GENIE USER CLUB

und Colour-Genie

USER CLUB BREMERHAVEN

CLUB-INFO
CLUB-INFO
CLUB-INFO

3108585

HACKTORY DOS 2.3 C

G-DOS 2.1b * modif. 1984 durch Arnulf Sopp

NEWDOS/80

APPARAT, INC.
VERSION 2.0
Genie II

GENIE DOS 2.1b

BAYERNDOS

Spieß, INC.
d'Nummer is 2.8
Schenie zwoa

3. JAHRG. | 10. AUSG | 1985

Red.: Peter Spieß, Trugenhofenerstr. 27, 8859 Rennertshofen 1
* Sortiert von: Edeltraud *** Auflage: 085 Exempl. *****

Inhalt

* Dieser Artikel stammt vom TRS80-User-Club München

In allerletzter Minute:

ACHTUNG: KOPIEREN UND WEITERGEBEN VON GEKAUFTER SOFTWARE IST STRAFBAR !!!

80-Zeichen-Karte f. Genie 1,2,2s TRS80M1 u. CG; 80*25, 8 Zeichensätze, voll CP/M2.2, Banker eingebaut. ~~225,-DM~~ Clubpreis: 215,-DM (mind.Abnahme 6 St. im Club). Info bei P. Spieß
CP/M2.2 ca. 385,-DM

H.-P. Schmid hat einen neuen Club!

Vor ein paar Tagen bekam ich einen Fragebogen von Herrn
Dr. med. Friedrich Lücke
Deisterallee 14 A
3250 Hameln 1,
der offenbar den Zweck verfolgte, die gemeinsame Hard- und Softbasis der Mitglieder des Hamburger TRS-80- und VG-User-Clubs herauszufinden. Merkwürdig daran war lediglich, daß ich von diesem Club nie gehört habe und höchstens in Abwesenheit und Unkenntnis zur Mitgliedschaft verurteilt worden sein kann. Daß der Club keine Beiträge erhebt, machte mich ebenfalls stutzig, denn wer spendiert das Porto und die Kopierkosten für Infos usw.? Also bat ich Herrn Dr. Lücke um Aufklärung und kriegte einen offensichtlich für viele Empfänger konzipierten Formbrief, der folgendes aufdeckte: Zwei weitere Adressaten (die geantwortet hatten; wer weiß wer noch alles!) kannten den Club überhaupt nicht. Das wirft ein deutliches Licht auf die "Mitglieder"-Verwaltung dieses Clubs. Herr Dr. Lücke weiß nicht, wer für die Mitgliederliste verantwortlich ist!!! Ist der Boß des Clubs der Große Unbekannte?

Nein, es ist ein allzu Bekannter, glaube ich: Mit dem Formbrief kam auch eine Mitgliederliste (mit einigen Streichungen, versteht sich, denn sie enthielt auch Leute wie unsereinen).

Hans-Peter Schmid ist dort Mitglied!

Die vielen Merkwürdigkeiten lassen einen interessanten Schluß zu: Sollte der bekannte Jäger von Raubkopierern, der nur so Geld für die von ihm vertriebenen Programme erlangen kann, wieder mal einen Club gegründet haben? Wenn man dann den Kreis der Mitglieder scheinbar durch ein paar Ahnungslose künstlich erweitert, erweitert man damit auch die Möglichkeiten, Geld zu verdienen. Denn irgendwer wird schon so unvorsichtig sein, Software mit ihm oder einem seiner Helfer zu tauschen.

Herr Lücke hatte vor meiner Antwort an ihn wahrscheinlich keine Ahnung, in welch prominenter Gesellschaft er sich befindet, sonst hätte er mir nicht ausgerechnet ein solches Intimum wie die Liste der Mitglieder geschickt. Er ist demnach wohl kaum zu zeihen. Ebenso wenig sind es die Mitglieder, die gleichzeitig zu unserem Club gehören. Überhaupt ist der Hamburger Club - von ihm gegründet oder vielleicht auch nicht - wohl nur ein willkommenes Werkzeug für Schmid, ansonsten aber vermutlich ein Haufen von netten Kollegen wie wir, mit Zielen wie den unseren. Zu warnen ist deshalb nicht vor Dr. Lücke, auch nicht vor dem Club schlechthin, aber davor, den Fragebogen auszufüllen, wo es um die Software geht. Und vor allem davor, mit Angehörigen dieses Clubs Programme zu tauschen, die man nicht selber geschrieben hat.

Seien wir mal ehrlich; fast jeder von uns kennt einen, der einen kennt, der schon einmal etwas raubkopiert hat. Wahrscheinlich ist Gevatter Schmid sogar formaljuristisch im Recht. Also haben wir genug Anlaß, auf den Fragebogen gar nicht erst zu reagieren. Schon gar nicht mit einem "geklauften" NEWSCRIPT, das Schmid vertreibt.

Arnulf Sopp

SONDERBEILAGE

AUF GRUND VERGANGENER VORFÄLLE HABE ICH DIESEN ARTIKEL DER WICHTIGKEIT WEGEN VORGEZOGEN. ICH BITTE UM BEACHTUNG !

Internes vom Betreuer

INTERNES VOM BETREUER

Im letzten Clubinfo hatte ich schon darauf aufmerksam gemacht, daß ich die Betreuung unseres Club's zum 01.01.1986 aus beruflichen Gründen abgeben muß. Ralf Folkerts hat sich bei mir gemeldet und würde das Amt des Betreuers weiterführen.

Anschrift und Bankverbindung des GENIE/TRS80 User-Club Bremerhaven:

Peter Spieß
Trugenhofenerstr. 27
8859 Rennertshofen 1

Raiffeisenbank Rennertshofen
BLZ: 721 697 55
Konto-Nr.: 1000 1940 2

Geburtstagsecke im Oktober

Thilo Brake
Waldemar Grundmann
Friedrich Horn
Othmar Stark
Wolfgang Frey
Wilhelm Hemme
Theodor Scharnhölz
Horst Weikamp

Die besten Glückwünsche im neuen Lebensjahr !

Als neues Clubmitglied begrüße ich Margit Schiegl. Seit sehr langer Zeit hat wieder einmal eine Dame um Aufnahme in unserer Runde gebeten.
Herzlich willkommen !

Umfrage an alle EPSON-Leute !

= * = * = * = * = * =

Für die Drucker EPSON ...80 (MX, RX usw.) kann ich bei r&r-Elektronik, Heidelberg, Farbbänder für nur DM 13.45 bekommen; allerdings erst bei Abnahme von mindestens 10 Stück. Wer interessiert ist, schreibe mir dies schnellstens (z.B. per Postkarte). Dies ist noch keine Bestellung!
Wenn mind. 10 Stück beisammen sind, werde ich die Betreffenden unterrichten. Die Sammelbestellung nehme ich vor, sobald alle Bestellungen mitsamt dem genannten Betrag + Porto (z.B. per Scheck) bei mir eingegangen sind (mit dem Vorlegen habe ich trübe Erfahrungen...)
Anm. Bestellungen für 3 Farbbänder liegen bereits vor.

= * = * = * = * = * =

Anschrift: siehe unten

Frage Nr. 1:

Von allerlei Lumpen möchte ich selten was wissen (es sei denn, ich hätte Langeweile) -
doch von den "LUMPS" möchte ich alles wissen!
Noch fand ich keine saubere Auskunft; gute Lehrbücher gibt es kaum!
WER, Ihr lieben Clubkameraden, erklärt mir hingebungsvoll und geduldig das "WIE", "WO" und "WAS" eines "LUMP" ? Vor allem seine Nutzanwendung ??
Der Kerl- nein: der/die/das LUMP ist ja nicht immer mit dem Track (der Spur) auf der Diskette identisch. Soviel habe ich immerhin schon mitbekomen.
Zeichnerische Ergänzungen der Erläuterung sind mir besonders willkommen!
* * * Der Erste, dessen Erläuterung ich nicht nur ver=* * *
* * * stehe, sondern auch begreife, erhält von mir * * *
* * * eine 5.25-Zoll-Diskette zum Nulltarif! * * *

(Anschrift s.u.)

Frage Nr. 2:

Noch einmal: V I S I C A L C !

Version VC3 besitzt gegenüber der ersten Version zusätzliche Befehle und belegt dadurch 10 KB mehr Speicher bzw. nach der Verbesserung durch Othmar Stark immer noch 4 KB mehr als vorher. (Neue Befehle u.a.: IF, AND, OR, NOT, TRUE, FALSE.)
Wer leiht mir kurzfristig das Handbuch oder auch nur die Seiten daraus, in denen diese Befehle erklärt sind oder ihre Syntax beschrieben ist, zum Kopieren? (Schonendste Behandlung wird zugesagt. Marginalien werden - auch bei scheußlichsten Umtrieben des Sprach- und Druckfehlerteufels - tunlichst ver=mieden...Portofreie Rücksendung versteht sich ohnehin !)

KaJot

Hier die Anschrift: K.-J.Mühlenbein, Am Mönchgarten 28, 6940 Weinheim

II
10185

Datenschutz

Ein paar Gedanken zum Thema Datenschutz, oder:
Wie kann man seine Briefe vor neugierigen Mitmenschen schützen ?

Die einfachste Art einen Text zu codieren, ist die, die Buchstaben z.B. in Zahlen umzuwandeln.

```
D  I  E  S     I  S  T     E  I  N     T  E  S  T  T  E  X  T
68 73 69 83    73 83 84    69 73 78    84 69 83 84 84 69 88 84
```

Wer aber weiß, wie häufig die einzelnen Buchstaben durchschnittlich in einem Text vorkommen, kann sich wohl relativ einfach ausrechnen, wie die Zuordnung lautet.

Kluge Leute sind dann auf die Idee gekommen, das ganze etwas schwieriger zu machen: man bringt die Zahlen etwas durcheinander.
Meistens wird dazu ein Schlüsselwort verwendet, dessen Buchstaben auch in Zahlen umgewandelt und dann zu den ursprünglichen Zahlen addiert werden.
Das könnte dann so aussehen:

Schlüsselwort: GENIE (71,69,78,73,69)
139 142 147 156 142 154 153 147 146 147 155 138 161 157 153 140
157 162
(hoffentlich stimmt's)

Sieht schon schon besser aus.
Das Dumme an der Sache ist nur, daß der Versatz sich nach fünf Stellen immer wiederholt. Also doch nicht so gut.
Lösung: längeres Schlüsselwort
Problem: Schreibkrampf und Gedächtnisschwäche

Woher kann man nun ein (beliebig) langes Schlüsselwort nehmen, das aber ohne große Mühe eingtippt werden kann ?

Praktischerweise ist dieses 'Wörtchen' schon in unserem Rechner eingebaut: der 'Zufallsgenerator'.
Wie die Anführungszeichen schon andeuten ist dieses Ding nämlich nicht so zufällig, wie es sein möchte. Wie ich im letzten Info gesagt habe, kann man ihn mit drei POKEs an die Kette legen. Wenn man in 16554, 16555 und 16556 feste Werte POKEd, erhält man bei RND(0) immer die gleichen Zahlen !
Damit hätten wir auch unser 'Schlüsselwort': drei Zahlen

Mit dem folgenden Programm kann man seine Texte auf der Disk meiner Meinung nach sehr wirkungsvoll codieren. Der Trick besteht darin, den Wertbereich der Buchstaben und Zahlen (32-127, Satzzeichen und Umlaute o.ä. werden also auch verdaut) auf den Bereich von 32-254 auszudehnen, wobei die Zuordnung aber nur pseudozufällig, also reproduzierbar ist.
Wenn man nach dem weiter oben beschriebenen Verfahren vorgegangen ist und das Codewort bekannt ist, kann jeder den Text leicht entschlüsseln. Mit dem 'Zufallsprinzip' ist das aber nicht oder nur sehr schwer möglich, da man weder aus den Codezahlen, noch aus dem vorangegangenen Text auf die weiteren Zahlen schließen kann (vor allem, wenn eine selbstgestrickter Zufallsgenerator verwendet wurde), wenn man nicht gerade einen Hellseher in der Verwandtschaft hat.

