HARDWARE + SOFTWARE

COMPUTER-
TANDY-/ BENIE- ANWENDER
CLUB 80

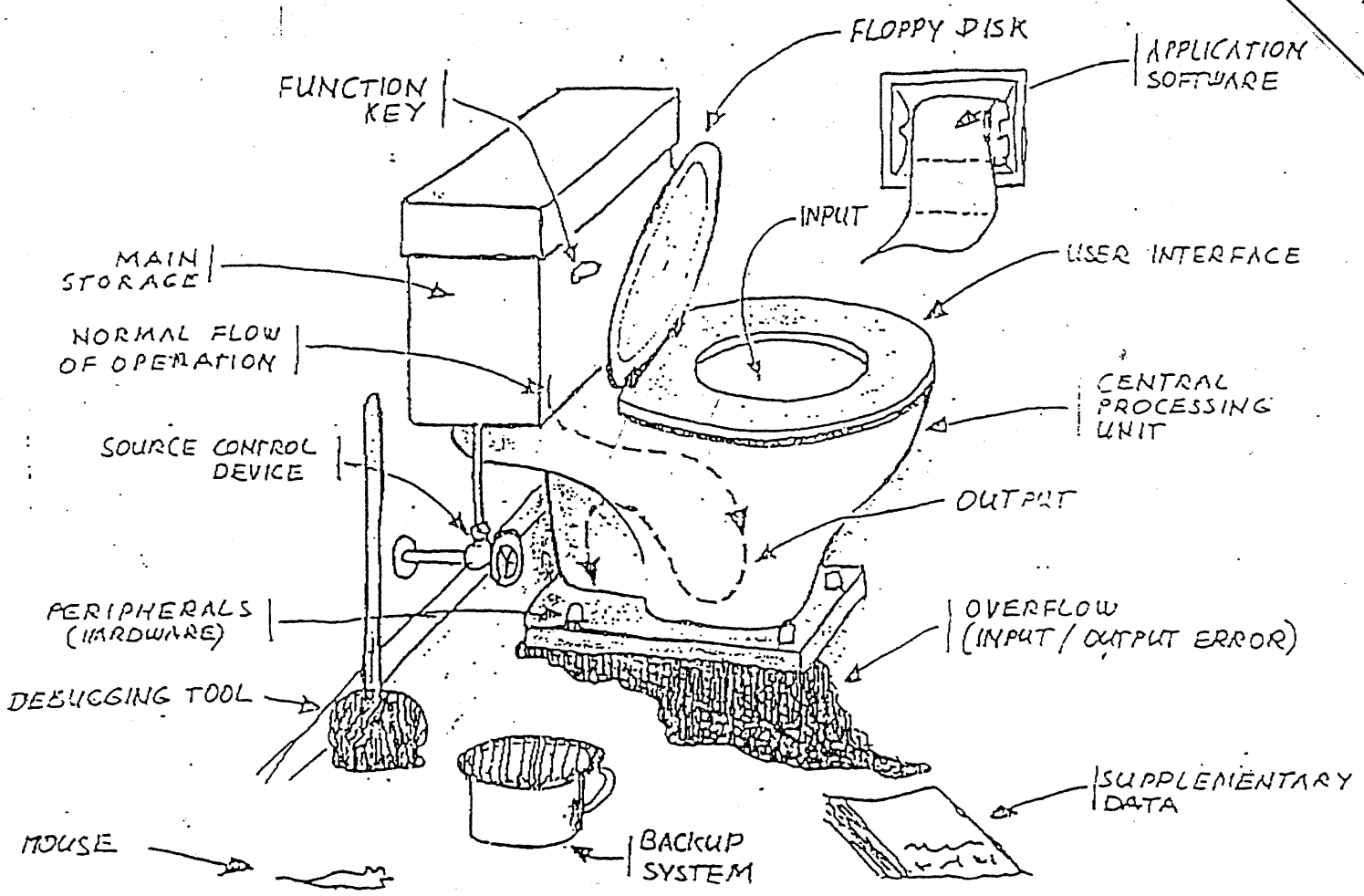
ERFAHRUNGSAUSTAUSCH

Achtung !!
wichtiger Termin !

Hobbytronik-Treffen

am Sonntag, den 8.11.92

gleich morgens zum Einlaßbeginn
oder
mittags, 12.00 Uhr
bei der Kartenkontrolle
am unteren Eingang



CLUBINFO

37. Ausgabe

Kontaktadresse:
Club 80
Fritz Chwolka
Saarstrasse 34
5173 Aldenhoven
Tel.: 02464/8920

Inhaltsverzeichnis

Achtung !!
wichtiger Termin !

Hobbytronik-Treffen

am Sonntag, den 8.11.92

gleich morgens zum Einlaßbeginn
oder
mittags, 12.00 Uhr
bei der Kartenkontrolle
am unteren Eingang

VORWORT

oder andersrum: We are back again!

Endlich habe ich wieder etwas Zeit. Ich muß mich bei Euch allen erst mal für das lange Ausbleiben des Clubinfos und das teilweise Nichtbeantworten von Briefen entschuldigen. Neben meiner beruflichen Tätigkeit im Wechselschichtbetrieb auf einem Kraftwerk machte ich per Fernschule den Maschinenbautechniker und habe jetzt alle Prüfungen abgeschlossen und die ersehnte Qualifikation.

Dieses Jahr hatten wir ein kurzfristig von Hartmut Obermann initiertes Clubtreffen, welches natürlich auch von mir besucht wurde. In einer zu nächtlicher Stunde anberaumten Sitzung hat der anwesende Vorstand eine Senkung des Mitgliedsbeitrages verabschiedet.

Der Vorstand war Mangels Anwesenheit leider nicht zur Wahl eines neuen Vorsitzenden beschlußfähig. Ich hätte mein Amt gerne zur Verfügung gestellt, vor allem wegen Zeitmangel , dies ist aber auf eine nächste Vorstandssitzung verschoben. Ich würde gerne die Arbeit des Softwarearchivatord uebernehmen, da sich die meisten doch direkt an mich wenden und Ruediger auch kaum noch Zeit hat.

Meine Meinung: Ich arbeite gerne im Club und helfe wo ich kann, wenn nur nicht diese administrativen Tätigkeiten währen.

Allgemein ist der Mitgliederstand konstant geblieben. Neuzugaenge und Abmeldungen halten sich die Waage. Allerdings werden die Nutzer von CP/M und Erweiterungen dazu (ZCPR...) mehr.

Vielen Dank fuer Eure Geduld mit mir
und viele Gruesse

F.Chwolka

Impressum

1. Vorsitzender — Fritz Chwolka
Tel.: 0 24 64/89 20 — Saarstraße 34
5173 Aldenhoven

2. Vorsitzender — Gerald Schröder
Tel.: 0 41 05/26 02 — Am Schützenplatz 14
2105 Seevetal 1

Hardewarekoordinator — Andreas Magnus
Tel.: 02 09/87 02 30 — Bismarckstraße 29
4650 Gelsenkirchen

Newdos-Diskothekar — Oliver Volz
Tel.: 07 11/6 85 30 13 — Am Ochsenwald 37A
7000 Stuttgart 80 (Rohrerhöhe)

CP/M-Diskothekar — Rüdiger Sörensen
Tel.: 0 61 31/32 06 60 — Thomas-Mann-Straße 3a
6500 Mainz 1

Clubbücherei — Kurt Müller
Tel.: 0 41 52/7 06 43 — Sophie-Scholl-Ring 3b
2054 Geesthacht

Redaktion — Jens Neueder
Tel.: 07 81/4 26 77 — Rudolf-Then Straße 32
7178 Gschlachtenbretzingen

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

Bankverbindung des CLUB 80
Postgirokonto Sonderkonto CLUB 80
Obermann H. 8870 Günzburg
Konto Nr. 496 071-605 Postgiroamt Frankfurt BLZ 500 100 60

Eine Zensur oder Kontrolle der INFO-Beiträge erfolgt nicht.
Die Redaktion.

Termine... Termine... Termine... Termine... Termine

Orgatec	Köln	22.10.- 27.10.92
Electronica	München	10.11.- 14.11.92

Redaktionsschluß für das nächste Clubinfo ist:

Mitte Dezember 92, damit rechtzeitig zum Jahreswechsel das INFO auch fertig wird.

Liebe Clubfreunde,

seit zwei Monaten bin ich Mitglied im Club-80 und möchte mich kurz vorstellen. Ich bin 42 Jahre alt, Physiklehrer (auch Chemie und Informatik) und habe 1980 bei einer Lehrerfortbildung an meiner Schule den TRS-80 Model I von TANDY kennengelernt (BASIC-Lehrgang). Weil die Geräte unverwüstlich sind, funktioniert die ganze Anlage bis heute. Ein Reporter der FRANKFURTER RUNDSCHAU hat am Tag der offenen Tür (8.2.1992) ein Photo von unserem Computerraum II (Im Computerraum I stehen neue 386-er SX-Rechner) geschossen. Weil die alten TRS-80-Geräte alle mit PIOs, AD-Wandlern usw. ausgestattet sind und mein Interesse dem Gebiet Messen, Steuern und Regeln gilt, bin ich mehr im Computerraum II als im Computerraum I zu finden.
Nachdem ich jahrelang mit BASIC programmiert hatte, habe ich mich nun mehr der Sprache PASCAL zugewandt. Leider habe ich mit ALCOR-PASCAL nicht den Durchbruch geschafft. Ich glaube, daß die von mir benutzte Version fehlerhaft war. Dann entdeckte ich PASCAL-80 und bin damit sehr zufrieden. Leider ist die Beschreibung sehr knapp. Wenn man wissen will, wie man z.B. den Stapelspeicher verändern will, so erhält man keinen Hinweis.
Da wir neben den TRS-80 noch drei alte PCs haben, auf denen inzwischen die TURBO-PASCAL-Version 6.0 läuft, war ich gezwungen, die Programme so zu schreiben, daß sie mit wenig Änderung auf beiden Geräten lauffähig sind. Das Programm zur Dekodierung der Funkuhrsignale läuft auf einem modernen PC genauso wie auf dem Model I. Dies ist möglich, weil kleine Zusatzprogramme wie TURBO1/PAS und PIO1/PAS die Unterschiede zwischen PASCAL-80 und TURBO-PASCAL ausgleichen. Es folgt nun das Programm mit ein paar Hinweisen, wie eine PIO (hier 8255) an den TRS-80 angeschlossen wird. Ich bin gerne bereit, weitere Auskünfte zu dieser Arbeit zu geben. Gibt es noch jemanden, der mit PASCAL-80 arbeitet? Bitte melden!

Ich wünsche allen noch viele neue Ideen auf den alten Rechnern

Wolfgang Brunk


```
(*************************************************************)
(*                                                           *)
(*   TURBO1/PAS     VERSION 1.0      -       18.01.1992     *)
(*                                                           *)
(*   ---- T R S - 8 0  ----   P A S C A L  8 0 ----        *)
(*                                                           *)
(*   ERLAUBT:     ANPASSUNG VON PASCAL-80                    *)
(*                AN TURBO-PASCAL                            *)
(*                                                           *)
(*************************************************************)

PROCEDURE DELAY (D: INTEGER);
VAR I: INTEGER;
BEGIN
  FOR I:=1 TO D DO
END;

PROCEDURE CLRSCR;
BEGIN
  CLS;
END;

FUNCTION KEYPRESSED: BOOLEAN;
BEGIN
  IF INKEY=CHR(0) THEN KEYPRESSED:=FALSE
                  ELSE KEYPRESSED:=TRUE
END;

FUNCTION READKEY: CHAR;
VAR CH: CHAR;
BEGIN
  REPEAT
    CH:= INKEY
  UNTIL ORD(CH)>0;
  READKEY:= CH
END;

PROCEDURE SETTIME(STUNDEN, MINUTEN, SEKUNDEN, HUNDERTSTEL:
          INTEGER);
BEGIN
  POKE(16449,SEKUNDEN);
  POKE(16450,MINUTEN);
  POKE(16451,STUNDEN);
END;

PROCEDURE SETDATE(JAHR, MONAT, TAG: INTEGER);
BEGIN
  POKE(16452,JAHR-1900);
  POKE(16453,TAG);
  POKE(16454,MONAT)
END;

(* WIRD FORTGESETZT *)
```

```
(*************************************************************)
(*                                                           *)
(*   PIO1.PAS      VERSION 1.0       DATUM :  05.03.1992     *)
(*                                                           *)
(*   ------ P C ------   T U R B O - P A S C A L  -----     *)
(*                                                           *)
(*                                                           *)
(*   ERLAUBT:    PIO('AUS','EIN','AUS');                     *)
(*               OUT('A',BYTE);                              *)
(*               BYTE:=INP('A');                             *)
(*                                                           *)
(*   ACHTUNG:    BEI MANCHEN COMPUTERN SIND NUR EIN ODER     *)
(*               ZWEI KANÄLE ('A','B') ZU BENUTZEN!          *)
(*                                                           *)
(*************************************************************)

PROCEDURE OUT(CH: CHAR; BYTE: INTEGER);
BEGIN
  IF (CH='A') OR (CH='a') THEN PORT[888]:=BYTE;
END;

FUNCTION INP(CH: CHAR): INTEGER;
BEGIN
  IF (CH='A') OR (CH='a') THEN INP:=PORT[888]
END;

PROCEDURE PIO(A,B,C: STR3);
BEGIN
  IF (A='AUS') OR (A='aus') OR (A='Aus') THEN OUT('A',0);
  IF (A='EIN') OR (A='ein') OR (A='Ein') THEN OUT('A',255)
END;
```



```
(*************************************************************)
(*                                                           *)
(*   PIO1/PAS      VERSION 1.0      -      18.01.1992        *)
(*                                                           *)
(*   ---- T R S - 8 0  ----   P A S C A L  8 0  ----       *)
(*                                                           *)
(*   ERLAUBT:    PIO('AUS','EIN','AUS');                     *)
(*               OUT('A',2);                                 *)
(*               OUT('C',5);                                 *)
(*               X:=INP('B');                                *)
(*                                                           *)
(*************************************************************)

PROCEDURE OUT(CH: CHAR; B: INTEGER);
VAR A,BYTE,NAME: INTEGER;
BEGIN
  CASE CH OF
    '7': A:=7;
    'A': A:=4;
    'B': A:=5;
    'C': A:=6
  END;
  POKE(-5536,  62);
  POKE(-5535,   B);
  POKE(-5534, 211);
  POKE(-5533,   A);
  POKE(-5532, 201);
  NAME:=CALL(-5536,BYTE)
END;

FUNCTION INP(CH: CHAR): INTEGER;
VAR A: INTEGER;
BEGIN
  CASE CH OF
    'A': A:=4;
    'B': A:=5;
    'C': A:=6
  END;
  POKE(-5530, 219);
  POKE(-5529,   A);
  POKE(-5528, 201);
  INP:=CALL(-5530,0)
END;

PROCEDURE PIO(A,B,C: STR3);
VAR X,Y,Z,ZAHL: INTEGER;
BEGIN
  X:=0;
  Y:=0;
  Z:=0;
  IF A='EIN' THEN X:=16;
  IF B='EIN' THEN Y:= 2;
  IF C='EIN' THEN Z:= 9;
  ZAHL:=128+X+Y+Z;
  OUT('7',ZAHL)
END;
```

Viele neugierige Besucher beim Tag der offenen Tür in der Friedberger Gesamtschule

08. FEB. 1992

Einblicke für künftige Schüler

FRIEDBERG. „Die Anlage hier ist ganz neu. Wir haben sie erst vor drei Tagen bekommen", sagt Walter Scheibner. „Maximal 18 Schüler — jeweils zwei an einem Platz — können hier arbeiten." Zuvor hatte sich die buntgemischte Gruppe von Müttern, Vätern und Kindern einen anderen EDV-Raum angesehen, wo fleißig getüftelt wurde. Wie es sich für einen Hausherrn gehört, führte der Leiter des Realschulzweigs der Friedberger Gesamtschule am Samstag seine Gäste durch das Haus. Und die kamen beim „Tag der offenen Tür" in Scharen.

Während andere erst mal gucken wollten, war für Thomas die Entscheidung bereits gefallen. Ab dem kommenden Schuljahr soll der Junge, der im April zehn Jahre alt wird, den Realschulweig der Gesamtschule besuchen, erzählt seine Mutter. Da ist der „Tag der offenen Tür" eine gute Gelegenheit, schon einmal einen Blick in die künftige „Lernstätte" zu werfen. Und wie gefällt sie ihm? „Ganz nett", findet Thomas und trollt sich.

Im Foyer sind Werke aus dem Kunst-Unterricht ausgestellt. Ein Katzenkopf, der aus einem Kupferblech getrieben wurde, Bilder in Pop-Art, Federzeichnungen, Foto-Collagen. Mehr als das interessieren Thomas aber die Computerterminals, an denen einige ältere Schüler arbeiten: Ein Elektronenhirn nimmt über vier Fotowiderstände an den Fenstern Informationen von draußen auf. Ein anderes fragt die Außentemperatur ab. Spätestens im Sommer eine ganz nützliche Einrichtung, um nachzusehen, ob nicht bald die Temperatur für „hitzefrei" erreicht ist.

Beim Gang durch die Räume der Gesamtschule mit ihren über 900 Schülern beläßt es Scheibner aber nicht bei solchen Vorzeigeprojekten. Der Lehrer weist auch auf Probleme der Schule hin: Neue Anstriche für einzelne Räume sind

Etliche Besuchergruppen mußte Walter Scheibner, der Leiter des Realschulzweigs, am Samstag durch die Friedberger Gesamtschule — im Bild einer der EDV-Räume — führen. (FR-Bilder: Eberhardt)

gen machen Scheibner Pläne, der Schule einen Teil ihres Gebäudekomplexes wegzunehmen. Der Kreis möchte dort einen Jahrgang der Augustinerschule unterbringen.

Für die Schule hat das tiefgreifende Folgen für das pädagogische Konzept: Wo Hauptschul-, Realschul- und Gymnasiumsklassen eines Jahrgangs nicht mehr wie bisher in einem Flügel untergebracht werden können, wird auch der Wechsel zwischen den einzelnen Schulzweigen

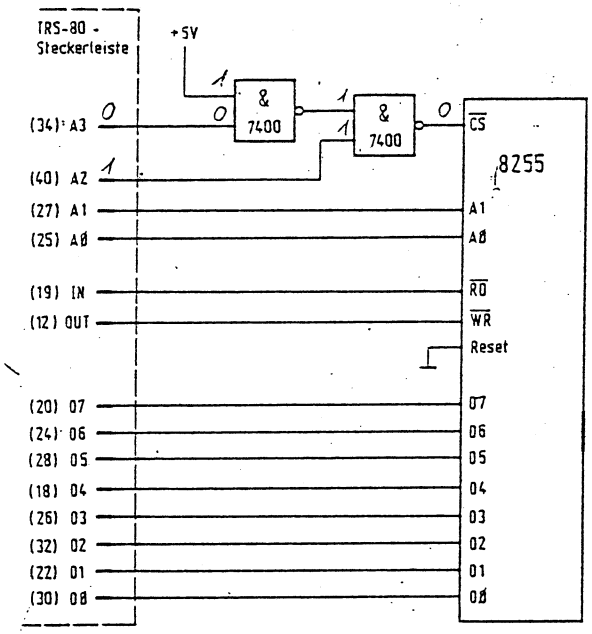

```
(*************************************************************)
(*                                                           *)
(*    PROGRAMM :   >FUNK3/PAS<          DATUM :   05.03.1992 *)
(*                                                           *)
(*                                                           *)
(*    COMPUTER :   TRS-80       UND     PC                   *)
(*                                                           *)
(*    SPRACHE  :   PASCAL-80    UND     TURBO-PASCAL-6.01    *)
(*                                                           *)
(*                                                           *)
(*    GESAMTSCHULE FRIEDBERG - WAHLPFLICHTKURS KLASSE 10     *)
(*                                                           *)
(*************************************************************)

PROGRAM FUNK3;

 USES CRT; DOS;      (*       <--- BEI TURBO-PASCAL       *)

TYPE STR3     = ARRAY(.1..03.) OF CHAR;          (* FUER PIO *)
     WERTTYP  = ARRAY(.0..59.) OF INTEGER;
     CHARTYP  = ARRAY(.0..59.) OF CHAR;
VAR  ZEIT                : BOOLEAN;
     WERT                : WERTTYP;
     WERTZEICHEN         : CHARTYP;
     MINUTEN, STUNDEN,
     TAG, WOCHENTAG,
     MONAT, JAHR, X,
     ZAHL1, ZAHL2, PC    : INTEGER;
```

```
(*************************************************************)
(*                                                           *)
(*    COMPUTERTYP:            ---------->    !    ENTFERNEN  *)
(*                                                           *)
(*-----------------------------------------------------------*)
(*                                                           *)
(*    TRS-80 ----->    *)                (*!$  TURBO1/PAS   *)
(*    TRS-80 ----->    *)                (*!$  PIO1/PAS     *)
(*                                                           *)
(*    PC     ----->    *)                (*$I PIO1.PAS      *)
(*                                                           *)
(*************************************************************)

PROCEDURE COMPUTER_TYP;
VAR WAHL: CHAR;
BEGIN
  WRITELN('AUSWAHL DES COMPUTERS:');
  WRITELN;
  WRITELN('>1<    TRS-80          ');
  WRITELN('>2<    PC              ');
  REPEAT
    WAHL:= READKEY
  UNTIL WAHL IN ['1','2'];
  IF WAHL='1' THEN PC:=1
              ELSE PC:=110
END;

PROCEDURE INIT;
BEGIN
  CLRSCR;
  COMPUTER_TYP;
  ZAHL1:= 70*PC;
  ZAHL2:=  7*PC;
  PIO('EIN','AUS','AUS')
END;
```

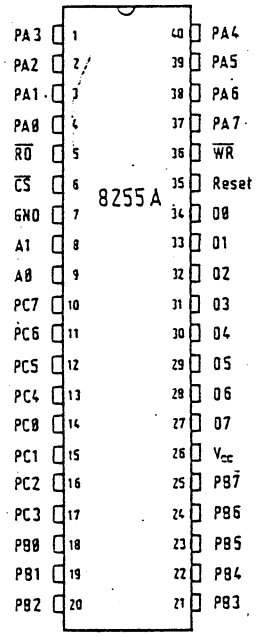
```
PROCEDURE MINUTENANFANG_SUCHEN;
VAR ZAEHLER, MARKE: INTEGER;
BEGIN
  WRITELN('MINUTENANFANG SUCHEN, BITTE WARTEN');
  REPEAT
    ZAEHLER:=0;
    REPEAT
      MARKE:= INP('A');
      MARKE:= MARKE MOD 2;
      IF MARKE=1 THEN ZAEHLER:= ZAEHLER+1
    UNTIL MARKE=0;
    WRITE(ZAEHLER,' ');
  UNTIL ZAEHLER>ZAHL1;
END;

PROCEDURE INFO_EINLESEN;
VAR ZAEHLER, MARKE: INTEGER;
BEGIN
  WRITELN;
  WRITELN('INFOEINLESEN, BITTE 1 MINUTE WARTEN');
  X:= 0;
  REPEAT
    ZAEHLER:=0;
    REPEAT
      MARKE:=INP('A');
      MARKE:=MARKE MOD 2;
      IF MARKE=0 THEN ZAEHLER:= ZAEHLER+1
    UNTIL MARKE=1;
    IF ZAEHLER>0 THEN
      BEGIN
        IF ZAEHLER>ZAHL2 THEN WERT(.X.):=1
                         ELSE WERT(.X.):=0;
        X:=X+1;
        WRITE(X)
      END
    UNTIL X=59;
  WRITELN;
END;

PROCEDURE INFO_AUSGEBEN;
BEGIN
  FOR X:=1 TO 59 DO
    BEGIN
      IF WERT(.X.)=0 THEN WERTZEICHEN(.X.):='.'
                     ELSE WERTZEICHEN(.X.):='-';
      WRITE(WERTZEICHEN(.X.))
    END;
  WRITELN
END;
```

```
PROCEDURE INFO_DEKODIEREN;
BEGIN
  WRITELN('INFO DEKODIEREN');
  MINUTEN    := WERT(.21.)* 1+WERT(.22.)* 2+WERT(.23.)* 4+
                WERT(.24.)* 8+WERT(.25.)*10+WERT(.26.)*20+
                WERT(.27.)*40;
  STUNDEN    := WERT(.29.)* 1+WERT(.30.)* 2+WERT(.31.)* 4+
                WERT(.32.)* 8+WERT(.33.)*10+WERT(.34.)*20;
  TAG        := WERT(.36.)* 1+WERT(.37.)* 2+WERT(.38.)* 4+
                WERT(.39.)* 8+WERT(.40.)*10+WERT(.41.)*20;
  WOCHENTAG  := WERT(.42.)* 1+WERT(.43.)* 2+WERT(.44.)* 4;
  MONAT      := WERT(.45.)* 1+WERT(.46.)* 2+WERT(.47.)* 4+
                WERT(.48.)* 8+WERT(.49.)*10;
  JAHR       := WERT(.50.)* 1+WERT(.51.)* 2+WERT(.52.)* 4+
                WERT(.53.)* 8+WERT(.54.)*10+WERT(.55.)*20+
                WERT(.56.)*40+WERT(.57.)*80+1900;
END;

PROCEDURE ZEIT_HOLEN;
BEGIN
  MINUTENANFANG_SUCHEN;
  INFO_EINLESEN;
  INFO_AUSGEBEN;
  INFO_DEKODIEREN;
END:

PROCEDURE UHR_STELLEN;
BEGIN
  SETDATE(JAHR,MONAT,TAG);
  SETTIME(STUNDEN,MINUTEN,0,0)
END;

FUNCTION MINUTEN_KONTROLLE: BOOLEAN;
VAR I, BIT_SUMME: INTEGER;
BEGIN
  BIT_SUMME:= 0;
  FOR I:=21 TO 28 DO
    IF WERT(.I.)=1 THEN BIT_SUMME:= BIT_SUMME+1;
  IF BIT_SUMME MOD 2=0 THEN MINUTEN_KONTROLLE:=TRUE
                       ELSE MINUTEN_KONTROLLE:=FALSE
END;

FUNCTION STUNDEN_KONTROLLE: BOOLEAN;
VAR I, BIT_SUMME: INTEGER;
BEGIN
  BIT_SUMME:= 0;
  FOR I:=29 TO 35 DO
    IF WERT(.I.)=1 THEN BIT_SUMME:= BIT_SUMME+1;
  IF BIT_SUMME MOD 2=0 THEN STUNDEN_KONTROLLE:=TRUE
                       ELSE STUNDEN_KONTROLLE:=FALSE
END;

FUNCTION TAG_KONTROLLE: BOOLEAN;
VAR I, BIT_SUMME: INTEGER;
BEGIN
  BIT_SUMME:= 0;
  FOR I:=36 TO 58 DO
    IF WERT(.I.)=1 THEN BIT_SUMME:= BIT_SUMME+1;
  IF BIT_SUMME MOD 2=0 THEN TAG_KONTROLLE:=TRUE
                       ELSE TAG_KONTROLLE:=FALSE
END;
```

```
PROCEDURE ZEIT_KONTROLLE;
VAR M, S, T: BOOLEAN;
BEGIN
  M:= MINUTEN_KONTROLLE;
  S:= STUNDEN_KONTROLLE;
  T:= TAG_KONTROLLE;
  IF (M=TRUE) AND (S=TRUE) AND (T=TRUE) THEN ZEIT:= TRUE
                                        ELSE ZEIT:= FALSE
END;

PROCEDURE ZEIT_ANZEIGEN;
BEGIN
  CLRSCR;
  WRITE('HEUTE IST ');
  CASE WOCHENTAG OF
    1: WRITE('MONTAG     ');
    2: WRITE('DIENSTAG   ');
    3: WRITE('MITTWOCH   ');
    4: WRITE('DONNERSTAG ');
    5: WRITE('FREITAG    ');
    6: WRITE('SAMSTAG    ');
    7: WRITE('SONNTAG    ')
  END;
  WRITE('DER  ');
  WRITE(TAG:2,'.',MONAT:2,'.',JAHR:4,'     ');
  WRITE(STUNDEN:2,':',MINUTEN:2,' UHR');
END;

PROCEDURE ENDE;
BEGIN
  REPEAT
  UNTIL KEYPRESSED
END;

BEGIN (* HAUPTPROGRAMM *)
  INIT;
  REPEAT
    ZEIT_HOLEN;
    UHR_STELLEN;
    ZEIT_KONTROLLE;
  UNTIL ZEIT=TRUE;
  ZEIT_ANZEIGEN;
  ENDE
END.
```

Schaltbild des Spezial-Interface. Die angegebenen Zahlen sind die Sockelstift-Nummern des TRS-80 Steckers



Anschlußbezeichnungen

D_0 - D_7	Daten-Bus (Zweiweg)
RESET	Rücksetz-Eingang
CS	Baustein-Auswahl
$\overline{RD}$	Lese-Eingang
$\overline{WR}$	Schreib-Eingang
A0, A1	Kanal-Adresse
PA_0 - PA_7	Kanal A (Bit 0 bis 7)
PB_0 - PB_7	Kanal B (Bit 0 bis 7)
PC_0 - PC_7	Kanal C (Bit 0 bis 7)
V_{cc}	Versorgungsspannung (+ 5V)
GND	Masse (0 V)

Anschluß-Belegung und Anschlußbezeichnungen des Bausteins 8255

Tabelle 1. Schema der beim Sender DCF77 verwendeten Zeitcodierung

Sek.	Bed.	Pegel	Wert	Zuordnung	Beispiel
0	M	0	–	Minuntenbeginn	0
1...					
14	–	X	–	z. Z. nicht belegt	–
15	R	X	–	Reserveantenne	0
16	A	X	–	Ankündigung Sommerzeit	0
17	Z1	X	2^1	Zonenzeit-bits	1
18	Z2	X	2^0	Zonenzeit-bits	0
19	–	X	–	z. Z. nicht belegt	–
20	S	1	–	Start der Zeitcod.	1
21		X	1		1
22		X	2		0
23		X	4	Einer	1
24		X	8		0
				Minuten	
25		X	10		1
26		X	20	Zehner	1
27		X	40		0
28	P1	X	–	Prüfbit 1	0
29		X	1		1
30		X	2		0
31		X	4	Einer	0
32		X	8		1
				Stunden	
33		X	10		1
34		X	20	Zehner	0
35	P2	X	–	Prüfbit 2	1
36		X	1		1
37		X	2		0
38		X	4	Einer	1
39		X	8		0
				Kalendertag	
40		X	10		1
41		X	20	Zehner	0
42		X	1		0
43		X	2	Wochentag	0
44		X	4		1
45		X	1		1
46		X	2	Einer	1
47		X	4		1
48		X	8		0
				Kalendermonat	
49		X	10	Zehner	0
50		X	1		0
51		X	2		1
52		X	4	Einer	0
53		X	8		0
				Kalenderjahr	
54		X	10		0
55		X	20		0
56		X	40	Zehner	0
57		X	80		1
58	P3	X	–	Prüfbit 3	1
59	–	–	–	Marke fehlt	–

Pegel: 0 = fest LOW; 1 = fest HIGH; X = LOW oder HIGH
Beispiel: Do (= 4), 15. 07. 82, 19.35 Uhr

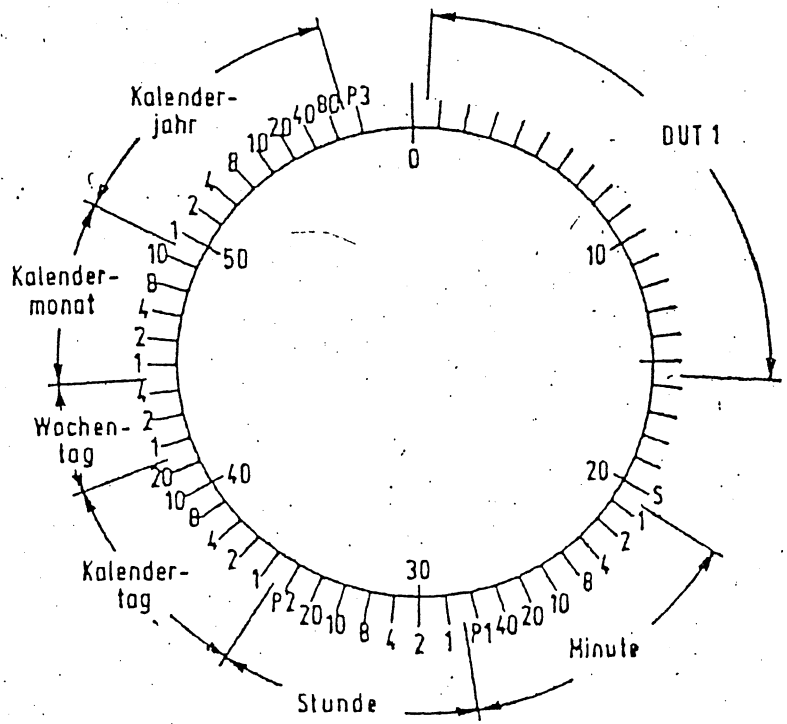


Bild 1. Schema der codierten Zeitinformation. P 1, P 2, P 3 Prüfbits, S auf 0,2 s verlängerte Sekundenmarke zum Start der Codierung der Zeitinformation

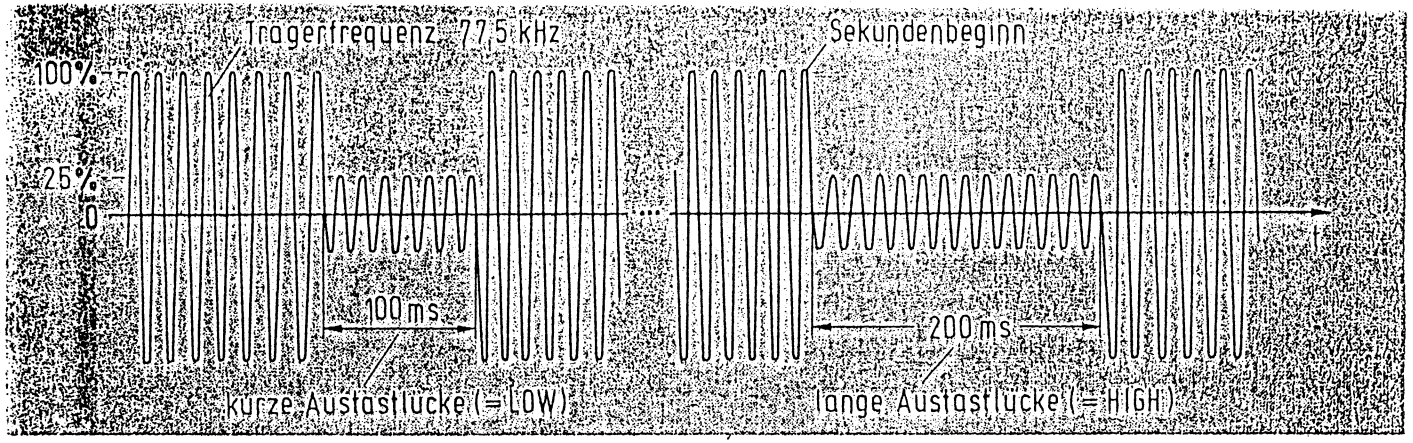


Bild 1: Die codierte Zeitinformation „steckt" in den unterschiedlich langen Trägerabsenkungen (Sekundenmarken).

