch 2 und notieren uns den Rest, der ja nur 0 oder 1 sein kann.
b) Wir ersetzen x durch den ganzzahligen Teil der Division von Schritt a.
c) Die beiden vorigen Schritte werden wiederholt, bis wir bei Null angekommen sind. Die Reste werden dabei nach links fortlaufend notiert, auch wenn sie Null sind. Das Ergebnis ist die aus den Resten gebildete binäre Zahl.

Kompliziert? Dann machen wir ein Beispiel mit x=50.

```
50 : 2 = 25   Rest        0
25 : 2 = 12   Rest       1
12 : 2 = 6    Rest      0
6  : 2 = 3    Rest     0
3  : 2 = 1    Rest    1
1  : 2 = 0    Rest   1         fertig
Ergebnis:            110010
```

Eine Begründung für dieses Verfahren findet ihr am Ende dieses Abschnitts als Vertiefungsstoff.

Übungsaufgabe 2.1-1: Schreibe ein BASIC Programm zur Umwandlung von dezimalen Zahlen in binäre. (Tip: Modulo-Funktion!)

110010 ist also die binäre Darstellung der Zahl 50. Wenn wir diese Bitkombination, mit einer führenden Null ergänzt, jedoch als ASCII Zeichen interpretieren würden, käme "2" heraus. Dies ist ein Zeichen und keine Zahl! Man sieht: dasselbe Bitmuster kann man (als Mensch) verschieden interpretieren. Für die CPU ist das belanglos, sie "sieht" immer dasselbe Bitmuster. Erst wir Menschen (und unsere Programme) entscheiden darüber, ob es sich um die Zahl 50 oder das Zeichen "2" handelt. Es ist sehr wichtig, dies verstanden zu haben. Deshalb solltest du obiges Beispiel noch einmal mit der Zahl 65 durchgehen. Auf der Assemblerebene "verschwimmt" der Unterschied zwischen Zahlen und Zeichen und es macht der CPU überhaupt nichts aus, "A" und "B" zu addieren, wenn wir es so wollen.

Wie alle Daten, werden auch die Zahlen byteweise verarbeitet. Ein Byte hat 8 Bit, das ergibt 2^8 = 256 verschiedene Zustände. Damit lassen sich alle ganzen Zahlen von 0 bis 255 darstellen. Wenn wir größere Zahlen darstellen wollen, müssen wir eben mehrere Bytes dafür verwenden. Die meisten Assemblerprogramme kommen mit einer Zwei-Byte-Darstellung aus. So eine Binärzahl sieht also zum Beispiel so aus: 1001101011011101.

Ihr seid sicher auch der Meinung, daß solche binären Bandwürmer für menschliche Programmierer unzumutbar sind. Der Computer "verträgt" aber keine Dezimalzahlen. Also müssen wir einen Kompromiß suchen: Ein Zahlensystem, das für den Menschen und den Computer einigermaßen verständlich ist. Dieses System ist das Hexadezimal- oder Sechzehnersys-

A s s e m b l e r: Kurs

tem. Statt Hexadezimal sagt man oft auch nur Hex. (Linguisten bevorzugen die Bezeichnung sedezimal, die eigentlich richtig ist, aber sich nicht so recht durchgesetzt hat.) Auch hier gelten wieder die bekannten Regeln für Zahlensysteme, nur ist diesmal die 16 die Basis des Systems. Wir brauchen also 16 Ziffern, von denen uns 0 bis 9 schon bekannt sind. Für die anderen 6 Ziffern verwenden wir Buchstaben:

A bedeutet Zehn
B bedeutet Elf
C bedeutet Zwölf
D bedeutet Dreizehn
E bedeutet Vierzehn
F bedeutet Fünfzehn

Wie nicht anders zu erwarten, wird der Wert dieser Ziffern durch multiplizieren mit 16, 16*16 u.s.w. berechnet:

13A4 = **1***16*16*16 + **3***16*16 + **10***16 + **4** = 5028

Die Hex-Zahl 13A4 kann sich ein Mensch leichter merken, als ihre binäre Form 1001110100100, aber der Computer muß die von uns eingegebene Hex-Zahl genauso übersetzen, wie eine Dezimalzahl. Also könnten wir doch gleich die uns gewohnten Zahlen eingeben. So einfach ist es leider nicht. Da 16 eine Potenz von 2 ist, sind die Hex-Zahlen viel enger mit den binären Zahlen 'verwandt' als die Dezimalzahlen. Deshalb lassen sich binäre und hexadezimale Zahlen sehr leicht ineinander umrechnen. Der Programmierer, der mit Hex-Zahlen arbeitet, erreicht dadurch eine größere 'geistige Nähe' zur binären Welt der Computer-Hardware. Zum Beispiel hat der MTX 500 dezimal betrachtet 32768 Bytes RAM. Hexadezimal sind es glatte 8000 Bytes. In vielen Fällen sind die bei der Assemblerprogrammierung vorkommenden Zahlen nur dann 'runde' Zahlen, wenn man sie hexadezimal schreibt (oder gleich binär). Wenn Ihr erst tiefer in die Assemblerprogrammierung eingestiegen seid, werdet Ihr selbst feststellen, daß dabei die hexadezimalen Zahlen den dezimalen vorzuziehen sind.