Das vorliegende Proramm liest einen Text (/TXT) von der Diskette ein, verschlüsselt ihn und schreibt ihn dann mit demselben Namen wieder zurück.

```
100 '    Textverschlüsselung
102 '
104 '    Alexander Schmid          18.07.85
106 '    St. Cajetan Str. 38/VII
108 '    8000 München 80
110 '
120 CLS
130 CLEAR 20000:DIM A$(500)
140 DEFINT A-Z
150 '
160 INPUT"(V)erschlüsseln oder (E)ntschlüsseln ";F$
170 IF INSTR("VvEe",F$)=0 GOTO 160
180 '
190 PRINT:INPUT"Name des Files ";N$
200 IF INSTR(N$,"/")=0 LET N$=N$+"/TXT"
      ELSE IF INSTR(N$,"TXT")=0 AND INSTR(N$,"txt")=0
      CLS:PRINT"Falscher Filetyp!"STRING$(5,7):GOTO 190
210 PRINT:PRINT"Lese File "N$" ein.":PRINT
220 '
230 Z=1
240 OPEN"I",1,N$
250 LINEINPUT#1,A$(Z):IF NOT EOF(1) LET Z=Z+1:GOTO 250
260 CLOSE
500 '
510 '    * Bearbeitung *
520 '
530 PRINT"6-stellige Codezahl XXXXXX"STRING$(6,24)CHR$(14);
540 C$="":L=0
550 A$=INKEY$:IF A$="" GOTO 550
560 IF A$<CHR$(32) AND A$<>CHR$(8) GOTO 550
570 IF A$=CHR$(8) IF L>0 LET C$=LEFT$(C$,LEN(C$)-1):L=L-1:
    PRINT"X"CHR$(24)CHR$(24);:GOTO 550 ELSE 550
580 IF A$>CHR$(31) LET L=L+1:C$=C$+A$:IF L<6 PRINT CHR$(25);:
    GOTO 550
585 PRINT CHR$(15)
590 '
600 POKE 16554,VAL(LEFT$(C$,2))
610 POKE 16555,VAL(MID$(C$,3,2))
620 POKE 16556,VAL(RIGHT$(C$,2))
630 '
640 PRINT:PRINT"Bearbeitung läuft":PRINT
650 '
660 IF F$="E" OR F$="e" GOTO 2030
1000 '
1010 '    * Verschlüsseln *
1020 '
1030 FOR M=1 TO Z
1040   FOR N=1 TO LEN(A$(M))
1050     IF LEN(A$(M))=0 GOTO 1080
1060     A=ASC(MID$(A$(M),N,1)):GOSUB 3010
1070     IF A<128 THEN MID$(A$(M),N,1)=CHR$(A+R)
1080   NEXT
1090 NEXT
1100 OPEN"O",1,N$
1110 FOR N=1 TO Z:PRINT#1,A$(N):NEXT
1120 CLOSE
1130 END
```

2
10185

```
2000 '
2010 '    Entschlüsseln
2020 '
2030 FOR M=1 TO Z
2040   FOR N=1 TO LEN(A$(M))
2050     IF LEN(A$(M))=0 GOTO 2080
2060     A=ASC(MID$(A$(M),N,1)):GOSUB 3010
2070     IF A<255 THEN MID$(A$(M),N,1)=CHR$(A-R)
2080   NEXT
2090 NEXT
2100 FOR N=1 TO Z:PRINT A$(N):NEXT
2110 OPEN"O",1,N$
2120 FOR N=1 TO Z:PRINT#1,A$(N):NEXT
2130 CLOSE
2140 END
2150 '
3000 '    Hier Zufallsgenerator (Wertbereich 0-127)
3005 '
3010 R=RND(127)
3020 RETURN
```

Es Würde mich sehr interessieren, ob es jemandem gelingt, den Text zu entschlüsseln, den ich mit auf die Diskette schreibe. (Natürlich kann es jeder auch mit einem eigenen Text versuchen)

Wer also krumme Finger hat oder gerne Rätsel löst, der soll mir eine Diskette schicken.

Viel Spaß beim Ausprobieren

Alexander Schmid

noch eine echte FREAK-Frage: (von KaJott) Anm. d. R.
Trotz vorhandener Treiber oder Hilfsprogramme (wie sie sich auch schon in der INFO fanden) schaffe ich es nicht, eine PIXEL-Grafik vom Bildschirm auf meinen EPSON RX80 zu bekommen. Er druckt stattdessen Kursivzeichen; das ist nämlich sein Zeichensatz unter ASCII 128-255. Daher bringt auch die Addition von 128 oder 64 oder so zum ASCII-Wert nichts, ebenfalls nicht 'JKL'. Was muß ich da machen? (Auf keinen Fall auf dem schwer erreichbaren Mäuseklavier spielen!)

Wie bringt man die Grafik von "TSCRIPS/GRA" zu Papier (ohne JKL) ?

auch von KaJott

Unterschied zwischen dem Kurzzeichen für REM und PRINT

Es gibt im Basic für zwei Befehle Abkürzungen.Gemeint sind PRINT (bzw. ?) und REM (bzw. ').
Man kann ohne Beeinträchtigung der Funktionsweise eines Basicprogrammes anstatt PRINT ein ? und anstatt REM ein ' eintippen.Das Programm wird immer noch einwandfrei funktionieren.
Und trotzdem gibt es einen Unterschied zwischen diesen beiden "Kurzbefehlen" (außer der Funktion) :

Gibt man bei der Programmeingabe anstatt PRINT ein Fragezeichen ein,so erscheint dennoch beim Listen des Programmes das Wort "PRINT" und nicht das eingegebene Fragezeichen.
Bei dem Hochkomma (') für REM wird hingegen nichts verändert.Wird einmal ein Hochkomma eingetippt,so bleibt ein Hochkomma im Programm stehen,bis es gelöscht oder verändert wird.Auch ein eingegebenes REM bleibt als REM erhalten.
Der Grund dafür liegt darin,daß der Interpreter diese beiden Abkürzungen unterschiedlich behandelt.
Nach der Eingabe der Programmzeile in den Buffer wird ein Unterprogramm zur Umwandlung des Textes in den Zwischencode (Befehlswörter werden in Token umgewandelt) aufgerufen.
Dort sind auch die Programmteile,die das Fragezeichen und das Hochkomma behandeln.
Ab der Adresse 1BE4H steht der Teil für das "?" :

```
CP      3FH
LD      A,B2H
JP      Z,1C5BH
```

Zuerst wird geprüft,ob überhaupt ein Fragezeichen vorliegt (Zeichen wurde vorher in den Akku geladen).Wenn ja,dann wird das Zeroflag gesetzt.Danach wird der Akku mit B2H , dem Token für PRINT,geladen.Dieses kann in jedem Fall (ob Fragezeichen oder nicht) gemacht werden,da für den Fall,daß kein Fragezeichen im Akku ist,der Akku neu belegt wird.
Nach dem Ladevorgang erfolgt ein Sprung falls,daß das Zeroflag gesetzt ist d.h. daß im Akku der ASCII-Wert des Fragezeichens ist.Das Programm verzweigt bei "zero" zu der Routine,die PRINT Token abspeichert.Dadurch,daß auch das Fragezeichen als PRINT Token abgespeichert wird,kann beim Listen nicht mehr unterschieden werden,ob nun ein "?" oder ein "PRINT" eingegeben wurde und folglich wird alles als "PRINT" gelistet.
Bei dem Hochkomma sieht die ganze Sache etwas schwieriger aus.Der Teil,der dieses Problem behandelt,steht ab Adresse 1C4AH :

```
CP      FBH
JR      NZ,1C5AH
LD      (HL),3AH
INC     HL
LD      B,93H
LD      (HL),B
INC     HL
        .
        .
JR      1C77H
```

An dieser Stelle ist schon das entsprechende Token im Akku.
Zuerst wird wieder durch einen Vergleich geprüft,ob überhaupt das Hochkommatoken im Akku ist.Falls dies nicht der Fall ist,verzweigt das Programm.Ansonsten wird erst einmal durch ein LD (HL),3A ein Doppelpunkt in den Zwischencode gebracht und der Zeiger um eins erhöht.Dann wird das Token für REM in das B-Register geladen und im Zwischencode abgespeichert.Nach dem Erhöhen des Zeigers erfolgt (einige

Befehle später) ein Sprung zur Adresse 1C77H,wo dann zuerst das Token für den Hochpfeil (!) und dann der nachfolgende Text im Zwischencode abgelegt wird.
D.h. Ein Hochkomma als REM-Ersatz wird als ein Doppelpunkt gefolgt von einem REM Token gefolgt von einem Hochpfeil Token abgespeichert.Folglich muß in einem Basicprogramm vor einem Hochkomma-Kommentar kein Doppelpunkt stehen (er darf aber).Bei der Ausgabe jedoch wird ein eingegebenes Hochkomma als Hochkomma ausgegeben.Denn beim Listen wird das vorhergehende ":REM" (vergleiche auch LIST-Routine ab 2B2EH) aus dem Buffer für die Ausgabe erst gelöscht, bevor das Hochkomma hineingeschrieben wird.
Als Beispiel ist noch der Hexdump des folgenden Basic-programmes abgebildet :

```
10 REM Kommentar 1
20 ' Kommentar 2
30 PRINT"Ende"'Kommentar 3
```

Jörg Seelmann-Eggebert

Token für REM

```
#D6A46
6A46: 58 6A 0A 00 93 20 4B 6F 6D 6D 65 6E 74 61 72 20
6A56: 31 00 6C 6A 14 00 3A 93 FB 20 4B 6F 6D 6D 65 6E
6A66: 74 61 72 20 32 00 86 6A 1E 00 B2 22 45 6E 64 65
6A76: 22 3A 93 FB 4B 6F 6D 6D 65 6E 74 61 72 20 33 00
6A86: 00 00 04 3C ED 5B 2D 40 D5 2A 2D 40 EB B7 ED 52
6A96: 7D 32 52 40 E7 ED 43 00 3C 3E 5D 32 02 3C D1 3E
6AA6: 0C 01 00 8F CD 09 5D EB 11 8E 6A D5 F5 3A 1C 40
6AB6: FE 04 11 DF 6A 28 03 11 C7 6A F1 CD 26 5A C3 9B
6AC6: 6E 50 8C 6F 47 B8 6F 4C 15 6F 2B FC 6F 2D 01 70
```

Bytefolge für ' (Hochkomma) ist unterstrichen

DISKETTENANGEBOT: MARKENDISKETTE DISKY 1D STÜCK 2,90 DM
BESTELLUNGEN BEI PETER SPIEß

NICK BINNS SUCHT DRINGEN EIN DISK-LIBRARY PROGRAMM.
WER WEIß ÜBER VISICALC BESCHEID UND KANN
BEI PROBLEMEN WEITERHELFEN ?

Experten vom Thema Computer-Viren infiziert

Eine Woche später, am 5. Juli, erscheint die erste Folge der Artikelserie des Fachmanns aus der Deutschen Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (DFVLR, Oberpfaffenhofen) in der noch jungen Sicherheitspublikation KES, die es aufgrund ihres als sensationell empfundenen Inhalts schlagartig auf einige hundert Abonnenten bringt. Als Autor zeichnete Dierstein jedoch noch nicht; schließlich zählt die Versuchsanstalt nahe Münchens auch und gerade das Bundesverteidigungsministerium zu ihren besten Kunden.

KES-Verleger Peter Hohl berichtet von ersten Resonanzen: „Manche haben sofort kapiert; andere stehen auf dem Standpunkt, das kann doch alles überhaupt nicht wahr sein." Der Blattmacher kommt sich vor „wie einer, der vor zehn Jahren vor dem Waldsterben gewarnt hat".

In der Tat, Warnungen scheinen am Platze. Nicht nur die Geheimhaltungsbemühungen der staatlichen Stellen sprechen dafür, auch der lokkere Plauderton, mit dem junge Informatiker behaupten, für Experten sei es nicht schwierig, „Viren" zu implantieren. Jeder Systemzugang sei als Infektionsstelle denkbar. Ferner, das hat Franz Peter Heider von der GEI, Bonn, ausprobiert, „ist es einfach, sich selbst reproduzierende Programme zu schreiben", also Viren zu erzeugen. Der Chiffrier-Experte wundert sich darüber, daß sich Informatiker über das Virus-Phänomen wundern. Mit zwei Sätzen faßt er die ganze Aussichtslosigkeit zusammen, Virus-Programme auf eine einzige Art und Weise in den Griff zu bekommen: „Jeder Informatiker müßte eigentlich wissen, daß man kein Programm zur Prüfung von Programmen universeller Art machen kann. Aber genau das muß man machen, um Viren zu entdecken."

Bekanntschaft mit den virulenten Software-Partikeln, die vom Mikrocode bis zum Betriebssystem und zur Anwendungsebene ihr Unwesen treiben können, machten die ersten deutschen Fachleute nach eigenem Bekunden jedoch mehrheitlich erst auf der National Computer Security Conference im vergangenen September in den USA. Erstmals klar wurde hier offenbar die ganz konkrete Gefahr „diesseits der theoretischen Möglichkeiten". Selbst die „Graumänner" vom Verfassungsschutz seien überrascht gewesen.

Nachdem das Thema nun auf dem Tisch ist, mußte sich auch die Gesellschaft für Mathematik und Datenverarbeitung (GMD) zu einem Statement bereit finden. Auf Anfrage der COMPUTERWOCHE, welche Gegenmaßnahmen denn die GMD der öffentlichen Verwaltung, als deren Forschungs- und Dienstleistungsunternehmen die Gesellschaft in St. Augustin vor den Toren Bonns ja auch zu betrachten ist, empfehlen würde, erhielten wir folgende Antwort: „Die GMD betrachtet diese und ähnliche Anfälligkeiten von Computersystemen seit langem und ist sich der Gefahren bewußt, die insbesondere durch die zunehmende Vernetzung akut werden können. Die GMD weist die Hersteller ihrer eigenen Rechner (Siemens und IBM) immer wieder auf erkannte Sicherheitsschwachstellen hin, denn in puncto Sicherheit sind in erster Linie die Hersteller von Rechnern und Betriebssystemen gefordert. Einzellösungen werden von Herstellern mit Entzug der Wartungsverpflichtungen geahndet; darüber hinaus sind Einzellösungen bei den jeweiligen Anwendern wirtschaftlich nicht vertretbar."

Dies bedeutet, daß vorerst nur eine Flucht in noch mehr Sicherheitsmaßnahmen organisatorischer Art möglich ist. Am Rechner oder am Betriebssystem „drehen" geht nicht.

Denn in diesen Basisbereichen der heutigen Computerei hätten die Entwicklungslinien schon vor Jahren anders verlaufen müssen, erklärt ein Fachmann für Software-Engineering und Rechnerarchitektur: „Man hat sich in den geläufigen Rechnerarchitekturen auf das untere Muß geeinigt. Eine Trennung von Programmen und zu verarbeitenden Daten ist in diesen schlampigen Architekturen nicht vorgesehen." Auch vermutet der Sachverständige, der nicht genannt sein will, daß bestimmte Programmiersprachen, die Springmanipulationen ermöglichen, besonders anfällig für Viren sind, „C zum Beispiel".

*KES — Zeitschrift für Kommunikations- und EDV-Sicherheit, Peter Hohl Verlag, 6507 Ingelheim.

