CLUBINFO

28. AUSGABE

KONTAKTADRESSE : CLUB 80 / ALEXANDER SCHMID / ST. CAJETAN STR. 38 VII / 8000 MÜNCHEN 80
TEL.: 089 / 495328

INHALTSVERZEICHNIS

Neues vom Vorstand

Vielleicht hat der eine oder andere von Euch unsere Anzeige in der c't gelesen. Bis jetzt haben sich immerhin acht Interessierte gemeldet, aber nur einer, Detlef Behrendt, hat schon den Sprung ins kalte Wasser gewagt. Er hat ein Genie I mit zwei Laufwerken und einen NEC P6. Die Amateurfunker können sich schon mal auf einen Neuen in Ihren Reihen freuen, das steht nämlich u.a. auch bei seinen Interessen. Mit Hardware- und C-Begeisterten wird er sich wahrscheinlich auch gut verstehen. Herzlich willkommen unter den letzten der Tafelrunde.

Die Infos lassen immer länger auf sich warten und ich danke Euch, daß soviel Vertrauen in den Club habt und weder ausgetreten seid, noch mir eine Bombe geschickt habt. Ich glaube, Ihr werdet DIESEN Grund aber mit Sicherheit verstehen und nicht doch noch auf die Barrkiaden gehen. Unser Jens hat nämlich geheiratet und hat sich in den Flitterwochen mit Recht um alles andere als um den Club gekümmert. Auch hierzu nochmal offiziell herzlichen Glückwunsch und alles Gute. Töpfe, Pfannen und Socken wird er inzwischen wohl genug haben, über rein schriftliche Beileidsbekundungen freut er sich bestimmt genauso.

über Heinrich Betz haben wir jetzt endlich eine Möglichkeit gefunden, das Info zu einem annehmbaren Preis drucken zu lassen. Ohne 'höhere Gewalt' wäre das Info also pünktlich erschienen und das nächste tut das dann wohl hoffentlich mal wieder. Wenn nicht, möchte ich Euch schon mal fröhliche Weihnachten und ein gutes Neues Jahr wünschen.

Soviel vorläufig vom Guru.

Die Leiden des jungen Helmuts

Unser allseits bekannter und beliebter Helmut Bernhardt hat mir in seinem letzten Brief sein Leid über eine leider scheinbar weit verbreitete Unsitte im Club geklagt. Da schreibt er als einer der wenigen wirklich Aktiven viele viele interessante Artikel und bekommt zum Dank viele viele Anfragen, auf die er stundenlang antworten muß (was er, wie er sagt, gerne tut), aber auf die er außerdem noch aus der eigenen Tasche das Porto draufkleben darf. Findet Ihr das fair ? Da wird er auch noch dafür bestraft, daß er was für's Info und den Club tut. Was würdet Ihr davon halten, wenigstens das Rückporto beizulegen ? Vom Arbeitsaufwand spricht ja gar keiner (den könntet Ihr bei Helmuts Durst sowieso nicht bezahlen), aber bei den vielen Anfragen kommen für ihn leicht zwei bis drei große Blaue im Jahr zusammen. Als Alternative könnte der Club natürlich in solchen Fällen die Portokosten übernehmen, oder den Beitrag nach der Menge der geschriebenen Artikel berechnen. In beiden Fällen würden die Beiträge wohl für die meisten Mitglieder eher steigen als sinken. Überlegt es Euch mal, wie wir das regeln könnten. Um Meinungen wird gebeten.

Euer Erster

Alle Jahre wieder

gibt es neue Gesichter im Vorstand des Club 80.
So ist es auch dieses Jahr nicht anders. Aproppos Gesicht, da ich erst jetzt dazu gekommen bin zu schreiben wird dieser Text erst im Info NR. 28 erscheinen, so das ihr mein Gesicht schon kennt. Nun aber zu meiner Vorstellung. Ich bin der neue Hardware-Koordinator des Club's. Wie der Name schon sagt, soll ich Hardware koordinieren.(was immer das heissen mag.) Aber dazu später. Erst zu meiner Person. Ich bin erst seit Januar im Club so das viele mich noch nicht kennen. Da ich nicht mehr weis was ich in meiner Vorstellung als neues Clubmitglied geschrieben habe hier noch einmal das wichtigste, meine Hardware. Ich besitze ein Genie IIIs mit 4 Laufwerken sowie einer 20 MB-Festplatte und einer 10-MB in vorbereitung. Das Gerät hat 1 MB Speicher sowie eine 256 KB Grafik. Als Betriebssystem benutze ich G-Dos (Calva-Dos von Arnulf) sowie CP/M.
Außerdem habe ich noch ein Genie II mit 2 Laufwerken (alles 80 Trk), 80-Zeichen-Karte und HRG B1 sowie einigen Erweiterungen (der G II befindet sich in einem PC-Gehäuse und der IIIs in einem Tower.)

Der Haupteinsatz meines Rechners besteht darin Hardwareänderungen einzubauen und die Software dazu zu schreiben. Wobei die Software allerdings überwiegt.Klingt zwar blöd als Hardware-Koordinator aber Software zu schreiben ist billiger. (Ach so: Ich schreibe fast ausschließlich in Assembler) Also wenn Ihr Probleme habt könnt Ihr euch in beiden Fällen an mich wenden.

Nun aber zu meiner Aufgabe im Club.
Erstens: Wenn Ihr irgendwelche Vorschläge zu Hardware-Projekten habt, die bitte sofort an mich. Ich werde dann die Sache ans laufen bringen.
Zweitens: Es ist vor seeehr langer Zeit mal ein ECE-Projekt angefangen worden, welches dann im Sande verlaufen ist. Ich finde das schade denn es ist eine schöne Sache wenn unser Club etwas eigenes hat auf das wir stolz sein können. Es ist abzusehen das dieses Projekt, etwas moderner, aber trotzdem gut, wieder aufgenommen werden kann. Und diesmal sind die, die sich mit der Herstellung der Platienen befassen zuverlässiger als damals. Die genaue Vorstellung wird Helmut geben wenn es soweit ist. Aber Trotzdem schon jetzt meine Frage: Besteht noch (oder erneut) Interesse an so einem Projekt ?
Wenn ja dann sofort bei mir melden (Postkarte genügt). Dann kann ich sehen ob es sich lohnt das Projekt wieder aufzunehmen was ich mir sehr wünsche.(Ich bin auf jeden Fall dabei).

Soweit dazu. Ich selber habe auch meine Vorstellungen.
Da wäre zum einen eine Grafikkarte die's in sich hat. Ich stelle mir da vor: 512 X 512 Punkte Farbgrafik, eigenes RAM sowie ein eigener Prozessor der die Sache Koordiniert. Der Vorteil ist, die Karte wird über Ports gesteuert so das die Software für alle Rechner gleich sein wird.
Außerdem Spiele ich mit dem Gedanken eines Video-Digitalisierers. Eventuell in Verbindung mit der Grafikkarte.
Und eine Sprach-Ein-Ausgabe wäre ein schönes Projekt was zu realisieren Spaß machen würde.

Also Schreibt mir was das Herz begehrt. Ich werde mich um den Rest kümmern.

Noch etwas in anderer Sache. In einem der letzten Info's, ich weiß nicht mehr in welchem, fragte Arnulf(Hallo Arnulf) nach einer G IIIs Ecke, ich wäre sehr ineressiert zumal sich die Anzahl der G IIIs Besitzer mit meinem Eintritt um grob gerechnet 100 % erhöht hat.(das sagt der zweitschönste G-IIIs-Besitzer).

Andreas Magnus

F r ö h l i c h e n G e b u r t s t a g !

Hallo Ihr Freunde des CLUB 80,

wieder einmal ist es soweit. Das Info ist erschienen und es kann losgehen mit den vielen Grüßen und Wünschen zum Geburtstag.
Auch diesmal wieder appelliere ich an Euch, schreibt mir, wenn Ihr Euch auf der nachfolgenden Liste vermißt. Es hat sich bisher zwar noch niemand beschwert, daß er hier noch nicht vertreten war, doch ich bin sicher, daß ich nicht von allen Mitgliedern die Daten habe.

Meine zweite Bitte lege ich Euch besonders nahe.

L e u t e , s c h r e i b t A r t i k e l

Wie ich Euch im letzten Info schon erklärt habe, besteht das Info nicht nur aus Witzen und Terminen, sondern vor allem aus E u r e n Artikeln.

So, aber nun zum eigentlichen Grund dieses Artikels.

Im Monat S e p t e m b e r hatten Geburtstag:

Rychlik Andreas	11.09.
Sopp Arnulf	23.09.

Im Monat O k t o b e r haben Geburtstag:

Meklenburg Heinz Dieter	01.10.
Magnus Andreas	17.10.
Held Manfred	30.10.

Im Monat N o v e m b e r haben Geburtstag:

Krispin Michael	02.11.
Hentz Werner	03.11.
Mühlenbein Klaus-Jürgen	07.11.
Schäfer Walter	08.11.

Jetzt wird das Jahr noch voll gemacht.

Im Monat D e z e m b e r hat Geburtstag:

Rinio Gerd	03.12.

Der Vorstand gratuliert allen genannten und ungenannten, hier aufgeführten und nicht aufgeführten, freudigen und nicht freudigen "Geburtstagskindern" ein fröhliches Wiegenfest und alles Gute auch weiterhin.

Eure Jutta

-- Termine -- Termine -- Termine --

Nächster Redaktionsschluß ..... Mitte / Ende Dezember

Bitte denkt an kleinere INFO-Artikel (ca. 1 - 4 Seiten) für unser Clubinfo !!

Hallo liebe Clubmitglieder,

als neues Clubmitglied möchte ich mich kurz vorstellen.

Ich heiße Christof Neumann, bin 28 Jahre alt und von Beruf Bauingenieur (Wasserwirtschaft/Tiefbau).

Durch den Beruf kam ich durch Zufall günstig zu einem **Tandy Model II**, der mir mit seinen zwei 8"-Laufwerken und einem dazuerworbenen Typenraddrucker (**JUKI 6000**) seit 1987 hauptsächlich für die Textverarbeitung gute Dienste leistete. Hierzu verwende ich das gute alte Wordstar (gepatcht für den Juki-Drucker) unter CP/M.

Eigene Programme erstelle ich in MBASIC; ich würde aber auch gern in PASCAL oder C programmieren, wenn ich es für das Model II bekommen kann.

Nachdem seit Mitte '88 der Rechner defekt war, hätte ich ihn in meiner Verzweifelung (denn niemand konnte mir helfen) fast zum Mond, oder doch zumindest in die Wüste geschickt, bis ich durch eine kleine Anzeige auf den 'CLUB 80' aufmerksam wurde. So kam es, daß er seit Anfang Mai durch tatkräftige Hilfe aus dem Club wieder läuft, und wenn er nicht gestorben ist ....

Mit dem Wunsch auf gute Zusammenarbeit grüße ich alle Clubmitglieder!

C. Neumann

Hallo Jens,

ich möchte mich hiermit dir und den anderen Clubmitgliedern vorstellen:

Name: Kemmer Jürgen

Alter: 19 Jahre

Adresse: Wohnheim: (nur Werktags)
1. Rangierbahnhof Bau 3
8500 Nürnberg 50
Tel.: 0911 / 48 20 90
2. normaler Wohnort:
Dorfberg 7
8701 Sulzdorf
Tel.: 09334 / 10 50

Beschäftigung: Ausbildung zum Informationselektroniker in Nürnberg bei der DB (bis nachstes Jahr).

Hobby's: Computern, Lesen (meistens Science Fiction)
Radio Hören, Experimentieren mit Eletronik

Hardware: Schneider CPC 464 (64k)
9-Nadel-Drucker (EPSON LX-800)
3'-Floppy mit 180K (Schneider-System-Laufwerk)
5 1/4'-Zweitlaufwerk mit 180K
Fischertechnik Computing Interface:
4 Motorausgänge,8 Digital- u. 2 Analogeingange
D/A-,A/D-Wandlerkarte fuer PIO-Karte
Eprom-Karte für 8 Steckplätze

Hallo Club 80-Freunde,

seid gegrüßt und gestattet, daß ich mich vorstelle:
Ich heiße Harald Hirsekorn, gehöre zum Jahrgang 1927, wohne in 2359 Henstedt-Ulzburg und übe den Beruf eines Elektro-Mechanikers aus.

Vor drei Jahren habe ich mir einen Sharp MZ-3541 angeschafft. Seitdem versuche ich, einigermaßen vernünftige Programme unter CP/M zu schreiben. Der Erfolg ist allerdings recht dürftig. Vielleicht verhilft mir der Kontakt zum Club-80 und seinen Mitgliedern dazu, meine Programmierkenntnisse zu verbessern.

Mit freundlichen Grüßen
Harald Hirsekorn

Harald Hirsekorn

Hallo Clubfreunde, -kameraden, -mitglieder,

als relativ neues Clubmitglied (ab Heft Nr, 23) möchte ich mich stichwortartig vorstellen:
Jahrgang 1938, Berufssoldat, Techniker, verheiratet, 2 Kinder m.

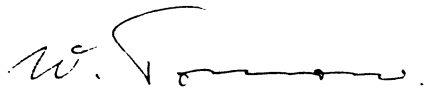
Auf den "Club" gekommen durch Helmut Bernhardt bei einem CP/M-Usertreffen in Kiel. Durch aktive "Hardware-Tips" von Hartmut Obermann und Harald Mand sowie der kameradschaftlichen Stimmung und spontaner Hilfsbereitschaft beim "Nordlichtertreffen" zum Dabeibleiben verpflichtet.

Meine Anlagen:
- Speedmaster 5.3 (Genie II S baugleich) mit 1x 40 Trk SS, 1x 40 Trk DS, 1x 80 Trk SS 5.25" Laufwerke und ein 8" DS Laufwerk sowie ITOH 8510 A Drucker und Akustikkoppler.
- TRS-80 Mod. 4/P mit 2 umschaltbaren 40/80 Trk DS 5.25" Laufwerken und Typenraddrucker Juki 2200.
- ATARI 1040 ST mit zusätzlichem 40/80 Trk 5.25" Laufwerk und einen STAR LC 24-10 Drucker.
- Software für alle Rechner satt.

Aufgrund meines zweiten Steckenpferdes, das der Astronomie, nutze ich meine Rechner mit Astronomieprogrammen ernsthaft, als Schreibmaschine privat als auch dienstlich und zum Spaß als Spielkamerad.

Vom Club erwarte ich, daß er weiterhin bestehen bleibt und sich auch andere (mich eingesclossen) an der Veröffentlichungen von Beiträgen für unsere Clubzeitung aufraffen. Denn dann bleiben auch diejenigen bei der Stange, die bisher alles "gemanagert" haben.

In diesem Sinne Euer



(Wilhelm Tornow)

Hallo Clubmitglieder

Ich möchte mich kurz vorstellen.

Mein Name ist Stefan Nitschke und wohne in Walzbachtal 1 (bei Karlsruhe). Ich bin 25 Jahre alt und studiere Physik an der Universität Karlsruhe.

Zu CP/M kam ich durch den günstigen Einkauf eines ECB-Bus Rechners Anfang diese Jahres. Es stellte sich jedoch, für mich, als schwierig heraus, Software für CP/M-Rechner zu bekommen, da meine Bekannten Atari ST oder MSDOS Rechner benutzen.

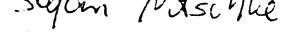
Das Betriebssytem meines Rechners (ZDOS) ist CP/M 2.2 kompatibel. Meine Aktivitäten am Rechner beschränken sich derzeit, wegen Softwaremangel, auf Textverarbeitung mittels Wordstar.

Interresse habe ich an fast allen (Höheren)Programmiersprachen, und Hardwareprojekten. So würde mich das Einbinden einer Harddisk (Soft/Hardware), vielleicht mittels eines RSM-Modules, welches während des booten eingeladen und anschließend aktiviert wird, desweiteren das Betriebssystem MP/M und Numerik interessieren. Für Numerikprogramme habe ich hier noch einen Arithmetikprozessor 8231A von Intel herumliegen, welcher auf Anschluß wartet.
Ich selbst kenne CP/M zu wenig um eigene Hard/Software Projekte zu starten.

Mein Rechner besteht aus:

CPU-Karte	:	Z80 mit 4Mhz Takt von Janich+Klass HKM 105X
Floppycontr.	:	JK 82 FDC 8/5 ebenfalls von Janich+Klass
Speicher	:	64 KBytes Eigenbau, 2 x 5.25'' 40-Spur-Laufwerke
Terminal	:	Infoton nicht graphikfähig
Drucker	:	Speedy 100-80

Mit Grüßen



Hallo Clubfreunde,

als weiterer CP/M-Benutzer möchte ich meine Person und meine hobbybezogene Vergangenheit und Zukunft vorstellen.

Mein Name ist Norbert Giese, 38 Jahre alt, und ich wohne in Gärtringen in der Nähe von Böblingen. Von der Herkunft bin ich eher ein "Nordlicht". Gelernter Starkströmer (etwas zum Anfassen), Elektroniker (lang ist es her), Außendiensttechniker für "etwas größere Computer", Planer (da muß man seine Fehler nicht selber ausbaden), heute Technischer Schreiber (Papier kann sich nicht wehren); kurz gesagt, vom Seitenschneider zum elektronischen Bleistift.

Zum Hobby der Computerei bin ich über einen programmierbaren Taschenrechner (TI-59) etwa 1982 zu einem SHARP MZ-80B nebst originaler Peripherie gekommen. Nachdem ich grob verstanden hatte, was solch ein Ding macht, folgte CP/M 2.2 und diverse Ausflüge in verschiedene Programmiersprachen. Dann wollte ich auch noch wissen, wie ein solches Ding arbeitet und griff wieder zum schon verstaubten Lötkolben. Eine selbstentwickelte doppelte serielle Schnittstellenkarte für den MZ-80B wurde gebaut, ein Akustikkoppler und Ausflüge in diverse Mailboxen folgten. EPROMs wurden geändert, das BIOS wurde vollständig disassembliert, kommentiert, und geändert.

Nun liegt noch eine ECBus-1Mbyte-RAM-Karte auf dem Tisch und muß angeschlossen und ins BIOS implementiert werden, eine Typenradschreibmaschine wartet noch darauf, als "LST:" angesprochen zu werden, eines der beiden 40-Spur Laufwerke soll noch zwei 80-Spur Slimline-Laufwerken weichen, die Graphikkarten des MZ-80B sind auch noch ungenutzt, und, und, und.....

Da dies alles für mich nur ein Hobby ist, sind alle in der Vergangenheit aufkommenden Sehnsüchte nach einer hochintegrierten, lötkolbenfeindlichen Hochleistungs-Produktionsmaschine bisher erfolgreich unterdrückt worden.

Weitere Hobbys: Fotografie mit Schwerpunkt Bildgestaltung, um die andere Gehirnhälfte auch etwas zu trainieren, regelmäßig Standard- und Latein-Tanz zusammen mit der "besseren Hälfte", als ausgleichende Gerechtigkeit für die langen Nächte vor dem Computer.

Ich freue mich auf eine kreative Zusammenarbeit.

* Ein letzter Zehnzeiler *

(Determinanten-Berechnung)

Hallo, Betriebs- und Volkswirte und die es werden wollen! Leider kommt ihr in der Berufspraxis nicht ganz um Mathe und schon gar nicht um die Berechnung von Matrizen (in der Tat: ohne t!) und Determinanten herum.

Nachdem sich einer meiner Schüler eine Zeitlang mit dieser endlosen Zahlenrechnerei herumgequält hatte, baute ich ihm hierfür ein Prgrämmchen - wohl wissend, daß es hierfür natürlich längst eine Anzahl Programme gibt; doch wem das Programmieren ein lieber, aber auch nützlicher Zeitvertreib ist (sogar in BASIC!), der gibt sich erst zufrieden, wenn er eine Aufgabe selbander gelöst hat, und holt nicht Fertiges vom Markt bei ("hob-by" kommt eben nicht von "bei-holen" <Schiffahrt>, sondern von "hob<el>-by-yourself"...)

Der Algorithmus hat einige Tücken. Das Programm mußte deshalb durch eine Anzahl variantenreicher Versuche mit mehrreihigen Determinanten von Hand getestet werden; ich probierte drei- bis fünfreihige aus. Das Ergebis findet ihr nachstehend. Man kommt mit einer einzigen "Subroutine" aus, nämlich für den Fall, daß sich in der Hauptdiagonalen eine Null befindet. Und es geht genau so schnell wie in "TURBO-PASCAL". Wer es nicht glaubt, probiere es aus! (Für den Zeitvergleich bei einer 20-reihigen Determinante will ich mich allerdings nicht verbürgen! Aber wer braucht die schon!)
(Für die Mathematiker: Es wurde nicht der "Laplace'sche Entwicklungssatz" angewendet, sondern das "Dreiecksverfahren", wobei alle Elemente unterhalb der Hauptdiagonalen zu Null gemacht werden und das Produkt der dabei entstandenen neuen Elemente der Hauptdiagonalen den Wert der Determinante ergibt. Letztere Methode finde ich eleganter; sie führt auch schneller zum Ziel.)

Der Eleganz des "Dreiecksverfahrens" ist es auch zu verdanken, daß sich das Programm letztenendes in 7/10 Zeilen unterbringen ließ, wie ihr seht (die Kommentare natürlich nicht mitgezählt). Die Flasche guten Pfälzer Weins dafür (vgl. das uralte Preisausschreiben) habe ich mir selbst spendiert. Eine ebensolche spendiere ich demjenigen, der es (in BASIC) kürzer macht... Gruß! - *Ka-Det*

Und hier das Programm:
(10 Druckzeilen - aber nur 7 Programmzeilen!)

```
10 CLS:DEFINTI-Z:INPUT"Anzahl Reihen der Determinante ";R:DIMA(R,R):PRIN
T:PRINT"Eingabe :":PRINT:FORZ=1TOR:PRINT"Zeile";Z:FORS=1TOR:PRINT"   Spa
lte";S;":";:INPUTA(Z,S):NEXTS:PRINT:NEXTZ
20 '
30 'Spalten unterhalb der Hauptdiagonalen nullen:
40 '
50 FORS=1TOR-1:IFA(S,S)=0THENGOSUB150
60 FORJ=0TOR-S-1:F=A(S+J+1,S)/A(S,S):A(S+J+1,S)=0:FORI=1TOR-S:A(S+J+1,S+
I)=A(S+J+1,S+I)-F*A(S,S+I):NEXTI,J,S
70 '
80 'Berechnung der Determinante:
90 '
100 D=1:FORI=1TOR:D=A(I,I)*D:NEXT:PRINT
110 PRINT"Die Determinante hat den Wert D = ";D*(-1)^X:PRINT:PRINT:END
120 '
130 'Subroutine "Zeilentausch"
140 '
150 FORI=S+1TOR:IFA(I,S)<>0THENGOTO160ELSENEXTI
160 FORU=1TOR:CMD"F=SWAP",A(S,U),A(I,U):NEXT:X=2*(I-S)-1:RETURN
```

* * * NELLEMAKELLO * * *

(Ein Beitrag zur Europa-Wahl!)

Wieviele Sitze erhält der **CLUB 80** im Europa-Parlament? Wird er sich gegen andere Clubs, Vereine, Lobbys, Parteien und was es sonst noch an Interessengemeinheiten gibt durchsetzen können?
Das ist eine schwerwiegende Frage! Wir wissen nur, daß wir N=70 Stimmen erhalten (Stand Januar 89) - nämlich die unseren! Die "Konkurrenz" scheint mir (was die Masse betrifft) stärker... "Köpfchen" zählen leider nicht nach unserem Recht. So läßt sich's leichter zählen. Sonst müßte man unsere Mitgliederzahl ja "hochnehmen"... (auf den Arm?? Gott bewahre! Ich meinte natürlich: "potenzieren".)
Doch auch den Kleinen soll und kann geholfen werden.
Dafür sorgte das Köpfchen eines Belgiers namens *Victor d'HONDT*. Er erfand 1882 das

Höchstzahlverfahren.

Es fand Eingang in das Bundeswahlgesetz §6, Abs.1. Ich will es euch und mir ersparen, die "Erläuterungen" dazu zu zitieren. Vielleicht verstehst du, lieber Leser, meine folgende:
Die auf die Parteien (oder Personen) auf einer Wahlvorschlagsliste entfallenen Stimmenzahlen werden fortlaufend durch die Folge der natürlichen Zahlen (1,2,3,4,...) dividiert. Dabei ergeben sich immer kleiner werdende Zahlen (logisch!) Man notiert nun, mit der *höchsten Zahl* beginnend (daher der Name) fortlaufend die *nächstniedrigeren* Zahlen und die jeweils dazu gehörende Partei (bzw. Person), und zwar solange, bis man soviele "Größtzahlen" beisammen hat, wie Sitze zu vergeben sind. Nun zählt man, wieviele solcher "Größtzahlen" jede Partei (Person) erhalten hat: soviele Sitze bekommt sie! Ganz einfach, nichtwahr, lieber Leser? Jedenfalls für die Wahlhelfer einfacher als umständliches Proportionalitätsrechnen...
Ein Leckerbissen für den Programmierfreak!

Hier nun mein Programm "d'HONDT" in BASIC. Du ahnst es, l.L.: Hier wird nach allen Regeln der Rechenkunst sortiert: Nämlich mit dem CMD"O"-Kommando! Und zwar absteigend. Das heißt, CMD"O" reiht die Quotienten (das sind die durch die natürlichen Zahlen dividierten Stimmen) nach abnehmender Größe. Deshalb das Minus vor der ersten Variablen (Zeile 120). Natürlich handelt es sich um eine *zweidimensionale* Variable, also eine Matrix. An anderer Stelle in diesem Heft wird die mathematische Seite des 2-dimensionalen Falles gesondert behandelt, da er etwas komplexer ist als der eindimensionale Fall und deshalb (leider) selten angewendet wird.
Durch einen Trick (die Z-Schleife in Zeile 70 und die I-Schleife in Zeilen 140-150) gelang es, die erhaltenen Höchstzahlen den einzelnen "Parteien" wieder zuzuordnen, so daß sie pro Partei ausgezählt werden können. Diese Auszählung wird schließlich in Zeile 170 ausgeworfen.
Vor Anwendung des Programmes vergesse man nicht, die Namen der Parteien (Personen) in die DATA-Zeile 180 einzutragen!

"Das wär's" (wie man sagt...)
Ach so: Was die Überschrift bedeute, fragst du, l.L.?
Na, dann *ORDNE* sie mal! <CMD"O",12,-X(I)>
Vergessen, was das "Minus" bedeutet?
Na, dann lies eben alles noch einmal!

-toJaK

Sitzverteilung nach d'HONDT

```
10 CLS
   PRINT "                d'Hondtsches Höchstzahlverfahren
     zur Ermittlung der Sitzverteilung im Parlament"
   PRINT
20 PRINT "              (C) KaJot Mühlenbein, Weinheim, 6/89
30 CLEAR 70
   PRINT STRING$ (61,"=")
40 DEFSTR P
   DEFINT A,I,M,N,S,Z              'S=Spalte  Z=Zeile  N="Spalten-Tiefe"
50 PRINT
   INPUT "Anzahl der Parteien (muß mit DATA-Zeile 180 übereinstimme
   n)    ";N
   N = N - 1
   INPUT "Anzahl der Sitze im Parlament ";ZZ
   M = (N + 1) * ZZ                'M=max. Matrix-Größe
   ZZ = ZZ - 1
60 DIM ZP(ZZ),V(N,ZZ),P(N),PP(N,ZZ)
70 FOR S = 0 TO N
   READ P(S)
   FOR Z = 0 TO ZZ
   PP(S,Z) = P(S)
   NEXT Z
80 PRINT "Stimmenzahl für "P(S);
   INPUT V(S,0)                    'V=Votum
   NEXT S
90 PRINT
   PRINT "                      * * * Ich rechne ! * * *
100 FOR Z = 1 TO ZZ
   FOR S = 0 TO N
110 V(S,Z) = V(S,0) / (Z + 1)
   NEXT S,Z
120 CMD "O",M, - V(0,0),PP(0,0)
130 FOR Z = 0 TO ZZ
   FOR S = 0 TO N
140 FOR I = 0 TO N
   IF PP(S,Z) = P(I),SS(I) = SS(I) + 1
150 NEXT I
160 A = A + 1
   IF A = ZZ + 1
   THEN GOTO 170
   ELSE NEXT S,Z
170 FOR I = 0 TO N
   PRINT "Die Partei der "P(I)" erhält" TAB( 40);SS(I)" Sitze"
   NEXT
180 DATARäuber,Stinker,Trinker,Heuler,Hacker  <- Beispiel für 5 "Parteien"
```

KaJot

* * * Ordnung muß sein! * * *

Soeben bin ich erfrischt und gestärkt dem Entmüdungsbecken entstiegen

> (der Gebrauch moderner Wortschöpfungen weist mich als modernen Menschen aus, der wenigstens in der Tagessprache up-to-date ist, wenn schon nicht in der Maschinensprache...)

und schon packt mich der Ordnungssinn: Die Seife hierhin, das Handtuch dorthin usw. Dies ist für mich also der erste Maßstab - oder, um langsam zur Sache zu kommen, die erste Dimension!
Doch schon erhalte ich gelinden Widerspruch: "Sie" meint, die Seife müsse dorthin und das Handtuch hierhin. Also ordnen wir beides um. Dies war - i h r e, die z w e i t e Dimension.
Die übrigen Sachen durfte ich lassen, wo ich sie hintat.
Nun haben wir also eine zweidimensionale Ordnung neben dem "Entmüdungsbecken", weil ihr und mein Wille geschah.
Könnte man das nicht gleich mittels Computer steuern, bevor Disput entsteht?
Man kann.
Das heißt: "Er" kann.
Sowohl ein- als auch zwei- und mehrdimensional. Und - vor allem - natürlich auch in BASIC!
All dies bewältigt das Kommando:

```
      a)    CMD"O",z,A(0,0,0,...),B(0,0,0,...),...
bzw.  b)    CMD"O",z,*I(0,0,0,...),A(0,0,0,...),B(0,0,0,...)...
```

Fangen wir "eindimensional" an.
Hierüber hatte ich mich bereits in einem INFO ausgelassen.
Daher hier in aller gebotenen "Kürze":
Gewöhnliche (feldfreie) Variablen können nicht mit diesem Kommando sortiert werden. Dafür könnte man ein Programm schreiben, das allerdings recht umständlich würde.
Elegant wird das Sortieren erst mit Feldern.
Im Feld A(J) ist J der "Index" oder die "Hausnummer" der Variablen A. Das Feld habe N+1 Elemente. Die Zählung beginnt mit J=0, der höchste Index ist also N, und diesen nennen wir die "Tiefe" der Felddimension. Beim Programmieren wird sie durch das statement "DIM A(N)" festgelegt. A ist, wie man sieht, "eindimensional".
Es können maximal 9 Feldvariablen gleichzeitig sortiert werden. Die Sortierung richtet sich nach dem ersten Feld; alle übrigen Felder machen diesen Neuordnung zwangsläufig mit, d.h. alle Felder mit gleicher "Haus-Nummer" bleiben stets beieinander!
Das 'z' im CMD"O"-Kommando bestimmt, wieviele Elemente des ersten Feldes sortiert werden; der Index, der für die Variable eingegeben wurde, bestimmt, ab welcher "Hausnummer" sortiert wird. Im allgemeinen ist dies die Null, also das erste Element. Es kann jedoch auch ab einem beliebigen anderen Element mit dem Index X sortiert werden. Sortiert wird in aufsteigender (ASCII-)Reihenfolge des ersten Feldes. Soll es absteigend gehen, also mit dem höchsten Wert der A(J) begonnen und abwärts sortiert werden, so wird ein Minuszeichen vor A(J) gesetzt.

Nun ein ganz wichtiger, interessanter Unterschied zwischen Form a) und b).
Benutzt man Form a), so werden die Felder sortiert, indem ihre Indizes (=Mehrzahl von Index; Betonung auf der ersten Silbe, das e wird lang gesprochen! <Also nicht mit den "Indizien" verwechseln...>) so verändert werden, daß sich bei der Auflistung mit einer J-Schleife die neue Reihenfolge ergibt (= "direkte Sortierung").
Benutzt man jedoch Form b), so bleiben die Indizes aller Variablen unverändert. Stattdessen werden die Werte (nicht der Index) des in dieser Form zusätzlich eingeführten, durch den Stern * gekennzeichneten "Indexfeldes" I(J) sortiert! Der Index J weist auf das zugehörige Element A(J) hin; der Wert von I(J) jedoch bestimmt dessen neue Position, letztlich also die Reihenfolge der sortierten Elemente A(J) (="indirekte Sortierung").
Will man diese auflisten, so muß die Schleife lauten:

```
FOR J = 0 TO Z-1
    <L>PRINT A(I(J))        } Z ≤ N+1
NEXT
```

Das Indexfeld I(J) muß übrigens ein Integerfeld sein.

Die "Hausnummern" (Indizes) der Elemente beginnen stets mit Null. Das erste Element ist also stets A(0). Man kann diese Hausnummern deshalb auch als "relative Element-Nummern" (R.E.N.) bezeichnen. Die "REN" (wir lassen die Punkte künftig weg) ist beim eindimensionalen Feld also identisch mit der Hausnummer (dem Index).
Nicht so bei *mehrdimensionalen* Feldern. (Ein Haus kann ja nicht mehrere Nummern haben!) Bei 2-dimensionalen Feldern z.B. wird die REN wie folgt berechnet.
Man denke sich die Elemente in einer Matrix angeordnet:

```
A(0,0) A(1,0) A(2,0) ... A(N,0)
A(0,1) A(1,1) A(2,1) ... A(N,1)
A(0,2) A(1,2) A(2,2) ... A(N,2)
  .      .      .     ...   .
  .      .      .     ...   .
A(0,M) A(1,M) A(2,M) ... A(N,M)
```

Der erste Index ist die Spalten-Nummer, der zweite die Zeilen-Nummer. Nach diesem Prinzip arbeitet CMD"O"!
N ist die Spalten-"Tiefe", M die Zeilen-"Tiefe". Die Anzahl aller Elemente ist mithin (N+1)*(M+1). - Alles klar?
Die REN ist hier nichts weiter wie die Durchzählung aller Matrixelemente von links oben nach rechts unten, und zwar zeilenweise, d.h. von links nach rechts! Also:

```
Den Indizes 0,0 entspricht REN=0,
 "      "    1,0      "      REN=1,
 "      "    2,0      "      REN=2,
                      ...    REN=...
 "      "    N,0      "      REN= N (="Tiefe"),
 "      "    0,1      "      REN=N+1,
 "      "    1,1      "      REN=N+2
 "      "    2,1      "      REN=N+3   u.s.w.
```

Allgemein gilt für eine 2-dimensionale Feldvariable mit der Dimensionierung DIM A(N,M), wobei die Spaltennummern S=0...N und die Zeilennummern Z=0...M lauten:

REN von Element A(S,Z) = Z * (N+1) + S (1)

wie sich an der Matrix leicht ablesen läßt.
Die Zeilentiefe (M) geht bei dieser Zählweise nicht ein. Sie würde die entsprechende Rolle spielen, wenn man die Elemente spaltenweise (statt zeilenweise) - also senkrecht von oben nach unten - abzählen würde. Das tut CMD"O" jedoch nicht!
Wenn wir mehrdimensionale Felder sortieren wollen, lautet das Kommando genauso wie bei eindimensionalen, nämlich einfach
(Form a:) CMD"O", z, A(J,K<,...>),..<ggf.weitere Feldvariablen>
mit z=Anzahl der zu sortierenden Elemente und J,K,...=Indizes desjenigen Elementes, mit dem die Sortierung beginnen soll; das ist i.a. A(0,0<,0,...>). Sortiert wird diese erste Variable; die übrigen Felder werden lediglich "mitgenommen". Es dürfen wieder insgesamt bis 9 Felder sein.
Der Algorithmus ändert die Indizes so, daß die Elemente in der sortierten Reihenfolge erscheinen, wenn man sie mit dem folgenden Schleifensystem auflistet:
(Bsp.für eine 3-dimensionale Variable A(J,K,L) mit DIM(N,M,P):

```
FOR L = 0 TO P
    FOR K = 0 TO M
        FOR J = 0 TO N
          <L>PRINT A(J,K,L)
        NEXT J
    NEXT K
NEXT L
```

Soviel zur *direkten* Sortierung von mehrdimensionalen Feldern.
Wie sieht aber bei diesen die indirekte Sortierung aus?
Hierbei spielen die oben erklärten "Relativen Element-Nummern" die entscheidende Rolle. Bei der direkten Sortierung werden diese von der Routine automatisch als neue Indizes eingesetzt. Bei der indirekten Sortierung werden die Indizes der Variablen selbst jedoch nicht verändert, sondern lediglich die Werte des Feldes der relativen Elementnummern - des REN-Feldes -, das den Variablen im CMD"O"-Kommando vorangestellt und durch einen * als "Indexfeld" gekennzeichnet wird:
CMD"O", z, *I(0,0,...), ±A(0,0,...), B(0,0,...), ...
Das ± vor der ersten Variablen A (nach der sortiert wird) steht zur Wahl für auf- oder absteigende Sortierung. Für die übrigen Felder gilt das bereits oben Gesagte.
Übrigens: Bei allen Sortierverfahren - ganz gleich, welche Dimension die Variablen haben - gelten sog. "Sortierlevel". Das bedeutet: Wenn mehrere Werte der ersten Variablen gleich sind, wird nach den *entsprechenden* Werten der nächsten Variablen (Level 2) solange weitersortiert, bis die erste Vartiable einen neuen Wert aufweist. Sind Werte der zweiten Variablen untereinander auch gleich, wird aufgrund der dritten Variablen sortiert (Sortierlevel 3) usw.

Das Auflisten indirekt sortierter mehrdimensionaler Variablen ist komplizierter als das der indirekt sortierten eindimensionalen. Das könnte der Grund dafür sein, daß indirekte Sortiervorgänge mehrdimensionaler Variablen so selten vorkommen, weil die Wiedergabe der Sortierung etwas umständlich ist. Es genügt ja nicht, wie bei den eindimensionalen Variablen, einen einzigen Index durch die neue REN zu ersetzen. Vielmehr müssen *alle* Indizes einer mehrdimensionalen Variablen irgendwie durch die neue REN, also durch das sortierte Indexfeld, ausgedrückt werden.
Aus obiger Gleichung (1) lassen sich für die REN" folgende Formeln für die neuen Indizes von 2-dim. Variablen entwickeln:

Spaltenindex (=1.Index):
S = I(S,Z) - INT(I(S,Z)/(N+1)) * (N+1)
Zeilenindex (=2.Index):
Z = INT(I(S,Z)/(N+1))

worin N die "Tiefe" des Spaltenindexes, also, wie schon gesagt, N+1 die Anzahl der Spalten ist.
Diese Formeln sind für S und Z im Ka,Jot-Schleifensystem (s.o.) einzusetzen. Die Variablen werden dann in der sortierten Reihenfolge angezeigt bzw. gedruckt. Dabei hat jedes Element seinen alten Index S für die Spalten- und Z für die Zeilen-Nummer behalten! Das kann sehr nützlich sein, wenn die Variablen noch in anderem Zusammenhang in der ursprünglichen Reihenfolge benötigt werden.

Zum Schluß einen wichtigen Hinweis auf eine Bedingung, "die nicht im Handbuch steht" (APPARAT hat hier geschlafen), und hinter die ich erst jetzt nach langer Fehlersuche bei der Anwendung zweidimensionaler Sortierung gekommen bin (und das war mit ein Grund für diesen Beitrag):

> Sämtliche Variablen - einschl. eines Indexfeldes - sind auch dann durch "DIM A(.....) usw." zu dimensionieren, wenn die "Tiefe" der Dimensionen kleiner als 11 ist!
>
> (Anmerkung: Im allgemeinen ist eine DIMensionierung bei N≤10 nicht erforderlich, weil NEWDOS von vornherein je Index 11 Plätze bereithält.)

Vor einem CMD"O"-Kommando muß jedoch auch dann DIMensioniert werden, wenn N≤10 ist, weil die Berechnung der REN sonst nicht funktioniert. Die Routine muß nämlich das Zeilenende erkennen können.
An anderer Stelle in diesem Heft findet sich ein Anwendungsbeispiel für zweidimensionale Sortierung (bei dem allerdings die direkte Sortierung genügte).

Die entsprechenden Rekursionsformeln für drei- und mehrdimensionale Sortierung sind wesentlich komplizierter. Ich habe deshalb beschlossen - nein, nicht "Politiker zu werden", sondern: sie nur auf Anfrage zu erläutern!
Denn es kennt seine Club'nheimer: der
(Sprich: Klapp'nheimer)

Klapp

[-NEXT L,K,J]

* P A S C A L + G A L T O N *

und der liebe Zufall

Nicht für Fakire ist das *Nagelbrett* von GALTON (geb.1822). Es sei denn, einer fühlt sich "als wie ein Fakir", der sich den Leiden der Mathematik unterzieht. Doch Galton war Biologe. Mit der dreieckigen Anordnung von Nägeln auf einem Brett, wie sie in der oberen Hälfte von Bild 1 zu sehen ist, wollte er zeigen, wie sich Erbeigenschaften auf die Nachkommen verteilen, wenn man die Erbbedingungen "dominant oder rezessiv" zugrundelegt, und zwar im Sinne von zwei Erbeigenschaften, die sich gegenseitig ausschließen; wobei das Verhältnis der Wahrscheinlichkeiten beider Eigenschaften durchaus nicht 1:1 sein muß (genauer gesagt, 0,5:0,5, denn die Summe muß stets 1 ergeben), sondern auch 0,7:0,3 oder 0,45:0,55 sein kann. Bild 1 bildet das Verhältnis 0,5:0,5 ab. Für andere Verhältnisse brauchen die Nagelreihen nur entsprechend gegeneinander verschoben zu werden. Doch wir wollen es hier nicht komplizieren.