Der Atari Portfolio,
ein "echter" MSDOS-Computer ?

Möglichkeiten eines Pocket PC's

1. Vorwort

Trotzdem der Artikel einen Außenseiter unter den Rechnern (und einen MSDOS-Kompatiblen dazu) beschreibt, wird dieser hoffentlich für die Club Mitglieder interessant sein.
Zumindest von einem im Club (Hallo Claus !) ist mir bekannt, daß er einen Atari Portfolio besitzt. Da die neueste SMD-Technologie aber auch andere im Club interessieren dürfte, wird dieser Artikel hoffentlich nicht nur für 2 Club-Mitglieder und zum Füllen der Info-Seiten geschrieben worden sein.

2. Allgemeines

Die Reaktionen auf das 20x10x2 cm große schwarze Kästchen gleichen sich: "Was ist denn das für ein Taschenrechner ?" oder "Ich wußte gar nicht, daß Atari Taschenrechner herstellt". Die nächste Phase des Gespräches hört sich dann etwa so an: "DAS soll ein PC sein" gefolgt von einem verächtlichen herabziehen der Lippen. Es besteht aber auch die Möglichkeit das zwei Freak's aufeinander treffen (soll es auch außerhalb der Clubtreffen geben): "Haben Sie ihre Schnittstelle dabei ?". Das 40x8 Zeichen große LCD--Display wird durch Betätigung eines Schiebeschalters hochgeklappt, daß 63 Tasten umfassende "Mäuseklavier" steht zur Verfügung und der Daten, Text oder Programmtransfer via parallele oder serielle Schnittstelle kann beginnen. Was verbirgt sich nun in dem knapp 500 g schweren "Taschen-PC" an Hardware ?

3. Hardware

Kennt jemand noch die Daten des aus dem Jahr 1983 stammenden original IBM-PC ?: 64 KByte Hauptspeicher und 4,77 MHz Taktfrequenz. Der Portfolio wartet da in seiner ursprünglichen Version mit 4,93 MHz, 128 KByte Speicher und 256 KByte ROM auf. Den Hauptspeicher von 128 KByte RAM bilden vier S-RAM Bausteine vom Typ 43256-15. Zwei SMD-ROM's beherbergen das Betriebssystem sowie die internen Programme. Es gibt inzwischen Anbieter, die den Portfolio mit 640 KByte intern aufgerüstet ausliefern. Der Portfolio besitzt aufgrund seiner Größe keinen Bildschirm von 80x25, wie bei MDA-Darstellung üblich, sondern ein 40x8 Zeichen großes nicht hintergrundbeleuchtetes LC-Display (bei ungünstigem Blickwinkel prima als Rasierspiegel zu gebrauchen). Die Punktmatrix der Buchstaben beträgt 8x8 Pixel. Dadurch rücken Unterlängen wie 'g' etwas unschön nach oben.
Anstelle der herkömmlichen 5.25"- oder 3.5" Laufwerke können zur Speicherung Chipkarten verwendet werden. An der linken Gehäuseseite befindet sich der Schlitz für die Speicherkarten. Dieses "Laufwerk A:" kann sowohl ROM als auch RAM-Cards lesen.
RAM-Cards gibt es von 32KByte bis 1MByte. Durch einen Schalter an den RAM-Cards kann man diese gegen ungewolltes Schreiben schützen.
Erweitert man den Portfolio extern mit der von Atari vertriebenen 256K Speichererweiterung, so erhält man zusätzlich noch ein "Laufwerk B:".
Durch diese Speichererweiterungen verliert der Portfolio sein handliches Format, allerdings ist das Laufwerk B: recht nützlich. An der rechten Gehäuseseite hinter einer Plastikkappe verborgen befindet sich der Erweiterungsport an dem man wahlweise die Erweiterungsmodule (max. 2) und/oder eine RS 232 Schnittstelle oder eine parallele Schnittstelle anschließen kann. Stellt man sich nun die maximale externe Hardware-Erweiterung von 2 Speichererweiterungen a 256K + parallele oder serielle Schnittstelle vor, so erreicht der 20 cm Breite Rechner einen 21 cm langen "Anbau". Da Atari keine Abhilfe schaffte, wird eine Erweiterung von anderen Firmen vertrieben, welche 512K + Laufwerk B: + parallele/serielle Schnittstelle enthält. Länge: Sagenhafte 5 cm. Die Portbelegung des Portfolio ist mir leider nicht bekannt.

4. Steckerbelegung der Schnittstellen

4.1. Parallelschnittstelle

Pol	Richtung	Signal
1	AUS	Strobe
2	AUS	Data0
3	AUS	Data1
4	AUS	Data2
5	AUS	Data3
6	AUS	Data4
7	AUS	Data5
8	AUS	Data6
9	AUS	Data7
10	EIN	Acknowledge
11	EIN	Busy
12	EIN	Papierfehler
13	EIN	Select
14	AUS	Auto Feed
15	EIN	Fehler
16	AUS	Drucker Initialisieren
17	AUS	Select Input
18-25		Ground

4.2. Serielle Schnittstelle

Pol	Signal	Funktion	
1	CD	Carrier Detect	(Empfangssignal)
2	RD	Receive Data	(Empfangsdaten)
3	TD	Transmit Data	(Sendedaten)
4	DTR	Data Terminal Ready	(DEE betriebsbereit)
5	GND	Signal Ground	(Signal Erde)
6	DSR	Data Set Ready	(Betriebsbereitschaft)
7	RTS	Request To Send	(Sendeteil einschalten)
8	CTS	Clear To Send	(Sendebereitschaft)
9	RI	Ring Indicator	(Ankommender Ruf)

5. Eingebaute Software

Die ROMs des Portfolio enthalten einige nützliche Anwendungsprogramm. Alle Programme belegen die Funktionstasten einheitlich, so daß man sich schnell an deren Funktion gewöhnt. Alle Funktionen können über Popdown-Menüs aufgerufen werden. Ein Klemmbrett (Clipboard) ermöglicht den Austausch von Daten zwischen den integrierten Programmen. Aufgerufen werden die Programme über die 'ATARI'-Taste oder über ein Menü, welches mit dem Befehl APP gestartet wird. Verfügbar sind:

- Adreßbuch
- Rechner
- Zeitplaner
- Textverarbeitung
- Tabellenkalkulation

Hinzu kommen noch die Möglichkeiten der Systemvorbereitung (RS232 initialisieren etc.)
Das Adreßbuch dient zum Speichern von Namen, Adressen und Telefonnummern und wählt diese automatisch.
Der einfache Rechner besitzt 5 Speicher, eine korrigierbare

Eingabezeile, Quadratwurzel, verschiedene Zahlenformate und Druckausgabe (Datei/Drucker).
Der einfach zu handhabende Texteditor besitzt die üblichen Funktionen wie Blockbildung, Verschieben, Kopieren, Suchen und Ersetzen. Den Ersatz eines Notizblocks gewährleistet dieses Programm allemal.
Eine recht interessante Anwendung ist der Zeitplaner, der einen Kalender und ein Tagebuch mit Weckfunktion bei jedem Eintrag enthält. Termine können abgelegt und einmalig, täglich, arbeitstäglich, wöchentlich, monatlich oder jährlich wiederholt werden.
Die Tabellenkalkulation ist Lotus 1-2-3 kompatibel und generiert 127 Spalten mal 255 Zeilen.
Das Problem des zu kleinen Bildschirm hat ATARI bei diesem Programm oder auch MSDOS-Programmen, die nicht auf den Portfolio Bildschirm angepaßt wurden elegant umsegelt: der intere Bildschirmspeicher umfaßt die PC-üblichen 80x25 Zeichen. Je nach Einstellung im Menü "Systemvorbereitung" wandert das 40x8 Zeichen Fenster dynamisch mit dem Cursor mit oder zeigt statisch einen Ausschnitt an, der mit der ATARI und den Cursor-Tasten über den virtuellen Bildschirm geschoben wird.
Dieses Verfahren ist sehr gewöhnungsbedürftig. Betrachtet man den im Vergleich zum kleinen LCD-Bildschirm riesigen PIEPSER (Lautsprecher wäre wohl übertrieben), so hätte man ruhig etwas mehr als 40x8 Zeichen investieren können.
Eingebaut ist auch die serielle Datenübertragung, welches den Rechner nun auch für die Z80-Welt interessant macht. Via serieller Datenübertragung ist damit ein Austausch zwischen PC und Z80 Rechner möglich.

6. Externe Software

An externen Programmen gibt es mittlerweile eine ganze Menge. Beispielsweise existiert ein BTX-Programm, diverse Modem-Programme inklusive Hardware (Mini-Modem), ein auf das ATARI-Display abgestimmtes dBASE, diverse Fremdwörter Lexika, welche den Portfolio in einen echten Übersetzungscomputer verwandeln. Generell kann man sagen, daß alle *.COM Files der MSDOS Welt fehlerfrei auf dem Portfolio laufen, bei *.EXE Files kann es schon mal Schwierigkeiten geben.
Turbo Pascal 3.0 läuft auf dem Portfolio ebenso fehlerfrei wie GW-BASIC. Erstaunlich ist, daß die entsprechenden Befehle zur Steuerung einer Farbgraphikkarte in den Programmen keinerlei Schwierigkeiten bereiten.
Einen CP/M Rechner kann man durch entsprechende Emulationsprogramme leider nicht aus dem Portfolio machen. Zwar läßt sich beispielsweise das Programm Z80MU, ein CP/M 2.2 Emulator für MSDOS-Rechner, noch fehlerfrei auf dem Portfolio installieren und starten. Der entsprechende Prompt taucht auch auf dem Display auf und der Speicher wird auch in 64K-Segmente unterteilt, allerdings lassen sich nicht mehr als ca. 10 Zeichen in der Tastaturzeile eingeben. Danach stürzt der Rechner ab.
Das man auch die Dos-Bios Interrupts per Programm nutzen kann, zeigt folgendes Beispiel-Programm in Turbo-Pascal 3.0:
Das Programm sperrt Dateien gegen versehentliches löschen und wird durch cmode s dateiname für Sperren, bzw. cmode f dateiname für Freigabe einer Datei aufgerufen.

```
program filechmode;
type
  registers = record
    ax, bx, cx, dx, bp, si, di, ds, es, flags:integer;
  end;

var
  fn   : string!64!;      -> ! = eckige Klammer
  mode : string!2!;
  regs : registers;
  atr  : byte;

begin
  if paramcount <> 2 then begin
    write('Dateiname: ');
    readln(fn);
    write('Mode F(rei S(perren: ');
    readln(mode);
   end

   else begin
     mode := paramstr(1);
     fn   := paramstr(2);
end;

case mode!1! of
  'S','s' : atr := 1;
  'F','f' : atr := 0;
end;

fn := fn + chr(0);
with regs do begin
  ax := $4301;
  cx := atr;
  ds := seg(fn);
  dx := ofs(fn) + 1;
  msdos(regs);
  if (flags shl 15) shr 15 = 1 then begin
    write('Error ',ax);
    case ax of
      3 : writeln(': Datei existiert nicht');
      5 : writeln(': Kein gültiger Dateiname');
    end;
    halt;
  end;
  writeln('Done');
end;
end.
```

Alles in allem representiert das Maschinchen eine beachtliche Ingenieur-Leistung.

Egbert Schröer, April 1992

7. Literaturnachweis

1. Portfolio-Bedienungshandbücher (Portfolio-Grundgerät, RS232c-Erweiterung, parallelschnittstelle, Speichererweiterung)
2. Systemhandbuch zum ATARI-Portfolio, Fischel GmbH 1991, S. Gad
3. ct Juli 1990, S.114, "Ungleiche Brüder"
4. Happy Computer Sept. 1989, S. 116, "Der kleinste PC der Welt"

ZEUS spricht Intel-Hex

"ZEUS 1.Ø,(c) 1983 C.E.C" erscheint auf dem Bildschirm, wenn ich meinen Editor-Assembler auf dem Model 4P aufrufe. Neun Jahre ist das gute Stück Software also schon alt. An der Lebensdauer heutiger Programme gemessen ist er ein Relikt aus der Computer-Steinzeit. Doch das Werkzeug, das ich schon viele Jahre benutze, ist keineswegs stumpf geworden. Im Gegenteil: Mit einem Assemblerprogramm habe ich dem ZEUS neue Anwendungen erschlossen, die ihn über seinen ursprünglich geplanten Einsatz hinaus zum Universalwerkzeug für die Z8Ø-Programmentwicklung werden lassen.

Lohnt es sich heute in der Zeit der High Performance Chips noch, den Z8Ø Mikroprozessor für Neuentwicklungen zu verwenden? Schon ein Blick auf die Verkaufszahlen beantwortet diese Frage. Zahlenmäßig dominieren die 8-Bit-Prozessoren nach wie vor den Markt. Etwa die Hälfte von ihnen basiert auf dem Z8Ø-Befehlssatz. Wegen seiner geradlinigen I/O-und Interruptstruktur, sowie der großen vorhandenen Programmiererfahrung gehört der Z8Ø und seine Nachfolgetypen für Anwendungen in der Prozeßtechnik und Datenkommunikation immer noch zur ersten Wahl.

Natürlich gibt es inzwischen Assembler für den Z8Ø, die auf MSDOS-Rechnern laufen. Aber versucht einmal, eine Kombination von Editor und Assembler zu finden, die ähnlich leistungsfähig und leicht zu bedienen wie der ZEUS ist. Bis jetzt kenne ich nur ein Produkt, das Gleiches (und mehr) leistet. Es ist der ECAL von Maxim Technology, für den man jedoch über DM 600,- anlegen muß.

Der ZEUS war konzipiert als eine Art verbesserter EDTASM, um Programme zu entwickeln, die auf dem TRS8Ø laufen. Er erzeugt /CMD-Files auf Diskette, die man dann von der DOS-Ebene aus starten kann. Nicht vorgesehen ist der Fall, daß der TRS8Ø gar nicht das Zielsystem für die selbst entwickelte Software ist, sondern ein anderes Gerät, das über die serielle Schnittstelle an den TRS8Ø angeschlossen ist. Da der ZEUS den Z8Ø-Code mit jeder beliebigen Startadresse erzeugt, braucht man nur noch ein Hilfsprogramm, das den auf der Diskette vorhandenen Objectcode umformatiert und über die Schnittstelle überträgt.

Für die serielle Übertragung von Binärdaten ist das am häufigsten angewendete Format der Intel-Hex-Code. Die Erklärung dazu habe ich der Einfachheit halber aus der Zeitschrift c't auskopiert (siehe Information im Kasten). Mein Entwicklungsziel war es, die Objectfiles des ZEUS so umzucodieren, daß sie der Intel-Hex-Norm entsprechen und über die serielle Schnittstelle ausgegeben werden können. Das ist nicht ganz einfach, denn ZEUS und EDTASM lassen Blocklängen bis zu 256 Bytes zu, während Intel-Hex mit Blocklängen von 64 Zeichen oder weniger arbeitet. An den passenden Stellen müssen Adressmarken und Prüfsummen eingesetzt werden. Die Versuche fingen zunächst mit einem Basicprogramm an. Das lief zwar im Prinzip, war aber viel zu langsam. Erst mit dem Assemblerprogramm konnte ich die im TRS8Ø mögliche Baudrate von 96ØØ nutzen.

Der Intel-Hex-Ausgang aus dem TRS-8Ø ist wie ein Schlüssel zu neuen Möglichkeiten. Auf einmal ist der gute alte ZEUS wieder up to date. Man kann mit ihm Objectcodes in EPROM-Programmiergeräte oder EPROM-Simulatoren laden, man kann Programme ins RAM von Single-Board-Computern hinunterladen oder per Modem zu anderen Computern übertragen. Auch die modernsten Entwicklungssysteme schicken genau den gleichen Intel-Hex-Code über ihre seriellen Schnittstellen.

Die Gebrauchsanweisung für DLHEX/CMD:

1) Das Programm assemblieren z.B. mit der ZEUS-Anweisung ANO

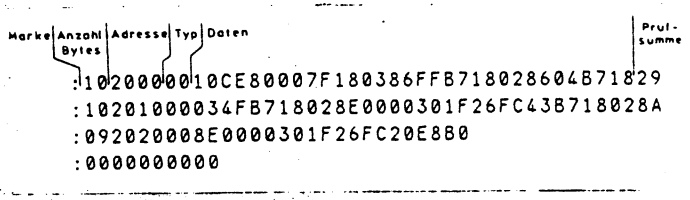
2) Als Namen des Objectfiles immer OBJ eingeben. Nur diesen Namen findet und bearbeitet DLHEX/CMD.

3) Den ZEUS verlassen mit dem Dreifingergriff Q-Enter-Clear

4) DLHEX aufrufen. Man sieht am Bildschirm die gleichen Daten, die über die serielle Schnittstelle gesendet werden.

5) DLHEX/CMD führt nach beendetem Programmlauf zu NEWDOS oder startet wahlweise gleich wieder den ZEUS.

Heinrich Betz



Intel-Hex-Format

Das Intel-Hex-Format ist eine Kodierungsvorschrift für beliebige byteorientierte, adreßbezogene Daten, wie sie etwa Maschinenprogramme für 8-Bit-Mikroprozessoren darstellen. Zur Kodierung der Daten werden ausschließlich Textzeichen verwendet (ASCII), wobei der Wert eines Bytes, der üblichen Schreibweise entsprechend, in zwei Hex-Ziffern (0..9, A..F) erscheint.

Der Hex-Ziffern-Strom der umgesetzten Daten ist nun aber nicht 'endlos', sondern in Blöcke unterteilt, die jeweils mit der Steuercodefolge CR/LF enden, so daß eine Datei in diesem Format wie eine normale Textdatei behandelt werden kann. Dies betrifft insbesondere die Übertragung über Schnittstellen, die keine 8-Bit-Transfers erlauben (vornehmlich serielle Schnittstellen).

Jeder Block stellt eine in sich geschlossene, mit einem Vorspann und einer Prüfsumme versehene Informationseinheit dar; was bei Übertragungsfehlern eine relativ genaue Aussage über den Fehlerort zuläßt beziehungsweise nicht gleich die Wiederholung der kompletten Sendung verlangt. Das erste Zeichen des Vorspanns ist ein Doppelpunkt, gefolgt von zwei Hex-Ziffern (= ein Byte), die die hexadezimale Anzahl Datenbytes im jeweiligen Block verkörpern. Die nächsten vier Hex-Ziffern bilden die (absolute) 16-Bit-Adresse, unter der das erste Datenbyte des Blockes im Speicher abgelegt werden soll. Den Vorspann beschließt ein Byte, dessen Wert den Typ des Blockes angibt: 0 = Datenblock, 1 = Endblock. Auf diese Unterscheidung kann jedoch verzichtet werden, wenn sich ein Endblock auch durch eine Blocklänge gleich Null eindeutig kennzeichnen läßt. (So verfahren die meisten Assembler unter CP/M, auch der XASM09; das Typbyte ist dann immer Null).

Nach den Datenbytes, als letztes vor dem Block- oder Zeilenende (CR/LF), steht die 8-Bit-Prüfsumme, die als negativer Wert der Summe aller übrigen Bytes des Blocks gebildet wird. Oder anders ausgedrückt: Die Summe aller Bytes eines Blocks (einschließlich der Prüfsumme) beträgt Null. Durch die Addition auftretende Überträge bleiben dabei unberücksichtigt.


```
                    00001 ;DLHEX/SRC      29.08.92          Heinrich Betz
                    00002
                    00003 ;Das Programm speichert das File OBJ/CMD ins RAM (OBJBUF)
                    00004 ;und laedt es von dort ueber die serielle Schnittstelle
                    00005 ;im Intel-Hex-Code zum EPROM-Programmer oder zum SBC.
                    00006 ;Schnittstellen-Parameter: siehe Zeilen 14 bis 19.
                    00007
4424                00008 OPEN    EQU     4424H           ;Open file
4436                00009 READ    EQU     4436H           ;Read one record
4428                00010 CLOSE   EQU     4428H           ;Close file
4409                00011 ERRDSP  EQU     4409H           ;Display DOS errors
                    00012
5200                00013         ORG     5200H
                    00014 ;Baudraten: 0=50Bd,;17=75Bd;34=110Bd;52=150Bd;85=300Bd
                    00015 ;102=600Bd;119=1200Bd;170=2400Bd;204=4800Bd;238=9600Bd
5200  3EAA          00016 START   LD      A,170           ;Baudrate..
5202  D3E9          00017         OUT     (233),A         ;waehlen
5204  3EA5          00018         LD      A,165           ;ev. Prty, 1 Stopbit
5206  D3EA          00019         OUT     (234),A         ;7 Datenbits, DTR=true
5208  3E0F          00020         LD      A,15            ;Master Reset..
520A  D3E8          00021         OUT     (232),A         ;UART
520C  216654        00022         LD      HL,OBJBUF       ;ab OBJBUF den..
520F  226154        00023         LD      (PTSTOR),HL     ;Objcode abspeichern
5212  210553        00024         LD      HL,MSOBJ        ;Text 'OBJ/CMD'
5215  114153        00025         LD      DE,FCB
5218  010800        00026         LD      BC,MSOEND-MSOBJ
521B  EDB0          00027         LDIR                    ;File Name -> FCB
521D  216153        00028         LD      HL,FILBUF
5220  114153        00029         LD      DE,FCB
5223  0600          00030         LD      B,0
5225  CD2444        00031         CALL    OPEN            ;Open File
5228  C2EE52        00032         JP      NZ,SYNERR       ;evtl. Error anzeigen
522B  114153        00033 RDSEC   LD      DE,FCB
522E  CD3644        00034         CALL    READ            ;read sector
5231  C24652        00035         JP      NZ,RDERR
5234  216153        00036         LD      HL,FILBUF
5237  ED5B6154      00037         LD      DE,(PTSTOR)
523B  010001        00038         LD      BC,256
523E  EDB0          00039         LDIR                    ;store -> RAM
5240  ED536154      00040         LD      (PTSTOR),DE
5244  18E5          00041         JR      RDSEC
                    00042
5246  FE1C          00043 RDERR   CP      1CH             ;Check for EOF
5248  CA5052        00044         JP      Z,FILEND        ;EOF=File gelesen
524B  FE1D          00045         CP      1DH             ;Check if past end
524D  C2E952        00046         JP      NZ,ERROR
5250  114153        00047 FILEND  LD      DE,FCB
5253  CD2844        00048         CALL    CLOSE           ;Close File
5256  C20944        00049         JP      NZ,ERRDSP       ;Display error, goto DOS
                    00050 ;.....................................................
                    00051 ;Programmteil fuer Download -> SBC
5259  CDC901        00052         CALL    01C9H           ;CLS
525C  216654        00053         LD      HL,OBJBUF       ;Anfang Obj.-File
525F  7E            00054 EPLOOP  LD      A,(HL)          ;1.Byte=Satzkennz.
5260  FE01          00055         CP      01              ;01=Normaler Datensatz
5262  C2F352        00056         JP      NZ,ENDE         ;02=End-Datensatz
5265  23            00057         INC     HL              ;HL pt ZEUS-Bytezahl
5266  7E            00058         LD      A,(HL)          ;Zahl der Bytes
5267  3D            00059         DEC     A               ;Zeus hat 2 zu viel,
5268  3D            00060         DEC     A               ;(zaehlt Startadr. mit)
5269  F5            00061         PUSH    AF
526A  D1            00062         POP     DE              ;Flag in E
526B  326554        00063         LD      (GESBYT),A
526E  23            00064         INC     HL              ;HL pt LSB der Startadr.
526F  7E            00065         LD      A,(HL)
5270  326354        00066         LD      (STARTA),A
5273  23            00067         INC     HL              ;HL pt MSB der Startadr.
```

```
5274  7E          00068           LD      A,(HL)
5275  326454      00069           LD      (STARTA+1),A
5278  3E3A        00070 STMARK    LD      A,':'           ;Startmarke..
527A  CD3353      00071           CALL    SIOOUT          ;ausgeben
527D  CB73        00072           BIT     6,E             ;Z-Flag von vorher
527F  CBB3        00073           RES     6,E
5281  2008        00074           JR      NZ,BYT256       ;256 Bytes?
5283  3A6554      00075           LD      A,(GESBYT)
5286  FE21        00076           CP      33              ;<= 32 Bytes?
5288  DA9F52      00077           JP      C,KLGL32
528B  3E20        00078 BYT256    LD      A,32            ;Bytezahl=32..
528D  47          00079           LD      B,A
528E  CD1E53      00080           CALL    BIN2HX          ;ausgeben
5291  3A6554      00081           LD      A,(GESBYT)
5294  D620        00082           SUB     32
5296  326554      00083           LD      (GESBYT),A
5299  CDAD52      00084           CALL    SEND
529C  C37852      00085           JP      STMARK
              00086
529F  3A6554      00087 KLGL32    LD      A,(GESBYT)      ;Zahl der restl. Bytes
52A2  47          00088           LD      B,A             ;fuer DJNZ -> B..
52A3  CD1E53      00089           CALL    BIN2HX          ;und ausgeben
52A6  CDAD52      00090           CALL    SEND
52A9  23          00091           INC     HL
52AA  C35F52      00092           JP      EPLOOP
              00093
52AD  3A6454      00094 SEND      LD      A,(STARTA+1)    ;MSB der Startadr.
52B0  F5          00095           PUSH    AF
52B1  80          00096           ADD     A,B             ;Checksu. bilden
52B2  4F          00097           LD      C,A             ;C=Checksu.-Speicher
52B3  F1          00098           POP     AF
52B4  CD1E53      00099           CALL    BIN2HX          ;in Hex-ASCII ausgeben
52B7  3A6354      00100           LD      A,(STARTA)      ;LSB der Startadr.
52BA  F5          00101           PUSH    AF
52BB  81          00102           ADD     A,C             ;Checksu. aufaddieren
52BC  4F          00103           LD      C,A
52BD  F1          00104           POP     AF
52BE  CD1E53      00105           CALL    BIN2HX          ;LSB Startadr. ausgeben
52C1  AF          00106           XOR     A               ;Record Type (00)..
52C2  CD1E53      00107           CALL    BIN2HX          ;ausgeben
52C5  AF          00108           XOR     A
52C6  81          00109 SATZLP    ADD     A,C             ;Checksu. aufaddieren
52C7  4F          00110           LD      C,A
52C8  23          00111           INC     HL              ;HL pt Datenbyte
52C9  7E          00112           LD      A,(HL)          ;Daten..
52CA  F5          00113           PUSH    AF
52CB  CD1E53      00114           CALL    BIN2HX          ;senden
52CE  F1          00115           POP     AF
52CF  DD216354    00116           LD      IX,STARTA
52D3  DD3400      00117           INC     (IX)
52D6  2003        00118           JR      NZ,NUR1X        ;Uebertrag?
52D8  DD3401      00119           INC     (IX+1)          ;Hi Byte erhoehen
52DB  10E9        00120 NUR1X     DJNZ    SATZLP
52DD  81          00121           ADD     A,C             ;Checksu. bilden..
52DE  ED44        00122           NEG                     ;und..
52E0  CD1E53      00123           CALL    BIN2HX          ;ausgeben
52E3  3E0D        00124           LD      A,0DH           ;CR
52E5  CD3353      00125           CALL    SIOOUT
52E8  C9          00126           RET
```

Club 80
INFO 37
Okt. 92


```
              00127
52E9  F5          00128 ERROR     PUSH    AF
52EA  CD2844      00129           CALL    CLOSE
52ED  F1          00130           POP     AF
52EE  F640        00131 SYNERR    OR      40H
52F0  C30944      00132           JP      ERRDSP
              00133
52F3  060C        00134 ENDE      LD      B,12            ;Laenge v. EMSG
52F5  210D53      00135           LD      HL,EMSG
52F8  7E          00136 E1        LD      A,(HL)
52F9  CD3353      00137           CALL    SIOOUT
52FC  23          00138           INC     HL
52FD  10F9        00139           DJNZ    E1
-------------------------------------------------------------------------------
52FF  211953      00140           LD      HL,ZMSG         ;ZEUS aufrufen mit..
5302  C30544      00141           JP      4405H           ;DOS Einsprung
              00142 ;.....................................................
5305  4F          00143 MSOBJ     DM      'OBJ/CMD',0DH
0000              00144 MSOEND    DS      0
530D  3A          00145 EMSG      DM      ':00000001FF',0DH
5319  5A          00146 ZMSG      DM      'ZEUS',0DH
              00147
              00148 ;BIN2HX, gibt das Byte in A als zwei ASCII-Zeichen
              00149 ;'00'..'FF' ueber die serielle Schnittstelle und den
              00150 ;Bildschirm des TRS80 aus. Zerstoert den Inhalt von A.
              00151 ;Beispiel: '?' (3FH) wird als '3F' (33H,46H) ausgegeben.
531E  F5          00152 BIN2HX    PUSH    AF
531F  E6F0        00153           AND     0F0H
5321  0F          00154           RRCA
5322  0F          00155           RRCA
5323  0F          00156           RRCA
5324  0F          00157           RRCA
5325  CD2B53      00158           CALL    NASCII
5328  F1          00159           POP     AF
5329  E60F        00160           AND     0FH
532B  FE0A        00161 NASCII    CP      10
532D  3802        00162           JR      C,NAS1
532F  C607        00163           ADD     A,7
5331  C630        00164 NAS1      ADD     A,'0'
5333  F5          00165 SIOOUT    PUSH    AF
5334  DBEA        00166 RSLP      IN      A,(0EAH)
5336  CB77        00167           BIT     6,A
5338  28FA        00168           JR      Z,RSLP
533A  F1          00169           POP     AF
533B  D3EB        00170           OUT     (0EBH),A        ;Zch. ausgeben
533D  CD3A03      00171           CALL    033AH           ;darstellen
5340  C9          00172           RET
              00173
              00174 ;Variable:
0020              00175 FCB       DS      32              ;File Control Block
0100              00176 FILBUF    DS      256             ;File Buffer
0002              00177 PTSTOR    DS      2               ;Pointer ->RAM
0002              00178 STARTA    DS      2               ;Startadr. f. Intel-Hex
0001              00179 GESBYT    DS      1               ;Gesamt-Bytezaehler
0000              00180 OBJBUF    DS      0               ;Hier beginnt der Buffer
              00181
5200              00182           END     START

00000 Fehler
```

Aufbacken der harten Scheiben.

Wer ein Winchesterlaufwerk in seinem Computer eingebaut hat, sollte ab und zu diesselbe 'backup'-pen. Ich habe auf meinem DOS-Laufwerk 5 zum Beispiel bei der Installation ein defektes Directory angelegt. Nach dem ich gerade alle meine Lieblingsprogramme auf der Scheibe installiert hatte, und eines Abends noch zwei Programme dazutun wollte, meldete der Rechner beim Kopieren plötzlich 'Lesefehler Inhaltsverzeichnis'. Eins der beiden Programme wurde auf die ersten beiden Sektoren des Inhaltsverzeichnisses kopiert und mein Gesicht wurde länger und länger.

Falls mir so etwas noch einmal passieren sollte, will ich gewappnet sein. Das Maschinenprogramm 'BACKUP/CMD' kopiert alle Files, die sich auf der Festplatte befinden, auf 5 Disketten und löscht dann die Bearbeitungskennzeichen der Files auf der Harddisk. Nur die Files, die seit dem letzten Backup neu auf die Platte gekommen sind und solche, die über DOS-Routinen verändert oder erweitert wurden, werden beim nächsten Backup übertragen.

Der 'COPY'-Befehl benutzt hierbei den Parameter 'BEA' für 'Bearbeitungskennzeichen gesetzt' und kopiert eben gerade nur solche Files, bei denen dieses Flag gesetzt ist. Das BK wird allerdings nicht gesetzt, wenn man mit SUPERZAP zum Beispiel sektorweise auf einen File, man müßte hier von Sektor und nicht von File reden, zugegriffen wird.

Wird 'BACKUP <ENTER>' eingegeben, wird von allen 5 Laufwerken ein Backup auf Diskette erstellt, das Programm fordert dann zu 5 Diskettenwechseln auf. Falls so eine Sicherungsdiskette einmalvoll ist, schließlich fassen Festplattenlaufwerke die mit dem 'HDSYSOEA/SRC' von Andreas Magnus betan werden immerhin 1.9MByte, muß man eben eine mehere Backup-Disketten benutzen. Man kann auch nur ein Laufwerk sichern, indem, man die betreffende Laufwerksnummer mit angibt.

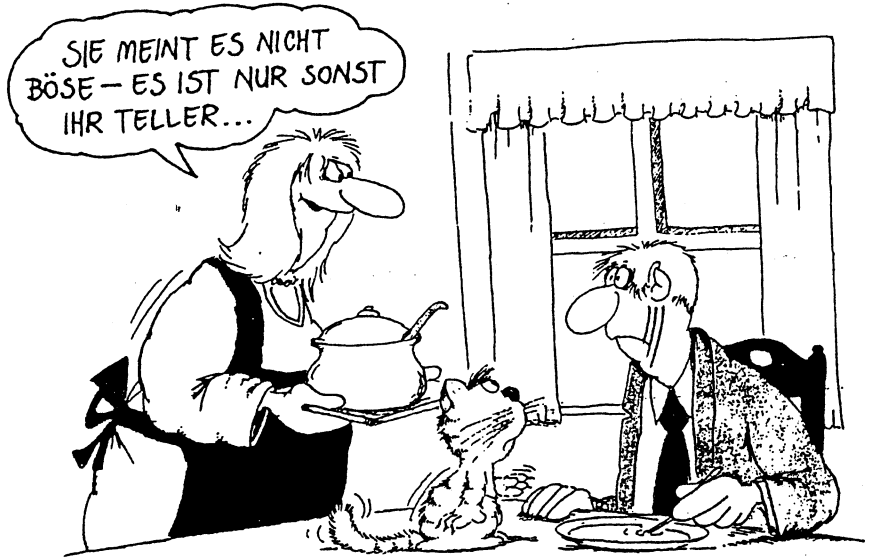
Das Programm ist eigentlich ziemlich einfach, es können sehr viele DOS-Calls benutzt werden. Man könnte dieses Backupprogramm übrigens auch sehr leicht als JOB-File schreiben. Die Startadresse ist 3000h, weil dieser Speicherbereich beim Kopieren garantiert nicht überschrieben wird. Soviel zum Thema Datensicherheit beim Betrieb von Festplattenlaufwerken.

Ein weiterer Vorteil ist noch, daß durch regelmäßiges Sichern zum einen ältere Versionen von Dateien erhalten bleiben und man kann relativ gefahrlos Files löschen wobei man eben sicher sein kann, diese File einfach wiederzufinden, wenn man der Meinung ist, dieses File ist denn nun doch nicht so unnütz wie früher vermutet.

