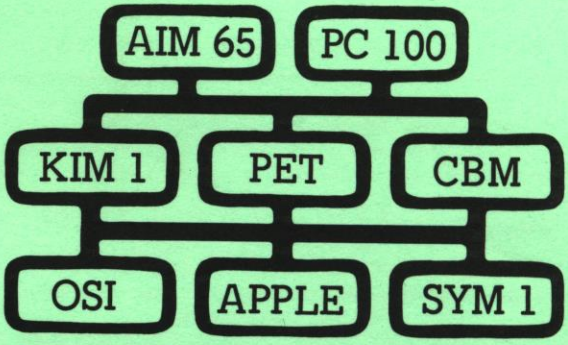
65xx MICRO MAG

COMPUTING · SOFTWARE · HOBBY

DM 9,50 Nr.32 August 1983



Inhaltsverzeichnis

CBM 710 !

DUO Plott Interface

für MX80 F/T und
ITOH 8510
und COMMODORE-Rechner 30/40/80

Paralleles IEEE 488 - Interface, genormter IEEE-Stecker und Kabel
kompletter Zeichensatz des CBM-Computers
zwei Geräteadressen für Groß-/Grafikmodus und Textmodus
alle Funktionen der Drucker bleiben erhalten, Floppy-Kompatibilität
Deutsche Umlaute, ß und Paragraph
Problemloser Einbau, inklusive deutsches EPSON-Handbuch
Komplettpreis einschl. Kabel, CBM-Grafiksatz und Handbuch incl. MWSt
für EPSON DM 298,--
für ITOH DM 298,--

Umrüstsatz MX80 F/T auf MX80 Typ 3

Aus Ihrem EPSON MX-80 F/T wird der MX-80 Typ 3
Einzelnadelansteuerung, erweiterter Befehlssatz, Elite-Schrift
Preis inkl. MWSt DM 98,--
Komplettpreis Interface, Kabel, Handbuch und Umrüstsatz inkl. MWSt DM 450,--

Für COMMODORE

Deutsche Tastatur mit Original-Tastensätzen (keine Aufkleber)
inkl. deutschem Zeichengenerator
für CBM 8032 DM 98,--
für CBM 8032 und 8096 DM 98,--
nur 8096, ohne Tasten DM 98,--

Speichererweiterung auf 96 K

LOS-kompatibel, inkl. MWSt DM 798,--

ISAM-Routinen
Datenhaltungsprogramm, Indexed Sequential Access Method
siehe Besprechung in Heft 23 des 65xx MICRO MAG, **DM 298,--**

Drucker:

RX 80 DM 998,-- + MWSt

ITOH 8510 DM 1495,-- + MWSt

EPSON-Druck-Computer FX 80 DM 1495,-- + MWSt

Stellberg Computer-Systeme

COMMODORE EPSON C. ITOH SOFTWARE INTERFACE

Blindenaaf 36 5063 Overath Tel.: 022 06 - 66 44

Commodore CBM 710

Seit einigen Wochen werden in der Klasse der Personal Computer die ersten Systeme der neuen Serien 700 und 600 ausgeliefert. Der wesentliche Unterschied zwischen beiden besteht darin, daß die Platinen ein unterschiedliches Layout haben und daß die Serie 700 einen eigenen entspiegelten Bildschirmmonitor mit 31 cm Diagonale hat. Letztere fordert einen vom Benutzer bereitgestellten Monitor. Die Festwertspeicher werden noch als EPROMs geliefert und noch nicht als ROMs. Dabei gibt es zwischen beiden Serien wohl auch noch graduelle Unterschiede. - Gegenüber den bisherigen 8000er Modellen ergeben sich erhebliche Leistungssteigerungen, wie noch zu zeigen sein wird.

Das Äußere

Das Gehäuse der Serie 700 gleicht dem stromlinienförmigen der SK-8000-Geräte. Es ist mit abgerundeten Ecken und Kanten aus stabilem Kunststoff geformt. Mit vorgelegter Tastatur sind die Maße ca. 45x60x45 cm (LTH). Die Tastatur ist mit dem Computer über Spiralkabel und Stecker verbunden und somit frei beweglich. Der Bildschirm kann bewegt werden: um 90 Grad in der Horizontalen und ist um 30 Grad neigbar, so daß Tastatur und Bildschirm ergonomischen Gesichtspunkten für eine wechselnde und günstige Arbeitshaltung entsprechen. Das Monitorbild steht absolut ruhig, die Buchstaben sind aus einer 9x14 Punktmatrix deutlich gebildet, es werden 24 Zeilen mit 80 Zeichen dargestellt, so daß sich eine angenehme Lesbarkeit ergibt, die Zeichen haben sogar Seriphen (ausgeformte kleine Spitzen).

Hardware und Interfaces

Bei Abfassung des Berichtes lagen hier keine Datenblätter und kein Schaltplan vor, so daß einige Angaben ggfs. später ergänzt werden müssen. Herz der Maschine ist die CPU 6509 mit 2 MHZ Takt, 8 Bit Datenbus und 20 Adreßbuspins. Der Adreßbus ist DMA-fähig, d.h. die Pins gehen beim Anhalten in den hochohmigen Zustand (Tri-State). Dies ist ein deutlicher Unterschied zur bisherigen CPU 6502. Die DMA-Fähigkeit erlaubt damit auch einen Co-Prozessor vom Typ Z80 oder 8088 um z.B. Programme unter CP/M zu benutzen. Dieser Punkt bedarf aber noch weiterer Aufklärung.

Wie aber, wenn die CPU mit ihren Registern eigentlich nur 64 KB adressieren kann? Der Trick besteht darin, daß die CPU die in Zelle 00 (Zero Page) eingeschriebenen niederen 4 Bit auf den Adreßbus aussendet, wenn sie Instruktionen holt. Wenn sie dagegen den Datenspeicher anspricht, sendet sie die unteren 4 Bit aus der Speicherzelle 01 auf den Adreßbus. Insgesamt kann die CPU damit 1 Megabyte Speicher adressieren. Beim CBM 710 liegt die Bank für BASIC, Betriebssystem und E/A in der Bank hex 0F.

Neben den Festwertspeichern von 24 KB finden wir bei diesem Modell einen RAM-Speicher von 128 KB (späteres Modell 730: 256 KB). Folgende Interfacebausteine sind im Bereich ab Adresse D800 dekodiert:

> 6545 CRT-Controller (der das 2 KB Video-RAM ab D000 zur Anzeige bringt), 6851 SID Sound Interface Device (Synthesizer mit 3 Stimmen, Hüllkurvenbeeinflussung unf Frequenzfiltern - wie beim Computer VC64), 6526 und 6525 Portbausteine für IEEE488 Interfacebus, Anwenderport und Tastaturanschluß. Eine ACIA 6551 besorgt über eine RS232C-Schnittstelle den seriellen Datenverkehr bis 9600 Baud, der befehlsmäßig auch in das BASIC eingebunden ist.

Auf der Rückseite des Gehäuses befinden sich die Buchse für die Netzschnur, der Netzschalter, ein Reset-Knopf (der den Rechner ohne Programmzerstörung zurückholt), die übliche Kontaktleiste für den IEEE488-Bus, eine 25-polige Buchse für die RS232-Schnittstelle, eine Audio-Buchse für den Ausgang des SID, eine Kontaktleiste für Datasette sowie der 30-polige Cartridge oder User-Connector. Nach der Dokumentation findet sich im Inneren des Gehäuses ein 40-poliger Anschluß

für einen Co-Prozessor (Z80 oder 8088) und ein 60-poliger Erweiterungsanschluß. Für die drei letztgenannten Anschlüsse sind die Signale zwar bezeichnet, ihre Bedeutung und Verwertung muß aber noch aufgeklärt werden.

Tastatur mit Zehnerblock und Funktionstasten

Die ersten Geräte wurden mit ASCII-Tastaturen ausgeliefert, später soll auch eine DIN-Ausführung erhältlich sein. Wir finden 5 Tastenfelder:

Das Haupttastenfeld entspricht einer Schreibmaschinentastatur. Es hat 48 Zeichentasten, 2x Shift, Shift Lock, Escape, CTRL, TAB und Delete/Insert und eine Commodore-Taste, mit der sich Abläufe so unterbrechen lassen, daß man zum Wiederstart nicht mit CONT arbeiten muß, sondern irgendeine andere Taste drückt.. Sie ist eine große Hilfe z.B. beim LIST.

Rechts auf dem Keyboard finden wir einen 'Zehnerblock', der um die Tasten 00 (doppelte Null), PRINT, Dezimalpunkt, die vier arithmetischen Operatoren, 2 Vorzeichentasten und die ENTER-Taste (wie CR) erweitert ist. Die Taste CE (Clear Entry) wirkt auf die letzte Eingabe. War sie eine Zahl (mit Dezimalpunkt), so wird sie mit einmaligem Tastendruck mit allen Ziffern gelöscht. Auf Buchstaben wirkt CE wie die Taste DELETE einzelzeichenweise. Der 'Zehnerblock' ist mit seinen 19 Tasten für den schnellen Tischrechnermodus sehr komfortabel.

Über dem Zehnerblock liegen 4 Tasten mit CLR/HOME (für Cursor und Bildschirmlöschung), OFF/REV (inverse Zeichendarstellung) NORM/GRAPH (Zeichensatzumschaltung jetzt nicht mehr mit POKE) und RUN/STOP.

Über dem alphabetischen Feld befinden sich zwei weitere Tastenfelder: Rechts 4 Kontrolltasten für die Cursorbewegungen und links 10 Funktionstasten, die mit Texten bis 160 Zeichen oder vollständigen Statements einschließlich dem auslösenden Carriage Return belegt werden können. Bei Betätigen der Shift- und Funktionstasten stehen 10 weitere Funktionen zur Verfügung. Auf den Gebrauch dieser Tasten werden wir an anderer Stelle eingehen.

Alle 95 Tasten lassen sich leicht und sicher betätigen und sind mit automatischem REPEAT ausgestattet. Es gibt kein hohles Klappern wie bei früheren CBM-Tastaturen. Das Bedienerinterface Tastatur darf wegen dieser Ausstattung und wegen der freien Bewegleichkeit als komfortabel und gelungen bezeichnet werden.

Commodore BASIC 128, V4.8

Der Rechner meldet sich beim Einschalten mit dieser Überschrift. Dieses BASIC und das Betriebssystem bieten zahlreiche Verbesserungen gegenüber dem bereits bewährten und beliebten BASIC 4. An dieser Stelle wird zunächst nur im Überblick auf die Neuerungen hingewiesen. Ein besonderer Artikel in diesem Heft wird den Gebrauch der neuen Befehle darstellen, zumal das schmale Handbuch zum Rechner den Benutzer hier nicht ausreichend informiert.

Die neuen Befehle und Systemvariablen sind (zunächst für die Fehlerbehandlung):

TRAP n Definition der Fehlerbehandlungsroutine mit Zeile n

DISPOSE mit RETURN oder NEXT. Nach Fehlereintritt zum Bereinigen des Datenstacks, wenn der Fehler nach GOSUB oder zwischen FOR .. NEXT eintrat.

EL Variable enthält Zeilennummer, in der der Fehler eintrat.

ER Variable mit der Fehlernummer.

ERR$(ER) Zuweisung des Fehlertextes aus der Fehlernummer.

RESUME Programmfortsetzung nach Fehlerbehandlung.

DELETE (von/bis/Zeilennummer), Editierkommando

INSTR, Instringfunktion

PRINT USING Ausgabeformatierung

PRINT /# USING Ausgabeformatierung für logische Datei Nr.

PUDEF Umdefinition für PRINT USING der Füll- und Währungszeichen, des Dezimalkommas und der Tausenderabteilung.

RESTORE (Zeilennummer) Gezieltes Umsetzen des DATA-Pointers in einem Programm.

IF THEN : ELSE Zweiwegeverzweigung innerhalb einer Programmzeile.

BANK n Umschaltung auf die Datenbank n für die Befehle PEEK, POKE, BLOAD, BSAVE

BLOAD, BSAVE mit Optionen: Laden und Speichern eines binären Files in eine Speicherbank (auch Segment genannt), mit oder ohne Ladeoffset (Verschiebung), bzw. Abspeichern von .. bis aus einer Bank.

DLOAD und DSAVE sind um die Option bereichert, daß Programme an die alte Stelle oder mit Umrechnung an den üblichen Beginn des BASIC-Speichers geladen werden können.

INPUT, INPUT # Es können bis zu 160 Zeichen eingelesen werden.

OPEN Das Statement ist um Parameterübergaben für die Eröffnung eines RS232-Kanals erweitert.

FRE(n) Mit dem Argument n wird jetzt die zu untersuchende Speicherbank bezeichnet.

SAVE, LOAD Der Betrieb einer Datasette ist (trotz Kontaktleiste) im System des Autors nicht mehr unterstützt.

PEEK und POKE wirken auf die mit BANK eingestellte Speicherbank.

TI$ Die Zeitvariable ist jetzt 7-stellig, es werden auch 1/10 Sekunden angezeigt.

Der Screen Editor

Zusammen mit der Vortaste ESC können die 26 Buchstaben für Editierfunktionen und für das Arbeiten mit Bildschirmfenstern benutzt werden, was eine Erweiterung der bei der Serie 8000 gegebenen Möglichkeiten bedeutet. Wir haben jetzt folgende Funktionen:

Automatischer Insert-Modus/Ende desselben
Set TOP/Set BOTTOM
Insert/Delete Line
Flashing/Non Flashing Cursor
Enable/Disable Bell
Move to Start/to End of Line
Enable/Disable Scrolling
Normal/Inverse Screen
Erase to Start/to End of Line
Cancel Insert, Quote and Reverse Mode
Cancel Escape Sequence
Voller Cursor/Unterstreichungs-Cursor
Scroll Up/Down
Normaler/alternativer Zeichensatz

Dokumentation

Den bisherigen Lieferungen liegt nur ein 113-seitiges kleines englisches Handbuch bei: Commodore 500/600/700 Series User's Guide. Es erklärt die Inbetriebnahme, Besonderheiten der Tastatur, Laden und Speichern von Programmen, Liste der BASIC-Befehle. Enthalten sind verschiedene Übersichtstabellen und Anschlußbelegungen. Bei den neuen Befehlen wird vermißt, daß ihre vollen Möglichkeiten und die Syntax nicht immer ausreichend dargestellt sind, weil es offensichtlich wieder einmal einfacher war, einen neuen Computer in die Serie zu schicken als seine Dokumentation auf's Papier zu bringen.

Ab Anfang August wird von hier aus und von einigen Händlern eine mehr als 300-seitige ausführliche Dokumentation beziehbar sein, das ROM-Listing für die Serie 700 und 600 als wieder

assemblierfähiger kommentierter Assembler-Quelltext mit Original CBM-Labels, der Belegung der Zero Page, des weiteren RAM und des I/O-Bereiches, zusammen mit einer Cross-Referenzliste (s. Anzeige). Es wird viele notwendige Aufklärungen geben. Weitere Informationen müssen wohl weiterhin zunächst von engagierten Betreibern beigesteuert werden. Die Leser sind ausdrücklich gebeten, ihre Beobachtungen mitzuteilen, denn der CBM 710 wird von hier aus weiter betreut werden.

Zusammenfassung

Der CBM 710 ist ein für professionellen Einsatz ausgelegtes und empfehlenswertes System mit gefälligem Gehäuse, einem einwandfreien Monitorbild (hohe Zeichenauflösung in einer 9x14 Punktmatrix) und einer komfortablen Tastatur als Hauptbedienungseinheiten.

Das RS 232c-Interface ist ein echter Gewinn für die an Bedeutung gewinnende Datenübertragung zwischen Computern in Netzen. Es ist in das BASIC eingebunden. Das erweiterte BASIC 4 bringt viele Befehle, die die Programmierung einfacher machen: IF THEN : ELSE, RESTORE, PRINT USING, INSTRING und die Befehle zur Fehlerbehandlung. Es ist leider noch immer kein dezimal rechnendes BASIC, so daß es auch weiterhin die bekannten Rundungsfehler produziert.

Der Rechner ist für Erweiterungen ausgelegt, für die aber noch Information fehlt: Anschluß eines Co-Prozessors, um die unter CP/M zu ladenden Sprachen und Anwenderprogramme zu erschließen, Einbau von Floppies in das Gehäuse, Speichererweiterungen auf den vollen Adressierbereich der CPU 6509.

In der Summe darf das neue System des hiesigen Marktführers als gelungen bezeichnet werden, auch wenn noch einige Wünsche offen geblieben sind. Bei einem Preis knapp unter DM 3000 zuzüglich Mehrwertsteuer bietet der Computer ein hohes Preis-Leistungsverhältnis, das vom Wettbewerb so schnell nicht eingeholt werden kann. Rückschauend muß man feststellen, daß Commodore seit dem Erscheinen des PET 2000 zum Ende des Jahres 1978 (als die eingeführten großen Firmen über das 'Spielzeug' lächelten) einen weiten Weg gegangen ist. Es folgten in dieser Klasse die Serien 3000, 4000 und 8000 u.a., begleitet von einem zunehmenden Angebot steckerfertiger Peripherie. Wenn auch die Entwicklung schnell weiterschreitet, heute kann man sagen, daß der 710 der erste richtige 'Profi' aus diesem Hause ist.

R.L.

BASIC 4: Die neuen Befehle

Wie bei der Besprechung des CBM 710 schon erwähnt, sind viele neue Befehle des erweiterten BASIC 4 für die Anwendung nicht ausreichend dokumentiert. Daher der nachfolgende Überblick (ohne Gewähr):

DELETE Zeilen im BASIC-Quelltext. Anwendung analog zum Befehl LIST.

INSTR abzusuchende Zeichenkette/ gesuchter String/ab Zeichenstelle
Instringfunktion, sie liefert die Zeichenstelle ab, ab der der gesuchte String enthalten ist. Ist er nicht enthalten, so wird Null abgeliefert. Im nachfolgenden Beispiel ergibt sich A=3 und B=0 (weil hinten im String angesetzt wurde).

```
10 A$="ABCDE"              :REM ABZUSUCHENDER STRING
20 A=INSTR(A$,"CD",1)      :REM STRING WIRD GEFUNDEN
30 B=INSTR(A$,"CD",4)      :REM WIRD NICHT GEFUNDEN
40 PRINTA,B
50 :
60 DATA 1,2,3              :REM ZU UEBERGEHEN
70 RESTORE 70              :POSITIONIERUNG VOR GEW. ZEILE 80
80 DATA 4,5,6
90 READ A:PRINTA           :REM ES WIRD 4 AUSGEGEBEN
```

IF ... THEN : ELSE, bedingte Zweiwegeverzweigung. Das vollständige Statement muß in einer Zeile stehen, ELSE darf also nicht in einer Folgezeile stehen. Man beachte den not-

wendigen Doppelpunkt vor ELSE. Das Konstrukt kann auch mehrfach geschachtelt werden

RESTORE Zeilennummer: Es muß nicht unbedingt auf die gewünschte DATA-Zeile positioniert werden. Ein Beispiel ist ab Zeile 60 in der vorstehenden Liste enthalten.

KEY n, "Text". Mit dem Schlüsselwort KEY wird den Funktionstasten ein Text zugewiesen, der in Gänsefüßchen stehen muß. Er kann 160 Zeilen lang sein. Sofern der Text eine wie üblich formulierte Anweisungsfolge ist, kann man durch Anhängen von CHR$(13) (entspricht der Betätigung der Return-Taste) diese auf Tastendruck zur sofortigen Ausführung bringen. Steuerzeichen müssen also als CHR$( ) geschrieben werden. n von 1-10 für Steuertasten ohne Shift, von 11-20 mit Shift-Taste. Die ersten 10 Texte werden bereits bei der Initialisierung des Systems belegt mit PRINT, LIST, DLOAD", DSAVE", DOPEN, DCLOSE, COPY, DIRECTORY, COPY und CHR$(.

PRINT USING "Zeichenkette" oder Stringvariable; Variable oder Direktwert
PRINT # (logisches File),USING "Zeichenkette" o. Stringvariable; Variable o. Direktwert
Befehl zur Ausgabeformatierung. Es können grundsätzlich Zahlen oder Strings formatiert werden, die Wirkung ist etwas unterschiedlich und sollte ggfs. etwas erprobt werden. Die Philosophie dieses Befehles besteht darin, für die Ausgabe eine Maske vorzuhalten, die mit stellvertretenden Zeichen (#), ggfs. einem Währungssymbol (auch gleitend vor die Ziffern gesetzt), Füllzeichen vor gültigen Ziffern/Zeichen (z.B. Schutzsterne) und für Zahlen mit der Position des Dezimalpunktes/-kommas aufgebaut ist. Diese Maske, zuvor als 'Zeichenkette' oder 'Stringvariable' bezeichnet, wird als Formatstring 'benutzt' (engl. use). Die Formatierungsmöglichkeiten sind außerordentlich vielseitig, wie die nachfolgenden Beispiele zeigen. Mit der weiteren BASIC-Anweisung PUDEF (Print Using Definition) ist es ferner möglich, die vier Standardzeichen für Füller, Währungszeichen, Dezimalpunkt und Abteilung der Tausendergruppen (in Zahlen) nach den Wünschen des Benutzers umzudefinieren. Von den reservierten Maskenzeichen abgesehen, kann der Formatierungsstring auch anderen Text enthalten, der bei PRINT USING mit in die Ausgabe gelangt. Dieser Befehl ist in einer Form implementiert, wie man sie auch in Geräten der mittleren und größeren Datentechnik findet. Im Zweifel kann man dort in den Handbüchern einmal nachschlagen.

PUDEF Diese Anweisung beeinflußt die Ausgabe im PRINT USING, und zwar für das zu verwendende Füllzeichen (Standard ist der Space), für das Trennzeichen zwischen Tausender-Zahlengruppen (Standard: Komma), für den Dezimalpunkt (Standard: Punkt) und für das Währungszeichen (Standard: das Dollarzeichen). PUDEF erwartet in dieser genauen Reihenfolge ein Argument mit bis zu 4 Zeichen, die zwischen Gänsefüßchen eingeschlossen sind oder die als CHR$( ) erklärt werden.

Für die vielen Möglichkeiten von PRINT USING in Verbindung auch mit PUDEF hier nun einige Bespiele. Man beachte die Kommentare in der Programmliste.

```
100 OPEN4,4:PRINT#4,CHR$(27)CHR$(71)    :REM DOPPELTER DRUCK
110 A$="###,###.##"                      :REM MASKE

118 PRINT#4,"ZEILE 120"
120 PRINT#4, USING A$;1234.567           :REM ES WIRD GERUNDET
125 :
130 A$=".###"                            :REM KEINE FUEHRENDE NULL
135 PRINT#4,"ZEILE 140"
140 PRINT#4, USING A$;0.1234

150 A$="#.###"                           :REM MIT NULL
160 PRINT#4,"ZEILE 170"
170 PRINT#4, USING A$;.1234
180 PRINT#4,"ZEILE 190"
190 PRINT#4, USING A$;555.66             :REM ERZEUGT OVERFLOW, STERNE
```

```
 200 A$="#####,###"                          :REM GLEITENDES $-ZEICHEN, RUND
 205 PRINT#4,"ZEILE 210"
 210 PRINT#4, USING A$;12.56

 220 PUDEF "*,"                              :REM DEUTSCHE ZAHLENSCHREIBWEISE
 230 PRINT#4,"ZEILE 240"                     :REM MIT SCHUTZSTERN
 240 PRINT#4, USING A$;12.789

 250 A$="$###,###.##-"                        :REM FESTER $, VORZEICHEN
 260 PRINT#4,"ZEILE 270"
 270 PRINT#4, USING A$;-123456.77
 280 PRINT#4,"ZEILE 290"                      :REM MIT SCHUTZSTERNEN
 290 PRINT#4, USING A$;12

 300 A$="###,###.## DEUTSCHE MARK"            :REM MIT TEXT VERMISCHT
 310 PRINT#4,"ZEILE 320"
 320 PRINT#4, USING A$;1234.55

 330 A$="########## DEUTSCHE MARK"            :REM TEXTMASKE
 334 PRINT#4,"ZEILE 336"
 336 PRINT#4, USING A$;"ZEHN"

 340 A$="##########> DM"                      :REM RECHTSBUENDIG DRUCKEN
 350 PRINT#4,"ZEILE 360"
 360 PRINT#4, USING A$;"HUNDERT"

 370 A$="LINKS =######################## RECHTS" :REM ZENTRIERUNG DURCH =
 380 PRINT#4,"ZEILE 390"
 390 PRINT#4, USING A$;"ZENTRIERT"

 410 A$="##.##^^^^"                           :REM EXPONENTIALSCHREIBWEISE
 420 PRINT#4,"ZEILEN AB 430"
 430 PRINT#4, USING A$;123456
 440 PRINT#4, USING A$;.0000456789
 450 PRINT#4, USING A$;CHR$(13)
 460 CLOSE4
READY
```

```
  ZEILE 120
    1,234.57
  ZEILE 140
  .123
  ZEILE 170
  0.123
  ZEILE 190
  *****
  ZEILE 210
        $13
  ZEILE 240
  ******$13
  ZEILE 270
  $123.456,77-
  ZEILE 290
  ******12,00*
  ZEILE 320
  **1.234,55 DEUTSCHE MARK
```

```
ZEILE 336
ZEHN       DEUTSCHE MARK
ZEILE 360
     HUNDERT DM
ZEILE 390
LINKS          ZENTRIERT          RECHTS
ZEILEN AB 430
12,35E+04
45,68E-06
```

BANK n Umschaltung der Speicherbank in Zelle 01 für Befehle wie PEEK und POKE auf Bank n. Im Commodore 710 ist Nr. 15 die Systembank mit BASIC und Betriebssystem ab hex 8000. Im Bereich von 0002 bis 07FF liegt RAM, das vom System benutzt wird (gleichwohl ergibt FRE(0)=0). Bank 1 enthält Programmtext und Variable. Am Ende der Bank 2 werden Strings dynamisch abgespeichert, etwa ab FA00. Darüber liegen offensichtlich die Texte für die KEYs.

V n Mit SYS4 kann man in den TIM-Monitor gelangen. Mit V 01 und M xxxx yyyy kann man nun wie gewohnt Speicherzellen aufschlagen, in diesem Beispiel in Bank 1. n bezeichnet also die Bank.

BLOAD und BSAVE Laden und Abspeichern binärer Files. Die BANK kann im Kommando erklärt werden, fehlt der Begriff ON B n (nach Bank n), so gilt die Verfügung des letzten BANK-Statements. Es ist ein verschiebliches Laden möglich. Ohne Angabe eines Offsets hinter 'P' wird ab Speicherstelle 02 geladen. Beim Speichern ist eine Bereichsangabe von .. bis als Parameter hinter den 'P' möglich, fehlt sie, so wird bis zum Ende der Bank abgespeichert. Die Syntax für diese Befehle ist:

BLOAD "Name" ON Bn, Pxxxx n=Bank, xxxx=Verschiebung (dezimal)
BSAVE "Name" ON Bn, Pxxxx TO Pyyyy xxxx=von ... yyyy= bis Speicherstelle (dez.)

Bei den Befehlen DLOAD und DSAVE werden die Programme von anderen CBM-Rechnern auf die neue Speicherbelegung umgerechnet und bleiben damit lauffähig. Angeblich soll auch ein Laden an alte Stelle möglich sein. Das konnte hier aber noch nicht verifiziert werden.

