LOOP

11

2. JAHRGANG

Zeitung für Computer-Bauer, -Anwender, -Programmierer und -Starter DM 3,–

Uwe Koch
Frankenstraße 25
5880 LÜDENSCHEID
Tel. (0 23 51) 2 61 92

Erste Software-Partnertagung in Kempten

Entwickler der NDR-Software stellen sich und ihre Produkte vor. Software-Nachschub für die nächsten Jahre gesichert!

Vom 10. bis 12. 10. fand bei GES in Kempten die erste „Software-Partner-Tagung" für den NDR- und mc-CP/M-Computer statt.

Diese Tagung hatte mehrere Ziele:

- Kennenlernen der Partner untereinander
- Vorstellen der Programme, an denen gearbeitet wird, um Doppelentwicklungen zu vermeiden
- Vorstellen der „Wunschliste" für Programme. Diese Wünsche stammen von unseren Mitarbeitern, Kunden und *LOOP*-Lesern
- Vergabe der „Wunsch"-Software an Partner
- Diskussion über Hardware, Verbesserungsvorschläge.

Die Tagung wurde von allen Beteiligten als großer Erfolg gewertet. Die Diskussionen reichten oft bis in die späten Nachtstunden!

Einige Ergebnisse:

– Hardware

Der verbesserungswürdigste Teil ist die GDP64K, hier besonders der langsame Scroll. GES sagte zu, die Baugruppe mit einem Hardware-Scroll-Register zu versehen und völlig aufwärts-kompatibel zu halten. Eine Leiterplatte mit Sockeln soll als preisgünstiger Umbausatz herauskommen (Termin: 1987).

– Hardware-Unterstützung

Hardware-Baugruppen, wie etwa die Analog-Digital-Wandler, sollten durch mehr TOOL-Programme (im Handbuch) unterstützt werden. Beispiel: Einlesen oder Ausgeben unter BASIC oder PASCAL.

– Logic-Analyzer zum NDR-Computer

Eine 10 MHz Logik-Analysator-Baugruppe für den NDR-Computer wurde vorgestellt. (Siehe: Neue Produkte!) Besonders beeindruckend ist die Software dazu, die ganz in Turbo-Pascal geschrieben ist: Analysen lassen sich einlesen, vergleichen, speichern und disassemblieren (Z80). Auch der Datenstrom auf seriellen Schnittstellen kann disassembliert und angezeigt werden!

– Software Einsteiger Z80

Besonders für die Einsteiger fehlen Programme, Tips und Tricks sowie Hinweise. Konkret werden nun einige Programme für das Einsteigerpaket erstellt.

– Lehrsystem

Der Wunsch nach einem Lernprogramm-Editor wurde gestellt. Dieser Editor soll dazu dienen, interaktive Lernprogramme erstellen zu können. Besonders soll die Möglichkeit der Animation, der bewegten Graphik, die die GDP bietet, genutzt werden.

– CP/M für Einsteiger

Besonders Einsteiger wissen mit dem „A)", das am Bildschirm erscheint, wenig anzufangen. Hier soll durch Programme und Lehrmaterial geholfen werden.

– 68000-Betriebssystem

Das „JADOS"-Betriebssystem für 68000 hat sich neben CP/M68K als Standard herausgestellt. Es ist so ausgelegt, daß Programme, die unter JADOS laufen, sofort auf allen drei Prozessoren (68008, 68000, 68020) laufen. Software-Partner, die bestehende Programme an JADOS anpassen wollen, erhalten JADOS kostenlos.

Weitere Betriebssysteme

In Zusammenarbeit mit dem Elektronikladen Detmold sollen weitere Betriebssysteme für den NDR-Computer installiert werden. Fortsetzung Seite 2

Der neue Katalog ist da!

Kaum fertig gedruckt, und schon über 2000 mal verkauft: Unser neuer 168 Seiten dicker, vierfarbiger Katalog ist (endlich) da! Er wurde etwa 5 mal so dick (und so teuer) als wir erwartet hatten. Deshalb können wir unsere Zusage, ihn kostenlos an *LOOP*-Abonnenten zu senden, leider nicht einhalten. Wir bitten um Verständnis: Die Herstellkosten liegen weit über denen eines *LOOP*-Abos! Wir liefern aber sofort ab Lager (auch gegen Rechnung oder Bankeinzug!): Er kostet DM 10.–, Porto und Verpackung sind frei. Noch nicht gesehen? Unbedingt gleich bestellen!

– RL-BASIC mit Gleitkomma-Unterstützung

Das bewährte RL-BASIC für die 68000-Linie wird den Gleitkommaprozessor 68881 auf der CPU68020 unterstützen. Damit dürfte es das schnellste BASIC bezüglich der Arithmetik am Markt sein.

– Fraktal-Darstellung mit 68020 und 68881

Kei Thomsen stellte ein Demo-Programm zur Berechnung von Fractals mit der CPU68020 und dem Coprozessor 68881 vor. Die Berechnung eines Bildes dauerte weniger als eine Minute! Mit einem Graphik-Cursor kann nun ein beliebiger Teilbereich eingegrenzt und wieder berechnet werden. Die Demo war eindrucksvoll!

– Disketten-Editor unter 68000

Der wohl umfangreichste Disk-Editor, der – bildschirmgesteuert – keine Wünsche mehr offen läßt, wurde gezeigt. Er wird in *LOOP* 12 vorgestellt.

– Der NDR-Computer als „Vorleser"

Noch nicht ganz fertig, aber vorführbereit: Ein beliebiger, mit dem RDK-Editor erfaßter Text kann mittels der Sprachausgabe-Baugruppe „vorgelesen" werden. Gut, die Sprache klingt etwas englisch, aber absolut verständlich. Computerei – nun auch für Blinde?

– Spiele, Spiele, Spiele

Was kommt? Ein Schachprogramm, ein 3D-Labyrinth, Mühle mit Graphik und vieles mehr.

– NDR- und MS-DOS: Ein erster Schritt!

Ein Transferprogramm, um mit dem NDR-Computer IMB-Disketten lesen und beschreiben zu können. Texte und Programme können so transferiert werden. Fast fertig, lediglich die Umschaltung der Laufwerke auf 40 Spuren macht noch Probleme.

– Unter dBasell: Hausverwaltung und Rechnungsschreibung.

Für alle, die nie Geld haben: Mit dem Haushaltsprogramm wissen Sie (vielleicht) warum! Siehe „Jetzt lieferbar".

Für kleine Firmen interessant: Eine komplette Fakturierung und Lagerverwaltung, mit Quelltext für unter DM 100.–. Durch den Quelltext und übersichtliche Programmierung leicht zu ändern!

– Video-Verwaltung unter 68008-CPU

Ein Assemblerprogramm für alle Video-Freaks! Bis zu 350 Kasetten können verwaltet, Titel geordnet und gesucht sowie Freiräume gefüllt werden! Siehe neue Produkte!

– Mehr SOUND für den NDR!

Das neue Sound-Programm ermöglicht das Spielen von Melodien mit vielen Sounds und der Sound-Baugruppe. Melodien können gespeichert und editiert werden. Besonderer Gag: Die Plastikhülle, die aus der Tastatur eine Klaviatur machte!

– Heißer Programm-Editor für 6800x-Anwender.

Alle, die immer über den langsamen Scroll geschimpft haben: Der Text mit diesem Editor scrollt so schnell, daß man nicht mehr mitlesen kann! Scroll aber auch in X-Richtung: Bis zu 256 Spalten können erfaßt werden!