Mit der binären Zahl 10110101 möchte ich euch zeigen, wie einfach sich binäre und hexadezimale Zahlen ineinander umrechnen lassen. Die Zahl hat 8 Stellen, paßt also genau in ein Byte. Dieses Byte teilen wir nun in zwei Teile mit je 4 Bit. So eine Gruppe von vier Bit nennt man Nibble. Mit einem Nibble lassen sich 16 verschiedene Zustände darstellen; also zum Beispiel die 16 Ziffern des Hexsystems. Jetzt betrachten wir jedes Nibble als eine eigene Zahl. Das vordere Nibble 1011 ergibt 11 dezimal oder B hex. Das hintere Nibble 0101 ergibt 5 dezimal und hex. Jetzt setzen wir die beiden Nibbles wieder zusammen und erhalten die Hex-Zahl B5. Ihr könnt jetzt 10110101 und B5 in dezimale Zahlen umrechnen, und ihr erhaltet jedesmal die gleiche Zahl (Welche?). Das Verfahren läuft analog wie oben beschrieben, wenn ihr 2 durch 16 ersetzt. Um klarzustellen, daß es sich um eine Hexzahl handelt, schreibt man häufig B5H oder B5h oder #B5 (MTX-Assembler) oder manchmal auch $B5. Bei B5 wäre das nicht nötig, aber wie sieht es mit 13 aus? 13H ist nämlich ___

Umgekehrt ist die Sache natürlich genauso einfach: Wir zerlegen die Hex-Zahl in ihre einzelnen Ziffern, und ersetzen jede Ziffer durch ihre entsprechende vierstellige Binärzahl. Beispiel: 31H = 0011 0001. Falls es euch nicht aufgefallen ist: In ein Byte passen haargenau zwei Hex-Ziffern, das sind ungefähr 2,5 dezimale Ziffern. Die 'geistige Nähe' zur binären Hardware wird schon spürbar.

Übungsaufgabe 2.2-1: Schreibe ein BASIC Programm zur Umwandlung einer

A s s e m b l e r: Kurs

Dezimalzahl in eine Hexzahl. Wenn du die vorige Aufgabe gelöst hast, sollte es nicht allzu schwer fallen. Du mußt jedoch bedenken, daß die Reste jetzt die Werte 0 bis F annehmen können (0 bis 15 dezimal).

Vertiefungsstoff
Begründung der "Restemethode" zur Umwandlung dezimal --> binär:
Wir nehmen als Beispiel wieder die Dezimalzahl 50. Da sie kleiner als 64, aber größer als 32 ist, werden wir 6 Binärziffern benötigen. Da wir diese Ziffern noch nicht kennen, nennen wir sie erstmal allgemein b_5, b_4, ... b_0. Damit können wir also schreiben
$50=b_5*32 + b_4*16 + b_3*8 + b_2*4 + b_1*2 + b_0*1$
Nun teilen wir beide Seiten ganzzahlig durch 2 und erhalten
$25=b_5*16 + b_4* 8 + b_3*4 + b_2*2 + b_1*1$ Rest b_0
Da sich auf der linken (dezimalen) Seite kein Rest ergab, können wir schließen, daß $b_0=0$ sein muß. Nun wiederholen wir diesen Schritt. Auf der linken Seite ergibt sich nun 25/2=12 Rest 1. Auf der rechten Seite erhalten wir $b_5*8 + b_4*4 + b_3*2 + b_2*1$ Rest b_1. Also muß $b_1=1$ sein. Das weitere Vorgehen ist wohl jetzt klar. Mit ein bißchen Überlegung wirst du auch dahinterkommen, wie sich das Verfahren auf die Umwandlung dezimal --> hex abändern läßt.
Ende Vertiefungsstoff

2.3 Der ASCII-Code (Herbert Oppmann, 8522)

ASCII =
american standard code for information interchange
oder auf Deutsch:
amerikanischer Standardcode für Informationsaustausch.

Warum "amerikanisch"? Nun, die Amerikaner waren halt die Ersten, die einen solchen Standard vorgeschlagen und benutzt haben.