Galton war ein so großer, genialer Biologe, daß er 1909 - zwei Jahre vor seinem Tode - vom englischen König in den Adelsstand erhoben wurde. Vielleicht war aber *Sir Francis Galton* - wie er seitdem anzureden ist - kein so großer Mathematiker. Sonst hätte er vielleicht auf seinen 200 Jahre älteren Vorläufer PASCAL (geb. 1623) hingewiesen, oder wenigstens auf den nur 45 Jahre älteren C.F.GAUSS, als er mit seinem dreieckigen Nagelbrett die natürliche Entwicklung der *Binomialkoeffizienten* mithilfe eines Zufallsexperimentes darstellte. * Oder hat er? Mein Mathelehrer hat das nicht verraten. (Ich finde, geistes- und naturgeschichtliche Zusammenhänge sollte man aufzeigen!)

Ihr habt das "Spielchen" vielleicht schon einmal irgendwo gesehen. Vor langer Zeit hing es noch als Spielautomat in Gaststuben (bevor Roulette- und Computerspiele Augen, Ohren und Nerven zerstörten...)
Man läßt eine kleine Kugel auf den obersten Nagel fallen. Dieser Ärmste muß sich nun "entscheiden", ob er nach links oder nach rechts fallen will (soll!) Beides ist im Falle des dargestellen "Galtonbrettes", bei welchem die Nägel genau mittig gegeneinander versetzt sind, gleich wahrscheinlich. Fällt die Kugel immer abwechselnd nach rechts und links, so landet sie schließlich genau in der Mitte des Bodens unter dem Brett. (Man stelle sich hier soviele Auffangbehälter vor, wie die letzte Nagelreihe Zwischenräume hat, aus denen die Kugeln fallen können.) Der "Zufall" will es jedoch, daß die Kugel manchmal zwei- oder dreimal nach der gleichen Seite fällt; dann wird sie irgendwo seitlich versetzt auf dem "Boden" landen.
Läßt man genügend Kugeln auf diese Weise durch die Nagelreihen wandern (wobei sie nur der Schwerkraft und dem "Zufall" unterworfen sind), so bilden sich am Boden Kugelhäufchen unterschiedlicher Höhe, die schließlich ein Stapeldiagramm ergeben, das sich um so besser der *GAUSS*ischen "Glockenkurve" nähern würde, je mehr Kugeln man fallen läßt - vorausgesetzt, man verbindet die Stapel durch eine stetige Kurve. Diese läßt sich durch eine etwas komplizierte Integralfunktion berechnen, auf deren Wiedergabe wir aber hier verzichten wollen, weil es nichts mit dem Programm zu tun hat.
Ich weiß wie gesagt nicht, ob *Galton* den *Gauß* kannte; als dieser starb, war *Galton* 33 Jahre alt; aber Engländer wollten ja oft nichts von den Deutschen wissen <was ich verstehen kann>. Da war ihm der Franzose *Pascal* schon lieber.

Dieser baute auch ein Dreieck auf; aber nicht aus Nägeln, sondern Zahlen. Nach dem Schema, das Bild 2 zeigt. Es ist für jeden leicht durchschaubar - oder? (Für den mathematisch Frustrierten: Man schreibe einen Kegelmantel(schnitt) aus lauter Einsen; dann schreibe man in die Zwischenräume immer die Summe der beiden darüberstehenden Zahlen - so einfach ist das!)
Die Zahlen, die man so erhält, sind ebenfalls die sog. "Binomialkoeffizienten", die sich ergeben, wenn man a+b beliebig potenziert. *Pascal* erfand dieses Dreieck als Trick für diejenigen, die nicht gerne mit Formeln umgehen. Denn man kann die Bin... - also die "B.K." auch aus "*n über k*"

$\binom{n}{k}$ (mit n=Zeilen-Nr., k=Spalten-Nr. im Dreieck, wobei Zeilen- und Spaltenzählung mit Null beginnen)

berechnen.
Doch *Galton* begnügte sich mit Nägeln (statt Zahlen) und mit Stapeln (statt stetigen Kurven) und erhielt so ebenfalls eine Darstellung der "B.K."! Zwar keine zahlenmäßige, aber eine anschauliche. Biologen denken eben praktisch.

Wenn ihr mein kleines Programm eingebt, könnt ihr die Kugeln laufen sehen. So etwa nach 100±20 Kugeln (sie werden links unten mitgezählt) bildet sich ein Stapeldiagramm aus, das sich nicht schlecht der Verteilung einer Fifty-fifty-Wahrscheinlichkeit nähert - also eine "Glockenkurve" andeutet. Natürlich cum grano salis - - - schließlich ist General *Zufall* im Spiel! Aber das "Gesetz der Großen Zahl" ist stärker als jeder Zufall - oft genug provoziert, muß er sich "dem Zwange edler Form" ergeben!

Übrigens:
Dies Spiel ist einmal nichts für Maschinensprache.
Die Kugeln würden schneller als die eines Maschinengewehrs durch die Nägel rasen und niemand würde was sehen! Denn aufs Sehen kommt es hier an. Es ist fast interessanter, dem Lauf der (hier eckigen) Kugeln zuzusehen, als dem Anwachsen der Stapel!
Es sei denn, der M.L.-Bauer baut komische Pausen zwischen jedem Fall einer "Kugel" ein.
(Wer war das? Bitte nicht lächeln, sondern einsenden!!)

KP

Darstellung einer Zufallsverteilung am GALTON'schen Nagelbrett

```
10 CLS:DEFINTI-K,X,Y
20 PRINT"              G A L -                    T O N";
30 FORY=2TO24STEP2
40 FORX=66-YTO62+YSTEP4
50 SET(X,Y):NEXTX,Y
60 FORX=35TO92:SET(X,47):NEXT
70 X=64:SET(X,0):FORK=1TO50:NEXT
80 FORY=0TO22STEP2:SET(X,Y):RESET(X,Y)
90 RANDOM:R=RND(0):S=SGN(.5-R)
100 SET(X+2*S,Y+1):RESET(X+2*S,Y+1)
110 SET(X+2*S,Y+2):RESET(X+2*S,Y+2)
120 X=X+2*S:NEXT
130 N=N+1:PRINT§972,N;
140 FORJ=25TO47
150 IFPOINT(X,J)THENSET(X,J-1):IFJ=25THEN170ELSEGOTO70
160 NEXT
170 GOTO170
```

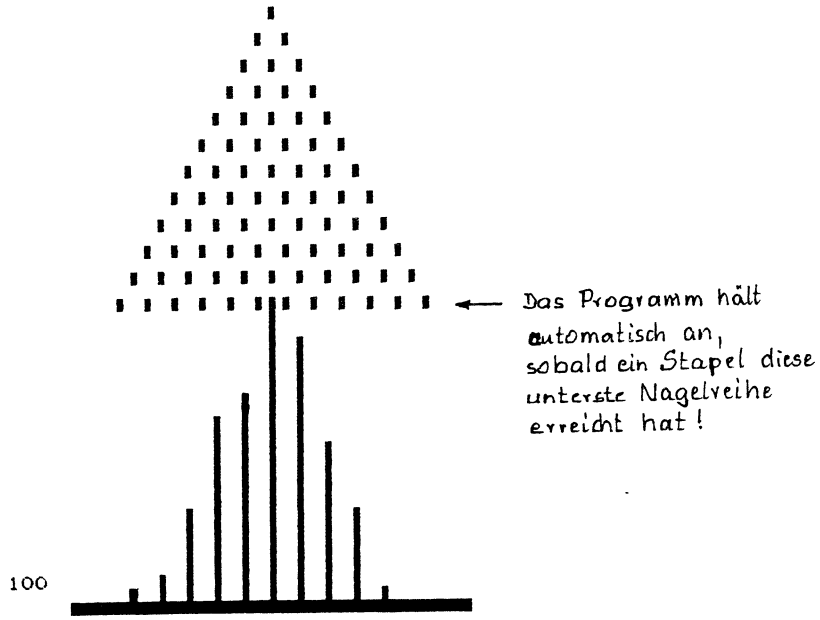
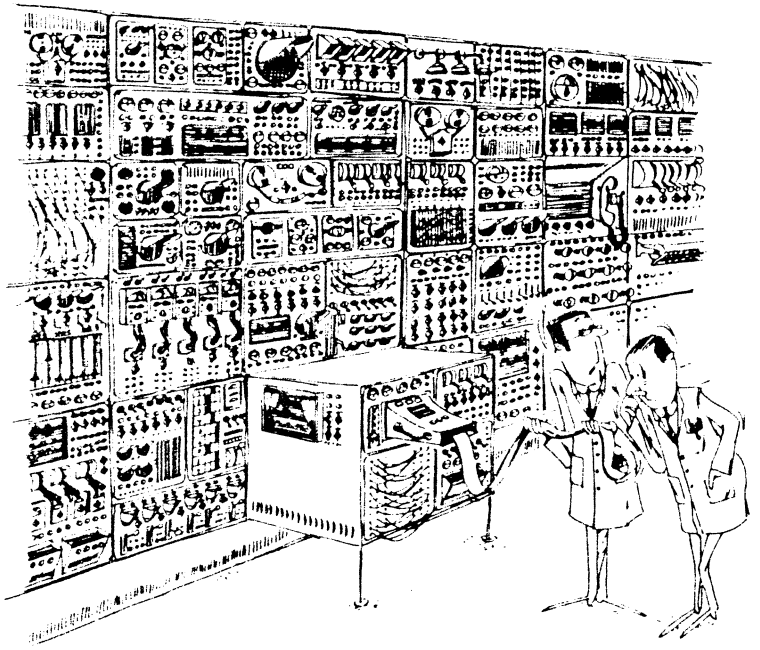


Bild 1

Das PASCAL'sche Dreieck:

```
                    1
                  1   1
                1   2   1
              1   3   3   1
            1   4   6   4   1
          1   5  10  10   5   1
        1   6  15  20  15   6   1
      1   7  21  35  35  21   7   1
    1   8  28  56  70  56  28   8   1
  1   9  36  84 126 126  84  36   9   1
1  10  45 120 210 252 210 120  45  10   1
1  11  55 165 330 462 462 330 165  55  11   1
1  12  66 220 495 792 924 792 495 220  66  12   1
.....................................................
      u.s.w.                          u.s.w.
```

Bild 2



»Ich werd' verrückt: Das Ding sagt: COGITO ERGO SUM!*«
* ich denke, also bin ich

Oskar Drechsler

Als Neuer möchte ich auch etwas zur Info beisteuern, weiß aber nicht, unter welchem Oberbegriff ich das anbieten soll. Es ist nämlich viel Zeugs, das größtenteils zusammengehört bzw. zusammenarbeitet. Vielleicht fangt Ihr einfach mal von vorne an:

Startprogramm für Basic-Programme

Wer sich Anfang der achtziger Jahre einen Computer gekauft hat, schreibt seine Programme selbst. Da sammelt sich im Laufe der Jahre einiges an. Wenn man Programme wegen höherer Geschwindigkeit über den Accel3 laufen läßt und auch noch eine HPG-Platine besitzt, muß man ständig darauf achten, daß das richtige Hilfsprogramm geladen ist. Außerdem kommt es vor, daß man ein Programm laden will und hat den genauen File-Namen vergessen. Irgendwann ging mir das gewaltig auf den Wecker und so entstand ein Programm, das das Löschen und Laden der Hilfsprogramme erledigt und alle ständig benötigten Programme in alphabetischer Reihenfolge mit einer Nummer auf dem Bildschirm zeigt. Man muß nur noch die Nummer tippen und schon rattert das Laufwerk. Wofür hat man einen Computer?

Das Programm heißt Start/Bas. Dort werden unter DATA die Programm-Namen eingetragen, das Laufwerk, wo sie zu finden sind und die Programmart-Kennung:
A = Accel3-Programm, benötigt Puntime-Poutinen
H = HPG-Programm, benötigt HPG/CMD
B = Basic-Programm, wegen der Größe kein Hilfsprogramm zulässig
X = Basic-Programm, kann mit geladenem Accel oder HPG laufen
C = Maschinen-Programm
I = für Accel3-Compiler und PACKER

Alle Hilfsprogramme werden automatisch geladen. HPG wird automatisch gelöscht. Für Löschen von Accel3 wird man aufgefordert ' /restore' und 'cont' einzugeben.

Wenn man im Menü des aufgerufenen Programms als Programmende RUN"Start/Bas" programmiert, muß man nur noch Knöppsche drücke.

Das Ablegen der Programm-Namen unter DATA erspart einen unnötigen Disketten-Zugriff und einen zusätzlichen File. Da die Programm-Namen selten verändert werden, gibt man sie in alphabetischer Reihenfolge ein und braucht nicht bei jedem Programmlauf den Diskettenzugriff für CMD"D".

Das Start-Programm alleine ist aber nicht viel wert, wenn der andere Kram unpraktisch ist. Deshalb jetzt ein anderes Thema.

Dateiprogramm

Eigentlich wollte ich das Programm später vorstellen, aber es arbeitet in der vorliegenden Form nur mit den im nächsten Abschnitt beschriebenen Programmen.

Obwohl ich universelle Dateiprogramme nicht leiden kann, möchte ich das nachfolgend beschriebene Programm nicht mehr missen. Es ist schnell zu erlernen, leicht zu bedienen und enthält keine Funktionen die sowieso niemand braucht. Es enthält allerdings keine Datenausgabe-Routine um möglichst viel Platz für Daten zu schaffen.

Das oben ausgesprochene Lob gilt nicht mir, sondern einem Bekannten, der inzwischen zu IBM-AT geflüchtet ist und gegen eine Aufnahme seines Programmes in unsere Programm-Bibliothek nichts einzuwenden hat. Das Programm lief recht ordentlich, hatte aber zwei kleine Fehler. Bei der Fehlersuche hatte ich noch einige Ideen und habe das Programm umgeschrieben und ergänzt. 30-40 % sind von mir.

Mit dem Maskengenerator (Generat/Bas) wird die Bildschirmmaske erstellt. Dabei kann innerhalb des vorgegebenen freien Raumes eine Maske vollkommen nach eigenen Wünschen gebastelt werden. Die Datensatzlänge kann von 1-255 gehen. Es können 1-25 Datenfelder vorkommen. Soll ein Feld linksbündig gefüllt werden, ist das Datenfeld mit '*' zu markieren, für rechtsbündig mit Doppelkreuzen. (Ich schreibe mit Genie-Text, dort sind Doppelkreuz und Dollarzeichen Steuerzeichen und somit nicht darstellbar). Wenn man anschließend die ENTER-Taste drückt, wird die Maske interpretiert und zusammen mit den notwendigen Parametern als xxxxx/MSK abgespeichert.

Datei/Bas nimmt die Maske und die Parameter als Futter und arbeitet damit. Die letzten zwei Zeilen des Bildschirmes geben Bedienerhinweise, sodaß zur Benutzung nicht viel zu sagen ist.

3 Funktionen sind näher zu erklären:
Bei Eingabe des Dollarzeichens an der 1. Stelle eines Datenfeldes wird der Inhalt des Datenfeldes des vorher eingegebenen Datensatzes in dieses Feld kopiert. Das ist recht nützlich, wenn in einem Datenfeld öfter gleiche oder ähnliche Eingaben vorkommen, die dann nur leicht geändert werden brauchen.

Bei der Suche eines Datensatzes kann in jedem Feld gesucht werden. Das Suchwort muß allerdings am linken Rand des Feldes beginnen und kann ein Teilwort sein, das aber am Wortanfang beginnen muß (keine Instring-Suche). Das Suchwort ist mit einem Doppelkreuz abzuschließen. Das Suchwort wird gespeichert. Wenn das Suchwort wieder verwendet werden soll, ist anstatt dessen nur ein Dollarzeichen einzugeben.

Die Daten werden bei der Eingabe nach dem 1. Datenfeld, innerhalb diesem nach dem 2. Datenfeld ... usw. sortiert. Abgespeichert wird sequentiell mit dem FF-File. Alle Zahlen liegen wegen der Speicherung im RAM (nicht als Variable) als String vor und werden auch so abgespeichert.

Bei der Auswahl der Funktionen gab es wie üblich Probleme mit den Anfangsbuchstaben von Speichern und Suchen. Deshalb erfolgt die Auswahl in englisch. Siehe Hardcopy's am Ende.

Das Programm befindet sich einmal als Datei/Ori in Originalform mit Kommentaren auf Diskette, in der es nicht lauffähig ist und einmal als Datei/Bas, gekürzt mit dem PACKER.

Für das Auslesen der Daten gebe ich mein Programm Elohefte/Bas als Muster dazu, ebenfalls die Daten. Das erleichtert den Einstieg.

1. Bild

```
Eingabemasken - Generator

Eingabefelder: * = Kennung für Feld linksbündig
               # = Kennung für Feld rechtsbündig
Vor dem ersten und hinter dem letzten '*' '#' muß mindestens
ein ' ' stehen.

               Die maximale Feldzahl beträgt 25.

Satzlänge      : 1 ... 255 Zeichen

Steuerzeichen  : <--/-->  / Auf-/Abpfeil = Cursorbewegung
                 Sh <--/-->               = Chr löschen/einfügen
                 <New Line>               = Interpretation
    Zum Fortfahren bitte 'Taste' drücken .......
```

Beispiele für Erstellung einer Maske

```
    Add        Update        Delete        Inquire        Save
|Dateiname:              |  |Satznummer:  |  |Satzzahl:    |

Titel      : ********************************
Fundstelle: ****************
Kode       : *

El     = Elektor
PE     = Populäre Elektronik
ELV
elrad
Buchstabe = Fundstelle im Ordner
```

```
    Add        Update        Delete        Inquire        Save
|Dateiname:              |  |Satznummer:  |  |Satzzahl:    |

Titel      : ********************************   <- linksbündig
Fundstelle: ################ <- rechtsbündig
Kode       : *

El     = Elektor
PE     = Populäre Elektronik
ELV
elrad
Buchstabe = Fundstelle im Ordner
```

Datei Bas

Befehlsmodus erscheint nach Programmstart und Laden der Daten.


```
    Add        Update        Delete        Inquire        Save
|Dateiname: elohefte/SEQ|  |Satznummer:  |  |Satzzahl: 422|

Titel      :
Fundstelle:
Kode       :

El     = Elektor
PE     = Populäre Elektronik
ELV
elrad
Buchstabe = Fundstelle im Ordner

 <A>dd     <U>pdate     <D>elete     <I>nquire     <S>ave     <Q>uit
```

wurde gewählt

```
    Add        Update        Delete        Inquire        Save
|Dateiname: elohefte/SEQ|  |Satznummer:  |  |Satzzahl: 422|

Titel      :
Fundstelle:
Kode       :

El     = Elektor
PE     = Populäre Elektronik
ELV
elrad
Buchstabe = Fundstelle im Ordner

 Shift <--/-->  Chr löschen/einfügen      <CLEAR> Feld löschen
 Bitte Datensatz eingeben... oder   <§> für Befehlsmodus
```

wurde gewählt

```
    Add        Update        Delete        Inquire        Save
|Dateiname: elohefte/SEQ|  |Satznummer:  |  |Satzzahl: 422|

Titel      :
Fundstelle:
Kode       :

El     = Elektor
PE     = Populäre Elektronik
ELV
elrad
Buchstabe = Fundstelle im Ordner

 Suchbegriff mit <#> abschließen  oder  <§> für Befehlsmodus
```

Zum Schluß zur Speicherbelegung

Ich hatte Routinen aus Basic f&b oben im Speicher liegen und wenn ich nach Benutzung von HRG/CMD vergaß, sie neu zu laden, stürzte der Rechner beim Sprung auf eine der nicht mehr vorhandenen Routinen ab. So faßte ich den Keyboard-Treiber und diese Routinen im File MaschUnt/USR zusammen und legte sie unter Basic-Anfang, der auf 7051H angehoben wird. Somit steht er unbehelligt da unten rum und stört nur, wenn man Basic mit mehr als 3 Filebereichen lädt. Aber wann kommt das schon vor!

Die Diskette, die ich Oliver schicken werde, enthält folgendes:

NEWDOS 2.052 und Basic mit deutschen Fehlerkommentaren, formatiert mit 40 Tr SS/DD.

Mit Autostart wird das Programm 'Anfang/Bas' geladen, das den Basic-Anfang anhebt und das Programm 'Start/Bas' rennen läßt. Dieses Programm lädt 'MaschUnt/USR' nach, startet den Keyboard-Treiber und zeigt alle Programme, die abrufbar sind.

Der Chainfile 'Allstart/JCL' macht beim Aufruf von HRG/CMD und den Runtime-Routinen von Accel3 'AccelRun/CMD' den Automatismus und ruft den 'Accel3/CMD'-Compiler und den PACKER auf.

Die in HRG/CMD enthaltene LPRINT-Routine ist für Epson-kompatible Drucker und beinhaltet ESC K n1 n2. Läuft mindestens auf Epson FX 85, CPA 80 und Nec P 5. Ich habe die Druckerinitialisierung vor und nach LPRINT herausgenommen, um mehrere Hardcopys nahtlos aneinanderfügen zu können.

Wer schon Accel3-Programme hat, muß sie durch Aufruf des Accel3 über das Start-Programm neu compilieren, weil der Basic-Anfang beim Compilieren genauso stehen muß wie beim Programmlauf.

Außerdem stehen folgende Routinen aus Basic f&b zur Verfügung: ID-Array, Quick-Array, LStrip, Search2, Sort3, Movex, RStrip. RStrip befindet sich allerdings auf dem File Basic/CMD zusammen mit CLS für die HRG 1B. Da habe ich noch etwas freien Platz gefunden. Einsprungadressen siehe nachfolgende Speicherbelegung.

Wer noch Fragen hat, kann mich täglich von 1900-2400 Uhr anrufen. Wenn ich ins Bett gehe, stelle ich das Telefon ab. Ihr könnt mich also nie stören. ☎ 02202/55282

Viele Grüße

Olivier

Speicherbelegung

DB00-FF17	56064-65303	HRG/CMD
E9E0-EFBF	59872-65471	Accel3/CMD
E9E0-EFE2	59872-61410	AccelRun/CMD
EF79-FF79		Packer/CMD

7001	28673	frei für Basic-Programme
7000	28672	enthält '0', muß so bleiben
6F26-6FFF	28454-28671	frei
6F1B-6F25	28443-28453	KbdNeu ausschalten
6D88-6F1A	28040-28442	KBDNEU
6D87	28039	Zeichensatz HRG geladen
6D86	28038	Programm-Nr. in Start/Bas
6D85	28037	Programm auf Laufwerk
6D84	28036	geladenes Hilfsprogramm
6D83	28035	MaschUnt/USR geladen
6D82	28034	Kontrollzahl

6CD8-6D81	27864-28033	Search2	USR8
6C62-6CD7	27746-27863	ID-Array	USR5
6BDD-6C61	27613-27745	Search1	USR4
6B57-6BDC	27479-27612	Quick-Array	USR3
6B38-6B56	27448-27478	LStrip	USR1
6AE0-6B37	27360-27447	Movex nur:	USR7
6A47-6ADF	27207-27359	Sort3	USR2

Die Maschinenprogramme befinden sich auf dem File MaschUnt/USR bzw. MaschUnt/SRC

Folgende Routinen befinden sich auf BASIC/CMD:
siehe auch BasicMod/SRC

669E-66BB	RStrip
6698-669D	Chr$(27) auf Drucker
668B-6697	Zeichen in Akku auf Drucker
6682-668A	Chr$(27);Chr$(33) auf Drucker
6669-6681	#CLS

Basic-Anfang wird durch Anfang/Bas höher gesetzt
Es können nur 3 Files gleichzeitig geöffnet sein

Bei Änderung Basic-Anfang, Accel-Progr. neu kompilieren!

Beim künftigen Start aus dem DOS ist immer wie folgt zu verfahren:
Basic,run'ANFANG/BAS'
Die letzte Zeile von Anfang/Bas muß ein RUN auf ein anderes Basic-Programm enthalten, das als erstes enthalten muß:
CMD'load Maschunt/USR':DEFUSR &H6D88:X=USR(X)

Geheimnisse eines Zeichengenerators

von Norbert Giese

September 1989

In diesem Artikel möchte ich Euch meine Erfahrungen und Erkenntnisse mitteilen, die ich bei der Änderung zweier Charaktergenerator-EPROMs gemacht bzw. gewonnen habe. Dieses auf den ersten Blick einfache Unterfangen entpuppte sich zu einem umfangreichen Projekt, gab aber auch Einblicke in eine Methode, einen vollständigen Charaktersatz in möglichst wenig Speicherplatz unterzubringen.

Konkret beziehen sich die folgenden Angaben auf einen Sharp MZ-80B Computer mit Z80 CPU und dem dazugehörenden Drucker MZ-80P5 mit 8085 CPU (ab Produktionsdatum Oktober 1981). Beide Geräte haben in ihren Zeichensätzen u. a. einen Unterstrich (ASCII 95, bzw. hexadezimal '5F'), der aber aus mir unerklärlichen Gründen als "Oberstrich" auf dem Bildschirm und auf dem Papier erschien. Die üblichen Pascal-Listings mit ihren Variablennamen und den verbindenden Unterstrichen sahen folglich etwas seltsam aus, auch sonst war mit dem "Oberstrich" nicht viel anzufangen. Dieser Tatbestand sollte geändert werden.

Aus beiden Geräten wurden die entsprechenden EPROMs ausgebaut, auf dem EPROMMER eines Freundes ausgelesen und als Hex-Dump aufs Papier gebracht. Das EPROM 2716 (2 Kilobyte) aus dem MZ-80B enthält den Zeichensatz in Form einer Matrix aus je acht Zeilen und acht Spalten pro Zeichen. Der folgende Auszug aus einem Hex-Dump, schon etwas aufbereitet, zeigt dies deutlich.

```
0000    00 00 00 00 00 00 00 00     ;ASCII 00H
0008    1C 14 14 77 22 14 08 00     ;ASCII 01H
.
0100    00 00 00 00 00 00 00 00     ;ASCII 20H
0108    08 08 08 08 00 00 08 00     ;ASCII 21H
0110    24 24 24 00 00 00 00 00     ;ASCII 22H
.
0200    1C 22 4A 56 4C 20 1E 00     ;ASCII 40H
0208    18 24 42 7E 42 42 42 00     ;ASCII 41H
0210    7C 22 22 3C 22 22 7C 00     ;ASCII 42H
.
02F8    FF 00 00 00 00 00 00 00     ;ASCII 5FH
.
06F8    00 FF FF FF FF FF FF FF     ;ASCII DFH
.
07F8    00 00 01 3E 54 14 14 00     ;ASCII FFH
```

Jeweils acht Bytes formen ein Zeichen. Wenn man für jedes dieser acht Bytes deren Bitfolge zu Papier bringt, repräsentiert jedes gesetzte Bit einen Punkt auf dem Bildschirm. Die folgenden Zeichen "A", "B", und der "Oberstrich" zeigen dies.

```
0208 18  ...XX...      0210 7C  .XXXXX..      02F8 FF  XXXXXXXX
0209 24  ..X..X..      0211 22  ..X...X.      02F9 00  ........
020A 42  .X....X.      0212 22  ..X...X.      02FA 00  ........
020B 7E  .XXXXXX.      0213 3C  ..XXXX..      02FB 00  ........
020C 42  .X....X.      0214 22  ..X...X.      02FC 00  ........
020D 42  .X....X.      0215 22  ..X...X.      02FD 00  ........
020E 42  .X....X.      0216 7C  .XXXXX..      02FE 00  ........
020F 00  ........      0217 00  ........      02FF 00  ........
```

Der "Oberstrich" beginnt im Hex-Dump auf der Adresse 02F8H als Folge von acht Bytes, nämlich "0FFH, 000H, 000H, 000H, 000H, 000H, 000H, 000H". Ändert man dies in die Bytefolge "000H, 000H, 000H, 000H, 000H, 000H, 000H, 0FFH", dann wird aus dem "Oberstrich" der gewünschte Unterstrich. Da der Zeichensatz des MZ-80B alle druckbaren Zeichen auch noch in invertierter Form, sprich schwarze Punkte auf weissem Hintergrund, enthielt, wurde auch hier der invertierte "Oberstrich" geändert (Adresse 06F8).

Bevor ich auf des EPROM im Drucker komme, noch eine kurze Rechnung zum Platzbedarf. Der Zeichensatz im MZ-80B spezifiziert die ASCII-Kodes 00 bis 255, also 256 Zeichen. Acht Bytes pro Zeichen mal 256 ergibt 2048 Bytes, also genau 2 Kilobyte. Im Hex-Dump entspricht dies den Adressen 0000H bis 0FFFH. Das EPROM ist also vollständig belegt.

Nun galt es noch das EPROM aus dem Drucker zu ändern. Beide Zeichensätze sollten ja identisch sein. Da es ein 2732 EPROM (4 Kilobyte) ist, vermutete ich je 2 Kilobyte für den Zeichensatz und für das Betriebsprogramm des Druckers. Jedoch lies sich nur gelegentlich ein vollständigen Zeichens entdecken, oft nur die Andeutung davon. Das Service Manual, etwas genauer gelesen, sprach von einem 'verdichteten Charaktersatz', sonst schwieg es beharrlich.

Wie aber sollte ich diesen entschlüsseln? Einen Disassembler für den 8085 Prozessor besitze ich nicht. Dessen Mnemonics wollte ich auch nicht unbedingt lernen.

Irgendwann erinnerte ich mich, dass der 8085 Prozessor 100% softwarekompatibel zum 8080 ist, dieser wiederum eine Untergruppe des Z80 ist. Folglich sollte doch mit einem Z80-Disassembler etwas zu machen sein. Gedacht, getan. Der Inhalt des EPROMs 2732 wurde disassembliert und das Listing Stück für Stück kommentiert. Meine Achtung vor den Designern stieg mit jedem Byte. Nie zuvor hatte ich daran gedacht, so weit in das Innenleben eines Druckers vorzudringen. Es dauerte einige lange Nächte, bis ich eine Stelle im Programmlisting gefunden hatte, die offensichtlich ein vom Drucker empfangenes Byte in eine Kette von Bytes für den Druckkopf aufbereitete. Da der Druckkopf nur acht Nadeln hat, lag es nahe, dass jeweils ein Byte die abzufeuernden Nadeln spezifiziert, bevor der Schrittmotor des Druckkopftransportes einen winzigen Schritt in Schreibrichtung macht. Auf diese Weise werden Schritt für Schritt aus einzelnen Bytes Zeichen und letztendlich eine ganze Textzeile. Die Zeichenhöhe ist somit acht Punkte, was sich auch unter einer Lupe auf dem Papier zeigte.

Wie aber wird aus einem einzelnen, vom Drucker empfangenen Byte eine Folge von Bytes für den Druckkopf? Am einfachsten wäre eine Tabelle, in der für jeden empfangenen ASCII-Wert z.B. acht Bytes für den Druckkopf stehen. Diese Tabelle könnte ähnlich wie beim MZ-80B aufgebaut sein und wäre genauso gross. Dies war aber offensichtlich nicht der Fall. Erstens hatte das Betriebsprogramm im EPROM 2732 schon mehr als die Hälfte der 4 Kilobyte belegt, zweitens hätte ich dann ja die einzelnen Zeichen finden müssen, und drittens hiess es ja 'verdichteter' Charaktersatz.

Die Ingenieure von Sharp haben sich damals eine, wie ich meine, elegante Methode einfallen lassen, um dieses Platzproblem zu lösen, vermutlich auch aus Kostengründen. Ein ausgeklügelter Algorithmus und zwei Tabellen sind der Kern der Lösung. Der Algorithmus empfängt einen zu druckenden ASCII-Wert und bereitet zuerst einmal einen internen Puffer von 14 Bytes vor, den er mit 00 füllt. Dies bedeutet, dass später 14 Bytes für jeden empfangenen ASCII-Wert an den Druckkopf weitergegeben werden. Die Darstellung in diesem Puffer bewirkt, das später für jedes gesetzte Bit eines Bytes eine Nadel des Druckkopfes abgefeuert wird. Zwischen den einzelnen Bytes wird natürlich ein Vorschub des Druckkopfes in Schreibrichtung ausgeführt.

Nun gilt es, diese 14 Bytes im Puffer entsprechend des empfangenen ASCII-Werts zu setzen. Dazu bedient sich der Algorithmus der zwei Tabellen, die hier verkürzt und der Übersicht halber nebeneinander gezeigt sind.

```
DEFB 00000001B          DEFB 000H               ;ASCII 20H
DEFB 00110010B          DEFB 000H, 05FH         ;ASCII 21H
DEFB 11100010B          DEFB 007H,0000H         ;ASCII 22H
DEFB 11000011B          DEFB 014H,07FH,014H     ;ASCII 23H
DEFB 00001001B          DEFB 004H,02AH,000H,02AH ;ASCII 24H
                        DEFB 055H,02AH,000H,02AH
                        DEFB 010H
DEFB 00001001B          DEFB 003H,040H,023H,010H ;ASCII 25H
                        DEFB 008H,004H,062H,001H
                        DEFB 060H
.
DEFB 00001001B          DEFB 032H,000H,049H,000H ;ASCII 40H
                        DEFB 079H,000H,041H,022H
                        DEFB 01CH
DEFB 01000101B          DEFB 078H,004H,012H,001H ;ASCII 41H
                        DEFB 010H
.
DEFB 11000011B          DEFB 001H,001H,001H     ;ASCII 5FH
DEFB 10100011B          DEFB 001H,002H,004H     ;ASCII 60H
.
```

Die Tabelle auf der linken Seite enthält für jedes druckbare Zeichen ein Byte, die Tabelle auf der rechten Seite für das korrespondierende Zeichen eine variable Anzahl von Bytes. Jedes Byte der linken Tabelle hat folgende Bedeutung:

Bit	7	Flag für Leerspalten einfügen
Bit	6	Flag für Spiegelung
Bit	5,4	Offset in Puffer
Bit	3,2,1,0	Anzahl Bytes aus rechter Tabelle

Die Bits 3,2,1 und 0 bestimmen die Anzahl der Bytes, welche für den empfangenen ASCII-Wert aus der rechten Tabelle in den Puffer zu stellen sind. Die Bits 5 und 4 geben an, ab welcher Stelle im Puffer diese Bytes zu speichern sind. Ist Bit 6 gesetzt, werden die Bytes aus der rechten Tabelle noch einmal gespiegelt in den Puffer geschoben (Erklärung folgt). Ist Bit 7 gesetzt, wird zwischen jede zu druckende Spalte eine Leerspalte eingefügt.

Dieses ist zugegeben nicht einfach und bedarf eines Beispiels. Der Drucker empfängt z.B. den ASCII-Wert 73 ='I', bzw. 49H. Irgendwann entscheidet sich das Betriebsprogramm des Druckers, dieses Zeichen zu drucken und übergibt es an den besagten Algorithmus. Der interne 14-Byte Puffer wird vorbereitet:

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Inhalt	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Die für Hex 49 gültigen Einträge in den beiden Tabellen sehen wie folgt aus:

```
DEFB    11100010B       DEFB    041H,07FH       ;ASCII 49

   Bit  76543210
```

Die Bits 5 und 4 in der linken Tabelle, hier 0010B, bestimmen den Offset in den Puffer. Zwei Positionen (1 und 2) bleiben also unverändert.

Position	1	2
Inhalt	00	00

Die Bits 3, 2, 1, 0 in der linken Tabelle, hier 10B, bestimmen, wieviele Bytes aus der rechten Tabelle in den Puffer geschoben werden sollen. Also zwei Bytes. Zusätzlich ist aber Bit 7 aktiv, es soll also eine Leerspalte nach jedem Byte eingefügt werden. Der Puffer sieht nun so aus:

Position	1	2	3	4	5	6
Inhalt	00	00	41	00	7F	00	00	00	00	00	00	00	00	00

Nun ist noch das Bit 6 in der linken Tabelle gesetzt. Es weist den Algorithmus an, die Bytes aus der rechten Tabelle noch einmal gespiegelt in den Puffer zu schieben, dabei aber das letzte Byte zu ignorieren. Also wird 07FH ignoriert und 041H in den Puffer geschoben, wieder gefolgt von einer Leerspalte (Bit 7 aktiv). Der Puffer sieht nun so aus:

Position	1	2	3	4	5	6	7	8
Inhalt	00	00	41	00	7F	00	41	00	00	00	00	00	00	00

Schickt man diese Folge von Bytes gedanklich zum Druckkopf und schiebt diesen nach jedem Byte einen Schritt nach rechts, ergibt sich folgendes Bild für das Zeichen 'I':

```
Nadel 1  . . X . X . X .
Nadel 2  . . . . X . . .
Nadel 3  . . . . X . . .
Nadel 4  . . . . X . . .
Nadel 5  . . . . X . . .
Nadel 6  . . . . X . . .
Nadel 7  . . X . X . X .
Nadel 8  . . . . . . . .

         00 00 41 00 7F 00 41 00 00 00 00 .....
```

Es ist also gelungen, mit einem Byte aus der linken Tabelle und zwei Bytes aus der rechten Tabelle ein 'I' vollständig darzustellen. Spielt man dieses Beispiel mit anderen Zeichen durch, stellt man sehr leicht fest, dass viele Zeichen unseres Alphabets sich vertikal spiegeln lassen (A, H, I, M, O, T, U, ...). Man braucht folglich nur einen Teil dieser Zeichen in der Tabelle angeben, ohne dass Information verlorengeht. Auch ist der Abstand zum benachbarten Zeichen durch den Offset und die Länge des Puffers eindeutig bestimmt.

Rechnet man dieses Verfahren der Platzoptimierung durch, ergibt sich folgende Bilanz: Der Algorithmus und die Tabellen zur Spezifizierung von 256 Zeichen belegen zusammen etwa 1700 Bytes, dazu im Vergleich das Bildschirm-EPROM im MZ-80B, das genau 2 Kilobyte belegt. Hier muss man aber berücksichtigen, das dieser die einzelnen Zeichen nur in einer Breite von acht Punkten auflöst, dagegen der Drucker eine horizontale Auflösung von 14 Bytes hat. Also eine erhebliche Platzeinsparung.

Einige Leser mögen sich nun fragen, warum man über Bit 7 noch die Einfügung von Leerspalten steuert. Hier hätte man doch offensichtlich die horizontale Auflösung besser machen können. Hier muss ich nun etwas spekulieren, da ich die Sharp-Ingenieure nicht gefragt habe. Würde man die offensichtlich vorhandene horizontale Auflösung dadurch ausnutzen, dass man die Zeichen unter Weglassung der Leerspalten in der linken Tabelle genauer spezifiziert, müsste man dies konsequenterweise für alle Zeichen mit gesetztem Bit 7 machen. Dadurch würde die Tabelle grösser (Platzproblem). So aber kann man die Auflösung selektiv für die Zeichen nutzen, welche schräge Linien haben, wie zum Beispiel das Zeichen 'A'. Hier ist das Bit 7 zurückgesetzt. Ein weiterer Grund wird durch die Mechanik des Druckkopfes und die Druckgeschwindigkeit gegeben sein. Würde man zum Beispiel das Zeichen 'I' unter Weglassung der Leerspalten spezifizieren, dann würden die Nadeln 1 und 7 über die gleiche Wegstrecke fünf anstatt drei mal feuern müssen. Dies verlangt, soll es mit gleicher Geschwindigkeit passieren, eine schnellere Mechanik im Druckkopf und definitiv ein stärkeres

Netzteil im Drucker. Wenn man z.B. das Zeichen 'A' genau betrachtet, hat auch hier jede Nadel eine 'Pause', bevor sie wieder gefeuert wird.

Nach diesen Erkenntnissen war die Lösung des "Oberstrich"-Problems eine Kleinigkeit. Das Byte in der linken Tabelle blieb unverändert, in der rechten Tabelle wurde aus der Bytefolge "001H, 001H, 001H" die Folge "080H, 080H, 080H", und für den invertierten "Oberstrich" wurde aus "0FEH, 0FEH, 0FEH, 0FEH, 0FEH, 0FEH" die Folge "07FH, 07FH, 07FH, 07FH, 07FH, 07FH". Es tat mit fast leid, den Sharp-Ingenieuren ihr Geheimnis entrissen zu haben, als der erste Unterstrich auf dem Bildschirm und dem Drucker erschienen war.

Den Lesern, denen in diesem Text merkwürdige Umlaute und falsche "ss" aufgefallen sind: deutsche Umlaute habe ich noch nicht eingebaut, aber jetzt weiss ich ja, wie so etwas gemacht wird.