„Als Computer-Virus wird im folgenden ein Programm bezeichnet, das die Eigenschaft hat, andere Programme zu infizieren, das heißt, immer dann eine Kopie von sich selbst in ein anderes Programm einzufügen, wenn dieses Programm bisher noch keine Kopie des Virus enthielt, also noch nicht infiziert war. Jedes Programm, das einmal infiziert wurde, kann sofort wieder als Virus auftreten und andere, noch ‚keimfreie' Programme infizieren. Die Infektion breitet sich, genauso wie bei den biologischen Viren, lawinenartig in einem DV-System oder einem Netz aus. Dies ist die erste entscheidende Eigenschaft der Computer-Viren."
Quelle: KES 3/85, Hohl-Verlag

gefunden von
H. Thönnißen

6 1018

Mini-RAM-Floppy im "sicheren Plätzchen"

Helmut Bernhardt stellt in c't 5/85 unter dem Titel "Ein sicheres Plätzchen" eine Schaltung vor, die im Adreßbereich 3900-3BFF RAM verfügbar macht (diese wie auch fast alle folgenden Zahlenangaben in Hex). Dort liegen ursprünglich die oberen nicht genutzten Adressen der Tastatur. Der Autor schlägt vor, dort Maschinenprogramme unterzubringen. Da alle gängigen Anwenderprogramme im Adreßraum ab 4300 (Level 2) bzw. 5200 (DOS) residieren, liegt es nahe, im neu gewonnenen RAM allgemeine Systemerweiterungen unterzubringen. Hier soll eine Methode vorgestellt werden, nach der NEWDOS-80 und seine Abkömmlinge G-DOS und H-DOS mit wertvollen zusätzlichen Möglichkeiten ausgestattet werden können.

Eine DOS-Anforderung, d. h. das Laden und Anspringen eines SYS-Files, wird über die RST-28-Routine abgewickelt. Dazu muß ein Code im Akku stehen, dessen binäre Bitkonfiguration darüber entscheidet, welche Systemdatei geladen und welche Routine innerhalb dieses Files angesprungen wird. Dieser Code muß mindestens 20 (hex, wohlgemerkt) betragen, andernfalls kehrt RST 28 unverrichteter Dinge zurück. In der Praxis kommt aber nur ein einziger Fall vor (abgesehen von Programmierfehlern), in dem der Requestcode kleiner als 20 ist: Wird im ROM-Tastaturtreiber die BREAK-Taste erkannt, dann wird RST 28 mit 01 im Akku angesprungen (und ohne Wirkung sofort wieder verlassen, wie gesagt).

Bei diesem Requestcode entscheiden die drei unteren Bits 0-2 darüber, in welchem Sektor des Inhaltsverzeichnisses der Systemdiskette die SYS-Datei zu suchen ist. Die Bits 3 und 4 geben an, welches der vier dort eingetragenen Systemfiles gemeint ist. BOOT/SYS (oder GDOS/SYS bzw. HDOS/SYS) wird im Prinzip nur nach dem Einschalten aufgerufen. Sein Requestcode entspräche dem Bitmuster xxx-00-000. Das bedeutet, daß im "nullten" Dateieintrags-Sektor die "nullte" Datei gemeint ist, also im Sektor 02 des Inhaltsverzeichnisses der 1. Eintrag. Ein Aufruf des Bootfiles mit RST 28 kommt nicht vor. Die acht möglichen Requestcodes mit dem Muster xxx00000 stehen deshalb für unsere Zwecke zur Verfügung.

Es wird noch wesentlich mehr: Die Bedingung, daß der Code im Akku mindestens 20 betragen muß, verringert die theoretisch möglichen 256 (diesmal dez) Codes glatt auf die Hälfte, was wir nicht hinnehmen müssen. Stattdessen kann man auf den BREAK-Code 01 testen und bei Übereinstimmung zurückspringen, um nicht bei jedem BREAK die Floppy in Gang zu setzen. Anschließend kann geprüft werden, ob der Requestcode höchstens 1F beträgt. Falls nein, handelt es sich um eine Anforderung a la Apparat, Inc. bzw. TCS. Dann geht es eben in der alten Routine im DOS-Kern weiter. Andernfalls ist es ein ehemals wirkungsloser Code bis 1F, mit dem der Anwender nun etwas anstellen kann.

Außerdem ist die Tatsache interessant, daß bei einem RST 28 alle Register zunächst unverändert in der Bearbeitungsroutine ankommen. So können beliebige Parameter an eine selbstgeschriebene DOS-Erweiterung übergeben werden. NEWDOS-80 (G-DOS, H-DOS) macht sich das zunutze, indem es dem Register C bei den meisten Library-Befehlen eine Zeigerfunktion zuordnet.

Um die oben skizzierten neuen Möglichkeiten auszunutzen, muß man wissen, wie RST 28 arbeitet. Der Einsprung ist natürlich bei 0028 in der "zero-page", der "Seite 0", also im Bereich der ersten 256 Bytes des ROM. Dort steht ein Vektor nach 400C, wo wiederum nach 4BC2 weiterverzweigt wird. An der Adresse 4BC2 wird der Stapelzeiger SP (stack pointer) zweimal inkrementiert. Die Wirkung ist, daß die RET-Adresse sozusagen vom Stack verschwindet. Dadurch verliert der RST 28 im Gegensatz zu den anderen RSTs seinen CALL-Charakter. Es wird ein gewöhnlicher JP daraus. Anders als bei einem Unterprogrammaufruf geht deshalb die Kontrolle endgültig an die angesprungene Routine über. Unter welchen Umständen sie dennoch mit einem RET verlassen werden kann, soll später erläutert wer-

cen.

Nach dem Quasi-Löschen der RET-Adresse folgt die oben angesprochene Prüfung auf 20. Wenn der Requestcode kleiner ist, geht das Carry-Flag auf 1 und es erfolgt ein Sprung nach 4312, von dort nach 45B0, wo nur der Akku auf 00 gesetzt und aus der RST-28-Routine zurückgekehrt wird. Dieser bedingte Sprungbefehl JP C,4312h kann nun leicht durch einen Sprung in die eigene Routine im Bereich 3900-3BFF ersetzt werden (wer das "sichere Plätzchen" nicht hat, kann natürlich auch sonstwohin springen). Ein Teil dessen, was dort bei H-DOS passiert, geht aus dem Listing am Ende dieses Artikels hervor:

Im residenten Teil von SYS0 ist an der Adresse 4BC6 der bedingte Sprungbefehl durch einen JP rst28 ersetzt. Dieses Label steht für die Adresse 3A5B. Dort wird geprüft, ob die BREAK-Taste mit 01 im Akku den RST verursachte (s. o.). Bei Übereinstimmung wird der Sprung nach 4312 nachgeholt. Sonst wird getestet, ob eines der beiden Bitmuster 000xxxxx oder xxx00000 vorliegt. Falls ja, ist unser neues RAM-SYS-File zuständig und wird angesprungen. Andernfalls geht im DOS-Kern bei 4BC9 die Bearbeitung wie gewohnt weiter. Auf diese Weise sind 39 zusätzliche Requestcodes möglich. Wie aus dem Listing hervorgeht, wird davon bisher nur ein Teil genutzt. Zukünftige Erweiterungen werden diesen Vorrat nach und nach verkleinern.

Die Tatsache, daß der RST 28 wie ein JP behandelt wird, hat gute Gründe: Bei vielen Systemdateien ist ein Rücksprung zum Caller nicht sinnvoll, manchmal, etwa beim Auftreten eines I/O-Fehlers, u. U. sogar fatal. Gleichwohl kann eine SYS-Datei wie ein Unterprogramm aufgerufen werden. Dazu ist es lediglich nötig, den RST 28 nicht direkt zu programmieren, sondern stattdessen einen CALL an eine Adresse, wo ein RST 28 steht. Das ist z. B. bei 4402 der Fall oder auch im ROM bei 0456, wo BREAK den RST aufruft. Ein RST ist für den Z80 ein CALL. Wenn man nach dem angegebenen Muster einen CALL callt, befindet man sich deshalb bereits in der zweiten Unterprogrammebene. Das zweimalige Inkrementieren des Stackpointers am Beginn der RST-28-Routine geht eine Ebene höher, so daß bei einem RET nun die richtige Rückkehradresse gefunden wird.

So erklärt sich das RET in Zeile 164 des Listings. Mehr möchte ich zum gelisteten Teil des Programms nicht sagen, denn es geht hier nur um die Methode, RST 28 für eigene Anwendungen nutzbar zu machen. Die hier nicht interessierenden Teile des Programms sind durch LIST OFF ausgespart. Wer Interesse daran hat, kann einen großen Teil davon aus dem darüberstehenden Sektordump rekonstruieren. Das hier Gelistete ist darin unterstrichen. Es handelt sich um den relativen Sektor 10h von SYS0/SYS. Ursprünglich hat SYS0 nur 15 Sektoren, wurde aber für die Erweiterungen mit APPEND um weitere 5 Sektoren verlängert.

Arnulf Sopp

```
001000: 0102 003A FF10 FE41 3CD3 FF10 FE0D 4110 ...:...A<.....A.
001010: F0F1 C93A 8038 FE01 200A 3A40 38FE 1020 ...:.8.. .:@8..
001020: C33A 1038 E6FE 28BC C31C 3BCD 5444 280C .:.8..(...;.TD(.
001030: 7EE6 DFFE 4ACA E73A FE4E 2804 3E2F 185E ~...J..:.N(.>/.^
001040: 2153 4822 0845 3E3A 32D3 4521 1038 22D4 !SH".E>:2.E!.8".
001050: 453E DA32 C64B 2112 4322 C74B AF18 40FE E>.2.K!.C".K..@.
001060: 01CA 1243 F5E6 1F28 0AF1 F5E6 E028 04F1 ...C...(.....(..
001070: C3C9 4BF1 FEE0 2007 3E01 D3F0 3EFD EFE5 ..K... .>...>...
001080: 05C5 FE60 CA06 3BF3 FE40 2838 FE80 285B ...`..;..@(8..([
001090: FEA0 2897 FEC0 28A8 FE0F 3008 3E37 B7C1 ..(...(...0.>7..
0010A0: D1E1 FBC9 CB67 2004 0EF0 1807 0EDF 3E0A .....g .......>.
0010B0: ED79 3CED 7921 0030 7E2F 77BE 2F77 ED78 .y<.y!.0~/w./w.x
0010C0: 3E08 18DB 3E08 D3DF 3CD3 DF3C 3CD3 DF21 >...>...<..<<..!
0010D0: 0000 545D 7E2F 77BE 2804 3E08 18C0 AFD3 ..T]~/w.(.>.....
0010E0: DF01 E037 EDB0 DBDF AF18 B421 F439 2208 ...7.......!.9".
0010F0: 453E CD32 D345 210F 3A22 D445 3EC3 32C6 E>.2.E!.:".E>.2.
```

```
4BC6                    00001           ORG     4bc6h           ;RST-28h-Routine
4BC6  C35B3A            00002           JP      rst28           ;umleiten
                        00003
                        00114 ;Ansprung DOS-Request (RST 28h):
3A5B  FE01              00115 rst28     CP      01h             ;RST 28h nach BREAK?
3A5D  CA1243            00116           JP      Z,4312h         ;falls ja
3A60  F5                00117           PUSH    AF              ;Requestcode retten
3A61  E61F              00118           AND     1fh             ;Requestbits isolieren
3A63  280A              00119           JR      Z,ramsys        ;falls Code xxx00000b
3A65  F1                00120           POP     AF              ;sonst Requestcode rest.
3A66  F5                00121           PUSH    AF              ;und wieder retten
3A67  E6E0              00122           AND     0e0h            ;Requestbits ausmaskieren
3A69  2804              00123           JR      Z,ramsys        ;falls Code 000xxxxxb
3A6B  F1                00124           POP     AF              ;sonst Code restaurieren
3A6C  C3C94B            00125           JP      4bc9h           ;und RST 28h a la G-DOS
                        00126
                        00127 ;Ansprung hier, falls RAM-SYS-File zuständig:
3A6F  F1                00128 ramsys    POP     AF              ;Requestcode restaurieren
3A70  FEE0              00129           CP      0e0h            ;V24?
3A72  2007              00130           JR      NZ,nov24        ;falls nein
                        00131
3A7B  E5                00140 nov24     PUSH    HL              ;Register retten
3A7C  D5                00141           PUSH    DE
3A7D  C5                00142           PUSH    BC
3A7E  FE60              00143           CP      60h             ;HRG-Speicher löschen?
3A80  CA063B            00144           JP      Z,hrgcls        ;falls ja
3A83  F3                00145           DI                      ;bloß keine Störungen!
3A84  FE40              00146           CP      40h             ;ROM -> RAM kopieren?
3A86  2838              00147           JR      Z,copy          ;falls ja
3A88  FE80              00148           CP      80h             ;INI,J?
3A8A  285B              00149           JR      Z,inij          ;falls ja
3A8C  FEA0              00150           CP      0a0h            ;INI?
3A8E  2897              00151           JR      Z,ini           ;falls ja
3A90  FEC0              00152           CP      0c0h            ;INI,N?
3A92  28A8              00153           JR      Z,inin          ;falls ja
3A94  FE0F              00154           CP      0fh             ;auf Bank-RAM testen?
3A96  3008              00155           JR      NC,ramtest      ;falls ja
                        00156
                        00157 ;raus mit oder ohne Fehlercode:
3A98  3E37              00158           LD      A,37h           ;Code für DOS-Fehler
3A9A  B7                00159 error     OR      A               ;Fehlerflag NZ
3A9B  C1                00160 exit      POP     BC              ;Register restaurieren
3A9C  D1                00161           POP     DE
3A9D  E1                00162           POP     HL
3A9E  FB                00163           EI                      ;INTs wieder zulassen
3A9F  C9                00164           RET                     ;und raus

00000 Fehler
```

9 10185

HORST WEIKAMP DL9YAP
4290 Bocholt Fontanestr.77 Tel.: 02871/12835
QTH-Loc: JO31HU DoK:N17 Datum: 06.10.85

HORST WEIKAMP FONTANESTR. 77 4290 BOCHOLT
An Herrn
Peter Spieß
Trugenhofenerstraße 27

8859 Rennertshofen 1

Liebe Clubmitglieder:

Ich freue mich nun auch Mitglied sein zu können und möchte mich kurz vorstellen:
Ich bin 42 Jahre alt, wohl einer der ältesten nehme ich an, so eine Art Computer-Opa.
Meinen Rechner habe ich seit Oktober 1980
Der Rechner ist ein TRS 80 mit 48K zwei Laufwerken SS und DS/DD HRG 1b 3. 5 Mhz Funkfernschreibinterface und einem FX80/Graftrax.
Bis auf den Drucker alles in einem Modell III Gehäuse.