Sobald die erwähnten Produkte vertriebsbereit sind, werden wir in der *LOOP* darüber informieren!

In eigener Sache

Ab 1987 haben wirs vor: Die *LOOP* soll dann wirklich regelmäßig erscheinen. Erscheinungstermin: Jeweils am Anfang des ungeraden Monats, also erste Januar-, März-, Mai-... Novemberwoche. Es müßte auch klappen, da nun neben Gerd Graf auch Axel Granel an der *LOOP* mitarbeitet.

Unsere „Großtat" im Herbst war unser Katalog, auf den wir sicher zu Recht stolz sind. Wer ihn noch nicht hat: Einfach gleich bestellen!

Vielen Dank für die Rücksendung der *LOOP*-Umfrage. Besonders gefreut hat uns, daß sich fast alle *LOOP*-Leser bereiterklärt haben, ihre Adresse weiterzugeben und so den Kontakt zu anderen Lesern und Bastlern zu fördern. Wir werden in nächster Zeit unsere Datenbank erweitern (Platte) und diese Anschriften, nach PLZ selektierbar, bereitstellen. Nicht-Datenbankzugreifer können natürlich (ab 1. 1. 87 gerne bei uns anrufen!

Die zweite „Großtat" vieler Engagierter war die Software-Partner-Tagung (siehe Artikel). Wir bekamen weit mehr Anfragen als wir überhaupt einladen konnten. Nicht böse sein: Auch die, die diesmal nicht dabei waren, sind uns als Software-Partner willkommen! Direkter Ansprechpartner bei uns ist A. Granel.

Eine unschöne Geschichte, die aber wohl zu den Kavaliersdelikten zählt, möchten wir noch ansprechen: Das leidige Kopieren von Programmen! Von der Zusammenkunft einer Münchner „Benutzergruppe" wurde uns berichtet: Ein Anwender wollte ein Spielprogramm kopieren und bekam gleich MBASIC, den BASIC-Compiler, HEBAS und Turbo-Pascal mit kopiert. Fein! Wie solls weitergehen? Welches Software-Haus oder welcher engagierte Programmierer ist denn unter diesen Umständen noch bereit, überhaupt etwas Neues zu schaffen?

Wir bemühen uns, die Preise für SW so festzulegen, daß sich Kopieren nun wirklich nicht mehr lohnt. Auch DM 200.– für ein so hervorragendes Werkzeug wie Turbo-Pascal oder dBasell sind sicher nicht zu viel!

P.S. Auch die *LOOP* lebt von den *bezahlten* Ausgaben – also bitte nicht kopieren!

TCM bei Großcomputern

Ein Besuch bei IBM in Montpellier

von Gerd Graf

Im August hatte ich Gelegenheit, auf Einladung der IBM Deutschland das Großrechnerwerk Montpellier (Frankreich) zu besichtigen.

Hier werden fast ausschließlich die größten IBM-Computer – System 3090 – gefertigt. Besonders interessant war ein Vortrag über die technologischen Probleme bei der CPU.

Um extrem hohe Verarbeitungsgeschwindigkeiten zu erreichen, müßten die Signal-Laufzeiten so kurz wie möglich gehalten werden. Eine „konventionelle" Verdrahtung über eine Leiterplatte scheidet also aus.

Die Integration der CPUs und peripheren Bausteine wurde zunächst einmal sehr hoch getrieben, etwa Faktor 10 über der der CPU 68020. Hier taucht nun das Problem auf: Wie bekommt man alle Signale zum und vom Chip?

IBM verwendet hier nun keine gebonderten Golddrähte mehr (auch wegen der Laufzeit und mechanischen Problemen), sondern kontaktiert *direkt auf dem Substrat* mit derzeit 179 winzigen Lötzinn-Kügelchen, die aufgebracht werden. Der „Deckel" ist ein Keramik-Substrat, das gleichzeitig die „Leiterbahnen" zur Verbindung zu weiteren Chips enthält.

Dieses Keramik-Substrat entspricht einer hochgenauen, 36-lagigen (!!) Multi-Layer-Leiterplatte mit etwa 18.000 (!) Verbindungen. Die Größe dieses Substrats ist etwa 9 x 9 cm, die Packungsdichte also extrem hoch. Bis zu 100 Chips können damit verbunden werden!

Ein weiteres Problem ist die bei dieser extremen Integrationsdichte auftretende Verlustwärme. IBM macht das, was unsere Cracks schon längst erkannt haben: Wasserkühlung!

Die abzuführende Wärmeenergie entspricht etwa 40 ... 50 Watt pro cm^2; zum Vergleich hat ein Bügeleisen auf der höchsten Stufe etwa 6W/cm^2.

Neue Wege werden hier schon gesucht: So taucht z. B. CRAY seine gesamten Substrate in eine Flüssigkeit, um zu kühlen. Raten Sie mal, welche Flüssigkeit sich dazu bewährt hat? Nein, nicht Wasser: Blutplasma! Es ist dies eine der wenigen Flüssigkeiten, die sich ohne Druck auf – 22 Grad abkühlen läßt.

Dies nur als ein Detailproblem. Speicher wurden ebenso besprochen, Größenordnungen von Terrabyte (Arbeitsspeicher) gehen den Ingenieuren dort so leicht von der Zunge wie uns das erste Megabyte. (1 Gigabyte = 1000 MByte; 1 TByte = 1000 GByte!)

Was wird kommen? Nun, das größte Problem stellt die Betriebssoftware dar, und dies besonders bei weltweit vernetzten Rechnern, Rechnersysteme, die über 100 oben erwähnter CPUs und Speicher enthalten, sind bald keine Seltenheit mehr; die Vernetzung wird z. B. durch eigene Rechner-Satelliten erfolgen.

Einen „Operator" vor seiner Konsole und ein paar Lämpchen wird es wohl bald nicht mehr geben. Ein solches System, so die Aussage eines Entwicklers, kann wohl bald nicht mehr von einem Menschen bedient werden.

Von wem dann?

Jetzt lieferbar - Jetzt lieferbar

Das Haushaltsprogramm

... oder wie man das Übel an der Wurzel packt!

Benötigtes System: CP/M2.2 System, dBase II

Geht es Ihnen wie uns: Am Monatsende ist man völlig überrascht, wo das Geld geblieben ist! Das Haushaltsprogramm kann Ihnen, falls Sie alle Bewegungen eintragen, darüber Auskunft geben.

Es ist in dBase II geschrieben – der Quelltext wird mitgeliefert. Somit können Sie es leicht an Ihre speziellen Bedürfnisse anpassen und nebenbei auch lernen, wie man unter dBase programmiert und z.B. die Flomon-Graphik aufruft.

Es läuft auf dem NDR-Computer *und auf dem mc-CP/M-Computer!*

Ihre Ein- und Ausgaben werden Hauptrubriken (z.B. PKW, Haus) und darin Unterrupriken (z.B. für PKW: Benzin, Steuer, Pflege, ...) zugeordnet. Weiter wird Ihr Privatkonto geführt – den aktuellen Kontostand erhalten Sie bei jeder Eingabe.

Die Bewegungen innerhalb der Rubriken und Ihres Kontos können Sie nun graphisch in Form von Balken- und Liniendiagrammen oder tabellarisch darstellen. Die Linie des Girokontos entspricht meist einer Sägezahnkurve mit überlagertem Gleichspannungsanteil. Leider stößt sie früher oder später durch die X-Achse...

