

CLUB 80

Clubinfo
der

TANDY –
GENIE –
und KOMTEK –
ANWENDER

15. AUSGABE



Vergangenheit

... eine Schöpfungsgeschichte?

..... Zukunft



... wahrscheinlich ein Kultgegenstand einer untergegangenen Zivilisation!

Clubinternes

Fragebogenauswertung	81 - 83
Termine / Gemischtes	84
ECB-Bus-TRS-80 allgemein	85
Vorstellung eines Mitgliedes	86

Software

Programmierwettbewerb	87
Kreis ohne Trigonometrie	88
Logische Operationen in BASIC	89 - 10
HRG-Bildübertragung ohne HRG	11 - 12
Berichtigung: Mein erstes	14
Mein 2. ASSEMBLER-Programm	15 - 20
DIR-/ INHALT- SYS ?	21 - 24
The Disappearing DOS	25 - 26
Lib-Befehl: GO(,adresse	27 - 28
Tscrips-Verschlüsselung	29 - 31
Ach du liebe Zeit!	32 - 36
Nachtrag: Ach du liebe Zeit	37 - 38
M4/4p Grafikshorty	39
M4/4p Grafikprogramm	40
M4/4p TANDY-Logo	41 - 42
M4/4p ROM-Image-Loader	43 - 46
M4/4p UHRUNTEN	47 - 48
CALL um die Ecke	49 - 50
Korrektur: CALL um die Ecke	51 - 52
Real Time Black Box?	53 - 56
RPNL- Programmiersprache	57 - 64

Hardware

ECB-Adapterplatine	65 - 70
ECB-Bus-Projekt: Grundeinheitenaufbau	71 - 72
ECB-Bus-Projekt: Eingangsplatine	73 - 76
ECB-Bus-Projekt: ZK01/02	77 - 83
ECB-Bus für M 3/477	84 - 86

Börse

Wer hat was ???--Wer sucht was ???	87 - 88
Wer kann weiterhelfen	87 - 88

Sonstiges

Norung und Rechtschreibung	89 - 90
Sprach-Entwicklung	92

Programmbibliothek

Einleitung	93
Public-Domain-Software	94 -106

Die letzten Seiten

Impressum	107
Schluß	108
Clubmitgliederadressen	am INFO-Ende
Eingangskarten-Layout	am INFO-Ende
ZK01/02-Layout	am INFO-Ende

Nachdem ich Euch diesmal an dieser Stelle, wie sonst üblich, kein Vorwort von unserem 1. Vorsitzenden bieten kann, möchte ich den Platz nutzen und die schon lange fällige Fragebogenauswertung bringen. Ich hoffe, daß genügend Informationen, für Euch, darin enthalten sind. Leider hatte ich keine Zeit mehr, die Antworten zu jedem Thema, zu sortieren.

FRAGEBOGEN MITGLIEDERDATEN UND -INTERESSEN

AUSWERTUNG

Es wurden durch das Club-INFO 53 Fragebogen an die Mitglieder versandt.
Davon kamen bis zum Auswertungstag, den 31.08.1986, 29 Stück zurück.

2. Hardwaredaten :

Computer : TRS-80 MI 7x, TRS-80 MIII 1x, TRS-80 M4/M4p 5x
VG EG3003 2x, VG I 10x, VG II 6x, VG IIs 1x, VG III 2x, VG IIIs 1x
Kontek 1 1x, MC CP/M, ECB-TRS-80 1x

Drucker : EPSON MX 80 FT 3x davon mit Grafrax 1x, RX 80 FT 2x, TA DRH 80 2x
NEC 8023B-C 2x, ITOH 8510A 3x, DP 510 3x, STAR RADIX 10 1x, SG-10 1x
Quen Data DMP-1180 1x, HEATH H14 1x, CP80 1x, Gemini 10X 5x
Tandy LP VI 1x, LP VIII 1x, SD 4035 1x, KX-P1091, FX 80 1x

Laufwerke : 40 Trk,DS,DD 14x, 40 Trk,SS,SD 7x, 40 Trk,SS,DD 11x
80 Trk,DS,DD 39x, 35 Trk,SS 1x
Teac 55A+55B, SA 405 1x, SD 5211x, ohne Angabe 2x

HRG : 640 x 240 1x, 384 x 192 13x, 256 x 512 1x, 480 x 192 1x, IIs-HRG 1x

CP/M : 12x davon Montezuma 1x, 2.2 5x, 3.0 1x, diverse 2x

Telefon-Modem : Tandy AC3 2x, Dataphon s21d 1x, RB 300 1x, ohne Angabe 1x

Modemfahrung: JA 3x, gering 5x

sonstiges Zubehör : RS 232 2x, Prommer 4x, Spooler 64K 2x
Exatron Stringy-Floppy 2x, Gripp II 1x
Z 80-Zeichenkarte 2x, V24 1x, 3.54 MHz 3x
Banker: 256K RAM, ROM/RAM 2x
I/O /RAM 1x, Grafik 512 x512 1x, Basiccode 2x
Analog/Digital -Wandler 1x, Supertape 2x
Schnittstellen 1x, Joystick 2x, Lightpen 1x
Video-Snoval 1x, ECB-Bus 1x, Inverse Zeichend. 1x
Meßwertaufnahme 1x
Nixdorf Kugelkopfdruker nicht anschlussfertig 1x
Telefax-Gerät 1x, Typenradschreibmaschine 4x

3. Deine Interessen :

Mein Hauptinteresse liegt bei: Software 17x Hardware 4x Beidem 8x

Ich "computere" seit:

2 J. 2x, 3 J. 7x, 4 J. 6x, 5 J. 5x, 6 J. 5x, 7 J. 1x, 8 J. 2x, 9 J. 1x

Im Moment beschäftige ich mich mit folgendem:

Software : HRG-Software 4x davon M4 1x, HRG1B 2x, VG IIs 1x
alles 2x, Z80 Assembler 2x, RPNL-Compiler 1x, mathematisches 2x
Analyse + Änderung VG IIs 1x, NEWDOS 2x, C 1x, CP/M 1x
TURBO-PASCAL 2x, Programmanpassung MI auf M4/4p/3 1x
Spiel / Postspiele 1x, 80-Zeichenanwendung 1x, Textprogramme 3x
DBASE 1x, Fortran 1x, Bau-Programme 1x, Vereinsverwaltung 1x
Datenverwaltung(Sport) 1x, Programmiersprachen 1x
Datenaustausch C64/TRS-80 1x, Terminalprogramme 1x

Hardware : ECB-Bus 5x, Telefax Signale auf HRG 1x, Reperaturen 1x
Erweiterung eines Z80 Selbstbaues 1x, HRG1B-Selbstbau 2x
VG III aufwärtskompatibler Selbstbau 1x, EPROM brennen 1x
Analog/Digital 1x, Steuern/Messen 2x, Echtzeit-System 1x, RS232 1x
Spooler 2x, Direktgekoppeltes Modem 1x, Uhr im 4p 1x

Allgemein : Programmanwendungen 3x, CP/M Software 2x, Computer + Recht 1x
Maschinensprache 1x, BASIC 1x, Elektronik 1x, Modellfliegen 1x
Altbier 1x, alles 1x, wenig 1x

Ich würde mich für folgendes interessieren:

Software : HRG-Software 7x davon M4 1x, HRG1B 4x, VG IIs 1x, Drucker 1x
Utilitys 3x davon M4 1x, CP/M 3x, Amateurfunk allg. 1x
Softwareschutz/-Knack 1x, Funktionsbelegung 1x
dt. Anleitung Mumath 1x, Fortran 80-Anwendungen 1x, alles 2x
gebanktes CP/M 3.0 f. VG I und a. BANKER 2x, Videodat 300 1x
Mathematische Probleme 2x, 80-Zeichen 1x, Lisp 1x, Ram-Disk 1x
Simulationen 1x, Lehrprogramme 1x, ständiger Hintergrundmonitor 1x

Hardware : ECB-Bus-TRS-80 selbstbau 5x, ECB-Bus 3x, Modemselbstbau 1x
ECB-Adapter 1x, Meßwertfassung 2x, Maus 1x, HRG 2x, Drucker-HRG 1x
prellfreie Tastatur 1x, Videodat 300 1x, VG-Ausbau 1x, alles 2x

Allgemein : PASCAL 1x, Funktionen f. Grafik 1x, Visicalc-Tastenbelegung GIII 1x
68020 CPU 1x, Pils 1x, alles 2x

Ich fände es gut, wenn es in unserem Club eine
Assembler-Gruppe 2x, CP/M-User-Gruppe 2x, HRG-Ecke 4x, Lisp-Ecke 1x
BASIC-Gruppe 1x, Modem-Ecke 2x, (Norddeutsche) Hardware-Gruppe 2x
Anwender > VG I 1x, Textverarbeitungs-Ecke 1x, Schutz/Knack-Ecke 1x
Mathe/Naturwissenschaft-Ecke 1x, Anfänger-Ecke 1x geben würde.

sonstiges : disassemb. Dos-Listing ab SYS1 1x
öftere Clubtreffen 1x, regionale Treffen 1x

HEFT
15
September
1986

12

4. Clubleben:

- Wie siehst Du Dich als Clubmitglied ?
 mehr passiv 10x mehr aktiv 13x unentschieden 6x
 Welche Themengruppen würden Dich in der Info interessieren ?
 Bitte prozentual angeben (Wichtigkeitswertung) !!
- a) Grundlagen 556,95%: Allgemeines 15x, Basic 12x, Maschinensprache 21x
 b) sonstiges: Hardware 2x, Programmstrukturierung 1x
 DOS-Vergleich 1x, Hardwareelektronik 1x
 Ram-Erweiterung 1x, BANK-Switching 1x
 c) Allgemeines 282,53%: Neuigkeiten 21x, Unterhaltendes 12x
 d) Clubinternes 221,07%: was besonders: Computer + Recht 1x
 prakt. Erlebnisse 3x (Kauf, Test)
 Börse 1x, Gruppenarbeit 1x
 Bereichstreffen 1x, wer macht was 2x
 Hardware der Mitglieder 3x
 e) Programmiersprachen 526,01%: welche: FORTH 3x, BASIC 7x, PASCAL 13x, C 5x, COBOL 2x
 FORTRAN 5x, RPNL 1x, Z80-Assembler 8x
 alles 1x
 f) Software 489,77%: Thematik: Grafik 6x, Utilitys 4x, Mathematik 3x
 Programmiertechniken 2x, Hardware-Treiber 1x
 Dos-Patches 1x, Datenbanken 1x, Softwarelösungen 1x
 Erklärungen 1x, Verbesserungen 2x, Assembler 1x
 Betriebssysteme 1x, Modem 1x, Banker 1x, allgemein 1x
 Programmvorstellungen 2x, Anwendungen 1x, Physik 1x
 Dateienverwaltung /Datenbanken 1x, Terminalsoftware 1x
 Simulationen 1x
 g) Hardware 523,67%: Thematik: ECB-Bus (alles) 6x, DFÜ 1x, RAM-Disk 2x
 Interface 1x, Verbesserungen 3x, Modem 2x
 kleinere/mittlere Projekte 2x, Kompatible 1x
 Amateurfunkweiterungen 1x, alles 4x
 Messen/Regeln/Steuern 1x
 Computerkommunikation 1x
 h) Dein Vorschlag dazu: TIPS/TRICKS 1x, Fragebogenbeantwortung 1x
 mehr eigene Beschreibungen/Kritiken d. Mitglieder zu
 Hard und Soft 1x, Lehrg. Künstl. Intelligenz 1x

--- einmal keine Angaben deshalb nur 2000% von 29 Antworten (100% freibleibend)
 Daraus ergibt sich ein effektiver prozentualer Anteil (umgerechnet/gerundet):
 a) 19,89% -- c) 10,09% -- d) 7,9% -- e) 18,79% -- f) 24,63% -- g) 18,7%

- Hattest Du Lust dich in irgendeiner Art und Weise
 für den Club zu engagieren ? Ja 22x Nein 4x Keine Angabe 3x
 Wenn ja, wobei: Papierbeiträge 5x, kleine Hardware-Probleme 1x
 Anfängerkurs Hardware 1x, Programmiersprachen 1x
 Hardware 5x, je nach Thematik 1x, Anleitungen 1x
 Software 5x, Übersetzungen 1x, laufende Clubarbeit 5x
 Was gefällt Dir an der INFO nicht ? Keine Angaben 9x
 Lesbarkeit der Listings 10x
 " " Tel.-Nr. auf der Titelseite 1x
 teilweise falsche Rechtschreibung 1x
 zu viele Zeitschriftenartikel 2x
 Klebebindung 1x, zu wenig Autoren 1x
 nicht computerbezogene Witze 1x
 teilweise zu Systemspezifisch 2x
 zu wenig Hardware 1x
 uninformative Überschriften 1x
 Was gefällt Dir an der INFO ? bis auf ... in Ordnung 16x
 gutes Layout 9x, hohes Niveau 2x
 gute Autoren 1x
 computerbezogene Witze 2x

-- Termine -- Termine -- Termine -- 04

Hobbytronik in Stuttgart vom15. - 19. Oktober 1986

Mir treffen uns am Sonntag den 19.10.86
 jeweils 2-stündig an der Information am
 Eingang. Erster Treff gleichzeitig mit
 Beginn der Eröffnung um 9.00 Uhr.

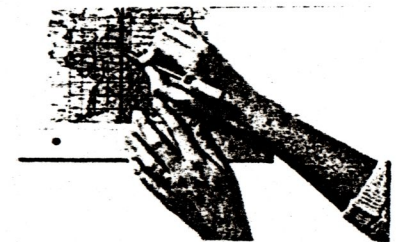
Platinenbestellschluß für ECB-TRS-80 ..12. Oktober 1986

Nächster Redaktionsschluß31. Oktober 1986

Norddeutsches Regionaltreffen25. + 26. April 1987

Aus beruflichen Gründen ist Hartmut Obermann
 die Woche über nicht zu erreichen!
 Dies gilt von jetzt bis zum 27.11.86
 Wer noch Post von ihm zu bekommen hat muß sich
 leider etwas gedulden. Telefonisch kann der
 Hartmut samstags erreicht werden.

HRG-Pack-Tip !!
 Möchte man zu HRG-Pack ein spezielles
 BASIC geladen haben (z.B. Basic/10),
 wird RUNTIME mit Extension IV eingegeben.
 Ulrich Böckling



**Mit einem Elektronikstift gibt
 man eine Landkarte auf einem
 »Tablett« in den Computer ein.**

Digitizer

Um dem Computer schnell
 und zuverlässig zu sagen, was
 er auf dem Bildschirm zeichnen
 soll, gibt es »Digitizer«, Digitali-
 sierungsgeräte, die eine ana-
 loge Zeichnung in Bruchteilen
 von Sekunden in digitale Si-

gnale umwandeln. Diesen Code
 des Zeichentablets versteht der
 Computer und läßt die entspre-
 chenden Bildpunkte auf dem
 Monitor aufleuchten. Doch wo-
 her weiß das Zeichenbrett, wo
 ich gerade meinen Stift ansetze?
 Der Stift ist mit einem Radiosen-
 der vergleichbar. Eine Spule
 sendet elektromagnetische Wel-
 len aus. Das Tablett ist die An-
 tenne. Unter der Oberfläche
 liegt ein dichtes Netz feinsten
 Antennendrähte, die von Chips
 abgefragt werden. Wo der Emp-
 fang am stärksten ist, da muß
 der Griffel gerade zeichnen. ★

Nach dem Bestellauftrag -für die Grundeinheit- im INFO 14 haben sich zehn Clubmitglieder gemeldet. Acht von Ihnen sind aktiv in die Sache eingestiegen. Für diese acht "Mutigen" habe ich die entsprechenden Bauteile Anfang September bestellt und bekommen bzw. selbst beim Hersteller besorgt. Die Bauteileauslieferung erfolgt nach dem 28. September, da ich erst zu diesem Zeitpunkt die Netzteile vom Bernd bekommen kann.

Zwischenzeitlich sind auch die ersten Anleitungen für unser Projekt fertig. Einmal ist dies der Grundeinheitsaufbau und zum Anderen die ersten beiden Platinen: Eingangsplatine TRS-80 und ZK01/02

Beide Karten dienen jetzt im Moment in erster Linie zur Überprüfung des Bus, sind aber später auch weiterhin verwendbar.

Die Eingangsplatine stellt die Verbindung zwischen dem schon vorhandenen Computer und dem ECB-Bus her. Dadurch hat man die Möglichkeit, die restlichen neun Steckplätze der Busplatine, auf die anliegenden ECB-Bus-Signale zu überprüfen oder später diese Steckplätze mit den in den folgenden INFO's vorgestellten Zusatzkarten zu belegen. Eine davon wäre zum Beispiel die in diesem INFO vorgestellte ZK01/02. Auch "Selbstgestricktes" ist durch Bus und Eingangsplatine an den "alten" Computer dann anschließbar.

Zur Überprüfung des Bus ist es notwendig, jeden freien Kartenplatz einmal auszutesten, dies geschieht am einfachsten mit den Testprogrammen von Seite 78 in Verbindung mit ZK01/02.

Der weitere Aufbau unseres ECB-Computers geht wie folgt von statten:

Nachdem nun die Grundeinheit aufgebaut und getestet ist, kann jeder nach Belieben seine "ECB-Träume" verwirklichen. Wir werden dazu beitragen, indem wir in den folgenden INFO's ECB-Zusatzkarten und natürlich die vier ECB-Computerkarten vorstellen. Die Platinen dazu können -von dem der gerne alles machen möchte- selbst erstellt werden. Die entsprechenden Vorlagen im Maßstab 1:1 liegen jeder INFO bei. Ansonsten sind die Platinen für ca. 12,- bis 15,- DM (gegen Vorkasse) fertig geätzt aber ungebohrt vom Bernd erhältlich. Die Bauteile, die in den Schaltungen verwendet werden, sind in jedem Elektronikladen erhältlich. Da es sich in der Regel um herkömmliche Bauteile handelt, möchten wir darum bitten, sie Euch -entsprechend der jeweiligen Stückliste- selbst zu besorgen. Porto- und Verpackungskosten würden hier den eventuellen günstigeren Einkauf wieder verteuern. Auch wäre unser Zeitaufwand für die Sache nicht mehr vertretbar. Wo es sich aber um spezielle oder auch kostenaufwendige Bauteile handelt, werden wir auf jeden Fall eine Sammelbestellung rechtzeitig anbieten.

Falls sich doch einmal Versorgungsprobleme ergeben sollten, sind wir natürlich gerne bereit diese zu beseitigen. Bitte meldet Euch rechtzeitig bei einem von uns wenn es für Euch irgendwelche Probleme beim Nachbauen gibt.

Viel Spaß beim Basteln und bis zur nächsten "ECB-Bus-Vorstellung" Euer Jens

Rich. Rensch, Orgelbaumeister
Bahnhofstr. 100, 7128 Lauffen

Postfach 226

W.Germany

Tel. 07133-8415



Ein neues Mitglied stellt sich vor

Juli 1986

RENSCH über RENSCH

Vor etwa 10 Jahren fing es an: mit programmierbaren Taschenrechnern. Ab April 81 erste Erfahrungen mit BASIC an einem SHARP PC 1211. Nach etwa 1 Jahr waren aber doch die mageren 1024 Bytes ausgereizt, und es kam ein gebrauchtes Video Genie I (EG 3003) mit „unglaublichen“ 16 KB her. Es lebt heute noch, natürlich längst mit Expander, Doubler, einem 80- und zwei 40-Track-Laufwerken, alles DS/DD und zum Umschalten zwischen LW 0 (=40 Tr.) und LW. 2 (=80 Tr.) eingerichtet (ähnlich Vorschlag in Info 14, S. 87). Aber schon 1982 brachte der Weihnachtsmann den neuen GENIE III. Im CHIP-Preisausschreiben gewann ich auch noch den HAUPTGEWINN, ein „Textsystem“ (GENIETEXT 2.0, der erste und wohl einzige Hauptgewinn in meinem Leben!). Seitdem verstauben die verschiedensten Scripsit-Versionen (mit Kurs in Deutsch auf CC), Zorloff und Newscript (trotz Korrektur-Lexikon und automatischem Glossarschreiber), weil es alles nicht an den GTXT herankommt!

Der jüngste Filius (das letzte unserer 9 Kinder) strickte ein recht brauchbares Nachkalkulationsprogramm (in Basic) für Orgelbauten, mit dem wir nun schon seit 1983 recht erfolgreich arbeiten. Weitere professionelle Programme kamen bald dazu (GP 4, Visicalc, Lohnabrechnung usw.). Nur für die Buchführung hatten wir seit 1979 schon einen TA 20 MP (mit erstaunlichen 8 KB für 25.000,- DM und Magnetbandstreifen auf den Kontokarten als „Massen-Speicher“), der bei der nächsten Steuergesetzänderung prompt nicht mehr mitkam. Jetzt machen wir die Buchführung mit einem kompatiblen (Commodore PC 20 mit Festplatte).

Vor etwa 1 Jahr streikte der GENIE III plötzlich total, gerade als Trommeschläger pleite machte. „Phönix“ ließ kopfschüttelnd mitteilen, daß sie „solche alten Modelle“ nicht mehr unterstützen könnten. Der Monat ging dem Ende entgegen, und die Lohnabrechnung war fällig, für die wir früher immer ca. 4 Tage gebraucht hatten (mit dem Computer nur noch 1/2 Tag!). Was tun? Keiner wollte so einen Uraltcomputer noch reparieren. So schaute ich in die Kleinanzeigen von CHIP und fand tatsächlich einen zu verkaufenden GENIE III („wegen Umstellung“) für „lumpige“ zweieinhalb Tausender (mit Software - nochmal ein originaler GENIETEXT, sogar mit Kalkulation!). Der Zahltag war gerettet, und schließlich fand sich der (manchmal m.E. zu Unrecht gescholtene) Schmidtke in Aachen bereit, den anderen GENIE III für lumpige 600,- DM wieder flott zu machen: er löttete sämtliche Lötstellen auf den Platinen nach! Das hätte ich schließlich auch noch hingebacht. Löten 20,- DM, „Gewußt wo“ 580,- DM!

Immerhin habe ich nun ganz allein für mich einen „Seniorchef-Computer“, und ich diktiere kaum mehr, sondern schreibe fast alle Briefe selber.

Dem CLUB 80 gehöre ich nun seit März 86 an. Es ist der dritte Computerclub, die beiden Vorgänger sind sanft entschlafen. Es scheint auch der erste Club zu sein, dessen Club-Info funktioniert. Leute, seid denn bewußt und pflegt Eure fleißigen Officials. Auch ich will gerne etwas dazu helfen, soweit es meine schwachen Kenntnisse erlauben. Ich hoffe auch, daß der Kajo mit seinem Baby- und Opa-Assemblierkurs (das meine ich durchaus positiv, denn sowas fehlt meist!) weiter macht (bitte nicht zu schnell!). Vielleicht können damit einige von uns doch noch die höheren Computerweihen der Assemblerkunst erlangen. Also auf ans Werk!

Richard Rensch

HEFT
15
September
1986

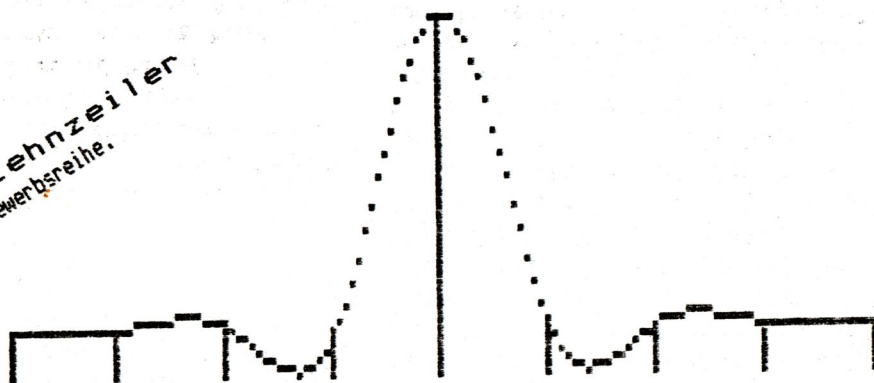
86

Gerald Schröder

```

1 READ N:N=N-1:DIM X(N),Y(N),D(N),R(N),E(N),C(N),Z(N),XX(128),YY(128):FOR I=0 TO N:
2 D(0)=X(1)-X(0):FOR I=1 TO N-1:D(I)=X(I+1)-X(I):C(I)=2*(D(I-1)+D(I)):NEXT I:Z(2)=D(1)
3 D(1)=D(0):FOR I=2 TO N-1:Z(I)=D(I-1)/(C(I-1)-D(I-2))*Z(I-1):NEXT I:FOR I=1 TO N-1:R(I)=(Y(I
4 +1)-Y(I))/D(I)-(Y(I)-Y(I-1))/D(I-1):R(I)=3*R(I):NEXT I:Tridiag GLS
5 B(1)=R(1)/C(1):FOR I=2 TO N-1:B(I)=(R(I)-D(I-1)*B(I-1))/(C(I)-D(I-1)*Z(I)):NEXT I:C
6 (N-1)=B(N-1):FOR I=N-2 TO 1 STEP -1:C(I)=(B(I)-Z(I-1)*D(I+1)):NEXT I:C(0)=0:C(N)=0
7 FOR I=0 TO N-1: B(I)=(Y(I+1)-Y(I))/D(I):E(I)=B(I)-(2*C(I)+C(I+1))*D(I)/3:D(I
8 )=(C(I+1)-C(I))/(3*D(I)):NEXT I:W=(X(N)-X(0))/127:XX(0)=X(0):FOR I=1 TO 127:XX(I)=XX
9 I-1+W:NEXT I
10 FOR J=0 TO 127: XX=XX(J):FOR I=0 TO N-1: IF (X(I+1)-XX)>0 THEN XX=X(I):YY
11 (J)=Y(I)+XX*(E(I)+XX*(C(I)+XX*D(I))): GOTO 7
12 NEXT J
13 NEXT J:IN=YY(0):SU=IN:FOR I=1 TO 127:Y=YY(I): IF Y>SU THEN SU=Y:GOTO 8 ELSEIF
14 Y<IN THEN IN=Y
15 NEXT J:FOR I=1 TO N:Y=Y(I):IF Y>SU THEN SU=Y:NEXT I:ELSEIF Y<IN THEN IN=Y:NEXT I
16 LSE NEXT:CLS:FOR I=0 TO N:Y=Y(I):GOSUB 10:Y=(X(I)-X(0))/W:FOR J=47 TO Z STEP -1:SET
17 (I,J):NEXT J,I:FOR I=0 TO 127:Y=YY(I):GOSUB 10:SET(I,Z):NEXT I
18 FOR I=0 TO N: Y=Y(I): GOSUB 10: K = (X(I)-X(0))/W: FOR J=47 TO Z STEP -1: SET
19 (K,J): NEXT J,I : FOR I=0 TO 127 : Y=YY(I): GOSUB 10 : SET (I,Z) : NEXT:GOT
20 O 1
21 DATA 9,
22 1,2,3,4,5,6,7,8,9,
23 0,0,0,0,0,0,0,0,0

```



... und wieder ein Zehnzeiler
in unser Wettbewerbsreihe.

„Unsere Zeit ist so aufregend, daß man die Menschen eigentlich nur noch mit Langeweile schockieren kann.“

Kreis ohne Trigonometrie

In der mc 8/86 steht ein kleines Plot-Programm, das Kreise ohne die Verwendung von trigonometrischen Funktionen zeichnet. Das Programm ist für diejenigen interessant, die sich wie ich ohne CIRCLE-Befehl durch das Leben zeichnen müssen.

Ich habe das Programm nur in einen Punkt abgeändert: der Divisor IN war im Original mehr oder weniger zufällig gewählt, so daß entweder kleine Kreise stundenlang gezeichnet wurden oder große Kreise ebensogroße Lücken aufwiesen. Die folgende Version sollte bei jeder Kreisgröße einen lückenlosen Kreis in maximaler (Basic-)Geschwindigkeit ergeben.

```

10 MX=Mittelpunkt, X-Koordinate
20 MY= " " , Y= " "
30 R=Radius
40 X=R : Y=0 : IN=1.1*R
50 X=X-Y/IN
60 Y=Y+X/IN
70 PLOT MX+X,MY-Y : PLOT MX+X,MY+Y : PLOT MX-X,MY+Y : PLOT MX-X,MY-Y
80 IF X>0 THEN 50

```

Gerald Schröder

Vorlage von Jürgen Zahn in mc 8/86

„Die Frau wird erst an dem Tag mit dem Mann wirklich gleichberechtigt sein, an dem man auf einen bedeutenden Posten eine inkompetente Frau beruft.“

Logische Operationen in BASIC

OR, AND und NOT, die sog. logischen Operationen, sind für viele Basic-Programmierer ein Buch mit sieben Siegeln. Dabei sind diese drei Operationen nur die Grundfunktionen, von denen alle weiteren Logikoperationen, wie NAND, NOR, XOR abgeleitet werden können. Um wenigstens ein wenig Verständnis für diese Funktionen zu wecken, hier ein paar Grundlagen und weiterführende Gedanken!

Am besten veranschaulicht man sich die logischen Operationen anhand sog. Funktionstabellen. Sie enthalten, je nach Funktion, ein oder zwei Eingangszustände und den Ausgangszustand. Die Funktionstabelle für die AND-Verknüpfung verdeutlicht dies:

A	B	C	Wie man hier sehen kann, sind auf der linken Seite die beiden Eingänge in den jeweiligen Zuständen gezeigt, die sie annehmen können. Die rechte Seite zeigt den daraus resultierenden Ausgang. Die Funktionstabelle verdeutlicht sehr gut, was eigentlich schon der Name der
0	0	0	
0	1	0	
1	0	0	
0	1	0	

Funktion aussagt:

Am Ausgang C erscheint nur dann eine "1", wenn Eingang A und Eingang B eine "1" führen!

Auch bei der OR-bzw. ODER-Verknüpfung macht die Funktionstabelle sehr schnell deutlich, wie die Sache funktioniert:

A	B	C	Der Ausgang C wird immer dann eine "1" sein, wenn mindestens einer der Eingänge eine "1" führt. Ob dies Eingang A oder B ist, spielt keine Rolle.
0	0	0	
0	1	1	
1	0	1	
1	1	1	

In Basic sieht die Sache so aus:

```
10 FOR A = 0 TO 1
20   FOR B = 0 TO 1
30     C = A OR B
40     PRINT A,B,C
50   NEXT B
60 NEXT A
```

Was jedoch passiert, wenn wir von den 0'en und 1'sen weggehen? Nehmen wir einmal an, A soll 7 sein, B hat den Wert 1! Wie sieht das Ergebnis in C aus? Eigentlich ist die Sache ganz einfach, auch wenn man sich etwas unter die Oberfläche des dezimalen Zahlensystems begeben muß. Die BASIC-Operationen OR, NOT und AND behandeln die ihnen übergebenen Argumente immer in ihrer binären Darstellungsweise. Es wird also innerhalb der Maschine die Operation 111 b AND 101 b durchgeführt. Setzen wir diese beiden Binärzahlen in eine etwas abgewandelte Funktionstabelle um, sehen wir schnell, warum die BASIC - Operation 7 AND 5 als Ergebnis 5 bringt!

1 1 1 b	7 d
AND 1 0 1 b	AND 5 d
= 1 0 1 b	= 5 d

A C Sehen wir uns jetzt die Operation NOT an. Sie hat nur einen Eingang und bewirkt am Ausgang die genaue Umkehrung des Eingangswertes! Aus einer 0 am Eingang wird eine 1 am Ausgang und umgekehrt!

Was bringt die BASIC-Zeile "PRINT NOT A", wenn A zuvor mit dem Wert 9 geladen wurde???

Ganz einfach:	1 0 0 1 b	9 d
NOT	-----	NOT ---
	0 1 1 0	6 d

Soweit also die Grundfunktionen. Was hat es nun mit den weiterhin genannten Operationen NAND (NOT AND), NOR (NOT OR) und XOR (exklusiv ODER) auf sich? Diese sind im BASIC nicht realisiert, können jedoch aus den Grundfunktionen zusammengesetzt werden! Die Funktionstabellen zeigen wie:

NOR	NAND
A B C	A B C
0 0 1	0 0 1
0 1 0	0 1 1
1 0 0	1 0 1
1 1 0	1 1 0

Als BASIC-Zeile wäre:

NOR -> C = NOT (A OR B)

NAND -> C = NOT (A AND B)

Bei beiden Funktionen wird also jeweils das Ergebnis der Grundfunktion negiert (NOT-verknüpft!).

Bei der XOR-Operation wird's etwas komplizierter. Hier erst mal die Funktionstabelle:

A B C	Der Ausgang ist immer nur dann '1', wenn die beiden Eingänge ungleich sind! Haben beide Eingänge gleiche Wertigkeit, ist der Ausgang '0'!
0 0 0	
0 1 1	
1 0 1	
1 1 0	

Wie hat nun aber die BASIC-Zeile auszusehen, in der wir mit Hilfe der implementierten Logikoperationen OR, AND und NOT die XOR-Funktion simulieren?

Schamhaft muß ich gestehen, daß ich auch nicht ohne fremde Hilfe auf die Lösung des Rätsels kam (manchmal habe ich wirklich ein gewaltig dickes Brett vorm Kopf!). Ich mußte mir von Ulrich Böckling mit Rat und Tat zur Seite stehen lassen. Trotzdem, oder gerade deshalb, möchte ich einen kleinen Wettbewerb starten und biete demjenigen, der mir als erster die Lösung des Problems als BASIC-Zeile schickt, eine Flasche Wein aus eigenem Weinberg (garantiert glycolfrei!!!). Disqualifiziert werden all die Teilnehmer, die jetzt schnell Ulrich Böckling anrufen, um sich dort, genau wie ich, Hilfe zu holen. Als Datum gilt das Datum des Poststempels und nicht etwa die Ankunft der Lösung bei mir, damit diejenigen, die weiter weg wohnen, nicht benachteiligt sind. Telefonanrufe zählen nicht! Hier nochmals das Problem bzw. die Frage:

C = A XOR B

Wie muß die BASIC-Zeile aussehen, die mit Hilfe der Funktionen OR, AND und NOT diese Verknüpfung realisiert!

Im nächsten Heft folgt dann die Auflösung, die Nennung der Teilnehmer und des Gewinners und noch ein paar Tips und Tricks für den Umgang und die Verwendung der Logikoperationen! Viel Glück und Erfolg wünscht Euch

Karlheut Obermann

HEFT
15
September
1986

10

Mit dem unten abgedruckten Programm können im WDR-Format abgespeicherte, hochauflösende Bilder direkt auf einen grafikfähigen Drucker ausgegeben werden. Dadurch kommen auch Grafikfreaks ohne HRG in den Genuß hochauflöster Bilder! Das Programm ist auf einem Model 4P und für den EPSON RX-80 geschrieben. Für andere Drucker und das Model 1 müssen natürlich Änderungen vorgenommen werden. Das Programm ist ab sofort bei Kajott in der Clubdiskothek zu haben. Dort bekommt ihr auch Bilder im WDR-Format!

Karlsruhe Obermann

```

110 REM DRUCKBILD - fuer TRS-80 Model 4 mit Level II - BASIC
140 REM W.BACK MAI 85 geaendert 6.86 Hartmut Obermann
160 CLEAR 1000 : CLS
161 IF INF(248)<>61 THEN PRINT0,"Drucker nicht bereit!";GOTO161
162 'Zeilenvorschub auf 'eine Pixelhoehe' einstellen
163 LPRINTCHR$(27);"3";CHR$(3)
165 ON ERROR GOTO 390
170 DEFINT A-Z
180 CLS:PRINT:PRINTTAB(10)* * * B I L D - D R U C K E R * * *:PRINT
185 PRINT"Dieses Programm druckt im WDR-Format gespeicherte Bilder auf
    grafikfaehigen EPSON-Druckern aus!":PRINT
190 PRINT "Dateiname : ";:LINE INPUT NAM$
200 PRINT"Laufwerk : ";:LINE INPUT LAUF$
210 OPEN"1" , 1 , NAM$ + ".:" + LAUF$
220 LINE INPUT #1,N$ : X = INSTR(N$,"BILD") : IF X=0 THEN 220
230 LINE INPUT #1,R$ : PRINT R$
240 LINE INPUT #1,V$ : VEZ = VAL(MID$(V$,2,3)) : PRINT VEZ
250 LINE INPUT #1,H$ : HOZ = VAL(MID$(H$,2,3)) : PRINT HOZ
260 LINE INPUT #1,D$ : LINE INPUT #1,D$
265 PRINT"Werte OK!?"
266 J$=INKEY$ : IF J$="" THEN 266
267 PRINTJ$ : IF J$<>"J" AND J$<>"j" THEN 220
268 'N$ = kennbuchstabe fuer normale (K) oder doppelte (L) Dichte
    N1 und N2 representieren die Anzahl der zu druckenden Pixel
269 IF HOZ>480 THEN N$="K" ELSE N$="L":
    N2=CINT(HOZ/256) : N1=CINT(HOZ-N2*256)
270 FOR V = 0 TO VEZ
280 IF EOF(1) THEN GOTO 382
290 LINE INPUT #1 , A$ : X=0 : IF LEN(A$)=0 THEN NEXT V
291 IF INSTR(A$,"uebertragung beendet")<>0 THEN GOTO 381
294 'EPSON-Drucker auf den benoetigten Grafikmodus einstellen
295 LPRINTCHR$(27);N$; : Z=N1 : GOSUB 10000 : Z=N2 : GOSUB 10000
300 FOR H = 0 TO HOZ STEP 6 : X = X + 1
310 A = ASC(MID$(A$,X,1)) - 32
320 IF A AND 1 THEN Z=128 ELSE Z=0 : GOSUB 10000
330 IF A AND 2 THEN Z=128 ELSE Z=0 : GOSUB 10000
340 IF A AND 4 THEN Z=128 ELSE Z=0 : GOSUB 10000
350 IF A AND 8 THEN Z=128 ELSE Z=0 : GOSUB 10000
360 IF A AND 16 THEN Z=128 ELSE Z=0 : GOSUB 10000
370 IF A AND 32 THEN Z=128 ELSE Z=0 : GOSUB 10000
380 NEXT H : Z=13 : GOSUB 10000 : NEXT V
381 PRINTA$
382 CLOSE: PRINT"Bild gedruckt!"
389 LPRINTCHR$(27);"0":END
390 PRINT0,"*":PRINT0,"*":RESUME 380
9999 'Ausgabe der Grafik an den Drucker unter Umgehung des normalen
    Druckertreibers, da dieser bestimmte Zeichen 'verschluckt'!
10000 I=INF(248):IF I<>61THEN10000
10010 OUT 248,Z : RETURN

```



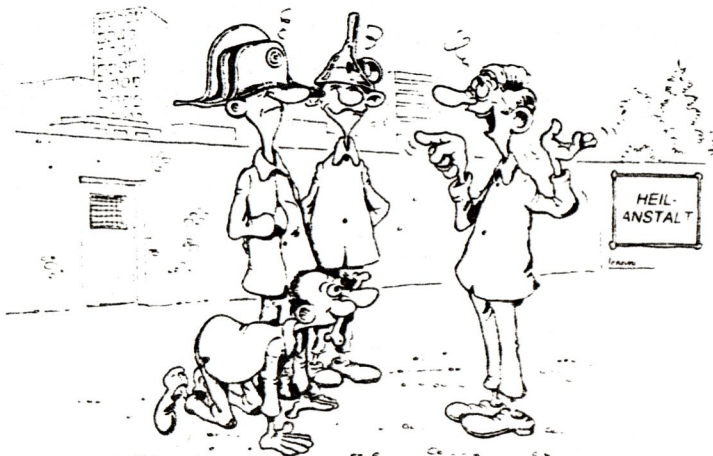
Es stimmt! Die Inventur ist wirklich viel einfacher mit dem Computer.



Man erzählt sich, daß alles mit einem Fehler im Programm begann...