Volker Dan

```
00001 ;       BACKUP/CMD
00002 ;
00003 ;
00004 ; dieses Programm soll von der Festplatte Backup's erstellen.
00005 ; Es tut dies indem nur Files kopiert werden, deren Bearbeitungs-
00006 ; kennzeichen gesetzt ist. Dies ist der Fall, wenn ein File neu
00007 ; in ein Directory eingetragen wurde oder wenn das File verändert
00008 ; wurde. Man wird aufgefordert, die Backup-Disketten 5 bis 9
00009 ; nacheinander einzustecken.
00010 ;
00011 ;
00012 ;
00013 gibaus  EQU     4467h
00014 dosrdy  EQU     402dh
00015 doscal  EQU     4419h
00016 doserr  EQU     4409h
00017 datum   EQU     4470h
00018 ;
00019 ;
00020 ;
00021         ORG     3000h
00022 start   LD      A,(HL)
00023         CP      0dh     ; nur <ENTER> ohne Parameter ?
00024         JR      Z,alle
00025         SUB     '5'             ; gültige LW# sind 5,6,7,8,9
00026         JR      C,error
00027         CP      10
00028         JR      C,eintrag
00029 error   LD      A,32
00030         OR      A
00031         JP      doserr
00032 eintrag ADD     A,'5'   ; restauriert die in Z.25 ausgef. Subtraktion
00033         LD      (cpnum),A
00034         LD      (prnum),A
00035         LD      (lgnum),A
00036         LD      (afnum),A
00037         CALL    backup
00038         JR      ende
00039 alle    LD      BC,0500h        ;5 HD-laufwerke
00040 loop    PUSH    BC              ; BC wegen Registerveränderungen pushen
00041         LD      A,(speich)
00042         INC     A               ; a => '5', usw.
00043         LD      (speich),A      ; zwischenspeichern
00044         LD      (cpnum),A
00045         LD      (prnum),A
00046         LD      (lgnum),A
00047         LD      (afnum),A
00048         CALL    backup
00049         POP     BC              ; BC vom Stack, schön warm und trocken
00050         DJNZ    loop
00051 ende    LD      HL,endtex
00052         CALL    gibaus
00053         JP      dosrdy
00054 endtex  DM      'Der Backupvorgang ist abgeschlossen.',0dh
00055 ;
00056 backup  LD      HL,auffor
00057         CALL    gibaus
00058 loop1   LD      A,(3840h)
00059         CP      01h             ; <ENTER> gedrückt??
00060         JR      NZ,loop1
00061         LD      HL,losgeh
00062         CALL    gibaus
00063         LD      HL,copy
```

```
0064            CALL    doscal
0065            CALL    NZ,doserr
0066            LD      HL,prot
0067            CALL    doscal
0068            CALL    NZ,doserr
0069            LD      HL,buff
0070            CALL    datum   ; datum trägt das aktuelle Datum in 'buff' ein
0071            LD      HL,prot0                ; neuestes Datum auf Disk 0
0072            CALL    doscal
0073            CALL    NZ,doserr
0074            RET
0075 ;
0076 auffor     DM      07h,20h,0bh,'Lege Backup-Disk für HD LW# '
0077 afnum      DM      '5'
0078            DM      ' ein. Dann <ENTER>.',0dh
0079 losgeh     DM      'Backup von Festplattenlaufwerk '
0080 lgnum      DM      '5'
0081            DM      '.',0dh
0082 copy       DM      '> '
0083 cpnum      DM      '5'
0084            DM      ',0,,nfmt,kdwa,edk,bea',0dh
0085 prot       DM      'prot :'
0086 prnum      DB      '5'
0087            DM      ',bkl',0dh
0088 prot0      DM      'prot :0,datum='
0089 buff       DM      '00.00.00',0dh
0090 speich     DM      '4'
0091            END     start
```



Es lebe der Fortschritt.

Als glücklicher Besitzer eines Genie IIIs (übrigens jetzt auch mit Harddisk) genieße ich es sehr, auf dem Bildschirm bis zu 2000 Zeichen im Format 80x25 darstellen zu können. In diesem Bildschirmmodus befindet sich der Rechner fast immer, wenn ich mit ihm herumspiele.

Leider sind die meisten Programmme für diese Rechnergeneration original für den TRS-80 bzw. VideoGenie geschrieben, der ja bekanntlich nur 64x16 Zeichen anzeigt, so auch das Textverarbeitungsprogramm, mit dem dieser Artikel entstand. Wenn ich also so ein älteres Programm nutzen will, muß ich beim GIIIs zunächst den DOS-Befehl '64' eingeben und dann das Programm starten. Habe ich genug von diesem oder jenem Programm, so mußte ich nach Beendigung so einer Session den Dos-Befehl '80' eingeben. Diese beiden Befehle hab ich mir auf die ganz linke und ganz rechte Funktionstaste gelegt, dies beschleunigt den Umschaltvorgang schon ganz schön. Aber das genügte mir nicht.

Ich entsann mich der Organisation von /CMD-Programmen und dem DOS-Call 0033h, der ein Zeichen auf den Bildschirm bringt. Wenn man das Zeichen 10h oder 16d über diese Routine ausgibt, wird der Videocontrollerbaustein umprogrammiert und ein Clear-Screen wird ausgegeben. Danach erfolgt die Bildschirmausgabe im 64x16 Zeichen-Modus. Um den 80x25 Zeichen-Modus einzuschalten, muß das Zeichen 13h (19d) ausgegeben werden.

/CMD-Programme sind folgendermaßen auf Diskette abgespeichert: Das erste Byte ist immer 01h. Dann folgt die Angabe über die Menge der in den Speicher zu ladenden Zeichen plus zwei und dann die Adresse, an der die Zeichen in den Speicher geladen werden sollen. Ein Zähler wird mit dem zweiten Byte geladen, und wenn er auf 0 runtergezählt wurde,zeigt der Zeiger wieder auf so eine 01, was eine neue Menge von Zeichen bedeutet. Ist diese Zeichen aber 02, ist das nächste auch 02. Dies ist die Ende-Kennung für dieses File, danach folgen noch zwei Bytes, die die Startadresse des Programms angeben.

Es gibt noch den Sonderfall, das /CMD-Files zunächst oder auch mitten in der Datei Kommentare enthalten, dann ist dieses erwähnte Zeichen welches übrigens Kennbyte des Object Code Records genannt wird auch einmal 00 oder sogar 03 bis OFFh. Das ist allerdings relativ selten.

Ich habe nun bei einigen mir wichtigen Programmen dieselben so geändert, daß vor dem Programmstart zunächst auf 64x16 Zeichen geschaltet wird und bevor dieselben wieder den Ausgang über DOS-Ready (DOS-Call 402Dh ist hier ziemlich oft verwendet) finden, wird wieder auf 80x25- Zeichen zurückgeschaltet. Der zweite Teil funktioniert nicht bei Programmen, die als Unterprogramm programmiert sind, die enden nämlich mit einem 'RET'. Um so eine Umschaltung in die Files einzuZAPen, benutze man am besten den 'FED' oder noch besser das Programm 'DED', was dieselben Funktionen wie 'FED' kann, allerdings auch Systemfiles bearbeitet.

In ihm wird mit der Funktion 'Z', was das Aufsuchen des nächsten Object Record Headers bedeutet, so lange, bis man den Start Header mit der Kennung 02 02 <startadresse> gefunden hat. Wenn man diese äußerst praktischem Programme nicht hat, muß man halt von Hand den letzten Sektor nach der Endekennung duchsuchen.

Die Umschaltung auf 64x16 in Assembler:

```
3E 10          64x16  LD     A,10h
CD 33 00              CALL   0033h
C3 XX XX              JP     <startadresse>
```

Das ist schon fast alles. Anstatt der Ende Kennung 02 02 usw. muß nun dieses kleine Programm eingezapt werden, erst muß jedoch die erste 02 in 01 umgewandelt werden (nicht Endekennung) und als zweites Byte muß die Anzahl der Objektbytes plus zwei (damit der Zeiger auf dem nächsten Kennbyte steht) eingetragen werden. Es sind also 8 + 2 Bytes, was hexadezimal 0Ah heisst. Jetzt muß die Ladeadresse der kleinen Routine kommen, sie kann entweder nach dem eigentlichen Programm stehen' oder am Ende des Eingabepuffer ab 4318h oder im Prinzip sonstwo. Die 8 Bytes können ja getrost überschrieben werden. Natürlich darf die Routine nicht im Code des eigentlichen Programmmes liegen. Nehmen wir an, das Programm an dem der Zusatz angehängt werden soll, liegt von 5200h bis 5400h. Unser Zusatz kann also zum Beispiel bei A000h liegen, einzutragen ist folgender Code :

```
01 0A  00 A0    3E 10  CD 33 00  C3 00 52
```

Falls man im Besitz des FED oder DED ist, sollte man sich anschauen, wo das letzte Byte des Programms geladen werden soll, und die Routinen ab der nächsten Speicherstelle positionieren. Jetzt soll nach dem Ausstieg des Programms wieder auf 80x25 zurückgeschaltet werden. Das ist etwas komplizierter. Es muß zunächst der Ausgang des Programmes gefunden werden. Hierzu gibt man bei 'FED' folgende Sequenz ein

```
F       - für FIND HEX
0d 42   - für 420dh im Z80-Format, also erst MSB dann LSB
```

Der Cursor sollte dann auf die Bytes 0D 42 zeigen, falls sich das Programm auf diesem Wege verabschiedet. Vor dem 0D 42 muß ein 'C3' stehen, es heißt 'JUMP' und stellt sicher, das man nicht den Code in einer Tabelle gefunden, sondern den tatsächlichen Ausgang. Es können auch mehrer solcher Sprünge nach 402Dh im Programm sein.Hier muß jetzt ein Sprung zu einer Routinen stehen, die zunächst auf 80x25 Zeichen schaltet, und erst dann nach DOS-Ready springt. Sie sieht ähnlich aus wie '64x16', bloss daß das Zeichen, mit dem der Accu geladen wird, ein 13h ist.

Diese Routine darf aber nicht irgendwo stehen, da das Rumpfprogramm diese Stelle ja überschreiben könnte, sondern sie muß an einem sicheren Plätzchen verborgen bleiben. Ich würde das Ende des DOS-Eingabepuffers empfehlen, er ist schließlich 80 Zeichen lang, und so lange DOS-Befehle wird man wohl kaum in einem Programm eingeben. Im folgenden also 80x25:

```
4338  3E 13       LD    A,13h
433A  CD 33 00    CALL    0033h
433D  C3 2D 40    JP      402Dh
```


Dieser Code sieht in der Datei dann folgendermaßen aus

```
01 0A 38 43  3E 13 CD 33 00 C3 2D 40
```

Und jetzt zum Schluß die Endekennung :

```
02 02 00 A0
```

, damit zunächst das dort liegende '64x16' abgearbeitet wird.

Schon fertig.
Aber ich denke die Arbeit lohnt sich. Falls dieses Späsken auf einem anderen Rechner laufen soll, muß erst das Handbuch zu Rate gezogen werden, um die Codes für die Bildschirmumschaltung in Erfahrung zu bringen.

Bei einigen Programmen sind Variationen denkbar, der 'DED' zum Beispiel läßt mitten im Programmtext einige Bytes für Erweiterungen frei, das '80x25' oder sogar auch '64x16' läßt sich also dort unterbringen. Das wärs für heute abend

Happy Hacking

Vath Da

Nie mehr das verfluchte DOS-Handbuch suchen.

Es gibt für das Betriebssystem NewDOS-80 V.2 ein Programm namens 'HELP/CMD' von Matt Wakeley, das zu jeden DOS-Befehl einen mehr oder weniger aussagekräftigen Erläuterungstext auf den Bildschirm bringt. Dieses Programm ist genial einfach programmiert. Der Maschinencode selbst ist nur knapp 400 Bytes lang. Die nächsten zwei Sektoren enthalten eine Liste der DOS-Befehle, jeder Befehl hat maximal 6 Zeichen und danach folgen immer zwei Zeichen, deren Funktion wir später kriegen.

Diese Liste hat als Endekennung ein 00h, und sie kann im Prinzip auch länger als zwei Sektoren sein. Nach den Sektoren mit der Liste folgen mehere Sektoren mit den 'HELP'-Texten, sie sind getrennt durch ein '00h'. Die jeweiligen Texte sind folgendermaßen strukturiert :

Zunächst kommt die Beschreibung des Befehls, dieser Text wird nach der Wiederholung des gesuchten Begriffs angezeigt. Dann folgen Textzeilen, die mit einem Zeichen größer C0h (größer 191) beginnen. Diese Codes stehen für eine bestimmte Menge Leerzeichen, C5h steht für 5 Spaces, das spart halt Platz. Abgeschlossen werden die einzelnen Zeilen mit 0Dh, ENTER. Somit können sie über den DOS-Call 4467h (gibt Zeichenketten bis zur Endekennung 03h oder 0Dh aus) bequem auf den Bildschirm gebracht werden. Nach der letzten Textzeile steht dann auch '00h'. In der ersten Zeile steht immer derBefehl mit allen Parametern.

So ein Hilfe-File wollte ich für G-DOS schon immer haben, bei Befehlen die ich selten benutze, muß ich sonst halt immer das Handbuch rauskramen, und das gibt ja auch nicht immer alles her, was man sich so wünscht. Ich habe mit TSCRIPS also für jeden Befehl aufgeschrieben, was mir wichtig und notwendig erschien. Um Platz zu sparen, habe ich vor jede Zeile, die mit so einem Tabulatorspezialzeichen (CXh) anfangen sollte, einen Stern '*' gemacht. Zwischen den einzelnen Texten habe ich immer ein 0Dh, Carriage Return, gelassen. Als der Text endlich fertig war, schrieb ich mir in Basicprogramm, welches solche 0Dh's in 00h und jeden Stern in 'C5h' umwandelte. Mit 'CREATE DUMMI/CMD,ANZ=5' hatte ich mit ein File erzeugt, welcher genau so groß war, wie der Maschinendcode von 'HELP/CMD' plus 3 Sektoren für die Befehlstabelle, da mein CalvaDOS mehr als 64 Befehle ausführen kann. Mit Superzap kopierte ich die ersten 5 Sektoren von 'HELP/CMD' nach 'DUMMI/CMD' und hängte die Textdatei 'HILF/TXT' mit 'APPEND HILF/TXT DUMMI/CMD' an den Rumpf meines im werden begriffenen 'HILF/CMD'. Die Tabelle ab Sektor 2 im 'HELP/CMD' ist nämlich genial einfach aufgebaut : Nach dem Befehlswort folgen 2 Bytes, die angeben in welchem Sektor ab welchem Byte der gewünschte Hilfe-Text steht.

'HELP/CMD' liesst dann den angegebenen und die folgendenSektoren und gibt die Textzeilen aus, bis die Endekennung 00 erreicht ist. Die Schwierigkeit war dann eben genau dies herauszufinden und in die ersten beiden Sektoren hereinzuZAPen. Aber dank 'FED' war es dann doch relativ schnell getan. Das Programm ließe sich auch noch erweitern, wenn man kleine Hilfen zu Programmen oder eine Liste der Controlcodes für den Videocontroller ab und zu mal braucht. Man schreibt sich den Text, ändert die Sterne in 'C5h', erzeugt ncoh die Endeklennung und hängt das File an 'HILF/CMD' heran. Dann muß nur ncoh der Suchbegriff in der Tabelle und die Position des Hilfstextes im File eingetragen werden. Eine praktische Sache, finde ich. Als Kostprobe hab ich den Text zum Befehl 'DR' ausgdruckt, die Kontollcodes für den Drucker kann und will ich mir einfach nicht in den Schädel hämmern. Das File ist allerddings auch ziemlich lang geworden, bis jetzt knapp 37KByte, es ist also für Festplattenbenutzer prädestiniert. Aber auch auf ein 80 Spuren Diskette passen ja fast 720kByte. Der Abdruck des Programms selbst erscheint mir ziemlich sinnlos, wer Interesse hat, soll sich bei mir melden, er oder sie kann dann die Files zugeschickt bekommen. Im Folgenden also 'HILF DR<ENTER' :

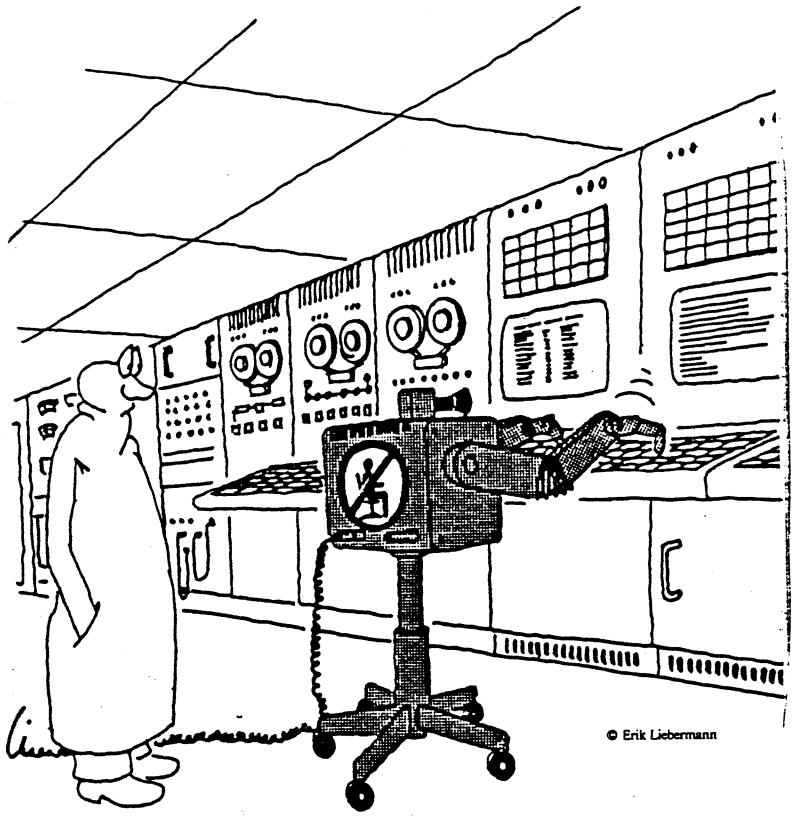

```
HILF (Ver.1) für G-DOS und CalvaDOS 2.4
nach einer Idee von Matt Wakeley

DR --- Bemerkung drucken.
   DR,<$>,<#>,<text>
    ermöglicht die direkte Ausgabe von Text und Sonderzeichen an

   den Drucker. Maximale Zeilenlänge ist 76 Zeichen.
   '$' ersetzt das Zeichen ESC (ASCII 27).
   '#' bedeutet die Umwandlung des nachfolgenden Buchstabens in
   einen Controlcode (z.B. #N ergibt 14).
   Eingige Umschaltsequenzen für Schriftarten:
   DR #§  =>  ist ASCII 00, #A ist ASCII 01
   DR $W#A=>  Doppelt breite Zeichen
   DR $W#§=>  Doppelt breite Zeichen aus
   DR #N  =>  Breitschrift
   DR #O  =>  Komprimierter Druck
   DR #Q  =>  Aufhebung des komprimierten Druckens
   DR #S  =>  Aufhebung der Breitschrift
   DR $P  =>  Elite-Schrift
   DR $§  =>  Drucker-Reset
   DR $-  =>  Unterstreichen
   DR $S  =>  Tieferstellen
   DR $S#§=>  Hochstellen
   DR $E  =>  Hervorgehobener Druck
   DR $F  =>  Ausschalten von $E
   DR $G  =>  Doppeltdruck
   DR $H  =>  Ausschalten von $G
   DR $4  =>  Kursiver Druck
   DR $5  =>  Kursivdruck aus
:7 Workbench >
```

Volk Da

```
00002 ;*****************************************************************
00003 ;       I       N       V       /       C       M       D
00004 ;
00005 ;       INV/CMD zeigt auf dem angegebenen Laufwerk nur die
00006 ;       unsichtbaren Files an.
00007 ;
00008 ;       SYNTAX :=  INV <LW#><ENTER>
00009 ;
00010 ;
00011 ;*****************************************************************
00012 ;
00013 dosrdy  EQU     402dh
00014 doserr  EQU     4409h
00015 debug   EQU     440dh
00016 dirrea  EQU     490ah
00017 gibaus  EQU     4467h
00018 tesdsk  EQU     445eh
00019 ;
00020 ;
00021 ;
00022 ;
00023 ;
00024         ORG     5200h
00025 lwnum   DM      '0'
00026 debuf   DW      1806h   ;18h=24 Zeilen mit 6 Filenames /Zeile
00027 fdeanz  DW      0000h   ; hier steht wieviele Files angezeigt wurden
00028 secanz  DB      01h     ; Anzahl der DIR-Sektoren, wird incrementiert
00029 secbuf  DB      00h     ; Hier steht die Anzahl immer !
00030 ;
00031 ;
00032 ;
00033 ;
00034 error   LD      A,2fh   ; 'schlechte parameter
00035         JP      doserr  ;
00036 istzahl SUB     '0'     ; ascii korrektur
00037         JR      C,error ;
00038         CP      10      ;
00039         JR      C,eindir;
00040         JR      error   ;
00041 ;
00042 ;
00043 ; Hier fängt das Hauptprogramm an
00044 ;
00045 start   LD      IX,fdeanz       ; IX wird der Zähler für die angz.Files
00046         LD      (IX+0),00h      ;
00047         LD      A,(HL)          ;
00048         CP      0dh             ; wenn ENTER, hole lw# von 43a0h
00049         CALL    NZ,istzahl      ;
00050         LD      A,(43a0h)       ; hier aktuelles laufwerk
00051 eindir  LD      (lwnum),A       ; hier die Routine für genau 1 Laufwerk
00052         ADD     A,'0'           ; binär=>ASCII
00053         LD      (dnum),A        ; ist für U-Prog. 'zeile'
00054         CALL    zeile           ; hier wird die Lw# sowie der Name
00055         CALL    diskdir         ; des Lw's auf den Bildschirm gebracht
00056         LD      A,(IX+0)        ; Wieviele Files sind angezeigt ?
00057         CP      00h             ; wenn kein File der Spezi. genügte
00058         JP      NZ,dosrdy       ; dann wird die 'zeile' wieder gelöscht
00059         LD      A,1bh           ; Cursor eine Zeile höher
00060         CALL    0033h           ; auf den Bildschirm
00061         JP      dosrdy          ; jetzt aber Abgang
00062 diskdir CALL    ermittl
00063 dirwei  LD      A,(secanz)
00064         INC     A
00065         LD      (secanz),A      ; 1 erhöhen und zurückschr.
00066         PUSH    HL
00067         LD      HL,secbuf
00068         CP      (HL)
00069         POP     HL
00070         RET     Z               ; hier ist der AUSGANG der U-Routine
00071 dirsec  CALL    dirrea          ; ließt einen Dirsector
00072         JP      NZ,doserr       ; falls ein Fehler auftrat
00073         CALL    auswert
00074         JP      dirwei
00075 ermittl LD      A,01h           ; HIT-Sector
00076         CALL    dirrea
00077         LD      A,(421fh)       ; hier steht die Anzahl der
00078         ADD     A,0ah           ; Dirsec, minus 10 inkl.HIT&GAT
00079         LD      (secbuf),A
00080         RET
00081 auswert LD      BC,0800h        ; 8 Einträge pro Sector
00082 awloop  LD      A,(HL)          ; erstes Zeichen des FDE's
00083         CP      18h             ; ist es ein FPDE ?
00084         JR      NZ,fxde         ; wenn nicht war b7=1 oder b4=0
00085         LD      A,L             ; jetzt soll HL auf den Filename zeigen
00086         ADD     A,05h           ; der ist bei FDE+5
00087         LD      L,A             ; HL auffrischen
00088         CALL    verglei         ; vergleichen und anzeigen
00089         LD      A,1bh           ; 5 weniger als bei keinem Eintrag
00090                                 ; weil der Zeiger im Sektorpuffer
00091                                 ; in Zeile 86 mit 5 addiert wurde
00092         JR      fpde
00093 fxde    LD      A,20h           ; alle 20h Zeichen beginnt ein FDE
00094                                 ; (FileDirectoryEintrag)
00095 fpde    ADD     A,L
00096                                 ; FPDE=FilePrimaryDirectoryEntry
00097                                 ; also erster Eintrag eines Files
00098 ;
00099                                 ; FXDE=FileExtensionDirectoryEntry
00100                                 ; also Erweiterungseintrag
00101         LD      L,A             ; HL zeigt auf den nächsten FDE
00102         DJNZ    awloop          ; bis 8 FDE's verglichen wurden
00103         RET                     ; und wieder eine Ebene höher
00104 verglei PUSH    BC              ; einen Zähler brauch ich selbst
00105         PUSH    HL
00106 eintrag LD      DE,(debuf)
00107         CALL    ausgabe
00108         LD      (debuf),DE      ;
00109         INC     (IX+0)          ; fdeanzahl=fdeanz+1
00110 zurück  POP     HL
00111         POP     BC
00112         RET
00113 dirfert JP      dosrdy
00114 ;
00115 ;
00116 ;
00117 ;
00118 zeile   LD      A,(lwnum)       ;Hier wird die Laufwerksnummer und der
00119         CALL    tesdsk          ;Diskettenname auf dem Bildschirm
00120         JP      NZ,doserr       ; angezeigt.
00121         LD      A,00h           ; Name ist im GAT-Sector an Position
00122         CALL    dirrea          ; Byte D0h bis D7h
00123         LD      HL,42d0h
00124         LD      DE,dname
00125         LD      BC,0008h        ; 8 Zeichen
00126         LDIR
00127         LD      HL,zeitxt
00128         CALL    gibaus          ; den gesamten String ausgeben
00129         RET                     ; ist ein Unterprogramm
```

```
00130 zeitxt  DM      'in '
00131 dnum    DB      00h
00132         DM      ':'
00133 dname   DM      'DISKNAME  ',0dh
00134 ;
00135 ;
00136 ;
00137 ;
00138 ;
00139 ;       Ab hier liegen Unterroutinen, die ich von Andreas Magnus
00140 ;       einfach übernommen habe. Moin, moin Andreas !
00141 ;       Sie stammen aus seinem höchst nützlichen Programm
00142 ;
0143 ;            DIRVERGL/CMD
0144 ;
0145 ;       Und machen die Bildschirmausgabe so schön einfach.
0146 ;
0147 ;UP AUSGABE: gibt den Filenamen in HL auf dem Bildschirm oder Drucker aus
0148 ;            und zwar 6 Files nebeneinander und 24 files pro Schirm
0149
0150 ausgabe LD      B,08h           ;8 Byte Filenamen
0151         LD      C,H             ;H sichern, wird noch gebraucht
0152         CALL    aushl           ;und ausgeben
0153         LD      A,20h           ; ein Space zum Trennen
0154         CALL    ausa
0155         LD      B,03h           ;3 Byte Extension
0156         CALL    aushl           ;auch ausgeben
0157         LD      A,':'
0158         CALL    ausa            ; hier den Doppelpunkt
0159         DEC     E               ;(Filenamen nebeneinander)
0160         LD      H,C             ;H vorsichtshalber zurück
0161         RET     NZ              ;alles klaro
0162
0163         LD      E,6             ;wieder auf 6 stellen
0164         DEC     D               ;Zeilenzähler -1
0165         JR      NZ,neuz         ;neue Zeile anzeigen
0166         LD      D,18h           ;wieder auf 24 Zeilen stellen
0167         EX      DE,HL           ;Zähler sichern
0168         CALL    warte           ;auf Tastendruck warten
0169         EX      DE,HL           ;und Zähler zurück
0170 neuz    LD      A,0dh           ;CR => A
0171         CALL    ausa            ;und anzeigen und zurück
0172         LD      H,C             ;H jetzt entgültig zurück
0173         RET
0174
0175 ;UP AUSHL: gibt B Zeichen lang (HL) ff aus
0176
0177 aushl   LD      A,(HL)          ;Zeichen in den ACCU
0178         CALL    ausa            ;und ausgeben
0179         INC     HL              ;auf nächste Stelle
0180         DJNZ    aushl           ;bis alle Angezeigt
0181         RET
0182
```

```
0183 ;UP AUSA: gib den ACCU auf dem Bildschirm oder Drucker aus
0184
0185 ausa    PUSH    DE              ;Zeilenzähler retten
0186         CALL    0033h           ;oder 3bh für Drucker
0187         POP     DE
0188         RET
0189
0190 ;UP WARTE: wartet bis CR gedrückt ist
0191
0192 warte   CALL    0049h           ;auf Tastendruck warten
0193         CP      0dh             ;war es Enter ?
0194         RET     Z               ;zurück wenn ja
0195         JR      warte           ;weiter warten
0196 ;
0197 ;
0198 ;
0199 ;
0200         END     start
```

Wer suchet, der FINDet.

Auch auf die Gefahr hin mich zu wiederholen:
Ich bin überglücklicher Besitzer einer Harddisk !

Aufbauend auf dem Rumpf meiner Programme 'KOPIERE' und 'ERASE' entschloß ich mich, diese Wildcards auch beim Aufruf des Inhaltsverzeichnisses nutzbar zu machen. Das Programm 'FIND/CMD' kann dies, mit 'FIND APFEL*/*:9' werden alle Dateien, die irgendwas mit Äpfeln zu tun haben (bei mir alles, was mit Mandelbrot zusammenhängt) auf den Bildschirm gebracht.

Außerdem macht mir seit einiger Zeit mein Gedächnis ein wenig zu schaffen, deswegen kann ich mich mit FIND davon überzeugen, ob sich ein Programm, welches ich auf einer meiner Disketten finde und welches ich eigentlich ganz interessant finde schon auf einem meiner Harddiskaufwerke befindet. Dies wird erreicht durch das Weglassen der Laufwerksbezeichnung im Parameterstring. Jetzt werden nämlich die logischen Laufwerke 3 bis 9 nach irgendeinem oder sogar nach meheren Dateien durchsucht, das ist ganz nützlich, wenn man den genauen Namen einer Datei vergessen hat oder wenn man nicht mehr weiß, auf welchem Laufwerk sich die Datei denn nun verbirgt. Gibt man als Laufwerksbezeichner das Zeichen '*' an, so werden auch die Diskettenlaufwerke mit durchsucht, das dauert ja bekanntlich seine Zeit und muß ja nicht immer sein.

Das Programm arbeitet folgendermaßen :
Nachdem festgestellt wurde, welche Laufwerke oder welches denn durchsucht werden soll, wird ein Unterprogramm aufgerufen, daß einen Inhaltsverzeichnissektor nach dem anderen in den Speicher lädt und jeden Eintrag mit der eingegebenen Maske vergleicht. Ist der Dateiname würdig angezeigt zu werden, wird er das auch und zwar mit einer Routine, die ich wieder von Andreas Magnus übernahm. Ich habe den Teil, der fast original aus 'KOPIERE/SRC' stammt, nicht ausgedruckt, er stand ja im letzten INFO schon.

Das Programm ist zwar nicht besonders elegant programmiert, aber es erfüllt halt seinen Zweck. Der einzige Fehler, den ich finden konnte, geht so : Wenn man FIND */* eingibt, so stürzt der Rechner nach einigen ausgegebenen Zeilen einfach ab, wahrscheinlich habe ich die Unterroutinen von Andreas nicht sauber genug eingefügt. Dies wird man aber wohl ohnehin selten tun und ich war bis jetzt zu faul zum Suchen. Meinen Zwecken genügt es vollkommen.

Nach einiger Zeit und dem Besuch eines Freundes, der mir auf seinem IBM-Kompaktiblen das Programm 'WO/COM' zeigte, hatte ich das Gefühl, das mein 'FINDER' einfach zu langsam sei. Ich schloß messerscharf, das es daran liegt, das ich jeden DIR-Sektor nacheinander einlesen mußte, um ein Laufwerk zu durchforsten. Ich errinnerte mich daran, wie das DOS das Problem des Findens löst: Jeder Eintrag im Inhaltsverzeichnis hat seine Entsprechung in der HIT-Tabelle, welche den zweiten Sektor des 'INHALT/SYS' belegt. HIT steht für Hash Index Tabelle. In dieser Tabelle ist zu erkennen, ob ein Eintrag frei oder belegt ist, ist er frei, so ist der Eintrag in der HIT ein 00h. Jedem FDE im Inhaltsverzeichnis ist ein Byte in der HIT zugeordnet, außerdem entspricht die Position in der HIT genau dem DEC (Directory Entry Code), welcher sich über einen DOS-Call bequem einlesen läßt. Der Filename wird nach einem bestimmten und sehr einfachen Algorithmus kodiert und dieser 'Hash-Code' wird dann in der HIT eingetragen. Soll ein File vom DOS gefunden werden, so wird aus dem Filenamen der Hash-Code errechnet und die HIT des jeweiligen Laufwerkes wird nach diesem Code durchsucht. Wird der Code in der Tabelle gefunden, wird der zuständige FDE eingeladen und dieser Filenameneintrag wird mit dem zu findenden verglichen. Ist er nicht gleich, was durchaus nicht unüblich ist, da die Information von 11 Zeichen (FILENAME/EXT) nicht in einem Byte gespeichert werden kann, wird die HIT weiter durchsucht. Wenn der errechntete Hash-Code gar nicht in der HIT zu finden ist, kann sich das File nicht auf der Diskette befinden.

Diese Funktion realisiert das Programm 'WO/CMD'. Es durchsucht alle angeschlossenen Laufwerke in aufsteigender Folge nach einem einzigen Dateinamen. Findet es diesen, so wird eine Meldung auf dem Bildschirm ausgegeben. Dieser Vorgang ist erheblich schneller als der beim 'FIND/CMD' angewendetet, so daß jetzt immer auch die Disketenlaufwerke mit durchsucht werden. Es kann selbstverständlich keine Wildcards im Eingabestring akzeptieren.

Diese Programme sind zwar nur für Leute mit Harddisk interessant (das FIND stellt allerdings auch ein 'DIR */*:!w#' zur Verfügung) aber ich hoffe daß der oder die eine so ein Gerät eingebaut hat. Ansonsten muß der oder die das eben bald tun.