Das Haushaltsprogramm ist ab Lager lieferbar und kostet **DM 49,–.**

Bestellnummern:
10768 5¼" 80 Spuren
10769 3½" 80 Spuren
10770 8" SSSD

Musik auf dem 68008

von Manfred Zehner
Schwenckestraße 2, 2000 Hamburg 20

Wer seinem 68008er endlich die „Flötentöne" beibringen will, braucht jetzt nur noch zwei Dinge:

- die altbekante *Sound-Karte,*
- das nagelneue *Sound-Programm.*

Damit wird das „Zusammenbasteln" der Lieblingsmelodien zum Vergnügen. Mit dem Soundprogramm lassen sich zum Beispiel

- Töne auf der Tastatur festlegen (damit wird die Tastatur zum „Klavier" – jedes Tastaturfabrikat kann verwendet werden – bis zu 4 Oktaven, d.h. 48 Töne sind möglich),
- Sounds in der Soundbank ablegen (bis zu 255 Klänge oder Geräusche können gespeichert werden, z.B. zum Nachbilden von Instrumentenklängen),
- Töne eingeben (in Einzelschritt- oder in Echtzeit-Eingabe lassen sich über 65000 Töne als Sequenz speichern),
- Musikstücke zusammenstellen (255 Songs, Lieder, Geräuschfolgen usw., lassen sich aus den Ton-Sequenzen zusammenstellen).

Natürlich sind viele Korrekturmöglichkeiten vorgesehen, schließlich macht auch ein Musikgenie seine Fehler.

Darüberhinaus können die Töne bearbeitet werden, wie zum Beispiel

- Tempo bestimmen (hiermit wird die Abspielgeschwindigkeit verändert),
- Transponieren (über 2 Oktaven, d.h. um 24 Halbtonschritte kann man die Töne für ein Musikstück nach oben „verstimmen"),
- Stimmen (wer mit seinem Computer im Duett spielen will, kann damit eventuelle Tonanpassungen vornehmen),
- Verstimmen der Soundprozessortöne (damit können Obertöne und Schwebungen erzeugt werden).

Für jeden, der Zusammenhänge der (Computer-) Musik verstehen oder sich kreativ damit beschäftigen will, ist das Sound-Programm durch seine Hilfen und Möglichkeiten bestens geeignet.

Das Programm ist ab Lager lieferbar und kostet **DM 39,–.**

Bestellnummer:
10681 (Esound) Ein Eprom, relokativ

Laser-Drucker am NDR-Computer

Das ideale Ausgabemedium

Wir bieten ab sofort den Laser-Drucker Kyocera F 1010 zum NDR-Computer an.

Wie funktioniert ein Laser-Drucker?

Nun, prinzipiell wie ein Fotokopierer. Das „Herz" der beiden Geräte ist eine lichtempfindliche Trommel, die auf knapp 1000 Volt negativ aufgeladen wird. Dort, wo Licht auf diese Trommel kommt, wird sie wieder entladen. Das Toner-Pulver wird ebenfalls negativ geladen und haftet nur dort, wo die Trommel entladen, also belichtet wird.

Beim Fotokopierer geschieht diese Belichtung durch eine Optik – beim Laserdrucker durch einen Laser-Strahl, der von einer Laser-Diode erzeugt wird. Die Ablenkung des Strahls erfolgt durch einen rotierenden Spiegel, der Bildaufbau ist ähnlich dem des Fernsehbildes.

In der c't 11/86 wurde der F 1010 im Vergleich mit anderen Systemen getestet und schnitt, besonders was die Bedienung und Programmierung angeht, als deutlich bester ab.

Die Steuerung des Druckers erfolgt über einen 68000-Prozessor mit einem (freien) Speicher von 1.25 MByte. Die Kommandos, um z.B. eine Schriftart zu wechseln, können in den Text gemischt werden. Beispiel:

!R! FONT 10; EXIT;

schaltet auf Font 10 um. Genau so einfach ist es, Graphik zu erzeugen.

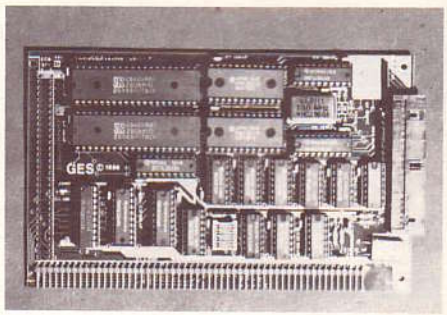
Ein interessantes Anwendungsgebiet eines Laser-Druckers ist das „Desktop Publishing". Darunter versteht man den Sofort-Satz und -Druck, etwa für Anzeigen, Bedienungsanleitungen, Handbücher, Werbeprospekte usw.

Da Dienstleistungsberufe immer mehr gefragt werden, könnten sich hier einige Chancen eröffnen. Was brauchen Sie dazu? Den Laser-Drucker, den NDR-Computer, z.B. mit der COL256 und „etwas" Software.

Dieses Programm soll die Befehle des Laser-Druckers interpretieren und auf dem Bildschirm darstellen, um so die Gestaltung auf dem Bildschirm vorzunehmen. Wer hat Lust zu einem solchen Programm?

Der Drucker ist ab Lager lieferbar und kostet knapp unter DM 10000,—. Detaillierte Prospekte versenden wir gerne auf Anfrage.

Neu: 10 MHz Logikanalysator für den NDR-Klein-Computer



Mit der LOG16-Baugruppe wird aus dem NDR-Computer in der CP/M2.2-Version (Z80, Floppy) ein 10 MHz Logik-Analysator! Ein solches Meßgerät war bisher für den Hobby-Anwender fast unerschwinglich. Die Abtastrate von 10 MHz erlaubt eine Darstellung von z.B. drei Funktionen um die CPU Z80. Sowohl beim Bausatz als auch beim Fertiggerät ist die umfangreiche Software enthalten. Diese Software enthält alle notwendigen Files zum Messen und Darstellen, als auch Files zum Abspeichern von Messungen und Parametern sowie einen Z80 Disassembler (Software-Funktionen siehe unter Abb. 1 und 2). Wie der Bildaufbau aussieht wird in Abb. 3 und 4 dargestellt.

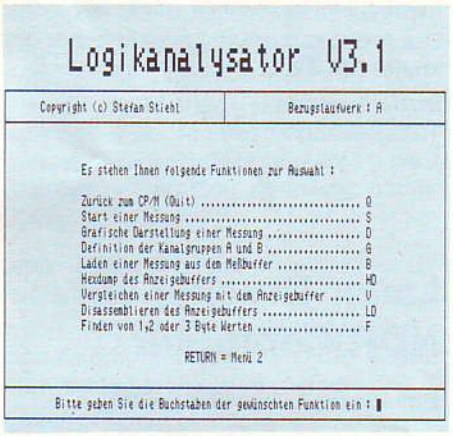


Bild 1

Bild 2

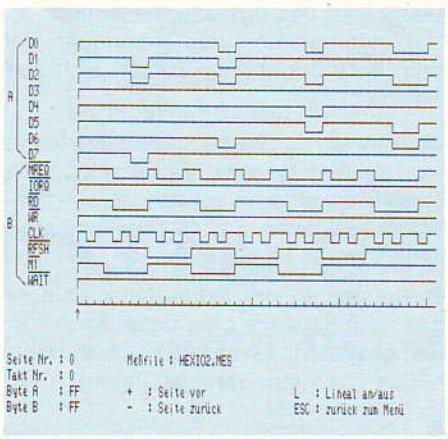


Bild 3

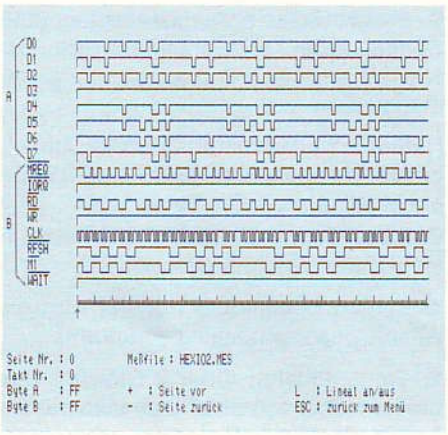


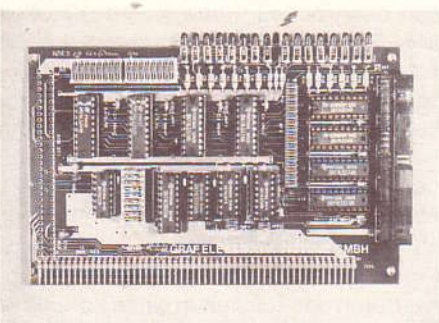
Bild 4

Preise:

10741	LOG16F Fertiggerät + SW	**598,–**
10743	LOG16P Leiterplatte + SW	**298,–**
10742	LOG16B Bausatz + SW	**498,–**
10744	LOG16H Handbuch	**20,–**

Technische Daten:

Steuerbausteine:	2 x Z80 A PIO
Meßspeicher:	2 x 2 KB RAM (2048 Meßwerte pro Kanal, 2 Byte Abtastbreite = 16 Kanäle)
Anschluß:	direkt auf die Busplatine stecken
Abtastfrequenz:	intern 10 MHz constant extern max. 10 MHz
Triggermöglichkeit:	intern (= manuel), extern, Triggerbyte
Stromaufnahme:	ca. 0.4 A
Befehle der Software:	'S' = Start Messung 'D' = Darstellung einer Messung
(Auszug)	'ML' = Messung von Diskette laden 'MS' = Messung auf Diskette schreiben 'B' = Laden einer Messung in den Anzeigebuffer 'HD' = Hexdump des Anzeigebuffers 'LD' = Disassemblieren des Anzeigebuffers 'V' = Vergleich mit Anzeigebuffer usw.



IOE2

Die IOE2 ist eine universelle Ein/Ausgabe-Baugruppe mit jeweils vier Ein- und vier Ausgabe-Ports. Zwei der Ein- und Ausgabe-Ports sind absolut gleich wie bei der IOE. Dadurch ist die IOE2 voll aufwärtskompatibel zur IOE. Die beiden anderen Ports sind mit DIL-Schaltern bzw. mit Leuchtdioden versehen. Diese DIL-Schalter und Leuchtdioden eignen sich hervorragend als Aus- und Eingänge für SPS. Außerdem wurde auch noch ein Stecker vorgesehen, auf dem die Signale für die Centronics-Schnittstelle liegen.

Technische Daten:

Spannungsversorgung:	+ 5 V
Stromverbrauch:	max. 400 mA
Busformat:	NDR- oder ECB-Bus
Größe der Leiterplatte:	100 x 160 mm
Ports:	4 Eingabeports 4 Ausgabeports

- aufwärtskompatibel zur IOE
- 16 Schalter und 16 LED's für direkte Ein- und Ausgabe, besonders geeignet für SPS
- Stecker für Centronicsschnittstelle
- Ports voll ausdekodiert und frei einstellbar (mit DIL-Schalter)

Preise:

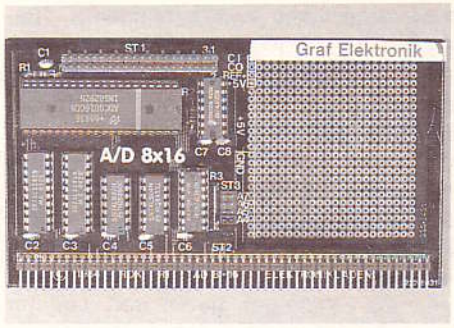
10625	Fertiggerät	**200,–**
10626	Bausatz	**150,–**
10627	Leiterplatte	**30,–**
10628	Handbuch	**20,–**

AD 8/16

Diese Baugruppe wandelt eine Eingangsspannung von 0 bis 5 V in 100 usec in ein 8-Bit-Datenwort. Dabei kann der Wandlerbaustein (ADC0916) 16 Analogspannungseingänge von je 0 bis 5 V multiplexen. Damit ist diese Baugruppe für Meßplätze mit mehreren Meßquellen hervorragend geeignet.

Technische Daten:

Betriebsspannung:	+ 5 V
Stromaufnahme:	max. 150 mA
Leiterplattengröße:	145 x 75 mm
Bus:	NDR-Bus
Wandlungszeit:	100 usec
Auflösung:	8 Bit
Analogspannungs-eingänge:	16
Eingangsspannung:	0 – + 5 V



Betriebstemperatur:	– 40 bis + 80 C
Betriebsfrequenz des Wandlers:	660 kHz

Preise:

10072	Fertiggerät	**215,00**
10071	Bausatz	**147,99**
10073	Leiterplatte	**30,00**
10732	Handbuch	**10,00**

CENT2

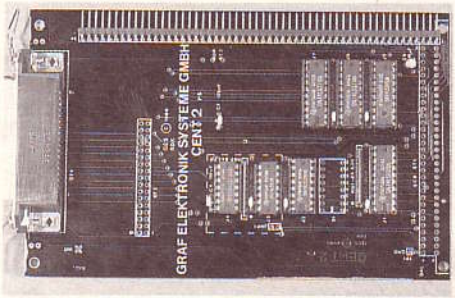
Die CENT2 ist eine eigenständige Baugruppe (nicht wie die CEN eine Aufsteckplatine auf die IOE). An diese Baugruppe kann direkt das Centronicskabel zum Drucker angeschlossen werden. Außerdem ist die CENT2 für Interruptsteuerung vorbereitet.

Technische Daten:

Betriebsspannung:	+ 5 V
Stromverbrauch:	max. 180 mA
Bus:	NDR oder ECB-Bus
Leiterplattengröße:	100 x 160 mm
Ausgang:	Centronics-Schnittstelle (parallel)

Preise:

10136	Fertiggerät	**195,–**
10135	Bausatz	**129,–**
10138	Leiterplatte	**39,–**
10137	Handbuch	**10,–**



IN LETZTER MINUTE:

Ausstrahlung neuer Sendungen NDR 3

1. 05. 11. 1986
17.40 Uhr **Roboter-Steuern**

2. 12. 11. 1986
17.40 Uhr **Springender Punkt**

3. 18. 11. 1986
17.15 Uhr **Kreise mit Ecken**

4. 27. 11. 1986
16.30 Uhr **Daten durch's Telefon**

Für Einsteiger Z80 SBC2

Der Einsteiger und sein Paket

von Mike Hayduk

Der Christiani-Kurs: Mikroelektronik und das Einsteigerpaket

Das Einsteigerpaket, ob als Bausatz oder Fertiggerät, ist gedacht für diejenigen, die nicht damit zufrieden sind, nur die Tasten eines PC zu bedienen sondern wissen wollen, was denn nun in diesem 40-beinigen Kästchen mit dem Namen CPU passiert. Sie wollen, oft autodidaktisch, nicht vorhandenes Wissen erwerben und sind also an mehr als der reinen Anwendung interessiert.

Insofern war es nur folgerichtig, in den Lieferumfang des „Einsteigerpaketes" den ersten Teil des Christiani-Kurses „Mikroelektronik, Einführung mit dem NDR-Computer" aufzunehmen (vgl. *LOOP* Nr. 2), auch wenn das eine Preiserhöhung zur Folge hatte.