Wie aus dem Briefkopf zu entnehmen ist, betreibe ich auch noch Amateurfunk auf KW und UKW (Funkdatenübertragung) auf beiden Bändern.
Eine Computerrunde findet Donnerstags um 19. 30 Uhr Mez auf 3750 Khz statt.

Bekannte habe ich schon seit längerem hier im Club.
Das ich nicht früher dem Club beigetreten bin liegt daran, daß ich schon lange, daß heißt seit 1982 Kontakt zum TRS 80 Club München habe.
Ich hoffe hier und da auch mal eine Kleinigkeit zur Clubinfo beisteuern zu können, so hier eine Anmerkung zum Newdos 80 XX Zaps.

10
10/85

NEWDOS 80

Das Newdos 80 V 2. 0 mit XX Zaps ist wohl jedem von uns ein Begriff.
Da gibt es Versionen mit vielen Zaps, andere mit wenigen.
Welche Version ist es nun die man nehmen sollte???
Na ja so einfach ist das nicht, daß man das Dos mit den meisten Zaps nimmt und hat seine Ruhe!
Da gibt es Betriebssysteme, die können zwar den Line Befehl, aber Lineinput ist ihnen fremd.
Der Name Befehl zum Listen ist sehr schön, aber manches Betriebssyste ist so eingestellt daß ein Editieren kaum noch möglich ist.
Da rufe ich Route auf und nichts geht, ein anderes addiert die Tabulatormarken statt an der angegebenen Stelle zu drucken.
Noch ein anderes sucht garnicht erst auf den anderen angesclossenen und innitialisierten Laufwerken wenn der File nicht auf Laufwerk 0 gefunden wurde, oder wenn man ein Copy von der Disk machen will ist plötzlich der Gat Sector zu klein und keiner weiss warum und so weiter und so weiter!!!!!!!
Gdos ??? na ja sehr schön aber hier geht das Chaos weiter!!
Da gefallen mir zwar die deutschen Fehlermeldungen, aber fast alle Eingaben sind geändert worden, so daß ich wieder neu lernen soll ??

Nun ja Ihr geplagten Freaks (ich denke diese Erscheinungen sind nicht nur mir bekannt)
versucht es mal mit einem anderen Dos wenns mal nicht klappt!!
Ich habe mir ein Dos gebastelt welches zwar lange nicht alle Zaps enthält, aber deutsche Texte, englische (Original) Eingaben und so nützliche Sachen wie SYS29 oder die HRG JKL Routine vom Arnulf allerding als JKLHRG/SYS auf dem Platz vom SYS28 weil SYS22 schon vom HRG Pack belegt wird.
Gerade hier ist die Möglichkeit der HRGJKL Routine sehr schön.
Nun ja auch wieder ein gefummeltes Dos, richtig! aber für meine Anwendungen ganz optimal.

Ich muß aber noch sagen daß ich im HDOS vom Arnulf noch keine Macke entdeckt habe, ausser daß es sich hier um GDOS handelt und ich will mich nun mal nicht umgewöhnen.

Solltet Ihr auch mal Fehler feststellen so denkt auch mal daran,

daß es sich um euer Dos handeln könnte und der Fehler nicht in der eigenen Software zu finden ist.
Sollte noch jemand unter euch Fehler im Dos kennen die hier nicht aufgeführt sind, dann verfaßt Ihr vieleicht auch mal einen Brief für Clubinfo und ich werde dann mein Dos hierhingehend auch mal untersuchen. Es wird sicher auch noch die eine oder andere Macke haben die ich noch nicht kenne.

Ein Trick sei hier noch erwähnt ! !
Wenn man ein Copy vom Betriebssystem machen will und eine andere Dichte wählen möchte, z. b. von DD auf SD oder von Doppelkopf auf einseitig, dann schreibt man
copy, 0, 1, , cbf, fmt, /sys
so wird das Bertr. syst. in jede beliebige Confirgation gebracht.
, fmt ist wichtig dabei, anschließend Basic/cmd zu Fuß rüberholen.

mit freundlichen Grüßen
Horst Weikamp
H. Weikamp

Mein EG 64 MBA ist zu haben! Noch immer schwärme ich von ihm, es fehlt ihm auch nicht das geringste, aber ein Kumpel hat einen entwickelt, der noch mehr kann. Wer so gerne wie ich im Computer herumlötet, mag auf die Veröffentlichung der Schaltung in unserer Clubpostille warten, wer aber lieber die Finger davon läßt, kann meinen für DM 100,- (VB) kriegen (Neupreis DM 195,-, neuerdings soll er wohl DM 150,- kosten). An meinem hängt eine zusätzliche I/O-Platine dran, die alle Leitungen (inkl. IORD und IOWR) enthält, die man für portgesteuerte Peripherie braucht. Es lassen sich damit gleichzeitig ca. 30 Geräte über lauter verschiedene Ports ansteuern. Die Platine ist fest mit dem MBA verbunden und natürlich im Preis inbegriffen.

Ein Freund bietet einen Typenraddrucker TP-II von Smith-Corona für DM 600,- an. Die Daten:
12 Z/s, 10 Z/inch, 105 Z/Zeile
unidirektional, max. Papierbreite 33 cm
Schnittstellen parallel Centronics, seriell RS232C, Puffer 256 Bytes
Wer Interesse hat, wende sich bitte an
Frank Helferich, Schneidemühler Str. 20b, 7500 Karlsruhe 1 (0721-628828).

Arnulf Sopp

12 10/85

Selbstbaubanker

Angeregt durch den c't-Banker, der leider einige Nachteile hat (siehe Clubinfo 10/84 S. 3-6), hatte ich mich schon vor längerer Zeit entschlossen, einen eigenen Banker zu entwickeln. Dieses Projekt ist nun abgeschlossen, und ein Prototyp funktioniert seit einigen Tagen problemlos.
Aufgabe der Schaltung ist es, die untersten acht 2k-Blöcke unabhängig voneinander zwischen RAM und ROM I/O hin und her zu schalten.

Die Schaltung arbeitet wie folgt:
Der Decoder Z2 gewinnt aus den Signalen A11-A15 die acht Select-Signale für die untersten 2k-Blöcke. Diese werden invertiert (Z2/Z3). Die invertierten Signale und die acht Ausgänge des Latches Z1, die angeben, welche Blöcke gebankt werden sollen, werden NAND verknüpft (Z5/Z6). Die Open-Collector-Ausgänge der NAND-Gatter werden Wire-OR verknüpft, um das PHANTOM-Signal für das Genie zu bekommen. Das Latch (Z1) ist auf der Eingangsseite mit dem Z80-Datenbus verbunden und wird mit Hilfe des Decoders Z7 und der OR-Gatter Z8 selektiert. Um einen Normalbetrieb nach dem Reset zu ermöglichen, ist der CLEAR-Eingang des Latches mit der Leitung CPU-RESET verbunden.
Je nachdem welche Brücke von Z7 nach Z8 Pin 9 benutzt wird, ergeben sich folgende Portadressen für das Latch:

Y0: 40H	Y1: 48H	Y2: 50H	Y3: 58H
Y4: 60H	Y5: 68H	Y6: 70H	Y7: 78H

Bedeutung der einzelnen Bits im Datenlatch:

Bit nicht gesetzt (=0)	ROM I/O eingeschaltet
Bit gesetzt (=1)	RAM eingeschaltet

Bit 0 ist zuständig für den Bereich 0000-07FFH
.....
Bit 8 ist zuständig für den Bereich 3800-3FFFH

Aufbau der Schaltung:
Ich habe mir die Schaltung auf einer Lochrasterplatine in Fädeltechnik aufgebaut. Da ich an meinem Rechner eine zusätzliche Busplatine angeschlossen habe, mußte ich den Prototyp nur dort einstecken. Wer eine solche Busplatine nicht besitzt, kommt nicht daran vorbei, sich die benötigten Signale auf der CPU-Platine zu suchen und diese dann einzeln mit der Platine zu verbinden.

Falls jemand noch Probleme beim Verständnis oder dem Aufbau der Schaltung hat, kann er sich gerne an mich wenden.

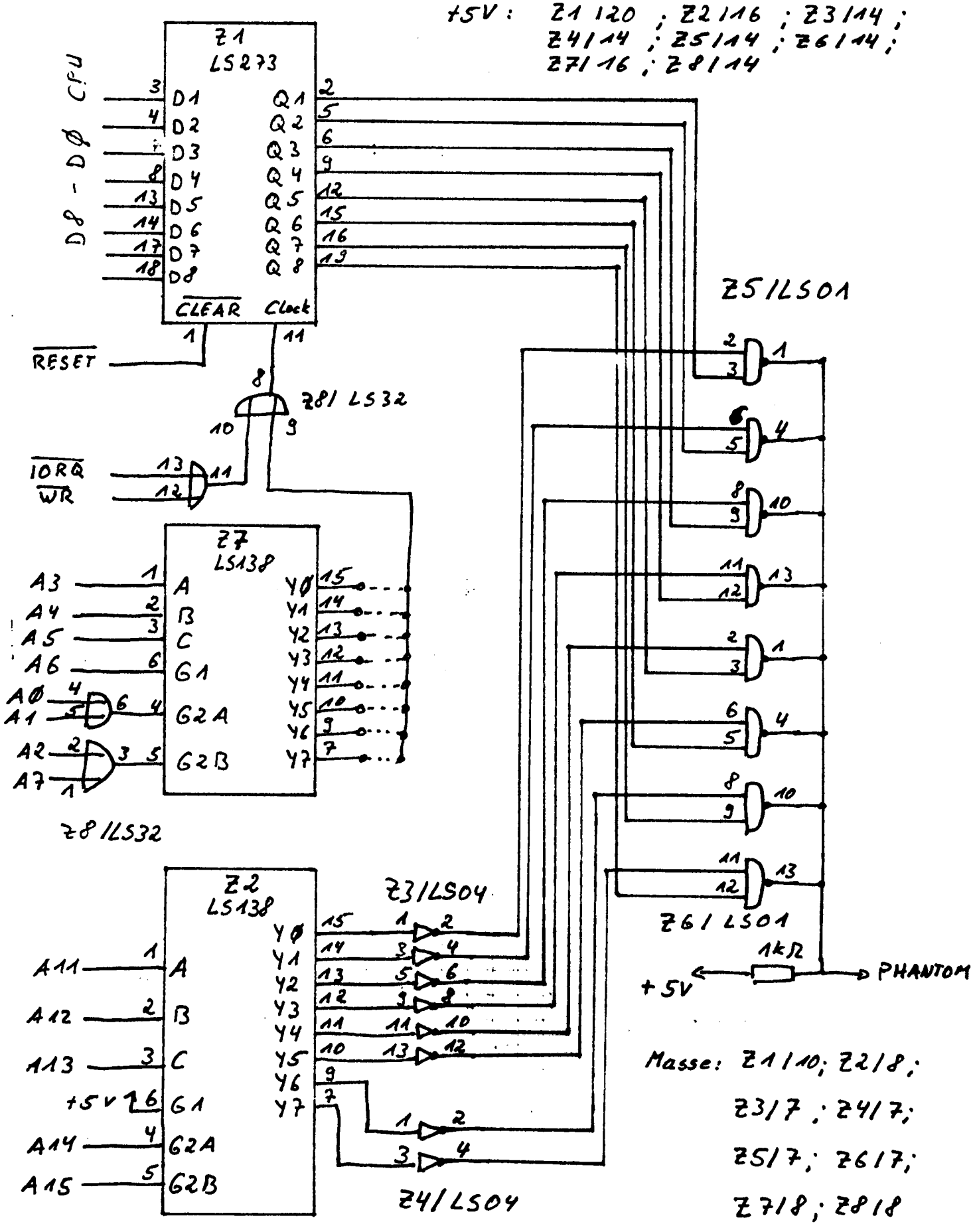
Olaf Thun

Stichwörter für KJ: Banker, Hard, Schaltplan

Schaltplan

zum Selbstbau banker

+5V: Z1 12Ø ; Z2 16 ; Z3 14 ;
Z4 14 ; Z5 14 ; Z6 14 ;
Z7 16 ; Z8 14



14 101 85

Wie werden Basic-Programme abgespeichert?
(und Basic-List vom DOS)

Jeder kennt bestimmt schon das Problem: Man ruft DIR auf, und nun steht da, unter anderem, ein komischer Name wie z.B. ALGOZXY/BAS. Was zum Teufel war das nochmal für ein Programm? Ich tippe eifrig vom DOS aus 'LIST ALGOZXY/BAS' und schon kommt das gewünschte Ergebnis. Oder?

```
Yjr.. AU1 = 10 L 5.ajö.2 A.xj... Z$Uy(z(G,A,1),2)..j..2 G,Z$..j... A..
```

Nanu? Das ist doch kein Basic, oder? Um es zu verstehen, müßen wir uns etwas ins Basic vertiefen, nicht ins Programmieren, sondern in wie das Basic die Programme abspeichert.
Tipt nun volgendes Programm ein:

```
1010 FOR A=1 TO 10 STEP 5
1020 PRINT A
1030 LET Z$=RIGHT$(MID$(TIME$,A,1),2)
1035 PRINT TIME$,Z$
1040 NEXT A
```

