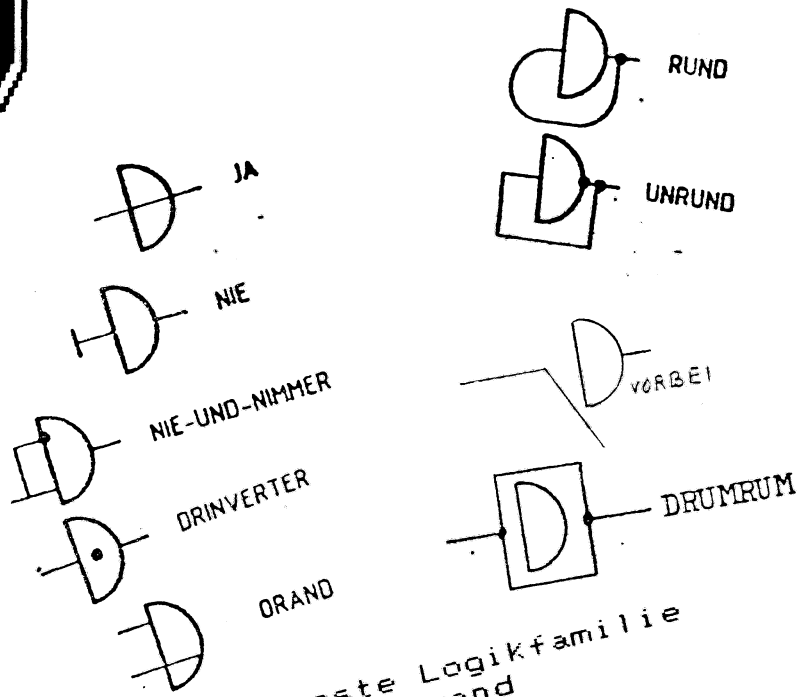
CLUB 80

Clubinfo der

CANDY - GENIE - und KOMTEK - ANWENDER



17. AUSGABE

KONTAKTADRESSE : CLUB 80 / HARTMUT OBERMANN / SCHWALBACHER STRAßE 6 / 6209 HEIDENROD 1
TEL.: 06124 / 3913

Inhaltsverzeichnis

Seite:

Clubinternes

Software

Hardware

Sonstiges

Programmbibliothek

liegt als Sonder-Heft bei

Die letzten Seiten

Neues vom Vorstand

Diesmal habe ich zwei Nachrichten für euch, eine gute und eine schlechte!

Zunächst die gute Nachricht:
Dank der intensiven Bemühungen von Jürgen Wucherer (bei dem ich mich herzlich bedanken möchte) haben wir wieder ein preisgünstiges und schönes Hotel für unser Clubtreffen gefunden. Näheres dazu siehe bei "Clubtreffen"!

Die weniger gute Nachricht lautet:
nachdem ich erst vor kurzem die Geschäftsführung des Clubs von Peter Stevens übernehmen musste, werde ich sie wohl genauso schnell wieder abgeben müssen. Der Grund dafür liegt bei meinem Dienstherren, dem BMVg (für Leute denen das nichts sagt: Manfred Wörner). Man wird mich (ich hoffe es kommt nichts dazwischen) auf eigenen Wunsch ab 05.01.1986 auf einen neuen Dienstposten versetzen. Dieser Dienstposten zwingt mich leider dazu, für ein paar Monate eine sog. Wochenendehe zu führen. Ich werde also die Woche über unterwegs und nur an Wochenenden, und längst nicht an allen, daheim sein. Ich hoffe trotzdem, mit Hilfe meiner besseren Hälfte, die Clubarbeit mindestens bis zum Treffen im März korrekt und ohne übergroße Verzögerungen erledigen zu können.
Damit sind wir wieder genau da angelangt, wo wir auch im letzten Jahr standen:

Der Club 80 sucht einen Vorsitzenden!

Hier einmal eine Aufzählung der Arbeiten, die ein Vorsitzender im Laufe eines Jahres zu erledigen hat:
- Betreuung der Clubmitglieder und vor allem der Neuzugänge
- Anregen und Koordinierung von Aktivitäten (z.B. Clubtreffen)
- Verwaltung der Clubkasse
- Vorworte zu den Infos
- Verbindung zu anderen Clubs knüpfen/halten

Wieviel Zeit für diese Arbeiten benötigt wird, läßt sich nur schwer sagen, denn je nach Engagement schwankt der Arbeitsaufwand stark.

Sollte sich irgend jemand mehr für die Arbeit des Vorsitzenden interessieren und mehr Informationen benötigen, bin ich gerne bereit, Auskunft zu erteilen. Ich befürchte aber, daß sich wohl auch diesmal (wie schon in den letzten Jahren) niemand für diesen so wichtigen Posten bewerben wird und der Club wieder einmal nur mit Mühen an einer Auflösung vorbeikommt. Die gesamte Suche nach Vorständen bei den beiden Clubtreffen erinnert mich immer mehr an einen Witz über meinen Berufsstand.

Frage: Was tut ein Soldat, wenn es heißt "Freiwillige einen Schritt vortreten!"?
Antwort: Er tritt einen Schritt zurück, um die Freiwilligen vorbeizulassen!!!

Damit genug für heute über dieses traurige Thema und hin zu erfreulicheren Dingen.

Clubtreffen '87

Wie schon im letzten Info erwähnt, findet das Clubtreffen 1987 in der Zeit von Freitag dem 13.03.87 (hoffentlich ist niemand abergläubisch) und Sonntag dem 15.03.87 statt. Wie weiter oben schon erwähnt, haben wir auch diesmal, dank der schnellen und aktiven Hilfe unseres Freundes Jürgen Wucherer, ein schönes, gut gelegenes und preisgünstiges Hotel für das Treffen gefunden.

Der Ort des Treffens ist diesmal Alsfeld, ein direkt an der Autobahn A5/E4 (zwischen Giessen und Bad Hersfeld) gelegenes, romantisches Städtchen mit historischem Stadtkern. Alsfeld ist sowohl für die "Nordlichter" als auch für die "Südländer" gleich gut zu erreichen und erspart durch seine Autobahnnähe mühsehlige Fahrten auf unbekannten Landstraßen (ist also besser gelegen als die letzten beiden Tagungsorte).

Das Hotel Klingelhöffer, welches uns diesmal Unterkunft und Tagungsräume zur Verfügung stellt, liegt mitten in der Stadt und fällt vor allem durch die Fachwerkfassade und die rustikalgemütliche Innenausstattung auf. Der Preis für übernachtung mit Frühstück beträgt pro Person in einem Einzelzimmer 35, in einem Doppelzimmer 30 Mark. Der Tagungsraum wird uns kostenlos zur Verfügung gestellt. Natürlich kann man bei Klingelhöffers auch preisgünstig und sehr gut Essen. Hier die genaue Adresse:

Hotel Klingelhöffer
Hersfelder Straße 47/48
6320 Alsfeld (Hessen)
Tel.: 06631 / 2073.

Eine Tagesordnung für das Treffen steht noch nicht fest, ich verspreche euch aber, daß es weniger Administratives als bei den letzten Treffen geben wird. Der Freitag wird sowieso ohne festes Programm ablaufen, der offizielle Teil beginnt erst Samstags nachmittags und der Sonntag gehört wieder ganz dem Erfahrungstausch der Mitglieder. Die genaue Themenaufstellung für den offiziellen Teil des Treffens geht den Teilnehmern noch rechtzeitig zu.

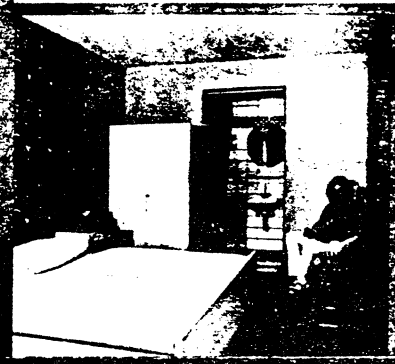
Mit dem Hotel wurde schriftlich eine vorläufige Zimmerreservierung für 25 Einzel- und 3 Doppelzimmer vereinbart. Es ist dringend nötig, bis spätestens Ende Februar eine genaue Reservierung vorzunehmen. Aus diesem Grunde muß das dem Info beiliegende Anmeldeformular bis spätestens 20. Februar bei mir eingegangen sein. Wer sich bis zu diesem Termin nicht darüber im klaren ist, ob er kommen kann oder nicht, muß für seine Zimmerreservierung selbst sorgen.

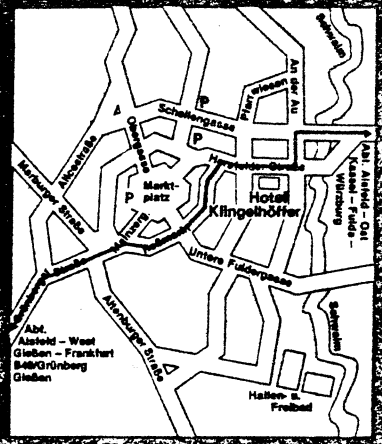
Ich hoffe, daß auch diesmal wieder viele Mitglieder die Möglichkeit nutzen, sich mit Gleichgesinnten zu treffen, Erfahrungen und Informationen auszutauschen und gute alte Bekannte und Freunde wiederzusehen. Vielleicht kann man schon einige Ergebnisse unserer ECB-Bastler sehen und sicher bekommt man neueste Informationen über die ständig wachsende Club-Programmbibliothek. Eventuell kann ich auch den einen oder anderen Spezialisten dazu überreden, eine kleine, verständliche Einführung in sein Gebiet zu geben. Alles in allem weiß ich jetzt schon, daß sich der Besuch des Treffens unter Garantie lohnt und freue mich auf jeden der kommt!

Damit Schluß für heute. Viele Systemcrashs wünscht euch Euer

Hartmut Ebermann

Die hier abgebildeten Bungalowzimmer sind ebenerdig erreichbar und inmitten eines Gartens gelegen.





6320 Alsfeld (Hessen)
Telefon (06631) 2073
Hersfelder Straße 47/48
Abfahrt: BAB Alsfeld-Ost

Von Süden kommend (Frankfurt) der Deutschen Autobahn Abfahrt Alsfeld-Ost, Richtung Alsfeld – Gießen
Von Norden kommend (Kassel) Abfahrt Alsfeld-West
Aus Richtung Frankfurt – Gießen Abfahrt Alsfeld-West
Aus Richtung Kassel – Fulda – Würzburg Abfahrt Alsfeld-Ost

Herzlich Willkommen

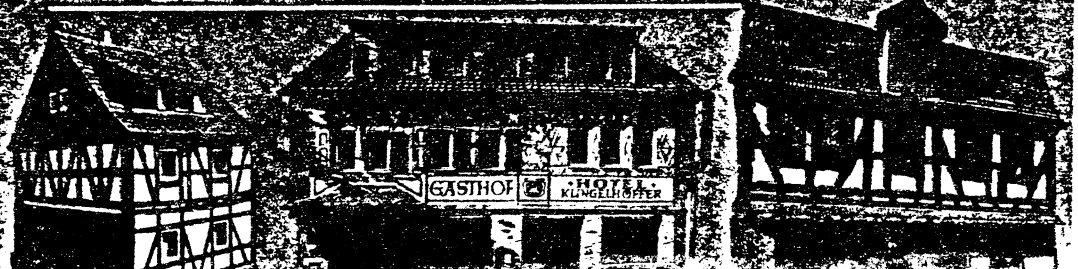
Leider ist der Prospekt
über unseren nächsten Tagungsort
nicht besser kopierbar.
Ich wollte es Euch aber dennoch nicht vorenthalten.

Die Redaktion

WIR STELLEN VOR...

Hotel Klingelhöffer

MIT SEINEN GASTHÄUSERN



* * Eine Antwort * *

an Dieter Kasper

(Zu Peter Spieß' Programmier-Wettbewerb)

Jeder Beitrag ist willkommen; ein kritischer ganz besonders.
Also auch *Dieter's* in Heft 16, Seite 5.
Ich will auch keine Gegenkritik zu seiner Stellungnahme zu *Peters* Wettspiel-Angebot üben. Das wäre *Peters* Sache.

> Apropos: Auch ich könnte Peter z.B. vorwerfen, daß seine Aufgabe seine Fachkenntnisse überstieg, so daß ich aus dem Rennen von vornherein ausschied! (Öfterer liefen aber alle unter "ferner liefen")... Doch das verkneife ich mir.

Ich möchte *DIETER* nur auf einen - wie ich glaube - Irrtum hinweisen, wenn er sagt, "daß die Übersicht enorm darunter leidet", wenn man beim Programmieren möglichst rationell (sparsam) auf die Zeilenzahl achtet, diese also nach Möglichkeit einschränkt. Nach einer IF-Bedingung geht das in der Tat meistens nicht, wenn man nicht mit lauter "ELSEs" um sich schlagen will (eine Else am Halse genügt im allgemeinen...)
Aber Zeilennummern kosten Speicher. Das ist zwar bei BASIC i.a. irrelevant, aber es stellt ein ökonomisches Prinzip dar.
Ich strebe möglichste Verdichtung (nicht nur Dichtung) an. Eine Zeile kann 240 Bytes aufnehmen und mehrere, durch Doppelpunkte getrennte Anweisungen enthalten. Nun, das weiß jeder Programmierer. Auch, daß das sehr "unübersichtlich" ist.

ABER:
1) Man kann erst mal gemütlich ("übersichtlich") programmieren und dann, wenn alles läuft, die einzelnen Zeilen mit einem Bildschirm-Editor (z.B. SEDIT o.a.) maximal zusammenfügen; und
2) kann man bei der Veröffentlichung (oder späteren eigenen Betrachtung) das Listing mit NAME ausdrucken.
"NAME" trennt die statements wieder und ersetzt die Doppelpunkte durch Line Feeds - s. mein kleines Listing in Heft 16, Seite 23, zur FIBONACCI-Folge! "NAME" spaltet nicht nur die langen Zeilen wieder auf, sondern macht auch FOR...NEXT--Schleifen sowie die häßliche IF...THEN...ELSE-Formel (die man sich mit logischen Operationen ersparen kann, siehe Zeile 60 ab "ON") sehr übersichtlich! - Findest Du nicht auch, Dieter? (Zugegeben: Mein Beispiel ist zu schlicht - aber nicht ganz schlecht...)

Last not least: Auch ich würde mich sehr "freuen, wenn das nächste Problem in dieser (von Dir gewünschten) Form gestellt würde" - wie Du am Schluß schreibst. Nur: ich frage mich, an wen sich diese Aufforderung richtet? Nicht auch an Dich selbst? Ich hätte mich (und sicher viele Kollegen auch) also noch mehr gefreut, wenn Du bei dieser Gelegenheit mit Deinem eigenen Vorschlag einer Problemstellung auch gleich den Anfang gemacht hättest! * Denn auch von uns ist niemand dazu "verpflichtet".

* * * Grüß' Dich, *Dieter!* * * * Xaver Mayr

PS: Auf *Gerald Schröders* "Flugreichweiten-Problem" hat sich m.W. seinerzeit auch niemand gerührt! Hierüber siehe a.a.O.

Gesucht: Grafik-Standard

Der gute alte TRS80 hatte eine niedliche Klötzchengrafik, mit der sich kaum ein Anwender nach Ablauf der Garantiezeit zufrieden gab. Nur leider gibt es ein Problem dabei: die eingebauten Grafikkarten, sind nicht genormt, werden also unterschiedlich angesteuert und besitzen keine einheitliche Auflösung. Der heimliche Standard dürfte die 384x192-Grafik sein, aber genauso gibt es 640x400 (Mod. IV), 512x512 (IIIs) und 480x192 (IIs). Dazu kommen noch Eigenkonstruktionen oder Anbauten über den ECB-Bus.
Nun gibt es sicherlich für jede Karte mehr oder weniger gute Grafik-Programme, aber nur die Bilder lassen sich (mit einiger Umformung) untereinander austauschen. Warum soll das bei den Programmen nicht auch so sein? Bestes Beispiel ist der IBM-PC, bei dem es keine wirkliche Standard-Karte gibt (Color, Hercules, EGA). Trotzdem bedient jedes bessere Grafik-Programm alle Karten!
Es ist wohl unmöglich, die Hersteller bzw. Software-Firmen im TRS80-Bereich auf eine Norm einzuschwören, aber club-intern sollten wir dies meiner Meinung nach versuchen.
Deshalb folgender Vorschlag: ein paar Grafik-/Assembler-Freaks tun sich zusammen und knobeln einen CP/M-ähnlichen Standard aus. Das soll heißen, daß wir uns auf einen Einsprungpunkt einigen, von dem jedes Programm Gebrauch machen kann. In den Registern sind beim Einsprung genormte Werte zu finden, nach denen ein individueller Grafik-Treiber die Grafik bedient. Als Anfang wären für jede Grafik-Karte folgende Funktionen zu definieren:
1) Abfrage max. X-Koord., max. Y-Koord.
2) Punkt X/Y setzen (SET)
3) Punkt X/Y löschen (RESET)
4) Punkt X/Y abfragen (POINT)

Aus diesen Funktionen lassen sich m.E. alle anderen Grafik-Funktionen zusammenbauen. Alle höheren Funktionen wie Linien und Kreise sind dann allgemeingültig. Natürlich wären auch Basic-Treiber möglich, aber diese brauchten nur auf einem Computer gebaut zu werden und würden dann mit jeder anderen Karte laufen.
Der Treiber, der die obengenannten Funktionen realisiert, müßte natürlich auch seinen Platz haben. Wahrscheinlich ist ein Mindest-Raum anzusetzen, in dem sich für jede Karte diese Funktionen realisieren lassen.

Dies ist als ein Vorschlag zu verstehen, der sich an alle Leser richtet. Es bringt nichts, wenn ich mich allein hinsetze und einen Standard für Newdos und evtl. CP/M ausarbeite, den dann niemand annimmt. Ich möchte deshalb die Interessierten bitten, mir zumindest ihre Meinung zu schreiben. Außerdem habe ich allein natürlich keine Chance, etwas Vernünftiges auf die Beine zu stellen und brauche deshalb Unterstützung von Programmierer-Seite. Da ich sowieso nur eine Grafik-Karte kenne und die nur einmal im Club vertreten ist, wäre meine Arbeit sinnlos. Ich hoffe, daß einige Freaks sich mit mir zusammenraufen und diesen Standard etablieren, damit unsere Rechner auch eine Grafik-Zukunft haben. Ich habe auch keinerlei Ambitionen, bei diesem Projekt die Führung zu übernehmen, sondern wäre froh, wenn sich berufenere Hände als meine dazu finden würden.

Ich warte auf Eure Reaktion!