„Selbst Gott braucht
die Werbung. Er hat Glok-
ken.“

„Die Freiheit des
Menschen liegt nicht
darin, daß er tun kann, was
er will, sondern daß er
nicht tun muß, was er nicht
will.“



CTRL SHIFT RETURN BREAK INSERT CLEAR GOTO
SELECT START < DELETE BAKS → SHIFT LOCK
SYSTEM RESET ! RETURN LINE FEED ESC RPT
SHIFT + RETURN BS → SELECT INSERT +
ENTER SHIFT 1 = CTRL START= RETURN + > , 0
CLEAR BACKSPACE TAB PRINT REM

"Mein erstes Assembler-Programm" -
eine Berichtigung zur Beschwichtigung!

"Na so ein Heini! Der will uns was erzählen?" - das hat so mancher von euch, der bereits in den ersten Kinderschuh der Fa. Assembler steckt, gedacht, sofern er meinen Beitrag im letzten Heft gelesen hat. "Wie kann man nur 'REPEAT' mit 'ER' abkürzen und uns glauben machen, es hieße 'LDIER'!" (14. Ausgabe, Juli 86, S. 12 Mitte)

Recht so! Nur daß dieser Mist-Ake nicht auf meinem Assembler-Mist gewachsen war, sondern an meinem allzu flüchtigen Umgang mit TSCIPS lag! Hatte ich doch versehentlich statt des Steuerzeichens für Fettdruck (§E) ein # getippt - und so blieb hier ein unbeabsichtigtes "E" stehen.

Doch dies ist beileibe keine Entschuldigung!

Wer was schreibt, sollte es mindestens ein-, besser dreimal (3x) auf Fehler durchlesen, bevor er es veröffentlicht! Hier tat ich (und noch jemand...!) es nur je einmal; und wir erwiesen uns als "betriebsblind".

Eine größere (fast nicht mehr "läßliche") Sünde war es, euch falsch über die Plazierung von Definitionen und Gleichsetzungen zu belehren! Ich danke an dieser Stelle nochmals Arnulf SOPP, der sich als einziger zu meinem Beitrag meldete und auf diesen Kardinalfehler hinwies!

Ich hatte bereits hervorgehoben, daß man beim Assembler zwischen "Anweisungen" und "Befehlen" zu unterscheiden hat. In meiner Rekrutenzeit (nicht beim "Bund", sondern im "388-jährigen Reich") gab es da keinen Unterschied - aber man lernt ja nicht aus. *

Der Unterschied ist, daß Definitionen wie 'DEFM' (DEFine Message), 'DEFB' (DEFine Byte), 'DEFS' (DEFine Storage) usw. keine Befehle an die CPU, sondern nur Anweisungen an den Assembler sind, die er beim Assembliervorgang befolgen soll und die nachher im fertigen Assemblat (die Leute nennen es mal den "OPCODE" = Operation Code und mal den OBJECT-CODE - warum soll ich dann nicht einen dritten wohlklingenden Namen, der eindeutiger ist, dafür erfinden??) gar nicht mehr in Erscheinung treten; sie haben nämlich heimlich bewirkt, daß der Code bereits alle Zutaten enthält, die ihn der CPU schmackhaft machen. * * Das gleiche (doch! Heute schreibt man's klein; ich hab nachgesehen) gilt für die Gleichsetzungen mit 'EQU'.

Weil diese "Anweisungen" (die Bezeichnung "Definitionen" wäre m.E. logischer!) die CPU nichts angehen, ja für sie nach der Assemblierung überflüssig geworden sind, haben sie mitten im Programm, das sie durchlaufen soll, nichts zu suchen!! Also bitte: Gleich an den Anfang des Source-Codes damit! Und zwar vor 'ORG'! (Oder notfalls auch an das Ende, oder teils vor, teils hinter - aber nie mitten hinein!)

Wer bis hierher mitgedacht hat, sollte fragen: "Wieso hat dann das Programm überhaupt funktioniert?" (Denn alle haben es ja ausprobiert: keiner hat sich beklagt...)

Um das herauszufinden, habe ich die Bytes, die in den DEF... und EQU-Zeilen stehen, mal so betrachtet, wie die CPU sie sieht: nämlich als "Befehle", indem ich diese einer Disassembliertabelle entnahm (eine "reziproke Mnemonics-Sammlung"). Siehe da: dabei stellte sich heraus, daß es sich im wesentlichen um bedingte Sprungbefehle handelte, wobei deren Bedingung nie erfüllt war, so daß der PC (ihr wißt ja: PC = 1e Postal Courreur oder Pfadfinder, der der CPU jeden nächsten Schritt zeigt!) fröhlich mir-nichts-dir-nichts drüber hinwegging! * G.C.C. = Glücks-Sache CeJot?? Nein: Serialer Source Code!! - Zur Nachahmung nicht empfohlen!

Diese Aufmerksamkeitstests werden nicht die letzten gewesen sein...

Bis auf ein neues!

Euer xelat

HEFT
15
September
1986

14

KA + KA-LA = MU

=====

Wer kennt "Das Große LaLuLa" ?

Lest Christian Morgenstern!

Er assemblierte schon vor 80 Jahren unsere Hochsprache ("GINGGANZ") und stellte im LaLuLa die größte Mnemonics-Sammlung seiner Zeit zusammen ("BIFZI BAFZI ..."). Im Gegensatz zum Hochdeutsch von C.M.'s LaLuLa kann K.M.'s MaMaLaMu nur Englisch interpretiert werden:

MATHEMATICS + MACHINE-LANGUAGE = MUSIC

Diesmal also nach dem Augenschmaus in der letzten Ausgabe einen Ohrensmaus!

* ZILOG = "Zauberei In LOGik" macht's möglich. *

Zuvor die unvermeidliche Theorie:

Wird der Verstärker mit schwankenden Spannungen beschickt, so gerät die Membran ins Wackeln. Sind die Schwankungen regelmäßig, so wackelt auch die Membran gleichmäßig, d.h. die Abstände Höchstpunkt-Ruhepunkt-Tiefpunkt sind zeitlich konstant (je nach Flächen-Trägheitsmoment ihres Querschnittes auch geometrisch, d.h. die Schwingungen sind symmetrisch). Dann spricht man nicht mehr von Wackeln (= diffuse Geräusche), sondern von Schwingungen. Die sind mal "schön" (konsonant), mal "hässlich" (dissonant). Aber das ist bereits Geschmackssache. PENDERECKI (polnischer Komponist, *1933) ist selten "schön", aber stets hinreißend! (Dies trifft übrigens besser als das Allerweltswort "interessant".)

Schwingungen in Sinusform, die sich in ganzzahligen Frequenz-Verhältnissen überlagern - teils harmonisch, teils hinder- oder zerreißen - können wir unserem Tasteninstrument zwar nicht entlocken. Aber so scharf hören wir nicht hin: die schiefen Rechtecksignale, die wir dem Verstärker aufzwingen, genügen uns voll und ganz, um uns zu bestätigen, daß wir "erfolgreiche Assemblerer" sind, und seien wir noch so (un)mu-, dafür aber um so mehr physikalisch!

Als "Eichwert" gilt der Kammerton 'a', heute mit 440 Schwingungen pro Sekunde (440Hz). Jeden Halbtonschritt darüber bzw. darunter erhalten wir durch Multiplikation mit bzw. Division durch $\sqrt[12]{2}$ (zwölfte Wurzel aus zwei). Nach zwölf Halbtönen erhalten wir wegen $(\sqrt[12]{2})^{12}$ den Faktor 2, das ist wieder ein 'a', diesmal genau eine Oktave höher.

Mit diesem Wissen können wir Das Große LaLuLa leicht starten!

Einen Taschenrechner, der beliebige Wurzeln ziehen kann, hat wohl jeder; andernfalls hilft die gute alte Logarithmentafel (notfalls ein Dentist). Auch hat jeder, der ins Assemblerreich aufbrechen will, ein gutes Z80-MaLa-Book (inzwischen ist wohl klar, wie man Machine Language abkürzt...), in dem die Taktzyklen und -zeiten stehen, die jeder Z80-Befehl bei der Ausführung benötigt.

Wenn wir die Taktzeiten aller Einzelbefehle, die die Membran zum Hin- und Herschwingen veranlassen, zusammenzählen, hängt das

Ergebnis natürlich von der Struktur ab, die wir dem Programm geben. Hier kommt es nicht auf elegante Kürze an, sondern nur darauf, daß die Summe der einzelnen Taktzeiten möglichst genau mit der "Periode" der gewünschten Tonhöhe übereinstimmt (absolut genaue Übereinstimmung ist übrigens physikalisch unmöglich, weil HEX keine Dezimalzahlen kennt <Widerspruch erbeten!>). Die Periode ist <Mus-, Elektr-, Stat- und andere-iker wissen das> der Reziprokwert der Frequenz:

Periode T in $\mu s = 1 / (\text{Frequenz in MHz})$

(1MHz = 1 Mega-Hertz = eine Million Schwingungen in der Sekunde)

Da der Kammerton 'a' ---> 0,00044 Schwingungen/ μs macht, ergeben sich für die Töne, die wir beispielsweise für diese Melodie



brauchen, gemäß der Wurzelformel folgende Perioden (notfalls sagt uns das auch ein gutes Physikbuch, z.B. der GRIMSEHL, Band 1 Mechanik, Kapitel Akustik <nicht mit dem Akku-Register A der CPU zu verwechseln>):

Ton	Frequenz f (Hz)	Periode T (μs)	C (Hex)
e	329.63	3033.70	B3
g	392.00	2551.02	96
h	493.88	2024.78	76
c	523.25	1911.13	6F

Die letzte Spalte nenne ich "Frequenz-Index" C; sie ergibt sich aus der einfachen Formel $C = (T - 69) / 16.5$, die ich aus meinen Programmzeilen 170 bis 230 plus 350 bis 380 errechnet habe (s. Anhang). Diesen (vom Programm abhängigen) Frequenz-Index laden wir später ins C-Register.

Nun charakterisiert jeden Ton nicht nur eine Schwingungsdauer (die wir ihrer Kürze wegen gar nicht hören können), sondern zwecks Hörbarkeit auch eine Verweildauer B, während welcher wir ihn hören. Deshalb bedienen wir uns einer Schleife (Zeilen 170 bis 230), in der das Frequenz-Unterprogramm (Zeilen 350 bis 380) immer wieder abgearbeitet wird. Diese Verweildauer laden wir vor dem Schwingungsteil, der von C umfaßt wird, ins Register B (Zeile 150).

Warum verwende ich B und C? Ich hatte einen Hintergedanken, der mir zunächst frech und gewagt erschien (typisch Verfasser). Ich war dann selbst verwundert, daß es klappte. Ja: Zauberer-80 lehrt uns noch wundern...

Die Idee: Wenn ich B und C verwende, kann ich sie vielleicht als Doppelregister (16Bit-Register) behandeln und mit der "Anweisung" (Mathematiker sagen stattdessen "Erklärung" oder "Definition"; "Anweisung" ist für den Anfänger irreführend, weil er dies leicht mit "auszuführendem Befehl" verwechselt. So schlecht sind MaLa-"Lehrbücher"! - also mit der Erklärung

DEFW (=DEFINE Word)

als 2-Byte-Kombination in den Speicher laden, aber jedes Byte einzeln wieder herausholen, siehe Zeilen 20 bis 120 (im Gänsemarsch in Speicherzellen schreiben). In Zeile 150 wird dann zuerst die Tondauer (Reg. B) und in Zeile 350 danach der Frequenz-Index (Reg. C) wieder aus der Gedächtnis (memory) geholt.

Diese Tonhöhen und -längen sind in den Zeilen 20 bis 120 als 16 Bit lange "Wörter" abgelegt, bevor das eigentliche Musikprogramm in Zeile 140 (hier in Speicheradresse 8017h) beginnt. Das niederwertige Byte, das stets zuerst abgelegt und auch wieder herausgeholt wird (siehe 2. Spalte = Operation Code, kurz "OPCODE" genannt), entspricht der Tondauer, das höherwertige der Tonhöhe (Frequenz).

In Zeile 260 wird das A-Register jedesmal, wenn es sich das dritte, fünfte, siebente usw. Byte für die Tondauer geholt hat, gefragt, ob die "Melodie zu Ende" ist. Dies ist der Fall, wenn das Nullbyte gemäß der Definition "DEFB" von Zeile 130 erreicht, d.h. wenn A=0 ist. Das Mnemonic, das diese Frage stellt, lautet: CP 00h, was soviel bedeutet wie: Vergleiche den Inhalt von A mit Null! Faktisch wird nicht "gefragt", sondern die Differenz zwischen dem Akku-Inhalt und dem Argument von CP gebildet; bei Gleichheit ist sie natürlich Null. Sobald dies eingetreten ist, springt das Programm in der nächsten Zeile ins DOS zurück: "JP Z, -->dorthin", das ist ein bedingter Sprung, der nur dann ausgeführt wird, wenn das Z-Bit im "Flag-Register" "gesetzt", d.h. gleich Eins ist und somit anzeigt, daß das letzte Ergebnis im Akku eine Null war! Ist das nicht der Fall (weil die Melodie noch weiter geht), so wird der Sprungbefehl übersprungen und der Programmzähler PC macht gehorchen im laufenden Programm weiter.

Die Übergabe der so definierten Parameter an den Verstärker erfolgt über Port 255d = OFFh durch einen "OUT"-Befehl (Zeilen 180 bzw. 210), der abwechselnd eine Eins oder eine Null an ihn schickt, wobei die Eins ihn auf ein Potential von 0.85V (hoher Wert) und die Null ihn auf ein solches von 0.46V (mittlerer Wert) legt. Über die beiden Bits 0 und 1 des Port(s) kann natürlich auch eine Zwei (10h) gesandt werden, die ihn an Masse legt und damit die Schwingungsamplitude (Lautstärke) mehr als verdoppelt; doch wir wollen nichts übertreiben, schließlich sitzt vor dem Lautsprecher normalerweise noch ein Poti.

Nachdem wir mit diesen Behauptungen uns aber bereits verdammt nahe an Hardware-Belange herangetastet haben, mit denen ich mit Rücksicht auf die Umwelt wegen überhöhter Zinn-Verschmutzung absolut nichts im Sinn habe, wollen wir uns edleren Dingen zuwenden und uns die Musik anhören, die wir da erzapft haben! (Nein, das "v" wurde nicht vergessen. Wer will, kann apf gegen eug eintauschen.)

Ich sagte schon, daß ich selbst über den halbwegs gelungenen Effekt überrascht war; stand es doch wieder mal in keinem "Leerbuch", daß man ein 16-Bit-Wort (DEFW) wieder zerreißen und seine beiden Hälften einzeln und unabhängig voneinander als vollgültige SteuerCodes, hier als kusalische Komponenten (Frequenz&Dauer), einsetzen könne! Im allgemeinen entsprechen zwei halbe Fliegen eher einer toten Fliege; hier aber haben wir zwei vollwertige Brummer nach der Teilung! * Jedoch:

Wie man sieht, enthält die Melodie (für Nichtschwimmer, die sich nicht ins tiefe ASS-Becken trauen, habe ich sie in klassischer Mensural-Notation bereits verraten - er summe sie vor sich hin; aber bitte con sordino, um niemand zu kränken...) viermal das g hintereinander. In meiner ersten Version erwies sich dies als ein einziger Ton mit der Länge von zweieinhalb Taktschlägen (ein Viertel im Viervierteltakt - durch c gekennzeichnet - entspricht einem Taktschlag; 3 Achtel + 1 Viertel sind also 2,5 Taktschläge); d.h. der gewünschte Rhythmus kam nicht zustande, weil die einzelnen frequenzgleichen Töne ineinander

Französisch: Le torchon brûle entre eux.

Das Spültuch brennt zwischen ihnen.



Der Hausseggen hängt schief.

Amerikanisch: That's the way the cookie crumbles.
Wie ein Keks halt so krümelt.
Wie das Leben halt so spielt.
Markus Hubers, Gescher

verschwammen. Also mußten sowas wie "Zwischenräume" her. Zu diesem Behufe fädelte ich ab Zeile 280 bis 330 eine kleine Trennschleife ein. Die Werte für die Haltereister D und E wurden einfach ausprobiert.

A propos:

Das Salz beim ASSEMBLIEREN ist grad' das AUSPROBIEREN!

Wenn's dann mal klappt, freut man sich, der Frust schwindet, das Erfolgserlebnis ist da und man kann's vergessen und neuen Ufern (= Untaten) zustreben... Hierin weiß ich mich eins mit größ'ren Geistern.

Übrigens: Wenn man glaubt, man könne statt D=OFFh plus E=0Ah gleich das Doppelregister DE mit OFF0Ah laden, "so irrt sich der" und fällt beim Versuch, die Sache dann mit "DEC DE" in Gang zu bringen, wie Busch's Frosch vom Baum - er stürzt ab. Warum, weiß ich bis heute nicht. Welch ein Widerspruch zu der freudigen Entdeckung der Leistungsfähigkeit des Doppelregisters BC, siehe oben!! Vielleicht verrät es mal ein Experte, der an den Fortschritten der Halbassembler interessiert ist! (???)

Der Frequenz-Index C bestimmt übrigens, genauer als oben gesagt, nicht die ganze, sondern nur die halbe Periode (Zeile 340, Label "HLRPER"), weil ja genau nach jeder halben Periode die Fortbelegung, sprich das Ausgangspotential von Hoch nach Tief vice versa geändert werden muß.

Denit wären wir am Ende des Programmchens. Entstanden ist weder ein Beethoven-Plagiat noch eine Tanz-Orgie, sondern nur der Anfang eines Gassenhauers. Wer kennt seinen Text? Einstmals kleine Balina haik'n noch ümma jefiff'n! (Aber der 'J' in Kajott is keen verstümmeltet 'G', gloobt man blos nich!!)

Nun kennst du die bekannte Melodie mal zu Ende!

Kajott

HEFT
15
September
1986

18

19

8000 7F96 00010 ANF DEF 8000H ; VIERTELNOTE G
 8000 7F96 00020 ANF DEF 967FH ; PUNKTIERTE ACHTEL C
 8002 5D6F 00030 DEF 6F5DH ; SECHZEHNTEL E
 8004 1FB3 00040 DEF 0B31FH ; ACHTEL G
 8006 3F96 00050 DEF 963FH ; DTO.
 8008 3F96 00060 DEF 967FH ; VIERTELNOTE G
 800A 7F96 00070 DEF 963FH ; ACHTELNOTE E
 800C 3F96 00080 DEF 0B33FH ; ACHTELNOTE E
 800E 3FE3 00090 DEF 963FH ; ACHTELNOTE G
 8010 3F96 00100 DEF 6F3FH ; ACHTELNOTE C
 8012 3F6F 00110 DEF 76FFH ; HALBE NOTE H
 8014 FF76 00120 DEF 00H ; SCHLUSS DER MELODIE.
 8016 00 00130 DEF HL,ANF ; ZEIGER AUF ERSTE NOTE.
 8017 2100B0 00140 START LD B,(HL) ; TONDauer.
 801A 46 00150 TON LD HL ; ZEIGER AUF FREQUENZ-INDEX.
 801B 23 00160 INC A,01H ; ERSTE HALBPERIODE.
 801C 3E01 00170 DAUER LD (OFFH),A ; 0,85V AUF VERSTÄRKERPORT.
 801E D3FF 00180 OUT FREQU ; ZUM UP FREQUENZGEBER.
 8020 CD40B0 00190 CALL A,00H ; ZWEITE HALBPERIODE.
 8023 3E00 00200 LD (OFFH),A ; 0,46V AUF VERSTÄRKERPORT.
 8025 D3FF 00210 OUT FREQU ; ZUM UP FREQUENZGEBER.
 8027 CD40B0 00220 CALL DAUER ; WIEDERHOLUNG DER PERIODE,
 802A 10F0 00230 DJNZ ; SOLANGE TON ANDAUERT.
 802C 23 00240 INC HL ; ZUM NÄCHSTEN TONLAENGEN-
 802D 7E 00250 LD A,(HL) ; PRUEFUNG, OB MELODIE
 802E FE00 00260 CP 00H ; ZU ENDE IST;
 8030 CA2D40 00270 JP 2,402DH ; WENN JA ==> DOS,
 8033 1E0A 00280 LD E,0AH ; SONST TRENNUNG (ZAESUR)
 8035 16FF 00290 TREN2 LD D,OFFH ; ZWISCHEN DEN TONEN,
 8037 15 00300 TREN1 DEC D ; DA GLEICHE TONE SONST
 8038 20FD 00310 JR NZ,TREN1 ; NICHT
 803A 1D 00320 DEC E ; ZU UNTERSCHIEDEN
 803B 20F8 00330 JR NZ,TREN2 ; SIND.
 803D C31A50 00340 JP TON ; ZUM NÄCHSTEN TON.
 8040 4E 00350 FREQU LD C,(HL) ; UNTERPROGRAMM
 8041 02 00360 HLEPER DEC C ; ZUR FESTLEGUNG DER
 8042 C241B0 00370 JP NZ,HLBPER ; FREQUENZ (TONHOEHE)
 8043 00375 ; DURCH DEN BERECHNETEN
 8044 00376 ; 'FREQUENZ-INDEX'.
 8045 C9 00380 RET ; ZUR NÄCHSTEN HALBPERIODE.
 8017 00390 END START
 00000 TOTAL ERRORS
 33635 TEXT AREA BYTES LEFT

ANF 8000 00020 00140
 DAUER 801C 00170 00230
 FREQU 8040 00350 00190 00220
 HLEPER 8041 00360 00370
 START 8017 00140 00390
 TON 801A 00150 00340
 TREN1 8017 00300 00310
 TREN2 8035 00290 00330

Anm. Leider sendet EDTASM (min)
 keine Minuskeln; es sähe besser aus!

Berechnung der Frequenz-Indices

Die Dauer einer Schwingung - das ist die reziproke Tonfrequenz - entspricht der Zeit, die die CPU benötigt, um je eine positive und eine negative Halbschwingung zu erzeugen, indem sie abwechselnd ein höheres und ein niederes Potential über den Port 255 zum Verstärker schickt. Dies macht sie, indem sie der Befehlsfolge ab Label "DAUER" in Zeile 170 bis zur Dekrementierung in Zeile 230 gehorcht, wobei zweimal ein CALL zum Unterprogramm für den Frequenzindex C in Zeile 350 bis 390 durchgeführt wird. Wir müssen also die Taktzeiten aller Befehle, die hierbei anfallen, zusammenzählen.

Hierzu stellen wir uns zunächst die Anzahl Taktzyklen zusammen, die die Befehle machen, die in den vorgenannten Zeilen vorkommen:

Mnemonic	Anz. Taktzyklen
LD A,n	7
OUT n,A	11
CALL nn	17
DJNZ e (Sprungweite)	13 (solange E<00)
DEC r (Register)	4
JP NZ,nn	10
RET	10

Zählt man die für jede der oben genannten Zeilen jeweils gültigen Zahlen zusammen - wobei die Zeilen 350 und 380 doppelt und die beiden Zeilen 360 und 370 sogar 2Cmal zu zählen sind, s.o. - so erhält man für die Anzahl Taktzyklen die Gleichung:
 $TZ = 117 + 28C$. Da (bei meinem alten Z80) ein Taktzyklus ziemlich genau 0.59 Mikrosekunden dauert, (was von mir am 1.4.0815 mit einer Stoppuhr exakt gemessen wurde,) dauert die Abarbeitung der Befehlsfolge für die Tonperiode (indirekt also für die Frequenz) $T = 0.59(117 + 28C) \mu s$, woraus die im Beitrag erwähnte Formel für den "Frequenz-Index" C folgt. Da in Tabellenwerken i.d.R. nicht die Periodendauern, sondern die Frequenzen der Töne in Hz aufgeführt sind, erhält man die richtige Größenordnung der Periode, indem man den Kehrwert der in MHz verwandelten Tabellenwerte in die Formel einsetzt.

*"DAUER" = Schwingungsdauer (nicht Tondauer!)

** Wie (Matthie-) üblich, wurde der Mult.-Operator fortgelassen:

"2Cmal" bedeutet also "2 * C-mal"; "28C" bedeutet "28 * C".

DIR/SYS oder INHALT/SYS?

WEGE ZUR GDOS/NEWDOS-KOMPATIBILITÄT

Ein zunächst kaum ins Auge fallender Unterschied zwischen NEWDOS-80V.2 und GDOS besteht in der Benennung des Directory-Files. Formatiert man eine Diskette mit NEWDOS, so heißen die dabei kreierten SYS-Files BOOT/SYS und DIR/SYS. Tut man dasselbe mit GDOS, so heißen diese beiden Files GDOS/SYS und INHALT/SYS. Da man auch in GDOS das Directory mit DIR (oder mit I) aufrufen kann, fällt das wie gesagt auch dem Umsteiger zunächst kaum auf. Im Gegenteil, man empfindet es als Umsteiger nach kurzer Gewöhnung als großen Vorteil, das Directory statt mit drei Tastenanschlägen (DIR) mit nur einem (I) aufrufen zu können.

Trotzdem stellen sich nach einiger Zeit gewisse Nachteile heraus. Es gibt eine große Anzahl Programme, die auf den DIR/SYS-File zugreifen, z.B. Programme, die die Directory-Einträge (ohne die SYS-Files) alphabetisch sortieren und sortiert auf die Diskette zurückschreiben, eine besonders bei DD und/oder 80-Track-Laufwerken mit bis zu ca. 200 Einträgen fast unentbehrliche (max. 15 x 4 = 60 Einträge auf einer Bildschirmseite) Utility. Auch die Katalogisierungsprogramme greifen auf DIR/SYS als File zu. Heißt nun aber der Directory-File INHALT/SYS, während das Programm auf DIR/SYS zugreifen will (und umgekehrt), so gibt es eine Fehlermeldung „Datei nicht im Inhaltsverzeichnis“ (bzw. "FILE NOT IN DIRECTORY"): Inkompatibilität zwischen NEWDOS und GDOS.

Natürlich ist eine Abhilfe relativ einfach möglich, indem man den File je nachdem in DIR/SYS bzw. INHALT/SYS (mit RENAME bzw. N - auch eine der so angenehmen Befehlsabkürzungen im GDOS, diesmal gleich fünf Tastenanschläge weniger!) - umbenennt.

Weit eleganter wäre es natürlich, die GDOS-Systemdiskette durch einen Patch zu veranlassen, von vornherein beim Formatieren den Directory-File gleich DIR/SYS statt INHALT/SYS zu benennen. Der GDOS-„Erfinder“ J.H.L. Heicke gibt als Grund für die Benennung des Inhalt-Files mit INHALT/SYS statt DIR/SYS an: „Diese Namensänderung wurde wie bei GDOS/SYS beschlossen, um eine Zerstörung der abweichenden internen Aufteilung durch direkt zugreifende Programme zu erschweren“ (JHL Heicke, GDOS-Systembeschreibung, 1984). Der Zugriff wird, wie gezeigt, tatsächlich erschwert. Eine Zerstörungsgefahr ist mir hingegen noch nicht begegnet, wie z.B. das Programm DIRSORT von Heinfried Cznotka, Braunschweig, beweist: es greift offensichtlich direkt auf den Directory-File zu und schreibt die Einträge alphabetisch sortiert auf die Diskette zurück, aber (im Gegensatz zu anderen Programmen gleichen Namens und gleicher Funktion) unabhängig vom Namen des Directory-Files!

So schaute ich mir den in NEWDOS und GDOS für das Formatieren und Kopieren zuständigen Systemfile SYS6/SYS genauer an. Auch als Nicht-Assemblergewandter findet man leicht im relativen File-Sektor (FRS) 24d/18h beim relativen Sektor-Byte (SRB) 84h die Einträge „GDOS/SYS“ und (SRB A4h) „INHALT/SYS“. Also gleich mal umgezappt in BOOT/SYS und DIR/SYS (die restlichen drei Bytes werden mit Blanks - 20h - aufgefüllt) und probiert!

Tatsächlich heißen nach dem Formatieren die beiden SYS-Files einer neu formatierten Diskette nun BOOT/SYS und DIR/SYS, wie ein Auflisten mit I,S (bzw. DIR,S) beweist. Alles geschafft - denkt man. Aber leider hat man sich getäuscht! Beim Ausprobieren von Programmen mit einem Zugriff auf DIR/SYS gehen diese weiterhin mit „Datei nicht im Inhaltsverzeichnis“ auf Fehler.

Noch schlimmer: auch wenn man im Programm den Namen DIR/SYS in INHALT/SYS umwandelt (was vorher zum Erfolg führte), bleibt das Programm mit der Fehlermeldung „Datei nicht im Inhaltsverzeichnis“ hängen. Und selbst wenn man nun mit DDE oder SUPERZAP den File DIR/SYS zu listen versucht, erscheint die gleiche Fehlermeldung.

Der relative Laufwerkssektor (DRS), in welchem das Directory einer Diskette beginnt, ist leicht aus den PDRIVE-Werten dieser Diskette zu errechnen:

NEWDOS	GPL	x 5 x	DDSL	= DRS des Directory-Beginnes
GDOS	EIB		SBIV	
<u>Beispiele:</u>				
	6	x 5 x	48	= 1440 (für GENIE III bzw. 80-Tr.-Laufwerke DS/DD)
	2	x 5 x	17	= 170 (40 od. 35-Tr.-Laufwerke SS/SD)

GPL = Grans Per Lump = EIB = Einheiten Im Block; 1 Einheit (Gran) = 5 Sektoren in NEWDOS und GDOS.

DDSL = Disk Directory Start Lump = SBIV = StartBlock des Inhaltsverzeichnisses.

In der normalen PDRIVE-Einstellung beginnt also bei 80-Track Laufwerken Double Sided mit Double Density der Directory-File im relativen Drive-Sektor (DRS) 1400d/5A0h. Also mit SUPERZAP mal nachgesehen, was auf DRS 1440d steht: der Directory-Beginn! Im DRS 1443d und 1444d stehen die File-Einträge für BOOT/SYS und DIR/SYS. Sie scheinen zu stimmen.

SUPERZAP erlaubt mit dem Kommando DNTH (Display Name/Type Hashcode) die Berechnung des Hashcodes für jeden Filenamen. Dabei ergibt sich

A2 für BOOT/SYS
A1 für GDOS/SYS
C4 für DIR/SYS
CE für INHALT/SYS.

Überprüft man nun die Hashcodes im DRS 1441d/5A1h. SRB 00 für BOOT/SYS und SRB 01 für DIR/SYS, so stellt man fest, daß hier weiterhin die Codes A1 und CE für GDOS/SYS und INHALT/SYS stehen, obwohl die die Filenamen im DRS 1403d/5A3h und 1404d/5A4h entsprechend der gemachten Änderung in SYS6/SYS in BOOT/SYS und DIR/SYS umgeändert sind.

Das kann nichts anderes bedeuten, als daß durch diesen Systemzap zwar die Filenamen geändert sind, nicht aber die zugehörigen Hashcodes, so daß das System beim Aufruf der Filenamen die entsprechenden Files BOOT/SYS und DIR/SYS nicht finden kann, weil der entsprechende Hashcode in DRS 1441d nicht zu den geänderten Namen paßt. Im Gegensatz zur Benennung eines Files mittels CREATE oder RENAME (bzw. N) aus dem DOS oder OPEN aus dem BASIC wird also der Hashcode beim Formatieren nicht nach dem im FRS 24d/18h von SYS6/SYS festgelegten Namen errechnet, sondern er muß an einer anderen Stelle festgelegt sein und danach eingetragen werden. ABER WO ???

Mit SUPERZAP habe ich den GDOS-File SYS6/SYS nach dem Byte CE (= Hashcode für INHALT/SYS) durchsucht und dabei 19maliges Vorkommen festgestellt und notiert. Beim Durchsuchen des SYS6/SYS-Files von NEWDOS nach CE fehlte CE gegenüber dem entsprechenden GDOS-File im FRS 25d/19h bei SRB 98h; es hieß hier C4, das ist der Hashcode für DIR/SYS! Auf die gleiche Weise fand sich im gleichen FRS 25d/19h dicht daneben, bei SRB 95h, A1 (Hashcode für GDOS/SYS) im GDOS-File und A2 (Hashcode für BOOT/SYS) im NEWDOS-File.

Nach entsprechendem Umzapfen ergab die Probe endlich das gewünschte Ergebnis: die Formatierung zeigt die umgeänderten

HEFT
15
September
1986

22

2 3
Filennamen BOOT/SYS und DIR/SYS mit den richtigen Hashcodes A2 und C4.

Hier noch einmal die erforderlichen Laps im GDOS-File SYS6/SYS. Ich hielt es für angebracht, den Namen des BOOT/SYS-Files weiterhin bei GDOS/SYS zu belassen, da das nicht stört und doch einen gewissen Beleg auf jeder mit GDOS formatierten Diskette darstellt.

FRS 25d/19h, RSB 95h von 36 A1 23 nach 36 A2 23
gezappt ergibt den Hashcode für BOOT/SYS beim
Formatieren (bei mir nicht geändert).

FRS 25d/19h, RSB 98h von 36 C4 3A nach 36 C4 3A
gezappt ergibt den Hashcode für DIR/SYS beim
Formatieren.

Die File-Namen müssen entsprechend dem eingesetzten Hashcode in FRS 25d/19h im Sektor davor, also FRS 24d/18h beginnend bei SRB 84h von GDOS in BOOT (wenn man will) und bei SRB A4h von INHALT in DIR geändert werden, wobei die SRB A7h bis A9h mit Blanks (20h) aufgefüllt werden. Die beiden Sektoren sehen also so aus:

SYS6/SYS, FRS 18h
GDOS original:

```
DRV 00 3701 0051 CDD7 427E E683 E281 421A 0203 7..Q..Bß....B...
1 10 CB4E C287 42CB 4EC2 8742 CB4E 20EF CB46 .N..B.N..B.N...F
1H 20 280B CB4E 20E7 CB7E 28E6 7E36 D0C1 D1E6 (...N..ß(ß6....
30 FC20 0C1C 7BD6 0020 0314 1E00 D97E C9CD ....ä.....ß..
DRS 40 D742 360B 1098 21DD 427E FE03 28FB 23CD .B6...!Bß..(.#.
979 50 3300 18F5 CDD7 42CB 4620 FC7E C93E 063D 3.....B.F.ß.>.=
3D3H60 20FD C91C 1F52 4553 4554 031C 1F01 09A1 .....RESET.....
70 6547 444F 5320 3F03 0100 B765 0000 005E eGDOS.?....e...^
80 0000 0000 4744 4F53 2020 2020 5359 5360 ....GDOS....SYS'
90 7F1F B205 0000 00FF FFFF FFFF FFFF FF5D.....Ü
A0 0000 0000 494E 4841 4C54 2020 5359 53A7 ....INHALT..SYS.
B0 1DF9 E51E 0030 05FF FFFF FFFF FFFF FFCD .....0.....
FRS C0 9263 D218 52CD A063 E521 9759 CBD6 E1CD .c..R..c..!Y....
24 D0 B56E DC95 4FCD B86F CDB5 6E30 07CD 2550 .n..0..o..n0.%P
18H E0 ED53 8159 0640 CDA7 4E21 9459 7EE6 F920 .S.Y.ß..N!.Yß...
F0 047E F680 77CD B363 21F3 6DCD 6744 CD10 .ß..w..c!.m.gD..
```

GDOS gezappt:

```
DRV 00 3701 0051 CDD7 427E E683 E281 421A 0203 7..Q..Bß....B...
0 10 CB4E C287 42CB 4EC2 8742 CB4E 20EF CB46 .N..B.N..B.N...F
0H 20 280B CB4E 20E7 CB7E 28E6 7E36 D0C1 D1E6 (...N..ß(ß6....
30 FC20 0C1C 7BD6 0020 0314 1E00 D97E C9CD ....ä.....ß..
DRS 40 D742 360B 1098 21DD 427E FE03 28FB 23CD .B6...!Bß..(.#.
151950 3300 18F5 CDD7 42CB 4620 FC7E C93E 063D 3.....B.F.ß.>.=
5EFH60 20FD C91C 1F52 4553 4554 031C 1F01 09A1 .....RESET.....
70 6547 444F 5320 3F03 0100 B765 0000 005E eGDOS.?....e...^
80 0000 0000 4744 4F53 2020 2020 5359 5360 ....GDOS....SYS'
90 7F1F B205 0000 00FF FFFF FFFF FFFF FF5D.....Ü
A0 0000 0000 4449 5220 2020 2020 5359 53A7 ....DIR.....SYS.
B0 1DF9 E51E 0030 05FF FFFF FFFF FFFF FFCD .....0.....
FRS C0 9263 D218 52CD A063 E521 9759 CBD6 E1CD .c..R..c..!Y....
24 D0 B56E DC95 4FCD B86F CDB5 6E30 07CD 2550 .n..0..o..n0.%P
18H E0 ED53 8159 0640 CDA7 4E21 9459 7EE6 F920 .S.Y.ß..N!.Yß...
F0 047E F680 77CD B363 21F3 6DCD 6744 CD10 .ß..w..c!.m.gD..
```

FRS 19h
GDOS original:

2 4

```
DRV 00 67CD F34D 212D 55E5 CDC3 673A 9759 CB5F g..M!-U...g:~Y...
1 10 C03A 9659 CB7F C011 0000 CDB4 573A 9459 ..Y.....W:~Y...
1H 20 CB4F C021 DA6D CD67 44CD AA67 11B5 6E01 .0!..m.gD..g..n.
30 0200 CCB4 67C2 1A52 3EFF CDB0 64ED 5BD1 ....g..R>...d.A.
DRS 40 5921 0042 221A 5B4D E5CD 6257 E1CB C63A Y!..B".AM..bW....
980 50 CD59 32C5 592A 9943 7706 005F 3ACE 594F .Y2.Y#.Cw...:YO
3D4H60 D521 0042 CD1C 6221 7E59 11CB 4201 1600 .!..B..b!BY..B...
70 EDB0 E13A CA59 6707 0784 0100 B566 CD92 ....Yg.....f..
80 4C22 215B 0E00 183C C5AF CDB0 6479 3D20 L"!A...<....dy=.
90 1E21 0042 36A1 2336 C43A CE59 D602 4F07 .!..B6.#6.:Y..0.
A0 0781 321F 42C6 0A32 EE65 320C 6718 1421 ..2.B..2.e2.g...!
B0 BA65 FE02 3805 200B 21DA 6511 0042 0120 .e..8...!e..B..
FRS C0 00ED B0C1 DDCB 00C6 CDD4 57DD CB00 86C2 .....W.....
25 D0 1A52 0C79 FE0A 38B0 C921 C559 DDE5 E5DD .R.y..B...!Y....
19H E0 E1DD 7E04 DD6E 03CD 924C DD75 0ADD 740B ..ß..n...L.u..t.
F0 CDF2 5CDD 750C DD74 0DDD 7E05 CDB4 4CB7 ..ö.u..t..ß...L.
```

GDOS gezappt:

```
DRV 00 67CD F34D 212D 55E5 CDC3 673A 9759 CB5F g..M!-U...g:~Y...
0 10 C03A 9659 CB7F C011 0000 CDB4 573A 9459 ..Y.....W:~Y...
0H 20 CB4F C021 DA6D CD67 44CD AA67 11B5 6E01 .0!..m.gD..g..n.
30 0200 CCB4 67C2 1A52 3EFF CDB0 64ED 5BD1 ....g..R>...d.A.
DRS 40 5921 0042 221A 5B4D E5CD 6257 E1CB C63A Y!..B".AM..bW....
152050 CD59 32C5 592A 9943 7706 005F 3ACE 594F .Y2.Y#.Cw...:YO
5F0H60 D521 0042 CD1C 6221 7E59 11CB 4201 1600 .!..B..b!BY..B...
70 EDB0 E13A CA59 6707 0784 0100 B566 CD92 ....Yg.....f..
80 4C22 215B 0E00 183C C5AF CDB0 6479 3D20 L"!A...<....dy=.
90 1E21 0042 36A1 2336 C43A CE59 D602 4F07 .!..B6.#6.:Y..0.
A0 0781 321F 42C6 0A32 EE65 320C 6718 1421 ..2.B..2.e2.g...!
B0 BA65 FE02 3805 200B 21DA 6511 0042 0120 .e..8...!e..B..
FRS C0 00ED B0C1 DDCB 00C6 CDD4 57DD CB00 86C2 .....W.....
25 D0 1A52 0C79 FE0A 38B0 C921 C559 DDE5 E5DD .R.y..B...!Y....
19H E0 E1DD 7E04 DD6E 03CD 924C DD75 0ADD 740B ..ß..n...L.u..t.
F0 CDF2 5CDD 750C DD74 0DDD 7E05 CDB4 4CB7 ..ö.u..t..ß...L.
```

Es sei noch bemerkt, daß bei mir auch noch andere GDOS- und NEWDOS-Fassungen existieren, bei denen aus irgendwelchen Gründen die betreffenden Stellen etwas verschoben sind. Anhand der in den Sektorenlistings aufgeführten Zusammenhänge wird jeder in seiner Fassung die entsprechenden Stellen leicht auffinden.