Teuer, aber überflüssig

Eigentlich können Sie es auch gleich sein lassen, vor allem dann, wenn Sie versuchen, ein modernes System zu erwerben. Denn beim Computer gilt eine eiserne Faustregel: Spätestens alle drei Monate wird eine neue Maschine auf den Markt geworfen, die nicht nur alles bisher Dagewesene technisch in den Schatten stellt, sondern auch nur noch einen Bruchteil des Vormodells kostet. Außerdem werden die Dinger auch immer kleiner. Computer, die noch vor zwanzig Jahren mehrere Stockwerke eines Rechenzentrums füllten, haben heute bequem in einer Würstchenbude Platz.
Irgend jemand hat mal folgenden Vergleich angestellt: Wenn man diese Entwicklung auf die Automobilindustrie überträgt, dann wäre ein Rolls-Royce, der vor 20 Jahren 80000 Mark kostete, heute schon für fünf Mark achtzig zu haben. Dafür hätte er allerdings auch in einer Streichholzschachtel Platz.
Da stellt sich die Frage: Wer will schon seinen Rolls in der Hosentasche herumtragen? Und ganz nebenbei: Wie soll man da noch den Knopf fürs Autoradio erreichen?
Die Computer könnte man schon längst noch viel kleiner bauen, wenn nur der Mensch nicht so sperrig wäre. Aber unsere Fingerchen sind eben dafür entwickelt worden, Keulen und Kanonen zu bedienen und nicht mit Mikrochips zu hantieren. Merken Sie sich also die Regel Nummer eins beim Kauf der neuen Wundermaschinen:

- **Fragen Sie sich, wofür Sie die Kiste eigentlich brauchen können**

Wenn Sie nicht zufällig der Innenminister sind oder Chef einer großen Versicherungsanstalt, dann überlegen Sie sich die Sache noch mal. Wie haben Sie denn die letzten zehn Jahre Ihre Probleme gelöst? Mal ehrlich: Um Ihren Kontostand zu ermitteln, genügt doch schon eine Kinderrechenmaschine mit Holzperlen, und den Biervorrat im Eisschrank überschauen Sie am besten, indem Sie einfach mal die Türe öffnen und nachzählen: 1, 2, 3!
Leider leben wir in einer Zeit, in der man der Werbung ausgeliefert ist. Jedes Produkt wird mit einem passenden Image versehen: Die harten Kerle rauchen die ungesunden Zigaretten, die Softies ziehen die teuren italienischen Klamotten an. Wer sich für sportlich hält, kauft nur einen weißen Wagen mit schwarzen Gummiflossen und Streifen am Heck. Eine Hausfrau, die nicht mit »Rakete« aufwischt und mit »Blitzy« spült, kann nicht einmal mehr wagen, ihre Freundinnen zum Kaffee einzuladen.

Im technischen Bereich scheint die Sache ebenfalls klar zu sein: Wer heute dazugehören möchte, wer dynamisch, modern, aufgeschlossen, tolerant, aber trotzdem noch ein bißchen individualistisch ist, braucht einen Computer – sagt die Werbung.
So ein Kasten mit dem kleinen grünen Flimmerschirm ist angeblich für alle da. Denn so ein Ding ist einfach »wahnsinnig praktisch«. Das kann alles und kostet dabei nur halb soviel wie ein Auto. Ein Computer gehört einfach zum Image des modernen Menschen.
Das Genialste an der Erfindung des Computers ist, daß die Medien uns erfolgreich eingeredet haben: Jeder, der sich der Kiste verweigert, ist ein weltfremder Typ von vorgestern!
Nie hat uns jemand aufgefordert, einen Fahrkartenautomaten, Zigarettenautomaten oder eine Telefonzelle in unserer Wohnung aufzustellen. Auch eine Druckmaschine oder eine zusammenklappbare Autowerkstatt wären Dinge, mit denen fast jeder etwas anfangen könnte. Aber das ist von den Marketing-Leuten verpennt worden.
Seit der Vermarktung des Automobils hat niemand in der Werbebranche mehr daran geglaubt, daß es jemals wieder etwas geben könnte, das nicht nur unnütz und gefährlich, sondern auch noch immens teuer ist und das man trotzdem jedem halbwegs vernünftigen Mann zwischen sechs und sechzig mühelos andrehen kann. Jetzt ist es endlich gefunden: das teure Ding für jedermann!
Nachdem Sie nun schon einmal festgestellt haben, daß das Leben bisher auch ohne Computer ganz gut funktionierte, fragen Sie sich:

- **Wann habe ich Zeit für den Kasten?**

Schließlich arbeiten Sie ja von neun bis um halb fünf. Wenn Sie nach Hause kommen, will erst mal Ihre Frau von ihnen wissen, wie der Tag gewesen ist, und dann möchte Ihre kleine Tochter aus dem Buch über die Schweine vorgelesen bekommen.
Außerdem ist es höchste Zeit, daß Sie endlich mal wieder ihren Hund kämmen. Und kaum ist das Abendbrot fertig, lockt Frank Elstner mit der neuesten Ausgabe von »Wetten, daß ...?« zu einem gemütlichen Nickerchen vor der Glotze.
Also, wo ist denn da Zeit, um am Computer zu arbeiten?
Von Arbeiten kann sowieso keine Rede sein. Die Untersuchungen zeigen: Mit der programmierbaren Schreibmaschine wird mehr gespielt als gearbeitet. Das ist schon wieder so ein genialer Streich der PR-Strategen. Aufs Knöpfchen gedrückt wurde schon immer gerne: Ob es die Flipper-Automaten in der Kneipe um die Ecke oder die Druckknöpfe an den Jeans der ersten Freundin auf dem Rücksitz des VW-Käfers waren. Mit dem Computer sind die Spielchen erwachsen geworden. Plötzlich ist die Jagd auf die Ufos kein Kinderkram mehr, sondern wir bereiten uns vor »auf die beruflichen Aufgaben des nächsten Jahrzehnts«.
Gemessen an den meistverkauften Programmen im Bereich der Home-Computer wollen demnach die meisten Menschen in Zukunft Weltraumpiloten werden.

Ordnung muß sein!

Bis vor kurzem kokettierte ich noch gerne mit der nicht ganz wahrheitsgemäßen Bemerkung, ich besäße nur eine Diskette. Es war nämlich so, daß die paar Programme und Datensätze, die ich häufig brauche, sämtlich einträchtig auf meiner Workdisk versammelt waren. Nun hat mir aber Andreas Magnus eine Harddisk eingebaut, die sich wie 5 Laufwerke verhält. Klar, daß ich bei dieser bequemen Art des Zugriffs unterschiedliche Software-Pakete je einem Drive zugeordnet habe, um leichter die Übersicht behalten zu können. Früher hätte ich solche Ordnung nur mit dem leidigen Türchen-auf-Türchen-zu an den Floppy-Laufwerken halten können.

So weit die guten Nachrichten. Hier eine schlechte (eine zweite gute: die schlechte stimmt seit ein paar Minuten nicht mehr): Die SYSTEM-Optionen AN=xx und AO=xx legen fest, welches Laufwerk ohne weitere Parameter beim Befehl DIR und bei der Eröffnung neuer Dateien benutzt werden soll. Andreas hat das DOS so gepatcht, daß ein Teil-Laufwerk der Harddisk hierfür benutzt wird. Gut und schön. Meine Textverarbeitung mit allen Formularen liegt aber auf einem anderen. Neue Texte sollen auch dorthin. Es macht wenig Laune, jedesmal die Laufwerksnummer angeben zu müssen, um die liebe neue Ordnung auch durchzuhalten.

Übrigens tritt dieses Problem auch ohne Harddisk auf, falls ihr die Disziplin habt, grundsätzlich in Lw. 0 nur das System und in einem anderen Laufwerk z. B. den Assembler-Kram zu beharken.

Die Abhilfe ist so simpel, daß ich mich geniere, nicht schon seit Jahren damit zu arbeiten: Das zuletzt angesprochene Laufwerk ist in 4308h vermerkt. Wenn ich gerade ZEUS angekurbelt habe, der bei mir in Drive 6 (Harddisk) wohnt, steht dort also eine 6. Da sollen bitteschön auch alle Sources hingeschrieben werden, die ich gerade schöpfe. Also braucht nur der Inhalt von 4308h nach 43A0h (SYSTEM,AN=xx) und 43A1h (SYSTEM,AO=xx) kopiert zu werden, um das DOS auf das zuletzt benutzte Laufwerk einzuschwören.

Hacker sind faul, deshalb sollte das gerne automatisch passieren, sobald ZEUS (oder z. B. auch TSCRIPS in Lw. 7) aufgerufen wird. Dazu braucht nur eine kleine Maschinenroutine (s. Assemblerlisting) an das Hauptprogramm angehängt zu werden, die das beim Einprung mal eben erledigt:

Die Laufwerksnummer in 4308h wird geladen. Hier kommt das Register L zur Anwendung; wenn anschließend H aus L geladen wird, können gleichzeitig 16 Bit (zweimal die Laufwerksnummer) nach 43A0h und 43A1h kopiert werden. Auf diese Weise werden nur 10 Bytes für das Prográmmchen verschlissen. Nach dieser Verlade muß natürlich das Gastgeberprogramm besprungen werden. Seine Einsprungsadresse wurde zuvor aus dem Ende-Code in seinem letzten Disksektor ermittelt. Die kommentierten Sektordumps sollen diese Zusammenhänge aufklären.

Die alte Frage: Wohin mit solchen Erweiterungen? Beim einfachen Aufruf eines Programms wird vom DOS-Eingabepuffer nur ein bißchen verbraucht, je nach Länge der File Specifications. Der riesige Rest ist frei. Dort kann die Routine arbeiten (s. ORG).

Die fortgeschrittenen DOS-Frevler unter den Lesern werden sich nun fragen, wieso nicht mit lausigen 2 Bytes, nämlich

```
ORG     43A0h       ;Lw# für DIR und CREATE
DEFW    lw#lw#      ;zweimal die Lw#
```

derselbe Job, sogar ohne Rechenzeit erledigt werden soll. Deswegen: Die Diskette liegt vielleicht auch mal nicht im Lieblingslaufwerk. Und wenn das Programm von der RAM-Disk kommt, sieht es wieder anders aus. Daher war die etwas umständlichere Methode vorzuziehen.

Um die Erweiterung an ein Host-Programm anzuhängen, genügt leider nicht der APPEND-Befehl. Man kann es mit der CDD-Funktion von SUPERZAP tun oder "zu Fuß", indem man die Codes einzeln eintippt. Wenn dabei die Sektorgrenze überschritten wird, wird es leider haarig. Mann sollte dann mit APPEND eine kurze Datei, die höchstens einen Sektor lang ist, an das Hauptprogramm anhängen. Wenn der neue Sektor diskrelativ direkt an das Hauptprogramm anschließt, wird die ganze Schlußarie auf einmal mit CDD übertragen, sonst in zwei Raten bis bzw. ab Sektorgrenze. Eine direkte Manipulation am Inhaltsverzeichnis mag schneller sein, ist aber kritischer, besonders dann, wenn der neue Sektor in einem neuen Granule liegt.

Computer heißt auf spanisch *ordenador*, weil er ordnet. Meiner jetzt endlich auch.

Arnulf

```
424E            00001 program EQU     424eh         ;Startadresse Host-Programm, hier ZEUS
                00002
435E            00003         ORG     435eh         ;am Ende des DOS-Eingabepuffers
435E  2A0843    00004 start   LD      HL,(4308h)    ;L (- aktuelle Laufwerknummer
4361  65        00005         LD      H,L           ;HL (- dasselbe zweimal
4362  22A043    00006         LD      (43a0h),HL    ;jetzt Default-Lw. fÜr DIR und CREATE
4365  C34E42    00007         JP      program       ;Host-Programm anspringen
435E            00008         END     start         ;dort Einsprung

00000 Fehler
```

ZEUS/CMD (letzter Sektor) vor der Änderung:
(Einsprungsadresse unterstrichen)

```
FRS C0  E521 0042 1174 0601 1900 DBFA F5E6 DFD3
39  D0  FAED B01E D20E 35ED B0F1 D3FA E13E 13CD
27H E0  3300 C312 5302 024E 4200 0000 0000 0000
    F0  0000 0000 0000 0000 0000 0000 0000 0000
```

Die Wurmfortsatz-Routine:
(die Einsprungsadresse von ZEUS/CMD und ihre eigene unterstrichen)

```
DRU 00  010C 5E43 2A08 4365 22A0 43C3 4E42 0202
2   10  5E43 0000 0000 0000 0000 0000 0000 0000
2H  20  0000 0000 0000 0000 0000 0000 0000 0000
```

ZEUS/CMD (letzter Sektor) nach der Änderung:
(Wurmfortsatz-Routine komplett unterstrichen –
Achtung! Der Ende-Code von ZEUS/CMD muß vom Record-Header des Anängsels überschrieben werden!)

```
FRS C0  E521 0042 1174 0601 1900 DBFA F5E6 DFD3
39  D0  FAED B01E D20E 35ED B0F1 D3FA E13E 13CD
27H E0  3300 C312 5301 0C5E 432A 0843 6522 A043
    F0  C34E 4202 025E 4300 0000 0000 0000 0000
```

WORDSTAR AUF ALLEN LAUFWERKEN

Ist es Euch auch schon so gegangen? Diskette mit Wordstar ins Laufwerk, gestartet und freundliche Mitteilung, die Overlays sind nicht zu finden, obwohl sie da sind ?!

AHA, wieder mal falsches Laufwerk genommen. Diskette wechseln! Alles noch mal von vorn...

Das muß aber keinesfalls so sein. MicroPro hat bei WS lediglich ein paar Bytes "vergessen" einzubauen. Nun soll es ja Betriebssysteme geben, die diese Unterlassungssünde ausbügeln, mein CP/M aber nicht. Der FATALE ERROR liegt in der Adresse 02DCH von WS.COM begründet. Hier steht die Variable DEFDSK, die festlegt, auf welchem Laufwerk WS seine Overlays suchen soll. Steht dort nun "01" (entspricht Drive A) und die Diskette mit WS und seinen Overlays steckt in Drive B, dann sind die eben diese Overlays für WS nicht existent.

Abhilfe läßt sich durch eine Mini-Routine bei Start von WS.COM erreichen. Wie alte CP/M-er wissen, steht der Aktuelle Laufwerks-Code in den unteren 4 Bit der Adr. 0004. Der Inhalt dieser Adresse wird nur bei Aufruf der BIOS-Routine SELDSK beeinflußt. Nach dem Laden von WS.COM wird also dort noch der Laufwerks-Code des Laufwerkes stehen, in dem die Dikette mit WS gesteckt hat. Dieser Code braucht nur in DEFDSK eingetragen werden und schon "weiß" WS.COM auf welchem Laufwerk seine Overlays stehen.

Die einfachste Art diese Maßnahme zu realisieren ist das Umleiten des Startsprunges auf Adr. 0100H zu einer entsprechenden Routine und danach der Sprung zur Original-Startadresse. Die Routine kann in einem freien Bereich von MORPAT, dem Bereich in WS.COM für derartige User-Routinen, (Adr. 02E0-035BH) abgelegt werden.

```
0100    JP      NEWST       ;Start-Adr. ersetzen

NEWST:  LD      A,(0004)    ;Laufwerks-Code holen, A=00, B=01,...
        AND     0F,A        ;nur Bit 0-3
        INC     A           ;Laufwerk A=01, B=02,...
        LD      (02DC),A    ;Laufwerks-Code nach DEFDSK
        JP       OLDST      ;Sprung zu der Adresse, die im Sprung
                            ;auf 0100H im Original stand. Bei Ver-
                            ;sion 2 und 3 meist 2D08H
```

Mit DU oder ZSID ist diese Angelegenheit in 10 Minuten erledigt. Einen Schönheitsfehler hat der Vorschlag allerdings. WS muß vom aktuellen Laufwerk gestartet werden. Wenn etwa folgendes gemacht wird:

A>B:WS

wird WS in Adr. 0004 den Wert 00 (Drive A) vorfinden und eintragen. Unter der Annahme, daß WS und seine Overlays auf Drive B stecken, führt diese Methode mit Sicherheit ins Abseits.

Ursache dafür ist die Arbeitsweise des CCP, der nur für die Dauer des Ladevorganges der WS.COM auf Drive A umschaltet. Vor der Übergabe der Kontrolle an das Programm wird der alte Zustand wieder hergestellt und WS findet daher Drive B als aktuelles Laufwerk vor!

Funktionieren würde es so:

A>B:

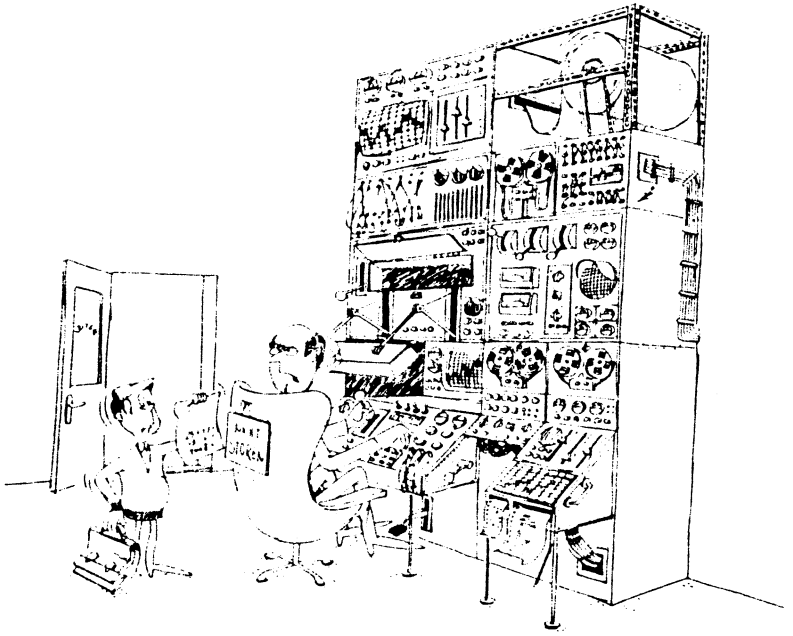
B>WS

Selbst dieser Haken ist, wenn auch mit etwas mehr Aufwand, vermeidbar. Geheimtip: BDOS-Fkt. 17! Wie das geht werde ich im nächsten Info vorstellen.

Noch ein kleiner Tip: Bei WS ist oftmals nicht zwischen harten Trennstrichen und weichen Trennstrichen, die durch Zeilenumbruch erzeugt werden, zu unterscheiden. Nach mehrmaliger Neuformatierung stehen die Trennstriche dann recht sinnverwirrend auf dem Schirm.

Der Zeichencode für den weichen Trennstrich ist in der Variablen SOFHYC (Adr. 03B8H) festgelegt. Der Code hat keinen Einfluß auf den Dateiinhalt oder das Druckbild, d.h. man kann ohne die Dateikompatibilität zu gefährden ändern. Zweckmäßig ist der Ersatz von "-" (2DH) durch "_" (5FH). Damit ist eine eindeutige Zuordnung möglich.

Frank-Michael Schober



»Ist mir egal, ob dein Opa hier der Chef ist – mach' deine Schulaufgaben allein!«

Teamwork

Durch einen Brief von Helmut Bernhardt bin ich auf folgendes Problem gestoßen: man hat eine große Datei mit Namen und Adressen (z.B. unsere Mitgliederverwaltung) unter DBASE II und möchte an eine bestimme Gruppe, z.B. die Nordlichter, ein Rundschreiben schicken, wozu man natürlich (?) Wordstar und Mailmerge nimmt. Wie bringt man die beiden Programme nun zur Zusammenarbeit, damit man nicht alles per Hand abtippen muß ? In einer alten c't (8/84) stand, wie's geht. Ausgehend davon habe ich ein kleines Beispiel gebaut, an dem man hoffentlich sieht, was dabei alles beachtet werden muß. Zuerst sollte man sich die Länge der einzelnen Felder im DBASE ausgeben lassen, sonst gibt's bei dem Vergleich weiter unten Probleme.

```
.. Drucken von Adressenaufklebern mit den Daten aus der Mitgliederverwaltung
.. unter WordStar 4.0
..
..      Erzeugen des Files in DBASE mit:
..
..      .USE pdaten                         <- hier stehen die Adressen
..      .DISPLAY STRUCTURE                  <- Länge der Felder anzeigen
..      .COPY TO xxx DELIMITED WITH ,       <- Komma nicht vergessen !
..
..      Variable          Länge
..
..      NAME          C     020
..      VORNAME       C     020
..      STRASSE       C     025
..      LAND          C     003
..      PLZ           C     005
..      WOHNORT       C     025
..      TELP          C     020
..      TELG          C     020
..      GEBOREN       C     010
..      EINTRITT      C     010
..      BEITRAG       L     001
..      NUMMER        N     004
..
.. Feld im IF muß exakt die Länge haben, die oben angegeben ist !
..
.DF mixxx.txt                 <- diese Datei wird eingelesen
.RV
name,vorname,strasse,land,plz,wohnort,telp,telg,geboren,eintritt,beitrag,
    nummer                    <- alle Elemente einlesen
.IF &plz& #<4000              <- z.B. alle Nordlichter. Hinter der '4000' ist
Herrn                         <- ein Leerzeichen, da das Feld 5 Zeichen enthält
&vorname& &name&
&strasse&
&plz& &wohnort&

.EI                           <- Ende des IF
```

Einzelne Aufkleber kann man natürlich einfacher direkt aus DBASE drucken, aber wenn dann noch ein kleiner Brief dazu kommt, braucht man doch ein Textprogramm. Ebenso kann man auch Ausgaben des DBASE-Listengenerators in den Texteditor laden und noch nach Wunsch nachbearbeiten, z.B. um die Texte einzudeutschen oder Kommentare anzufügen, indem man die Ausgabe mit 'SET ALTERNATE TO listout' und 'SET ALTERNATE ON' zusätzlich auf die Floppy lenkt.

Alexander Schmid

* * * TSCRIPS retten * * *

Kurt Müller ergänzte in Heft 26, S.18, meine früheren Ausführungen, wie man einen verunglückten TSCRIPS-Text rettet, durch den Hinweis auf den (NEW)DOS-Befehl DUMP.
Hiermit läßt sich das von mir sz. geschilderte Verfahren wesentlich vereinfachen - vorausgesetzt, daß der Text im Speicher noch nicht zu sehr verunstaltet oder gar ganz kaputt ist.
Leider beschreibt Clubkollege *Kurt* das Vorgehen nicht im Detail, insbesondere nicht, wie man den korrigierten Text nun wieder in das Textsystem TSCRIPS integriert. Ich habe einige Zeit herumprobiert, bis ich es heraushatte. In der Vermutung, daß auch andere mit dem DUMP-Befehl nicht so vertraut sind, möchte ich das Vorgehen hier Schritt für Schritt kurz aufzeigen. Sicher kann man es außer für TSCRIPS auch zum "Retten" anderer Programme anwenden.
Es funktioniert so:

1) Gehe ins DOS
2) Lade SUPERZAP
3) Wähle DM (Display Memory). Der TSCRIPS-Text beginnt in Speicherstelle 9177H (37239d).
4) Korrigiere die Fehler mit MOD (es braucht nur das letzte Byte der Speicheradresse eingegeben zu werden)

Falls gleich weitergeschrieben wird ==> gehe nach 5)
falls der korrigierte Text gespeichert werden soll ==> 5a)

5) Verlasse SUPERZAP mit EXIT
6) Lade TSCRIPS und wähle A<LT>. Drücke SHIFT/Hochpfeil, dann CLEAR

Der korrigierte Text sollte jetzt auf dem Bildschirm sein!

---

* E N D E *

5a) Notiere die Adresse des letzten zu speichernden Bytes nach der korrigierten Stelle = Endadresse.
6a) Verlasse SUPERZAP
7a) Gib ein:

DUMP FILENAME/CMD:dr,37239,Endadresse

8a) Sobald der korrigierte Text mit TSCRIPS weiterverarbeitet werden soll, gib im DOS ein: LOAD FILENAME/CMD:dr
9a) Gehe nach 6)

Kurt, schönen Dank für Deinen Hinweis!

(Man wird alt wie'n Haus
und lernt nicht aus.
Grüß Gott! Jot-Klaus)

Vergleich BASIC mit TURBO-PASCAL

Der Vergleich "VW gegen PORSCHE" ist überhaupt *kein* Vergleich dagegen. Eher schon der Vergleich "Käfer gegen Windhund"! Wer sich noch nicht mit TURBO-PASCAL befaßt hat, sollte sich einmal für diesen ersten kurzen und bescheidenen "Erfahrungsbericht" interessieren. (Wer TP kennt und auch benutzt, sollte zum nächsten Beitrag weitergehen: Er findet hier nichts Neues!)
Als steinalter BASIC-Freak - und auch heute noch der Überzeugung, daß man mit BASIC *alle* Probleme lösen kann und nur dann eine andere Sprache braucht, wenn man es eiliger hat (oder auf irgendeiner engen Schiene dahinfährt) - habe ich mich nun endlich mit *TURBO-PASCAL* befaßt! Das war mit meinem (ur)-alten TRS-80 Model 1 allerdings nur möglich, seit ein geduldiger und aufopferungsbereiter Clubkamerad mir eine CP/M-Karte verpaßt und die nötige Software dafür spendiert hat (oder muß es "CP/M-Banker" heißen? Ich glaube nicht, denn mein CP/M+ ist nicht "gebankt". Aber dieses Wissen ist - mir jedenfalls - völlig unwichtig. Als kleiner "User" bitte ich daher den Fachmann ggf. um Verzeihung!)

Um künftig ein TP-User zu werden, habe ich zunächst einige dünne sowie dicke Bücher über PASCAL studiert; für Interessierte seien sie hier aufgeführt:

A.Schmuck: PASCAL - Schritt für Schritt DM 8.80
(Humboldt-Taschenbuch Nr.551)
R.Baumann: Programmieren mit PASCAL (antiqu.DM 13.-)
Vogel-Verlag (sog."CHIP-Wissen")
H.Feller : PASCAL-Datenstrukturen u. Anwendungen
Luther-Verlag; auf einer Messe gekauft; Preis?
(Dieser Verlag scheint ja inzwischen pleite zu sein.)

Speziell für *TURBO-PASCAL* empfehle ich jedoch:
K.-H.Rollke: Grundkurs Turbo-Pascal, Band 1 & 2, je DM 29.80
Das zweibändige Werk ist bis Version 4 zu benutzen. (Inzwischen gibt es schon Version 5; aber das ist für den Einsteiger sicher wurs<ch>t).
Dieses im SYBEX-Verlag erschienene Werk scheint mir am klarsten geschrieben und ist so recht geeignet für geistige Schnecken wie mich. Schwerwiegende Druckfehler halten sich in Grenzen (ich fand bisher nur drei), so daß man das Buch (289 + 320 Seiten) auch nach längerem Lesen noch ertragen kann. SYBEX scheint mir ohnehin der beste EDV-Verlag zu sein.
Ein guter Clubkamerad kopierte mir außerdem 375 Seiten eines Werkes (das hier nicht genannt werden soll), auf dessen Lektüre an Winterabenden ich mich schon besonders freue, weil es recht wissenschaftlich anmutet...

Zu Beginn der Lektüre erschien mir als schlichtem BASIC-Jünger alles furchtbar umständlich. Du meine Güte: Jede noch so kleine Handlung ("Prozedur") bekommt ihre eigene Überschrift mit einem eigenen Namen; dauernd muß etwas erklärt ("definiert") werden; kleine Selbstverständlichkeiten, die man unter NEWDOS sich einfach aus dem ROM holt, müssen ausführlich Punkt für Punkt beschrieben (programmiert) werden, um dann als selbständige "Funktion" zu fungieren (unter NEWDOS meist schlicht mit der Funktion 'DEFFN' darstellbar) und so weiter und auch nicht weiter...

Der "enorme" Vorzug der zwangsläufigen Strukturiertheit dieser förmlich starr fixierten Sprache kann nur demjenigen imponieren, der sich bisher bei seinen Programmierbemühungen stets in einem Urwald von GOTO-Befehlen, Schleifen und sonstigen Anweisungen verlaufen hat und gar nicht gemerkt hat, daß er daran selber schuld war, weil sein Geist so undiszipliniert war. In der Tat: "Strukturiert programmieren" kann man in BASIC auch! Ich habe das in einem früheren Beitrag gezeigt.
Natürlich habe ich das eine und andere TP-Programm von *Rollke* abgeschrieben und ausprobiert - kritisch. Nun aber wollte ich es genau wissen, wo der Vorteil liegt und warum ich mein Hirn mit neuen Vokabeln quälen sollte: Was kann BASIC eigentlich nicht im Vergleich mit Turbo-Pascal?
Ich nahm mir also eins von den ein bißchen mathematischen Programmen des Herrn Rollke vor (er wird mir verzeihen, hoffe ich), welches sich auf S.186 des ersten Bandes befindet. Nachdem man zwei ganze Zahlen eingegeben hat, sagt dieses Programm, welche von beiden "die kleinere" ist (was ja für einen Erstkläßler wirklich enorm schwer zu übersehen ist! Gottlob gibt's dafür Computer...). Es verkündet aber auch, welches der "ggT" (der größte gemeinsame Teiler) dieser beiden Zahlen ist (dies erfordert i.a. schon einen höheren Grad mathematischer Bildung') und schließlich offenbart es dem Zuschauer, ob die eine Zahl eine Primzahl ist oder nicht, und verrät den gleichen Sachverhalt danach auch hinsichtlich der zweiten eingegebenen Zahl!
Ich hätte mir zwar gewünscht, in jenen Fällen, in denen die eingegebene Zahl *keine* Primzahl ist, auch noch zu erfahren, wodurch sie denn nun teilbar ist - aber das sollte dem Adepten sicher noch nicht zugemutet werden: vielleicht hätte es das Programm gesprengt...
Nun programmierte ich das gleiche Thema in BASIC - nichts leichter als das! Hierbei habe ich mich bewußt nicht um besondere "Strukturierung" bemüht, sondern etliche "GOTOs" benutzt!
Nachdem ich mein BASIC-Programm getestet (man mußte ja einbauen, daß die "1" nicht versehentlich als Primzahl und die "2" nicht als Nichtprimzahl bewertet wird sowie daß die "1" und die eingegebene Zahl selbst sog. "unechte" Teiler sind und daher außer Konkurrenz liegen, d.h. aus dem Rennen auszuscheiden sind; solcherlei Kleinkram kriegt man natürlich erst nach einigen Tests hin) -
aber nun konnte es losgehen: Stoppuhr zur Hand!
Was heißt hier Stoppuhr?
Es genügte der Sekundenzeiger meiner Armbanduhr, die "ich, wo ich gehe, stets bei mir trage" (*Löwe*) - und nachher sogar mußte der Minutenzeiger herbei - je nachdem, was für ein Zahlenpärchen ich eingab. Folgende Liste zeigt die Geschwindigkeits-, genauer gesagt: die Zeitunterschiede zwischen beiden Sprachen:

Gewählte Zahlen		Zeitbedarf in Sek.	
		BASIC	Turbo-Pascal
117	226	3	< 0.5
551	323	8	< 0.5
1001	557	12	0.5
11107	6543	145 (!)	5

Das letzte Beispiel zeigt den Zeitgewinn unter TP überdeutlich; er ist das 29fache! Diese lockeren Beispiele ließen sich bis zur größten darstellbaren Ganzen Zahl (65535) erweitern; doch hatte ich keine Lust mehr, auf mein schläfriges BASIC-Programm zu *warten*. Offenbar steigt der Zeitgewinn logarithmisch an; ich will das jetzt nicht genauer untersuchen.
(Sicher ist er aber nicht so groß, wie er bei Umsetzung des Programmes in Maschinensprache wäre - gell, *Arnulf*?)

Beide Programme findet ihr zum Vergleich anschließend.
Nunmehr werde ich mich aufgrund der Anhebung der Geschwindigkeitsbegrenzung innerhalb meines TRS-80-Dorfes mit eitler Lust ins TP-Vergnügen stürzen - denn "Zeit ist Geld" (ZIG). Und billiger kann ich nicht zu Geld kommen.
Wer's noch nicht tut: Wer möchte nicht mit mir Zeit (Geld) sparen? * Der melde sich zum Erfahrungsaustausch! Eins kann ich dem Neuling - falls es ihn noch gibt - versprechen: Er kann aus seinem VW-Käfer in einen TURBO-PORSCHE umsteigen! Fast kostenlos...
Dennoch werde ich auch in Zukunft auf mein Allerwelts-BASIC nicht ganz verzichten; denn:

Wer die Qual
einer Wahl
scheut, der nehme allemal
einfach TURBO-BAS(I)CAL!

Grüß Gott!
Ka-Jot

P.S. Die wesentliche Ursache für die Beschleunigung ist der Umstand, daß Turbo-Pascal eine Compiler-Sprache ist. [BASIC kann aber oft auch kompiliert werden!]


```
program primzahlGgt;

var zahl1, zahl2 : integer;
    wort : string(.5.);

function min (a,b:integer):integer;
 begin
  if a<b then min:=a
         else min:=b
 end;

function ggT (a,b:integer):integer;
 var teiler:integer;
  begin
   for teiler:=1 to min(a,b) do
     if a mod teiler = 0 then
     if b mod teiler = 0 then ggT:=teiler;
  end;

function prim(zahl:integer):boolean;
 var teiler:integer;
     test:boolean;

 begin
  teiler:=3;
  test:=true;
    if zahl<2then test:=false else
     if zahl>3then
      if zahl mod 2=0 then test:=false else
       repeat
        if zahl mod teiler=0then test:=false;
        teiler:=teiler+2;
       until teiler>round(sqrt(zahl));
      prim:=test
     end;

function weiter:boolean;
 var ch:char;

 begin
   write('weitermachen (J/N) ?');
   read(ch);
   weiter:=(ch='J')or(ch='j');clrscr
 end;

begin
clrscr;
 repeat
  write('Gib eine erste Zahl ein: ');
  readln(zahl1);
  write('Gib noch eine Zahl ein: ');
  readln(zahl2);
  writeln('Der groesste gemeinsame Teiler');
  writeln('ist ggT = ',ggT(Zahl1,Zahl2):3);
  if prim(zahl1) then wort:='eine' else wort:='keine';
  writeln(zahl1:3,' ist ',wort,' Primzahl');
  if prim(zahl2) then wort:='eine' else wort:='keine';
  writeln(zahl2:3,' ist ',wort,' Primzahl')
 until not weiter
end.
```

```
10 CLS
   PRINT "            PRIMZAHL oder nicht ?"
   DEFINT A - N
   PRINT
   PRINT "Bitte zwei ganze Zahlen <A, B> eingeben!
20 INPUT A,B
   IF A < B
   THEN MIN = A
   ELSE MIN = B
30 PRINT MI"ist das Minimum"
   FOR N = 1 TO MI
   IF ((A / N = INT (A / N)) AND (B / N = INT (B / N)))
   THEN GG = N
40 NEXT
   IF GG < > 1
   THEN PRINT "Der ggT ist"GG

   ELSE PRINT "Kein gemeinsamer Teiler!
50 IF A = 1
   THEN PRINT "Die 1 ist keine Primzahl!"
        GOTO 110
60 IF A = 2
   THEN GOTO 90
70 N = 2
   GOSUB 120
80 FOR N = 3 TO SQR (A) STEP 2
   GOSUB 120
   NEXT
90 PRINT A"ist eine Primzahl"
   GOTO 110
100 PRINT A"ist keine Primzahl -"C"ist der größte Teiler"
   [redacted]
110 IF F = 1
   THEN END
   ELSE A = B
        F = 1
        GOTO 50
120 IF A < > N AND A / N = INT (A / N)
   THEN C = A / N
        GOTO 100

   ELSE RETURN
```


von Rüdiger Sörensen.

Um eine Diskussion auf dem letzten Clubtreffen und verschiedene Aufrufe verschiedener Mitglieder noch einmal aufzugreifen, will ich ein Zitat aus einem in letzter Zeit recht erfolgreichen Film etwas abgewandelt zum besten geben : "Die Hauptaussage des CLUB 8Ø ist nicht : 'Jeder kämpft für sich allein.'"
Mir ist aufgefallen, daß vor allen anderen Ka-Jot die Problematik des Bankens untersucht und mit Programmen verdeutlicht hat. An diesem Punkt kann ich etwas hinzufügen, das dem einen oder anderen vielleicht hilft, auf alle Fälle aber zur Diskussion anregen soll! In meinem Model 4 Tischmodell ist ein RAM-Erweiterung um 256K eingebaut, die Anleitung stammt aus der 8Ø MICRO. Damit verfüge ich über insgesamt 32ØK RAM, die, wie üblich, nur zu jeweils 64K erreichbar sind. Unter TRSDOS kann man damit eine 256K RAMDisk einrichten, mehr ist mit meinen bescheidenen Systemkenntnissen nicht machbar. Aber unter CP/M! Denn da ist alles, wenn schon nicht selbstgemacht, so doch im Quellcode vorhanden und weitgehend verstanden. Wie bereits mehrfach erwähnt, gliedert sich CP/M in 3 Teile, CCP, BDOS und BIOS. Letzteres nimmt im Model 4 ca. 6K in Anspruch. Davon bestehen 1.5K aus Diskettenpuffern und ähnlichen Bereichen (DPH, DPB etc.).
Die einzelnen Routinen des BIOS habe ich nach Funktionen gegliedert :

CRESET	Kaltstartroutine
WRESET	Warmstartroutine
DISKIO	Lesen und Schreiben einzelner Sektoren
PPDRV	Druckertreiber
SPDRV	RS232-Treiber
VDDRV	Video-Treiber
KBDDRV	Tastatur-Treiber

Das Memory-Layout des Model 4 sieht so aus :

ØØØØH-7FFFH	untere 32K-Bank
8ØØØH-FFFFH	obere 32K-Bank
F8ØØH-FFFFH	Memory-Map für Bildschirm und Tastatur

Die Memory-Map läßt sich Port-gesteuert ein- und ausschalten. Des weiteren wird über einen Port gesteuert, welche der jeweiligen 64K aus der 256K-Erweiterung gerade aktiv sind und als alternativer Speicher parallel zum 64K-Hauptspeicher zur Verfügung stehen. Ferner kann man aus eben den 64K Alternativspeicher eine Bank von 32K in den Hauptspeicher einblenden, und zwar in die obere oder untere Adreßhälfte. Eine 256K Ramdisk ist dann recht einfach zu installieren, aber das reicht mir nicht. Denn man kann ja auch Programme im Alternativspeicher ablegen!
Und genau das habe ich getan, Teile meines BIOS sitzen nicht im Hauptspeicher sondern im AS (Alternativspeicher). Bekanntlich muß das BIOS oberhalb des BDOS residieren. Im Model 4 ist das eh zwingend, wenn man überhaupt Banken will. Also muß in den residenten BIOS-Teil eine Routine, welche das Banken übernimmt, in diesem Fall wird die untere 32K-Bank

ausgewechselt. Für den Transfer von Daten ist das gar kein Problem, es muß lediglich ein Puffer im residenten Teil sitzen, über den der Transfer laufen kann. Beispiel RAMDisk : Wenn ein Anwenderprogramm die DMA-Adresse in der unteren Bank sitzen hat und einen Sektor dorthin haben will, muß das BIOS zuerst die entsprechende Bank berechnen, diese in die untere Adreßhälfte einblenden und den richtigen Sektor in den residenten Puffer laden, die TPA-Bank wieder einblenden und den Puffer in die DMA transferieren. Das hört sich komplizierter an, als es ist.
Man stelle sich nun vor, das einige Routinen des BIOS selbst nicht resident sind, sondern in einer AS-Bank sitzen. Diese kann sich natürlich nur in der unteren Adreß-Hälfte befinden, da ja die Umschaltung vom residenten BIOS in der oberen Hälfte erledigt wird. Die gebankten Routinen müssen ferner auch in dem unteren Adreßraum lauffähig sein, das heißt, mit ORG ØØØØH oder was immer assembliert werden. Welche Routinen kommen nun in Frage? Die DISKIO-Routinen sind schwierig, da sie ihre Puffer im residenten Teil halten müssen. Auch aus einem anderen Grund, nämlich der dynamischen Änderbarkeit des gebankten BIOS, sollte man zunächst davon absehen. CRESET scheidet aus, denn hier wird der gebankte Teil initialisiert und auch (bei mir) in die Bank geladen. WRESET lohnt sich nicht, geht auch nicht, wie später noch zu berichten sein wird.
Bleiben noch die CHARIO-Routinen. Ich habe mich für folgendes Vorgehen entschieden :

Beim Booten des Systems wird CCP, BDOS und BIOS in den residenten Teil des Speichers geladen. Das BIOS enthält allerdings nur rudimentäre CHARIO-Routinen, welche für die untere Adreßhälfte assembliert sind und also oben gar nicht laufen können. Nach dem Laden des BIOS sitzen diese Routinen an der Stelle der Diskettenpuffer, können also deren Länge nicht überschreiten. CRESET schaltet dann die AS-Bank ein, in der das gbankte BIOS sitzen soll, transferiert die Routinen dorthin und initialisiert anschließend die Disk-Buffer. Die Sprungleiste im residenten Teil verweist natürlich nicht auf die gebankten Routinen, sondern auf kleine residente Treiber, die das Banken übernehmen und die entsprechende Routine anspringen.
Im letzten Info stellte Frank-Michael Schober den RSX-Trick vor, mit dem das Nachladen des CCP und BDOS verhindert werden kann. Mit Banken was besseres : CRESET schafft CCP und BDOS ebenfalls in eine Bank, dann kann WRESET aus dem Speicher nachladen! Das geht in Nullzeit (fast). Ein Nachteil der RSX-Methode ist, daß der CCP nicht mehr überschrieben werden kann und damit 2K für Anwenderprogramme verloren gehen. Außerdem bin ich nicht ganz sicher, daß kein Programm das BDOS überschreibt, meines Wissens macht das z.B. TURBO-MODULA2. Aber das ist sicher eine Ausnahme, wenn überhaupt. Weiß jemand Näheres?
Das größte Problem beim Verwenden gebankter Routinen ist der Stack. Wenn er in der unteren Bank residiert, falls XBIOS gerufen wird (sei XBIOS der gebankte Teil des BIOS), resultiert unweigerlich ein Systemabsturz daraus. Die residente Bankroutine muß also unbedingt den Stack sichern und einen Neuen anlegen. Bei mir wird zuerst die XBIOS-Bank eingeschaltet und dann der Stack dorthin verlegt. Dann die Routine geCALLt (grundsätzlich, um wieder in der oberen Bank zu landen, JP's sind sehr gefährlich!) und anschließend der

Stack wieder restauriert und die TPA-Bank eingeschaltet. Das funktioniert einwandfrei. Ein Problem taucht auf : Das BIOS ist nicht reentrant. Man muß sicherstellen, daß Interrupts ausgeschaltet snd, solange in der Bank gewerkelt wird, und daß XBIOS sich nicht selbst via BIOS-Sprungleiste ruft, da dann der Stackpointer verloren geht. Eine entsprechende Verwaltung des Stacks ist viel zu aufwendig, und diese zwei Einschränkungen sind gering.
Was habe ich denn nun damit gewonnen ? Nun, ca. 1.5K an TPA, denn den nehmen nun die gebankten Teile nicht mehr weg. Das ist nicht viel, man muß auch noch etwas abziehen für die hinzugekommenen Verwaltungsroutinen.