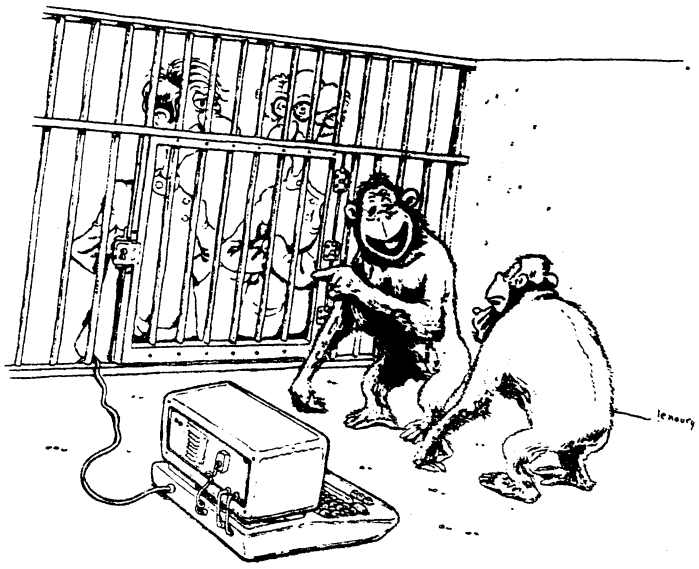
Gerald Schröder

-- Termine -- Termine -- Termine --

Nächster Redaktionsschluß .....28. Februar 1987
Jahreshauptversammlung 1987 .....13. - 15. März 1987
Norddeutsches Regionaltreffen .....25. + 26. April 1987

-- Messen '86 --

Hobby-tronik	Dortmund	18. - 22. Februar 1987
CeBIT	Hannover	4. - 11. März 1987
Internationale Computer-Ausstellung	Köln	11. - 14. Juni 1987
HOBBY ELEKTRONIK 87	Stuttgart	5. - 8. November 1987
PRODUCTRONICA	München	10. - 14. November 1987

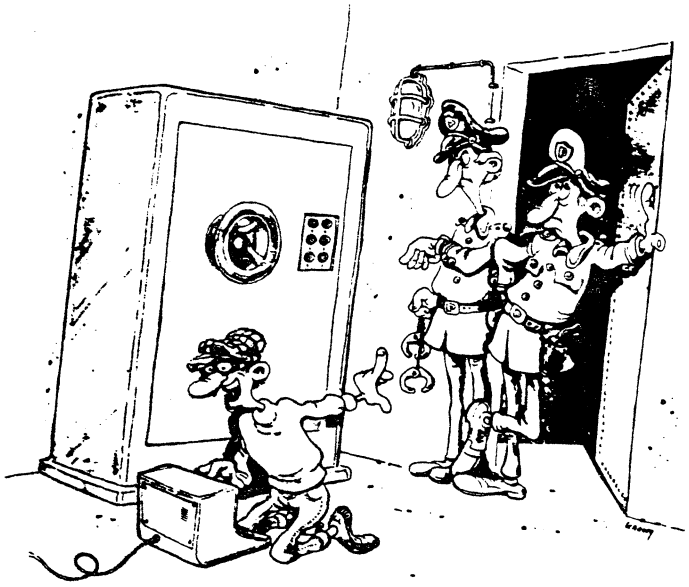


Laß die nicht sehen, was du damit machen kannst, sonst sitzt du nachher auch acht Stunden im Büro.

Amerikanisch: Having a ball.
Einen Ball haben.
Sich köstlich amüsieren.

Go fly a kite.
Geh, laß einen Drachen fliegen.
Mach, daß du wegkommst.

Erwin Knop, Big Bear City, California



Mach weiter, Kalle... wir ham noch genau sieben Minuten bis die Bullen hier sind.

Persisch: Dare baq-e ssabs be kasi nišān dadan.
Jemandem die Tür zum grünen Garten zeigen.
Jemanden anlocken, um ihn zu übervorteilen.

Wie Phoenix aus der Asche oder
Vier Computer in einem Schrotthaufen

Die Beilage von Arnulf und Helmut zum letzten Info hat Euch sicherlich geschockt, denn schließlich verfügt nicht jedermann über einen IIIs, den er aufmotzen kann. Mir fehlt zwar das Kleingeld für einen IIIs, aber deshalb gönne ich den IIIs-Besitzern diesen Triumpf noch lange nicht. Deshalb habe ich kurzerhand Arnulfs Idee abgekupfert und auf den schon in der Überschrift erwähnten TRS80 angewandt. Wer sich jetzt extra einen IIIs gekauft hat, um mit Arnulf gleich ziehen zu können, ist selbst schuld.

Wie sieht das in der Praxis aus? Ganz einfach, Ihr belagert einfach den schon mehrmals erwähnten Helmut, bis er Euch einen Banker einbaut. Dann tippt Ihr das folgende Programm ab oder laßt es Euch vom nächsten Diskothekar schicken. Wenn Ihr dann <Shift><Down-Arrow><Ziffer> betätigt, wobei <Ziffer>:= 0 bis 3, könnt Ihr zwischen vier Computern hin- und herschalten. In jedem Computer kann ein Programm laufen, solange dieses Programm
1. die Interrupts angeschaltet läßt und
2. keine fiesen Nebeneinbauten benutzt.

Punkt 2 bezieht sich auf Grafikkarten o.ä. Damit wir uns richtig verstehen: benutzen darf das Programm sie schon, solange beim Sprung zu einem anderen Computer die Sache wieder ausgeschaltet ist. Der weiß nämlich dann überhaupt nichts davon.

Gerettet wird der Speicher ab 3000h aufwärts. Dadrunter sollte ein festes ROM sein, das gefälligst nicht geändert wird.

Einige interne Notierungen:
- die Umschaltung ist in die normale Interrupt-Routine des Gdos/Newdos-SYS0 eingebaut
- alle Routinen liegen in Bank 1
- die Banks 1,2 und 3 enthalten die "unteren" Hälften der Computer 1-3, jeweils von B000-FFFF
- die Banks 0 und 4-6 enthalten die "oberen" Hälften der Computer 0-3
- die "untere" Hälfte von Computer 0 wird jeweils mit der des momentan aktiven Computers getauscht
- es werden jeweils nur die "unteren" Hälften vertauscht (weil das Common nun mal eine "untere" Bank ist), während die "oberen" Hälften nur umgeschaltet werden müssen
- d.h.: wenn von Computer 1 auf 3 umgeschaltet wird, wird zuerst Computer 0 selektiert, bevor Computer 3 endgültig ans Rohr kommt
- deshalb spielt sich auf dem Bildschirm bei einer solchen Umschaltung kurz ein Wirrwarr ab, das aber nur mühsam zu umgehen wäre
- das Vertauschen läuft folgendermaßen ab:
  (von Bank 1 aus) 7E00-7FFF nach Bank 1 retten
  Tausch-Routinen nach 7E00 (ins Common) übertragen
  Tausch-Routinen starten
  gerettetes 7E00 ff. mit dem aus sel. Bank tauschen
  Common 3000-7DFF mit sel. Bank tauschen
  (dabei Buffer 7F00-7FFF benutzt)
  zurück nach Bank 1
  geretteten Bereich 7E00 ff. ins Common
  starten des Computers
- um unerwünschte Nebeneffekte (wie Druckerausgabe) auszuschließen, wird der I/O-Bereich 3700-37FF nicht gerettet
- das Programm ist lange nicht so elegant wie Arnulfs Lösung, aber es tut seinen Dienst

So das war's schon, besorgt Euch 256Kb und los geht's mit SIDEKIC

Gerald Schröder

Hardware: Helmut
Literatur: Arnulfs Listings

```
00001 ;       SIDEKICK für TRS80-Kompatible mit Helmut's Banker
00002 ;       im Dez. 1986 von Gerald Schröder
00003 ;       nach einer Idee von Arnulf Soop
00004
00005
00006         ORG     5300h           ;Stack ab hier abwärts!
00007
00008 start   LD      (sobuf),SP      ;SP retten
00009
00010         LD      HL,retpro       ;Rückkehr-Adresse (bei 1.
00011         PUSH    HL              ; Aufruf Comp. 1-3)
00012         LD      B,4             ;Stack vorbereiten
00013 loop1   PUSH    AF
00014         DJNZ    loop1
00015         LD      (rettsp),SP     ;retten für Comp. 1-3
00016
00017         LD      A,1             ;Bank 1
00018         DI
00019         OUT     (0ech),A        ;einschalten
00020         LD      HL,schalt       ;Umschaltroutine
00021         LD      DE,schalt-diff  ;in Bank 1
00022         LD      BC,ende-schalt
00023         LDIR
00024         XOR     A
00025         OUT     (0ech),A        ;wieder Bank 0
00026
00027         LD      HL,umleit       ;Umleitung in Interrupt-
00028         LD      DE,umleit-diff1 ;Routine anlegen
00029         LD      BC,ende1-umleit
00030         LDIR
00031
00032         LD      HL,zurück       ;Rücksprung
00033         LD      DE,zurück-diff2 ;in SYS0 anlegen
00034         LD      BC,ende2-zurück
00035         LDIR
00036
00037         LD      SP,start        ;SP neu setzen
00038
00039         LD      B,3             ;drei Computer
00040 loop2   CALL    copy            ;in die Banks
00041         DJNZ    loop2
00042
00043         EI
00044         LD      SP,0000         ;SP zurück
00045 sobuf   EQU     $-2
00046
00047 retpro  RET                     ;ab
00048
00049
00050 ;       kopiert oberen und unteren Teil in Banks
00051
00052 copy    PUSH    BC              ;Zähler retten
```

```
00053           LD      A,B             ;als Bank-Nr.
00054           OUT     (0ech),A        ;für unterer Teil
00055           LD      DE,8000h+3000h  ;Ziel=b000h
00056           LD      B,50h           ;Länge = 5000h
00057           LD      C,E
00058           LD      H,30h           ;Anfang = 3000h
00059           LD      L,E
00060           LDIR                    ;(3000-7fff)=>(b000-ffff)
00061
00062           ADD     A,3             ;für oberer Teil
00063           EX      AF,AF'          ;retten
00064           LD      D,60h           ;Buffer: 6000h-6100h
00065
00066 loop3     INC     B               ;ein Sektor
00067           XOR     A               ;Bank 0
00068           OUT     (0ech),A        ;ein
00069           LDIR                    ;Sektor aus 0 in Buffer
00070           DEC     D               ;Sektor-Zeiger auf 6000h
00071           LD      A,H             ;Quelle- als Zielzeiger
00072           LD      H,D
00073           LD      D,A
00074           DEC     D
00075           INC     B
00076           EX      AF,AF'          ;Bank-Nr. zurück
00077           OUT     (0ech),A        ;anwählen
00078           EX      AF,AF'          ;wieder retten
00079           LDIR                    ;aus Buffer in Bank
00080           DEC     H               ;Buffer auf 6000h
00081           LD      A,D             ;und wieder tauschen
00082           LD      D,H
00083           LD      H,A
00084           OR      L               ;schon über ffffh hinaus?
00085           JR      NZ,loop3        ;nein
00086           OUT     (0ech),A        ;ja, Bank 0 ein
00087           POP     BC              ;Zähler zurück
00088           RET
00089
00090
00091
00092 ;         Umleitung in der Interrupt-Routine
00093
00094 diff1     EQU     $-45f6h
00095 umleit    LD      A,1             ;Bank 1
00096           OUT     (0ech),A        ;ein
00097           JP      schalt-diff     ;und anspringen
00098 ende1     EQU     $
00099
00100
00101 ;         Rückkehr-Routine
00102
00103 diff2     EQU     $-4053h
00104 zurück    OUT     (0ech),A        ;Bank x für oben an
00105           JP      45fdh           ;weiter
00106 ende2     EQU     $
00107
00108
00109
00110 ;         Schaltroutine in Bank 1
00111 ;         Ansprung von Interrupt-Routine aus
00112
```



```
00113 diff     EQU     $-8000h
00114 schalt    LD      HL,3880h        ;Tast.-zeile
00115           BIT     0,(HL)          ;SHIFT?
00116           JR      Z,goon3         ;nein, ab
00117           SRL     L               ;Tast.zeile 3840h
00118           BIT     4,(HL)          ;Down-Arrow?
00119           JR      Z,goon3         ;nein, ab
00120           SRL     L
00121           SRL     L               ;Tast.zeile 3810h
00122           LD      A,(HL)
00123           AND     0fh             ;Taste 0 bis 3?
00124           JR      Z,goon3         ;nein, ab
00125           LD      L,A             ;retten
00126           XOR     A               ;Zähler=0
00127 goon1     INC     A               ;+1
00128           SRL     L               ;solange
00129           JR      NC,goon1        ;bis Taste erreicht
00130           DEC     A               ;Zähler korrigieren
00131
00132           LD      (rettsp),SP     ;SP retten
00133
00134           LD      DE,goon4-diff   ;Rückkehradresse
00135           JR      spsuch          ;neuen SP berechnen
00136
00137 goon4     PUSH    IX              ;Register dieses Comp.
00138           PUSH    IY              ;retten
00139           EX      AF,AF'
00140           EXX
00141           PUSH    AF
00142           PUSH    BC
00143           PUSH    DE
00144           PUSH    HL
00145           LD      SP,ende-diff+150        ;zeitweilig neu
00146
00147           EX      AF,AF'          ;Comp.-Nr. zurück
00148           CP      0               ;schon eingeschaltet?
00149 momba     EQU     $-1
00150           JR      Z,abgang        ;ja
00151           LD      (banku-diff),A  ;dort unterer Teil
00152           OR      A               ;Comp. 0?
00153           JR      Z,goon2         ;ja, oben = Bank 0
00154           ADD     A,3             ;sonst oben = u +3
00155 goon2     LD      (banko-diff),A  ;Bank oberer Teil
00156           LD      A,(momba-diff)  ;Comp. 0 eingeschaltet?
00157           OR      A
00158           CALL    NZ,tausch-diff  ;nein, Comp. 0 holen
00159           LD      A,0             ;Bank unterer Teil
00160 banku     EQU     $-1
00161           LD      (momba-diff),A  ;sichern: mom. Comp.
00162           OR      A               ;falls nicht 0:
00163           CALL    NZ,tausch-diff  ;Computer holen
00164
00165 abgang    LD      DE,goon5-diff
00166           JR      spsuch
00167 goon5     LD      B,12            ;SP korrigieren
00168 ab1       DEC     SP
00169           DJNZ    ab1
00170 ;                                 Register des
00171           POP     HL              ;Computers zurück
00172           POP     DE
```

```
00173           POP     BC
00174           POP     AF
00175           EXX
00176           EX      AF,AF'
00177           POP     IY
00178           POP     IX
00179           LD      SP,(rettsp)     ;SP restaurieren
00180
00181 goon3     LD      A,(37ech)       ;urspr. Int.-rout.
00182           LD      A,(37e0h)
00183           RLCA
00184           LD      A,0             ;Bank-Nr. für oberer Teil
00185 banko     EQU     $-1
00186           JP      zurück-diff2    ;nach SYS0 für Abgang
00187
00188
00189 ;         Unterprogramm: Stack-Pos. für geg. Comp. A
00190 ;         berechnen
00191
00192 spsuch    LD      L,A             ;Nr. nach L
00193           LD      H,0
00194           LD      SP,ende-diff+20 ;Stack-Offset
00195           LD      B,5             ;+32*Comp.-Nr.
00196 sps1      SLA     L
00197           DJNZ    sps1
00198           ADD     HL,SP
00199           LD      SP,HL           ;SP in Bank 1 setzen
00200           EX      DE,HL           ;Rückspr.adr. nach HL
00201           JP      (HL)            ;und ab
00202
00203
00204
00205 ;         Unterprogramm Tausch:
00206 ;         vertauscht unteren Teil von Computer 0 mit dem
00207 ;         eines der drei anderen Computer
00208
00209 tausch    LD      (bankx1-diff),A ;gew. Bank-Nr. sichern
00210           LD      (bankx2-diff),A
00211           LD      (bankx3-diff),A
00212
00213           LD      HL,anf-diff3    ;Bereich Tausch-Rout.
00214           PUSH    HL              ;retten
00215           LD      DE,0a000h       ;Buffer für Original
00216           LD      BC,0200h
00217           LDIR                    ;Orig.-Code retten
00218
00219           POP     DE              ;7e00h
00220           PUSH    DE
00221           LD      HL,anf-diff     ;Routinen zum Tauschen
00222           LD      BC,ende-anf
00223           LDIR                    ;in die untere Hälfte
00224 ;                                  bei 7e00h
00225           LD      (stack-diff),SP ;SP retten
00226           LD      SP,7f00h        ;auch nach unten
00227           LD      HL,0fe00h       ;7e00h aus Bank x
00228           LD      DE,7f00h        ;Buffer
00229           LD      BC,9f00h        ;MSB (Buffer in Bank) -1
00230           CALL    anf-diff3       ;Aufruf Tausch-Rout.
00231           LD      SP,0000         ;SP zurück
00232 stack     EQU     $-2
```



```
00233           POP     DE              ;7e00 zrück
00234           LD      H,9fh           ;von 9f00 aus
00235           INC     B               ;BC=0200h
00236           LDIR                    ;7e00-7fff neu füllen
00237           RET
00238
00239
00240 diff3     EQU     $-7e00h
00241 anf       PUSH    BC              ;9f00 (x-1) auf Stack
00242           PUSH    DE              ;7e00 (z)    "    "
00243           LD      B,1
00244           LD      A,2             ;2 Durchläufe
00245
00246 ;7e00-7fff tauschen
00247 loopp     EX      AF,AF'          ;Zähler retten
00248           LD      A,0
00249 bankx1    EQU     $-1
00250           OUT     (0ech),A
00251           POP     DE              ;z zurück
00252           POP     AF              ;(x-1) zurück
00253           PUSH    HL              ;y retten
00254           PUSH    AF              ;(x-1) retten
00255           PUSH    DE              ;z retten
00256           LDIR                    ;y -> z, Bank x -> Buffer
00257           INC     B
00258           LD      A,1
00259           OUT     (0ech),A
00260           POP     HL              ;z zurück
00261           POP     DE              ;(x-1) zurück
00262           PUSH    HL              ;z retten
00263           LDIR                    ;z -> (x-1)
00264           INC     B               ; Buffer -> Bank 1
00265           LD      H,D             ;x
00266           POP     DE              ;z zurück
00267           PUSH    HL              ;x retten
00268           PUSH    DE              ;z retten
00269           LDIR                    ;x -> z, Bank 1 -> Buffer
00270           INC     B               ; (Original-Code)
00271           LD      A,0
00272 bankx2    EQU     $-1
00273           OUT     (0ech),A
00274           POP     HL              ;z zurück
00275           POP     AF              ;x zurück
00276           POP     DE              ;y zurück
00277           PUSH    AF              ;x retten
00278           PUSH    HL              ;z retten
00279           LDIR                    ;z -> y, Buffer -> Bank x
00280           INC     B               ; (Orig. in Bank x)
00281           LD      H,D             ;(y+1)
00282           EX      AF,AF'
00283           DEC     A
00284           JR      NZ,loopp
00285
00286 ;Stack: (x+1),z
00287
00288           POP     DE              ;Buffer z
00289           POP     AF              ;(x+1) weg
00290           LD      H,30h           ;Anfang mit unten (3000h)
00291           LD      A,0
00292 bankx3    EQU     $-1
```

```
00293           OUT     (0ech),A        ;Bank x ein
00294
00295 loop      PUSH    DE              ;"Buffer z" retten
00296           PUSH    HL              ;"von y" retten
00297           LDIR                    ;1.: Sektor in Buffer
00298
00299           POP     DE              ;"nach y" zurück
00300           LD      H,D             ;"von x" (aus Bank)
00301           SET     7,H             ;deshalb: immer >=8000h
00302           PUSH    HL              ;retten
00303           INC     B               ;ein Sektor
00304           LDIR                    ;2.: aus Bank nach unten
00305
00306           POP     DE              ;"nach x" (Bank)
00307           POP     HL              ;"von z" (Buffer)
00308           INC     B               ;ein Sektor
00309           LDIR                    ;aus Buffer in Bank
00310
00311           INC     B               ;ein Sektor
00312           DEC     H               ;Buffer bleibt
00313           LD      A,D             ;"von y":
00314           AND     7fh             ;immer unter 8000h
00315           LD      D,H             ;Buffer in DE
00316           LD      H,A             ;"von y" in HL
00317           CP      37h             ;I/O-Bereich?
00318           JR      NZ,lop          ;nein, weiter
00319           INC     H               ;ja, überspringen
00320 lop       CP      7eh             ;=fertig?
00321           JR      NZ,loop         ;weiter, wenn <> 7e00
00322
00323           LD      A,1             ;Bank 1
00324           OUT     (0ech),A        ;wieder ein
00325           RET                     ;fertig
00326 ende      EQU     $
00327
00328
00329
00330 rettsp    EQU     4058h
00331
00332
00333           END     start
```

Türkisch: Iki karpuzu bir koltugu sigderamazsin.
Du kannst keine zwei Melonen unter einen Arm klemmen.
Man kann nicht auf zwei Hochzeiten gleichzeitig tanzen.

* * * BOOLE XOR ABEL * * *

(oder: "Ent- oder -weder: d a s war hier die Frage!")