Warum "Standard"? Na wenn jeder die Zeichen anders verschlüsseln würde, könnte man Texte nicht so ohne weiteres austauschen.

Und warum dann ASCII-Code, wo doch das C in ASCII bereits Code heißt? Nun, nur von dem ASCII zu sprechen, wäre verwirrend. Und das C als Code aussprechen geht noch weniger (ASCodeII oder irgend sowas Schreckliches). Also nochmal Code, wenn ich auch sonst gegen solche Wiederholungen bin, denn sie zeigen im Allgemeinen, daß derjenige die Bedeutung der Abkürzung nicht kennt (Beispiel zum Schmunzeln aus einem Werbeprospekt von Kaufhof: "Taschenrechner mit LCD-Display-Anzeige").

Bevor noch weiter erklärt wird, gleich mal die

A s s e m b l e r: Kurs

Tabelle 2.3-1

x	0	1	2	3	4	5	6	7
0	NUL	DLE	SPC	0	§	P	'	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	Ä	k	ä
C	FF	FS	,	<	L	ö	l	ö
D	CR	GS	-	=	M	Ü	m	ü
E	SO	RS	.	>	N	^	n	ß
F	SI	US	/	?	O	_	o	DEL

Erläuterungen zur Tabelle:

Die horizontal angetragenen Ziffern sind das höherwertige Nibble (ein Nibble sind 4 Bits), die vertikal angetragenen Ziffern das niederwertige Nibble des Codes. Gleich am Beispiel: Welchen Code hat der Buchstabe "A"? Zunächst schaue ich in der Spalte nach oben und merke mir die 4. Dann schaue ich nach links und klebe die 1 an die 4. Der Buchstabe "A" hat also den Code 41 (hexadezimal) bzw. 16*4+1=65 (dezimal). Um den Bezug zu Basic herzustellen: die Basic-Variable A enthalte den Wert 66. Die Funktion CHR$(A) liefert nun das Zeichen, das den ASCII-Code 66 hat, also 'B'. Umgekehrt: die Basic-Variable A$ enthalte die Zeichenkette 'EGON'. Dann liefert die Funktion ASC(A$) den ASCII-Code des ersten Zeichens des Strings A$ (das ist ein "E"), also 69.

SPC ist die Abkürzung für Space und heißt auf Deutsch Leerzeichen. Das Nichts läßt sich halt schwer darstellen!

Alle anderen mehrbuchstabigen Einträge sind Abkürzungen von Steuerbefehlen. Dazu folgende

A s s e m b l e r: Kurs

Tabelle 2.3-2

Kurz	Hex	Ctrl	Englisch	Deutsch
NUL	00	§	null	nix tun
SOH	01	A	start of heading	Anfang eines Kopfes
STX	02	B	start of text	Anfang des Textes
ETX	03	C	end of text	Ende des Textes
EOT	04	D	end of transmission	Ende der Übertragung
ENQ	05	E	enquiry	Anfrage
ACK	06	F	acknowledge	Bestätigung
BEL	07	G	bell	Glocke betätigen
BS	08	H	backspace	um ein Zeichen zurück
HT	09	I	horizontal tabulation	horizontale Tabulation
LF	0A	J	line feed	eine Zeile tiefer
VT	0B	K	vertical tabulation	vertikale Tabulation
FF	0C	L	form feed	Seitenvorschub
CR	0D	M	carriage return	Wagenrücklauf
SO	0E	N	shift out	
SI	0F	O	shift in	
DLE	10	P	data link escape	
DC1	11	Q	device control 1	
DC2	12	R	device control 2	
DC3	13	S	device control 3	
DC4	14	T	device control 4	
NAK	15	U	negative acknowledge	statt ACK bei Fehler
SYN	16	V	synchronous idle	
ETB	17	W	end of transmission block	
CAN	18	X	cancel	unterbrechen
EM	19	Y	end of medium	
SUB	1A	Z	substitute	
ESC	1B	Ä	escape	"Fluchtsymbol"
FS	1C	ö	file separator	
GS	1D	Ü	group separator	
RS	1E	^	record separator	
US	1F	_	unit separator	
DEL	7F		delete	Zeichen löschen

Die Bezeichnungen sind für die Datenübertragung gedacht gewesen und daher haben für uns die meisten keine Bedeutung. Wichtig dagegen sind die sog. Kontrollcodes. Sie werden dadurch gebildet, daß man von einem Code zwischen 0 und 1F (hexadezimal) in Tabelle 2.3-1 vier Spalten nach rechts rutscht und sich dort den Buchstaben nimmt. Beispiel: von SOH gelangen wir nach "A", also ist SOH gleich Ctrl-A (in der Zirkumflexschreibweise: ^A). Das ist auch der Code, den unsere Tastatur liefert, wenn man die Ctrl- und die A-Taste gleichzeitig drückt. Die Ctrl-Taste bewirkt einfach, daß die Bits 7, 6 und 5 des Buchstabencodes auf Null gesetzt werden. In Basic läßt sich die Umwandlung von Buchstabe zu Ctrl-Code realisieren, indem man vom Großbuchstaben 64 subtrahiert (bei Kleinbuchstaben 96).