ABER!

Wenn man das gebankte XBIOS etwas geschickt aufbaut, ist es änderbar! Bei mir startet das XBIOS bei Adresse ØØØØH mit einer Sprungleiste, genau wie sein residentes Gegenstück. Das hat den Vorteil, daß BIOS gar nicht wissen muß, wo die XBIOS-Routinen liegen, sondern einfach in die Sprungleiste hüpft, die damit das Einzige ist, das in der Reihenfolge und Adreßlage immer beibehalten werden muß. Solange diese Bedingung beachtet wird, kann man nach Herzenslust ein neues XBIOS laden! (Deswegen habe ich auch die Disk-Routinen resident gelassen, sie würden dabei überschrieben werden). Also ein kleines Programm geschrieben, das ein XBIOS-File überprüft, ob es ein solches ist, und an die richtige Stelle schafft. **SOFORT NACH DEM KALTSTART TRITT DIESES PROGRAMM IN AKTION!** Und überhaupt, wie groß darf denn jetzt das BIOS werden? Auf die Systemspuren muß es ja nicht mehr passen, wir haben eine 32K Bank abzüglich CCP und BDOS sowie etwas Stack zur Verfügung! Massig Platz! Zusätzlich zu den CHARIO-Routinen habe ich die Sprunglast im residenten BIOS um einen Sprung erweitert. Dieser führt wieder in das XBIOS, wo anhand des Akku-Inhaltes ein Programm ausgesucht und gestartet wird. Dabei werden BC, DE und HL als Parameter aufgefaßt, je nach Routine. Man kann dahin die Grafik stecken, oder, wie bei mir, Routinen zum Stellen der RS232-Parameter, Modemsteuerung und so weiter. Außerdem kann man die Treiber fast beliebig erweitern. Mein Video-Treiber ist jetzt ein Fenster-Treiber. Default ist ein 8Ø*24 Fenster. Man kann die Grenzen setzen, wie man will, alle Operationen, Zeichenausgabe, Cursorsteuerung, CLS, CLEOL, DELLIN, INSLIN, SCROLL arbeiten im jeweils aktiven Fenster und kein Anwenderprogramm ahnt auch nur etwas davon!
Der Tastaturtreiber kann zwischen deutscher und US-Tastaur umschalten, es gibt eine Bildschirm-Hardcopy-Routine, der Druckertreiber erkennt Grafik-Modus und steuert bei 8-Nadel-Grafik 24 Nadeln an, Tastaturklick (abschaltbar), der Phantasie sind keinerlei Schranken auferlegt.

Ich gerate ins Schwärmen. Noch eine Idee : bekanntlich liegen CCP und BDOS inzwischem als leistungsfähige Z8Ø-Versionen und im Quellcode vor. Warum also nicht CCP und BDOS ebenfalls um gebankte Teile erweitern? Kommandozeilen-Editor? Disk-Buffering? Wo liegt die Grenze ?

An die Arbeit ! *Rüdiger*

Ich hatte beschrieben, wie es mir gelungen ist, mehr TPA zu ergattern, indem ich Teile des BIOS gebankt habe. Inzwischen hat sich da einiges geändert. Mein Spezi Lucius, Besitzer eines Model 4 Tischmodells mit 320K RAM, wie ich auch, regte mich an/auf, doch das gesamte Betriebssystem in eine ausführbare Datei zu stecken, so daß ein neues OS einfach durch Aufruf eines Programmes gestartet werden kann. Das habe ich getan. Auf meinen Disketten liegen jetzt zwei Files namens BOOT.COM und ZBOOT.COM rum. Der erste lädt ein 57K TPA System mit Z80DOS 2.34 und Z80CCP. Der zweite lädt ein ZCPR33 System mit 55K TPA und demselben BDOS. Zusätzlich zu den gebankten Teilen aus dem ladbaren XBIOS (wie beschrieben) habe ich nun auch den Floppy-Treiber gebankt. Das XBIOS kann folgendes : Fensterverwaltung, d.h. Cursoransteuerung, Ausgabe von Zeichen, scrolling, insert/delete Line erfolgen in einem definierbaren Fenster. Es können unterschiedliche Fenster definiert werden und das jeweils aktive Fenster gewechselt werden. Der Tastaturtreiber unterhält einen 128 Byte Puffer. Dieser kann von außen mit beliebigen Strings gefüllt werden. Dadurch läßt sich eine Pseudo-Batch-Steuerung realisieren, die sogar direkten Tastatur-I/O unterstützt. Außerdem sind alle Tastaturcodes in einer Tabelle abgelegt, die änderbar ist. Man kann also seine WordStar-Tastaturbelegung laden und danach zur Standardbelegung zurückkehren. Der Video-Treiber ist umschaltbar, auf Tastendruck zeigt er entweder die deutschen Umlaute nebst 'ß' oder die amerikanische Belegung mit geschweiften/eckigen Klammern an. Das geht natürlich nicht bei memory-gemappten Editoren. Es ist übrigens prinzipiell unmöglich, CP/M *beides* auf einmal darstellen zu lassen. XBIOS unterstützt auch die Ausgabe von Zeichen in die HRG. Momentan bin ich gerade dabei, Grafik-Primitive einzubauen. Ferner ist es möglich, Speicher in der XBIOS-Bank zu allokieren und zu lesen/beschreiben. Die RS232-Parameter können gesetzt werden. Der Druckertreiber erkennt Grafik-Codes und setzt sie in 24-Nadel-Grafik um.

Das ganze System benötigt eine 32K-Bank. (It's nice to have 32K for your BIOS...). Damit bleibt mir noch eine RAM-Disk von 224K. Warmstarts laden CCP aus der RAM-Bank nach.
Ich möchte anregen, dieses XBIOS-System zu standardisieren, um eine einheitliche Schnittstelle für Anwenderprogramme zu schaffen. Dazu eine Kurzbeschreibung :

Die Einsprünge in XBIOS erfolgen über einen RST 28H. Diese Adresse ist von DRI freigegeben, es darf damit keine Probleme geben. Beim Warmstart wird der Vektor bei 39H neu initialisiert, falls ein Programm ihn doch einmal zerstört haben sollte. Der Einsprung in's XBIOS muß folgenden Konditionen genügen :
In A steht die Nummer der gerufenen Funktion (0..255).
A = 0 ist die Initialisierung des XBIOS. Nicht implementierte Funktionen RETurnen einfach. Alle anderen Register können zur Parameterübergabe benutzt werden. Register B hat eine Sonderstellung : hier kann eine Unter-Funktion übergeben werden. Alle Register außer IX und IY werden verändert. Über Fehlerbedingungen habe ich mir noch keine Gedanken gemacht, ich denke, das BIOS-Fehlerschema sollte übernommen werden. Die Vergabe der Funktionsnummern an die entsprechenden Aktivitäten ist bei mir historisch bedingt, über Änderungen lasse ich gerne mit mir reden. Hier eine Liste :

Nummer in A	Beschreibung
0	Initialisierung des XBIOS
1	N/A
2	N/A
3	N/A
4	N/A
5	N/A
6	BaudSet

IN : C = Baudrate Code.
OUT : -

Bedeutung von C :

0 = 50 bd
1 = 75 bd
2 = 110
3 = 134.5
4 = 150
5 = 300
6 = 600
7 = 1200
8 = 1800
9 = 2000
10 = 2400
11 = 3600
12 = 4800
13 = 7200
14 = 9600
15 = 19200

7.....spopts

Set Serial Port Options

IN : C = Option Byte

Bits :

0 : 1 = RTS
1 : 1 = DTR
2 : 1 = BREAK
3 : 1 = disable parity
4 : 1 = 2 Stop Bits, 0 = 1 Stop Bit
5+6 : Word Length
7 : 1 = even parity

8.....WDefine

IN : HL, BC

H = linke Spalte
L = obere Zeile

B = Anzahl Spalten
C = Anzahl Zeilen

!!! A C H T U N G !!!

Es werden keinerlei checks auf korrekte Parameter durchgefuehrt! Insbesondere führen zu kleine oder zu grosse Fenstergrenzen zum Systemabsturz!

9.....MAlloc

IN : HL OUT : HL

Speicher in der XBIOS-Bank allokieren.

Input : HL = Anzahl der zu allokierenden Bytes
Output :
HL = 0000H : Fehler
HL # 0000H : Adresse des Speichers

10.....MFree

Allokierten Speicher wieder freigeben

IN : HL OUT : HL

Input : HL = Adresse des freizugebenden Speicherbereichs
Output :
HL = 0000H : Fehler
HL # 0000H : Speicher freigegeben.

11.....FKBuf

Den Tastaturpuffer mit einem String fuellen.
IN : HL = Zeiger auf Buffer.
HL muss groesser als 7FFFH sein.
Der Buffer darf hoechstens 128 Byte gross sein und muss mit 00H enden.

12.....RetBufAdr

HL liefert die Adresse des 128 byte langen Buffers (WkBuf) in der oberen RAM-Haelfte, der z.B. fuer XBIOS-Datentransfer benutzt werden kann.

13.....MovBuf

Kopiert den Inhalt des durch (12) gegebenen Buffers WkBuf an die durch HL spezifizierte Stelle in der unteren XBIOS-Bank oder umgekehrt.

IN :

HL = Zeiger auf Buffer (HL > 07FFF -> Fehler)
C = 0 : XBIOS -> WkBuf
C <> 0 : WkBuf -> XBIOS
B = Anzahl der zu uebertragenden Bytes.
Falls B > 128, werden 128 bytes uebertragen.

14.....GetWin

Gibt in HL einen Zeiger auf den aktuellen Fenster-DCB zurueck.

15.....SelectWin

Selektiert einen neuen Fenster-DCB

IN : HL

HL = Zeiger auf neuen Fenster-DCB

16.....NewWin

Erzeugt einen neuen Fenster-DCB.

OUT : HL

HL = Zeiger auf neuen DCB (HL = 0 , falls Fehler).
Die Parameter des neuen DCB entsprechen denen des gegenwaertig aktiven DCB.
Der neue DCB wird nicht selektiert. Das muss explizit durch Aufruf von (15) getan werden.

17.....RestoreDefaults

Loescht alle Fenster-DCB's ausser dem urspruenglichen.
Traegt default-Werte in den DCB ein (80*24).
Alle allokierten Speicherbereiche werden freigegeben.

18.....N/A
19.....N/A
20.....N/A
21.....N/A
22.....N/A
23.....N/A

24.....Grafik-Zeichen schreiben u.a.

IN : B : Funktionsunternummer

B = 0 : Schreibe Zeichen in C in Grafikkarte
B = 1 : Setze Grafik-Cursor nach
H : Spalte (0..79)
L : Zeile (0..239)
B = 2 : Gib zurueck Cursor-Adresse
H : Spalte
L : Zeile
OUT : HL, je nach Funktion

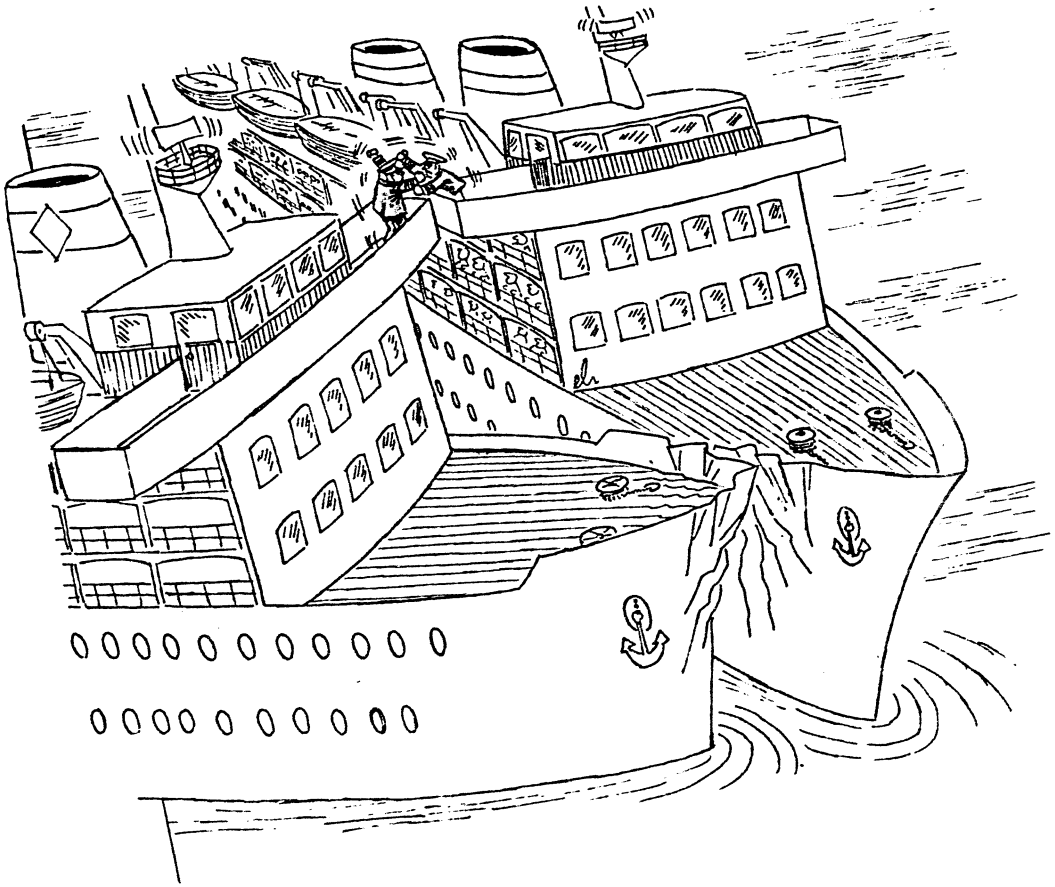
Anm. Der Cursor ist nicht sichtbar und wird intern verwaltet. Steuerzeichen werden nicht erkannt. Legal sind nur Zeichen zwischen 20H und 07FH. Alle anderen werden ignoriert. Bei Erreichen eines Zeilenendes wird ein Umbruch durchgefuehrt. Dabei wird ein Zeilenabstand von 10 Grafikzeilen angenommen.

Anm. Das Erzeugen verschiedener Fenster kann mit (16) geschehen. Die Zeiger auf die DCB's muessen gespeichert erden. Dann alle DCB's nacheinander selektieren (15) und Parameter eintragen(8). Die Ausgabe von Zeichen erfolgt nur in das gegenwaertig aktive Fenster. SetWin kann zwischen den verschiedenen Fenstern wechseln.

Ich denke, daß das Konzept recht interessant ist. Wie denkt Ihr darüber? Wer hat Interesse an einer Zusammenarbeit zwecks Standardisierung? Bitte schreibt mir Eure Meinung dazu!

Ach, übrigens, wer das System haben will, soll mir eine formatierte 80TrDSDD Monte SYSTEM Diskette zukommen lassen. Der Speicher des Model 4/4p muß allerdings 128K groß sein. Als Zugabe erhält jeder Besteller ein vollständiges ZCPR33-System. Die RAM-Disk ist bei 128K Speicher allerdings nur noch 32K groß. Wen's stört...

Euer auf Echo wartender *Rüdiger*



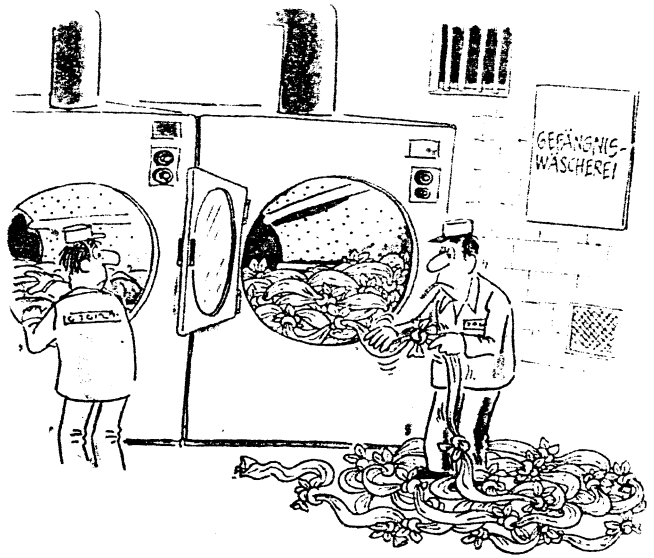
Unterbrechung gestattet?

Im Betriebssystem gibt es immer zeitkritische oder aus sonstigen Gründen anfällige Routinen, bei denen Interrupts gesperrt werden müssen. Das stellt ja auch kein Problem dar, ein DI an der richtigen Stelle sorgt für den ungestörten Verlauf des Programmes. Wenn dieses jedoch fertig ist, sollen dan die Interrupts wieder zugelassen werden ? Oder nicht ? Das hängt davon ab, ob *vor* dem Abschalten Interrupts zulässig waren. Aber wie kann man das herausbekommen ?
Man betrachte die Z80-Befehle

```
     LD   A,I
oder
     LD   A,R
```

Bei jedem dieser Befehle wird das Flag P/V auf den Wert von IFF2 gesetzt, man erhält also Aufschluß darüber, ob Interrupts erlaubt sind! Also : Vor dem DI ein LD A,I, das P/V-Flag in geeigneter Weise speichern und schon kann man hinter der kritischen Routine die Interrupts selektiv zulassen oder sperren!

Rüdiger



ZCPR33 -- die Vierte.

Noch einmal : Installation von ZCPR33 auf den verschiedenen Z80-Maschinen.
Anscheinend ist es doch nicht so leicht, sein CP/M System auf das wesentlich bessere ZCPR-System umzurüsten. Das liegt vermutlich an meinen etwas unverständlichen Ausführungen zu diesem Thema. Da noch keinerlei Erfolgsmeldungen bei mir eingetroffen sind, versuche ich nochmals, in Form einer Schritt-für-Schritt-Anleitung, die Installation zu erklären. Im Übrigen sind die Bugs, die ich das letzte Mal beschrieben hatte, ausgemerzt und können nun vermieden werden.

Was brauche ich ?

HARDWARE

Einen Computer mit Z80, Z180, Z280 oder HD64180 CPU.
Mindestens ein Diskettenlaufwerk (am besten 2x80T).

SOFTWARE

Ein laufendes CP/M 2.2.
Quellen des BIOS.
ZCPR33.LBR (Public Domain, bei mir erhältlich).

Im folgenden differieren die Vorgehensweisen. Mein Ansatz war Standard, deshalb wird er hier behandelt.)

Einen Z80-Assembler (z. B. ZAS.COM).
Einen Text-Editor.
SYSGEN.COM
evtl. MOVCPM.COM
Einen Debugger (WADE, DSD, SID, DDT o.ä.)

Das ist alles.

1. Schritt.

Welche Features soll das ZCPR33-System bekommen? Alles kostet seinen Platz. In meinem System habe ich nur auf die Implementation des Input/Output Packages verzichtet. Der Rest nimmt etwa 3.5 KBytes in Anspruch. Jedes Byte lohnt sich! Hier ist der Auszug aus meinem BIOS, der das System beschreibt. Bei mir beginnen die Z-Buffer bei Adresse 0EA00H. Das ist natürlich unterschiedlich. Rechnet aus, wie lang die Buffer werden und verschiebt das CP/M um den entsprechenden Betrag nach unten (mit MOVCPM). Wer die Quellen zu BDOS und BIOS hat, lebt leichter, er muß lediglich neu assemblieren. TESTET DAS NEUE SYSTEM! Es muß genau so arbeiten wie zuvor, nur das die TPA kleiner ist. Alles klar ? Betrachtet den Auszug aus meinem BIOS. Z3RCP liegt hinter dem eigentlichen BIOS und vor den Disk-Tabellen. Das ist aber egal. Die Z-Buffer müssen nicht mal am Stück aneinderhängen, aber das ist von Vorteil.


```
        if ZCPR33

;       I/O package not implemented

z3rcp   equ     0ea00h          ; resident command package

;   RCP beginnt bei 0EA00H (bei mir!)

z3fcp   equ     z3rcp+2048      ; Flow Command Package
z3env   equ     z3fcp+512       ; environment descriptor
z3shstk equ     z3env+256       ; shell stack
z3mbf   equ     z3shstk+128     ; message buffers
z3fcb   equ     z3mbf+80        ; ext. fcb
z3ndrb  equ     z3fcb+48        ; named directory buffer
z3mclb  equ     z3ndrb+256      ; multiple command line buffer
z3xstk  equ     z3mclb+208      ; external stack
z3whb   equ     z3xstk+49       ; Wheel Byte
z3path  equ     z3whb+1         ; external Path
eobios  equ     z3path+48       ; end Of BIOSZ

        else

eobios  equ     $

        endif                   ; IF ZCPR33
```

Wenn das System mit weniger TPA einwandfrei läuft, kommt der

2.Schritt.

Im BIOS gibt es eine Kaltstartroutine. Das BIOS hat immer eine Sprungleiste, diese steht i.allg. am Anfang des BIOS. Der erste Sprung verzweigt in die Kaltstartroutine. Diese muß modifiziert werden. Sie erhält die zusätzliche Aufgabe, die Z-Buffer zu initialisieren.

Hier ein Auszug aus meiner Kaltstartroutine :

```
        if      ZCPR33
z3init

; Alle ZCPR-Buffer mit 00H füllen

        call    clrmem
        DW      z3rcp           ; clear ZCPR3 buffers
        DW      ((z3path+48)-z3rcp) (**)

;   Kaltstartkommando -> externen Kommandopuffer

        ld      hl,intcom
        ld      de,z3mclb
        ld      bc,cmdlsz
        ldir                    ; Initialize for Cold Boot Command

;   externen Path initialisieren

        ld      hl,path
        ld      de,z3path
        ld      bc,4            ; Initialize path
        ldir
```

Wheel-byte auf Supervisor setzen

```
        ld      a,0ffh
        ld      (z3whb),a       ; Initialize Wheel to non-protect
        ret
```

Das ist der Initial-Suchpfad,
A0: -> A15:

```
path    defb    'A'-'§',0,'A'-'§',15


intcom  defw    z3mclb+4                ; nxtchr
        defb    208                     ; bufsiz
        defb    zcmdend                 ; chrcnt
```

Dieser Kommandostring wird beim KALTSTART ausgeführt :

```
zcmdstr defb    'LDR SYS.ENV;STARTUP'
zcmdend equ     $-zcmdstr
cmdlsz  equ     4+zcmdend

        endif
```

Das ist genau der Code, der notwendig ist, um die Z-Buffer zu initialisieren. Mehr braucht es nicht. Und damit ist das Einbinden von ZCPR33, was das BIOS angeht, beendet.
Ändert die Kaltstartroutine und TESTET DAS SYSTEM!

3. Schritt.

Notiert alle Adressen der Z-Buffer (RCP, FCP etc.). Ferner die BDOS-Startadresse (steht an Stelle 06H im Speicher) und die Adresse des CCP (BDOS-806H). Kopiert ZCPR33.LBR auf eine leere Diskette. Packt die LBR aus (mit NULU oder sowas) und löscht die LBR von der Diskette. Die Dateien sind gecrunched oder gesqueezed und müssen entsprechend expandiert werden. Löscht dann alle .?Q? und .?Z? Dateien von der Diskette.
Editiert die Datei Z3BASE.LIB. In sie müssen alle Adressen eingetragen werden, die Ihr zuvor notiert habt. Editiert die Datei Z33HDR.LIB und wählt die gewünschten Features des Systems aus. Versucht es mit JUMP, GO und SAVE. Assembliert ZCPR33 in ein HEX-File.

```
A>ZAS ZCPR33 H
```

4. Schritt.

Als nächstes wird der alte CCP mit ZCPR33 überschrieben und auf den Systemspuren abgelegt. Dazu wird SYSGEN und ein Debugger benötigt. Das System muß von der Diskette mit dem geänderten BIOS gelesen werden (hier B:)

```
A>SYSGEN
     ...
     PRESS A..P TO READ SYSTEM TRACKS OR <ENTER>:      B
     INSERT DISK IN B: AND PRESS <ENTER>       :      <ENTER>
     PRESS A..P TO WRITE SYSTEM TRACKS OR ENTER :     <ENTER>

     SYSGEN COMPLETED.

A>SAVE 44 CPMREV.BIN
```


Die Zahl 44 ist von System zu System unterschiedlich, sie gibt die Größe in Pages (256 Bytes) an. SYSGEN sollte die Information rausrücken. Jetzt gibt es eine Datei CPMREV.BIN, die das System enthält. Wo war doch gleich die Adresse des CCP? sagen wir mal, CC00H. Nun muß der Debugger gestartet werden.

```
A>DDT CPMREV.BIN
```

Damit wird gleich das System geladen. Wie gesagt, es enthält auch den CCP. Er sollte so um die Adresse 800H im Systemabbild liegen, sucht ihn mit dem Debugger. Am Anfang des CCP steht eine Copyrightnotiz von Digital Research. Sucht um 800H, eher weniger. Sagen wir, Ihr findet den CCP bei 800H.

```
H800,CC00           berechnet den Offset für das Laden von ZCPR33
     D400 3C00      Summe und Differenz von 800H und CC00H
IZCPR33.HEX         Initialisiert einen FCB
R3C00               lädt ZCPR33.HEX mit einem Offset von 3C00.
                    Prüfen!
^C
A>SAVE 44 CPM.BIN   sichert das neue System.
A>SYSGEN CPM.BIN usw. -> System auf die Spuren schreiben.
```

(Anm. Diese Vorgehensweise gilt sinngemäß genauso für Änderungen des BIOS und BDOS. Allerdings eben nur, wenn in ein HEX-File assembliert wird. Es gibt Assembler, die die direkte Erzeugung eines COM-Files unterstützen.)

Uff! Bootet das System. Es sollte sagen :

```
A0>LDR SYS.ENV;STARTUP?
```

Dann ist alles in Ordnung.

5. Schritt.

Nun müssen die System-Segmente vorbereitet werden. Dies sind zunächst SYS.ENV, SYS.RCP, SYS.FCP, SYS.NDR und SYS.Z3T. Letztere kann erstmal weggelassen werden. Editiert SYSENV.LIB und SYSENV.ASM und erzeugt daraus ein HEX File.

```
A>MLOAD SYSENV
     Warning! program origin not at 100H!
A>REN SYS.ENV=SYSENV.COM
```

Macht dasselbe mit ZCPR33xx.RCP und .FCP, so daß Ihr die Files SYS.RCP und SYS.FCP erhaltet. Legt diese Dateien in A15: der ZCPR33 Boot-Diskette ab.

Letzter und sechster Schritt.

Es müssen nur noch die Utilities installiert werden, besonders VALIAS.COM und LDR.COM. Erzeugt eine Datei, die, Zeile für Zeile, die Namen (*.COM) aller ZCPR-Utilities enthält. Bringt SYS.ENV, Z3INS.COM und die Liste, die SYS.INS heißen sollte, nach A0:.

```
A0>Z3INS SYS.ENV SYS.INS
```

Das dauert ein paar Minuten, dann sind alle Utilities installiert und können benutzt werden. Bootet das System! LDR sollte SYS.ENV laden, über STARTUP? darf er noch meckern. Legt einen STARTUP.COM an mit

```
A>VALIAS STARTUP
     (vorausgesetzt, VALIAS ist installiert)
```

Er sollte folgende Befehle enthalten :

```
LDR SYS.RCP,SYS.FCP,SYS.Z3T,SYS.NDR
PATH $$$$: A0: A15:
```

Legt STARTUP.COM in A0: oder A15: ab.
Das System sollte jetzt booten, außer .Z3T und .NDR alle Segmente laden und eine Suchpfad einrichten über
current drive/user
drive a user 0
drive a user 15

Nun müßt Ihr nur noch mit MKDIR.COM eine Datei SYS.NDR einrichten, wenn Ihr Directories mit Namen haben wollt. Falls SYS.ENV richtig ist, könnt Ihr vorerst auf SYS.Z3T verzichten.

Falls Ihr ZFILER.COM oder ZF.COM habt, startet es, sonst vielleicht VFILER.COM. Es wird sich über fehlende Terminal-Installation schon beschweren. Wenn alles aber funktioniert, dann

*(** Aus drucktechnischen Gründen (!) habe ich die eckigen Klammern, die ZAS für seine Arithmetik verlangt, durch runde Klammern ersetzt. Das müßt Ihr ändern.)*

HERZLICHEN GLÖCKWUNSCH!!!

Rüdiger

Prommer 80 unter G-DOS

Ja, Ihr habt richtig gelesen. Ab jetzt müssen diejenigen von Euch, denen es nicht gegönnt ist, CP/M zu 'fahren', aus welchen Gründen auch immer, nicht mehr auf das EPROM-Brennen verzichten. Es geht auch unter G-Dos bzw. Newdos.

Der Grund dafür ist meine Faulheit und Unerfahrenheit. Da ich noch keine nennenswerten Erfahrungen mit CP/M habe programmiere ich noch unter G-Dos mit ZEUS. Da ich in der letzten Zeit mein Boot-Eprom modifiziert habe, mußte ich den Inhalt immer mit CPMAC auf CP/M-Disketten kopieren, dann CP/M booten, Prommer anwerfen, programmieren and so on. Diese Prozedur ging mir mit der Zeit an meine Nerfen. Also mußte ein Programm her welches unter G-Dos lief. Nein, ich habe keines geschrieben, dazu war ich zu faul (s.o.), außerdem gibt es ja ein sehr gutes, das von Becker. Nur das läuft unter CP/M.

Beim durchstöbern meiner Erinnerung sowie einiger CP/M Bücher fiel mir auf, daß CP/M ein Rechnerunabhäniges Betriebssystem ist und eigentlich alles was mit der Hardware zu tun hat über eine Sofwareschnittstelle erledigt. Als kleinen Exkurs in CP/M für diejenigen die von CP/M noch weniger wissen als ich, hier eine Erläuterung wie CP/M das macht.

Alle Ein- und Ausgaben die den Bildschirm oder die Diskette betreffen werden über das BDOS abgewickelt. BDOS steht für ' Basic Disk Operation System '. Das BDOS wird angesprungen über die Adresse 0005H. In Register C wird die Funktionsnummer übergeben. Weitere Parameter in anderen Registern. (siehe auch Lising) Die Befehlsfolge

```
LD   E,41H
LD   C,02H
CALL 0005h
```

zum Beispiel gibt den Buchstaben 'A' auf dem Bildschirm aus. Genauso macht es auch das Programm von Becker genannt PROM. Soweit zu CP/M.

Ich habe nun das Programm mit CPMAC nach G-Dos kopiert, disassembliert und dann alle Call-Befehle nach 0005H in 'bdos' umbenannt. Jetzt habe ich eine Routine geschrieben die das macht was BDOS im CP/M auch macht. Siehe Listing. Die übergebene Funktionsnummer wird abgefragt und dann zur entsprechenden Routine verzweigt. Dort werden die zusätzlichen Parameter in die, die G-Dos braucht, umgewandelt und dann die entsprechende G-Dos-Funktion aufgerufen. Der Sprung von 'bdos' nach 'system' ist nötig, weil das Hauptprogramm diese Adresse " umbiegt " um die Register zu retten. Im Hauptprogramm mußte fast nichts geändert werden, bis auf das Trennzeichen bei Filenamen. Dieses ist bei CP/M ein Punkt "." und bei G-Dos ein "/". Auch scheinen die Funktionen 'read' und 'write' sehr ungewöhnlich. Zur Erklärung: CP/M benutzt 128-Byte Sektoren und G-Dos 256-Byte. Um einer komplizierten Umrechnung zu entgehen, habe ich einfach Byteweise, und zwar immer 128 an der Zahl, übergeben, das war der einfachste Weg. Ich stimme Euch zu, wenn Ihr der Meinung seid das ginge auch einfacher, (besonders Arnulf) aber mir kam

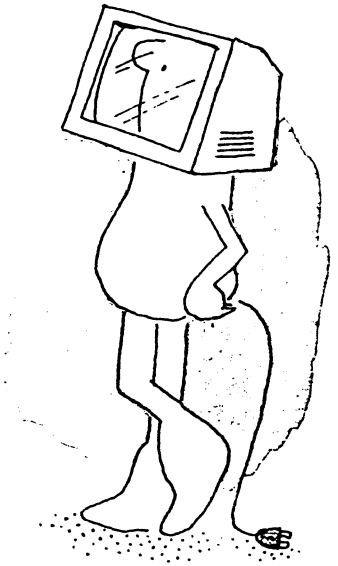
es nur auf die Funktion an und wenn Ihr nach der Geschwindigkeit fragt, so ist G-Dos sowieso viel schneller.

Das abgedruckte Listing zeigt natürlich nicht das ganze Programm, dafür müßte ein Extra-Info erscheinen, sondern nur die Bearbeitung der gebrauchten Funktionen. Es ist nur auf das eine Programm abgestimmt, aber wer Lust hat kann es ja für andere Dinge "mißbrauchen". Da wären wir auch gleich beim Urheberrecht, das bleibt natürlich beim Verfasser, und die kommerzielle Weitergabe ist nicht erlaubt. Anders ist es bei meinem Zusazprogramm, das darf, und soll geändert, erweitert und weitergegeben werden, aber auch nur kostenlos.

Jetzt zu Eurer nächsten Frage, wie Ihr an das Programm rankommt. Bitte schreibt nicht mir wenn Ihr es haben wollt, ich habe das Programm sowie die Scorces dazu an Oliver unseren G-Dos/Newdos-Disk-Jokey geschickt, bitte wendet euch an Ihn. Das lauffähige Programm heist 'PROMGDOS'. Als weiteres gibt es ein Programm 'PROMINST', dieses dient dazu das Hauptprogramm an Euern Rechner und Prommer anzupassen. Wenn Ihr allerdings Fragen zum Programm selbst, oder Beschwerden habt, dann wendet Euch an mich. Aproppos Beschwerden, ich übernehme natürlich keine Garantie damit, aber das kennt Ihr ja.

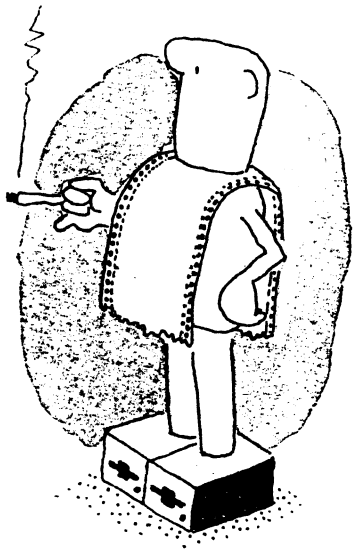
Zu Erwähnen wäre noch, daß PROMINST den Namen mit Extension haben will, sonst stürzt es ab. Als zweites, das Format der Dateien die PROMGDOS generiert, oder haben will ist immer noch CP/M. Das heißt, Ihr müßt die Dateien von Ladecodes befreien, oder diese hinzufügen, das geschieht mit den Programmen CMDER/BAS und COMER/BAS. Die sind auch mit auf der DISK die Oliver hat.