Dieses Programm hat überhaupt keinen Sinn, es geht mir nur zu überprüfen wieviel Speicher es belegt. Laut Rechnung sind es 103 Buchstaben (inklusive Space). Das heißt, es müßten mindenstens 103 Bytes, wahrscheinlich mehr, verwendet werden. Ich tippe nun MEM ein und ich bekomme 38194 (Ohne RUN!!!). Wenn ich nun NEW und dann MEM aufrufe ergibt sich eine Diferenz von 69 Bytes!!! Das ist fast die Hälfte!!! Wir können aber Froh sein darüber, denn sonst würden wir beim Programmieren an die Grenzen des Speichers ankommen. Das Basic verwandelt jedes Keyword (PRINT, LIST, FOR, NEXT, etc.) in einen TOKEN. Ein Token ist ein Byte, das ein bestimmten Befehl representiert. Somit werden sogar lange Wörter wie RESTORE in einen einziegen Byte komprimiert. Das erspart unheimlich viel Platz. Schauen wir uns mal das Programm von vorhin an, wie es abgespeichert wurde.

```
6A46 ////  ////  A2  03  81  20  41  D5
6A4E 31  20  BD  20  31  30  20  CC
6A56 20  35  00  ////  ////  A2  03  B2
6A5E 20  41  00  ////  ////  00  00  8C
6A66 20  5A  24  D5  F9  28  FA  28
6A6E C7  2C  41  2C  31  29  2C  32
6A76 29  00  ////  ////  0B  04  B2  20
6A7E C7  2C  5A  24  00  00  00  00
6A86 04  87  20  41  00  00  00  00
6A8E 00  00  00  00  00  00  00  00
6A96 00  00  00  00  00  00  00  00
6A9E 00  00  00  00  00  00  00  00
6AA6 00  00  00  00  00  00  00  00
```

//// = Zeiger
~~ = Zeilennummer
0 = EOLN
(00 00 00) = Programmende

Ich habe die besonderen Bytes markiert. Die ersten zwei Bytes sind Zeiger und zeigen auf den Anfang der nächsten Zeile, also bei 6A59h. Die nächsten zwei Bytes sind die Zeilennummer 03F2h = 1010d. Dannach folgt dann der Programmtext, der durch das End of Line Zeichen 00 beendet wird. Dann geht das ganze wieder von vorne los: Zeiger, Zeilennummer, Programmtext und EOLN. Am Schluss des Program müßen dann 3 Nullen stehen. In diesem Fall merkt man das nicht, da alles Nullen sind, es ist

aber immer so. Wenn man NEW eingibt, wird nicht das Programm gelöscht, sondern es werden nur 3 Nullen am Anfang des Programms geschrieben.
Wenden wir uns nun dem Programmtext zu. Es reicht wenn wir uns die erste Zeile anschauen. Hinter der Zeilennummer steht 81. Dies ist der Token für FOR. Dann kommt Space und A und dann der Token für = u.s.w.
Es ist klar, das man so viel Platz spart und außerdem das Basic viel schneller wird, als müste es jedesmal das ganze Wort lesen.

Genau in der selben Form wird es auf Diskette abgespeichert, um Platz zu sparen. Deswegen läßt es sich nicht listen. Wenn man das Programm aber als SAVE "TEST/BAS",A abspeichert, dann wird es im ASCII Format abgespeichert, und man kann es vom DOS aus listen. Es verbraucht aber unheimlich viel mehr Diskettenkapazität.

Nun habe ich einen Programm geschrieben, daß es erlaubt, Basic-Programme im Binären Format zu lesen und als ASCII Format auszugeben. Ich habe es BLIST/CMD genannt. Es ist nicht sehr lang und am besten Ihr tippt es gleich ein. Wie das Programm genau funktioniert, werde ich im nächsten Heft vorstellen. So habt ihr Zeit darüber nachzudenken, und vielleicht kommt ihr ja selber drauf.

Sind noch Fragen offen?
Bis dann und happy typing!!!

Alfonso Sanz
Santa Virgilia 16
28033 MADRID
SPANIEN

Frage an alle UNIDAT80 benutzer!!! Hat jemand es geschaft mit einen 40 SD/DD Laufwerk mit UNIDAT zu arbeiten, ohne daß es nach einer Zeit zum katastrofalen Datenunfug kommt? Ich habe es nur mit PDRIVE - Daten für SD zum laufen gebracht, doch wenn ein Track vollgeschrieben ist, dann will es nichts mehr Schreiben. Kann mir da jemand dringend helfen?
Vielen Dank schon im Voraus.

Alfonso Sanz
Santa Virgilia 16
Esc. izda. 1 º - C
28033 Madrid
SPANIEN

```
                00010 ;BLIST/CMD   Programm um Basicfiles vom Dos aus in Klartext
                00020 ;zu drucken. (C) by Alfonso Sanz
5200            00030          ORG     5200H
5200 11B852     00040 OPEN     LD      DE,DCB  ;Filenamen vom DOS übernehmen
5203 7E         00050 OP1      LD      A,(HL)  ;und als DCB abspeichern.
5204 12         00060          LD      (DE),A
5205 13         00070          INC     DE
5206 23         00080          INC     HL
5207 FE0D       00090          CP      0DH
5209 20F8       00100          JR      NZ,OP1
520B 1B         00110          DEC     DE
520C 3E03       00120          LD      A,03    ;Muß mit 03 enden.
520E 12         00130          LD      (DE),A
520F 11B852     00140          LD      DE,DCB
5212 21CD52     00150          LD      HL,BUFFER       ;HL muß auf einen BUFFER
                00151                                  ;zeigen
5215 CD2444     00160          CALL    4424H   ;File öffnen
5218 C20944     00170          JP      NZ,4409H        ;Error?
521B 21CB54     00171          LD      HL,TXTBUF
521E 22A740     00172          LD      (40A7H),HL      ;Buffer fuer Klartext in
                00173                                  ;(40A7h) abspeichern.
5221 C9         00180          RET
5222 11B852     00190 READ     LD      DE,DCB  ;Vom File im DCB einen Byte lesen
5225 CD1300     00200          CALL    0013H   ;und in A abspeichern.
5228 C20944     00210          JP      NZ,4409H        ;Fehler?
522B C9         00220          RET
522C CD9A0A     00230 ZNR      CALL    0A9AH   ;Zeilennummer ausgeben.
522F AF         00240          XOR     A       ;Muß in Dezimal umgewandelt werden
5230 CD3410     00250          CALL    1034H   ;Es werden Routinen im ROM benutzt
5233 B6         00260          OR      (HL)
5234 CDD90F     00270          CALL    0FD9H
5237 E5         00280          PUSH    HL
5238 AF         00290          XOR     A
5239 23         00300 LOOP1    INC     HL      ;Es wird jetzt das Ende der ZNR
                00301                          ;gesucht.
523A BE         00310          CP      (HL)    ;um einen Space und einen 03 als
523B 20FC       00320          JR      NZ,LOOP1        ;Markierung einzusetzen.
523D 3620       00330          LD      (HL),' '
523F 23         00340          INC     HL
5240 3603       00350          LD      (HL),03H
5242 E1         00360          POP     HL
5243 CD6744     00370          CALL    4467H
5246 C9         00380          RET
5247 CD0052     00390 INIT     CALL    OPEN    ;File eröffnen
524A CD2252     00400          CALL    READ    ;Byte lesen. Gleich 0FFh?
524D FEFF       00410          CP      0FFH    ;Wenn nicht, kein BASIC-File.
524F 204D       00420          JR      NZ,ERROR
5251 CD2252     00430 NEXT     CALL    READ    ;Byte lesen.
5254 47         00440          LD      B,A     ;Wenn 2 Nullen hintereinander folgen
5255 CD2252     00450          CALL    READ    ;Dann ist das File zu ende gelesen.
5258 B0         00460          OR      B       ;Sonst zwei Bytes überspringen
5259 2849       00470          JR      Z,ENDREA
525B CD2252     00480          CALL    READ    ;ZNR lesen
525E 6F         00490          LD      L,A     ;und in HL abspeichern
525F CD2252     00500          CALL    READ
5262 67         00510          LD      H,A
5263 CD2C52     00520          CALL    ZNR     ;ZNR ausdrucken
5266 21CC53     00530          LD      HL,BUF  ;HL zeigt auf einen Buffer
5269 CD2252     00540 LOOPBF   CALL    READ    ;Byte lesen
526C B7         00550          OR      A       ;=0?
526D 77         00560          LD      (HL),A  ;In den Buffer abspeichern
526E 2803       00570          JR      Z,BUFFIN        ;Ja, Ende Zeile.
5270 23         00580          INC     HL      ;nächstes Byte lesen.
5271 18F6       00590          JR      LOOPBF
```

```
5273 21CC53    00600 BUFFIN  LD      HL,BUF   ;HL zeigt auf den Anfang der Zeile
5276 CD7E2B    00610         CALL    2B7EH    ;Aus Tokens lesbaren Text machen.
5279 2AA740    00620         LD      HL,(40A7H)       ;Buffer wo sich der neue Tex
t
               00621                                  ;befindet.
527C E5        00630         PUSH    HL
527D AF        00640         XOR     A
527E 23        00650 LOOP2   INC     HL       ;Ende des Textes suchen
527F BE        00660         CP      (HL)
5280 20FC      00670         JR      NZ,LOOP2
5282 360D      00680         LD      (HL),0DH         ;und mit 0Dh markieren.
5284 E1        00690         POP     HL
5285 CD6744    00700         CALL    4467H    ;Text ausgeben
5288 3A4038    00710         LD      A,(14400)        ;Pfeiltastenkontrolle
528B FE40      00720         CP      64       ;Rechtspfeil
528D CC9652    00730         CALL    Z,WAIT   ;Ja? -> Warten
5290 FE08      00740         CP      8        ;Pfeil oben
5292 2810      00750         JR      Z,ENDREA         ;Ja? -> Ende
5294 18BB      00760         JR      NEXT     ;Nächste Zeile lesen.
5296 3A4038    00770 WAIT    LD      A,(14400)        ;Warten bis
5299 FE01      00780         CP      1        ;New Line (<ENTER>)
529B 20F9      00790         JR      NZ,WAIT  ;gedrückt wird.
529D C9        00800         RET
529E 21A852    00810 ERROR   LD      HL,TEXT  ;Errortext ausgeben, wenn es kein
52A1 C36744    00820         JP      4467H    ;Basic-File war
52A4 CD2844    00830 ENDREA  CALL    4428H    ;File schließen (CLOSE)
52A7 C9        00840         RET      ;Zurück zum DOS
52A8 4E        00850 TEXT    DEFM     'NO BASIC FILE'
52B5 0707      00860         DEFW     0707H    ;nur für H-Dos benutzer
52B7 0D        00870         DEFB     0DH
0015           00880 DCB     DEFS     21       ;DCB - Data Control Block
00FF           00890 BUFFER  DEFS     255      ;I/O Buffer fuer Diskette
00FF           00900 BUF     DEFS     255      ;Zeile mit Tokens
00FF           00901 TXTBUF  DEFS     255      ;Zeile als Klartext
5247           00910         END      INIT
00000 Fehler
32291  Zeichen verfügbar


BUF     53CC 00900    00530 00600
BUFFER  52CD 00890    00150
BUFFIN  5273 00600    00570
DCB     52B8 00880    00040 00140 00190
ENDREA  52A4 00830    00470 00750
ERROR   529E 00810    00420
INIT    5247 00390    00910
LOOP1   5239 00300    00320
LOOP2   527E 00650    00670
LOOPBF  5269 00540    00590
NEXT    5251 00430    00760
OP1     5203 00050    00100
OPEN    5200 00040    00390
READ    5222 00190    00400 00430 00450 00480 00500 00540
TEXT    52A8 00850    00810
TXTBUF  54CB 00901    00171
WAIT    5296 00770    00730 00790
ZNR     522C 00230    00520
```

Achtung!
Entrypoint ist INIT d.h. 5247h.

18 10/85

Einfacher Graphikausdruck

Ich habe mir vor ungefähr drei Wochen den Drucker SG-10 zugelegt.Der SG-10 ist das Nachfolgemodell des Druckers Star Gemini 10 x/i.
Schon nach den ersten paar Tagen kämpfte ich mit dem Ausgeben von Graphiken auf dem Drucker.Es gab Probleme (wie üblich beim ROM des Genie's) mit der Ausgabe bestimmter Codes (z.B. 0AH).Doch für diesen Fall ging das Handbuch sogar speziell auf den TRS-80 ein,indem es einen kleinen aber recht nützlichen Druckertreiber (16 Bytes) angab,der ab der Adresse 16571 geladen wird.
Ich habe die Datas des Basictreibeprogrammes mal disassembliert :

```
            LD    HL,37E8H
label1:     BIT   7,(HL)
            JR    NZ,label1
            LD    HL,0011H
            ADD   HL,SP
            LD    A,(HL)
            LD    (37E8H),A
            RET
```

(Besser noch ist der Druckertreiber von Arnulf : vergl. Info 3/85)

Zuerst wird das HL Register mit der Adresse des Druckerports im I/O RAM-Bereich geladen.Danach wartet der Treiber in einer Schleife solange,bis das Bit 7 dieser Adresse zurückgesetzt ist (Bit 7 ist das Busybit und gibt darüber Auskunft,ob der Drucker noch beschäftigt ist).
In den nächsten zwei Befehlen wird die Position berechnet, an der das zu druckende Zeichen auf dem Stack steht.Der Wert ist zwar auch im C Register,aber dort steht der veränderte Wert (z.B. wurde dort ein 0AH schon in ein 0DH umgewandelt).Zum Glück wurde der Orginalwert aber vorher auf dem Stack abgelegt.Und weil nicht nur der Wert sondern auch andere Register danach (oder Rücksprungadressen von CALLs) auf dem Stack abgelegt wurden,ist die Stelle des Bytes 17 Bytes über dem Stackpointer (auf dem "Weg vom LPRINT" zum Ansprung des Druckertreibers wurden nach dem Retten des Orinalzeichens z.B. an den Adressen 039CH ff,001BH und 03C2H ff Register "gepusht".Außerdem kommt noch ein CALL an der Stelle 03BBH dazu.).