Denn wir werden bei den meisten Anfängern gewisse Elekronikkenntnisse (Widerstand, Strom, Spannung, usw.) voraussetzen dürfen, nicht aber das Wissen um die Grundlagen der Computertechnik (Sedezimalsystem, Register etc. als Stichworte).

Das Buch von Rolf-Dieter Klein, das ebenfalls dem Paket beiliegt (Titel: „Mit Hexmon Programme entwickeln") ist eigentlich für Fachleute geschrieben, die schon erhebliche Kenntnisse der Hardware und der Maschinensprache mitbringen. In diesem Werk stößt man auf den Satz: „Am Besten man blättert einmal im Listing (. . .) und lernt so HEXMON verstehen." Ich habe das als echter Anfänger, als Einsteiger, getan und war mir nach der dritten Adresszeile klar: So geht es nicht! Der wirkliche Einsteiger kann dieses Buch getrost zur späteren Verwendung auf einem der oberen Regale im Bücherschrank deponieren.

Meine eigenen Erfahrungen mit dem NDR-Computer begannen mit dem Zusammenbau des Inhaltes des Einsteigerpaketes. Mit der schon vorhandenen Lötpraxis und dem erforderlichen kleinen Lötkolben schritt die Arbeit schnell voran. Die Baubeschreibungen sind auch einem Anfänger durchaus zuzumuten, aber man muß genau und sorgfältig lesen. Daß auch meine Wenigkeit nicht vor Flüchtigkeitsfehlern gefeit ist, erfuhr ich nach dem Vollenden der Hexio-Ein-/Ausgabe, als erst die LED's die Funktion verweigerten und ich dann dahinterkam, daß die Bedienungstasten verkehrt herum eingelötet waren. 24 mal 4 Kontakte schimpfend und schmachvoll wieder auslöten: Der Kenner weiß, wovon ich spreche. Nachträglich findet man natürlich im Text genügend Hinweise auf die richtige Lage der Tasten; also nochmals: lesen, überlegen, und dann erst zur Tat schreiten.

Kurze Zeit später bekam ich den ersten Teil des Christiani-Kurses in die Hand. Da ich schon mit dem Kursus „Elektronik" Erfahrungen sammeln konnte (vgl. *LOOP 6*), machte ich mich sofort an die Arbeit. Um es vorweg zu sagen: Mit dem gewohnten Christiani-System der engen Verflechtung von Theorie und Experiment (dem das Thema „Mikrocomputer" sicher besonders entgegen kommt) ist es den Lehrmeistern aus Konstanz wiederum gelungen, ein kompliziertes Thema für den Anfänger „von Punkt Null" an aufzubereiten.

Die ersten zwanzig Seiten mögen den Einsteiger langweilen, der sein selbstgebautes System endlich laufen lassen möchte (sie sind der Theorie der Programmerstellung gewidmet): sie machen klar, daß ohne Problemanalye und Lösung des Problems durch den Programmierer keine sinnvolle Arbeit mit Computern möglich ist.

Ein längeres Programm, dessen Arbeitsweise zunächst völlig schleierhaft bleibt, wird dennoch schon hier eingebaut (Ziffernanzeige Uhr), um den Drang zur praktischen Arbeit zu befriedigen. Ausführlich wird das Sedezimalsystem („Hexadezimalsystem") erläutert, dessen Kenntnis zur Programmierung in Maschinensprache unerläßlich ist.

Im zweiten Teil des Lehrganges geht es dann wirklich zum Kern der Sache: der inneren Architektur der CPU. Die verschiedenen Register werden besprochen, es wird erläutert wie man Registerinhalte betrachtet, verändert oder im Speicher ablegt. Jeder „Op-Code" wird mit kleinen Beispielprogrammen erläutert (und nachgeprüft) wobei hilfreicherweise die Reihenfolge der zu betätigenden Tasten angegeben ist; erfahrungsgemäß kommt man hier anfangs immer wieder einmal durcheinander. Auch das Sedezimalsystem wird weiter vertieft (addieren, subtrahieren) und das Dualsystem ausführlich erörtert.

Am Ende des 2. Kursteiles kann man immerhin schon indirekt addressierte Speicherzellen laden oder etwa Register in- oder dekrementieren.

Im dritten Kursteil wird die Umgebung der CPU, die „Peripherie" genauer betrachtet. Die „Ports" und ihre Addressierung werden erläutert. Für eine Reihe von Versuchen wird hier eine zweite IOE-Baugruppe und acht LED mit Vorwiderständen benötigt. (Die IOE läßt sich später selbstverständlich weiterverwenden, z.B. zum Anschluß eines Schnelldruckers o.ä.). Ausführlich wird auf das Abspeichern von Daten auf einen Kassettenrekorder eingegangen.

Verknüpfungsbefehle (Exclusiv- Oder, Und-Verknüpfung), die Arithmetik- und die Schiebebefehle sowie die unbedingten und relativen Sprünge sind ebenfalls Themen des dritten Kursteiles.

Der vierte Abschnitt des Kurses schließlich ist dem Statusregister gewidmet, das ja zu einem wesentlichen Teil die „Intelligenz" der CPU durch bedingte Sprünge darstellt.

Die Verwendung von Unterprogrammen (in Maschinensprache, wohlgemerkt) wird erläutert und weitere Befehle (Push, Pop) sowie die restlichen Register werden besprochen. Auch die Einzelheiten der LED-Anzeige des Computers und die Art und Weise, wie der Rechner eine bestimmte Zahl "ausgibt" und darstellt birgt kein Geheimnis mehr [(Eine Übung zum Verständnis: man überlege sich, wie man den Computer dazu bringen kann, ein vorher definiertes „unsinniges" Anzeigemuster auszugeben!)]

Erhellt wird auch die Frage, wie der Rechner durch das Schließen eines Kontaktes (= drücken einer Taste auf der HEXIO-Tastatur) sich eine bestimmte Zahl einverleibt. Neben vielen anderen Einzelheiten wird alles ergänzt durch eine Befehlsliste des Z 80 und ein Register des Gesamtkurses.

Was läßt sich rückblickend besonders hervorheben?

Erstens: Die Materie ist sehr komplex und verlangt oft Rückgriffe auf anfänglich Gelerntes, das heißt, es erfordert auch hier Konzentration und Lerndisziplin, will man wirkliches Verständnis erreichen.

Zweitens: Mit dem Einsteigerpaket lassen sich durchaus schon sehr viele Übungen durchführen, zur praktischen Anwendung der Einheit ist noch mindestens eine IO-Karte erforderlich.

Drittens: Nach einem Durcharbeiten des Kurses wird man die Z80-CPU sicherlich noch nicht beherrschen, aber einen sehr intensiven Einblick in ihre Arbeitsweise erhalten haben und auch imstande sein, eigene Problemstellungen anzugehen, gemachte Fehler zu beschreiben (auch: warum).

Was blieb mir noch unklar?