Tschüss



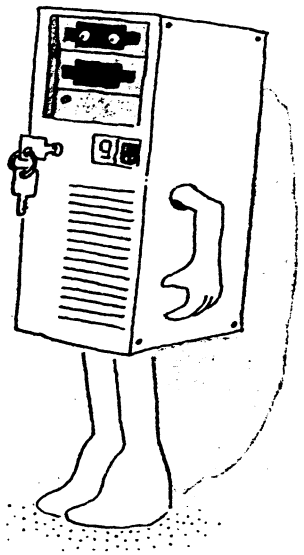

```
00001 ;       BDOS/SRC
00002 ;*************************************************
00003 ;
00004 ; Diese Routine simuliert die CP/M-BDOS-Funktionen,
00005 ; die für das Programm PROM von Becker gebraucht
00006 ; werden, unter G-DOS.
00007 ;
00008 ; Änderungen, Erweiterungen sowie Unkommerzielle
00009 ; Unkommerzielle Weitergabe sind ausdrücklich erwünscht.
00010 ;
00011 ; (c) Mai 1989 by Andreas Magnus HACKNUS-SOFTWARE
00012 ;
00013 ;*************************************************
00014         ORG     5200h
00015         DI
00016         JP      m0100           ;Start des CP/M-Programmes
.
00017
00018 ;SYSTEM-Routinen zur Anpassung von PROM/CMD an G-DOS.
00019
00020 bdos    JP      system          ;SYSTEM ist der Einsprung
in das
00021                                 ;Beriebssystem
00022
00023 system  LD      A,C             ;Parameter in den ACCU
00024         CP      01
00025         JP      Z,getkey        ;warte bis Taste gedrückt
00026         CP      02
00027         JP      Z,out           ;Zeichen an Bildschirm
00028         CP      09
00029         JP      Z,display       ;Zeichenkette an den
00030                                 ;Bildschirm
00031         CP      10
00032         JP      Z,zeile         ;Zeile von Tastatur lesen
00033         CP      11
00034         JP      Z,inkey         ;Tastatur abfragen
00035         CP      13
00036         JP      Z,sysres        ;SYSTEM zurücksetzen
00037         CP      14
00038         JP      Z,select        ;Laufwerk anwählen
00039         CP      15
00040         JP      Z,open          ;Datei öffnen
00041         CP      16
00042         JP      Z,close         ;Datei schließen
00043         CP      20
00044         JP      Z,read          ;sequentiell lesen
00045         CP      21
00046         JP      Z,write         ;sequentiell schreiben
00047         CP      22
00048         JP      Z,init          ;neue Datei anlegen
00049         CP      25
00050         JP      Z,getdrv        ;Aktuelles Laufwerk abfrag
en
00051         CP      26
00052         JP      Z,setdma        ;Sectorpuffer setzen
00053         RET                     ;Fehler
00054 getkey  PUSH    DE              ;DE wird noch gebraucht
00055         CALL    0049h           ;auf Zeichen warten
00056         CALL    0033h           ;Zeichen ausgeben
00057         POP     DE              ;wieder zurück
00058         RET                     ;das wars
00059
00060 out     LD      A,E             ;Zeichen in den Accu
00061 out1    CP      0ah             ;ist es LF ?
00062         RET     Z               ;wird nicht ausgegeben
00063         CP      08h             ;Backspace ?
   64         JR      NZ,normal
   65         LD      A,24            ;24 daraus machen (Rückschritt)
```

```
00066                                   ;ohne löschen)
00067 normal  PUSH    DE
00068         CALL    0033h             ;Zeichen ausgeben
00069         POP     DE
00070         RET
00071
00072 display LD      A,(DE)            ;Zeichen in den ACCU
00073         CP      '$'               ;Ende der Zeichenkette ?
00074         RET     Z                 ;fertig
00075         CALL    out1              ;Zeichen ausgeben
00076         INC     DE                ;nächstes Zeichen
00077         JR      display           ;bis alle fertig
00078
00079 zeile   PUSH    DE
00080         PUSH    HL
00081         PUSH    BC
00082         EX      DE,HL             ;Pufferanfang nach HL
00083         INC     HL
00084         INC     HL                ;hier steht das erste Zeic
hen
00085         LD      B,80              ;maximale Zahl der Zeichen
00086         CALL    05d9h             ;B-Zeichen einlesen
00087         DEC     HL                ;BUFFER + 1
00088         LD      (HL),B            ;Eingegebene Zeichen
00089         DEC     HL                ;Buffer + 0
00090         LD      (HL),C            ;Max. Zahl der Zeichen
00091         POP     BC
00092         POP     HL
00093         POP     DE
00094         RET                       ;fertig
00095
00096 inkey   PUSH    DE
00097         CALL    002bh             ;Tastatur abfragen
00098         POP     DE
00099         RET
00100
00101 sysres  RET                       ;wird bei G-DOS nicht gebr
aucht
00102
00103 select  RET                       ; dito
00104
00105 open    CALL    umw               ;FCB von CP/M nach G-DOS w
andeln
00106         CALL    4424h             ;OPEN FILE
00107 rück    RET     Z                 ;alles klar
00108         CP      18h               ;Datei nicht gefunden ?
00109         RET     NZ                ;anderer FEhler
00110 fehler  LD      A,0ffh
00111         RET
00112
00113 close   LD      DE,fcb            ;FCB des zu schließenden F
iles
00114         CALL    4428h             ;Dos-Call Datei schließen
00115         JR      rück              ;weiter wie nach open
00116
00117 read    PUSH    DE                ;FCB retten
00118         LD      DE,fcb            ;FCB des geöffneten Files
00119         LD      HL,(dma)          ;HL = Adresse des Buffers
00120         LD      B,128             ;128 Byte lesen
00121 readlop CALL    0013h             ;nächstes Byte lesen
00122         JR      NZ,err            ;wenn Fehler aufgetreten
00123         LD      (HL),A            ;und in den Puffer
00124         INC     HL
00125         DJNZ    readlop           ;bis alle Bytes gelesen
00126 normaus XOR     A                 ;alles OK
00127 err     POP     DE
00128         RET
00129
00130 write   PUSH    DE                ;FCB retten
```

```
00131         LD      DE,fcb            ;FCB des geöffneten Files
00132         LD      HL,(dma)          ;HL= Adresse des Buffers
00133         LD      B,128             ;128 Byte schreiben
00134 writelo LD      A,(HL)            ;Byte aus dem Puffer
00135         CALL    001bh             ;nächstes Byte schreiben
00136         JR      NZ,err            ;wenn Fehler aufgetreten
00137         INC     HL                ;nächste Pufferstelle
00138         DJNZ    writelo           ;bis alle Byte's geschrieb
en
00139         JR      normaus           ;normaler Rücksprung
00140
00141 init    CALL    umw               ;s.o.
00142         CALL    4420h             ;INIT FILE
00143         LD      HL,m3080          ;Sectorpuffer für Ein-Ausg
abe
00144         LD      (dma),HL          ;als Sectorpuffer abspeich
ern
00145         RET     Z                 ;alles klar
00146         CP      1ah               ;Inhaltsverzeichnis voll
00147         JR      Z,fehler          ;dann Fehler
00148         CP      1bh               ;Disk voll ?
00149         JR      Z,fehler          ;dann auch
00150         RET                       ;anderer Fehler
00151
00152 getdrv  RET
00153
00154 setdma  LD      (dma),DE          ;DE in das DMA-Flag
00155         RET                       ;das wars
00156
00157
00158 umw     INC     DE                ;DE zeigt auf FILENAMEN
00159         LD      HL,filenam        ;Zwischenspeicher für File
name
00160         LD      B,8               ;8 Zeichen übertragen
00161 fnamelo LD      A,(DE)            ;Zeichen aus CP/M Filename
00162         CP      20h               ;ist der Filename zuende ?
00163         JR      Z,ext             ;dann bei der Extension we
itermachen
00164         LD      (HL),A            ;in G-Dos Filename übertra
gen
```

```
00165           INC     HL
00166           INC     DE              ;beide ein höher
00167           DJNZ    fnamelo         ;bis 8 Zeichen übertragen
00168 ext       CP      ' '             ;war Fname zuende
00169           JR      NZ,ext1         ;nein
00170 exloop    INC     DE              ;DE + 1
00171           DJNZ    exloop          ;DE auf EXT stellen
00172 ext1      LD      (HL),'/'        ;Trennungsstrich
00173           INC     HL              ;hier steht die EXT
00174           LD      B,3             ;3 Byte für EXT
00175 extloop   LD      A,(DE)          ;Zeichen in Accu
00176           CP      ' '
00177           JR      Z,ok            ;EXT zuende
00178           LD      (HL),A
00179           INC     HL
00180           INC     DE
00181           DJNZ    extloop         ;bis 3 Byte übertragen
00182 ok        LD      (HL),0dh        ;Endekennung für G-Dos
00183           LD      HL,filenam
00184           LD      DE,fcb          ;FCB für Files
00185           CALL    441ch           ;Filename in FCB übertrage
n
00186           LD      B,0             ;Recordlänge
00187           LD      HL,buffer       ;Buffer für den File
00188           RET                     ;zurück
00189
00190 filenam   DM      'FILENAME/EXT',0dh      ;Platz für den FIL
ENAMEN
00191 fcb       DS      32                      ;Platz für den FCB
00192 dma       DW      0000                    ;Platz für Buffera
dresse
00193 buffer    DS      256                     ;Platz für die Dat
en
00194
00195           END     5200h
```



Fensterln mit CP/M

Erstes Ergebnis des Modula-Projekts

Wer hat sich nicht schon immer gewünscht, diese niedlichen kleinen Fensterchen mit dem Rahmen drumherum auf seinem CP/M-Rechner zu haben? Ich weiß: Keiner. Aber jeder poplige PC kann das auch; und den benutzt Ihr ja auch jeden Tag im Büro! Also habe ich mich an die Arbeit gemacht und gemäß diversen Vorbildern (Chip-Special-Turbo-Pascal, GEM) ein kleines Modul erstellt, das genau solche Fensterchen ermöglicht. Allerdings nur mit Texten und nicht mit Grafik.

Sinn und Zweck: Beispiele

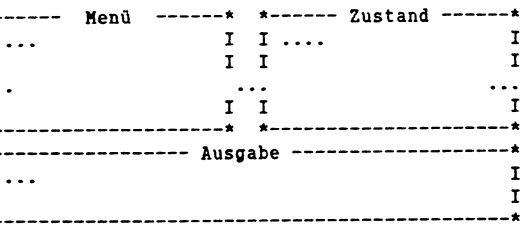
Wozu sind Fenster eigentlich zu gebrauchen? Nun, eigentlich steht uns nur der Bildschirm als "großes" Fenster zur Verfügung. Das reicht auch vollkommen, solange wir nur einen Text editieren wollen, nur ein Menü sehen usw. Die Betonung liegt auf "ein". Windows kommen ins Spiel, sobald wir uns von dem "ein" lösen und "zwei und mehr" (Texte, Menüs usw.) auf unserer Wunschliste stehen. Dazu zwei Beispiele:

Beispiel 1: Wir möchten zwei Texte gleichzeitig editieren. Wie ordnen wir die beiden Text-Ausschnitte auf dem Bildschirm an, so daß sie sich nicht stören? Und wie verhindern wir, daß eine Änderung in einem Text (z.B. das Löschen einer Zeile) sich auf den anderen Text auswirkt? Die Lösung: Jeder Text-Ausschnitt wird in einem Window (mit Rahmen drumrum) ausgegeben. Dabei dürfen sich die beiden Windows ruhig überlappen; wir können die Teile in beliebiger Größe anzeigen und an einer beliebigen Stelle auf dem Bildschirm plazieren. Die Änderungen in einem Text wirken sich nur auf sein Window aus. Das andere Window wird davon überhaupt nicht berührt.

Beispiel 2: Wir haben ein Menüsystem mit einem Hauptmenü; zu jedem Menüpunkt im Hauptmenü gibt es evtl. wieder ein Untermenü usw. Wenn wir nur einen Bildschirm zur Verfügung haben, müssen wir erst das Hauptmenü anzeigen, dann den Bildschirm löschen und das Untermenü anzeigen (das Ganze evtl. mehrere Stufen tief). Bei "Hochsteigen" aus den Menüs muß dann wieder jedes Mal der Bildschirm gelöscht und das Menü angezeigt werden usw. Unter Windows sieht die Lösung anders aus: Am Anfang des Programms Öffnen wir für jedes Menü ein Fenster und schreiben den Text dort hinein; der Text erscheint aber noch nicht auf dem Bildschirm. Dann geben wir die Anweisung: "zeige das Fenster *Hauptmenü*" und, schwupp, erscheint es auf dem Bildschirm. Bei jedem Untermenü geht es genauso. Dabei lassen wir das Hauptmenü-Fenster (mit seinen folgenden Untermenü-Fenstern) auf dem Bildschirm stehen. Wenn wir uns jetzt wieder "hochhangeln" geben wir nur die Anweisung "lasse das Fenster *Untermenü x* vom Bildschirm verschwinden" und schon taucht das darunterliegende Fenster mit dem nächsthöheren Menü wieder auf. Das könnte dann so aussehen:

```
*--- Hauptmenü ---*
I  a. Punkt 1     I
I  *--- Untermenü 1 ---*
I  I   x. Punkt 1.1    I
I  I   y. Punkt 1.2    I
I  I  *--- Untermenü 1.2 ---*
I  I  I   1. Punkt 1.2.1    I
I  I  I   2. Punkt 1.2.2    I
*--I  I   3. Punkt 1.2.3    I
   I  I   4. Punkt 1.2.4    I
   *--I   5. Punkt 1.2.5    I
      I   6. zurück zu 1    I
      *---------------------*
```

Soweit zu den Beispielen. Natürlich brauchen sich Fenster nicht zu überlappen. Sie können auch neben- oder untereinander stehen. Dann dienen sie nur dazu, den Bildschirm übersichtlich aufzuteilen. Beispiel:



Das läßt sich natürlich auch ohne das Modul "Windows" so oder ähnlich erledigen. Aber warum sollten wir für jedes Programm das Rad neu erfinden? Jetzt müssen wir nur noch sagen: "Öffne ein Fenster an der Position (7,4). Es soll 5 Zeilen lang und 40 Zeichen breit sein. Als Titel soll über dem Fenster "A:Text.DOC" stehen." Das liest sich in Modula so:

```
...
VAR TextWindow: WindowHandle;
...
TextWindow := WdOpen (7,4,40,5);
WdSetTitle (TextWindow, "A:Text.DOC");
...
```

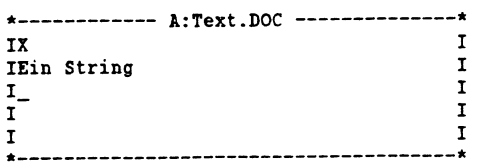
Wir können jetzt Text in dem Fenster ausgeben; dieser Text erscheint aber noch nicht auf dem Bildschirm, sondern wird intern gespeichert:

```
WdWriteC   (TextWindow, "X");          (* ein Zeichen *)
WdGotoXY   (TextWindow, 0,1);          (* Cursor in die nächste Zeile *)
WdWriteStr (TextWindow, "Ein String"); (* ein String *)
WdWriteC   (TextWindow, EoLn);         (* Cursor in die nächste Zeile *)
```

Wie Ihr seht, müssen wir immer den Namen des Windows ("TextWindow") mit angeben, wenn wir Text ausgeben oder den Cursor positionieren usw. Aber nur so kann das Modul wissen, auf welches Window (von den beliebig vielen Windows) wir uns beziehen.

Wenn wir jetzt das Fenster anzeigen lassen, erscheint es an der richtigen Position mit dem richtigen Inhalt:

```
WdTop (TextWindow);   (* es erscheint folgendes: *)
```



Evtl. liegen noch vorher geöffnete Windows darunter, daneben usw. Der "_" (Unterstrich) soll hier die Cursor-Position markieren; bei Eurem Rechner blinkt da vielleicht ein Block o.ä. Wie Ihr seht, ist die Innenfläche 40 Zeichen breit und 5 Zeilen hoch. Das ist nun unser Mini-Bildschirm für "A:Text.DOC".

Wenn nun ein anderes Window "nach vorne geholt" wird, verdeckt es evtl. teilweise dieses Window. Auf jeden Fall blinkt der Cursor dann in dem anderen Window. Das heißt aber nicht, daß wir nicht in "TextWindow" noch Text ausgeben könnten; der Cursor ist im "TextWindow" zwar nicht mehr sichtbar, aber er ist noch vorhanden, und nach

```
WdWriteStr (TextWindow, "Noch ein String");
```

würde diese Zeile an der Position angezeigt werden, wo der Cursor vorher stand. Diese Zeile wird auch auf dem Bildschirm angezeigt, wenn der Teil nicht von dem neuen obersten Window verdeckt wird.

Wenn wir das Window nicht mehr auf dem Bildschirm sehen wollen, geben wir

```
WdHide (TextWindow);
```

und schon verschwindet es. Wir können es aber jederzeit mit "WdTop" in alter Form und mit dem alten Inhalt wieder anzeigen lassen. Erst nach

```
WdClose (TextWindow);
```

gibt es unser "TextWindow" gar nicht mehr. Der ganze Inhalt ist jetzt auch verloren.

Wir hätten das Window auch zwischendurch mal woanders plazieren ("WdSetPos") oder seine Größe verändern ("WdSetSize") können. Der Inhalt wäre dabei erhalten geblieben (soweit möglich, wenn das Window kleiner wird).

Außerdem gibt es noch einige Prozeduren zur Text-Manipulation: "WdInsLine" fügt an der Cursorposition im Fenster (und nur dort!) eine Leerzeile ein, "WdDelLine" löscht eine Zeile. "WdScrollUp" bzw. "WdScrollDown" scrollen den ganzen Fensterinhalt eine Zeile hoch bzw. runter. "WdWriteStrLn" schreibt Strings (wie "WdWriteStr"), aber der Text darf länger als eine Zeile im Window sein; der Rest wird dann in die nächste(n) Zeile(n) geschrieben.

Außerdem unterstützt Windows auch noch einige Text-Attribute, wenn Euer Rechner das kann. Bei meinem Televideo-Terminal arbeite ich jetzt mit "HighLight" (Buchstaben leuchten heller), "Revers" (bzw. invers), "UnderLine" (unterstrichen) und "Flash" (die Zeichen blinken). Das kann alles ein-/ausgeschaltet werden, indem das Zeichen "On[Attribut]" bzw. "Off[Attribut]" ausgegeben wird. Entweder erledigt man das per "WdWriteC" oder Ihr baut die Zeichen in den auszugebenden String ein:

```
WdWriteC (TextWindow, On[Flash]);
WdWriteStr (TextWindow, "Tatü-tata, die Post ist da!");
WdWriteC (TextWindow, Off[Flash]);
```

und schon steht der Alptraum jedes echten Hackers blinkenderweise im "TextWindow", das hoffentlich gerade sichtbar ist. Ansonsten ist das selbstgebaute Modem futsch.

Das Windows-Paket (mit Dokumentation, Hilfs- und Test-Modulen) könnt Ihr bei Rüdiger (CP/M-Diskothek) bekommen. Die neuesten Infos gibt's natürlich bei mir, aber ich werde Rüdiger immer auf dem laufenden halten. Für die Italiener bietet sich noch Alexander als Bezugsadresse an. Bei der (kinderleichten) Anpassung helfe ich gerne; die Größe des zusätzlichen Programmcodes kann von 3 bis 14 Kb reichen (je nach gewünschter Geschwindigkeit).

Gerald

Der flüsternde Portable

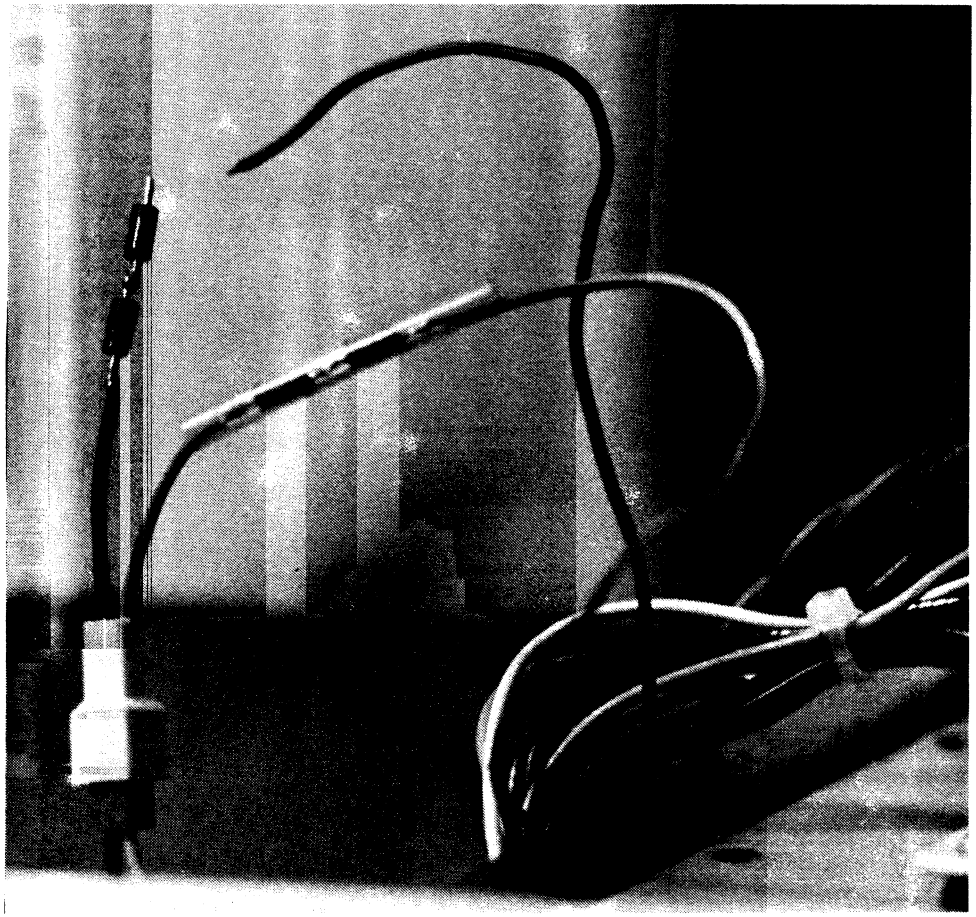
****

Dank Hartmut Obermanns ausführlicher Anleitung im Info Nr. 26 war der Einbau spurumschaltbarer Diskettenlaufwerke in meinen tragbaren 4p-Computer kein Problem. Die Mitsubishi-Laufwerke haben ihre Befestigungslöcher an anderer Stelle als die original montierten Tandon-Laufwerke, so daß man den Bohrer ansetzen muß. Wenn aber die mechanische Arbeit getan ist, fällt einem schon beim ersten Probelauf das dumpf schnurrende Positioniergeräusch als angenehmer Gegensatz zum lautstarken "Sägen" der Tandon-Drives auf.

Ein Blick ins Datenblatt der Mitsubishi-Laufwerke zeigt, daß sie weniger Strom brauchen als ihre Vorgänger. Dieser Vorteil erlaubt, den 4p-Computer mit weniger Luftdurchsatz und damit noch leiser zu betreiben. Den radikalsten Schritt, den Lüfter ganz abklemmen, wollte ich nicht tun. Nach halbstündigen Hin- und Herkopieren konnte man nämlich bei ausgeschaltetem Gebläse eine deutliche Erwärmung der rechten Gehäusehälfte oberhalb der Diskettenlaufwerke feststellen. Aber zwischen voller Leistung und Null gibt es ja noch einen Mittelweg. Versuche ergaben, daß schon eine geringe Verminderung der Spannung den Lüfter viel langsamer laufen läßt. Mit etwa 2,5 Volt weniger Betriebsspannung wird das singende Lüftergeräusch deutlich gedämpft und die Umwälzung staubhaltiger Luft verringert.

Die einfachste Art, Gleichspannungen zu "vernichten", ist entweder das Vorschalten einer Zenerdiode, oder bei kleinen Spannungen das Vorschalten normaler Dioden in Flußrichtung. Wie das Foto zeigt, habe ich das Zuleitungskabel zum Lüfter aufgetrennt und in beide Adern je zwei Dioden 1N4003 eingelötet. Beim roten (hellen) Kabel ist die Verbindung fertig. Der durchsichtige Schrumpfschlauch wurde schon übergezogen, aber noch nicht erhitzt. Am schwarzen Kabel kann man sehen, wie die Diodenkette verlötet wird.

Heinrich Betz



Umlaute auf der Tastatur des Genie I

Bekanntlich hat die Tastatur keine Umlaute und kein ß. Wenn man jedoch das Gehäuse öffnet, findet man über der Zahlenreihe noch 10 freie Tastenplätze, von denen 8 angeschlossen sind.
Von links nach rechts:

1. nicht angeschlossen
2. Abpfeil
3. Ä oder Hochpfeil
4. ö mit Shift ö
5. Ü mit Shift ü
6. - mit Shift ß
7. Chr 95 mit Shift Chr 127
8. Linkspfeil
9. Rechtspfeil
10. nicht angeschlossen

Sicher ist das bekannt. Blöd ist nur, daß die Umlaute in Großschrift und mit SHIFT in Kleinschrift kommen. Außerdem kommt ä überhaupt nicht. Deshalb schrieb ich einen Treiber, der die Funktion dieser Tasten umdreht und das ä erzeugt.

Kaum war ich damit zufrieden, stellte sich mir die Frage: Was soll ich mit den anderen 3 Pfeiltasten? Die Chr 95 und 127 waren auch recht nutzlos. Das Studium der Tastatur-Matrix ergab, daß die Adresse 3880H nur mit der Shift-Taste belegt war und somit noch eine Abfragemöglichkeit für 7 weitere Tasten zur Verfügung stand. So erweiterte ich meinen Keyboard-Treiber um die Abfrage dieser Adresse. Was ich mit den Tasten gemacht habe, weiß ich heute nicht mehr, denn kurz danach bekam ich eine Tastatur mit deutscher Tastenbelegung und 23 Funktionstasten, mehr als die Adresse 3880H hergab. Aber wozu gibt es Dioden? Ich holte also den Lötkolben hervor und heraus kam der Plan, der hoffentlich am Schluß dieses Artikels abgedruckt ist. Jetzt habe ich mehr Abfragemöglichkeiten als Tasten, denn der Plan ist noch längst nicht ausgereizt.

Wer den Plan studiert, soll sich nicht an den Namen der zusätzlichen Tasten stören. Meine Tasten sind eben so beschriftet.

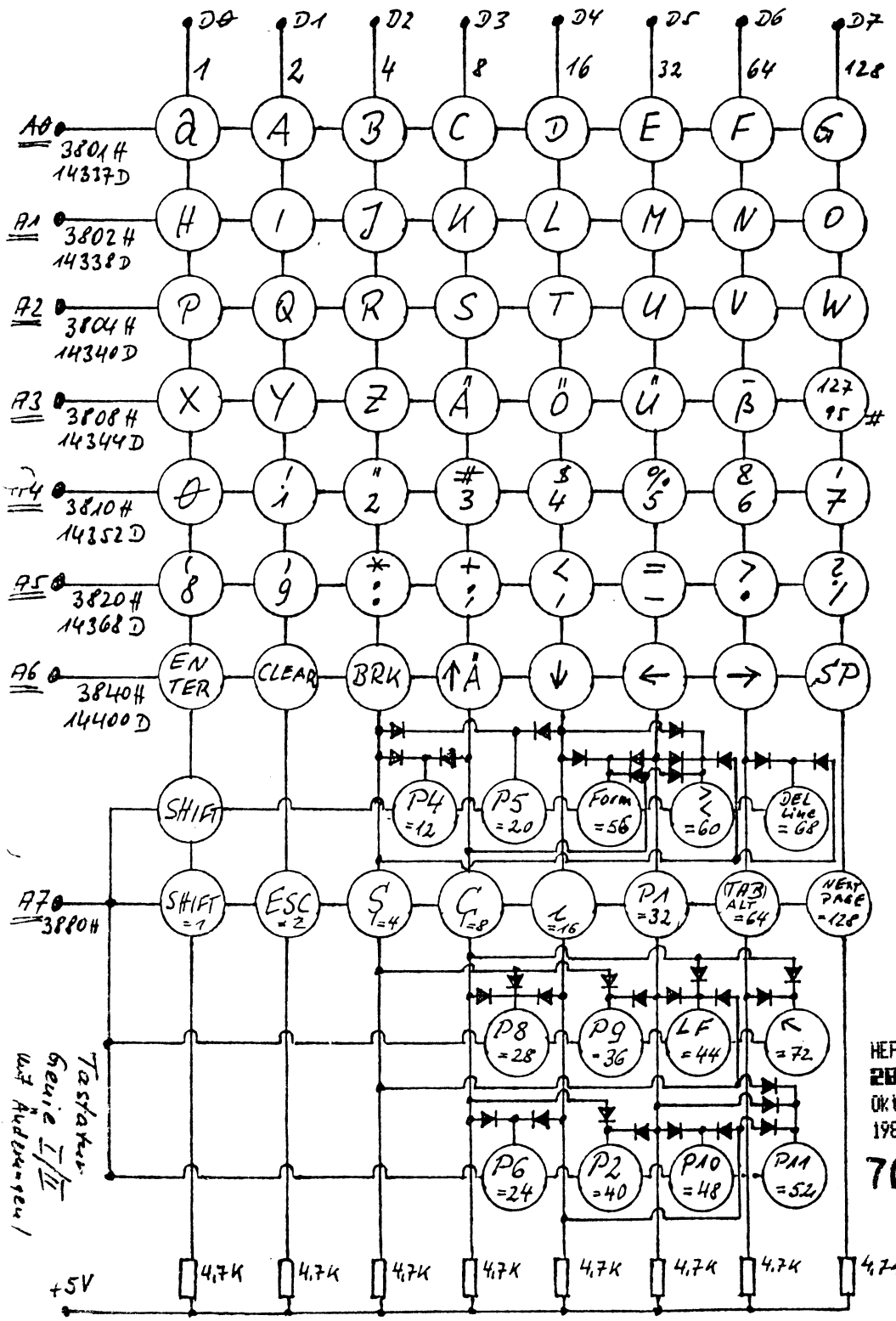
Die Abfrageart der Sondertasten ergibt 4-fache Tastenbelegung:

SHIFT addiert 1 zum abgefragten Tasten-Code.
ESC addiert 2 zum abgefragten Tasten-Code.
Das ist natürlich kein echtes Escape, die Taste ist so beschriftet.

Als Beispiel die Taste mit dem komischen S in meinem Plan (das ist wie die nächsten 2 ein türkisches Zeichen):
Tastendruck = 4
SHIFT+Taste = 5
ESC +Taste = 6
SHIFT+ESC+Taste=7

Belegt habe ich die Sondertasten bisher mit 3 türkischen Zeichen, Open, Close und CLS für die HPG-1B und einigen Druckersteuerungen wie Initialisierung, TOF, LF, FF, Elite, Pica, double strike, Breitschrift, linker Rand 4 und 6. Mit der TAB-Taste (inwischen umbenannt in ALT) und einer 'normalen' Taste werden die Grafikzeichen ausgegeben.

Auf der Diskette, die ich dem Club zur Verfügung stelle, kann man alles in MaschUnt/SRC nachlesen. (geschrieben mit Zeus)

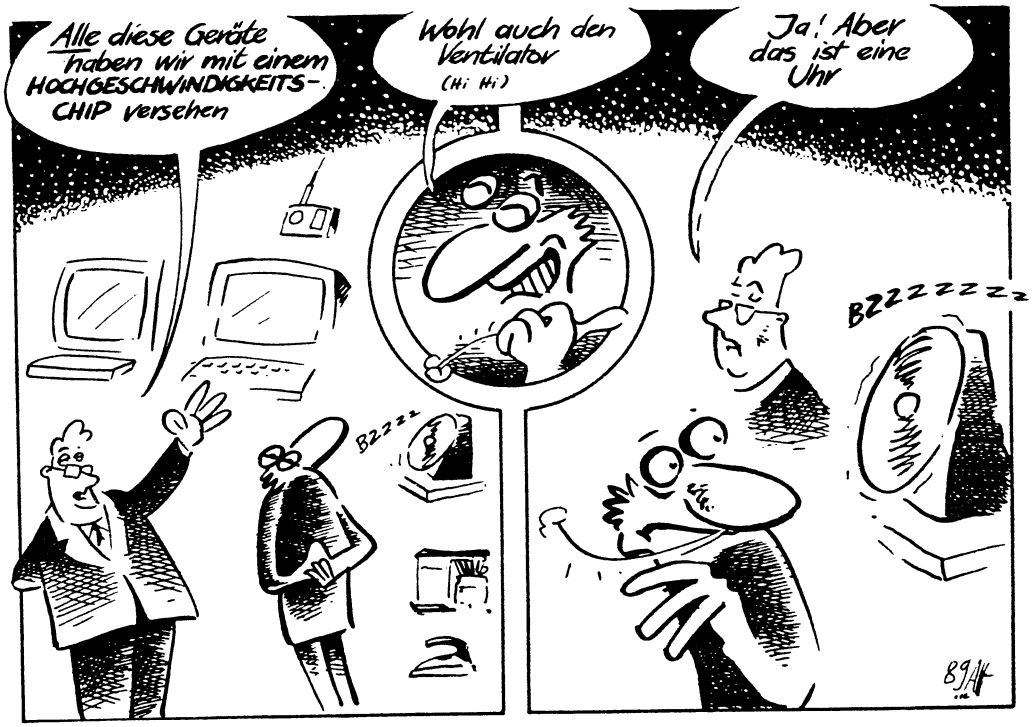


Ein klebriges Problem

71

Lange Zeit haben mir meine Mitsubishi-Laufwerke jetzt gute Dienste geleistet, aber so allmählich hat die Kiste immer schlechter gebootet und besonders auf einem Laufwerk immer wieder Schreib- und Lesefehler gebracht. Am Ende war es so schlimm, daß ich den Kopf erst etliche Male per Diskmonitor von einem Ende zum anderen fahren mußte, bevor er mit einem deutlich hörbaren Klack aufgesetzt und der Rechner gebootet hat. Nachdem ich das Laufwerk dann ausgebaut und zerlegt hatte, stellte sich schließlich ein Gummi der Marke 'Klebrig spezial' als übeltäter heraus. Die Idee, auf die Anschläge des Kopflademagneten einen dünnen Gummi zu kleben ist sicher im Interesse der Geräuschdämpfung, aber im Laufe der Zeit hat sich der, an den der Anker in der Ruhestellung drückt, mehr oder weniger in Wohlgefallen aufgelöst und einen sehr klebrigen Rückstand hinterlassen. Da konnte der Magnet ziehen, wie er will, erst durch die Vibrationen beim Fahren des Kopfes hat sich der Kleister für einige Zeit gelöst; bis zum nächsten längeren Stillstand. Ein Scheibchen Pappe und ein Tropfen Sekundenkleber haben dem dann sehr schnell ein Ende gemacht und seitdem laufen sie wieder einwandfrei.
Es ist gar nicht so schlimm, ein Laufwerk auseinander zu nehmen. Solange man den Köpfen nicht zu nahe kommt, kann eigentlich gar nichts passieren. Natürlich sollte man die leicht erreichbaren Potis nicht zu Reinigungszwecken auch verdrehen, aber mit etwas Vorsicht ist das kein Problem. Die nach dem Zusammenbau übriggebliebenen Schrauben und Teile sind eine gute Ergänzung für den häuslichen Werkzeugkasten.

Alexander Schmid



Star NL10 abwechselnd am CP/M-Rechner und am PC

Helmut Bernhardt

72

Eigentlich sollte es dem Drucker doch egal sein, ob jenseits des Centronics-Eingangs ein CP/M-Rechner oder ein Big Blue Nachbau Ursache für zu beschmutzendes Papier ist. Das ist es ihm auch und dadurch hat man die Schwierigkeiten mit den deutschen Umlauten, wenn der Drucker nicht der gleichen Kaste angehört wie der Computer. Der Drucker geht davon aus, daß es den ASCII-Code gibt und der für ihn genauso verbindlich wie für alle anderen ist. Schließlich ist der ASCII-Code ja ein allgemeiner Standard.

Das war vor einigen Jahren mal ein einheitlicher Standard. Damals genügten 7 Bit, um damit 128 Zeichen zu definieren. Mit 128 Zeichen waren alle amerikanischen Zeichen und eine Reihe Steuerzeichen zu codieren. Um der Vielfalt der nationalen Sonderzeichen in der übrigen Welt Rechnung zu tragen, hatten Drucker ein Mäuseklavier, auf dem man beispielsweise einstellen konnte, daß anstelle der amerikanischen eckigen und geschweiften Klammern unter den gleichen Codes (5BH-5FH, 7BH-7FH) dann die entsprechenden deutschen Umlaute zu Papier gebracht werden. Andere Einstellungen machten den Drucker auch für Engländer, Franzosen, Spanier und noch manch anderes Völkchen nutzbar.

Bei der Einführung des PC hat der Urheber desselben einen neuen 8 Bit ASCII-Code zum Standard erhoben. Der IBM Zeichensatz Nr 2 wird heute von allen Druckern und natürlich auch von allen PC-Videokarten unterstützt. Diesem Standard haben sich auch andere Computer (z.B. Atari ST) angeschlossen. Der 8 Bit ASCII-Code besteht zur Hälfte (Codes 00H - 7FH) aus dem alten 7 Bit ASCII-Code. Sämtlichen von IBM als berücksichtigenswert erachteten nationalen Sonderzeichen wurden Codes mit gesetztem Bit 7 zugeteilt.

Da Big Blue aber auch so manchen Unsinn wie die Spielkarten-Farben, eine Teilmenge der TRS80-Grafik-Bauklötzchen und eine Menge von Linien-Grafikzeichen (für die Darstellung der furchtbar wichtigen Rahmen bei Pulldown- und Popup-Menüs) als wichtig erachtet hat, sind natürlich etliche Zeichen unter den Tisch gefallen. Anstelle des deutschen "Eßzett" muß man halt ein griechisches "Beta" nehmen.

Mein jüngst erworbener Drucker Star NL10 hat das Glück, in einer Zeit herausgekommen zu sein, als nicht nur PCs sondern auch noch CP/M-Computer eine Rolle spielten. Entsprechend war Star so umsichtig, den Drucker mit verschiedenen Betriebsprogrammen und Zeichensätzen (im 27256-EPROM) herauszubringen.

Wenn man nun einen NL10 mit PC-kompatiblem Betriebsprogramm-EPROM hat und jemanden kennt, der den gleichen Drucker mit dem CP/M-kompatiblen Betriebsprogramm hat, und jemanden kennt, der 27256-EPROMs lesen und 27512-EPROMs brennen kann (falls man das nicht selbst kann), dann kann man sich die Inhalte der beiden 27256-EPROMs in ein 27512-EPROM füllen (lassen) und über A15 des EPROMs (Pin 1) durch Anlegen von +5V oder GND das jeweilige Betriebsprogramm für den jeweiligen Rechner einstellen. Das Umschalten darf natürlich nicht bei eingeschaltetem Drucker erfolgen. Wegen des sonst beim Absturz vom Drucker erzeugten Lärms hat man sich aber schnell an diese Maßnahme gewöhnt.

Eleganter Adreßdecoder für GENIE 1

Helmut Bernhardt

Als ich bei der Fehlersuche in meinem GENIE 1 durch die zahlreichen selbstgeschnitzten Erweiterungen und das damit verbundene Strippengewirr selbst nicht mehr durchblickte, keimte der Gedanke, das alles sehr viel übersichtlicher und völlig neu zu gestalten. Als Ergebnis kam ein kleines Board heraus, das in den Sockel des Z80 zu stecken ist und den Z80 selbst mit aufnimmt. Zusätzlich wird auf dem Board ein 27128-EPROM (das anstelle der dann überflüssigen Original-ROMs das Level-II-Basic enthält), ein 74LS273-Latch, ein PAL16L8 (das ein Freigabesignal für das PAL16L8 und einige zusätzliche Freigabesignale erzeugt) und ein Adreßdecoder-PAL20L8 (das die Freigabesignale für alle memory mapped Systemkomponenten erzeugt) untergebracht.

Die Ausgänge des Latch haben folgende System-Steuerfunktionen:

Q0 = 0 : 0000H-2FFFH ist beim Lesen ROM und beim Schreiben RAM
= 1 : 0000H-2FFFH ist beim Lesen und Schreiben RAM

Q1 = 0 : 37E0H-37EFH ist Floppy, Drucker, INT-Status, Drive-Select
= 1 : 37E0H-37EFH ist RAM; Floppy u.s.w. sind über die Ports E0H-EFH erreichbar

Q2 = 0 : 3800H-38FFH ist beim Lesen die Tastatur und beim Schreiben RAM
= 1 : 3800H-38FFH ist beim Lesen und Schreiben RAM

Q3 = 0 : 3C00H-3FFFH ist Video-RAM
= 1 : 3C00H-3FFFH ist RAM

Q4 = 0 : die memory mapped Systemkomponenten liegen im Bereich 0000H-3FFFH
= 1 : die memory mapped Systemkomponenten liegen im Bereich C000H-FFFFH

Unabhängig vom Zustand von Q0-Q4 ist dauerhaft bei 3000H-37DFH, 37F0H-37FFH und 3900H-3BFFH RAM eingeblendet. Anstelle eines Sonder-ROM war mir RAM bei 3000H wichtiger.

Das Board erzeugt die Freigabe-Signale /ROM, /RAM, /FLO, /KB und /VID und das Steuersignal /MRD für die Lesetreiber des RAMs. Diese Signale sind an die entsprechenden Punkte auf dem CPU-Board zu führen; die ursprünglichen Signale sind zu durchtrennen.

Frei zum CPU-Board und zum RB-EXP1 zu verdrahtende Signale

Signal	Pin des 20L8	zu verbinden mit	zu durchtrennende Leitung	
/RAM	22	Z37,74LS367,Pin15	von Z36,74LS04,Pin2	#1
/MRD	17	Z9,74LS367,Pin1	von Z21,74LS20,Pin8	#2
/KB	20	Z7,74LS368,Pin1	von Z35,74LS32,Pin3	
/VID	19	Interboard-Conn. Pin10	von Z37,74LS32,Pin11	
/FLO	18	RB-EXP1, Z18, 74LS15,Pins2,14	RB-EXP1, Z14, 74LS32,Pin11	#3

#1 bei Aufrüstung auf 64K (256K) RAM besteht diese Verbindung als freie Verdrahtung; dieser Draht wird entfernt
#2 bei Entfernen von Z21,74LS20 entfällt das Durchtrennen
#3 Z13,74LS30 auf dem EXP1 kann entfernt werden
anstelle von /MWR von Z15,74LS32,Pin3 auf dem CPU-Board muß /BWR von Z16,74LS367,Pin11 an das EXP1 geführt werden, damit der Betrieb über die Ports E0H-EFH möglich ist

Das Freigabesignal /ROM ist auf dem Huckepack-Board direkt an die Pins 20 und 22 des EPROMs geführt.

Die Erzeugung dieser Freigabesignale ist aus dem Assembler-Listing für das PAL20L8 zu ersehen. Die Adreßeingänge A14 und A15 dieses PALs sind nicht die Adressen A14 und A15 der CPU. Diese Signale werden aus 2 XOR-Gattern (74LS86) aus den Adressen A14 und A15 der CPU erzeugt. Mit Q4 des Latch läßt sich damit steuern, ob A14 und A15 invertiert oder direkt an das PAL gelangen.

PAL-Assembler-Listing für das Adreßdecoder PAL20L8

```
PAL20L8
GENIE 1 Adreßdecoder mit mm-I/O-Switching
                    04.09.88  H. Bernhardt

A13 A11 MERQ RD A8 A9 A10 Q3 A15 Q1 PEX GND
ADR A14 MRD Q0 ROM FLO VID KB Q2 RAM A12 VCC

/ROM  = /Q0 */MERQ */RD */A15 */A14 */A13
      + /Q0 */MERQ */RD */A15 */A14 * A13 */A12
/FLO  = /Q1 */MERQ */A15 */A14 *A13 *A12 */A11 *A10 *A9 *A8 */ADR
      +  Q1 */PEX
/KB   = /Q2 */MERQ */RD */A15 */A14 *A13 *A12 *A11 */A10 */A9 */A8
/VID  = /Q3 */MERQ */A15 */A14 *A13 *A12 *A11 *A10
/RAM  = /MERQ *ROM *FLO *KB *VID
/MRD  = /MERQ */RD *ROM *FLO *KB *VID
```

Um bei den zur Verfügung stehenden Pins des PAL20L8 auch das Signal /FLO voll ausdecodieren zu können (A4-A15), mußte eine Gruppe von Adressen (A4-A7) schon zusammengefaßt (/ADR) und an einen Pin des PALs geführt werden. Auch für das alternativ von einer Portadresse herzuleitende /FLO war ein vorheriges Ausdecodieren von /IORQ, A4-A7 nötig (/PEX). Diese Signale werden vom PAL16L8 erzeugt, an das für das Erzeugen von /OUTFC (Freigabe des Steuer-Latch) ohnehin schon die Adressen A0-A7 und /IORQ gelangen.

PAL-Assembler-Listing für das I/O-Decoder-PAL16L8

```
PAL16L8
GENIE 1 Portdecoder
                    04.09.88  H.Bernhardt

A7 A6 A5 A4 A3 A2 A1 A0 WR GND
RD OFC IOF0 IOF4 IORQ IFC OFB PEX ADR VCC

/OFC  = /IORQ */WR *A7 *A6 *A5 *A4 *A3 *A2 */A1 */A0   ;Pin12
/IFC  = /IORQ */RD *A7 *A6 *A5 *A4 *A3 *A2 */A1 */A0   ;Pin16
/OFB  = /IORQ */WR *A7 *A6 *A5 *A4 *A3 */A2 *A1 *A0    ;Pin17
/IOF0 = /IORQ *A7 *A6 *A5 *A4 */A3 */A2                ;Pin13
/IOF4 = /IORQ *A7 *A6 *A5 *A4 */A3 *A2                 ;Pin14
/PEX  = /IORQ *A7 *A6 *A5 */A4                         ;Pin18
/ADR  = A7 *A6 *A5 */A4                                ;Pin19
```

Da das PAL16L8 noch einige freie Ausgangs-Pins hatte, wurden noch einige weitere Port-Freigabe-Signale decodiert.