Im vorletzten INFO verkündete Hartmut (unser Obermann) ein Preisausschreiben, für dessen Gewinner er eine Flasche Wein aussetzte.
Damit hat er in dankenswerter Weise einen "frischen Wind" in unser Clubleben gebracht, und ich möchte sagen, so ein Wettbewerb sollte eigentlich in jeder unserer INFO-Nummern stehen! Doch wer bezahlt den vielen Wein bei so vielen Könnern? Unsere Clubkasse wäre sicher bald in den berüchtigten "Roten Zahlen". Es kann also immer nur der Initiative des Einzelnen überlassen bleiben.

Wie Hartmut inzwischen mitteilte, gingen mehr als eine richtige Lösung ein. Da das ja zu erwarten war - denn die sog. "logischen Funktionen" (auch Logic Operands genannt) kann eigentlich jeder begreifen, der logisch denken kann; und eben das ist wohl ohnehin die Voraussetzung für die halbwegs seriöse Beschäftigung mit unserem Hobby! -

also: Da das zu erwarten war, hatte Hartmut den Preis zunächst demjenigen ausgesetzt, dessen Lösung als erste einging!
Um so nobler war es von ihm, letztenendes doch j e d e m Einsender einer richtigen Lösung je eine Flasche aus "seinem" Weinberg zu verpassen...
Ein **PROST!** auf den edlen Spender und meinen besonderen Dank auch noch einmal an dieser Stelle, Hartmut!

Nun aber kommt, wie so oft, hier "das dicke Ende"! Sehen wir uns doch noch einmal Hartmuts Aufgabenstellung im INFO-Heft 15 vom September 1986, Seite 10, an!

Dort heißt es wörtlich:

> "Wie muß die BASIC-Zeile aussehen,
> die mit Hilfe der Funktionen OR, AND und NOT
> diese (XOR-)Verknüpfung realisiert?" !!!

Logisch, daß die Lösung sich auch auf diese Operatoren beschränken muß!

Die Einsender haben dies - sine ira et studio (Tacitus) sei es gesagt: mit einer Ausnahme - sorgfältig beachtet und mehr oder weniger geschickte Vorschläge angeboten. Soweit OK.

Die "eine Ausnahme" (na: wer wohl?) hat ein bißchen anders herum gedacht. Doch davon nachher.

Wenn man schon außer Logik-Altmeister *BOOLE* auch den norwegischen Algebra-Meister *ABEL* (1824 Traktat über die algebraische Gleichung 5.Grades) heranzieht und eine **SUBTRAKTION** zuhilfe nimmt (und auch dies akzeptiert wird), so müßte es auch erlaubt sein, eine noch wesentlich kürzere und somit "elegantere" Lösung ins Feld zu führen, die auf die Boole'sche Algebra ganz verzichtet:

Was wird denn verlangt? C soll immer dann Null sein, wenn A und B gleich sind; und es soll Eins sein, wenn sie verschieden sind. Daraus folgt sofort (nicht nur algebraisch, sondern auch "logisch"):

C = - (A<>B) !

Das Minuszeichen deshalb, weil für "Logisch Wahr" in der "BASIC-Algebra" '-1' statt '+1' erzeugt wird.

Schauen wir uns die entsprechende "WAHRHEITSTAFEL" (dies ist übrigens die korrekte Bezeichnung in der BOOLE'schen Algebra) an:

A	B	A<>B	C=-(A<>B)
0	0	0	0
0	1	-1	1
1	0	-1	1
1	1	0	0

Es kommt also genau das heraus, was "XOR" tut: eine Eins nur dann, wenn ENTWEDER A ODER B gleich Eins ist - aber Null, wenn BEIDE gleich, also auch gleich Eins sind: im Gegensatz zum "OR"-Operanden, der im letzteren Fall gleich Eins ist! * ("Logisch", ja?) *[Wer entdeckt den Trugschluß ?] **)

Das komplette Prgrämmchen lautet somit:

```
10 CLS: FOR A = 0 TO 1 : FOR B = 0 TO 1
20 C = - (A <> B)
30 <L>PRINT A, B, C
40 NEXT B, A : END
```

← Hier liegt der Hund...

Übrigens ein prima "Einzeiler"! Denn das geht natürlich alles auch in eine einzige Zeile. Ich habe es nur der Deutlichkeit halber getrennt.

Unser Arnulf (die "Ausnahme") hat mir im letzten INFO die undankbare Aufgabe übertragen, soz. den "Schiedsrichter" in der "Kontroverse" (klingt zu hart; gemeint ist der Strategie--Unterschied) zwischen ihm und Hartmut zu spielen. Ich wage es:

Beide sind aus ihrer jeweiligen Perspektive im Recht. Zwar beinten - ich meine: fußten beide auf *BOOLEs* schöner Algebra. Doch Boole (1815-1864) wußte gewiß noch nichts von Bits und schon gar nicht von Bytes. Seine Lehre befaßte sich mit der *Aussagenlogik*, und zwar so allgemein, daß aus ihr in unserem Jahrhundert die euch sattsam bekannte Mengenlehre entstand (mit welcher unsere Erstklässler einige Jahre spielen durften).

Um nun die von *BOOLE* definierten Logik-Operatoren auch schlichteren Menschen - z.B. Informatikern - verständlich zu machen, werden sie immer am bitteren BIT-Beispiel exerziert, und wer nur oder vorzugsweise BASIC macht, denkt hieran zuerst und zuletzt (so myself!)

Arnulf hingegen, der selbst zugibt, kein Jünger des BASIC zu sein, hat schon als GENIE-Baby (oder noch früher?) den süßen BYTE-Honig geschleckt! Diese Byte-Biene reproduziert in ihrem Geiste grundsätzlich wieder Bytes, denn er steht der CPU wesentlich näher als wir BASICisten - und die futtert bekanntlich keine Bi(scui)ts, sondern nur Bytes in ihrer Gänze, oft sogar zwei auf einmal (ich versuchte das bereits in meinem "1.Assembler-Programm" anzudeuten).

Hier lag also der tiefere Unterschied zwischen beiden Herren: Dem Byte-Beflissenen offenbart sich natürlich 256^2 mal soviel wie dem billigen BIT-Steller! Deshalb vermißte Arnulf eine integrale (alles umfassende) Lösung.

Allerdings: Hartmut hatte danach ja auch gar nicht

gefragt! (s.o.)

Was heißt überhaupt: "BIT"?

Ich glaube: "BASIC Ist Trumpf".

Aber Glaube gehört woandershin. Z.B. in die Politik. Oder Religion. Bei BASIC nützt er nichts. Und schon gar nichts im Umgang mit Bytes. Diese bedeuten nämlich:

"Bißchen Ybung Tut Es Selten!"

Aber bei dieser Gelegenheit fällt mir etwas anderes ein, das nur indirekt mit logischen Operatoren, aber auch sehr viel mit Logik zu tun hat.

Rosenfelder beschreibt in seinem hervorragenden Buch "BASIC FASTER & BETTER" eine Kurzschreibweise für logische Bedingungen (oder Entscheidungen), die die oft umständliche Formulierung "IF...THEN...ELSE" überflüssig machen. Ich habe das Gefühl - na so'n Quatsch; ein Programmierer und Software-Freak "fühlt" nicht: er denkt (oder denkt nicht XOR!) - also ich habe den Gedanken, daß sich hierin noch weniger Kollegen auskennen als mit den "logischen Operaturen"! Und deshalb habe ich im letzten INFO ein kleines Privatissimum darüber begonnen - allerdings ohne meine Weinvorräte anzupreisen, denn sie sind so alt, daß ich nicht für deren $(CH_2)_2(OH)_2$-Freiheit garantieren kann...

So läßt uns BOOLE
gar oft recht cool.
Denn schon der junge Recke ABEL
ergriff die Algebra beim Nabel!
Ich sage das ganz ohne Spott
mit Hochachtung für beid': KaJott.

(wie manches Weib...)

Der Trugschluß:

Diese "Wahrheitstafel" gilt nur für A,B = 0 oder 1, nicht für beliebig große Zahlen! Schon bei 1-Byte-Zahlen (0 bis FFh=255d) gibt es unter 256*256 = 65536 Möglichkeiten nur 256 Fälle, in denen A=B, also nach meiner "Kurzgleichung" C=0 ist (die Wahrscheinlichkeit hierfür beträgt somit rd. 4 Promille, was ausnahmsweise mal zu wenig ist...), während in rd. 996 Promille der Fälle mein C=1 wird. Dies widerspricht der "Gleichverteilung", die obige Wahrheitstafel voraussetzt.

XOR hingegen wird bei der Kombination zweier 1-Byte-Zahlen zwar auch in 256 Fällen gleich Null, nämlich immer dann, wenn A=B; aber in den übrigen 65280 Fällen wird es nicht 1, sondern kann alle Werte von 1 bis 255 annehmen!

- Meine schöne kurze Gleichung für C ist daher zwar "elegant" - aber fatalistisch, weil sie "XOR" nicht abdeckt!

(* HEFT 11 Dezember 1986

[L.UHLAND: „Nun muß sich alles, alles wenden...!"
- aus: Frühlingslieder <!>] Kp

PS: Ich danke Arnulf für seinen Einsatz!

Assembler kontra LISP: Die Türme von Hanoi

Das Problem: Gegeben sind drei "Bauplätze" A, B und C. Auf einem Platz befindet sich ein Stapel (Turm) von Scheiben, nach Größe aufgeschichtet (die größte zuunterst). Bewege den Turm von Platz A nach B unter Benutzung des Hilfsplatzes C und unter Einhaltung der Regeln:
- Es darf nur eine Scheibe zur Zeit bewegt werden.
- Es darf keine (größere) Scheibe auf eine kleinere gelegt werden.

Lösungshinweis: Benenne die Scheiben von 1 bis n (die größte heißt n). Das Ergebnis soll eine Liste der Züge sein, z.B. bei n=2: (1 A C)(2 A B)(1 C B).
(soll bedeuten: Scheibe 1 von Turm A nach Turm C, dann Scheibe 2 ...)

(aus: Peter Schefe: "Informatik - Eine konstruktive Einführung")

So, Leute, mit diesem Problem wurde ich vor einigen Wochen konfrontiert, und zwar im Rahmen einer Erstsemster-Vorlesung Informatik bei dem Prof, der auch das oben genannte Buch verbrochen und bei uns armen Studenten dafür ganz schön kassiert hat. Nun bin ich zwar leidenschaftlicher Assembler-Freak, aber der Herr wollte den Kram gern in LISP. Falls Euch das nichts sagt: macht nichts. Das ist angeblich die Sprache der "Künstlichen Intelligenz", was immer das auch sein mag. Sollte sich jemand ernstlich dafür interessieren, schreibe ich im nächsten Info gern mehr darüber - dann ist nämlich das Semester zuende und ich sollte es dann können. Aber dann müßt Ihr mich erst brieflich oder sonstwie dazu auffordern. Wenn Ihr Pech habt, schreib' ich auch so was dazu, aber verlaßt Euch nicht drauf.

Weiter im Text. Auf die Lösung kann man ungefähr so kommen (hat mir jedenfalls mein Übungsgruppen-Leiter gesagt): Der erste Arbeiter hat die Aufgabe bekommen, den Turm von A nach B zu versetzen, wobei er C benutzen darf. Nun sagt er sich: "Verdammt, es wäre besser, wenn alle Scheiben bis auf die letzte (die größte) auf C wären, dann könnte ich die unterste (größte) Scheibe nach B packen und müßte dann nur noch den Rest-Turm von C nach B schaffen." Also stellt er dem nächsten untergebenen Arbeiter die unangenehme Aufgabe: "Schaffe alle Scheiben bis auf die unterste von A nach C. Du kannst dabei B benutzen." Dieser Arbeiter denkt nun ähnlich wie sein Chef und delegiert die Aufgabe weiter, bis ein Arbeiter erreicht wird, der die kleinste Scheibe bewegen muß, was er sofort tun kann. Man könnte auch erst dann aufhören, wenn es keine zu bewegenden Scheiben mehr gibt, dann kann dieser Arbeiter sagen: "Danke, Chef, Aufgabe ist schon erledigt."
Die jeweiligen "Rest-Türme" werden nach dem gleichen Prinzip bewegt.

Die Lösung in LISP sieht ungefähr so aus:

```
(DE HANOI (Scheiben)
    (HAN1 Scheiben 'A 'B 'C NIL))

(DE HAN1 (Scheibe von nach Hilfsturm Ergebnis)
    (COND ((ZEROP Scheibe) Ergebnis)
          (T (HAN1 (SUB1 Scheibe)
                   von
                   Hilfsturm
                   nach
                   (CONS (LIST Scheibe von nach)
                         (HAN1 (SUB1 Scheibe)
                               Hilfsturm
                               nach
                               von
                               Ergebnis))))))
```


Dazu wäre folgendes zu sagen:
Es handelt sich bei dem verwendeten LISP um UCI-LISP, das auf einer DEC-10 läuft (noch, bald kommt das Ding auf den Schrotthaufen). Die (hier benutzten) eingebauten Funktionen:
(DE name (Lambda-Ausdruck)): definiert Funktion, die unter "name" aufgerufen werden kann.
'x=(QUOTE x): gibt an, daß "x" eine Konstante ist (keine Variable)
NIL: leere Liste oder Null oder Wahrheitswert FALSE
(COND ...): Konditional, eine Art IF...THEN...ELSE...
(ZEROP x): wird TRUE, wenn x=0 und FALSE, wenn x<>0
T: ist der Wahrheitswert TRUE, also wie (1=1)
(SUB1 x): vermindert x um 1 bzw. hat als Ergebnis (x-1)
(CONS x l): setzt das Element x vorne auf die Liste l auf bzw. hat als Ergebnis die Liste aus dem Element x und der Liste l
(LIST x y z ...): setzt aus den Elementen x... eine Liste zusammen bzw. hat als Ergebnis die Liste mit den Elementen x, y, z ...
HANOI, HAN1: zwei Funktionen bzw. deren Namen
Scheiben, Scheibe, von, nach, Hilfsturm, Ergebnis: Variablen

Wie Ihr seht, ist alles auf Listen und Funktionen aufgebaut. Die Arbeitsweise des Programms:
Durch Eingabe von "(HANOI x)", wobei x eine Zahl ist, wird das Programm gestartet. Die einzige Ausgabe ist das Ergebnis dieses Funktionsaufrufs (Funktion HANOI mit dem Argument x, also genauso wie f(x) oder cos(x)).
Als erstes erhält die Variable "Scheiben" den Wert x zugewiesen. Dann wird eine zweite Funktion (HAN1) aufgerufen, diesmal mit folgenden Zuweisungen:
- Scheibe := Anzahl der Scheiben
- von := "Von-Turm" = A
- nach := "Ziel-Turm" = B
- Hilfsturm := C
- Ergebnis := NIL (leere Liste, noch kein Zug)

Nun wird abgefragt, ob die zu bewegende Scheibe "0" ist (ZEROP Scheibe). Wenn dies zutrifft, braucht keine Scheibe bewegt zu werden ("Danke, Chef ...").
Ansonsten wird immer (durch das "T") ein etwas komplizierter, doppelt rekursiver Ausdruck ausgeführt: Es wird wieder die Funktion HAN1 aufgerufen, wobei die Anzahl der zu bewegenden Scheiben um eins vermindert wird, also ist das neue "scheibe":=(altes "scheibe" -1). "Nach"- und Hilfsturm werden vertauscht, denn danach sollen die Scheiben bis auf die größte auf dem Hilfsturm liegen. Also neues "von":=altes "von"; neues "nach":=altes "Hilfsturm" und neues "Hilfsturm":=altes "nach". Außerdem muß noch ein "Ergebnis" übergeben werden. Dieses Ergebnis besteht aus einer Liste (die mit "CONS" gebaut wird). Diese Liste hat als erstes Element den Zug, den der Arbeiter ausführt, nämlich die Scheibe "Scheibe" von Turm "von" nach Turm "nach" zu packen (diese Liste wird durch "LIST" erzeugt). Dahinter folgen noch alle Züge, die gemacht werden müssen, nachdem dieser Zug gemacht wurde, d.h. hier wird der "Restturm" von dem "Hilfsturm" endgültig zum "nach"-Turm bewegt. Also muß hier wieder die Funktion HAN1 aufgerufen werden, nur mit entsprechend vertauschten Parametern.
Mit anderen Worten: eigentlich drehen wir die Arbeit um. Zuerst wird der letzte Aufruf von HAN1 abgehandelt, es wird also der Restturm zum Zielturm gebracht, wobei das Ergebnis der Funktion die Liste der dazu nötigen Züge ist. Dann fügen wir vorne an die Liste den Zug an, den der Arbeiter machen sollen. Nun erst kann der zweite Aufruf von HAN1 erfolgen, bei dem als "Ergebnis" die Liste der danach folgenden Züge übergeben wird (die Anführungsstriche bedeuten: die Variable mit dem Namen "Ergebnis" ist gemeint). Diese Übergabe des Ergebnisses nennt man Akkumulieren.
Das hört sich alles sehr schwierig an, aber nach dem fünften Lesen müßte es langsam dämmern. Schließlich habe ich auch einige Minuten (Rechenzeit) und einige Stunden (Arbeitszeit) gebraucht, bis diese Lösung stand, so kurz sie auch ist. Und vorher habe ich etliche Stunden in der Vorlesung mein Sitzfleisch geschult.

Aber Ihr fragt Euch: was soll das Ganze? Will der mir etwa was beibringen?
Weit gefehlt! Denn nun folgt der interessante Teil: die Lösung in anderen Programmiersprachen. In Pascal soll es ähnlich aussehen wie in LISP, aber den Artikel überlassen ich anderen, denn Pascal kommt erst im nächsten Semester dran. Aber die gute alte Maschinensprache kann das auch! Wer glaubt, daß Rekursionen nicht möglich sind, lasse sich mit nachfolgendem Programm eines besseren belehren. Es macht für die Rekursion intensiv vom Stack Gebrauch, der deswegen auch überwacht werden muß, sonst überschreibt er plötzlich das Programm.
Interessant ist nun natürlich der Vergleich: von der Länge her gewinnt LISP klar, aber sicherlich wäre das Teil compiliert ziemlich lang. Auf der DEC 10 haben wir aber nur einen Interpreter, also kann ich nichts darüber sagen. Aus dem gleichen Grunde fallen Geschwindigkeitsvergleiche flach, denn je nach Tageszeit sind bis zu 12 Leute gleichzeitig am Werkeln, was natürlich die Reaktionszeit herabsetzt und außerdem müssen die Daten erst mühsam von/zu den Terminals übertragen werden, was unheimlich verlangsamt. Aber eine Kapazitätsprüfung ist angebracht. Ohne besondere Platz-Reservierung schafft es die DEC 10 gerade bis zu einem Turm mit 8 Scheiben (in Worten: acht, immerhin ein Großrechner, auf dem die Rechenstunde 2500 DM kosten soll, wenn er auch nur 256 Kb Hauptspeicher (in 36-Bit-Worten) hat!). Mein kleiner IIs mit seinen 64Kb Hauptspeicher, von denen in diesem Programm nur 32Kb für den Stack benutzt werden dürfen, schafft 16 in ca. 2 Minuten. Da sich mit jeder neuen Scheibe die Rechenzeit verdoppelt, habe ich dort die Tests abgebrochen. Aber bei 64 Scheiben hat er ca. 20 min gerechnet (bis mir der Geduldsfaden riß), ohne einen Stacküberlauf zu produzieren. Ich würde sagen, daß die gesamte Rechenzeit bei 64 Scheiben ca. 800 Mio Jahre beträgt, kann das aber selbst nicht glauben und bin über jeden Gegenbeweis dankbar.
Außerdem warte ich auf weitere Vergleiche und eventuelle Anfragen wegen LISP.