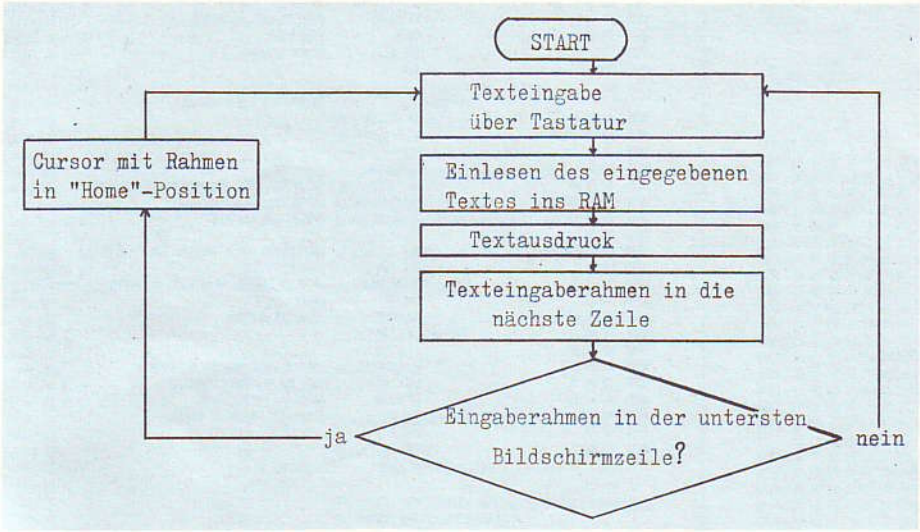
Sicherlich wird bei den Meistern hier der Wunsch entstehen, die vorhandenen Möglichkeiten durch Bildschirm, Tastatur etc. zu erweitern und komfortabler zu gestalten. Und dies scheint mir der sinnvolle Weg zu sein: von der CPU und ihrem Verständnis in die Peripherie vorzustoßen, und nicht etwa ein vollständiges Mikrocomputer-System herzunehmen und dann sozusagen in die Tiefe zum Mikroprozessor vorzudringen.

Allen Wissensdurstigen viel Erfolg und Freude beim Durcharbeiten des Kurses.

Kleine „Schreibmaschine" mit der SBC2

```
8800 Eingabe:    21 16 00    ld HL,22d      22 Zeilen für den Text
8803             cd 0f 00    cd schleife
8806             21 11 88    ld HL,8811     Textblockbeginn
8809             0e 01       ld C,1         mit Eingaberahmen
880B             cd 21 00    cd read        Texteingabe
880E             c3 e8 88    c3 print       Druckerausgabe
8811             03 00                      Cursorposition X-Richtung
8813             ef 00                      Cursorposition Y-Richtung
8815             11 00                      Schriftgröße
8817             4f 00                      80 Zeichen pro Zeile
8819-            00 00 00 00...             dieser Bereich wird mit
885F          ...00                         Nullen gefüllt
8860-            23 23 23 23...             dieser Bereich wird mit
88BF          ...23                         Nummernzeichen gefüllt
```

```
88C0 Centronics:  c5          push BC         Druckroutine mit kleinen
88C1              d5          push DE         Änderungen aus
88C2              e5          push HL         LOOP 5 Seite 3
88C3              0e 49       ld C,49h        von W. Reimann

88C5 Busy:        ed 40       in B,(C)
88C7              cb 40       bit 0,B
88C9              20 fa       jr nz,busy
88CB              d3 48       out (48h),A
88CD              06 ff       ld B,ffh
88CF              ed 41       out (C),B
88D1              06 00       ld B,00
88D3              ed 41       out (C),B
88D5              06 ff       ld B,ffh
88D7              ed 41       out (C),B
88D9              e1          pop HL
88DA              d1          pop DE
88DB              c1          pop BC
88DC              c9          ret

88DD Crlf:        3e 0d       ASCII-Code für CR (Carridge Return)
88DF              cd c0 88    cd centronics
88E2              3e 0a       ASCII-Code für LF (Line Feed)
88E4              cd c0 88    cd centronics
88E7              c9          ret
88E8 Print:       21 19 88    ld HL,8819      Textanfang

88EB Zeichen:     7e          ld A,HL
88EC              fe 23       cp 23h          Vergleich, ob Nummernzeichen
88EE              28 06       jr z,88F7       wenn ja, dann ausdrucken
88F0              cd c0 88    cd centronics
88F3              23          inc HL          nächste Zeichenadresse
88F4              18 f5       jr zeichen
88F7              cd dd 88    cd crlf
88F9              2a 13 88    ld HL,(8813)    in 8813 abgelegte Cursor-
88FC-             2b 2b 2b 2b..               position um 11 Bild-
8906          ...2b           dec HL          schirmzeilen verringern
8907              22 13 88    ld (8813),HL    neue Position nach 8813
890A              01 50 00    ld BC,80d       80 Zeichen löschen
890D              21 70 88    ld HL,(8870)    Löschtext (Nummernzeichen)
8910              11 19 88    ld DE,8819      ab 8819 wird überschrieben
8913              ed b0       Blocktransfer   80 Zeichen-Übertragung
8915              3e 23       ld A,23h        letztes Zeichen immer
8917              32 6a 88    ld 886A,A       ein Nummernzeichen
891A              cd 12 00    cd endschleife
891D              3e ef       ld A,efh        ursprüngl. Y-Position
891F              32 13 88    ld 8813,A       Y-Position in 8813
8922              3e 00       ld A,00         auch MSB auf Null setzen
8924              32 14 88    ld 8814,A       da ein Überlauf entsteht
8927              c3 00 88    c3 eingabe      Rücksprung zum Pr.-beginn
```



PROGRAMM-SCHIEBER

Z80-Programme intelligent verschoben – von Christof Hübner, Lessingstraße 2, 6729 Rheinzabern

Beim Arbeiten mit dem Grundprogramm ist es sehr umständlich, Korrekturen an bereits geschriebenen Programmen anzubringen. Man kann zwar zwischen den einzelnen Programmteilen oder auch im Programm selbst Lücken lassen und diese mit NOP's füllen, doch trägt das nicht gerade zur Übersichtlichkeit bei. Ganz übel ist man dran, wenn man ein Unterprogramm, welches häufig aufgerufen wird, verschieben muß, da man sämtliche JP- bzw. CALL-Befehle hierzu ändern muß. Schon ein kleiner Fehler, und man hat den schönsten „Salat". (Wehe, wenn's dann noch nicht auf Cassette gespeichert ist!) Durch zusätzliche Sprünge zu „ausgelagerten" Erweiterungen verliert man ebenfalls schnell die Übersicht, denn wer schreibt schon minutiös sämtliche Änderungen aus!

Mein Programm hat daher folgende Aufgaben:

1. Verschieben von Programmen oder Programmteilen.

2. Löschen von Speicherbereichen.

Beim Verschieben von Programmen werden folgende Änderungen vorgenommen:

1.1 Der gesamte Speicher der SBCII (falls erwünscht auch ein Bereich der vor der Adresse 8000h liegt, z.B. auf einer ROA64) wird nach Sprungbefehlen in den transportierten Bereich durchsucht. Falls einer gefunden wird, wird er auf die neue Adresse umgeschrieben.

1.2 Die Symbole, die auf Speicherplätze im transportierten Bereich weisen, werden so geändert, daß sie auf den neuen Speicherplatz zeigen.

Achtung: Verschiebt man Texte oder reservierte „Merker", so muß man die Stellen im Programm, an denen die Adressen der entsprechenden Speicherstellen (durch die Symbole definiert) in bestimmte Register der CPU geladen werden (wie z.B. für die Grundprogrammroutine „WRITE"), ändern. Dies geschieht im „aendern" Modus des Grundprogrammes durch Eintragen des Symbolnamens an der jeweiligen Stelle, denn das Symbol zeigt ja nach dem Programmlauf schon auf den neuen Platz.