Mit wunden Augen und platten Fingern

Schöne Grüße aus dem Norden

Vadda Dan

```
00001 ;
00002 ;
00003 ;       F       I       N       D       /       C       M       D
00004 ;
00005 ;       soll auf der Hard-, Ram- und eventuell Floppydisk
00006 ;       Programme finden, ähnlich einem DIR *:*:*
00007 ;       also Filenamen, Extensions und auch Laufwerknummer
00008 ;       können beliebig angegeben werden.
00009 ;       Die Directoryunterfunktionen I(-nvisible),S(-ystem),
00010 ;       A(-lles) werden nicht implementiert.
00011 ;       File ist File. Die Laufwerksangabe ist * für alle,
00012 ;       bzw. explizite Lw#-angabe.
00013 ;       Man kann auch den :* weglassen und enfach ENTER nach der
00014 ;       Filespezifikation angeben
00015 ;
00025 dosrdy  EQU     402dh
00026 doserr  EQU     4409h
00027 debug   EQU     440dh
00028 dirrea  EQU     490ah
00029 gibaus  EQU     4467h
00030 tesdsk  EQU     445eh
00031 ;
00032 ;
00033 ;
00034 ;
00035 ;
00036 ;       Dies ist ein Teilstück des Programmes
00037 ;               KOPIERE/CMD
00038 ;       und zwar wird hier die richtige Eingabe des Filenamens
00039 ;       kontrolliert, und zwar Test auf ?,* inklusive.
00040 ;       Man kann jetzt allerdings auch Zahlen und das Zeichen '_'
00041 ;       eingeben(allerdeings auch an erster Stelle, was eigentlich
00042 ;       nicht in Ordnung ist).
00043 ;
00044         ORG     5200h
00045 fname   DM      'FILENAME';wird vom Progr. eingesetzt
00046 exten   DM      'EXT'     ;hier steht dann der Vergleichsstring
00047 ende    DM      0dh
00048 lwnum   DM      '0'
00049 debuf   DW      1806h   ;18h=24 Zeilen mit 6 Filenames /Zeile
00050 fdeanz  DW      0000h   ; hier steht wieviele Files angezeigt wurden
00051 secanz  DB      01h     ; START der FDE-Sektoren in DIR/SYS
00052 secbuf  DB      00h     ; Hier wird eingetragen, wie viele DIR-Sektoren
00053 ;
00151 ;  Hier fängt das Hauptprogramm an
00152 ;
00153 progr   LD      IX,fdeanz       ; IX wird der Zähler für die angz.Files
00154         LD      (IX+0),00h
00155         INC     HL      ; zeigt jetzt auf lwnummer
00156         LD      A,(HL)
00157         CP      '*'
00158         JP      Z,alle
00159         CALL    istzahl
00160 eindir  LD      (lwnum),A       ; hier die Routine für genau 1 Laufwerk
00161         ADD     A,'0'           ; binär=>ASCII
00162         LD      (dnum),A        ; ist für U-Prog. zeile
00163         CALL    zeile           ; hierwird die Lw# sowie der Name
00164         CALL    diskdir         ; des Lw's auf den Bildschirm gebracht
00165         LD      A,(IX+0)        ; Wieviele Files sind angezeigt ?
00166         CP      00h             ; wenn kein File der Spezi. genügte
00167         JP      NZ,dosrdy       ; dann wird die 'zeile' wieder gelöscht
00168         LD      A,1bh           ; Cursor eine Zeile höher
00169         CALL    0033h           ; auf den Bildschirm
00170         JP      dosrdy          ; jetzt aber Abgang
00171 diskdir CALL    ermittl
00172 dirwei  LD      A,(secanz)
00173         INC     A
00174         LD      (secanz),A      ; 1 erniedriegen und zurückschr.
00175         PUSH    HL
00176         LD      HL,secbuf
00177         CP      (HL)
00178         POP     HL
00179         RET     Z               ; hier ist der AUSGANG der U-Routine
00180 dirsec  CALL    dirrea          ; ließt einen Dirsector
00181         JP      NZ,doserr       ; falls ein Fehler auftrat
00182         CALL    auswert
00183         JP      dirwei
00184 ermittl LD      A,01h           ; HIT-Sector
00185         CALL    dirrea
00186         LD      A,(421fh)       ; hier steht die Anzahl der
00187         ADD     A,0ah           ; Dirsec. minus 10 inkl.HIT&GAT
00188                                 ; + noch einen weiß der Geiger warum
00189         LD      (secbuf),A
00190         RET
00191 auswert LD      BC,0800h        ; 8 Einträge pro Sector
00192 awloop  LD      A,(HL)          ; erstes Zeichen des FDE's
00193         AND     90h             ; alle außer b7 & b4 werden gelöscht
00194         CP      10h             ; ist es ein FPDE ?
00195         JR      NZ,fxde         ; wenn nicht war b7=1 oder b4=0
00196                                 ; herumrechnen zu müssen.
00197         LD      A,L             ; jetzt soll HL auf den Filename zeigen
00198         ADD     A,05h           ; der ist bei FDE+5
00199         LD      L,A             ; HL auffrischen
00200         CALL    vergiei         ; vergleichen und anzeigen
00201         LD      A,1bh           ; 5 weniger als bei keinem Eintrag
00202         JR      fpde
00203 fxde    LD      A,20h
00204 fpde    ADD     A,L
00205         LD      L,A             ; HL zeigt auf den nächsten FDE
00206         DJNZ    awloop          ; bis 8 FDE's verglichen wurden
00207         RET                     ; un wieder eine Ebene höher
00208 vergiei PUSH    BC              ; einen Zähler brauch ich selbst
00209         PUSH    HL
00210         LD      B,11            ;
00211         LD      DE,fname        ; Vergleichsfilespezifikation
00212
00213 verloop LD      A,(DE)
00214         CP      '?'
00215         JR      Z,veregal               ; wenn ?, dann ist alles erlaubt
00216         CP      (HL)            ; ist es gleich dem FDE ?
00217         JR      NZ,zurück       ; falls nciht, noch mal POPen
00218 veregal INC     DE
00219         INC     HL              ; beide Zeiger ein Weiter
00220         DJNZ    verloop         ; 11 zeichen
00221         POP     HL
00222         PUSH    HL
00223 einträg LD      DE,(debuf)
00224         CALL    ausgabe
00225         LD      (debuf),DE
00226         INC     (IX+0)          ; fdeanzahl=fdeanz+1
00227 zurück  POP     HL
00228         POP     BC
00229         RET
00230 dirfert JP      dosrdy
00231 ;
00232 ;
```

```
00233 ;
00234 ;
00235 zeile   LD      A,(lwnum)       ;Hier wird die Laufwerksnummer und der
00236         CALL    tesdsk          ;Diskettenname auf dem Bildschirm
00237         JP      NZ,doserr       ; angezeigt.
00238         LD      A,00h           ; Name ist im GAT-Sector an Position
00239         CALL    dirrea          ; Byte D0h bis D7h
00240         LD      HL,42d0h
00241         LD      DE,dname
00242         LD      BC,0008h        ; 8 Zeichen
00243         LDIR
00244         LD      HL,zeitxt
00245         CALL    gibaus          ; den gesamten String ausgeben
00246         RET                     ; ist ein Unterprogramm
00247 zeitxt  DM      'in :'
00248 dnum    DB      00h
00249         DM      '
00250 dname   DM      'DISKNAME ',0dh
00251 ;
00252 ;
00253 ;       Einsprung in 'alle', wenn nur RAMDisk und HARDDISK durchsucht
00254 ;       werden soll
00255 abram   LD      A,03h           ; RAMDISK ist LW# 3
00256         JR      aloop           ; und sonst genau wie alle
00257 ;
00258 alle    LD      A,00h           ; alle heißt das ein oder mehrer Files
00259 aloop   LD      (lwnum),A       ; auf allen verfügbaren Laufwerken gesuch
00260         CALL    tesdsk          ; ist überhaupt erreichbar ?
00261         JR      NZ,is_nich      ; zerobit =0 bei keinem fehler
00262 is_doch LD      A,(lwnum)
00263         ADD     A,'0'           ; binär => ASCII
00264         LD      (dnum),A        ; in zeitxt
00265         LD      (IX+0),00h      ; fdezähler
00266         CALL    zeile           ; Diskname usw.
00267         CALL    diskdir         ; Inhaltsverzeichnis durchstöbern
00268         LD      A,(IX+0)        ; ist mindestens ein File gefunden worden
00269         CP      00h
00270         JR      NZ,is_nich
00271         LD      A,1bh           ; cursor eine Zeile höher
00272         CALL    0033h
00273 is_nich LD      A,0bh           ; zeilenvorschub wenn nicht schon
00274         CALL    0033h
00275         LD      A,01h           ; bei 'alle' muß jedesmal der secanz neu
00276         LD      (secanz),A      ; auf den ersten DEC-Sektor gesetzt werdn
00277         LD      A,(lwnum)
00278         INC     A
00279         LD      (lwnum),A
00280         CP      0ah             ; 9 laufwerke durch ?
00281         JR      NZ,aloop        ; sonst weiter suchen
00282         JP      dosrdy
00283 ;
00284 ;
00285 ;       Ab hier liegen Unterroutinen, die ich von Andreas Magnus
00286 ;       einfach übernommen habe. Moin, moin Andreas !
00287 ;       Sie stammen aus seinem höchst nützlichen Programm
00288 ;
00289 ;          DIRVERGL/CMD
00290 ;       Und machen die Bildschirmausgabe so schön einfach.
00291 ;
00347         END     start
```



```
00001 ;
00002 ;       WO/CMD
00003 ;
00004 ;       Programm, um genau ein bestimmtes Programm auf allen
00005 ;       angeschlossenen Laufwerken zu finden.
00006 ;       Der Suchvorgang funktioniert mit der Errechnung des
00007 ;       HASH-Codes, es muß von jedem Laufwerk also nur der HIT-
00008 ;       Sector eingelesen werden.
00009 ;       Das geht natürlich erheblich schneller.
00010 ;
00011 dosrdy  EQU     402dh
00012 gibaus  EQU     4467h
00013 doserr  EQU     4409h
00014 debug   EQU     440dh
00015 tstdsk  EQU     445eh
00016 dirrea  EQU     490ah
00017 getfde  EQU     4936h
00018 ;
00019         ORG 5200h
00020 filspe  DM      20h,20h,20h,20h,20h,20h,20h,20h; 8 Zeichen Filespec.
00021 extspe  DM      20h,20h,20h     ; 3 Zeichen für Extension
00022 aktlw   DM      00      ; Suche starten bei LW# 0, wird hochgezählt
00023 hashco  DM      00h     ;hier wird der errechnete Hashcode eingetragen
00024 ;
00025 start   PUSH    HL      ; zeigt auf den eingegenben Filenamen
00026         LD      DE,filspe
00027         LD      B,09h   ; 8 Zeichen für                 "
00028 nloop   LD      A,(HL)
00029         CP      0dh     ; ENTER ?
00030         JR      Z,rechne
00031         CP      '/'     ; Trennzeichen zur Extension ?
00032         JR      Z,ext
00033         LD      (DE),A
00034         INC     DE
00035         INC     HL
00036         DJNZ    nloop   ; bis 8 Zeichen in Buffer übertragen(Höchstens)
00037         CP      '/'     ; folgt jetzt der slash ?
00038         LD      A,13h
00039         JP      NZ,doserr
00040 ext     LD      B,03h   ; 3Zeichen für Extension
00041         INC     HL
00042         LD      DE,extspe
00043 eloop   LD      A,(HL)
00044         CP      0dh
00045         JR      Z,rechne
00046         LD      (DE),A
00047         INC     DE
00048         INC     HL
00049         DJNZ    eloop
00050         LD      A,(HL)
```

```
00051          CP      0dh     ; hernach also hier muß!! Enter sein
00052          LD      A,13h
00053          JP      NZ,doserr
00054 rechne   LD      HL,filspe       ; diese Routine errechnet den HASH-Code
00055          LD      B,0bh           ; sie ist abgeschrieben aus SYS2/SRC
00056          XOR     A
00057 rloop    XOR     (HL)
00058          INC     HL
00059          RLCA
00060          DJNZ    rloop
00061          JR      NZ,fertig
00062          INC     A
00063 fertig   LD      (hashco),A      ; abspeichern.
00064 ;        Der HASH-Code ist errechnet, jetzt müssen alle HIT-Sectoren
00065 ;        eingelesen und der Inhalt derselben mit dem hashco verglichen
00066 ;        werden. Dann muss der jeweilige FDE geholt werden und mit
00067 ;        filspe verglichen werden. Erst dann ist der Filename gefunden.
00068 ;
00069 drloop   LD      A,(aktlw)       ; LW# laden
00070          CALL    tstdsk          ; ist eine Disk eingelegt ??
```


```
00071          CALL    Z,liehit        ; wenn ja, lies den HIT-Sector
00072          LD      A,(aktlw)
00073          INC     A
00074          CP      0ah             ; schon über 9 hinaus ?
00075          JP      Z,ende
00076          LD      (aktlw),A
00077          JR      drloop
00078 ende     LD      HL,text1
00079          CALL    gibaus
00080          JP      dosrdy
00081 liehit   LD      A,01h           ; HIT-Sector
00082          CALL    dirrea
00083          JP      NZ,doserr       ; falls Fehler(sehr unwahrscheinlich)
00084          LD      B,0ffh
00085          LD      A,(hashco)
00086 sloop    CP      (HL)            ; sucht den gesamten Sectorpuffer ab
00087          CALL    Z,su_fde        ; wenn der hashcode stimmt, muß der
00088          INC     HL              ; zuständige FDE mt filspe verglichen
00089          DJNZ    sloop           ; werden.
00090
00091          RET
00092 su_fde   PUSH    HL
00093          PUSH    AF
00094          PUSH    BC
00095          LD      A,L             ; L ist LSB des Sectropufferzeigers,
00096          CALL    getfde          ; und ist genau der FDE im Directory
00097          LD      DE,filspe
00098          LD      BC,0005h        ; FILENAME steht an 5.ter Stelle
00099          ADD     HL,BC           ; jetzt zeigt HL auf den FILENAMEEXT
00100          LD      B,0bh
00101 fdloop   LD      A,(DE)
00102          CP      (HL)
00103          JR      NZ,nein
00104          INC     DE
00105          INC     HL
00106          DJNZ    fdloop
00107 ja       LD      A,(aktlw)
00108          ADD     A,'0'           ; binär => ASCII
00109          LD      (lwnum),A
00110          LD      HL,text2
00111          CALL    gibaus
00112 nein     LD      A,01h           ; der HIT-Sector muss wieder eingelesen
00113          CALL    dirrea          ; werden, er wurde überschrieben
00114          POP     BC
00115          POP     AF
00116          POP     HL
00117          RET
00118 text2    DM      'File gefunden auf LW# '
00119 lwnum    DM      '0.',0dh
00120 text1    DM      'Alle Laufwerke sind durchsucht.',0dh
00121          END     start
```

Das Schlußwort unseres Club-Freundes Jens, war in der 35.Ausgabe des Club-Infos ein Aufruf an alle Computerbesitzer die mit BASIC arbeiten.

Auch ich wäre der Meinung, daß man sich innerhalb des CLUB 80 zu einer BASIC-Anwendergruppe zusammentun sollte. Ich selber arbeite seit 8 Jahren mit G-DOS der Firma Trommelschläger, wovon ich nicht abzubringen bin. Mit diesem G-DOS,was ja die Ableitung bzw. Erweiterung von NEWDOS 80 ist, habe ich mich sehr stark angefreundet.

Wer hat Interesse an der Gründung einer BASIC-Anwendergruppe ?

Meldet Euch bitte bei: Willi Johnen, Hansemannstr.1
5160 Düren, Tel. 02421/501305

BASIC

Liebe Basic-Freunde,

wie die Zuschriften hier im INFO und auch Gespräche zeigten, ist BASIC ein Treffpunkt, an dem alle Computer unseres Club's wieder zusammenfinden.

Mich freut es ganz besonders, wieder einmal eine Gemeinsamkeit innerhalb des Club's -trotz der verschiedensten Rechnertypen- gefunden zu haben. Aus diesem Grund ist es, bei entsprechender Beteiligung, sinnvoll eine BASIC-Ecke ins Leben zu rufen. Interessenten melden sich bitte bei Willi Johnen oder bei mir.

Sicherlich wird es nicht ganz einfach sein, die verschiedenen BASIC-Dialekte unter einen Hut zu bringen. Was bei dem einen als Befehl im Basic schon vorhanden ist, kann ein anderer erst mit Hilfe seiner Toolkiste ausführen. Hier liegt, meiner Meinung nach, ein wichtiger Aktivitätspunkt der BASIC-Anwendergruppe:
Bereitstellen von Tools, die die Arbeit erleichtern und sich in andere Basicprogramme einfügen oder anpassen lassen.
Es muß ja nicht jeder seine eigene ...xy-Routine geschrieben haben. Gleichzeitig können durch die Gruppe Anregungen zur Optimierung gemacht werden. Somit haben eigentlich alle Beteiligten einen Nutzen an dieser Aktion.

Ich hoffe auf vielfältige Aktivität und freue mich schon auf die nächsten Veröffentlichungen -und das nicht nur wegen der CLUB-INFO!

Jens Neueder

B A S I C - Neuen Wind für eine totgeglaubte Sprache

Mittlerweile gibt es viele, viele schlaue, hilfreiche, übersichtliche, effektive, schnelle und was weiß ich nicht noch alles für Programme und Programmiersprachen, so daß kaum noch Eine(r) von der antiquarisch-amateurhaft anmutenden Sprache BASIC redet.

S C H A D E !

Jens hat sich im Info Nr. 36 etwas verschämt dazu bekannt und ich möchte diese Anregung einer BASIC-Anwendergruppe aufgreifen und für das INFO eine 'BASIC-Ecke' eröffnen.

'Damals' habe ich das Programmieren auch mit BASIC gelernt, daher liegt mir diese Sprache verständlicherweise immer noch sehr leicht auf der Zunge. Hauptsächlich aus folgenden Gründen heraus ist sie bei mir auch jetzt noch beliebt, wenn auch nicht mehr für alle Anwendungen:

- ich kann Ein- und Ausgabedialoge komplett steuern und so gestalten, wie ich es schön und praktisch finde
- ich kann Fehler abfragen und das Programm ohne Abbruch weiterführen

Um dies verständlicher zu machen, gibt es jetzt ein paar Beispiele (bekanntlich läßt man sich davon ja am besten stimmulieren) und ich bin der festen Überzeugung, daß es dann zukünftig nur so von Beispielen, Anfragen, Tips und Tricks aus den CLUB-Reihen für unsere neue Ecke hagelt...

Euer Christof

Christof Neumann
Zeitblomstr. 22/2
7900 Ulm
Tel. 0731/6022568

P.S.: Meine Adresse lautet richtig:

1. ALLGEMEINES VORWEG

Grundsätzlich habe ich die meiste Zeit mit einem Microsoft-BASIC namens MBASIC unter dem Betriebssystem CP/M gearbeitet; alle Programmierbeispiele sind damit verfaßt. Mein Computer hört auf den Namen 'Tandy Model II' und alle zitierten Steuerzeichen sind für ihn gedacht.

Je nach Computer und verwendeter BASIC-Version ergeben sich also Unterschiede, die zu beachten sind.

Je größer und rechenintensiver ein BASIC-Programm ist, umso wichtiger wird die Übersetzung des BASIC-Quellprogrammes in die Maschinensprache: Das COMPILIEREN. Hierzu gibt es spezielle Software, wobei mir allerdings ein Überblick über Angebot, Leistungen und Möglichkeiten fehlt; ich weiß nur, daß mein COMPILER auch auf meinem Model IVp mit dem Montezuma-CP/M läuft.

2. STEUERUNG DER DIALOGE

Ich habe es ganz gerne, wenn meine Programme idiotensicher auch von anderen bedient werden können. Dazu ist nicht nur eine Bedienungsanleitung und ein eindeutiger Bildschirmdialog nötig, sondern obendrein auch - je nach Bedarf - eine Abfrage dessen, was dem Computer eingepickt wird.

In dem folgenden Beispiel sind ein paar grundsätzliche Definitionen aufgelistet, die bei fast allen Programmen so dastehen.

```
DEF FNCU$(X,Y)  =  Positionierung des Cursors auf X,Y
                (das geht bei anderen Rechnern sicher einfacher)
EIN$            =  Inverse Bildschirmdarstellung ein
AUS$            =  dasselbe wieder aus
CLS$            =  gesamten Bildschirm löschen
```

```
170 WIDTH 255
175 DIM BV$(7),BVK$(7),GMNR(7),GMKG$(7),KGMK$(7),ANFL(7),SBET#(7),SBEI#(7)
180 DEF FNCU$(X,Y)=CHR$(27)+CHR$(61)+CHR$(32+Y)+CHR$(32+X)
190 EIN$=CHR$(14):AUS$=CHR$(15):CLS$=CHR$(26)
200 PRINT CLS$
210 PRINT FNCU$(9,2);EIN$;SPC(62)
220 FOR I=1 TO 5 :PRINT FNCU$(9,2+I);"  ";FNCU$(69,2+I);"  ":NEXT
230 PRINT FNCU$(9,8);SPC(62);AUS$
```

Die Zeilen 210 - 230 malen mir ein kleines Eingabefenster auf den Bildschirm.

Jetzt kommt ein Eingabebeispiel, wo ich an einer vorgegebenen Stelle auf dem Bildschirm (Zeile 1680) ein Zahl mit maximal 2 Stellen eingeben lasse (andere Zeichen sind nicht zugelassen):

```
1670 FL$=""
1680 PRINT FNCU$(41,Y%+5);"";
1690 ANT$=INKEY$:IF ANT$="" THEN 1690
1700 IF ANT$=CHR$(13) THEN 1770
1710 IF ANT$=CHR$(8) AND LEN(FL$)>0 THEN FL$=LEFT$(FL$,LEN(FL$)-1):GOTO 1740
1720 IF ASC(ANT$)<48 OR ASC(ANT$)>57 THEN 1690
1730 IF LEN(FL$)<2 THEN FL$=FL$+ANT$
1740 PRINT FNCU$(41,Y%+5);"  "
1750 PRINT FNCU$(41,Y%+5);FL$;
1760 GOTO 1690
```

Weitere Möglichkeiten, die denkbar sind:

- Umwandlung von Kleinbuchstaben automatisch in Großbuchstaben (z. Bsp. für Dateinamen)
- freies Definieren von Sonderzeichen (z.Bsp. wenn man Komma eingibt, wandelt der Rechner dies in Punkt um)

Der Phantasie sind keine Grenzen gesetzt. Da dies bei der Eingabe von vielen Werten im Programm einen Wust von Programmtipperei gibt, kann man das ganze auch als Unterroutine anspringen; hier z. Bsp. zur Eingabe von Texten mit einer Eingabemaske, die die Wortlänge mit der Variablen 'EL' beschränkt:

```
9000 '     --> UP Eingabe Texte <--
9010 PRINT FNCU$(EX,EY);">";STRING$(EL,"_");"<"
9020 PRINT FNCU$(EX+1,EY);"";
9030 LINE INPUT "",E$
9040 E$=LEFT$(E$,EL):RETURN
```

3. FEHLERBEHANDLUNG

In dem folgenden Beispiel wird vorausgesetzt, daß am Programmanfang irgendwo ein 'ON ERROR GOTO 10000' steht. Findet das Programm ein bestimmtes File auf der Diskette nicht, gibt es normalerweise eine Fehlermeldung und das Programm bricht ab. Hier wird in Zeile 10010 stattdessen bei Auftreten des Fehlers (Fehlernummer 53) die gewünschte Datei angelegt und eine Null reingeschrieben. Alle anderen Fehler werden mit Nummer und Programmzeile angegeben und das Programm steigt aus.

```
10000 '    --> Fehlerroutine  <--
10010 IF ERR=53 THEN CLOSE:OPEN "O",#1,DAT$:PRINT #1,"0":CLOSE:RESUME
10020 PRINT FNCU$(17,22);EIN$;"Fehler Nr. ";ERR;" in Zeile ";ERL;AUS$::END
```

4. AUSBLICK AUF WEITERE THEMEN

Ich möchte in einer späteren Folge auf die Themen

- Schnelligkeit durch richtiges Programmieren
- Sortieren mit BASIC - mit welcher Logik?

eingehen und außerdem interessiere ich mich auch noch für das Ansprechen der Schnittstellen mit BASIC. Wer kann mir dabei mit seinen Erfahrungen, Literatur, Beispielen etc. hilfreich unter die Arme greifen ???

Hallo alle GENIE III S -Besitzer !!

Ich hoffe Ihr habt geneuso wie ich die Karnevals- bzw. Faschingstage gut überstanden, was auch der Grund war, wieso ich Euch in der Nr.36 des CLUB 80-INFOS habe 'hängen' lassen. Da ich in Karneval sehr aktiv bin, war ich nicht in der Lage meinen Beitrag zum CLUB-Info zu schreiben. Ich bitte Euch herfür Verständnis zu haben.

In dieser Ausgabe liefere ich Euch den III.Teil des ADRESS-TEXT-90, die Adressenverwaltung. Mit diesem Programm könnt Ihr

*** ca. 2000 Adressen speichern

*** den gesamten Adressenbestand sortieren nach Familiennamen

*** eine Adressenliste formatiert ausdrucken

Hinweise: Dieses Programm läuft nur mit hochauflösender Graphik und einer --------- installierten Festplatte (LW5 + LW6)

Nachfolgende Zeilen müssen jenachden geändert werden:
2.0150 Hochauflösende Graphik
2.0160 Laufwerksangabe
2.1670 Laufwerksangabe
2.1700 Laufwerksangabe
2.1730 Laufwerksangabe
2.1740 Laufwerksangabe

Die Hochauflösende Graphik zu diesen Programm wurde bereits in Nr.34 des CLUB-INFOS vorgestellt.

Bereits erfolgte Veröffentlichungen:

I.Teil - Club-Info Nr.34: Textverarbeitung
II.Teil - Club-Info Nr.35: Startprogramm und Hauptmenü

```
10 '#)) Programmname  (( JOHADRES/BAS ))                 ((#
20 '#)) Adressenverwaltung des ADRESS-TEXT-90 - Privat   ((#
25 '#)) (C)opyright by W.Johnen,5160 Düren               ((#
30 '#)) Stand: 19.01.1992          12.00 UHR             ((#
40 '#)) Hochauflösende Graphik                           ((#
50 '#)) Graphikbild:  )JOHMEN01/RDL(                     ((#
60 '#)) Datenspeicherung in LW 6                         ((#
70 '#)) Sortierprogramm                                  ((#
80 '#)) M I T      H A R D D I S K                       ((#
90 HOFF:HCLS:HON:CLEAR10000:DEFSTRA-D,G-H,J-K:DEFINTN,X,Z,W:DEFSNGY:DIMDX(7)
100 '#---------------------- Graphik $ -----------------#
110 CC=STRING$(15,140):CY=STRING$(79,32):KR=STRING$(20,144):KY=STRING$(30,144):KC=STRING$(16,144)
120 AD="NEUEINGABE":AE="ÄNDERUNG":AI="LISTE-DRUCKEN":AF="SORTIERUNG"
130 HX="####"
140 '#--------Datendiskette prüfen ---------------------#
150 CLS:HLOAD"JOHMEN01/RDL":WP=0:HBIG30,225,1,1,1,'* ADRESSEN - VERWALTUNG *':HBIG536,225,1,1,1,LEFT$(TIME$,8);:HBIG536,202,1,1,1,RIGHT$
(TIME$,8);:IFWA=6THENRETURN
160 GA="":ONERRORGOTO170:OPEN"I",1,"JOHADRES/DAT:6":CLOSE:GOTO220
170 RESUME180
180 PRINT@405,"Im Laufwerk 6 ist keine  (JOHADRES/DAT - Datei)":PRINT@494,"Prüfen sie nochmal mit ( W )";
190 GA=INKEY$:IFGA="w"THENGOSUB200:GOTO160ELSE190
200 X7=405:FORX=1TO6:PRINT@X7,CY;:X7=X7+80:NEXTX:RETURN
210 '#-------- Erstellung der Programmnamen ------------#
220 K1="1 Anrede              : ":K2="2 Vorname/Firmenbez.    : ":K3="3 Name/Firmenname       : ":K4="4 Straße/Postf./Hausnr. : ":K5="
5 PLZ/Wohnort          : ":K6="6 Telefonnummer         : ":K7="7 Ansprechpartner       : "
230 GOTO260
240 '#---------------- Menüanzeige ---------------------#
250 CLOSE:WA=6:GOSUB150:WA=0
260 ONERRORGOTO0:PRINT@461,CC:PRINT@541,"PROGRAMM - MENÜ":PRINT@621,CC:PRINT@779,"(1) ";AD:PRINT@939,"(2) ";AE:PRINT@1099,"(3) ";AI:PRIN
T@1259,"(4) ";AF:PRINT@1419,"(5) HAUPT-MENÜ";
270 WU=VAL(INKEY$):IFWU(1ORWU)5THEN270ELSEONWUGOTO300,360,810,1640,280
280 CLEAR50:RUN"JOHMENÜ/BAS"
290 '#---------- Neueingabe von Daten ----------------#
300 CLS:WU=0:GOSUB1560:PRINT@131,"Neueingabe";:GOSUB1530:GOSUB1200
310 LSETDX(1)=DT(1):LSETDX(2)=DT(2):LSETDX(3)=DT(3):LSETDX(4)=DT(4):LSETDX(5)=DT(5):LSETDX(6)=DT(6):LSETDX(7)=DT(7)
320 PUT1,LQ:CLOSE
330 WA=0:GOSUB1600:PRINT@1848,"( W ) eitere Neueingaben machen          ( M ) zurück zum Menü";CHR$(7);
340 GA=INKEY$:IFGA="w"THEN300ELSEIFGA="m"THEN250ELSE340
350 '#---------- Ändern von Daten --------------------#
360 CLS:GOSUB1560:PRINT@131,"Ändern";
370 IFGO="y"THEN400
380 GOSUB1590:GOSUB1620:PRINT@1842,"Nach welchem Suchbegriff wollen Sie suchen ? ";KR:PRINT@1887,;:LINEINPUTGN:IFLEFT$(GN,1)="/"THEN250
390 IFGN=""THEN400ELSEIFLEN(GN))30THEN380
400 WP=9:GOSUB1530:WP=0
410 IFEOF(1)THENGOSUB770:CLOSE:GOTO360
420 IFGO="y"THENX1=X4:GO="":X4=0
430 X1=X1+1:GET1,X1
440 K2=DX(1)+DX(2)+DX(3)+DX(4)+DX(5)+DX(6)+DX(7)
450 PRINT@141,USING":#### ";X1;:IFINSTR(K2,GN))0THEN470ELSE410
460 '#--- Änderung / Bildschirmanzeige des D-Block----#
470 GOSUB1590:PRINT@321,K1;DX(1):PRINT@481,K2;DX(2):PRINT@641,K3;DX(3):PRINT@801,K4;DX(4):PRINT@961,K5;DX(5);:PRINT@1121,K6;DX(6):PRINT@
1281,K7;DX(7)
480 PRINT@1841," (---( Z ) - ( V )---) - ( N )eusuchen - ( Ä )ndern - ( L )öschen - ( M )enü ";:ONERRORGOTO1500
490 GA=INKEY$:IFGA="ä"THEN560
500 IFGA="n"THENCLOSE:GOTO360
510 IFGA="z"THENGA="":X1=X1-1:GET1,X1:GOTO440
520 IFGA="v"THENGA="":GOTO410
530 IFGA="m"THEN250
540 IFGA="l"THEN690ELSE490
550 '#--------- Ändern des Datenblcks ---------------#
560 PRINT@131,"Änderung";:ONERRORGOTO0:GA="":PRINT@459,"Neue Daten eingeben";:GOSUB1180
570 '#----- Speichern des geänderten D-Blocks --------#
```

```
580 IFDT(1)=""THENDU(1)=DX(1)ELSEDU(1)=DT(1)
590 IFDT(2)=""THENDU(2)=DX(2)ELSEDU(2)=DT(2)
600 IFDT(3)=""THENDU(3)=DX(3)ELSEDU(3)=DT(3)
610 IFDT(4)=""THENDU(4)=DX(4)ELSEDU(4)=DT(4)
620 IFDT(5)=""THENDU(5)=DX(5)ELSEDU(5)=DT(5)
630 IFDT(6)=""THENDU(6)=DX(6)ELSEDU(6)=DT(6)
640 IFDT(7)=""THENDU(7)=DX(7)ELSEDU(7)=DT(7)
650 LSETDX(1)=DU(1):LSETDX(2)=DU(2):LSETDX(3)=DU(3):LSETDX(4)=DU(4):LSETDX(5)=DU(5):LSETDX(6)=DU(6):LSETDX(7)=DU(7)
660 PUT1,X1:CLOSE
670 WA=0:PRINT§1841," Die Änderung wurde durchgeführt!   (W)eitere Adressdaten ändern      (M)enü ";CHR$(7);:GOTO730
680 'H----- Löschen eines Adressblocks ----------------H
690 GA="":PRINT§131,"Löschung";
700 FORQ=1TO7:LSETDX(Q)="":NEXTQ
710 PUT1,X1:CLOSE
720 PRINT§1841," Die Löschung wurde durchgeführt !   (W)eitere Adressblöcke löschen   (M)enü ";CHR$(7);
730 GA=INKEY$:IFGA="m"THEN250
740 IFGA="w"THENCLOSE:GO="y":X4=X1:GOTO360  'Datenrücksprung bei Änderungswiederholung
750 GOTO730
760 'H---- Ende der Adress-Datei (EOF) ----------------H
770 PRINT§1841,"  Kein weiterer Adress-Block vorhanden   (W)eitersuchen m.a.Suchkr.  (M)Menü ";CHR$(7);
780 GA=INKEY$:IFGA="m"THEN250
790 IFGA="w"THENRETURNELSE770
800 'H----- Adress-Liste drucken ----------------------H
810 ONERRORGOTO0:CLS:GOSUB1560:PRINT§131,"Drucken";
820 PRINT§1841," Welche Adress-Liste wollen Sie drucken ?   ( G )esamtbestand --- ( A )auswahl ";CHR$(7);
830 B1=INKEY$:IFB1="a"THEN840ELSEIFB1="g"THEN870ELSE830
840 GOSUB1590:GOSUB1620:PRINT§1842,"Nach welchem Suchbegriff wollen Sie drucken? ";KY:PRINT§1887,;:LINEINPUTGN:IFLEFT$(GN,1)="/"THEN250
850 IFGN=""THEN840ELSEIFLEN(GN))30THEN840ELSEIFASC(GN)(480RASC(GN))126THEN840ELSE870
860 'H---------- Drucker einstellen ------------------H
870 XY=0:IFINP(253)=63THEN890ELSEPRINT§1841," Ihr Drucker ist nicht startbereit.      Prüfen Sie den Drucker (SEL-Lampe) !";:PRINT§1841,
CY;CHR$(7);:GOTO870
880 'H---------- Formular einstellen ----------------H
890 PRINT§1841,"     Ist das Formular richtig eingestellt ?          (D)rucken     (M)enü      ";CHR$(7);
900 GA=INKEY$:IFGA="d"THEN920ELSEIFGA="m"THEN250ELSE900
910 'H--------- Druck der Daten-Liste ----------------H
920 WP=9:GOSUB1530:WP=0
930 IFEOF(1)THENCLOSE:GOTO1150
940 LQ=LQ+1:GET1:KZ=DX(1)+DX(2)+DX(3)+DX(4)+DX(5)+DX(6)+DX(7)
950 IFB1="g"THEN970
960 IFINSTR(KZ,GN))0THEN970ELSE930
970 KN=DX(1):KV=DX(2):KU=DX(3):KS=DX(4):KO=DX(5):KW=DX(6):KP=DX(7)
980 JB=CHR$(14):JC=CHR$(15):JG=CHR$(27)+"N":JH=CHR$(27)+"E":J1=CHR$(27)+"Q":JN=CHR$(27)+"!":JO=CHR$(27)+CHR$(34)
990 IFX8=9THENLPRINT:LPRINT:LPRINT:X8=0:X7=0:GOTO1030
1000 IFXY)0THEN1060
1010 LPRINTJH+JN+JB;TAB(18)"ADRESS-VERZEICHNIS":LPRINT:LPRINTTAB(16)"P R I V A T"+JC:LPRINT:LPRINT
1020 LPRINTTAB(35)"STAND: ";LEFT$(TIME$,8):LPRINT:LPRINT
1030 LPRINTJI+JO+"  "+STRING$(128,45):LPRINT"  ANREDE"
1040 LPRINT"  VORNAME / FIRMENBEZEICHNUNG      PLZ / WOHNORT"
1050 LPRINT"  NAME / FIRMENNAME                STRASSE / POSTFACH / HAUSNR.     TELEFONNUMMER                    ANSPRECHPARTNER":LPRINT"
  "+STRING$(128,45):LPRINT
1060 X7=X7+1:LPRINTCHR$(27)+"(003,"
1070 LPRINTCHR$(09)KN;
1080 LPRINTCHR$(27)+"(003,036,"
1090 LPRINTCHR$(09)KV;CHR$(09)KO;
1100 LPRINTCHR$(27)+"(003,036,069,101,"
1110 LPRINTCHR$(09)KU;CHR$(09)KS;CHR$(09)KW;CHR$(09)KP;:LPRINT
1120 XY=XY+1
1130 FORX=0TO2A:S(X)=0:NEXTX:IFXY=12THENLPRINT:LPRINT:LPRINT:X8=9
1140 IFX7=15THENLPRINT:LPRINT:LPRINT:X8=9:X7=0:GOTO930ELSE930
1150 PRINT§1841," Die Adress-Liste ist fertig !        (W)eiterdrucken m.a.Suchkr.  (M)Menü    ";CHR$(7);
```


```
1160 GA=INKEY$:IFGA="w"THEN810ELSEIFGA="m"THEN250ELSE1160
1170 '▮-------- Dateneingabe mit Maske ----------▮
1180 GOTO1450
1190 '▮---Anrede                K1/KN ▮
1200 PRINT§321,K1;KY;CY;:GOSUB1600:GOSUB1620
1210 PRINT§1850,"Herrn - Frau - Eheleute - Architekt - Firma - Dr. - Sonstige";CHR$(7);
1220 PRINT§347,;:LINEINPUTDT(1):IFLEFT$(DT(1),1)="/"THENCLOSE:GOTO250
1230 IFLEN(DT(1))>30THEN1200ELSEIFWA=1THEN1430
1240 '▮---Vorn./Firmenbez.   K2/KV ▮
1250 PRINT§481,;K2;KY;CY;:GOSUB1590:PRINT§1854,"Vornamen oder Firmenbezeichnung eingeben!";CHR$(7);
1260 PRINT§507,;:LINEINPUTDT(2):IFLEN(DT(2))>30THEN1250ELSEIFWA=2THEN1430
1270 '▮---Name/Firmenname    K3/KU ▮
1280 PRINT§641,K3;KY;CY;:GOSUB1600:PRINT§1858,"Familiennamen oder Firmennamen eingeben!";CHR$(7);
1290 PRINT§667,;:LINEINPUTDT(3):IFLEN(DT(3))>30THEN1280ELSEIFWA=3THEN1430
1300 '▮---Straße/Hausnr.     K4/KS ▮
1310 PRINT§801,K4;KY;CY;:GOSUB1600:PRINT§1861,"Straße und Hausnr. oder Postfach eingeben!";CHR$(7);
1320 PRINT§827,;:LINEINPUTDT(4):IFLEN(DT(4))>30THEN1310ELSEIFWA=4THEN1430
1330 '▮---PLZ/Wohnort        K5/KO ▮
1340 PRINT§961,K5;KY;CY;:GOSUB1600:PRINT§1862,"PLZ und Wohnort eingeben!";CHR$(7);
1350 PRINT§987,;:LINEINPUTDT(5):IFLEN(DT(5))>30THEN1340ELSEIFWA=5THEN1430
1360 '▮---Telefonnummer      K6/KW ▮
1370 PRINT§1121,K6;KC;CY;:GOSUB1600:PRINT§1874,"Telefonnummer eingeben !";CHR$(7);
1380 PRINT§1147,;:LINEINPUTDT(6):IFLEN(DT(6))>16THEN1370ELSEIFWA=6THEN1430
1390 '▮---Ansprechpartner    K7/KP ▮
1400 IFDT(1)="Herrn"ORDT(1)="Frau"ORDT(1)="Fräulein"ORDT(1)="Eheleute"THEN1430 ___
1410 PRINT§1281,K7;KY;CY;:GOSUB1600:PRINT§1854,"Ansprechpartner eingeben - z.B. )Herr Bong( )Frau Meier(";CHR$(7);
1420 PRINT§1307,;:LINEINPUTDT(7):IFLEN(DT(7))>30THEN1400ELSEIFWA=7THEN1430
1430 N=0:FORZ1=1TO7:N=N+1:K(N)=DT(N):NEXTZ1:PRINT§1600,CY;:GA="":GOSUB1600:PRINT§1857,"Sind die obigen Eingaben richtig  ( J/N ) ?";CHR
$(7);
1440 GA=INKEY$:IFGA="j"THENRETURNELSEIFGA="n"THEN1450ELSE1440
1450 WA=0:GOSUB1600:PRINT§1858,"Geben Sie die zu ändernde Zahl ein  ( 1-7 )";CHR$(7);
1460 WA=VAL(INKEY$):IFWA=00RWA)7THEN1460
1470 IFWU=2THENN=0:FORZ2=1TO7:N=N+1:K(N)=DX(N):NEXTZ2
1480 N=WA:PRINT§1602,"Alte Eingabe -----)))))  ";K(N);BN;BN;:DT(N)=""
1490 ONWAGOTO1200,1250,1280,1310,1340,1370,1400
1500 RESUME1510
1510 ONERRORGOTO0:GOSUB770:CLOSE:GOTO360
1520 '▮--------- Datei öffnen ---------------------------▮
1530 LQ=0:OPEN"R",1,"JOHADRES/DAT:6":X1=0:FIELD1,30ASDX(1),30ASDX(2),30ASDX(3),30ASDX(4),30ASDX(5),16ASDX(6),30ASDX(7)
1540 LQ=LOF(1):IFWP=9THENRETURNELSELQ=LQ+1:PRINT§141,USING"#####";LQ;:RETURN
1550 '▮--------- Daten auf 0 setzen --------------------▮
1560 FORQ=1TO7:DX(Q)="":NEXTQ:FORQ=1TO7:DU(Q)="":NEXTQ:FORQ=1TO7::DT(Q)="":NEXTQ
1570 KN="":KV="":KU="":KS="":KO="":KW="":KP="":RETURN
1580 '▮---- Rechte Textspalte auf 0 setzen ------------▮
1590 X6=298:FORX=1TO18:PRINT§X6,STRING$(20,32);:X6=X6+80:NEXTX
1600 PRINT§1840,CY;:RETURN
1610 '▮------------ Texteingabe ----------------------▮
1620 PRINT§540,"Nach der Eingabe";:PRINT§619,"( ENTER )  drücken";:PRINT§778,"Mit ( / ) kommen Sie";:PRINT§860,"zurück ins Menü!";:RETUR
N
1630 '▮------------ Sortierprogramm -----------------▮
1640 CLEAR5000:DEFSTRA-G:CV=STRING$(19,140):CW=STRING$(54,140)
1650 CLS:PRINT§131,"Sortieren";
1660 ONERRORGOTO1690
1670 KILL"JOHADRES/BAK:6":GOTO1710
1680 GOTO1710
1690 RESUME1700
1700 OPEN"R",1,"JOHADRES/BAK:6":CLOSE:GOTO1660
1710 CLOSE:ONERRORGOTO0:PRINT§322,CW;:PRINT§402,"Das gesamte  JOHADRES/DAT-File  wird nun bereinigt.Die":PRINT§482,"alten und neuen Date
n werden alphabetisch sortiert und":PRINT§562,"     erhalten eine neue fortlaufende Datennummer.":PRINT§642,CW;
1720 PRINT§379,CV;:PRINT§459,"Daten einlesen";:PRINT§539,"LOF-Datensatz:";:PRINT§619,"Daten-Nummer :";:PRINT§699,CV;
1730 CMD"N JOHADRES/DAT:6 JOHADRES/BAK"
1740 OPEN"R",1,"JOHADRES/BAK:6":OPEN"R",2,"JOHADRES/DAT:6"
1750 FIELD#1,90ASX$      '▮)) Länge des zu sortierenden Blocks ((▮
1760 L1=LOF(1):L2=L1:PRINT§553,L1;;PRINT§'▮)) Dateilänge alt/neu ((▮
1770 DIMN$(L1),NS(L1):FORN=1TOL1:GET1,N:PRINT§633,N;
1780 Z$=MID$(X$,61,30)      '▮)) Sortieren nach Familiennamen bzw.Firmenbez.
1790 N$(N)=Z$      '▮)) Sortierkriterium ((▮
1800 NS(N)=N      '▮)) Zugehörige Sektor-Nummer ((▮
1810 IFLEFT$(N$(N),1)=" "THENN$(N)=CHR$(255):L2=L2-1      '▮)) Gelöschter Eintrag ((▮
1820 GOSUB1890:NEXTN:CMD"O",L1,N$(1),NS(1)
1830 PRINT§813,"Das Sortierprogramm ist beendet.";:PRINT§891,"Das neue Datenfile wird geschrieben.";:PRINT§962,CW;
1840 PRINT§779,"Daten speichern";:PRINT§859,"LOF-Datensatz:";L2;:PRINT§939,"Daten-Nummer :";:PRINT§1019,CV;
1850 FIELD#2,255ASY$:FIELD#1,255ASX$:FORN=1TOL2
1860 GET1,NS(N):PRINT§953,N;:LSETY$=X$      '▮)) Nach sortierter Reihenfolge ((▮
1870 PUT2,N:NEXTN:CLOSE      '▮)) Neue Datei ist richtig sortiert ((▮
1880 CLEAR50:RUN
1890 IFINSTR(N$(N),"Ä")THEN1960
1900 IFINSTR(N$(N),"Ö")THEN1970
1910 IFINSTR(N$(N),"Ü")THEN1980
1920 IFINSTR(N$(N),"ä")THEN1990
1930 IFINSTR(N$(N),"ö")THEN2000
1940 IFINSTR(N$(N),"ü")THEN2010
1950 IFINSTR(N$(N),"ß")THEN2020ELSERETURN
1960 A="Ae":D="▮A":H=INSTR(N$(N),"Ä"):GOSUB2030:GOTO1920
1970 A="Oe":D="▮O":H=INSTR(N$(N),"Ö"):GOSUB2030:GOTO1920
1980 A="Ue":D="▮U":H=INSTR(N$(N),"Ü"):GOSUB2030:GOTO1920
1990 A="ae":D="▮a":H=INSTR(N$(N),"ä"):GOSUB2030:RETURN
2000 A="oe":D="▮o":H=INSTR(N$(N),"ö"):GOSUB2030:RETURN
2010 A="ue":D="▮u":H=INSTR(N$(N),"ü"):GOSUB2030:RETURN
2020 A="ss":D="▮ß":H=INSTR(N$(N),"ß"):GOSUB2030:RETURN
2030 B=LEFT$(N$(N),H-1):C=MID$(N$(N),H+1):N$(N)=B+A+C+D:RETURN
```

D/A-Wandler selbstgebaut

Musikwiedergabe mit dem PC muß nicht teuer sein. Wenige Bauteile schaffen eine kleine Alternative zu Soundkarten.

Der Ton macht die Musik

Nichts mehr hören und sehen. Das Gekreische und Gepiepse einfach abschalten und der wohltuenden Stille lauschen. Diesen Wunsch hegt bestimmt jeder, der schon mal versucht hat, Musik-Files über den PC-Lautsprecher wiederzugeben. Denn schöne Töne gehören nicht zu dessen Stärken. Unüberhörbar sind die Unzulänglichkeiten des winzigen Lautsprechers — Tonqualität läßt er schmerzlich vermissen, von ausreichender Lautstärke ganz zu schweigen.

Das sind Schwächen, mit denen die hier vorgestellte Schaltung Schluß macht. Sie wird an der parallelen Schnittstelle betrieben und wandelt die digitalen Daten in analoge Signale um — erheblich besser, als es der im Rechner vorhandene Baustein nebst Lautsprecher tun könnte.

Das Herzstück der Platine bildet ein DAC-Baustein (DAC = Digital/Analog-Converter). Er stellt die benötigten analogen Signale zur Verfügung, die dann über eine leistungsfähige HiFi-Anlage oder Aktivbox wiedergegeben werden können. Das schont nicht nur das Gehör, sondern nützt auch dem Geldbeutel.

Rund 25 Mark kostet der Eigenbau in Mono-Ausführung. Die Stereo-Version schlägt dagegen mit 50 Mark zu Buche. Dann allerdings kann sich die Schaltung mit so mancher Soundkarte messen. Herkömmliche Soundkarten, die ebenso wie die einfache Ausführung des Eigenbaus nur Mono-Wiedergabe erlauben, läßt sie jedoch weit hinter sich.

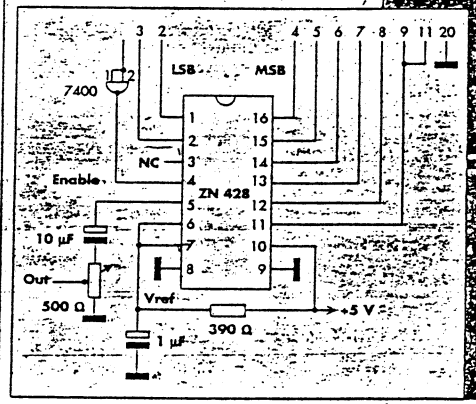
So bietet beispielsweise die alte Soundblasterkarte von der Firma Creative Labs ebenfalls nur 8-Bit-Wandlung. Mit der neuen Soundblasterkarte, die Stereoklang und 8-Bit-Wandlung pro Kanal beherrscht, kann der Eigenbau ebenfalls konkurrieren. Die Klangqualität einer Next-Workstation jedoch, die mit einem 16-Bit-Wandler für jeden Kanal CD-Sound bietet, erreicht sie nicht.

Mitentscheidend für Tonqualität und Dynamik ist die Abtastfrequenz. Und in diesem Punkt besitzt die analoge Technik einige Schwächen: So bereiten ihr Höhen und leise Passagen Probleme; denn je höher der Ton, desto höher muß die Abtastfrequenz sein. Hier stoßen herkömmlichen Schaltungen schnell an ihre technischen Grenzen. Die Umwandlung der digitalen Informationen wird ungenau, die Störungen nehmen zu.

Leistungsfähiger D/A-Wandler

Der CHIP-Tip stößt zwar ab 10 kHz Tonfrequenz an seine Grenzen, darunter gibt er aber das Audiosignal relativ sauber und exakt wieder. Für diese Maximalfrequenz müssen aber bereits 20 000 Signale pro Sekunde an dem D/A-Wandler bereitgestellt werden. Soll es höher hinausgehen, benötigt man professionelle Soundkarten. Mit einem Geräuschspannungsabstand von ungefähr 48 Dezibel liegt er jedoch gar nicht schlecht im Rennen. Und leistet ebensoviel wie preiswerte Kofferradios.

Außer an den D/A-Wandler stellt die Wiedergabe digitaler Musikstücke auch hohe Anforderungen an die Rechenkapazität des PC. Der Grund liegt auf der Hand. Um den Schwingungsverlauf auch der hohen Töne möglichst exakt wiedergeben zu können, müssen pro Sekunde 20 000 Werte an den D/A-Wandler ausgegeben werden. Das bedeutet einen enormen Rechenaufwand: 20 000mal je Sekunde muß eine Programmschleife abgearbeitet werden, die aus den Musikdaten die Ausgabewerte berechnet. Das überfordert PC der Klasse 286 in den meisten Fällen.



Der Aufbau der Schaltung ist recht einfach, da sie aus sehr wenigen Bauteilen besteht:

- eine Platine;
- ein DAC (ZN 428);
- ein TTL-Baustein 7400;
- ein Elko mit1Mikro-Farad/16Volt;
- ein Elko mit10Mikro-Farad/16Volt;
- ein Widerstand mit 390 Ohm;
- ein Drehpotentiometer mit 500 Ohm;
- eine Cinchbuchse;
- ein Kabel mit 25poligem Sub-D-Stecker sowie ein Netzteil oder ein Akku.

Im wesentlichen besteht die Schaltung aus zwei integrierten Schaltkreisen, aus dem DAC ZN428, der die digitalen Musiksignale in analoge umwandelt, und dem TTL-Baustein 7400, der ein Steuersignal anpassen muß. Signal 1 des Druckerports wird an Pin 1 und 2 angeschlossen und invertiert an Pin 3 ausgegeben. Wichtig ist: Pin 7 muß mit Masse und Pin 14 mit +5 Volt Versorgungsspannung verbunden sein.

Die 8-Bit-Signale werden über die Anschlüsse 1, 2 und 11 bis 16 über die parallele Schnittstelle an den DAC-Baustein gesendet. Liegen sie richtig an, erfolgt die Freigabe der Daten über die Stromleitung. Das Ausgangssignal kann an Pin 5 abgegriffen werden. Hier verhindert zusätzlich der zwischengeschaltete Elektrolytkondensator (Elko) Störungen zwischen DAC-Baustein und angeschlossenem Verstärker (siehe Schaltplan).

Zusätzlich läßt sich über das Drehpotentiometer der Ausgangspegel variieren. Sollte der angeschlossene Verstärker die Musikstücke nämlich nur verzerrt wiedergeben, so liegt dies häufig daran, daß der Ausgangspegel zu hoch liegt — Abhilfe schafft das Verdrehen des Widerstands. Der Pegel wird an den jeweiligen Verstärkereingang angepaßt, so daß die Störungen entfallen.

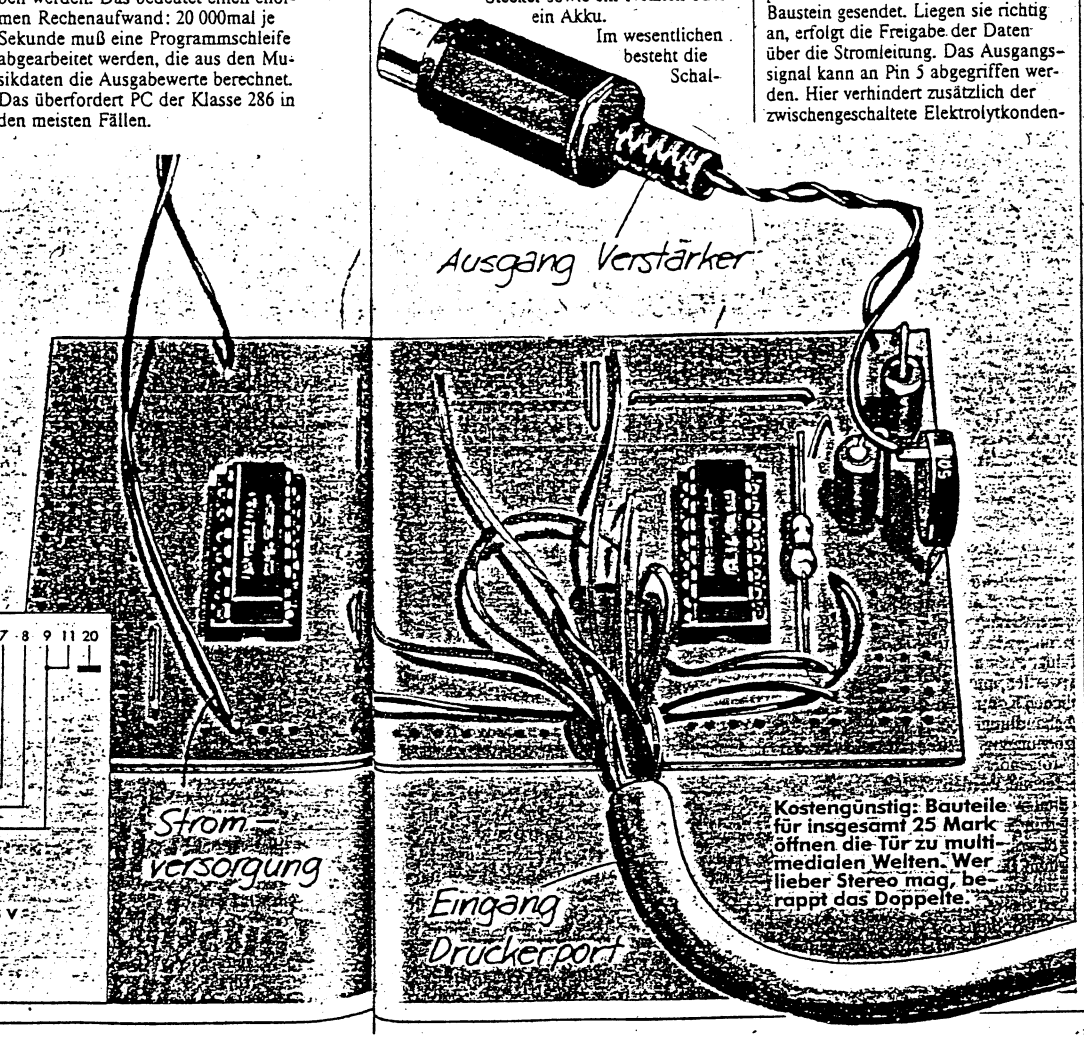
Die Stromversorgung der Schaltung übernimmt entweder ein herkömmliches externes Netzteil (5 Volt, 500 mA) oder ein Akku. Auch die Verbindung zum Druckerport des PC bereitet keine Probleme: Die Anschlußbelegung für den 25poligen Druckerport-Stecker ist im Schaltplan angegeben.

Ausgangspegel muß angepaßt werden

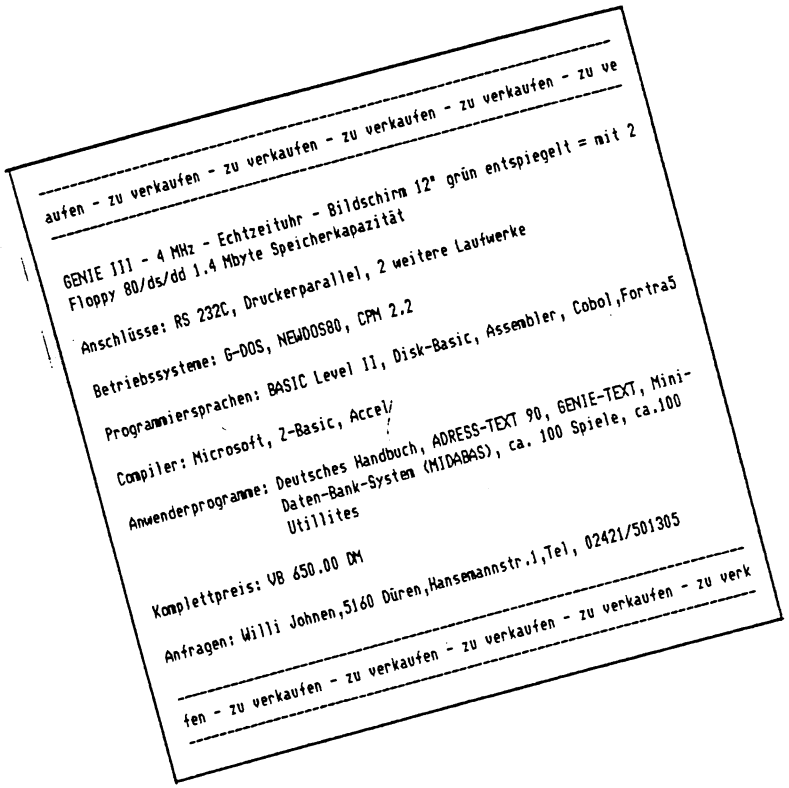
Etwas mehr Aufwand erfordert die Wiedergabe in Stereo. Für jeden Kanal werden je eine Schaltung und Schnittstelle benötigt. Dann allerdings steht dem ungetrübten Klangvergnügen dank zahlreich angebotener Publicmain- und Sharewareprogramme nichts mehr im Wege.

So können etwa die MOD-Player Trackblaster 2.0 von Volker Zinke oder WOW von Jan Ole Suhr direkt den Hardwaretip ansteuern. Sie spielen die breite Palette der vom Amiga bekannten Soundmodule (Mod-Files) auf dem PC ab — egal, ob in Mono oder in Stereo.

Andreas Filip



Know-how im Überblick



aufen - zu verkaufen - zu verkaufen - zu verkaufen - zu verkaufen - zu ve

GENIE III - 4 MHz - Echtzeituhr - Bildschirm 12" grün entspiegelt = mit 2
Floppy 80/ds/dd 1.4 Mbyte Speicherkapazität

Anschlüsse: RS 232C, Druckerparallel, 2 weitere Laufwerke

Betriebssysteme: G-DOS, NEWDOS80, CPM 2.2

Programmiersprachen: BASIC Level II, Disk-Basic, Assembler, Cobol,Fortra5

Compiler: Microsoft, Z-Basic, Accel

Anwenderprogramme: Deutsches Handbuch, ADRESS-TEXT 90, GENIE-TEXT, Mini-
Daten-Bank-System (MIDABAS), ca. 100 Spiele, ca.100
Utillites

Komplettpreis: VB 650.00 DM

Anfragen: Willi Johnen,5160 Düren,Hansemannstr.1,Tel, 02421/501305

fen - zu verkaufen - zu verkaufen - zu verkaufen - zu verkaufen - zu verk

Zwecks Platzgewinnung habe ich folgende Teile billig abzugeben:

1 * ATARI Megafile 30 für DM350.--

1 * PC-Steckerweiterung PC-Spedd V1.4 für DM100.--
(Nur für ATARI-Besitzer geeignet !)

1 * ATARI GFA-Assembler V1.5 für DM50.--

1 * ATARI GFA-BASIC 2.02 mit Compiler für DM25.--

1 * ATARI GFA-Draft v2.0 (CAD) für DM20.--

1 * Video-Genie EG3003 m. 256KByte RAM über Banker, 3,5 MHz Takt, HRG1B incl. HRG-Basicerw., Monitor u. 2*360K Disklaufwerke, NewDOS80 V2.0 und noch div. andere Software.

Da ich den Kram beim letzten Versuch nicht losgeworden bin, hier nochmal das super Sonderangebot: alles was ich hier genannt habe plus das was sonst noch zum System rumliegt, für DM 100.-- Wegen der vielen Einbauten ist der Rechner jedoch nicht mehr ohne entsprechende Maßnahmen postversandtfähig. Auf Wunsch versuche ich jedoch die Innereien des Rechners (und nur diesen betrifft es) durch Füllmaterial so zu fixieren, daß er nach Empfang noch komplett ist. Der Versandt erfolgt UNFREI !, da das ganze Material mit sicherheit mehrere Packete umfaßt und ich bei dem Angebotspreis nicht auch noch die Post finanzieren will.

Meine Anschrift:

KURT MÜLLER
Sophie-Scholl-Ring 3 b
2054 Geesthacht

Club 80 Börse --- Club 80 Börse

Zehn Jahre ZCPR

von Jay Sage

(Anm des Übersetzers: Dieser Artikel wurde der Ausgabe 54 des "Computer Journal" entnommen und mit freundlicher Genehmigung des Verlages veröffentlicht. Jay Sage ist - was ZCPR betrifft - ein Mann der ersten Stunde. Ich habe mich bemüht, den Artikel so wörtlich wie möglich zu übersetzen. Teilweise ist dabei der Stil etwas auf der Strecke geblieben. Einige Fachausdrücke, bei denen ich davon ausgehen mußte, daß eine Übersetzung sie nur unverständlich machen würde, habe ich übernommen. Hartmut Schulte)

Am 02. Februar 1992 sind genau zehn Jahre seit der Herausgabe der ersten Version des ZCPR vergangen. Mehr oder weniger habe ich mich während der gesamten Zeit damit beschäftigt und ich glaube, es ist erstaunlich, wie kraftvoll die Aktivitäten auf diesem Gebiet noch sind.

Der Herausgeber (des Computer Journal, d.Ü.), Chris McEwen, hatte ursprünglich gehofft, daß wir aus diesem Heft eine spezielle ZCPR-Jubiläumsausgabe mit Beiträgen von einigen der ursprünglichen Entwickler, in erster Linie Richard Conn, machen konnten. Ich wechselte mehrere Briefe mit Richard darüber, er griff die Idee aber leider nicht auf. Wenn ich davon ausgehe, daß ich von all denen, die in der Z-Gemeinde aktiv sind, mit am längsten dabei bin, scheint es mir passend zu sein, daß ich nunmehr den Artikel schreibe.

Die Geschichte des ZCPR

Die Masse des Materials für diese Abhandlung habe ich dem Einführungskapitel meines Buchs "ZCPR33 User Guide" entnommen. Als Echelon meine Version des ZCPR3 als offizielle Version herausgab, wurde selbstverständlich ein begleitendes Handbuch gefordert. Neben all den notwendigen technischen Informationen, z.B., was der neue Command Processor macht und wie er installiert werden sollte, fügte ich zwei für mich sehr wichtige Kapitel hinzu:
eine Feststellung, was ich versuchte, mit ZCPR33 zu erreichen und die Geschichte, die zu seiner Entwicklung führte.
Im Computer Journal schreibe ich oft über die Ziele des Z-Systems, heute geht es darum, einiges aus seiner Vergangenheit zu beleuchten.

ZCPR1

"Was, Du hast noch nichts von ZCPR gehört?" Ich erinnere mich sehr gut daran, daß ich mit diesem Ausruf von einem der Clubveteranen begrüßt wurde, als ich als frischgebackener Computerbenutzer an dem Treffen eines CP/M-Computerclubs teilnahm.
Er konnte es nicht fassen, daß es noch jemanden gab, der das Standard-CP/M benutzte. Bald dachte ich genauso und tue es auch heute noch!
Das ZCPR, das er meinte, würden wir heutzutage ZCPR1 nennen. ZCPR, was soviel bedeutet wie "Z80 Command Processor Replacement", war das Ergebnis der Arbeit einer Gruppe von Computerliebhabern, die sich "The CCP Group" nannte. Sie bestand aus Frank Wancho, Keith Peterson, Ron Fowler, Charlie Strom, Bob Mathias und Richard Conn. Wie wir noch sehen werden, war Richard die treibende Kraft.

Ron Fowler ist wohl bekannt als der Autor des MEX Kommunikationsprogramms, daß ich immer noch benutze und liebe. Keith Peterson schrieb eine vereinfachte Version von Ward Christensens CBBS, dem ursprünglichen computerisierten Schwarzen Brett. Keith Programm nannte sich MINICBBS, bis heute läuft es auf meiner Z-Node in einer von mir angepaßten Version. Um ehrlich zu sein, ist es überholt, aber es ist für mich eine Art Verbindung zur Vergangenheit, die ich sehr schätze.

Keith Peterson war lange Zeit Sysop eines der besten BBS Systeme im Lande, dem Royal Oak. Obwohl es zur MS-DOS Software expandierte, ließ es CP/M nie außer acht. Vielleicht war es einzigartig, daß sich Anrufer bei Royal Oak sofort am Prompt des Operationssystems wiederfanden. Man brauchte weder Namen noch Passwort. Wollte man MINICBBS benutzen, mußte man es selbst aufrufen.

Frank Wancho ist an der Verwaltung des SIMTEL20-Computers auf dem White Sands Raketenschießstand beteiligt. Dieser Rechner beherbergt ein riesiges Archiv von CP/M-Programmen (und anderen natürlich). Keith arbeitet bei SIMTEL20 an der Bearbeitung der Sammlungen. In den Informationen, die mir Frank via Computer gab, tauchten ihre Namen immer wieder auf.

Um 1981 begeisterte Richard Conn die Gruppe, indem er vorschlug, den CP/M Console Command Processor oder CCP neu zu schreiben, um die Vorteile der leistungsfähigeren und eleganteren Opcodes des neuen Zilog Z80 Mikroprozessors zu nutzen. Da die Mitglieder der CCP-Gruppe weit auseinander wohnten, hielten sie, ebenso wie wir heutzutage, über Computer Kontakt zueinander. Für diesen Zweck stellte Frank Wancho den Zugriff zum Rechner sicher.

Nachdem im CCP etwas Platz geschaffen worden war, konnten die Programmierer einige brauchbare neue Eigenschaften einfügen. Die allerwichtigste war die Schaffung eines Suchpfades für ausführbare Dateien. Ab CP/M 2.0 hatte Digital Research Usernummern eingeführt, jedoch waren sie faktisch wertlos, weil es keine Möglichkeit gab, von einem Userbereich zum anderen Programme laufen zu lassen oder sie anzusprechen. Mit ZCPR und seiner Fähigkeit, automatisch Laufwerk A/ User 0 zu suchen, wurde dieses Problem aus der Welt geschaffen, so daß sich die Möglichkeit ergab, die neugeschaffenen Userbereiche effektiv zu nutzen.

Eine weitere Neuheit des ZCPR war das Kommando GO, daß den sofortigen Neustart des letzten ausgeführten transienten Programms erlaubte, ohne daß dies neu von Diskette geladen werden mußte. Das war in den Zeiten der langsamen Diskettenlaufwerke ein echter Segen. Die residenten Befehle wurden in vielen nützlichen Kleinigkeiten verbessert. So brechen Befehle wie REN oder SAVE unter CP/M einfach ab, wenn sie auf eine schon existierende Zieldatei treffen. Der Nutzer muß dann langwierig diese Datei löschen und wieder von vorne beginnen. Unter ZCPR wird der Nutzer einfach gefragt, ob die alten Dateien überschrieben werden sollen.

Das original ZCPR wurde auf einer Diskette der SIG/M (Special Interest Group/Microcomputers) veröffentlicht, dem Zweig der Amateur Computer Group of New Jersey (ACGN), der für die Verteilung von Public Domain Software verantwortlich war. Es war Ausgabe Nr. 54 vom 02. Februar 1982. Interessanterweise wurde der Artikel, den Sie gerade lesen, zehn Jahre später in der Ausgabe 54 des Computer Journal herausgegeben!

Andere Programmierer verfeinerten ZCPR weiter zum NZCPR (New ZCPR), so daß Ende Oktober 1982 mit der SIG/M Nr.77 die Version 1.6 des NZCPR herausgegeben werden konnte. Schließlich wurde Version 2.1 erreicht, die allerdings nie auf Diskette, sondern nur über Netzwerke verteilt wurde.

Jim Byram von der Boston Computer Society CP/M Group verfaßte eine persönlich verteilte Version des NZCPR, die nur Intel 8080 Codes benutzte und an der man erkennen konnte, daß nicht einfach nur die Anwendung der neuen Z80 Opcodes, sondern vielmehr wirtschaftliches Programmieren der wichtigste Faktor bei der Verbesserung des Command Processors war. Übrigens, ich glaube, es war Jim, der die eingangs erwähnte Frage bezüglich ZCPR an mich stellte. Ich wurde schließlich Leiter dieser Gruppe, die sich mit anderen zusammenschloß und letztendlich die Zi/Tel Group wurde, bei der ich heute der Verantwortliche für CP/M und der Sysop des Schwarzen Bretts bin. Die Gruppe unterstützt CP/M, das Z-System und MS-DOS.