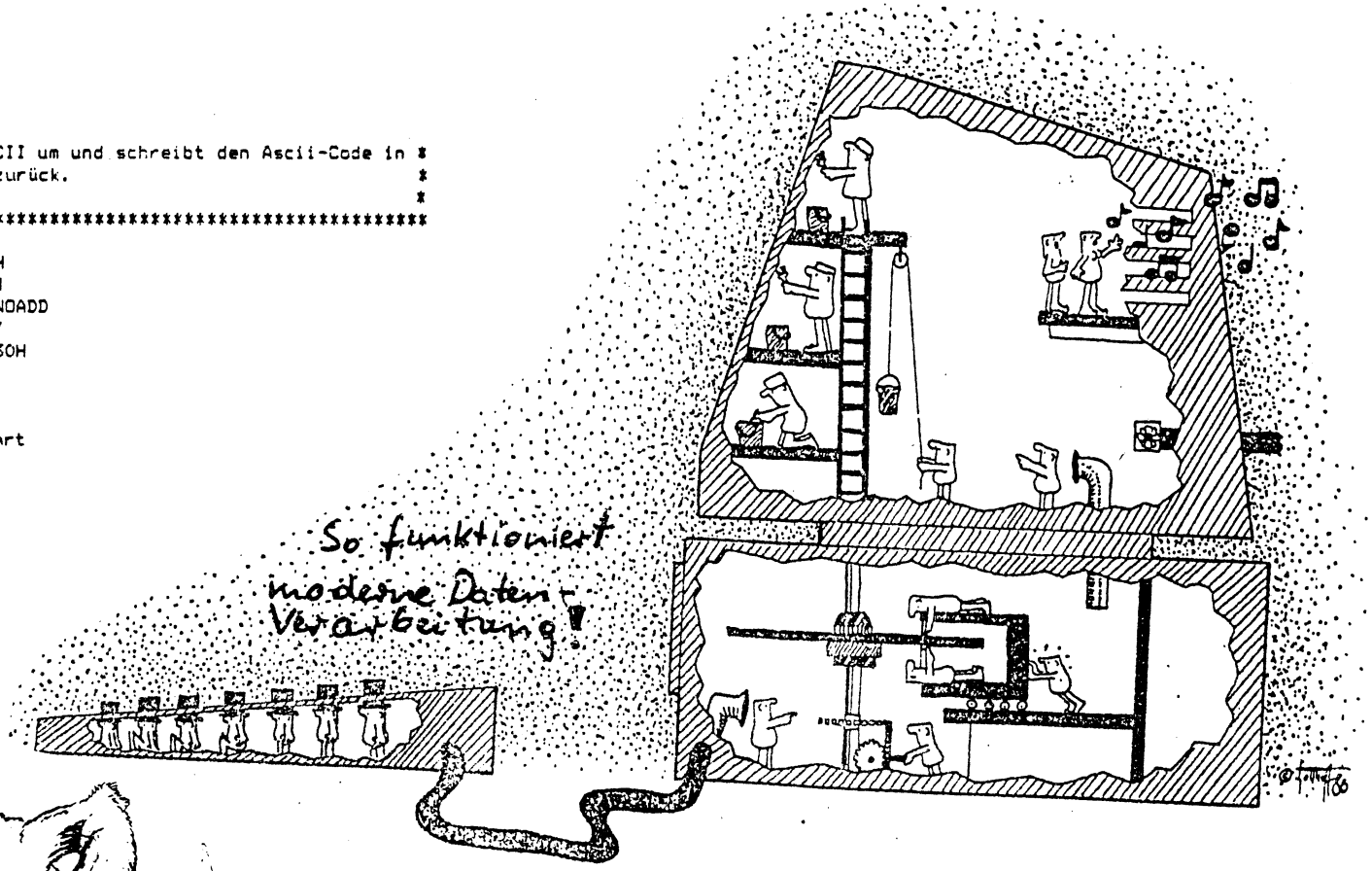
Gerald Schröder

(Folgende Unterprogramme des nachfolgenden Listings stammen nicht von mir, sondern von Anton Hummel: HEXJ, PRTHX, ASC.)

```
00001 ;       Hanoi
00002 ;       rekursive Lösung des Problems "Türme von Hanoi"
00003 ;       von Gerald Schröder
00004
00005 ;       im Dez. '86 nach eigener LISP-Vorlage
00006
00007 ;       Aufruf durch "HANOI/CMD xx", wobei xx die
00008 ;       Höhe des Turms in Hex angibt
00009
00010 out     EQU     33h             ;Ausgabe-Routine (o. 3Bh für Drucker)
00011
00012
00013         ORG     5200h
00014 start   CALL    holarg          ;Zahl holen
00015         LD      B,A             ;Anzahl Scheiben nach B
00016         CALL    PRTHX           ;Höhe ausgeben
00017         LD      A,0dh           ;mit CR
00018         CALL    out
00019         LD      (stack),SP      ;SP retten
00020         LD      SP,0000         ;neu setzen
00021
00022         LD      H,'a'           ;"von"-Turm
00023         LD      L,'b'           ;"nach"-Turm
00024         LD      C,'c'           ;"Hilfsturm"
00025         LD      IY,3880h        ;für BREAK-Abfrage
00026
00027         CALL    hanoi           ;Tausch aufrufen
00028
00029         LD      A,0dh           ;CR
00030         CALL    out             ;ausgeben
00031 ab      LD      SP,0000         ;SP zurück
00032 stack   EQU     $-2
00033         RET                     ;ab ins DOS
00034
00035
00036 ;       ****************************************
00037 ;       * Haupt-Unterprogramm: Packe B Scheiben *
00038 ;       * von Turm H nach L mithilfe von C.     *
00039 ;       * dieser Teil ist rekursiv!!!           *
00040 ;       ****************************************
00041
00042 hanoi   INC     B               ;0 Scheiben?
00043         DEC     B
00044         RET     Z               ;ja fertig
00045         BIT     0,(IY+0)        ;Shift?
00046         JP      NZ,402dh        ;ja, BREAK
00047         PUSH    HL              ;rette von/nach
00048         PUSH    BC              ;rette Hilfst., Scheiben
00049
00050         CALL    sphigh          ;ab, wenn Stack zu groß
00051         DEC     B               ;eine Scheibe weniger
00052         LD      A,C
00053         LD      C,L             ;"nach" als "Hilfsturm"
00054         LD      L,A             ;"Hilfsturm" als "nach"
00055         CALL    hanoi           ;rekursiver Aufruf
00056
00057         POP     BC              ;Hilfsturm, Scheiben
00058         POP     HL              ;von/nach
00059         CALL    ausgabe         ;Bewegung ausgeben
00060
00061         DEC     B               ;eine Scheibe weniger
00062         LD      A,C
00063         LD      C,H             ;"von" als "Hilfsturm"
00064         LD      H,A             ;"Hilfsturm" als "von"
00065         JP      hanoi           ;Abgang mit Tauschen
00066
00067
00068 ;       ****************************************
00069 ;       * gibt Tauschaktion aus im Format:     *
00070 ;       *       (xx v n)                       *
00071 ;       * wobei xx = Nr. der Scheibe (in hex)  *
00072 ;       *        v = von Turm (a, b oder c)    *
00073 ;       *        n = nach Turm (a, b oder c)   *
00074 ;       ****************************************
00075
00076
00077 ausgabe LD      A,'('           ;Klammer-auf
00078         CALL    out             ;ausgeben
00079         LD      A,B             ;Scheiben-Nr. nach A
00080         CALL    PRTHX           ;in Hex ausgeben
00081         LD      A,20h           ;Leerzeichen
00082         CALL    out             ;ausgeben
00083         LD      A,H             ;von-Turm
00084         CALL    out             ;ausgeben
00085         LD      A,20h           ;Leerzeichen
00086         CALL    out             ;ausgeben
00087         LD      A,L             ;nach-Turm
```

```
00088           CALL    out             ;ausgeben
00089           LD      A,')'           ;Klammer-zu
00090           CALL    out             ;ausgeben
00091           LD      A,20h           ;Leerzeichen
00092           CALL    out             ;ausgeben
00093           RET
00094
00095
00096 ;         ****************************************
00097 ;         * überprüft, ob SP unterhalb von 8000h *
00098 ;         * wenn ja: zu groß, ab                 *
00099 ;         ****************************************
00100
00101 sphigh    PUSH    HL
00102           LD      H,0
00103           LD      L,H
00104           ADD     HL,SP
00105           BIT     7,H             ;>=8000h?
00106           POP     HL
00107           RET     NZ              ;ja, Bit 7 = 1
00108
00109           LD      HL,spmes        ;nein, Meldung
00110           CALL    4467h           ;ausgeben
00111           JR      ab              ;und Abgang
00112
00113 spmes     DM      0ah,'Stack zu groß! Abbruch.',0dh
00114
00115
00116
00117 ;         ***********************************
00118 ;         * holt Argument (Höhe des Turms *
00119 ;         * in Scheiben, Hex-Zahl)        *
00120 ;         ***********************************
00121
00122 holarg    CALL    4cd5h           ;nächstes Zeichen
00123           JR      Z,error         ;=CR: Fehler
00124           LD      A,(HL)          ;holen
00125           CALL    wertaus         ;Zahl auswerten
00126           LD      A,B             ;Ergebnis nach A
00127           RET     NC              ;kein Fehler: fertig
00128
00129 error     LD      HL,errmes       ;bei Fehler:
00130           CALL    4467h           ;Meldung ausgeben
00131           JP      402dh           ;ab ins DOS
00132
00133 errmes    DM      'Fehler: kein oder falsches Argument!',0dh
00134
00135
00136 ;         ************************************************
00137 ;         * wertet ein- oder zweistellige Hex-Zahl aus *
00138 ;         ************************************************
00139
00140
00141 wertaus   CALL    HEXJ            ;ASCII in binär
00142           RET     C               ;Fehler: zurück
00143           LD      B,A             ;nach B retten (einst.)
00144           RRCA                    ;falls zweistellig:
00145           RRCA                    ;höherwertiges Nibble
00146           RRCA                    ;erzeugen
00147           RRCA
00148           LD      C,A             ;und nach C
00149           INC     HL
00150           LD      A,(HL)          ;nächstes Zeichen
00151           CP      0dh
00152           RET     Z               ;CR: fertig
00153           CALL    HEXJ            ;in binär
00154           RET     C               ;falsch: Fehler
00155           OR      C               ;sonst höheres N. dazu
00156           LD      B,A             ;für Ergebnis-Ausgabe
00157           RET
00158
00159
00160 ;*****************************************************************
00161 ;*                                                               *
00162 ;* CALL HEXJ                     geschützte Register:            *
00163 ;*                               alle außer AF                   *
00164 ;* Wandelt Ascii-Zeichen aus dem Register A in Binär-             *
00165 ;* zahl um und speichert Ergebnis in A.                           *
00166 ;* Ausgabe-Flags: C=0 = alles Ok                                  *
00167 ;*                C=1 = Fehler                                    *
00168 ;*****************************************************************
00169
00170
00171 HEXJ    CP      30H
00172           RET     C
00173           CP      3AH
00174           JR      C,HEX1
00175           SUB     7
00176           CP      40H
00177           JR      NC,HEX2
00178 HEX1    AND     0FH
00179           RET
00180 HEX2    SCF
00181           RET
00182
00183 ;*****************************************************************
00184 ;*                                                               *
00185 ;* CALL PRTHX                    GESCHÜTZTE REGISTER:            *
00186 ;*                               ALLE AUßER AF                   *
00187 ;* Zeigt den Inhalt des Registers A als 2-stellige                *
00188 ;* Hexadezimalzahl auf dem Bildschirm an.                         *
00189 ;*                                                               *
00190 ;*****************************************************************
00191
00192
00193 PRTHX   PUSH    AF
00194           RRCA
00195           RRCA
00196           RRCA
00197           RRCA
00198           CALL    ASC
00199           CALL    out     ;(AUSGABE)
00200           POP     AF
00201           CALL    ASC
00202           JP      out
00203
00204
00205 ;*****************************************************************
00206 ;*                                                               *
00207 ;* CALL ASC                      geschützte Register:            *
00208 ;*                               alle außer AF                   *
00209 ;* Wandelt den Inhalt der unteren 4 Bit des Registers A *
```

```
00210 ;* von Binär in ASCII um und schreibt den Ascii-Code in *
00211 ;* das Register A zurück.                                 *
00212 ;*                                                       *
00213 ;*******************************************************
00214
00215 ASC     AND     0FH
00216         CP      0AH
00217         JR      C,NOADD
00218         ADD     A,7
00219 NOADD   ADD     A,30H
00220         RET
00221
00222
00223         END     start
```



Erklär du ihm doch, daß wir Pac-Man nicht haben! . . .

Computerlexikon:

directory

In gewöhnlichem Englisch bedeutet das Wort Adreßbuch oder auch Telefonbuch, im Computer-Englisch dagegen ist die Bedeutung eng begrenzt. Das directory ist das Inhaltsverzeichnis aller auf einem Massenspeicher abgespeicherten Dateien, sei es nun ein Magnetband, eine Floppy oder eine Festplatte. Bei einem Personal Computer bewirkt der Befehl DIR, wenn man im Betriebssystem arbeitet, daß sämtliche Dateinamen aufgelistet werden, meist mit dem Umfang des belegten Speicherplatzes; also der Anzahl der Bytes, die sie einnehmen. Das erleichtert nicht nur das Suchen, sondern zeigt auch gleich, wieviel Speicherplatz noch frei ist. Natürlich kennen auch Heimcomputer so einen Befehl, doch ist er hier von Gerätetyp zu Gerätetyp verschieden. Will man bei dem weit verbreiteten C 64 einen Blick ins »Inhaltsverzeichnis« einer Diskette tun, genügt der Befehl LOAD "$",8 schon hat man alles auf dem Bildschirm. ★

Z80, HD64180 und Illegals im ROM-Modus

Das Herz unseres kleinen Haufens Silikonmasse besteht bekanntermaßen aus einem Z80. Dieser Z80 kann über 1100 Befehle verarbeiten, von denen aber nur 694 von den Erbauern vorgesehen waren. Der Rest ist also Abfallprodukt.
Nun gibt es einen neuen Prozessor, den HD64180, der Z80-kompatibel ist, aber einge Sachen mehr kann (512Kb RAM verwalten, DMA, Schnittstelle...). Nur haben die Erbauer dieses Dings leider vergessen, die Befehle des Z80 zu berücksichtigen, die nicht offiziell dokumentiert sind. Das kann böse Probleme geben, denn wenn der neue Proz auf einen ihm unbekannten Befehl trifft, macht er einen Brutalo-Reset nach 0000h, was in unserem Fall jedes Programm ins Jenseits befördert.
Es wäre dies kein Problem, wenn niemand die "Illegals" bzw. die nicht dokumentierten Befehle benutzen würde, aber viele Programmierer tun es doch. Prominentes Beispiel: der Editor/Assembler ZEUS (3 Illegals).

Ihr könnt nun
1. den neuen Prozessor vergessen (sehr schwer!),
2. alle Programme mit Illegals vergessen (nicht ganz so schwer),
3. alle Illegals durch normale Befehle ersetzen (schwer) oder
4. dem HD64180 die unbekannten Befehle beibringen (fast leicht).

Punkt 3 erschien mir brauchbar, bis sich ZEUS nach den Änderungen so merkwürdig benahm. Leider sind die Ersatz-Befehle für die Illegals nämlich einen kleinen Tick länger als die Illegals und das gibt big problems.
Also auf zu Punkt 4! Vorraussetzung: Veränderliches RAM/ROM bei 0000h: bei jedem CP/M-fähigen Rechner der Fall.

Außerdem habe ich in meinem kleinen Programm noch einiges ausgenutzt:

Arnulf hett seggt, daß RST 00h nie gebraucht wird. Also benutze man RST 00h für etwas anderes. In diesem Fall bot sich ein Banking an, das Helmuts alten Banker simuliert. Verfahrensweise: Der Speicher von 8000h-ffffh wird als Common 1 definiert und je nach gewünschter Bank wird der Offset von Common 1 neu gesetzt. Bank 0 ist die normale Bank, also sind dann Interrupts erlaubt. Ansonsten werden die Interrupts gesperrt. Die Bank-Nr. wird vor dem RST 00h in A geladen.
Mein kleiner IIs kennt ein Parallel-RAM. D.h. Neben Tastatur, Bildschirm etc., die ja memory-mapped-I/O sind, gibt es einen RAM-Bereich von 3400h-3fffh. Je nach Wunsch kann ich nun entweder Ein-/Ausgaben vornehmen oder in diesem Extra-RAM einige zusätzliche Programme laufen lassen. Wiederum müßte jeder TRS80-CP/M-Rechner irgendetwas in der Richtung haben.
Nun versteht Ihr auch, warum ich in der Überschrift von "ROM-Modus" sprach. Der ganze Kram klappt bei CP/M in dieser Form nicht. Dafür andere Lösungen zu suchen, überlasse ich den CP/M-Freaks.
Zu guter Letzt kann ich auch das ROM/RAM überschreiben (s. wiederum CP/M-fähige Rechner). Da sich in dem ROM allerhand Müll angesammelt hat, kann man gewisse Teile ohne Skrupel killen, was ich auch getan habe.

Mit diesen ganzen Einschränkungen wird das Programm zwar sehr auf eine Maschine festgelegt, aber wer wird sich schon einen HD64180 anschaffen? Wohl doch nur Leute, die sowieso am Basteln sind und bei deren Drahtansammlungen nicht viel vom TRS80 übrig geblieben ist.

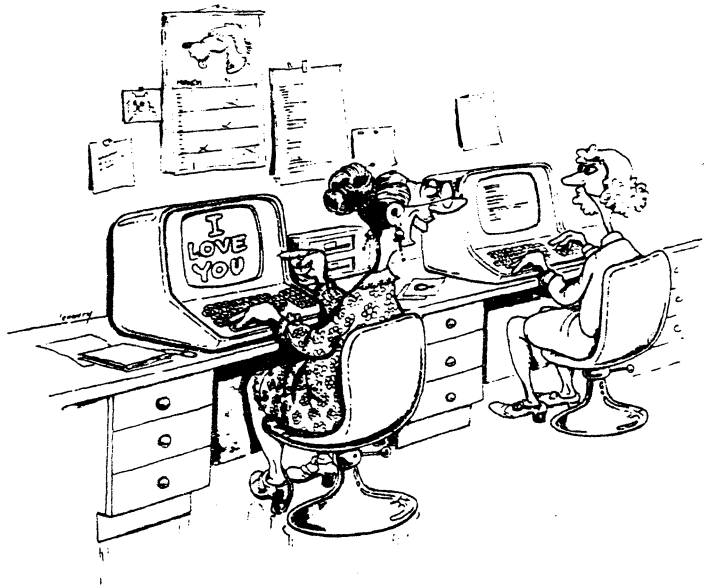
Noch kurz zum eigentlichen Programm: Der erste Teil ist nur der Ansprung der Routine von 0000h aus. Der zweite Teil ist ein Ausgang aus der Routine, bei dem die Interrupts ausgeschaltet bleiben. Die anderen Ein/Aus-Routinen stellt OVL3/SYS aus dem Gdos 2.4 zur Verfügung. Der dritte Teil ist derjenige, der beim Aufruf die Routine ins Parallel-RAM bringt. Danach folgt die eigentliche Routine, die im gegenwärtigen Zustand noch nicht mal 100h Bytes lang ist. Sie berücksichtigt alle Illegals, die in einem Artikel der "data welt" im Mai/Juni 1986 erwähnt wurden.

Die HD64180-spezifischen Befehle IN0/OUT0 wurden über DBs realisiert, weil mein alter ZEUS sie natürlich nicht kennt.

Der Programmierstil ist leider sehr "unschön", wie mein Prof sagen würde, vor allem aufgrund der vielen Patches direkt im Programmablauf, aber es tut seinen Dienst.