Und nun viel Spaß - noch mehr Spaß mit kompatibleren GDOS- und NEWDOS-Systemen!

Richard Rensch

The Disappearing DOS

Five utilities let you access Model I/III system functions from DOS Ready.

While you can't deny that a disk operating system is a necessity, in some cases it's also a hindrance. For example, if you want to execute a low-level function like modifying memory, initializing a printer, sending data out a particular port, or calling a ROM subroutine, you have to leave DOS, load either Basic or Debug, execute the function, and then return to DOS.

I've written five utilities that break this DOS barrier and let you use almost all your Model III capabilities directly from DOS Ready. This is especially useful in writing Build files, where a transparent DOS gives you direct access to system functions.

Typing in the Utilities

The five utilities presented below all use the same shell, which appears in the listing for POKE/SRC (see the Program Listing). Type in and assemble POKE, then write the file to disk. Load POKE/SRC and, for each of the other four utilities, make the changes indicated in the Table. Then assemble and save those four programs to disk with the appropriate file name.

The Five

POKE/SRC modifies up to 20 contiguous bytes of memory with a single command so you can change the cursor character, write to the screen, change case, and so on from DOS Ready.

POKE's format is POKE nnnn.nn, where the first parameter is a four-digit hexadecimal (hex) address and the second is a two- to 40-digit hex data stream. For example, POKE 4023.nn changes the cursor character, with nn the hex code for the new character. POKE 3C00.54455354 displays the word "TEST" in the upper left-hand corner of the display. POKE 0419.00 sets the keyboard for lowercase characters; 01 instead of 00 sets it for uppercase.

LPRINT/CMD, a line printer utility.



System Requirements

Models I and III
32K RAM
DOS
Editor/assembler
Printer

sends up to 20 control bytes to your line printer. This lets you execute a form feed or change modes on a programmable printer such as an Epson MX-60 without leaving DOS. The utility's format is LPRINT nn, where nn is a two- to 40-digit hex data stream. For example, LPRINT OC sends a stream-of-form control character to your line printer. LPRINT 1B451B47 sends an escape sequence; it sets the Epson MX-60 to emphasized, double-strike print mode.

OUT/CMD, a port output utility, sends a data stream of up to 20 bytes to any Z80 port. You can initialize any port-mapped peripheral, such as a universal asynchronous receiver/transmitter (UART) or a Z80 speed-up modification, from DOS Ready. The command's format is **OUT nn.nn**, where the first parameter is a two-digit hex port address and the second is a two- to 40-digit hex data stream. For example, **OUT EC.04** sets your screen to 32-character mode; **CLS** returns you to 64-character mode. **OUT FF.01020102010201020102** generates a beep from the cassette port.

CALL/CMD, a machine-language subroutine call utility, executes any machine-language subroutine and then returns to DOS. You can use this to reenter resident programs after a hang-up, to test subroutines, or to execute ROM subroutines. The command's format is **CALL nnnn**, where

nnnn is a four-digit hex address. For example, CALL 0049 stops execution until you press a key, useful as part of a Build file. CALL 01D9 prints the screen contents on a line printer, also useful as part of a Build file.

Execute/CMD, a machine-code execution utility, lets DOS serve as a machine-language interpreter. You pass hex digits as a parameter, and **Execute** converts them to binary and puts them in a buffer beginning at 5300H. The buffer is padded with No Operations (NOPs) and terminated with a jump to DOS (JP 402DH). Thus, unless the machine-language routine contains a jump external to the buffer, or contains an infinite loop, DOS will regain control after execution. You use this utility to test short routines, move blocks of memory, or pass parameters to subroutines. **Execute's** format is **EXECUTE nn**, where **nn** is a two- to four-digit hex data stream. For example:

EXECUTE 21003C1100F0010004EDB0
moves screen data to the buffer at
0F000H. Following is the Assembly-lan-
guage code for the above statement:

LD HL,3C00H
LD DE,0F0001H
LD BC,0400H
LDIR

The sequence below turns your TRS-80 into a typewriter:

```
LPRINT/SRC - LPRINT/CMD
Add:      412          EXX
Change:   268         JR     HI,PARAM2
          406          EXX
          418          CALL   3BE
Delete:    lines 388-388

OUT/SRC - OUT/CMD
Change:   496         LD     C,D
          418         OUT    (C),A
Delete:    lines 328, 338

CALL/SRC - CALL/CMD
Add:      292         PUSH  IX,EXIT
          294         LD     I,X
Change:   348         EX     DE,HL
          358         JP     (BL)
Delete:    lines 368-428

EXECUTE/SRC - EXECUTE/CMD
Add:      938        DEFN   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          540        DEFN   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          920        DEFB   0C3B
          968        ERFW   EXIT
          978        END    START
Change:   388 PARAM1   LD     D,START+100H
          468         JP     I,START+100H
          520         JP     I,START+100H
          578         ORG    START+100H
Delete:    lines 318-378
```

Table. Changes to POKESRC for the four other utilities.

Program Listing. POKE/SRC

```

5200 00100 START EQU 52000
5020 00170 EXIT EQU 4020H
5010 00180 PRINT EQU 0210H
5200 00190 ;
00200 ORG START
00210 ;
5200 7E 00220 SPACE LD A,(HL) ;FIND FIRST PARAMETER
5201 FF0D 00230 CP #DH
5203 C8 00240 RET ;
5204 F120 00250 CP ;
5206 2003 00260 JF NE,PARAM1
5208 23 00270 INC HL
5209 18F5 00280 JR SPACE
00290 ;
520B CD2152 00300 PARAM1 CALL GETBYTE ;get address
520E 57 00310 LD D,A
520F CD2152 00320 CALL GETBYTE
5212 5F 00330 LD E,A
5213 7E 00340 LD A,(HL)
5214 F02C 00350 CP ;
5216 C26452 00360 JP NE,ERROR
5219 23 00370 INC HL
00380 ;
521A CD2152 00390 PARAM2 CALL GETBYTE ;get data string
521D 32 00400 LD (DE),A
521E 13 00410 INC DE
521F 18F9 00420 JR PARAM2
00430 ;
5221 7E 00440 GETBYTE LD A,(HL)
5222 FE0D 00450 CP #DB
5224 CA2D40 00460 JF S,EXIT
5227 CD4252 00470 CALL LEGAL
522A DA6452 00480 JF C,ERROR
522D 07 00490 ADD A,A
522E 87 00500 ADD A,A
522F 87 00510 ADD A,A
5230 87 00520 ADD A,A
5231 4F 00530 LD C,A
5232 23 00540 INC HL
5233 7E 00550 LD A,(HL)
5234 FE0D 00560 CP #DH
5236 CA2D40 00570 JF S,EXIT
5239 CD4252 00580 CALL LEGAL
523C DA6452 00590 JF C,ERROR
523F 81 00600 ADD A,C
5240 23 00610 INC HL
5241 C9 00620 RET
00630 ;
5242 FE20 00640 LEGAL CP 2FH
5244 CA2D40 00650 JF S,EXIT
5247 D630 00660 SUB 3FH
5249 FE00 00670 CP 0
524B 3F 00680 CCF
524C FA6252 00690 JF N,ILLEGAL
524F FE0A 00700 OR 0AH
5251 3F 00710 CCF
5252 F0 00720 RET N
5253 D607 00730 SUB 7
5255 FE0A 00740 CP 10
5257 3F 00750 CCF
525E FA6252 00760 JF N,ILLEGAL
525F FE10 00770 CP 16
5261 3F 00780 CCF
5262 F26252 00790 JF P,ILLEGAL
5263 C9 00800 RET
00810 ;
5262 37 00820 ILLEGAL SCF
5263 C9 00830 RET
00840 ;
5264 216D52 00850 ERROR LD HL,ERROR
5267 CD18F2 00860 CALL PRINT
526A C32D40 00870 JP EXIT
00880 ;
526D 0A 00890 MERROR DEFR 0AH,'*** Command Parameter Error ***'
2A 2A 2A 2F 43 6F 6D 6D
61 6E 64 20 5F 61 72 61
6D 65 74 65 72 20 45 72
72 6F 72 2F 2A 2A 2A
528D 0A 00900 DEFB 0AB,0DH
00
00910 ;
528F 00920 ; END START

```

21

EXECUTE CD4900CD3300CD3B00C30053

Following is the Assembly-language code for the above statement:

```
CALL 0049H: FROM INKEY routine
CALL 0033H: Display character
CALL 003BH: Line print character
JP 5300H: Jump to beginning
```

DOS Ready

The five utilities described above let you access almost all the Model I/III features

from DOS Ready. If a needed DOS command doesn't exist, you can simulate it with one of these utilities. And if the excessive typing starts to bother you, you can type in Commando [see "Macro Economics," February 1986, p. 66], which lets you rename long, hard-to-remember commands to short, meaningful, logical names. ■

You can reach Craig Chaiken at 32 Beverly Drive, Avon, CT 06001.

DRV	00	060E	1994	132C	20DA	01EA	F850	10F8	D1E1#.....F.....
O	10	01F8	3509	27E2	584E	204F	4E4C	5927	2041	..S.'RUN.ONLY'.A
OH	20	4282	7155	4748	0344	6873	6868	65C0	6571	EBRUCH.Dosfehler
	30	2666	8174	512D	0721	212F	2052	4553	4554	.fatal.!!!.RESET
DRS	40	204D	6974	205F	617C	7145	20E2	710D	51C2	.mit.Taste.R!
148950		994E	7886	50FE	40CA	2040	23FE	680C	604D	.Nn.P.E.-f.k.M
5D1H60		CDDE	4020	042A	0743	E92E	CD6C	51E7	CDc0	..L..X.D.+1GW.1
	70	5179	E257	CD6C	51EF	CD6C	5179	8C6F	62EF	Gy.W.10_1Gv.ob.
TRK	80	D738	04FE	473C	0EFE	4138	0AD6	07E6	0F4F	.S..EO..AS.....0
41	90	0707	0707	053E	2FE7	C309	4130	0000	0000>...D.....
29H	A0	0000	0000	0000	0000	0000	0000	0000	0000
	B0	0000	0000	0000	0000	0000	0000	0000	0000
TRF	C0	0000	0000	0000	0000	0000	0000	0000	0000
13	D0	0000	0000	0000	0000	0000	0000	0000	0000
DH	E0	0000	0000	0000	0000	0000	0000	0000	0000
	F0	AFC3	0000	0202	47E1	0000	0000	0000	0000GG.....

Tscrips-Verschlüsselung

Im letzten Clubinfo habe ich darüber berichtet, wie man durch Abschaffen der Kassetten-E/A viel Platz im Tscrips erhält, den man nur nutzen muß. Eine Möglichkeit will ich Euch heute vorführen.

Beim Clubtreffen zeigte mir Hartmut eine Verschlüsselungsroutine in Pascal und fragte mich, ob man diese nicht in das Tscrips einbauen könnte. Diese Routine hat zwei Vorteile: die von ihr verschlüsselten Texte werden beim zweiten Durchlauf wieder entschlüsselt und das Codewort (bzw. Paßwort) existiert nur im Kopf des Benutzers. Da die Länge des Codeworts in dieser Version 1-53 Buchstaben beträgt, dürfte eine Entschlüsselung äußerst kompliziert sein.

Der Aufruf erfolgt beim Laden / Speichern mit dem Parameter "V": "S.V filename" beim Speichern und "L.V,K,C> filename" beim Laden. Es wird jeweils nach dem Codewort gefragt. Da die Eingabe über die <BREAK>-Routine erfolgt, sind Kleinbuchstaben nur möglich, wenn der erste Buchstabe "F", "D" oder "R" lautet (Find, Delete, Replace).

Die Erweiterung wurde für Tscrips 5.4 geschrieben. Mit einer Änderung kann sie auch für Tscrips 4.0 übernommen werden. In Zeile 101 muß es dann heißen:

```
101 LD DE,8E4Bh ;anderer Anfang des Textspeichers
```

Vielen Dank an Hartmut!

Gerald Schröder

```
00001 : PATCH #2
00002 : 07/86 BY GERALD SCHRÖDER
00003 :
00004 : VERSCHLÜSSELUNGS-ROUTINE FÜR TSCRIPS 5.4 / 4.0
00005 : IDEE UND LITERATUR VON HARTMUT OBERMANN
00006 :
00007 ORG 5D42H
00008 CALL 00DLOA ;CODEWORD EINGEBEN (LOAD)
00009 :
00010 ORG 5DB8H
00011 CALL 00DSAV ;CODEWORD EINGEBEN (SAVE)
00012 :
00013 ORG 5DEEH
00014 CALL VERLOA ;ENTSCHLÜSSELN BEI LOAD
00015 :
00016 ORG 5E0AH
00017 CALL VERSAV ;VERSCHLÜSSELN BEI SAVE
00018 :
00019 ORG 626AH ;ROUTINEN IN TAPE-E/A
00020 :
00021 00DLOA CALL 00DEEIN ;CODEWORD EINGEBEN
00022 LD A,(70BDH) ;ALTES KOMMANDO IM LOAD
00023 RET
00024 :
00025 00DSAV CALL 00DEEIN ;CODEWORD EINGEBEN
00026 CALL 72A9H ;ALTES KOMMANDO IM SAVE
00027 RET
00028 :
00029 00DEEIN BIT 2,(IY+0EH) ;FLAG "V" GEBETZT
00030 RET Z ;NEIN, KEINE VERSCHLÜSSELUNG
00031 PUSH HL
00032 PUSH DE
00033 PUSH BC
00034 LD HL,7A25H ;GRUNDWERTE FÜR RND-ROUTINE
00035 LD DE,7C74H ;ÜBERTRAGEN
00036 LD EC,0004
00037 LD7F
00038 LD (IY+07H),BFH ;OLDSCHREIBEN
00039 NOCHMAL LD HL,CODEW ;ZEIGER AUF ABFRAGE-TEXT
```

```
00040 CALL 6B0BH ;IN DIE LETZTE ZEILE
00041 LD E,53 ;BIS 53 ZEICHEN EINGEBEN
00042 CALL 5E72H ;EINGABE
00043 CALL 6BE9H ;LETZTE ZEILE LÖSCHEN
00044 :
00045 CALL 53FFH ;1. ZEICHEN AUS DEM BUFFER
00046 JR Z,NOCHMAL ;=1E? (ENTER)
00047 POP BC ;JA, NOCHMAL EINGEBEN
00048 POP DE
00049 POP HL
00050 RET
00051 :
00052 CODEW DB CODEWE-CODEW ;LÄNGE DES TEXTES
00053 DM 'Codeword?'
00054 CODEWE EQU $-1
00055 :
00056 :
00057 VERLOA CALL VERSCHL ;ENTSCHLÜSSELN
00058 LD HL,7CD1H ;ALTES KOMMANDO
00059 RET
00060 :
00061 VERSAV CALL VERSCHL ;VERSCHLÜSSELN
00062 LD DE,7CB1H ;ALTES KOMMANDO
00063 RET
00064 :
00065 VERSCHL BIT 2,(IY+0EH) ;FLAG "V" GES. ?
00066 RET Z ;NEIN, KEIN ENT-/VERSCHLÜSSELN
00067 PUSH HL
00068 PUSH DE
00069 LD DE,7DD2H ;ANFANG CODEWORD
00070 LD E,0 ;LÄNGE DES SEKTOREN (=256)
00071 LD HL,7CD1H ;ZEIGER AUF SEKTOR-BUFFER
00072 LOOP LD A,(DE) ;CODEWORD-BUCHSTABE LADEN
00073 CP 1EH ;ENTER? (ENDE DES WORTES?)
00074 JR NZ,601 ;NEIN
00075 LD DE,7DD2H ;ZEIGER AUF CODEWORD-ANFANG
00076 LD A,(DE) ;1. BUCHSTABEN LADEN
00077 SDI LD C,(HL) ;BUCHSTABE AUS DEM SEKTOR-BUFFER
00078 BIT 0,L ;ZEIGER GERADE?
00079 JR Z,602 ;JA
00080 RLCA ;CODEWORD-BUCHSTABE LINKS ROTIEREN
00081 JR 603
00082 602 RRC A ;CODEWORD-BUCHSTABE RECHTS ROTIEREN
00083 603 XOR C ;XOR MIT TEXT-BUCHSTABE
00084 LD C,A ;ERGEBNIS NACH C
00085 PUSH HL
00086 CALL 72B2H ;RND-ZAHL IN A ERZEUGEN
00087 POP HL
00088 XOR C ;XOR MIT BUCHSTABEN
00089 LD (HL),A ;IN DEN SEKTOR-BUFFER
00090 INC DE
00091 INC HL
00092 DJNZ LOOP ;BIS SEKTOR ZUENDE
00093 POP DE
00094 POP HL
00095 RET
00096 :
00097 :
00098 ORG 66DBH ;ÄNDERUNGEN IM SAVE/LOAD-AUFRUF
00099 JP Z,5D97H ;SPRUNG ZU SAVE
00100 BIT 0,(IY+0EH) ;FLAG "C" = VERKETTET
00101 LD DE,9177H ;TEXTSPEICHER-ANFANG (8E4BH IN 4.0)
00102 JR Z,604 ;NEIN, NICHT VERKETTET
00103 LD DE,(702BH) ;ENDE DES TEXTES
00104 604 JP 5D23H ;SPRUNG ZU LOAD
```

HEFT
15
September
1986

11


```

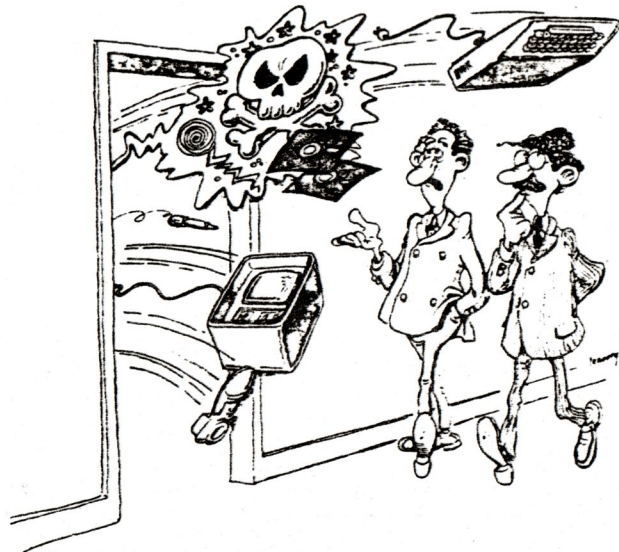
00105 ;
00106 :
00107     DRG     6ACFH
00108     DB      "V"      ;PARAMETER "V" STATT "T"
00109     END

```

Kleiner Tip für Tscrips

Bei allen Tscrips-Versionen kann man sich ein böses Fingerleiden zuziehen, wenn man zu oft TABT (mit Klammeraffe Rechts Pfeil). ZEUS hat da eine bessere Lösung: SHIFT SPACE. Wer das auch gerne in Tscrips hätte, muß nur in die Speicherstelle 7992h das Byte 1Fh statt 20h schreiben.

Gerald Schröder



Und das ist das Büro unseres Programmierers.

Die in unserem Club vertretenen Computer, die eine eingebaute Echtzeituhr haben, werden vom DOS leider teilweise im Stich gelassen. Es ist ein Relikt aus den alten Tagen des TRS-80, daß nach dem Wochentag nie gefragt wird. Ihn durch ein Programm aus dem Datum zu errechnen, wäre zu speicherplatzaufwendig geworden. Der zweite Grund, weshalb ich mich erneut mit der Uhr beschäftige, ist, daß das Boot-EPROM des Genie 3 s (und wohl auch anderer Rechner) die Uhr überhaupt nicht zur Kenntnis nimmt.

In meinem Artikel "Real Time Black Box?" ist bereits ausführlich beschrieben, wie die Uhr funktioniert. Deshalb baue ich heute darauf auf und verzichte auf allzu weitschweifende wiederholende Erklärungen. In jenem Artikel ist mir übrigens ein Fehler unterlaufen, der hiermit richtiggestellt werden soll: In dem kleinen Programm, das den Sonntag programmiert, muß anstelle der 7 eine 6 auf den Port 5A ausgegeben werden. Die 7 Wochentage werden nämlich ab 0 gezählt, so daß der höchste die Nr. 6 ist. Es war wirklich nicht meine Absicht, eine arbeitnehmerfeindliche Woche einzuführen.

Überhaupt ist dieser Punkt etwas heikel. Mein CP/M behauptet am Wochentag "0" steif und fest, er heiße Montag. Der im letzten Artikel erwähnte Auszug einer anderen Publikation findet, daß der Sonntag der erste Tag der Woche sei, auch für diesen Chip. Bibel beiseite, es ist egal! Wer nach dem vorgestellten Strickmuster einen Wochentag programmieren will, kann ihm jede beliebige laufende Nummer geben. Hauptsache, alle Programme oder Betriebssysteme benennen ihn gleich - und gemäß einer für alle gleichen Vereinbarung.

Um das Auslesen der Uhr sowohl im EPROM als auch unter DOS zu ermöglichen, muß die Ausgabe auf eine möglichst neutrale Art erfolgen. Beide Systeme verwalten nämlich den Bildschirm sehr unterschiedlich, so daß ein allgemeingültiger Algorithmus zur Anzeige nicht möglich ist. Daher wird hier beim Einsprung davon ausgegangen, daß HL als Zeiger auf einen Puffer geladen ist, in den der ASCII-String mit der Zeitangabe geladen werden soll. Jedes beliebige System kann anschließend diesen String auf seine Weise in den Bildschirm übertragen.

Dieser String hat folgende Gestalt: Fr, 22.08.86, 13:48:43. Es ist die komplette Zeit, wobei die einzelnen Angaben durch Komma und ein folgendes Leerzeichen getrennt sind. An der Stelle, wo hier der Satzende-Punkt (nicht Bestandteil der Zeitanzeige) steht, wird der String durch das Programm mit einem OD-Byte (ENTER) abgeschlossen. Nach dem Anzeigen der Zeit wird deshalb immer in die nächste Zeile gesprungen.

Das Listing zeigt außer dem Programm selbst eine kleine Routine (ab Label start), die die Adresse eines Puffers nach HL lädt und dann das Ausleseprogramm aufruft. Zuletzt wird über die DOS-Routine an 4467 der Pufferinhalt auf den Bildschirm übertragen. Wenn das Programm immer parat ist (SYS-File, EPROM, Helmut's "sicheres Plätzchen"), dann genügen zum Abruf einer Zeitanzeige ein paar Bytes, wie man sieht.

Die nun folgenden Erläuterungen zum Verständnis des Programms sind grundsätzlich für alle Uhrenchips, also auch für alle Echtzeituhren im Club gültig. Nur das Auslesen des Chips selbst, das von dem Unterprogramm rdclk erledigt wird, dürfte von Wecker zu Wecker variieren.

Es beginnt damit, daß der Pufferzeiger HL auf den Stack gerettet wird. Damit steht er bei abschließenden POP (Z. 33) wieder auf dem Pufferanfang, so daß das aufrufende Programm einen mündgerechten Zeiger zurückbekommt. Das Y-Indexregister wird sodann als Zeiger auf eine Maskentabelle geladen. Die verschiedenen Register des RTC-Bausteins enthalten nämlich nicht nur die jeweiligen Ziffern einer Zeitangabe, sondern daneben gibt es noch Bits, die etwas von Schaltjahren, Nachmittagen und

12/24-Stunden-Anzeige erzählen. Sie müssen ausmaskiert werden, weil nur die Ziffern selbst interessieren.

Das erste auszulesende Register ist Nr. 6, wo die Wochentage verraten werden. Die chipinterne Adresse 6 kommt beim 63s in das obere Nibble eines Bytes, das gleichzeitig im unteren den Befehl "lesen" (xC) enthält. Deshalb wird das Register A mit dem Adressbyte 6C geladen. Mit dieser teuren Fracht geht's ab ins UP rdneg, wo das gewünschte RTC-Datenregister über den Port 5B adressiert wird, um über den Port 5A ausgelesen werden zu können. Danach wird diese BCD-Ziffer gleich mit OR '0' in ihre ASCII-Entsprechung verwandelt, weil dies wegen der übrigen Zeiteinheiten ökonomischer erschien. Nachteil: Nach dem CALL in Z. 12 muß mit AND OF gleich wieder der Binärwert gebildet werden. Er ist nun die laufende Nummer eines Wochentags in der Tabelle daytab.

Es wird hier übrigens nicht rdneg ab Z. 49, sondern rdneg+1 ab Z. 50 aufgerufen. Der Umweg über das Register C, dessen Bedeutung später erläutert wird, ist hier nicht nötig.

Die Tagesnamen haben zwei Zeichen. Um einen Zeiger auf den korrekten Tag setzen zu können, muß der Akkuinhalt deshalb verdoppelt werden. Das geht am schnellsten mit RLCA. Als Zeiger wird DE geladen. Die Addition des LSB E und der Tagesnummer führt zum richtigen Byte in der Tabelle. Wer diese Routine in ein eigenes Programm (z. B. ein SYS-File) einbauen möchte, muß dabei allerdings höllisch aufpassen, daß die Tagestabelle komplett in einer Page liegt, d. h., das Adress-MSB muß für alle Namen gleich sein. Z. B. die Grenze zwischen 61FF und 6200 darf nicht mitten in der Tabelle liegen.

Alternativvorschlag, falls das nicht geht:

```
ADD A,E
LD E,A
LD A,0
ADC A,D
LD D,A
```

Dann geht es ab Z. 18 normal weiter.

Nach dem Einstellen des Wochentagszeigers muß der Pufferzeiger um eine Stelle zurückgesetzt werden, denn das UP rdneg erhöhte ihn. Dann werden die beiden zutreffenden Zeichen in den Puffer geschrieben. Im UP delia kommen noch ein Komma und ein Blank dahinter, um die Zeiteinheiten optisch voneinander zu trennen.

In Z. 24 wird BC mit 034C geladen. Dabei wird B als Zähler für drei Datumseinheiten (Tag, Monat, Jahr) aufgesetzt, C beinhaltet das Adressregister 4 der Uhr. Im oberen Nibble, wohlgeordnet (s. o.). Das untere ist wieder für das Lesen konfiguriert. Ähnlich verhält es sich mit dem Befehl LD DE,402E. E enthält dabei einen Punkt als Trennzeichen der Datumseinheiten, dessen ASCII-Wert 2E beträgt. Was es mit D auf sich hat, ist komplizierter:

Wir brauchen zur Anzeige zuerst die Zehnerstelle des Kalendertages, dann die Einerstelle, dann dasselbe für den Monat, dann für das Jahr. Im UP rdneg wird aber das Adressregister (in C gespeichert) immer um 10 (weil im oberen Nibble; sonst nur 1) verändert. Angesichts dieser Tatsache und eingedenk der Reihenfolge der Register, die in der Tabelle des letzten Artikels steht, ist also C nach einem Lesevorgang immer um 4 Register zu niedrig. Also muß 4 addiert werden. Wegen der Nibble-Akrobatik lautet der Wert tatsächlich 40. Und dieser Wert kommt in Z. 25 ins Register D. Nach der Addition der RTC-Adresse in Register C und D am Beginn des UP rdloop beträgt sie also nicht mehr 4, sondern 8. Das ist das Register für die Monatszehnerstelle, die wir nun brauchen. Uff!

Das Unterprogramm rdloop wird nun aufgerufen. Dort passiert zunächst das soeben Erwähnte. Dann geht es gleich in rdclk weiter, einem UP, das

**„Technik ist wie ein
Messer. Man kann damit
morden oder damit Brot
schneiden.“**

**„Es ist anatomisch
schwierig und immer ein
wenig lächerlich, sich gra-
tulierend auf die eigene
Schulter zu schlagen.“**

zwei RTC-Register ausliest. Es ruft nämlich zunächst rdneg auf, das wir vom Auslesen des Wochentags her bereits kennen und geht anschließend in rdneg über. Dort wird das zuständige Datenregister adressiert und ausgelesen. Die uninteressanten Bits werden ausmaskiert. Der Binärwert wird gleich ins ASCII-Format umgewandelt und in den Puffer geschrieben.

Nach der Rückkehr von rdclk kommt das Trennzeichen, beim Datum ein Punkt, aus E in den Puffer (Z. 39). Wenn alle drei Einheiten des Datums geschrieben sind, stört natürlich der Punkt hinter der Jahreszahl. In Z. 42 wird deshalb der Pufferzeiger auf ihn zurückgestellt, damit er mit einem Komma und einem anschließenden Blank überschrieben werden kann.

Nun ist die Uhrzeit dran. In Z. 27 wird wieder BC mit der Anzahl von Schleifen (in B) und der RTC-Adresse (in C) geladen. Diese Adresse lautet jetzt 5, weil dort die jetzt fällige Zehnerstelle der Stunden steht. Wie aus der erwähnten Tabelle hervorgeht, können wir jetzt immer ein Register tiefer rutschen, denn die Reihenfolge der Register deckt sich mit der Reihenfolge der Ziffern in einer Zeitangabe. Die Subtraktion in Z. 51 im UP rdneg trifft also immer genau das jeweils gewünschte Register. So kann auch DE mit ':', dem Trennzeichen für die Zeiteinheiten geladen werden. Auf diese Weise kommt 00 nach D, also der Summand für die Addition in Z. 36 ist 0.

Der Rest ist Handwerk. Wieder wird rdloop aufgerufen, um diesmal die Uhrzeit in den Puffer zu schreiben. Danach stehen dahinter wieder ein Komma und ein Blank, was ganz am Ende des Strings jedoch stört. Der Pufferzeiger HL wird deshalb in Z. 30/40 auf das Komma zurückgestellt, wo dann stattdessen 0D, also ENTER gepatcht wird. Nachdem der Zeiger mit POP wieder auf den Pufferanfang gesetzt ist, hat Reserve Ruh'.

Solche Selbstkorrekturen und verschiedene Andere machen das Programm ziemlich kompliziert, nahezu unübersichtlich. Das Problem ist jedoch, daß es im EPROM residieren soll. Es kann sich also nicht selbst modifizieren wie die Uhrenroutine des 63s im RAM. Bei dem gleichzeitigen Anspruch, es sehr kurz zu halten, sollte eben ein UP möglichst alles können.

Das Listing und der darunterstehende Sektordump geben eine Demoversion wieder, die bestenfalls als Spielkram zu bezeichnen wäre. Da jede Armbanduhr dasselbe leistet, lohnt ein solches Programm im Anwenderspeicher nicht. Interessant wird es erst, wenn es in einem SYS-File, im (EP-) ROM oder in einem Speicherbereich ständig bereitliegt, der Systemroutinen vorbehalten ist.

Arnulf Sopp

HEFT
15
September
1986

34


```

5200      00001      ORG      5200h
00002
00003 ;Rahmenprogramm, je nach dem anzupassen
5200 217F52 00004 start LD HL,buffer ;einen Puffer festlegen
5203 CD0952 00005 CALL prog ;Uhr dorthin auslesen
5206 C36744 00006 JP 4467h ;im Bildschirm anzeigen
00007
00008 ;Programm zum Auslesen der Uhr
5209 E5 00009 prog PUSH HL ;Pufferzeiger retten
520A FD216452 00010 LD IY,clktab ;Tabelle d. Ziffernmasken
520E 3E6C 00011 LD A,6ch ;ab Reg. 6 (Wochentage)
5210 CD5352 00012 CALL rdtreg+1 ;Wochentag auslesen
5213 E60F 00013 AND 0fh ;ASCII -> binär
5215 07 00014 RLCA ;* 2, weil 2 Zeichen/Tag
5216 117152 00015 LD DE,daytab ;Tabelle der Tagesnamen
5219 83 00016 ADD A,E ;auf richtigen N. stellen
521A 5F 00017 LD E,A ;Zeiger korrigieren
521B 2B 00018 DEC HL ;auf Pufferanfang zurück
521C EB 00019 EX DE,HL ;Quelle <-> Ziel tauschen
521D EDA0 00020 LDI ;2 Buchstaben des Tages-
521F EDA0 00021 LDI ;namens puffern
5221 EB 00022 EX DE,HL ;rücktauschen
5222 CD4852 00023 CALL delim ;", " hinter Tagesnamen
5225 014C03 00024 LD BC,034ch ;3 Durchl. ab Reg. 4+4=8
5228 112E40 00025 LD DE,402eh ;Summand 40, Trennz. "."
522B CD3D52 00026 CALL rdtloop ;3 Datumeinheiten ausl.
522E 015C03 00027 LD BC,035ch ;3 Durchl. ab Reg. 5
5231 113A00 00028 LD DE,": " ;Summand 00, Trennz. ":"
5234 CD3D52 00029 CALL rdtloop ;3 Uhrzeiteinh. auslesen
5237 2B 00030 DEC HL ;zurück auf letztes ", "
5238 2B 00031 DEC HL ;stattdessan CR puffern
5239 360D 00032 LD (HL),0dh ;Zeiger auf Pufferanfang
523B E1 00033 POP HL
523C C9 00034 RET
523D 79 00035 rdtloop LD A,C ;RTC-Register
523E 82 00036 ADD A,D ;je nach dem korrigieren
523F 4F 00037 LD C,A ;neu schreiben
5240 CD4F52 00038 CALL rdtclk ;1 Zeiteinheit auslesen
5243 73 00039 LD (HL),E ;Trennzeichen puffern
5244 23 00040 INC HL ;Zeiger nachstellen
5245 10F6 00041 DJNZ rdtloop ;bis 3 Einheiten ausgel.
5247 2B 00042 DEC HL ;auf letztes Trennzeichen
5248 362C 00043 delim LD (HL),',' ;mit ", " überschreiben
524A 23 00044 INC HL ;Zeiger nachstellen
524B 3620 00045 LD (HL),' ' ;Blank dahinter
524D 23 00046 INC HL ;Zeiger nachstellen
524E C9 00047 RET
524F CD5252 00048 rdtclk CALL rdtreg ;Einerstelle einer Einh.
5252 79 00049 rdtreg LD A,C ;RTC-Adressregister
5253 D35B 00050 OUT (5bh),A ;Datenreg. adressieren
5255 D610 00051 SUB 10h ;auf nächstes Register
5257 4F 00052 LD C,A ;neu laden
5258 DB5A 00053 IN A,(5ah) ;Zeit-Ziffer auslesen
525A FDA600 00054 AND (IY) ;relevante Bits maskieren
525D F630 00055 OR '0' ;binär -> ASCII
525F 77 00056 LD (HL),A ;Pufferzeiger nachstellen
5260 23 00057 INC HL ;cto. Maskenzeiger
5261 FD23 00058 INC IY
5263 C9 00059 RET
00060
00061 ;Tabelle der Ziffernmasken für die Zeiteinheiten
5264 07 00062 clktab DB 07h ;Wochentag, 0-6
5265 07 00063 DB 03h ;Zehner Datum, 0-3
5266 0F 00064 DB 0fh ;Einer Datum, 0-9

```

```

5267 01 00065 DB 01h ;Zehner Monat, 0-1
5268 0F 00066 DB 0fh ;Einer Monat, 0-9
5269 0F 00067 DB 0fh ;Zehner Jahr, 0-9
526A 0F 00068 DB 0fh ;Einer Jahr, 0-9
526B 03 00069 DB 03h ;Zehner Stunde, 0-2
526C 0F 00070 DB 0fh ;Einer Stunde, 0-9
526D 07 00071 DB 07h ;Zehner Minute, 0-5
526E 0F 00072 DB 0fh ;Einer Minute, 0-9
526F 07 00073 DB 07h ;Zehner Sek., 0-5
5270 0F 00074 DB 0fh ;Einer Sek., 0-9
00075
00076 ;Tabelle der Wochentagsnamen
5271 4D 00077 daytab DM 'Mo'
5273 44 00078 DM 'Di'
5275 4D 00079 DM 'Mi'
5277 44 00080 DM 'Do'
5279 46 00081 DM 'Fr'
527B 53 00082 DM 'Sa'
527D 53 00083 DM 'So'
00084 buffer
00085 END start

```

00000 Fehler

```

buffer 527F      clktab 5264      daytab 5271      delim 5248      prog 5209
rdclk 524F      rdtloop 523D      rdtreg 5252      start 5200

```

```

DRV 00 0181 0052 217F 52CD 0952 C367 44E5 FD21 ...R! R..R.gD...!
0 10 6452 3E6C CD53 52E6 0F07 1171 5283 5F2B dR>1.SR....qR.+
OH 20 EBED A0ED A0EB CD48 5201 4C03 112E 40CD .....HR.L...s.
30 3D52 015C 0311 3A00 CD3D 522B 2B36 0DE1 =R.ö.....=R++6...
DRS 40 C979 824F CD4F 5273 2310 F62B 362C 2336 .y.O.DRs#...+6,#6
252050 2023 C9CD 5252 79D3 5BD6 104F DB5A FDA6 #...RRy.A..O.Z...
9DBH60 00F6 3077 23FD 23C9 0703 0F01 0F0F 0F03 ..Ow#. #.....
70 0F07 0F07 0F4D 6F44 694D 6944 6F46 7253 .....MoDiMiDoFrS
80 6153 6F02 0200 5200 0000 0000 0000 0000 aSo...R.....
90 0000 0000 0000 0000 0000 0000 0000 0000 .....
A0 0000 0000 0000 0000 0000 0000 0000 0000 .....
B0 0000 0000 0000 0000 0000 0000 0000 0000 .....
FRS C0 0000 0000 0000 0000 0000 0000 0000 0000 .....
0 D0 0000 0000 0000 0000 0000 0000 0000 0000 .....
OH E0 0000 0000 0000 0000 0000 0000 0000 0000 .....
F0 0000 0000 0000 0000 0000 0000 0000 0000 .....

```

Arnulf Sopp

Nachtrag zu "Ach du liebe Zeit!"

In dem oben genannten Beitrag war die Rede davon, daß die Routine zum kompletten Auslesen der Uhr das Boot-EPROM des Genie III s zieren sollte. Sie zielt mittlerweile. Und noch etliches mehr. Alle Programme, die sich sinnvoll callen lassen, sind im Anschluß aufgeführt. Es sind teilweise solche, die schon immer drin waren, solche, die zwar drin waren, sich aber aus verschiedenen Gründen nicht als Unterprogramme eigneten (aber inzwischen eignen) und schließlich solche, die erst neuerdings drin sind.

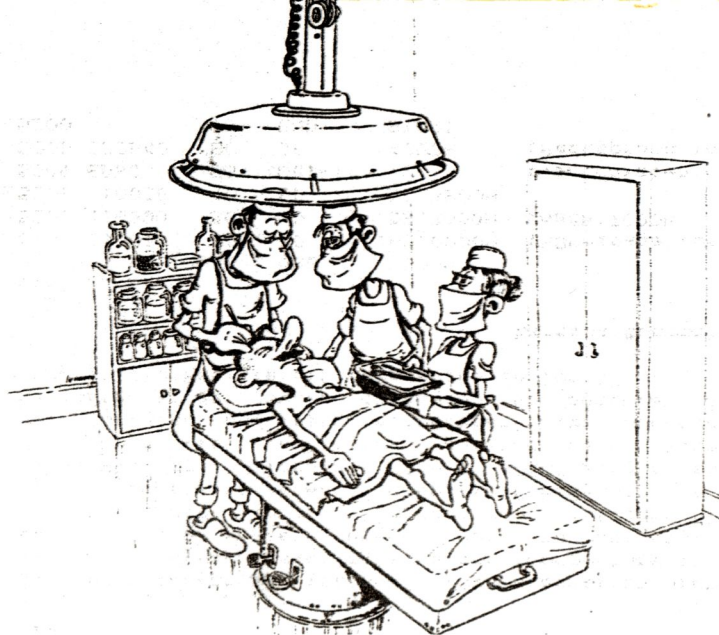
Nicht erwähnt ist dort, daß der Zeichensatz jetzt gefälliger aussieht. Das ist zwar Geschmackssache, aber Unterlängen auf Kosten der Bäuche des p und des q und ähnlich faule Kompromisse kommen nicht mehr vor.