USR Ausführung eines vom Anwender vektorierten Maschinenprogramms. Der Vektor ist in Bank 15 in den Zellen 03 und 04 zu hinterlegen (POKE), der Gleitkomma-Akku liegt ab Adresse hex 71. Vermutlich wird ein zweistufiges Vorgehen notwendig sein, wenn das Anwenderprogramm in einer anderen Bank liegt. Man muß dann wohl in Bank 15 noch die Bank umschalten, der Folgebefehl wird dann in der neuen Bank gesucht.

OPEN log. File-Nummer, 2, Kommando, Open-String
Dieses ist das OPEN-Kommando für den RS232-Kanal. Der Parameter 'Kommando' bestimmt die Richtung (Eingabe, Ausgabe, bidirektional) der Open-String besteht aus 4 Byte für die Betriebsparameter von denen nur 2 belegt sind, gleichwohl müssen 4 Byte enthalten sein, so daß man mit 'dummies' auffüllt. Es ist mit den Kommandos 129, 130 und 131 eine Zeichenumwandlung von CBM-ASCII auf normales ASCII und umgekehrt möglich. Für Kommando und Open-String gelten folgende Tabellen:

RS232-Kommandos für die Datenrichtung

Wert	Bedeutung
1	Open Ausgabekanal
2	Open Eingabekanal
3	Open Ein-/Ausgabekanal
129	Open Ausgabekanal, Umwandlung CBM zu ASCII
130	Open Eingabekanal, Umwandlung ASCII zu CBM
131	Open bidirektionaler Kanal mit Umwandlung in beiden Richtungen

Beim Senden von Zeichen, die nicht druckbar sind, sollte auf eine Umwandlung CBM ZU ASCII verzichtet werden um die Übermittlung nicht zu stören.

Bitmuster im Open-String

Das erste Byte bestimmt die Baudrate,
die Clock des Receivers und die Wortlänge wie folgt:

Bit	Bedeutung	
0-3	Baudrate	
	Bitmuster	Baudrate
	0000	16x externe Clock
	0001	50
	0010	75
	0011	109,82
	0100	134,58
	0101	150
	0110	300
	0111	600
	1000	1200
	1010	1800
	1011	2400
	1100	3600
	1101	4800
	1110	9600
	1111	19200
4	Bestimmung der Receiver-Clock	
	0= externe Clock	
	1= Baudraten-Generator	
5-6	Wortlänge	
	00	8 bit
	01	7 Bit
	10	6 Bit
	11	5 Bit
7	Zahl der Stop-Bits	
	0	1 Stop-Bit
	1	1 Stop-Bit bei Wortlänge 8 mit Parity
	1	1,5 Stop-Bits b. Wortlänge 5 ohne Parity
	1	2 Stop-Bits in allen anderen Fällen

Das zweite Byte im Open-String beinflußt
das Parity-Bit und den Echo-Modus

Bit	Bedeutung	
0-3	unbenutzt	
4	Echo-Modus	
	0 normale Übermittlung	
	1 Empfänger echot alle empfangenen Zeichen	
	Im Echomodus kann man keine Daten senden.	
7-5	Paritätskontrolle	
	000	ohne Parity-Bits beim Senden und Empfangen
	001	Odd Parity auf beiden Seiten
	011	Even Parity auf beiden Seiten
	101	Mit Mark-Bit, ohne Paritätserzeugung
	111	Space-Parity-Bit wird übermittelt, aber Paritätsprüfung ausgeschaltet.

Auf dem RS232-Kanal sollte mit GET# gearbeitet werden, nicht mit INPUT#, weil letzteres ein CR erwartet.

Wenn GET# leer zurückkommt, weil kein Zeichen auf dem Kanal ankam, dann ist die Variable ST (Status) in Bit 4 auf 1 gesetzt. Die Status-Bits haben für RS232 folgende Bedeutung:

Bit	Bedeutung
0	Parity-Fehler
1	Framing-Fehler
2	Empfänger durch zu schnellen Eingang überlaufen
3	nicht benutzt
4	leerer Eingabepuffer nach GET#
5	Data Carrier Detect-Fehler (unterbrochene Verbindung)
6	Data Set Ready-Fehler (DSR)
7	nicht benutzt.

Wie die nachfolgenden Beispiele zeigen, ist die Fehlerbehandlung im erweiterten BASIC 4 auf eine einfache und übersichtliche Art gelöst: Zentrales Statement ist TRAP n (n=Zeilennummer), mit dem auf die Zeile verwiesen wird, in der die Fehlerbehandlung beginnt. Zur Fehleruntersuchung stehen jetzt die Systemvariablen EL (Error Line, in der der Fehler vorkam) und ER (Error Number Fehlerart) zur Verfügung. Die Stringvariable kann wahlweise z.B. für die Anzeige mit dem Fehlertext gefüllt werden, der im anderen Fall auf dem Bildschirm erscheinen würde PRINT ERR$(ER). Nach einem Fehler zwischen FOR ... NEXT kann nach der Fehlerbehandlung mit dem Statement RESUME NEXT n (n=Zeilenummer) in der Schleife fortgefahren werden. Man kann aber auch das Kontrukt verlassen mit DISPOSE FOR, oder wenn es ein Unterprogramm war, in dem der Fehler auftrat, mit DISPOSE GOSUB. Durch das DISPOSE wird der BASIC-Stack von Müll befreit.

```
 100 OPEN4,4:PRINT#4,CHR$(27)CHR$(71)
 150 TRAP 500
 200 FORI=-3TO5
 250 PRINT#4, 88/I
 300 NEXT
 400 PRINT#4,CHR$(13):CLOSE4
 450 END
 500 DISPOSE FOR
 550 PRINT#4,"NULLSTELLE GEFUNDEN I= ";I
 600 PRINT#4,CHR$(13):CLOSE4
 650 END
READY.

-29.3333334
-44
-88
NULLSTELLE GEFUNDEN I=  0
```

```
 50 OPEN4,4
 100 TRAP 500
 110 FORI=-1TO1
 120 PRINT#4, 77/I
 130 NEXT
 135 END

 500 PRINTERR$(ER)
 510 RESUME NEXT 120

READY.
-77
DIVISION BY ZERO
 77
```

Dipl.-Ing. (FH) Rudolf Kirchgäßner, 6831 Brühl

CROSSREF für BASIC-Programme

Crossreferenz-Listen zeigen die Verwendung von Variablen und Sprungzielen innerhalb eines Programmes. Werden diese Listen durch BASIC-Programme erzeugt, ist entweder die Laufzeit des Programms sehr groß oder die Informationen sind nicht ausreichend, meistens trifft beides zu. Jim Butterfield entwickelte 1981 ein Programm namens XREF, das in punkto Schnelligkeit alle Wünsche übertrifft. Dieses Programm wurde vom Autor analysiert, kommentiert und erweitert. Nun werden Wertzuweisungen an Variable dargestellt, das Zeichen dafür ist '*'. Ebenfalls durch '*' markiert werden Verweise auf Zeilennummern bei Aufruf durch GOSUB. Weiterhin werden ausgegeben alle Funktionsvariablen sowie alle Zeilen, die POKE, PEEK, SYS oder WAIT-Befehle enthalten.

Programmbeschreibung

CROSSREF ist gedacht zur Erweiterung eines Tools. Deshalb ist die Steuerroutine kurz gehalten, und alle Label im Maschinenprogramm beginnen mit CR. CROSSREF belegt rund 1300 Byte im Speicher. Da das Testprogramm von Diskette gelesen wird, steht der restliche Speicher für Tabellenarbeit zur Verfügung. Die Verwaltung der Verweise geschieht in zwei Tabellen. Alle Einträge einer BASICZeile werden in eine Zeilentabelle aufgenommen, wobei doppelte Einträge verhindert werden. Die Zeilentabelle wird nach dem Lesen des Zeilenendes in die Referenztabelle eingemischt. Durch dieses Verfahren wird sowohl die Suche nach schon vorhandenen Einträgen als auch das Einsortieren neuer Verweise optimiert. Die Liste für ein 16K-BASIC-Programm wird bereits nach 60 Sekunden ausgedruckt!

Ebenso optimal ist die Strategie zur Verarbeitung der BASIC-Zeichen: Jedes Zeichen wird umgewertet in einer Codetabelle. Bit 5-7 des Codes regeln die *-Markierung. Die anderen Bits dienen zur indirekten Adressierung einer Arbeitstabelle. Die von dort erhaltenen Werte steuern die Bildung der Label und adressieren die für das nächste BASIC-Zeichen gültige Arbeitstabelle.

Literatur:

Dohmann, B.: Cross Reference List für BASIC-Variable (CBM), 65xx MICRO MAG, Heft 20, August 1981.
Kirchgäßner, R.: Crossreferenzliste für Sprünge und Variablen, Computer Journal, Jan/Feb 83.

```
******************
CBM 3032  11.04.83
******************

    1 '     XREF BY  JIM BUTTERFIELD
    2 '     UPDATE   RUDOLF KIRCHGAESSNER
    3 '              NIBELUNGENSTR.4
    4 '              6831 BRUEHL
    5 '
    6 S=1559:W=40:POKE 32768,1:IF PEEK(33792)<>1 THEN W=80
   10 OPEN 15,8,15:PRINT"<CLEAR><RVS>CROSSREF  V 1.0
   20 INPUT"<2DOWN>WELCHES PROGRAMM";P$:OPEN 1,8,2,P$+",P,R"
   30 INPUT#15,E,E$:IF E THEN PRINT E$:CLOSE 1:GOTO 20
   40 GET#1,X$,Y$:IF Y$<>CHR$(4) THEN GOSUB 100:GOTO 20
   50 IF X$<>"" AND X$<>CHR$(1) THEN GOSUB 100:GOTO 20
   60 INPUT"<DOWN>LISTE DRUCKEN? N<3LEFT>";X$:O=3:IF X$="J" THEN O=4:W=80
   70 SYS S:CLOSE 1:CLOSE 15:OPEN 4,O
   80 PRINT#4,"CROSS REFERENCE MAP"SPC(W-20-LEN(P$))P$
   90 POKE 195,W/8+1:SYS S+3:PRINT#4:CLOSE 4:END

  100 CLOSE 1:PRINT P$" IST KEIN BASIC-PROGRAMM":RETURN


532 BYTES
```

```
******************
CROSSREF  SEITE:01
******************
```

```
 CROSS REFERENCE MAP

   20:      30      40      50
  100:     40*     50*
:PEEK:      6
:POKE:      6      90
:SYS :     70      90
E    :     30*
E$   :     30*
D    :     60*     70
P$   :     20*     80     100
S    :      6*     70      90
W    :      6*     60*     80      90
X$   :     40*     50      60*
Y$   :     40*
```