Gerald Schröder

Hardware: Helmut Bernhardt
Literatur: "TAV's Z80 Tuning" aus "data welt" 5 und 6 '86
Z80-Reference-Manual



Du siehst, ich bin immer noch attraktiv!

```
                        00001 ;       Error-Trap für Genie IIs mit HD64180
                        00002 ;       Weihnachten '86 by Gerald Schröder
                        00003
                        00004
06A0                    00005 pramon  EQU     06a0h           ;schaltet Par.-RAM ein
06AB                    00006 pramoff EQU     06abh           ;schaltet Par.-RAM aus
                        00007
                        00008
                        00009 ;-------------------------------------------------------
                        00010
                        00011 ;       0000: Ansprung Error-Trap und RST 00h
                        00012
0000                    00013         ORG     0
0000  CDA006             00014         CALL    pramon          ;Par.-RAM ein
0003  C30034            00015         JP      routin-offset           ;anspringen
                        00016
                        00017
                        00018 ;-------------------------------------------------------
                        00019
                        00020 ;       neuer Ausgang: Par.-RAM aus und Interrupts
                        00021 ;       bleiben disabled
                        00022
06D2                    00023         ORG     06d2h
06D2  DBFE              00024 prama   IN      A,(0feh)
06D4  CB87              00025         RES     0,A
06D6  D3FE              00026         OUT     (0feh),A
06D8  F1                00027         POP     AF
06D9  C9                00028         RET
                        00029
                        00030
                        00031 ;-------------------------------------------------------
                        00032
                        00033 ;       Initialisierungsroutine
                        00034
5200                    00035         ORG     5200H
5200  3E80              00036 start   LD      A,80h           ;Common 1 ab 8000h
5202  ED                00037         DB      0edh,39h,3ah    ;setzen (OUT0 (3ah),A)
                        00038
5205  CDA006            00039         CALL    pramon          ;Par.-RAM ein
5208  211652            00040         LD      HL,routin       ;Routine
520B  110034            00041         LD      DE,routin-offset        ;dorthin
520E  01C900            00042         LD      BC,ende-routin          ;übertragen
5211  EDB0              00043         LDIR
5213  C3AB06            00044         JP      pramoff         ;Par.-RAM aus und ab
                        00045
                        00046
                        00047 ;-------------------------------------------------------
                        00048
                        00049 ;       Routine im Par.-RAM für Error-Trap und RST 00h
                        00050 ;       bei Error: fehlende Codes simulieren
                        00051 ;       bei RST 00h: Bank A einschalten
                        00052
1E16                    00053 offset  EQU     $-3400h
5216  F5                00054 routin  PUSH    AF
5217  ED                00055         DB      0edh,38h,34h    ;IN0 A,(34h)
521A  CB7F              00056         BIT     7,A             ;TRAP gesetzt?
521C  2010              00057         JR      NZ,illopc       ;ja, illegaler Opcode
                        00058
                        00059 ;       **********************************
                        00060 ;       * Banking: Bank A einschalten *
                        00061 ;       **********************************
                        00062
521E  F1                00063         POP     AF
521F  F5                00064         PUSH    AF              ;retten
5220  07                00065         RLCA
5221  07                00066         RLCA                    ;korr. Offset erzeugen
5222  07                00067         RLCA                    ;(Basis Common 1 neu)
5223  ED                00068         DB      0edh,39h,38h    ;OUT0 (38h),A
5226  B7                00069         OR      A               ;Bank 0?
5227  C2D206            00070         JP      NZ,prama        ;nein, keine Interrupts
522A  F1                00071         POP     AF
522B  C3AB06            00072         JP      pramoff         ;Interrupts ein
                        00073
                        00074
                        00075
                        00076 ;       **********************************
                        00077 ;       * illegaler Occode: simulieren *
                        00078 ;       **********************************
                        00079
522E  CBBF              00080 illopc  RES     7,A             ;TRAP löschen
5230  ED                00081         DB      0edh,39h,34h    ;OUT0 (34h),A
5233  322534            00082         LD      (ufo-offset),A  ;UFO retten
5236  F1                00083         POP     AF              ;A vom Stack
5237  E3                00084         EX      (SP),HL         ;HL mit Err.-Adr. tausch.
5238  F5                00085         PUSH    AF              ;A wieder retten
5239  C5                00086         PUSH    BC              ;BC als Hilfsregister
523A  3E00              00087         LD      A,0             ;UFO
523B                    00088 ufo     EQU     $-1
523C  CB77              00089         BIT     6,A             ;gesetzt?
523E  2801              00090         JR      Z,nosub         ;nein, Opc. direkt davor
5240  2B                00091         DEC     HL
5241  2B                00092 nosub   DEC     HL              ;Zeiger auf ill. Opc.
5242  7E                00093         LD      A,(HL)          ;laden
5243  47                00094         LD      B,A             ;und retten
5244  FECB              00095         CP      0cbh            ;CBxx? (SLIA)
5246  23                00096         INC     HL              ;Zeiger weiter
5247  7E                00097         LD      A,(HL)          ;2. Byte laden
5248  200F              00098         JR      NZ,ddfd         ;nein, Oper. mit IX/Y
                        00099
                        00100 ;       **************************************
                        00101 ;       * illegaler Opcode: CBxx = SLIA x *
                        00102 ;       **************************************
                        00103
524A  D620              00104         SUB     20h             ;korrigieren
524C  323F34            00105         LD      (CBxx-offset),A ;in den Programm-Text
                        00106
524F  C1                00107         POP     BC              ;Register zurück
5250  F1                00108         POP     AF
5251  23                00109         INC     HL              ;PC neu setzen
5252  E3                00110         EX      (SP),HL         ;und HL zurück
5253  37                00111         SCF                     ;für invertiert!!!
5254  CB10              00112         RL      B               ;und rotieren
5255                    00113 CBxx    EQU     $-1             ;10-17 für Register B-A
5256  C3AB06            00114 ab      JP      pramoff         ;weiter im Programm
                        00115
                        00116
                        00117 ;       **************************************
                        00118 ;       * illegaler Opcode: erstes Byte DD/FD *
                        00119 ;       **************************************
                        00120
```

```
5259 FECB   00121 ddfd   CP    0cbh              ;danach CBh?
525B 2828   00122       JR    Z,ddfdcb          ;ja
525D 326834 00123       LD    (ersatz-offset),A       ;in den Text
5260 FE26   00124       CP    26h               ;"LD hX/Y,konstante"?
5262 2807   00125       JR    Z,ldkon           ;ja
5264 FE2E   00126       CP    2eh               ;"LD IX/Y,konstante"?
5266 2803   00127       JR    Z,ldkon           ;ja
            00128
5268 AF     00129       XOR   A                 ;keine Konstante: NOP
5269 1802   00130       JR    weiter
            00131
            00132 ;     Konstante nach MSB/LSB von IX/Y
            00133
526B 23     00134 ldkon INC   HL                ;Zeiger auf Konstante
526C 7E     00135       LD    A,(HL)            ;diese laden
            00136
526D 326934 00137 weiter LD   (konst-offset),A  ;u. in den Text
5270 78     00138       LD    A,B               ;DD/FD zurück
5271 326534 00139       LD    (ddfd1-offset),A  ;setzen
5274 326B34 00140       LD    (ddfd2-offset),A
            00141
5277 C1     00142       POP   BC                ;alle Register zurück
5278 F1     00143       POP   AF
5279 23     00144       INC   HL                ;PC neu
527A E3     00145       EX    (SP),HL           ;HL zurück
            00146
527B DDE5   00147 ddfd1 PUSH  IX                ;bzw. IY
527D E3     00148       EX    (SP),HL           ;nach HL
527E 7F     00149 ersatz LD   A,A               ;Operation ausführen
527F 00     00150 konst NOP                     ;evtl. Konstante
5280 E3     00151       EX    (SP),HL           ;neues IX/Y zurück
5281 DDE1   00152 ddfd2 POP   IX                ;bzw. IY
5283 18D1   00153       JR    ab
            00154
            00155
            00156 ;     ****************************************
            00157 ;     * DD/FD gefolgt von CB                  *
            00158 ;     * RLC/RRC/RL/RR/SLA/SRA/(SLIA)/SRL,     *
            00159 ;     *  (IX/Y+d),B/C/D/E/H/L/A               *
            00160 ;     ****************************************
            00161
5285 78     00162 ddfdcb LD   A,B               ;DD/FD zurück
5286 32C034 00163       LD    (ddfd3-offset),A  ;setzen
5289 32C534 00164       LD    (ddfd4-offset),A
528C 23     00165       INC   HL                ;Zeiger auf Offset
528D 7E     00166       LD    A,(HL)            ;"d" aus (IX/Y+d) laden
528E 32C234 00167       LD    (offs1-offset),A  ;setzen
5291 32C734 00168       LD    (offs2-offset),A
5294 23     00169       INC   HL                ;Zeiger auf Operation
            00170
            00171 ;     Lade-Befehl errechnen und in den Text
            00172
5295 7E     00173       LD    A,(HL)            ;Befehl laden
5296 E607   00174       AND   7                 ;nur die Bits 0-2
5298 3C     00175       INC   A                 ;+1
5299 47     00176       LD    B,A               ;als Zähler
529A 3E3E   00177       LD    A,3eh             ;Opcode 46h-7eh erzeugen
            00178
529C C608   00179 loop1 ADD   A,8               ;(46/4e/56/5e/66/6e/7e)
529E 10FC   00180       DJNZ  loop1             ;=LD B/C/D/E/H/L/A,(IX..)
```



```
52A0 32C634 00181       LD    (lade-offset),A ;setzen
            00182
            00183 ;     Rotations-/Schiebe-Befehl erzeugen
            00184
52A3 7E     00185       LD    A,(HL)
52A4 E6F8   00186       AND   0f8h              ;die Bits 0-2 löschen
52A6 0F     00187       RRCA                    ;und raus rotieren
52A7 0F     00188       RRCA
52A8 0F     00189       RRCA
52A9 3C     00190       INC   A                 ;+1
52AA 47     00191       LD    B,A               ;als Zähler
52AB 3EFE   00192       LD    A,0feh            ;6/e/16/1e/26/2e/3e=RLC,
            00193
52AD C608   00194 loop2 ADD   A,8               ;RRC,RL,RR,SLA,SRA,SRL
52AF 10FC   00195       DJNZ  loop2             ;(u. RES/SET x)
            00196
52B1 48     00197       LD    C,B               ;NOP erzeugen (für reta)
            00198
52B2 FE36   00199       CP    36h               ;SLIA?
52B4 200B   00200       JR    NZ,noSLIA         ;nein
            00201
            00202 ;     SLIA (IX/Y+d) oder SLIA (IX/Y+d),r
            00203
52B6 0637   00204       LD    B,37h             ;SCF erzeugen
52B8 7E     00205       LD    A,(HL)            ;SLIA (IX/Y+d)?
52B9 FE36   00206       CP    36h
52BB 2002   00207       JR    NZ,nopro          ;nein, kein Problem
52BD 0EC9   00208       LD    C,0c9h            ;RET für reta erzeugen
            00209
52BF 3E16   00210 nopro LD    A,16h             ;RL erzeugen (statt SLIA)
            00211
52C1 32C334 00212 noSLIA LD   (opera-offset),A  ;setzen
52C4 78     00213       LD    A,B               ;SCF oder NOP
52C5 32BF34 00214       LD    (cflag-offset),A  ;setzen
52C8 79     00215       LD    A,C               ;RET oder NOP
52C9 32C434 00216       LD    (reta-offset),A ;setzen
            00217
52CC C1     00218       POP   BC                ;Register zurück
52CD F1     00219       POP   AF
52CE 23     00220       INC   HL                ;PC korrigieren
52CF E3     00221       EX    (SP),HL           ;und HL zurück
            00222
52D0 CDBF34 00223       CALL  cflag-offset      ;Operation ausführen
52D3 1881   00224       JR    ab
            00225
            00226 ;     Unterprogramm: Operation simulieren
            00227
52D5 00     00228 cflag NOP                     ;oder SCF
            00229
52D6 00     00230 ddfd3 NOP                     ;DD o. FD für IX/Y
52D7 CB     00231       DB    0cbh              ;immer CBh
52D8 00     00232 offs1 NOP                     ;Offset d (IX/Y+d)
52D9 00     00233 opera NOP                     ;Operationsanweisung
            00234
52DA 00     00235 reta  NOP                     ;RETurn bei SLIA (IX/Y+d)
            00236
52DB 00     00237 ddfd4 NOP                     ;DD/FD für IX/Y
52DC 7F     00238 lade  LD    A,A               ;Ladebefehl
52DD 00     00239 offs2 NOP                     ;Offset (s. offs1)
52DE C9     00240       RET
            00241
52DF        00242 ende  EQU   $
            00243
            00244
5200        00245       END   start

00000 Fehler
```

TAV's Z80 Tuning

Illegals

Die Z80-CPU ist mit ihren 694 Befehlen ein überaus leistungsfähiger Mikroprozessor. Dieser riesige Befehlsschatz läßt sich durch sogenannte "Illegals" auf über 1100 Op-Codes erweitern. Ein Illegal ist ein vom Hersteller nicht definierter Op-Code, der normalerweise eine Fehlfunktion des Prozessors verursacht. Viele dieser Fehlfunktionen repräsentieren jedoch eine sinnvolle Befehlserweiterung.

Im Bereich der Realtime-Programmierung kann es vorkommen, daß eine Routine einige tausendmal innerhalb einer Programmschleife aufgerufen wird. In solchen Fällen macht sich das Einsparen jedes Taktzyklus bemerkbar. Ein gezielter Einsatz der folgenden Operationen kann insbesondere dazu beitragen, kritische Bereiche zu unterschreiten und somit ungewollte Fehlfunktionen eines Programmes zu vermeiden. Weiterhin können in Grenzfällen Speicherplatzprobleme gelöst werden, da ein Illegal mindestens durch zwei konventionelle Befehle ersetzt werden muß.

Illegals lassen sich darüberhinaus hervorragend in einen Programmschutz einbauen, da sie kein mir bekannter Disassembler in sinnvoller Form interpretieren kann.

Neue Shift- und Rotations-Befehle

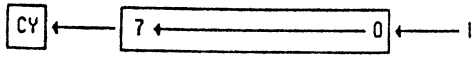
Als erstes möchte ich Ihnen eine gänzlich neue Shift-Operation vorstellen - die **SLIA**-Funktion. (Shift Left Inverted Arithmetic). Sie entspricht der **SLA**-Operation mit dem Unterschied, daß hierbei das relative Bit 0 mit logisch 1 (High) aufgefüllt wird.

Bitte beachten Sie, daß ihr Assembler die oben vorgestellten symbolischen Befehlscodes nicht direkt verstehen kann. Bei der Programmierung müssen die ihnen zugeordneten Op-Codes mittels den DEFB- und DEFW-Anweisungen folgendermaßen in den Source-Code implementiert werden.

Beispiel:

```
LD      A,(MEM1)
;Hohe Testzahl
DEFW    +37C9
;SLIA A (Testzahl=Testzahl*2+1)
LD      (MEM1),A
;Ergebnis zurückschreiben
RET ;Test beendet
```

In diesem Beispiel wird ein beliebiger Testwert aus der Speicherzelle MEM1 in den Akkumulator geladen, mittels SLIA A Operation manipuliert und in die Speicherzelle MEM1 zurückgeschrieben. Die Implementierung des SLIA-Befehles erfolgte in unserem Beispiel mit Hilfe der **DEFW**-Anweisung. Es ist in diesem Zusammenhang unbedingt zu beachten, daß High- und Low-Byte des benötigten 2-Byte Op-Codes vertauscht werden müssen (siehe Tabelle). Falls die **DEFB**-Anweisung verwendet werden sollte, müssen die Op-Codes in normaler Reihenfolge eingegeben werden.
An dieser Stelle endet der 1. Teil über Z80-Operationen, die in keinem Handbuch stehen. Im nächsten Monat werden wir uns unter anderem mit neuen Befehlen zur gezielten High/Low-Byte Manipulation der 16-Bit Index-Register beschäftigen. Bis dahin wünsche ich Ihnen viel Spaß beim Experimentieren mit der Z80-CPU und immer ein Bit zuviel.(tav) ●



Name	Op-Code	Name	Op-Code
SLIA B	CB 30	SLIA C	CB 31
SLIA D	CB 32	SLIA E	CB 33
SLIA H	CB 34	SLIA L	CB 35
SLIA (HL)	CB 36	SLIA A	CB 37

Links orientiertes indiziert-indirektes invertiert-arithmetisches "Shiften" mit Transfer des Endergebnisses in ein beliebiges 8-Bit Register.

Name:	Op-Code	Name	Op-Code
SLIA (IX+d),B	DD CB d 30	SLIA (IY+d),B	FD CB d 30
SLIA (IX+d),C	DD CB d 31	SLIA (IY+d),C	FD CB d 31
SLIA (IX+d),D	DD CB d 32	SLIA (IY+d),D	FD CB d 32
SLIA (IX+d),E	DD CB d 33	SLIA (IY+d),E	FD CB d 33
SLIA (IX+d),H	DD CB d 34	SLIA (IY+d),H	FD CB d 34
SLIA (IX+d),L	DD CB d 35	SLIA (IY+d),L	FD CB d 35
SLIA (IX+d)	DD CB d 36	SLIA (IY+d)	FD CB d 36
SLIA (IX+d),A	DD CB d 37	SLIA (IY+d),A	FD CB d 37

d = Distanzadresse (00 - FF)

RLC (xr+d),r

Name:	Op-Code	Name	Op-Code
RLC (IX+d),B	DD CB d 00	RLC (IY+d),B	FD CB d 00
RLC (IX+d),C	DD CB d 01	RLC (IY+d),C	FD CB d 01
RLC (IX+d),D	DD CB d 02	RLC (IY+d),D	FD CB d 02
RLC (IX+d),E	DD CB d 03	RLC (IY+d),E	FD CB d 03
RLC (IX+d),H	DD CB d 04	RLC (IY+d),H	FD CB d 04
RLC (IX+d),L	DD CB d 05	RLC (IY+d),L	FD CB d 05
RLC (IX+d),A	DD CB d 07	RLC (IY+d),A	FD CB d 07

Indiziert-indirekte Verschiebeoperationen mit Transfer des Endergebnisses in ein Register.
RR (xr+d),r

Name:	Op-Code	Name	Op-Code
RR (IX+d),B	DD CB d 18	RR (IY+d),B	FD CB d 18
RR (IX+d),C	DD CB d 19	RR (IY+d),C	FD CB d 19
RR (IX+d),D	DD CB d 1A	RR (IY+d),D	FD CB d 1A
RR (IX+d),E	DD CB d 1B	RR (IY+d),E	FD CB d 1B
RR (IX+d),H	DD CB d 1C	RR (IY+d),H	FD CB d 1C
RR (IX+d),L	DD CB d 1D	RR (IY+d),L	FD CB d 1D
RR (IX+d),A	DD CB d 1F	RR (IY+d),A	FD CB d 1F

RL (xr+d),r

Name:	Op-Code	Name	Op-Code
RL (IX+d),B	DD CB d 10	RL (IY+d),B	FD CB d 10
RL (IX+d),C	DD CB d 11	RL (IY+d),C	FD CB d 11
RL (IX+d),D	DD CB d 12	RL (IY+d),D	FD CB d 12
RL (IX+d),E	DD CB d 13	RL (IY+d),E	FD CB d 13
RL (IX+d),H	DD CB d 14	RL (IY+d),H	FD CB d 14
RL (IX+d),L	DD CB d 15	RL (IY+d),L	FD CB d 15
RL (IX+d),A	DD CB d 17	RL (IY+d),A	FD CB d 17

RRC (xr+d),r

Name:	Op-Code	Name	Op-Code
RRC (IX+d),B	DD CB d 08	RRC (IY+d),B	FD CB d 08
RRC (IX+d),C	DD CB d 09	RRC (IY+d),C	FD CB d 09
RRC (IX+d),D	DD CB d 0A	RRC (IY+d),D	FD CB d 0A
RRC (IX+d),E	DD CB d 0B	RRC (IY+d),E	FD CB d 0B
RRC (IX+d),H	DD CB d 0C	RRC (IY+d),H	FD CB d 0C
RRC (IX+d),L	DD CB d 0D	RRC (IY+d),L	FD CB d 0D
RRC (IX+d),A	DD CB d 0F	RRC (IY+d),A	FD CB d 0F

SRA (xr+d),r

Name:	Op-Code	Name	Op-Code
SRA (IX+d),B	DD CB d 28	SRA (IY+d),B	FD CB d 28
SRA (IX+d),C	DD CB d 29	SRA (IY+d),C	FD CB d 29
SRA (IX+d),D	DD CB d 2A	SRA (IY+d),D	FD CB d 2A
SRA (IX+d),E	DD CB d 2B	SRA (IY+d),E	FD CB d 2B
SRA (IX+d),H	DD CB d 2C	SRA (IY+d),H	FD CB d 2C
SRA (IX+d),L	DD CB d 2D	SRA (IY+d),L	FD CB d 2D
SRA (IX+d),A	DD CB d 2F	SRA (IY+d),A	FD CB d 2F

SLA (xr+d),r

Name:	Op-Code	Name	Op-Code
SLA (IX+d),B	DD CB d 20	SLA (IY+d),B	FD CB d 20
SLA (IX+d),C	DD CB d 21	SLA (IY+d),C	FD CB d 21
SLA (IX+d),D	DD CB d 22	SLA (IY+d),D	FD CB d 22
SLA (IX+d),E	DD CB d 23	SLA (IY+d),E	FD CB d 23
SLA (IX+d),H	DD CB d 24	SLA (IY+d),H	FD CB d 24
SLA (IX+d),L	DD CB d 25	SLA (IY+d),L	FD CB d 25
SLA (IX+d),A	DD CB d 27	SLA (IY+d),A	FD CB d 27

SRL (xr+d),r

Name:	Op-Code	Name	Op-Code
SRL (IX+d),B	DD CB d 38	SRL (IY+d),B	FD CB d 38
SRL (IX+d),C	DD CB d 39	SRL (IY+d),C	FD CB d 39
SRL (IX+d),D	DD CB d 3A	SRL (IY+d),D	FD CB d 3A
SRL (IX+d),E	DD CB d 3B	SRL (IY+d),E	FD CB d 3B
SRL (IX+d),H	DD CB d 3C	SRL (IY+d),H	FD CB d 3C
SRL (IX+d),L	DD CB d 3D	SRL (IY+d),L	FD CB d 3D
SRL (IX+d),A	DD CB d 3F	SRL (IY+d),A	FD CB d 3F

TAV's Z 80-Tuning

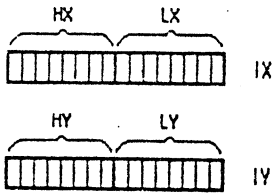
Noch mehr Illegals

In der letzten Ausgabe haben wir erfahren, was "Illegals" sind und wie durch sie die Leistungsfähigkeit der Z80-CPU gesteigert werden kann. Heute soll dieses Wissen ergänzt und neue, bisher unbekannte Befehle vorgestellt werden.