Neu sind auch zwei Befehle, E (für EXECUTE) und U (für UHR). Letzterer tut natürlich das oben Beschriebene. E ruft eine Routine als Unterprogramm auf. Bisher konnte nur mit dem G-Befehl die Kontrolle endgültig an ein Programm im Speicher übergeben werden, das nicht in den Monitor zurückkehrt. EXECUTE hätte ich lieber CALL genannt und mit C aufgerufen, aber dieses Kommando ist bereits für COPY reserviert. Schließlich soll das EPROM mit der Originalversion voll kompatibel bleiben. Immerhin ist hier eine Analogie zu ZETBUG gegeben.

Da ich momentan mit ziemlicher Ausschließlichkeit am EPROM arbeite, ist es sehr wahrscheinlich, daß es zum Zeitpunkt des Erscheinens unseres Infos bereits noch mehr weiß und kann. Diese Aufstellung ist daher vermutlich bereits übermorgen sehr unvollständig.

Wer daran interessiert ist, mag sich bei mir melden. Für die BASIC-Experten unter uns dürfte es allerdings ziemlich uninteressant sein. Der veränderte Zeichensatz, der unter DOS ohnehin überschrieben wird und das gefälligere Logo lohnen nicht einmal das Porto. Wer aber in Maschinensprache arbeitet, bekommt interessante Ware, wie die nachfolgende Aufstellung zeigt. Komplexe Routinen, die sonst jeweils neu programmiert werden müßten, können nun einfach mit einem CALL abgerufen werden.

Interessenten möchten mir bitte die Registriernummer ihres EPROMs nennen. Man erfährt sie, indem man bei Reset oder beim Einschalten die F1-Taste gedrückt hält. Ich verlasse mich auf diese Angabe und wahre damit das Urheberrecht von Uwe Böker. Da immer noch 90% des Programms von ihm stammen, möchte ich nur 15 Mark in gängigen Briefmarken dafür. Ich habe nicht etwa das Gelübde des Reichtums abgelegt sondern kalkuliere folgendermaßen: Ein EPROM kostet z. Zt. ca. 8 Mark. Dazu gibt es Fotokopien der Bedienungs- und Einbauleitung. Das Porto entspricht dem Gewicht. Den Rest von etwa 5 Mark werde ich auf das Wohl des Erwerbers direkt der Witwen- und Waisenkasse der hiesigen Gastwirtschaft zufließen lassen - im wahrsten Sinne des Wortes. Prost!



Das ist alles!? Keine Gehäuseschrauben? Keine Platinen?

*** neue Befehlskürzel: ***
E adresse Unterprg. ab adresse aufrufen
U Uhr vollständig auslesen

*** verwendbare Subroutinen: ***

frstscr	oder	
019A		Kleinbildschirm löschen, Hello anzeigen
chrset	oder	
027B		Zeichensatz (DE) laden (256 Zeichen ab ASCII 00)
engchr	oder	
027D		B Zeichen ab ASCII A aus (DE) laden
scr3c00	oder	
08DF		Scroll des Kleinbildschirms
scroll	oder	
08E8		Scroll (DE Anf. 1. Z., HL Anf. 2. Z., BC Rest)
delay15	oder	
0A63		ca. 15 ms Verzögerung
cphlde	oder	
0CA3		HL und DE vergleichen (wie RST 18h)
gethex	oder	
0CD4		DE bin. <- Hex-ASCII-String ab (HL) bis Blank Cy: falsches Zeichen, Z: 4 Zeichen oder Blank
binchr	oder	
0EB9		A bin. <- gültige Hex- und Deziffer
calend	oder	
0F36		vollst. Uhr-Anzeige als ASCII-String nach (HL)
getdec	oder	
0FAC		ASCII-Dezimalstring binär nach HL laden (Ende bei der ersten Nicht-Ziffer)
putdec	oder	
0FCD		Binärzahl in DE als ASCII-Dez-String nach (HL)

Arnulf Sopp

HEFT
15
September
1986

In letzter
Minute
eingetroffen!!

E I N L A D U N G

Liebe Leute,

in einem der vorherigen INFO's wurden Regionaltreffen angeregt. Im südlichen Teilen der Bundesrepublik finden solche Treffen bereits mehr oder weniger häufiger statt. Das sowas im nördlichen Teil nicht klappt, hat glaube ich weniger mit den sog. Nodlichtern zu tun, als mehr mit der geringen Anzahl der Mitglieder! Aber trotzdem will ich einen Versuch starten und Euch nach Osnabrück einladen. Nicht, weil ich hier zufällig wohne, sondern weil Osnabrück recht günstig für ein solchen Vorhaben liegt. Denn von Kiel bis Koblenz ist Osnabrück bei Richtgeschwindigkeit in gut drei Stunden zu erreichen! Als Termin habe ich das nächste Frühjahr ins Auge gefasst, und zwar den

Samstag und Sonntag 25.+ 26. April 1987

Falls Ihr Interesse an einem solchen Treffen habt, lasst es mich bitte bald wissen, damit ich die notwendigen Vorbereitungen einleiten kann. Ein Pensionshaus in der Umgebung von Osnabrück wird sich zu einem ähnlichen Preis finden lassen wie bei unserer Jahreshauptversammlung. Und falls jemand mit der Bahn anzureisen wünscht, kann der Transport zur Versammlungsstätte auch organisiert werden.

Bleibt mir nur noch, auf eine große Beteiligung zu hoffen! Ich würde mich jedenfalls sehr freuen, Euch im nächsten Jahr in Osnabrück begrüßen zu können.

Sans-Martin Siephau

GRAFIK - SHORTY FUER MODELL 4/4P

```
=====
1 CLEAR.-4993:CLS:V=2944:PRINT CHR$(15):POKE 120,134:WHILE 1:D=D-5*(D=0):Y=Y+(Y<
71)*(D<4)-(Y>0)*(D>6):D=D MOD 3:X=X+(X<159)*(D=0)-(X>0)*(D=1):P=80*(Y&3)+X&2-204
8:C=PEEK(P):B=2^(2*Y MOD 6+X MOD 2):C=C*(C>127)*(C<192)AND-1+B*(M=2)OR 128-B*(M=
1)XOR-B*(M=3)
2 POKE P,C XOR B:A$=INKEY$:D=VAL(A$):N=INSTR("0XX.",CHR$(13),A$)+(A$=""):POKE P
,C:WHILE N>4:U=-V*(N=6):V=V-U:FOR I=-2048 TO-129:POKE I-U,PEEK(I-V):NEXT I:IF U TH
EN POKE 120,135:PRINT CHR$(14)ELSE N=1:WEND:IF D+N THEN M=N+(N>0)-M*(N=0):WEND E
LSE 2
```

m4/4p ecke

Kleines Grafikprogramm fuer Model 4/4P aus BOMicro 07/86/160

Mit diesem Zweizeiler lassen sich Grafiken mittels des Ziffernblocks erstellen. Die Ziffern 1-9 (ausser 5) werden fuer die entsprechende Richtung benutzt. Die Ziffer 0 benutzt man, um aus dem momentanen Modus zu kommen.

Den Modus waehlt man mit den Funktionstasten:
F1 - Zeichnen F2 - Loeschen F3 - Invertieren

Mit ENTER wird die erstellte Grafik gespeichert: mit
SYSTEM"DUMP Filename (START=60544,end=62463)"
wird die Grafik auf die Disk gespeichert. Mit
SYSTEM"LOAD Filename"

kann man die Grafik wieder von der Disk laden und innerhalb des Programms mit < . > aufrufen.

Hinweis zur Programmeingabe:
die drei "XXX" in Zeile 2 werden durch das jeweilige Zeichen der Funktionstasten F1,F2 und F3 ersetzt (es handelt sich dabei um Pixel)

So und nun viel Spass beim Grafikeln

Klaus Hermann

m4/4p ecke

TANDY - LOGO (TRSDOS 6.x)

=====

Nicht jedem gefällt wohl das Tandy-Logo, das beim booten auf der Bildfläche erscheint. Außerdem sieht man sich mit der Zeit satt daran. Der eine oder andere hätte vielleicht auch sein eigenes Erkennungszeichen oder ähnliches auf dem Schirm. Deshalb eine kurze Anleitung für eine Änderung.

Das Logo ist in den letzten 3 Records von SYS0/SYS abgespeichert. Mit einem entsprechenden Programm oder auch mit DEBUG lassen sich die Grafikzeichen leicht in Leerzeichen oder andere Grafikzeichen umwandeln. Es ist jedoch unbedingt erforderlich, daß die Ladeadressen nicht überschrieben werden, ansonsten hängt sich das System beim booten auf. Die Zeichen für das Logo sind in den Listings durchgestrichen.

Die Copyright Meldung steht ebenfalls in SYS0/SYS (Record 7). Die Verfahrensweise für eine Änderung ist dieselbe wie oben bereits beschrieben.

So und zuletzt noch die Zeile mit der Meldung "LS-DOS62Level-AN". Diese Meldung steht in BOOT/SYS, Record 2 ab Byte 10H.

Nach einer Änderung in Leerzeichen hat man nun nur noch einen leeren Bildschirm vor sich. Datum- und Zeiteingabe lassen sich mit dem SYSTEM-Befehl abschalten.

Klaus Hermann

SYS0/SYS Drive 0 Record X'0007'

0123456789ABCDEF	BYTE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
.....<.....0.	<00>	CD	26	1A	47	07	07	07	E6	07	3C	CD	3B	19	C5	4F	06
...#.B.....Day.	<10>	0B	AF	CB	23	17	B9	38	02	91	1C	10	F6	4F	7B	59	D1
..W...+...oz(....7	<20>	09	57	3E	04	CD	2B	1A	CE	6F	7A	2B	01	87	C9	01	37
..LS-DOS 06.02.0	<30>	0D	F8	4C	55	2B	44	4F	E3	20	30	36	2E	30	32	2E	30
OD- Copyright 19	<40>	30	44	2B	20	43	6F	70	79	72	69	67	6B	74	20	34	39
84 Logical Syste	<50>	3B	34	20	4C	6F	67	67	65	61	6E	20	53	77	73	74	65
ms Inc..50.All R	<60>	6B	73	20	49	6E	67	2E	01	35	5E	F8	41	6C	6C	20	52
ights Reserved.	<70>	69	67	6B	74	73	20	52	65	73	65	72	73	65	64	2E	20
Licensed to Tand	<80>	4E	69	63	65	6E	73	65	64	20	74	6F	20	54	61	6E	64
y Corporation...	<90>	79	20	43	6F	72	70	6F	72	61	74	69	6F	6E	2E	01	02
....(B.E.#.=.	<A0>	00	01	F2	1B	2B	06	42	06	3B	06	45	06	23	06	3D	06
v.5...-.....0.(.	<B0>	79	19	35	06	85	05	2D	05	03	05	00	05	30	05	2B	05
.....	<C0>	D0	0B	82	03	87	19	AF	07	8D	07	89	06	08	1B	0B	1B
..8.a.....	<D0>	F4	1A	7E	19	7B	19	0F	1B	A0	19	F5	1C	DA	1C	D7	1C
.....	<E0>	EB	1C	D0	1C	93	19	AF	19	AC	19	F4	1A	F4	1A	F4	1A
.....	<F0>	F4	1A	B5	19	BC	19	C0	19	C4	19	C8	19	CC	19	D0	19

SYS0/SYS Drive 0 Record X'0010'

0123456789ABCDEF	BYTE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
.....<.....0.	<00>	00	01	82	80	03	00	00	C5	3A	7C	00	CB	5F	C4	8C	03
...#.B.....Day.	<10>	C1	0B	7B	B1	20	FB	C9	C5	E5	7B	E6	07	07	21	D1	03
..W...+...oz(....7	<20>	4F	7B	06	00	09	4E	23	6E	0F	0F	0F	E6	1F	3C	67	3A
..LS-DOS 06.02.0	<30>	7C	00	E6	0B	28	06	CB	24	CB	25	CB	21	F3	E5	41	3E
OD- Copyright 19	<40>	01	D3	90	10	FE	41	3C	D3	90	10	FE	2D	20	F0	E1	25
84 Logical Syste	<50>	20	EB	FB	E1	C1	C9	50	4C	56	4B	5C	44	62	40	6B	3C
ms Inc..50.All R	<60>	72	3B	79	34	80	30	01	00	00	7E	D6	30	D8	FE	0A	D0
ights Reserved.	<70>	C5	E3	29	29	09	29	06	00	4F	09	44	4D	E1	23	1B	E9
Licensed to Tand	<80>	F3	AF	D3	B4	C7	01	26	57	F9	02	AF	BF	BF	BF	BF	BF
y Corporation...	<90>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
....(B.E.#.=.	<A0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
v.5...-.....0.(.	<B0>	F9	0B	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<C0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
..8.a.....	<D0>	A1	04	01	21	FA	F9	02	AF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<E0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<F0>	9B	A1	00	9B	01	01	1D	4C	FA	04	BF	BF	BF	BF	BF	BF

SYS0/SYS Drive 0 Record X'0011'

0123456789ABCDEF	BYTE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
.....<.....0.	<00>	BF	BF	BF	BF	AA	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
...#.B.....Day.	<10>	02	0B	A1	06	01	1B	9D	FA	02	AF	BF	BF	BF	BF	BF	BF
..W...+...oz(....7	<20>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
..LS-DOS 06.02.0	<30>	04	01	17	EF	FA	0B	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
OD- Copyright 19	<40>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
84 Logical Syste	<50>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
ms Inc..50.All R	<60>	04	01	11	92	FB	AB	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
ights Reserved.	<70>	96	AB	04	96	01	11	E2	FB	96	AB	04	96	01	11	E2	FB
Licensed to Tand	<80>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
y Corporation...	<90>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
....(B.E.#.=.	<A0>	7F	FC	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
v.5...-.....0.(.	<B0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<C0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
..8.a.....	<D0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<E0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
.....	<F0>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF

SYS0/SYS Drive 0 Record X'0012'

0123456789ABCDEF	BYTE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
.....<.....0.	<00>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
...#.B.....Day.	<10>	A1	0B	92	04	05	00	01	23	B9	FD	0B	BF	BF	BF	BF	BF
..W...+...oz(....7	<20>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
..LS-DOS 06.02.0	<30>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
OD- Copyright 19	<40>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
84 Logical Syste	<50>	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF	BF
ms Inc..50.All R	<60>	02	02	00	1E	00	00	00	00	00	00	00	00	00	00	00	00
ights Reserved.	<70>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
Licensed to Tand	<80>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
y Corporation...	<90>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
....(B.E.#.=.	<A0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
v.5...-.....0.(.	<B0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
.....	<C0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
..8.a.....	<D0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
.....	<E0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
.....	<F0>	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

HEFT
15
September
1986

42

Wie schon im letzten Info erwähnt, muß man das Model 4P, wenn es als Model 3 laufen soll, mit einem "ROM" füttern. Dazu liefert Tandy das File MODEL3/III auf einer TRSDOS 1.3-Diskette mit. Diese legt man in den Bootdrive und drückt RESET (ev. Optionen könnt ihr im letzten Info nachlesen!). Die Prozedur des Ladens dauert ziemlich lange, was angesichts des recht kurzen Laders im Boot-ROM nicht weiter verwunderlich ist.

Dieser Umstand allein hätte mich aber bestimmt nicht dazu gebracht, einen anderen Weg zum Laden des ROM's zu suchen. Den eigentlichen Antrieb erhielt ich erst, als ich feststellen mußte, daß Tandy aus mir unerfindlichen Gründen nicht etwa das Original Model 3-, sondern ein geändertes ROM mitliefert! Wäre dieses "ROM" eine Verbesserung des M. 3-ROM, könnte man die Sache ja noch verstehen, aber das Gegenteil scheint der Fall zu sein!

Während Tandy im Original Model 3-ROM z.B. eine Abfrage der Funktionstasten vorgesehen (aber nicht aktiviert) hat, ist diese Möglichkeit für's Model 4 nicht mehr vorgesehen. Noch unsinniger erscheint dieser Umstand wenn man weiß, daß das Model 3 keine Funktionstasten eingebaut hatte, die bekam erst das Model 4 spendiert. Ähnlich verhält es sich mit der Druckroutine des Model 3. Hier haben die ROM-Macher eine Tabelle zum Umsetzen der internen Codes in die entsprechenden DruckerCodes eingebaut. Diese Maßnahme erscheint relativ unsinnig, da man die Tabelle im ROM ja nicht ändern kann. Beim Model 4 ist diese Tabelle nicht mehr zu finden, obwohl sie hier wohl angebracht (weil im "Pseudo-ROM" änderbar!) gewesen wäre.

Es gibt noch einige solcher Beispiele, die aus Platzgründen nicht alle hier aufgezählt werden sollen. Erwähnen will ich nur noch, daß auch manche Programme nur mit den Original-ROM's laufen. So verweigert z.B. TASMOM (für Model 3) seine Arbeit sowohl mit dem MODEL3- als auch mit dem MODEL4-ROM und selbst mit einem Original Model 3-ROM für die deutsche Tastatur gibt es erhebliche Probleme. Richtig funktioniert das Programm nur mit dem amerikanischen Original-ROM des Model 3. Grund genug finde ich, die verschiedenen ROM's auf einer Diskette bereitzuhalten und je nach den gegebenen Bedürfnissen zu wählen, welches man benutzen will!

Leider sieht das Boot-ROM des 4p diese Möglichkeit nicht vor. Es erkennt immer nur das File mit dem Namen "MODEL3/III" als das zu ladende und ignoriert alle anderen ROM-Images. Man müsste also, um immer auf alles gefasst zu sein, vier "ROM"-Disketten bereithalten. Diese Platzverschwendung veranlasste mich, eine Lösung für das Problem zu suchen (und zu finden).

Wie oben schon erwähnt, ist der ROM-Lader nicht gerade der Schnellste. So lädt z.B. TRSDOS 6.2 oder CP/M ein 12k langes Programm (und nichts anderes ist das ROM-Image) in weniger als einem Drittel der Zeit, die der Lader dazu braucht. Aus dieser Erkenntnis heraus entstand folgendes Konzept:

1. Alle verfügbaren ROM-Images werden auf einer TRSDOS 6.2 - Systemdiskette gespeichert und
2. von hier in den Speicher geladen und gestartet!

Probleme bei der Realisierung des Konzeptes:

1. Die ROM-Images können nicht direkt in ihren tatsächlichen Speicherbereich (0000h - 3000h) geladen werden, da hier TRSDOS residiert und überschrieben werden würde!

2. TRSDOS benutzt eine andere Speicheraufteilung als die, die für den Model 3 - Modus notwendig ist!
3. Der Stackpointer des TRSDOS liegt mitten im späteren ROM-Bereich und kann dort ganz schön für Durcheinander sorgen!

Lösungen:

zu 1.:

Die ROM's wurden mittels eines kleinen Maschinenprogramms (siehe Listing 1) nach oben (7000h) verschoben. Dazu wird das entsprechende ROM auf die alte Weise geladen, NewDOS gebootet und dann ROMSHIFT/CMD gestartet. Danach wird TRSDOS 6.2 gebootet und das ROM unter dem entsprechenden Namen auf die Diskette gedumpt.

Beispiel: DUMP ROM1/CIM (START=X'7000', END=X'A000h').

Die Original Model 3-ROM-Images muß man sich natürlich zuvor auf ähnliche Weise aus einem Model 3 beschaffen!

Die ROM's werden später von dem Programm "ROM-Lader" (siehe Listing 2) auf Befehl geladen und, nachdem die zu bootende Model 3-Diskette eingelegt wurde, wieder auf ihren richtigen Platz verschoben und gestartet (Z. 76 - 82)!

zu 2.:

Das Model 4p (und natürlich auch das Tischmodell) wurde, um mit dem Model 3 kompatibel zu bleiben, mit einer raffinierten Bankinglogik ausgestattet. Über Steuerports kann man die RAM-Konfigurationen so verändern, daß entweder die Memory-Map des M. 3 (ROM bis 3000h, danach Keyboard, Videospeicher und dann RAM bis FFFFh) eingestellt ist oder aber die Tastatur und der Video-RAM nach oben "unters Dach" (2k Video-RAM von F7FFh - FFFFh, Tastatur von F3FF-F7FE) verschoben wird. Das ist die Konfiguration für TRSDOS und CP/M.

Um das Model 4p aus dem TRSDOS-Modus in den M. 3-Modus zu versetzen, muß nach dem Verschieben des ROM-Images auf den Port 84h der Wert 00 ausgegeben (Z. 80 und 81) werden (06 im TRSDOS-Mode).

zu 3.:

Um dem Stack keine Möglichkeit zu geben im "ROM" Unheil anzurichten (wie einigemal bei Vorversuchen passiert!), wird der Stackpointer (CPU-Register SP), kurz vor dem Verschieben des ROM-Images auf seinen alten Platz, nach oben (E000h) verlegt.

Zu dem Programm in Listing 2 ist eigentlich nicht viel zu sagen, es erklärt sich mittels der Kommentare praktisch von selbst. Eine kurze Anmerkung noch zu den sog. Supervisor-Calls. Das sind für den Programmierer nutzbare DOS-Routinen, ähnlich denen in NewDOS oder im BASIC-ROM, die allerdings nicht wie dort üblich mit "CALL ADDR" aufgerufen werden. Um einen Supervisor-Call aufzurufen wird dessen Nummer in den Accu geladen und dann ein ReStart 28h durchgeführt. Sind Parameter an den SVC zu übergeben, so geschieht das über die anderen CPU-Register (zum Beispiel Zeilen 13 u. 61 - 64).

Kartmut Overmann

Listing 1:

10	ORG	5200h	
5200	210070	20	LD HL,0000H ;ROM-Image von 0000h
5203	110000	30	LD DE,7000H ;nach 7000h
5206	010038	40	LD BC,3800H
5209	EDB0	50	LDIR ;verschieben!
520B	C32D40	80	JP 402DH ;Rücksprung ins DOS
5200		90	END START

m4/4p ecke m4/4p ecke

Listing 2

```

00001 ;*****
00002 ;*           Model 4p - ROM-Lader           *
00003 ;*   Lädt verschiedene ROM-Versionen von einer TRSDOS 6.x - Diskette   *
00004 ;*****
00005 ;* Hartmut Obermann, Schwalbacher Str. 6, 6209 Heidenrod, 06124 / 3913 *
00006 ;*****
00007 ;
00008 ORG     5000H           ;Hier (oder anderswo) solls sein!
00009 ;
00010 ;***** Supervisor-Calls *****
00011 ;
00012 $DSPLY: EQU     10           ;Text auf Screen
00013 $FSPEC: EQU     78           ;FCB erstellen
00014 $KEY:   EQU     1           ;Tastatur abfragen
00015 $LOAD: EQU     76           ;File laden
00016 $CLS:  EQU     105          ;Bildschirm löschen
00017 ;
00018 ;***** Definitionen *****
00019 ;
00020 FCB:   DEFS     32           ;Platz für FileControlBlock
00021 ROM:   DEFM     'ROM'       ;Name des einzulesenden Files
00022 NUM:   DEFM     ' '         ;Nummer des einzulesenden Files
00023       DEFM     '/CIM'       ;und dessen Extension
00024       DEFB     ODH          ;Ende des Filenames
00025 TEXT1: DEFM     'Model 4P - ROM-Lader' ;Hier steht was,
00026       DEFB     OAH
00027       DEFM     '-----'
00028       DEFB     OAH
00029       DEFM     'C by Hartmut Obermann' ;und von wem es ist!
00030       DEFB     OAH,OAH
00031       DEFM     'MODEL A/III   = 1' ;Und was man machen kann!
00032       DEFB     OAH
00033       DEFM     'MODEL D/III   = 2'
00034       DEFB     OAH
00035       DEFM     'MODEL 3US/ROM = 3'
00036       DEFB     OAH
00037       DEFM     'MODEL 3GE/ROM = 4'
00038       DEFB     OAH,ODH      ;Ende des Textes
00039 TEXT2: DEFM     'ROM-Image ist geladen!
           Diskette wechseln und <ENTER> drücken!!!'
00040       DEFB     ODH
00041 ;
00042 ;***** Hautprogramm *****

5108 3E69 00043 ;
510A EF   00044 START: LD     A,$CLS           ;Bildschirm löschen
510B 212950 00045 RST     28H
510E 3E0A 00046 LD     HL,TEXT1          ;und Text ausgeben!
5110 EF   00047 LD     A,$DSPLY
5111 3E01 00048 RST     28H
5113 EF   00049 ASK:  LD     A,$KEY           ;Tastatur abfragen,
5114 FE31 00050 RST     28H
5116 280C 00051 CP     '1'
511B FE32 00052 JR     Z,LOAD           ;bis eine Taste zwischen 1 und 4
511A 2808 00053 CP     '2'
511C FE33 00054 JR     Z,LOAD
511E 2804 00055 CP     '3'
5120 FE34 00056 JR     Z,LOAD
5122 20ED 00057 CP     '4'
5124 212350 00058 JR     NZ,ASK           ;gedrückt wurde!
5127 77   00059 LOAD: LD     HL,NUM          ;Nummer des zu ladenden Files
512B 212050 00060 LD     (HL),A           ;in den Filenames einfügen!
512B 110050 00061 LD     HL,ROM          ;FileControlBlock initialisieren
512E 3E4E 00062 LD     DE,FCB
5130 EF   00063 LD     A,$FSPEC
5131 210050 00064 RST     28H
5134 3E4C 00065 LD     HL,FCB           ;und File laden!
5136 EF   00066 LD     A,$LOAD
5137 21BF50 00067 RST     28H
513A 3E0A 00068 LD     HL,TEXT2          ;Aufforderung "<ENTER> drücken!"
513C EF   00069 LD     A,$DSPLY          ;anzeigen und
513D 3E01 00070 RST     28H
513F EF   00071 LOOP: LD     A,$KEY          ;warten bis <ENTER>
5140 FE0D 00072 RST     28H
5142 20F9 00073 CP     ODH
5144 3100E0 00074 JR     NZ,LOOP          ;gedrückt wurde!
5147 210070 00075 LD     SP,0E000H          ;Stack nach "oben" verlegen!
514A 110000 00076 LD     HL,7000H          ;ROM-Image von 7000h
514D 010038 00077 LD     DE,0000H          ;nach 0000h
5150 EDB0 00078 LD     BC,3800H
5152 3E00 00079 LDIR          ;verschieben!
5154 D384 00080 LD     A,00H           ;Operation-Port richtig
5156 C30000 00081 OUT     (84H),A          ;initialisieren
5108      00082 JP     0000H          ;und "ROM" starten!!!
510B      00083 END     START
00000 Total errors

```

HEFT
15
September
1986

Nach der Lektüre des Artikels von Arnulf Sopp "Wenn die Uhr mal stört" (Info Nr. 14 Seite 39ff), war ich natürlich höchst wissbegierig, wie die Sache sich auch beim Model 4p im Model 3-Mode realisieren läßt. Also nichts wie ran an den Speck!

Aber schon beim Ausprobieren der Funktion CLOCK,Y bekam ich erstmal große Augen; es tat sich (fast) nichts! Die Floppy lief zwar an und auch die eigentliche Aufgabe des Befehls, nämlich ein Flagbit im Byte 4210h zu setzen, wurde (wie ich später nach längerer Spurverfolgung herausfand) einwandfrei erfüllt, aber die Uhrzeit erschien trotz dem nicht auf dem Bildschirm. Ein Fehler im DOS!? Kann nicht sein, wenn ich das MODEL4/III-ROM (die Model 4p-ROM-Version für die amerikanische Tastatur) benutze, läuft die Sache einwandfrei. Also ein Bug in der deutschen Version des ROM-Image! Aber wo?

Der Fehler ist schnell (alles relativ!) gefunden. Zum Glück stimmt das Model 4p-ROM gerade an dieser Stelle einigermaßen mit dem Original Model 3-ROM überein, so daß das ROM-Listing (Grosser/Röckrath) weiterhelfen kann. Ein einziges Byte ist, durch welchen unglücklichen Umstand auch immer, um den Wert 5 zu niedrig geraten. Um die Uhr wieder zum Laufen zu bekommen, muß also nur an der Speicherstelle 355Dh aus dem Wert 19h ein 1Eh gemacht werden. Das macht man aber nicht im Speicher (da geht's zwar auch, aber nur mit einem kleinen Trick), sondern direkt im ROM-Image-File, damit nicht nach jedem Neuladen der Ärger wieder anfängt!

Und schon geht der Ärger wieder los! Das MODEL4/III-File befindet sich auf einer Diskette im TRSDOS 1.3 - Format. Dort kann es nicht ohne weiteres bearbeitet werden, da weder TRSDOS 6.2 noch NewDOS direkten Zugriff auf dieses Format haben und unter TRSDOS 1.3 gibt es keinen Diskeditor (bzw. ich kenne keinen). Also muß das File erst von seiner Originaldiskette auf NewDOS, und nach der Änderung wieder zurück kopiert werden. Dazu muß man den PDRIVE folgendermaßen einstellen:

TI=AM,TD=E,TC=40,SPT=18,TSR=3,GPL=6,DDSL=17,DDGA=2.
Vorsicht! Auch nach der Einstellung dieser Parameter kann man mit "DIR" nicht auf die Diskette schauen (dazu gibt es in der Tandy-Public-Domain das Programm TRSDIR) oder etwa mit SUPERZAP ans Ändern denken. Man kann sich aber mit "COPY MODEL4/III:1:0" das ROM-Image auf seine Floppy holen und nach der Änderung wieder zurückschreiben. Die eigentliche Änderung findet im relativen Sector 36h (54d), relatives Byte 34h (52d) statt. Dort wird, wie oben schon gesagt, statt 19h der Wert 1Eh eingetragen!

Nun aber endlich doch zu "UHRUNTEN"! Ich finde nämlich Arnulfs Idee gar nicht so schlecht, noch dazu da das Problem der Verlegung für 4p-Besitzer sehr einfach zu lösen ist. Man muß nur der ROM-Routine, die die Uhrzeit in die rechte obere Bildschirmcke schreibt, mitteilen, wo sie sie statt dessen hinpacken soll. Das könnte man natürlich auch mittels einer festen Änderung im ROM-Image machen, es gibt aber eine elegantere Möglichkeit.

Wie im Beitrag "ROM-Image-Loader" schon kurz erwähnt, ist das 4p mit einer raffinierten Bankinglogik ausgestattet. Die Logik kann über die Ports 84h (132d) und 9Ch (156d) gesteuert werden.

Mit Hilfe des Port 9Ch kann man das BOOT-ROM ein- bzw. ausblenden (siehe auch "BOOT-ROM des 4p", Info Nr. 14 Seite 63-65). Der Port 84h dient zur Steuerung der RAM-Bänke. An diesem Port interessiert uns zur Zeit nur das Bit 0, mit dem der Bereich zwischen 0000h und 37FFh schreibgeschützt (Bit 0 = 0) oder beschreibbar (Bit 0 = 1) gemacht werden kann. Normalerweise wird für den 3'er-Modus der Port mit dem Wert 00h geladen, also alle Bits auf 0 gestellt und damit das RAM zum Pseudo-ROM, sprich schreibgeschützt. Das kann man durchaus mal mit DEBUG oder vom BASIC aus mit POKE und PEEK probieren, alle Änderungsversuche werden fehlschlagen!

Will man, wie wir in unserem Fall, Änderungen am "ROM" vornehmen, setzt man das erste Bit von Port 84h, indem man ihn mit 01h lädt. Danach lassen sich ohne weiteres Speicherstellen, sowohl mit DEBUG als auch vom BASIC oder von Anwenderprogrammen heraus ändern. Aber Vorsicht! Allzuleicht zerstört man sich dabei die eigene Arbeitsgrundlage und der Rechner verschwindet "im Keller" und es hilft nur erneutes Laden des ROM-Image. Aus diesem Grunde sollte man auch nach kontrollierten Änderungen, wie wir sie jetzt vorhaben, wieder 00h in den Port 84h schreiben, um versehentliche ROM-Änderungen zu vermeiden.

Nun aber endlich zur Sache! Um die Uhr "nach unten" zu verlegen, müssen wir in Adresse 359Eh den gewünschten Wert, sagen wir z.B. 3FF8h, eintragen. Dann landet die Uhr ganz unten rechts, aber natürlich kann man jeden Wert innerhalb (und sogar ausserhalb) des Bildschirmspeichers eintragen, ganz nach eigenem Geschmack. Aber Vorsicht, auch hier lauert wieder einmal eine Gefahr. Wie immer in der Maschinensprache wird die Adresse, und nichts weiter ist der von uns eingetragene Wert, im Format LSB/MSB eingetragen. Hier zur Verdeutlichung zwei kurze Listings in BASIC und Assembler.

```
10 REM UHRUNTEN in BASIC für Model 4p
20 OUT 132,1 : REM Aus Pseudo-ROM mach RAM!
30 POKE 13726,248 : REM LSB der neuen Adresse setzen
40 POKE 13727,63 : REM MSB der neuen Adresse setzen
50 OUT 132,0 : REM Aus Sicherheitsgründen wieder ROM!
60 CMD"CLOCK,Y" : REM Effekt gleich mal ausprobieren!
```

5200	10	ORG	5200h	
5200	3E01	20 START	LD	A,00h ;Aus ROM
5202	D384	30	OUT	(84h),A ;mach RAM!
5204	21F83F	40	LD	HL,3FF8h ;die neue Adresse
5207	229E35	50	LD	(359Eh),HL ;eintragen und
520A	3D	60	DEC	A ;aus RAM wieder
520B	D384	70	OUT	(84h),A ;ROM machen. Dann
520D	C32D40	80	JP	402Dh ;zurück zum DOS!
	90	END	START	
0000 TOTAL ERRORS				

Natürlich muß man nicht zurück ins DOS springen, sondern kann gleich mit irgendeinem Programm (z.B. SCRIPSIT) weitermachen; aber da fallen euch bestimmt selbst genug Anwendungen ein.

Damit Schluß für diesmal! Viel Spaß beim Zäpen und Zapfen, euer

Karimut Obermann

m4/4p ecke

CALL um die Ecke

Was ich hier vorstellen möchte, ist in dieser Form nur auf dem Genie I/II oder vergleichbaren Computern lauffähig. Das Strickmuster, nach dem es aufgebaut ist, eignet sich jedoch für alle in unserem Club vertretenen Maschinen, überhaupt für alle Z80-Rechner. Es geht darum, Speicherbereiche zu erreichen, die ohne Umwege nicht verfügbar sind. Dazu gehören beispielsweise die Boot-EPRoMs der neueren Genies, aber auch Programme, die bei einfacheren Computern etwa mit Hilfe eines Bankers im Adreßbereich des ROMs, der Tastatur oder des Bildschirms laufen. Die hierfür notwendigen Umschaltsequenzen kann man sich schenken.

Vorbild waren dafür das Level-4-ROM von TCS (Genie I/II) und CP/M. In Genie-Sonder-ROM kann ein String auf den Bildschirm ausgegeben werden, der sich unmittelbar hinter dem Ausgabebefehl befindet. So braucht man keinen Zeiger zu laden. CP/M ruft alles Mögliche mit einem CALL nach 0005h auf, je nach dem, was in den Registern übergeben wird. Ähnlich ist es hier:

CALL	call0	is. Listing
DEFW	4080h	Unterprogramm-Adresse

Mit dieser Befehlssequenz wird ein Unterprogramm an der Adresse 4080h in der normalerweise nicht zugänglichen Bank 0 des G3s aufgerufen. Um das tun zu können, müßte man sonst zuerst die Bank 0 selektieren, dann die Unterprogrammadresse anspringen, zuletzt wieder in die Arbeitsbank (normalerweise 1) zurückkehren. Aber bereits beim Umschalten auf Bank 0 würde sich ein solches Programm selber in die ewigen Jagdgründe begeben, es wäre bei RET nicht mehr auffindbar.

Die hier vorgestellte Methode arbeitet im Common-Bereich, der unabhängig von der eingestellten Bank immer selektiert ist. Von hier aus kann die Umschaltung risikolos vorgenommen werden. Die häufiger gebrauchten Umschaltsequenzen liegen bereits als Unterprogramme vor, anders wurden (ab Label bank1) hier eigens angefügt. So ist es mit dem G3s nunmehr möglich, Unterprogramme in allen Speicherbereichen aufzurufen. Folgende Routinen besorgen die Selektion:

RAM "heben" den memory-mapped Devices:	06A0h
dieses RAM wieder ausblenden:	06A2h
RAM in Bank 0:	35D4h (Label bank0)
wieder Bank 1 selektieren:	35D5h (Label bank1)
Boot-EPRoM selektieren:	35E5h (Label rom)
Boot-EPRoM wieder ausknipsen:	35E6h (Label ram)
ein Unterprogramm in Bank 0 aufrufen:	35E9h (Label call0)
(Restoration des alten Zustandes nicht erforderlich)	

Das EPRoM oder der Speicher im memory-mapped-Adreßbereich sind mit einem gewöhnlichen CALL an die genannten Adressen erreichbar. Nur bei Bank 0 geht das aus den oben skizzierten Gründen nicht, daher ist die Routine call0 implementiert. Wenn sie mit einem CALL aufgerufen wird, steht die RET-Adresse im Stack. Es ist die Adresse unmittelbar hinter dem CALL. Nichts spricht dagegen, dort sonstwas unterzubringen, in unserem Fall die Adresse des gewünschten Unterprogramms. Hiervon geht die Routine call0 aus. Deshalb erhöht sie die RET-Adresse auf dem Stack um zwei Stellen, um die CALL-Adresse zu überspringen. Bei der Rückkehr aus dem Unterprogramm wird nun wieder sinnvoller Code vorgefunden.

Alle genannten Routinen, auch call0 und die zitierten System-Unterprogramme können gefahrlos aufgerufen werden, auch unter BASIC mit USR. Der User sollte jedoch ziemlich genau wissen, was er da tut. So knipst er sich beispielsweise mit dem Programm an 06A0h (s. o.) die Tastatur und den Bildschirm aus. Da hilft nur noch abschalten. Daher muß der Anwender in seinen Programmen immer für eine Restoration des alten Zustands sor-

gen. Die zuständigen Routinen sind oben benannt. Auch der Interrupt-Status muß bedacht werden. Alle aufgezählten Routinen, die einen besonderen Speicherbereich selektieren, schalten die Interrupts ab. Unter denen, die den Normalzustand restaurieren, läßt nur 06ABh (s. o.) sie anschließend wieder zu. Der User muß deshalb selbst für den gewünschten INT-Status sorgen (unter BASIC mit CMD"I" und CMD"R").

Für andere Computer sind diese Adressen ungültig. Der Leser muß sich jeweils darüber informieren, wie es bei seiner eigenen Maschine aussieht. Dazu gehört natürlich auch die Suche nach einem freien Speicherplatz, an dem neu hinzukommende Routinen untergebracht werden können. Für die weitaus meisten von euch dürfte es der Adreßraum ab 3000h sein, der beim Genie I/II mit dem ES 64 MPA nutzbar gemacht werden kann.