Übungsaufgabe 2.3-1: Schreibe ein Basic-Programm, das ein Zeichen von der Tastatur liest und wieder auf den Bildschirm ausgibt. Dabei sollen Control-Zeichen in der Zirkumflexdarstellung ausgegeben werden, also z.B. SOH = CTRL-A als "^A". Das Programm soll sich ständig wiederholen, bis ^C (= BREAK) eingegeben wird.

Übungsaufgabe 2.3-2: Schreibe ein Basic-Programm, das ein Zeichen von der Tastatur liest und dieses in die folgenden Kategorien einordnet:

1) Kontrollzeichen (^A, ^U, ...)
2) Ziffern (0, 1, ..., 9)

A s s e m b l e r: Kurs

3) Großbuchstaben (A, ..., Ä, ö, Ü)
4) Kleinbuchstaben (a, ..., ä, ö, ü)
5) Sonderzeichen (der Rest, also z.B. SPC,".",":")

Das Programm soll die Kategorie des eingegebenen Zeichens ausgeben. Es soll sich ständig wiederholen, bis ^C (= BREAK) eingegeben wird.

Nun gibt es beim ASCII-Code noch eine Schwierigkeit, die bei der Tabelle 2.3-1 nicht sichtbar wird: Da der ASCII-Code ja ein amerikanischer Code ist, sind eigentlich keine deutschen Sonderzeichen eingeplant gewesen, und Sonderzeichen anderer Länder auch nicht. Man behalf sich damit, daß man einige (hoffentlich selten benutzte) Zeichen rauswarf zugunsten der Sonderzeichen. Da im Deutschen nur wenige Sonderzeichen benötigt werden, geht das auch noch einigermaßen. Bei Dänisch oder Schwedisch fängt dann das große Gewusel an. Deshalb möchte ich darauf auch nicht näher eingehen, sondern nur die amerikanisch-deutschen Unterschiede aufzeigen. Eine Zusammenstellung findet sich in folgender

Tabelle 2.3-3

Sonder-zeichen	Hex	im Original
§	40	@ Klammeraffe (ein "a" in einem Kringel)
Ä	5B	[left square bracket (eckige Klammer auf)
ö	5C	\ backslash (umgedrehter Schrägstrich)
Ü	5D	] right square bracket (eckige Klammer zu)
ä	7B	{ left brace (geschweifte Klammer auf)
ö	7C	! bar (ein senkrechter Strich)
ü	7D	} right brace (geschweifte Klammer zu)
ß	7E	~ Tilde (horizontale Schlangenlinie)

Wenn also amerikanische Software dazu auffordert, eckige Klammern oder sonst was einzugeben, muß man beim MTX die entsprechenden Sonderzeichen eingeben, denn die Software "sieht" ja nur den Code (und der ist identisch). Wie das Zeichen auf unserem Bildschirm und der Tastatur aussieht, merkt die Software aber nicht. Einige Programme erlauben auch andere Schreibweisen, wenn die geforderten Zeichen auf diesem Computer anders definiert sind. Turbo Pascal akzeptiert z.B. (. für eckige Klammer auf, .) für eckige zu, (* für geschweifte auf usw..

Problematisch sind die Sonderzeichen, wenn es darum geht, von Groß- nach Kleinbuchstaben oder umgekehrt umzuwandeln: Ä, ö und Ü sollten, da sie für uns ja Buchstaben sind, umgewandelt werden. Eckige oder geschweifte Klammern dürfen aber auf keinen Fall umgewandelt werden! Da die Software prinzipiell aber nicht herausfinden kann, ob die Sonderzeichen auf dem Bildschirm und der Tastatur als Umlaute oder Klammern zu verstehen sind (ich wiederhole mich), geht amerikanische Software stets von Klammern und deutsche Software stets von Umlauten aus. Das ist auch der Grund, warum bei Turbo Pascal die UpCase-Funktion mit Umlauten nicht funktioniert und NewWord in der ausgelieferten Version Worte mit Umlauten nicht als ein Wort ansah.
Vom eben gezeigten Problem abgesehen ist die Umwandlung von Groß- nach Kleinbuchstaben und umgekehrt recht einfach. Von Groß- nach Kleinbuchstabe muß man nur 32 addieren, umgekehrt 32 subtrahieren.