Jedem Z80-Programmierer ist sicherlich bekannt, daß er zwei 8 Bit-Register zu einem 16 Bit-Register zusammenfassen kann, wobei die darin enthaltenen 8 Bit-Register weiterhin getrennt behandelt werden können. In diese sinnvolle Registerorganisation konnten die beiden Indexregister IX und IY bisher leider nicht einbezogen werden.

Vier neue 8 Bit-Register

Durch die Illegals ist es jedoch möglich, die Indexregister aufzuspliten d.h. in jeweils zwei 8 Bit-Register zu zerlegen. Die so gewonnenen 8 Bit-Register bezeichnen wir am besten mit HX bzw. HY und LX bzw. LY.



Wie wir an diesem Schaubild erkennen können, beziehen sich die Low-Byte-Register (L) auf die relativen Bits 0 bis 7 und die High-Byte-Register auf die relativen Bits 8-15. Die acht folgenden Befehlstabellen enthalten alle Op-Codes, die Sie benötigen, um die Indexregister so effektiv wie möglich einzusetzen. Bitte beachten Sie, daß kein Assembler die vor den Op-Codes stehenden symbolischen Befehlsformen verstehen kann. Sie dienen nur zu Ihrer Orientierung.

Lade 8-Bit-Indexregister mit Konstanten

Name:	Op-Code	Name	Op-Code
LD HX,dd	DD 26	LD HY,dd	FD 26
LD LX,dd	DD 2E	LD LY,dd	FD 2E

Lade 8-Bit-Register mit 8-Bit-Indexregister

Name:	Op-Code	Name	Op-Code
LD A,HX	DD 7C	LD A,HY	FD 7C
LD A,LX	DD 7D	LD A,LY	FD 7D
LD B,HX	DD 44	LD B,HY	FD 44
LD B,LX	DD 45	LD B,LY	FD 45
LD C,HX	DD 4C	LD C,HY	FD 4C
LD C,LX	DD 4D	LD C,LY	FD 4D
LD D,HX	DD 54	LD D,HY	FD 54
LD D,LX	DD 55	LD D,LY	FD 55
LD E,HX	DD 5C	LD E,HY	FD 5C
LD E,LX	DD 5D	LD E,LY	FD 5D

Lade 8-Bit-Indexregister mit 8-Bit-Register

Name:	Op-Code	Name	Op-Code
LD HX,A	DD 67	LD HY,A	FD 67
LD LX,A	DD 6F	LD LY,A	FD 6F
LD HX,B	DD 60	LD HY,B	FD 60
LD LX,B	DD 68	LD LY,B	FD 68
LD HX,C	DD 61	LD HY,C	FD 61
LD LX,C	DD 69	LD LY,C	FD 69
LD HX,D	DD 62	LD HY,D	FD 62
LD LX,D	DD 6A	LD LY,D	FD 6A
LD HX,E	DD 63	LD HY,E	FD 63
LD LX,E	DD 6B	LD LY,E	FD 6B

Lade 8-Bit-Indexregister mit 8-Bit-Indexregister

Name:	Op-Code	Name	Op-Code
LD HX,LX	DD 65	LD HY,LY	FD 65
LD LX,HX	DD 6C	LD LY,HY	FD 6C

Additionsbefehle (8-Bit-Indexregister)

Name:	Op-Code	Name	Op-Code
INC HX	DD 24	INC HY	FD 24
INC LX	DD 2C	INC LY	FD 2C
ADD A,HX	DD 84	ADD A,HY	FD 84
ADD A,LX	DD 85	ADD A,LY	FD 85
ADC A,HX	DD 8C	ADC A,HY	FD 8C
ADC A,LX	DD 8D	ADC A,LY	FD 8D

Subtraktionsbefehle (8-Bit-Indexregister)

Name:	Op-Code	Name	Op-Code
DEC HX	DD 25	DEC HY	FD 25
DEC LX	DD 2D	DEC LY	FD 2D
SUB HX	DD 94	SUB HY	FD 94
SUB LX	DD 95	SUB LY	FD 95
SBC HX	DD 9C	SBC HY	FD 9C
SBC LX	DD 9D	SBC LY	FD 9D

Logische Operationen (8-Bit-Indexregister)

Name:	Op-Code	Name	Op-Code
AND HX	DD A4	AND HY	FD A4
AND LX	DD A5	AND LY	FD A5
XOR HX	DD AC	XOR HY	FD AC
XOR LX	DD AD	XOR LY	FD AD

Vergleichs-Operationen (8-Bit-Indexregister)

Name:	Op-Code	Name	Op-Code
CP HX	DD BC	CP HY	FD BC
CP LX	DD BD	CP LY	FD BD

Mit dem Thema "Illegals" sind wir in ein Gebiet eingedrungen, das bis jetzt nur teilweise erforscht worden ist. Es ist also durchaus denkbar, daß die Z80-CPU weitere sinnvolle Befehle beinhaltet und Sie während Ihrer eigenen Programmentwicklungen auf weitere Illegals stoßen. Diesbezügliche Anregungen nehmen wir jederzeit dankend entgegen.

Zum Schluß noch ein kleiner Tip, der nichts mit Illegals zu tun hat: Ein 16 Bit-Register kann in Null Taktzyklen durch 256 geteilt werden! Wie? Rechnen Sie doch einfach mit dem High-Byte des verwendeten Doppelregisters weiter.

(tav)

Op-Code für IX: DD CB of XX
Op-Code für IY: FD CB of XX

Wie Sie sehen, eröffnet uns die 16 Bit High/Low Byte-Manipulation der Indexregister völlig neue Möglichkeiten. In der Praxis erspart uns gerade ihr gezielter Einsatz jede Menge Bytes und Taktzyklen, da aufwendige Shift- und Maskenoperationen in vielen Fällen durch einen einzigen Befehl ersetzt werden können. Außer der direkten High/Low-Byte Manipulation gibt es noch weitere Möglichkeiten, die Indexregister in bisher nicht bekannter Weise einzusetzen. Die zwei folgenden Tabellen beziehen sich auf eine indiziert-indirekte Einzelbitmanipulation mit anschließendem Transfer des Resultats in ein von uns definiertes 8 Bit-Register.

Indeziert-Indirektes Rücksetzen eines Bits mit anschließendem Transfer des Ergebnisses in ein Register

Name:	A	B	C	D	E	H	L
RES 0,(XR+of)	87	80	81	82	83	84	85
RES 1,(XR+of)	8F	88	89	8A	8B	8C	8D
RES 2,(XR+of)	97	90	91	92	93	94	95
RES 3,(XR+of)	9F	98	99	9A	9B	9C	9D
RES 4,(XR+of)	A7	A0	A1	A2	A3	A4	A5
RES 5,(XR+of)	AF	A8	A9	AA	AB	AC	AD
RES 6,(XR+of)	B7	B0	B1	B2	B3	B4	B5
RES 7,(XR+of)	BF	B8	B9	BA	BB	BC	BD

Eine äquivalente Sicherung des Ergebnisses haben wir übrigens im letzten Monat in Bezug auf einige neue Shift- und Rotationsbefehle kennengelernt. Die neuen Einzelbit-Befehle sind sowohl in Verbindung mit dem IX- als auch dem IY-Register anwendbar. Nachfolgend finden Sie zwei entsprechende Op-Code-Gerüste, die sich auf alle in den Tabellen vorhandenen Befehlen beziehen. In diesem Gerüst muß XX durch einen Tabellenwert und of durch eine 8 Bit-Distanzadresse (Offset) ersetzt werden.

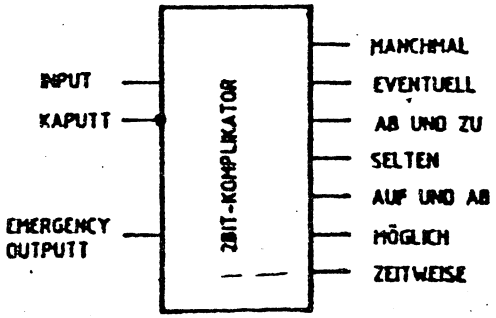
Indeziert-Indirektes Setzen eines Bits mit anschliessendem Transfer des Ergebnisses in ein Register

Name:	A	B	C	D	E	H	L
SET 0,(XR+of)	C7	C0	C1	C2	C3	C4	C5
SET 1,(XR+of)	CF	C8	C9	CA	CB	CC	CD
SET 2,(XR+of)	D7	D0	D1	D2	D3	D4	D5
SET 3,(XR+of)	DF	D8	D9	DA	DB	DC	DD
SET 4,(XR+of)	E7	E0	E1	E2	E3	E4	E5
SET 5,(XR+of)	EF	E8	E9	EA	EB	EC	ED
SET 6,(XR+of)	F7	F0	F1	F2	F3	F4	F5
SET 7,(XR+of)	FF	F8	F9	FA	FB	FC	FD

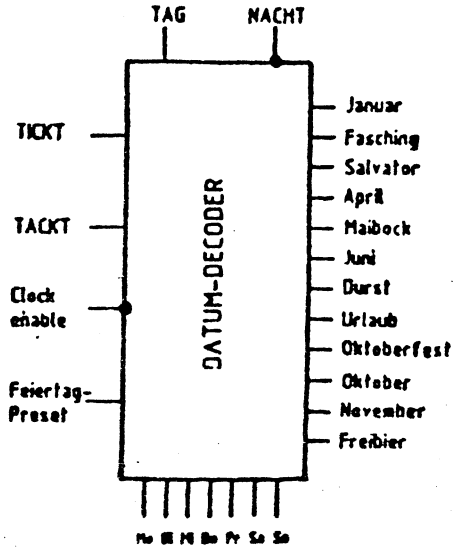
CLUB 80 HARD - - - - CLUB 80 HARD

Weitere neue IC's:
Sicher habt Ihr auch gleich die entsprechende Beschaltung dazu gefunden. Viel Spaß beim Löten und Ausprobieren.

Der 2Bit-Komplikator



Datum-Decoder



Jens Neueder

Hardware - Liste CLUB 80

Nun folgend erscheint die schon lange angekündigte Hardware - Liste unserer Mitglieder.

Wie schon mehrmals angedeutet ist sie, wegen mangelnder Beteiligung von Eurer Seite aus, unvollständig. Der Stand dieser Datenauflistung ist der Dezember 1986. Ich möchte Euch nun herzlich bitten, änderungen, die Eure Geräte-Konfiguration betreffen, an mich weiterzuleiten. Ich hoffe, daß ich doch eines Tages einmal eine vollständige und aktuelle Liste bieten kann. Einstweilen soll und muß Euch dieses Werk genügen.

Die Liste wurde in folgende Punkte aufgeteilt:

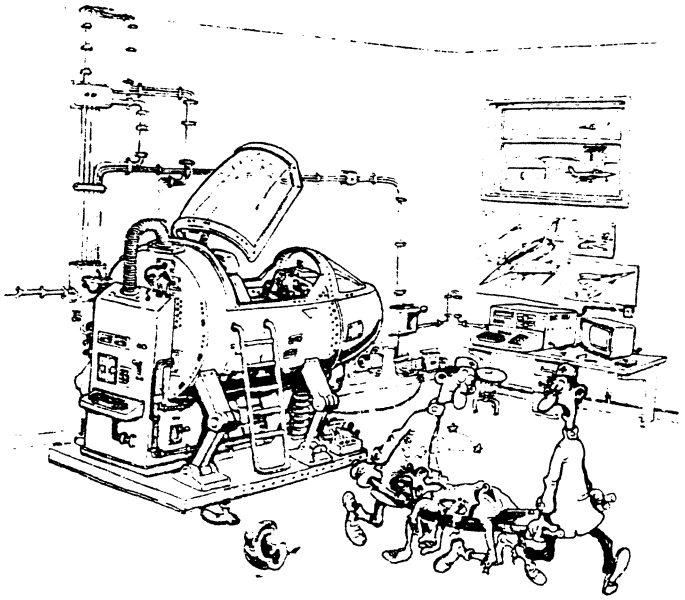
Punkt 1 ... Computer -- hierzu sind sicher keine weiteren Erklärungen nötig.
2 ... Drucker -- Die STAR- und die Epson-Drucker (FX,MX,RX) dürften im Normalbetreib kompatibel sein. Gleiches gilt auch für die ITOH und NEC'ler. Zu den weiteren Typen kann ich keine Aussage treffen.
3 ... Laufwerke -- Die Anzahl der Laufwerke wurde in diese Aufstellung vernachlässigt. Auch wurde nur das jeweils vielseitigere Laufwerk angegeben.
4 ... CP/M -fähig
5 ... HRG -- 384 x 192 entspricht der HRG 1B, 640 x 240 läuft auf dem M4
6 ... Sonstiges -- hier werden weiter zusätzliche Geräte aufgezählt

1. Computer

VG I : Herbert Albers, Helmut Bernhardt, Dieter Kasper, Holger May, Waldemar Misioch, Kurt Müller, Richard Rensch Bernd Retzlaff, Paul-Jürgen Schmitz, Walter Schäfer, Hans-Martin Stephan
VG II : Manfred Held, Rainer Jablotschkin, Waldemar Misioch, Peter Spieß, Jürgen Wucherer
VG IIs : Gerald Schröder
VG III : Richard Rensch, Helmut Bernhardt
VG IIIs : Arnulf Sopp
TRS 80 MI L2 : Ulrich Böckling, Werner Föster, Josef Konrad, Karl-Herbert Krüger, Eckehard Kuhn, Klaus-Jürgen Mühlenbein, Jens Neueder, Hans Raggan, Andreas Rychlik
Tandy M III : Helmut Emmerich
TRS 80 M4 : Rüdiger Sörensen
TRS 80 M4p : Gerald Dreyer, Klaus Hermann, Hartmut Obermann, Dieter Reichelt, Peter Stevens, Günther Wagner
ECB-Bus-TRS 80: Bernd Drowälder
Komtek 1 : Helmut Bernhardt
mc CP/M : Helmut Bernhardt
C 64 : Peter Stevens

2. Drucker

STAR DP-510 : Herbert Albers, Rainer Jablotschkin, Jürgen Wucherer
STAR SG-10 : Dieter Kasper
STAR Gemini 10x : Holger May, Gerald Schröder, Walter Schäfer, Hans-Martin Stephan
FX 80 : Peter Stevens
MX 80 : Waldemar Misioch
MX 80 FT : Gerald Dreyer, Karl-Herbert Krüger
RX 80 : Werner Förster
RX 80 F/T : Klaus-Jürgen Mühlenbein, Rüdiger Sörensen
RX 80 F/T + : Hartmut Obermann
ITOH 8510 A: Helmut Bernhardt, Ulrich Böckling, Richard Rensch
NEC 8023B-C : Jens Neueder, Peter Spieß
NEC P 6 : Paul-Jürgen Schmitz, Arnulf Sopp
HEATH H14 : Kurt Müller
KX-P1091 : Bernd Drowälder
LP VI : Günther Wagner
LP VIII : Helmut Emmerich
TA DRH 80 : Manfred Held, Josef Konrad
QD DMP-1180 : Eckehard Kuhn
CP 80 : Hans Raggan
TCS Printstar 10 : Dieter Reichelt



Diese computergesteuerten Flugsimulatoren sind ein bißchen zu realistisch – besonders bei Bruchlandungen.

3. Laufwerke

Typ 5 1/4 Zoll
40 SS SD : Holger May, Hans Raggan
SS DD : Werner Förster, Bernd Retzlaff
DS SD : -
DS DD : Gerald Dreyer, Klaus-Jürgen Mühlenbein, Hans-Martin Stephan
80 SS SD : -
SS DD : -
DS SD : -
DS DD : Helmut Bernhardt, Ulrich Böckling, Bernd Drowälder, Manfred Held, Dieter Kasper, Josef Konrad, Karl-Herbert Krüger, Waldemar Misioch, Jens Neueder, Hartmut Obermann, Dieter Reichelt, Richard Rensch, Andreas Rychlik, Paul-Jürgen Schmitz, Gerald Schröder, Walter Schäfer, Arnulf Sopp, Peter Spieß, Günther Wagner

Stringy-Floppy : Jens Neueder
Typ 3 Zoll : Bernd Drowälder
Typ 3 1/2 Zoll : Jens Neueder

SA 405 : Herbert Albers
SD 521 : Herbert Albers
TEAC-55FV-13 : Rainer Jablotschkin
TEAK 55 A+B : Kurt Müller

Hardware - Liste CLUB 80

4. CP/M -fähig

Böckling	Ulrich
Dreyer	Gerald
Drowälder	Bernd
Förster	Werner
Held	Manfred
Kasper	Dieter
Krüger	Karl-Herbert
Reichelt	Dieter
Retzlaff	Bernd
Rychlik	Andreas
Schmitz	Paul-Jürgen
Sopp	Arnulf
Stevens	Peter
Sörensen	Rüdiger
Wagner	Günther

5. Grafik

HRG - Pixelauflösung

384 x 192

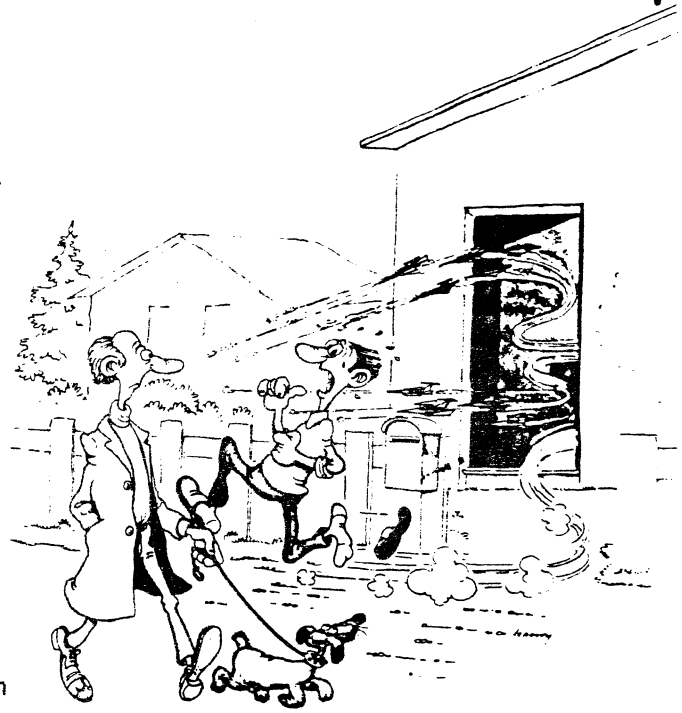
Bernhardt	Helmut
Böckling	Ulrich
Drowälder	Bernd
Förster	Werner
Jablotschkin	Rainer
Krüger	Karl-Herbert
Kuhn	Eckehard
Misioch	Waldemar
Mühlenbein	Klaus-Jürgen
Müller	Kurt
Neueder	Jens
Rychlik	Andreas
Spieß	Peter

640 x 240

Dreyer	Gerald

480 x 192

Schröder	Gerald



Die neueste Version vom „Krieg der Sterne"!