Mit diesem Druckertreiber habe ich ein kleines Basicprogramm geschrieben,welches die auf dem Bildschirm befindlichen Graphikzeichen ausdruckt.Damit die ausgedruckte Graphik nicht so klotzig wie die Graphik auf dem Bildschirm aussieht,habe ich einen Graphikpunkt durch zwei untereinanderliegende Druckpunkte ersetzen lassen.
Ich habe mich,nach längerem Ausprobieren,für eine Auslösung von 90 Punkten/inch entschlossen.Bei einer noch höheren Auflösung ist das Bild zu schmal.
Doch nun zum Basicprogramm :
In Zeile 0 wird erst einmal der Treiber geladen (gosub110) und dann der Zeilenvorschub auf 8/72 inch eingestellt.In Zeile 1 wird dann das Graphikbild auf den Bildschirm geladen.Hier ist es ein Bild,das mit dem 3-Tastenbefehl von Alfonso Sanz abgespeichert wurde.
Zeile 1 kann auch durch ein Gosub ersetzt werden,wo dann das Bild gezeichnen oder von Kassette geladen wird.
Zeile 15 stellt dann die Auflösung von 90 Punkten/inch ein.
Zeile 30 dient zur Umwandlung von 4 untereinanderliegenden Punkten in eine 1 Bytezahl.Wenn der obere Punkt gesetzt ist,dann entspricht das dem Setzen von Bit 7 und 6,beim

zweiten Punkt von oben werden Bit 5 und 4 gesetzt usw.
Dieses Byte wird dann zum Drucker geschickt.
Der Zeilenvorschub des Druckers wird in Zeile 90 dann wieder
auf den normalen Abstand (1/6 inch) eingestellt.
Dauert einem dieses Basicprogramm zu lange,so wird es sicher
nicht schwer sein,das entsprechende Maschinen-
spracheprogramm (z.B. für einen neuen 3-Tasten Befehl) zu
schreiben.Man muß lediglich eine Point-Routine schreiben und
das gebildete Byte in 37E8H laden (oder Port FDH).
Ich habe noch zwei Graphiken ausdrucken lassen,damit Ihr Euch
mal ein Bild davon machen könnt,wie das Gedruckte nachher
aussieht.

Jörg Seelmann-Eggebert

```
0 GOSUB110:LPRINTCHR$(27)CHR$(65)CHR$(8);
1 CMD"load f9/cmd"
10 FORI1=0TO11
15 LPRINTCHR$(27)"g"CHR$(6)CHR$(128)CHR$(0);
20 FORI2=0TO127
30 LPRINTCHR$(-192*POINT(I2,I1*4)-48*POINT(I2,I1*4+1)-12*POINT(I2,I1*4
+2)-3*POINT(I2,I1*4+3));
40 NEXTI2
50 LPRINT
60 NEXTI1
90 LPRINTCHR$(27)CHR$(50)
100 END
110 AD=16571
120 FORI=0 TO 15
130 READA:POKEAD + I,A
140 NEXT
150 POKE16422,187:POKE16423,64
155 RETURN
160 DATA33,232,55,203,126,32,252,33,17,0,57,126,50,232,55,201
```

zwei Probeausdruecke

Subroutinen im Newdos80.2 (ständig im System)
==========

Von Michael Schau

4063H ********** DE in Hex ausgeben
-> DE enthält den Wert, HL die Pufferadresse
<- HL zeigt auf das folgende Byte

4068H ********** A in Hex ausgeben
-> A enthält den Wert, HL die Pufferadresse
<- HL zeigt auf das folgende Byte

44A4H ********** Uhrzeit in rechte obere Bildschirmecke
-> keine Bedingungen
<- HL = 3C3DH, DE = 4040H

44A7H ********** Uhrzeit in Puffer (HL) schreiben
-> HL zeigt auf Puffer
<- HL = Pufferende+1, DE = 4040H

44C2H ********** Datum in Puffer (HL) schreiben
-> HL zeigt auf Puffer
<- HL = Pufferende+1, DE = 4043H

45B5H ********** Großbuchstaben - Umwandlung
-> Zeichen in A
<- Großbuchstabe in A

4630H ********** Sektor lesen
-> DE = absolute Sektornummer, HL = Pufferadresse,
(4308H) = aktuelle Drivenummer
<- NZ = Fehler aufgetreten, Code in A

4640H ********** Sektor schreiben
-> DE = absolute Sektornummmer, HL = Pufferadresse,
(4308H) = aktuelle Drivenummer
<- NZ = Fehler aufgetreten, Code in A

4762H ********** FDC-Status lesen
-> (4309H) enthält maskiert aktuelle Drivenummer
<- C, wenn FDC Ready, NC, wenn erneute Driveanwahl

4773H ********** Drive (4308H) aktivieren (DCT)
-> aktuelle Drivenummer in (4308H)
<- NZ = Fehler aufgetreten Code in A

4776H ********** Drive (A) aktivieren (DCT)
-> Drivenummer in A
<- NZ = Fehler aufgetreten, Code in A

490AH ********** Directory-Sektor lesen
-> Drivenummer in (4308H), DIR-Sektornummer in A
<- NZ = Fehler aufgetreten, A = 11H = DIR-read-error

491FH ********** Directory-Sektor schreiben
-> Drivenummer in (4308H), DIR-Sektor mußte vorher
gelesen worden sein, Nummer in (4930H)
<- NZ = Fehler aufgetreten, A = 12H = DIR-write-error

4922H ********** Directory-Sektor schreiben
-> Drivenummer in (4308H), DIR-Sektornummer in A
<- NZ = Fehler aufgetreten, A = 12H = DIR-write-error

492FH ********** DIR-Entry lesen
-> Drivenummer in (4309H), DEC in A
<- HL zeigt auf DIR-Entry, NZ = Fehler aufgetreten

4968H ********** FCB NEXT-Wert mit EOF-Wert vergleichen
-> IX zeigt auf den FCB, High-Order-Byte in H, Middle-Order-Byte in L, Low-Order-Byte in C
<- Flags (NEXT - EOF)

4C92H ********** L mit A multiplizieren
-> L und A enthalten die Faktoren
<- Produkt in HL, A = MSB

4C94H ********** HL mit A multiplizieren
-> HL und A enthalten die Faktoren
<- Produkt in HL, A = MSB

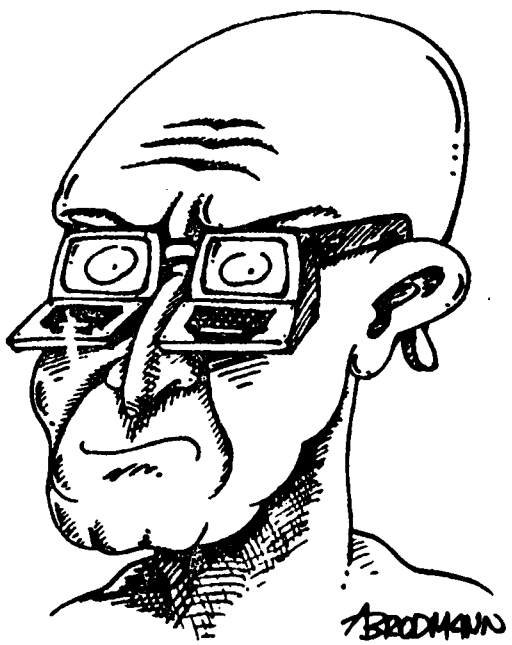
4CB2H ********** HL durch 5 teilen
-> HL enthält Zahl
<- Quotient in L, Rest in A, Z wenn A=0

4CB4H ********** HL durch A teilen
-> HL und A enthalten Divisor und Dividenden
<- Quotient in L, Rest in A, Z wenn A=0

4CC5H ********** Stringvergleich
-> HL zeigt auf String, BC auf Tabelle, Tabellenstring muß mit 00H enden
<- NZ = Strings nicht gleich, Z = Strings gleich, HL zeigt auf Tabellenwortanfang

4CD5H ********** Test auf Separator/Terminator
-> HL zeigt auf Trennzeichen
<- Z,NC : Trennzeichen = CR
NZ,NC : Trennzeichen Space oder Komma
C : Illegal keyword/separator/terminator-Code in A (34H)

4CEDH ********** Lange Warteschleife
-> Schleifendurchläufe in B, berücksichtigt Takterhöhung



```
100 CLS:CLEAR 7000
110 '=---->DISKA095/BAS Version 1.1 * 6/1985<---=
120 '=---->Bildschirmgrafik über Prog.-Information, Copyright u.s.w<---=
130 FOR X=31 TO 99 STEP 2:SET(X,1):NEXT
140 FOR Y=1 TO 37 STEP 2:SET(31,Y):SET(99,Y):NEXT
150 FOR X=99 TO 31 STEP -2:SET(X,37):NEXT
160 PRINT@85,"Programm:  DISKA095/BAS  ":PRINT@217,"von  H. Thönnißen"
170 PRINT@341,"für  TANDY / TRS-80 M1":PRINT@466,"NEWDOS80 (2.052)  BASIC L. II"
180 PRINT@599,"Copyright (C) 3/1982":PRINT@725,"by H. Thönnißen / BREMEN"
190 FOR Y=1 TO 37 STEP 2:SET(31,Y):SET(99,Y):NEXT
200 FOR X=97 TO 33 STEP -2:SET(X,1):SET(X,6):SET(X,13):SET(X,18):SET(X,25):SET(X,
30):SET(X,37):NEXT
210 PRINT@903,"Zum Programm-Start bitte <<<E N T E R>> drücken ";:INPUT Y$:IF Y$=
"" THEN CLS:GOTO 300
220 IF LS=0 THEN LPRINT"Lfd.-Nr.:   DISK.-Nr.:   DISK.-Name:   Arb./Sys.-Disk.:
 Bemerkung:"
230 LPRINT STRING$(80,CHR$(131)):LPRINT:RETURN
240 ' UNTERPROGRAMM ZUM EINORDNEN DES UNTERSATZES IN-
250 ' NERHALB DES 256 BYTES LANGEN PHYSISCHEN SATZES.
260 DF=LS-4*INT((LS-1)/4)-1
270 PRINT@393,"UNTERSATZ-NR. = ";DF
280 FIELD1,DF*44 AS G$,2 AS A1$,8 AS A2$,12 AS A3$,22 AS A4$
290 RETURN
 90 DIM A$(4):Z1$="..":Z2$=".......":Z3$=".....":Z4$=STRING$(22,"."):ZE=0
310 PRINT TAB(11)"D I S K E T T E N - A U F L I S T U N G"
320 PRINT CHR$(151);STRING$(23,CHR$(131));CHR$(171);STRING$(21,CHR$(131));CHR$(17
1);STRING$(15,CHR$(131));CHR$(171)
330 PRINT CHR$(149)"  Daten-File-Programm  "CHR$(170);"  Datum :  tt.mm.jj";DT$;
340 PRINT@174,CHR$(170)" Sätze ges. 95";TAB(62);CHR$(170)
350 PRINT STRING$(63,CHR$(131)):PRINT
360 PRINT@403,"Tages-Datum : ";:INPUT DT$:PRINT
370 PRINT@164,DT$
380 PRINT STRING$(63,CHR$(131)):PRINT
390 PRINT@904,"Weiter im Programm bitte  <E N T E R>  drücken ";:INPUT W$
400 CLS:PRINT TAB(11)"D I S K E T T E N - A U F L I S T U N G"
410 PRINT CHR$(151);STRING$(23,CHR$(131));CHR$(171);STRING$(21,CHR$(131));CHR$(17
1);STRING$(15,CHR$(131));CHR$(171)
420 PRINT CHR$(149)"  Daten-File-Programm  "CHR$(170);"  Datum :  ";DT$;
430 PRINT@174,CHR$(170)" Satz-Nr.: ";LS;TAB(62);CHR$(170)
440 PRINT STRING$(63,CHR$(131))
450 PRINT@338,"Wahlmöglichkeiten ...(1-4)"
460 PRINT@466,"DISKETTEN-File ....."
 70 PRINT@530,"   -/-    schreiben .....1"
480 PRINT@594,"   -/-    lesen .........2"
490 PRINT@658,"   -/-    drucken .......3"
500 PRINT@786,"Programm - E N D E ......4"
510 PRINT@914,"Ihre Eingabe bitte ....";:INPUT N$:N=VAL(N$)
520 IF N<1 OR N>4 THEN CLS:GOTO 400
530 IF N=4 THEN CLS:GOTO 1150
540 IF N=1 THEN CLOSE:OPEN"R",1,"Diskette/Auf":GOTO 570
550 IF N=2 THEN CLOSE:OPEN"R",1,"Diskette/Auf":GOTO 720
560 IF N=3 THEN CLOSE:OPEN"R",1,"Diskette/Auf":GOTO 870
570 CLS:PRINT TAB(11)"D I S K E T T E N - A U F L I S T U N G"
580 PRINT CHR$(151);STRING$(23,CHR$(131));CHR$(171);STRING$(21,CHR$(131));CHR$(17
1);STRING$(15,CHR$(131));CHR$(171)
590 PRINT CHR$(149)" Daten-File: SCHREIBEN "CHR$(170);"  Datum :  ";DT$;
600 PRINT@174,CHR$(170)" Satz-Nr.: ";LS;TAB(62);CHR$(170)
610 PRINT@185,LS
620 PRINT STRING$(63,CHR$(131))
630 PRINT@329,"NUMMER DES LOGISCHEN SATZES EINGEBEN ";:INPUT LS
640 IF LS=0 THEN CLS:CLOSE:GOTO 400
650 GOSUB 240:PS=INT((LS-1)/4)+1
660 GET 1,PS:PRINT@457,"PHYSISCHE SATZ-NR. = ";PS
670 PRINT@592,CHR$(30);:PRINT@609,Z1$;CHR$(138);:PRINT@585,"1.  Disk.-Nr.:
 ";:INPUT A$:LSET A1$=A$
680 PRINT@656,CHR$(30);:PRINT@673,Z2$;CHR$(138);:PRINT@649,"2.  Disk.-Name:
 ";:INPUT A$:LSET A2$=A$
```