Cross-Referenz
für vorstehendes BASIC-Programm

```
                0010 ; CROSSREF  V01  27.03.83
                0020
                0030             .BA $0617              ;ADRESS-PEGEL
                0040             .OS                    ;OBJECT-CODE SPEICHERN
                0050
                0060 ;      SYSTEM-ADRESSEN CBM BASIC3  (BASIC4)
                0070 ;      ===================================
                0080 STRADR      .DE $1F                ;STRINGADRESSE
                0090 STREND      .DE $2E                ;ENDE STRINGSPEICHER
                0100 FAC1        .DE $5E                ;FLOATING-AKKU1
                0110 KBDMAT      .DE $97                ;KEYBOARD-MATRIX
                0120 STACK       .DE $0100
                0130 CRLFBAS     .DE $C9E2              ;$BADF AUSGABE CRLF
                0140 STROUT      .DE $CA23              ;AUSGABE STRING
                0150 ADRFP       .DE $DB55              ;$CD7F ADRESSE -> FLOATING
                0160 FPSTR       .DE $DCE9              ;$CF93 FLOATINGP.-> STRING
                0170 SPACE       .DE $FDCD              ;$D531 AUSGABE SPACE
                0180 CHKIN       .DE $FFC6              ;CMD FUER INPUT
                0190 CKOUT       .DE $FFC9              ;CMD FUER OUTPUT
                0200 CLRCH       .DE $FFCC              ;STANDARD I/O
                0210 BSOUT       .DE $FFD2              ;AUSGABE ZEICHEN
                0220 STOPR       .DE $FFE1              ;BEI STOP -> READY
                0230 GETIN       .DE $FFE4              ;LESEN ZEICHEN
                0240
                0250 ;           PROGRAMMADRESSEN
                0260 ;           ================
                0270 CRNEWLABEL  .DE $16                ;BYTE 1-5: LABEL
                0280             ;                       BYTE 6-7: ZEIL# HIGH/LOW
                0290             ;                       BYTE 8 : * MARKIERUNG
                0300             ;          POINTER AUF:
                0310 CRREFPT     .DE $B4                ;ELEMENT IN REFERENZTAB
                0320 CRHILFPT    .DE $BC                ;  ''     IN   ''
                0330 CRZEILPT    .DE $BC                ;  ''     IN ZEILTAB
                0340 CRARBPT     .DE $D6                ;ARBEITSTABELLE
                0350
                0360 CRSTREND    .DE $BA                ;BEGINN FREIER SPEICHER
                0370 CRREFEND    .DE $BE                ;ENDE DER REFERENZTAB
                0380 CRZEILEND   .DE $C0                ;ENDE DER ZEILTAB
                0390 CRHIFE      .DE $DD                ;HILFSFELD
                0400 CRBASCODE   .DE $C2                ;MERKER FUER BASIC-CODE
                0410 CRZREF      .DE $C2                ;ZAEHLER FUER VERWEISE
                0420 CRREFMAX    .DE $C3                ;MAX.ANZAHL VERWEISE/ZEILE
                0430 ;=================================================================
0617- 4C 20 06  0440             JMP CRBEGINN           ;PROGRAMMSTART
                0450
061A- 4C 33 08  0460             JMP CRDRUCK            ;EINSPRUNG FUER DRUCKEN
                0470
061D- 4C CC FF  0480 CRENDE      JMP CLRCH              ;PROGRAMMENDE
                0490
0620- A2 01     0500 CRBEGINN    LDX #1                 ;LESEN VON KANAL 1
0622- 20 C6 FF  0510             JSR CHKIN
0625- 38        0520             SEC
0626- A5 2E     0530             LDA *STREND            ;FREIEN SPEICHER MERKEN
0628- 85 BA     0540             STA *CRSTREND
062A- 85 BE     0550             STA *CRREFEND
062C- E9 08     0560             SBC #8                 ;AUF NULL-EINTRAG STELLEN
062E- 85 B4     0570             STA *CRREFPT
```

```
0630- A4 2F     0580             LDY *STREND+1
0632- 84 BB     0590             STY *CRSTREND+1
0634- C8        0600             INY
0635- 84 BF     0610             STY *CRREFEND+1       ;1 PAGE FUER ZEILTAB
0637- 98        0620             TYA
0638- E9 00     0630             SBC #0
063A- 85 B5     0640             STA *CRREFPT+1
063C- A9 09     0650             LDA #H,CRARBTAB00
063E- 85 D7     0660             STA *CRARBPT+1
0640- A0 0F     0670             LDY #$0F              ;ANFANGSWERTE IN
0642- B9 DB 08  0680 CRNULL1     LDA CRNULLTAB,Y       ;REFTAB
0645- 91 B4     0690             STA (CRREFPT),Y
0647- 88        0700             DEY
0648- 10 F8     0710             BPL CRNULL1
064A- 20 B2 07  0720             JSR CRCLRLAB
064D- 20 E4 FF  0730 CRNEWLINE   JSR GETIN             ;LINKADRESSE
0650- 20 E4 FF  0740             JSR GETIN             ;     *
0653- F0 C8     0750             BEQ CRENDE            ;PROGRAMM-ENDE
0655- A5 BA     0760             LDA *CRSTREND         ;ZEILTABELLE
0657- 85 C0     0770             STA *CRZEILEND        ;AUF ANFANG STELLEN
0659- A5 BB     0780             LDA *CRSTREND+1
065B- 85 C1     0790             STA *CRZEILEND+1
065D- 20 E4 FF  0800             JSR GETIN             ;ZEILEN # LOW
0660- 85 1C     0810             STA *CRNEWLABEL+6
0662- 20 E4 FF  0820             JSR GETIN             ;    *      HIGH
0665- 85 1B     0830             STA *CRNEWLABEL+5
0667- A9 00     0840             LDA #0                ;WEICHE IN
0669- 85 D6     0850             STA *CRARBPT          ;AUSGANGSSTELLUNG
066B- F0 0D     0860             BEQ CRNEWBYTE1
                0870
066D- A9 20     0880 CRNEWBYTE   LDA #$20
066F- 24 C2     0890             BIT *CRBASCODE
0671- 30 0A     0900             BMI CRNEWBYTE2        ;BIT 7
0673- 70 0C     0910             BVS CRNEWBYTE3        ;BIT 6
0675- D0 03     0920             BNE CRNEWBYTE1        ;BIT 5
0677- A0 23     0930             LDY #'#               ;BLEIBT GOSUB
0679- 2C        0940             .BY $2C               ;BIT-BEFEHL
067A- A0 2A     0950 CRNEWBYTE1  LDY #'*
067C- 2C        0960             .BY $2C
067D- A0 20     0970 CRNEWBYTE2  LDY #'
067F- 84 1D     0980             STY *CRNEWLABEL+7
0681- 20 E4 FF  0990 CRNEWBYTE3  JSR GETIN             ;BASIC-ZEICHEN
0684- F0 15     1000             BEQ CRNEWBYTE5        ;ZEILENENDE
0686- C9 21     1010             CMP #$21              ;< = SPACE ?
0688- 90 F7     1020             BCC CRNEWBYTE3        ;UEBERLESEN
068A- C9 C3     1030             CMP #$C3              ;> PEEK ?
068C- B0 41     1040             BCS CRNEWCODE3
068E- C9 22     1050             CMP #$22              ;QUOTE ?
0690- F0 63     1060             BEQ CRNEWBYTE7
0692- C9 8F     1070             CMP #$8F              ;REM ?
0694- D0 14     1080             BNE CRNEWCODE
0696- 20 E4 FF  1090 CRNEWBYTE4  JSR GETIN
0699- D0 FB     1100             BNE CRNEWBYTE4
069B- A4 D6     1110 CRNEWBYTE5  LDY *CRARBPT          ;ZEILENENDE
069D- C0 24     1120             CPY #L,CRARBTAB24     ;LABEL GESPEICHERT ?
069F- 90 03     1130             BCC CRNEWBYTE6        ;NEIN
06A1- 20 3E 07  1140             JSR CRINSZTAB         ;EINFUEGEN
06A4- 20 BD 07  1150 CRNEWBYTE6  JSR CRMERGE           ;ZEILTAB -> REFTAB
06A7- 4C 4D 06  1160             JMP CRNEWLINE
                1170
06AA- AA        1180 CRNEWCODE   TAX
06AB- BD 27 09  1190 CRNEWCODE1  LDA CRCODETAB-33,X    ;BASICZEICHEN BEWERTEN
06AE- 85 C2     1200             STA *CRBASCODE
06B0- C9 1B     1210             CMP #$1B              ;SONDERBEFEHL ?
06B2- B0 1F     1220             BCS CRNEWCODE5        ;NEIN
06B4- C9 02     1230             CMP #$02              ;FN ?
06B6- F0 53     1240             BEQ CRNEWCODE9
06B8- A8        1250             TAY                   ;ALSO PEEK, POKE ...
06B9- A2 04     1260             LDX #4
06BB- B9 EB 08  1270 CRNEWCODE2  LDA CRLABTEXT,Y       ;TEXT -> LABEL
06BE- 95 16     1280             STA *CRNEWLABEL,X
```

```
06C0- 88        1290              DEY
06C1- CA        1300              DEX
06C2- D0 F7     1310              BNE CRNEWCODE2
06C4- A9 3A     1320              LDA #':
06C6- 85 16     1330              STA *CRNEWLABEL
06C8- A9 20     1340              LDA #' 
06CA- 85 1D     1350              STA *CRNEWLABEL+7
06CC- 20 3E 07  1360              JSR CRINSZTAB         ;LABEL UEBERNEHMEN
06CF- A9 80     1370 CRNEWCODE3   LDA #$80
06D1- 85 C2     1380 CRNEWCODE4   STA *CRBASCODE
06D3- 29 0F     1390 CRNEWCODE5   AND #$0F
06D5- A8        1400              TAY
06D6- B1 D6     1410              LDA (CRARBPT),Y       ;NEUE WEICHENSTELLUNG
06D8- 10 07     1420              BPL CRNEWCODE6
06DA- 29 7F     1430              AND #$7F
06DC- 48        1440              PHA
06DD- 20 17 07  1450              JSR CRLABSTORE        ;VAR ODER SPRUNGZIEL
06E0- 68        1460              PLA
06E1- A4 D6     1470 CRNEWCODE6   LDY *CRARBPT
06E3- 85 D6     1480              STA *CRARBPT          ;WEICHE STELLEN
06E5- C4 D6     1490              CPY *CRARBPT          ;MIT ALTEM WEICHENWERT
06E7- F0 17     1500              BEQ CRNEWCODE8        ;VERGLEICHEN
06E9- 90 82     1510              BCC CRNEWBYTE         ;KLEINER
06EB- C0 24     1520              CPY #L,CRARBTAB24     ;LABEL GESPEICHERT ?
06ED- 90 03     1530              BCC CRNEWCODE7        ;NEIN
06EF- 20 3E 07  1540              JSR CRINSZTAB         ;EINFUEGEN
06F2- 4C 6D 06  1550 CRNEWCODE7   JMP CRNEWBYTE
                1560
06F5- 20 E4 FF  1570 CRNEWBYTE7   JSR GETIN             ;INNERHALB QUOTE
06F8- F0 A1     1580              BEQ CRNEWBYTE5        ;ZEILENENDE
06FA- C9 22     1590              CMP #$22              ;QUOTE ?
06FC- D0 F7     1600              BNE CRNEWBYTE7        ;NEIN
06FE- F0 B1     1610              BEQ CRNEWBYTE3
                1620
0700- C9 3F     1630 CRNEWCODE8   CMP #L,CRARBTAB3F      ;Z.B. PRINT A$B$
0702- D0 EE     1640              BNE CRNEWCODE7
0704- 20 3E 07  1650              JSR CRINSZTAB         ;1.VAR EINFUEGEN
0707- 84 D6     1660              STY *CRARBPT          ;YR=0
0709- F0 A0     1670              BEQ CRNEWCODE1        ;2.VAR BEARBEITEN
                1680
070B- A0 46     1690 CRNEWCODE9   LDY #'F                ;FN-TEXT SCHREIBEN
070D- 84 16     1700              STY *CRNEWLABEL
070F- A0 4E     1710              LDY #'N
0711- 84 17     1720              STY *CRNEWLABEL+1
0713- A9 40     1730              LDA #$40
0715- D0 BA     1740              BNE CRNEWCODE4
                1750
0717- A8        1760 CRLABSTORE   TAY
0718- 8A        1770              TXA                   ;BASICZEICHEN
0719- A2 00     1780              LDX #0
071B- C0 24     1790              CPY #L,CRARBTAB24     ;VARIABLE ODER
071D- F0 09     1800              BEQ CRLABSTOR2        ;SPRUNGZIEL
071F- B4 16     1810 CRLABSTOR1   LDY *CRNEWLABEL,X     ;SUCHEN BIS
0721- C0 20     1820              CPY #'                ;ZUM 1.SPACE
0723- F0 16     1830              BEQ CRLABSTOR4
0725- E8        1840              INX
0726- D0 F7     1850              BNE CRLABSTOR1
                1860
0728- A4 1D     1870 CRLABSTOR2   LDY *CRNEWLABEL+7
072A- C0 2A     1880              CPY #'*               ;GING THEN VORAUS?
072C- D0 04     1890              BNE CRLABSTOR3
072E- A0 20     1900              LDY #'                ;*MARKE LOESCHEN
0730- 84 1D     1910              STY *CRNEWLABEL+7
0732- B4 17     1920 CRLABSTOR3   LDY *CRNEWLABEL+1,X   ;MULTIPLY BY 10
0734- 94 16     1930              STY *CRNEWLABEL,X
0736- E8        1940              INX
0737- E0 04     1950              CPX #4
0739- D0 F7     1960              BNE CRLABSTOR3
073B- 95 16     1970 CRLABSTOR4   STA *CRNEWLABEL,X     ;DANN SPEICHERN
073D- 60        1980              RTS
                1990
```

```
073E- A5 C0      2000 CRINSZTAB  LDA *CRZEILEND        ;SUCHEN, OB LABEL SCHON
0740- A4 C1      2010            LDY *CRZEILEND+1      ;IN ZEILTABELLE
0742- D0 04      2020            BNE CRINSZTAB2
                 2030
0744- A5 B4      2040 CRINSZTAB1 LDA *CRREFPT          ;VORIGES ELEMENT
0746- A4 B5      2050            LDY *CRREFPT+1        ;ADRESSIEREN
0748- 38         2060 CRINSZTAB2 SEC
0749- E9 08      2070            SBC #8
074B- 85 B4      2080            STA *CRREFPT
074D- B0 01      2090            BCS CRINSZTAB3
074F- 88         2100            DEY
0750- 84 B5      2110 CRINSZTAB3 STY *CRREFPT+1
0752- C5 BA      2120            CMP *CRSTREND         ;ALLES DURCHSUCHT?
0754- 98         2130            TYA
0755- E5 BB      2140            SBC *CRSTREND+1
0757- 90 16      2150            BCC CRSPACE           ;JA, NOCH KEIN EINTRAG
0759- A0 04      2160            LDY #4
075B- B9 16 00   2170 CRINSZTAB4 LDA CRNEWLABEL,Y
075E- D1 B4      2180            CMP (CRREFPT),Y       ;LABEL GLEICH?
0760- D0 E2      2190            BNE CRINSZTAB1        ;NEIN
0762- 88         2200            DEY
0763- 10 F6      2210            BPL CRINSZTAB4
0765- A5 1D      2220            LDA *CRNEWLABEL+7     ;JA, BEREITS EINGETRAGEN
0767- A0 07      2230            LDY #7
0769- 11 B4      2240            ORA (CRREFPT),Y
076B- 91 B4      2250            STA (CRREFPT),Y       ;* MARKE UEBERNEHMEN
076D- D0 43      2260            BNE CRCLRLAB
                 2270
076F- A5 C0      2280 CRSPACE    LDA *CRZEILEND        ;ELEMENT IN ZEILTAB
0771- A4 C1      2290            LDY *CRZEILEND+1      ;EINFUEGEN
0773- D0 04      2300            BNE CRSPACE2
                 2310
0775- A5 B4      2320 CRSPACE1   LDA *CRREFPT
0777- A4 B5      2330            LDY *CRREFPT+1
0779- 85 BC      2340 CRSPACE2   STA *CRHILFPT
077B- 84 BD      2350            STY *CRHILFPT+1
077D- 38         2360            SEC
077E- E9 08      2370            SBC #8
0780- B0 01      2380            BCS CRSPACE3
0782- 88         2390            DEY
0783- 85 B4      2400 CRSPACE3   STA *CRREFPT          ;= HILFPT - 1 ELEMENT
0785- 84 B5      2410            STY *CRREFPT+1
0787- C5 BA      2420            CMP *CRSTREND         ;ZEILTABANF ERREICHT?
0789- 98         2430            TYA
078A- E5 BB      2440            SBC *CRSTREND+1
078C- 90 0F      2450            BCC CRELINS           ;JA, EINFUEGEN
078E- A0 07      2460            LDY #7
0790- 38         2470            SEC
0791- B1 B4      2480 CRSPACE4   LDA (CRREFPT),Y       ;EINTRAG UM 1 ELEMENT
0793- 91 BC      2490            STA (CRHILFPT),Y      ;NACH OBEN SPEICHERN
0795- F9 16 00   2500            SBC CRNEWLABEL,Y      ;MIT LABEL VERGLEICHEN
0798- 88         2510            DEY
0799- 10 F6      2520            BPL CRSPACE4
079B- B0 D8      2530            BCS CRSPACE1          ;NAECHSTES ELEMENT
079D- A0 07      2540 CRELINS    LDY #7
079F- B9 16 00   2550 CRELINS1   LDA CRNEWLABEL,Y      ;EINTRAG SPEICHERN
07A2- 91 BC      2560            STA (CRHILFPT),Y
07A4- 88         2570            DEY
07A5- 10 F8      2580            BPL CRELINS1
07A7- 18         2590            CLC                   ;ZEILTAB UM 1 ELEMENT
07A8- A5 C0      2600            LDA *CRZEILEND        ;VERGROESSERN
07AA- 69 08      2610            ADC #8
07AC- 85 C0      2620            STA *CRZEILEND
07AE- 90 02      2630            BCC CRCLRLAB
07B0- E6 C1      2640            INC *CRZEILEND+1
07B2- A0 05      2650 CRCLRLAB   LDY #5                ;LABEL LOESCHEN
07B4- A9 20      2660            LDA #'
07B6- 99 15 00   2670 CRCLRLAB1  STA CRNEWLABEL-1,Y
07B9- 88         2680            DEY
07BA- D0 FA      2690            BNE CRCLRLAB1
07BC- 60         2700 CRRETURN   RTS
```

```
                        2710
07BD- A5 BE     2720 CRMERGE    LDA *CRREFEND       ;POINTER AUF
07BF- A4 BF     2730            LDY *CRREFEND+1
07C1- 85 B4     2740            STA *CRREFPT        ;ENDE DER REFTAB
07C3- 84 B5     2750            STY *CRREFPT+1
07C5- 38        2760            SEC
07C6- A5 C0     2770            LDA *CRZEILEND      ;LAENGE DER ZEILTAB
07C8- E5 BA     2780            SBC *CRSTREND
07CA- 85 DD     2790            STA *CRHIFE         ;NACH CRHIFE
07CC- A5 C1     2800            LDA *CRZEILEND+1
07CE- E5 BB     2810            SBC *CRSTREND+1
07D0- 85 DE     2820            STA *CRHIFE+1
07D2- 05 DD     2830            ORA *CRHIFE
07D4- F0 E6     2840            BEQ CRRETURN        ;KEIN EINTRAG IN ZEILTAB
07D6- 18        2850            CLC
07D7- A5 DD     2860            LDA *CRHIFE         ;LAENGE ZEILTAB
07D9- 65 B4     2870            ADC *CRREFPT        ;+ ENDE DER REFTAB
07DB- 85 BE     2880            STA *CRREFEND       ;= NEUES ENDE REFTAB
07DD- 85 BC     2890            STA *CRZEILPT
07DF- A5 DE     2900            LDA *CRHIFE+1
07E1- 65 B5     2910            ADC *CRREFPT+1
07E3- 85 BF     2920            STA *CRREFEND+1
07E5- 85 BD     2930            STA *CRZEILPT+1
07E7- 20 27 08  2940            JSR CRVORELE
07EA- A5 C0     2950 CRMERGE1   LDA *CRZEILEND      ;ZEILTAB UM
07EC- 38        2960            SEC                 ;1 ELEMENT VERMINDERN
07ED- E9 08     2970            SBC #8
07EF- 85 C0     2980            STA *CRZEILEND
07F1- B0 02     2990            BCS CRMERGE2
07F3- C6 C1     3000            DEC *CRZEILEND+1
07F5- C5 BA     3010 CRMERGE2   CMP *CRSTREND       ;ZEILTAB ABGEARBEITET?
07F7- A5 C1     3020            LDA *CRZEILEND+1
07F9- E5 BB     3030            SBC *CRSTREND+1
07FB- 90 BF     3040            BCC CRRETURN        ;JA
07FD- A5 BC     3050 CRMERGE3   LDA *CRZEILPT

07FF- 38        3060            SEC
0800- E9 08     3070            SBC #8
0802- 85 BC     3080            STA *CRZEILPT
0804- B0 02     3090            BCS CRMERGE4
0806- C6 BD     3100            DEC *CRZEILPT+1
0808- A0 07     3110 CRMERGE4   LDY #7
080A- 38        3120            SEC
080B- B1 B4     3130 CRMERGE5   LDA (CRREFPT),Y      ;ELEMENT IN REFTAB
080D- 91 BC     3140            STA (CRZEILPT),Y    ;HOCHSCHIEBEN
080F- F1 C0     3150            SBC (CRZEILEND),Y   ;EINTRAG IN ZEILTAB
0811- 88        3160            DEY                 ;KLEINER?
0812- 10 F7     3170            BPL CRMERGE5
0814- 90 06     3180            BCC CRMERGE6        ;JA
0816- 20 27 08  3190            JSR CRVORELE
0819- 4C FD 07  3200            JMP CRMERGE3
                3210
081C- A0 07     3220 CRMERGE6   LDY #7               ;ELEMENT AUS ZEILTAB
081E- B1 C0     3230 CRMERGE7   LDA (CRZEILEND),Y
0820- 91 BC     3240            STA (CRZEILPT),Y    ;UEBERNEHMEN
0822- 88        3250            DEY
0823- 10 F9     3260            BPL CRMERGE7
0825- 30 C3     3270            BMI CRMERGE1
                3280
0827- A5 B4     3290 CRVORELE   LDA *CRREFPT         ;VORHERGEHENDES
0829- 38        3300            SEC                 ;ELEMENT ADRESSIEREN
082A- E9 08     3310            SBC #8
082C- 85 B4     3320            STA *CRREFPT
082E- B0 02     3330            BCS CRVORELE1
0830- C6 B5     3340            DEC *CRREFPT+1
0832- 60        3350 CRVORELE1  RTS
                3360
0833- A2 04     3370 CRDRUCK    LDX #4              ;REFTAB AUSDRUCKEN
0835- 86 16     3380            STX *CRNEWLABEL     ;LABEL "ENTWERTEN"
0837- 20 C9 FF  3390            JSR CKOUT
083A- A5 BA     3400            LDA *CRSTREND       ;ANFANG REFTAB
083C- 85 B4     3410            STA *CRREFPT
```

```
08C8- 8A          4130              TXA
08C9- 38          4140              SEC
08CA- E9 06       4150              SBC #6                ;+1 FUER ABSTAND
08CC- A8          4160              TAY                   ;FUEHRENDE SPACES
08CD- 20 CD FD    4170 CRPRZEILN2   JSR SPACE
08D0- C8          4180              INY
08D1- D0 FA       4190              BNE CRPRZEILN2
08D3- C8          4200              INY
08D4- 84 1F       4210              STY ±STRADR           ;=1
08D6- 84 20       4220              STY ±STRADR+1
08D8- 4C 23 CA    4230              JMP STROUT            ;STRING AUSGEBEN
                  4240
08DB- 00 00 00    4250 CRNULLTAB    .BY 0 0 0 0 0 0 0 0 ;NULL-EINTRAG
08DE- 00 00 00
08E1- 00 00
08E3- 4E 4F 4E    4260              .BY 'NONE' 32 32 0 0
08E6- 45 20 20
08E9- 00 00
08EB- 50 4F 4B    4270 CRLABTEXT    .BY 'POKEPEEKSYS USR WAIT'
08EE- 45 50 45
08F1- 45 4B 53
08F4- 59 53 20
08F7- 55 53 52
08FA- 20 57 41
08FD- 49 54
                  4280
                  4290              .BA $0900             ;NEUE PAGE!!
                  4300 ;            ================================
0900- 00 AD 1B    4310 CRARBTAB00   .BY $00 $AD $1B               ;GRUNDSTELLUNG
0903- 00 00 09    4320              .BY $00 $00 $09 $00 $00 $12
0906- 00 00 12
0909- 09 09 09    4330 CRARBTAB09   .BY $09 $09 $09               ;DATA
090C- 09 09 09    4340              .BY $09 $09 $09 $00 $09 $09
090F- 00 09 09
0912- 00 AD A4    4350 CRARBTAB12   .BY $00 $AD $A4               ;GOTO RUN GOSUB +
0915- 00 00 09    4360              .BY $00 $00 $09 $00 $12 $12 ;THEN
0918- 00 12 12
091B- 00 1B 1B    4370 CRARBTAB1B   .BY $00 $1B $1B               ;0-9
091E- 00 00 09    4380              .BY $00 $00 $09 $00 $00 $12
0921- 00 00 12
0924- 00 00 A4    4390 CRARBTAB24   .BY $00 $00 $A4               ;SPRUNGZIEL
0927- 00 00 09    4400              .BY $00 $00 $09 $00 $12 $12 ;WEITERE STELLEN
092A- 00 12 12
092D- 00 B6 B6    4410 CRARBTAB2D   .BY $00 $B6 $B6               ;2.STELLE LABEL
0930- BF B0 09    4420              .BY $BF $B0 $09 $00 $00 $12
0933- 00 00 12
0936- 00 36 36    4430 CRARBTAB36   .BY $00 $36 $36               ;$ %  3.ZEICHEN
0939- BF B0 09    4440              .BY $BF $B0 $09 $00 $00 $12
093C- 00 00 12
093F- 00 3F 00    4450 CRARBTAB3F   .BY $00 $3F $00               ;(    4.ZEICHEN
0942- 00 B0 09    4460              .BY $00 $B0 $09 $00 $00 $12
0945- 00 00 12
                  4470 ;-------------------------------------------------
                  4480 ;UMWERTUNG DER BASIC-ZEICHEN:
                  4490
                  4500 ; $.0 = STANDARD      I   $02 = FN
                  4510 ; $.1 = A-Z           I   $03 = POKE
                  4520 ; $.2 = 0-9           I   $07 = PEEK
                  4530 ; $.3 = $ %           I   $0B = SYS
                  4540 ; $.4 = (             I   $0F = USR
                  4550 ; $.5 = DATA          I   $13 = WAIT
                  4560 ; $.6 = :             I ------------------------
                  4570 ; $.7 = ,             I   $8. = KEINE WERTZUWEISUNG
                  4580 ; $.8 = GOTO RUN      I   $4. = KEINE VERAENDERUNG
                  4590 ;       THEN GOSUB    I   $2. = WERTZUWEISUNG
                  4600
0948- 80 FF 40    4610 CRCODETAB    .BY $80 $FF $40 ;ERST AB ZEICHEN:$21
094B- 43 43 80    4620              .BY $43 $43 $80 $80 $84 $80 $80 $80 $47
094E- 80 84 80
0951- 80 80 47
```

```
083E- A4 BB      3420              LDY #CRSTREND+1
0840- C8         3430              INY
0841- 84 B5      3440              STY #CRREFPT+1
0843- A0 04      3450 CRDRNEWELE   LDY #4            ;NOCH GLEICHES LABEL ?
0845- B9 16 00   3460 CRDRNEWEL1   LDA CRNEWLABEL,Y
0848- D1 B4      3470              CMP (CRREFPT),Y
084A- D0 05      3480              BNE CRDRNEWEL2    ;NEIN
084C- 88         3490              DEY
084D- 10 F6      3500              BPL CRDRNEWEL1
084F- 30 1B      3510              BMI CRDRREF1      ;JA
                 3520
0851- 20 E2 C9   3530 CRDRNEWEL2   JSR CRLFBAS       ;ZEILENWECHSEL
0854- A0 00      3540              LDY #0
0856- B1 B4      3550 CRDRLAB      LDA (CRREFPT),Y   ;LABEL MERKEN +
0858- 99 16 00   3560              STA CRNEWLABEL,Y
085B- 20 D2 FF   3570              JSR BSOUT         ;AUSGEBEN
085E- C8         3580              INY
085F- C0 05      3590              CPY #5
0861- 90 F3      3600              BCC CRDRLAB
0863- A9 3A      3610              LDA #':           ;TRENNZEICHEN
0865- 20 D2 FF   3620              JSR BSOUT
0868- A9 00      3630 CRDRREF      LDA #0
086A- 85 C2      3640              STA #CRZREF
086C- E6 C2      3650 CRDRREF1     INC #CRZREF
086E- A5 C2      3660              LDA #CRZREF
0870- C5 C3      3670              CMP #CRREFMAX
0872- 90 0D      3680              BCC CRDRREF3
0874- 20 E2 C9   3690              JSR CRLFBAS       ;ZEILENWECHSEL
0877- A0 05      3700              LDY #5            ;TABULATOR
0879- 20 CD FD   3710 CRDRREF2     JSR SPACE
087C- 88         3720              DEY
087D- 10 FA      3730              BPL CRDRREF2
087F- 30 E7      3740              BMI CRDRREF
                 3750
0881- 20 AD 08   3760 CRDRREF3     JSR CRPRZEILNR    ;AUSGEBEN ZEILEN#
0884- A0 07      3770              LDY #7
0886- B1 B4      3780              LDA (CRREFPT),Y   ;AUSGABE MARKE
0888- C9 20      3790              CMP #'
088A- F0 02      3800              BEQ CRDRREF4
088C- A9 2A      3810              LDA #'*
088E- 20 D2 FF   3820 CRDRREF4     JSR BSOUT
0891- 20 E1 FF   3830 CRDRREF5     JSR STOPR
0894- A4 97      3840              LDY #KBDMAT       ;ANDERE TASTE GEDRUECKT ?
0896- C8         3850              INY
0897- D0 F8      3860              BNE CRDRREF5      ;JA
0899- 18         3870              CLC
089A- A5 B4      3880              LDA #CRREFPT      ;NAECHSTES ELEMENT
089C- 69 08      3890              ADC #8
089E- 85 B4      3900              STA #CRREFPT
08A0- 90 02      3910              BCC CRDRREF6
08A2- E6 B5      3920              INC #CRREFPT+1
08A4- C5 BE      3930 CRDRREF6     CMP #CRREFEND     ;REFTABELLEN-ENDE ?
08A6- A5 B5      3940              LDA #CRREFPT+1
08A8- E5 BF      3950              SBC #CRREFEND+1
08AA- 90 97      3960              BCC CRDRNEWELE    ;NEIN
08AC- 60         3970              RTS
                 3980
08AD- A0 05      3990 CRPRZEILNR   LDY #5            ;ZEILEN # AUSGEBEN
08AF- B1 B4      4000              LDA (CRREFPT),Y
08B1- 85 5F      4010              STA #FAC1+1
08B3- C8         4020              INY
08B4- B1 B4      4030              LDA (CRREFPT),Y
08B6- 85 60      4040              STA #FAC1+2
08B8- A2 90      4050              LDX #$90          ;WANDELN HEX IN
08BA- 38         4060              SEC               ;DEZIMALSTRING UND
08BB- 20 55 DB   4070              JSR ADRFP         ;ABLAGE IN $0100
08BE- 20 E9 DC   4080              JSR FPSTR
08C1- AA         4090              TAX               ;=0
08C2- E8         4100 CRPRZEILN1   INX               ;LAENGE ERMITTELN
08C3- BD 01 01   4110              LDA STACK+1,X
08C6- D0 FA      4120              BNE CRPRZEILN1
```

```
0954- 80 80 80  4630          .BY $80 $80 $80 $42 $42 $42 $42 $42 $42
0957- 42 42 42
095A- 42 42 42
095D- 42 42 42  4640          .BY $42 $42 $42 $42 $26 $40 $80 $80 $80
0960- 42 26 40
0963- 80 80 80
0966- 80 80 41  4650          .BY $80 $80 $41 $41 $41 $41 $41 $41 $41
0969- 41 41 41
096C- 41 41 41

096F- 41 41 41  4660          .BY $41 $41 $41 $41 $41 $41 $41 $41 $41
0972- 41 41 41
0975- 41 41 41
0978- 41 41 41  4670          .BY $41 $41 $41 $41 $41 $41 $41 $41 $41
097B- 41 41 41
097E- 41 41 41
0981- 41 80 80  4680          .BY $41 $80 $80 $80 $80 $80 $80 $80 $80
0984- 80 80 80
0987- 80 80 80
098A- 80 80 80  4690          .BY $80 $80 $80 $80 $80 $80 $80 $80 $80
098D- 80 80 80
0990- 80 80 80
0993- 80 80 80  4700          .BY $80 $80 $80 $80 $80 $80 $80 $80 $80
0996- 80 80 80
0999- 80 80 80
099C- 80 80 80  4710          .BY $80 $80 $80 $80 $80 $80 $80 $80 $80
099F- 80 80 80
09A2- 80 80 80
09A5- 80 80 80  4720          .BY $80 $80 $80 $20 $80 $85 $20 $20 $80
09A8- 20 80 85
09AB- 20 20 80
09AE- 20 20 88  4730          .BY $20 $20 $88 $88 $80 $80 $18 $80 $FF
09B1- 88 80 80
09B4- 18 80 FF
09B7- 80 80 13  4740          .BY $80 $80 $13 $80 $80 $80 $20 $03 $80
09BA- 80 80 80
09BD- 20 03 80
09C0- 80 80 80  4750          .BY $80 $80 $80 $80 $80 $0B $80 $80 $20
09C3- 80 80 0B
09C6- 80 80 20
09C9- 80 80 80  4760          .BY $80 $80 $80 $02 $80 $28 $80 $80 $80
09CC- 02 80 28
09CF- 80 80 80
09D2- 80 80 80  4770          .BY $80 $80 $80 $80 $80 $80 $80 $80 $80
09D5- 80 80 80
09D8- 80 80 80
09DB- 80 80 80  4780          .BY $80 $80 $80 $0F $80 $80 $80 $80 $80
09DE- 0F 80 80
09E1- 80 80 80
09E4- 80 80 80  4790          .BY $80 $80 $80 $80 $80 $07
09E7- 80 80 07
                4800          .EN
END OF MAE PASS!
```

--- LABEL FILE: ---

```
ADRFP =DB55
CKOUT =FFC9
CRARBTAB00 =0900
CRARBTAB1B =091B
CRARBTAB36 =0936
CRBEGINN =0620
CRCODETAB =0948
CRDRNEWEL2 =0851
CRDRREF1 =086C
CRDRREF4 =088E
CRDRUCK =0833
CRENDE =061D
CRINSZTAB =073E
BSOUT =FFD2
CLRCH =FFCC
CRARBTAB09 =0909
CRARBTAB24 =0924
CRARBTAB3F =093F
CRCLRLAB =07B2
CRDRLAB =0856
CRDRNEWELE =0843
CRDRREF2 =0879
CRDRREF5 =0891
CRELINS =079D
CRHIFE =00DD
CRINSZTAB1 =0744
CHKIN =FFC6
CRARBPT =00D6
CRARBTAB12 =0912
CRARBTAB2D =092D
CRBASCODE =00C2
CRCLRLAB1 =07B6
CRDRNEWEL1 =0845
CRDRREF =0868
CRDRREF3 =0881
CRDRREF6 =08A4
CRELINS1 =079F
CRHILFPT =00BC
CRINSZTAB2 =074B
```

```
CRINSZTAB3 =0750
CRLABSTOR2 =072B
CRLABSTORE =0717
CRMERGE =07BD
CRMERGE3 =07FD
CRMERGE6 =081C
CRNEWBYTE1 =067A
CRNEWBYTE4 =0696
CRNEWBYTE7 =06F5
CRNEWCODE2 =06BB
CRNEWCODE5 =06D3
CRNEWCODE8 =0700
CRNEWLINE =064D
CRPRZEILN1 =08C2
CRREFEND =00BE
CRRETURN =07BC
CRSPACE2 =0779
CRSTREND =00BA
CRZEILEND =00C0
FAC1 =005E
KBDMAT =0097
STOPR =FFE1
STROUT =CA23
//0000,09EA,09EA
Ü
```

```
CRINSZTAB4 =075B
CRLABSTOR3 =0732
CRLABTEXT =08EB
CRMERGE1 =07EA
CRMERGE4 =0808
CRMERGE7 =081E
CRNEWBYTE2 =067D
CRNEWBYTE5 =069B
CRNEWCODE =06AA
CRNEWCODE3 =06CF
CRNEWCODE6 =06E1
CRNEWCODE9 =070B
CRNULL1 =0642
CRPRZEILN2 =08CD
CRREFMAX =00C3
CRSPACE =076F
CRSPACE3 =0783
CRVORELE =0827
CRZEILPT =00BC
FPSTR =DCE9
SPACE =FDCD
STRADR =001F
```

```
CRLABSTOR1 =071F
CRLABSTOR4 =073B
CRLFBAS =C9E2
CRMERGE2 =07F5
CRMERGE5 =080B
CRNEWBYTE =066D
CRNEWBYTE3 =0681
CRNEWBYTE6 =06A4
CRNEWCODE1 =06AB
CRNEWCODE4 =06D1
CRNEWCODE7 =06F2
CRNEWLABEL =0016
CRNULLTAB =08DB
CRPRZEILNR =08AD
CRREFPT =00B4
CRSPACE1 =0775
CRSPACE4 =0791
CRVORELE1 =0832
CRZREF =00C2
GETIN =FFE4
STACK =0100
STREND =002E
```

Peter W. Arps, 2000 Hamburg 73

Alpha-Korrelation

In der Zeitschrift CHIP Nr. 6/83 wurde für den ZX81 unter dem Titel 'Tippfehler besiegt' ein BASIC-Programm zur Erkennung von Wortähnlichkeiten (Alpha-Korrelation) beschrieben. Es wird ein Suchbegriff mit einem anderen verglichen. Je nach der ermittelten Übereinstimmung wird al Ergebnis ein Wert zwischen Null (keinerlei Übereinstimmung) und Eins (völlige Übereinstimmung) geliefert.

Diese Routine läßt sich sinnvoll für das Durchsuchen von Dateien (z.B. Namen, Telefonnummern usw.) einsetzen. Als BASIC-Routine ist sie aber aufgrund iherer Stringoperationen viel zu langsam und damit nicht zu gebrauchen. Ich habe daher den Algorithmus zu einer Maschinensprache-Routine für CBM mit BASIC 3 verarbeitet, die ca. um den Faktor 100 schneller ausführt.

Der Aufruf erfolgt mit SYS (adr) (Suchbegriff, Vergleichswert, Ergebnis). Die Variablen können beliebig gewählt werden. Ist das Ergebnis kleiner als der kritische Wert, so ist der Aufruf mit vertauschten Begriffen zu wiederholen. Beispiel:

```
100 SYS(800) (S$, V$,R): IF R < .75 THEN SYS(800) (V$, S$, R)
```

Das Originalprogramm legt in der Länge des Vergleichswertes eine Matrix an, die soviele Elemente enthält, wie der Suchbegriff lang ist. Da nur bestimmte Informationen aus der Matrix benötigt werden, konnte das Programm, um eine höhere Performance zu errreichen, noch optimiert werden. Die Matrix wurde auf eine Merkerzeile in Tapebuffer und zwei Pointer reduziert.

```
0010              ;++++++++++++++++++++
0020              ;+                  +
0030              ;+ APHA-KORRELATION +
0040              ;+                  +
0050              ;++++++++++++++++++++
0060              ;
0070              .OS
0080              .BA $0320
0090 ;
0100 SU.LNG       .DE $E1
```

```
                        0110 SU.PNT      .DE $B2
                        0120 VGL.LNG     .DE $B9
                        0130 VGL.PNT     .DE $BA
                        0140 KORRE       .DE $BC
                        0150 SU.ZHL      .DE $BD
                        0160 VGL.ZHL     .DE $BE
                        0170 FIRST.0     .DE $BF
                        0180 LAST.1      .DE $C0
                        0190 T.BUFF      .DE $27A
                        0200 ;                          BASIC 4.0
                        0210 KOMMA       .DE $CDF8       ;BEF5
                        0220 KLAUF       .DE $CDF5       ;BEF2
                        0230 KLZU        .DE $CDF2       ;BEEF
                        0240 ARGUM       .DE $CC9F       ;BD98
                        0250 ARGSTR      .DE $D57D       ;C7B5
                        0260 FVAR        .DE $CF6D       ;C12B
                        0270 FAC1MEM     .DE $DAE0       ;CD0A
                        0280 DIV         .DE $DA1E       ;CC48
                        0290 FAC12       .DE $DB1B       ;CD45
                        0300 INTFP       .DE $D26D       ;C4BC
                        0310 ;
                        0320 ;
0320- 20 F5 CD          0330 ENTRY       JSR KLAUF       ;SYS(800)(S$,V$,R)
0323- 20 9F CC          0340             JSR ARGUM
0326- 20 7D D5          0350             JSR ARGSTR
0329- 85 B1             0360             STA *SU.LNG
032B- 86 B2             0370             STX *SU.PNT
032D- 84 B3             0380             STY *SU.PNT+1
032F- 20 F8 CD          0390             JSR KOMMA
0332- 20 9F CC          0400             JSR ARGUM
0335- 20 7D D5          0410             JSR ARGSTR
0338- 85 B9             0420             STA *VGL.LNG
033A- 86 BA             0430             STX *VGL.PNT
033C- 84 BB             0440             STY *VGL.PNT+1
033E- 20 F8 CD          0450             JSR KOMMA
                        0460             ;
0341- A9 00             0470             LDA #0
0343- 85 BC             0480             STA *KORRE
0345- 85 BD             0490             STA *SU.ZHL
0347- 85 BE             0500             STA *VGL.ZHL
0349- 85 C0             0510             STA *LAST.1
034B- 85 BF             0520             STA *FIRST.0
034D- C6 BF             0530             DEC *FIRST.0
034F- A6 B9             0540             LDX *VGL.LNG
0351- 9D 7A 02          0550 LOESCH      STA T.BUFF,X
0354- CA                0560             DEX
0355- 10 FA             0570             BPL LOESCH
                        0580 ;
0357- A9 00             0590 LOOP        LDA #0
0359- 48                0600             PHA
035A- A4 BE             0610             LDY *VGL.ZHL
035C- B9 7A 02          0620             LDA T.BUFF,Y
035F- 30 16             0630             BMI A2          ;BELEGT
0361- 68                0640             PLA
0362- B1 BA             0650             LDA (VGL.PNT),Y
0364- A4 BD             0660             LDY *SU.ZHL
0366- D1 B2             0670             CMP (SU.PNT),Y
0368- D0 05             0680             BNE A1
```

```
036A- E6 BC      0690              INC *KORRE
036C- A9 FF      0700              LDA #255
036E- 2C         0710              .BY $2C
036F- A9 01      0720 A1           LDA #1
0371- 48         0730              PHA
0372- A6 BE      0740              LDX *VGL.ZHL
0374- 9D 7A 02   0750              STA T.BUFF,X
0377- A4 BD      0760 A2           LDY *SU.ZHL
0379- C4 B1      0770              CPY *SU.LNG
037B- B0 49      0780              BCS FERTIG
037D- A6 BE      0790              LDX *VGL.ZHL
037F- 68         0800              PLA
0380- F0 1A      0810              BEQ LEER
0382- 30 2A      0820              BMI EINS
                 0830 ;
0384- A5 BF      0840 NULL         LDA *FIRST.0
0386- 10 02      0850              BPL NULL0
0388- 86 BF      0860              STX *FIRST.0
038A- E8         0870 NULL0        INX
038B- E4 B9      0880              CPX *VGL.LNG
038D- 90 08      0890              BCC NULL1
038F- E6 BD      0900              INC *SU.ZHL
0391- A6 BF      0910              LDX *FIRST.0
0393- A9 FF      0920              LDA #255
0395- 85 BF      0930              STA *FIRST.0
0397- 86 BE      0940 NULL1        STX *VGL.ZHL
0399- 4C 57 03   0950              JMP LOOP
                 0960 ;
039C- E8         0970 LEER         INX
039D- E4 B9      0980              CPX *VGL.LNG
039F- 90 08      0990              BCC LEER1
03A1- E6 BD      1000              INC *SU.ZHL
03A3- A6 C0      1010              LDX *LAST.1
03A5- A9 FF      1020              LDA #255
03A7- 85 BF      1030              STA *FIRST.0
03A9- 86 BE      1040 LEER1        STX *VGL.ZHL
03AB- 4C 57 03   1050              JMP LOOP
                 1060 ;
03AE- E6 BD      1070 EINS         INC *SU.ZHL
03B0- A9 FF      1080              LDA #255
03B2- 85 BF      1090              STA *FIRST.0
03B4- E8         1100              INX
03B5- E4 B9      1110              CPX *VGL.LNG
03B7- 90 02      1120              BCC EINS1
03B9- A2 00      1130              LDX #0
03BB- 86 BE      1140 EINS1        STX *VGL.ZHL
03BD- E4 C0      1150              CPX *LAST.1
03BF- 90 02      1160              BCC EINS2
03C1- 86 C0      1170              STX *LAST.1
03C3- 4C 57 03   1180 EINS2        JMP LOOP
                 1190 ;
03C6- 68         1200 FERTIG       PLA
03C7- A4 BC      1210              LDY *KORRE
03C9- A9 00      1220              LDA #0
03CB- 20 6D D2   1230              JSR INTFP
03CE- 20 1B DB   1240              JSR FAC12
03D1- A4 B1      1250              LDY *SU.LNG
03D3- A9 00      1260              LDA #0
```

```
03D5- 20 6D D2  1270            JSR INTFP
03D8- 20 1E DA  1280            JSR DIV
03DB- 20 6D CF  1290            JSR FVAR
03DE- AA        1300            TAX
03DF- 20 E0 DA  1310            JSR FAC1MEM
03E2- 4C F2 CD  1320            JMP KLZU
                1330 ;
                1340            .EN
LABEL FILE: C / = EXTERNAL ]
/SU.LNG=00B1            /SU.PNT=00B2            /VGL.LNG=00B9
/VGL.PNT=00BA           /KORRE=00BC             /SU.ZHL=00BD
/VGL.ZHL=00BE           /FIRST.0=00BF           /LAST.1=00C0
/T.BUFF=027A            /KOMMA=CDF8             /KLAUF=CDF5
/KLZU=CDF2              /ARGUM=CC9F             /ARGSTR=D57D
/FVAR=CF6D              /FAC1MEM=DAE0           /DIV=DA1E
/FAC12=DB1B             /INTFP=D26D             ENTRY=0320
LOESCH=0351             LOOP=0357               A1=036F
A2=0377                 NULL=0384               NULL0=038A
NULL1=0397              LEER=039C               LEER1=03A9
EINS=03AE               EINS1=03B8              EINS2=03C3
FERTIG=03C6
//0000,03E5,03E5
```

Peter Porbadnigk, 2070 Ahrensburg

VC-20 am IEEE488-Bus

Angeregt durch den Artikel AIM 65 am IEEE 488-Bus in MICRO MAG, Heft 19, sollte versucht werden, auch den VC-20 an diesem parallelen Bus zu betreiben. Dabei sollte möglichst nicht der Modulschacht benutzt werden, und es sollten von dem zusätzlichen VIA 6522 keine Adressen im BASIC-Bereich hex A000-BFFF benutzt werden.