Etwas störend bei der Umwandlung von hexadezimalen Zahlen ist die Tatsache, daß nach der "9" im ASCII Code nicht gleich "A" kommt. Es liegen hier noch sieben andere Zeichen dazwischen. Achte darauf bei folgender

A s s e m b l e r: Kurs

Übungsaufgabe 2.3-3: Schreibe ein Basic-Programm, das ein Zeichen von der Tastatur liest und den ASCII-Code dieses Zeichens hexadezimal ausgibt. Das Programm soll sich ständig wiederholen, bis ^C eingegeben wird.

Übungsaufgabe 2.3-4: (Umkehrung der vorigen Aufgabe) Schreibe ein Basic-Programm, das eine Hexzahl von der Tastatur liest und das zu dieser Zahl gehörende Zeichen auf dem Bildschirm anzeigt (Kontrollzeichen in Zirkumflexdarstellung). Das Programm soll sich solange wiederholen, bis ^C eingegeben wird.

Der ASCII-Code ist ein 7-Bit-Code, d.h. zur Abspeicherung eines Zeichens werden 7 Bits benötigt (plus noch eins, das frei bleibt, macht 8 = 1 Byte). 7 Bits zu nehmen kommt nicht vom Alter des Codes. Wenn man einen Standardcode heute erfinden müßte, würde wohl etwas sehr Ähnliches herauskommen. Hier einige Gründe:

* Wenn man nicht mit einem Steuerzeichen zwischen Groß- und Kleinbuchstaben umschalten will oder (wie beim Baudot-Code) auf Kleinbuchstaben verzichten möchte, braucht man allein für die Ziffern und Buchstaben 2*26+10=62 Zeichen. D.h. man braucht mehr als 6 Bits für ein Zeichen.
* Die in einem 7-Bit-Code darstellbaren 128 Zeichen reichen für die meisten Fälle aus. Wenn mehr Zeichen benötigt werden schaltet man mit einem Spezialzeichen (Escape, Erklärung später) in einen anderen Modus um, in dem die Codes eine andere Bedeutung haben.
* Man kann das noch freie 7te Bit dazu nehmen, um z.B. die Parität (das ist: ob eine Zahl in binärer Darstellung eine gerade oder ungerade Anzahl von gesetzten Bits enthält) darin zu speichern. Dadurch kann man Übertragungs- und Speicherfehler mit einiger Wahrscheinlichkeit erkennen. Oder man läßt das 7te Bit auf Null und setzt es nur beim letzten Zeichen eines Strings. Dadurch muß man keine zusätzliche Information bezüglich Länge oder Position des Endes mitführen. (Dieser Trick wird z.B. bei WordStar und NewWord im Dokumentmodus angewendet).
* Einige Rechner stellen bei ihren parallelen oder seriellen Schnittstellen nur 7 Bits pro Zeichen zur Verfügung. So hat z.B. bei den Schneider CPC's die Centronics-Schnittstelle nur 7 Bits.

Während der Standard schon bei den Sonderzeichen etwas verwässert wurde, schaut es bei den Escape-Sequenzen (s. u.) noch viel schlimmer aus. Nachdem die Druckerhersteller jahrelang vor sich hingewurstelt haben, scheint sich der Epson-Standard doch langsam durchzusetzen. Die Terminalhersteller haben sich jetzt auch einen Standard einfallen lassen, aber der paßt natürlich (natürlich ?!?) nicht zum Epson-Standard. Wir werden also noch eine Weile auf "die große Vereinheitlichung" warten müssen. Was hat es mit den Escape-Sequenzen nun auf sich? Zunächst schickt man dem Gerät Escape, d.h. ein Fluchtsymbol. Das sagt ihm, daß das nächste Zeichen nicht gedruckt werden soll, sondern eine der Möglichkeiten, die das Gerät bietet, auswählen soll. Ob weitere Zeichen nun gedruckt werden oder nicht, hängt davon ab, ob die ausgewählte Funktion noch zusätzliche Informationen benötigt und ist gerätespezifisch. Man spricht deshalb von einer Escape-Sequenz, da sie mit Escape eingeleitet wird und ihre Länge im Allgemeinen nicht bekannt ist.