Verfasser

23
10/85

```
690 PRINT§720,CHR$(30);:PRINT§737,Z3$;CHR$(138);:PRINT§713,"3.  Arb./Sys.-Disk.
 ";:INPUT A$:LSET A3$=A$
700 PRINT§784,CHR$(30);:PRINT§801,Z4$;CHR$(138);:PRINT§777,"4.  Bemerkung:
 ";:INPUT A$:LSET A4$=A$
710 PUT 1,PS:GOTO 610
720 CLS:PRINT TAB(11)"D I S K E T T E N - A U F L I S T U N G"
730 PRINT CHR$(151);STRING$(23,CHR$(131));CHR$(171);STRING$(21,CHR$(131));CHR
1);STRING$(15,CHR$(131));CHR$(171)
740 PRINT CHR$(149)" Daten-File:   LESEN   "CHR$(170);"  Datum :  ";DT$;
750 PRINT§174,CHR$(170)" Satz-Nr.: ";LS;TAB(62);CHR$(170)
760 PRINT§185,LS
770 PRINT STRING$(63,CHR$(131))
780 PRINT§329,"NUMMER DES LOGISCHEN SATZES EINGEBEN ";:INPUT LS
790 IF LS=0 THEN CLS:CLOSE:GOTO 400
800 GOSUB 240:PS=INT((LS-1)/4)+1
810 GET 1,PS:PRINT§457,"PHYSISCHE SATZ-NR. = ";PS
820 PRINT§592,CHR$(30);:PRINT§609,Z1$;CHR$(138);:PRINT§585,"1.  Disk.-Nr.:
   ";A1$
830 PRINT§656,CHR$(30);:PRINT§673,Z2$;CHR$(138);:PRINT§649,"2.  Disk.-Name:
   ";A2$
840 PRINT§720,CHR$(30);:PRINT§737,Z3$;CHR$(138);:PRINT§713,"3.  Arb./Sys.-Disk.:
   ";A3$
850 PRINT§784,CHR$(30);:PRINT§801,Z4$;CHR$(138);:PRINT§777,"4.  Bemerkung:
   ";A4$
860 PUT 1,PS:GOTO 760
870 CLS:PRINT TAB(11)"D I S K E T T E N - A U F L I S T U N G"
880 PRINT CHR$(151);STRING$(23,CHR$(131));CHR$(171);STRING$(21,CHR$(131));CHR$(17
1);STRING$(15,CHR$(131));CHR$(171)
890 PRINT CHR$(149)" Daten-File:  DRUCKEN  "CHR$(170);"  Datum :  ";DT$;
900 PRINT§174,CHR$(170)" Satz-Nr.: ";LS;TAB(62);CHR$(170)
910 PRINT§185,LS
920 PRINT STRING$(63,CHR$(131))
930 IF LS=0 THEN PRINT§403,"Vorspann drucken (J/N)"
940 IF LS=0 THEN PRINT§531,"Drucker  R E A D Y ? ";:INPUT P$:PRINT§384,CHR$(30);:
PRINT§512,CHR$(30)
950 IF LS=0 AND P$="n" OR P$="N" THEN 1020
960 IF LS=0 THEN LPRINT CHR$(27)"E";CHR$(14)"Bestand vohandener Disketten ! ";DT$
:LPRINT
970 IF LS=0 THEN LPRINT CHR$(14);STRING$(40,"*")
980 IF LS=0 THEN LPRINT"Auflistung von: TRSYS...   GDSYS...   NDSYS...   HDSYS...
   HWORK...   NWORK..."
990 IF LS=0 THEN LPRINTCHR$(14);STRING$(40,"*"):LPRINT:LPRINT
1000 IF LS=0 THEN LPRINT"Lfd.-Nr.:   DISK.-Nr.:   DISK.-Name:   Arb./Sys.-Disk.:
  Bemerkung:"
1010 IF LS=0 THEN LPRINT CHR$(27)":";STRING$(80,CHR$(131)):LPRINT
1020 PRINT§329,"NUMMER DES LOGISCHEN SATZES EINGEBEN ";:INPUT LS
1030 IF LS=0 THEN CLS:CLOSE:GOTO 400
1040 GOSUB 240:PS=INT((LS-1)/4)+1
1050 GET 1,PS:PRINT§457,"PHYSISCHE SATZ-NR. = ";PS
1060 PRINT§592,CHR$(30);:PRINT§609,Z1$;CHR$(138);:PRINT§585,"1.  Disk.-Nr.:
    ";A1$
1070 PRINT§656,CHR$(30);:PRINT§673,Z2$;CHR$(138);:PRINT§649,"2.  Disk.-Name:
    ";A2$
1080 PRINT§720,CHR$(30);:PRINT§737,Z3$;CHR$(138);:PRINT§713,"3.  Arb./Sys.-Disk.:
    ";A3$
1090 PRINT§784,CHR$(30);:PRINT§801,Z4$;CHR$(138);:PRINT§777,"4.  Bemerkung:
    ";A4$
1100 ZE=ZE+1
1110 LPRINT TAB(2);ZE;TAB(16);A1$;TAB(26);A2$;TAB(40);A3$;TAB(58);A4$
1120 IF ZE=55 THEN FOR LP=1 TO 6:NEXT
1130 IF ZE=55 GOSUB 220
1140 PUT 1,PS:GOTO 910
1150 END
```

24
10185

Eingesandt von . Kajott

PROGRAMMSERVICE

```
960 'DATEN EINGEBEN************************
970 BORDER 13:GOSUB 1510:LOCATE #1,7,1: PRINT #1,"D a t e n - e i n g e b e n":A
=A+1:FOR Z=1 TO Q
980 LOCATE #2,11,1:PRINT #2,">>> ZEILE NR. ";Z;" <<<":LOCATE #2,3,2:PRINT #2,"EI
NGABE-";Q;"ZEILEN <ENTER>=LEERZEILE"
990 LOCATE #4,3,3:PRINT #4,"Datensatz Nr.";A:GOSUB 1520
1000 PRINT #4,CHR$(7):LOCATE #4,1,Z+5:INPUT #4,D$(Z,A):NEXT Z
1010 CLS #2:LOCATE #2,8,2:PRINT #2,"Möchten Sie etwas ändern <J/N>"
1020 A$="":WHILE A$="":A$=UPPER$(INKEY$):WEND
1030 IF A$="J" THEN 690
1040 IF A$="N" THEN 130
1050 GOTO 1020
1060 'AUFLISTEN***************************
1070 BORDER 10:GOSUB 1510:LOCATE #1,4,1:PRINT #1,"A u f l i s t e n -<1>- Z e i
l e"
1080 LOCATE #2,2,2:PRINT #2,"<1>-listen              <2>-verarbeiten"
1090 LOCATE #3,1,1:PRINT #3,"Start:";CHR$(7):LOCATE #3,1,3:INPUT #3,ST:IF ST<1 T
HEN ST=1 ELSE IF ST>A THEN ST=A
1100 LOCATE #3,1,8:PRINT #3,"Stop:";CHR$(7):LOCATE #3,1,10:INPUT #3,STO:IF STO<S
T THEN STO=ST ELSE IF STO>A THEN STO=A
1110 FOR D=ST TO STO STEP 20
1120 FOR Z=0 TO 19
1130 LOCATE #4,1,Z+1:PRINT #4,D+Z;D$(1,D+Z):NEXT Z
1140 WW$="":WHILE WW$="":WW$=INKEY$:WEND
1150 IF WW$="1" THEN 1180
1160 IF WW$="2" THEN 790
1170 GOTO 1140
1180 CLS #4:NEXT D:GOTO 1110
1190 'DATENSATZ LOESCHEN*****************
1200 BORDER 15:GOSUB 1510
1210 PRINT #1,"   D a t e n s a t z   l ö s c h e n"
1220 PRINT #2,CHR$(7):LOCATE #2,1,1:INPUT #2,"Welchen Datensatz moechten Sie lös
chen?           Datensatz Nr.";D
1230 LOCATE #4,3,3:PRINT #4,"Datensatz Nr.";D
1240 GOSUB 1520
1250 FOR Z=1 TO Q
1260 LOCATE #4,3,Z+5:PRINT #4,D$(Z,D)
1270 NEXT Z
1280 CLS #2:LOCATE #2,8,2:PRINT #2,"Sind Sie ganz sicher <J/N>";CHR$(7)
1290 L$="":WHILE L$="":L$=UPPER$(INKEY$):WEND
1300 IF L$="J" THEN 1330
1310 IF L$="N" THEN 790
1320 GOTO 1290
1330 FOR Z=1 TO Q
1340 D$(Z,D)=" "
1350 NEXT Z
1360 GOTO 790
1370 STOP
1380 'DRUCK-AUSGABE ALS LISTE***********
1390 BORDER 18:GOSUB 1510:PRINT #1,TAB(5);"D r u c k a u s g a b e / gesamt"
1400 LOCATE #2,5,2:PRINT #2,"Drucker auf 'on line' - any key";CHR$(&7):CALL &BB0
6
1410 PRINT #8,TAB(10);CHR$(14);"* ";Na$;"-Datei *":PRINT #8
1420 FOR D=1 TO A
1430 LOCATE #4,3,3:PRINT #4,"Datensatz Nr.";D:GOSUB 1520
1440 PRINT #8,CHR$(18);"<";D;">";
1450 FOR Z=1 TO Q
1460 LOCATE #4,1,Z+5:PRINT #4,D$(Z,D)
1470 IF Z=1 OR Z=3 OR Z=5 OR Z=7 OR Z=9 THEN PRINT #8,TAB(10);D$(Z,D); ELSE PRIN
T #8,TAB(44);D$(Z,D)
1480 NEXT Z
1490 PRINT #8,TAB(10);"- - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - -":CLS #4:NEXT D
1500 GOTO 130
1510 CLS #1:CLS #2:CLS #3:CLS #4:RETURN
1520 LOCATE #3,1,1:PRINT #3,"-Bytes--free";FRE(0)
1530 FOR X=1 TO Q:LOCATE #3,1,5+X:PRINT #3,Bez$(X):NEXT X
1540 RETURN
1550 NEW
1560 END
```

Colour-Genie – Hardcopy für die Bildschirmgrafik

Richtig aufs Papier

Die Bildschirmgrafik des Colour-Genie läßt sich mit dieser Routine ausdrucken. Sie ist für den Drucker Epson FX-80 ausgelegt und erlaubt verschiedene Vergrößerungen.

Das Colour-Genie bietet mit den „alten" ROM eine hochauflösende Grafik mit 160 × 96 Punkten. Diese Grafik wird vom BASIC durch Befehle wie PLOT und CIRCLE unterstützt. Zeichnet man allerdings einen Kreis, so stellt man enttäuscht fest, daß er vertikal gedehnt dargestellt wird. Dieser Fehler läßt sich auch durch Einstellung am Fernseher oder Monitor nicht beheben. So bleibt der Wunsch, wenigstens auf dem Drucker ein unverzerrtes Bild zu erhalten. Hierzu ist ein Drucker erforderlich, der mit einer Dichte von 640 Punkten auf 20 cm druckt. Diese Eigenschaft weisen bisher nur wenige Drucker auf, beispielsweise aber der Epson FX-80.

Läßt man je Grafikpunkt einen Punkt ausdrucken, so erhält man tatsächlich einen runden Kreis, aber der Ausdruck ist ziemlich klein. Da sich die Auflösung des Colour-Genie nicht vergrößern läßt, bleibt die Möglichkeit, für jeden Bildschirmpunkt mehrere Punkte drucken zu lassen. Soll der Ausdruck weiterhin unverzerrt bleiben, muß waagrecht und senkrecht um den gleichen Faktor vergrößert werden. Da horizontal bis zu 640 Punkte gedruckt werden können und die hochauflösende Grafik waagrecht mit 160 Punkten arbeitet, ist bis zu vierfache Vergrößerung möglich. Dies bedeutet, daß jeder Bildschirmpunkt durch ein Quadrat mit vier mal vier Punkten dargestellt wird. Das Programm erlaubt auch die Vergrößerungen 2 und 3. Ein vergrößerter Ausdruck ergibt natürlich keine höhere Auflösung, wohl aber ein größeres Bild und erlaubt damit die Betrachtung aus größerem Abstand.

In der hochauflösenden Grafik sind vier Farben möglich. Entsprechend der Darstellung auf einem monochromen Sichtgerät wird für jeden nicht schwarzen Bildschirmpunkt ein Punkt gedruckt.

Neuere Modelle des Colour-Genie bieten eine auf 160 × 102 Punkte erhöhte Auflösung. Hiervon werden nur 160 × 96 Punkte ausgedruckt. Ansonsten arbeitet das Programm mit den alten und den neuen ROM.

Ein Hardcopy-Programm läßt sich in BASIC oder in Assembler schreiben. Wird in Assembler programmiert, so ist die Ausführungszeit ausschließlich durch den Drucker bestimmt, das heißt, der Drucker arbeitet ununterbrochen. Während er eine Zeile ausgibt, berechnet das Programm schon die nächste Druckzeile. Ich habe je nach Vergrößerung Ausführungszeiten zwischen 11 s und 112 s ermittelt.

Hochauflösende Grafiken werden ab 4800H im RAM abgelegt. Jeweils ein Byte entspricht vier horizontal nebeneinanderliegenden Punkten. Für jeden Punkt stehen damit 2 bit zur Speicherung von vier Farben zur Verfügung. Sind beide Bit nicht gesetzt, so bedeutet das einen dunklen Punkt.