Zur Hardware:

Im Bereich der Schnittstellenbausteine und des Video-Chips ist der Bereich hex 9190-918F nicht genutzt und wir auch nicht durch Kopien von anderen Chips belegt. Somit liegt die Adresse für das VIA mit 9180 fest. Das Treiberprogramm liegt im Moment noch im Bereich 6000, aber im Bereich 9800-9FFF ist auch noch ungenutzter Adreßraum, so daß ein 2K-EPROM sicher auch hier einen geeigneten Platz findet, so daß dann von der parallelen Schnittstelle keine Adressen benötigt werden, die evtl. von anderen Programmen belegt werden könnten (Module usw.). Die Schaltung ist auf einer Lochrasterplatte aufgebaut und in Fädeltechnik verdrahtet. Um das Interface im VC-20 zu installieren, wurde das VIA auf Sockel UAB3 herausgezogen. Auf der neuen Platte wird für dieses VIA ein Wire Wrap-Sockel eingelötet. Somit stehen alle Signale auf einmal auf der neuen Platine zur Verfügung, bis auf CA7, das von der Mutterplatine abgenommen werden muß (Pin 1 von einem ROM z.B.). Nähere Beschreibung der Hardware wie in o.a. Quelle.

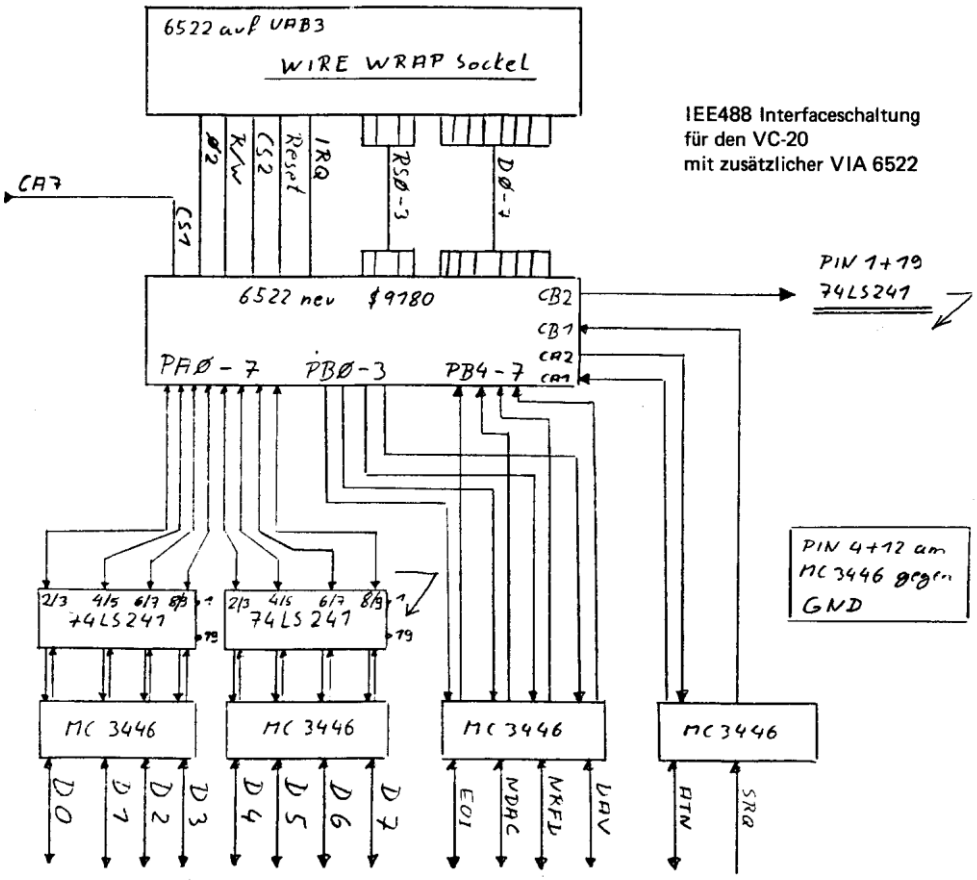
Zur Software:

Das Programm leitet nach dem Initialisieren alle Ein- und Ausgaben, die über eine Device-Nummer größer als 3 erfolgen, auf die parallele Schnittstelle um. Die serielle Schnittstelle ist nun außer Betrieb. So lassen sich alle Befehle wie SAVE, LOAD, VERIFY, OPEN, CLOSE, PRINT#, GET# und INPUT# wie gewohnt in BASIC nutzen. Die Syntax ist identisch mit der der seriellen Schnittstelle. Von einem Maschinenprogramm aus funktioniert die parallele Schnittstelle, solange wie über die Pointer hex FFC0-FFE7 gearbeitet wird. Programme, die im Betriebssystem direkt Routinen ansprechen, müssen dann geändert werden, was aber kein großes Problem darstellen dürfte, da die Bezeichnung der Routinen im Treiberprogramm weitgehendst identisch mit denen des Betriebssystem ist. Mit Hilfe dieser Routinen ist auch eine BASIC-Erweiterung mit Befehlen aus BASIC 4 denkbar. Ein Besuch dahingehend ist schon unternommen.

Nach einem Reset mit STOP und RESTORE muß das Programm neu initialisiert werden, da das System die Pointer wieder auf sich setzt. Es kann dann vorkommen, daß nach einem Initialisieren die Befehle SAVE oder LOAD zu einem Device Not Present Error führen. Diesem kann man wie folgt abhelfen, so daß der VC-20 die Floppy auch bei SYVE und LAOD nachfolgend erkennt:

```
OPEN 15,8,15:PRINT#15,"I0":CLOSE15
```

Literatur: 65xx MICRO MAG Nr, 19 und 21, Systemlistings für VC-20 und CBM 4032.



IEE488 Interfaceschaltung
für den VC-20
mit zusätzlicher VIA 6522

```
0001  0000  ;**********************************
0002  0000  ;* vc-20 ieee-488 bus interface *
0003  0000  ;*=================================*
0004  0000  ;* peter porbadnigk    21.02.83 *
0005  0000  ;**********************************
0006  0000
0008  0000
0009  0000  ;definition der portadressen
0010  0000
```

```
0011  0000                    *= $9180
0012  9180
0013  9180                    portb  =*                ;steuerport
0014  9180                    porta  =*+1              ;datenport
0015  9180                    ddrb   =*+2
0016  9180                    ddra   =*+3
0017  9180                    acr    =*+11
0018  9180                    pcr    =acr+1
0019  9180
0020  9180                    ;konstanten
0021  9180
0022  9180                    write  =$c0
0023  9180                    read   =$e0
0024  9180                    atnhig =$0e
0025  9180                    atnlow =$0c
0026  9180                    unlisn =$3f
0027  9180                    untalk =$5f
0028  9180
0029  9180                    ;vc-20 system adressen
0030  9180
0031  9180                    filopn =$98              ;anzahl der offenen file
0032  9180                    fnlen  =$b7              ;länge des filenamens
0033  9180                    fn     =$b8              ;filenummer
0034  9180                    sa     =$b9              ;sekundär adr.
0035  9180                    dn     =$ba              ;device nummer
0036  9180                    fnlo   =$bb              ;lobyte filenameadr.
0037  9180                    status =$90              ;status byte
0038  9180                    lbflag =$94              ;flag für last byte
0039  9180                    outreg =$95              ;ausgabe register
0040  9180                    indev  =$99              ;eingabe gerät
0041  9180                    outdev =$9a              ;ausgabe gerät
0042  9180                    stop   =$f770            ;stoptaste abfragen
0043  9180                    *=$6b00
0044  6b00
0045  6b00                    ;alle system adressen für's ieee-bus handling
0046  6b00                    ;werden auf dieses programm gerichtet
0047  6b00
0048  6b00                    init
0049  6b00  a2 14                     ldx #20          ;ansprungadressen in
0050  6b02  bd 22 6b          loop    lda tabl,x       ;den bereich 031a-
0051  6b05  9d 19 03                 sta $319,x       ;032d übertragen
0052  6b08  ca                       dex
0053  6b09  d0 f7                    bne loop
0054  6b0b  a9 79                    lda #<load       ;load und
0055  6b0d  a0 6e                    ldy #>load
0056  6b0f  8d 30 03                 sta $0330
0057  6b12  8c 31 03                 sty $0331
0058  6b15  a9 17                    lda #<save
0059  6b17  a0 6f                    ldy #>save       ;save routine auf
0060  6b19  8d 32 03                 sta $0332        ;ieee bus richten
0061  6b1c  8c 33 03                 sty $0333
0062  6b1f  4c 7f 6d                 jmp output       ;alle ports output
0063  6b22
0064  6b22  00               tabl    .byt 0
0065  6b23  3a 6b                    .wor open,close,chkin,chkout
0065  6b25  ae 6b
0065  6b27  6f 6c
0065  6b29  e6 6b
```

```
0066  6b2b  55 6c           .wor clrchn,chrin,chrout,stop
0066  6b2d  cd 6c
0066  6b2f  2c 6c
0066  6b31  70 f7
0067  6b33  bf 6c           .wor getin,clall
0067  6b35  51 6c
0068  6b37  ff              .byt $ff,$ff,$ff
0068  6b38  ff
0068  6b39  ff
0069  6b3a
0070  6b3a                 ;fortsetzung der system-routinen
0071  6b3a                 ;==============================
0072  6b3a
0073  6b3a                 open
0074  6b3a  a6 b8                 ldx fn          ;get filenummer
0075  6b3c  d0 03                 bne lab107
0076  6b3e  4c 8d f7              jmp $f78d       ;not input file
0077  6b41  20 cf f3       lab107 jsr $f3cf        ;get file parameter
0078  6b44  d0 03                 bne lab10f
0079  6b46  4c 81 f7              jmp $f781       ;file open error
0080  6b49  a6 98         lab10f ldx filopn
0081  6b4b  e0 0a                 cpx #10         ;schon 10 files offen ??
0082  6b4d  90 03                 bcc lab118      ;nein -> weiter
0083  6b4f  4c 7e f7              jmp $f77e       ;too many files open
0084  6b52  e6 98         lab118 inc filopn       ; +1
0085  6b54  a5 b8                 lda fn          ;file parameter in
0086  6b56  9d 59 02              sta $259,x      ;tabelle eintragen
0087  6b59  a5 b9                 lda sa          ;sekundär adresse
0088  6b5b  09 60                 ora #$60        ;set bit 5 u. 6
0089  6b5d  85 b9                 sta sa          ;sekundär adr neu
0090  6b5f  9d 6d 02              sta $26d,x
0091  6b62  a5 ba                 lda dn          ;devicenummer
0092  6b64  9d 63 02              sta $263,x
0093  6b67  f0 0c                 beq lab189
0094  6b69  c9 03                 cmp #3          ;bildschirm
0095  6b6b  f0 08                 beq lab189
0096  6b6d  90 09                 bcc lab13a      ;cassette
0097  6b6f  20 7b 6b              jsr chopen      ;ieee open
0098  6b72  4c 42 f4              jmp $f442       ;open fortsetzen
0099  6b75  4c 93 f4       lab189 jmp $f493        ;-> rts
0100  6b78  4c 44 f4       lab13a jmp $f444        ;adr<3 (cassette rs 232)
0101  6b7b
0102  6b7b                 ;open für ieee-bus (parallel)
0103  6b7b
0104  6b7b                 chopen
0105  6b7b  20 7f 6d              jsr output      ;all output
0106  6b7e  a5 b9                 lda sa
0107  6b80  30 2a                 bmi lab1bb
0108  6b82  a4 b7                 ldy fnlen
0109  6b84  f0 26                 beq lab1bb
0110  6b86  20 46 6d              jsr listen
0111  6b89  a5 b9                 lda sa
0112  6b8b  09 f0                 ora #%11110000  ;sekundär adresse
0113  6b8d  20 77 6d              jsr second
0114  6b90  a5 90                 lda status      ;statusbyte
0115  6b92  10 05                 bpl lab1a8      ;bit 7 = 0
0116  6b94  68                    pla             ;return adresse entfernen
0117  6b95  68                    pla
```

```
0118  6b96  4c 8a f7        jmp $f78a       ;device not present
0119  6b99  a5 b7        lab1a8 lda fnlen
0120  6b9b  f0 0c              beq lab1b8
0121  6b9d  a0 00              ldy #$00
0122  6b9f  b1 bb        lab1ae lda (fnlo),y
0123  6ba1  20 40 6c            jsr ciout
0124  6ba4  c8                  iny
0125  6ba5  c4 b7              cpy fnlen
0126  6ba7  d0 f6              bne lab1ae
0127  6ba9  20 1a 6e      lab1b8 jsr unlis
0128  6bac  18            lab1bb clc
0129  6bad  60                  rts
0130  6bae
0131  6bae                ;close routine für ieee-bus
0132  6bae
0133  6bae                close
0134  6bae  20 d4 f3            jsr $f3d4       ;file# suchen
0135  6bb1  f0 02               beq lf351
0136  6bb3  18                  clc
0137  6bb4  60                  rts
0138  6bb5  20 df f3      lf351 jsr $f3df        ;get file parameter
0139  6bb8  8a                  txa
0140  6bb9  48                  pha
0141  6bba  a5 ba               lda dn
0142  6bbc  f0 09               beq label1
0143  6bbe  c9 03               cmp #3          ;bildschirm
0144  6bc0  f0 05               beq label1
0145  6bc2  b0 06               bcs close2
0146  6bc4  4c 60 f3            jmp $f360       ;rs 232 ??
0147  6bc7  4c b1 f3      label1 jmp $f3b1        ;file daten löschen
0148  6bca  24 b9          close2 bit sa
0149  6bcc  30 14               bmi lab3
0150  6bce  20 7f 6d            jsr output      ;all outp.
0151  6bd1  a5 ba               lda dn
0152  6bd3  20 46 6d            jsr listen
0153  6bd6  a5 b9               lda sa
0154  6bd8  29 ef               and #%11101111
0155  6bda  09 e0               ora #%11100000  ;sec.adr. e?
0156  6bdc  20 77 6d            jsr second
0157  6bdf  20 1a 6e            jsr unlis
0158  6be2  18            lab3  clc
0159  6be3  4c b1 f3            jmp $f3b1       ;file daten löschen
0160  6be6
0161  6be6                ;definition als output device
0162  6be6
0163  6be6                chkout
0164  6be6  20 cf f3            jsr $f3cf       ;file# suchen
0165  6be9  f0 03               beq lf311
0166  6beb  4c 84 f7            jmp $f784       ;file not open
0167  6bee  20 df f3      lf311 jsr $f3df        ;file parameter holen
0168  6bf1  a5 ba               lda dn
0169  6bf3  d0 03               bne lf31b
0170  6bf5  4c 90 f7      lf318 jmp $f790        ;not output file
0171  6bf8  c9 03          lf31b cmp #3          ;bildschirm
0172  6bfa  f0 0f               beq lf32e
0173  6bfc  b0 11               bcs lf332
0174  6bfe  c9 02               cmp #$02
0175  6c00  d0 03               bne lf328
```

```
0176  6c02  4c bc f0          jmp $f0bc       ;file an rs 232
0177  6c05  a6 b9       lf328 ldx sa
0178  6c07  e0 60             cpx ##$60
0179  6c09  f0 ea             beq lf318
0180  6c0b  85 9a       lf32e sta outdev
0181  6c0d  18                 clc
0182  6c0e  60                 rts
0183  6c0f             lf332
0184  6c0f  48                 pha
0185  6c10  20 7f 6d          jsr output      ;all outp.
0186  6c13  68                 pla
0187  6c14  aa                 tax             ;ieee-chkout
0188  6c15  20 46 6d          jsr listen
0189  6c18  a5 b9              lda sa
0190  6c1a  10 05              bpl lf33f
0191  6c1c  20 03 6e           jsr atnhi
0192  6c1f  d0 03              bne lf342
0193  6c21  20 77 6d    lf33f jsr second
0194  6c24  8a          lf342 txa
0195  6c25  24 90              bit status
0196  6c27  10 e2              bpl lf32e
0197  6c29  4c 8a f7           jmp $f78a       ;device not present
0198  6c2c
0199  6c2c                  ;character output to ieee 488
0200  6c2c
0201  6c2c                  chrout
0202  6c2c  48                  pha
0203  6c2d  a5 9a               lda outdev
0204  6c2f  c9 03               cmp #3          ;bildschirm
0205  6c31  d0 04               bne lf285
0206  6c33  68                  pla
0207  6c34  4c 42 e7           jmp $e742       ;auf bildschirm
0208  6c37  90 04       lf285   bcc lf28b
0209  6c39  68                  pla
0210  6c3a  4c 40 6c           jmp ciout       ;auf bus
0211  6c3d  4c 8b f2    lf28b   jmp $f28b        ;rs 232 ???
0212  6c40  24 94       ciout   bit lbflag      ;bit 7 = 1 ??
0213  6c42  30 04              bmi leeed
0214  6c44  c6 94              dec lbflag      ;flag für letztes zeicher
0215  6c46  d0 05              bne leef2
0216  6c48  48          leeed   pha
0217  6c49  20 aa 6d           jsr send        ;vorheriges senden
0218  6c4c  68                  pla
0219  6c4d  85 95       leef2   sta outreg       ;neues speichern
0220  6c4f  18                  clc
0221  6c50  60                  rts
0222  6c51
0223  6c51                  ;clear channel routine für ieee
0224  6c51
0225  6c51  a9 00       clall   lda #0           ;clear all
0226  6c53  85 98              sta $98
0227  6c55             clrchn
0228  6c55  20 7f 6d           jsr output      ;all outp.
0229  6c58  a2 03               ldx #3          ;bildschirm
0230  6c5a  e4 9a               cpx outdev
0231  6c5c  b0 03               bcs lf3fc
0232  6c5e  20 1a 6e           jsr unlis
0233  6c61  e4 99       lf3fc   cpx indev
```

```
0234  6c63  b0 03        bcs lf403
0235  6c65  20 12 6e     jsr untal
0236  6c68  86 9a     lf403  stx outdev
0237  6c6a  a9 00        lda #0
0238  6c6c  85 99        sta indev
0239  6c6e  60           rts
0240  6c6f
0241  6c6f              ;definition als input device
0242  6c6f
0243  6c6f              chkin
0244  6c6f  20 cf f3        jsr $f3cf       ;file# suchen
0245  6c72  f0 03           beq lf2cf
0246  6c74  4c 84 f7        jmp $f784       ;file not open
0247  6c77  20 df f3  lf2cf  jsr $f3df       ;parameter holen
0248  6c7a  a5 ba           lda dn
0249  6c7c  f0 16           beq lf2ec
0250  6c7e  c9 03           cmp #3          ;bildschirm
0251  6c80  f0 12           beq lf2ec
0252  6c82  b0 14           bcs lf2f0
0253  6c84  c9 02           cmp #$02
0254  6c86  d0 03           bne lf2e3
0255  6c88  4c 16 f1        jmp $f116       ;input von rs 232
0256  6c8b  a6 b9     lf2e3  ldx sa
0257  6c8d  e0 60           cpx #$60
0258  6c8f  f0 03           beq lf2ec
0259  6c91  4c 8d f7        jmp $f78d       ;not input file
0260  6c94  85 99     lf2ec  sta indev
0261  6c96  18           clc
0262  6c97  60           rts
0263  6c98  aa           lf2f0  tax
0264  6c99  20 7f 6d        jsr output      ;all outp.
0265  6c9c  20 43 6d        jsr talk
0266  6c9f  a5 b9           lda sa
0267  6ca1  10 06           bpl lf2fe
0268  6ca3  20 03 6e        jsr atnhi
0269  6ca6  4c 01 f3        jmp $f301       ;chkin ende
0270  6ca9  85 95     lf2fe  sta outreg      ;sek adr
0271  6cab  20 aa 6d        jsr send
0272  6cae  ad 00 91        lda portb
0273  6cb1  29 09           and #%00001001  ;nrfd ndac 1
0274  6cb3  8d 00 91        sta portb
0275  6cb6  20 03 6e        jsr atnhi
0276  6cb9  20 95 6d        jsr input
0277  6cbc  4c 01 f3        jmp $f301       ;chkin ende
0278  6cbf
0279  6cbf              ;get character from bus
0280  6cbf
0281  6cbf              getin
0282  6cbf  a5 99           lda indev
0283  6cc1  d0 03           bne lf201
0284  6cc3  4c f9 f1        jmp $f1f9       ;routine fortsetzen
0285  6cc6  c9 02     lf201  cmp #2          ;rs 232 ??
0286  6cc8  d0 12           bne lf21d
0287  6cca  4c 05 f2        jmp $f205       ;routine fortsetzen
0288  6ccd
0289  6ccd              ;input character from bus
0290  6ccd
```

```
0291  6ccd                    chrin
0292  6ccd  a5 99                     lda indev
0293  6ccf  d0 0b                     bne lf21d
0294  6cd1  a5 d3                     lda $d3          ;save screen pointer
0295  6cd3  85 ca                     sta $ca
0296  6cd5  a5 d6                     lda $d6
0297  6cd7  85 c9                     sta $c9
0298  6cd9  4c 4f e6                  jmp $e64f
0299  6cdc  c9 03             lf21d   cmp #3            ;bildschirm
0300  6cde  d0 03                     bne lf22a
0301  6ce0  4c 21 f2                  jmp $f221        ;routine fortsetzen
0302  6ce3  b0 03             lf22a   bcs lf264
0303  6ce5  4c 2c f2                  jmp $f22c        ;rs 232
0304  6ce8  a5 90             lf264   lda status        ;status?
0305  6cea  f0 03                     beq lf26c
0306  6cec  4c 68 f2                  jmp $f268        ;i/o test
0307  6cef                    lf26c
0308  6cef  ad 80 91                  lda portb
0309  6cf2  09 04                     ora #%00000100 ;nrfd-h
0310  6cf4  8d 80 91                  sta portb
0311  6cf7  a9 ff                     lda #255         ;64 ms (timeout)
0312  6cf9  8d 85 91                  sta $9185
0313  6cfc  2c 8d 91          test    bit $918d
0314  6cff  70 3a                     bvs error
0315  6d01  2c 80 91          w1      bit portb         ;wait dav-l
0316  6d04  30 f6                     bmi test
0317  6d06  ad 81 91                  lda porta
0318  6d09  48                        pha
0319  6d0a  ad 80 91                  lda portb
0320  6d0d  29 0b                     and #%00001011 ;nrfd-l
0321  6d0f  8d 80 91                  sta portb
0322  6d12  18                        clc
0323  6d13  ad 80 91                  lda portb
0324  6d16  29 10                     and #%00010000 ;eoi ???
0325  6d18  f0 01                     beq acc3
0326  6d1a  38                        sec
0327  6d1b  b0 04             acc3    bcs acc4      /
0328  6d1d  a9 40                     lda #%01000000 ;status 64
0329  6d1f  85 90                     sta status
0330  6d21  ad 80 91          acc4    lda portb
0331  6d24  09 02                     ora #%00000010 ;ndac-h
0332  6d26  8d 80 91                  sta portb
0333  6d29  2c 80 91          w2      bit portb         ;wait for dav-h
0334  6d2c  10 fb                     bpl w2
0335  6d2e  ad 80 91                  lda portb
0336  6d31  29 0d                     and #%00001101 ;ndac-l
0337  6d33  8d 80 91                  sta portb
0338  6d36  68                        pla
0339  6d37  49 ff                     eor ##ff
0340  6d39  18                        clc
0341  6d3a  60                        rts
0342  6d3b                    error
0343  6d3b  a9 02                     lda #2           ;timeout
0344  6d3d  85 90                     sta status
0345  6d3f  a9 0d                     lda #13          ;cr
0346  6d41  18                        clc
0347  6d42  60                        rts
0348  6d43
```

```
0349  6d43                    ;zuweisung für talk oder listen
0350  6d43                    ;-------------------------------
0351  6d43                    ;(unlisten und untalk) sowie
0352  6d43                    ;kennzeichen des letzen bytes
0353  6d43                    ;mit eoi
0354  6d43
0355  6d43  a9 40             talk    lda #%01000000  ;talk
0356  6d45  2c                       .byt $2c        ;dummy-byte
0357  6d46  a9 20             listen  lda #%00100000  ;listen
0358  6d48  48               unli    pha
0359  6d49  ad 80 91                  lda portb
0360  6d4c  09 06                     ora #%00000110 ;nrfd ndac h
0361  6d4e  8d 80 91                  sta portb
0362  6d51  24 94                     bit lkflag     ;bit 7 = 1 (last byte)
0363  6d53  f0 12                     beq noeoi
0364  6d55  20 09 6e                  jsr eoilo      ;put eoi low
0365  6d58  20 aa 6d                  jsr send       ;send last byte
0366  6d5b  a9 00                     lda #0
0367  6d5d  85 94                     sta lkflag     ;reset last byte flag
0368  6d5f  ad 80 91                  lda portb
0369  6d62  09 01                     ora #%00000001 ;restore eoi
0370  6d64  8d 80 91                  sta portb
0371  6d67  68               noeoi   pla
0372  6d68  05 ba                     ora dn         ;prim adresse
0373  6d6a  85 95                     sta outreg
0374  6d6c  2c 80 91          w8      bit portb       ;wait for dav-h
0375  6d6f  10 fb                     bpl w8
0376  6d71  20 fd 6d                  jsr atnlo
0377  6d74  4c aa 6d                  jmp send
0378  6d77  85 95             second  sta outreg
0379  6d79  20 aa 6d                  jsr send       ;sekundär adresse
0380  6d7c  4c 03 6e                  jmp atnhi
0381  6d7f
0382  6d7f                    ;unterprogramme für bus handling
0383  6d7f
0384  6d7f                    output
0385  6d7f  a9 0f                     lda #$0f       ;port init for output
0386  6d81  8d 82 91                  sta ddrb
0387  6d84  8d 80 91                  sta portb
0388  6d87  a9 ff                     lda #$ff       ;all ports output
0389  6d89  8d 83 91                  sta ddra
0390  6d8c  8d 81 91                  sta porta
0391  6d8f  a9 c0                     lda #write     ;multiplex for write
0392  6d91  8d 8c 91                  sta pcr
0393  6d94  60                       rts
0394  6d95                    input
0395  6d95  a9 0f                     lda #$0f       ;port init for input
0396  6d97  8d 82 91                  sta ddrb
0397  6d9a  a9 09                     lda #%00001001 ;nrfd-l ndac-l
0398  6d9c  8d 80 91                  sta portb
0399  6d9f  a9 00                     lda #0
0400  6da1  8d 83 91                  sta ddra       ;port input mode
0401  6da4  a9 e0                     lda #read
0402  6da6  8d 8c 91                  sta pcr        ;multiplex for input
0403  6da9  60                       rts
0404  6daa
0405  6daa                    ;ausgabe eines bytes aus outreg ($95)
0406  6daa
```

```
0407  6daa  ad 80 91      send    lda portb
0408  6dad  09 08                 ora #%00001000 ;dav high
0409  6daf  8d 80 91              sta portb
0410  6db2  ad 80 91              lda portb
0411  6db5  29 60                 and #%01100000 ;nrfd ndac isolieren
0412  6db7  c9 60                 cmp #%01100000 ;nrfd ndac high ??
0413  6db9  f0 2e                 beq dnperr
0414  6dbb  a5 95                 lda outreg
0415  6dbd  49 ff           send2  eor #$ff
0416  6dbf  8d 81 91              sta porta
0417  6dc2  ad 80 91              lda portb       ;flip dav
0418  6dc5  29 01                 and #%00000001 ;eoi nicht ändern
0419  6dc7  09 06                 ora #%00000110 ;dav low
0420  6dc9  8d 80 91              sta portb
0421  6dcc                 send1
0422  6dcc  ad 80 91              lda portb
0423  6dcf  29 60                 and #%01100000 ;nrfd ndac
0424  6dd1  c9 20                 cmp #%00100000 ;ndac-h ??
0425  6dd3  d0 f7                 bne send1       ;nein dann warten
0426  6dd5  a9 0f                 lda #%00001111
0427  6dd7  8d 80 91              sta portb
0428  6dda  a9 ff                 lda #$ff
0429  6ddc  8d 81 91              sta porta
0430  6ddf                 sample
0431  6ddf  ad 80 91              lda portb
0432  6de2  29 60                 and #%01100000 ;nrfd ndac
0433  6de4  c9 40                 cmp #%01000000 ;nrfd-h???
0434  6de6  d0 f7                 bne sample      ;nein dann warten
0435  6de8  60                    rts
0436  6de9  a9 80           dnperr lda #%10000000
0437  6deb  05 90                 ora status      ;bit 7 setzen
0438  6ded  85 90                 sta status
0439  6def  ad 80 91              lda portb
0440  6df2  09 08                 ora #%00001000 ;dav auf high
0441  6df4  8d 80 91              sta portb
0442  6df7  a9 ff                 lda #$ff
0443  6df9  8d 81 91              sta porta
0444  6dfc  60                    rts
0445  6dfd                 atnlo
0446  6dfd  a9 cc                 lda #write+atnlow
0447  6dff  8d 8c 91              sta pcr
0448  6e02  60                    rts
0449  6e03                 atnhi
0450  6e03  a9 ce                 lda #write+atnhig
0451  6e05  8d 8c 91              sta pcr
0452  6e08  60                    rts
0453  6e09                 eoilo
0454  6e09  ad 80 91              lda portb       ;set eoi  low
0455  6e0c  29 fe                 and #$fe
0456  6e0e  8d 80 91              sta portb
0457  6e11  60                    rts
0458  6e12                 untal
0459  6e12  a9 5f                 lda #untalk     ;send untalk
0460  6e14  20 48 6d              jsr unli
0461  6e17  4c 03 6e              jmp atnhi       ;atn-h
0462  6e1a                 unlis
0463  6e1a  a9 3f                 lda #unlisn     ;send unlisn
0464  6e1c  20 48 6d              jsr unli
0465  6e1f  4c 03 6e              jmp atnhi
```

```
0467  6e22                    ;test auf floppy fehler nach open
0468  6e22
0469  6e22                    check
0470  6e22  20 1a 6e               jsr unlis
0471  6e25  20 fd 6d               jsr atnlo
0472  6e28  a9 48                  lda #$48          ;talk device
0473  6e2a  20 bd 6d               jsr send2         ;direkt zum port
0474  6e2d  a9 6f                  lda #$6f          ;error channel
0475  6e2f  20 bd 6d               jsr send2
0476  6e32  20 03 6e               jsr atnhi         ;end of command
0477  6e35  20 95 6d               jsr input         ;listen
0478  6e38  20 ef 6c               jsr lf26c         ;get char
0479  6e3b  c9 30                  cmp #"0           ;error null (ok)
0480  6e3d  f0 23                  beq ok
0481  6e3f  48                     pha
0482  6e40  a9 0d                  lda #13           ;return
0483  6e42  20 d2 ff               jsr $ffd2
0484  6e45  68                     pla
0485  6e46  20 d2 ff               jsr $ffd2         ;erstes zeichen schreibe
0486  6e49  a9 00                  lda #0
0487  6e4b  85 90                  sta status        ;reset status
0488  6e4d  20 ef 6c    errlop     jsr lf26c          ;get byte
0489  6e50  20 d2 ff               jsr $ffd2
0490  6e53  a5 90                  lda status
0491  6e55  c9 40                  cmp #64           ;last byte received
0492  6e57  d0 f4                  bne errlop
0493  6e59  20 7f 6d               jsr output        ;port to output
0494  6e5c  20 69 6e               jsr zu15
0495  6e5f  4c 67 e4               jmp $e467         ;restart basic
0496  6e62  20 7f 6d    ok         jsr output
0497  6e65  20 1a 6e               jsr unlis
0498  6e68  60                     rts
0499  6e69  20 fd 6d    zu15       jsr atnlo
0500  6e6c  a9 28                  lda #$28          ;close error channel
0501  6e6e  20 bd 6d               jsr send2
0502  6e71  a9 ef                  lda #$ef
0503  6e73  20 bd 6d               jsr send2
0504  6e76  4c 1a 6e               jmp unlis
0505  6e79
0506  6e79                    ; load und save routine vc-20
0507  6e79
0508  6e79                    load
0509  6e79  85 93                  sta $93
0510  6e7b  a9 00                  lda #$00          ;status löschen
0511  6e7d  85 90                  sta status
0512  6e7f  a5 ba                  lda dn            ;device number
0513  6e81  d0 03                  bne lf556
0514  6e83  4c 96 f7    lf553      jmp $f796          ;illegal device
0515  6e86
0516  6e86  c9 03       lf556      cmp #$03           ;bildchirm ??
0517  6e88  f0 f9                  beq lf553         ;illegal device
0518  6e8a  90 74                  bcc lf5ca         ;cassette
0519  6e8c  a4 b7                  ldy fnlen         ;filename ??
0520  6e8e  d0 03                  bne lf563
0521  6e90  4c 93 f7               jmp $f793         ;missing filename
0523  6e93  20 bc e4    lf563      jsr $e4bc          ;searching for
0524  6e96  a9 60                  lda #$60
0525  6e98  85 b9                  sta sa            ;sec adr = 60/lesen
```

```
0526  6e9a  20 7b 6b          jsr chopen      ;open ieee
0527  6e9d  a5 ba             lda dn
0528  6e9f  20 43 6d          jsr talk        ;set dev to talker
0529  6ea2  a5 b9             lda sa
0530  6ea4  20 77 6d          jsr second      ;sec adr senden
0531  6ea7  20 95 6d          jsr input       ;flip ports
0532  6eaa  20 ef 6c          jsr lf26c       ;1. byte holen
0533  6ead  85 ae             sta $ae
0534  6eaf  a5 90             lda status      ;test status 64
0535  6eb1  4a                lsr a
0536  6eb2  4a                lsr a
0537  6eb3  b0 48             bcs lf5c7
0538  6eb5  20 ef 6c          jsr lf26c       ;2. byte (addr)
0539  6eb8  85 af             sta $af
0540  6eba  20 c1 e4          jsr $e4c1       ;pointer-loading
0541  6ebd  a9 fd          lf58a lda ##fd
0542  6ebf  25 90             and status
0543  6ec1  85 90             sta status
0544  6ec3  20 e1 ff          jsr $ffe1       ;stop taste
0545  6ec6  d0 03             bne lf598
0546  6ec8  4c 60 6f          jmp lf6cb       ;close file
0547  6ecb
0548  6ecb  20 ef 6c       lf598 jsr lf26c      ;get data byte
0549  6ece  aa                tax
0550  6ecf  a5 90             lda status
0551  6ed1  4a                lsr a
0552  6ed2  4a                lsr a
0553  6ed3  b0 e8             bcs lf58a       ;next byte
0554  6ed5  8a                txa
0555  6ed6  a4 93             ldy $93         ;verify flag
0556  6ed8  f0 0c             beq lf5b3
0557  6eda  a0 00             ldy ##00
0558  6edc  d1 ae             cmp ($ae),y     ;compare
0559  6ede  f0 08             beq lf5b5
0560  6ee0  a9 10             lda ##10
0561  6ee2  20 6a fe          jsr $fe6a       ;set bit 4 in status
0562  6ee5  2c                .byt $2c
0563  6ee6  91 ae          lf5b3 sta ($ae),y    ;store to mem
0564  6ee8  e6 ae          lf5b5 inc $ae        ;count pointer
0565  6eea  d0 02             bne lf5bb
0566  6eec  e6 af             inc $af
0567  6eee  24 90          lf5bb bit status     ;test status
0568  6ef0  50 cb             bvc lf58a       ;next byte
0569  6ef2  20 7f 6d          jsr output
0570  6ef5  20 12 6e          jsr untal       ;ready? >send untalk
0571  6ef8  20 6f 6f          jsr lf6da       ;close file
0572  6efb  90 06             bcc lf641       ;end of load
0573  6efd  4c 87 f7       lf5c7 jmp $f787     ;file not found
0574  6f00  4c ca f5       lf5ca jmp $f5ca     ;weiter mit tape
0575  6f03  4c 41 f6       lf641 jmp $f641     ;end of load
0576  6f06  a4 b7          lf659 ldy fnlen
0577  6f08  f0 0c             beq lf669
0578  6f0a  a0 00             ldy #$00
0579  6f0c  b1 bb          lf65f lda (fnlo),y
0580  6f0e  20 d2 ff          jsr $ffd2
0581  6f11  c8                iny
0582  6f12  c4 b7             cpy fnlen
0583  6f14  d0 f6             bne lf65f
0584  6f16  60             lf669 rts
0585  6f17
```

```
0586  6f17                    save
0588  6f17  a5 ba                   lda dn          ;device ??
0589  6f19  d0 03                   bne lf68c
0590  6f1b  4c 96 f7      lf689     jmp $f796        ;illegal device
0591  6f1e  c9 03        lf68c     cmp #$03         ;bildschirm
0592  6f20  f0 f9                   beq lf689       ;illegeal device
0593  6f22  90 62                   bcc lf6f1       ;tape
0594  6f24  a9 61                   lda #$61        ;sec adr = 61/schreiben
0595  6f26  85 b9                   sta sa
0596  6f28  a4 b7                   ldy fnlen       ;test file name
0597  6f2a  d0 03                   bne lf69d
0598  6f2c  4c 93 f7                jmp $f793       ;missing file name
0599  6f2f  20 7b 6b      lf69d     jsr chopen       ;open file
0600  6f32  20 22 6e                jsr check       ;error ??
0601  6f35  20 8a 6f                jsr lf728       ;writing
0602  6f38  a5 ba                   lda dn
0603  6f3a  20 46 6d                jsr listen      ;listener
0604  6f3d  a5 b9                   lda sa
0605  6f3f  20 77 6d                jsr second      ;sec adr senden
0606  6f42  a0 00                   ldy #$00
0607  6f44  20 d2 fb                jsr $fbd2       ;tape adr. retten
0608  6f47  a5 ac                   lda $ac
0609  6f49  20 40 6c                jsr ciout       ;ausgabe der start-
0610  6f4c  a5 ad                   lda $ad
0611  6f4e  20 40 6c                jsr ciout       ;adresse auf den bus
0612  6f51  20 11 fd      lf6bc     jsr $fd11        ;adr. f. save und load
0613  6f54  b0 16                   bcs lf6d7
0614  6f56  b1 ac                   lda ($ac),y     ;put mem to bus
0615  6f58  20 40 6c                jsr ciout
0616  6f5b  20 e1 ff                jsr $ffe1       ;test auf stop
0617  6f5e  d0 07                   bne lf6d2
0618  6f60  20 6f 6f      lf6cb     jsr lf6da        ;close file
0619  6f63  a9 00                   lda #$00
0620  6f65  38                      sec
0621  6f66  60                      rts
0622  6f67  20 1b fd      lf6d2     jsr $fd1b        ;count save adr.
0623  6f6a  d0 e5                   bne lf6bc       ;next byte
0624  6f6c  20 1a 6e      lf6d7     jsr unlis        ;unlisn to bus
0625  6f6f  24 b9         lf6da     bit sa
0626  6f71  30 11                   bmi lf6ef
0627  6f73  a5 ba                   lda dn
0628  6f75  20 46 6d                jsr listen      ;listen
0629  6f78  a5 b9                   lda sa
0630  6f7a  29 ef                   and #$ef
0631  6f7c  09 e0                   ora #$e0
0632  6f7e  20 77 6d                jsr second      ;sec. adr. senden
0633  6f81  20 1a 6e                jsr unlis       ;unlisn- bus closed
0634  6f84  18            lf6ef     clc
0635  6f85  60                      rts
0636  6f86  4c f1 f6      lf6f1     jmp $f6f1        ;weiter mit tape
0637  6f89  60            lf727     rts
0638  6f8a  a5 9d         lf728     lda $9d
0639  6f8c  10 fb                   bpl lf727
0640  6f8e  a0 51                   ldy #$51        ;message
0641  6f90  20 e6 f1                jsr $f1e6
0642  6f93  4c 06 6f                jmp lf659
0643  6f96
0645  6f96                    .end                                    □□
```