ZCPR2

Obwohl ZCPR1 eine deutliche Verbesserung gegenüber CP/M bedeutete, so war es doch keine revolutionäre Entwicklung. Richard Conn träumte aber von einem wirklich fortschrittlichen Betriebssystem und setzte darum die Entwicklung fort. Fast genau ein Jahr nach dem Erscheinen des ZCPR1 wurde ZCPR2 auf zehn SIG/M Disketten (98-107) herausgegeben, ein beispielloser und großartiger Beitrag zur Public Domain Software.

Der konzeptionelle Fortschritt des ZCPR2 war beachtlich: es nutzte Memoryspeicher im geschützten Memory oberhalb des BIOS zur Speicherung neuentwickelter Module des Betriebssystems. Die Befehlszeile, die vorher Bestandteil des Command Processors war, wurde in einem dieser Puffer gespeichert, so daß sie nicht durch Warmstarts, bei denen eine neue Kopie des Command Processors von Diskette geladen wird, zerstört werden konnte. Auf diese Weise konnten mehrere Befehle auf einer Zeile aneinandergefügt werden.

Der Befehlssuchpfad befand sich ebenfalls in einem dieser Puffer, anstatt ihn fest im Command Processor einzuprogrammieren. Auf diese Weise konnte der Suchpfad jederzeit vom Nutzer geändert werden. Gleichzeitig wurde das Konzept der mit Namen versehenen Directories eingeführt, wobei ein weiterer Memorypuffer zur Aufnahme des Namensindex dient.

Viele uns heute wohlvertraute Utilities erschienen zum ersten Mal unter ZCPR2, so z.B. ZEX, WHEEL, HELP, PATH, PWD, MKDIR und MENU. Das letztgenannte Programm nutzte ein neues Shellkonzept, indem es jedesmal, wenn es einen Befehl in den Befehlszeilenpuffer schrieb, seinen Namen an das Ende der Befehlsfolge anfügte, so daß nach Abarbeitung der Befehle die Kontrolle wieder an MENU überging. Dieses Prinzip funktionierte hervorragend, solange es nicht mehr als eine Shellebene gab. Unter ZCPR2 wurde auch die erweiterte Befehlsverarbeitung eingeführt.

Alleine die ZCPR2-Dokumentation war größer als ein halbes Megabyte! Sie bestand aus einem konzeptionellen Handbuch, einem Installationshandbuch, einem Benutzerführer und einem Handbuch, das die Grundprinzipien erklärte und wohl entstand, weil Rick sich und uns beweisen mußte, daß es sich bei dem Ganzen nicht um eine Marotte handelte.

Kurz nach der Herausgabe durch die SIG/M erschien Version 2.3 auf Diskette 108. Bisher war ZCPR2 den Traditionen des ZCPR1 gefolgt und nutzte Zilog Opcodes. Da die Eigenschaften des ZCPR2 jedoch so überzeugten, wollten immer mehr Besitzer von Rechnern, die mit Intel 8080 und 8085 Mikroprozessoren arbeiteten, am neuen System teilhaben. Charlie Strom, ein Mitglied der ursprünglichen CCP Group und später wohlbekannt als Sysop der Compuserve CP/M Special Interest Group, schrieb den Kode des Command Processors und einige wichtige Utilities um in einen Intel-kompatiblen Kode und veröffentlichte das Ergebnis auf SIG/M 122. Ob Sie es glauben oder nicht, zu jener Zeit benutzte ich bei meiner Arbeit ein Intel MDS-800 miroprozessorgestütztes Entwicklungssystem, eben das, für das Gary Kildall während seiner Zeit bei Intel CP/M erfand und ich erinnere mich gut an das Erscheinen der 8080-Version des ZCPR2. Es war großartig!

ZCPR3

Doch mit ZCPR2 endete keineswegs die Entwicklung. Am französischen Nationalfeiertag, dem 14. Juli 1984, noch nicht ganz eineinhalb Jahre nach Herausgabe des ZCPR2, bot Richard Conn eine Version 3 in Form von weiteren neun SIG/M-Disketten (184-192) an. Bis zu diesem Zeitpunkt stammten mehr als 10% aller Software, die die SIG/M je herausgebracht hatte, aus der Feder eines Mannes - Richard Conn!

Während eines Gesprächs mit Richard muß ich wohl meine Bewunderung über diese unglaubliche Menge an Software, die er geschrieben und veröffentlicht hat, geäußert haben. Richards Antwort beeindruckte mich ebenso. Er sagte, daß die Programme, die andere veröffentlicht hatten, ihn soviel gelehrt und ihm so geholfen hatten, daß er sich selbst weit über das normale Maß hinaus verpflichtet fühle, das an die Gemeinschaft zu verteilen, was er geben können. Das hat er mit Sicherheit getan! Und von genau diesem Geist ist die 8-Bit-Gemeinde durchdrungen.

ZCPR3 brachte sowohl bedeutsame neue Konzepte als auch größere Verfeinerungen. Drei wesentliche Neuerungen waren flow control, error handling und der message buffer.

Flow control erlaubte einen erheblich höheren Grad der automatischen Verarbeitung, weil der Command Processor nicht mehr vom Bediener und seinen Eingaben abhängig war, sondern weitreichende Entscheidungen selbst treffen konnte. Der message buffer schuf die Möglichkeit zur Kommunikation zwischen dem Command Processor und den Programmen bzw zwischen nacheinander ablaufenden Programmen.

Mit error handlern war es möglich, falsch eingegebene Befehle zu korrigieren, eine wichtige Hilfe im Zusammenhang mit Mehrfachbefehlen in einer Zeile. Es war schon schlimm genug, nach einem Fehler einen Befehl neu zu tippen, aber wegen eines einzigen Fehlers eine lange Befehlsfolge neu eingeben zu müssen, konnte einem die Freude an der Nutzung der Mehrfachbefehle schnell nehmen.

Übrigens war ZCPR3 anders als seine Vorgänger so geschrieben, daß es sowohl mit Intel als auch mit Zilog Opcodes assembliert werden konnte. im ersten Fall war der Code verständlicherweise länger und es konnten weniger Möglichkeiten eingebaut werden, aber er konnte auf einem 8080- oder 8085-Rechner laufen.

ZCPR31

Im März 1985 begann jene Kette von Verbesserungen des ZCPR3, die zur Version 3.3 führte, mit meiner privaten Experimentalversion ZCPR31, die ich auf meiner Z-Node nutzte. Sie war so abgewandelt, daß der Command Processor die Werte für das letzte Laufwerk und den letzten Userbereich vom Environment Descriptor (dazu später mehr) erhielt.

Dies war meine erste Begegnung mit dem Kode des Operationssystems, vor dem ich bisher, wie bestimmt viele andere auch, großen Respekt hatte. Das Wort "Operationssystem" hat etwas Mystisches, das einen vermuten läßt, daß nur die allererfahrensten Programmierer eventuell den Kode verstehen könnten. Tatsächlich stellte ich dann fest, daß er sich gar nicht so sehr von dem ordinärer Utilityprogramme unterschied. Zu meinem Erstaunen war ich in der Lage, lauffähige Änderungen durchzuführen, die den CCP verbesserten. Die offensichtlichsten Fortschritte machte ich im August 1985, als mir gleich drei Verbesserungen gelangen.

Als erstes unterbrach ich die Endlosschleife, in die ZCPR30 geriet, wenn der vorher aufgerufene Error Handler nicht gefunden wurde, vielleicht, weil der Suchpfad oder der Programmname geändert worden war. In dieser Situation würde ein Befehlsfehler den Error Handler aufrufen. Wenn dieser nicht gefunden würde, würde dies einen weiteren Fehler bewirken, der wiederum den Error Handler aufrufen würde und so weiter, bis jemand die Resettaste drückt oder den Strom ausschaltet.
Zweitens änderte ich den Kode so, daß er aus dem Environment die Adressen der RCP-, FCP- und NDR-Module entnehmen und so dynamisch auf Änderungen reagieren konnte.
Schließlich schrieb ich den Kode so um, daß der erweiterte Command Processor die Kontrolle an den Command Processor zurückgeben konnte, falls er nicht in der Lage war, das Kommando auszuführen. In diesem Fall rief der Command Processor dann den Error Handler auf. Damit war der erweiterte Command Processor als Erweiterung des CCP tatsächlich flügge geworden und ein ZCPR3-System konnte Vorteile sowohl aus der erweiterten Befehlsverarbeitung als auch aus dem Error Handling ziehen, das nunmehr auch von normalen Programmen eingeleitet werden konnte.

Im Januar 1986 wurden dann erste Schritte unternommen, um einige schwerwiegende Fehler bei der Berechnung des kürzesten Suchpfads zu beseitigen. Leider lösten wir damit andere Fehler aus, so daß es Howard Goldstein erst im Juni 1986 gelang, eine vollständige und saubere Lösung anzubieten.

Zum nächsten größeren Verbesserungsschub kam es im März 1986, als Al Hawley, Sysop der Z-Node #2 und heute ein bekannter Autor des "Computer Journal", verschiedene neue Ideen einbrachte. Eine davon war ein neuer Ansatz, wheelgeschützte Befehle (Befehle, die nur ein berechtigter Nutzer geben darf) einzufügen. Unter ZCPR30 wurde dieser Schutz direkt in den Command Processor (und den RCP) einprogrammiert, so daß eine Fehlermeldung erschien, wenn ein eingeschränkter Befehl bei ausgeschaltetem Wheel gegeben wurde. Al schlug vor, das höchstwertige Bit des ersten Buchstabens des Befehls zu setzen, um anzuzeigen, daß dieser Befehl für die Allgemeinheit unzulässig war.

Dieses Konzept hatte mehrere wichtige Vorteile: Erstens verkürzte sich der Kode. Zweitens wandte der neue Kode automatisch die selbe Technik bei Befehlen des residenten und flüchtigen Befehlsspeichers (RCP und FCP) an, so daß der Kode, der die Zugriffsberechtigung überprüfte, aus diesen Modulen verschwinden konnte.

Drittens verschwanden geschütze Programme einfach, soweit es den Command Processor betrifft, statt eine Fehlermeldung auszulösen. Damit konnten transiente Programme oder Alias mit dem selben Namen wie der geschützte residente Befehl automatisch eingreifen und das tun, was der Systemverwalter in einem solchen Fall von ihnen wünschte.

Al Hawley führte auch zwei Konzepte ein, die die Behandlung geschützter Systeme einfacher machten. Er ermöglichte es dem Command Processor, flexibel festzulegen, ob er die DU: Form des Inhaltsverzeichnisses entsprechend der gesetzten DUCK Flag im Environment erkennen sollte oder nicht und er umging die Überprüfung des Paßworts, wenn das Wheelbyte gesetzt war. Diese beiden Neuschöpfungen erleichterten Sysops und Systemergänzern den Umgang mit gesicherten Systemen, ohne es dem eingeschränkten Nutzer leichter zu machen.

Der letzte größere Fortschritt bei der Entwicklung des ZCPR31 entsprang einer Unterhaltung, die ich im Juli 1986 mit Bruce Morgen hatte. Wir diskutierten über die lästige Art, wie ZEX in Shells funktionierte, weil das Shellprogramm für jede Befehlszeile neu geladen wurde, nur um festzustellen, daß ZEX gerade lief und dann die nächste Befehlszeile aus dem ZEX in den Mehrfachbefehlszeilenpuffer (multiple command line buffer) zu laden. Ich wechselte eine Kleinigkeit im Kode und das Problem verschwand wie der Blitz.

ZCPR33

In den letzten Tagen des Januar 1987 erhielt ich einen Anruf von Echelon. Richard Conn wollte nicht mehr am ZCPR3 weiterarbeiten und Echelon fragte mich, ob ich auf der Basis meiner Experimentierversion 31 an einer offiziellen Version 3.3 weiterarbeiten wollte. Ich stimmte zu. Von Februar bis April 1987 war ein enormer Aufwand an zusätzlichen Entwicklungen zu bewältigen, deren Ergebnisse ich im Detail im ZCPR33 User Guide beschrieben habe. Hier will ich nur einige Kernpunkte erwähnen.

Die Entscheidung, nicht länger 8080/8085-Rechner zu unterstützen, fiel. Der Kode wurde in Zilog mnemonics geschrieben und es wurde intensiver Gebrauch von Z80-spezifischen Befehlen einschließlich relativer Sprünge, Blockbewegungen, direkter Wortübertragungen an Registerpaare außer HL, 16-Bit Subtraktionen und das alternative Registerset gemacht. Diese Entwicklung wurde bis zum heutigen ZCPR34 fortgeführt. Soweit ich weiß, hat niemand auch nur versucht, eine 8080-Version zu schreiben - es gibt ja auch nicht mehr allzu viele Rechner dafür.

Eine der schönsten Eigenschaften des ZCPR33 war die automatische Installation von Programmen. Bis zu diesem Zeitpunkt mußte auch ein Programm, das später unter ZCPR laufen sollte, auf das spezielle System eingestellt werden. Falls das vergessen wurde, konnte sich ein Programm unter Umständen ganz komisch benehmen, was sowohl junge als auch erfahrene Nutzer in Schwierigkeiten bringen konnte.

Unter ZCPR2 war die Installation ein aufwendiges Verfahren, bei dem ein großer Kodeblock mit GENINS gepatcht werden mußte. Unter ZCPR3 wurden Informationen über die Systemzusammensetzung in einem Puffer gespeichert (dem sogenannten Environment oder ENV), zu dem alle Programme Zugriff hatten. Oder was noch wichtiger war: die Konfiguration des Systems konnte geändert werden, ohne gleichzeitig alle Programme ändern zu müssen. Das Patchen der Programme konnte auf das Einfügen der ENV-Adresse reduziert werden.

Sobald ich hörte, daß Richard Conn einen Weg gefunden hatte, um diesen lästigen Installationsschritt aus der Welt zu schaffen, war mir die Lösung auch klar. Wenn der Command Processor schon das Programm von der Diskette lädt und er gleichzeitig auch die ENV-Adresse kennt, warum kann er dann nicht gleich automatisch die Adresse in das geladene Programm schreiben? Das war doch die Lösung...

Eine wirklich revolutionär neue Idee wurde mit ZCPR33 eingeführt. Bisher wurden unter CP/M alle transienten Programme ab der Standardadresse 100H geladen und von dort aus gestartet. Unter CP/M gab es keinen Grund, dies zu ändern, wohl aber unter ZCPR3. Schon seit ZCPR1 hatte ich mich sehr daran gewöhnt, Programme mit GO wiederzustarten. Umso mehr verwunderte es mich, daß GO unter ZCPR3 manchmal merkwürdige Ergebnisse lieferte.

Unter CP/M werden Programme nur geladen, wenn das System angewiesen wird, diese laufen zu lassen. Unter ZCPR3 jedoch wurden eine ganze Menge von Programmen automatisch vom Command Processor geladen und ausgeführt. Dazu gehörten erweiterte Command Processoren, Error Handlers, Shells und transiente COM-Versionen ansonsten residenter Befehle, wie z.B. ERA oder REN.
So kam es ab und zu, daß diese Programme statt der vom Nutzer erwarteten wiedergestartet wurden, wenn man GO eingab.

Als ich eines Tages am ZCPR33-Kode arbeitete, bemerkte ich, daß es nur einer Kleinigkeit bedurfte, damit der Command Processor eine Datei in eine andere Adresse als 100H lud. Das, so war mir klar, konnte das Problem mit dem GO lösen. Anwenderprogramme konnten, wie üblich, bei 100H geladen werden, aber automatisch durch den Command Processor aufgerufene Programme konnten in eine höhere Adresse wie z.B. 8000H deponiert werden. Auf diese Weise wurden Anwenderprogramme nicht überschrieben und GO könnte sie jederzeit wieder laufen lassen.

Es gab noch eine weitere Gruppe von Neuerungen. ZCPR30 bot eine Anzahl an Sicherheitsmaßnahmen, die es besonders geeignet für Datenfernübertragungssysteme (BBS) machten. Das sogenannte Wheelbyte steuerte den Zugriff sowohl zu residenten als auch zu transienten Programmen. Beliebig veränderbare Grenzen im Bereich der Laufwerke und Usernummern und Inhaltsverzeichnisse mit Namen und Paßwort konnten Anrufer aus bestimmten Bereichen fernhalten.

Diese Vorsichtsmaßnahmen ermöglichten es Nutzern, das System direkt vom Befehlszeilenprompt aus fernzusteuern, dies im scharfen Gegensatz zu MS-DOS Systemen, bei denen ein Nutzer, der erstmal beim Prompt angelangt ist, frei über Nutzung oder auch Zerstörung des Systems entscheiden kann.

War das Sicherheitssystem voll aktiviert, konnte es unter ZCPR30 allerdings auch ein unnützes Ärgernis sein. So konnte man sich Situationen vorstellen, in denen ein Nutzer zwar mit einem Namen in ein Directory kam, weil es kein Passwort hatte, jedoch die Möglichkeit, nur mit Laufwerk und User hineinzukommen, versperrt blieb, weil es hier außerhalb der erlaubten Reichweite lag. Unter ZCPR33 ist ein Directory auch mit Laufwerk/User erreichbar, wenn es unter einem Namen angesprochen werden kann.

ZCPR34

Der augenblickliche Bearbeitungsstand des ZCPR Command Processors ist Version 3.4. Die erste Herausgabe erfolgte irgendwann im März 1988 zusammen mit NZCOM und Z3PLUS; sie wurde im Heft 32 des Computer Journal beschrieben. Im Vergleich zu ZCPR33 war es ein Fortschritt, eine Verfeinerung. Es gab keine radikal neuen Ideen wie beim ZCPR33. Trotzdem waren die Änderungen bedeutsam und nützlich. Seit der ersten Veröffentlichung gab es mehrere kleinere Überarbeitungen und Erweiterungen.

Eine dieser Erweiterungen unter ZCPR34 war die Schaffung eines erweiterten environment descriptors ("Umgebungsbeschreibers"). Nach Entfernung einiger unwichtigen Details fügten wir neue Informationen hinzu. Die wichtigste war die Festlegung einzelner Laufwerke mit einem 16-bit Wort. Schon immer gab es im ENV ein Byte, das das letzte in einem System ansprechbare Laufwerk festlegte. Dies reichte jedoch für ein System, in dem nur einzelne Laufwerke, die nicht fortlaufend durchbuchstabiert waren, nicht aus, z.B. A:, B: und E:. Mit dem neuen Wort konnte jetzt exakt jedes Laufwerk definiert werden, indem "sein" Bit gesetzt wurde.

Der neue ENV enthält auch die Adressen des CCP, BDOS und BIOS und die Größen der ersten beiden. Dies ist eine Maßnahme für die Zukunft, wenn wir uns nicht mehr unbedingt an die Standardgrößen des alten CP/M halten werden. Hal Bower und Cam Cotrill, haben schon im Rahmen der Entwicklung eines neuen gebankten ZSDOS (das kurz vor der Veröffentlichung steht) mit einem CCP experimentiert, der größer ist als die üblichen 2K und einem gebankten DOS, das deutlich kleiner sein wird als die üblichen 3.5K.

Ich erwähnte schon, daß unter ZCPR33 jedes Directory, das unter einem Namen erreichbar ist, auch über DU: aufzurufen ist, obwohl es außerhalb der vom Environment gesetzten Grenzen liegt. Unter ZCPR34 war die Symetrie vollständig. Falls hier ein paßwortgeschütztes Directory mit dem DU: frei angesprochen werden kann, wird das Paßwort ignoriert. Somit können Directories jetzt immer auf beide Arten erreicht werden.

Das Interface des erweiterten Command Processors wurde liberalisiert, so daß Kommandos, die früher nicht erlaubt waren und einen sofortigen Fehler reproduzierten, wie z.B. Jokerzeichen ("?" oder "*") oder bestimmte Dateitypen, an den ECP weitergegeben werden. Damit kann etwa das ALIAS.CMD, das Alias für den für ARUNZ erweiterten Befehlsprozessor definiert, solche Befehle wie "?" enthalten, mit dem dann HELP aufgerufen würde.

Der offensichtlichste Fortschritt unter ZCPR34 war die Unterstützung der sogenannten Typ 4-Programme. Typ 3-Programme werden, wie ich schon ausführte, an eine andere Adresse als 100H geladen, aber diese Adresse ist dann auch für das Programm festgelegt. Für mich war beim Schreiben des ZCPR33 schon klar, daß es ideal sein würde, wenn die zu ladende Adresse erst beim Laden durch den CCP festgelegt würde.
Ich entschloß mich damals für den sehr einfachen Kode, der für Typ 3-Programme ausreichte. Mit diesem Kompromiß war Joe Wright nicht einverstanden und er schrieb darum eine erste Version für Typ4-Programme, der den Kode automatisch an das obere Ende des freien Memories setzte. In echter Gemeinschaftsarbeit polierten wir so lange, bis es wirklich schön funktionierte und brauchten dann dem Command Processor nur sehr wenig Kode hinzuzufügen.

Das Geheimnis lag darin verborgen, daß Joe ein PRL (page relocateable) Programm für ausführbare Dateien anwendete. Einzelheiten hierzu stehen in der Ausgabe 32 des Computer Journal. Die normale PRL-Datei beginnt mit zwei 128-Byte Header Records und ich schlug vor, daß der zum Berechnen der Adresse und zum Verschieben benötigte Kode hier statt im Command Processor stehen sollte. Joe fand einen wunderbaren Weg, dies in die Tat umzusetzen.

Hierdurch wurde nicht nur der Kode des CCP kürzer, sondern das Typ-4 Programm wurde flexibler, weil es vom Command Processor unabhängig war. In meinem nächsten Artikel im Computer Journal werde ich über die ersten oben beschriebenen Laderoutinen für Typ 4-Programme schreiben. Sie können vom Nutzer in jedes bestehende Type 4-Programm installiert werden. Auch dies ist wieder ein wunderschönes Beispiel für den modularen Aufbau, der eines der herausragenden Kennzeichen des ZCPR ist.

Weiter werde ich über die letzte Überarbeitung des ZCPR34, die Version 3.4E berichten. Vorbereitet wurde diese Version von Howard Goldstein, indem er eine Anzahl neuer Ideen einfügte, am erwähnenswertesten ein kleiner Wechsel im Typ 4-Lader, der das Ganze noch flexibler macht.

So entwickelt sich nach zehn Jahren, einer Ewigkeit für die Computerindustrie, ZCPR - das Konzept, das Richard Conn entworfen hat - immer noch weiter und fordert immer noch die Kreativität von Programmierern und Nutzern gleichermaßen. Wie immer können solche Entwicklungen nur in einer großen Gemeinschaft von Menschen wachsen, die willens und begierig sind, ihre Ideen miteinander zu teilen.

(Eine letzte Anmerkung des Übersetzers: So, das war die Übersetzung eines - hoffentlich nicht nur im Original - wirklich interessanten Artikels aus dem "Computer Journal". Wer jetzt wissen möchte, wo und wie man dieses Journal beziehen kann oder was so komische Abkürzungen und Begriffe wie "RCP", "Wheel" oder "Error Handler" bedeuten, der sollte sich mit Vorstandsmitgliedern unseres Clubs oder aber mit mir in Verbindung setzen. Ich würde auch gerne wissen, ob Bedarf an weiteren Übersetzungen oder Abhandlungen über ZCPR besteht - so einfach ins Leere zu arbeiten, ist doch etwas zu frustierend. Hartmut Schulte, Tel 05173/1248)

Deterministisches Chaos als ergänzende Betrachtungsweise

Eckhard Freuwört, Seesen · Teil 2 · (Teil 1 CLB-Heft 5/90)

Fraktale Geometrie

Die nichtlineare fraktale Geometrie erregt seit den siebziger Jahren mit der Abbildung der nach ihrem Entdecker *Benoit Mandelbrot* benannten, aus Julia- und Cantor-Mengen bestehenden Mandelbrot-Menge Aufsehen. Diese Struktur errechnet sich rekursiv nach

Gl. 19 $Z_{i+1} = Z_i^2 + W$ (Abb. 1).

Z und W sind dabei komplexe, d. h. aus Real- und Imaginäranteil bestehende Zahlen, wobei W den Kontrollparameter der Rekursion bildet. Jede Zahl „W", die den Betrag des Ergebnisses nach unendlich vielen Iterationsschritten endlich läßt (der Realanteil von „W" liegt dabei etwa zwischen -1,6 und +1,6 und der Imaginäranteil nimmt Werte etwa zwischen -1,2 und +1,2 an), gehört zu einer zusammenhängenden Julia-Menge (nach dem französischen Mathematiker *Gaston Julia*, 1893–1978, benannt). Bildet man diese Zahl „W" der Iterationsfolge von Gl. 19 nun als Computergraphik in der Ebene der komplexen Zahlen ab (wobei z. B. für jedes „W" einer zusammenhängenden Julia-Menge ein schwarzer Punkt gesetzt wird), so erhält man die Mandelbrot-Menge [17, 18]. Betrachtet man die Ränder dieser Menge vergrößert, so findet sich eine zur Umgebung unendlich dünne Trennlinie - eine scharfe, definierte Trennung ist jedoch nicht vorhanden. Hingegen tauchen immer wieder kleinere, der Mandelbrot-Menge ähnliche Strukturen auf. Diese Eigenschaft der Selbstähnlichkeit wird als Skaleninvarianz bezeichnet und stellt Symmetrie in verschiedenen Maßstäben dar, wobei man ständig auf dieselben Grundelemente trifft. In diesem Sinne bildet die Mandelbrot-Menge einen Bildspeicher von unerhörten Ausmaßen, denn relativ geringfügige Variationen von Gl. 19, etwa

Gl. 20 $Z_{i+1} = Z_i^3 + W$

erlauben die rekursive Darstellung von Bildern durch eine einzige Formel - so führt Gl. 20 beispielsweise zum Bild einer Radiolarie [19]. Die Fachrichtung des „Artificial Life" der Biologie wendet solche Rekursionen u. a. zur Computersimulation von Wachstums- und Evolutions-Prozessen an [20]. Die damit erzeugten Abbildungen von Lebewesen werden Biomorphe genannt.

Doch zurück zur Skaleninvarianz. Schon bei der Untersuchung von Turbulenzen bzw. Konvektionsströmungen in der Physik und bei der Bildung von Rohrleitungsverkrustungen oder der Entwicklung von Schlamm-Rinnen auf Behälterböden in der chemischen Technik treten selbstähnliche Strukturen auf. Die Bezeichnung für diese Strukturen lautet Fraktale; sie werden mit den Methoden der linearen fraktalen Geometrie untersucht. Zu diesem Zweck verwendet man als Maß für die Selbstähnlichkeit die sogenannte Hausdorff-Besicovitch-Dimension (*Felix Hausdorff*, 1868–1942, war deutscher Mathematiker), die häufig der Fraktaldimension gleichgesetzt wird. Das sich von dem englischen Wort „fraction" und dem lateinischen Verb „frangere" für Bruch bzw. Brechen ableitende Wort Fraktal weist bereits darauf hin, daß die besagte Dimension nicht ganzzahlig ist.

Die Bestimmung dieser Maßzahl soll nun näher erläutert werden: Wie bereits mit Gl. 15 und Gl. 18 angedeutet wurde, bewegt sich der Endwert eines chaotischen Systems in einem bestimmten Bereich, ohne sich je zu wiederholen. Dieser Bereich ist graphisch darstellbar, indem die sich verändernden Einzelergebnisse der rekursiven Berechnungen gegeneinander aufgetragen werden. Die dabei entstehende Struktur (das Phasenbild bzw. der Attraktor im Phasenraum) weist bei vergrößerter Betrachtung Selbstähnlichkeit auf - es handelt sich um ein Fraktal. Die *Brown*sche Molekularbewegung eines Teilchens in der Ebene kann hierfür als anschauliches Beispiel dienen. Dessen irreguläre Bewegung führt zu einem verwobenen Muster ineinander verworrener, zurückgelegter Strecken. Betrachtet man dieses Muster bei verschiedenen Vergrößerungen und zählt letztlich immer nur die Anzahl

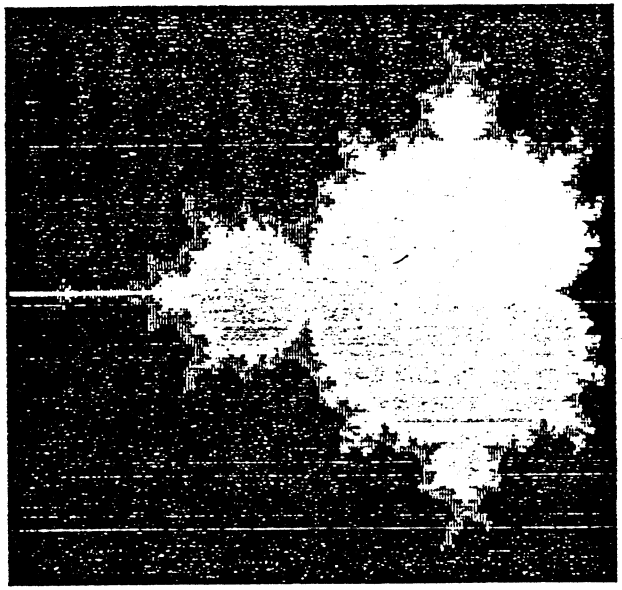


Abb. 1

a der gleichgroßen Ereignisse (hier die Anzahl der mit gleicher Länge zurückgelegten Strecken) in einem willkürlich gewählten Skalierungs-Ausschnitt mit dem Radius r, so erhält man eine Beziehung, welche einem Potenzgesetz folgt:

Gl. 21 $a = r^D$

mit D als Fraktaldimension. Nach Umformung und Logarithmieren der Gleichung läßt sich deren Wert näherungsweise bestimmen [18, 21]:

Gl. 22 $D = \frac{\ln a}{\ln r}$.

In der Praxis stellt diese Berechnung die Bestimmung der Steigung einer Geraden im doppelt logarithmischen Netz mit gegeneinander aufgetragenen Meßwerten für a und r dar (lineare Regressionsanalyse, Listing 1). Die Kenntnis der Fraktaldimension zur Charakterisierung der geometrischen Eigenschaften eines seltsamen Attraktors ermöglicht dann die vergleichende Untersuchung komplexer Systeme auf dem Wege der Computersimulation: z. B. beim Flockengrößenwachstum (als Umkehrung der Rohrleitungsverkrustung) [22] oder beim Mischen hochviskoser Flüssigkeiten [23], wie sie bei der Polymerverarbeitung oder in der Geochemie bei Gesteinsschmelzen auftreten. Ferner wurde empirisch festgestellt, daß sich das Aussehen einer Oberfläche und damit deren Topographie anhand der Fraktaldimension gut beurteilen läßt – dies ist z. B. in der Oberflächenmeßtechnik anhand der Oberflächenmaßzahlen (nach DIN 4762 und DIN 4768) allein nicht möglich. Im Gegensatz zu der eingangs erwähnten deterministischen Mandelbrot-Menge handelt es sich bei diesen zuletzt genannten Beispielen um zufällige Fraktale – wobei Zufall als allgemeine Tendenz in Richtung auf Entropieerhöhung zu verstehen ist. Hierbei werden die Kontrollparameter der den jeweiligen Simulationen zugrundeliegenden rückgekoppelten Gleichungen durch (statistische) Verteilungsgesetze beeinflußt. Da dies ohne Auswirkungen auf die Selbstähnlichkeit bleibt, läßt sich argwöhnen, daß derartige Untersuchungen am Computer realitätsfern sind – die Laserforschung belegt jedoch die Korrektheit solcher Simulationen [24]. Die Computersimulationen zeigen dann, daß sich entwickelnde komplexe Systeme qualitativ unterschiedliche Zustände annehmen können. Im Chaos ist die Bildung von Inseln der Ordnung

```
10 CLS:CLEAR:KEY OFF
20 PRINT "ANZAHL WERTEPAARE ":::INPUT N1
30 IF N1<=0 OR N1<>INT(ABS(N1)) THEN GOTO 10
40 DIM L(2,N1):B=0:X=0:Y=0:X1=0
50 Y1=0:X2=0:Y2=0:Z=0:R=0:A=0
60 FOR N=1 TO N1:FOR Q=1 TO 2
70 IF Q=2 THEN PRINT "EREIGNIS ";N;" (z.B.ERHOHUNG/VERTIEFUNG) ":
80 IF Q=1 THEN PRINT "UMFELD   ";N;" (z.B.KREISRADIUS/FLACHE ) ":
90 INPUT L(Q,N):IF L(Q,N)<1 THEN GOTO 90
100 L(Q,N)=LOG(L(Q,N))
110 NEXT Q:NEXT N
120 FOR N=1 TO N1
130 X=X+L(1,N):Y=Y+L(2,N):X2=X2+L(1,N)*L(1,N):Y2=Y2+L(2,N)*L(2,N)
140 Z=Z+L(1,N)*L(2,N)
150 NEXT N
160 X1=X/N1:Y1=Y/N1
170 IF N1*X2-X*X=0 THEN GOTO 190
180 B=(N1*Z-X*Y)/(N1*X2-X*X)
190 A=Y1-B*X1
200 X0=SQR(N1*X2-X*X):Y0=SQR(N1*Y2-Y*Y)
210 IF X0*Y0=0 THEN R=0:GOTO 230
220 R=(N1*Z-X*Y)/(X0*Y0)
230 PRINT "FRAKTALDIMENSION D= ";B
240 PRINT "KORRELATIONSKOEFFIZIENT r= ";R
250 PRINT :PRINT "NOCHMAL ?  (J/N)      ":::INPUT A$
260 IF A$="J" OR A$="JA" THEN GOTO 10
270 END
```

Listing 1

durchaus normal. Unter welchen Bedingungen dies erfolgt, ist Gegenstand der Untersuchungen der Synergetik.

Synergetik

Die Synergetik als Lehre vom Zusammenwirken untersucht sich entwickelnde, gegenseitig beeinflussende komplexe Systeme [25], wobei ein System als Kontrollparameter (Ordner) des anderen auftritt. Die Systementwicklung ist dabei durch die Rekursion einer als Instruktionssatz zur Durchführung eines Prozesses aufzufassenden rückgekoppelten Funktion vorgegeben und basiert auf positiver Rückkoppelung. Wie bereits früher dargestellt wurde, kommt es dann über kurz oder lang zur Ausbildung eines instabilen Gleichgewichts (Bifurkationspunkt), an dem das System – ähnlich einem Phasenübergang – einen qualitativ neuen Zustand einnehmen kann. Dies ist mit der Zufuhr von Aktivierungsenergie zum Starten einer exothermen Reaktion vergleichbar und soll am Beispiel der Bildung von FeS aus den Elementen beschrieben werden. Bekanntermaßen erfolgt die exotherme Reaktion nach

Gl. 23 Fe + S → FeS,

wobei eine ausreichende Aktivierungsenergie Voraussetzung ist. Betrachtet man eine Pulvermischung dieser Elemente als System, so kann dieses 3 Arten von Stabilität aufweisen:

1. Stabilität im Sinne von Beharrlichkeit bzw. Statik, wobei beide Elemente über einen langen Zeitraum hinweg unverändert vorliegen (Pulvermischung).
2. Stabilität im Sinne von Robustheit, wobei diese angibt, wie ein System mit dem Zuwachs bzw. Verlust von System-Komponenten (beispielsweise der Änderung des Energiehaushalts) zurechtkommt. Wird die Aktivierungsenergie unterschritten, so setzt keine Reaktion ein, und das System erweist sich als robust.
3. Stabilität im Sinne von Elastizität als Maß dafür, wie schnell ein System nach einer Störung (z. B. durch Energiezufuhr) in einen energetisch günstigen Zustand (dies kann die Ausgangslage oder aber ein qualitativ neuer Zustand sein) übergeht. In Gl. 23 entspricht dies der FeS-Bildung.