Wird auf den FX-80 eine Punktgrafik ausgegeben, so entspricht jedes Byte

acht untereinanderliegenden Punkten. Es müssen somit jeweils acht Bildschirmzeilen zu einer Druckzeile zusammengefaßt werden. Hierzu wird ein Puffer verwendet, in den eine ganze Druckzeile geschrieben wird, die anschließend an den Drukker ausgegeben wird. Als Puffer bot sich der vom BASIC verwendete I/O-Puffer ab 41E8H an.

Soll mit einer Vergrößerung größer eins ausgedruckt werden, so muß jeder Punkt waagrecht und senkrecht mehrfach ausgegeben werden. Horizontal wird einfach jedes Byte mehrfach an den Drucker ausgegeben. Vertikal wird bei Aufbereitung der Druckzeile jede Bildschirmzeile entsprechend der Vergrößerung mehrfach durchgegangen. Dieser Weg scheint zunächst unnötig zeitaufwendig, die Praxis zeigt aber, daß der Drucker bei allen Vergrößerungen mit maximaler Geschwindigkeit arbeitet.

Und wohin nun mit dem Maschinenprogramm? Es kann an beliebiger Stelle im RAM abgelegt werden. Sofern zur Erstellung der hochauflösenden Grafik keine Shapes verwendet wurden, bietet sich die Shape-Table an. Die Anfangsadresse ist dann 7F00H bei 16K und BF00H bei 32K. Alternativ bietet sich die Ablage am oberen oder unteren Ende des BASIC-Programmspeichers an. Hierbei ist auf Konflikte mit eventuell gleichzeitig im Speicher befindlichen BASIC-Erweiterungen zu achten. Das Maschinenprogramm kann auch einfach irgendwo in das RAM gelegt werden. Hierbei nimmt man dann eben in Kauf, daß das Anwenderprogramm nach dem Ausdruck neu geladen werden muß.

Der Aufruf der Hardcopy erfolgt mit
A=USR(n).

Hierbei ist n die Vergrößerung, also eine Zahl von eins bis vier. Eine unsinnige Vergrößerung führt zur Meldung FC-Error.

Entwickelt wurde das Programm für den Drucker Epson FX-80. Es müßte auch mit RX-80 und FX-100 arbeiten. Zur Anpassung an andere Drucker ist der Anfang des Unterprogramms PRINT (siehe Assemblerlisting) anzupassen. Unter Verzicht auf die unverzerrte Wiedergabe ist jeder punktgrafikfähige Drucker geeignet.

Wolfgang Ottenweller

```
01000 ; FSR-HARDCOPY FUER COLOUR-GENIE
01010 ; ==============================
01020 ; speziell fuer Epson FX-80!
01030 ;
01040 ; von W.Ottenweller,7988 Wangen
01050 ;
01060 ; Registerbelegung:
01070 ; ----------------
01080 ;
01090 ; D  Bildschirmzeilenzaehler
01100 ; E  Vergroesserung
01110 ; HL Bildschirmzeiger
01120 ; B' Bytes je Bildschirmzeile
01130 ; C' Vergroesserungszaehler
01140 ; D' Punktezaehler Druckerbyte
01150 ; E' Druckerpunktmaske
01160 ; H' Bildschirmpunktmaske
01170 ; L' Punkte je Bildschirmbyte
01180 ; IX Bildschirmzeiger
01190 ; IY Druckpufferzeiger
01200 ;
01210 ;
01220 ; geaenderte Belegung fuer PRINT:
01221 ; ------------------------------
01230 ;
01240 ; B  Druckbytezaehler
01250 ; C  Ausgabebyte
01260 ; D  Vergroesserungszaehler
01270 ; E  Vergroesserung
01280 ; HL Druckpufferzeiger
01290 ;
01300 ;
01310 ;
01320 ;          ORG 0B000H ;oder wo Sie wollen
01330 ;
01340 ; Zuweisungen:
01341 ; ------------
01350 ;
4800  01360 RAM     EQU 4800H ;FSR-RAM-Anfang
41E8  01370 BUFFER  EQU 41E8H ;Eingabepuffer, dient als Druckpuffer
00A0  01380 BUFLEN  EQU 160 ;Druckpufferlaenge
00C8  01390 CMASK   EQU 0C8H ;Anfangswert Bildschirmpunktmaske
0080  01400 PMASK   EQU 80H ;Anfangswert Druckerpunktmaske
0004  01410 CDOTS   EQU 4 ;Punkte je Bildschirmbyte
0008  01420 PDOTS   EQU 8 ;Punkte je Druckerbyte
0028  01430 CBYTES  EQU 40 ;Bytes je Bildschirmzeile
0060  01440 CLINES  EQU 96 ;Bildschirmzeilenzahl
01450 ;
01460 ;
01470 ; ROM-Unterprogramme:
01480 ; --------------------
01490 ;
1E4A  01500 FCERR   EQU 1E4AH ;FC-Error-Einsprung
0529  01510 BUSY    EQU 529H ;testet, ob Drucker bereit
053C  01520 OUTPUT  EQU 53CH ;Register C an Drucker ausgeben
01530 ;
01540 ;
408E  01550 USRADR  EQU 16526 ;Ansprungadresse fuer USR-Befehl
01560 ;
01570 ;
01580 ;
01590 ; Hauptprogramm
01591 ; -------------
01600 ;
B000 CD7F0A 01620 ENTRY   CALL 0A7FH ;Uebergabewert der USR-Funktion nach HL
B003 7C     01630         LD A,H
B004 B7     01640         OR A
B005 C24A1E 01650         JP NZ,FCERR ;hohes Byte muss 00 sein
B008 7D     01660         LD A,L
B009 3D     01670         DEC A ;sollte jetzt 0...3 sein
B00A E6FC   01680         AND 0FCH ;0...3 AND 0FCH ergibt 00
B00C C24A1E 01690         JP NZ,FCERR ;war zu gross
B00F 5D     01700         LD E,L ;Vergroesserung nach E
B010 210000 02420         LD HL,0
B013 11A800 02430         LD DE,BUFLEN
B016 19     02440 PLUS    ADD HL,DE
B017 10FD   02450         DJNZ PLUS ;HL=Druckpufferlaenge*Vergroesserung
B019 F5     02460         PUSH AF
B01A 48     02470         LD C,L
B01B CDC1B0 02480         CALL SEND
B01E 4C     02490         LD C,H
B01F CDC1B0 02500         CALL SEND ;Druckbytezahl an Drucker
B022 F1     02510         POP AF
B023 5F     02520         LD E,A ;Vergroesserung nach E
B024 21E841 02530         LD HL,BUFFER ;Druckpufferanfang
B027 0640   02540         LD B,BUFLEN ;Druckpufferlaenge
B029 53     02550 REPEAT  LD D,E ;Vergroesserungszaehler
B02A 4E     02560 LOOP    LD C,(HL) ;Druckbyte holen
B02B CDC1B0 02570         CALL SEND ;und ausgeben
B02E 15     02580         DEC D ;bei Vergroesserung das gleiche Druckbyte
B02F 20F9   02590         JR NZ,LOOP ;mehrmals ausgeben
B031 23     02600         INC HL ;Zeiger auf naechstes Druckbyte
B032 10F5   02610         DJNZ REPEAT ;und wiederholen
B034 0E0D   02620         LD C,13
B036 CDC1B0 02630         CALL SEND ;Zeilenvorschub
B039 E1     02640         POP HL
B03A D1     02650         POP DE
B03B C1     02660         POP BC ;Register zurueck
            02670 ;
            02680 ;
            02690 ; Druckpuffer loeschen
            02700 CLRBUF  PUSH HL
            02710         PUSH BC ;Register retten
            02720         LD HL,BUFFER ;Druckpufferanfang
            02730         LD B,BUFLEN ;Druckpuffer loeschen
            02740         XOR A ;A loeschen
            02750 CLEAR   LD (HL),A
            02760         INC HL
            02770         DJNZ CLEAR ;Druckpuffer loeschen
            02780         POP BC
```

```
B010 1640      01710          LD D,CLINES ;Bildschirmzeilenzahl
B012 21D847    01720          LD HL,RAM-CBYTES ;eine Zeile vor FOR-RAM
B015 D9        01730          EXX
B016 1608      01740          LD D,PDOTS ;Punktezahl Druckerbyte
B018 1E80      01750          LD E,PMASK ;Druckerpunktmaske setzen
B01A CDB2B0    01760          CALL CLRBUF ;Druckpuffer loeschen
B01D 26C0      01770          LD H,CMASK ;Bildschirmpunktmaske setzen
B01F D9        01780 NXLINE   EXX
B020 D5        01790          PUSH DE
B021 112800    01800          LD DE,CBYTES
B024 19        01810          ADD HL,DE ;Bildschirmzeiger eine Zeile weiter
B025 D1        01820          POP DE
B026 7B        01830          LD A,E
B027 D9        01840          EXX
B028 4F        01850          LD C,A ;Vergroesserungszaehler setzen
B029 D9        01860 AGAIN    EXX
B02A E5        01870          PUSH HL
B02B DDE1      01880          POP IX ;Bildschirmzeiger kopieren
B02D D9        01890          EXX
B02E 0628      01900          LD B,CBYTES ;Bytes je Bildschirmzeile
B030 FD21E841  01910          LD IY,BUFFER ;Druckpufferzeiger setzen
B034 2E04      01920 NXBYTE   LD L,CDOTS ;Punkte je Bildschirmbyte
B036 DD7E00    01930 NXDOT    LD A,(IX+0) ;Bildschirmbyte holen
B039 A4        01940          AND H ;Bildschirmpunkt maskieren
B03A 2807      01950          JR Z,NODOT ;dunkel?
B03C FD7E00    01960          LD A,(IY+0) ;Druckbyte holen
B03F B3        01970          OR E ;Punkt setzen
B040 FD7700    01980          LD (IY+0),A ;Druckbyte ablegen
B043 CB0C      01990 NODOT    RRC H
B045 CB0C      02000          RRC H ;Bildschirmpunktmaske einen Punkt weiter
B047 FD23      02010          INC IY ;Druckbytezeiger ein Byte weiter
B049 2D        02020          DEC L ;Zaehler Punkte je Bildschirmbyte
B04A 20EA      02030          JR NZ,NXDOT ;naechster Punkt
B04C DD23      02040          INC IX ;Bildschirmzeiger erhoehen
B04E 10E4      02050          DJNZ NXBYTE ;naechstes Bildschirmbyte
B050 CB0B      02060          RRC E; Druckerpunktmaske einen Punkt weiter
B052 15        02070          DEC D ;Zaehler Punkte je Druckerbyte
B053 CC5FB0    02080          CALL Z,PRINT ;wenn Druckzeile fertig, ausgeben
B056 0D        02090          DEC C ;Vergroesserungszaehler
B057 20D0      02100          JR NZ,AGAIN ;bei Vergroesserung gleiche
               02101 ; Bildschirmzeile mehrmals durchgehen
B059 D9        02110          EXX
B05A 15        02120          DEC D ;Bildschirmzeilenzaehler
B05B D9        02130          EXX
B05C 20C1      02140          JR NZ,NXLINE ;naechste Bildschirmzeile
B05E C9        02150          RET ;fertig, zurueck nach BASIC
               02160 ;
               02170 ;
               02180 ;
               02190 ; Unterprogramme:
               02191 ; ---------------
               02200 ;
               02210 ; Druckpuffer an Drucker ausgeben
B05F 1608      02220 PRINT    LD D,8 ;Punktzaehler Druckerbyte neu setzen
B061 C5        02230          PUSH BC
B062 D5        02240          PUSH DE
B063 E5        02250          PUSH HL ;Register retten
B064 21E841    02260          LD HL,BUFFER ;Druckpufferzeiger setzen
B067 06A0      02270          LD B,BUFLEN ;Druckpufferlaenge
B069 0E1B      02280          LD C,27
B06B CDC1B0    02290          CALL SEND ;Byte an Drucker ausgeben
B06E 0E31      02300          LD C,49
B070 CDC1B0    02310          CALL SEND ;ESC 1  = 7/72 Zoll Zeilenabstand
B073 0E1B      02320          LD C,27
B075 CDC1B0    02330          CALL SEND
B078 0E2A      02340          LD C,42
B07A CDC1B0    02350          CALL SEND
B07D 0E04      02360          LD C,4
B07F CDC1B0    02370          CALL SEND ;ESC * CHR$(4)  = 640 Punkte je Zeile
B082 D9        02380          EXX
B083 7B        02390          LD A,E ;Vergroesserung holen
B084 D9        02400          EXX
B085 47        02410          LD B,A ;und nach B
```

```
B0BF E1        02790          POP HL ;Register zurueck
B0C0 C9        02800          RET ;fertig
               02810 ;
               02820 ;
               02830 ; Register C an Drucker ausgeben
B0C1 CD2985    02840 SEND     CALL BUSY ;Drucker empfangsbereit?
B0C4 20FB      02850          JR NZ,SEND ;nein, warten
B0C6 C33C85    02860          JP OUTPUT ;Register C ausgeben
               02870 ;
               02880 ;
               02890 ;
               02900 ; Initialisierung
B0C9 210080    02910 START    LD HL,ENTRY ;USR-Adresse auf
B0CC 22BE40    02920          LD (USRADR),HL ;dieses Programm setzen
B0CF C36600    02930          JP 66H ;nach 'READY'
               02940 ;
               02950 ;
B0C9           02960          END START
00000 TOTAL ERRORS
```

```
AGAIN   B029        OUTPUT  053C
BUFFER  41E8        PDOTS   0008
BUFLEN  00A0        PLUS    B0BC
BUSY    0529        PMASK   0080
CBYTES  0028        PRINT   B05F
CDOTS   0004        RAM     4800
CLEAR   B0BA        REPEAT  B09F
CLINES  0060        SEND    B0C1
CLRBUF  B0B2        START   B0C9
CMASK   00C0        USRADR  40BE
ENTRY   B000
FCERR   1E4A
LOOP    B0A0
NODOT   B043
NXBYTE  B034
NXDOT   B036
NXLINE  B01F
```