Andreas Zilker, 8300 Landshut

Hi-Plot

Hochauflösendes Plotprogramm für Grafik und mehrdimensionale Funktionen

Das nachstehende Programm lehnt sich an das 'Grafik-Plot' von F. Geissler in den Heften 23 und 24 dieser Zeitschrift an, das auf den Thermodrucker des AIM 65 zugeschnitten war. Hier nun wird auf einen breiten Drucker mit einer Auflösung von 256x256 oder auch 512x512 Bildpunkten ausgegeben, im speziellen Fall auf einen NEC 8023.

Im BASIC-Teil wurden unnötige 'REM's an zeitintensiven Stellen entfernt und, wo sinnvoll, Konstanten durch Variable ersetzt. Die Rechenintervalle wurden an das vergrößerte Format angepaßt. Bei Grobauflösung ist mit einer Rechenzeit von ca. 10 Min. zu rechnen (1 MHz Takt), im Fein-Modus 'verbrät' das Programm auch einmal eine Stunde CPU-Zeit (je nach Funktion). Es ist daher empfehlenswert, um schöne Plots zu erhalten, die Funktion mathematisch nach Extrema, Grenzwerten und Polen abzusuchen und von Anfang an vernünftige Rechenintervalle zu benutzen. Der Autor arbeitet auch schon an einer schnelleren FORTH-Version. Nun zum Assembler-Teil:

Der Teil von SET bis STOP wurde im wesentlichen nur an das größere Format angepaßt. Er verwendet einen 8K-Bildspeicher, der zwischen TABAN und TABEND irgendwo im RAM liegen kann. Die Ausgabe des Speichers dürfte auf jeden Matrixdrucker mit Einzelnadelsteuerung möglich sein. Hier wird ein NEC 8023 mit Centronix-Schnittstelle an der User–VIA betrieben (siehe MICRO MAG Nr. 22, S. 30). Die Routine NECINI initialisiert die Schnittstelle und selektiert den Drucker, NECPR2+1 gibt ein Zeichen im Akku auf die Schnittstelle aus und NECOFF schiebt ein Formfeed nach und deselektiert den Drucker. GRAPHM ist speziell auf die Grafiksteuerung des NEC ausgelegt und muß für andere Drucker entsprechend angepaßt werden (GRATB: ESC/T legt den Zeilenabstand fest, ESC/S schaltet auf 256 Zeichen im Graphik-Mode). Der NEC verarbeitet die Bits außerdem 'kopfstehend', deshalb wurde ROLP eingefügt, um ein Byte aus dem Bildspeicher zu stürzen.

Da ein Bild mit 256x256 Punkten auf dem großen Drucker recht mager aussieht, wurde Die Routine PRINT2 geschrieben. Sie ermöglicht den Ausdruck eines 512x512-Formates, ohne die BASIC-Zeit zu verlängern. Wenn die Frage 'Großformat?' im BASIC-Teil mit 'J' beantwortet wurde, dann wird diese Routine verwendet. Sie benötigt zwei zusätzliche Buffer CBUF und PBUF von 1 bzw. 1/4K Länge. Zwischen die berechneten Bildpunkte werden nun jeweils in X- und Y-Richtung 0-Bits eingeschoben, die so die Bildfläche vervierfachen.

Das Programm wurde 'straightforward' und ohne große Tricks geschrieben, ist nahezu selbstdokumentierend. Bezüglich der druckerspezifischen Kommandos gilt das vorher Gesagte (in GRATAB zusätzlich ESC/L zur Randformatierung).

Wer so ungern wie der Autor eintippt, erhält gegen DM 5,-- eine AIM-Cassette. Anschrift: Schoeplergasse 20 b, 8300 Landshut 1.

```
1 REM PLOTPROGRAMM FUER MEHRDIMENSIONALE FUNKTIONEN
3 REM VERSION 2.1
4 REM NACH F. GEISSLER (OPTIMIERT)
5 REM AUFLOESUNG 256*256 PUNKTE
10 IL=42104:IT=42103:IS=42102:CO=0.38:SI=0.5
15 REM
20 REM-LOESCHEN
30 REM
40 POKE4,60:POKE5,88:Q=USR(0)
50 REM
60 REM-EINGABE
70 REM
75 PRINT"FUNKTION IN 2000 !"
```

```
80 INPUT"X - BEREICH";A,B:IFA>=BTHEN80
90 INPUT"Y-BEREICH";C,D:IFC>=DTHEN90
100 INPUT"Z-BEREICH";E,F:IFE>=FTHEN100
110 XS=(B-A)/205:YS=(D-C)/128:ZS=(F-E)/166
120 XN=-A/XS:YN=-C/YS:ZN=-E/ZS
130 PRINT"  GROB ?  (J/N)"
135 GETA$:IFA$="J"GOTO500
140 IF A$<>"N"GOTO135
150 REM
160 REM-MAXIMALE PUNKTDICHTE
170 REM
180 FORX=ATOBSTEPXS
190 PRINTINT((X-A)*.488/XS*10)/10;"%BERECHNET"
200 FORY=DTOCSTEP-YS:GOSUB2000
250 POKEIL,X1:POKEIT,Y2:POKEIS,Y1
260 POKE4,109:Q=USR(0)
300 POKEIT,Y1:POKE4,90:Q=USR(0)
320 Y2=Y1:NEXTY,X
330 REM
340 REM-AUSGABE DES BILDES
350 REM
360 PRINT"GROSSFORMAT ? (J/N)"
365 GETA$
370 IFA$="J"THENG=245:GOTO390
375 IFA$<>"N"GOTO365
380 G=166
390 POKE4,G:Q=USR(0)
400 PRINT" FERTIG !":END
500 REM
510 REM-GROBES NETZ FUER UEBERBLICK
520 REM
530 PRINT"AUGENBLICK BITTE !"
540 FORX=ATOBSTEPXS*5
550 FORY=DTOCSTEP-YS:GOSUB2000
570 POKEIT,Y1:POKEIL,X1:POKE4,90:Q=USR(0)
580 NEXTY,X
590 FORX=ATOBSTEPXS
600 FORY=DTOCSTEP-YS*5:GOSUB2000
610 POKEIT,Y1:POKEIL,X1:Q=USR(0)
620 NEXTY,X
630 GOTO330
2000 Z=X*X+Y*Y:IFZ=0THENZ=1:GOTO2100
2010 Z=SIN(Z)/Z
2060 IFZ>FTHENZ=F:IFZ<ETHENZ=E
2080 X1=X/XS+XN:Y1=Y/YS+YN:Z1=Z/ZS+ZN
2090 X1=INT(X1+CO*Y1+SI):Y1=INT(Z1+Y1+SI)
2100 RETURN

0000          ; ==================
0000          ; =  P L O T - 5  =
0000          ; ==================
0000
0000          ; VERSION 2.3
0000          ; PLOTPROGRAMM FUER GRAPHIK UND
0000          ; MEHRDIMENSIONALE FUNKTIONEN
0000          ; FUER NEC 8023 PRINTER
0000          ; AUFLOESUNG 256*256 PUNKTE
0000          ; AUSDRUCK AUCH 512*512 PUNKTE
0000          ; NACH F. GEISSLER (MM 23/43)
```

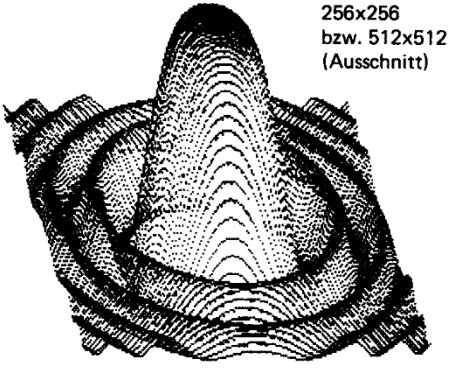
```
0000 NECINI          =$9249          ;DRUCKER-ROUTINEN
0000 NECOFF          =$9257
0000 NECPR1          =$925C
0000 NECPR2          =$9265
0000
0000 TABAN           =$60            ;TABELLENANFANG
0000 TABEND          =$80            ;UND ENDE
0000 CBUF            =$5B00          ;PUFFER FUER ORGINAL-BILDZEILE
0000 PBUF            =$5C00          ;PUFFER FUER 1024 BILD-BYTES
0000
0000                 *=$00A0
00A0 BYTE            *=*+2           ;POINTER AUF BILDSPEICHER
00A2 YBYTE           *=*+1           ;BYTE-POSITION
00A3 YBIT            *=*+1           ;BIT-MASKE
00A4 YBIT1           *=*+1           ;BIT-POSITION
00A5 PPNT                            ;POINTER AUF PRINTER-PUFFER
00A5
00A5                 *=$A476
A476 IOFFS           *=*+1           ;Y2,
A477 IDOT            *=*+1           ;Y1 UND
A478 IOUTL           *=*+1           ;X AUS BASIC
A479
A479                 *=$5800
5800 SET      A960   LDA #TABAN      ;WANDELT X- UND Y-WERT IN
5802          85A1   STA BYTE+1      ;EINE ADRESSE FUER DEN
5804          AD77A4 LDA IDOT        ;BILD-SPEICHER UM
5807          4A     LSR A
5808          4A     LSR A
5809          4A     LSR A
580A          85A0   STA BYTE        ;BYTE=INT(Y/8)
580C          0A     ASL A
580D          0A     ASL A
580E          0A     ASL A
580F          85A4   STA YBIT1
5811          AD77A4 LDA IDOT
5814          38     SEC
5815          E5A4   SBC YBIT1
5817          85A4   STA YBIT1       ;BIT-NUMMER IN YBIT1
5819          A900   LDA #0
581B          85A2   STA YBYTE
581D K3       A5A0   LDA BYTE        ;BERECHNUNG DER EFFEKTIVEN
581F          F007   BEQ K2          ;BASIS-ADRESSE FUER EINEN BILDP
5821          E6A1   INC BYTE+1
5823          C6A0   DEC BYTE
5825          4C1D58 JMP K3
5828 K2       A901   LDA #01         ;UMWANDELN DER BIT-POSITION
582A          85A3   STA YBIT        ;IN EINE MASKE
582C K1       A5A4   LDA YBIT1
582E          F007   BEQ K4
5830          06A3   ASL YBIT
5832          C6A4   DEC YBIT1
5834          4C2C58 JMP K1
5837 K4       A5A2   LDA YBYTE
5839          85A0   STA BYTE
583B          60     RTS
583C
583C LOE      A220   LDX #32         ;LOESCHEN DES GESAMTEN
583E          A0FF   LDY #255        ;BILDSPEICHERS
```

```
5840           A900    LDA #0
5842           85A0    STA BYTE
5844           A960    LDA #TABAN
5846           85A1    STA BYTE+1        ;POINTER SETZEN
5848 F1        A900    LDA #0
584A           91A0    STA (BYTE),Y
584C           88      DEY
584D           C0FF    CPY #$FF          ;EINE PAGE ?
584F           D0F7    BNE F1
5851           CA      DEX
5852           3005    BMI ENDE
5854           E6A1    INC BYTE+1
5856           4C4858  JMP F1
5859 ENDE      60      RTS
585A
585A SETZE     AD77A4  LDA IDOT          ;SETZEN EINES PUNKTES (X,Y)
585D           AD78A4  LDA IOUTL         ;IM BILDSPEICHER
5860           200058  JSR SET
5863           AC78A4  LDY IOUTL         ;PARAMETER AUS "SET"
5866           B1A0    LDA (BYTE),Y
5868           05A3    ORA YBIT
586A           91A0    STA (BYTE),Y
586C           60      RTS
586D
586D LOET      AD77A4  LDA IDOT
5870           CD76A4  CMP IOFFS         ;Y1<Y2 ?
5873           300A    BMI Q3
5875           A8      TAY               ;VERTAUSCHEN VON Y1 UND Y2
5876           AD76A4  LDA IOFFS
5879           8D77A4  STA IDOT
587C           8C76A4  STY IOFFS
587F Q3        CE76A4  DEC IOFFS
5882           EE77A4  INC IDOT
5885 Q4        AD77A4  LDA IDOT          ;LOESCHEN EINER SENKRECHTEN
5888           CD76A4  CMP IOFFS         ;LINIE AN DER STELLE X
588B           1018    BPL STOP          ;VON Y1 BIS Y2 (AUSSCHL.)
588D           200058  JSR SET
5890           A5A3    LDA YBIT
5892           49FF    EOR #$FF
5894           85A3    STA YBIT
5896           AC78A4  LDY IOUTL
5899           B1A0    LDA (BYTE),Y
589B           25A3    AND YBIT
589D           91A0    STA (BYTE),Y
589F           EE77A4  INC IDOT
58A2           4C8558  JMP Q4
58A5 STOP      60      RTS
58A6
58A6 PRINT     A900    LDA #0            ;AUSGABE AUF NEC-PRINTER
58A8           85A0    STA BYTE          ;FORMAT 256*256 PUNKTE
58AA           A97F    LDA #TABEND-1
58AC           85A1    STA BYTE+1        ;"VON OBEN NACH UNTEN"
58AE           A0FF    LDY #255
58B0           204992  JSR NECINI        ;PRINTER ON
58B3 PRTLP1    20DD58  JSR GRAPHM        ;GRAPHIK-MODUS INITIALISIEREN
58B6 PRTLP2    B1A0    LDA (BYTE),Y
58B8           A208    LDX #8
58BA ROLP      0A      ASL A             ;BYTE STUERZEN
58BB           6E76A4  ROR IOFFS
```