H a r d w a r e: Test BROTHER M-1509 Drucker

Gerd Rüdiger v.Pezold * Hildesheimer Str.96 * 3 H 1

GERÄTEINFORMATION über den Nadel-Drucker ,,Brother M-1509''

Ich hatte Ursache, mir einen neuen Drucker anzuschaffen. Nach einigen Vergleichen und etlichem Überlegen fiel mir dieser Drucker auf: Er bietet mir das genügend breite Schreibformat um, wie gefordert, DIN-A4-Einzelblätter querformatig verarbeiten zu können. Weil der Neuling schon einen vorgegebenen Platz hatte, mußte das Gehäuse auch sehr platzsparend gestaltet sein. Wegen der vielseitigen Verwendbarkeit kam nur ein Matrix-Mosaik-Drucker in Frage und wegen des Preises nur Nadeln; bezogen auf die Güte des Ausdruckes reichten mir neun Nadeln aus. Geschrieben werden sollte allerdings nicht mit dem Carbon(-Kohle-)band, sondern mit einem Textilband wie bei einer klassischen Schreibmaschine. Wunsch, jedoch nicht Bedingung, war der zweifachgerichtete Papiervorschub, um mindestens plotähnliche Arbeiten durchführen zu können. Zu guter Letzt mußte auch ein (CENTRONIC)Parallel-Rechneranschluß eingebaut sein und die Ansteuerung EPSON entsprechen. Natürlich sollte er so billig wie möglich sein, denn ich wollte erheblich weniger als 2.000,--DM bezahlen!

Inzwischen sind alle neueren Druckermodelle sowohl mit der Schreibwalze (Friktionstrieb) einer Schreibmaschine als auch mit den Stachelrädern (Traktor) für die randgelochten Faltpapiere ausgerüstet. Auf gelegentliche Einzelblattbeschriftung braucht schon lange nicht mehr verzichtet zu werden! DIN-A4-Quer- oder, was die selbe Zeilenlänge erfordert, DIN-A3-Hochformat verkraften schon recht viele Modelle. Oft ist das Gehäuse für den vorgegebenen Platz zu groß.Von den nun noch in die engere Wahl kommenden boten die gewünschten Drucker Textilschreibbänder. Den Datenübersichten und -zusammenstellungen nach schien ich auf den rückwärts gerichteten Papiervorschub verzichten zu müssen. Bei der Überprüfung des Platzbedarfes wurde auch nicht mehr gleichgültig, wie die Steckeranschlüsse verteilt waren; denn bei seitlicher Anordnung hätte womöglich die Frage, ob abgewinkelte Stecker mitgeliefert werden oder wenigstens nachträglich erhältich sind, die Unterbringung entschieden. Das Möbel war schon vorhanden: Der Phonowürfel ,,ALTA''(Fa.IKEA) ist hoch 63, tief 45 und das wichtigste Maß: breit 61cm. Da der Drucker in das Möbel eingesetzt werden soll, ist die lichte Breite auch gleichzeitig die größte gewünschte Gehäusebreite einschließlich Walzendrehknopf und der vielleicht seitlich angeordneten Stecker. Für die Seitenwände werden üblicher Weise 19mm dicke Bretter verwendet, so daß 57cm nicht überschritten werden durften; andernfalls hätte ich den Drucker obenauf stellen oder ein neues Möbel suchen müssen ...

Ziemlich früh fielen mir die Drucker von ,,BROTHER'' auf, weil sie im Verhältnis zu der genutzten Papierbreite geradezu erstaunlich platzsparende Gehäuse hatten! Bei den in Frage kommenden Modellen M-1509 und M-1709 blieben immerhin noch 8cm für Drehknopf und Stecker! Da die Anschlüsse auf beiden Seiten verteilt sind, braucht man unbedingt zu beiden Seiten mehr Platz als das Gehäuse für sich alleine beansprucht. Bei diesen 8cm kann man

überprüfen, ob die Schnüre erforderlichenfalls abzuknicken sind.

Auffällig ist auch die an der Hinterkante des Gehäuses eingebaute Schreibwalze. Wahlweise kann eine Traktor- oder eine Einzelblattzuführung angesetzt werden. Somit haben die Anschlüsse auf der Rückseite natürlich keinen Platz mehr.

Die Überprüfung nach den Papierbreiten ergab,daß die dazu gehörige breiteste Papiereinführung, bei Faltpapierbreiten von 14 3/4'', für die diese Druckergruppen gebaut sind, 16'' beträgt, das sind 406mm! Wenn man nun aber weiß, daß die DIN-A3-Lang- als auch die DIN-A2-Schmalseite 420mm beträgt, das ist Breite eines aufgeschlagenen DIN-A4-Heftes, dann wäre wohl die Frage naheliegnd, warum man gerade hier sich 16mm ersparte! Ich glaube, außer BROTHER waren ein oder zwei, die die Verwendung von DIN-A3-Quer- und DIN- A2-Hochformate erlauben, ohne die lächerlichen, jedoch ärgerlichen 1 1/2cm abschneiden zu müssen!