Funktion der Port-Freigabesignale des PAL16L8

/OFC (/OUT FCH) wird intern zur Freigabe des Sytem-Steuerlatch benutzt
/IFC (/IN FCH) steht für die Freigabe eines über Port FCH zu lesenden Treibers zur Verfügung

/OFB (/OUT FBH) Freigabe-Signal, um über Port FBH ein Latch zu beschreiben; kann für die Freigabe des 74LS273 (Pin11) auf dem 256K-Banker benutzt werden (A0-A5 brauchen dann nicht mehr zum Banker geführt zu werden, A6 und A7 sind für das Erzeugen einer 8Bit-Refreshadresse dort aber weiterhin nötig; das 74LS30 auf dem Banker kann entfallen)

/IOF0 und /IOF4 (/In/OUT F0H-F3H und /IN/OUT F4H-F7H) Freigabe-Signale für z.B. Z80-Peripherie-Bausteine mit 4-Portadressen (PIO, SIO, CTC, DMAC, PPI8255...)

Die Verwendung eines 27128-EPROMs (wovon es auch 150ns-Versionen gibt) erlaubt bedeutend höhere Taktraten als die ursprünglichen ROMs. Wenn dann auch noch das Timing der Signale /RAS, /MUX und /CAS anders gelöst wird, spielen auch die RAMs mit. 5,3MHz sind auch ohne große Eingriffe möglich.

Da das ROM nun innerhalb der Treiber der CPU-Signale liegt, muß zur Steuerung der Daten-Lesetreiber auch das Signal /ROM herangezogen werden. Diese Treiber dürfen nicht freigegeben werden, wenn das ROM gelesen wird.

Um dies zu erreichen, wird Pin9 von IC1,74LS14 durchgekniffen und hochgebogen. Auf IC1 wird ein 74LS08 mit den Pins 7 und 14 huckepack aufgelötet. Alle anderen Pins des 74LS08 werden hochgabogen. Der abgekniffenen und hochgebogene Pin9 des IC1 wird mit dem seitlich gebogenen Pin8 des 74LS08 verlötet. Pin10 des 74LS08 wird mit Pin22 des 27128-EPROMs und Pin9 des 74LS08 mit Pin3 von IC14,74LS00 verbunden (Abbildung).

Zur Freigabe der Speicherlese-Treiber erzeugt das PAL20L8 das Signal /MRD, das an Pin15 von IC9,74LS367 zu legen ist. Durch Entfernen von IC21,74LS20 wird das bisherige Signal entfernt.

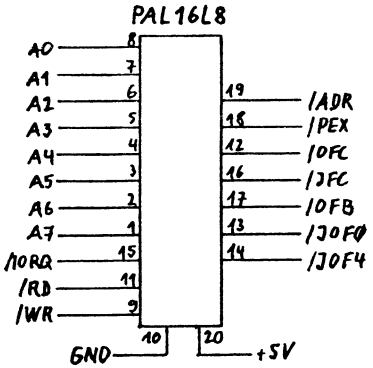
Außer IC21 sind auch IC22,74LS156 und IC25,74LS139 überflüssig geworden. Diese ICs können ebenfalls entfernt werden. Und natürlich können auch die alten ROMs rausgeschmissen werden.

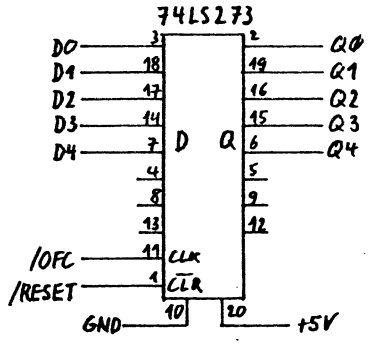
Beim Aufbau des Huckepack-Boards sind folgende Punkte zu beachten:

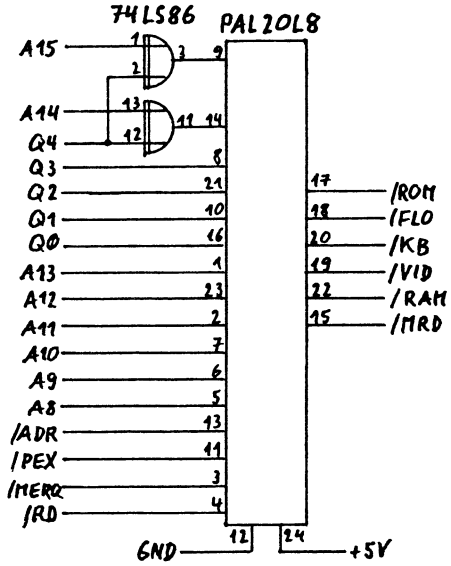
Vor dem Einlöten irgendwelcher Bauteile sind zunächst sämtliche Drahtbrücken zu legen. Als erstes sind dann der Sockel für den Z80 und der Stecker für das Anflanschen des Boards in den Z80-Sockel auf dem CPU--Board in geeigneter Reihenfolge einzulöten. Der Stecker wird auf der Lötseite und der Sockel auf der Bestückungsseite eingebaut. Dabei erschwert das zuerst eingebaute Teil den Einbau des anderen. Was zuerst angelötet wird, sollte eine gründliche Besichtigung der Teile ergeben.

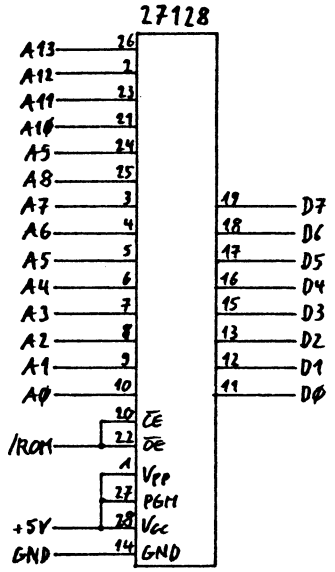
Es sollten alle ICs gesockelt werden. Sowohl beim EPROM als auch bei den PALs kann irgendwann der Wunsch nach geändertem Inhalt aufkommen. Und für Testzwecke bei der Fehlersuche ist es auch bei Standard-ICs zweckmäßig, wenn man diese auswechseln kann oder einfach nur einzelne Pins aus der Fassung biegen kann.

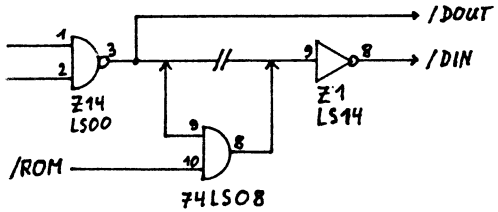
Ungebohrte Boards, mit Level-II befüllte 27128-EPROMs und gebrannte PALs kann ich zum Selbstkostenpreis zur Verfügung stellen.

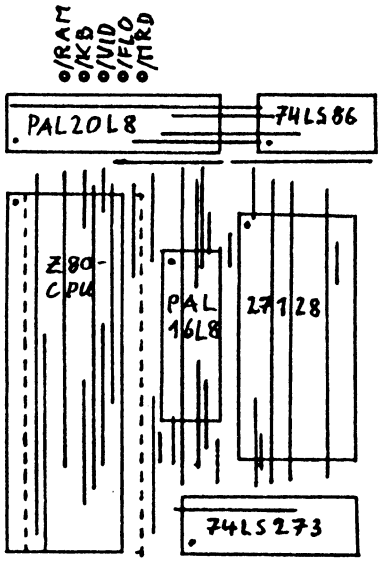


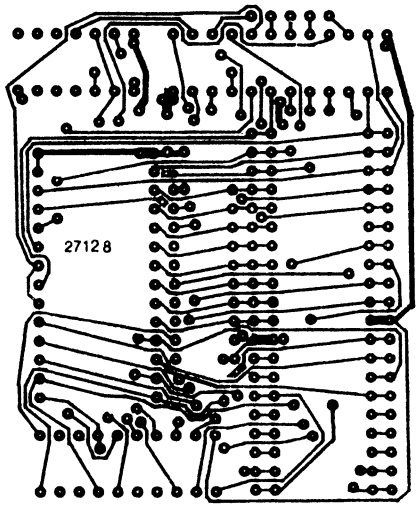


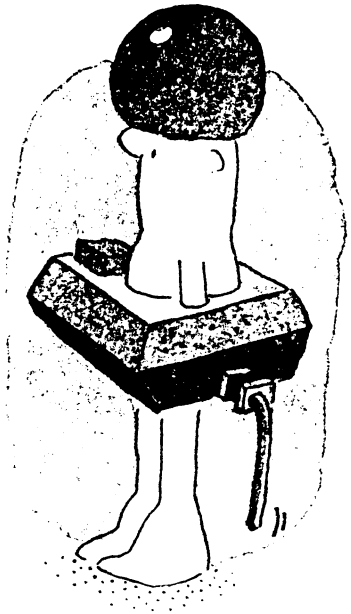












Es lebe der Standard !

Hercules Graphics Adaptor meets Trash 80

Helmut Bernhardt

Genau besehen hat es auch Vorteile, daß der Industrie Standard einen so niedrigen Standard hat. Insbesondere die Herkules-Karte, die dort schon nicht unbedingt als schlechteste Video-Karte gilt, ist schon fast zum TRS-80 kompatibel. Na ja, nicht ganz zu dem, aber doch schon zum GenieIIIs-Video-Interface. Wie im GIIIs gibt es dort eine 2K große Text-Video-Seite (80x25 Zeichen) und ebenso memory mapped. Zusätzlich bietet die Hercules aber auch noch einen 2K großen Attribut--Speicher, womit sich für jedes Zeichen auf dem Bildschirm einzeln Attribute setzen lassen. Und schließlich wird auf der Hercules wie im Genie IIIs mit 2 HRG-Pages mit je 32K Größe (ebenfalls bei beiden memory mapped) Grafik dargestellt. Bei beiden Geräten wird alles von einem CRTC6845 gesteuert.

Die maßgeblichste Inkompatibilität ist die Tatsache, daß bei der Hercules Text- und Attribut-Speicher ineinander verschoben sind. Dort werden innerhalb eines linearen 4K-Bereichs alle geradzahligen Adressen für Textzeichen und alle ungeradzahligen Adressen für die dazugehorigen Attribute verwendet. Weniger störend sind die 10Bit-Portadressen des Industrie-Standards, mit denen die internen Register der Karte angesprochen werden. A8 und A9 kann man der Karte bei Z80--Portzugriffen einfach vorgaukeln.

Folgende Tabelle stellt die Ähnlichkeiten und Abweichungen zwischen Hercules und GenieIIIs-Video-Interface zusammen:

Funktion	Hercules	GenieIIIs
Text-Speicher	B0000-B0FFF (mit A0=0)	3800(3C00)-3FFF
Attribut-Speicher	B0000-B0FFF (mit A0=1)	nicht vorhanden
HRG-Seite 0	B0000-B7FFF	8000-FFFF
HRG-Seite 1	B8000-BFFFF	8000-FFFF
CRTC6845-Ports	3B4H/3B5H	F6H/F7H

Diese Ähnlichkeit ließ mich vor längerer Zeit schon mal darüber brüten, wie man die Hercules anstelle des dort recht lausigen Video--Interface in TRS-80- und Genie-Rechnern einsetzen könnte. Zunächst dachte ich an einen Adapter, der mir die nötige Adreßübersetzung macht und die Verbindung zwischen Genie-CPU-Board und Hercules herstellt. Dann dachte ich daran, daß sich dieser Aufwand für das ebenfalls recht lausige Genie-CPU-Board nicht lohnt. Inzwischen ist um diesen reinen Adapter ein ganzes CPU-Board gewachsen, das noch einiges mehr bietet als das Genie-CPU-Board und dabei auf die Maße einer Europa-Karte geschrumpft ist. Das Board enthält all das, was mir bislang lieb geworden ist:

[1] HD64180-CPU
[2] 2 serielle Schnittstellen
[3] 1MB RAM bestehend aus 8 mal 511000-Megachips
[4] auf 1MB erweiterte MMU des HD64180
[5] 27128-EPROM mit Level II BASIC und Initialisierungen

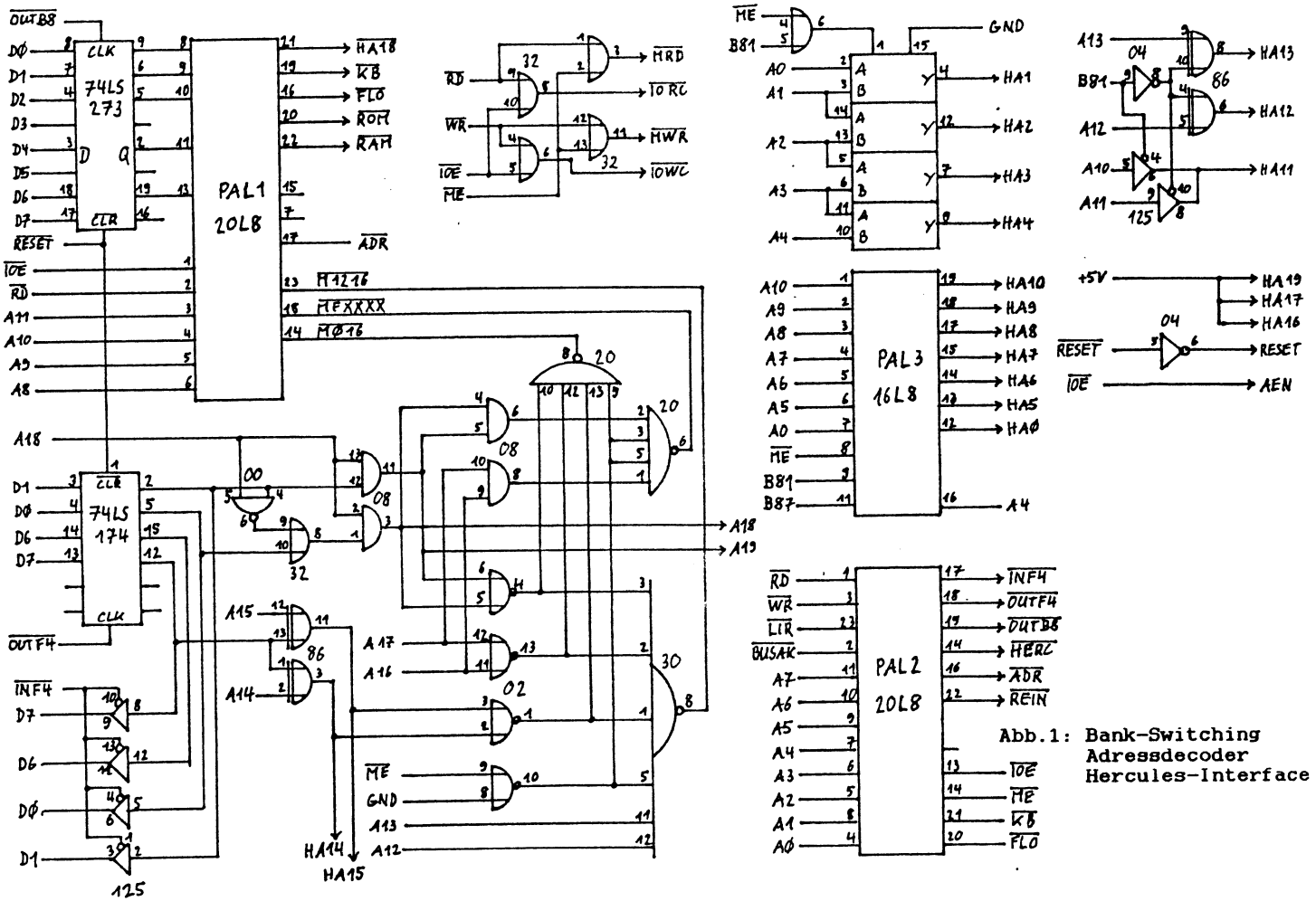


Abb.1: Bank-Switching
Adressdecoder
Hercules-Interface

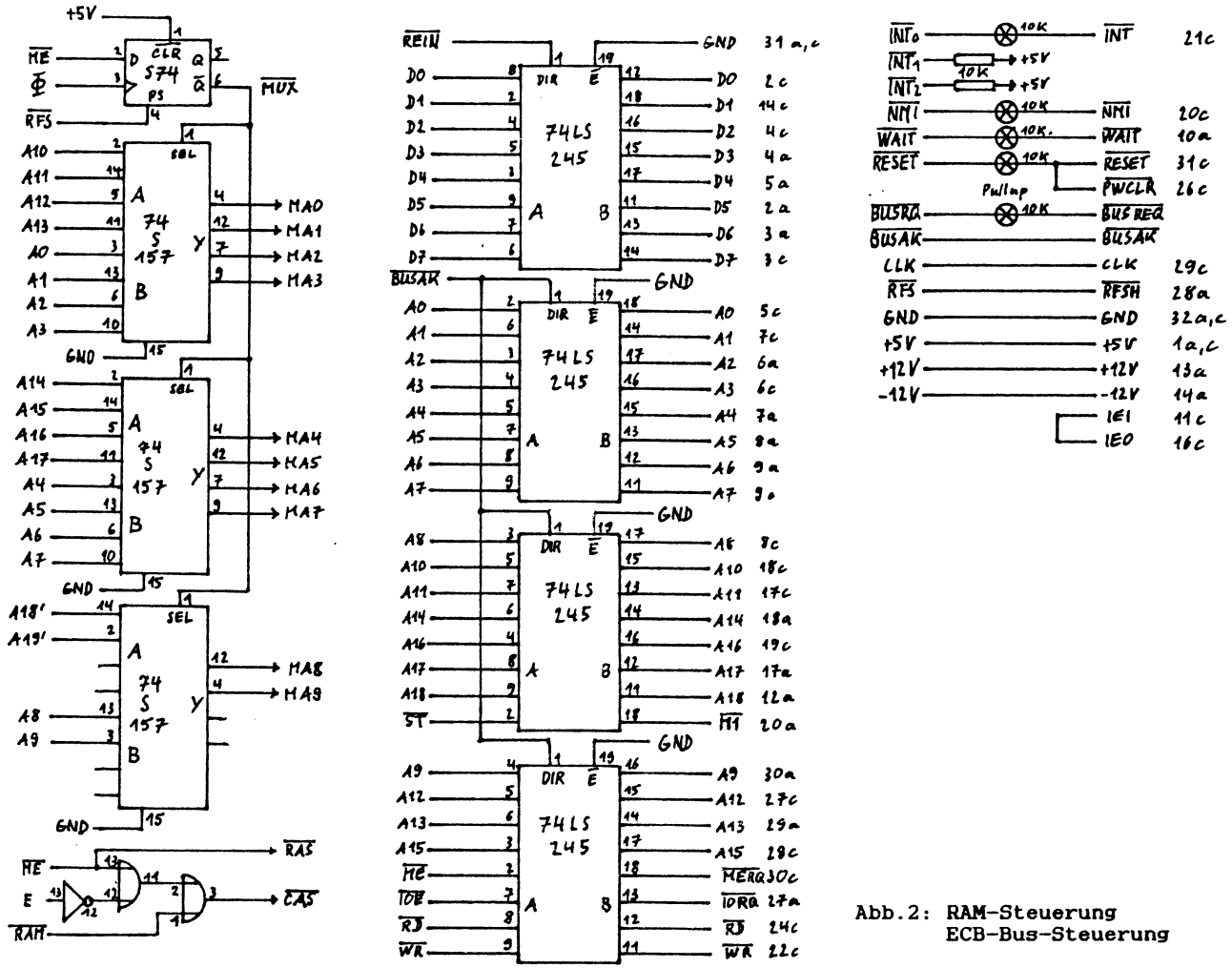


Abb.2: RAM-Steuerung
ECB-Bus-Steuerung

[6] ECB-Bus mit Buspufferung und IM2- und DMA-fähiger Steuerung
[7] Interface zur Hercules
[8] komfortables Switching für memory mapped I/O

Damit steht mir ein voll Geniel-kompatibler (nur Spiele lassen sich ohne Bauklötzchengrafik nicht so richtig spielen) Computer zur Verfügung. Die Hercules stellt mir dafür den Standard-64x16-Bildschirm, eine 80-Zeichen-Karte und zwei hochauflösende Grafikseiten mit je 720x348 Punkten zur Verfügung, und das alles für 100,-DM.

Die eigentlichen Kosten fallen bei der Beschaffung der RAMs an. Zugegeben - 256K hätten's beim Geniel auch getan, aber das Teil-Layout mit CPU, RAMs, EPROM, RAM-Steuerung und Treibern für die V24-Schnittstellen brauchte ich nur aus der Schublade zu ziehen und beim Rest habe ich mir dann auch keine große Mühe mehr gemacht. So ist

Der Aufbau der Karte

zu einem richtigen kleinen Abenteuer geworden. Das Layout beschränkt sich auf einseitige Leiterbahnführung und versucht, dort so viele Verbindungen unterzubringen, wie irgend möglich. Nur im hinteren Teil der Karte (der aus der Schublade), war es möglich, alles mit Drahtbrücken zu erledigen - bis auf das EPROM, das muß auch von Hand verdrahtet werden, dort sind nur D0-D7, +5V und GND angeschlossen. Was von Hand zu verdrahten ist, kann aus der Verdrahtungsliste ersehen werden.

Wer nun immer noch glaubt, daß er dieses Projekt durchstehen kann, und von mir zum Selbstkostenpreis ein geätztes, ungebohrtes Board bekommen hat, muß zunächst mal 1-2 Stunden mit der Bohrmaschine zubringen. Die 3 Stecker für VG64-Leiste, für die Pfostenfeld-Leiste zum Aufstecken der Hercules-Karte und die Pfostenfeld-Leiste für die seriellen Schnittstellen müssen 1mm-Bohrungen erhalten, alle anderen Bohrungen sind in 0,8mm niederzubringen.

Wenn man sich davon wieder erholt hat, legt man sämtliche Drahtbrücken (Drahtbrucken- und Bestückungs-Plan) und fängt dann erst an, Kleinteile und IC-Sockel einzulöten. Da die Drahtbrücken teilweise recht eng liegen, sollte grundsätzlich dünner Kupfer-Lackdraht verwendet werden. Bei Drahtbrücken, die +5V und GND führen sollen (solche Brücken enden meistens an etwas breiteren Leiterbahnen), ist etwas dickerer isolierter Schaltdraht vorzuziehen.

Es sei noch gesagt, daß im dichter besiedelten vorderen Bereich (beim Busstecker) in der ersten Reihe ein IC nicht zu bestücken ist. Weil der Platz halt da war, habe ich da vorsichtshalber mal die Lötaugen hinterlegt, damit es spätere Bastler bei Erweiterungen einfacher haben.

Auf der Busplatine oder frei improvisiert auf dem CPU-Board müssen die CPU- Signale /BUSRQ, /RESET, /WAIT, /NMI und /INT0 mit 10kOhm an +5V gelegt werden. /INT1 und /INT2 müssen direkt am HD64180 über 10k an +5V gelegt werden. Wenn diese Eingänge nicht genutzt werden, genügt ein gemeinsamer Widerstand.

Die 62polige Steckleiste zum Anschluß der Hercules ist für eine spezielle, besonders kleine Karte gedacht, die direkt neben dem für PC-Karten üblichen Platinenrand-Stecker zwei Reihen Durchkontaktierungen aufweist, die freundlicherweise im RM2,54 Rastermaß angelegt sind und eine Weite von 1mm haben. Wenn hier von der Bestückungsseite eine zweireihige Buchsenleiste eingelötet wird, und auf dem CPU-Board die Pfostenfeld-Leiste mit etwa um 1cm verlängerten Beinchen eingelötet wird, können beide Karten direkt aufeinander gesteckt werden.

Der Fernseher darf leider nicht der alte vom Geniel bleiben. Die Hercules benötigt einen TTL-Monitor (ohne Gehäuse und Netzteil bekommt man sowas im Elektronik-Ramsch-Versandhandel für ca. 40,-DM). Andererseits gibt es auch TTL/BAS-Wandler, denen man aber ein aus "Video" und "Intensity" zusammengeführtes Signal anstelle des reinen Video--Signals zuführen sollte, um nicht auf die Helligkeitssteuerung der Hercules verzichten zu müssen.

Programmierung der Hercules-Karte

Interne Ports (die Portadressen sind die auf die 8Bit-Decodierung durch das Interface reduzierten Adressen; im PC müssen zusätzlich A8 und A9 high sein)

B4H Adreßregister des CRTC-6845
B5H Datenregister des CRTC-6845
B8H Display-Betriebsart-Steuerkanal
B9H Display-Status-Kanal
BBH Light-Pen FlipFlop zurücksetzen
BCH Drucker-Datenport
BDH Drucker-Statusport
BEH Drucker-Steuerkanal
BFH Konfigurations-Schalter

Für die Programmierung des CRTC-6845 ist ein Datenblatt erforderlich. Durch Trial-and-Error Methoden soll man es schaffen können, den Monitor dauerhaft zu disablen. Es sei nur soviel gesagt, daß der CRTC eine ganze Menge Datenregister besitzt, wovon nur die ersten 16 relevant sind. Sie haben alle die Adresse B5H. Auf welches dieser Register jeweils über Port B5H zugegriffen werden kann, bestimmt der Eintrag in das Adreßregister B4H. Die Initialisierungsroutine im ROM enthält an der Adresse 3000H 5 Tabellen mit je 16 Daten für die ersten 16 Datenregister des CRTC und eine Routine, die dem CRTC diese Daten einverleibt, wenn HL auf den Anfang der jeweiligen Tabelle zeigt. Dort mag sich der Softi informieren, der lieber noch exotischere Bildschirm-Formate einstellen möchte. Die grundsätzliche Bedeutung der CRTC-Register ist aus mc 1/88 S.113 zu entnehmen.

Der Display-Betriebsart-Steuerkanal (Port B8H) der Hercules reagiert nur auf die Bits 1, 3, 5 und 7. Die geradzahligen Datenbits haben dort keine Funktion. Da die Einstellungen dieses Registers für die Adreßübersetzung des Interface zur Hercules von Bedeutung sind, wurde dieser Port auf dem CPU-Board nachgebildet und um die geradzahligen Bits bereichert. Setzen und Zurücksetzen einzelner Bits dieses Ports durch Auslesen, Bit verändern und Zurückschreiben ist also nicht moglich. Man muß dafür im RAM über den Zustand des Ports Buch führen.

In der folgenden Tabelle sind die Funktionen der einzelnen Bits des

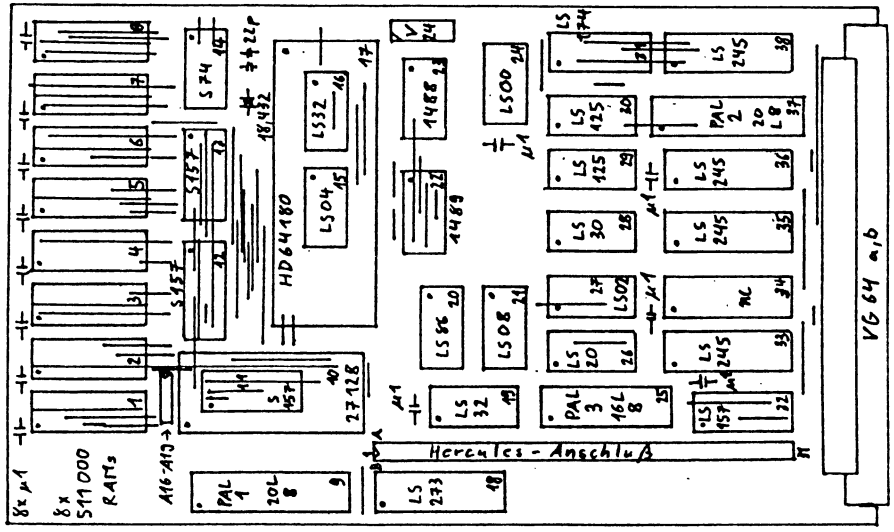
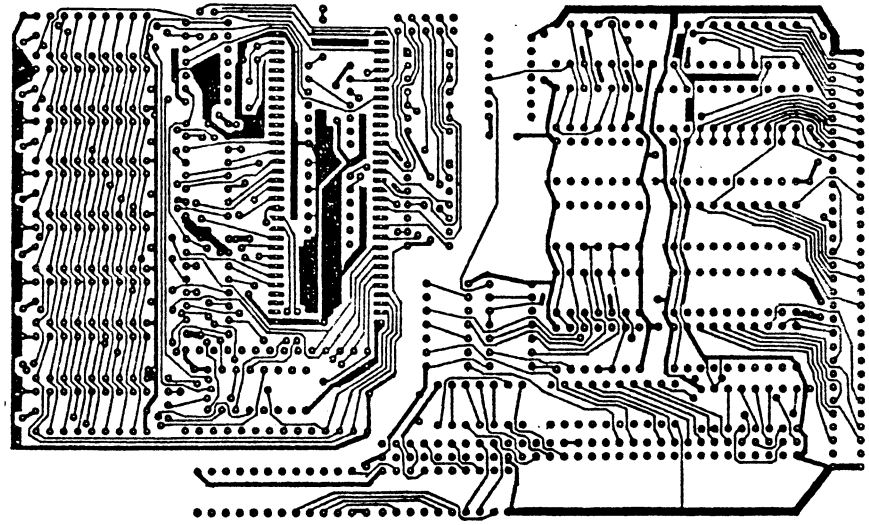


Abb.3: Bestückung- und Dratbrückenplan, Teil-Layout



erweiterten Ports B8H zusammengestellt. Die urgsprünglichen Funktionen des Ports auf der Hercules, die nach wie vor gelten, sind mit # markiert.

Display-Mode-Control-Port, B8H

```
D0 =0 : 1K Video-RAM bei 03C00-03FFF (64x16), Keyboard bei 03800-038FF
   =1 : 2K Video-RAM bei 03800-03FFF (80x25), Keyboard disabled
D1 =0 : Text-Video-Mode enabled, HRG disabled                          #
   =1 : HRG enabled, Text-Video disabled                               #
D2 =0 : ROM bei 00000-036FF (nur bei /RD, bei /WR RAM)
   =1 : ROM disabled, bei 00000-036FF liegt RAM
D3 =0 : die Hercules gibt kein Video-Signal an den Monitor             #
   =1 : das Video-Signal der Karte gelangt zum Monitor                 #
D4 =0 : Floppy, Drucker u.s.w. bei 037EX memory mapped
   =1 : 037EX ist RAM, Floppy u.s.w. über Ports EXH
D5 =0 : D7=1 des Attribut-Bytes bedeutet "heller Hintergrund"          #
   =1 : D7=1 des Attribut-Bytes bedeutet "Blinken"                     #
D6 =0 : Text- bzw HRG-RAM sind freigegeben
   =1 : Text- bzw HRG-RAM gesperrt, dort RAM eingblendet
D7 =0 : HRG : Page 0 selected                                          #
        Text: 2K Textspeicher eingestellt
   =1 : HRG : Page 1 selected                                          #
        Text: 2K Attributspeicher eingstellt
```

Die Einteilung des 1MB-Speichers

Der HD64180 (in der hier verwendeten DIL64-Form) kann nur 512K Speicher adressieren. Um den vollen 1MB-Speicher zu nutzen, muß durch zwei weitere Bits eines Ports (D0 und D1 von Port F4H) ein Banking von 256K-Blocks dem Banking der MMU des HD64180 überlagert werden. Die Selektion eines der 4 256K-Blocks erfolgt mit A18 des HD64180 und D0/D1 von Port F4H. Aus diesen 3 Signalen werden die Adressen A18 und A19 für das RAM hergeleitet.

Port F4H D1	D0	CPU A18	RAM A19	A18	256K--Block
x	x	0	0	0	0
0	0	1	0	1	1
0	1	1	0	1	1
1	0	1	1	0	2
1	1	1	1	1	3

Wenn A18 der CPU low ist, wird immer der unterste 256K-Block angewählt. Wenn A18 high ist, wird einer der 3 anderen Blocks selektiert. Welcher dieser 3 Blocks selektiert wird, hängt von den Zuständen von D0 und D1 des Ports F4H ab. Diese Vorgaben gelten sowohl für die MMU als auch für den DMA-Controller.

Nach einem RESET ist die MMU des HD64180 immer so eingestellt, daß der logische 64K-Z80-Adreßraum in den unteren 64K des unteren 256K-Blocks liegt. Um die Standard-Software (NEWDOS/80, CP/M) zu fahren, braucht die MMU nicht umprogrammiert zu werden. Die Decodierung von ROM und

memory mapped I/O schließt auch die Adressen A16-A19 mit ein, so daß diese Baugruppen im physikalischen Adreßraum 00000H-03FFFH liegen. Nur der HRG-Speicher der Hercules-Karte mit 2mal 32K wäre bei 08000H eingeblendet etwas lästig und wurde deshalb an das obere Speicherende bei F0000H verbannt. Für die MMU und den DMAC des HD64180 sind dies die Adressen 70000H-7FFFFH, wobei D0 und D1 von Port F4H gesetzt sein müssen, damit der HRG-Seicher adressiert wird.

Neben der Grundeinstellung (D7 an Port F4H low, Zustand nach RESET) gibt es noch eine andere Lage für die memory mapped Baugruppen. Wenn D7=1 an Port F4H ausgegeben wird, liegen ROM, Floppy, Drucker, Tastatur und Text-Video-RAM bei 0C000H-0FFFFH und die untersten 16K sind RAM.

Und schließlich läßt sich innerhalb dieser 16K (wo auch immer die memory mapped Baugruppen eingeblendet sind) sehr komfortabel umkonfigurieren. Darüber gibt die Tabelle der Bedeutung der Bits des erweiterten Ports B8H und die Map der Speicheraufteilung Auskunft. In allen Bereichen, wo die jeweilige memory mapped Baugruppe ausgeblendet ist, liegt dann automatisch RAM vor.

Der Text-Bildwiederholspeicher

Die Hercules-Karte stellt im Text-Modus einen 2K großen Bildwiederholspeicher für Text und einen 2K großen Attribut-Speicher zur Verfügung. Diese Speicher sind ineinander verschoben; innerhalb eines linearen 4K-Bereichs stellen die geraden Adressen den Text-Speicher und die ungeraden Adressen den Attribut-Speicher dar.

Video-Attribute waren aber damals, als der TRS-80 geboren wurde, noch unbezahlbarer Luxus. Deshalb scheint der Attribut-Speicher der Hercules der Kompatibilität zum TRS-80 im Wege zu stehen. Der TRS-80 braucht einen zusammenhängenden Bildwiederhol-Speicher von 1K Größe (und manche jüngeren Brüder können auch mit 2K Textspeicher umgehen).

Dieses unlösbare Problem läßt sich dadurch lösen, daß beim Text-Modus die Adressen A0-A10 der CPU (2K-Bereich) auf die Adressen A1-A11 der Hercules-Karte gelegt werden. Diese Umsetzung darf aber nur bei Zugriff auf den Text- und den Attribut-Speicher erfolgen. Bei HRG-Betrieb und bei Zugriff auf die Ports der Karte müssen alle Adressen direkt durchgeschaltet werden. Die für das Selektieren von Text- oder Attribut-Speicher zuständige Adresse A0 (zur Hercules) wird im Text--Modus von D7 des erweiterten Ports B8H und bei Zugriff auf Hercules--Ports und HRG-Speicher von A0 der CPU bedient. Diese Adreßsteuerung wird von PAL3 und einem 74LS157-Multiplexer erledigt.

Die damit auf 2K reinen Text-Speicher (und nach Umschalten 2K reinen Attribut-Speicher) beschränkte Bildschirmseite ist aber für den TRS--80, der nur 1K bei 3C00H dafür vorsieht, immer noch zu groß. Deshalb kann mit D0=0 an Port B8H (Einstellung nach der Initialisierung im ROM) die Freigabe nur der oberen Hälfte des Text-Speichers im Bereich 3C00-3FFF vorgegeben werden. Für das TRS-80-Format 64x16 Zeichen muß dann aber auch der CRTC-6845 entsprechend programmiert werden. Dies wird ebenfalls durch die Initialisierung im ROM erledigt. Das Betriebsystem findet dann beim anschließenden Booten eine voll TRS-80--kompatible Maschine vor.

Wenn D0=1 an Port B8H ausgegeben wird, ist der volle 2K-Text-Speicher im Bereich 3800-3FFF verfügbar. Um eine vernünftige Darstellung auf dem Bildschirm zu haben, muß dann aber auch der CRTC-6845 entsprechend initialisiert werden. Im 80x25-Modus ist die Tastatur dann aber nicht mehr zugänglich. Für das Auslesen der Tastatur muß dann immer erst in den 64x16-Modus zurückgeschaltet werden und nach dem Auslesen der Tastatur für weitere Bildschirm-Ausgaben wieder der 80x25-Modus eingestellt werden. Wenn dabei die Register des CRTC-6845 in Ruhe gelassen werden und nur mit D0 von Port B8H geschaltet wird, bleibt die Darstellung auf dem Bildschirm unbeeinträchtigt. Die Umschaltung betrifft dann nur die Zugriffsmöglichkeiten der CPU auf den Bildwiederholspeicher. Das für die Darstellung des Bildspeichers auf dem Monitor verantwortliche Auslesen des 6845 wird dadurch nicht belangt.

Die Initialisierungs-Routine im ROM füllt zunächst den gesamten Attribut-Speicher mit dem Standard-Wert 7. Das bedeutet, daß die Zeichen auf dunklem Hintergrund mit normaler Helligkeit dargestellt werden, wenn der Attribut-Speicher bei der Textausgabe nicht berücksichtigt wird.

Es sind allerdings dies die Zeichen entsprechend Big Blue's Verständnis von ASCII-Code. Die Zeichen mit den Codes 5BH-5FH und 7BH-7FH sind grundsätzlich die amerikanischen Zeichen [(geschweifte und eckige Klammern u.s.w.)]. Den deutschen Umlauten und auch allen anderen nationalen Sonderzeichen sind feste ASCII-Codes mit gestztem Bit7 zugeordnet. Das hat den Nachteil, daß bei allen von Standard-Programmen unter NEWDOS und CP/M ausgegebenen Texten anstelle der deutschen Umlaute die entsprechenden amerikanischen Zeichen erscheinen. Ein Vorteil besteht aber auch darin, daß amerikanische Sonderzeichen und sämtliche nationalen Umlaute gleichzeitig dargestellt werden können. Für Soft-Freaks besteht hier ein weites Betätigungsfeld in der Anpassung von Tastatur-Treibern und Standard-Programmen.

Die Bildschirm-Attribute

Wenn mit D1=1 und D7=1 an Port B8H die Text-Betriebsart und der Attribut-Speicher eingeschaltet werden, kann im Adreßbereich des Text-Speichers ein Attribut-Byte eingetragen werden, das die Darstellung des ASCII-Zeichens auf gleicher Adresse im Text-Speicher beeinflußt. Die Attribut-Bits wirken nur auf das eine Zeichen und nicht auf den gesamten Bildschirm. Wenn mit Attributen gearbeitet werden soll, muß für jedes Zeichen im Text-Speicher auch ein Byte im Attribut-Speicher eingetragen werden. Beim Scrollen muß auch der Attribut-Speicher genauso wie der Text-Speicher verschoben werden.

Die Bits 3 und 7 eines Attribut-Bytes haben eigenständige Funktionen, während die Bits 0-2 und die Bits 4-6 in Kombination miteinander zu benutzen sind:

```
D7 :  die Wirkung  von D7 eines Attribut-Bytes  hängt davon ab,
      welcher Wert für D5 an Port B8H ausgegeben wurde.
   Port B8H, D5=0:
      D7(Attribut) =0 : Zeichen erscheint auf dunklem Hintergrund
                   =1 : Zeichen erscheint vor halbhellem Hintergrund
   Port  B8H,  D5=1:
```

D7(Attribut) =0 : kein blinkendes Zeichen
=1 : Zeichen wird blinkend dargestellt

D3 = 0 : normale Helligkeit der Darstellung
= 1 : Highlighting

D0-D2, D4-D6 :

D6	D5	D4	D2	D1	D0	Bedeutung
0	0	0	0	0	0	Zeichen nicht dargestellt
0	0	0	0	0	1	unterstrichen dargestellt
0	0	0	1	1	1	normale Darstellung
1	1	1	0	0	0	inverse Darstellung

Die HRG der Hercules-Karte

Mit 720x348 Punkten Auflösung und 2 Seiten läßt sich schon sehr schöne Grafik auf die Mattscheibe bringen. Diese Auflösung benötigt natürlich auch Speicherraum für die beiden 32K-Pages. Diese Bereiche analog zum GenieIIs einfach in den logischen Adreßraum der nach RESET eingestellten (unteren physikalischen) 64K zu legen, erscheint etwas unzweckmäßig zu sein. Es müßte dann eine Hälfte des ohnehin nur 64K großen, direkt erreichbaren Arbeitsspeichers ausgeschaltet werden. Für Grafik-Anwendungen müßte man sich festlegen, in welcher Hälfte der 64K die Grafik-Routinen laufen sollen.