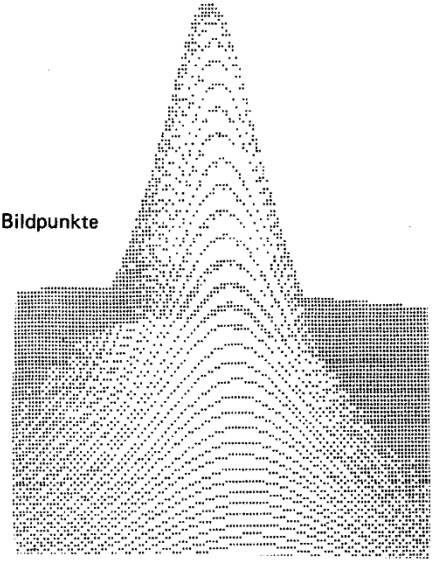
```
58BE            CA      DEX
58BF            D0F9    BNE ROLP
58C1            AD76A4  LDA IOFFS
58C4            206592  JSR NECPR2
58C7            88      DEY
58C8            C0FF    CPY #255
58CA            D0EA    BNE PRTLP2
58CC            A90D    LDA #$0D
58CE            205C92  JSR NECPR1
58D1            C6A1    DEC BYTE+1
58D3            A6A1    LDX BYTE+1
58D5            E060    CPX #TABAN      ;GANZER BILDSPEICHER ?
58D7            10DA    BPL PRTLP1
58D9            4C5792  JMP NECOFF      ;PRINTER AUS UND ZURUECK INS
58DC            60      RTS                                   BASIC
58DD GRAPHM     A200    LDX #0          ;STEUERZEICHEN FUER GRAPHIK
58DF GRALP      BDEB58  LDA GRATAB,X
58E2            206592  JSR NECPR2
58E5            E8      INX
58E6            E00A    CPX #LENGRA
58E8            30F5    BMI GRALP
58EA            60      RTS
58EB LENGRA=10
58EB GRATAB     1B      .BYT $1B,'T16',$1B,'S0256'
58EC            543136
58EF            1B
58F0            5330
58F5
58F5 PRINT2     A900    LDA #0          ;AUSGABE IM FORMAT
58F7            85A0    STA BYTE        ;512*512 PUNKTE MIT LEERRAUM
58F9            A97F    LDA #TABEND-1
58FB            85A1    STA BYTE+1
58FD            A0FF    LDY #255
58FF            204992  JSR NECINI
5902 STOLP      B1A0    LDA (BYTE),Y
5904            A208    LDX #8
5906 ROLP2      0A      ASL A
5907            6E76A4  ROR IOFFS
590A            CA      DEX
590B            D0F9    BNE ROLP2
590D            AD76A4  LDA IOFFS
5910            99005B  STA CBUF,Y      ;PUFFERN STATT AUSDRUCKEN
5913            88      DEY
5914            C0FF    CPY #255
5916            D0EA    BNE STOLP
5918            A95C    LDA #>PBUF      ;LOESCHEN DES GESAMTEN
591A            85A6    STA PPNT+1      ;DRUCKER-PUFFERS
591C            A000    LDY #0
591E            84A5    STY PPNT
5920            98      TYA
5921 CLRLP      91A5    STA (PPNT),Y
5923            C8      INY
5924            D0FB    BNE CLRLP
5926            E6A6    INC PPNT+1
5928            A6A6    LDX PPNT+1
592A            E060    CPX #>PBUF+4
592C            D0F3    BNE CLRLP
592E            A95C    LDA #>PBUF
```

```
5930            85A6    STA PPNT+1
5932            84A5    STY PPNT
5934 BLOLP      B9005B  LDA CBUF,Y      ;EINSCHIEBEN VON "0"-BYTES
5937            91A5    STA (PPNT),Y    ; ==> 512 ZEICHEN
5939            E6A5    INC PPNT
593B            C8      INY
593C            D0F6    BNE BLOLP
593E            A200    LDX #0
5940 DOLP1      A004    LDY #4          ;EINSCHIEBEN VON "0"-BITS
5942            BD005C  LDA PBUF,X      ; ==> 1024 ZEICHEN
5945 DOLP2      0A      ASL A
5946            3E005E  ROL PBUF+$200,X
5949            18      CLC
594A            3E005E  ROL PBUF+$200,X
594D            88      DEY
594E            D0F5    BNE DOLP2
5950            E8      INX
5951            E8      INX
5952            D0EC    BNE DOLP1
5954 DOLP3      A004    LDY #4          ;VIER "0"-BITS PRO HALBBYTE
5956            BD005D  LDA PBUF+$100,X
5959 DOLP4      0A      ASL A           ;NACH "OBEN" SCHIEBEN
595A            3E005F  ROL PBUF+$300,X
595D            18      CLC             ;"0"-BIT NACHSCHIEBEN
595E            3E005F  ROL PBUF+$300,X
5961            88      DEY
5962            D0F5    BNE DOLP4
5964            E8      INX             ;"0"-BYTES UEBERSPRINGEN
5965            E8      INX
5966            D0EC    BNE DOLP3
5968 DOLP5      A004    LDY #4
596A            BD005C  LDA PBUF,X
596D DOLP6      18      CLC             ;NACH "UNTEN" SCHIEBEN
596E            7E005C  ROR PBUF,X
5971            4A      LSR A
5972            7E005C  ROR PBUF,X      ;ADRESSIERUNG NUR "ABS,X"
5975            88      DEY
5976            D0F5    BNE DOLP6
5978            E8      INX
5979            E8      INX
597A            D0EC    BNE DOLP5
597C DOLP7      A004    LDY #4
597E            BD005D  LDA PBUF+$100,X
5981 DOLP8      18      CLC
5982            7E005D  ROR PBUF+$100,X
5985            4A      LSR A
5986            7E005D  ROR PBUF+$100,X
5989            88      DEY
598A            D0F5    BNE DOLP8
598C            E8      INX
598D            E8      INX
598E            D0EC    BNE DOLP7
5990            88      DEY             ;Y:=255 !
5991            A900    LDA #0
5993            85A5    STA PPNT
5995            A95D    LDA #>PBUF+1    ;UNTERE HAELFTE DES PUFFERS
5997            85A6    STA PPNT+1                      AUSGEBEN
5999            20CD59  JSR GRAPM2
```

```
599C            20C259  JSR PROUT
599F            C6A6    DEC PPNT+1
59A1            20C259  JSR PROUT
59A4            A95F    LDA #>PBUF+3      ;OBERE HAELFTE AUSGEBEN
59A6            85A6    STA PPNT+1
59A8            20CD59  JSR GRAPM2
59AB            20C259  JSR PROUT
59AE            C6A6    DEC PPNT+1
59B0            20C259  JSR PROUT
59B3            C6A1    DEC BYTE+1
59B5            A6A1    LDX BYTE+1
59B7            E060    CPX #TABAN        ;GANZER BILDSPEICHER
59B9            3003    BMI LPEND          AUSGEGEBEN?
59BB            4C0259  JMP STOLP
59BE LPEND      205792  JSR NECOFF
59C1            60      RTS
59C2 PROUT      B1A5    LDA (PPNT),Y      ;EINE PAGE AUSDRUCKEN
59C4            206592  JSR NECPR2
59C7            88      DEY
59C8            C0FF    CPY #255
59CA            D0F6    BNE PROUT
59CC            60      RTS
59CD GRAPM2     A90D    LDA #$0D          ;GRAPHIK-INIT. FUER GROSSFORMAT
59CF            205C92  JSR NECPR1
59D2            A200    LDX #0
59D4 GRALP2     BDE059  LDA GRATB2,X
59D7            206592  JSR NECPR2
59DA            E8      INX
59DB            E00F    CPX #LENGR2
59DD            30F5    BMI GRALP2
59DF            60      RTS
59E0 LENGR2             =15
59E0 GRATB2     1B      .BYT $1B,'L006',$1B,'T16',$1B,'S0512'
59E1            4C30
59E5            1B
59E6            543136
59E9            1B
59EA            5330
59EF                    .END
59EF             ERRORS= 0000
```



256x256
bzw. 512x512 Bildpunkte
(Ausschnitt)



Dipl.-Ing. Rüdiger Wollenberg und Dipl.-Phys. Artur Redder, 4600 Dortmund 50

Schnelles Replace für AIM 65

Nachdem durch viele in der letzten Zeit erschienene Veröffentlichungen /1,2,3/ das Tor zur Textverarbeitung am AIM 65 erheblich stärker geöffnet wurde, ist ein weiterer Mangel offenkundig geworden: Das Zeileninsert am Anfang eines sehr langen Quelltextes erfordert schon mehrere Sekunden. Ähnlich ist es mit dem Zeilenkill. Auch bei Verwendung des SCREEN-Editors /2/ treten bei vielen Kommandos diese Schwachstellen zutage. Die Ursache liegt in der Replace-Routine des Editors, die den nötigen Platz für eine längere Zeile schafft bzw. eine Textverkürzung vornimmt, bevor die alte Zeile des Textspeichers durch den Inhalt des DIBUFF ausgetauscht wird.

Im einzelnen wurden Änderungen vorgenommen, die dazu führten, daß die Verarbeitungsgeschwindigkeit um den Faktor drei erhöht werden konnte, ohne daß zusätzlicher Speicherplatz benötigt wird. Im Gegenteil: In der schnellen Version sind 26 Bytes unbenutzt.

In Fortführung des REMON-Konzepts /1/ muß die verbesserte Routine an die Stelle der alten gesetzt sein, indem der Original-Monitor durch zwei 2532-EPROMs ausgetauscht wird. Ab der REMON-Version 3.5 der Autoren ist diese Änderung standardmäßig mit aufgenommen.

Die Verbesserungen betreffen im einzelnen:

Der Aufruf von LDAY, der ein LDA (NNNN),Y mit einer Absolut-Speicheradresse gestattet, wird ersetzt durch den erheblich schneller wirkenden Zeropagepointer SAVE, der lediglich als Zwischenspeicher dient und für die REPLAC-Routine ohne Einschränkung genutzt werden kann.

Die Erhöhung bzw. Ernidedrigung der laufenden Pointer, die für die Textverschiebung benötigt werden, erfolgt nicht mehr durch Unterprogrammaufrufe, sondern ist linear ausprogrammiert. Dies konnte allerdings erheblich kürzer und damit schneller geschehen, als es im AIM-Monitor durchgeführt ist.

Die Überprüfung auf eine richtige Abspeicherung mittels der Routine GOGO wurde derart modifiziert, daß die Vergleichsabfrage ebenfalls linear in das REPLAC einprogrammiert wurde. Damit ist wieder ein kleiner Zeitvorteil erreicht.

Die Speicherplatzreduzierung trotz der vielen Ergänzungen wurde erreicht durch eine neue Strukturierung der REPLAC-Routine. Unökonomische Befehle sind weggefallen. Platz hat auch die Weglassung von unsinnigen Befehlen geschaffen (z.B. ein TAX, obwohl das X-Register gar nicht benutzt wird oder ein CMP#0 nach einem LDA-Befehl, der das Zero-Flag automatisch beeinflußt).

Die Wirksamkeit des schnellen Replace demonstriert der folgende Vergleich, der in der TOP-Zeile eines 30-KByte langen Textes durchgeführt wurde:

	AIM 65-REPLAC	Schnelles REPLAC
Zeileninsert	4.3 sec	1.6 sec
Zeilenkill	4.0 sec	1.4 sec

Das REPLAC ist eine in sich geschlossene Programmeinheit, die lediglich vom Editor genutzt wird. Deshalb wirkt sich die Änderung auch nicht auf die Compiler, Interpreter und anderen Softwareprogramme aus.

Literatur:

/1/ Wollenberg R., Redder A.: REMON - Redigierter Monitor. 65xx MICRO MAG Nr. 28, S. 3-14.
/2/ Wollenberg R., Redder A.: SCREEN - Bildschirmeditor für AIM 65. 65xx MICRO MAG Nr. 30, S. 10-22.
/3/ Redder A., Wollenberg R.: EXEDIT - Extended Editor für AIM 65. 65xx MICRO MAG Nr. 31.
/4/ Rockwell International: Monitor Program Listing.

```
**********************************************************
SCHNELLES REPLAC   V1.6  26.6.1983  WOLLENBERG/REDDER

0000 CR            =$D
0000 DIBUFF        =$A438
0000 SAVNOW        =$F934
0000 RESNOW        =$F8D0
0000 ADDR          =$A41C
0000 MEMERR        =$EB33
0000 SETBOT        =$F8C5
0000 ATEND         =$F8F9
0000 ENDERR        =$FA5C
0000 COUNT         =$A419
0000 CPIY          =$A42A
0000
0000               *=$DF
00DF NOWLN         *=*+2
00E1 BOTLN         *=*+2
00E3 TEXT          *=*+2
00E5 END           *=*+2
00E7 SAVE          *=*+2
00E9 OLDLEN        *=*+1
00EA LENGTH        *=*+1
00EB
00EB
00EB               *=$F93F
F93F         ;MOVE CURRENT TEXT AROUND TO HAVE
F93F         ;SPACE TO PUT IN THE NEW BUFFER
F93F REPLAC 2034F9 JSR SAVNOW      ;SAVE OLD NOWLN
F942        A4EA   LDY LENGTH
F944        C4E9   CPY OLDLEN
F946        F06A   BEQ R87
F948        B07A   BCS R100
F94A
F94A        ;LENGTH ( OLDLEN
F94A        A5E0   LDA NOWLN+1     ;NOWLN TO SAVE
F94C        85E8   STA SAVE+1
F94E        A5E9   LDA OLDLEN
F950        38     SEC
F951        E5EA   SBC LENGTH      ;GET DIFFERENCE IN LENGTHS
F953        A4EA   LDY LENGTH
F955        D007   BNE RQP
F957        AE19A4 LDX COUNT       ;C-COMM?
F95A        D002   BNE RQP         ;YES, JUMP
F95C        6900   ADC #0          ;INCLUDE (CR)
F95E RQP    8D2AA4 STA CPIY
F961        18     CLC
F962        65DF   ADC NOWLN
F964        85E7   STA SAVE
F966        9002   BCC R6
F968        E6E8   INC SAVE+1
F96A R6     B1E7   LDA (SAVE),Y
F96C        91DF   STA (NOWLN),Y   ;MOVE IT UP (DOWN IN ADDR)
F96E        D1DF   CMP (NOWLN),Y
F970        D03E   BNE R97
F972 R6B    A5E7   LDA SAVE
F974        C5E1   CMP BOTLN       ;DONE?
```

```
F976          D006    BNE R5
F978          A5E8    LDA SAVE+1
F97A          C5E2    CMP BOTLN+1
F97C          F00F    BEQ R7
F97E R5       E6DF    INC NOWLN
F980          D002    BNE R51
F982          E6E0    INC NOWLN+1
F984 R51      E6E7    INC SAVE
F986          D002    BNE R55
F988          E6E8    INC SAVE+1
F98A R55      4C6AF9  JMP R6
F98D R7       20D0F8  JSR RESNOW      ;RESTORE NOWLN
F990          A5E1    LDA BOTLN       ;CALCULATE NEW BOTTOM
F992          38      SEC
F993          ED2AA4  SBC CPIY
F996          85E1    STA BOTLN
F998          B002    BCS R9
F99A          C6E2    DEC BOTLN+1
F99C R9       AD19A4  LDA COUNT       ;C COMM OR K/I COMM ?
F99F          D004    BNE R10
F9A1          A4EA    LDY LENGTH
F9A3          D005    BNE R11
F9A5 R10      A4EA    LDY LENGTH
F9A7          D009    BNE R87
F9A9 REP2     60      RTS
F9AA R11
F9AA R8       A90D    LDA #CR
F9AC          91DF    STA (NOWLN),Y
F9AE          D1DF    CMP (NOWLN),Y
F9B0 R97      D00C    BNE R98
F9B2          ;LENGTH = OLDLEN
F9B2 R87      88      DEY
F9B3          C0FF    CPY #$FF
F9B5          F0F2    BEQ REP2
F9B7 R88      B938A4  LDA DIBUFF,Y
F9BA          91DF    STA (NOWLN),Y
F9BC          D1DF    CMP (NOWLN),Y
F9BE R98      D053    BNE R99
F9C0          88      DEY
F9C1          10F4    BPL R88
F9C3          60      RTS
F9C4
F9C4          ;LENGTH ) OLDLEN
F9C4 R100     98      TYA             ;NEW LINE IS LONGER
F9C5          E5E9    SBC OLDLEN
F9C7          A4E9    LDY OLDLEN
F9C9          D002    BNE R101        ;ALREADY HAVE ROOM FOR CR
F9CB          6900    ADC #0          ;ADD ONE TO DIFFERENCE
F9CD R101     48      PHA
F9CE          20C5F8  JSR SETBOT
F9D1          A000    LDY #0
F9D3 R102     B1DF    LDA (NOWLN),Y
F9D5          F009    BEQ R108
F9D7          E6DF    INC NOWLN
F9D9          D0F8    BNE R102
F9DB          E6E0    INC NOWLN+1
F9DD          4CD3F9  JMP R102
F9E0 R108     68      PLA
```

```
F9E1          48      PHA
F9E2          18      CLC
F9E3          65E1    ADC BOTLN        ;ADD DIFFERENCE TO END
F9E5          85E1    STA BOTLN        ;STORE NEW END
F9E7          9002    BCC R103
F9E9          E6E2    INC BOTLN+1
F9EB R103     20F9F8  JSR ATEND
F9EE          900B    BCC R107
F9F0          A5E7    LDA SAVE         ;RESTORE OLD BOTTOM
F9F2          85E1    STA BOTLN
F9F4          A5E8    LDA SAVE+1
F9F6          85E2    STA BOTLN+1
F9F8          4C5CFA  JMP ENDERR       ;RAN PAST BUFFER END
F9FB R107     A5DF    LDA NOWLN        ;SAVE CURRENT END
F9FD          85E7    STA SAVE
F9FF          A5E0    LDA NOWLN+1
FA01          85E8    STA SAVE+1
FA03          68      PLA
FA04          18      CLC
FA05          65DF    ADC NOWLN
FA07          85DF    STA NOWLN
FA09          9002    BCC R104
FA0B          E6E0    INC NOWLN+1
FA0D R104     B1E7    LDA (SAVE),Y
FA0F          91DF    STA (NOWLN),Y
FA11          D1DF    CMP (NOWLN),Y
FA13 R99      D027    BNE GOGO
FA15          A5E7    LDA SAVE
FA17          CD1CA4  CMP ADDR
FA1A          D007    BNE R105
FA1C          A5E8    LDA SAVE+1
FA1E          CD1DA4  CMP ADDR+1       ;BACK WHERE WE STARTED ??
FA21          F013    BEQ R106         ;BRANCH IF DONE
FA23 R105     A5DF    LDA NOWLN
FA25          D002    BNE R105A
FA27          C6E0    DEC NOWLN+1
FA29 R105A    C6DF    DEC NOWLN
FA2B          A5E7    LDA SAVE
FA2D          D002    BNE R105B
FA2F          C6E8    DEC SAVE+1
FA31 R105B    C6E7    DEC SAVE
FA33          4C0DFA  JMP R104
FA36 R106     20D0F8  JSR RESNOW
FA39          4C9CF9  JMP R9
FA3C
FA3C          ;IT DIDN'T WROTE INTO MEMORY
FA3C GOGO     2034F9  JSR SAVNOW
FA3F          4C33EB  JMP MEMERR
FA42
FA42                  .END
 ERRORS= 0000
```

□□

Wolfgang Radeloff, 2080 Pinneberg

User Output Arbiter

Die Ausgabe von Daten an die Peripherie ist im AIM 65 über den Vektor UOUT mit der Initialisierungsroutine WHEREO und der Transportroutine OUTALL organisiert (siehe /1/2/). Dem WHERO geht im allgemeinen ein Unterprogramm voraus, wie 'M' (Memory Dump) oder 'L' wie LIST. Diesen Caller kann man an seiner Rücksprungadresse auf dem Stack erkennen und in Abhängigkeit von ihr die anwenderbestimmte Ausgabe durch Setzen eines zweiten Anwendervektors UOUT2 auffächern. Das analoge Verfahren ist natürlich für die anwenderbestimmte Eingabe möglich, z.B. bei LOAD oder READ usw.. Der Arbiter erspart es, daß man für jede anwenderbestimmte Dienstleistung den Vektor UOUT bzw. UIN umsetzen muß.

Für die Ausgabeverteilung mit dem folgenden Arbiter ist UOUT daher nur einmalig auf den Arbiter zu richten. Wenn man jetzt von WHEREO herkommt (Carry=0), so wird mit den Befehlen ab USERO+2 initialisiert, kommt man von OUTALL, so wird direkt zu JOUT2 verzweigt. Die Dekodierung der Herkunft kann vom Anwender natürlich auf weitere Dienstleistungen ausgedehnt werden.

(Anm. d. Hrsg.: Statt der Zellen FE/FF in der Zeropage kann auch der Label JUMP+1 des REMON als indirekte Sprungadresse benutzt werden.)

/1/ Ein- und Ausgabe am AIM 65, R. Löhr, 65xx MICRO MAG Nr. 14
/2/ AIM 65 Monitor Program Listing, Rockwell 1979
/3/ AIM User Device Arbiter, J. Swank, MICRO, The 6502 Journal No. 47, 4/82

```
0000           0000 UOUT    =$010A
0000           0001 UOUT2   =$FE
0000           0002 MEMD    =$0F00 ;DUMMY ADRESSES !
0000           0003 OBJD    =$0F02
0000           0004 ELIST   =$0F04
0000           0005 ASMF    =$0F06
0000           0006 ;
0000           0007 ;-----------------------------------------------------------
0000           0008 ;'USERO' CHECKS RETURN ADDRESS ON STACK FROM ' JSR WHEREO '
0000           0009 ;         AND SET POINTER UOUT2 TO USERROUTINE
0000           0010 ;         ENTER PARAMETER: UOUT IS SET TO USERO
0000           0011 ;         RETURN VALUE   : ADDRESS USER OUTPUT ROUTINE IN
0000           0012 ;                          UOUT2, C=0 IF INITCALL
0000           0013 ;         REG.AFFECTED   : A, X, STATUS
0000           0014 ;-----------------------------------------------------------
0000           0015 ;
0000           0016         *=UOUT
010A 000E      0017         .WOR USERO
010C           0018         *=$0E00
0E00           0019 ;
0E00 B01C      0020 USERO   BCS JOUT2
0E02 68        0021         PLA
0E03 48        0022         PHA
0E04 A203      0023         LDX #3
0E06 DD210E    0024 UE1     CMP OUTTAB,X
0E09 F005      0025         BEQ UE2
0E0B CA        0026         DEX
0E0C 10F8      0027         BPL UE1
0E0E 300D      0028         BMI UE3   ;UNKNOWN CALL, USER MUST SET 'UOUT2'
0E10           0029 ;
0E10 8A        0030 UE2     TXA
0E11 0A        0031         ASL A
0E12 AA        0032         TAX
0E13 BD250E    0033         LDA OUTDO,X
0E16 85FE      0034         STA UOUT2
0E18 BD260E    0035         LDA OUTDO+1,X
0E1B 85FF      0036         STA UOUT2+1
0E1D 18        0037 UE3     CLC
0E1E 6CFE00    0038 JOUT2   JMP (UOUT2) ;JUMP TO USER OUTPUT ROUTINE
0E21           0039 ;
```

```
0E21 61       0040 OUTTAB .BYT #61 ;<D>     E45F JSR WHEREO   5F +2 =61
0E22 AE       0041        .BYT #AE ;<OBJ>   D0AC JSR WHEREO   AC +2 =AE
0E23 EC       0042        .BYT #EC ;<L>     F7EA JSR WHEREO   EA +2 =EC
0E24 83       0043        .BYT #83 ;<LIST>  D003 JSR WHEREO   03 +2 =83
0E25          0044 ;
0E25 000F     0045 OUTDO  .WOR MEMD ;FORMATTED MEMORY DUMP
0E27 020F     0046        .WOR OBJD ;FORMATTED OBJ. CODE DUMP (ASSEMB)
0E29 040F     0047        .WOR ELIST ;EDITOR LISTING W. PRINTER COMMANDS
0E2B 060F     0048        .WOR ASMF ;ASSEMBLER FORMATTER
0E2D          0049 ;
0E2D          0050        .END USERO

  ERRORS= 0000                                                    □□
```

Dipl.-Ing. Rüdiger Wollenberg, 4600 Dortmund

Assembler-Formatierer

Der Assembler des AIM 65 ist für die nur 20-stelligen Ausgabemöglichkeiten auf dem Display und Printer konzipiert worden. Viele Betreiber haben inzwischen aber ein Videointerface, einen externen Drucker oder ein Terminal an ihren Rechner angeschlossen und müssen das schlechte Ausgabeformat verwenden. In der Vergangenheit sind daher schon mehrfach Assembler-Formatierprogramme vorgestellt worden, die eine gute Lesbarkeit gestatten. Leider treiben diese Programme zumeist nur feste Schnittstellen, etwa IEEE, V24, 20mA-Stromschleife oder ein bestimmtes Videointerface.

Dieser Einschränkung unterliegt man nicht, wenn man die Print-Routine im Assembler-ROM verändert. Verzichtet man auf eine gute Dokumentation über den 20-Zeichen-Drucker und das Display, so erhält man die größtmögliche Flexibilität bei der Ausgabe. Diese Lösung ist besonders attraktiv, wenn ein Benutzer den Dxxx-Bereich als RAM ausgelegt hat oder sich für ca. DM 20,- ein 2532-EPROM selbst programmiert.

Im einzelnen wurden folgende Änderungen durchgeführt:

Unterdrückung der automatischen Displayausgabe während des zweimaligen Zugriffes auf den Quelltext. Durch diese Maßnahme wird die Übersetzungszeit drastisch verkürzt.

Für den .PAG-Befehl wurde das '—' -Symbol in den häufiger gebräuchlichen Stern (*) umgesetzt, der statt 20mal jetzt 60mal ausgegeben wird.

Vor jeder Zeile wird grundsätzlich der aktuelle Programmzähler ausgegeben.

Die Label werden zur Verbesserung der Lesbarkeit mit Blanks aufgefüllt, wenn sie weniger als 6 Zeichen besitzen. In Zeilen ohne Label werden grundsätzlich 6 Blanks ausgegeben.