Inzwischen erwiesen sich die schon erwähnten BROTHER-Drucker als die großen Favoriten, wie wohl leicht festzustellen ist! Ich konnte den M-1509 für knapp über 1.000,--DM über eine Anzeige aus einer Computer-Zeitschrift liefern lassen. Nach dem Auspacken stellte ich fest, daß an der mitgelieferten Geräteschnur schon ein abgewinkelter Gerätestecker angeschweißt war; entsprechende Schnüre oder auch nur Stecker konnte ich selbst in den einschlägigen Fachgeschäften nicht auftreiben (Das war mein vorsorglicher Versuch vor der Anlieferung). Bei dem nun übriggebliebenen Platz bin ich jetzt nicht mehr darauf angewiesen, einen flachen Parallellstecker zu verwenden. Beim Anschließen an den Rechner entdeckte ich auch hinter einer kleinen Schutzkappe den Serienanschluß RS232c(V24). Wer weiß, wozu mir der ein Mal gut ist. Vorne links auf dem in beige gehaltenen Gehäuse sind die üblichen Funktionen an Folientastern einzustellen. Durch Farben sind die Funktionsgruppen deutlich gekennzeichnet. Sogar der schönschriftnahe Schreibzustand kann unmittelbar am Gehäuse eingeschaltet werden, ohne über den Rechner programmieren oder einstellen zu müssen. Bei schönschriftnaher Schreibweise (NLQ) entsteht ein schreibmaschinenähnliches Schriftbild, das recht angenehm lesbar ist, obwohl der Unterschied zum Typenanschlag immer noch deutlich erkennbar ist. Das wie gewünscht seidene Schreibband befindet sich in einem langen Gehäuse, dessen Länge der Schreibbreite, das ist etwa der Langseite einer DIN-B4-Seite, entspricht.

Alle zu Anfang genannten Forderungen sind durch dieses Modell erfüllt, nicht nur der 420mm breite Einzelblatteinzug, sondern sogar ein Steuerbefehl, mit dem das Papier auch schrittweise zurück bewegt werden kann! In den Übersichten über die technischen Daten sowohl in den Prospekten als auch im Handbuch fehlt dazu jeder Hinweis! Wohltuend empfand ich den verhältnismäßig leisen Druckvorgang, bei dem noch eine Unterhaltung in Zimmerlautstärke möglich ist (laut Handbuch 55dB). Auf den Rat von Herbert Herberg/zur Nedden hin stellte ich den Drucker auf einen etwa 1/2 cm dicken Filzstreifen, wie er sonst für Schreibmaschinen benutzt wird, was in der Tat eine zusätzliche Dämpfung des Schalls bewirkte.

H a r d w a r e: Test BROTHER M-1509 Drucker

Die Bedienung der Maschine ist, im Phonomöbel eingesetzt, bequem im Sitzen möglich und man kann ohne weitere Verrenkungen auch gleich lesen was geschrieben wurde. Neben der Wahl des Daten- und Steuerkabelanschlusses, kann auch an den Dip-Schaltern die EPSON- oder IBM-Ansteuerung sowie unabhängig davon auch der EPSON- oder IBM-Zeichensatz eingestellt werden. Ein eingebauter Pufferspeicher (3KByte) kann wahlweise für beliebig selbst eingegebene Zeichen (Download) genutzt werden. Mit einer von zwei kleinen Nachrüstkarten kann der Drucker auch 4 verschiedene Schriftarten verwirklichen oder statt der 4. den Zeichenpufferspeicher um 16KByte erweitern. Das Handbuch ist in einem recht ordentlichen Deutsch geschrieben und ausführlich illustriert. Die Befehle sind brauchbar systematisch gegliedert und erklärt. Lobend erwähnen muß ich die erklärenden Beispiele, die die Wirkungsweisen meist ausreichend veranschaulichen.

Die Ansteuerung scheint mit dem EPSON-MX80/100 einigermaßen übereinzustimmen, mindestens kann man in diesem Modus sofort ausdrukken. In NLQ-Modus werden die Zeilen ein klein wenig schief, kombiniert mit Elite (12/") geradezu unerträglich schräg! Auf meinen Wunsch hin versuchte Andreas Illgner den Papier-Rückwärtsschritt zu verwirklichen. Das gelang ihm mit Punktbefehlen. Hier die Erläuterung und ein Beispiel von Andreas:

Ein Drucker-Custom mit dem Code für nn Rückschritte muß wie folgt belegt werden:

Vor dem Anfang der betroffenen Zeile im Text in der ersten Spalte folgendes eingeben: .XQ1B6A24

Der Drucker setzt dann um den festgelegten Betrag zurück. Die 24 am Ende des Punktbefehles ist der Betrag für eine Zeile (1/6"), 48 (=2*24) für zwei usw.. An der Stelle, wo der Rückschritt erfolgen soll, wird im Text ^P^Q eingegeben.