Arnulf Sopp

35E9	00001	ORE	35bbh	hier freier Speicher
	00002			
	00003	;ein Unterprogramm in Bank 0 anspringen		
35EB	00004	call0	EX	(SP),HL ;HL retten, Stack laden
35ED	00005	PUSH	DE	;retten
35ED	00006	LD	E,(HL)	;E <- (Stack)
35EE	00007	INC	HL	;auf Stack + 1
35EF	00008	LD	D,(HL)	;D <- (Stack + 1)
35F0	00009	INC	HL	;auf RET-Adresse
35F1	00010	LD	(calladr),DE	;CALL-Adresse patchen
35F5	00011	POP	DE	;DE restaurieren
35F6	00012	EX	(SP),HL	;dto. HL. RET-Adr. -> St.
35F7	00013	CALL	bank0	;Bank 0 selektieren
35FA	00014	CALL	0000h	;dort Unterprg. aufrufen
35FB	00015	calladr	EQU	;-2 ;(Adresse variabel)
	00016			
	00017	;verschiedene Speicherbereiche zugänglich machen		
35FD	00018	bank1	PUSH	AF ;retten
35FE	00019	IN	A,(0Fh)	;Systembyte 0
35FF	00020	SET	6.A	;Bank 1 selektieren
3600	00021	JR	exit0	;dort weiter
	00022			
3604	00023	bank0	PUSH	AF ;retten
3605	00024	IN	A,(0Fh)	;Systembyte 0
3607	00025	AND	3fh	;Bank 0 selektieren
3609	00026	DI		;INTs sperren
360A	00027	exit0	OUT	(0Fh),A ;Systembyte neu schreiben
360C	00028	POP	AF	;restaurieren
360D	00029	RET		
	00030			
360E	00031	ram	PUSH	AF ;retten
360F	00032	IN	A,(0Fh)	;Systembyte 1
3611	00033	SET	2.A	;Boot-EPRoM ausschalten
3613	00034	JR	exit1	;dort weiter
	00035			
3615	00036	rom	PUSH	AF ;retten
3616	00037	IN	A,(0Fh)	;Systembyte 1
3618	00038	REB	2.A	;Boot-EPRoM selektieren
361A	00039	DI		;INTs sperren
361B	00040	exit1	OUT	(0Fh),A ;Systembyte neu schreiben
361D	00041	POP	AF	;restaurieren
361E	00042	RET		
	00043			
0000	00044	END		

00000 Fehler

HEFT
15
September
1986

58

Korrektur zu "CALL um die Ecke"

Man sollte wirklich auch den selbstverständlichsten Kleinkram dreimal checken und ausprobieren, bevor man unschuldige Leser damit frustet. In meinem Programm, das in dem oben zitierten Artikel vorgestellt wird, steckt ein fataler Fehler, der am Ende der Routine die Rückkehradresse in der eigenen Bank sucht - und fatalerweise sogar findet! Es fragt sich bloß, was für eine absonderliche Adresse das ist.

In der hier korrigierten Form läuft das Programm aber einwandfrei, denn der Stack wird nach Common verlagert. Da sich nun aber die Adressen hinter der Korrektur verschieben, ist der ganze Fest gelistet. Die veränderten Adressen für die in der Tabelle genannten Umschaltroutinen lauten jetzt so:

bank0 (die Bank 0 selektieren) 35D7h
 ram (das Boot-EPROM ausschalten) 35E8h
 rom (das Boot-EPROM selektieren) 35EFh

```

00017 :verschiedene Speicherbereiche zugänglich machen
00018 bank1 PUSH AF ;retten
00019 IN A,(0f9h) ;Systembyte 0
00020 SET 6,A ;Bank 1 selektieren
00021 LD SP,0000h ;Stack restaurieren
00022 spbuf EQU #-2 ;Adreß-Operand variabel
00023 JR exit9 ;dort weiter
00024
00025 bank0 PUSH AF ;retten
00026 LD (spbuf),SP ;Stackpointer retten
00027 LD SP,37d0h ;eigener Stack im Common
00028 IN A,(0f9h) ;Systembyte 0
00029 AND 3fh ;Bank 0 selektieren
00030 DI ;INTs sperren
00031 exit9 OUT (0f9h),A ;Systembyte neu schreiben
00032 POP AF ;restaurieren
00033 RET
00034
00035 ram PUSH AF ;retten
00036 IN A,(0fah) ;Systembyte 1
00037 SET 2,A ;Boot-EPROM ausschalten
00038 JR exita ;dort weiter
00039
00040 rom PUSH AF ;retten
00041 IN A,(0fah) ;Systembyte 1
00042 RES 2,A ;Boot-EPROM selektieren
00043 DI ;INTs sperren
00044 exita9 OUT (0fah),A ;Systembyte neu schreiben
00045 POP AF ;restaurieren
00046 RET

```


Real Time Black Box?

Für das Genie III s gibt es eine Echtzeituhr (real time clock, RTC), auf der der Uhren- und Kalenderchip MSM 5832 steckt. Da TCS bzw. Phönix aus Gründen der Wirtschaftlichkeit vermutlich dieselbe Modellpolitik betreibt wie VW (Audi und VW unterscheiden sich hauptsächlich am Firmensignet), ist anzunehmen, daß noch weitere Genies diese Uhr haben. Es macht mich nicht besonders stolz, den UHR- oder DATUM-Befehl korrekt eintippen zu können. Vielmehr muß es möglich sein, ohne Unterstützung des DOS die Uhr in eigenen Programmen auszulesen.

Zu diesem Zweck fragte ich bei Phönix an, erhielt aber außer dem Versprechen, man wolle sich erkundigen, keine Reaktion. So blieb mal wieder nichts anderes übrig, als die Uhr-Routinen des Betriebssystems zu analysieren. Willkommene Unterstützung bot ein Auszug eines Artikels über eine andere Uhr mit dem MSM 5832, woraus ich die Bedeutung der internen Register (s. Tabelle am Ende dieses Textes) ersehen konnte. Die Angabe der Quelle muß ich leider schuldig bleiben, weil ich sie selbst nicht kenne.

Der MSM 5832 hat 13 Register, in denen die in der Tabelle genannten Zeitinformationen im BCD-Format vorliegen. Das bedeutet, daß alle Zeiteinheiten (außer dem einstelligen Wochentag) als zweistellige Dezimalzahlen schreib- bzw. lesbar sind. Für eine Digitaluhr mit LCD-Anzeige ist das zweifellos von Vorteil. In binär "denkenden" Computern wäre allerdings eine duale Codierung sinnvoller. Nun ja, ist nun mal so.

Um zum Stellen oder Auslesen der Uhr die entsprechenden Register zu erreichen, müssen den Chip ihre 4 Bit breiten Adressen mitgeteilt werden. Zugleich muß eines von zwei weiteren Bits auf 1 stehen, um die Uhr entweder zum Lesen oder zum Schreiben vorzubereiten. Offenbar wird sie vollständig deaktiviert (läuft aber intern weiter), wenn als Abschluß einer Schreib-/Leseoperation beide Bits rückgesetzt werden. Über diesen Punkt bin ich mir nicht ganz sicher. Jedenfalls wird in der Genie-Routine so gefahren.

In Serie III s ist diese Ansprache der Zielregister folgendermaßen geset: Der Port 55 erhält einen Output mit dem Bitmuster 10011010. Die vier Bits beinhalten die Adresse des gewünschten Registers. Die Bits 7 und 8 unterscheiden zwischen Schreiben und Lesen. Beim Stellen (Schreiben) der Uhr lautet sie 01, beim Auslesen 10. So wird z. B. das Register 9 (Einerstelle der Monate) zum Schreiben mit dem Output 10011010 (=9A) auf den Port 55 adressiert. 9C würde das Lesen des Monats-Einers ermöglichen.

Nachdem das Zielregister angesprochen ist, kann die Zeiteinheit eingegeben oder ausgelesen werden. Hierfür ist beim Genie III s der Port 5A zuständig. Hierbei ist zu beachten, daß nur die unteren 4 Bit von Belang sind. Außerdem beinhalten die Register 5 und 8 zusätzliche Informationen über Schaltjahre, 12h- bzw. 24h-Anzeige sowie die Tageshalbe für die AM-PM-Anzeige. Die letztlich relevanten Bits müssen deshalb von der Anzeige maskiert werden. Umgekehrt müssen beim Stellen der Uhr diese Zusatzbits je nach den gesetzt oder rückgesetzt sein.

In Anschluß an diesen Text sind die beiden Routinen gelistet, die in Serie III s die Uhr stellen bzw. lesen (Labels wrlock und rdlock). Dabei handelt es sich um einen Auszug aus einem Listing des kompletten residenten Betriebssystems, das ich mir anfertigte, um die leider sehr bescheidene Dokumentation zu ergänzen. Daher fehlen einige Informationen, die mit LIST OFF ausgespart sind. Der Leser möge deshalb die jeweilige Ladeadresse der linken Spalte entnehmen. Labels, die nicht innerhalb des Listings definiert sind, ergeben sich aus der zweiten Spalte von links. Die Programmlogik dürfte sich aus den reichlichen Kommentaren ergeben.

Das Programm liegt "hinter" dem Bildschirm, ist also nicht ohne weiteres für Änderungen zugänglich. Zuvor muß auf den Port FA ein Byte mit gesetztem Bit 0 ausgegeben werden (also eine ungerade Zahl). Mit CALL 06A0 erledigt das auch das Betriebssystem. Nach der Manipulation wird mit CALL 06AB oder einem geradzahigen Output auf Port FA der Screen wieder zugeschaltet. Vorsicht mit diesem Port! Ein falscher Output kann die Arbeit von Stunden zunichte machen. Zuerst den Input lesen, dann das Bit 0 ändern, dann neu ausgeben!

Sagte ich weiter oben "Logik"? Der Algorithmus ist zwar wirklich ziemlich elegant und läßt sich kaum nennenswert verkürzen oder beschleunigen. Daß aber in Zeile 745 die Sekunden immer auf 0 gestellt werden, obgleich das DOS ohne weiteres die Eingabe von beispielsweise 13:54:32 zuläßt, ist nicht einzusehen. Es ist auch im Betriebssystem nicht vorgesehen, Schaltjahre zu berücksichtigen oder auch den Wochentag zu programmieren. Das ist peinlich, wenn CP/M ihn anzeigt - natürlich mit siebenfacher Chance den falschen. Mit dieser kleinen Routine läßt er sich ein- für allemal einstellen:

```

start  ORG      5200h
        CALL    06B5h      ;auf 1,78 MHz schalten
        LD      A,6ah      ;Register 6 schreiben
        OUT     (5bh),A     ;Befehl ausgeben
        LD      A,7h      ;Beispiel hier: Sonntag
        OUT     (5ah),A     ;Ausgabe an die Uhr
        XOR     A           ;A ← 00, Uhr "stillegen"
        OUT     (5bh),A     ;Befehl ausgeben
        JP      06beh      ;7,2 MHz und Ende
        END      start     ;port Einsprung
    
```

Die Triebfeder für mich, den Wecker auf die Couch zu legen, war zunächst meiner Wissensdurst. Der Leser möchte jedoch vielleicht etwas damit anfangen. Bei zeitkritischen Messungen kann beispielsweise das Register 0 via Foiling einen Sekundenzeit liefern, der von den Systeminterrupts unabhängig ist. Wenn für laienmännische Anwendungen die Wochentage interessieren, die vom DOS aus mit 1 abgerufen werden können, so sind sie, wie ich hoffe, nach meiner 8 Lektionen nun ebenfalls programmgesteuert verfügbar. Offen gestanden fällt mir im Moment nicht mehr dazu ein. Vielleicht lesen wir in nächste Info einen interessanten Vorschlag von einem anderen Mitglied?

Tabelle zur Adressierung des MSM 5832:

Register	Datenbits				Bedeutung
	D7	D6	D5	D0	
0	x	x	x	x	Sekunden Einer
1	x	x	x	x	Sekunden Zehner
2	x	x	x	x	Minuten Einer
3	x	x	x	x	Minuten Zehner
4	x	x	x	x	Stunden Einer
5	a	b	x	x	Stunden Zehner *)
6	x	x	x	x	Wochentag (Montag = 0)
7	x	x	x	x	Datum Einer
8	x	c	x	x	Datum Zehner *)
9	x	x	x	x	Monat Einer
A	x	x	x	x	Monat Zehner
B	x	x	x	x	Jahr Einer
C	x	x	x	x	Jahr Zehner

*) a: Register 5, Bit 0 gesetzt im 24h-Format
 b: Register 5, Bit 1 gesetzt ist Mittag (PM-Flag)
 c: Register 8, Bit 2 gesetzt in einem Schaltjahr

Arnulf Sepp

HEFT
 15
 September
 1986

54


```

00718 ; Routinen zum Stellen und Lesen der Hardware-Uhr des 63e
00719 ; (C) by TCS Computer GmbH
00720 wrclck CALL lsp ; auf 1,78 MHz umschalten
00721 LD HL,time ; Uhrzeit, Sekunden (RAM)
00722 LD C,02h ; 2 Durchl. (Zeit, Datum)
00723 LD A,23h ;= INC HL
00724 LD (indec1),A ;port patchen
00725 LD D,0ah ;WR ab RTC-Reg. 0 (Sek.)
00726 wr3val LD B,03h ;Zähler drei Werte
00727 wr1val LD A,(HL) ;Sekunden usw. laden
00728 EXX ;Register retten
00729 LD L,A ;HL <- Sekunden
00730 LD H,00h ;als 16-Bit-Wort
00731 LD A,0ah ;Divisor 10
00732 CALL diva ;Sekunden/10, Einer in A
00733 EXX ;Register tauschen
00734 CALL wrrtc ;RTC-Register schreiben
00735 EXX
00736 LD A,L ;Sekunden, Zehner
00737 EXX
00738 CALL wrrtc ;RTC-Register schreiben
00739 indec1 INC HL ;Uhrzeit Minuten usw.
00740 DJNZ wr1val ;bis ss/mm/hh fertig
00741 LD D,7ah ;ab RTC-Reg. 7 (Datum)
00742 LD A,2bh ;= DEC HL
00743 LD (indec1),A ;port patchen
00744 LD L,4ah ;Datum Tag (RAM 4046)
00745 DEC C ;von Zeit auf Datum
00746 JR NZ,wr3val ;falls noch nicht geschr.
00747 XOR A ;A <- 0. 0 Sek. (Quatsch)
00748 LD D,0ah ;RTC-Reg. 0 schreiben
00749 CALL wrrtc ;RTC-Schreibroutine
00750 CALL hsc ;auf 7,2 MHz umschalten
00751 rdclck PUSH IX ;retten
00752 LD IX,clktab-1 ;Differenzmaske-Tabelle
00753 LD L,4ah ;Datum Jahr (RAM 4044)
00754 LD C,02h ;2 Durchl. wie oben
00755 LD A,23h ;= INC HL
00756 LD (indec2),A ;port patchen
00757 LD D,00ch ;ab RTC-Reg. 12 (Jahr)
00758 rd3val LD E,00h ;Zähler 3 Werte
00759 rd1val CALL rrtc ;RTC auslesen (Zehner)
00760 ADD A,A ;*2
00761 LD E,A ;sichern
00762 ADD A,A ;*4
00763 ADD A,A ;*8
00764 ADD A,E ;*10
00765 LD E,A ;sichern
00766 CALL rrtc ;RTC auslesen (Einer)
00767 ADD F,E ;= Zehnerstelle
00768 LD (HL),A ;in den Puffer (RAM 4040)
00769 indec2 DEC HL ;1 Stelle tiefer
00770 DJNZ rd1val ;bis 3 Werte fertig
00771 LD D,5ch ;ab RTC-Reg. 5 (Stunden)
00772 LD A,2bh ;= DEC HL
00773 LD (indec2),A ;alten Befehl restaur.
00774 LD L,43h ;Zeit Stunden (RAM 4043)
00775 DEC C ;von Datum auf Zeit
00776 JR NZ,rd3val ;falls noch nicht gelesen
00777 POP F ;restaurieren
00778 XOR A ;A <- 00, RTC rücksetzen
00779 OUT (5bh),A ;auf RTC-Adresse port
00780 RET
00781 rdrtc LD A,D ;RTC-Adressreg., RD/WR-Bit

```

```

3F0B D35B 00782 OUT (5bh),A ;Register adressieren
3F0D 1610 00783 LD D,10h ;1 tiefer im ober. Nibble
3F0F 92 00784 SUB D
3F10 57 00785 LD D,A ;D <- neues Register
3F11 DD23 00786 INC IX ;nächste BCD-Ziffernmaske
3F13 DB5A 00787 IN A,(5ah) ;RTC Register auslesen
3F15 DDA600 00788 AND (IX+00h) ;relevante Bits maskieren
3F18 C9 00789 RET
3F19 5F 00790 wrrtc LD E,A ;Zeit/Datum retten
3F1A 7A 00791 LD A,D ;RTC-Adressreg., RD/WR-Bit
3F1B D35B 00792 OUT (5bh),A ;RTC-Register adressieren
3F1D 1610 00793 LD D,10h ;1 höher im oberen Nibble
3F1F 82 00794 ADD A,D
3F20 57 00795 LD D,A ;D <- neues Adressregister
3F21 E6F0 00796 AND 040h ;nur oberes Nibble
3F23 FE60 00797 CP 60h ;Std.-Zehner adressiert?
3F25 7B 00798 LD A,E ;Zeit/Datum
3F26 2002 00799 JR NZ,septrc ;falls anderes Digit
3F28 CBDF 00800 SET 3,A ;24h-Anzeige-Bit setzen
3F2A D35A 00801 septrc OUT (5ah),A ;Zeit stellen
3F2C C9 00802 RET
00803 ; Tabelle von Bitmasken für die Zeiteinheiten:
00804 clktab DB 0fh,0fh ;Jahr Zehner, Einer
00805 DB 01h,0fh ;Monat Zehner, Einer
00806 DB 03h,0fh ;Tag Zehner, Einer
00807 DB 03h,0fh ;Stunde Zehner, Einer
00808 DB 07h,0fh ;Minute Zehner, Einer
00809 DB 07h,0fh ;Sekunde Zehner, Einer

```

00000 Fehler



Das ist also interaktive Software!...

RPNL - Die andere Art zu programmieren

RPNL - was ist das ?

Bevor ich zum Thema komme sei gesagt, daß RPNL nicht auf meinem "Mist" gewachsen ist. Ich habe den Compiler nur an Newdos/80 angepaßt, bzw. nicht anpaßbare Teile (wegen CP/M) neugeschrieben (Disk-Befehle und die Ein-/Ausgabe etc.). Verbrochen wurde der Compiler von Herr Dipl.-Ing. G. Wostrack. Soviel zum Copy-Right und nun zur Sache.

RPNL bedeutet ausgeschrieben "Reverse Polish Notation Language". Als Eltern standen PASCAL und FORTH Pate, d.h. der Compiler-Befehlssatz ist an den von PASCAL angelehnt, die innere Struktur und die Art wie programmiert, wird stammt von FORTH. Herausgekommen ist ein Sprachkonzept, das soweit möglich nur die guten Seiten von beiden Eltern in sich vereinigt. Hierzu gehört zum Beispiel, daß die Programme strukturiert aufgebaut werden können. Es gibt Unterprogramme, Funktionen und das Hauptprogramm, welches alle Einzelteile miteinander verbindet. Für diejenigen, die ihre Probleme besonders schnell erledigt wissen wollen, können auch in Assembler geschriebene Routinen eingebunden werden. Dies erfolgt natürlich alles nach RPNL-Konvention.

Als Schleifen-Strukturen stehen REPEAT ... UNTIL ... LOOP und die FOR ... LOOP zur Verfügung. Mit ihnen können praktisch alle vorkommenden "Wiederholungs-Probleme" erledigt werden. Verzweigungen lassen sich einfach über IF ... ELSE ... ENDIF realisieren.

Wie FORTH, so kennt auch RPNL Dictionaries. Es gibt von ihnen insgesamt drei: Das Anwender-, das Compiler- und das Declarations-Wörterbuch. Der Programmierer hat jedoch nur auf jenes Zugriff, in das der Compiler die neu definierten Worte ablegt - das Anwender-Wörterbuch. Die verbleibenden beiden Wörterbücher sind nicht erweiterbar, sie können nur während der Compilierung vom Compiler aufgerufen werden.

Der hervorstechendste Unterschied zu FORTH ist jedoch die Tatsache, daß sich mit RPNL eigenständige Programme erzeugen lassen (mittels "SAVE name"), die direkt vom DOS-Level aus gestartet werden können. Es muß also nicht jedesmal das gesamte "System" geladen werden (wie z.B. zum Starten eines Basic-Programmes). Ermöglicht wird dies durch aufteilen des Systems in zwei Teile: dem Runtime-Package und dem Compiler. Das Runtime-Package beginnt am unteren Ende ab 5600H, der Compiler liegt am oberen Ende des Speichers ab F000H. Für eigene Treiber stehen z.Zt. ca. 500 Byte zur Verfügung (ab FE00H bis FFFFH). Das Anwenderwörterbuch wächst von F000H nach unten im Speicher.

Wird nun ein Programm compiliert, so wird der erzeugte Code an das Runtime-Package angehängt. Programmnamen etc. werden parallel in das Dictionary eingetragen, sodaß das System stets über alles informiert ist. Verdaulich sind für den

Compiler jedoch nur reine ASCII-Files ohne Zeilen-Nummern !. Es sollte also ein Texteditor, der ASCII-Files erzeugt, benutzt werden. Die max. Zeilenlänge ist auf 126 Zeichen begrenzt.

Wie wird programmiert ?

Die Programmerstellung erfolgt häppchenweise. Da in jeden Programm Variable, Konstante und dergleichen vorkommen, erfolgt die Definition dieser Dinge vor ihrem ersten Gebrauch. Hierfür wird die Anweisungsklammer DECLARE/DEND benutzt. Das sieht dann folgendermaßen aus:

```
DECLARE
  VAR TEMP    CONST TEST 100
  ARRAY BILD 220
  STRING FRAGE "Wie Bitte ?"
DEND
```

Das gezeigte Beispiel enthält alle Datentypen die RPNL kennt. Ich werde sie der Reihe nach abhandeln. Zuvor aber noch einige Worte zur Schreibart. RPNL benötigt KEINE Zeilennummern, auch ist es egal wie der Programmtext gestaltet ist. Das obige Beispiel könnte auch ebenso gut in einer Zeile geschrieben sein (sofern sie in den Input-Buffer paßt !). Ob eingerückt wird oder nicht ist gleichgültig, auf dererlei Dinge legt der Compiler keinen Wert. Die Schreibweise der Anweisungen interessiert ihn jedoch schon. Der bestehende Befehlssatz ist in Großbuchstaben verfaßt und muß auch so geschrieben werden, andernfalls werden die Anweisungen nicht erkannt. Neue Definitionen können dafür aber durchaus in gemischter Groß/Klein-Schreibung erfolgen, müssen dann aber in der einmal festgelegten Form benutzt werden !.

Nun zum Beispiel. Zuerst die Anweisung DECLARE selbst. Hier handelt es sich um eine Compiler-Anweisung, die das Flag "COMPILE" auf FALSE und somit das System in den "Compile-Modus" versetzt. Danach wird das Dictionary "DECLST" eröffnet. In diesem befinden sich alle gültigen Deklarations-Worte. Die Anweisung DEND nimmt eine Sonderstellung ein. Es ist in der Liste ebenfalls enthalten und hat die Aufgabe, das Compile-Flag wieder auf TRUE zu setzen und so den Compile-Modus zu beenden. Erkennt der Compiler nun beim Durchsuchen seiner Liste einen der Befehle, so wird die zugehörige Routine aufgerufen. Diese führt nun alle Aktivitäten durch, die notwendig sind, um die nachfolgenden Parameter zu verarbeiten.

VAR TEMP

Diese Anweisung bewirkt, daß im Dictionary ein Eintrag mit dem Namen TEMP erfolgt, gefolgt von einer Adresse. Diese Adresse dient bei späteren Bezugnamen auf diese Variable dazu, um an deren Inhalt heranzukommen. Die exakten Verhältnisse sind noch etwas komplizierter, da es sich hier um ein System von "Zeiger auf Zeiger auf ein Programm" handelt (Quasie von hinten rum durch die Küche ins Auge).

CONST TEST 100

Bei CONST liegen die Verhältnisse ähnlich, nur wird hier statt der Adresse auf den Speicherplatz der Variablen, der Wert der Konstanten auf den Stack gebracht. Während bei einer Variablen-Definition diese automatisch mit Null initialisiert wird, erhält die Konstante natürlich einen Wert. Dieser muß hinter dem Konstanten-Namen (hier TEST) angegeben werden.

ARRAY BILD 220

Vom Typ her ist die Anweisung ARRAY eine Variable. Lediglich bei Reservierung des Speicherplatzes wird anders vorgegangen. Es werden statt einem 220+1 16Bit Plätze vereinnahmt. Plus 1 deshalb, weil der Platz 0 mitzählt. Später im Programm wird bei Aufruf von BILD ein Zeiger auf den ersten (!) Speicherplatz des Arrays auf dem Stack übergeben. Dieser Umstand ist zu berücksichtigen, da es nämlich bedeutet, daß der Anwender seine eigenen Berechnungen für den jeweiligen Array-Platz durchführen muß. Man hätte auch anders verfahren können, dieser Weg bietet jedoch die größte Verwendungsvielfalt. Eine Überwachung der Grenzen (0 und 220) erfolgt nicht (!!!), hierauf hat der Programmierer selbst zu achten.

STRING FRAGE "Wie Bitte?"

Und als letztes die Anweisung STRING. Mit ihr kann dem Wort FRAGE eine Zeichenkette zugewiesen werden. Diese wird wie folgt vom Compiler im Speicher abgelegt:

- Ein Byte als Längenzähler (0...255)
- Die angegebene Zeichenkette (ohne CR oder andere Stringendekennungen !!!)

Der bei Aufruf übergebene Zeiger zeigt auf den Längenzähler. Natürlich erzeugt der Compiler auch hier wieder einen Eintrag im Dickey und hängt den erzeugten Programm-Code an das Runtime-Package an.

Das PROGRAM

Mit der DECLARATION allein ist noch nicht viel anzufangen. Interessant wird die Sache erst mit der Anweisung PROGRAM. Mit ihr kann "programmiert" werden. PROGRAM startet den Compiler über das bereits genannte Flag und eröffnet das Anwender-Wörterbuch. Die nachfolgende Zeichenkette wird als Name des Programms in das Dickey eingetragen. Alle nun folgenden Worte werden als Anweisungen aufgefaßt und compiliert. Ausgenommen hiervon sind Kontrollanweisungen wie IF...ELSE...ENDIF, FOR...LOOP etc. Sie veranlassen den Compiler zu besonderen Aktionen, die unter Umständen eine ganze Kette von Anweisungen dem Runtime-Package hinzufügen. Mit END wird der Compiler wieder verlassen (END ist in der Wirkung identisch mit DEND). Zum besseren Verständnis nun ein Beispiel:

PROGRAM BEISPIEL

```
;Gebe die Zahlen von 1 bis 100 aus
100 1 ;Laufbereich der Schleife
FOR ;Anfang der FOR-Schleife
I TEMP := ;Laufindex I nach TEMP
```

```
TEMP PRINT ;Inhalt von TEMP ausgeben
LOOP ;Ende der FOR-Schleife
END ;Programmende
```

Das Semikolon wird in RPNL dazu benutzt, Kommentare zu kennzeichnen, ein Blank zum nachfolgenden Text ist nicht erforderlich. Der Aktuelle Laufindex kann mit dem Wort 'I' auf den Stack geholt werden. Das man ihn von dort nicht unmittelbar ausgeben kann, ist beabsichtigt. Es soll mit der Zwischenspeicherung in einer Variablen eine bessere Lesbarkeit der Programme erreicht werden. Im gewählten Beispiel kommt diese Absicht nicht so deutlich zum Tragen, der Vorteil zeigt sich erst bei umfangreicheren Programmen. Dort ist es dann schon von Interesse sein, stets zu wissen, wo die Zahlen geblieben sind.

Wer bereits einmal in FORTH programmiert hat, wird wissen, wie schnell der Überblick über den Datenbestand auf dem Stack verloren gehen kann. Nach meinen bisherigen Erfahrungen bedeutet dieser "kleine Umweg" keinen Nachteil, denn das Hin- und Herschaufeln von Zahlen auf dem Stack beansprucht in der Regel die gleiche Zeit und ist unübersichtlicher.

Assembler wird mit CODE und CALL erst schön

Wer es besonders eilig hat, wird eine Lösung in Assembler vorziehen. Hierfür gibt es die Anweisungen CODE & CALL. Beide Befehle rufen wiederum den Compiler auf, unterliegen dabei aber einer gewissen Rangordnung. Es gilt:

```
Ebene 1: PROGRAM
Ebene 2: CODE
Ebene 3: CALL
```

Die Priorität verläuft vertikal, wobei eine Ebene nur jeweils die nächsttiefere oder eigene Ebene aufrufen kann. Das Überspringen einer Ebene ist nicht erlaubt und erzeugt nicht lauffähige Programme.

Da RPNL z.Zt. noch keinen Inline-Assembler enthält, muß der Maschinen-Code in Hexform eingegeben werden, was so aussieht:

```
CALL $BEISPIEL3
...
C9 ;das notwendige RET !
CEND

CODE BEISPIEL2
C1 D1 E1 C5 CD $BEISPIEL3
5F E1 ...
CEND
```

Wie zu erkennen, dürfen die Hex-Zahlen nur zweistellig sein. Abweichungen hiervon werden mit einer Fehlermeldung geahndet.

Die Länge des Assemblerteils ist nicht festgelegt. Das liegt im Ermessen des Programmierers. Eine Besonderheit stellt der Assemblerbefehl CD \$BEISPIEL3 dar. Hier wird ein Unterprogramm Namens \$BEISPIEL3 aufgerufen. CODE sucht nun im Anwender-Dicentry nach diesem Namen und trägt dessen zugehörige Code-Adresse im erzeugten Programm ein. Das Dollarzeichen vor dem Namen ist nicht zwingend, ich benutzte es nur zum Kennzeichnen von Unterprogrammen, die mit CALL...CEND erzeugt wurden. Diese Vorgehensweise ist so unmittelbar nur bei Unterprogrammen gangbar. Soll der Inhalt einer Variablen direkt per Maschinenbefehl in eines der Register geladen werden, so ist noch etwas zusätzlich zu tun. In diesem Fall muß die geladene Adresse um zwei erhöht werden. Dann erst kann unter Benutzung der "neuen" Adresse der Variableninhalt in ein Register geholt werden. Oder anders:

```
Reg. <- Adr.
Reg. = Reg. + 2
Reg. <- (Reg.)
```

Das ist zugegeben sehr umständlich, aber in diesem Fall schlägt die innere Struktur von RPML voll auf den Programmierer zurück. RPML ist eben nicht FORTRAN oder BASIC. Dafür hat man dann aber auch die Möglichkeit den ganzen Abfrage-Overhead dieser Sprachen nicht mitschleppen zu müssen.

Die Anweisung CEND schließlich schaltet den Compiler wieder aus. Die Anweisungsklammer CALL...CEND arbeitet nach dem gleichen Schema. Es muß aber unbedingt am Ende des Assemblerteils ein RET (= C9H) angefügt werden, sonst rennt der Z80 ins Leere (weil ja ein Unterprogramm!). Zu beachten ist hier ganz besonders, daß der Datenstack über den Stackpointer (SP) der CPU gebildet wird und somit die Rücksprung-Adresse zuoberst auf dem Stack liegt.

COMPILE name

Ist das Programm fertig, so kann es jetzt mit COMPILE übersetzt werden. Der Syntax für "name" ist wie gewohnt. Lediglich ein Passwort darf nicht benutzt werden, dieses wird z.Zt. nicht akzeptiert. Das Programm wird an das Runtime-Modul ancompiliert und kann bei Bedarf mit SAVE auf Diskette gesichert werden.

Treten bei der Compilierung Fehler auf, so wird abgebrochen und eine Fehlermeldung auf dem Bildschirm ausgegeben. Das fehlerhafte Programm wird in diesem Fall gelöscht und das System reinitialisiert. Eine Extension oder Laufwerk müssen nicht angegeben werden, hier werden vom Compiler optional "/RPN" und vom DOS gemäß der Eintragung unter SYSTEM das Laufwerk eingetragen.

SAVE name

Ist das Programm endlich compiliert, so möchte man es auch noch auf Diskette abspeichern. Hierfür ist die Anweisung SAVE mit nachfolgendem Programmnamen zuständig. Für den Namen gelten die Newdos-Konventionen mit Ausnahme der Passwörter. Diese können

z.Zt. nicht benutzt werden. Aber das läßt sich wohl verschmerzen. Die Extension und das Laufwerk müssen nicht unbedingt angegeben werden, diese werden automatisch angehängt (/Ext = CMD, das Laufwerk erledigt Newdos gemäß SYSTEM). Damit stände es nun auf der Diskette und kann von DOS-Ready aus aufgerufen werden.

Es ließe sich noch einiges mehr zu RPML sagen, aber der Redaktionsschluß und mein Urlaubstermin nahen. Als Praxisteil nun noch zwei Programme in RPML, in denen alles zusammengefaßt ist. Im ersten Programm werden darüberhinaus auch Diskbefehle verwendet. Wer es noch ausführlicher beschrieben lesen will, der besorge sich das Buch.

/1/ Vostrack, Gustav
RPML : eine FORTH-ähnliche Sprache mit struktur-
unterstützten Sprachkonstrukten / Gustav
Vostrack. - Sprendlingen : Luther, 1984.
ISBN 3-88707-022-4

Tschüß

Rud Müller

Das Programm PRINTF gibt einen ASCII-File inklusive der Steuercodes auf dem Monitor oder Drucker aus.

```
DECLARE
  VARF FILE          VAR TEMP
  VAR TEMP1          VAR TEMP2
  STRING (D) "D"     STRING (Y) "Y"

CEND

PROGRAM PRINTF
  SYSFIELD ?
  CLS WRITE "*****"
  CR WRITE "** File-Print Utility **"
  CR WRITE "*****"
  CR 14 OUTCHAR
  WRITE "Drucker (D) oder CRT (C) ? : "
  TEMP1 READ(CH)
  WRITE "Filename ? : "
  TEMP READ(CH)
  TEMP1 ? (D) =(S)
  IF
    WRITE "Form Feed (Y/N) ? : "
    TEMP2 READ(CH)
    TEMP2 ? (Y) =(S)
    SCROFF DRON
  IF
    PAGE
  ENDIF
ENDIF
```

NEFT
15
September
1984

62


```

TEMP ? ?(B)
0 > IF
    INCHAR
    TEMP ? FILE FILESPEC DROF
    FILE OPEN(1)
    REPEAT
    UNTIL
        FILE READF
        0= IF
            127 AND OUTCHAR
            INCHAR ZEICHEN ?(S)
            0= IF
                FALSE
            ELSE
                TRUE
            ENDIF
        ELSE
            TRUE
        ENDIF
    LOOP
    FILE CLOSE
    ENDIF
    DROFF SCRON
    SYSFIELD :=
END

```

Das Programm LINE zeichnet eine Linie in der HRG

```

CODE +2DY 21 DY 23 23 5E 23 56 EB 29 EF E1 19 E5 CEND
CODE +2DX 21 DX 23 23 5E 23 56 EB 29 EB E1 19 E5 CEND
CODE -2DY 21 DY 23 23 5E 23 56 EB 29 EB E1 A7 ED 52 E5 CEND
CODE -2DX 21 DX 23 23 5E 23 56 EB 29 EB E1 A7 ED 52 E5 CEND

```

```

PROGRAM LINE      ;(X1 Y1 X2 Y2 FLAG -> )
;Rette           einige           Parameter
MODE :=
Y2 :=      X2 :=
Y1 :=      X1 :=
;Errechne DX und bestimme aus dessen Vorzeichen, ob
; INCX = +1 oder -1 sein muß, anschließend wird der
; Absolutwert von DX gebildet
X2 ? X1 ? - 32768 + DUP
32768 >=
IF
    32768 - DX :=
    1 INCX :=
ELSE
    32768 SWAP - DX :=
    TRUE INCX :=
ENDIF
;Hier wird das Gleiche mit DY und INCY durchgeführt
Y2 ? Y1 ? - 32768 + DUP
32768 >=
IF
    32768 - DY :=
    1 INCY :=
ELSE
    32768 SWAP - DY :=

```

```

TRUE INCY :=
ENDIF
DX ? DY ? >      ;Bestimme die Steigung m
IF
    ;Schleife für Steigung < 45 Grad
    DY ? DUP + DX ? - 32768 + EE :=
    DX ? 0
    FOR
        X1 ? Y1 ? MODE ? DOT
        EE ? 32768 >=
        IF
            Y1 ? INCY ? + Y1 :=
            EE ? +2DY -2DX EE :=
        ELSE
            EE ? +2DY EE :=
        ENDIF
        X1 ? INCX ? + X1 :=
    LOOP
ELSE
    ;Schleife für Steigung >= 45 Grad
    DX ? DUP + DY ? - 32768 + EE :=
    DY ? 0
    FOR
        X1 ? Y1 ? MODE ? DOT
        EE ? 32768 >=
        IF
            X1 ? INCX ? + X1 :=
            EE ? +2DX -2DY EE :=
        ELSE
            EE ? +2DX EE :=
        ENDIF
        Y1 ? INCY ? + Y1 :=
    LOOP
ENDIF
END

```



Nun verstehen wir uns! Endlich habe ich den Durchbruch
bei dem verdammt Ding geschafft.

ECB-Adapterplatine

Ein erster Entwurf für eine ECB-Adapterplatine ist nun fertig.

Da die in Heft 9 vorgestellte Platine nicht funktionsfähig ist, muss die Adapterplatine vorläufig noch in Fädertechnik aufgebaut werden.

Schaltungsbeschreibung:

Alle Signale gelangen über ein Flachbandkabel vom TRS/VB/Komtek zu einem 2 X 25 - poligen Pfostenstecker auf der Interfacekarte. Dieser Pfostenstecker wird auf der Lötseite je nach Gerätetyp individuell mit den Eingängen (terminier-Widerständen) verdrahtet. Dies ist notwendig wegen

der unterschiedlichen Beschaltung der Geräte und damit, das Verbindungskabel ein einfaches 1 zu 1 Kabel sein kann.

(Anpresstecker)

Im Schaltplan sind an den meisten Eingängen 2 Kreise eingetragen. Diese deuten darauf hin, dass die Leitung, so wie ganz links oben gezeichnet, zu terminieren ist. Die 4 zum Computer führenden Leitungen, sowie der Takt sind ohne Terminierung.

Der Control-Bus und der Adressbus sind mit 74LS244 gepuffert, die fix das CS-Signal auf 0 haben. (immer aktiv) Der Datenbus-Puffer ist ein 74LS245, der seine Richtungssteuerung von MRD oder IN erhält. Sein CS erfolgt entweder durch IN/DOUT oder ein externes CS vom ECI-Bus auf B10.

(Wenn eine Karte IM Bus-System selbst seine Adresse decodiert, so muss das dabei entstandene CS-Signal auch auf Pin B10 gelegt werden, (open collector) damit der Datenbus-Puffer angesteuert wird. Das ist nur bei Memory-mapped mode notwendig)

Wenn man also für eine bestimmte Karte den Interrupt mode 2 verwenden will, so muss man diese Art der Ansteuerung wählen, da ja der Interrupt-Vektor auf den Bus gelegt werden muss.

Die 4 zum Computer führenden Control-lines sind mit einem 7407 gepuffert, da diese Leitungen voneinander unabhängig sein müssen.

Adressdecodierung:

Ziel war es so viele I/O Portadressen als möglich zu decodieren. Aus Platzgründen wurden nicht mehr als 3 Decoderbausteine 74LS138 samt den notwendigen Verknüpfungsbausteinen untergebracht.

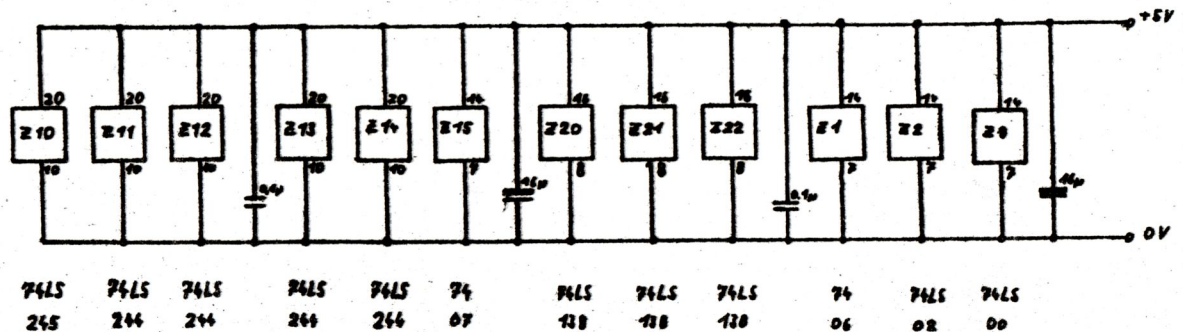
Z20 decodiert in Vierschritten von Port 0 an bis 1C. z.B.: 0, 4, 8, C, 10, 14, 18, 1C. Bei den dazwischenliegenden Adressen erfolgt natürlich ebenfalls ein CS-Signal. z.B.: liefert der Pin 15 von Z20 ein CS-Signal für jede I/O Adresse von 00H bis 03H.

Das IC 221 liefert eine Dreifachadressierung. Der Baustein bringt also ein CS-Signal auf Pin 15 für eine Adresse von 20H oder 40H oder 60H. Das hat den Vorteil, dass die Chance, irgendeine fertige Karte mit passender Adresse zu bekommen sehr hoch ist.