Bei Ausdruck des Operationscodes werden mehrfache Blanks reduziert.

Kommentare werden im Listing auf die 36. Spalte ausgerichtet.

Die Wirkungsweise zeigt das über den veränderten Assembler erstellte nachfolgende Listing. Ist für den Dxxx-Bereich ein RAM vorhanden, müssen die drei NOPs ab hex D107 unbedingt manuell gesetzt werden, weil die Assemblierung des Quelltextes den Übersetzungsvorgang stört. Der übrige Code kann von dem Assembler übersetzt werden, es sollte allerdings die List-Option nicht benutzt werden, um auch hier ein 'Aufhängen' des Prozessors zu vermeiden.

```
**********************************************************
ASSEMBLER - FORMATTER   V1.3  11.3.1983   WOLLENBERG

0000
0000 J              =$19
0000 JLABL          =$20
0000 IPC            =$32
0000 IMAXCL         =$35
```

```
0000 ICRD          =$46
0000 ABLK          =$DFFA
0000 OUTALL        =$E9BC
0000 WRITAZ        =$EA42
0000
0000
0000               *=$D107          ;UNTERDRÜCKE AUSGABE WÄHREND DES EINLESENS
D107        EA     NOP
D108        EA     NOP
D109        EA     NOP
D10A
D10A               *=$D6A7
D6A7        A03B   LDY #59          ;60 * '*'
D6A9        A92A   LDA #'*'         ;*-REIHE BEI PAGE-DIRECTIVE
D6AB
D6AB               *=$DA7E
DA7E        A533   LDA IPC+1        ;LADE AKTUELLEN PC
DA80        A632   LDX IPC
DA82        2042EA JSR WRITAZ       ;ANZEIGE
DA85        A200   LDX #0
DA87        A520   LDA JLABL        ;LABEL IN DER ZEILE?
DA89        F025   BEQ Q32          ;NEIN
DA8B        C8     INY              ;JA, SCHREIBE EIN BLAN
DA8C        20FADF JSR ABLK
DA8F Q34    B546   LDA ICRD,X       ;UNTERDRÜCKE FÜHRENDE BLANKS
DA91        E8     INX
DA92        C920   CMP #' '
DA94        D006   BNE Q33
DA96        E435   CPX IMAXCL       ;ALLES ABGEARBEITET?
DA98        F027   BEQ Q35          ;JA
DA9A        D0F3   BNE Q34          ;NEIN
DA9C Q33    C8     INY              ;SCHREIBE LABEL BUCHSTABENWEISE
DA9D        20BCE9 JSR OUTALL
DAA0        E435   CPX IMAXCL       ;ALLES ABGEARBEITET?
DAA2        F00C   BEQ Q32          ;JA
DAA4        B546   LDA ICRD,X       ;NÄCHSTES ZEICHEN
DAA6        E8     INX
DAA7        C920   CMP #' '         ;BLANK?
DAA9        F005   BEQ Q32          ;D.H. LABEL ZU ENDE
DAAB        C93D   CMP #'='         ;'='?
DAAD        D0ED   BNE Q33          ;D.H. LABEL ZU ENDE
DAAF        CA     DEX              ;EIN ZEICHEN ZU VIEL
DAB0 Q32    20FADF JSR ABLK         ;FÜLLE RAUM AUF, FÜR GLEICH VIELE ZEICHEN
DAB3        C8     INY
DAB4        C008   CPY #8           ;ACHT ZEICHEN GEDRUCKT?
DAB6        D0F8   BNE Q32
DAB8        EA     NOP
DAB9        EA     NOP
DABA        EA     NOP
DABB        EA     NOP
DABC        EA     NOP
DABD        EA     NOP
DABE        EA     NOP
DABF        EA     NOP
DAC0        EA     NOP
DAC1 Q35    8619   STX J            ;MERKE ZAHL DER ABGEARBEITETEN ZEICHEN
DAC3
DAC3               *=$DB2E
```

```
DB2E          A000    LDY #0
DB30          8420    STY JLABL       ;LÖSCHE LABEL
DB32          A619    LDX J           ;ALTER ZEIGER AUF QUELLTEXT-ZEILE
DB34          E435    CPX IMAXCL      ;ALLES ABGEARBEITET?
DB36          B07A    BCS Q80         ;JA
DB38 Q73      B546    LDA ICRD,X      ;NEIN, LIES ZEICHEN
DB3A          C920    CMP #' '        ;UNTERDRÜCKE BLANKS
DB3C          D007    BNE Q72
DB3E          E8      INX
DB3F          E435    CPX IMAXCL      ;ALLES ABGEARBEITET?
DB41          B06F    BCS Q80         ;JA
DB43          90F3    BCC Q73         ;NEIN, WEITERE BLANKS?
DB45 Q72      B546    LDA ICRD,X      ;VON BLANK VERSCHIEDENES ZEICHEN
DB47          C93B    CMP #';'        ;';'?
DB49          F031    BEQ Q20
DB4B          C927    CMP #$27        ;FIRST '''?
DB4D          F041    BEQ Q22
DB4F          C920    CMP #' '        ;BLANK?
DB51          F01E    BEQ Q70
DB53          C93D    CMP #'='        ;'='?
DB55          F004    BEQ Q77         ;JA, SCHREIBE DAS '='
DB57          C92A    CMP #'*'        ;'*'?
DB59          D007    BNE Q76         ;NEIN
DB5B Q77      20BCE9  JSR OUTALL      ;JA, SCHREIBE DAS '=' BZW. '*'
DB5E          E619    INC J           ;MERKER, UM FOLGENDE BLANKS ZU LOESCHEN
DB60          D007    BNE Q74
DB62 Q76      20BCE9  JSR OUTALL      ;NEIN, SCHREIBE DAS ZEICHEN
DB65          A900    LDA #0          ;LOESCHE BLANK-MERKER
DB67          8519    STA J
DB69 Q74      C8      INY             ;ERHÖHE TV-ZEIGER
DB6A Q75      E8      INX             ;ERHÖHE QUELLTEXT-ZEIGER
DB6B          E435    CPX IMAXCL      ;ALLES ABGEARBEITET?
DB6D          B043    BCS Q80         ;JA
DB6F          90D4    BCC Q72         ;NEIN
DB71 Q70      A519    LDA J           ;UNTERDRÜCKE MEHRFACHE BLANKS
DB73          D0F5    BNE Q75
DB75          20FADF  JSR ABLK        ;SCHREIBE BLANK
DB78          E619    INC J           ;SETZE BLANK-MERKER
DB7A          D0ED    BNE Q74
DB7C Q20      20FADF  JSR ABLK        ;POSITIONIERE AUF GLEICHE KOMMENTARZEILE
DB7         C8      INY
DB80          C010    CPY #16
DB82          90F8    BCC Q20
DB84 Q21      B546    LDA ICRD,X      ;SCHREIBE KOMMENTAR
DB86          20BCE9  JSR OUTALL
DB89          E8      INX
DB8A          E435    CPX IMAXCL      ;ALLES ABGEARBEITET?
DB8C          B024    BCS Q80         ;JA
DB8E          90F4    BCC Q21         ;NEIN
DB90 Q22      B546    LDA ICRD,X      ;SCHREIBE STRING IN QUOTES
DB92          20BCE9  JSR OUTALL
DB95          C8      INY
DB96          E8      INX
DB97          E435    CPX IMAXCL      ;ALLES ABGEAREITET?
DB99          B017    BCS Q80         ;JA
DB9B          B546    LDA ICRD,X      ;NEIN, NÄCHSTES ZEICHEN DES STRINGS
DB9D          C927    CMP #$27        ;ZWEITES '''?
```

```
DB9F        D0EF    BNE Q22          ;NEIN, ALSO WEITER
DBA1        F0BF    BEQ Q76          ;JA, KOMMT KOMMENTAR?
DBA3
DBA3                *=$DBB2
DBB2 Q80
DBB2                .END
 ERRORS= 0000
```

□□

Heimcomputer

Besprechung des LASER 110, des LASER 210 und des Aquarius

Im Frühjahr 1983 waren weitere z.T. drastische Preisermäßigungen bei Personal- und Heimcom putern festzustellen. Die letztgenannte Gruppe ist etwa wie folgt zu klassifizieren: Die Bedieneinheit (Konsole) mit Tastatur und Mikrocomputer liegt in der Preisklasse von ca. DM 150 (z.B. der ZX81) bis ca. DM 1500. Die Standard-Ausgabeeinheit ist ein vom Anwender vorzuhaltender Farb- oder S/W-Fernseher, den man eigentlich in den Preis einrechnen müßte.

Als Peripherie für den Heimgebrauch ist daneben mindestens ein Cassetten-Recorder für die Massenspeicherung (vor allem von Programmen) empfehlenswert. Auf manche Heimcomputer kann man Programmcassetten mit Festwertspeichern stecken, die Bildschirmspiele oder ernsthafte Programme enthalten.

Ein sehr erfolgreicher Heimcomputer ist der farbtüchtige VC-20 von Commodore, der von Anfang an mit deutschem Handbuch herauskam und der jetzt nur noch etwa DM 440,- kostet, bei wohl weiter fallender Preistendenz. Er wurde in Heft 23 dieser Zeitschrift besprochen, zusammen mit dem TRS-80 Color-Computer, letzterer liegt jetzt auch deutlich unter DM 1000. Bei diesem Preis bewegt sich auch der sehr verbreitete VC-64 von Commodore. - Die hier zu besprechenden Geräte fallen mehr in das untere Ende, in die Klasse des VC-20. Typisch ist hier, daß der Arbeitsspeicher des Grundgerätes nur wenige KB RAM enthält und daß sich die Stromversorgung außerhalb der Bedieneinheit befindet, und zwar als Modul in der Netzschnur.

Der LASER 110

Dieses Gerät stammt aus Hong Kong (Firma Video Technology Ltd.) und wird ab Juli 1983 von der Sanyo Video Vetrieb GmbH & Co. serienmäßig ausgeliefert. Er erzeugt ein schwarz/weißes Bild über Monitor- oder TV-Ausgang und liegt mit einem Endverbraucherpreis von DM 298,- an der unteren Preisgrenze. Der LASER 110 hat 4 KB RAM und meldet sich in Microsoft-BASIC (16 KB Betriebssystem und Sprache). Die Prozessorbusse sind auf eine Kontaktleiste an der Hinterseite herausgeführt, so daß man Ergänzungsmodule aufstecken kann (16 KB RAM DM 149,-, 64 KB DM 298,-, diese Preise gelten auch für den LASER 210). Eine weitere Kontaktleiste kann ein Printer-Interface aufnehmen, ferner gibt es eine Buchse für Cassetten-Recorderanschluß. Das Grundgerät wiegt 832 g und hat die Maße 290x165x48 mm, es ist damit leicht transportabel und kann in eine günstige Position zur Bildschirmausgabe gerückt werden. Das Monitor- oder Fernsehbild steht ruhig und klar und stellt 16 Zeilen zu 32 Zeichen dar.

Wohl aus Preisgründen hat dieses Gerät - wie auch andere - eine Tastatur mit Silikon-Kautschuk-Tasten, deren Betätigung anfangs etwas ungewohnt (weil wabbelig) ist. Die Tasten werden jedoch prellfrei und sicher ausgelöst, Tastendrücke werden mit einem Piepser bestätigt. Man muß sich allerdings fragen, wie lange wohl der Tastaturaufdruck (Siebdruck) nach Beanspruchung lesbar bleibt.

Durch Betätigen der CTRL-Taste zusammen mit einer anderen oder mit RETURN als Vortaste erhält man komplette BASIC-Statements auf dem Bildschirm. Insgesamt lassen sich so 67 Statements erzeugen, eine bemerkenswerte Bequemlichkeit für den Programmierer. Wir finden ferner 4 Kontrolltasten für die Cursorbewegungen sowie INSERT, RUBOUT und INVERSE und auch 16 Grafikzeichen. Wie bei Commodore kann man beim Bildschim-Editieren in angezeigte Bild-

zeilen hineinfahren und korrigieren: What you see is what you get. Für graphische Anwendungen stehen die Befehle SET, RESET und POINT zur Verfügung.

Eine deutliche Beschränkung muß darin gesehen werden, daß das BASIC nur 6 signifikante Ziffern ausgibt: Beträge über DM 9999,99 werden damit bereits gerundet. Bemerkenswert ist immerhin das Vorhandensein des PRINTUSING-Befehles. Mit dem LASER 110 kann man über den Befehl SOUND auch Töne ausgeben, und zwar 31 Frequenzen mit 9 verschiedenen Tondauern (notenlängen).

Das hier vorliegende Handbuch, das alle Hantierungen und Befehle erklärt, ist noch in Englisch geschrieben (149 Seiten). Eine deutsche Fassung soll mit Lieferbeginn zur Verfügung stehen. Als Software sind verschiedene Spiele und einfachere Anwenderprogramme in Vorbereitung (auf Tonbandcassette). Der LASER 110 darf damit in der Summe als preiswertes Einsteigersystem bezeichnet werden, mit dem man BASIC lernen, seine Schulaufgaben erledigen und verschiedene nützliche Anwendungen fahren kann.

Der LASER 210

Dieses Gerät entspricht weitgehend dem zuvor besprochenen LASER 110, hat wiederum 45 Tasten, jedoch 8 KB RAM in der Grundausstattung und kostet DM 388,-. Dieser LASER ist farbtüchtig in dem Sinne, daß man mit dem Befehl COLOR I,J die Farbe des Hintergrundes und die des Bildfensters verändern kann. Ansonsten sind keine Unterschiede erkennbar, auch nicht in der Rechengenauigkeit der hier geprüften Geräte, so daß für den Käufer eigentlich nur die Frage auftritt, ob er mehr Speicher in der Grundausstattung haben möchte und ob ihm Farbe auf dem Bildschirm wichtig ist.

Mattel's Aquarius

Im Herbst (Aug./Sept.) wird auch hierzulande ein weiteres Heimcomputersystem mit o.a. Namen ausgeliefert, das von der Mattel Electronics GmbH in Hamburg 76 vertrieben wird. In den USA wird dafür bereits in doppeltseitigen Farbanzeigen Werbung gemacht.Wie erinnerlich, war Mattel früher mit der Puppe Barbie und danach mit Videospielen sehr erfolgreich.

Das Design des Aquarius entspricht in etwa den vorerwähnten LASER-Computern: Konsole mit 48 Tasten aus Kautschuk + Resettaste, 34 BASIC-Statements können durch CTRL+Taste voll ausgeschrieben programmiert werden. Das Microsoft-BASIC (wohl einschl. Betriebssystem) residiert in Festwertspeichern von 8 KB. In der Grundausstattung hat man 4 KB RAM (Arbeitsspeicher), es gibt Erweiterungsmodule um 4 KB und 16 KB RAM. Der Computer ist farbtüchtig und erlaubt Farbansprache auch für Einzelzeichen. Das Bildschirmformat (TV-Ausgang) ist 24 Zeilen mit jeweils 38 Zeichen. Das Fernsehbild steht ruhig.

Der Aquarius ('Wassermann') akzeptiert Groß- und Kleinschreibung, natürlich (?) ohne deutsche Sonderzeichen (Japaner, wie z.B. Epson, haben von Anfang an diese Sonderzeichen!). Auch hier hat man magere 6 signifikante Ziffern im Rechenergebnis, womit der Computer in die untere Leistungsklasse fällt. Auch ist zu bemerken, daß der Tastendruck nicht immer angenommen und auch nicht durch ein akustisches Signal quittiert wird. Das verunsichert die Eingaben (es wurde hier eines der drei Voraus-Systeme getestet). Rechts neben der Tastatur befindet sich ein Schacht mit Kontaktleiste, auf die man Erweiterungsmodule aufschieben kann, 'paddles', ROM-Cassetten mit Spielen, Speichererweiterungen. Ein Thermodrucker mit 40 Zeichen in der Zeile ist seriell anzuschließen ebenso ein Daten-Rekorder. Ein Telefon-Modem und ein doppeltes Floppy-Laufwerke sind für 1984 geplant. Auch dieses System trägt auf der Rückseite das Schild 'Made in Hong Kong'. - Von Anfang an soll mit deutscher Dokumentation geliefert werden. Bei der hier geprüften englischen gefiel ein Aufsteller mit 'simplified instruction cards' und ein ausführlicher 'Guide to Home Computing'.

Nach Angaben der Firma Mattel ist mit folgenden Preisen zu rechnen: Grundgerät unter DM 400,-, Thermodrucker ebenfalls unter DM 400,-, Mini-Expander etwa DM 250,-, Datenrecorder unter DM 200,-.

Peter Engels, 5308 Rheinbach

Big Letters von CBM auf EPSON

Dieses Programm erlaubt die Ausgabe der ersten 7 Schirmzeilen eines CBM-Rechners auf einen Drucker mit Einzelnadelansteuerung (Epson Typ 2/3, Epson Typ RX/FX) in 8-fach vergrößerter Schrift. Es belegt mit Character-ROM den RAM-Bereich $77E0-8000 im CBM und wird mit SYS 30432 aufgerufen.

Beim ersten Aufruf schützt sich das Programm zuerst selbst vor BASIC (Zeilen 440-540). Dann ab dem 2. Aufruf erfolgt der Ausdruck der ersten 7 Zeilen. Dazu wird der Bit-Image-Mode der o.g. Drucker verwendet. Eine Änderung auf andere Drucker ist ohne weiteres durch Ändern der Steuerzeichen möglich.

Das Programm übergibt die Zeichen an den IEC-Bus wie in /1/ beschrieben. Eine weitere Erklärung braucht daher hier nicht zu erfolgen. Damit der Rechner weiß, wie die Zeichen für den Drucker gebildet werden, liegt im Bereich 7800-7FFF eine Kopie des jeweiligen Character-ROMs. Damit das Treiberprogramm von $76E0-77F0 kurzbleibt, wird das Character-ROM nicht direkt übernommen, sondern die Zeichen werden gedreht. Eine genaue Erklärung erfolgte bereits in Heft 30 des 65xxx MICRO MAG.

Literatur:

/1/ Kirchgäßner, R.: CBM Standardroutinen für Daten-I/O, MC 10/82

/2/ Engels, Peter: High Resolution Screen Dump, 65xx MICRO MAG, April 1983

Ein Probeausdruck (verkleinert) durch das Programm

65xx MICRO MAG

Computing Software Hobby

(c) by Roland Loehr

```
0010 ; '  BIG LETTERS CBM  '
0020 ;    ' FUER CBM-3032 '
0030 ; ' & EPSON TYP 2 + 3 '
0040
0050 ;    AUFRUF: SYS 30432
0060
0070 ZEIGER      .DE $21
0080 ZWSP1       .DE $BA              ; ZEICHEN-ZAEHLER
0090 ZWSP2       .DE $BC              ; BYTE-ZAEHLER
0100 ZWSP3       .DE $BD              ; BIT-ZAEHLER
0110 ZWSP4       .DE $BE              ; ROM-BYTE-BUFFER
0120 ZWSP5       .DE $BF              ; BIT-FLAG
0130 ZWSP6       .DE $C0              ; RVS FLAG
0140 ZWSP7       .DE $C1              ; DURCHLAUF-FLAG
0150 STRADR      .DE $1F
0160 FNLEN       .DE $D1
0170 LA          .DE $D2
0180 SA          .DE $D3
0190 FA          .DE $D4
0200 CHRROM      .DE $7800
0210 PCR         .DE $E84C
0220 OPENFILE    .DE $F524            ; ( $F563 )
0230 CLRCH       .DE $FFCC
0240 CLOSEFILE   .DE $F2AC            ; ( $F2E0 )
0250 CRLFBAS     .DE $C9E2            ; ( $BADF )
0260 BSOUT       .DE $FFD2
0270 CKOUT       .DE $FFC9
0280 NEW         .DE $C55D            ; ( $B5D4 )
0290 WRTWO       .DE $E784            ; ( $D731 )
```

```
                0300 SPAC2      .DE $FDCA               ; ($D52E)
                0310 STOPEQ     .DE $F301               ; ($F335)
                0320 ZEIL1      .DE $8000
                0330 ZEIL2      .DE $8000+40
                0340 ZEIL3      .DE $8000+80
                0350 ZEIL4      .DE $8000+120
                0360 ZEIL5      .DE $8000+160
                0370 ZEIL6      .DE $8000+200
                0380 ZEIL7      .DE $8000+240
                0390
                0400            .BA $76E0
                0410            .MC $76E0
                0420            .OS
                0430
76E0- A9 76     0440 SAVE       LDA #H,SAVE             ; HIMEM AUF
76E2- A2 E0     0450            LDX #L,SAVE             ; $SAVE SETZEN
76E4- 86 34     0460            STX *$34
76E6- 85 35     0470            STA *$35
76E8- A9 4C     0480            LDA #$4C                ; NACH 1. AUFRUF
76EA- A0 FA     0490            LDY #L,START            ; JMP START
76EC- A2 76     0500            LDX #H,START            ; NACH SAVE
76EE- 8D E0 76  0510            STA SAVE                ; BRINGEN, DA SAVE
76F1- 8C E1 76  0520            STY SAVE+1              ; NICHT MEHR NOETIG
76F4- 8E E2 76  0530            STX SAVE+2
76F7- 4C 5D C5  0540            JMP NEW
76FA- A9 7F     0550 START      LDA #127                ; OPEN 127,4,4
76FC- 85 D2     0560            STA *LA
76FE- A9 04     0570            LDA #4
7700- 85 D4     0580            STA *FA
7702- 85 D3     0590            STA *SA
7704- A9 00     0600            LDA #0                  ; FILENAME=""
7706- 85 D1     0610            STA *FNLEN
7708- 20 24 F5  0620            JSR OPENFILE
770B- A2 7F     0630            LDX #127                ; CMD 127
770D- 20 C9 FF  0640            JSR CKOUT
7710- A2 1B     0650            LDX #$1B                ; ESC "A"+8
7712- A9 41     0660            LDA #$41                ; LF = 8/72 INCH
7714- 20 B4 E7  0670            JSR WRTWO
7717- A9 08     0680            LDA #$08
7719- 20 D2 FF  0690            JSR BSOUT
771C- 20 E2 C9  0700            JSR CRLFBAS             ; CRLF
771F- A9 00     0710            LDA #$00
7721- 85 BA     0720            STA *ZWSP1
7723- A9 00     0730 READ1      LDA #$00
7725- 85 BC     0740            STA *ZWSP2
7727- A6 BA     0750 READ       LDX *ZWSP1
7729- A9 00     0760            LDA #$00
772B- 85 C1     0770            STA *ZWSP7
772D- BD F0 80  0780            LDA ZEIL7,X
7730- 20 8B 77  0790            JSR SEARCHOUT
7733- A9 FF     0800            LDA #$FF
7735- 85 C1     0810            STA *ZWSP7
7737- A6 BA     0820            LDX *ZWSP1
7739- BD C8 80  0830            LDA ZEIL6,X
773C- 20 8B 77  0840            JSR SEARCHOUT
773F- A6 BA     0850            LDX *ZWSP1
7741- BD A0 80  0860            LDA ZEIL5,X
7744- 20 8B 77  0870            JSR SEARCHOUT
7747- A6 BA     0880            LDX *ZWSP1
7749- BD 78 80  0890            LDA ZEIL4,X
774C- 20 8B 77  0900            JSR SEARCHOUT
774F- A6 BA     0910            LDX *ZWSP1
7751- BD 50 80  0920            LDA ZEIL3,X
7754- 20 8B 77  0930            JSR SEARCHOUT
7757- A6 BA     0940            LDX *ZWSP1
7759- BD 28 80  0950            LDA ZEIL2,X
775C- 20 8B 77  0960            JSR SEARCHOUT
775F- A6 BA     0970            LDX *ZWSP1
7761- BD 00 80  0980            LDA ZEIL1,X
7764- 20 8B 77  0990            JSR SEARCHOUT
7767- 20 E2 C9  1000            JSR CRLFBAS             ; CRLF
```

```
776A- E6 BC     1010            INC *ZWSP2
776C- A5 BC     1020            LDA *ZWSP2
776E- C9 08     1030            CMP #8
7770- D0 B5     1040            BNE READ
7772- E6 BA     1050            INC *ZWSP1          ; ZEILEN FERTIG ?
7774- A5 BA     1060            LDA *ZWSP1
7776- C9 28     1070            CMP #40
7778- D0 A9     1080            BNE READ1
777A- A2 1B     1090 EXIT       LDX #$1B            ; LF=1/6 INCH
777C- A9 32     1100            LDA #$32
777E- 20 84 E7  1110            JSR WRTWO
7781- 20 CC FF  1120            JSR CLRCH           ; STANDART I/O
7784- A0 7F     1130            LDY #127            ; CLOSE 127
7786- 84 D2     1140            STY *LA
7788- 4C AC F2  1150            JMP CLOSEFILE , -> BASIC
                1160 ;
778B- 85 BE     1170 SEARCHOUT  STA *ZWSP4
778D- A2 00     1180            LDX #$00
778F- A5 BE     1190            LDA *ZWSP4
7791- 10 02     1200            BPL NORVS
7793- A2 FF     1210            LDX #$FF
7795- 86 C0     1220 NORVS      STX *ZWSP6
7797- 29 7F     1230            AND #$7F
7799- 85 1F     1240            STA *STRADR
779B- A9 00     1250            LDA #$00
779D- 85 20     1260 BYTE       STA *STRADR+1
779F- 06 1F     1270            ASL *STRADR
77A1- 26 20     1280            ROL *STRADR+1
77A3- 06 1F     1290            ASL *STRADR
77A5- 26 20     1300            ROL *STRADR+1
77A7- 06 1F     1310            ASL *STRADR
77A9- 26 20     1320            ROL *STRADR+1
77AB- A5 20     1330            LDA *STRADR+1
77AD- 09 7B     1340            ORA #H,CHRROM
77AF- 85 20     1350            STA *STRADR+1
77B1- AD 4C E8  1360            LDA PCR
77B4- 29 02     1370            AND #$02
77B6- F0 07     1380            BEQ GRAFIC
77B8- A5 20     1390            LDA *STRADR+1
77BA- 18        1400            CLC
77BB- 69 04     1410            ADC #$04
77BD- 85 20     1420            STA *STRADR+1
                1430 ;
77BF- A4 BC     1440 GRAFIC     LDY *ZWSP2          ; BYTEZAEHLER
77C1- B1 1F     1450            LDA (STRADR),Y
77C3- 45 C0     1460            EOR *ZWSP6          ; EOR MIT RVS-FLAG
77C5- 85 BE     1470            STA *ZWSP4          ; BYTE SPEICHERN
                1480 ;
77C7- A5 C1     1490            LDA *ZWSP7
77C9- D0 0E     1500            BNE NOGRA
77CB- A2 1B     1510            LDX #$1B            ; ESC "K"+64+0
77CD- A9 4B     1520            LDA #$4B            ; =GRAFIC 8*8*7 ZEICHEN
77CF- 20 84 E7  1530            JSR WRTWO
77D2- A2 C0     1540            LDX #192
77D4- A9 01     1550            LDA #$01
77D6- 20 84 E7  1560            JSR WRTWO
                1570 ;
77D9- A9 07     1580 NOGRA      LDA #$07
77DB- 85 BD     1590            STA *ZWSP3
77DD- A5 BE     1600 BITLOOP    LDA *ZWSP4
77DF- 4A        1610            LSR A
77E0- 85 BE     1620            STA *ZWSP4
77E2- B0 03     1630            BCS OUT1            ; TEST OB BIT = 1/0
77E4- A9 00     1640 OUT0       LDA #00             ; BIT = 0
77E6- 2C        1650            .BY $2C
77E7- A9 FF     1660 OUT1       LDA #$FF            ; BIT = 1
77E9- A2 07     1670            LDX #$07
77EB- 20 D2 FF  1680 OUTLOOP    JSR BSOUT
77EE- CA        1690            DEX
77EF- 10 FA     1700            BPL OUTLOOP
77F1- C6 BD     1710            DEC *ZWSP3
```

```
77F3- 10 E8     1720            BPL BITLOOP
                1730 ;
77F5- 20 01 F3  1740            JSR STOPEQ          ; STOPTASTE ?
77F8- F0 01     1750            BEQ STOP
77FA- 60        1760            RTS
77FB- 68        1770 STOP       PLA
77FC- 68        1780            PLA
77FD- 4C 7A 77  1790            JMP EXIT
                1800 ;
                1810 ; AB $7800 FOLGT GEDREHTES
                1820 ; CBM-CHARACTER-ROM
                1830 ; ABGELEGT  MITTELS 'ROM-DREH'
                1840 ; WIE IN MICRO-MAG NR.30
                1850 ; BESCHRIEBEN
                1860 ;
                1870            .EN
ENDE VON MAE PASS !

BITLOOP =77DD           BSOUT =FFD2             BYTE =779D
CHRROM =7800            CKOUT =FFC9             CLOSEFILE =F2AC
CLRCH =FFCC             CRLFBAS =C9E2           EXIT =777A
FA =00D4                FNLEN =00D1             GRAFIC =77BF
LA =00D2                NEW =C55D               NOGRA =77D9
NORVS =7795             OPENFILE =F524          OUT0 =77E4
OUT1 =77E7              OUTLOOP =77EB           PCR =E84C
READ =7727              READ1 =7723             SA =00D3
SAVE =76E0              SEARCHOUT =778B         SPAC2 =FDCA
START =76FA             STOP =77FB              STOPEQ =F301
STRADR =001F            WRTWO =E784             ZEIGER =0021
ZEIL1 =8000             ZEIL2 =8028             ZEIL3 =8050
ZEIL4 =8078             ZEIL5 =80A0             ZEIL6 =80C8
ZEIL7 =80F0             ZWSP1 =00BA             ZWSP2 =00BC
ZWSP3 =00BD             ZWSP4 =00BE             ZWSP5 =00BF
ZWSP6 =00C0             ZWSP7 =00C1
//0000,7800,7800
```