Gestartet wird das Verschiebeprogramm auf Adresse 8D00h. Es fragt nach der Anfangsadresse und der Endadresse des zu verschiebenden Bereiches sowie nach der Zieladresse, wohin der Bereich verschoben werden soll (hier liegt dann das erste Byte, welches bisher bei der Anfangsadresse stand). Mit der Eingabe von (S) als erstem Zeichen wird das Programm abgebrochen, und man gelangt mit (M) ins Hauptmenue zurück.

Das Löschprogramm beginnt bei der Adresse 8E99 und füllt den Bereich von der Anfangsadresse bis zur Endadresse (einschließlich) – sie werden wieder erfragt – mit 00h.

Das Programm belegt zwar einen relativ großen Teil des Speicherplatzes auf der SBCII, doch hat es mir bei der Programmerstellung gute Dienste geleistet.

```
BLOCK 17.01.1986                                        PAGE  1

                        ;**********************************
                        ;**  BLOCKTRANSPORT UND LOSCHEN   **
                        ;**    (C) BY CHRISTOF HÜBNER     **
                        ;**          LESSINGSTR. 2        **
                        ;**          6729 RHEINZABERN     **
                        ;**********************************

                        TITLE BLOCK 17.01.1986
                        ORG 8000H

                WRITE   EQU 001EH
                READ    EQU 0021H
                WAIT    EQU 00F8H
                SEDRNG  EQU 0007H  ;GRUNDPROGRAMMROUTINE
                LENGTH  EQU 0D2AH  ;       "
                ANDBEG1 EQU 8800H  ;ANFANGSADRESSE DES ERSTEN DURCHSUCHTEN BEREICHES
                ANDEND1 EQU 8D00H  ;ENDADRESSE DES ERSTEN DURCHSUCHTEN BEREICHES
                ANDBEG2 EQU 2200H  ;ANFANGSADRESSE DES ZWEITEN DURCHSUCHTEN BEREICHES
                ANDEND2 EQU 8000H  ;ENDADRESSE DES ZWEITEN DURCHSUCHTEN BEREICHES

                BLOCK:          ;HAUPTPROGRAMM
8000 21 55 8F           LD HL,ÜBSRT
8003 CD 1E 00           CALL WRITE      ;ÜBERSCHRIFT DRUCKEN
8006 21 A6 8F           LD HL,ERKL
8009 CD 1E 00           CALL WRITE      ;ERKLARUNG DRUCKEN
800C CD 3A 80           CALL ANFEING
800F CD 50 80           CALL ENDEING
8012 CD 66 80           CALL ZIELEING
8015 CD EF 80           CALL TRANSP
8018 CD 2E 8E           CALL SYMBMOD
801B 21 00 88           LD HL,ANDBEG1
801E 22 DF 8F           LD (ANDERBEGINN),HL
8021 21 00 8D           LD HL,ANDEND1
8024 22 E1 8F           LD (ANDERENDE),HL
8027 CD CC 8E           CALL PROGAND    ;BEREICH AUF SBC11
802A 21 00 22           LD HL,ANDBEG2                   ;KANN WEGFALLEN, WENN
802D 22 DF 8F           LD (ANDERBEGINN),HL             ;NUR MIT SBC11 GEARBEITET
8030 21 00 80           LD HL,ANDEND2                   ;WIRD, MUSS DANN MIT
8033 22 E1 8F           LD (ANDERENDE),HL               ;NOP'S GEFÜLLT WERDEN
8036 CD CC 8E           CALL PROGAND    ;BEREICH VOR 8000H     ;
8039 C9                 RET

                ANFEING:        ;NACH ANFANGSADRESSE FRAGEN
803A 21 6A 8F           LD HL,BLOCKA
803D CD 1E 00           CALL WRITE
8040 21 96 00           LD HL,0096H
8043 22 C8 8F           LD (YPOFE),HL
8046 CD 7C 8D           CALL EINGAB
8049 21 D3 8F           LD HL,ANFADR
804C CD BB 8D           CALL UMWAND
804F C9                 RET

                ENDEING:        ;NACH ENDADRESSE FRAGEN
8050 21 7E 8F           LD HL,BLOCKE
8053 CD 1E 00           CALL WRITE
8056 21 64 00           LD HL,0064H
8059 22 C8 8F           LD (YPOFE),HL
805C CD 7C 8D           CALL EINGAB
805F 21 D5 8F           LD HL,ENDADR
8062 CD BB 8D           CALL UMWAND
8065 C9                 RET

                ZIELEING:       ;NACH ZIELADRESSE FRAGEN
8066 21 92 8F           LD HL,BLOCK2
8069 CD 1E 00           CALL WRITE
806C 21 32 00           LD HL,0032H


BLOCK 17.01.1986                                        PAGE  2

806F 22 C8 8F           LD (YPOFE),HL
8072 CD 7C 8D           CALL EINGAB
8075 21 D7 8F           LD HL,ZIELADR
8078 CD BB 8D           CALL UMWAND
807B C9                 RET

                EINGAB:         ;EINGABE UND TEST AUF RICHTIGKEIT
807C 21 00 00           LD HL,0
807F 22 CE 8F           LD (PU),HL
8082 22 D0 8F           LD (PU+2),HL    ;PUFFER LÖSCHEN
8085 21 C6 8F           LD HL,PUFFER
8088 0E 01              LD C,01
808A CD 21 00           CALL READ       ;EINLESEN DER ADRESSEN
808D 3A CE 8F           LD A,(PU)       ;ERSTES EINGEGEBENES ZEICHEN AUS PUFFER HOLEN
8090 CB AF              RES 5,A         ;s->S
8092 FE 53              CP 'S'
8094 CA EC 80           JP Z,SCHLUSS    ;FALLS 'S', ABBRUCH
8097 3A CD 8F           LD A,(AKT)
809A FE 04              CP 04           ;SIND 4 ZEICHEN EINGEGEBEN WORDEN ?
809C C2 7C 80           JP NZ,EINGAB    ;WENN NEIN, NOCHMAL EINGABE
809F FD 21 CE 8F        LD IY,PU
80A3 3E 00              LD A,00
80A5 CD F8 00           CALL WAIT
80A8 D3 70              OUT (70H),A     ;AUF POSITIVE SCHRIFT SCHALTEN
80AA FD 7E 00           TEST: LD A,(IY+0)
80AD FE 00              CP 00           ;0 IST SCHLUSSZEICHEN
80AF C8                 RET Z           ;FALLS '00', ABBRUCH
80B0 CD 07 00           CALL SEDRNG     ;TEST OB SEDEZIMALES ZEICHEN
80B3 FD 23              INC IY          ;NACHSTES ZEICHEN
80B5 DA 7C 80           JP C,EINGAB     ;FALLS NICHT SEDEZIMAL, NEUE EINGABE
80B8 D2 AA 80           JP NC,TEST

                UMWAND:         ;UMWANDLUNG DER ASCII-ZEICHEN IN SEDEZIMALZAHL
80BB FD 21 CE 8F        LD IY,PU        ;HL WIRD VORHER AUF DIE SPEICHERSTELLE EINGESTELLT,
80BF FD 7E 02           LD A,(IY+2)     ;IN DER DIE SED.ZAHL NACHHER STEHEN SOLL.
80C2 FE 40              CP 40H
80C4 D4 E9 80           CALL NC,KORR
80C7 ED 6F              RLD
80C9 FD 7E 03           LD A,(IY+3)
80CC FE 40              CP 40H
80CE D4 E9 80           CALL NC,KORR
80D1 ED 6F              RLD