222 wiederum ist nach demselben Schema wie Z20 eingesetzt. Nur ist diesmal das MSB auf on, das heißt: die Adressen

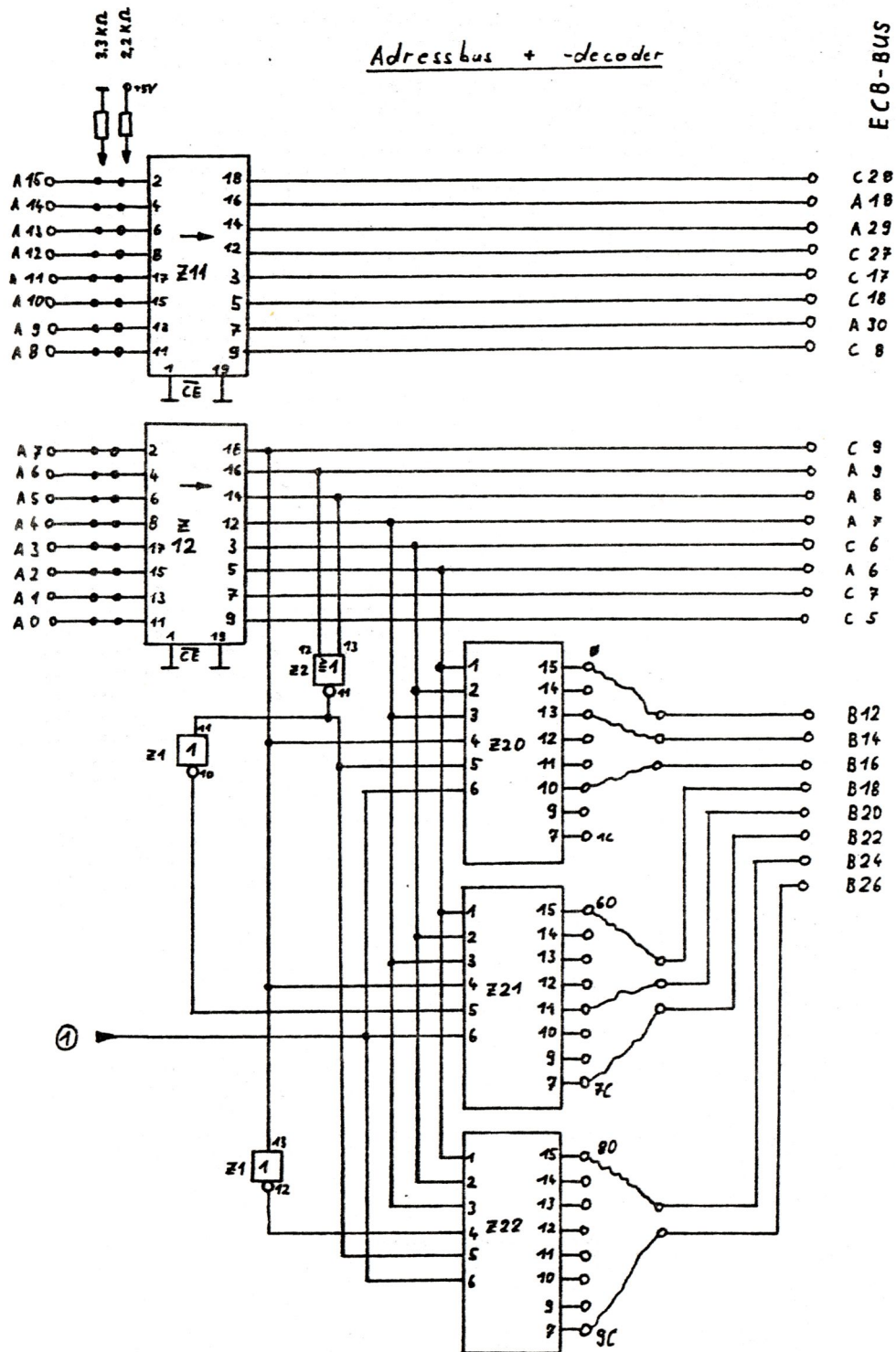
von 80H bis 9CH werden decodiert.

Alle Ausgänge der Decoderbausteine liegen auf Lötstiften. In ca. 1cm Abstand stehen die 8 Lötstifte, die zu den CS-Leitungen im ECI-System führen. So kann man mit kurzen Lötbrücken die gewünschten Adressen bestimmen und braucht auf einer ECB-Karte keine Adressdecodierung mehr vornehmen.



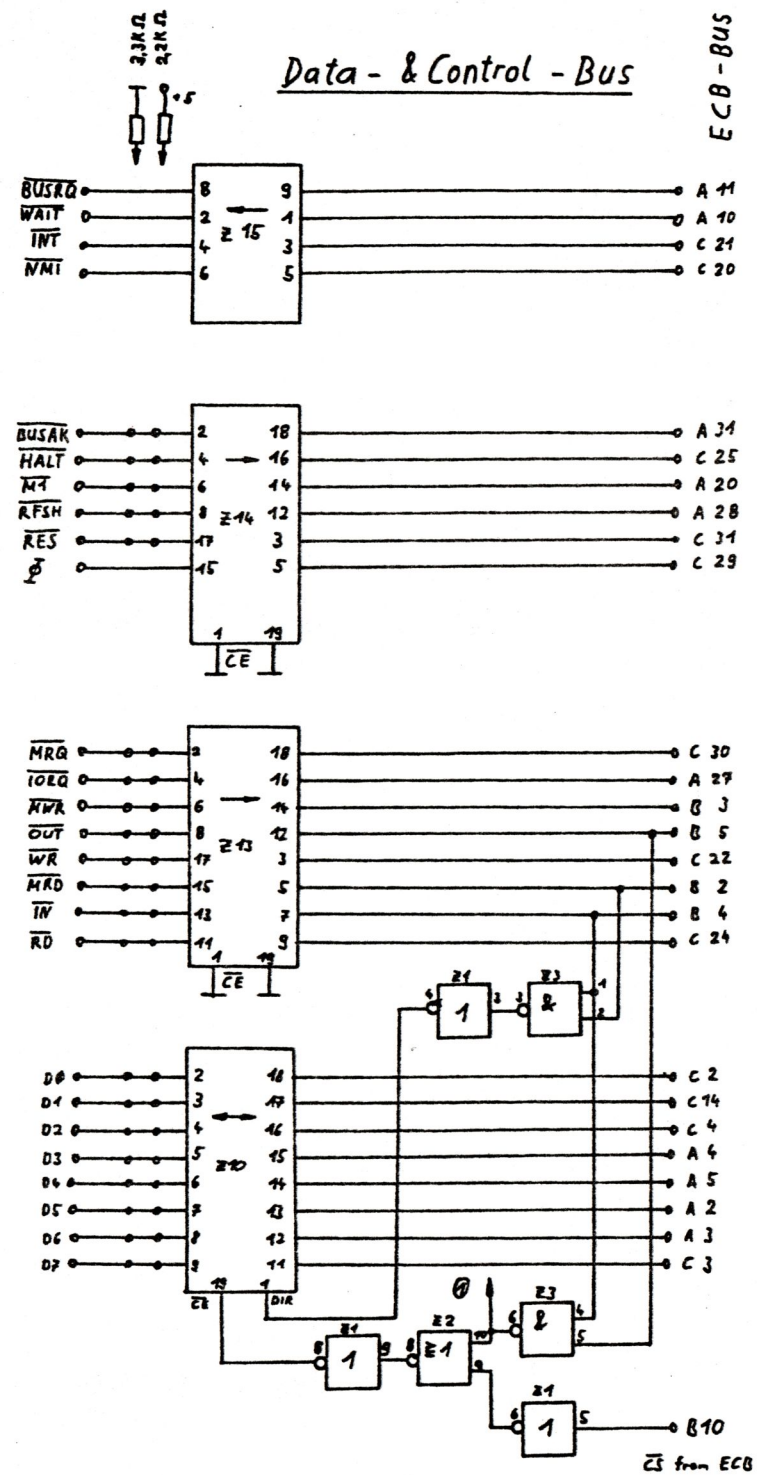
Adressbus + -decoder

ECB-BUS



Data - & Control - Bus

ECB-BUS



Verschiedene Anschlussleisten:

Leitung	TRS-80	ECB-Leiste	Kontext
A11	9	17C	6
A12	5	27C	11
A13	6	29A	14
A14	10	18A	13
A15	7	26C	16
<u>0</u>	16	29C	45
D4	18	5A	42
D3	26	4A	39
D5	28	2A	37
D6	24	3A	44
+5V		1ABC	
D2	32	4C	40
D7	20	3C	43
D0	30	2C	46
D1	22	14C	41
INT	21	21C	10
NMI	<u>17</u>	20C	5
HALT	<u>46</u>	25C	9
MEM	1	30C	47
IORQ	<u>20</u>	27A	35
RD	<u>27</u>	24C	38
WR	<u>22</u>	22C	36
BUSAK	<u>23</u>	31A	12
WAIT	33	10A	8
EUSRG	23	11A	7
RESET	2	31C	33
M1	<u>27</u>	20A	4
RFSH	3	28A	3
GND	B 29 37	32ABC	
A0	25	7C	34
A1	27	7C	32
A2	40	6A	30
A3	34	6C	28
A4	31	7A	26
A5	35	8A	24
A6	38	9A	22
A7	36	9C	20
A8	11	8C	19
A9	17	30A	21
A10	4	18C	27
MRD	13	2E	25
MWR	15	3E	46
IN	19	4E	
OUT	12	5E	

ECB-Bus Belegung

A	B	C
1	+5V	+5V
2	D5	D0
3	D6	D7
4	D3	D2
5	D4	A0
6	A2	A3
7	A4	A1
8	A5	A6
9	A6	A7
10	WAIT	
11	BUSRG	
12	A18	
13	+12V	
14		
15	-5V	
16	<u>1</u>	
17	A17	
18	A14	
19	+15V	
20	M1	
21		
22		
23		
24		
25		
26		
27	IORQ	
28	RFSH	
29	A13	
30	A9	
31	BUSAK	
32	GND	

CS für ECB

CS1

CS2

CS3

CS4

CS5

CS6

CS7

CS8

GND

Die ECB Norm verwendet nur die Stiftreihe A+C.
Die E-Reihe dient am Interface für CS-Adressdecodierung
26C und 31C sind auf der Platine verbunden
IEI: IEI = Daisy-Chain

Die eingekreisten Zahlen beim TRS-80 Interface sind keine Bus-Pins sondern diese Leitungen müssen direkt von der CPU aus verlitet werden.

HEFT
15
September
1986

70

Mit diesem Artikel möchte ich Euch für den Aufbau der Grundeinheit einige Tips geben. Diejenigen, die sie bestellt haben, kommen ja nun demnächst in den "Genuß" sie auch zusammenbauen zu dürfen.

An erster Stelle steht der mechanische Zusammenbau unseres Gehäuses. Einen groben Überblick dazu gibt ja schon die dem Gehäuse beiliegende Anleitung, aber ein paar Anregungen zur Erleichterung der Arbeit sind schon noch nötig.

Zu beachten ist die richtige Anordnung der sechs Schienen. Die vorderen beiden Schienen sind diejenigen, welche den Deckelfalz und die Löcher für die Platinen-Führungsschienen haben. Diese werden im ersten Seitenteilbefestigungspunkt, zusammen mit der Winkelschiene - an der vorher der Griff mit den selbstschneidenden Schrauben befestigt wurde-, verschraubt. Im dritten Befestigungspunkt des Seitenteiles werden dann die gelochten Schienen ohne Deckelfalz angebracht. Das zweite Loch von vorne bleibt frei. Die Lochbleche für den Einbau der Busplatine werden an der Schiene (auf Lochplatz drei des Seitenteiles) so befestigt, daß bis zur Vorderkante des Gehäuses der Abstand von 17,5 cm vorhanden ist. Später ist dann die Busplatine mit ihren Buchsenleisten von der Gehäusevorderseite einzulegen und daran anzuschrauben. Zur Entlastung der Platine werden alle zehn Buchsenleisten angeschraubt. Vorerst werden aber die restlichen zwei Winkelschienen über Loch vier des Seitenteiles durch Verschrauben, der Schienen mit Deckelfalz aber ohne Lochung, befestigt. Günstig ist es, in jede der "END-Schienen" drei "Einschiebemuttern" einzubringen. Sie dienen dann dem Anschrauben der Rückwand. Gleichzeitig sollten auch das Boden- und Deckelblech eingeschoben werden. Alles Weitere für den Gehäuse-Zusammenbau dürfte dann klar sein.

Zum Einlöten der zehn 96-poligen Busstecker auf die Busplatine ist zu beachten, daß die Buchsenleisten mit ihren Bezeichnungen übereinstimmend mit den Busbezeichnungen angebracht werden. Das heist, wenn die Stromversorgungsanschlüsse der Busplatine nach rechts zeigen und die Lötseite (beschriftete Platinenseite) nach unten zeigt, ist der Punkt 1A auf der Busplatine und auch auf der Buchsenleiste oben links zu finden. Dies gilt für jeden der zehn Steckplätze. Anhand der mitgelieferten Stecker kann dann überprüft werden, ob die Leitungen richtig durchverbunden sind. Besondere Beachtung ist der mittleren Reihe zu schenken, denn sie ist von der Stromversorgungsseite aus gesehen nach dem vierten Busplatz unterbrochen. Auf dieser Reihe haben wir unsere speziellen Signale, die wir von Fall zu Fall hier durchschalten bzw. auch auftrennen können. Weiterhin ist bei der linken Seite der jeweiligen Buchsenleiste die Verkettung von 23/25 und auf der rechten von 11/16 zu beachten. Hier wird jeweils ein Platztausch der Leitungen durchgeführt. Ist die Platine nun auf Durchgang und Isolation geprüft kann sie, wie links beschrieben, eingebaut werden. Danach ist das Netzteil anzuschließen. Die genaue Platzangabe für den Einbau kann noch nicht erfolgen, da ich die entsprechenden Netzteile noch nicht in Besitz habe und deshalb einen günstigen Einbauplatz noch nicht empfehlen kann. Dies wird aber bis zum entgeltlichen Zusammenbau noch nachgeholt.

Auf jeden Fall sollte der Anschluß der 220V nach VDE erfolgen. Die Mindestforderungen hier sind:

Berührungssicherheit der Spannungsführenden Leiter (Phase und Rückleiter)

Angeschlossenem Schutzleiter am Gehäuse.

Empfehlenswert ist auch die Verwendung eines zweipoligen Netz-Schalters mit beleuchtbarer Wippe zum Ein-/Ausschalten. Als weitere "Gestaltungsmöglichkeiten" wären Netzfilter zur Störabblockung und Steckdosen für externe Geräte denkbar. Dem jeweiligen Ausbau sind auch hier keinerlei Beschränkungen auferlegt.

Nun aber zur anderen Netzteilseite, der Niederspannung. Die Anschlußdrähte werden entsprechend ihrer Bezeichnung (zur Kontrolle vorher die Spannungen einmal messen) auf den gleichnamigen Punkt der Busplatine eingelötet. Danach sollten die entsprechenden Spannungen auf jeder Busbuchsenleiste greifbar sein. Bitte auch das überprüfen.

Das waren nun einige Tips für den Grundeinheitsaufbau. Sollten sich doch noch Probleme ergeben, bin ich gerne bereit sie mit Euch zu lösen.

Ist die Grundeinheit dann betriebsfertig geht es mit den hier nachfolgend vorgestellten Karten weiter.

Viel Erfolg beim Aufbau

Jens Neueder

Die Eingangsplatine hat für eine Anpassung der Erweiterungen an den TRS-80 zu sorgen. Eine galvanische Trennung ist nicht erforderlich. Aus diesem Grunde kann auf Impulsübertrager oder optoelektronische Koppellemente verzichtet werden. Die auf der Mutterplatine zur Verfügung stehenden Signale müssen mehrere TTL-Eingänge treiben können.

Diese Forderungen werden durch den TTL-Baustein 74LS243 erfüllt. In diesem Baustein sind 8 Leistungstreiber, von denen je 2 antiparallel geschaltet sind, integriert.

Um die 16 Adressleitungen, die 8 Datenleitungen, die Steuerleitungen RD*, WR*, IN*, OUT*, MUX, CAS* und RAS* in beiden Richtungen treiben zu können sind 8 dieser Bausteine nötig. Die ICs werden so beschaltet, daß die Leitungen jeweils immer in einer Richtung getrieben werden (GBA und GAB* miteinander verbunden).

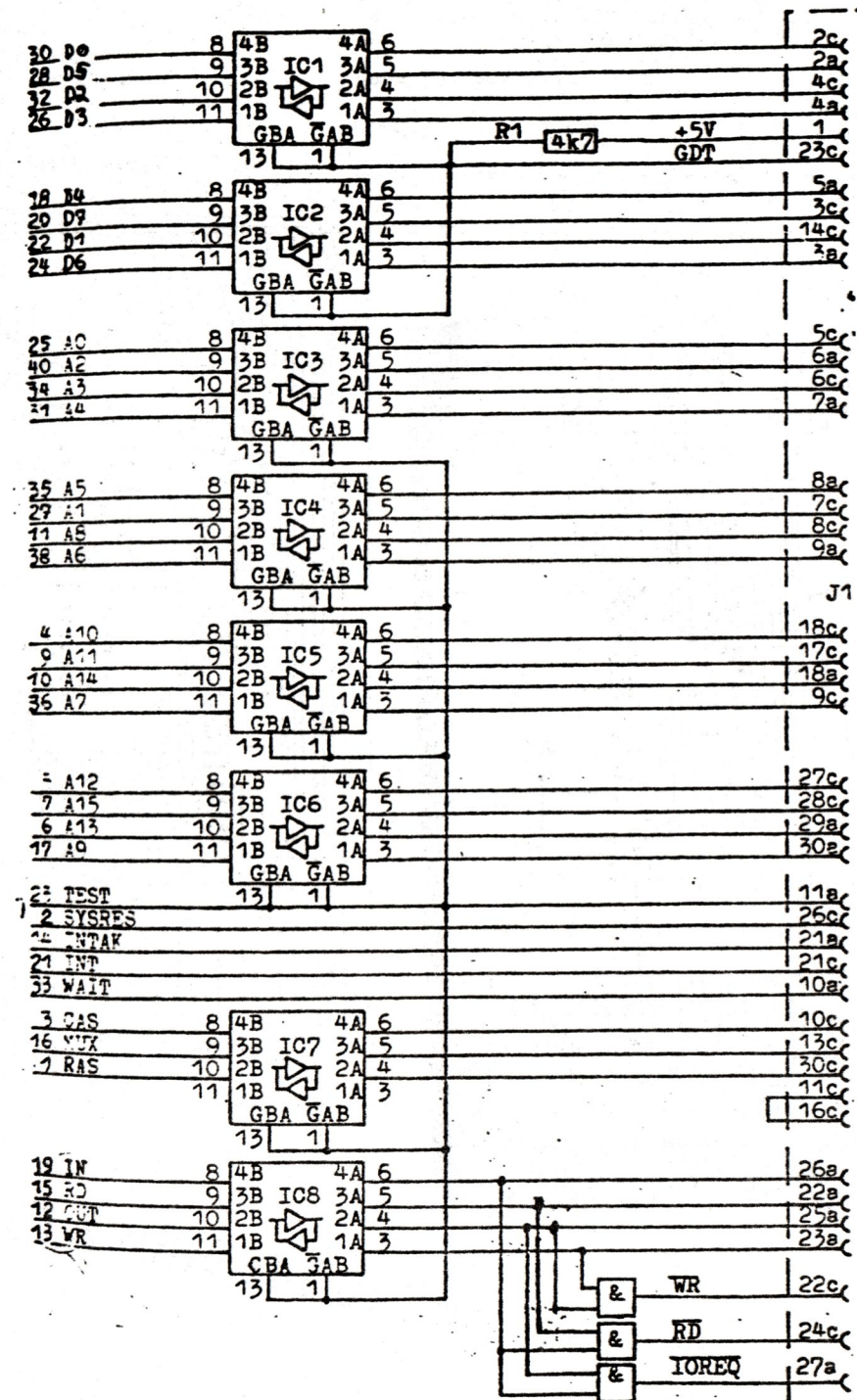
Die Steuerungen für die Richtungsumschaltung der Adress- und Steuerleitungen übernimmt das Signal TEST. Das TEST* Signal wird innerhalb des TRS-80 über einen Pull-Up-Widerstand auf 5 Volt gelegt, wodurch die Treiber vom TRS-80 aus zur Mutterplatine durchgeschaltet werden und die CPU Z-80 im aktiven Zustand bleibt. Wird das TEST* Signal auf 0 Volt gelegt, werden die Adress- und Steuerleitungstreiber in Richtung TRS-80 geschaltet. Die CPU Z-80 schaltet ihre Adress-, Daten- und Steuerleitungen auf hochohmig. Dieses ist z.B. bei einem DMA (Direct Memory Access) erforderlich.

Bei der Datenrichtungsumschaltung wäre bei Berücksichtigung aller Möglichkeiten eine aufwendige Decodierung der Steuer- und Adressleitungen erforderlich. Außerdem wäre bei einem DMA zu beachten, daß die Datenrichtungsumschaltung genau invers zum normalen Betrieb geschieht.

Eine Lösung, die dieses Problem ohne großen Aufwand bewältigt, ist die Einführung der GDT (Gate Data Transfer). Diese Leitung, die jeder Erweiterung über die Mutterplatine zugänglich ist, wird mit den Anschlüssen GBA und GAB der Datentreiber IC1 und IC2 verbunden und über einen Pull-Up-Widerstand auf H-Signal gelegt. Die Daten werden so vom TRS-80 aus in Richtung Mutterplatine getrieben. Soll die Richtung umgeschaltet werden, so muß die Leitung GDT auf L-Signal gelegt werden. Dies geschieht auf den Erweiterungsplatten durch die entsprechenden Decodierungen, die mit einem Open-Kollektor-Baustein oder mit einer einfachen Diode (siehe ZK01 diese Info-Ausgabe) abgeschlossen werden. Durch das Aufschalten dieser Open-Kollektor und Tri-State-Ausgänge auf die Leitung GDT erhalten wir ein Wired AND, d.h., daß sobald einer dieser Ausgänge auf L-Signal liegt, die Leitung GDT ebenfalls L-Signal führt. Diese Maßnahmen müssen bei der Auslegung von Erweiterungen berücksichtigt werden, da sonst Fehlfunktionen unvermeidbar sind! Käufliche Karten, können zu 90% mit einer einfachen Diode umgestellt werden.

Die Leitungen SYSRES*, INTAK*, INT* und Wait werden ohne Treiber auf die Eingangsplatine durchgeschleift, da sie nicht oft Verwendung finden und zum Betreiben von 2 TTL-Lasten ausreichend Leistung liefern.

Die Signale WR*, RD* und IORQ* der CPU Z-80 stehen am Ausgang des TRS-80 nicht direkt, sondern nur als deren Oder-Verknüpfungen IN*, OUT*, RD* und WR* zur Verfügung. Da eine Kompatibilität mit dem ELZET-80-BUS erwünscht ist und dort mit den Originalsignalen gearbeitet wird, werden sie über Und-Verknüpfungen zurückgewonnen. Das Signal RAS* ist identisch mit dem CPU-Signal MREQ*.

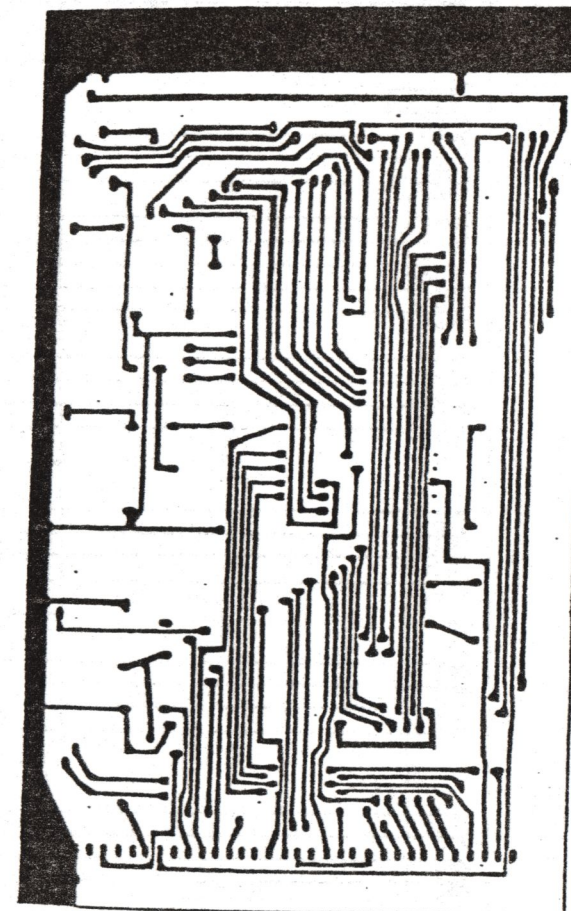
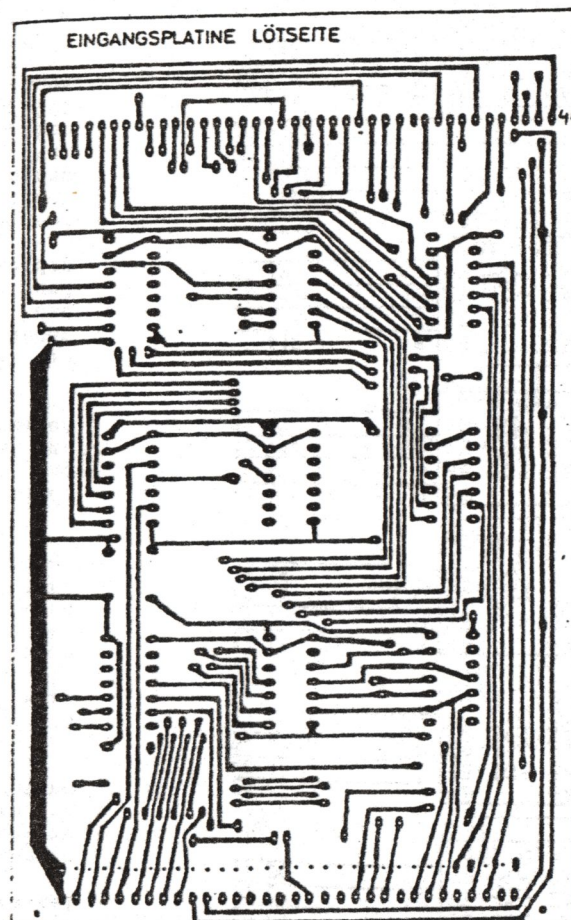
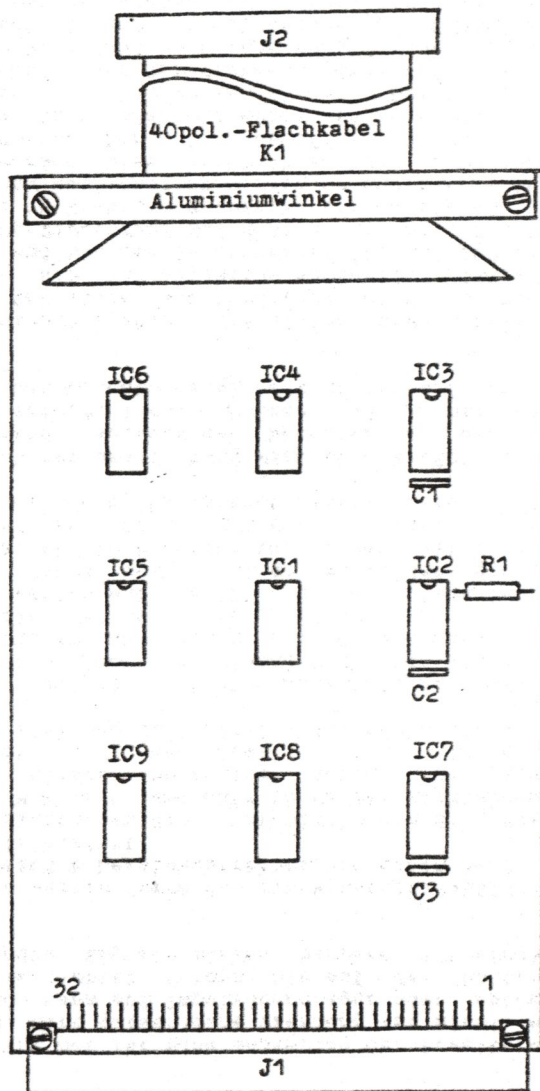


Platinenlayout Eingangskarte

Lötseite

Bestückungsplan Eingangskarte

ECB-TRS-80: EINGANGSPLATINE

Stückliste Eingangskarte

Halbleiter

IC 1-8 74LS243
IC 9 74LS08

Widerstände

R1 4,7k

Kondensatoren

C1-C3 100nF

Sonstige Bauteile

J1 64 pol. Steckerleiste
J2 40 pol. Steckverbinder (TRS-80)
K1 40 cm Flachbandkabel 40-adrig

ECB-Bus - Die erste Zusatzplatine !!!! (ZK01/02) Bernd Drowälder

Der folgende Artikel, beschreibt eine universelle 8-Bit-Parallelschnittstelle, für den ECB-Bus. Die Karte bietet folgende Möglichkeiten.

1. 8-Bit Parallelausgang. (Wert wird gespeichert!)
2. 8-Bit Paralleleingang. (zusätzlich benutzbar!)
3. Einstellbar auf jede der 256 möglichen I/O Adressen!

Wer will, kann 256 Karten gleichzeitig betreiben. Durch die freie Wahl der Adresse, dürfte wohl jeder, noch ein "freies Plätzchen", in seinem I-O Bereich finden. Die Auswahl erfolgt über 8-Diltschalter im Binärsystem.

Tabelle zur Adresseinstellung.

S1=2⁰=1 Stelle.
S2=2¹=2 Stelle.
S3=2²=4 Stelle.
S4=2³=8 Stelle.
S5=2⁴=16 Stelle.
S6=2⁵=32 Stelle.
S7=2⁶=64 Stelle.
S8=2⁷=128 Stelle.

Es folgen einige Beispiele.

gewünschte Adresse:		Diltschalter:							
Dezimal:	Hex:	S1	S2	S3	S4	S5	S6	S7	S8
255	FF	Ein	Ein	Ein	Ein	Ein	Ein	Ein	Ein
254	FE	Aus	Ein	Ein	Ein	Ein	Ein	Ein	Ein
253	FD	Ein	Aus	Ein	Ein	Ein	Ein	Ein	Ein
252	FC	Aus	Aus	Ein	Ein	Ein	Ein	Ein	Ein
...
003	03	Ein	Ein	Aus	Aus	Aus	Aus	Aus	Aus
002	02	Aus	Ein	Aus	Aus	Aus	Aus	Aus	Aus
001	01	Ein	Aus	Aus	Aus	Aus	Aus	Aus	Aus
000	00	Aus	Aus	Aus	Aus	Aus	Aus	Aus	Aus

Mit Hilfe der Tabelle, müßte eigentlich jeder seine gewünschte Adresse einstellen können. Es folgen noch ein paar Tips zum Testen der Karte. Dazu ist es jedoch erforderlich, acht Leuchtdioden, gemäß Bild 1.1 an die Karte anzuschließen.

Testprogramm A in Basic. (Ausgeben)

```
10REM EINGESTELLTE ADRESSE 0
20REM ALLE DILTSCHALTER AUS !
30INPUT "WELCHER WERT SOLL AUSGEGEBEN WERDEN";W
40OUT0,W
50goto30
```

Testprogramm B in Basic. (Ausgeben)

```
10REM EINGESTELLTE ADRESSE 0
20REM ALLE DILTSCHALTER AUS !
30REM LAUFLICHT
40FOR I=0 TO 7
50OUT0,2^I
60GOSUB 100:REM GEWÜNSCHTE VERZÖGERUNG
70NEXT I
80GOTO 40
90REM VERZÖGERUNG
100FOR V=1 TO 200
110NEXT V
120RETURN
```

Testprogramm C in Basic. (Einlesen)

```
10Rem EINGESTELLTE ADRESSE 0
20REM ALLE DILTSCHALTER AUS !
30REM DIE EINZELNEN DATENEINGÄNGE MÜSSEN AUF MASSE GELEGT WERDEN
40CLS
50W=INP(0):PRINT W:GOTO 50
```

Wird kein Eingang auf Masse gelegt, muß der angezeigte Wert 255 betragen !

Wird D0 auf Masse gelegt, dann 254. (D1=253, D2=251, D3=247 u.s.w.)

Weitere nützliche Beispiele folgen in den nächsten Infos.

HEFT
15
September
1986

78

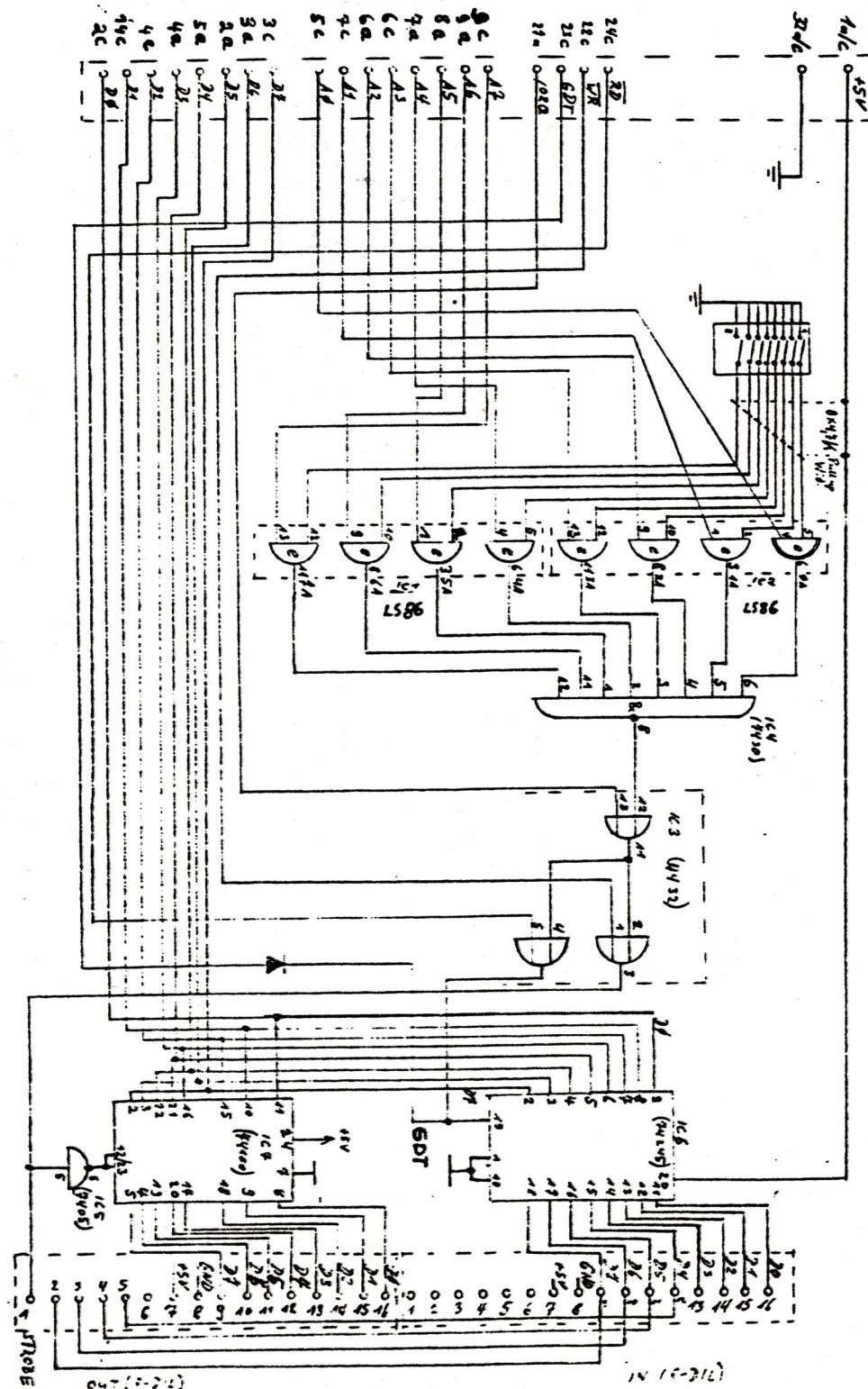
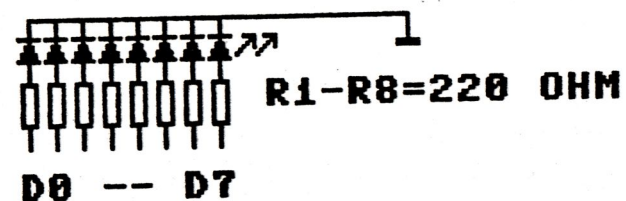


BILD 1.1



stückliste ZK01

Halbleiter

IC 1,2	74LS86
IC 3	74LS32
IC 4	74LS30
IC 5	74LS04
IC 6	74LS245
IC 7	74LS100

Widerstände

R1-R8 4,7k

Kondensatoren

C1-C2 100nF

Sonstige Bauteile

J1	64 pol. Steckerleiste
D1	1N4148
D1schalter	8*ON-OFF

P.S. Am besten alle Ic,s auf Fassungen !!!

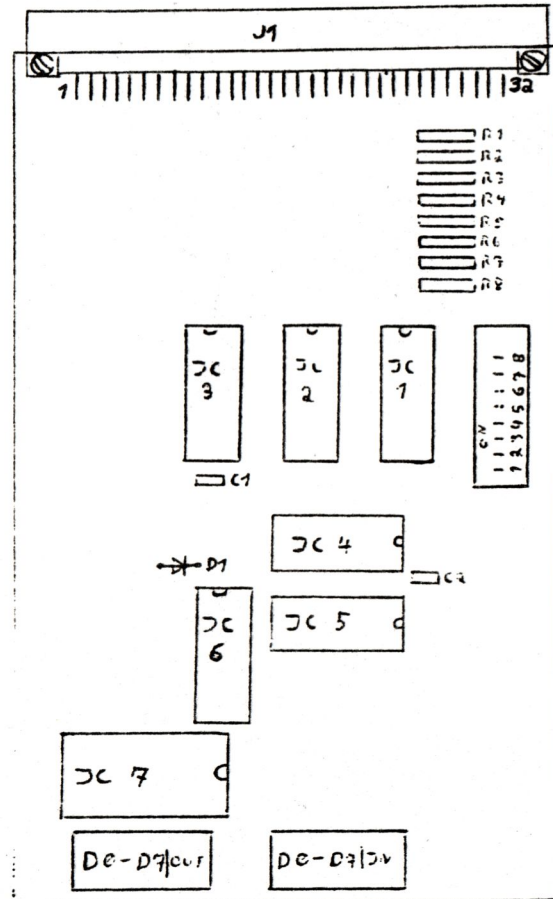
Um den Hartmut bei seinem Hardware-Grundlagen-Kurs zu unterstützen, folgt in der nächsten Info eine genaue Beschreibung der Hardware.