□□

FORTH-Splitter (3)

Wie auch andere strukturierte Sprachen setzt FORTH voraus, daß Befehle/Funktionen/Variable bereits vor ihrer Benutzung definiert worden sind. Ein Vorgriff auf im Quellprogramm weiter unten stehende Definitionen ist nicht möglich. - Herr Wolfgang Zweygart aus Böblingen schlägt hier nun ein Verfahren vor, das er 'Rückwärtsfädeln' nennt. Es wurde für ein Programm zum Plotten entwickelt, bei dem er nicht für jede neue Funktion eine komplette Kompilierung in Kauf nehmen wollte. Der Kunstgriff liegt darin, daß er, hinten im Dictionary stehend, die Codefeldadresse der neuen Funktion an die bei FUNKT reservierten Speicherstellen überträgt. Hier als Beispiel die Lösung:

```
: FUNKT TASK TASK ; ( AN DIESER STELLE SOLL DIE FORMEL )
( WIRKSAM SEIN )
  .
  .  ( HIER LIEGT DAS HAUPTPROGRAMM )
  .
' FUNKT CONSTANT FUNKT.ADR
: NEU CR
." Gib Funktion als 'Colon Definition' ein : " CR
." ?  " QUERY INTERPRET
( FUNKTION WIRD ALS LEZTES FORTHWORT EINGETRAGEN )
LATEST PFA CFA FUNKT.ADR !    ( CODE-FIELD-ADR WIRD UNTER )
( 'FUNKT' EINGETRAGEN )
;
```

Editorial

In den vergangenen Monaten war ein deutlicher Rückgang der Preise für persönliche und Heimcomputer zu verzeichnen. Die gleiche verbraucherfreundliche Entwicklung ist bei Peripheriegeräten festzustellen. Dieses Nachlassen der Preise ist im Zusammenhang mit der zunehmenden Liefermenge, mit verschärftem Wettbewerb und mit der Verbilligung großer Speicher zu sehen. Diese drei Einflüsse werden weiterwirken und das Angebot noch vielfältiger und interessanter machen. Man darf also auch künftig mit einem günstiger werdenden Preis-/Leistungsverhältnis rechnen.

Computerleistung ist bald für jeden erschwinglich, sowohl für den Oberschüler, der seine Aufgaben lösen und der spielen will, wie auch für die vielen Betriebe, die bisher noch keinen Computer benutzen. Man soll den Computer dabei nicht nur als rechnendes, textverarbeitendes und Verwaltungsarbeiten erledigendes Gerät sehen, sondern künftig mehr auch als ein Gerät zur Nachrichtenübermittlung, sei es nun innerbetrieblich oder in Wahlverbindung über die öffentlichen Netze. 1984 soll das BTX-System (Bildschirmtext) gestartet werden. Die dafür vorgesehenen Endgeräte, Terminals mit Tastatur und Bildschirm, werden in der vorgesehenen Form sicher nicht das Endprodukt sein, denn es liegt nahe, diesen notwendigen Komponenten auch noch ein Rechnerchassis beizufügen.

Die Konfiguration eines Rechners stellt von der Hardware her heute im allgemeinen kaum noch Probleme, es sei denn, es geht um besondere Interfaces. Deutlicher als früher aber hat gute Software einen größeren Wert und Preis im Verhältnis zum Anschaffungswert des Rechners. Zu erwähnen sind hier nicht nur gute Systemprogramme, Assembler, Sprachen, Wordprozessoren usw., sondern ebenso auch Anwenderprogramme, sei es nun für kaufmännische und verwaltende Aufgaben, Datenbanken, Textsysteme oder für das Messen, Steuern und Auswerten. Gute Programme fordern viel Fleiß, um die gewünschte Leistung zu erbringen und um sie für die Anwendung betriebssicher zu machen. Viel Zeit muß häufig auch aufgebracht werden, um ansonsten geeignete Systemprogramme auf die vom Anwender gewünschte Ein- und Ausgabeart anzupassen. Hier ist der Umstand beklagenswert, daß die Ein- und Aussprungspunkte selten bekannt sind.

Es kommt also weiterhin auf gute Software und auf die Analyse solcher Anpassungsmöglichkeiten an. Die Leser sind gebeten, hierzu nach Möglichkeit weiter beizutragen. Im Gegensatz zu anderen Zeitschriften druckt das 65xx MICRO MAG auch größere kommentierte Programmlistings ab, weil es ja fast immer nicht nur auf eine bestimmte Dienstleistung ankommt, sondern auf eine verstehbare Vorlage für ähnliche Arbeiten und für das Anpassen.

Bücher

Software-Auswahl leicht gemacht. Verlag Markt & Technik, Haar 1983, 2. völlig überarbeitete und aktualisierte Ausgabe, 423 S., kart., ISBN 3-922120-44-4, DM 58,--. Das Buch enthält mehr als 2000 Programmbeschreibungen aus allen Anwendungsbereichen für Personal Computer. Es handelt sich um in Deutschland angebotene, also käufliche Programme, die für berufliche oder betriebliche Anwendungen bestimmt sind. Sie werden in folgende Hauptgebiete gegliedert: Branchenneutrale Programme für Rechnungswesen und Verwaltung, Branchenpakete für Rechnungswesen und Verwaltung, Programme für Technik und Wissenschaft, Systemsoftware. Jeder Beschreibung sind Angaben über die Hardware, den Preis, die Zahlungsweise, den Autor und die Bezugsquelle zugefügt. Am Schluß des Buches sind Kreuzverweisungen: Programme nach Namen alphabetisch, Hardware- und Betriebssystemregister sowie geordnetes Anbieterverzeichnis. - Dieses Buch sollte eigentlich von jedem Softwareentwickler und Computeranwender zu Rate gezogen werden, denn es kann mühselige Eigenentwicklungen sparen, wo es schon Lösungen gibt. Andererseits ist man heute schon in der Lage, einen Computer unter dem Gesichtspunkt auszuwählen, daß für ihn die benötigte Software existiert.

Rompel, Helmut: Programmieren in Assembler und Maschinensprache mit dem Microprozessor 6502. MCA-Edition der Ges. für Microcomputeranwendungen mbH, Buchstr. 4 in 7120 Bietig-

heim-Bissingen, 1982, 201 S., DM 58,--. Wir finden folgende Gliederung: Grundlagen (Zahlensysteme, Codes, logische Verknüpfungen), Hardware der 6502, Befehlsstruktur, Adressierungsarten im Überblick, Instruktionen des 6502, der Monitor, Maschinensprache und BASIC. Der zweite Teil enthält: Neue BASIC-Befehle, E/A, Tastaturanschluß, IEEE-Bus, Interrupt und Parameterübergabe im Verkehr mit BASIC. Das Buch ist (wenn man einmal von den vier ersten allgemeinen Abschnitten absieht) sehr auf die Assemblerprogrammierung von CBM-Rechnern ausgerichtet, für die viele ausgelistete Beispiele einschließlich der sinnvollen Verwendung von Systemroutinen enthalten sind. Es ist damit sehr geeignet, den interessierten CBM-Anwender in den Gebrauch der Maschinensprache einzuführen.

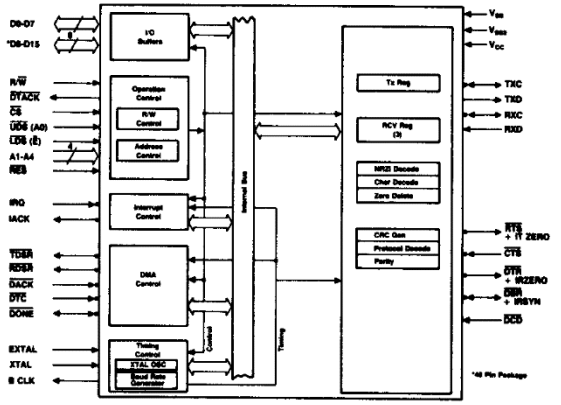
Schumny, H. (Hrsg.): BASIC-Anwenderprogramme. Braunschweig, Vieweg 1983, 89 S. ISBN 3-528 04218 4, kart., DM 19,80. Es handelt sich um den Band 4 der Vieweg Programmbibliothek für Microcomputer. Zusammen mit graphischen, tabellarischen oder textlichen Ablaufbeschreibungen finden wir Anwenderprogramme für Schnittstellen (HP-83/85), Zahnprofile, Solaranlagen, Beleuchtung, Netztransformatoren, Ampel, Rohrnetz, Dauerkalender und Stundenplangestaltung. Band 5 in dieser Reihe trägt den Titel: **BASIC-Programme für den PC 1211/1212** (75 S. ISBN 3-528 04239 7, DM 19,80). Wir finden Programme für Ballonnavigation, Lottozahlen, Skat, Bundesliga, druckender Tischrechner, Säurekonstante, Schrödingergleichung, Energieeigenwerte. Das Bemühen dieser Buchreihe ist, nicht nur verwertbare Programme zu vermitteln, sondern durch die Art der übersichtlichen Darstellung Beispiele zu setzen und ein Übertragen auf ähnliche Problemstellungen zu ermöglichen.

Aus der Branche

Rockwell International: Äußerst schnelle Mehr-Protokoll Übertragungsbausteine jetzt für 8- und 16-Bit Prozessoren in Mustern lieferbar. Es handelt sich um den R68561 (48 Pin) für Prozessoren mit der 68000 CPU (16 Bit) und um den MPCC R65560 (4o Pin) für 8-Bit Busse. Sie sollen mit einer seriellen Übertragungsrate von bis zu 4 Megabit/Sek. die schnellsten am Markt sein und 36 bzw. 30 Dollar in Stückzahlen kosten. Sie unterstützen alle gebräuchlichen Protokolle, synchron, asynchron und isochron, zeichenorientiert (wie IBM Bisync) in ASCII oder EDCDIC-Code oder bitorientiert wie SDLC/HDLC.

In der Familie der RM 65 Module trat das RM65-5303E hinzu für A/D-Konvertierung von bis zu 16 Eingangsspannungen bzw. 8 Differentialspannungen. Es kann für 5 Spannungsbereiche auf eine Genauigkeit von 12 Bit kalibriert werden.

R68561 (MPCC)



Valvo: Es wird derzeit unter dem Arbeitstitel SAA 5350 eine hochintegrierte Schaltung entwickelt, die für Bildschirmtext gem. CEPT-Standard alle nowendigen Attribute-Steuerungen sowie einen Zeichengenerator für sämtliche ca. 520 definierte Zeichen enthält. Es wird damit der preisgünstige Aufbau von Bildschirmtextdekodern ermöglicht.

FORCE Computers (68000, 16 Bit): Für die Platinencomputer PROFI KIT 2 und CPU1 wird jetzt die Sprache FORTH in 2 EPROMs angeboten (DM 387,60 inkl. MWST), und zwar als FIG-FORTH-Implementierung mit Datentypen in 16 und 32 Bit sowie mit Strings. Daneben gibt es einen Screen-Editor. – An Platinen traten die SASI-1 zur Ansteuerung von Floppy- oder Winchesterlaufwerken hinzu (DM 3.600) sowie die SIO-1 mit 6 seriellen E/A-Kanälen RS 232C, die vielfältige Protokolle verarbeiten.

Die GWK Gesellschaft für technische Elektronik in Herzogenrath bietet jetzt zum Profi Kit 2 der Fa. FORCE ein Systembusinterface an, das fast alle Karten der GWK anzuschließen gestattet. Es hat 20 Adreßbusleitungen und 8 für den Datenbus, ferner Stecksockel für Programme im 16 Bit-System. Dem Profi Kit stehen damit reichhaltige Anschlußmöglichkeiten zur Verfügung. Der Preis des Interface soll zwischen 500 und 600 DM liegen.

Grafik mit 64000 Bildpunkten als Overlay zur CBM-Bildausgabe für Systeme CBM 2/3/4/80xx wird als Zuszatzkarte zu DM 498 angeboten von Dipl.-Ing. Jürgen Bergeler, Oberer Laubenheimer Weg 56 in 6500 Mainz 1, Tel.: 06131-61 14 50 oder 85 593.

Interfaces auf Centronics-Schnittstellen (Drucker) werden von der Firma Reinhard Wesemann, Winchenbachstr. 3 a in 5600 Wuppertal 2, Tel.: 0202-51 04 44 angeboten, und zwar für VC-20 und VC-64-Computer (für deren seriellen Bus) und für Computer mit V24-Schnittstelle. Letzteres ist ein intelligentes Interface, daß sich selbstlernend auf alle Übertragungsparameter sowie auf unterschiedliche Steckerbelegungen der Schnittstelle einstellt. Es kann ferner 15 Codewandlungen vornehmen.

Disk-Interface für FIG-FORTH und CBM-Computer. Heft 29 dieser Zeitschrift enthielt einen Arkel nebst Programmscreens unter dieser Überschrift. Interessierte Leser können nun ein lauffähiges FORTH-System wie dort beschrieben beim Autor beziehen. Es arbeitet mit den Disketten 3040, 4040 und 4031 zusammen sowie mit den Rechnern der Serien 3000, 4000 und 8000. In dieses FORTH-System wurden nicht nur die Diskettenoperationen aufgenommen, sondern auch fundamentale Stringfunktionen und Editierfunktionen. Dieses leistungsfähige FORTH konnte hier auf Anhieb in Betrieb genommen werden, es ist für die Einarbeitung und für den ständigen Gebrauch sehr zu empfehlen. Es kann gegen Einsendung einer leeren Diskette und einen Unkostenbeitrag von DM 20,-- beim Autor bezogen werden: Dr. A. Schnell, Schloßweiherstraße 39, 5100 Aachen.

Grandmaster, Schachprogramm für Commodore-Rechner VC-20 (mit 8K RAM), VC-64, 4032 und 8032. Es handelt sich um eine Weiterentwicklung des berühmten Programmes BOSS. Es wurde hier auf dem 4032 getestet, und zeichnet sich durch eine hervorragende Grafik aus, die aus einem speziellen Zeichengenerator-EPROM erzeugt wird. Für VC-20 und VC-64 sollen daneben Farben und Geräusche einstellbar sein. Die Spielstärke läßt sich in 10 Stufen einstellen. Grandmaster kündigt ein Matt bis zu fünf Zügen im Voraus an und gibt auf Wunsch auch Vorschläge für den Gegenspieler. Ein Zug kann für beide Seiten zurückgenommen werden, auch ein Autoplay ist möglich, so daß sich das Programm gut auch für Lehrzwecke einsetzen läßt. In verschiedenen Vergleichen wird die Spielstärke und Überlegenheit dieses Programmes gegenüber anderen betont. Es kostet für die Volkscomputer DM 79 und für 4032/8032 DM 99 bei der Fa. Kingsoft, Schnackebusch 4 in 5106 Roetgen, Tel.: 02408-83 19.

□=

Kleinanzeigen

Interessante, nicht alltägliche Programme für AIM 65, Info gegen frankierten Rückumschlag durch R. Borchers, Habichtweg 19, 5600 Wuppertal 1.

AIM 65 REMON V3.5: Geänderter Monitor in 2*4k-EPROMs. Tastenrepeat, Groß-/Kleinschreibung, neue Monitorbefehle, deutscher Zeichensatz, Erweiterung der Monitor- und Editorbefehle, schnelles Replace u.v.a.m. für DM 98,-- incl. ausführlicher Dokumentation. Bestellungen bei Dipl.-Ing. Rüdiger Wollenberg, Stockumer Str. 234, 4600 Dortmund 50, Tel.: 0231-75 34 77.

Wir suchen Schaltpläne für Commodore-Drucker 3022 und 4022, insbesondere derjenigen Stellen, an denen die Peripherie-Nummer (Primäradresse) verstellt werden kann. KKB Köln, Herr Schünemann, Tel. 0221-20 04-140 (Durchwahl).

8" Siemens Floppy, neu Stck. DM 950,--. AIM 65 mit BASIC/Assembl./Editor in FELTRON-Gehäuse m. Netzteil nur DM 1400,--. **Centronics-Drucker 739, neu DM 1650,-- FD-Controller 1791 NEC** +Datenseparator + Schaltvorschl. DM 90,--. Niehus hobby electronic, Postfach 189, 2320 Plön, Johannisstr. 7, Tel.: 04522-27 42.

Assembler unter FORTH

CROSS–09

Cross-Assembler für MC6809 (Motorola). Tonbandcassette (6,5K), Diskette (CBM 3040/4040) oder 2 EPROMs, deren Inhalt nach hex 200-1B00 kopiert werden muß. DM 380,-- + MWSt.

CROSS–05

Cross-Assembler für MC6805/68705xx (Motorola), auf Cassette, Diskette oder EPROM für D000: DM 320,-- + MWSt.

Alle Assembler sind komfortabel ausgelegt, benutzen die Mnemonics des Herstellers und laufen unter FORTH des AIM 65 (für CBM-FORTH von Lowinski oder PHS ggfs. auf Anfrage). Sie erzeugen EPROM-fähigen Code für beliebige Speicherräume und enthalten alle üblichen Kontrollstrukturen wie BEGIN, WHILE, REPEAT, IF, ELSE, THEN usw. aber auch alle Branches (6809: Longbranches). Es kann mit externen Symbolen und Labels im Programmablauf gearbeitet werden. Zusätzlich sind viele Assembler-Anweisungen implementiert: Byte- Word- und Stringablage, .OPTLIS, Reserve Memory Bytes und binäre Argumente.

Mathe- ROM für 6502

Implementierung von P. Rix (s. Textteil) für Sockel D000. Fließkommaarithmetik und höhere mathem. Funktionen (wie in Microsoft-BASIC) für AIM 65-FORTH(komfortabler eigener Fließkommastapel) und für jedes 6502-Assemblerprogramm (dokumentierte Einsprungspunkte und Argumente). EPROM lieferbar ab Jan. 1983. DM 110,- + MWSt.

Commodore Serie 700 und 600 ROM-Listing DM 69,--

Ca. 300 Seiten, gebunden, ausführliche Dokumentation mit original CBM-Labeln, Belegung der Zero Page, des RAM und des I/O, Cross-Referenzliste. Assemblierfähiger Quelltext. Die Benutzung der Banks/Segmente wird erklärt. Lieferbar ab etwa Mitte August.

LASER 110

PERSONAL-COMPUTER

Prompt lieferbar: Computer-Konsole mit Microsoft-BASIC und Betriebssystem in 16 KB für Anschluß an TV-Gerät oder Daten-Monitor (TV- und Video-Ausgang), BASIC mit 6 signifikanten Ziffern und PRINT USING.

CPU Z80A, 4KB RAM, mit Aufsteckmoduln auf 16 und auf 64 KB erweiterbar (Busleiste). Bildschirmeditor (what you see ist what you get) mit 9 Editierfunktionen, 16 Zeilen, 32 Spalten, 16/4 Grafik (128x64 Pixels), unterstützt von den Befehlen SET, RESET und POINT (ob Pixel gesetzt), Tongenerator mit 32 Frequenzen und 8 Tondauern, 67 vollständige BASIC-Statements durch CTRL + Tastendruck zum Komfort des Programmierens. Cassetten-Interface (handelsübliche Recorder) mit Programm- und Datenabspeicherung, Verify-Befehl. Option: Centronics-Interface.

Mit ext. Netzteil, TV-Anschlußkabel, Kabel für Cassetten-Recorder, Demo-Cassette, deutsches (!) Anwenderhandbuch. Weitere Peripherie in Vorbereitung.

Ein leicht transportables, leistungsfähiges und preiswertes Gerät !

Preis inkl. MWSt, Verp., Porto bei Vorüberweisung/Scheck DM 300
bei NN-Versand DM 305

Roland Löhr, Hansdorfer Str. 4, 2070 Ahrensburg
Tel.: 04 102 - 55 816

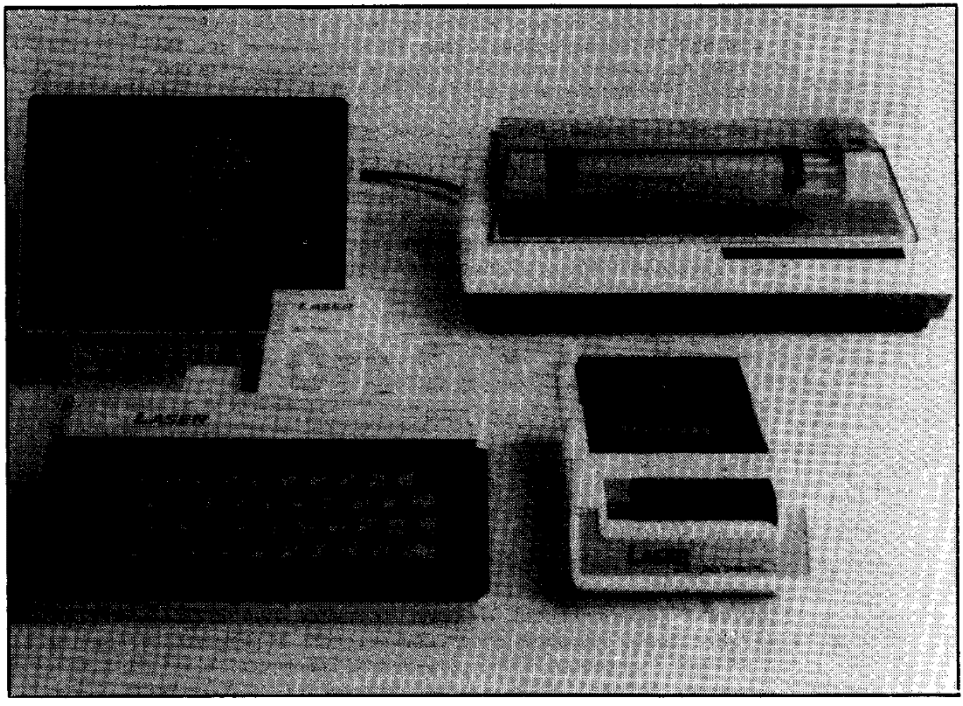
LASER

LASER 110: Mikroprozessor Z80A
16 Kbytes ROM
4 KBytes RAM
Tongenerator

Erweiterung:
16 KBytes RAM
bzw. 64 KBytes RAM
Drucker-Interface-Modul

LASER 220: Mikroprozessor Z80A
16 Kbytes ROM
8 Kbytes RAM
9 Farben
Tongenerator

Erweiterung:
16 KBytes RAM
bzw. 64 KBytes RAM
Drucker-Interface-Modul



SANYO

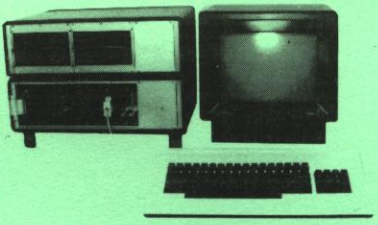
VIDEO VERTRIEB GMBH & CO.
Lange Reihe 29 - D-2000 Hamburg 1
Telefon 040/24 62 66 - Telex 2 174 757

GWK

GESELLSCHAFT FÜR TECHNISCHE ELEKTRONIK mbH.
HARDWARE SOFTWARE SYSTEMENTWICKLUNG

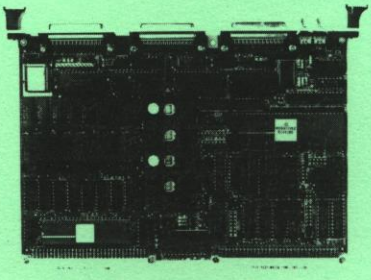
MICROCOMPUTER

FÜR DEN EINSATZ IN TECHNIK U. WISSENSCHAFT



GWK EURO BOARD COMPUTER SYSTEM

Modulares Europakarten System für die 8 Bit CPU s 6809 und 6502



SYS 68K VME

Modulares VME-BUS System mit 16 Bit CPU 68000

FORTUNE 32:16

Hochleistungs System CPU 68000 und
UNIX Betriebssystem
Multiuser Multitasking

D 5120 Herzogenrath Asternstr. 2
Tel.: 02406/6035 Telex: 832109 gwk d

65xx MICRO MAG

COMPUTING · SOFTWARE · HOBBY

Herausgeber:
Dipl.-Volkswirt Roland Löhr
Hansdorfer Straße 4
D–2070 Ahrensburg
Tel.: 04 102 - 55 816

65xx MICRO MAG erscheint zweimonatlich, jeweils Mitte Februar, April usw..Beiträge, die nicht besonders gekennzeichnet sind, stammen vom Herausgeber. - Offsetdruck: Druckartist Gerhard M. Meier, Hamburg 70.

Bezugsbedingungen: Abonnement ab laufender Ausgabe für 6 Hefte DM 54,-- (Inlandsendpreis). Ausland/foreign via surface mail DM 59,--, USA air 26 Dollar. Abonnements laufen bis auf Widerruf mit Kündigungsmöglichkeit bis zu zwei Wochen vor deren Ablauf.

Nachliefermöglichkeiten: Hefte 1-6 und 7-13 sind als Buch nachlieferbar (s. Anzeige unten). Hefte 14-31 können alle als Einzelhefte zu DM 7,80/St. + DM 2,50 je Sendung geliefert werden. Einzelpreis bei 10 und mehr Heften je Sendung DM 6,--.

Private Besteller werden um Überweisung/Scheck (auch Auslandsschecks) zusammen mit der Bestellung gebeten. Konto Roland Löhr, Nr. 654 70-202 Postscheckamt Hamburg, BLZ 200 100 20.