6. Weiteres Zubehör

Bernhardt	Helmut	... Banker (256K RAM /ROM/RAM) I/O/RAM /A/D-Wandl	256 x 512 HRG v. Garf/Kempten
Böckling	Ulrich	... 3.4 MHz /Inverse Zeichendarstellung	Basiccode /Supertape
Dreyer	Gerald	... RS 232 /Telefax-Gerät	
Drowälder	Bernd	... Eprommer /Meßwertaufnahme /usw.	
Förster	Werner	... EM 2005 /80 Zeichenkarte	
Held	Manfred	... Grip II /Prommer 80 /Spooler /V24	
Kasper	Dieter	... 3.54 MHz	
Krüger	Karl-Herbert	... Funk mit RTTY	
Kuhn	Eckehard	... Stringy-Floppy /80-Zeichenkarte /RS 232	/Nixdorfkugelkopfdr.
May	Holger	... Joystick /Siemens-Fernschreiber T100S	
Mühlenbein	Klaus-Jürgen	... Recorder /Verstärker	
Neueder	Jens	... Stringy-Floppy /Joystick /	
Rensch	Richard	... Doubler /Expander /STAR RADIX 10	
Rychlik	Andreas	... Eprommer /Lightpen /Supertape /Basiccode	3.54 MHz /Video-Snovel /ECB-Bus /RS232
Schröder	Gerald	... 80-Zeichenkarte	
Sopp	Arnulf	... div. Schnittstellen /Banker /Prommer-80	VG I
Spieß	Peter	... Prommer /Spooler 64K /Grafikkarte 512 x 512	EG 64 /Typenradschreibmaschine
Stevens	Peter	... Olivetti ETIII Typenradschreibmaschine	
Wucherer	Jürgen	... Olivetty Typenradschreibmashine .	

Selbstbau Doubler für EXP1

Helmut Bernhardt

Nachdem sich bei mir nun schon der zweite WD1771 Single Density Floppycontroller verabschiedet hat und der Chip wohl auch bald nicht mehr zu haben ist, entschloß ich mich, mir selbst einen Doubler zu bauen, der ohne diesen Käfer auskommt. Es ist auch nicht einzusehen, warum auf einem Doubler zwei Floppycontroller sitzen müssen, wo doch der WD1791 alles das, was der WD1771 kann (nämlich beim Booten die erste Spur der Diskette in Single Density und gelegentlich auch mal ganze Disketten in SD zu lesen oder zu schreiben, wenn ein unterausgerüsteter Zeitgenosse den digitalen Informationsaustausch nur in diesem Format leisten kann) nebenbei auch erledigen könnte.

Letzteres Argument soll nun aber auch gegenstandslos werden, denn den hier beschriebenen Doubler kann sich leicht jeder selbst bauen und auch finanzieren. Grundlage ist der WD2793 Floppycontroller, der schon aufgrund seiner Bescheidenheit bei der Spannungsversorgung (ausschließlich +5V und GND) mein Wohlwollen verdient hat. Außerdem enthält er auf dem Chip schon einen PLL-Datenseparator und eine Schreibvorkompensation. Dadurch können die meisten ICs auf den Doublern von TCS und dem davon abgekupferten von RB entfallen, die nur diesen Job leisten.

Der WD2793 kann intern den an Pin 24 angelegten Takt von 2MHz (für den Betrieb von 8" Floppies) auf 1MHz (5" Floppies) runterteilen, wenn an den Pins 17 und 25 Low-Pegel angelegt wird. über Pin 37 läßt sich vorgeben, ob in Single Density (high) oder Double Density (low) gelesen und geschrieben werden soll.

Durch die 5 zusätzlichen ICs links neben dem WD2793 auf dem Doubler Board lassen sich diese Features programmieren. Zwei Flip Flops geben über ihre Q-Ausgänge die Pegel für die Pins 17, 25 und 37 vor. Nach einem RESET ist der Floppycontroller auf Minifloppy und Single Density eingestellt. Durch Ausgabe folgender Werte an das Contollregister mit der memory mapped Adresse 37ECH bzw. das Sektorregister 37EEH lassen sich die Betriebszustände folgendermaßen einstellen:

37ECH:	1111 1xx0 bin (FFH)	Single Density
	1111 1xx1 bin (FEH)	Double Density
37EEH	1000 0xx0 bin (80H)	Mini Floppy (5 1/4")
	1100 0xx0 bin (C0H)	Maxi Floppy (8")

Diese Daten haben im Contoll- bzw. Sektorregister keine Wirkung. Aus den Daten und der Schreibadresse werden nur extern die Triggersignale für die Flip Flops decodiert, wobei für die Density D0 und für die Drive-Größe D6 den einzustellenden Pegel vorgibt.

Diese Schaltung ist in Abb.1 gezeigt. Das einseitige Layout und die Anordnung der Drahtbrücken und Bauteile ist aus den folgenden Abbildungen zu ersehen. Die zur Position des WD2793 leicht versetzte Dil40-Anordnung von Lötpunkten wird von der Lötseite mit einem entsprechenden Stecker bestückt, der in den Sockel des WD1771 auf dem EXP1 gesteckt wird. Zur Zugentlastung dieses Steckers sollten an ca. 10 der durchragenden Pins auf der Bestückungsseite kurze Drahtstücke angelötet werden. Das verhindert ein eventuelles Abreißen der Leiterbahnen beim Herausziehen des Doublers aus dem 1771-Sockel.

Beim Einsatz dieses Doublers muß der invertierende Datentreiber 74LS640 neben dem 1771-Sockel durch den nichtinvertierenden pinkompatiblen Typ 74LS245 ersetzt werden, da der WD2793 keine invertierenden Daten-Ein-/Ausgänge (wie WD1771 und WD1791) hat. Die Entscheidung fiel für den WD2793, der sich nur in seinen nichtinvertierenden Dateneingängen vom WD2791 unterscheidet, wegen unüberwindlicher Beschaffungsschwierigkeiten beim WD2791. Das bedeutet aber leider auch, daß dieser Doubler bei anderen Floppycontroller-Boards nicht verwendet werden kann, da dort nicht einfach ein Treiber ausgewechselt werden kann.

Auch ein weiterer Nachteil dieses Doublers soll noch erwähnt werden. Für die einwandfreie Funktion von Datenseparator und Schreibvorkompensation sind einige Trimmungen an den Potentiometern P1 und P2 und dem Trimmkondensator C1 vorzunehmen. Dazu ist ein Oszilloskop nötig. Wenn das nicht verfügbar ist, kann im Notfall auch mit Probieren getrimmt werden. Dann werden P1, P2 und C1 auf Mittelstellung gedreht und durch wiederholte Bootversuche und ständiges Verändern der Einstellung von P2 und C1 die günstigste Einstellung für das Lesen einer Diskette gefunden. Mit P1 muß dann noch die Schreibvorkompensation geregelt werden, was durch Schreiben auf Diskette und anschließendes Lesen geprüft werden kann. Die Enstellungen sind nicht sehr kritisch, so daß auch dieses Vorgehen möglich ist.

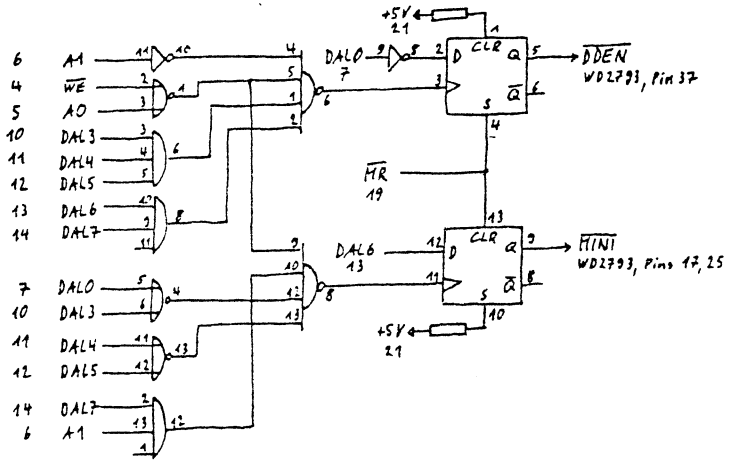
Die im Datenblatt vorgegebene Einstellung mit Hilfe eines Oszilloscops läuft folgendermaßen ab:

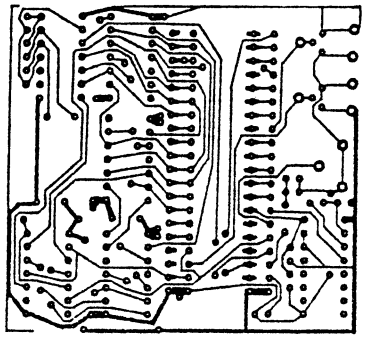
Der Computer wird mit gedrückter <BREAK> Taste eingeschaltet, um im Level II BASIC zu landen. Es wird der Befehl POKE 14316,255 (auf DD schalten) ausgeführt. Dann werden mit einem Jumper die beiden Stifte "TEST" kurzgeschlossen (Pin 21 des WD2793 an GND). Mit P1 kann dann die Breite der Pulse an Pin 31 (WD) unter Oscarkontrolle auf einen Wert zwischen 100 und 300ns eingestellt werden.

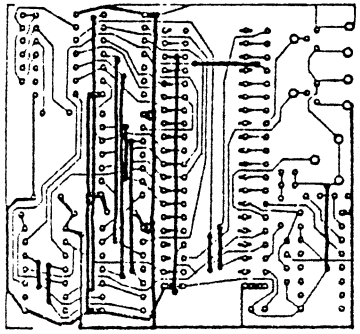
Mit P2 wird dann die Pulsbreite an Pin 29 (TG43) auf 60-65ns eingestellt. Mit C1 wird an Pin 16 (DIRC) eine Frequenz von 250kHz eingestellt.

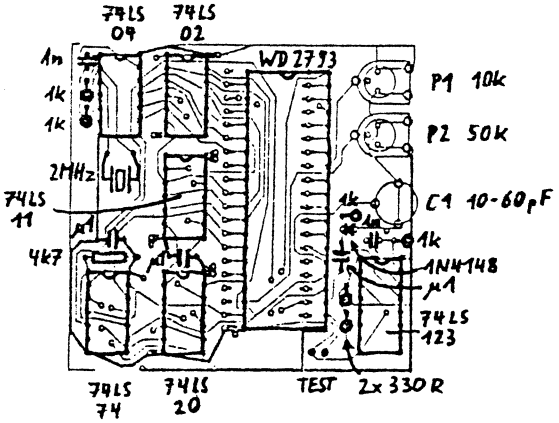
Um die Hardware-Voraussetzungen, auch 8"-Floppies betreiben zu können, zu nutzen, muß die Anordnung der Signale auf dem 34poligen Shugart-Bus für Minifloppies auf die für den 50poligen Bus für 8"-Floppies umsortiert werden. Bei älteren 8"-Drives muß diesen zusätzlich noch das Signal TG43 (WD2793, Pin29) über einen Open Collector Inverter zugeführt werden. Neuere 8"-Drives benötigen dieses Signal nicht mehr.

Wer sich diesen Doubler aufbauen will, kann für 10,-DM (incl. Rückporto) von mir ein geätztes, ungebohrtes Board bekommen und darf sich mit der Beschaffung der Bauteile, dem Bohren, Löten und Trimmen selbst abmühen.

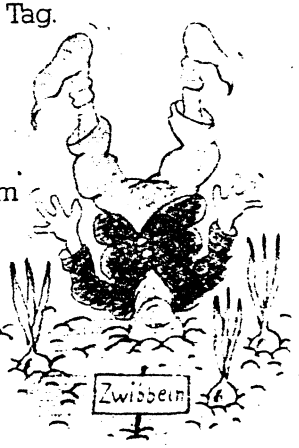








Jiddisch: A nechtige tug.
Ein nächtiger Tag.
Das wird nie passieren.



Di solst
wachsen wi
a zibbele mitm
kop in drerd.
Du sollst
wachsen wie
eine Zwiebel
mit dem Kopf
in der Erde.
Hol dich
der Teufel.

Rumänisch: La pastele cailor.
Wenn die Pferde Ostern feiern.
Wenn Ostern und Pfingsten auf einen Tag fallen.

Max Frankl, Hofolding

Spanisch: Hizo la vista gorda.
Er machte die Augen dick.
Er drückte ein Auge zu.

Echó la soga tras el calder.
Er warf den Strick hinter dem Schöpfeimer her.
Er warf die Flinte ins Korn.

Edelgard Moers, Darsten

N3: Minifloppy-Stecker

Belegung:				Funktion:	
01,02:	GND	/READY,/MINI	:	GND:	Signalmasse
03,04:	GND	/HOLD	:	/DS0-3:	Laufwerkauswahl
05,06:	GND	/DS3	:	/HOLD:	Kopf laden
07,08:	GND	/INDEX	:	/MOTOR ON:	Motor ein
09,10:	GND	/DS0	:	/DIR:	Spurwechsel-Richtung
11,12:	GND	/DS1	:	/STEP:	Spurwechsel-Impuls
13,14:	GND	/DS2	:	/WR DATA:	Schreibdaten
15,16:	GND	/MOTOR ON	:	/WR GATE:	Schreiben ein
17,18:	GND	/DIR	:	/SIDE:	Kopfauswahl
19,20:	GND	/STEP	:	/MINI:	Mini/Maxi-Umschaltung
21,22:	GND	/WR DATA	:		
23,24:	GND	/WR GATE	:	/READY:	Laufwerk bereit
25,26:	GND	/TRACK 0	:	/INDEX:	Indexloch-Impuls
27,28:	GND	/WR PT	:	/TRACK 0:	Kopf auf Spur 0
29,30:	GND	/RD DATA	:	/WR PT:	Schreibschutz ein
31,32:	GND	/SIDE	:	/RD DATA:	Lesedaten
33,34:	GND	/READY,/DCHG	:	/DCHG:	Diskettenwechsel

N4: Maxifloppy-Stecker

Belegung:				Funktion:	
01,02:	GND	/LOW CUR	:	GND:	Signalmasse
03,04:	GND	/MOTOR OFF	:		
05,06:	GND	--	:	/DS0-3:	Laufwerkauswahl
07,08:	GND	--	:	/HOLD:	Kopf laden
09,10:	GND	/TWO SIDED	:	/MOTOR OFF:	Motor aus
11,12:	GND	/DCHG	:	/DIR:	Spurwechsel-Richtung
13,14:	GND	/SIDE	:	/STEP:	Spurwechsel-Impuls
15,16:	GND	--	:	/WR DATA:	Schreibdaten
17,18:	GND	/HOLD	:	/WR GATE:	Schreiben ein
19,20:	GND	/INDEX	:	/SIDE:	Kopfauswahl
21,22:	GND	/READY	:	/LOW CUR:	Schreibstrom niedrig
23,24:	GND	--	:		
25,26:	GND	/DS0	:		
27,28:	GND	/DS1	:	/READY:	Laufwerk bereit
29,30:	GND	/DS2	:	/INDEX:	Indexloch-Impuls
31,32:	GND	/DS3	:	/TRACK 0:	Kopf auf Spur 0
33,34:	GND	/DIR	:	/WR PT:	Schreibschutz ein
35,36:	GND	/STEP	:	/RD DATA:	Lesedaten
37,38:	GND	/WR DATA	:	/DCHG:	Diskettenwechsel
39,40:	GND	/WR GATE	:	/TWO SIDED:	Zweiseitige Diskette
41,42:	GND	/TRACK 0	:		
43,44:	GND	/WR PT	:		
45,46:	GND	/RD DATA	:		
47,48:	GND	--	:		
49,50:	GND	--	:	-- = frei	

Tandy <-> Schneider

Vielen mag die oben angedeutete Kombination zwischen Tandy- und Schneider-Computern auf den ersten Blick etwas seltsam erscheinen, aber wenn man die Sache einmal genauer besieht, bekommt sie auch Sinn.

Gemeinsames
- Sowohl der gute "alte" Tandy (Model 1 - 4) als auch der Schneider CPC 464/664/6128/Joyce arbeitet mit einer Z80-CPU.
- Die Schneider-CPC-Modelle sind, sobald sie mit einer Diskstation ausgerüstet sind, CP/M-fähig und auch die Tandys können, nach gewissen Umbauten (4/4p direkt) dieses Betriebssystem benutzen.

Diese beiden Umstände machen den Tausch von Programmen und Daten zwischen den beiden Computertypen interessant. Behindert wird die Sache dabei dadurch, daß die CPC-Modelle normalerweise mit 3"-Diskettenlaufwerken ausgerüstet sind, die bei Tandy-Besitzern (und nicht nur dort) nicht gerade weit verbreitet sind.

Möglichkeiten
Dieses Hindernis wäre z.B. dadurch zu umgehen, daß der Daten und Programmaustausch über eine RS232-Verbindung abgewickelt wird. Aber auch diese ist beiweitem nicht bei allen CPC- und Tandygeräten vorhanden!
Die zweite Möglichkeit besteht darin, ein 3"-Laufwerk an einen Tandy anzuschließen. Diese Laufwerke sind z.Z. schon für knapp 100 DM (40/SS/SD) mit shugartkompatiblem Bus zu haben. Natürlich kann man auch das original Schneiderlaufwerk anschließen, muß dabei aber einiges beachten (siehe weiter unten)!
Als letzte Möglichkeit bietet sich ein Anschluß "normaler" Drives (5 1/4 Zoll) an den Schneider an. Für diese Möglichkeit, spricht vor allem, daß man dann nicht die doch recht teuren 3-zölligen Scheibchen verwenden muß.

5 1/4" Drive mit Shugart-Standardbus am Schneider CPC 464
Billige 5 1/4"-Laufwerke bekommt man schon zu Preisen um 150,- DM (z.B. Laufwerke aus IBM-kompatiblen, die gegen ein Festplattenlaufwerk oder einen 80-Track-Drive getauscht wurden). Diese Laufwerke kann man, nach einer kleinen Modifizierung, ohne Probleme am Schneider betreiben (alle folgenden Angaben beziehen sich auf einen CPC 464 mit Controller und 3"-Drive von Schneider). Hier zunächst einmal die Belegung des Schneider-Diskinterface im Gegensatz zum Shugart-Bus:

Pin-Nr.	Shugart-Bus	CPC-Diskinterface
1	GND	/Drive Ready
2	NC (bei neueren Laufwerken /Drive 3 select)	GND
3	GND	/Backside select
4	NC	GND
5	GND	/Read Data
6	NC	GND
7	GND	/Write Protect
8	/Index Pulse	GND
9	GND	/Track zero
10	/Drive 0 select	GND
11	GND	/Write Gate
12	/Drive 1 select	GND
13	GND	/Write Data
14	/Drive 2 select	GND
15	GND	/Step
16	/Motor On	GND
17	GND	/Direction select
18	/Direction select	GND
19	GND	/Motor On
20	/Step	GND
21	GND	+ 5 Volt !!!
22	/Write Data	GND
23	GND	/Drive 1 select
24	/Write Gate	GND
25	GND	/Drive 0 select
26	/Track zero	GND
27	GND	/Index Pulse
28	/Write Protect	GND
29	GND	+ 5 Volt !!!
30	/Read Data	GND
31	GND	+ 5 Volt !!!
32	/Backside select	GND
33	GND	+ 5 Volt !!!
34	NC (bei neueren Laufwerken /Drive Ready)	GND

Erklärungen: NC = Not Connected (nicht angeschlossen)
GND = Ground (Masse)
/ = Signal ist low-aktiv (Beispiel: /Drive 0 select ist low-aktiv d.h., das Laufwerk ist angewählt, wenn das Signal auf null Volt liegt.)