Viel Spaß beim Aufbau !!! Bernd Drowälder

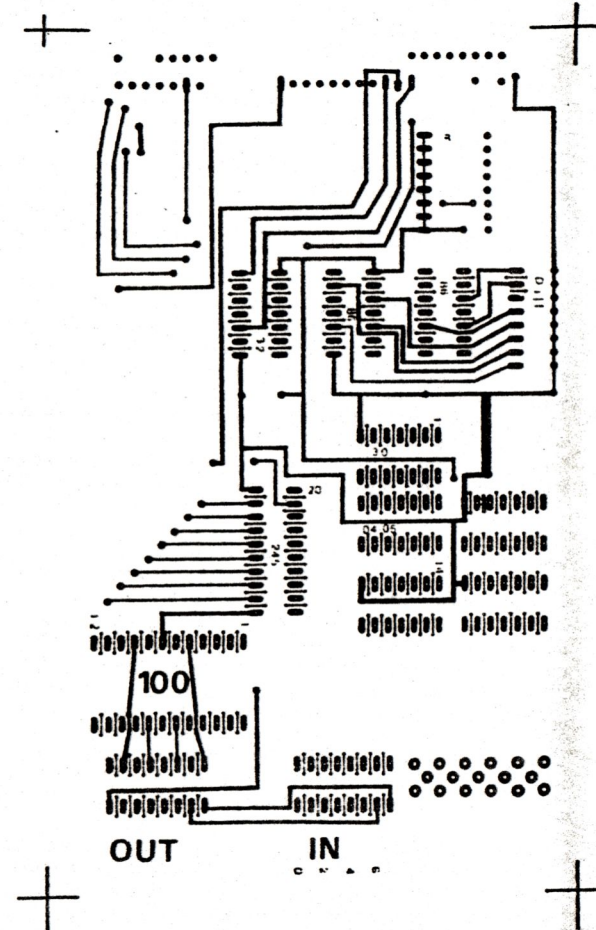
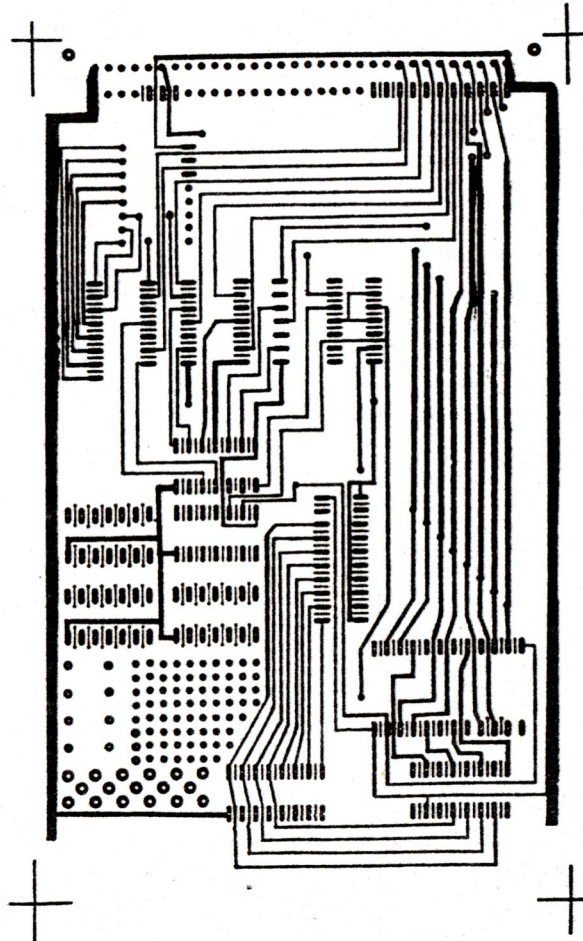
Platinenlayout (ZK01)

Lötseite

Bestückungsplan (ZK 01)



ECB-TRS-80: ZUSATZPLATINE ZK01/02



ECB-Bus Pin-Nr.	Signal	4p-I/O-Bus Pin-Nr.	Besonderheiten
a 14	---	---	nicht belegt!
a 15	- 5V	---	Spannung für ext. Erweiterungen
a 16	2 Phi	---	nicht im 4p-I/O-Bus vorhanden!
a 17	A 17	---	" " "!"
a 18	A 14	---	" " "!"
a 19	+15V	---	Spannung für ext. Erweiterungen
a 20	/M1	47	
a 21	---	---	nicht belegt!
a 22	---	---	" " "!"
a 23	BAI	---	nicht im 4p-I/O-Bus vorhanden!
a 24	VCMOS	---	Spannung für Ext. Erweiterungen
a 25	BAD	---	nicht im 4p-I/O-Bus vorhanden!
a 26	---	---	nicht belegt!
a 27	/IORQ	49	beim 4p /XIOREQ genannt
a 28	/RFSH	---	nicht im 4p-I/O-Bus vorhanden!
a 29	A 13	---	" " "!"
a 30	A 9	---	" " "!"
a 31	/BUSAK	---	" " "!"
a 32	GND	2-50	alle geraden Pins!
b 1	+ 5V	---	Spannung für ext. Erweiterungen
b 2	/MRD	---	nicht im 4p-I/O-Bus vorhanden!
b 3	/MWR	---	" " "!"
b 4	/IN	33	beim 4p /XIN genannt!
b 5	/OUT	35	beim 4p /XOUT genannt!
b 6-9	CP/M	---	projektspez. verwendete Pins!
b10-31	---	---	nicht belegt!
b 32	GND	2-50	alle geraden Pins!
c 1	+ 5V	---	Spannung für ext. Erweiterungen
c 2	D 0	1	Datenleitung 0
c 3	D 7	15	" " 7
c 4	D 2	5	" " 2
c 5	A 0	17	Adressleitung 0
c 6	A 3	23	" " 3
c 7	A 1	19	" " 1
c 8	A 8	---	nicht im 4p-I/O-Bus vorhanden!
c 9	A 7	31	Adressleitung 7
c 10	---	---	nicht belegt!
c 11	IEI	---	Signal der Interruptkette!
c 12	A 19	---	nicht im 4p-I/O-Bus vorhanden!
c 13	-12V	---	Spannung für ext. Erweiterungen
c 14	D 1	3	Datenleitung 1
c 15	-15V	---	Spannung für ext. Erweiterungen
c 16	IEO	---	Signal der Interruptkette!
c 17	A 11	---	nicht im 4p-I/O-Bus vorhanden!
c 18	A 10	---	" " "!"
c 19	A 16	---	" " "!"
c 20	/NMI	---	4p-intern verwendet!
c 21	/INT	39	beim 4p /IOBUSINT genannt!
c 22	/WR	---	nicht im 4p-I/O-Bus vorhanden!
c 23	/GTD	---	nicht im 4p-I/O-Bus vorhanden!
c 24	/RD	---	nicht im 4p-I/O-Bus vorhanden!
c 25	/HALT	---	nur bei mehrfach-CPU-Systemen!
c 26	/PWRCL	37	Peripherie-RESET!
c 27	A 12	---	nicht im 4p-I/O-Bus vorhanden!
c 28	A 15	---	" " "!"
c 29	Phi	---	" " "!"
c 30	/MREQ	---	" " "!"
c 31	/RESET	---	CPU-Reset! Nicht an /XRESET des 4p-I/O-Bus anschließen!!!
c 32	GND	2-50	alle geraden Pins!

Wie man sieht, bleiben die meisten Pins des ECB-Bus leer. Vor allem die für den Anschluß von RAM-Erweiterungen so wichtigen Adressleitungen A8-A18 und die Signale für den DMA (Direct Memory Excess) fehlen. Für die Ansteuerung von portorientierten Erweiterungskarten ist dagegen (fast) alles Nötige vorhanden.

Noch ein paar Worte zu einigen Signalen:

1. Die Spannungen auf dem ECB-Bus (+/- 5, 12 und 15 Volt) dienen der Stromversorgung der Erweiterungsplatinen und werden sowieso von einem externen Netzteil aufgebracht!

2. Die Taktfrequenz der CPU (Phi) sollte man nachträglich auf den nicht benutzten Pin 45 des 4p-I/O-Bus legen (also doch im Gerät löten, aber nur ein kleines bisschen!), da sie von manchen Karten (vor allem solche, die mit der PIO arbeiten) benötigt wird! Das Doppel der Taktfrequenz (2 Phi) wird nur selten benötigt und kann, falls nötig, durch einen mit Phi synchronisierten Taktgenerator erzeugt werden.

3. Das ECB-Signal /RESET ist normalerweise ein Eingang für die CPU-Karten um diese zurückzusetzen. Das 4p-I/O-Bus-signal /XRESET ist ein Ausgang und dient zum Zurücksetzen der Peripherie! Die beiden Signale sollte man nicht durcheinanderbringen bzw. koppeln!

4. Die zum Aufbau einer Interruptkette benötigten Signale IEQ und IEI müssen nicht zur CPU und damit auch nicht zum 4p-I/O-Bus durchgeschleift werden.

5. Wie schon gesagt fehlen die Signale A8-A18, die Speicherzugriffssignale /WR, /WR, /MREQ, /MRD, /MRD und die DMA-Signale BAI und BAD, sowie weitere Leitungen, die aber nur für Speicherzugriffe notwendig sind!

6. Alle Signale des I/O-Bus sind intern gebuffert und können ohne weiteres direkt auf die oben angegebenen Leitungen des ECB-Bus gelegt werden. Zu beachten ist nur noch die Funktion der Leitung /EXTIOSEL. Diese Leitung muß, bevor die CPU auf externe Daten zugreifen kann, auf "low" gelegt werden. Das bewerkstelligt man am besten mittels einer NAND-Verknüpfung von /XIN und der angesprochenen Fortadresse.

Was ist also machbar, was unmöglich? Unmöglich ist zunächst einmal nichts, vorausgesetzt man "besorgt" sich die fehlenden Signale direkt im Gerät, sprich man lötet wild darin herum! Aber auch wenn man das nicht will, bleibt der ECB-Bus interessant. Man kann dann aber "nur" portorientierte Erweiterungen wie Uhr, EPROMmer, Messwerterfassung, Ein/Ausgabe (Relaissteuerungen, Modelleisenbahnsteuerung) und sogar so umfangreiche Projekte wie den c't-Harddisk-Controller, realisieren! Ich hoffe aber, daß sich trotzdem ein paar Unentwegte finden werden, die sich für die ECB-Erweiterung ihres Model 3/4/4p interessieren und so die Gruppe der ECB-Nachbauer und -Begeisterten erweitern. In diesem Sinne viel Spaß und Glück beim Löten, euer

Karstaub Obermann

HEFT
15
September
1986

16

Hilfe!

Hilfe!

Ich bin ein Disketten-
Verschwendler!

Wer hat? DoublerWer weiß? gesucht

Wer weiß wo?
für GENIE

Herbert Albers
Zur Dänishöfen 14
2117 Wistedt
04182/8799

Hilfe!

Hilfe!

Ich suche Leute, die mit FORTRAN programmieren!

Günther Wagner
SUCHE aus den Jahren 1985/86 versch. LOAD 80 - Disketten.
Klaus Hermann, Tel. 07127/70024

Wer Interesse an einer
GENIE 3/3s - ECKE
in unserer Club-INFO hat,
sollte sich beim lieben Arnulf melden!!

Verkaufe:

GRIP 2.1 / 2.6 / 3.1
780 x 280 Punkte Grafik-Auflösung,
96 Buchstaben je Zeile — 33 Zeilen,
Hardware-Spooler, V24, Druckausgabe,
Tastatureingabe seriell oder parallel.
(Beschrieben in der c't)

VP: 350,— DM
Manfred Held

Normung und Rechtschreibung

Nachdem in unserer letzten "CLUB-Info" wieder der Druckfehlerteufel grausam zugeschlagen hatte, möchte ich Euch heute eine Vereinfachung in der Rechtschreibung vorschlagen. Mit dieser Rechtschreibreform werden in Zukunft sicher keine Fehler mehr auftauchen.

Die neue Rechtschreibreform sollte "Schritt für srit" folgendermaßen eingeführt werden:

Erster Schritt:

Wegfall der Großschreibung!

einer sofortigen einföhrung steht nichts mehr im wege, zumal schon viele grafiker und werbeleute zur kleinschreibung übergegangen sind.

zweiter schrit:

wegfall der dehnungen und schartungen!

dise masname eliminiert schon di größte felerursache in der grundschule, den sin oder unsin unserer konsonantenverdopelung hat onehin nimand kapirt.

dritter schrit:

v und p werden ersetzt durch f,

sch, tz und z durch s!

das alfabet wird um swei buchstaben redusiert, sreibmaschinen und sesmaschinen fereinfachen sich, wertfole arbeitskräfte können der wirtsaft sugeführt werden.

firter srit:

q und ch werden ersetzt durch k,

j und y werden ersetzt durch i,

fh (ph) wird ersetzt durch f!

iest sind son seks bukstaben drausen und di sfrake woltuend klar, di sulseit kan sofort fon neun auf swei iare ferkürzt werden. anstat aksig frosent reksreibunterikt können nüslikere fäker, wi fisik, kemi und reknen mer geflegt werden.

fünfter srit:

wegfal fon ä-, ö-, ü-seiken!

ales überflusige ist iest ausgemerst, di ortografi ist wider slikt und einfak, naturlik -benotigt es einige Zeit, bis dise fereinfakung ublical riktig ferdaut wird, fileikt sasungsweise ein bis swei iare.

dem folgt ein sekster srit:

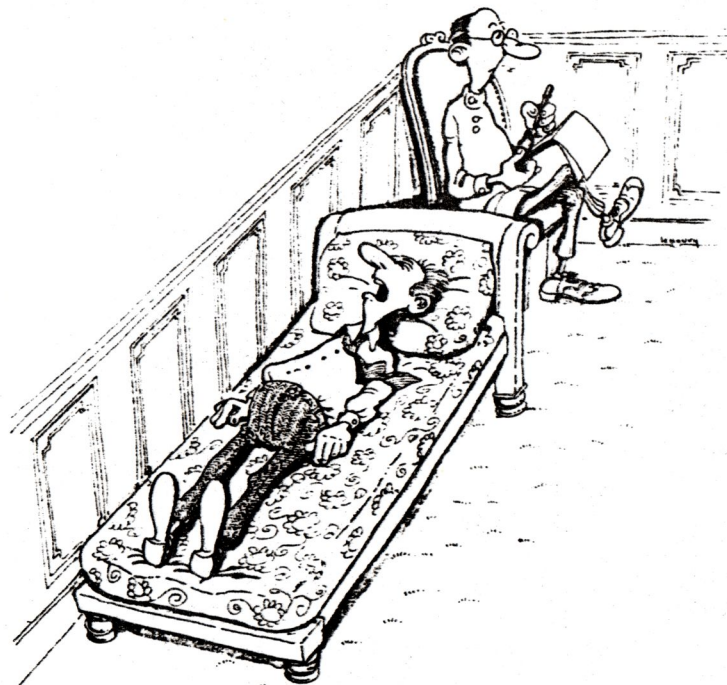
g ersetzt durch k, t ersetzt durch d,

und warum ersesd man nikd w durch f?

wen ale dise fordsride durkkessd sind, durfde als naksdes zil di fereinfakunk der nok sfirken und unsinikeren deudsen kramadik anfisird ferden.

Di Redakdion

PS: Wer solche Texte nachmacht oder verfälscht oder nachgemachte oder verfälschte sich verschafft und in die Club-INFO bringt, wird mit Computerentzug nicht unter zwei Jahren bestraft.



Mein Computer versteht mich nicht!
Immer Syntax Error, Syntax Error!

Assoziationen eines Assemblerers
"auf der Suche nach dem verlorenen Mnemonic"
(Text frei nach Marcel Proust; Ton nach Richard Strauß)

Ein Public Domain-Programm in JotCal, das auf allen Modellen lauffähig ist! Originaldiskette bei KalJot erhältlich.
Filename: "BABYLON"

1 ' Dunkler ASSEMBler	; A
2 'Finstere ASSOziationen	
3 'Keller-ASSeln	; u
4 'KellergerASSel	
5 'Untergrund-Geflüster	; f
6 'Grundtöne	
7 'Grundsprache	; S
8 'Basis-Sprache	
9 'BASIC-Sprache	; t
10 'BASF-Sprache	
11 'BAYER-Sprache	; i
12 '(Bayrisch)	
13 'Hochsprache	; e
14 'HOECHST-Sprache	
15 'Diskant	; g
16 'KANT's Sprache	
17 'Kannit-verstan	
18 'Kanni-balen-Sprache	; &
19 'Ver-bal-Sprache	
20 'Schriftsprache	
21 'Geheimsprache	; N
22 'Geheim-Code	
23 'Hieroglyphen	; i
24 'Hiro Hito	
25 'Heros-Hymnen	; e
26 'Heldengesänge	
27 '4 Letzte Lieder (Rich. Strauß)	; d
28 'Schwanengesang	
29 'Abgesang	; e
30 'Absprache	
31 'Abwärtssprache	; r
32 'Tiefsprache	
33 'Kellersprache	; g
34 'Kellerasseln	
35 'Rasseln	; a
36 'Röcheln	
37 'Röhren	; n
38 'Retörnen	
39 'RETURN	; g
40 'RET <<==== heureka (ich hab's) !	

END (Lied)

37



* Neues von der DISKO-Front ! *

Unsere DISKOTHEK ist gewaltig gewachsen!
Der Löwenanteil stammt diesmal aber nicht aus den eigenen Reihen.
Wie ist das zu erklären? Haben wir uns etwa doch entschlossen, allerlei Software zu kopieren, Copyrights zu umgehen, Protected Programs zu knacken???

Honny soit qui mal y pense!
Keineswegs!

Wir haben ein großes Paket sogenannte "Public Domain Software" an Land gezogen: gut 250 Programme aus USA, verteilt auf 26 Disketten!
Es ist eine Menge Nützliches und Interessantes darunter. Seht Euch den nachstehenden Katalog an!

Übrigens: Diese Programme wurden nicht zusätzlich in Katalog der Diskothek aufgenommen, weil ein erschöpfender Katalog mit Erläuterungen schon existiert, wie Ihr seht. Sie befinden sich auf 40Tr./SS/SD-Disketten. Wenn jemand etwas davon wünscht, so erhält er grundsätzlich eine Kopie der kompletten Diskette, auf der sich sein(e) Wu(e)nschle befindet! Meistens gehören ohnehin mehrere Files zu einem Programm. Und 40/SS/SD kann ja jeder lesen und sich ggf. unformatieren. - Eine wichtige Bitte:

Bei Bestellungen bitte die Disk-Nummer angeben!

Nun kann's losgehen! Ich rechne mit einer Sintflut von Programm-Anforderungen (möglichst auf eigenen, noch unformatierten Disketten). Wer sie gleich selbst formatieren will, der verwende für TRS80:
TI=A TD=A TC=39 (oder 40) SPT=10 TSR=be1. GPL=2 DDS1=17 DDGA=2
bzw. die entsprechenden deutschen Parameter für das GENIE.

Viel Erfolg + Spaß mit der "PD"!
Wünscht:

EVER GENIE

Unsere 'Public Domain' - Software

DISK 01 - NEWDOS80 PROGRAMME

APD/ASM	- Quellcode fuer ein PDRIVE-Automatikprogramm
APD/DOC	- Dokumentation fuer APD/ASM, APD1/CMD, APD3/CMD
APD1/CMD	- PDRIVE-Automatik fuer das Modell I
APD3/CMD	- PDRIVE-Automatik fuer das Modell III
CATALOG1/BAS	- Diskettenkatalog-Programm fuer NEWDOS80-Benutzer
COMFDIR/BAS	- Vergleicht die DIRECTORY zweier Disketten und gibt das Ergebniss auf dem Drucker, Bildschirm bzw. als Diskfile aus.
COMFDIR/DOC	- Dokumentation fuer COMFDIR/BAS
DIALND/CMD	- Terminalprogramm (waehlt Telefonnummern autom.)
EDTASMN/TXT	- Kurzanleitung fuer EDTASMN/CMD
FASTMENU/CMD	- Wahlmenue, ermoeeglicht Programmstart mit einem Tastendruck (nur Modell I)
HELP/CMD	- Erklaert alle NEWDOS-Befehle
NAMEIT/BAS	- Ermoeeglicht Aenderung der NEWDOS80-Meldung beim booten der Systemdiskette
NAMEIT2/BAS	- Ermoeeglicht Aenderung der NEWDOS80 READY-Meldung und des BASIC-Logos
PDRIVE/BAS	- Analysiert die PDRIVE-Daten einer Fremdiskette
REROUTE/ASM	- Quellcode fuer REROUTE/CMD, Dokumentation in Kommentarzeilen
REROUTE/CMD	- Leitet die Druckerausgabe auf ein Diskfile um
TRSDIR/ASM	- Quellcode fuer TRSDIR/CMD
TRSDIR/CMD	- Zeigt die Directory einer Modell III TRSDOS 1.3 Diskette unter NEWDOS80 (Modell I/III) an
TRSDIR/DOC	- Dokumentation fuer TRSDIR/ASM und TRSDIR/CMD

DISK 02 - VERSCHIEDENE MODELL I/III PROGRAMME

CALC/CMD	- Erlaubt Berechnungen auf Betriebssystemebene, Syntax: CALC Formel z.B.: CALC 2/4+5
CHECKER/BAS	- Vergleicht zwei im ASCII-Format abgespeicherte BASIC-Programme und druckt die Unterschiede aus
COMFARE/CMD	- Vergleicht zwei Files bytewise, Syntax: COMFARE FILE1 FILE2
DISKINDX/BAS	- Druckt DISKINDX/DAT (von DISKINDX/CMD)
DISKINDX/CMD	- Diskettenkatalogisierprogramm (max. 2500 Prg.)
DISKINDX/DOC	- Anleitung fuer DISKINDX/CMD
DOU/CMD	- Fuehrt eine Sequenz von DOS-Befehlen (aehnlich /JCL) aus
DOU/DOC	- Dokumentation zu DOU/CMD
LOADADDR/BAS	- Analysiert /CMD-Files und gibt die Ladeadresse im RAM an
NOTE/CMD	- Ermoeeglicht das Anlegen von Notizen, Kommentaren und Instruktionen auf der Disk. Selbsterklaerend
ORGAN/CMD	- Gibt Musik ueber den Cassettenport wieder
PERUSE/CMD	- Erzeugt ein Programm, welches Textfiles in Lesegeschwindigkeit auflistet. Syntax: PERUSE TEXT/TXT
PILOT/BAS	- Dieses BASIC-Programm poked einen PILOT-Interpreter in den oberen RAM-Bereich
PILOT/INS	- Instruktionen fuer PILOT/BAS in PILOT. Programmablauf: RUN "PILOT/BAS" LOAD "PILOT/INS" NAME
REPLACER/BAS	- Benennt Variablenamen in BASIC-Programmen um
REPLACER/DOC	- Dokumentation zu REPLACER/BAS
SCHEDULE/BAS	- Terminkalender in BASIC
SETDATE/ASM	- Quellcode fuer SETDATE/CMD; Dokumentation in Kommentarzeilen
SETDATE/CMD	- Verbesserte Datumseingabe auf DOS-Ebene

HEFT
15
September
1986

SUPERLST/CMD - Listet BASIC-Programme uebersichtlicher
 SUPERLST/DOC - Dokumentation zu SUPERLST/CMD

DISK 03 - TERMINAL- und KOMMUNIKATIONS-PROGRAMME

BINHEX/CMD - Konvertiert ASCII Hex Files in Binaerfiles und umgekehrt. Gibt die Checksumme für jedes File an
 BUSY/CMD - Beantwortet ankommende Anrufe mit "Computer is busy"; Backgroundtask fuer autom. antwortende Modems
 LHELP/TXT - Help file fuer LTERM/CMD
 LTERM/CMD - LTERM Terminalprogramm Version 3.0a
 LTERM/DOC - Dokumentation zu LTERM/CMD
 LYNXTERM/CMD - STERM 1.6 an das LYNX-Modem angepasst
 LYNXTERM/DOC - Dokumentation zu LYNXTERM/CMD
 MODEM/CMD - Modem Terminalprogramm Version 1.1b
 MODEM/DOC - Dokumentation zu MODEM/CMD
 ORCONV/CMD - Konvertiert ORCHESTRA-80/85/90 Files in ASCII und umgekehrt
 ORCONV/DOC - Dokumentation zu ORCONV/CMD
 STERM/CMD - STERM Terminalprogramm Version 1.6b
 STERM/DOC - Dokumentation zu STERM/CMD
 XMOD1200/CMD - 1200 Baud Version von XMODEM, ein Filetransferprogramm. Autom. Fehlererkennung und -korrektur
 MOD300/CMD - 300 Baud Version von XMODEM
 XMODEM/DOC - Dokumentation zu XMODEM1200 und XMODEM300

DISK 04 - VERSCHIEDENE MODELL 4, MULTIDOS, DOSPLUS, CASS.-PRG.

CLRDIR/CMD - Loescht unter MULTIDOS 1.x unbenutzte Directory Eintraege; Syntax CLRDIR :1
 DCTDSF/CMD - Zeigt das "DEVICE CONTROL TABLE" unter TRSDOS6.x an (Modell 4)
 DISKCAT/BAS - Diskettenkatalogisierungsprogramm unter DOSPLUS
 DISKCAT/DOC - Dokumentation zu DISKCAT/BAS
 FREEMAP/CMD - Zeigt die freien "GRANULES" unter MULTIDOS 1.x an Syntax: FREEMAP :1
 KBMOD/ASM - Quellcode fuer KBMOD/CMD; incl. Kommentarzeilen
 KBMOD/CMD - Modell 4 Programm; installiert einen "FULL-SCREEN-EDITOR" und die hohe Taktfrequenz im Modell III-Modus
 LABEL4/BAS - Druckt auf dem Modell 4 Disklabels
 LCOMM/TXT - Benutzungsanleitung fuer LCOMM
 MEMDISK4/CMD - Installiert auf dem Modell 4 eine RAM-DISK
 MOD4BASC/BAS - Konvertiert Modell I/III BASIC-Programme in Modell 4 BASIC-Programme
 MOD4BASC/DOC - Dokumentation zu MOD4BASC/BAS
 MOD4INFO/TXT - Patches fuer TRSDOS 6.0 und weitere M.4 Infos
 MODIII4F/BAS - BASIC-Subroutines, um auf dem Modell 4 im Modell III-Modus die Taktfrequenz umzuschalten
 MOVESYS4/JCL - Modell 4 JCL-File um alle SYS-Files auf die RAM-DISK zu uebertragen
 NEWLIST/CMD - BASIC Programmlister Version 6.5; unter DOSPLUS, eventuell auch unter anderen Betriebssystemen
 NEWLIST/DOC - Dokumentation zu NEWLIST/CMD
 ORGAN4/CMD - Ermoeeglicht Musikwiedergabe ueber das Modell 4 Sound Modul
 SCANDAT/BAS - Utility fuer DISKCAT/BAS; listet und sucht im Diskettenkatalog
 SOUND134/BAS - Demonstration von Sound-Routinen, fuer M.I/III/4
 SPEED/CMD - Utility fuer das Modell 4; SPEED 4 bzw. SPEED 2

stellen die gewuenschte Taktfrequenz der CPU ein
 TEXEDIT4/BAS - ASCII File-Editor Version 4.1 fuer das Modell 4
 TTERM16K/BAS - Terminal-Programm fuer 16 K Cassettsystem-User
 TTERM32K/BAS - Terminal-Programm fuer 32 K Cassettsystem-User

DISK 05 - PROGRAMME VON NORTHERN BYTES

ART6809/TXT - Dokumentation zu ASM6809/BAS, PONG/ASM, SETRES/ASM und VARS/TXT
 ASM6809/BAS - Ein 6809 Assembler in MICROSOFT BASIC
 CENTRJKL/ASM - Eine NEWDOS80-JKL-Routine für CENTR. 739 Drucker
 COCOPY/BAS - Ein Modell 100 Assembler File Transferprogramm
 Dokumentation in den ersten Programmzeilen
 EDPATCH/ASM - Passt EDTASMN PLUS fuer das M.1 an das M.3 an
 EDPATCH/TXT - Dokumentation zu EDPATCH/ASM
 EPSONJKL/ASM - NEWDOS80-JKL-Routine fuer EPSON Drucker
 FDCIII/BAS - WESTERN DIGITAL 1793 Floppy Disk Controller-Test
 FREEMAP/CMD - Zeigt die freien Granules unter MULTIDOS 1.x Syntax: FREEMAP :1
 JKL/TXT - Dokumentation zu CENTRJKL/ASM und EPSONJKL/ASM
 PONG/ASM - Ein Beispiel 6809 - Assemblerpr. für ASM6809/BAS
 SETDATE/CMD - Verbesserte Datumseingabe beim Booten der Disk. Verbesserte Version des Programms auf Disk 02
 Dokumentation in NORTHERN BYTES, Volume 5, No.2
 SETRES/ASM - Ein Beispiel 6809 - Assemblerpr. für ASM6809/BAS
 V/CMD - Filevergleichsprogramm, verbesserte Version von COMPARE/CMD auf Disk 02, Syntax: V File1 File2
 VARS/TXT - Liste aller benutzten Variablen von ASM6809/BAS
 ZAPDEBUG/ASM - QUELLCODE fuer ZAPDEBUG/CMD
 ZAPDEBUG/CMD - Zap Utility fuer TRSDOS 1.38
 ZAPDEBUG/TXT - Dokumentation zu ZAPDEBUG/ASM und ZAPDEBUG/CMD

DISK 06 - VORWIEGEND SPIELPROGRAMME

COMPDIR/BAS - NEWDOS80-Utility, vergleicht die Directories zweier Disketten und gibt das Ergebnis auf dem Schirm, Drucker und als Diskfile wieder. Verbesserte Version von COMPDIR/BAS auf DISK 01
 CONCEN/BAS - Konzentrationsspiel (PUZZLE)
 EMPIRE/BAS - Ein Wirtschaftssimulationsprogramm fuer mehrere Spieler, die Staaten regieren
 FLIGHTDC/BAS - Dokumentation zu FLIGHTSM/BAS
 FLIGHTSM/BAS - Flugsimulator
 OTHELLO/BAS - OTHELLO-Spiel fuer einen oder mehrere Spieler
 REACTOR/BAS - Simuliert einen Kernreaktor (mit Graphik)
 TTT/ASM - Quellcode fuer TTT/CMD
 TTT/CMD - Dreidim. TICTACTOE (in einer 4x4x4-Matrix)

DISK 07 - PROGRAMME AUS NORTHERN BYTES VOLUME 5 NO. 5

BASICOMP/BAS - Ein BASIC Compiler fuer die Modelle I u. III
 BASICOMP/DOC - Dokumentation zu BASICOMP/BAS u. SAMPLE/BAS
 BINCLOCK/ASM - Quellcode zu BINCLOCK/CMD
 BINCLOCK/CMD - "BINAERUHR" fuer die Modelle I u. III
 BINCLOCK/DOC - Dokumentation zu BINCLOCK/CMD
 CODE/ASM - Quellcode für CODE/CMD, Dokumentation in Kommentarzeilen
 CODE/CMD - Ver- und entschluesst ASCII-Files
 SAMPLE/BAS - Beispielprogramm zu BASICOMP/BAS
 SD456/ASM - Quellcode zu SD456/CMD (mit mehrfachen DEF-Statements, siehe SD456/DOC)

HDSTART/ASM - Quellcode fuer HDSTART/CMD
 HDSTART/CMD - Patch fuer NEWDOS80V2.5, ermoeeglicht das gleich-
 zeitige Betreiben einer HARDDISK mit dem HOLMES
 ENGINEERING SPRINTERMODUL (SPEED UP)
 HDSTART/DOC - Dokumentation zu HDSTART/CMD
 SD/ASM - Quellcode fuer SD/CMD
 SD/CMD - Sortierte Directory mit (mit wildcards!)
 SD/DOC - Dokumentation zu SD/CMD



*Tut mir leid, Chef! Wir spielten eines dieser Mikrospiele,
 und ich brauchte die ganze Nacht, um aus dem Labyrinth zu entkommen.*

DISK 12 - POTFOURRI

DATASET/ASM - Quellcode fuer DATASET/CMD; mit Kommentarzeilen
 DATASET/CMD - Ähnlich SETDATE (DISK 02); Datum im T/M/J-Format
 DISTANCE/BAS - Berechnet die Distanz und den Kurs zwischen zwei
 Staedten
 DS/ASM - Quellcode fuer DS/CMD
 DS/CMD - Wie DATESSET/CMD, schreibt das Datum als Diskfile
 (DATE/TXT) auf Disk, dadurch Uebernahme in die
 Textverarbeitung moeglich
 GM/BAS - Attraktives Menue-Programm, Anpassung notwend.
 MAP/CMD - Utility fuer MULTIDOS, gibt die Lage von Files
 auf der Diskette an. Syntax: MAP :drive
 MSTRMIND/NB - MASTERMIND in NEWBASIC (MODULAR SOFTWARE)
 P/CMD - Stellt die Druckmodi von ITOH-Druckern ein
 SD/ASM - Quellcode zu SD/CMD, mit Kommentarzeilen
 SD/CMD - Wie SETDATE/CMD (DISK 02), schreibt Datum als
 Textfile auf Disk; Uebernahme in Textver-
 arbeitung moeglich
 SETPRT/BAS - Quelltext zu P/CMD - fuer ZBASIC-Compiler
 SUNRISE/BAS - Angabe von Sonnen Auf- und Untergangszeiten fuer
 alle geographischen Orte (Angabe in GMT)
 WEEKDAYS/NB - Ein Spiel in NEWBASIC (MODULAR SOFTWARE)

DISK 13 - Programme aus NORTHERN BYTES VOLUME 6, No. 1-2

AUSDATE/ASM - Quellcode zu AUSDATE/CMD, mit Kommentarzeilen
 AUSDATE/CMD - Revidierte Version von DATESSET/CMD (DISK 12),
 speichert Datum im Tag/Monat/Jahr-Format
 BLANK/ASM - Quellcode zu BLANK/CMD, mit Kommentarzeilen
 BLANK/CMD - Modell III-Utility, blendet Bildschirm bei
 Nichtbenutzung des Rechners nach einer Wartezeit
 aus, nach beliebiger Tastatureingabe Restaura-
 tion des Bildschirmes (schont den Monitor)
 BLANK1/CMD - Modell I-Version von BLANK/CMD (Zeile 1810 im
 Quellcode ist von 447BH auf 4410H geaendert)
 DEMO - BASIC-Programm, Demo zu BLANK/CMD, Start mit
 START/JCL
 FIXHIT/ASM - Quellcode zu FIXHIT/CMD
 FIXHIT/CMD - NEWDOS80V2-Utility, rekonstruiert bei defekten
 Disketten das HASH INDEX TABLE (alle Formate)
 FIXHIT/DOC - Dokumentation fuer FIXHIT/CMD
 FKEY/ASM - Quellcode zu FKEY/CMD
 FKEY/CMD - M. 4-TRSDOS-Utility, weist den Funktionstasten
 neue Codes (vom Benutzer definierte Codes zu
 Dokumentation zu FKEY/CMD
 FKEY/DOC - Dokumentiert den Gebrauch von Funktionen in
 BASIC-Programmen
 FNDEMO/BAS - Dokumentation zu FNDEMO/BAS
 FNDEMO/DOC - Laedt die Standard-TRSD-Blockgraphik in den
 FX80, so das EPSON MX80-Graphik-Programme
 lauffaehig werden
 FX80/DOC - Dokumentation zu FX80/BAS
 PROMPT/BAS - Modell I-Utility, ermoeeglicht Aenderung des
 "BASIC-Prompts: READY"
 START/JCL - JCL-FILE, initialisiert BLANK/CMD und DEMO
 TRACE/ASM - Quellcode zu TRACE/CMD
 TRACE/CMD - Aehnlich der TRACE-Funktion unter TRSDOS2.3 aber
 fuer NEWDOS80V2
 TRACE/DOC - Dokumentation zu TRACE/CMD

DISK 14 - PROGRAMME AUS NORTHERN BYTES VOLUME 6, NO. 3

DGRAPH/BAS - BASIC-Demo fuer VDCTL/BAS
 FIXGAT/ASM - Quellcode zu FIXGAT/CMD
 FIXGAT/CMD - NEWDOS80V2-Utility, repariert den GAT-Sektor der Directory (SS und DS-Disks)
 FIXGAT/DOC - Dokumentation zu FIXGAT/CMD
 RD/ASM - Quellcode zu RD/CMD, inklusive Kommentarzeilen
 RD/CMD - Revidierte Version von DS/CMD (DISK 12), DOS-Patch speichert Datum im TT/MM/JJ-Format
 RD/DOC - Dokumentation zu RD/CMD
 TATUPNI/ASM - Quellcode zu TATUPNI/CMD
 TATUPNI/BAS - BASIC-Demo zu TATUPNI/CMD
 TATUPNI/CMD - NEWDOS80V2-Utility, BASIC-Erweiterung, Input an beliebiger Bildschirmposition. M. III-TRSDOS -Version: Zeilen 350-370 im Quellcode aendern!
 TATUPNI/DOC - Dokumentation zu TATUPNI/CMD
 VDCTL/ASM - Quellcode fuer VDCTL/OBJ
 VDCTL/DOC - Dokumentation zu VDCTL/OBJ
 VDCTL/OBJ - Modell 4-Utility, erlaubt direkten Zugriff auf das VIDEO-RAM

DISK 15 - "THE CHICAGO GREENE MACHINE BBS", UND WHERE/CMD

AD - *****
 BBS/JCL - * Folgende Programme gehoeren zu *
 BLDMENU/BAS - * "THE CHICAGO GREENE MACHINE BULLETIN BOARD" *
 CHATFLAG - * Von George Matyaszek, Sysop *
 CHATFLAG/ORG - * Infos unter HOWTORUN/TXT *
 COMMAND/BAS - * Nur fuer TRS80 Modell I unter NEWDOS80V2 *
 DTEGTS/CMD - * *
 DLMOD/BAS - * (EIN KOMPLETTES MAILBOX-SYSTEM) *
 DRIVER/CMD - * *
 EMAIL - * *
 ENTRY/BAS - * *
 HOWTORUN/TXT - * *
 INFO.FCL - * *
 LOGREC/BAS - * *
 MANAGER/BAS - * *
 MODEMBBS1/CMD - * *
 PACKER/BAS - * *
 USE - *****
 WHERE/ASM - * Quellcode fuer WHERE/CMD, verbesserte Version des Programmes auf DISK 10. *
 WHERE/CMD - * Genaue Angabe der Fehlerposition in BASIC-Programmzeilen *
 XMODEM/CMD - * Gehoert zu "THE CHICAGO GREENE MACHINE BBS" *

DISK 16 - VERSCHIEDENE MODELL I/III-PROGRAMME

CAT/CMD - NEWDOS80-UTILITY, gibt die Belegung der Diskette an. Nach Filenamen, relative Sektoren und Spur- und Sektornummer
 CAT/DOC - Dokumentation zu CAT/CMD
 CAT/SRC - Quellcode fuer CAT/CMD
 CLAWDOS/CMD - Modell III "MINI OPERATING SYSTEM" im TRSDOS1.3-FORMAT
 CLAWDOS/DOC - Dokumentation zu CLAWDOS/CMD
 CLAWDOS/SRC - Quellcode zu CLAWDOS/CMD
 CLONE1/CMD - NEWDOS80- und TRSDOS1.3-Utility, sehr schnelles "filecopy"

CLONE1/DOC - Dokumentation zu CLONE1/CMD
 CLONE1/SRC - Quellcode fuer CLONE1/CMD
 DIS/CMD - NEWDOS80-Utility, sortierte Directory, in fuenf Spalten (statt vier)
 DIS/DOC - Dokumentation zu DIS/CMD
 DIS/SRC - Quellcode fuer DIS/CMD
 FORM/CMD - Schnelles Diskettenformatierungsprogramm mit vielen Optionen, erzeugt keine Directories
 FORM/DOC - Dokumentation zu FORM/CMD
 FORM/SRC - Quellcode fuer FORM/CMD
 GRAPHPRO/BAS - Graphikeditor, gibt die Graphik in PRINT-Statements aus, die in BASIC-Programme eingefuegt werden koennen
 GRAPHPRO/DOC - Dokumentation zu GRAPHPRO/BAS
 PHONETXT/BAS - Untersucht amerikanische Telephonnummern auf enthaltene Buchstabenkombinationen
 SUPCALC2/BAS - Taschenrechnerprogramm in doppelter Genauigkeit (inclusive Konstanten: PI, E, etc.)
 SUPCALC2/DOC - Dokumentation zu SUPCALC2/BAS

DISK 17 - VIDEO STORE MANAGER UND DIV. PROGRAMME VON M. PATRICK

CONVBIN/ASM - Quellcode fuer CONVBIN/CMD
 CONVBIN/CMD - Gibt fuer jede Tastatureingabe de Binaercode an (Modell I/III)
 TRACK/CMD - Komplettes Verwaltungsprogramm fuer einen VIDEO-Laden (Unter TRSDOS1.3 oder DOSPLUS, ev. andere DOSSE). Der Quellcode benoetigt einen Assembler der Get bzw. Include Files unterstuezt
 TRA/ASM - Teil A des Quellcodes fuer TRACK/CMD
 TRE/ASM - Teil B des Quellcodes fuer TRACK/CMD
 TRC/ASM - TEIL C des Quellcodes fuer TRACK/CMD