Hier ist der HRG-Speicher ans obere Ende des physikalischen 1MB-Speichers verbannt worden. Das Besudeln des Grafik-Speichers erfolgt im einfachsten Fall durch blockweise Übertragung per DMA. Wer aber trotzdem gerne direkt in das HRG-RAM schreiben möchte, kann sich dieses durch die MMU in die untere oder obere Hälfte seines 64K-Arbeitsspeichers einblenden.

Nach einem RESET ist die Text-Betriebsart eingestellt. Mit D1=1 an Port B8H läßt sich auf die HRG-Betriebsart "Half-Mode" umstellen, wenn in Port BFH D0=1 eingetragen ist. Dabei ist die untere Hälfte (Page 0) erreichbar. Um auch die Page 1 erreichen zu können, muß auch D1=1 an Port BFH ausgegeben werden.

Während der Grafik-Betriebsart ist der Text-Bildspeicher nicht zugänglich. Da dieser aber physikalischer Bestandteil der Page 0 der HRG ist, wird der Inhalt des Text-Speichers bei Grafikausgaben an Page 0 überschrieben. Gleiches gilt für den Attribut-Speicher.

Die Tastatur

Dieses Gerät erleidet hier die größten Einbußen in der Präsenz. So ist der Bereich, in dem die Tastatur ausgelesen werden kann, auf den dafür nötigen Platz von 256 Adressen beschränkt. In der 64x16-Text-Betriebsart (D0=0 an Port B8H, die einzige Einstellung, in der die Tastatur erreichbar ist) stehen bei 3900H - 3BFFH ein paar Bytes frei verfügbaren Speichers bereit. Auch der Speicher parallel zur Tastatur kann bei eingeblendeter Tastatur beschrieben werden; auslesen läßt er sich nur, wenn auf Grafik-Betriebsart umgestellt wird.

Beim Arbeiten im 80x25-Textmodus ist die Tastatur nicht erreichbar. Dann liegt ab 3800H Bildwiederhol-Speicher vor. Um die Tastatur zwischendurch auszulesen, muß mit D0=0 an Port B8H vorübergehend auf 1K Video-RAM umgeschaltet werden. Da das aber nicht den Zugriff des CRTC-6845 auf den Bildspeicher beeinflußt, zeigt sich das Umschalten nicht in der Darstellung auf dem Bildschirm.

Wenn es auch Wahnsinn ist, eine Tastatur memory mapped als offene Matrix an einen Computer anzuschließen, weil bei höheren Taktraten und längeren Anschlußleitungen Probleme zu erwarten sind, kommt man bei einem TRS80-kompatiblen Rechner um diesen Unsinn nicht herum, wenn auch der Betrieb von OLDDOS/80 und der darunter laufenden Programme möglich sein soll.

Ein einfaches Interface (Abb.7) kann auf einem Stück Lochraster--Platine in Fädeltechnik aufgebaut werden und mit einer VG64--Steckerleiste auf den ECB-Bus gesteckt werden. Die Tastatur kann daran über ein nicht zu langes, 16poliges Kabel angeschlossen werden.

Die Freigabe des ROMs

Wie es sich für einen Trash-80-kompatiblen Computer gehört, liegen nach dem Einschalt-RESET die 12K Löffel-II-BASIC bei 0000H-2FFFH im ROM rum. Zusätzlich wurde hier auch der Bereich 3000H-36FFH als ROM decodiert. Hier sind die nötigen Initialisierungen der HD64180-CPU und der Hercules-Karte untergebracht. Der Einsprung bei 3050H muß im Level-II-ROM anstelle des ersten Sprungbefehls nach RST 00H gepatcht werden. Die Initialisierung endet mit einem Sprung an das dort vorher vorhandene Sprungziel 0674H.

Wenn D7 an Port F4H high ausgegeben wird, verschiebt sich die Lage des ROM-Inhalts von 0000-36FF nach C000-F6FF. Das ist zwar wenig sinnvoll, weil der Stoff auf dieser Adreßlage nicht läuft, ließ sich aber wegen des für CP/M nötigen Verschiebens des memory mapped I/O nach F7E0-FFFF nicht verhindern. Wenn das ROM aber ausgeschaltet wird (D2 an Port B8H high), stört es dort nicht mehr.

Das Abschalten des ROMs über D2 an Port B8H läßt sich natürlich auch in der ursprünglichen Lage bei 00000H durchführen. Da das ROM ohnehin nur beim Lesen freigegeben wird und beim Schreiben RAM eingeblendet ist (auch wenn das ROM nicht abgeschaltet ist) kann das Kopieren des ROM-Inhaltes in das RAM durch ein einfaches LDIR auf sich selbst (HL = DE = 0) erfolgen. Das wird in der Initialisierung vorsichtshalber auch schon mal gemacht.

Floppy, Drucker und der ganze Rest

Damit die Abwärtskompatibilität zum TRS-80 beibehalten wird, liegen diese Komponenten memory mapped bei 37EXH. Allerdings wurde die Option eingebaut, anstelle dieser Memory-Adressen auch die entsprechenden Ports E0H-EFH für das Ansprechen dieser Einheiten benutzen zu können. Dafür wird mit D4=1 an Port B8H auf Betrieb über Ports umgeschaltet.

Wer von seinem alten Genie oder TRS80 her noch einen RB-EXP1 Controller besitzt, hat damit ein weiterhin brauchbares Gerät vorliegen.

Diesem Board muß nur noch ein VG64-Stecker spendiert werden. Dieser Stecker wird auf einen 2 cm breiten Streifen Lochraster-Platine gelötet, der wiederum mit zwei schmalen Streifen Pertinax oder sonstigem Kunststoff mit dem RB-EXP1 verschraubt wird (Abb.6). Auf der Lötseite werden dann die ECB-Signale des Steckers mit den entsprechenden Punkten der beiden Anschlußstecker des EXP1 verbunden.

Anschluß des RB-EXP1 Controllers an den ECB-Bus

ECB-Bus Pin	Signal	RB-EXP1	
2c	D0	Pin 9	24poliger
14c	D1	Pin 10	Stecker
4c	D2	Pin 11	
4a	D3	Pin 13	
5a	D4	Pin 14	
2a	D5	Pin 15	
3a	D6	Pin 16	
3c	D7	Pin 17	
5c	A0	Pin 8	
7c	A1	Pin 7	
6a	A2	Pin 6	
6c	A3	Pin 5	
1a,c	+5V	Pin 24	
32a,c	GND	Pin 12	
13a	+12V	Pin 11	16poliger
15a	-5V	Pin 12	Stecker
21c	/INT	Pin 4	
31c	/RESET	Pin 5	
z.B. 23a	/FLO	Pins 2,14	74LS155

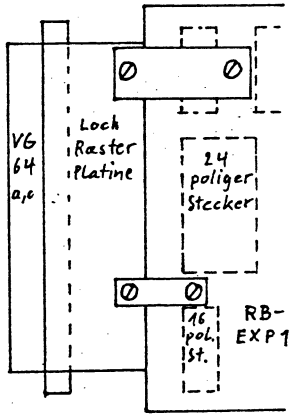


Abb.4: ECB-Bus-Anschluß der RB-EXP1

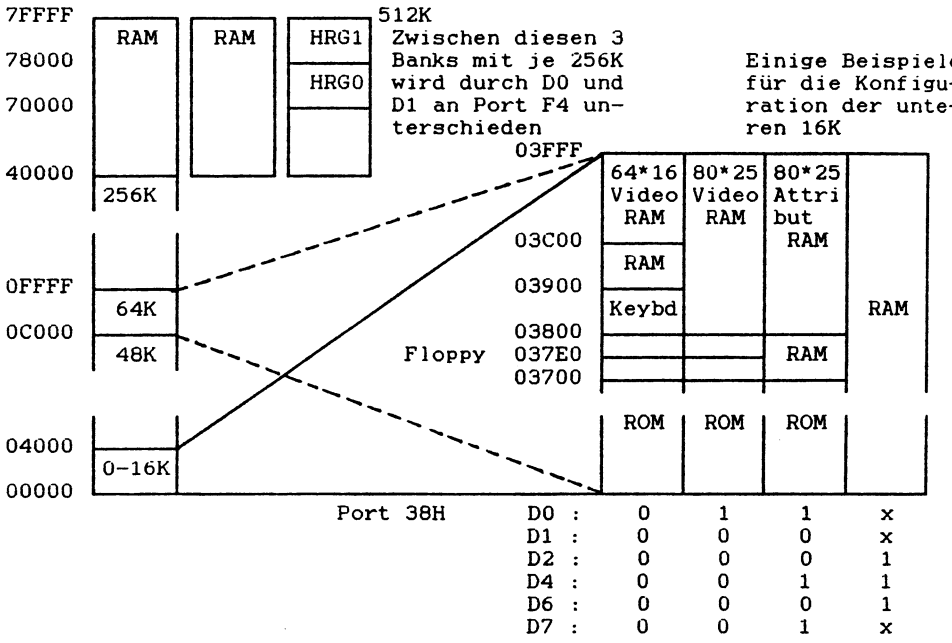
Da das Freigabe-Signal für den Controller bereits auf dem CPU-Board erzeugt wird und auch schon die Umschaltmöglichkeiten beinhaltet, sollte lieber dieses Signal verwendet werden als das auf dem EXP1 hergeleitete Signal /37EX. Dafür wird das Signal /FLO von PAL1 auf dem CPU-Board über eine nicht benutzte ECB-Leitung an den Controller geführt und dort an die Pins 2 und 14 des 74LS155 gelegt. Die beiden 74LS30 beim 16poligen Stecker, die vorher das Signal /37EX und die Drucker-Freigabe über Port FDH geliefert haben, können entfernt werden. Die Verwendung des Signals /FLO erübrigt auch die Zuführung der Signale, die in obiger Tabelle nicht angeführt sind und bei Verwendung des EXP1 im Genie oder TRS80 zusätzlich angeschlossen werden mußten.

RAM-Speicher satt

Grundsätzlich besteht der gesamte physikalische Adreßraum vom 1MB aus RAM-Speicher. Nur dort, wo irgendwelche memory mapped I/O-Baugruppen eingeblendet werden, ist der RAM-Speicher nicht erreichbar. Durch Abschalten der mm I/O-Baugruppen wird dort dann automatisch der entsprechende RAM-Bereich eingeblendet.

Im Bereich des ROMs (00000-036FF) und der Tastatur (03800-038FF), wenn diese im 64x16-Textmodus erreichbar ist, kann der RAM-Speicher auch beschrieben werden, ohne daß ROM bzw. Tastatur ausgeschaltet werden müssen. Nur zum Lesen dieser RAM-Bereiche müssen diese Einheiten ausgblendet werden.

Memory Map



Ein entscheidender Nachteil des Industrie-Standards sei zum Schluß noch erwähnt. Um das vom Genie her bekannte Flickern bei Ausgaben auf den Bildschirm zu verhindern, lassen modernere Hercules-Karten einen Zugriff auf das Video-RAM nur zu, wenn der Elektronenstrahl vom rechten zum linken Bildrand zurückläuft und dabei dunkel gesteuert ist. Die CPU wird während der Zeit bis zum Strahlrücklauf mit einem Wait (/IO_CH_RDY, Pin A10 des PC-Slots) ruhiggehalten. Offensichtlich ist das Wait-Timing bei HD64180 anders als beim 80286 im PC-AT, denn die Hercules macht im Genie nur bis 5MHz Systemtakt mit; bei höheren Takten erscheint nur noch Müll auf dem Fernseher.

Es gibt aber auch Hercules-Karten, die sich wie ein normales Genie-Video-Interface ohne Rücksicht auf Flickern beschreiben lassen. Eventuell läßt sich damit dann der für das CPU-Board als machbar zu erwartende Takt von 9,216MHz fahren. Wahrscheinlich muß dann aber ein Wait aus dem Freigabesignal /HA18 für die Speicher der Hercules-Karte erzeugt werden, das den Zugriff der CPU um 1 oder 2 Takte verlängert. Da ich eine solche Karte aber nicht habe, will ich darüber auch nicht weiter nachdenken.

Das System verfügt noch nicht über eine Reset-Schaltung. Diese kann im

einfachsten Fall durch Kurzschließen von /RESET (ECB-Bus, 31c) mit GND über einem Taster bereitgestellt werden. Da aber während der Zeit, in der der Taster gedrückt ist, kein Refresh der dynamischen RAMs erfolgt, gehen bei einem Reset Daten verloren. Eine einfache Schaltung mit einem 555-Timer und einem Taster erzeugt sowohl beim Einschalten als auch beim Betätigen des Tasters ein getimtes /RESET-Signal, das ein längeres Ausbleiben des Refresh verhindert (Abb.5).

Freie Verdrahtung auf der Lötseite

Signal	IC,Typ,Pin —— IC,Typ,Pin ...
D0	18,273,8 10,27128,11
D1	18,273,7 10,27128,12
D2	18,273,4 PC-Slot,A7 10,27128,13
D3	18,273,13 10,27128,15
D4	18,273,8 PC-Slot,A5 10,27128,16
D5	18,273,14 10,27128,17
D6	18,273,17 10,27128,18
D7	18,273,18 10,27128,19
D0	17,64180,34 30,125,5
D1	17,64180,35 30,125,2
D2	17,64180,36 38,245,4
D3	17,64180,37 38,245,5
D4	17,64180,38 38,245,3
D5	17,64180,39 38,245,9
D6	17,64180,40 30,125,11
D7	17,64180,41 31,174,13
/HALT	17,64180,56 VG64-Leiste,25c
/REF	17,64180,57 VG64-Leiste,28a
/IOE	1,PAL1,1 17,64180,58 19,32,10 PC-Slot,A10 33,245,7 37,PAL2,13
/ME	16,32,4 17,64180,59 19,32,13 33,245,2 37,PAL2,14
/LIR	17,64180,61 37,PAL2,23
/WR	17,64180,62 19,32,12 33,245,9 37,PAL2,3
/RD	1,PAL1,2 17,64180,63 19,32,1 33,245,8 37,PAL2,1
CLK	17,64180,64 VG64-Leiste,29c
/WAIT	17,64180,4 VG64-Leiste,10a PC-Slot,A10
/BUSAK	17,64180,5 37,PAL2,2
/BUSRQ	17,64180,6 VG64-Leiste,11a
/RESET	18,273,1 17,64180,7 15,04,5 31,174,1 VG64-Leiste,31c
/NMI	17,64180,8 VG64-Leiste,20c
/INT0	17,64180,9 VG64-Leiste,21c
/ST	17,64180,12 35,245,2
A0	17,64180,13 10,27128,10 25,PAL3,7 32,157,2 38,245,2
A1	17,64180,14 10,27128,9 32,157,14/3 36,245,6
A2	17,64180,15 10,27128,8 32,157,13/5 36,245,3
A3	17,64180,16 10,27128,7 32,157,11/6 36,245,4
A4	17,64180,17 10,27128,6 25,PAL3,16 32,157,10 36,245,5
A5	17,64180,18 10,27128,5 25,PAL3,6 36,245,7
A6	17,64180,19 10,27128,4 25,PAL3,5 36,245,8
A7	17,64180,20 10,27128,3 25,PAL3,4 36,245,9
A8	1,PAL1,6 10,27128,24 17,64180,21 25,PAL3,3 35,245,3
A9	1,PAL1,5 10,27128,25 17,64180,22 25,PAL3,2 33,245,4
A10	1,PAL1,4 10,27128,21 17,64180,23 25,PAL3,1 35,245,5
A11	1,PAL1,3 10,27128,23 17,64180,24 35,245,7
A12	17,64180,25 10,27128,2 20,86,5 28,30,12 33,245,5
A13	17,64180,26 10,27128,27 20,86,9 28,30,11 33,245,6
A14	17,64180,27 20,86,2 35,245,6
A15	17,64180,28 20,86,12 33,245,3
A16	17,64180,29 12,157,5 27,02,11 35,245,4
A17	17,64180,30 12,157,11 27,02,12 35,245,8
A18	17,64180,31 24,00,5 21,08,13/2 35,245,9
A18'	21,08,4 11,157,14
A19'	21,08,5 11,157,2
B81	18,273,6 25,PAL3,9 29,125,4 16,32,5 15,04,9
/B81	15,04,8 20,86,4 29,125,10
B87	18,273,16 25,PAL3,11
Q0	30,125,5 16,32,10
Q1	30,125,2 21,08,12
X	16,32,8 21,08,1
Y	24,00,6 15,32,9
/M016	26,20,8 1,PAL1,14
/M1216	28,30,8 1,PAL1,22
/MFxxxx	26,20,6 1,PAL1,18
/OUTB8	37,PAL2,19 9,273,11
/OUTF4	37,PAL2,18 31,174,9
/INF4	37,PAL2,17 30,125,10
/ADR	37,PAL2,15 1,PAL1,17
/ROM	1,PAL1,20 10,27128,20 10,27128,22
/FLO	1,PAL1,16 37,PAL2,19 VG64-Leiste,z.B.23c
/KB	1,PAL1,19 37,PAL2,20 VG64-Leiste,z.B.23a
/RAM	1,PAL1,22 16,32,1
/HA18	1,PAL1,21 PC-Slot,A13
HA15	20,86,11 PC-Slot,A16
HA14	20,86,3 PC-Slot,A17
HA11	29,125,6 PC-Slot,A20
RESET	15,04,6 PC-Slot,B2
SEL	16,32,6 32,157,1
+5V	22,1489,14 20,86,14 17,64180,32 10,27128,28 4,RAM,9
+5V	1,PAL1,24 2,RAM,9
+5V	25,PAL3,20 PC-Slot,A12/A14/A15/B29
GND	10,27128,14 20,86,7 26,20,7
GND	17,64180,33 15,04,7 V24-Stecker,9/10
+12V	23,1488,14 VG64-Leiste,13a
-12V	23,1488,1 VG-Leiste,14a

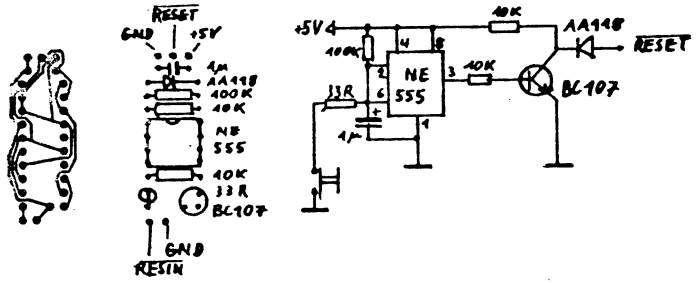


Abb.5: Getimeter /RESET beim Einschalten und Drücken des Tasters

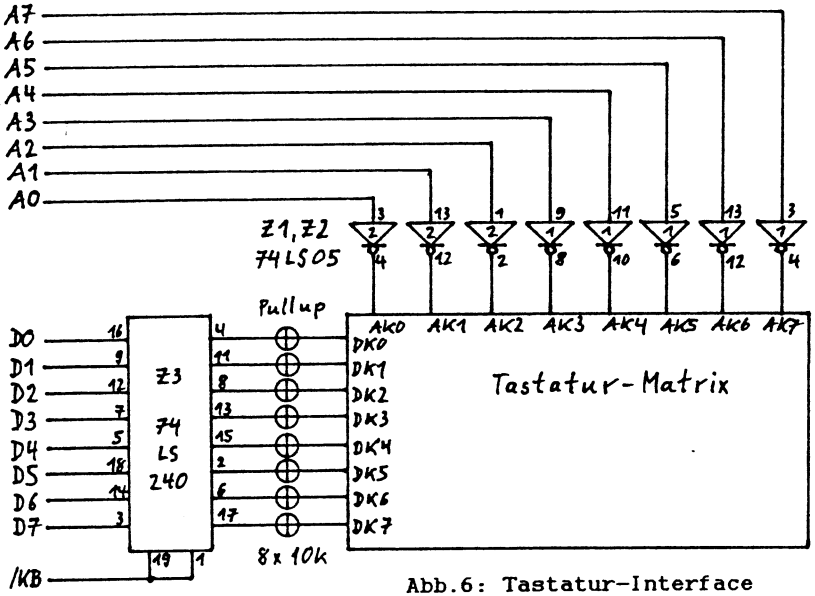
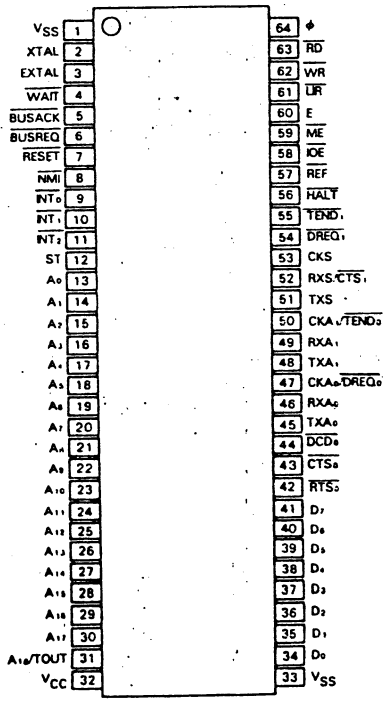
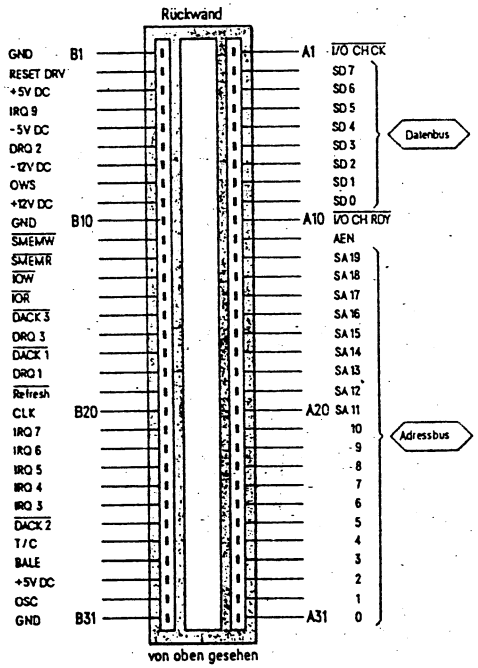


Abb.6: Tastatur-Interface



Pinout der HD64180-CPU



Pinout des PC-Slots


```
;Initialisierung der Hercules-Karte und der HD64180-CPU
;im Sonder-ROM-Bereich ab 3000H
;
;Diese Routinen werden aus dem ROM heraus angesprungen
;und übergeben zum Schluß wieder an die TRS-80 Boot-
;Routine. Dafür muß das ROM ab 0000H folgendermaßen
;geändert werden:
;     vorher                        nachher
;0000   DI                    0000   DI
;0001   XOR     A             0001   XOR     A
;0002   JP      0674H         0002   JP      3050H
;
;Die Initialisierung schließt mit JP  0674H ab.
;
;
attrib  EQU     07h     ;D7=0 nicht blinkend bzw
                        ;   normaler Hintergrund
                        ;D6-D4=0,D2-D0=1 normale
                        ;   Darstellung
mode    EQU     0       ;D7=0   HRG Page 0
                        ;D5=1   Blinken erlaubt
                        ;D3=1   Video-Signal aktiv
                        ;D0=0   Text-Betriebsart
config  EQU     3       ;D0=1   HRG erlaubt
                        ;D1=1   HRG-Page2 enabled
;
;
        ORG     3000h           ;Anfang Sonder-ROM
;
;Initialisierungdaten für den CRTC6845
;64*16-TRS80-Standard-Einstellung
tab     DEFB    6eh,40h,50h,0ah,14h,06h,10h,12h
        DEFB    02h,0eh,29h,09h,04h,00h,00h,00h
;80*25-Modus
        DEFB    6eh,50h,58h,0ch,1fh,02h,19h,1bh
        DEFB    02h,09h,29h,09h,00h,00h,00h,00h
;64*24-Modus
        DEFB    6eh,40h,50h,0ah,1bh,09h,18h,19h
        DEFB    02h,0ah,29h,09h,02h,00h,00h,00h
;64*32 interlaced
        DEFB    6eh,40h,50h,0ah,13h,02h,10h,11h
        DEFB    03h,0fh,2fh,0fh,00h,00h,00h,00h
;92*22-Modus
        DEFB    70h,5ch,61h,08h,1dh,08h,16h,19h
        DEFB    02h,0bh,29h,09h,00h,00h,00h,00h
;
        LD      SP,8000h
        LD      A,3             ;obere 256K einstellen
        OUT     (0f4h),A
        LD      A,81h           ;Attribut-Speicher mit
        OUT     (0b8h),A        ;Standard-Wert füllen
        LD      HL,3800h        ;Video-RAM ab 3800h
        LD      A,attrib
        LD      (HL),A
        LD      DE,3801h
        LD      BC,07ffh        ;2k Speicher
        PUSH    HL
        PUSH    DE
        PUSH    BC
        LDIR
;
        POP     BC
        POP     DE
        POP     HL
        LD      A,1             ;Umschalten auf Text-
```

```
        OUT     (0b8h),A        ;-Speicher
        LD      A,' '           ;mit Blanks füllen
        LD      (HL),A
        LDIR
;
        LD      HL,0            ;ROM-Inhalt ins RAM
        LD      DE,0            ;kopieren
        LD      BC,3700h
        LDIR
;
;HD64180 initialisieren
        LD      A,10h           ;WAITs: Memory 0, I/O 2
        DB      0edh,39h,32h    ;OUT0 (DCNTL),A
        LD      A,83h           ;Refresh-Control
        DB      0edh,39h,36h    ;optimieren
        LD      A,44h           ;Common 0-32K, Bank 32-
        DB      0edh,39h,3ah    ;64k einstellen
        XOR     A               ;Common1 als Bank benutzen
        DB      0edh,39h,38h    ;Bank bei 08000-0FFFF
;
;CRTC6845 für 64*16 Zeichen initialisieren
        LD      HL,tab          ;HL => CRTC-Parameter
        CALL    setcrtc         ;raus an CRTC
;
;Video_Signal einschalten
        LD      A,08h           ;Video-Signal aktiv
        OUT     (0b8h),A        ;Video bei 3c00-3fff
                                ;Text-Modus eingesch.
                                ;Blinken gesperrt
        JP      0674h           ;und weiter im TRS80-
                                ;Boot-Gebaren

;CRTC-6845 initialisieren (HL => Parameterliste)
setcrtc XOR     A               ;für Video-Mode-Umschal-
        OUT     (0bfh),A        ;tung nötig
        LD      B,16            ;16 Bytes ausgeben
        LD      C,0b5h          ;CRTC-Daten-Port
        XOR     A               ;CRTC-Register-Nr ab 0
crtclp  OUT     (0b4h),A        ;1. Register
        INC     A               ;A=> nächstes Register
        OUTI                    ;raus das Zeugs
        JR      NZ,crtclp
        LD      A,config        ;HRG voll freigeben
        OUT     (0bfh),A
        RET

        END
```


Assembler-Listing für PAL1

```
PAL20L8
Helmut Bernhardt            22.01.89
Erzeugen der mm-Freigabesignale

ioe rd a11 a10 a9 a8 nc b80 b81 b82 b84 gnd
b86 m016 nc flo adr mfxxxx kb rom ha18 ram m1216 vcc

/ha18 = /b86 */b81 */b80 */m1216 *a11 *a10
      + /b86 */b81 * b80 */m1216 *a11
      + /b86 * b81 */mfxxxx

/kb   = /b86 */b80 */rd */m1216 *a11 */a10 */a9 */a8

/flo  = /b84 */m1216 */a11 *a10 *a9 *a8 */adr
      +  b84 */ioe */adr

/rom  = /b82 */m016 *m1216 */rd
      + /b82 */m1216 */a11 */a10 */rd
      + /b82 */m1216 */a11 * a10 */a9 */rd
      + /b82 */m1216 */a11 * a10 * a9 */a8 */rd

/ram  = ioe *ha18 *kb *flo *rom
```

Assembler-Listing für PAL2

```
PAL20L8
Helmut Bernhardt            24.01.89
I/O-Decoder und Datentreiber-Steuerung

rd busak wr a0 a2 a3 a4 a1 a5 a6 a7 gnd
ioe me adr herc inf4 outf4 outb8 flo kb rein lir vcc

/inf4  = /ioe */rd *a7 *a6 *a5 *a4 */a3 *a2 */a1 */a0

/outf4 = /ioe */wr *a7 *a6 *a5 *a4 */a3 *a2 */a1 */a0

/cutb8 = /ioe */wr *a7 */a6 *a5 *a4 *a3 */a2 */a1 */a0

/adr   = a7 * a6 * a5 * /a4

/herc  = /ioe *a7 */a6 *a5 *a4

/rein  = busak */ioe */lir
       + /ioe */rd *inf4 *herc
       + /busak *rd
       + /flo */rd
       + /kb */rd
```

Assembler-Listing für PAL3

PAL16L8
Helmut Bernhardt 23.01.89
Adreßübersetzung für die Hercules-Karte

```
a10 a9 a8 a7 a6 a5 a0 me b81 gnd
b87 ha0 ha5 ha6 ha7 a4 ha8 ha9 ha10 vcc

/ha0  = /me * /b81 * /b87
      + /me *  b81 * /a0
      +  me * /a0

/ha5  = /me * /b81 * /a4
      + /me *  b81 * /a5
      +  me * /a5

/ha6  = /me * /b81 * /a5
      + /me *  b81 * /a6
      +  me * /a6

/ha7  = /me * /b81 * /a6
      + /me *  b81 * /a7
      +  me * /a7

/ha8  = /me * /b81 * /a7
      + /me *  b81 * /a8

/ha9  = /me * /b81 * /a8
      + /me *  b81 * /a9

/ha10 = /me * /b81 * /a9
      + /me *  b81 * /a10
```

Pinbelegung des V24-Steckers CN2

Signal	Pin	Pin	Signal
/CTS1	1	2	TXS1
/RTS0	3	4	RXA0
/CTS0	5	6	RXA1
TXA0	7	8	TXA1
GND	9	10	GND

Bei welchem VIDEO-GENIE- Besitzer läuft eine RB V.24-Schnittstelle.
Ich bin sehr an Erfahrungsaustausch interessiert. Meine
Schnittstelle läuft nämlich nicht.

Hans-Günther Hartmann, Adresse und Teflonnummer siehe Info!

Verkaufe

1 x TRS-80 Model I, Level II
mit 10er Block auf der Tastatur und HRG,
eingebaut in ein Stahlblechgehäuse
2 x Laufwerke 40 Spuren, DD
1 x entspiegelter Monitor

dazu diverse Software und Handbücher

Preis: VS

W. Camphausen
Florastr. 14
5600 Wuppertal 1
Tel.: 0202-750399

Suche folgende Zeitungsartikel leihweise oder als Kopie:

Creative Computing: Juli 79, S.34-37

Dr.Dobbs: 22:10-13;
48:34-37;
Feb.77, S.30-43;
Feb.80, S.18-25

Interface Age: Aug.78, S.152-156;
Dez.78, S.130-135;
Aug.80, S.88-90;
Sept.80, S.80-86,130-131;
Okt.80, S.106-107, 137-141

Microsystems: Jul/Aug.80, S.32-35

Kosten werden erstattet. Wer kann helfen?

Norbert Giese A.-Stifter-Weg 10 7034 Gärtringen 07034/29980

Betr.: Ihr Beitrag im c't-Club 7/89

Ich möchte zwar nicht dem Club beitreten, habe aber für den Tandy Modell 1 jede Menge an Software.

Den Tandy biete ich zum Verkauf an.

1 Tandy Modell 1 mit Expansion-Interface
alles in einem Gehäuse untergebracht.

2 serielle Schnittstellen eingebaut
1 Tandy Monitor
1 Tandy Drucker 100 ← Line Printer 100

Zeitschriften aus den USA "MICRO 80"

Alles zusammen für 400,-DM

Ich würde mich freuen von Euch zu hören.

Mit freundlichen Grüßen

Peter Schopen
Lochnerstr. 25
4030 Ratingen

Suche für SHARP MZ-80B:

Informationen über Bankumschaltung, speziell Funktion von PAL (IC 2 im Schaltbild).

Schaltbild oder Idee für einfachen EPROMMER, der die nötigen Programmierspannungen aus +5 Volt generiert. Gerät soll nur an Parallelport angeschlossen werden; ohne externes Netzteil.

Norbert Giese A.-Stifter-Weg 10 7034 Gärtringen 07034/29980

TRS 80 - Modell 1 mit HRG1B und 64 KB Ram, 3 BASF 5.25 " 1D DD, Exp. Interface, 256 KB - Banker Platine, CP/M - Platine. Netzteile, Monitor Sanjo bernsteinfarbig. Preis VS.

Genie 1 EG3003 16 KB mit Centronicsinterface, 200.- DM. Zero Eprommer mit Textolsockel für Genie jedoch Software auf Diskette, 80.- DM.

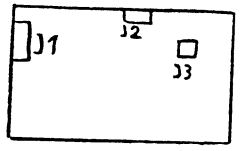
Reiner Stober Tel.: 05153/1564 ab 20.00Uhr

Außerdem möchte ich gleich mal eine Anfrage im Clubinfo starten:
Wie schon erwähnt haben meine beiden Laufwerke nur eine Kapazität von 180K. Ich möchte nun wissen, was ich an Hard- und Software ändern muß, um die Kapazität zu erhöhen.
Bei dem 3'-Floppy ist wahrscheinlich nichts mehr zu machen, dafür kann man bestimmt aus dem 5 1/4'-Floppy noch mehr herausholen. Laut Beschreibung handelt es scih um ein Original-Commodore-Amiga-Laufwerk, das ein meinen Bus angepaßt wurde. Zum Umschalten der beiden Köpfe wurde einfach ein Schalter vorne am Gehäuse angebracht.
Hinweise und Tips, die zur Erhöhung der Speicherkapazität beitragen, werden dankbar angenommen.

Bis zum nächsten Mal

Jürgen Kemna

Shuga meines 5¼'-Laufwerks



J1: DS ○○ ○
DD ○○
D1 ○○
D2 ○○
D3 ○○
DA ○○○

HH ○○
HS ○○
HL ○○
HM ○○
○○ MM
○○ MS

J2: ○○○○○○

J3: ○○○

GENIE III : Einzelteile zu verkaufen

CPU-Board, FDC-Board, Video-Board : je 50.-DM
Netzteil : 60.-DM
Gehäuse mit Einbaurahmen und Buskarte gibt's umsonst dazu
Video-Teil (60Hz-Monitor) und Floppy Drives sind defekt
Tastatur wollte ich nicht unbedingt loswerden; darüber läßt sich aber reden.

Helmut Bernhardt 0431/241907

Ein guter Meter TRS-80-Literatur zu verschenken

Enthalten sind die goldenen Jahrgänge von "80 Micro", die "Blah, blah, blah... and other Mysteries" - Bücher und noch so manche anderen Schmankerl.
Das alles verschenkt Christof Ueberschaar, wenn die Papierberge in Wilhelmshaven oder Kiel abgeholt werden oder die Portokosten übernommen werden. Da Christof momentan nicht weiß, wo es ihn in den nächsten Monaten hinverschlägt, und ich nicht weiß, wann das nächste Info erscheint, stelle ich mich als Kontaktadresse zur Verfügung.

Helmut Bernhardt 0431/241907

c't 12/83 bis 3/89 zu verkaufen

Alles, was c't über 8-Bit-Computer gebracht hat, ist hier zu finden. Bernd Bauer (Tel. 0431/673819, ab 19:Uhr) will das nicht noch mal alles bauen und sich daher von dem Meter Literatur trennen. Wer Interesse hat, kann sich mit ihm über den Preis streiten.

Suchmeldung,

suche Double Density Controler für Model 1. Wer kann helfen. Anleitung zum Eigenbau zwecklos, da Niete in Hardware

Egbert Schröer
Joachimstr. 16
4270 Dorsten 1
Tel.: 02362/75840

Mikrocomputer-Betriebssysteme - eine Einführung

von Gerald Schröder

Dieser Artikel soll der Auftakt zu einer kleinen Serie über CP/M sein. Allerdings werden auch Newdos sowie (am Rande) MS-DOS erwähnt. Hier soll es erst einmal um die Grundlagen gehen, die für alle Betriebssysteme gelten, die auf kleineren Rechnern zum Einsatz kommen. Bei Großrechnern sind ganz andere Anforderungen wichtig.

Ich habe mich bemüht, einen lockeren Ton anzuschlagen. Damit sollen vor allem die Leute angesprochen werden, die sich noch keine Gedanken über dieses Thema gemacht haben. Weiterhin muß ich zugeben, daß ich mich auf keinerlei Literatur berufen kann, weil sich die mir bekannten Bücher nur mit Großrechner-Betriebssystemen befassen. Alle Ausführungen sind also meine private Meinung, gewonnen aus 8 Jahren Erfahrung mit Mikrocomputern, die mich zu folgenden Forderungen brachten:

Freiheit für unsere Programme
Gleichheit aller Rechner-Umgebungen
Unabhängigkeit von der Hardware

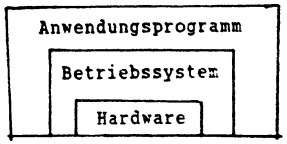
Am Anfang möchte ich Euch die Frage stellen: "Was erwartet Ihr von einem Rechner samt Betriebssystem?"

Meine Antwort: Ich möchte einen Rechner, bei dem ich mich nicht um jede Kleinigkeit selbst kümmern muß, auf dem ich schnell und einfach Programme herstellen kann, die auch auf anderen Rechnern laufen, bzw. auf dem ich problemlos Programme laufen lassen kann, die andere Leute auf anderen Rechnern geschrieben haben. Ich schätze, so dürften Eure Bedürfnisse auch aussehen; ich sehe von bekannten Ausnahmen ab, die den Rechner in- und auswendig kennenlernen wollen, nur Programme zum Eigengebrauch schreiben und auf fremde Programme ganz verzichten.

Nun sind Computeranlagen meist sehr unterschiedlich zusammengesetzt: Der eine hat das Geld für vier 1,2-Mb-Laufwerke nebst Festplatte, arbeitet grundsätzlich nur mit Laserdrucker, läßt eine Komfort-Tastatur eigens für seine Handgröße entwerfen und verwöhnt seine Augen mit einem hochauflösenden Grafik-Monitor. Der andere dagegen rödelt mit einem alten 100-Kb-Laufwerk, kann sich gerade die gehäuselose Tastatur von Bühler für 5,20 DM leisten, muß sich den Fernseher zum Monitor umbauen und hat einem ausgemusterten Fernschreiber das Drucken beigebracht. Beide möchten dieselben Programme benutzen und vielleicht sogar Programme austauschen. (Die echten Computer-Freaks kennen eben keine Skrupel bei der Überwindung von Klassengrenzen; Marx wäre schockiert.)

Da kommt unser Betriebssystem ins Spiel, mag es nun TRaShDos, Ce-Pe-eM, DoMeSDOS, oder TU-nix heißen. Man nehme ein beliebiges dieser Betriebssysteme, passe es so an, daß es einmal auf dem Turbo-GSI-Injection und einmal auf der Schrottmühle läuft, und schon können die beiden munter Programme tauschen, die mit etwas Glück auf beiden Computern laufen.

Wo ist der Trick? Ganz einfach: Jedes (Anwendungs-)Programm, das einer unser Freaks geschrieben hat, findet auf jedem der Rechner die gleiche Umgebung, nämlich das Betriebssystem. Das Programm merkt nichts davon, welcher Computer sich dahinter verbirgt. Bildlich sieht das so aus:



Das Betriebssystem "kennt" die Hardware und das Anwendungsprogramm "kennt" das Betriebssystem. Aber das Anwendungsprogramm weiß nichts über die Hardware. Das Programm hat nun meistens einige Bedürfnisse, die das Betriebssystem befriedigen muß: das Lesen und Schreiben von Texten (o.ä.) von/auf eine Diskette/Festplatte; das Einlesen von Tastendrücken des Users (einer unserer Freaks); die Ausgabe von Reaktionen darauf (meist auf einen mehr oder minder teuren Bildschirm); die Beschmutzung von Papier per Drucker; usw. Generell handelt es sich hier um Ein- und Ausgaben des Programms von/auf irgendwelche/n Geräte/n.

Das alles erledigt das Betriebssystem für das Anwenderprogramm. Das Programm sagt also (stark vereinfacht): "Ich hab' hier einen Text und der soll irgendwo so gespeichert werden, daß ich ihn später mal zurückholen kann; das Gerät soll A: sein und als Name wäre mir FILENAME.EXT recht."

Dies geschieht durch einen oder mehrere Unterprogrammaufrufe, wobei die Unterprogramme Teile des Betriebssystems sind, die von dem Programm aufgerufen werden. Nun kann das Betriebssystem nachsehen, was denn da so an Massenspeichern vorhanden ist, z.B. Diskettenlaufwerken, und welcher Massenspeicher die Bezeichnung "A:" trägt. Sollte es keinen davon geben, sagt das BS: "Nee, tut mir leid, A: haben wir heute nicht. Geht nicht was anderes?" Dann muß sich das Programm was anderes ausdenken.