Bei ausreichender Zufuhr von Aktivierungsenergie erfolgt daraufhin exotherme Reaktion. Wird die Energiezufuhr jedoch etwas knapper bemessen, so wird ein höchst instabiles Gleichgewicht erreicht. An diesem Verzweigungs- bzw. Bifurkationspunkt kommt es zu Schwankungen – sogenannten kritischen Fluktuationen –, welche entsprechend der Fremdbeeinflussung durch die zugeführte Energie einer statistischen Verteilung unterliegen. Aufgrund von positiver Rückkoppelung verstärken sich dann an sich vernachlässigbar kleine, zufällige Störungen selbst. Dieses Verhalten findet man u. a. auch bei der Untersuchung des Quellungsgrades von Gelen [26] (welcher bei der Entsorgung amphotermetallhaltiger Abwässer die Schlammalterung beeinflußt) oder im Bereich der Ökologie [27]. Die beschriebenen Störungen durch Fremdbeeinflussung (z. B. durch ein anderes System) führen bei dem rekursiven skaleninvarianten Prozeß zum Symmetriebruch und somit zur Bildung eines zufälligen Fraktals, das mittels linearer fraktaler Geometrie untersucht werden kann. Ferner bewirkt die Weiterentwicklung des Symmetriebruchs durch positive Rückkoppelung u. U. eine dramatische Systemänderung, die zu einer neuen Ordnung führt. Diese wiederum beeinflußt ihrerseits Fremdsysteme und deren Entwicklung. Ähnliche Prinzipien versucht man heute bei der Expertensystem-Programmierung mit Ansätzen wie Certainty-Factors, Fuzzy-Logic und Plausibilitätsintervallen [28] zu nutzen. Das *Lorenz*sche Wasserrad [29] mag als Gedankenexperiment zur Veranschaulichung dienen:

System A sei ein Wasserrad. System B sei ein Wasserstrom, der nicht wie üblich tangential, sondern mittig von oben auf das Wasserrad trifft, und System C sei eine Mechanik, die durch die Drehung des Wasserrades angetrieben wird und in ihrer Wirkung drehrichtungsabhängig ist. Erfolgt nun ein langsamer Wasserzufluß, so wird die Reibung des Rades nicht überwunden, und es kommt nicht zur Drehung. Bei verstärktem Zufluß erfolgt dann stetige Rotation, wobei die Drehrichtung und damit die Wirkung von System C nicht vorherbestimmbar, jedoch konstant ist. Bei sehr starkem Wasserzufluß hingegen entwickelt sich die Drehung chaotisch, da die Füllmenge der Schaufeln des Rades von der aktuellen Drehgeschwindigkeit abhängt – womit sich die Wirkung von System C mehrfach umkehrt. Beeinflußt System C nun seinerseits partiell durch Öffnen oder Schließen eines Schiebers den Wasserstrom des Systems B, so werden alle drei Systeme ständig zwischen den Zuständen konstante Rotation, indeterministisches Verhalten und instabilem Stillstandsgleichgewicht hin und hergeschleudert. Bildet man dieses Verhalten im Phasenraum ab, so erhält man einen seltsamen Attraktor mit fraktaler Struktur, bei dem an Bifurkationspunkten über die neue Ordnung entschieden wird. Normalerweise vernachlässigbar kleine Einflüsse führen dann die Entscheidung herbei und bewirken neue Verhaltensweisen aller drei Systeme.

Berücksichtigt man nun energetisch günstigere und daher stabilere Zustände, so ist die Zusammenlagerung von Subsystemen – die durch fraktale Strukturen darstellbar sind – nach dem Prinzip des kleinsten Widerstandes ebenfalls möglich (übrigens drängt sich bei diesem Prinzip die Ähnlichkeit mit der sehr rechenintensiven Problematik des kürzesten Netzwerkes von Ver- und Entsorgungsleitungen auf [30]). Die Zusammenlagerung kann zur Selbstorganisation führen, wobei Subsysteme ihre alten Eigenschaften zugunsten eines neuen Systemverhaltens verlieren – die Aussage der Gestaltpsychologie „Das Ganze ist mehr als die Summe seiner Teile" wird hierdurch bestätigt. Dies läßt sich mittels eines Beispiels aus der Elektrotechnik veranschaulichen (Abb. 2):

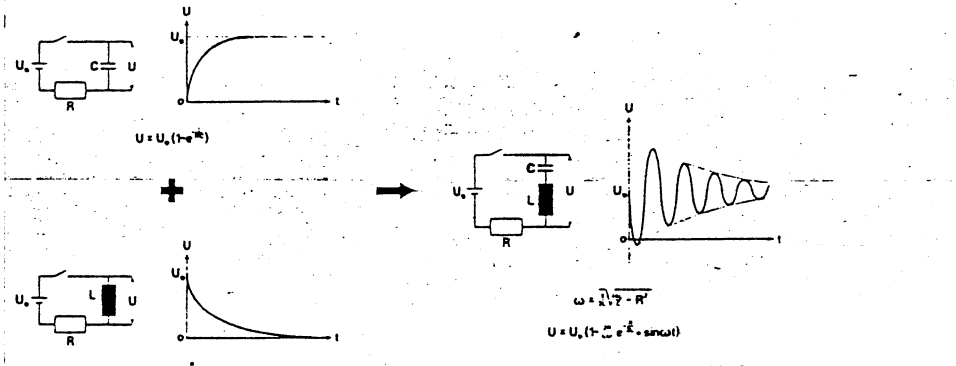


Abb. 2

Im Subsystem R-C-Glied kommt es nach Schließen des Stromkreises bedingt durch die Aufladung des Kondensators zur Spannungszunahme. Im Subsystem R-L-Glied tritt durch den Widerstand der Spule aufgrund von elektromagnetischer Induktion das Gegenteil ein. Bei deterministischer Betrachtungsweise ließe sich durch die Koppelung dieser Subsysteme ein konstanter Spannungsverlauf erwarten – dies erfolgt jedoch nicht, da sich ein Schwingkreis gebildet hat. Ähnliche Zusammenlagerungen von sich ergänzenden Eigenschaften führen in Biologie und Biochemie zur sogenannten kumulativen Selektion [31]. Von Interesse sind in diesem Zusammenhang beispielsweise die Theorie zu den an kristallinen Tonmineralen katalytisch synthetisierten organischen Replikator-Molekülen des englischen Chemikers *Graham Cairns-Smith* [32] sowie der Hyperzyklus der autokatalytischen Replikation von Biomolekülen von *Manfred Eigen* [33]. Beide Untersuchungen führen in letzter Konsequenz zu den von der Biochemie aufgeklärten metabolischen Zyklen wie u. a. dem Zitratzyklus. Derartige Zyklen sind zwar weit vom niedrigsten energetischen Gleichgewicht (wie es z. B. in einer kristallinen Struktur vorliegt) entfernt, erhöhen jedoch die Entropie entsprechend dem 2. Hauptsatz der Thermodynamik und stellen somit nur weitere Inseln der Ordnung auf der Linie der Gleichgewichtspunkte dar. Die Untersuchung solcher Zusammenhänge durch Computersimulation steht erst am Anfang: die entsprechende Forschungsrichtung befaßt sich mit dem deterministischen Chaos. Fortsetzung folgt

Fraktale – Vorkommen und Anwendung, Simulation auf dem Computer

Dipl.-Chem. Heiko Schinke, Doz. Dr. sc. Martin Schleiff, Dipl.-Biochem. Mirco Wahab, Dr. Wolfgang Brandt, Prof. Dr. habil. Alfred Barth
Biotechnikum und Sektion Mathematik der Martin-Luther-Universität Halle-Wittenberg, O-4020 Halle an der Saale

Insbesondere in populärwissenschaftlichen Zeitschriften finden sich in den letzten Jahren immer wieder Veröffentlichungen zum Thema „Fraktale", auch im Zusammenhang mit der theoretischen Beschreibung von Ordnung und Chaos. Besonders eindrucksvoll sind dabei die vielfältigen Computergrafiken, welche die Schönheit der einerseits so bizarren, aber andererseits unendlich regelmäßigen fraktalen geometrischen Gebilde anschaulich machen. Auch in unterschiedlichen Fachzeitschriften werden interessante Ergebnisse der Anwendung der Fraktale auf spezielle naturwissenschaftliche Probleme beschrieben. Man kann annehmen, daß viele Erscheinungen der uns umgebenden Natur fraktalen Charakter haben und sich mit Hilfe der fraktalen Geometrie adäquat simulieren lassen.

In diesem Beitrag soll zunächst eine kurze, allgemeinverständliche Einführung in die Fraktaltheorie gegeben werden. Danach werden Beispiele aus verschiedenen Gebieten der Naturwissenschaften betrachtet, die die vielfältigen Anwendungsmöglichkeiten des Fraktalbegriffs auf konkrete Erscheinungen nachweisen. Dabei wird sowohl in der Auswahl und im Umfang als auch in der angeführten Literatur kein Anspruch auf Vollständigkeit erhoben. Der Schwerpunkt der Beispiele liegt, dem Charakter dieser Zeitschrift entsprechend, auf den Gebieten der Chemie und Biochemie.

Aber auch auf die Erzeugung von Fraktalen mit Hilfe des Computers und ihre Darstellung als Computergrafik wird kurz eingegangen. Es ergeben sich so bei geeigneter Farbgebung Bilder von hoher ästhetischer Schönheit.

Was sind Fraktale?

Benoit Mandelbrot, der Begründer der modernen Fraktaltheorie, hat den Begriff „Fraktal" zunächst nicht exakt (im formal-mathematischen Sinn) definiert. Um die typischen und wesentlichen Eigenschaften herauszuarbeiten, betrachtet er ein anschauliches Beispiel: Er stellt die Frage nach der Länge der Küstenlinie von Großbritannien. In unterschiedlichen Nachschlagewerken findet man jeweils andere Werte. Mandelbrot schreibt hierzu: „Die Länge einer typischen Küstenlinie... ist schlecht bestimmt. Wenn man verschiedene Küstenlinien bezüglich ihrer Ausdehnung vergleichen will, dann ist die Länge ein ungeeigneter Begriff" [1].

Zur Verdeutlichung dieser Aussage sei ein Landvermesser betrachtet, der die Küste vollständig vermessen soll. Nach jeweils einer gleichen Länge l_1 (geradlinige Entfernung, Luftlinie) setzt er einen Markierungspunkt. Die Länge $L_1(l_1)$ der Küste ist dann die Anzahl dieser Markierungspunkte multipliziert mit dem Abstand l_1 zwischen zwei solchen Punkten. Nun wird dieser Meßprozeß wiederholt mit einem kleineren Abstand l_2 zwischen jeweils zwei Meßpunkten. Man hat dann eine größere Anzahl von Meßpunkten und erhält $L_2(l_2)$, einen Wert, der größer als $L_1(l_1)$ sein wird. Eine weitere Verkleinerung auf l_3 führt zu einem dritten Wert $L_3(l_3)$, der noch größer ist.

Während bei genügend glatten geometrischen Gebilden, z. B. einem Kreis, die so ermittelten Längen L_n schnell einem Grenzwert zustreben (in der Mathematik als Bogenlänge bezeichnet), scheinen bei natürlichen Grenzlinien, wie der Küstenlinie, die Werte L_n zunächst immer weiter zu wachsen.

Wodurch kommt dieser Effekt zustande, der die Küstenlinie deutlich von den glatten Kurven unterscheidet? Bei jeder Vergrößerung des Maßstabes werden neue Feinheiten der Küste zugänglich: Buchten teilen sich in Unterbuchten, Halbinseln lassen neue Unter-Halbinseln erkennen. Leicht nachvollziehbar ist diese Erscheinung auch auf Landkarten, wenn die gleichen Gebiete auf Karten mit unterschiedlichem Maßstab betrachtet werden. Das Typische für das andersartige Verhalten der Küstenlinie und anderer natürlicher Grenzen ist also die Selbstähnlichkeit. Halbinseln und Buchten sind auf Karten mit unterschiedlichem Maßstab ähnlich. In Abstraktion von dem konkreten Beispiel der Küstenlinie kann man also definieren: Eine Punktmenge heißt fraktal, wenn sie selbstähnlich ist, d. h. jeder noch so kleine Teil der Menge hat bei passender Vergrößerung das gleiche Aussehen wie die ursprüngliche Menge.

Mit dieser Definition wird das Typische und Gemeinsame solcher geometrischen Gebilde wie einer Küstenlinie herausgearbeitet. Gleichzeitig wird die typische Eigenschaft, die Selbstähnlichkeit, in der Weise umfassend verallgemeinert, daß sie bei beliebigen Ausschnittsvergrößerungen gültig sein soll. Auf diese Weise wird unter Abstraktion von dem konkreten Erscheinungsbild eine fraktale Menge der formal-mathematischen Untersuchung zugänglich. Man kann nun auf rein mathematischem Weg allgemeine Schlußfolgerungen herleiten, die dann automatisch für alle konkreten fraktalen Gebilde gültig sind. Das ist der Vorteil bei einer Anwendung mathematischer Methoden zur Untersuchung konkreter Gebilde. Man kann die universell gültigen mathematischen Ergebnisse nutzen, um unmittelbar zu Aussagen über konkrete Gebilde zu gelangen.

Allerdings muß dabei eine wesentliche Einschränkung gemacht werden. Konkrete Gebilde besitzen die fraktalen Eigenschaften nur in einem bestimmten Größenordnungsbereich. Bei der Anwendung von Fraktalen in der Chemie hat es beispielsweise keinen Sinn, eine Verkleinerung bis zu atomaren Größenordnungen vorzunehmen. Die aus der allgemeinen Theorie hergeleiteten Aussagen haben daher bei der Anwendung auf konkrete Gebilde nur bei einer bestimmten Größenordnung einen Sinn.

Mathematische Eigenschaften der Fraktale

Die hier gegebene mathematische Definition der fraktalen Menge geht von einer fertigen Menge aus. Es wird dabei nicht angegeben, wie eine solche Menge tatsächlich konstruiert werden kann (beispielsweise auf dem Bildschirm eines Computers). Im folgenden soll ein solcher konstruktiver Prozeß angegeben werden. Man geht von einer beliebigen Menge aus. Dann wird angegeben, wie verkleinerte Abbilder der ursprünglichen Menge (meistens drei oder vier Bildmengen) in bezug auf die Ausgangsmenge zu positionieren sind (Mehrfach-Verkleinerungs-Kopier-Regel). Dieser Prozeß wird nun auf alle Bildmengen angewendet und dann das Ganze immer wiederholt (iterativer, rekursiver Prozeß). Da dieser Prozeß auch auf jede Teilmenge beliebig oft angewendet wird, ist es klar, daß man dadurch eine selbstähnliche Menge, also eine fraktale Menge, erhält. Ein wichtiger mathematischer Satz sagt nun, daß die entstehende fraktale Menge unabhängig von der Ausgangsmenge ist, mit der der Prozeß begonnen wurde. Die fraktale Menge ist also allein durch die wenigen Abbildungsvorschriften der Mehrfach-Verkleinerungs-Kopier-Regel vollständig bestimmt. An den Bildern wird aber deutlich, welche bizarre Vielfalt von Punktmengen man so erhalten kann.

In dem einleitenden Beispiel von der Küstenlinie war die Frage nach der Länge dieser Linie gestellt. Es war ersichtlich, daß die in der üblichen Weise definierte Länge einer Kurve bei einer fraktalen Menge nicht zu sinnvollen Werten führt. Man führt daher als Maßzahl für die Größe einer fraktalen Menge die sogenannte **fraktale Dimension** ein, die auf eine Arbeit des Mathematikers *Hausdorff* (1868 – 1942) aus dem Jahre 1919 zurückgeht. Diese **Hausdorffdimension** ist folgendermaßen erklärt: Man überdeckt die Menge mit einem Quadrat (bzw. im Raum mit einem Würfel). Nun wird jede Seite des Quadrats in n gleiche Teile geteilt, so daß man ein Gitter von kleinen Quadraten erhält. Mit a_n wird die Anzahl derjenigen kleinen Quadrate bezeichnet, die einen Punkt der fraktalen Menge enthalten. Dann ist der Grenzwert

$$D_H = \lim_{n \to \infty} \frac{\ln (a_n)}{\ln (n)}$$

der im Fall einer fraktalen Menge stets existiert, die Hausdorffdimension. Man kann diese Formel auch noch anders deuten. Wenn man die Anzahl der überdeckenden kleinen Quadrate für verschiedene Unterteilungen n ermittelt und die Werte n und a_n als Punkte in einem doppeltlogarithmischen Koordinatenpapier einzeichnet, so liegen diese Punkte annähernd auf einer Geraden, deren Anstieg die Hausdorffdimension ist. Der Vorteil dieses Verfahrens ist, daß man damit auch die fraktale Dimension realer Objekte wie beispielsweise einer Küstenlinie bestimmen kann. Die Hausdorffdimension kann entsprechend ihrer Definition eine gebrochene Zahl sein. Aus der Tatsache der gebrochenen Dimension ist die Bezeichnung Fraktal entstanden (vgl. z. B. Fraktur, Fraktion). Für die Kochsche Schneeflockenkurve in Abb. 2 ist $D_H = 1{,}262$.

Es sei hier darauf hingewiesen, daß es in der Theorie der Punktmengen noch weitere Dimensionsbegriffe gibt. So hat eine Punktmenge die topologische Dimension D_t, wenn man sie durch Punktmengen der Dimension D_t-1 zerteilen kann. Die Kochsche Schneeflockenkurve hat also die topologische Dimension 1, da man sie durch einzelne Punkte, die die Dimension Null haben, zerlegen kann. Für fraktale Mengen gilt $D_t < D_H$. Diese Ungleichung bringt die anschauliche Tatsache zum Ausdruck, daß eine fraktale Menge stets verzweigt und bizarr ist.

In der Mathematik begegnet man fraktalen Mengen bei der Untersuchung dynamischer Systeme. Der Einfachheit wegen werden nur dynamische Systeme betrachtet, die durch wenige Zustandsparameter (z. B. drei Parameter) beschrieben werden. Es interessiert nun, welche Werte die Parameter nach einer langen zeitlichen Entwicklung annehmen werden.

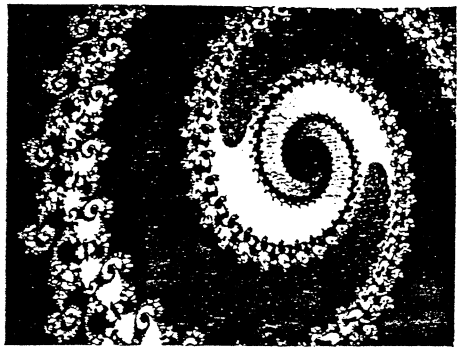


Abb. 1: Apfelmännchen in 256 Farben (Ausschnitt aus Abb. 8). Das Bild wurde mit dem Programm FRAKFILM (auf der Programm-Diskette) erzeugt.

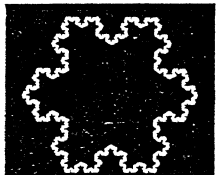


Abb. 2: Kochsche Schneeflocke (erzeugt mit dem Programm TURTLE)

Ein bekanntes Beispiel sind die idealen Planetenbahnen, wie sie sich aus den Keplerschen Gesetzen ergeben. Die Grenzmenge ist in diesem Fall eine gewöhnliche Kurve ($D_t = D_H = 1$). Bei komplizierteren dynamischen Systemen ist auch die Grenzmenge komplizierter, sie kann fraktale Eigenschaften aufweisen. Man spricht dann von chaotischem Verhalten. Es ist in diesem Fall nämlich nicht möglich, langfristig vorauszusagen, auf welchem Teil der fraktalen Grenzmenge die Parameterwerte nach einiger Zeit liegen werden.

Bei anderen dynamischen Systemen kann es mehrere verschiedene Grenzzustände geben. Man interessiert sich dann dafür, aus welchem Einzugsgebiet ein bestimmter Zustand erreicht wird. Die Grenze eines solchen Einzugsgebietes (z. B. Juliamenge, auch als Apfelmännchen bezeichnet) hat häufig fraktale Eigenschaften. Man sieht, daß die moderne Theorie dynamischer Systeme, der Wechsel von Ordnung und Chaos, vielfach mit dem Begriff der Fraktale zusammenhängt [1, 2, 13].

Fraktale in unserer Umwelt

Nachdem bisher die Fraktale mehr von ihrer theoretischen Seite betrachtet wurden, sollen nun Beispiele für das Vorkommen von fraktalen Eigenschaften in der Natur folgen. Dabei sei noch einmal darauf hingewiesen, daß der fraktale Charakter natürlicher Gebilde stets nur in einem bestimmten Größenordnungsbereich ausgeprägt ist.

Als einführendes Beispiel wurde bereits die fraktale Natur von Küstenlinien genannt. Allgemein haben natürliche Grenzen zwischen zwei Systemen (z. B. zwischen Land und Meer) häufig fraktalen Charakter. Auch Flüsse und die Einzugsgebiete von Flüssen weisen fraktale Eigenschaften auf, d. h. alle Flüsse und alle Flußeinzugsgebiete sind untereinander ähnlich. Die fraktale Dimension beträgt bei diesen Beispielen etwa 1,2.

In der Natur gibt es noch andere unregelmäßige Begrenzungslinien, beispielsweise die Oberfläche von Gebirgen. Aus dem Prozeß der Gebirgsentstehung wird der fraktale Charakter dieser Oberfläche deutlich: Ausgehend von einem Plateau (in der Computersimulation einer Dreiecks-Fläche, Abb. 3a) werden einzelne Punkte durch den Druck aus dem Erdinnern angehoben, andere durch die Erosion abgetragen (Abb. 3b). Dieses so entstandene, unregelmäßige Gebilde wird erneut teilweise angehoben und wieder abgetragen, d. h. den gleichen Vorgängen unterworfen (Abb. 3c – 3d). Im Laufe der Zeit entstehen so die unregelmäßigen Oberflächenformen der Erde. Dieser Entstehungsprozeß ist ein iterativer, rekursiver Prozeß, der hier zu einer Grenzfläche mit fraktalen Eigenschaften führt. Betrachtet man Wolken am Himmel, so stellt man fest, daß sie unregelmäßig begrenzt sind, aber andererseits selbstähnlich sind. Wolken stellen also fraktale Gebilde dar. Die fraktale Dimension ihrer Umfangslinie (der Grenzlinie zum wolkenfreien Gebiet) beträgt etwa 1,33.

Auch in der belebten Natur haben die Grenzflächen zwischen zwei Systemen häufig fraktalen Charakter. Allgemein bekannt ist, daß das Gehirn von Säugetieren gefurcht ist. Diese Furchungen sind sehr unregelmäßig, aber in gewisser Weise selbstähnlich. Die fraktale Dimension der Gehirnoberfläche konnte zu rund 2,75 ermittelt werden. Auch die Lungenoberfläche ist fraktal. Ermittelt man die

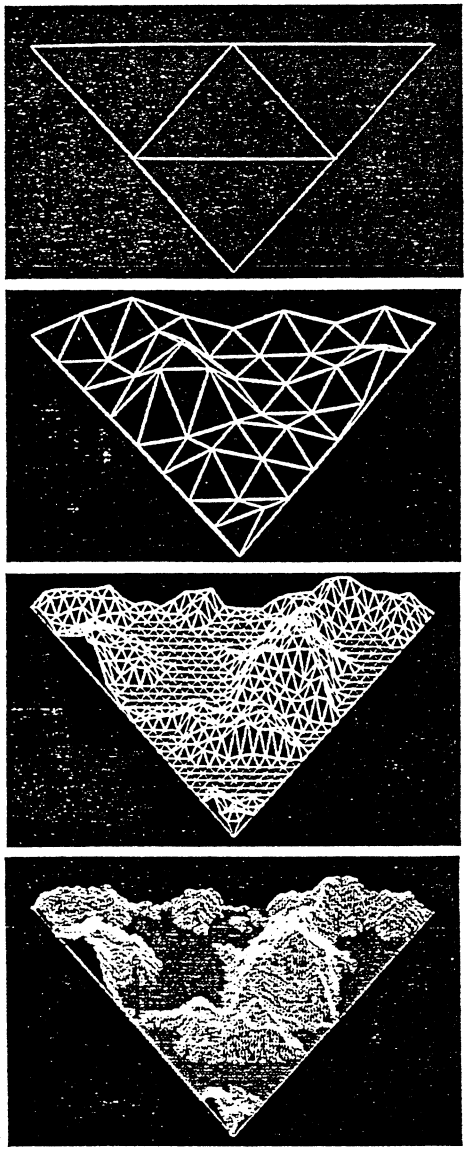


Abb. 3: Entstehung eines Gebirges auf dem Computer in 4 Abschnitten, dargestellt sind das 1., 3., 5. und 7. Bild (erzeugt mit dem Programm BERG)

Fläche der menschlichen Lungenalveolen mit Hilfe des Lichtmikroskops, so findet man 80 m², während man mit dem Elektronenmikroskop (und somit bei einer höheren Auflösung) 140 m² findet. Diese Fläche wird also besser durch ihre fraktale Dimension charakterisiert, die bei $D_H = 2{,}17$ liegt.

Es sei hier noch einmal an die Bedeutung der fraktalen Dimension erinnert. Ein Wert von D_H zwischen 2 und 3 bedeutet, daß das zu beschreibende Gebilde zwischen einem ebenen Gebilde und einem räumlichen Gebilde einzuordnen ist. Je näher der Wert von D_H bei der Zahl 3 liegt, um so stärker wird durch das geometrisch zweidimensionale Gebilde der Raum erfüllt.

Pflanzen bilden mit ihrem Wachstum ebenfalls fraktale Gebilde aus. So wird in der Struktur eines Baumes, bestehend aus Stamm, Ästen, Zweigen und Blättern, der selbstähnliche Charakter deutlich (Abb. 4a). Auch Blätter für sich haben häufig eine fraktale Struktur. Bäume und Blätter (insbesondere Farnblätter, Abb. 4b) werden daher gern als Demonstrationsbeispiele genutzt, um zu zeigen, wie gut die vom Computer erzeugten fraktalen Gebilde natürliche Gegenstände simulieren können. Mit etwas Zufall lassen sich noch zahlreiche weitere Formen erzeugen (Abb. 4c). Auch die Abb. 5a und 5b wurden mit einem Computer erzeugt und stellen einen simulierten Schnitt durch einen Pflanzenstengel dar.

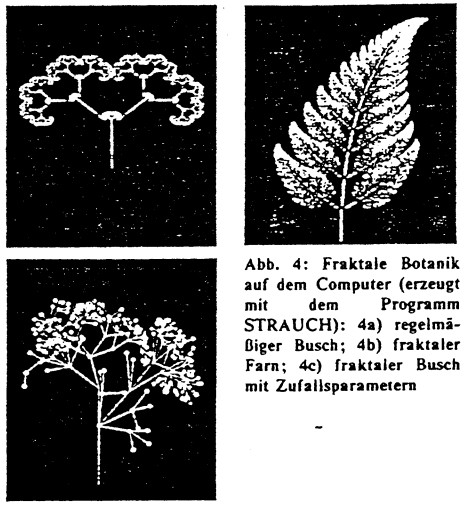


Abb. 4: Fraktale Botanik auf dem Computer (erzeugt mit dem Programm STRAUCH): 4a) regelmäßiger Busch; 4b) fraktaler Farn; 4c) fraktaler Busch mit Zufallsparametern

Fraktale in der Chemie

Viele chemische Prozesse laufen an Oberflächen ab (z. B. heterogene Katalyse). Es ist daher zu erwarten, daß hierbei auch Oberflächen mit fraktalen Eigenschaften eine Rolle spielen werden. In Abb. 6 ist ein Schnitt durch den sogenannten Mengerschwamm (in zwei Auflösungen) dargestellt, einem fraktalen Gebilde mit der Dimension 2,73, das als idealisiertes Modell für einen Katalysator mit durchgehenden Poren dienen kann. Wie bei einem realen Katalysator sind die Poren unregelmäßig verteilt und streuen in ihrer Größe über einen weiten Bereich. Die Oberfläche des Katalysators kann man daher, genauso wie bei der bereits beschriebenen Küstenlinie, nicht direkt bestimmen, da bei jeder Vergrößerung wieder neue Poren zugänglich werden. Zur Bestimmung der Oberfläche eines realen Katalysators wird diese mit einer Monoschicht von Adsorbatmolekülen (z. B. Stickstoff, Edelgase; idealisiert als Kugeln) bedeckt. Die Anzahl der bedeckenden Moleküle wird bestimmt und daraus die Oberfläche berechnet. In Abhängigkeit von den verwendeten Molekülen erhält man unterschiedliche Werte für die Oberfläche. Wenn man die Oberfläche mit verschieden großen Molekülen (Kugeln) mit den Radien $r_1, r_2, \ldots$ bedeckt, die jeweilige Anzahl $a_1, a_2, \ldots$ ermittelt, so kann man aus diesen Werten in der schon beschriebenen Weise die fraktale Dimension der Katalysatoroberfläche bestimmen (vgl. ausführlich in [19a – 19c]). Wie bereits oben erwähnt, hat diese empirisch bestimmte Dimensionszahl nur in dem Größenbereich einen Sinn, der durch die Radien der benutzten Moleküle eingegrenzt wird. Das ist stets zu beachten, wenn die fraktale Dimension als Katalysatorkenngröße verwendet wird.

Die in Abb. 2 dargestellte Kurve wird als Schneeflokkenkurve bezeichnet. Sie kann erzeugt werden, indem, ausgehend von einem gleichseitigen Dreieck, aus jeder Kante jeweils ein neues Dreieck ausgestülpt wird. Die Bezeichnung als Schneeflocke ist kein Zufall, denn auch bei der Bildung von Schneeflocken werden die neuen H_2O-Moleküle bevorzugt an geraden Kanten angelagert. Bei anderen Anlagerungsprozessen entstehen blumenartige oder baumartige Gebilde, beispielsweise bei mikrobiologischem Wachstum, bei elektrolytischen Kupferabscheidungen und bei Kolloidagglomerationen. Auch das sind fraktale Strukturen (siehe die Computersimulationen einer Anlagerung

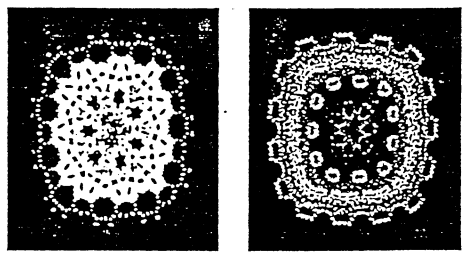


Abb. 5: Schnitt durch einen fraktalen Pflanzenstengel (erzeugt mit dem Programm PFLANZE). Geändert wurde nur der Parameter von b, in Bild a ist b = 0,01, in Bild b ist b = 0,005.

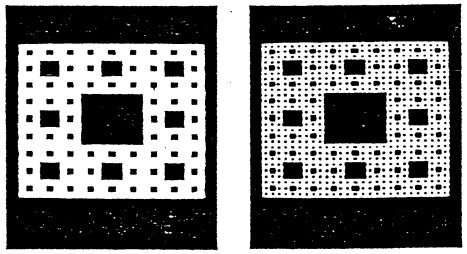


Abb. 6: Schnitt durch den Mengerschwamm in zwei verschiedenen Auflösungen

die Berechnungsprogramme dafür wurden [9] entnommen).

Die Oberfläche von gefalteten Proteinen weist ebenfalls fraktalen Charakter auf. Die fraktale Dimension beträgt hier etwa 2,2. Auch die RNA-Sekundärstruktur wird in der Literatur [17] als fraktal beschrieben. Dabei wurde eine gute Übereinstimmung zwischen dem Entwicklungsniveau der RNA und ihrer fraktalen Dimension beobachtet.

Weitere Beispiele für Fraktale in der Chemie seien hier nur kurz aufgezählt: Vernetzte und verzweigte Polymere in der makromolekularen Chemie, Struktur von Zeolithen, Aggregationsprozesse, Diffusionsfronten. Gute Übersichten bieten das Buch von *Avnier* [12] sowie das PC-Lernprogramm von *J. Bargon* [13].

Fraktale auf dem Computer

Die Berechnung fraktaler Gebilde ist mit einem iterativen, rekursiven Verfahren möglich. Das ist eine verhältnismäßig einfache Rechenvorschrift, die aber sehr oft angewendet werden muß (siehe die Programme). Die Ausführung einer solchen Rechnung mit Zettel und Bleistift ist sehr eintönig und ermüdend und wird daher kaum durchgeführt.

Umgekehrt eignen sich solche iterativen Rechenvorschriften sehr gut für die Ausführung auf einem Computer. Man hat meist ein einfaches und kurzes Rechenprogramm, welches wiederholt, mitunter tausendfach, abgearbeitet werden muß. Mit der Entwicklung einer leistungsfähigen Rechentechnik wurde es möglich, fraktale Gebilde in großer Vielfalt zu berechnen und auf dem Bildschirm grafisch darzustellen. Heute gibt es für jeden Heimcomputer und Personalcomputer fertige Programme zur Berechnung fraktaler Gebilde. Da diese Programme einfach aufgebaut sind; kann man sie leicht selbst abwandeln und so nach Belieben völlig neue Bilder von meist bizarrer Schönheit erzeugen.