.XQ1B6zz	Ein Punktbefehl in NW, zz bestimmt den Zeilensprungbetrag (zz =24 für eine Zeile 1/6Zoll).
^P^Q	Diese Markierung im Text veranlaßt den Zeilensprung.
.lm5	Ein Punktbefehl in NW, Setzen des linken Randes. Hier wird der linke Rand auf Spalte 5 gesetzt.

Ein Punktbefehl hat als erstes Zeichen einen Punkt. Erscheint ein Punkt in der ersten Spalte einer Zeile, so erwartet der Rechner einen entsprechenden Befehl und führt ihn aus wenn er ihn fand und verstand.Beispiel:

.XQ1B6A24	Einstellen des Druckers: Zeilensprung rückwärts um 1/6"
1.Spalte^P^Q	Text mit Sprungmarke
.lm10	folgender Text beginnt ab Spalte 10

<u>H a r d w a r e: Test BROTHER M-1509 Drucker</u>

2.Spalte Text

1.Spalte Ausdruck
2.Spalte ohne rückwärtigen Zeilensprung

1.Spalte 2.Spalte Ausdruck mit rückwärtigen Zeilensprung.

In dem Sinne ,,nichts ist vollkommen'' beobachtete ich zwei Mängel. Die Andruckwalze des Friktionstriebes läßt sich nicht lösen wie das sonst bei Schreibmaschinen und den meisten Druckern üblich ist. Ein schief eingespanntes Blatt muß wieder hinausgedreht und sorgfältig genau eingeführt werden. Auch ist die Papiereinführung in den Traktor nicht sehr narren(laien)sicher. Ein dritter Mangel betrifft die Bedienung der DIP-Schalter: Zwar sind die Schalter von oben leicht zugänglich, unter der Führungsbahn des Schreibkofpes mit einer Staubkappe abgedeckt, wurde einem Ausrutschen einer Einstellhilfe (Schraubendreher) nicht vorgebeugt, Verletzungen der Platinenbestückung sind allzu leicht möglich.

Zusammenfassung:
Der Matrixdrucker ,,Brother M-1509'' ist die etwas schwächere Ausführung gegenüber ,,M-1709'' in der 136cpl-Klasse. Statt der üblichen 16" (406mm) Papiereinführung haben beide die seltenen 16 1/2" (419mm), womit die 420mm breiten DIN-A2 im Hochformat und DIN-A3 im Querformat Einzelblätter verarbeitet werden können. Verglichen mit vergleichbaren Modellen anderer Hersteller erweisen sich die Gehäuseabmessungen als phantastisch schmal; Verwendung von Videokleinmöbeln als Druckermöbel bieten sich hierbei geradezu an, insbesondere wenn Papierver- und -entsorgung gelöst sind. Die Lärmentwicklung ist selbst bei Geräuschempfindlichkeit gut erträglich. Für Datenübertragung sind sowohl CENTRONIC als auch V24 bereits eingebaut und mit den von oben leicht zugänglichen DIP-Schaltern anwählbar. Mit diesen Schaltern wird sowohl der Steuermodus (IBM/EPSON) als auch der gültige Zeichensatz (IBM/EPSON) festgelegt. Bei EPSON-Modus scheint die Ansteuerung mit ,,EPSON MX80'' hinreichend übereinzustimmen. Mit Hilfe des reichhaltigen Befehlssatzes kann sogar das Papier in beiden Richtungen bewegt werden. Die Schrifttypen können nachträglich um drei oder vier erweitert werden, statt des vierten ist die Erweiterung des Puffers um 16KByte als Option möglich. Dieser kann auch als Speicher für selbst festgelegte Zeichen (DOWNLOAD) umgeschaltet werden. Die Preisspanne liegt zur Zeit bei 1.000,-- bis 1.500,--DM. Trotz kleiner Mängel scheint das Gerät sein Geld wert zu sein.

Zum Schluß sei auf das erste Sonderheft von ,,COMPUTER persönlich'' hingewiesen. Dessen Leitthema ist die Textverarbeitung. Einige Drucker als Vertreter ihrer Preisklassen wurden mit Testberichten vorgestellt. Hierbei wurde der größere ,,Verwandte'' M-1709 vorgestellt. Dieser Test erschien in der Ausgabe 6 vom 4. 3. 87 in ,,COMPUTER-PERSÖNLICH'' und wurde unverändert in das Sonderheft übernommen.