DISK 18 - HELP FILE GENERATOR, DOKUMENTATION FUEER VIDEO STORE

MANAGER UND DIVERSES (VON MAL PATRICK)

TRACK/DOC - Dokumentation fuer TRACK/CMD von Disk 17
 UNIHLP/ASM - Quellcode zu UNIHLP/CMD
 UNIHLP/CMD - Modell I/III-Utility fuer alle Betriebssysteme. Generiert und editiert HELP-FILES
 UNIHLP/DOC - Dokumentation fuer UNIHLP/CMD
 WEMAPPED/DOC - Text: Anleitung zum "patchen" von WordStar3.0 unter MONTEZUMA CPM2.2 (Modell 4). Die Bildschirmroutinen werden wesentlich beschleunigt

DISK 19 - KEYBOARD MACRO, FILE UTILITIES UND ANDERE PROGRAMME

- KEYMAC/ASM - Quellcode zu KEYMAC/CMD
 KEYMAC/CMD - Model 3-Utility, Tastaturerweiterung um Woerter bzw. um Wortfolgen. Abspeicherung der Tastaturdefinitionen auf der Disk. Die Definitionen duerfen auch "RETURNS" enthalten. UNMACRO/CMD beendet diese Utility
- KEYMAC/DOC - Dokumentation zu KEYMAC/CMD
 MASTER/ASM - Quellcode des Hauptmoduls von VFU/CMD
 NOCAT/ASM - Quellcode fuer NOCAT/CMD
 NOCAT/CMD - NEWDOS80-Utility, Modell 1/III, druckt sortierten Filekatalog auf dem Bildschirm. Benutzt eine "Screen Sort"-Routine. Fuer DOSPLUS3.4 sind zwei Patches notwendig.
- SECT1/ASM - Quellcode des 1. Untermoduls von VFU/CMD
 SECT2/ASM - " " 2. " " / "
 SECT3/ASM - " " 3. " " / "
 UNMACRO/ASM - Quellcode fuer UNMACRO/CMD
 UNMACRO/CMD - Entfernt KEYMAC/CMD aus dem RAM und setzt den HIGHMEM PDINTER auf den gewuenschten Wert
- VFU/CMD - Modell 1-NEWDOS80V2-Utility (die Patches fuer DOSPLUS sind im Quellcode vermerkt), vielseitige File-Utility mit eingebauter HELP-Funktion. Keine modifizierte Version von VFU/CMD von MULTIDOS!
- VFU/DOC - Dokumentation zu VFU/CMD

DISK 20 - WEITERE PROGRAMME VON MEL PATRICK

- ARROWFRT/ASM - Quellcode fuer ARROWFRT/CMD
 ARROWFRT/CMD - Undefinedion der Pfeiltasten. Die Pfeiltasten werden auf EPSON Druckern als Pfeile gedruckt. Alle Betriebssysteme, Modell 1/III
- DISCAT/ASM - Quellcode fuer DISCAT/CMD
 DISCAT/CMD - Creaert Diskettenkatalog (NEWDOS80V2 u. DOSPLUS)
 DOTPRINT/DOC - Textfile: Anleitung um ältere DOTPRINT-Versionen ALLWRITE anzupassen.
- HEXCAL/ASM - Quellcode fuer HEXCAL/CMD
 HEXCAL/CMD - Hexadecimal Calculator, arithm. Funktionen, log. Verknuepfungen (AND, OR, XOR). Eingebaute HELP-Funktion. Ferner sind Anzeige von Speicherinhalten und deren Modifikation moeglich. Alle Betriebssysteme, Modell 1/III
- MARQ/ASM - Quellcode fuer MARQ/CMD
 MARQ/CMD - Generiert einen Laufrahmen um den Bildschirm. Alle Betriebssysteme, Modell 1/III
- NEWBLD/ASM - Quellcode fuer NEWBLD/CMD
 NEWBLD/CMD - Ersetzt das CHAINBLD/BAS der NEWDOS80-Diskette. Benoetigt weniger Speicherplatz
 NEWBLD/DOC - Dokumentation zu NEWBLD/CMD

DISK 21 - DRUCKER PROGRAMME UND FILEVERGLEICHS-UTILITIES

- COMPARE/ASM - Quellcode fuer COMPARE/CMD
 COMPARE/CMD - Einfache Filevergleichs-Utility, Modell 1/III, alle Betriebssysteme
- EDTFMT/ASM - Quellcode fuer EDTFMT/CMD
 EDTFMT/CMD - Formatierte Druckerausgabe (Titel, Seitennummer) von Standard EDTASM-Files. Optionen: Aneinanderfuegen mehrerer Files, entfernen von Kommentarseiten. Modell 1/III, alle Betriebssysteme
- FONTDVR/ASM - Quellcode fuer FONTDVR/CMD

- FONTDVR/CMD - Utility fuer ELECTRIC PENCIL. Druckt den mit dem von FONTWRTR/CMD creierten Zeichensatz. Arbeitet auch mit beliebigen BASIC-Programmen zusammen. Modell 1/III, alle Betriebssysteme
- FONTDVR/DOC - Dokumentation zu FONTDVR/CMD
 FONTWRTR/ASM - Quellcode fuer FONTWRTR/CMD
 FONTWRTR/CMD - Creaert Zeichensatz fuer EPSON Drucker. Modell 1/III, alle Betriebssysteme. Benutzung zusammen mit FONTDVR/CMD
- FONTWRTR/DOC - Dokumentation zu FONTWRTR/CMD

DISK 22 - KOMMUNIKATIONSPROGRAMME UND VERMISCHTES (MEL PATRICK)

- ASCFMT/ASM - Quellcode fuer ASCFMT/CMD
 ASCFMT/CMD - Formatierte Druckerausgabe von BASIC-Programmen (im ASCII-Format), Option: normal/single statement pro Druckzeile, M. 1/3, alle Betriebss.
- BYE/ASM - Quellcode fuer BYE/CMD
 BYE/CMD - Utility fuer SMARTDVR/CMD. Erlaubt dem Anrufer, sich durch die Eingabe von BYE (im DOS READY Mod.) aus dem System auszuloggen. Dokumentation in SMARTDVR/DOC.
- DIALER/ASM - Quellcode fuer DIALER/CMD
 DIALER/CMD - Ergaenzung fuer Terminalprogramme. Erstellt Listen von Telefonnummern und waehlt daraus Nummern automatisch an. Start durch gleichzeitiges Druecken 5,6,7-Tasten. Fuer Patches siehe den Quellcode. Modell 1/III, alle Betriebssysteme
- HAYES/ASM - Quellcode fuer HAYES/CMD
 HAYES/CMD - Sendet "AT"-Code ans HAYES-Modem (zusammen mit HOST80/CMD), Modell 1/III, alle Betriebssysteme
- HOST80/ASM - Quellcode fuer HOST80/CMD
 HOST80/CMD - "HOST DRIVER PROGRAM", arbeitet mit den meisten Direktmodems zusammen. Viele Optionen, Modell 1 und III, alle Betriebssysteme
- MORSE/ASM - Quellcode fuer MORSE/CMD
 MORSE/CMD - Morseuebungsprogramm, Modell 1/III, alle Betriebssysteme
- SMARTDVR/ASM - Quellcode fuer SMARTDVR/CMD
 SMARTDVR/CMD - Bindet das HAYES SMARTMODEM ins System ein. Kann als Treiberprogramm fuer ein BBS-System verwendet werden
- SMARTDVR/DOC - Dokumentation zu SMARTDVR/CMD und BYE/CMD

DISK 23 - FILE MANAGER PROGRAMM (VON DAN SHELBY) UND PROGRAMME

AUS NORTHERN BYTES VOLUME 6, NO. 4-5

- BONDS/EE5 - Demonstrationsfile zu FILMGR48/BAS
 BONDS/RPG - " " " "
- FILMGR48/BAS - Siehe README/BAS und FILMGR48/DOC
 FILMGR48/BAS - FILE MANAGER PROGRAMM von DAN SHELBY. Siehe README/BAS und FILMGR48/DOC
- FILMGR48/DOC - Dokumentation zu DAN SHELBY'S FILE MANAGER PROGR. Weitere Informationen in README/BAS
- FILMGR48/FIX - FILE MANAGER Filekonvertierungsprogramm. Weitere Informationen in FILEGR48/DOC und README/BAS
- FM48/JCL - FILE MANAGER JCL-File
 MENU/DAT - Datenfile fuer das FILE MANAGER PROGRAMM
 NDAMPERSD/TXT - Patch fuer das NEWDOS80-BASIC. wird als Voreinstellung fuer die "-Funktion angenommen

PLANNER/BAS - Einfaches Finanzplanungsprogramm in BASIC
 PLANNER/DOC - Dokumentation zu PLANNER/BAS
 PLATES/EB5 - Demonstrationsfile zu FILMGR48/BAS
 PLATES/RPG - " " " " " "
 Siehe README/BAS und FILMGR48/DOC
 README/BAS - Ein BASIC-Programm, welches eine kurze Einführung in FILMGR48 gibt. Beinhaltet kurze Beschreibungen von : BONDS/EB5, BONDS/RPG, FILMGR48/BAS, FILMGR48/DOC, FILMGR48/FIX, FM48/JCL, MENU/DAT, PLATES/RPG, README/BAS, SPEED/CMD und MIDGET/CRF
 SCRPR1/BAS - BASIC-Programm mit derselben Funktion wie SCRPR1/CMD. Nur M. 3 (arbeitet mit Cassette-Systemen)
 SCRPR1/CMD - Modell III-Utility, schnelle Druckausgabe des Bildschirms auf einigen TANDY Druckern
 SCRPR1/DOC - Dokumentation zu SCRPR1/CMD, SCRPR1/CMD und SCRPR1/BAS
 SCRPR1/SRC - Quellcode fuer SCRPR1/CMD
 SCRPR11/CMD - Modell I-Version von SCRPR1/CMD. Nur Disk
 SPEED/CMD - MODELL 4-Utility, ermöglicht im Modell III-Modus eine Taktfrequenz von 4 MHz. Arbeitet mit FILMGR48/BAS zusammen. Siehe README/BAS und FILMGR48/DOC
 SZAF60/ASM - Quellcode fuer SZAF60/CMD
 SZAF60/CMD - Modell III-Utility, ermöglicht SUPERZAP im 60 x 24 Modus (VIDEO4)
 SZAF60/DOC - Dokumentation zu SZAF60/CMD
 WIDGET/CRF - Demonstrationsfile zu FILMGR48/BAS. Siehe README/BAS und FILMGR48/DOC
 WRDSRCH/BAS - Wirklich funktionierender Kreuzworträtselgenerator
 WRDSRCH/DOC - Dokumentation zu WRDSRCH/DOC

DISK 24 - SETDATE + PROGRAMME AUS NORTHERN BYTES VOL. 6, NO.4-5

BINTNIB/CMD - Verschlüsselt Files im umgekehrten NIBBLE FORMAT (mit NIBBIN/CMD verschlüsselt)
 GOSTMENU/ASM - Quellcode fuer GOSTMENU/CMD
 GOSTMENU/CMD - Programmauswahlmenue in Maschinensprache
 GOSTMENU/DOC - Dokumentation für GOSTMENU/CMD
 HEAPSORT/BAS - HEAPSORT-Demonstration in BASIC. Siehe SORTS/DOC
 MAZE/ASM - Quellcode für MAZE/CIM
 MAZE/BAS - BASIC-Programm mit einem Interface zu MAZE/CIM, zeichnet sehr schnell Labyrinth
 MAZE/CIM - MASCHINENSPRACHE-ROUTINE, Sehr schneller Entwurf von LABYRINTHEN.
 MAZE/DOC - Dokumentation zu MAZE/ASM, MAZE/BAS, MAZE/CIM und MAZE/GEN/BAS
 MAZE/GEN/BAS - Generiert Labyrinth, siehe MAZE/DOC
 NIBBIN/CMD - Entschlüsselt Files, die mit BINTNIB/CMD verschlüsselt worden sind. Syntax: NIBBIN FILENAME
 ROM/ASM - Quellcode fuer ROM/CMD
 ROM/CMD - Speichert das Modell 3-ROM im oberen RAM-Bereich ab, und ermöglicht damit ein schnelleres Booten des Modell 4p
 ROM/DOC - Dokumentation zu ROM/CMD
 SETDATE/CMD - Modell 4-Version von SETDATE (DISK 02), beendet die lästige Datumsabfrage, beim Booten des Betriebssystems
 SETDATE/FIX - Patch fuer TRSDOS6.2, verhindert Löschen des

Datums wenn SYSTEM (DATE=NO) gesetzt ist.

Syntax: DO SETDATE/FIX

SETDATE6/ASM - Quellcode der Modell 4-Version von SETDATE
 SETDATE6/DOC - Dokumentation zu SETDATE/CMD, SETDATE/FIX und SETDATE6/ASM
 SHELSORT/BAS - BASIC-Demo, demonstriert "SHELLSORT-Routine"
 SORTS/DOC - Dokumentation zu HEAPSORT/BAS und SHELSORT/BAS
 SPS/JCL - JCL-File zur Initialisierung von SPSMOD/CMD. Siehe SPSMOD/DOC fuer weitere Informationen
 SPSMOD/CMD - Modifiziert SCRIPSIT/LC, so dass einfache Berechnungen in der Textverarbeitung möglich sind
 SPSMOD/DOC - Dokumentation zu SPS/JCL und SPSMOD/CMD
 SPSMOD/SRC - Quellcode fuer SPSMOD/CMD



Genau wie man mir gesagt hatte, mein Computer hat mich wirklich von allen Zwängen befreit.

HEFT
 15
 September
 1986

1. Vorsitzende

Peter STEVENS
Postfach 56
4688 Dortmund 1
☎ 0231 /593883

2. Vorsitzende

Hartmut OBERMANN
Schwalbacher Straße 6
6289 Heidenrod 1
☎ 06124 /3913

Hardwarekoordinator

Eckehard KUHN
Im Dorf 14
7443 Frickenhausen 1
☎ 07022 /45417

Diskothekar

Klaus-Jürgen MÜHLENBEIN
Am Mönchgarten 28
6948 Weinheim -Ld.
☎ 06201 /55052

Redaktion

Jens NEUEDER
Panoramastraße 21
7178 Michelbach /Bilz
☎ 0791 /42877

Redakteure

Bernd Drohwälder	☒	Klaus Hermann
Dieter Kasper	☒	Eckehard Kuhn
Klaus-J. Mühlbein	☒	Kurt Müller
Jens Neueder	☒	Hartmut Obermann
Richard Rensch	☒	Gerald Schröder
Arnulf Sopp	☒	

sowie Artikel aus: 88 Micro

Bankverbindung des CLUB 88

Postgirokonto	Peter STEVENS
	Sonderkonto CLUB 88
Konto-Nummer	285 491 - 465
Postgiroamt	Dortmund
BLZ	440 100 46

Das INFO erscheint zweimonatlich.

Es erfolgt keine Zensur oder Kontrolle
der jeweiligen eingeschickten Infobeiträge
durch die Redaktion.

Schluß

Hallo Club-88er,

wie in den vorherigen INFO's möchte ich auch hier wieder einige
abschließende Worte schreiben.

Auch diesmal haben wir wieder ein über hundert Seiten starkes
INFO zusammenbekommen, wofür ich mich bei den Mitwirkenden recht
herzlich bedanken möchte. Gleichzeitig auch meinen Dank an
diejenigen, die bei der Fragebogenaktion mitgemacht haben. Leider
hat sich nur knapp über die Hälfte der Mitglieder daran
beteiligt, ich hoffe aber doch, daß die Auswertung etwas
Brauchbares für die Mitglieder enthält. Vielleicht sind doch
einige neue Trends zu entdecken, über die in einer der nächsten
INFO's zu berichten wäre. Vielleicht ist auch ein Thema dabei,
daß den ein oder anderen einmal reizt, den Bleistift zu schwingen
(in die Tasten zu hauen). Ich, als Redakteur, würde mich sehr
darüber freuen.

Als weiteres liegt mir natürlich unser ECB-Bus-Projekt am
Herzen. Nachdem nun die Grundeinheit fertig zum Ausliefern ist
und sicher bald von den Beteiligten zusammengebaut sein wird,
geht es nun an den "Harten Kern" der Sache -- den Computer --.
Aus diesem Grunde haben wir in nächster Zeit wieder ein kleines
"Hardwaretreffen" vereinbart. Dort werden wir dann die noch
notigen Einzelheiten absprechen. Auf jeden Fall geht es mit der
"Bastelei" im nächsten INFO wieder weiter.

Einstweilen wünscht Euch viel Spaß an der neuesten INFO

Euer

J. Neueder

schon eine Zahl steht! Und eine Speicher-Adresse ist ein reiner Zahlenwert.
 In diesem Fall wird das Register HL in Klammern gesetzt, dann weiß ASS bzw. die CPU: Nicht das Register HL, sondern die Speicherstelle, die in HL steht, soll mit dem Zahlenwert BFh gefüllt werden!
 Steht nicht ein Register-Buchstabe, sondern eine Zahl in Klammern, so ist damit eine "Hausnummer", d.h. eine Speicher-Adresse gemeint.
 Soll die Speicherstelle 3C00h mit BFh geladen werden (damit sie weiß wird), so müssen wir das also dem Register HL (in dem die Adresse 3C00h bereits steht) mitteilen, indem wir es einklammern.
 Allgemein gilt immer, wenn eine Klammer auftritt:
"Lade die Speicherstelle, die du in Sinn hast, mit dem Wert sowieso!"
 ("Sinn" = Klammer.)

In unserem Falle ist der "Wert" ein Code, nämlich der ASCII-Code "BFh".
 Und noch eine Falle hätte Herr ASS bei diesem Versuchs-Befehl vorgefunden: er hätte "BFh" für einen String, eben für die Buchstaben-Folge BFh gehalten. Wir meinten aber eine Hex-Zahl. Die akzeptiert er als solche nur, wenn sie mit einer Ziffer beginnt. Doch woher nehmen? Code 191 (Dez) fängt in HEX-Darstellung nunmal mit dem Buchstaben B an! "Was tun?" sprach ZEUS hier nicht; sondern: "Eine Null ist immer gut: sie schadet niemand - wenn was dahinter steckt!!" (Wie recht er hat...)
 Jetzt ist also alles klar (??):

LD (HL), 0FFh

ist richtig. Nicht HL wird mit BFh geladen, sondern (HL), das heißt:
die Speicherzelle, deren Adresse in HL steht,
 und das ist ja (siehe 1. Befehl) die erste Bildschirm-Position!
 Wir dürfen uns den Schweiß abwischen, den ZEUS vor unseren Erfolg gesetzt.
 Denn nun arbeitet ASS für uns. Es bedarf nur noch eines Fußtrittes, unentschlossen wie ASS ist - aber dann rast er!
 Der Tritt, der den Ball (Fußball 1986!) zum Sausen bringt:
 Laß den Inhalt der Quelle HL ins Ziel DE, kurz "LD";
 Inkrementiere (erhöhe) beide um 1, kurz "I";
 vermindere gleichzeitig den Zähler BC um 1;
 Repeat (wiederhole) diesen Vorgang, kurz "R"
 und zwar solange, bis in BC eine Null steht.

Fassen wir diese Abkürzungen zusammen, so erhalten wir das schlaue Kürzel

LDIR

das dies alles alleine zuwege bringt.

Jetzt ist der Bildschirm zwar nicht sauber - aber rein. So schnell schafft das kein Persil.

Was steht denn da in der 4. Spalte? Sieht aus wie Aushänge-Schilder. Sind es auch, wie sie vor wichtigen Häusern hängen (z.B. bei Friseuren, Bäckern und - Ähnlichem).

Gelegentlich muß man der CPU sagen, wo sie mit ihrer Arbeit anfangen und wo aufhören soll. Sie versteht "START" und "END". Nicht immer ist "START" identisch mit "ORG" (= origine = Ursprung, eigentlich also auch gleich Anfang; hier aber Anfang des Codes im Speicher, START hingegen ist das Startloch für die CPU, wie RUN für BASIC).

Übrigens, der Ordnung halber sei's gesagt, obschon der Club80 wohl namensgerecht nur den Mikroprozessor 280 oder dessen Abkömmlinge betreibt: Alles hier Gesagte betrifft eben diesen; ich weiß nicht, ob alles z.B. für den "6502" (hochwohlloblicher COMMODE) oder andere auch zutrifft. Wie gesagt: "OHNE GEWAHR!"

*Dies ist die
 "Ersatzseite"!*

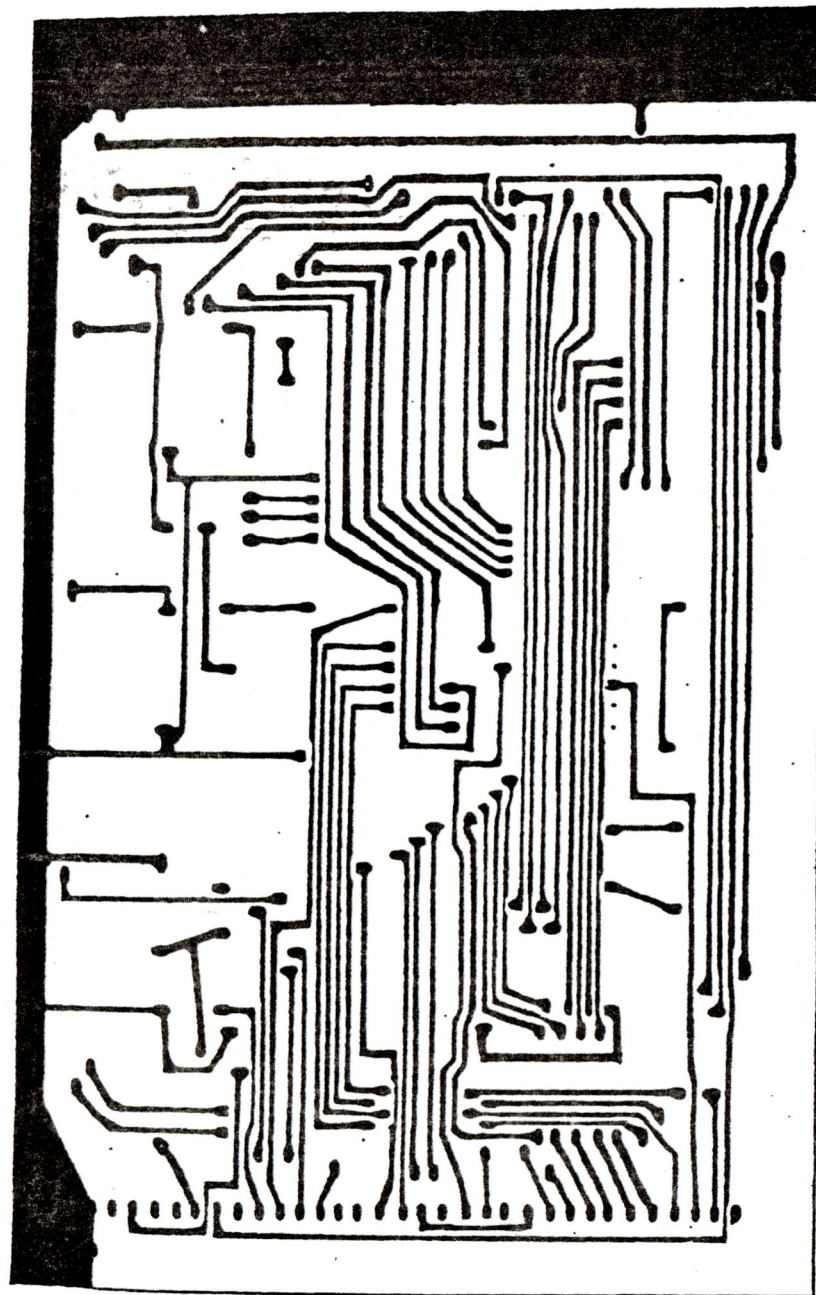
für S. 12 INFO 14

*(Nach Erscheinen der nächsten
 Adressenliste kann diese Seite dort
 eingeklebt werden)*

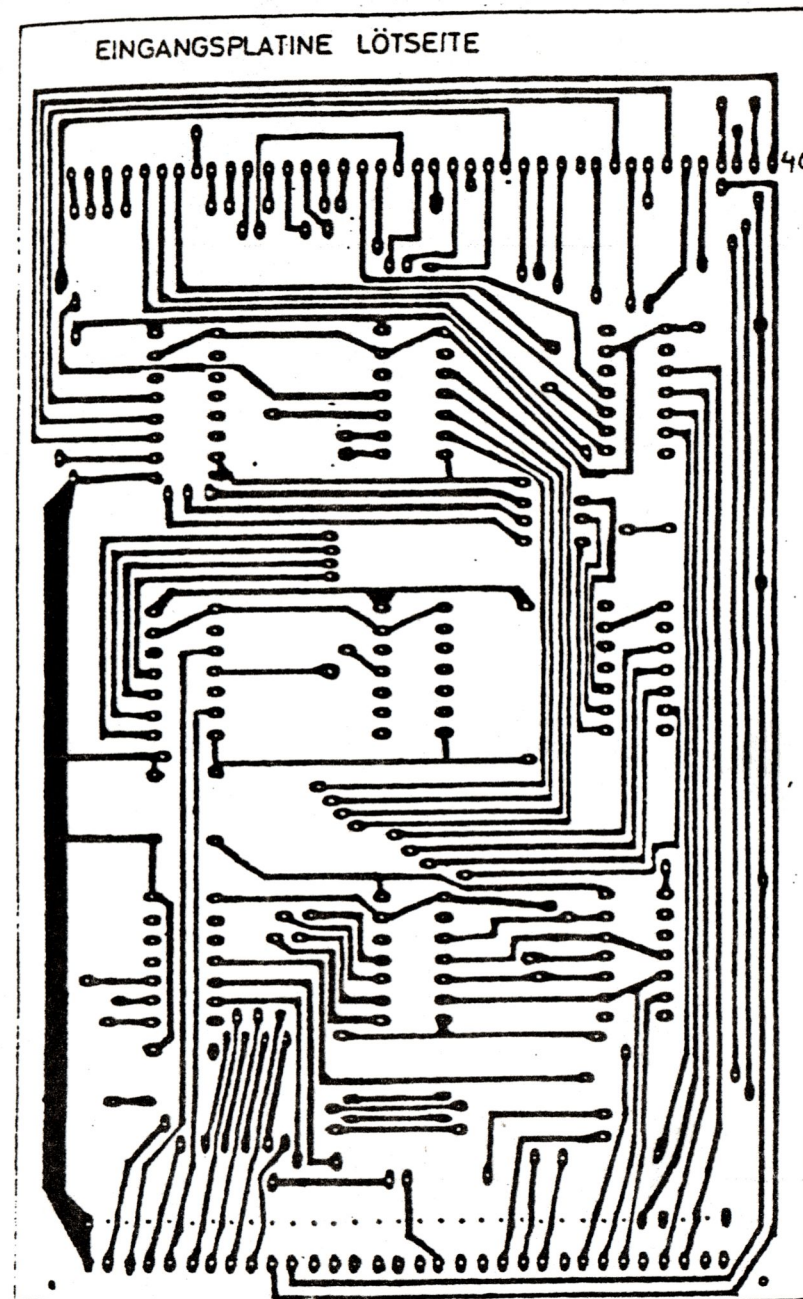
Name	Vorname	Straße	PLZ	Stadt	Telefon	privat	// geschäftlich
Alber	Herbert	Alemannenstraße 20	7732	Niedereschach	07721	/7182	// -
Albers	Herbert	Zum Düwelshöpen 14	2117	Wistedt	04182	/8799	// -
Beckhausen	Wolfgang	Vuerfelser-Kaule 30	5060	Bergisch-Gladbach 1	02204	/62781	//
Bernhardt	Helmut	Hafenstraße 7	2305	Heikendorf	0431	/241987	// 0431 /74047
Buskowiak	Thomas	Eschersheimer Landstr. 257	6000	Frankfurt 1	069	/5601621	//
Böcker	Dieter	Lehmweg 4	2930	Varrel 1	04451	/7640	//
Böckling	Ulrich	Am Sonnenhang 11	5414	Vallendar	0261	/69522	// 02631 /895168
Dreyer	Gerald	Am Speiergarten 8	6200	Wiesbaden-Bierstadt	06121	/508218	// -
Drowälder	Bernd	Hügel 1	4441	Wettringen	05233	/4320	// 02557 /1236
Emmrich	Helmut	Waldstraße 5	6682	Ottweiler	06824	/4114	// -
Frink	Thomas	An der Schleifmühle 2	5042	Erftstadt	02235	/76255	//
Grajewski	Werner	Zedernweg 29	4220	Dinslaken	02134	/54573	//
Hartel	Eberhard	Neenstetter Straße 20	7901	Breitingen	07340	/7281	//
Heile	Heinz-Dieter	Blankensteiner Straße 13	4320	Hattingen	02324	/28458	//
Held	Manfred	Stirnerstraße 22	8835	Pleinfeld	09144	/6563	// 0911 /2195245
Hermann	Klaus	Gartenstr. 22	7401	Pliezhausen	07127	/70024	//
Hill	Peter	Eckstraße 36	6750	Kaiserslautern 31	0631	/54782	//
Hummel	Anton	Schubertstr. 2	7612	Haslach	07832	/8289	//
Jablotschkin	Rainer	Thiekamp 29	4780	Lippstadt 8	02948	/542	// 02921 /70431
Kasper	Dieter	Zeppelinstr. 9	8952	Marktoberdorf	08342	/1630	// 089 /522071
Konrad	Josef	Anzengruberstraße 35	8038	Gröbenzell	08142	/8494	// -
Kopschina	Peter	Strandallee 138	2409	Scharbeutz	-		// -
Kuhn	Eckehard	Im Dorf 14	7443	Frickenhausen 1	07022	/45417	// -
May	Holger	Marienstr. 9	5768	Sundern 2	02935	/1668	// -
Misioch	Waldemar	Adenauerring 25	8505	Röthenbach a. d. Pegnitz	0911	/506051	// 0911 /668151
Mühlenbein	Klaus-Jürgen	Am Mönchgarten 28	6940	Weinheim -Lützelachsen	06201	/55052	// -
Müller	Kurt	Soltaustraße 24a	2050	Hamburg 80	040	/7246083	// 04103 /702662
Neueder	Jens	Panoramastraße 21	7178	Michelbach /Bilz	0791	/42877	// 0791 /44-667
Obermann	Hartmut	Schwalbacher Str. 6	6209	Heidenrod 1	06124	/3913	// 06124 /2051215
Perschbach	Patrick	Waldstr. 52	5000	Koeln 91	0221	/872118	//
Piller	Walter	Rohnenstraße 8	CH-8835	Feusisberg	01	/7847418	//
Raggan	Hans	Backnanger Weg 36	7146	Tamm	07141	/603611	// 0711 /2630473
Rank	Heinrich	Frühlingstraße 2	8000	Fürstenfeldbruck	08141	/3791	//
Rensch	Richard	Bahnhofstraße 100 (Postf. 226)	7128	Lauffen am Neckar	07133	/4167	// 07133 /8415
Retzlaff	Bernd	Kleiner Sand 98	2082	Uetersen	04122	/43551	// 04103 /7025310
Rychlik	Andreas	Königsberger Allee 120	4100	Duisburg 1	0203	/331383	// 0203 /331383
Schneider	Manfred	Rheinkasseler Weg 11	5000	Köln 71	0221	/707044	//
Schrewe	Christian	Fliederweg 32	4000	Düsseldorf 31	0203	/740897	//
Schröder	Gerald	Am Schützenplatz 14	2105	Seevetal 1	04105	/2602	// -
Schäfer	Walter	Rathausstr. 4	8160	Miesbach	08025	/1631	// 08025 /41247
Smerling	Frank	Tangstedter Str. 5	2030	Pinneberg	04101	/207284	//
Sopp	Arnulf	Wakenitzstr. 8	2400	Lübeck 1	0451	/791926	// -
Spieß	Peter	Trugenhofenerstraße 27	8859	Rennertshofen 1	08434	/454	// 08431 /7041684
Stephan	Hans-Martin	Am Glasesch 9a (Postf. 1207)	4506	Hagen a.TW.	05401	/99585	// 05401 /90036
Stevens	Peter	Postfach 56	4600	Dortmund 1	0231	/593883	// 0231 /593883
Trapp	Harald	Kranichstr. 46	4270	Dorsten 1	02362	/42497	// 02362 /23127
Troesch	Eberhard	Altenessener Str. 414	4300	Essen 12	0201	/342324	//
Volz	Oliver	Dusestraße 13	7000	Stuttgart 80	0711	/731285	//
Wagner	Günther	Gartenstraße 4	8201	Neubeuern	08035	/3361	// -
Weiß	Dieter	Bürglestraße 3	7209	Wehingen	07426	/7194	//
Wucherer	Jürgen	Menzelstraße 1	7750	Konstanz	07531	/29145	// -
Zwickel	Walter	Lengfelden 123	A- 5101	Bergheim	0043662/51130		// -

Stand: September 1986

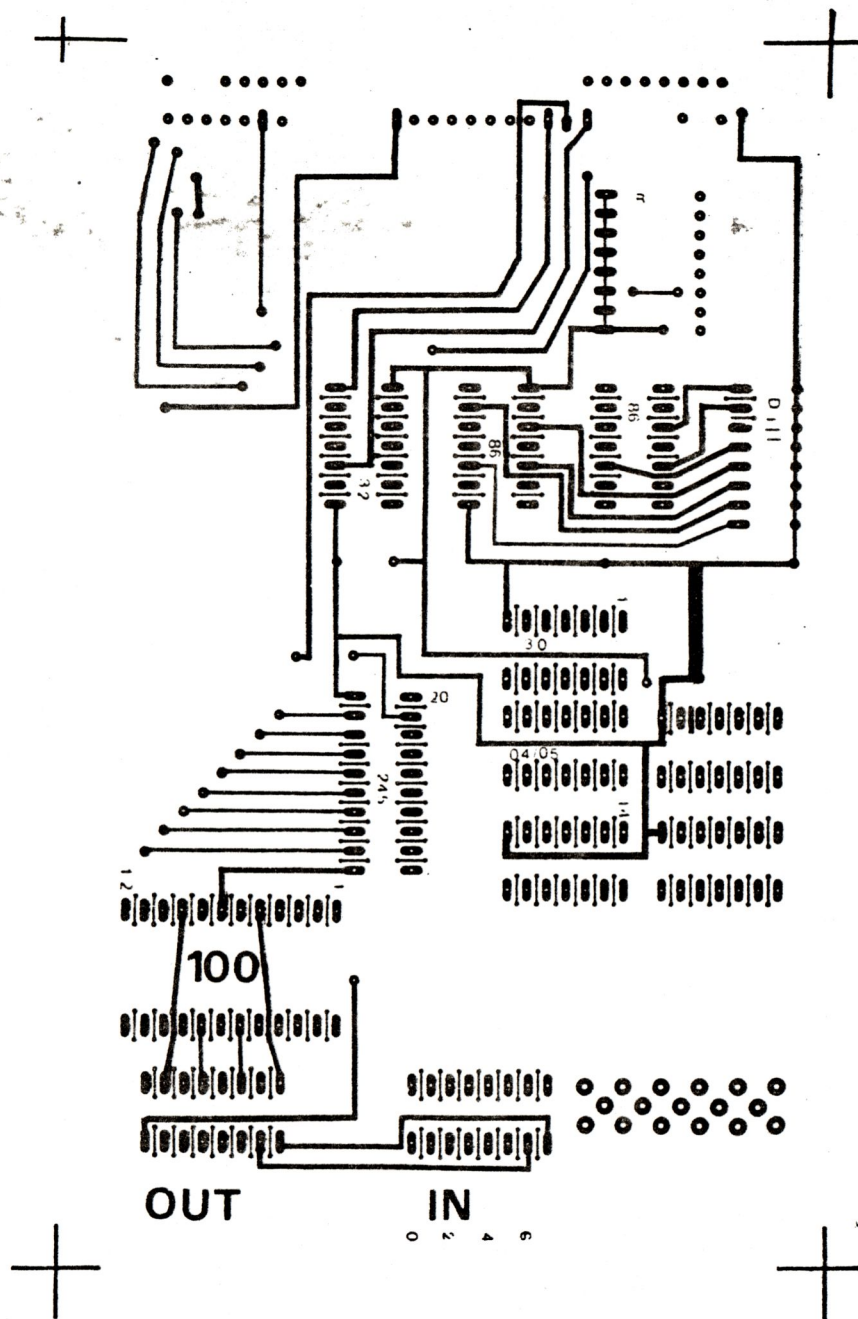
Bitte überprüft Eure Daten auf Richtigkeit
und teilt mir Unregelmäßigkeiten mit.
Die Redaktion



Platinenlayout Eingangskarte
Bestückungsseite

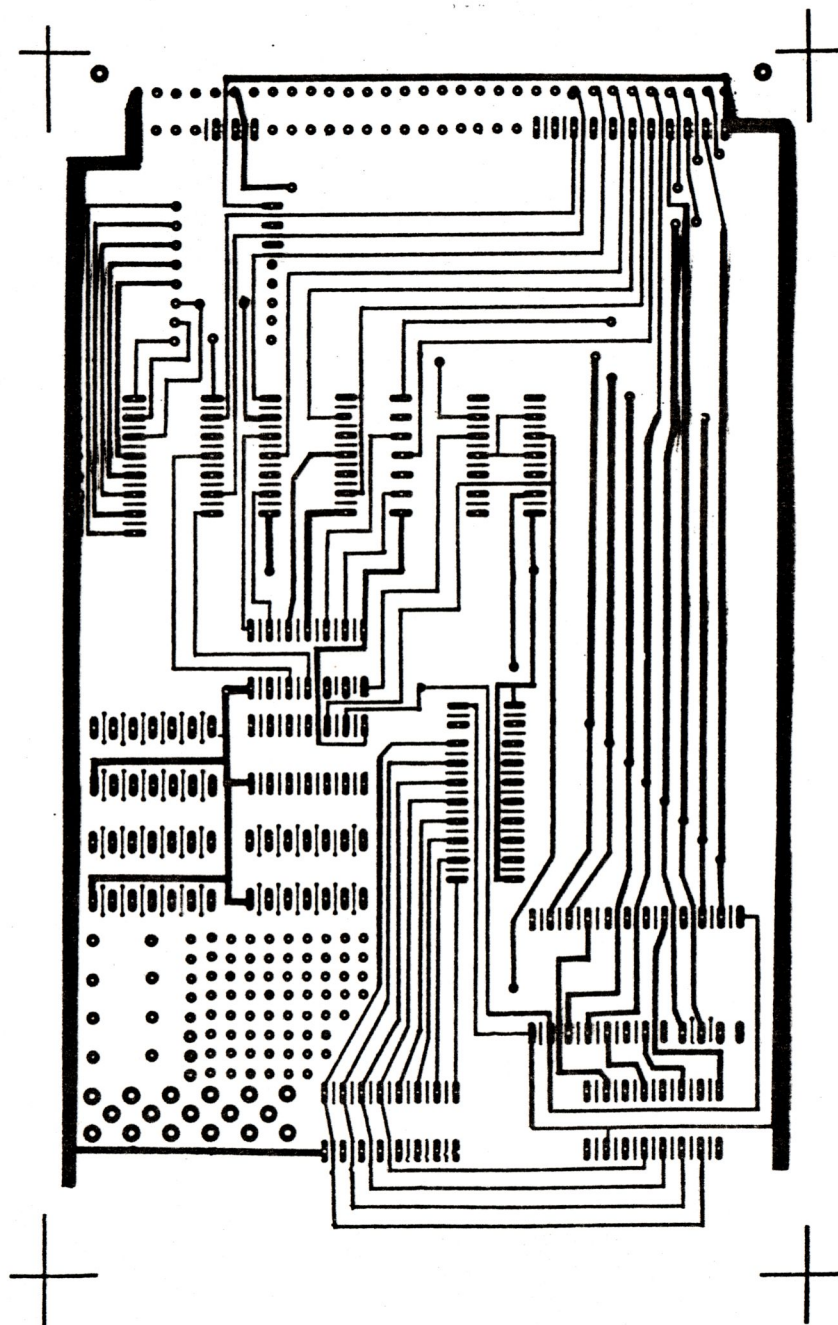


Platinenlayout Eingangskarte
Lötseite



Platinenlayout (ZK01)

Bestückungsseite



Platinenlayout (ZK01)

Lötseite