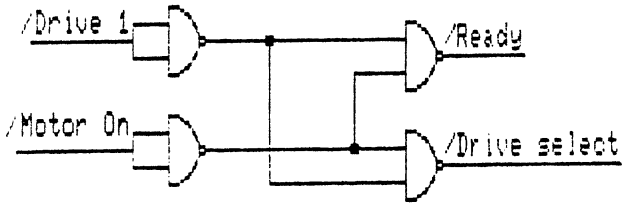
Wenn man die Signale der beiden Bussysteme einmal vergleicht, stellt man fest, daß sie beide die gleichen Signale verwenden. Was man aus dieser Tabelle nur relativ schwer ableiten kann, ist der Umstand, daß der Schneider-Floppybus nicht weiter als ein um 180 Grad gedrehter Shugart-Bus ist! Will man ein "normales" Laufwerk an diesen Bus anschließen, muß man einfach einen entsprechenden Edgecardconnektor (34-poliger Platinenstecker) zusätzlich auf das Schneider-Floppykabel aufpressen und diesen dann um 180 Grad verdreht (Pin 1 des Steckers auf Pin 34 des Laufwerks) aufstecken. So einfach ist das!

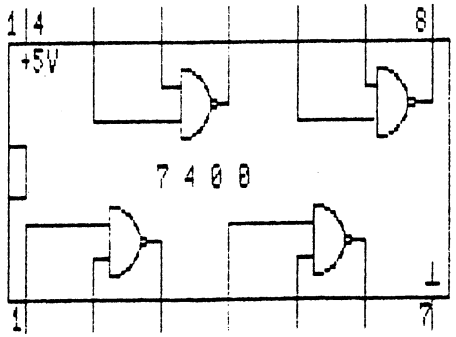
Ganz so einfach ist es aber doch nicht. Nachdem man der Floppy noch eine Stromversorgung verpaßt hat und das ganze ausprobieren will, stellt man zuerst einmal erschreckt fest, daß das zusätzliche Laufwerk dauernd selektiert zu sein scheint. Auf alle Fälle leuchtet die rote LED an der Frontblende ständig und dies, obwohl noch gar kein Floppyzugriff (weder auf 0 noch auf 1) erfolgt ist. Der Grund dafür findet sich in der etwas sparsam angelegten Schaltung des Schneidercontrollers. Das Signal /Drive 1 select wird nämlich durch eine einfache Negierung von /Drive 0 select erzeugt und damit ist Laufwerk 1 immer dann ausgewählt, wenn Drive 0 nicht selektiert ist! D.h., solange Drive 0 selektiert ist (z.B. bei DIR oder beim Laden eines Programms) erlischt die LED am Laufwerk 1 und geht sofort wieder an, nachdem die Schreib- oder Leseoperation beendet ist. Sehr schnell kommt man dahinter, daß dies nur ein unbedeutender Schönheitsfehler ist, da ja nur die LED leuchtet, der Motor sich aber nur dreht, nachdem auch das "/Motor On"-Signal gesendet wird.

Also ohne Bedenken weiterprobiert und im CP/M einfach mal "B:" eingeben, sprich Laufwerk 1 ansprechen. Denkste sagt der CPC, behauptet "Drive B not ready!" und er meint das ganz ernst! Dabei stimmt mit dem Laufwerk doch alles (Betriebsspannungen, Umdrehungsgeschwindigkeit usw.) und am Tandy läuft es einwandfrei. Woran kann es nur liegen, daß der CPC das Laufwerk auch nach viel Zureden nicht akzeptiert!? Das Problem stellt sich (wie so oft) als recht simpel heraus. Der Schneidercontroller möchte vom Laufwerk, nachdem dieses angesprochen wurde, auf dem Pin 1 (/READY) ein LOW-Signal geliefert bekommen, welches ihm sagt: "Laufwerk vorhanden und alles in Ordnung!". Genau dieses Signal liefern (abgesehen von neueren Laufwerken) die 5 1/4"-Drives nicht! Man könnte natürlich die Leitung ständig auf Masse legen und den Controller damit täuschen. Da aber das 3"-Laufwerk dieses Signal ordnungsgemäß liefert, käme diese Maßnahme einem Abbau der Betriebssicherheit gleich.

Abhilfe für beide Probleme schafft die unten gezeigte, kleine Schaltung. Die Signale /Drive 1 und /Motor On werden invertiert und dann NAND-verknüpft. Das erzeugte Signal ist immer nur dann low, wenn sowohl /Drive 1 und /Motor On aktiv (low) sind. Die NAND-Verknüpfung ist zweimal ausgeführt, um Rückwirkungen zwischen /READY- und /Drive select-Signal zu vermeiden. Das erzeugte /READY-Signal wird mit Pin 34 am Floppyanschluß verbunden; /Drive select wird dort angeschlossen, wo normalerweise der Jumper/DIP-Schalter sitzt, mit dem festgelegt wird, mit welcher Nummer das Laufwerk angesprochen wird (der Jumper muß dann natürlich entfallen bzw. der DIP-Schalter auf OFF gestellt werden!).

Mit dieser kleinen Zusatzschaltung
kann man ein "normales" Laufwerk
am Schneider CPC 464/664/6128 -
Disk-Interface betreiben!





Nach dieser kleinen Modifikation kanns losgehen. Zunächst einmal muß man eine Diskette in Laufwerk B formatieren. Nichts einfacher als das, denkt sich der (vermeintlich) CP/M-Erfahrene und ruft "FORMAT" auf. Seltsamerweise fehlt die gewohnte Frage, in welchem Laufwerk formatiert werden soll. Also Programm abbrechen, und "FORMAT B:" eingeben; ein Parameter-Error ist der Erfolg. Im Handbuch kann man dann nachlesen, daß man mit dem mitgelieferten Formatierungsprogramm nur Disketten in Drive A in die gewünschte Form bringen kann! Schei.....!!! Zum Glück mißfiel diese Einschränkung schon einem kleveren Assemblerprogrammierer, der fix eine Erweiterung zu FORMAT verfasste. Diese braucht man nur einzutippen (etwa eineinhalb Seiten Hexdump, wer das Programm für sich oder einen Bekannten benötigt, kann es bei mir gegen Einsendung einer 3"-Disk und Rückporto bekommen) und nun kanns wirklich losgehen.

Formate

Der CPC beherrscht mehrere verschiedene Formate, darunter das CP/M 86-Format des IBM-PC. Dieses bietet zwar nur 156k freien Diskettenplatz, ist aber als Tauschformat gut geeignet, da es von vielen anderen CP/M-Computern (u.a. den Tandy's) ohne Probleme bearbeitet werden kann. Wer aber auch das eigentliche CPC-Format benutzen möchte, kann es mit folgenden Informationen leicht auf seiner Maschine implementieren:

Schneider CPC 464 Systemdiskette

Tracks	40
Sides	1
Density	Double
Capacity	171k
Records per Track	36
Block Shift Count	3
Block Mask	7
Extent Mask	0
Disk Storage Maximum (größte Block-Nr. -1)	170
Directory Maximum (Anzahl der Einträge -1)	63
Allocation Vector 0	192
Allocation Vector 1	0
Directory Check Size	16
Track Offset (Anzahl der Systemspuren)	2
Sectors per Track (Anzahl der phys. Spuren)	9
Sector Size (Byte)	512

Reihenfolge, in der die (9) Sektoren gelesen werden (sog. Interleaving-Tabelle): 65,66,67,68,69,70,71,72,73

Für das Model 4 und Montezuma-CP/M muß mit einem Editor folgender Datensatz erstellt und in das ASCII-File DISK.FDF eingebunden werden: *Schneider CPC 464 (40T, SS, DD, 171K)
36,3,7,0,170,63,192,0,16,2,9,2,40,128
65,66,67,68,69,70,71,72,73

Danach kann man direkt und ohne weitere Probleme Daten mit dem CPC austauschen. Auch Programme lassen sich ohne weiteres transferieren, ob sie später auf der Zielmaschine auch laufen ist eine andere Frage, die ich hier nicht weiter ansprechen will.

3"-Schneider-Drive am Shugartbus

Wie wir schon gesehen haben, bereitet das Betreiben eines 5 1/4 Zoll-Laufwerkes am Schneider, von ein paar anfänglichen Schwierigkeiten abgesehen, keine weiteren Probleme. Wenn man aber kein zusätzliches Laufwerk für den CPC anschaffen möchte, kommt nur noch der Anschluß des "Schmalspurlaufwerks" an den Shugart-Bus in Frage.

Dieser bereitet noch weniger Probleme als das umgekehrte Verfahren. Man muß nur einen kleinen Adapter basteln, da das 3"-Laufwerk einen Pfostenfeldstecker als Anschluß benötigt und das direkte Aufpressen eines solchen nicht ohne weiteres möglich ist. Beim Erstellen des Adapters ist auf folgende Punkte zu achten:

1. die Leitungen 1-33 des Shugartbus können direkt mit 2-34 der 3-Zoll-Floppy verbunden werden.

2. Die Anschlüsse 21,29,31 und 33 am Floppy führen 5 Volt und dürfen nicht angeschlossen werden!

3. Das /READY-Signal (Pin 1 bzw. 34) sollte nicht angeschlossen werden.

4. Am Schneiderlaufwerk besteht nicht die Möglichkeit, durch Jumper oder DIP-Schalter die Nummer des Drives zu ändern. Aus diesem Grund muß der Anschluß 25 (Drive 0 select) des Laufwerks mit dem gewünschten Selectsignal des Shugartbus (10, 12, 14 oder 2) direkt verbunden werden. Die nicht benötigten Selectsignale sollten nicht angeschlossen werden.

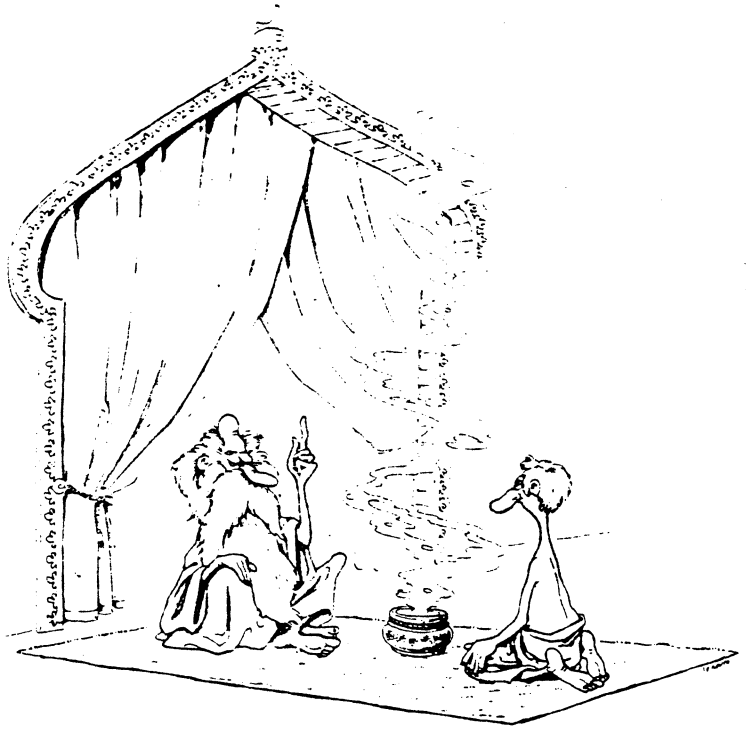
5. Die restlichen Signale können laut der oben abgedruckten Tabelle direkt verdrahtet werden.

Probleme mit dem 3"-Drive am 4p gibt es nicht. Eventuell kann es notwendig sein, im Betriebssystem die Steprate auf 12 ms heraufzusetzen, falls es öfters zu Schreib/Lesefehlern kommen sollte. Natürlich kann man das Laufwerk mit allen Betriebssystemen benutzen und auch alle möglichen Formate definieren (z.B. NewDOS 40/SS/DD). Tauschen kann man aber nur in den beiden oben schon erwähnten CP/M-Formaten.

Quellenhinweise:
De Snider un sin Disk (Detlef Grell in c't 6/85)
CP/M-Übungbuch für CPC 464/664/618 (Data Becker)
Betriebssystem CP/M (Plate / Franzis Verlag)

Sollte jemand Fragen zum Thema "Tandy <-> Schneider" haben, bin ich gerne bereit sie zu beantworten und, wo Probleme auftreten, zu helfen wo ich kann. Viel Spaß beim Löten und "schneidern"

Hartmut Obermann



Vernichte Dein Ego! Zerstöre die Illusionen der Wirklichkeit! Verbanne den Traum irdischen Vergnügens! Eine halbe Stunde mit unserem EDV-Rechnungsprogramm sollte dafür ausreichen.

Nachtrag zu "Dein G3s, die 4 unbekannten Wesen"

Wir mußten in diesem Artikel ratlos zugeben, daß die Vervierfachung des Graphikspeichers nicht geklappt hatte. *Hatte!* Sie klappt jetzt. Schuld war ein unbemerkter Lötzinntropfen, der den Pin 1 der RAMs (gemultiplexte Adressen A16 und A17) per Kurzschluß auch noch mit Refreshes beglückte.

Die Schaltung ist also vollkommen in Ordnung und kann nachgebaut werden. Man sollte aber etwas vorsichtiger löten als wir. Es lohnt sich: Vielleicht sind 8 Graphikseiten für alberne Bildchen wirklich überflüssig. Aber da es sich um gewöhnlichen RAM-Speicher handelt, ist der Zugewinn von 256 kB durchaus attraktiv. Welcher "Kompatible" kann schon mit 1 1/4 MB RAM, gleichzeitig 4 bzw. 8 Bildschirmen und 4 Zeichensätzen aufwarten?

Helmut Bernhardt
Arnulf Sopp

- I M P R E S S U M -

1. Vorsitzende
Peter STEVENS
Postfach 56
4600 Dortmund 1
☎ 0231 /593883

2. Vorsitzende
Hartmut OBERMANN
Schwalbacher Straße 6
6209 Heidenrod 1
☎ 06124 /3913

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

Diskothekar
Klaus-Jürgen MÜHLENBEIN
Am Mönchgarten 28
6940 Weinheim -Lü.
☎ 06201 /55052

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

Autoren dieser Ausgabe:
Helmut Bernhardt ¥ Klaus-J. Mühlenbein
Jens Neueder ¥ Hartmut Obermann
Gerald Schröder ¥ Arnulf Sopp

sowie Artikel aus: Datawelt

Bankverbindung des CLUB 80

PostgiroKonto	Peter STEVENS
	Sonderkonto CLUB 80
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.

printed by ß

SCHLUSS

Hallo Club-80er,

zum Abschluß des Jahres 1986 ist das 17. Clubinfo fertiggestellt.
Wie im letzten INFO versprochen, werden mit diesem INFO einige redaktionelle Aufräumarbeiten erledigt. Leider habe ich aber doch nicht alles erledigen können, wie ich es Euch angekündigt hatte. Zu guter Letzt gab auch noch der Kopierer, den ich zum Verkleinern verwende, den Geist auf. Es wurde also Zeit aufzuhören für dieses Jahr.

Die Publikation unseres Jahresinhaltsverzeichnises hat sich zu einem umfangreichen Werk gemausert. Aus diesem Grund möchte ich es als ein Sonderheft herausbringen. Dieses Sonderheft wird, weil es mir zeitlich nicht früher möglich ist, nun erst mit dem 18. INFO erscheinen. Ich hoffe, Ihr habt dafür Verständnis.

Beiliegend zu dem 17. INFO erhaltet Ihr den neuesten Programm-Katalog, einen Club-Kalender mit unseren wichtigsten Terminen und das Anmeldeformular für unser Clubtreffen im März. Bitte nehmt den Anmeldetermin rechtzeitig war. Am 20. Februar ist Anmeldeschluß. Mit den anderen beiden Beilagen wünsche ich Euch viel Spaß und erfolgreiches Schaffen.

Wieder ist ein Jahr Clubinfo's vorüber. Es ist das letzte Heft mit dem orangefarbenem Deckblatt. Ich hoffe, Ihr seid mit meiner Arbeit für das Jahr 1986 zufrieden gewesen. Gleichzeitig wünsche ich Euch, daß Ihr Weihnachten und den Jahreswechsel gut überstanden habt.
Im neuen Jahr geht es mit frischem Mut und neuer Deckblattfarbe weiter. Ich würde mich freuen, wenn Ihr mich im "Neuen Jahr" weiterhin mit Euren Beiträgen unterstützt. Bis dahin alles Gute.

Euer Jens

CLUB 80 Mitgliederadressenliste

Name	Vorname	Straße	PLZ	Stadt	Telefon privat // geschäftlich
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 /241907 // 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 //
Förster	Werner	Christoph-Krebs-Straße 9	8720	Schweinfurt	09721 /21841 // 09721 /51256
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 /20458 //
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	- // -
Krüger	Karl-Herbert	Bruchweg 65	4920	Lemgo	05261 /13686 // -
Kuhn	Eckehard	Im Dorf 14	7443	Frickenhausen 1	07022 /45417 // 07022 /77442
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 -Lützelsachsen	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 // -
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	8080	Fürstenfeldbruck	08141 /3791 //
Reichelt	Dieter	Philipp-Schmitt-Straße 30	6902	Sandhausen	06224 /52906 // -
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
Schmitz	Paul-Jürgen	Lübecker Straße 6	6236	Eschborn	- // -
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	2080	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 /90037
Stevens	Peter	Postfach 56	4600	Dortmund 1	0231 /593883 // 0231 /593883
Sörensen	Rüdiger	Thomas-Mann-Straße 3A	6500	Mainz 1	06131 /32860 // 06131 /395268
Trapp	Harald	Kranichstr. 46	4270	Dorsten 1	02362 /42497 // 02362 /23127
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 /54686 // -
Zwickel	Walter	Lengfelden 123	A- 5101	Bergheim	0043662/51130 // -

Stand: Dezember 1986

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

CLUB 80 TERMIN - KALENDER 1986

JANUAR

Mo	Di	Mi	Do	Fr	Sa	So
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

FEBRUAR

Mo	Di	Mi	Do	Fr	Sa	So
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	◆	

MÄRZ

Mo	Di	Mi	Do	Fr	Sa	So
						1
2	3	4	5	6	7	8
9	10	11	12	‡	‡	‡
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

APRIL

Mo	Di	Mi	Do	Fr	Sa	So
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
◆	28	29	30			

MAI

Mo	Di	Mi	Do	Fr	Sa	So
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JUNI

Mo	Di	Mi	Do	Fr	Sa	So
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	◆					

JULI

Mo	Di	Mi	Do	Fr	Sa	So
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

AUGUST

Mo	Di	Mi	Do	Fr	Sa	So
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
◆	25	26	27	28	29	30
31						

SEPTEMBER

Mo	Di	Mi	Do	Fr	Sa	So
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

OKTOBER

Mo	Di	Mi	Do	Fr	Sa	So
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
◆	27	28	29	30	31	

NOVEMBER

Mo	Di	Mi	Do	Fr	Sa	So
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

DEZEMBER

Mo	Di	Mi	Do	Fr	Sa	So
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
◆	29	30	31			

◆ --) Redaktionsschluß des jeweiligen INFO's

‡ --) 3. Clubtreffen in Alsfeld (siehe INFO 16/17)

Messe-Daten sind unterstrichen. Bitte schaut dazu in die Terminliste.

Bitte denkt an die Termine und nehmt sie rechtzeitig war !!!