Normalerweiser haben wir aber einen Massenspeicher A:; dabei ist egal, was für ein Typ Massenspeicher dies ist, also Diskette oder Festplatte oder umgebauter CD-Player o.ä. Das Betriebssystem sorgt nun dafür, daß dieser Text auf dem Massenspeicher so abgelegt wird, daß es ihn später unter dem Namen FILENAME.EXT wiederfinden kann. Je nach Betriebssystem kann/muß dieser Name und die Geräte-Bezeichnung übrigens verschieden aussehen. Später sagt das Programm dann mal: "Ey, ich hätte gern einen Text zurück, der auf dem Gerät A: steht und den Namen FILENAME.EXT trägt." In einem Prozent der Fälle bekommt das Programm den Original-Text fehlerlos zurück; streng nach Murphy tritt bei den restlichen 99% irgendein Fehler auf (Diskette im Kaffee gebadet, Bits gerade auf Reisen oder anderweitig unabkömmlich usw.).

Wie wir sehen, ist es dem Programm vollkommen schnurzpiepe, was für Gerät sich nun eigentlich hinter diesem ominösen "A:" verbirgt, solange die Daten da nur über das Ausschalten des Rechners hinaus verfügbar bleiben. Theoretisch könnte das beim Schreiben ein Drucker sein, der massenhaft Papier ausspuckt, und beim Lesen eine Tastatur, auf der eine unterbezahlte junge Dame wieder eintippt, was vorher auf das Papier gedruckt wurde. Da dies nicht so effizient ist, wurde die Idee trotz des drängenden Arbeitslosenproblems wieder fallengelassen.

Ähnlich sehen die Sachen nun für Tastatur, Bildschirm, Drucker usw. aus. Nur wird diesmal meistens nur ein Zeichen übergeben, was dann beim Bildschirm so ausieht: "Ich hab' hier ein Zeichen 'a', das der blöde User da irgendwie mitkriegen soll; schmier das mal auf 'nen Bildschirm oder so." Dabei ist es dem Programm egal, ob das Betriebssystem das Zeichen wirklich auf einen Bildschirm malt, oder auf einem Drucker ausgibt (weil der Bildschirm gerade zum 42.ten Mal kaputt ist), oder gar einen Text auf einer Diskette daraus baut (weil der User gerade auf'm Klo Geschäfte abwickelt und zu gegebener Zeit nachsehen möchte, was denn so in den letzten Stunden schiefgelaufen ist).

Für den Drucker sieht die Sache genauso aus, nur weiß das BS jetzt, daß das auszugebende 'a' etwas dauerhafter zur Verfügung stehen soll, z.B. zum Ablegen des verschmutzten Papiers in einen Briefumschlag zwecks Bereicherung der Post. Trotzdem kann sich das BS dafür entscheiden, dieses 'a' auf einem Bildschirm oder gar nicht auszugeben, wenn es das für sinnvoll hält

bzw. von dem User vorher diesen Befehl erhalten hat (der User meinte natürlich was ganz anderes).

Die entsprechende Frage für die Tastatur lautet: "Hat der User inzwischen mit seinen Fettfingern eine unschuldige Taste angefaßt?", worauf das BS antwortet: "Nee, der Junge braucht mal wieder 'ne Gedenkminute oder -stunde." oder "Ja, das arme 'a' mußte dran glauben. Hilft Dir auch nicht weiter, was?" Dabei kümmert sich unser Programm nicht darum, ob der User wirklich auf der Tastatur rumhämmert oder ob das BS sich diese Tastendrücke einfach ausdenkt.

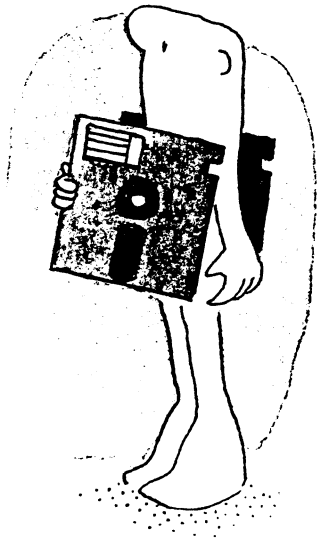
Niedlich ist es z.B., wenn der Mensch seine ganzen Tastendrücke in einem Text auf der Diskette ablegt und das BS diese nun Stück für Stück wieder rausholt und dem Programm gegenüber so tut, als hätte der User gerade seine Grabbeln an der Tastatur gehabt (während er in Wirklichkeit ganz was anderes begrabbelt). Das nennt sich "Batch-Betrieb" und der Text auf der Diskette heißt "Batch-Datei"; wenn Ihr jetzt im Wörterbuch über "Batch = Stapel" stolpert, denkt daran, daß in grauer Vorzeit mal Lochkartenstapel (statt Dateien) benutzt wurden, die sich das BS dann Karte für Karte reingezogen hat.

Wie Ihr seht, weiß das Anwenderprogramm überhaupt nicht, was denn eigentlich passiert, nachdem es dem BS die Aufgabe übertragen hat, etwas zu tun. Wenn es Pech hat, macht das BS sich einen faulen Lenz und das Programm merkt noch nicht einmal was. Das schönste Beispiel ist die Umlenkung der Drucker-Ausgabe auf ein Null-Device (das ist ein Gerät, das es nicht gibt, also eine Fata Morgana, eben nichts - null). Da vergißt das BS einfach alle Zeichen, die es eigentlich ausdrucken sollte - sagt aber dem Anwenderprogramm dauernd: "Alles klar, Kumpel, ich druck' das schon für Dich." Fragt mich nicht, was das soll, fragt unsere IBM-/MS-DOS-Freaks.

Was für einen Vorteil hat es für uns, daß das Betriebssystem alle diese Dinge (und noch mehr) für uns erledigt? Nun, wir (als Programmierer in irgendeiner Form) brauchen uns nicht darum zu kümmern, wie der Laserdrucker/Fernschreiber angesprochen werden muß, müssen nicht dauernd die Tastatur überwachen (sofern wir überhaupt wissen, wie das geht), kennen nicht jeden Bildschirm / jede Grafikkarte / jedes Terminal, sondern überlassen alle diese Sorgen dem Betriebssystem bzw. dem armen Schwein, das ein BS schreiben oder an den jeweiligen Rechner anpassen muß.

Ansonsten müßten wir jeden Fall, der irgendwann auf irgendeinem Rechner auftreten könnte, einkalkulieren und unser Programm darauf einrichten. Wir müßten jedes Gerät (Kassetten-/Disketten-/Festplattenlaufwerk, Bildschirm, Tastatur, Drucker...) kennen, wissen, wie es auf jedem Rechner angesprochen wird und was jeweils zu beachten ist usw. Das ist sogar schon bei der relativ kleinen Klasse der TRS-80/Genie-Rechner unmöglich. Wer's nicht glaubt, versuche mal ein Programm ohne jeglichen Newdos-Aufruf zu schreiben, das sowohl auf einem Original-TRS-80 als auch auf Arnulfs Genie-IIIs-Turbo läuft.

Fazit: Ohne Betriebssystem wäre unser Programmierer-Leben weitaus schwerer und wir würden alle ein Einzelgänger-Dasein fristen. Hier habe ich nur etwas dazu gesagt, wie Betriebssysteme uns helfen, daß unsere Programme sich auch auf vollkommen fremden Rechnern zurechtfinden. Mit Hilfe des Betriebssystems können sie auf jedem Rechner feststellen, welche Tasten der User gerade gedrückt hat, sie können einen angeschlossenen Drucker drucken lassen usw. Über andere Probleme werde ich mich in einem weiteren Artikel auslassen. Also: Keep listening to this radio station!



Ein neuer Diskothekar meldet sich zu Wort

Wie Ihr an anderer Stelle gelesen habt bin ich der neue Diskothekar für Neudos, HDOS, GDOS, etc.. Ich hoffe, daß ich rege Post von Euch bekomme. Am liebsten wäre es mir wenn Ihr mir Disketten 80 Spur DS/DD schicken würdet, da ich dieses Format benutze. Ihr solltet aber auf jeden Fall, bei Programmanforderungen, die Scheiben mit euerem Rechner formatieren da es sonst später Probleme bei Euch geben wird, vor allem bei 40 Tracklaufwerken. So jetzt will ich Euch noch verraten in welchen Formaten Ihr mir sie senden könnt bzw. in denen ich sie beschreibe.

80 Track DS/DD
TI=EHK,TD=G,SP=80,SEK=36,SWZ=3,EIB=6,SBIV=48,AEIV=6

40 Track DS/SD
TI=EKL,TD=G,SP=40,SEK=36,SWZ=3,EIB=6,SBIV=24,AEIV=6

40 Track SS/SD
TI=A,TD=A,SP=40,SEK=10,SWZ=3,EIB=2,SBIV=20,AEIV=2

Bei den Anforderungen von Kopien kann es etwas dauern denn ich habe noch ein paar Probleme mit der Dateiverwaltung. Ihr solltet also nicht verzweifeln wenn Ihr nicht gleich am nächsten Tag Post von mir bekommt.
Jetzt noch eine Frage, kann mir jemand bei TSCRIPS oder wie das sonst noch heist helfen.
Gehe ich richtig in der Annahme, daß es eine zu kopieren erlaubte Software ist und wie komme ich an ein für mich lauffähiges File und eine Bedienungsanleitung. Ich habe einen Genie IIS und der Drucker läuft vermutlich über einen PORT.

Am Schluß möchte ich mich bei allen bedanken die mir beim Clubtreffen Infomaterial versprochen und zahlreich zugesand haben. Ich hoffe daß ich bis zum nächsten Info alles im Griff haben werde, in Bezug auf unsere Diskothek.

Es tut sich was auf dem Sektor BDOS-Replacements. Es stehen momentan folgende Pakete zur Verfügung (aufgelistet nach dem Datum ihres Erscheinens) :

SUPRDOS2.LBR
eines der ersten Pakete, teilweise fehlerhaft, aber ist zu Ruhm gelangt durch die Tatsache, daß die meisten anderen DOSse auf ihm aufbauen.

Z80DOS10.LBR
Z8D-UTL1.LBR
direkter Nachfolger von SUPRDOS, nicht weniger Bugs.

Z80D20A.LBR
Neuere Beta-Test-Version

DOSPLS25.ARK
Das ist eigentlich ein ganzes Betriebssystem, natürlich ohne BIOS, aber mit erweitertem CCP. Erweitertes Batch-Processing, IF/THEN/ELSE/GOTO usw. eigentlich Shareware, vom Autor jetzt in die PD gegeben, inkl. Quellen. Wenn es jemand ausprobieren möchte, wäre ich für einen Erfahrungsbericht dankbar.

P2DOS23.LBR
Habe ich selbst eine Zeitlang benutzt, gut dokumentiert.

Z80DS231.LBR
Das Ergebnis der Fehlerkorrekturen von Vers. 2.0, den ^P Bug hat wohl niemand entdeckt. Der Autor hat sich große Mühe gegeben, Disk-Zugriffe, speziell Directory-Zugriffe zu beschleunigen.

NOVADOSH.LBR
Das sieht *sehr* interessant aus. Wie Z80DOS, aber zusätzlich ein Suchpfad definierbar. Natürlich auch Uhr, Datestamps, Public Files etc.

Z80DOVL.LBR
Z80D24SR.LBR
Zwei Files aus einem Paket von dreien, wobei mir die dritte noch fehlt. Die neueste Version, stammt aus der ersten Jahreshälfte 1989. ^P geht immer noch nicht. Das wichtigste sind sog. "Harte Disks", Disketten, die nicht gewechselt werden, kann man "hart" machen. Dann muß die Disk Allocation Table nicht jedesmal neu berechnet werden und das Einloggen geht schneller.

Zu allen BDOS-Replacements liegen die Quellen vor. Die Installation ist in der Regel simpel, man muß lediglivh die BDOS-Startadresse eintragen, Optionen setzen und assemblieren, mit Hilfe von SYSGEN und DDT das neue System auf die Systemtracks schaffen.
Ein Tip zum Assembler : alle Pakete sind in Zilog-Mnemonics abgefaßt. Allerdings oft für unterschiedliche Assembler, meist für den SLR. Ich benutze ZAS. Die Änderungen an den Quellen beschränken sich auf zweierlei:

Alle arithmetischen Ausdrücke, die geklammert sind. ZAS verlangt ECKIGE Klammern.

Das Einbinden einer Include-Datei sieht so aus :

```
.in  ANYCODE.LIB
```

oder

```
maclib ANYCODE.LIB
```

Die Extension der Datei *muß* .LIB sein!

Programme des Monats

Eins fällt in die Kategorie Print-Utilities und heißt BRADFORD. Unterstützt werden grafikfähige 8/9 Nadeldrucker, befehlskompatibel zu Epson FX80/MX80 oder Star Gemini. Das dürfte für die Mehrzahl der Drucker zutreffen. BRADFORD liest eine Text-Datei, die spezielle Kommandos beinhalten kann, und gibt sie unter Benutzung eigener Fonts aus. Mit Fonteditor. Nachteil : BRADFORD braucht viel TPA, läuft nicht unter Standard-Monte-CP/M. Aber unter meiner 128K-Version davon. Ein Demo-Ausdruck liegt bei.

Das zweite Programm ist ein Editor namens VDE, inzwischen Version 2.66. Er ist kompatibel zu Turbo-Pascal / WordStar. Sehr schnell, mit begrenzten Fähigkeiten als Textverarbeiter. Gut konfigurierbar, Macro-Fähigkeiten. Läßt sich memory-mappen. Nachteil : Er frißt nur Dateien bis Speichergröße.

So, das war's für diesmal. Bestellungen an :

Rüdiger Sörensen
Thomas-Mann-Straße 3a
6500 Mainz 1

Bitte legt eine Diskette ein. bevorzugt eine formatierte. Ich versende nur Monte 80 Tr DSDD SYSTEM. Punkt.

Bis denn, *Rüdiger*

Bradford

The printing program

A Concom Enterprises production

Are you tired of the low-quality printing you get from your dot-matrix printer? Then it's time to use Bradford, the printing program.

Bradford will print your ASCII and WordStar files in very high quality, using your choice of fonts (typestyles). Bradford requires no hardware adjustment and costs only $39.95. The version that you are now using is a completely functional copy, not just a crippled "demo version."

Bradford 2.0 new features include:

Up to five fonts PER line.

Much faster on many computers.
Understands more WordStar commands.
Double-height fonts (three shown above).
Automatically recognizes ends of paragraphs.
New three-pass mode on some printers.

Subscripts and superscripts.
Automatic chapter numbering.
Can overstrike two lines.
Headers and footers.
Macros.

Bradford can now do equations: $\mu_0 = 4\pi \cdot 10^{-7}$ Wb A^{-1} m^{-1}

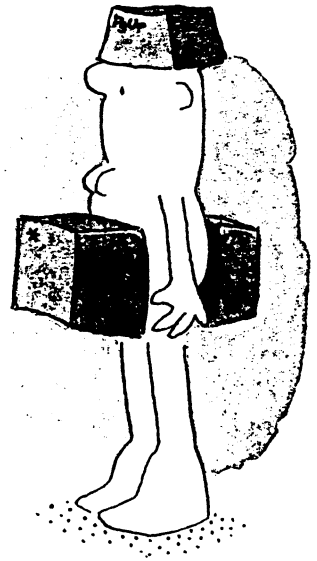
Bradford can now print Greek text: ΙΧΘΙΣ ιχθυσ

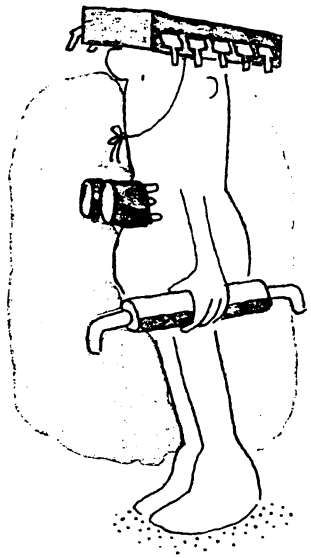
Try Bradford for a reasonable period of time. A set of basic operating instructions has been provided on the disk for you. If you like what Bradford can do for you, then it's time to pay for a "legitimate" copy. Included with this copy will be a thorough, printed instruction manual (over 50 pages long), and one or more disks containing the latest version of Bradford.

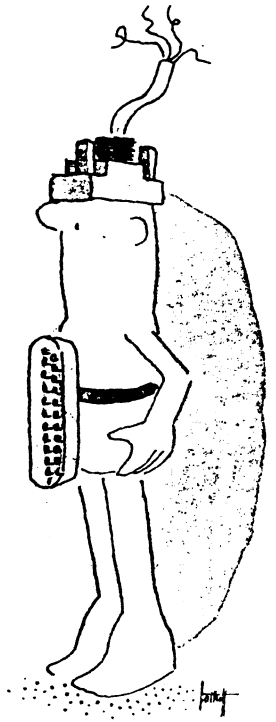
To order your own copy of *Bradford 2*, send a check or money order in U.S. dollars for $39.95 (which includes shipping) to:

Concom Enterprises
Post Office Box 5056
Champaign, IL 61820, USA

(Add $3 if you're outside North America.) Be sure to specify what type of computer you are using (so we will know what type of disks to send) and what type of printer you are using. If you do not wish to purchase your own copy of Bradford 2, you may order the manual alone for $25.







Die Betriebsart Packet-Radio (PR)

(Konzept eines Vortrags, gehalten von Hans-Günther Hartmann, DF7BX, anläßlich des Nordlichter-Treffens des Club 80 vom 20. bis 22.10.89 in Kiel)

A. Zusammenfassung

Die von den Funkamateuren genutzte, neueste Betriebsart **Packet-Radio** ist Gegenstand dieses Vortrags. Hierbei wird zunächst auf die herkömmlichen Betriebsarten im Amateurfunkdienst eingegangen. Weiter soll der Weg erkennbar werden, wie man überhaupt Funkamateur wird, welche Rechte und Pflichten ein Funkamateur hat. Nach einem kurzen "technischen" Streifzug durch die möglichen Betriebsarten im Amateurfunk soll endlich näher auf PR eingegangen werden. Wer keine Lust hat, etwas über Funkamateure und Gesetze und Verordnungen zu lesen, fängt beim Punkt C 1 an.

B. Der Amateurfunkdienst

1. Was ist Amateurfunk?

Nach der Vollzugsordnung für den Funkdienst (VO Funk) ist der Amateurfunk ein Funkdienst, der von Funkamateuren für die eigene Ausbildung, für den Verkehr der Funkamateure untereinander und für technische Studien wahrgenommen wird. Funkamateure sind ordnungsgemäß ermächtigte Personen, die sich mit der Funktechnik aus rein persönlicher Neigung und nicht aus wirtschaftlichem Interesse befassen.
Für die Verbreitung eignen sich nur Meldungen von geringem Nachrichteninhalt sowie technischem Inhalt. Der Amateurfunk ist kein Ersatz für das Telefon.
Der Amateurfunk existiert weltweit.
Es gibt rund 1 Million Amateurfunk-Stationen auf der Erde, nahezu 57000 befinden sich in der Bundesrepublik.

2. Wie wird man Funkamateur?

Funkamateur kann werden, der
- seinen Wohnsitz in der Bundesrepublik Deutschland einschl. Berlin(West) hat
- mindestens 18 Jahre alt ist (Ausnahmen möglich)
- nicht vorbestraft ist
- eine fachliche Prüfung für Funkamateure abgelegt hat

a) Anforderungen der Deutschen Bundespost an die Fachkunde

Im Rahmen der fachlichen Prüfung werden Kenntnisse in folgenden Bereichen erfragt:
- Kenntnis von Vorschriften
- betriebliche Kenntnisse
- technische Kenntnisse
- Morsekenntnisse

b) Lizenzklassen

Es gibt hier z.Zt. 3 Lizenzklassen: A, B und C.

Die "große" Lizenz der Klasse B erlaubt den Amateurfunkbetrieb auf allen freigegebenen Frequenzen auf KW und UKW in allen Betriebsarten.
Die Klasse A erlaubt eingeschränkten Betrieb auf KW, schränkt den Betrieb auf UKW jedoch nicht ein.
Die "kleine" Lizenz der Klasse C erlaubt nur den Funkbetrieb im UKW-Bereich. Für diese Klasse sind keine Morsekenntnisse erforderlich.

3. Betriebsarten im Amateurfunk

Neben den herkömmlichen Betriebsarten wie Telegrafie (Morsen; hierbei wird die Information in Form eines Punkt/Strich-Codes übermittelt)und Telefonie (Sprechfunk; hier gibt es verschiedene Modulationsarten wie wir es alle aus dem Rundfunk kennen: AM und FM)gibt es mittlerweile eine Vielzahl von "moderneren" Betriebsarten, die nachfolgend kurz erläutert werden sollen.

a) RTTY (Radio Tele Typing)

RTTY, auch Funkfernschreiben genannt, ist die am weitesten verbreitete Betriebsart.
Zunächst mit ausgemusterten Fernschreibern betrieben, wird heute immer mehr mit Hilfe von Computern Funkfernschreibverkehr betrieben. Platzbedarf und Lärmpegel konnten so erheblich verringert werden.
RTTY ist sehr nahe mit Btx verwandt. Der digitale Code eines Zeichens wird in Töne umgesetzt, per Funk ausgesendet, beim Empfänger wieder decodiert und auf den Bildschirm des Computers gebracht. Die Übertragungsgeschwindigkeit auf UKW liegt wie beim Btx auf 1200 Baud. Im herkömmlichen Funkfernschreiben liegt die Geschwindigkeit bei 45,45 Baud.

b) ATV (Amateur Television)

Das ATV ist die mit dem höchsten technischen Aufwand verbundene Betriebsart. Aufbereitung und Aussendung eines ATV-Signals erfolgt nach den gleichen Prinzipien und Normen wie beim öffentlichen Fernsehen.
Wegen der erforderlichen Bandbreite von 5,5 MHz eignet sich diese Betriebsart nur zur Aussendung im UKW-Bereich (70 cm-Band). Die verwendeten geringen Leistungen von ca. 10 W erfordern drehbare Richtantennen.

c) SSTV (Slow Scan Television)

Entgegen der Möglichkeit, die beim Fernsehen üblichen bewegten Bilder zu übertragen, kann man beim SSTV - auch Schmalbandfernsehen genannt - nur stehende Bilder übertragen. Die Reichweite ist allerdings nahezu unbegrenzt, da die Signale üblicherweise auf KW übertragen werden.
Die von Funkamateuren entwickelte Technik erlaubt das Abspeichern eines normalen Fernsehbildes und späterer Aussendung mit 120 Zeilen. Die Übertragung eines kompletten Bildes dauert rund 8 Sekunden.

d) FAX (Faksimile-Funk)

Mittels des Faksimile- oder Bildfunks kann man vom sendeseitigen Original über Funk auf der Empfangsseite eine Kopie anfertigen.
Man erhält also auf einem Papier eine Hard-Copy.
Als populärstes Beispiel sei das Aufzeichnen von Aussendungen der Wettersatelliten erwähnt.
Neben der Elektronik ist der Anteil der Elektromechanik nicht unerheblich, damit derartige Aussendungen auf das Papier gebannt werden können. FAX-Geräte sind fast nicht zu kaufen, daher muß der Funkamateur fast alles selber bauen.

e) Amateurfunk-Satelliten

Der erste Amateurfunksatellit wurde im Jahre 1961 gestartet. Seitdem hat diese extrem kostenaufwendige Betriebsart einen "raketenhaften" Aufstieg hinter sich. Waren zunächst noch Batterien an Bord, so wird die Stromversorgung mittlerweile mittels Solarzellen und aufladbaren Akkus sichergestellt.
1983 erfolgte der Start des ersten deutschen OSCAR.
Mit Hilfe dieser Satelliten ist es selbst Amateuren mir der Lizenzklasse C möglich, weltweite Funkverbindungen herzustellen.
Auch Packet-Radio und ATV können über Satelliten abgewickelt werden. Zum Betrieb über Satelliten sind horizontal und vertikal drehbare Richtantennen nötig.

f) AMTOR (Amateur Teleprinting Over Radio)

Amtor eignet sich hervorragend zur Datenübertragung selbst unter schwierigsten Bedingungen auf Kurzwelle. Hiermit ist praktisch fehlerfreie Datenübertragung zwischen entsprechend ausgerüsteten Amateurfunkstationen möglich. Die Fehlerfreiheit geht allerdings auf Kosten der Übertragungszeit.
Im Gegensatz zum asynchronen 5er-RTTY-Code verwendet Amtor die synchrone Aussendung eines 7-Bit-Codes.

C. Die Betriebsart Packet-Radio (PR)

1. Allgemeines

PR ist eine noch sehr neue Betriebsart im Amateurfunk. PR ermöglicht eine gesicherte Übertragung von Texten über Funk.

PR kann man auf KW (sehr langsam ==> 300Bd) und auf UKW betreiben.
Die Übertragungsgeschwindigkeit beträgt im allgemeinen zwischen den Funkgeräten 1200 Baud, während Computer und Modem mit 9600 Baud kommunizieren können.

Der Hauptzweck beim PR-Betrieb muß die Beschäftigung mit Datenübertragungstechniken sein. Daher wird Binärfile-Übertragung auch nicht gerne gesehen. Hier ist zumindest zweifelhaft, ob es sich dabei noch um die vorgeschriebene Übertragung von "offener" Sprache handelt.

2. Der Netzaufbau am Beispiel DB0IL in Kiel

PR hat eine so rasante Entwicklung genommen, daß heute schon fast ganz Europa mit Umsetzern -sogenannten Digipeatern --> Digi von Digitaltechnik; Repeater von Umsetzer- versorgt ist.
Diese Umsetzer sorgen für eine größere Reichweite meiner Aussendungen, die im UKW-Bereich sonst höchstens Stationen im Umkreis von 30 km erreichen würden. Dies ist natürlich abhängig vom Antennenaufwand.

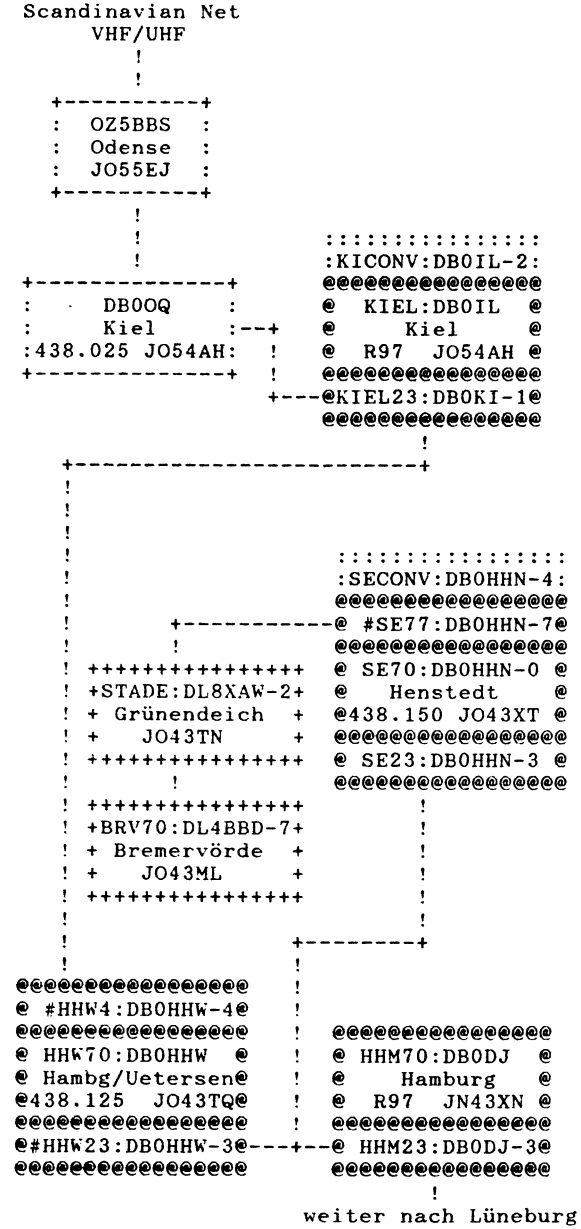
Mit Hilfe dieser Digipeater also kann ich PR-Verbindungen auf UKW von Norwegen bis nach Sizilien oder Spanien aufbauen.

Im Kurzwellenbereich ist die Sendung weltweit zu hören. Hierfür sind dann auch keine Umsetzer nötig.

In Kiel steht z.B. der Digipeater mit dem Rufzeichen DB0IL. Daran angeschlossen ist eine Mailbox, die in etwa 60 Rubriken zu allen möglichen Themen Dateien gespeichert hat. Diese Dateien können per Funk ausgelesen und dann auf meinem Datenträger, sei es nun Diskette oder Festplatte, gespeichert werden.

Weiter unten ist kurz aufgezeichnet, welche Verbindungen von Kiel aus möglich sind. Einerseits kann ich in Richtung Dänemark/Schweden versuchen, Verbindungen aufzubauen, andererseits ist der Aufbau einer Verbindung Richtung Süden über Lüneburg oder Hamburg möglich. Von jedem Digi gehen spezielle Abzweigverbindungen per Funk zu anderen Umsetzern.

Seit kurzem sind auch Funkamateure in der DDR in PR betriebsbereit und haben ihrerseits schon Digipeater aufgebaut.



Jeder Digipeater weiß, welche anderen Umsetzer er erreichen kann. Diese Information kann ich abrufen und mich dadurch in die gewünschte Richtung durcharbeiten.

Nehmen wir einmal an, daß ich von Kiel aus nach München möchte. Dazu brauche ich bei bestehender Verbindung von Kiel aus nur den Befehl "C DB0MWE" (DB0MWE ist das Rufzeichen des Digipeaters in München) einzugeben und schon weiß der Digi, was er zu tun hat. Nach etwa 3-5 Minuten bin ich in München angekommen, was mir der dortige Digi an meinem Bildschirm mit einer Meldung anzeigt.

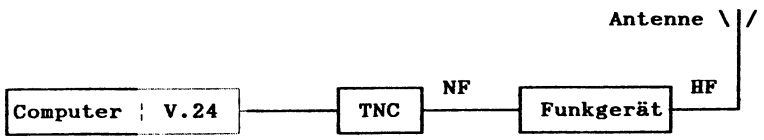
a) Digipeater,Links, Gateways

Nach Vorgaben des Deutschen Amateur-Radio-Clubs (DARC) muß der **Link** zwischen **Digipeatern** im 23cm-Band (1240 - 1300 MHz) erfolgen. Als Einstieg ist das 70cm-Band (430 - 440 MHz) vorgesehen.
Linkverbindungen laufen mit einer Leistung von 0,5 W bis ca.5 W. Als Antennen werden Richtantennen mit bis zu 20 dB Gewinn verwendet. Es gibt heute schon Richtfunkverbindungen, die im Vollduplexverkehr 9600 Bit/s übertragen.

Zu Verbindung von absolut inkompatiblen Netzen, die also mit unterschiedlichen Übertragungsprotokollen arbeiten, werden sogenannte Gateways benötigt. Jede PR-Station kann einen Gateway darstellen; z.B. zur Verbindung von Stationen einerseits mit AX.25-Protokoll und andererseits mit TCP/IP-Protokoll.

3. Was benötigt man für PR?

a) Hardware



Benötigt wird ein Computer mit V.24-Schnittstelle, ein TNC (Terminal **N**ode **C**ontroller), ein Funkgerät für das 70 cm-Band sowie eine Antenne.
Der TNC, der einfach an den NF-Ausgang des Funkgerätes angeschlossen wird, ist ein einfacher Computer mit Z80A-CPU, EPROM-Software, einer RS 232-Schnittstelle mit Z80A-SIO/0 und einem Modem mit dem IC TCM 3105.

Für manche Rechner (C64, Apple) gibt es PR-Software, so daß der TNC entfallen kann.

b) Software

Die erforderliche Software ist natürlich abhängig vom verwendeten Rechnertyp. Auf alle Fälle wird ein Terminalprogramm benötigt, das im Polling- oder Interrupt-Modus arbeiten kann. Nach meiner Erfahrung mit dem TRS 80 Model 4p kann ich sagen, daß die Verwendung von Programmen mit Polling-Modus die Transfergeschwindigkeit sehr verlangsamt. So ist z.B. der PR-Betrieb mit dem Model 4p nur mit 1200 Bd möglich, während sonst Übertragungsraten von 9600 Bd üblich sind.

Die Software ist nur lauffähig, wenn ich vorher mein Rufzeichen eingegeben habe. Hiermit soll verhindert, daß Nichtlizensierte diese Betriebsart anwenden.
Software gibt es für alle gängigen Rechner. Der Komfort der Programme für PC's und Atari-Rechner ist durchaus beachtlich. Der für mich größte Vorteil dieser komfortablen Programme liegt in der dreifachen Aufteilung des Bildschirms.
Die Größe dieser 3 Bereiche kann mit den Cursortasten verändert werden.
Der obere Bereich ist als "Vorschreibschirm" gedacht, nach Drücken der Return-Taste wird der oben geschriebene Text ausgesendet und erscheint im mittleren Bildschirmteil, in dem übrigens die für die eigene Station bestimmten Pakete erscheinen. Der untere Teil des Schirms ist als Monitorkanal zur Überwachung der Frequenz gedacht. Hiermit kann der Betrieb auf der Frequenz beobachtet werden.
Diese Programme sind von Funkamateuren für Funkamateure geschrieben worden und ausdrücklich Public Domain.

Diese eben beschriebenen Vorteile haben die mir bekannten, für uns als TRS80-User verfügbaren Programme leider nicht, weder in den Systemen NEWDOS/GDOS (GDOS habe ich hier noch eingefügt, nachdem ich aus Kiel zurückkam, weil mir Arnulfs Rechner mit Festplatte so imponiert hatte) noch unter CP/M!
Entweder muß hier ganz auf die Monitorfunktion verzichtet werden (was natürlich möglich ist, gleich bei welchem Rechner), oder fremde und eigene Pakete kommen gemischt auf meinen Bildschirm; es ist dann nur noch mit Mühe ein für mich bestimmter Text lesbar.

Angeblich soll das beste Programm für CP/M "Modem7" sein. Ich habe bisher aber nicht geschafft, die notwendigen Parameter einzustellen, so daß ich zu diesem Programm nichts sagen kann.

Was sich hier so (hoffentlich) schön liest, war dann leider in der Praxis doch nicht so schön, weil die Funkverbindung zur Mailbox recht mickerig war. Das lag beileibe nicht an der Unterkunft -hier war alles bestens organisiert-, sondern an der mit 1,5 Watt geringen Sendeleistung meines Funkgerätes und der nur ca. 15 cm langen Antenne. Aber die Hauptsache war, eine Verbindung aufzubauen ,das Prinzip zu erläutern und einmal mehr darzustellen, was man noch alles mit einem Computer machen kann außer Briefe schreiben.

Zum Schluß, Dir, lieber Helmut, nochmals herzlichen Dank für die Organisation des Treffens. Es hat mir sehr gut gefallen!

I M P R E S S U M

1. Vorsitzender
Alexander SCHMID
St.Cajetan Str. 38 VII
8000 München 80
☎ 089 /495326

2. Vorsitzender
Horst-Dieter SCHROERS
Breslauer Str. 9
8016 Feldkirchen
☎ 089 /9032615

Hard-warekoordinator
Andreas MAGNUS
Pommernstr. 4
4650 Gelsenkirchen
☎ 0209 /870230

NEWDOS-Diskothekar
Oliver VOLZ
Waldburgstr. 73
7000 Stuttgart 80
☎ 0711 /7353817

CP/M -Diskothekar
Rüdiger SöRENSEN
Thomas-Mann-Str. 3a
6500 Mainz 1
☎ 06131 /32860

Club-Bücherei
Christian MENK
Ollsener Str. 52
2116 Hanstedt
☎ 04189 /78251

Redaktion
Jens NEUEDER
Gschlachtenbretzingen
Rudolf-Then-Straße 32
7178 Michelbach /Bilz
☎ 0791 /42877

Autoren
Die Redaktion bedankt sich bei den im INHALTSVERZEICHNIS genannten Autoren für die Mitarbeit an der Club-INFO.

Bankverbindung
des CLUB 80
Postgirokonto
Sonderkonto CLUB 80
Obermann,H. 6209 Heidenrod
Konto-Nummer 496 071 - 605
Postgiroamt Frankfurt
BLZ 500 100 60

Eine Zensur oder Kontrolle der Infobeiträge erfolgt nicht.
Die Redaktion

S C H L U S S

Hallo Club-80er,

endlich kann ich Euch nun wieder einmal ein umfangreiches INFO präsentieren. Dafür möchte ich mich bei Euch, den Autoren, bedanken.

Nachdem beim letzten Termin -zu dem ich Euch leider kein aktuelles INFO liefern konnte- recht wenig an INFO-Material vorhanden war, habt Ihr nun recht kräftig aufgeholt und ca. 4/5 dieses Club-INFO's angeliefert. Ich hoffe und wünsche es uns, daß es auch in Zukunft so weiter geht. Es wäre ja schade, wenn die Kommunikation im Club abreißen würde.

Als nächsten Redaktionsschluß möchte ich Mitte/Ende Dezember ansetzen, sodaß ich Euch nach dem Jahreswechsel das nächste Club-INFO zusenden kann.

Weiterhin habe ich noch eine Bitte an Alle, wenn Ihr Artikel für unser INFO einsenden möchtet:
- Bitte verwendet kein Recycling-Papier, da es beim Verkleinern kein so guten Kontrast zum Text abgibt.
- Bitte haltet die Zeilenzahl von 60 Zeilen ein. Ich brauche auch ein bißchen Rand.

Für Eure Mitarbeit danke ich im voraus.

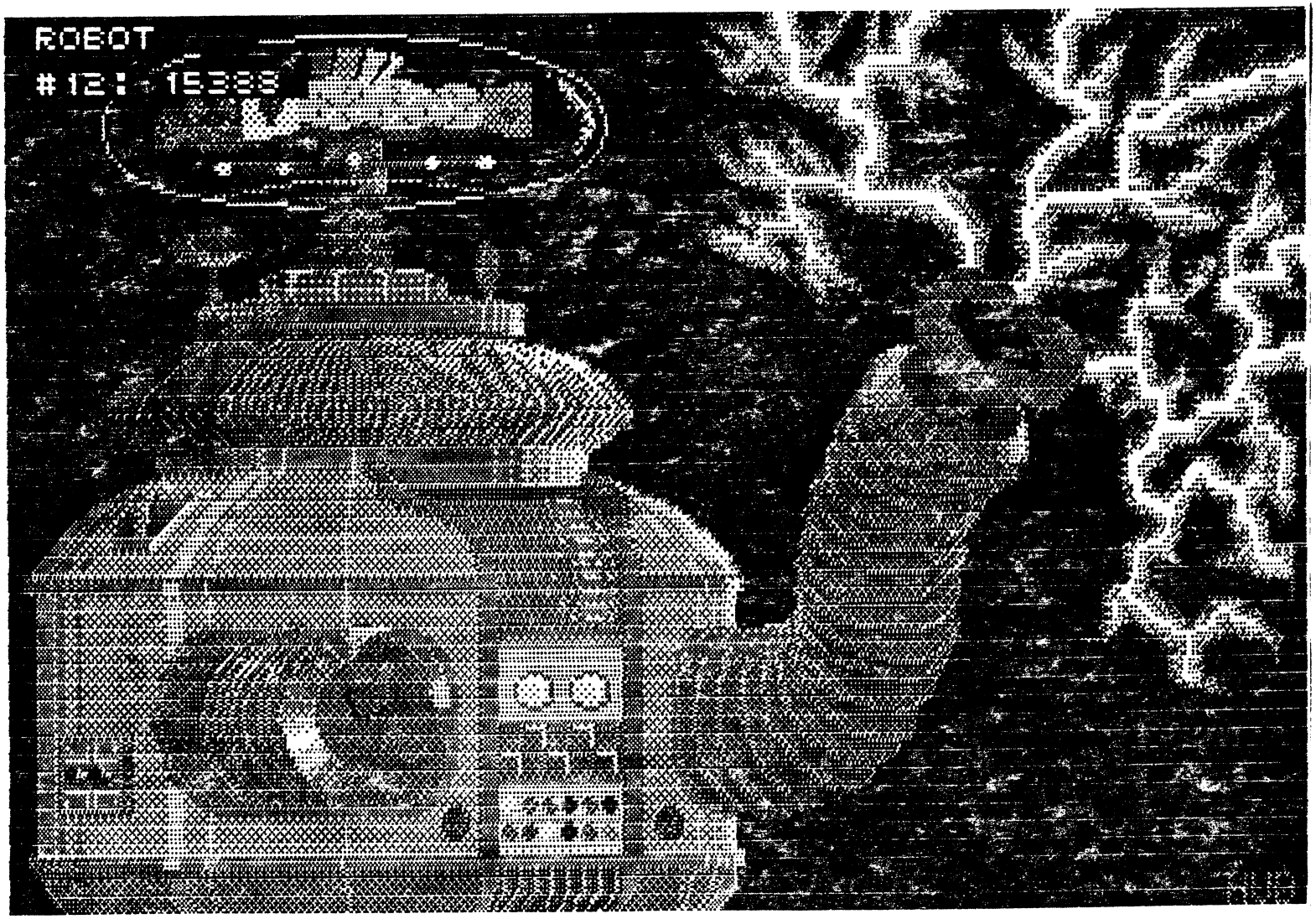
Als Beilage zu diesem INFO findet Ihr einmal das Sonderinfo
Floppy-Controller 1771 / 179x / 279x
sowie BIOS- / BDOS- Funktionen.

Nun wünsche ich Euch viel Spaß damit, an der Computerei und dem Trubel der kommenden Tage.

Es grüßt Euch Euer

Jens

HEFT 20 Oktober 1989


ROBOT
#12: 15388