Einen Eindruck geben die Abb. 1 sowie 8, 9 und 10. Sie stellen ein fraktales Gebilde dar, das sich als Grenze des

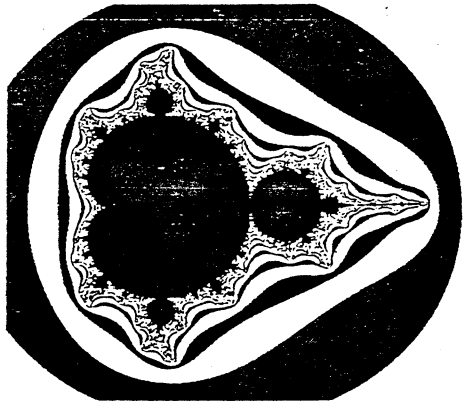


Abb. 8: Apfelmännchen (Gesamtansicht)

Abb. 7: Anlagerungsprozesse in der Chemie als Computersimulation: 7a) Anlagerung (Kristallisation); 7b) Sedimentation

Einzugsbereiches eines bestimmten dynamischen Prozesses ergibt (häufig als Apfelmännchen bezeichnet, siehe Listing 1). Abb. 8 stellt das vollständige Gebilde dar (in 2-Farbdarstellung). Dabei wurde der Bereich in der Mitte (das „Chaos", d. h. die Menge der Punkte, die nach einer bestimmten Anzahl von Iterationsschritten noch keine eindeutige Zugehörigkeit zu einem bestimmten Einzugsbereich erkennnen lassen) schwarz gezeichnet. Die weißen und schwarzen Ringe darum herum lassen die Anzahl der Iterationsschritte erkennen, die an den entsprechenden Punkten notwendig sind. Die Bezeichnung dieses Gebildes kommt von der Ähnlichkeit des inneren schwarzen Berei-

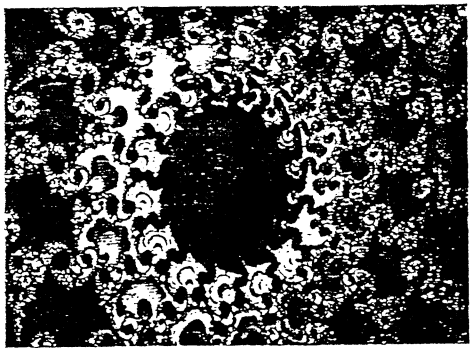


Abb. 9: Apfelmännchen (Ausschnitt aus Abb. 8; erzeugt mit dem Programm FRAKFILM)

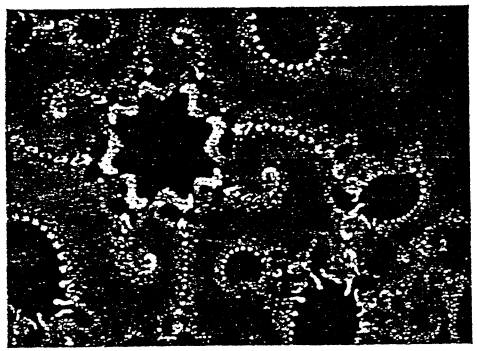


Abb. 10. Apfelmännchen (Ausschnitt aus Abb. 8; erzeugt mit dem Programm FRAKFILM)

len jeweils Ausschnitte daraus dar, wobei auch unterschiedliche Möglichkeiten der Farbgebung verwendet wurden. Die Berechnungsvorschrift ist auch hier iterativ und rekursiv, aber mathematisch von anderer Struktur als die Berechnungsvorschrift, die zur Schneeflockenkurve oder zu baumartigen bzw. blattartigen Fraktalen führt. Für interessierte Leser enthält das Literaturverzeichnis Hinweise auf Programme zur Berechnung fraktaler Strukturen auf dem Commodore C64 sowie auf dem Personalcomputer [3 – 11], außerdem ist zu diesem Artikel eine Programm-Diskette mit Fraktal-Programmen erhältlich (s. u.).

An den zwei Teilbildern der Abb. 5 soll noch einmal gezeigt werden, wie minimale Änderungen eines Ausgangsparameters zu grundlegenden Veränderungen in der Struktur des sich als Grenzfall ergebenden Bildes führen (siehe dazu Listing 3). Der Wert von b beträgt in Abb. 5a 0,01, in Abb. 5b 0,005. Nach 50 000 Iterationsschritten (Punkten auf dem Bildschirm) wurden diese beiden Bilder erhalten, zwischen denen sich kaum eine Ähnlichkeit finden läßt.

Es ist also offensichtlich, daß man durch eine andere Wahl gewisser Parameter im Berechnungsprogramm völlig neue Strukturen und Bilder erzeugen kann.

Der Vorteil der Bilderzeugung in der beschriebenen Weise besteht darin, daß man zur Erzeugung eines kompletten Bildes nur eine einfache und kurze Berechnungsvorschrift braucht. So werden Gebirgslandschaften in Computerspielen oft als Fraktale berechnet (z. B. bei dem Spiel „Rescue of Fractalus" für 8-bit Homecomputer). Das ist effektiver, als wenn das komplette Landschaftsbild gespeichert werden müßte. Auch Bäume und ganze Wälder werden auf diese Weise als Fraktale „synthetisch" erzeugt und liegen nicht als fertige Bilder vor (siehe Abb. 4c), wobei in begrenztem Umfang der Zufall bei der Berechnung mit eine Rolle spielt.

Diese vom Computer erzeugten Bilder sind häufig von einer faszinierenden Schönheit und stellen in der Beziehung durchaus Kunstwerke dar. Demgegenüber wird oft eingewandt, daß die Berechnungsvorschrift vollständig determiniert und reproduzierbar ist. Das ist richtig, aber die Wahl der Parameter, die zum fertigen Bild führt, und die passende Farbgebung erfordern die Intuition. Daher stellen diese „künstlichen" Fraktale durchaus kreative Leistungen dar. Der Computer ist dabei nur das Hilfsmittel, um die Ideen des Autors gegenständlich zu realisieren. Deutlich wird diese Aussage z. B. in den Abb. 1, 9 und 10 sowie bei dem Programm FRAKFILM auf der zu diesem Beitrag erhältlichen Programmdiskette (s. u.).

Einen umfassenden Eindruck von solchen „künstlichen" Bildern bietet [2]. Zahlreiche fertige Computerprogramme zum Thema Fraktale (geschrieben in der Programmiersprache C) sind in [3] veröffentlicht. Programmbausteine (geschrieben in PASCAL) zum Selberexperimentieren mit Fraktalen finden sich in [4].

Zu den Programmen

Zum eigenen Experimentieren mit Fraktalen am Computer werden hier drei kurze Programm-Listings vorgestellt. Sie sind alle in C geschrieben, wobei für die Grafik die speziellen Funktionen des Turbo C-Compilers der Firma Borland verwendet werden. Zur Compilierung auf anderen Compilern müssen die entsprechenden Funktionen

sten Programmzeilen).

Die Programme laufen ohne Änderung auf allen Farbgrafikkarten (CGA, EGA oder VGA). Ein Coprozessor ist – wegen der teilweise doch recht großen Rechenzeit – sinnvoll. Alle Zeichen zwischen „/*" und „*/" (teilweise auch über mehrere Zeilen reichend) sind Kommentare und brauchen nicht abgetippt zu werden. Aus Platzgründen wurde in allen Programmen auf Eingabemöglichkeiten über den Bildschirm verzichtet. Die Startwerte für die Berechnungen (in den Zeilen mit „define") können mit einem Editor verändert werden; danach sind die Programme erneut zu compilieren. Auf der Programm-Service-Diskette sind u. a. auch diese 3 Listings (in einer erweiterten Form mit interaktiver Parametereingabemöglichkeit) als Quelltexte sowie als lauffähige EXE-Files vorhanden.

Listing 1: APFEL.C

Mit diesem Programm können Apfelmännchen mit maximal 16 verschiedenen Farben berechnet werden. Die angegebenen Werte für rmin, rmax, imin und imax liefern das vollständige Bild. Ausschnitte sind durch die Wahl kleinerer Werte möglich. Gleichzeitig muß dann jedoch die Iterationstiefe (Variable „Tiefe") erhöht werden, was automatisch zu einer Erhöhung der Rechenzeit führt.

Zur Farbgebung werden die Anzahl der benötigten Iterationsschritte bis zum Erreichen des Ergebnisses gezählt, und dieser Zahl wird dann eine Farbe zugeordnet. Sollte kein Ergebnis erreicht werden, wird als Farbe „schwarz" gewählt.

Listing 2: DRACHE.C

Eng mit dem Apfelmännchen verwandt ist der San-Marcos-Drachen. Hiermit kann er als „Gebirge" dargestellt werden, d. h. die Anzahl der benötigten Iterationsschritte wird durch eine unterschiedliche Höhe der Punkte über der Grundfläche kenntlich gemacht. Speziell bei Verwendung einer CGA-Grafikkarte müssen die Farben verändert werden (COLOR1 = 1, COLOR2 = 2, COLOR3 = 3).

Listing 3: PFLANZE.C

Vollständig andere Bilder können mit diesem Programm erzeugt werden. Sie haben Ähnlichkeit mit Querschnitten durch Pflanzenstengel (siehe Abb. 5a – 5b). Für a, b und c kann jeder beliebige Wert eingesetzt werden, bei großen Werten sollten die Zoomfaktoren „x_zoom" und „y_zoom" verkleinert werden. Mit „bunt" kann die Farbigkeit des Bildes verändert werden, d. h., nach soviel Punkten, wie dort angegeben, ändert sich die Zeichenfarbe.

Aus technischen Gründen werden die drei Listings in der Beilage CLB-Memory abgedruckt. Wir bitten um Beachtung.

Erläuterungen zur Programm-Service-Diskette

Als Service für alle an Fraktalen auf dem Computer interessierten Leser ist zu diesem Artikel eine Programm-Service-Diskette erhältlich. Sie enthält neben den bereits hier abgedruckten Listings noch einige andere als C-Quelltexte, dazu von allen Programmen die lauffähigen EXE-Files (für MS-DOS), einige Beispielbilder zum sofortigen Anse-

hen, für jedes Programm eine Tabelle mit interessanten Parametern, Hinweise zur Compilierung der Quelltexte (sie sind Compiler-unabhängig) sowie eine ausführliche Beschreibung jedes der einzelnen Programme. Die Disketten (wahlweise 2 Stück 5 1/4 Zoll 360 KB oder 1 Stück 3 1/2 Zoll 720 KB, bei der Bestellung bitte mit angeben) sind für einen Zeitraum von sechs Monaten nach Erscheinen dieses Beitrags für DM 35,- (incl. Porto, Verpackung, Mehrwertsteuer) bei der CLB-Redaktion (Postfach 1247, 5840 Schwerte) erhältlich. Danach werden sie mit in das Vertriebsprogramm der CLB-Software aufgenommen und sind dann für DM 70,- (zuzüglich Porto, Verpackung, Mehrwertsteuer) erhältlich.

Auf den Disketten befinden sich im einzelnen die folgenden Programme (einschließlich Quelltexte):

APFEL.C

Das als Listing hier bereits abgedruckte Apfelmännchen-Programm, erweitert um die Möglichkeit des Bildspeicherns, der Ausschnittbestimmung unter Verwendung einer Maus sowie der Möglichkeit der interaktiven Parametereingabe.

DRACHE.C

Das hier als Listing abgedruckte Programm für den 3D-San-Marcos-Drachen, erweitert um die Möglichkeit der interaktiven Parametereingabe.

PFLANZE.C

Das als Listing hier abgedruckte Programm eines Schnittes durch einen Pflanzenstengel, ebenfalls erweitert um die Möglichkeit der interaktiven Parametereingabe. Es können hierbei gleichzeitig mehrere Muster auf dem Bildschirm dargestellt werden.

BERG.C

Programm zur Berechnung fraktaler Berge (siehe Abb. 3) mit interaktiver Parametereingabe. Das Gebirge kann durch Linien oder Polygone aufgebaut werden; die einzelnen Flächen können jeweils unterschiedlich farblich modelliert werden.

STRAUCH.C

Programm zur Berechnung fraktaler Bäume, Sträucher und Wälder (siehe Abb. 4) auf dem Bildschirm und Ausgabe auf einen HP-kompatiblen Plotter.

TURTLE.C

Programm zur Erzeugung von Fraktalen mit Turtle-Grafik (Schneeflocke, Hilbertkurve etc., siehe Abb. 2). Das Programm enthält einen Turtle-Editor, mit dessen Hilfe sich einfach selbstähnliche Strukturen generieren lassen. Die Ausgabe der Grafiken kann ebenfalls auf dem Bildschirm oder auf einem HP-kompatiblen Plotter erfolgen.

FRAKFILM.C

Dies ist das Spitzenprogramm der Programmsammlung. Mit diesem Programm können, ähnlich wie mit APFEL.C, Apfelmännchen berechnet werden, die Darstellung erfolgt aber mit bis zu 256 Farben gleichzeitig. Erforderlich ist dazu jedoch eine VGA-Grafik-Karte. Nach der Berechnung können die Farben in unterschiedlichen Variationen kontinuierlich geändert werden. So ergeben sich sehr schöne Filmeffekte, da die Farben durch die komplizierten Figuren ineinanderlaufen, und schöne Standbilder (siehe Abb. 1, 9 und 10).

Literaturübersicht (Auswahl)

[1] Mandelbrot, B.B.: Die fraktale Geometrie der Natur. Akademie-Verlag Berlin, 1987.
[2] Peitgen, H.O.; Richter, P.H.: The Beauty of Fractals. Springer-Verlag, Berlin 1986.

a) Fraktale und Computer

[3] Stevens, R.T.: Fractal-Programming in C (englisch). M & T Publishing Inc., Redwood City, 1989.
[4] Becker, K.-H.; Dörfler, M.: Dynamische Systeme und Fraktale. Vieweg-Verlag, Braunschweig/Wiesbaden 1989.
[5] Vilsmeier, S.: Reise in die fraktale Faszination. 64'er, 5/1987.
[6] Vilsmeier, S.: Vorstoß ins Chaos. 64'er, 9/1987, 10/1987, 12/1987, 2/1988, 3/1988, 4/1988.
[7] Chaos und Fraktale. Spektrum der Wissenschaft Verlagsgesellschaft, Heidelberg 1989.
[8] Computer-Kurzweil. Spektrum der Wissenschaft Verlagsgesellschaft, Heidelberg 1989.
[9] DOS extra, Sonderheft Grafik, Nr. 12, 1990. DMV-Verlag.
[10] Bandt, C.; Kuschel, T.: Fraktale mittels „Igelgeometrie". Wissen. Fortschr. 38 (1989) 264.
[11] Urian, R.: Wir bauen uns ein Monster. c't, 5/1988.

b) Fraktale und Wissenschaft

[12] Avnir: The Fractal Approach to Heterogeneous Chemistry. Surfaces, Colloids, Polymers. John Wiley & Sons, 1989.
[13] Bargon, J., u.a.: Fraktale in der Physikalischen Chemie. IBM-Anwendungsbrief Nr. 16, Bonn 1987.
[14] Pfeifer, P.: CHIMIA 39 (1985) 120.
[15] Amann, A.; Gans, W.: Angew. Chem. 101 (1989) 277-285.
[16] Purugganan, M.D.: Naturwissenschaften 76 (1989) 471.
[17] Luo, L.; Tsai, L.: Chin. Phys. Lett 5 (1988) 421.
[18] Verhandlungen der Gesellschaft Deutscher Naturforscher und Ärzte, 115. Versammlung, Tagungsband: Ordnung und Chaos in der unbelebten und belebten Natur. Wissenschaftliche Verlagsgesellschaft mbH. Stuttgart 1989.
[19a] Ackermann, W.-G.; Spindler, H.: Z. phys. Chemie 269 (1988) 1000.
[19b] Ackermann, W.-G.; Spindler, H.; Kraft, M.: Z. phys. Chemie 269 (1988) 1233.
[19c] Spindler, H.; Kärger, J.: Z. phys. Chemie 270 (1989) 225.
[20] Spindler, H.: Z. Chem. 28 (1988) 421.
[21] Heidenreich, E.: Chem. Techn. 38 (1986) 65.
[22] López-Quintela, M.A.; Tojo, C.; Buján-Núñez, M.C.: Molec. Phys. 65 (1988) 1195.
[23] Mögel, H.-J.: Wiss. Z. Univ. Halle XXXVII (1988), H. 5, 20.
[24] Leuenberger, H.; Holman, L.; Usteri, M.; Winzap, S.: Pharm. Acta Helv. 1989 (2) 34.
[25] Blumen, A.; Schnörer, H.: Angew. Chem. Int. Ed. Engl. 29 (1990) 113.

Listings zu dem Aufsatz über Fraktale in diesem CLB-Heft

Listing 1: APFEL.C

```
/*****************************************************
            Das Apfelmännchen in 'C'

 Für die graphische Darstellung werden folgende
 Routinen der Grafikbibliothek von TURBO-C (Version
 1.5, 2.0, TC++) verwendet:
 initgraph    :  Einschalten des Grafikmodus
 getmaxx      :  liefert höchste X-Koordinate
 getmaxy      :  liefert höchste Y-Koordinate
 getmaxcolor  :  liefert höchste Farbnummer
 putpixel     :  zeichnet Punkt bei x,y in der Farbe
 closegraph   :  Zurück in den Textmodus

*****************************************************/

#include <graphics.h>     /* TURBO-C Grafikheader */
#include <conio.h>

/*****************************************************
  Definierte Konstanten für die Darstellung des
  Apfelmännchens. Diese Werte sind für das Aussehen
  der Figur verantwortlich. Sinnvolle Werte liegen
  für rmax/rmin und imax/imin jeweils zwischen
  -2.5 und 2.5.
  Die Iterationstiefe (tiefe) sollte nicht unter 20
  liegen. Es gilt: je höher dieser Wert, desto
  detailierter die Darstellung, aber auch um so
  länger die Rechenzeit.
*****************************************************/

#define tiefe   30 /* Maximalzahl der Iterationen */
#define rmin   -2.5   /* Wertebereich für den     */
#define rmax    1.0   /* Realteil sowie für den   */
#define imin   -1.5   /* Imaginärteil des be-     */
#define imax    1.5   /* trachteten Bereichs      */
/*****************************************************/

int main()
{
int    tc_driver, tc_mode;
int    maxcolor, xres, yres, xr, yi, v, color;
double ar, ai, r, x, y, i, xalt, xx, yy;

tc_driver = DETECT;     /* Automatische Erkennung */
                        /* des Grafiksystems      */
initgraph(&tc_driver,&tc_mode,"");
xres = getmaxx() + 1;  /* maximale Auflösung X    */
yres = getmaxy() + 1;  /* maximale Auflösung Y    */
maxcolor = getmaxcolor() + 1;  /* max. Farbzahl   */
                       /* Skalierung von          */
ar = (rmax - rmin)./ xres;   /* Wertebereich zu   */
ai = (imax - imin) / yres;   /* Bildschirm        */
v = 0;                       /* Iterationszähler  */

for(yi=0; yi<yres; yi++)       /* Schleife über Y */
  for(xr=0; xr<xres; xr++) {   /* Schleife über X */
     r = rmin + xr * ar;
     i = imin + yi * ai;
     x = y = xx = yy = 0.0;
     v = 0;

       do {                          /* Iterationen */
          xx = x * x;
          yy = y * y;
          xalt = x;
          x = xx - yy  + r;
          y = 2.0 * xalt * y + i;
       }
       while(!(++v >tiefe || (xx + yy > tiefe)));
     /* Farbe für den nächsten Punkt bestimmen */
     color = (v == tiefe) ? BLACK : (v % maxcolor);
     putpixel(xr, yi, color);

     if(kbhit()) {  /* Nach betätigen von Tasten */
     getch();       /* erfolgt Schleifenabbruch */
        yi = yres, xr = xres;
     }
  }                 /* Ende der X-Schleife     */
                    /* Ende der Y-Schleife     */
getch();            /* Erst nach Betätigen     */
closegraph();       /* einer Taste wird das    */
return 0;           /* Programm beendet.       */
}
/*****************************************************/
```

Listing3: PFLANZE.C

```
/*****************************************************
            Das Programm 'Pflanze' in 'C'

 Für die graphische Darstellung werden folgende
 Routinen der Grafikbibliothek von TURBO-C (Version
 1.5, 2.0, TC++) verwendet:
 initgraph   :  Einschalten des Grafikmodus
 getmaxx     :  liefert höchste X-Koordinate
 getmaxy     :  liefert höchste Y-Koordinate
 getmaxcolor :  liefert höchste Farbnummer
 putpixel    :  zeichnet Punkt bei x,y in der Farbe
 closegraph  :  Zurück in den Textmodus

 *****************************************************/

 #include <graphics.h>    /* TURBO-C Grafikheader */
 #include <conio.h>
 #include <math.h>

/*****************************************************
  Definierte Konstanten für die Darstellung der
  'Pflanze'. Diese Werte sind für das Aussehen
  der Figur verantwortlich.  Sinnvolle Werte liegen
  etwa zwischen -3.0 und 3.0 für a, b und c.
  Die Konstante 'bunt' dient dazu,  erst  nach  dem
  Zeichnen einer größeren Menge von Punkten einen
  Farbwechsel zuzulassen.
 *****************************************************/

 #define a       -1.0     /* Parameter für Form und */
 #define b        1.0        /* Farbe der 'Pflanze' */
 #define c       -2.0
 #define x_zoom   5.0  /* Zoomfaktoren für X und Y */
 #define y_zoom   5.0 /* für Größe der Darstellung */
 #define bunt  4000 /* Verzögerung f. Farbwechsel */
/*****************************************************/

                        /* Makro für Absolutbetrag */
 #define FABS(x)   (((x) > 0.0) ? (x) : (x) * -1.0)

/*****************************************************/

 int main()
{
 int  tc_driver, tc_mode, maxc;
 int  x_mitte, y_mitte, x_coord, y_coord, color, cl;
 double  x0, y0, x1;

 tc_driver = DETECT;     /* Automatische Erkennung */
                         /* des Grafiksystems      */
 initgraph(&tc_driver,&tc_mode,"");
 x_mitte = getmaxx() / 2;     /* Zentrierung von X */
 y_mitte = getmaxy() / 2;     /* Zentrierung von Y */
 maxc = getmaxcolor() + 1;    /* Anzahl Farben */
                              /* cl wird 'über'- */
 cl = 0;                      /* laufend hochgezählt */
 x0 = y0 = 0.0;               /* Startwerte */

 while(!kbhit()) {       /* Bildschirmkoordinaten */
    x_coord = (int)(x0 * x_zoom) + x_mitte;
    y_coord = (int)(y0 * y_zoom) + y_mitte;

    color = ((++cl / bunt)  % maxc) + 1;
    putpixel(x_coord, y_coord, color);
                         /* neue Funktionswerte */
    x1 = y0 - sqrt(FABS(b * x0 - c))
         * ((x0 < 0.0) ? -1.0 : 1.0);
    y0 = a  - x0;   x0 = x1;
 };
          /* Da eine Taste gedrückt wurde, wird */
 getch();              /* das Zeichen eingelesen, */
 closegraph(); /* der Textmodus wiederhergestellt */
 return 0;             /* und das Programm beendet. */
}
/*****************************************************/
```

Listing 2: DRACHE.C

```
/*****************************************************
        Der San Marcos-Drachen als 3D-Bild in 'C'

 Für die graphische Darstellung werden folgende
 Routinen der Grafikbibliothek von TURBO-C (Version
 1.5, 2.0, TC++) verwendet:
 initgraph   :  Einschalten des Grafikmodus
 getmaxx     :  liefert höchste X-Koordinate
 getmaxy     :  liefert höchste Y-Koordinate
 getmaxcolor :  liefert höchste Farbnummer
 line        :  zeichnet Linie von x1,y1 nach x2,y2
 closegraph  :  Zurück in den Textmodus

 *****************************************************/

 #include <graphics.h>    /* TURBO-C Grafikheader */
 #include <conio.h>

/*****************************************************
  Definierte Konstanten für die Darstellung des
  Drachens. Diese Werte sind für das Aussehen der
  Figur verantwortlich.  Sinnvolle Werte liegen für
  rmax/rmin und imax/imin zwischen -2.5 und 2.5.
  Die Iterationstiefe (tiefe) sollte nicht unter 20
  liegen, es gilt: je höher dieser Wert,  desto de-
  tailierter die Darstellung, aber auch um so länger
  die Rechenzeit.
 *****************************************************/

 #define tiefe   50     /* Max.zahl der Iterationen */
 #define hoehe  100     /* Höhe über Grundfläche    */
 #define rmin   -2.5  /* Wertebereich für den      */
 #define rmax    2.5  /* Realteil sowie für den    */
 #define imin   -1.3  /* Imaginärteil des be-      */
 #define imax    1.3  /* trachteten Bereichs       */

 #define COLOR1  6          /* Farbe Vorderseite    */
 #define COLOR2  7          /* Farbe rechte Seite   */
 #define COLOR3  14         /* Farbe Oberfläche     */

/*****************************************************/

 int main()
{
 int     tc_driver, tc_mode;
 int     m, n, k, xm, ym;
 int     u, v, ul, vl;
 double a, x, x2, y, y1, y2, xc, yc, dx, dy;

 tc_driver = DETECT;     /* Automatische Erkennung */
                         /* des Grafiksystems      */
 initgraph(&tc_driver,&tc_mode,"");
 xm = getmaxx() / 2;     /* maximale Auflösung X   */
 ym = getmaxy() / 2;     /* maximale Auflösung Y   */

 dx = (rmax - rmin)/(double)xm; /* Skalierung von */
 dy = (imax - imin)/(double)ym; /* Wertebereich   */
                                /* zu Bildschirm  */
 xc = 1.0;
 yc = 0.0;
 a  = (double)tiefe / (double)hoehe;

/* Verhältnis Rechentiefe zu Höhe der Darstellung */

 for(n=0; n<ym; n++) {          /* Schleife über Y */
       y1 = imin + n * dy;
    for(m=0; m<xm; m++) {       /* Schleife über X */
          x = rmin + m * dx;
       y = y1;
       k = 0;
          while(++k <= tiefe) {   /* Iterationen */
                x2 = x * x;
                y2 = y * y;
                y  = 2 * x * y - yc;
                x  = x2 - y2 - xc;
                if(x2 + y2 >=(double)hoehe) break;
  /* Abbruch, wenn Höhe der Darstellung erreicht */
          }
         /* Berechnung der Bildschirmkoordinaten */
          u  = m + xm/2 - n/2 ;
          ul = u + 1;
          v  = n + ym;
          vl = v - (int)((double)k / a - 1.0);

       setcolor(COLOR1);     /* Farbe vorne setzen */
       line(u,v, u,vl);          /* Linie zeichnen */
       setcolor(COLOR2);   /* Farbe rechts setzen */
       line(ul,v, ul,vl);        /* Linie zeichnen */
       setcolor(COLOR3);     /* Farbe oben setzen */
       line(u,vl, ul,vl);        /* Linie zeichnen */
       if(kbhit()) n = ym, m = xm; /* Nach Betätigen
                          einer Taste Schleifenabbruch */
    }
 }
 getch();                  /* Erst nach Betätigen  */
 closegraph();             /* einer Taste wird das */
 return 0;                 /* Programm beendet.    */
}
/*****************************************************/
```


Schluß

Hallo Club 80'er,

endlich, werdet Ihr sagen, gibt es mal wieder eine INFO! Ein langer Zeitraum war jetzt zwischen INFO 36/37 - aber das Material aus vorliegendem INFO ist alles, was zwischenzeitlich bei mir eingetrudelt ist. Das lange Warten auf versprochene Beiträge hat sich nicht gelohnt -sie fehlen bis heute. Schade!

Vielleicht ergibt sich über die BASIC-Ecke mehr Feedback. Interesse dazu wurde bisher bekundet. Auf alle Fälle möchte ich Mitte Dezember mit der Produktion des nächsten INFO's beginnen. Ich hoffe, daß bis dahin noch einige Artikel bei mir ankommen.

Gleichzeitig wurde anläßlich Clubtreffen Süd beschlossen, daß zukünftig Artikel von mindestens eineinhalb DIN A 4 Seiten, die im INFO abgedruckt, mit 5,-DM vergütet werden. Dies in der christlichen Hoffnung auf eine sich bessernde Beteiligung am Club-INFO zum Gemeinwohl aller. Entlohnt werden maximal bis zu vier Beiträge pro Jahr. Der Betrag wird den jeweiligen Autoren zum Jahresende ihrem Jahresbeitragskonto gutgeschrieben. Wir hoffen auf diese Weise auch die Portokasse der Autoren etwas zu schonen. Begonnen wird die Aktion mit den Artikeln zum 38. (nächsten) INFO.

Ich wünsche Euch eine schöne Zeit
bis zum nächsten mal

Jens

Adressen-Liste Club80

Stand vom 30-10-1992
Seite 1

Nachname	Vorname	Straße	PLZ	Ort	Telefon privat	Telefon geschäftlich
Behrendt	Detlef	Schlosserbreite 1a	W- 8018	Grafing	08092 / 9173	
Benner	Achim	Vorm Mühlberg 1	W- 5910	Kreuztal 3	02732 / 3780	(ab 18.00Uhr)
Berndt	Wolfgang	Friedberger Straße 92c	W- 6360	Friedberg 2	06031 / 2963	
Berndt-Jochum	Ilse	Stachelsgut 24	W- 5060	Bergisch-Gladbach 1	02204 / 65254	
Bernhardt	Helmut	Hafenstraße 7	W- 2305	Heikendorf	0431 / 241907	0431 / 74047
Betz	Heinrich	St. Wolfgangstraße 13	W- 8551	Hausen	09191 / 31698	09191 / 611108
Brans	Jörg	Tieloh 55	W- 2000	Hamburg 60	040 /6906531	
Braun	Harald	Postfach 8011	W- 2300	Kiel 17		
Braun	Günter W.	Postfach 800 226	W- 8000	München 80		
Böckling	Ulrich	Jochaczstraße 61	W- 5410	Hör-Grenzhausen	0264 / 4861	02631 / 895168
Chwolka	Fritz	Saarstraße 34	W- 5173	Aldenhoven	02464 / 8920	
Dose	Volker	Dorfstraße 10	W- 2304	Brodersdorf	04343 / 1357	
Gill	Thomas	Maria-Eich-Str. 34	W- 8000	München 60	089 /8349527	
Günther	Jens	Bannerscheid 7	W- 5231	Neitersen	02681 / 1553	
Halgasch	Gert	Großschonauer Straße 26	O- 8805	Jonsdorf		
Hartmann	Hans-Günther	Möwenstraße 9	W- 2876	Berne 2	04406 / 6911	0421 /2482419
Hebecker	Ulrich	Büsumerstraße 15	W- 7000	Stuttgart	0711 / 734800	
Held	Manfred	Stirner Straße 22	W- 8835	Pleinfeld	09144 / 6563	0911 /2195245
Hermann	Klaus	Forchenstraße 8	W- 7401	Pliezhausen	07127 / 71945	
Hürdler	Manfred	Niederhofer Straße 29	W- 8709	Rimper	09365 / 4235	
Johnen	Willi	Hansemannstraße 1	W- 5160	Düren	02421 / 501305	
Kauka	Dietmar	Straße des Friedens 37	A- 7201	Neukirchen	0037404/850112	
Kemmer	Jürgen	Dorfberg 7	W- 8701	Sulzdorf	09334 / 1050	
Kostya	Mary Jo	Balberstraße 68	CH- 8038	Zürich	00411 /4828948	00411 / 258170
Kuhn	Eckehard	Im Dorf 14	W- 7443	Frickenhausen	07022 / 45417	
Littmann	Claus	Plockhorst, Zum Spring 15	W- 3155	Edemissen	05372 / 7796	
Lorenz	Walter	Mahräckerstraße 9	W- 6000	Frankfurt /Main 50	069 / 531656	
Magnus	Andreas	Bismarckstraße 29	W- 4650	Gelsenkirchen	0209 / 870230	
Mahlert	Herbert	Hohenbudbergerstraße 112 A	W- 4100	Duisburg	02135 / 47217	
Mand	Harald	Kl. Flintbeker Straße 7	W- 2302	Flintbek /Kiel	04347 / 3629	0431 /3013580
Menk	Christian	Ollsener Straße 52	W- 2116	Hanstedt	04184 / 7825	
Müller	Kurt	Sophie-Scholl-Ring 3b	W- 2054	Geesthacht	04152 / 70643	04151 / 889137
Mössel	Franz	Schafferstraße 12	I- 39012	Meran		
Neebe	Gerhard	Märkische Straße 186	W- 4600	Dortmund 1	0231 / 416549	
Neueder	Jens	Rudolf-Then-Straße 32	W- 7178	Gschlachtenbretzingen	0791 / 42877	0791 /44-4722
Neumann	Christof	Zeitblomstraße 22/2	W- 7900	Ulm	0731 /6022568	0731 /9749720
Nitschke	Stefan	Germanenstraße 5	W- 7519	Walzbachtal 1	07203 / 452	
Obermann	Hartmut	Mozartring 23	W- 8870	Günzburg	08221 / 30248	
Peters	Jürgen	Heukoppel 14	W- 2000	Hamburg 14	040 /6412371	
Reit	Hermann	Vechter Hof 40	W- 4500	Osnabrück	0541 / 16331	
Retzlaff	Bernd	Kleiner Sand 98	W- 2082	Uetersen	04122 / 43551	04103 / 605310
Riechmann	Michael	Letelner Heideweg 12	W- 4950	Minden	0571 / 36627	
Rinio	Gerd	Rennbahnstraße 9	W- 2000	Hamburg 74	040 /6552630	
Ruschinski	Claus	Pommernstraße 21	W- 4370	Marl	02365 / 34646	
Schilling	Werner	Ehndorfer Straße 340	W- 2350	Neumünster	04321 / 61116	
Schimmer	Jörg	Castellring 55	W- 6369	Nidderau 1	06187 / 25503	
Schmid	Alexander	Entmannsdorf 5	W- 8640	Kronach/Gehülz		
Schmitz	Rainer	Dornierstraße 17	W- 7320	Göppingen	07161 / 22549	
Schober	Frank-Michael	Weberweg 2	O- 7590	Spremberg	0037574/ 4565	
Schoberth	Uwe	Petrus-Waldus-Straße 14	W- 7136	Oetisheim	07041 / 7254	0711 /2011455
Scholz	Hans-Werner	Spitalstraße 54	W- 4054	Nettetal		
Schroers	Horst-Dieter	Breslauer Straße 9	W- 8016	Feldkirchen	089 /9032615	089 /9032615
Schröder	Gerald	Am Schützenplatz 14	W- 2105	Seevetal 1	04105 / 2602	
Schröder	Peter	Theodor-Fahr-Straße 32	W- 2000	Hamburg 62	040 /5311582	
Schröer	Egbert	Joachimstraße 18	W- 4270	Dorsten 1	02362 / 75840	02365 / 495183
Schulte	Hartmut	Entenschnabel 8	W- 3162	Uetze	05173 / 1248	
Schut	Andre	Sanderstraße 26	W- 1000	Berlin 44	030 /6917861	
Schwarz	Wolfgang	Schwedenring 6	W- 8850	Donauwörth	0906 / 3092	
Seelmann-Eggebert	Jörg	Henri-Spaak-Straße 96	W- 5305	Alfter 4	0228 / 643853	
Soerensen	Svend A.	Bogholder Allee 76A	DK- 2720	Vanloese		
Sonnemann	Harald	In den Eckwiesen	W- 6101	Fischbachtal	06131 /320860	
Stumpferl	Stefan	Hasenberglstraße 57	W- 8000	München 45	089 /3144001	
Sörensen	Rüdiger	Thomas-Mann-Straße 3a	W- 6500	Mainz 1	06131 / 320860	06131 / 395268
Tornow	Wilhelm	Görlitzer Straße 16	W- 2190	Cuxhaven 13	04723 / 1355	04723 / 3063
Vollkmer	Richart	Am Spörkel 69	W- 4600	Dortmund 50		
Volz	Oliver	Am Ochsenwald 37A	W- 7000	Stuttgart 80 (Rohrerhöh	0711 /744051	0711 /6853013
Werner	Heiko	Reichenberger Straße 5	O- 8032	Dresden		
Wittkamp	Heinz	Hindenburgstraße 37	W- 5630	Remscheid	02191 / 75132	
Wulf	Hans-Otto	Im Brahmkamp 38	W- 4250	Bottrop	02041 / 688972	

Bitte überprüft Eure Daten
und teilt uns Änderungen mit!