80D3 23                 INC HL
80D4 FD 7E 00           LD A,(IY+0)
80D7 FE 40              CP 40H
80D9 D4 E9 80           CALL NC,KORR
80DC ED 6F              RLD
80DE FD 7E 01           LD A,(IY+1)
80E1 FE 40              CP 40H
80E3 D4 E9 80           CALL NC,KORR
80E6 ED 6F              RLD
80E8 C9                 RET

                KORR:           ;KORREKTUR, FALLS ZEICHEN IN (A...F)
80E9 C6 09              ADD A,09
80EB C9                 RET

                SCHLUSS:        ;STACK REINIGEN
80EC F1                 POP AF
80ED F1                 POP AF
80EE C9                 RET

                TRANSP:         ;TRANSPORT DES BESTIMMTEN BEREICHES


BLOCK 17.01.1986                                        PAGE  3

80EF 2A D5 8F           LD HL,(ENDADR)
80F2 ED 5B D3 8F        LD DE,(ANFADR)
80F6 37                 SCF
80F7 3F                 CCF             ;CARRY LÖSCHEN
80F8 ED 52              SBC HL,DE
80FA E5                 PUSH HL         ;LANGE DES BEREICHES AUF STACK (FÜR TRANSP.-ROUTINEN)
80FB 7D                 LD A,L
80FC C6 01              ADD A,01
80FE 6F                 LD L,A
80FF 7C                 LD A,H
8100 CE 00              ADC A,00
8102 67                 LD H,A
8103 22 D9 8F           LD (LANGE),HL   ;LANGE+1 IN LANGE GESPEICHERT
8106 2A D7 8F           LD HL,(ZIELADR)
8109 7C                 LD A,H
810A 92                 SUB D
810B FA 25 8E           JP M,TRUNT      ;FALLS ZIEL<ANFANG : TRANSPORT NACH UNTEN
810E C2 1A 8E           JP NZ,TROB      ;FALLS ZIEL>ANFANG : TRANSPORT NACH OBEN
8111 7D                 LD A,L
8112 93                 SUB E
8113 FA 25 8E           JP M,TRUNT      ;NIEDERWERTIGES BYTE DER ADRESSE TESTEN,
8116 C2 1A 8E           JP NZ,TROB      ;WENN OBEN GLEICHHEIT
8119 C9                 RET

                TROB:           ;TRANSPORT NACH OBEN
811A C1                 POP BC          ;BC=ENDADR-ANFADR
811B 09                 ADD HL,BC       ;HL=ZIELADR+(ENDADR-ANFADR)
811C EB                 EX DE,HL        ;DE=      "           ; HL=ANFADR
811D 09                 ADD HL,BC       ;HL=ANFADR+(ENDADR-ANFADR)
811E ED 4B D9 8F        LD BC,(LANGE)   ;BC=LANGE=(ENDADR-ANFADR+1)
8122 ED B8              LDDR            ;TRANSPORT
8124 C9                 RET

                TRUNT:          ;TRANSPORT NACH UNTEN
8125 ED 4B D9 8F        LD BC,(LANGE)   ;BC=LANGE
8129 EB                 EX DE,HL        ;DE=ZIELADR , HL=ANFADR
812A ED B0              LDIR            ;TRANSPORT
812C C1                 POP BC          ;STACK SAUBERN
812D C9                 RET

                SYMBMOD:        ;ANDERN DER SYMBOLTABELLE
812E 2A D7 8F           LD HL,(ZIELADR)
8131 ED 5B D3 8F        LD DE,(ANFADR)
8135 A7                 AND A
8136 ED 52              SBC HL,DE
8138 22 DB 8F           LD (OFFSET),HL  ;IN OFFSET STEHT DER ABSTAND ZIELADR-ANFADR
813B FD 21 C9 81        LD IY,81C9H     ;IY=ZEIGER AUF SYMBOLTABELLE
813F FD 7E 00           HOLEN: LD A,(IY+0)      ;ERSTES ZEICHEN HOLEN
8142 FE 00              CP 00
8144 C8                 RET Z           ;RÜCKSPRUNG FALLS TABELLE LEER
8145 CB 7F              BIT 7,A         ;7.BIT GESETZT ?
8147 C2 4F 81           JP NZ,PRÜF      ;WENN JA :PRÜFEN OB ADRESSE IN TRANSPORTIERTEM BEREICH
814A FD 23              INC IY          ;WENN NEIN :
814C C3 3F 81           JP HOLEN        ;NÄCHSTES ZEICHEN
814F FD 6E 01           PRÜF: LD L,(IY+1)
8152 FD 66 02           LD H,(IY+2)
8155 22 DD 8F           LD (ADR),HL     ;IN ADR STEHT ADRESSE DES SYMBOLS
8158 CD 78 8E           CALL GROSSE     ;ADRESSE IN TRANSP. BEREICH ?
815B DA 67 81           JP C,MOD        ;WENN JA, MODIFIZIEREN
815E FD 23              RÜCK: INC IY    ;WENN NEIN, NÄCHSTES SYMBOL
8160 FD 23              INC IY
8162 FD 23              INC IY
8164 C3 3F 81           JP HOLEN

BLOCK 17.01.1986                                        PAGE  4

8167 2A DD 8F           MOD:    LD HL,(ADR)
816A ED 5B DB 8F        LD DE,(OFFSET)
816E 19                 ADD HL,DE       ;HL=ALTE ADRESSE+ABSTAND
816F FD 75 01           LD (IY+1),L
8172 FD 74 02           LD (IY+2),H     ;NEUE ADRESSE IN TABELLE
8175 C3 5E 81           JP RÜCK         ;NÄCHSTES SYMBOL

                GROSSE:         ;PRÜFUNG OB ADRESSE IN TRANSP. BEREICH
8178 ED 4B D3 8F        LD BC,(ANFADR)
817C A7                 AND A
817D ED 42              SBC HL,BC       ;HL=ADR-ANFADR
817F DA 97 81           JP C,NC1        ;C-) ADR<ANFADR -) RÜCKSPRUNG OHNE CARRY
8182 2A DD 8F           LD HL,(ADR)
8185 ED 4B D5 8F        LD BC,(ENDADR)
8189 A7                 AND A
818A ED 42              SBC HL,BC       ;HL=ADR-ENDADR
818C D2 92 81           JP NC,NUTE      ;NC-) ADR>ENDADR -) TEST OB ADR=ENDADR
818F C3 95 81           JP C1           ;ANFADR<ADR<ENDADR
8192 C2 97 81           NUTE:   JP NZ,NC1       ;ADR=ENDADR -) RÜCKSPRUNG OHNE CARRY
8195 37                 C1:     SCF
8196 C9                 RET             ;CARRY
8197 A7                 NC1:    AND A
8198 C9                 RET             ;OHNE CARRY

                LÖSCHEN:        ;LÖSCHEN VON ANFADR BIS ENDADR
```

 - Für Einsteiger - Z80, SBC2

```
8E99  21 B7 8F                  LD HL,LÖÜB
8E9C  CD 1E 00                  CALL WRITE      ;ÜBERSCHRIFT DRUCKEN
8E9F  21 A6 8F                  LD HL,ERKL
8EA2  CD 1E 00                  CALL WRITE      ;ERKLARUNG DRUCKEN
8EA5  CD 3A 8D                  CALL ANFEING
8EA8  CD 50 8D                  CALL ENDEING    ;EINGABEN
8EAB  2A D5 8F          LÖ:     LD HL,(ENDADR)
8EAE  ED 5B D3 8F               LD DE,(ANFADR)
8EB2  A7                        AND A
8EB3  ED 52                     SBC HL,DE
8EB5  22 D9 8F                  LD (LANGE),HL   ;LANGE=ENDADR-ANFADR
8EB8  2A D3 8F                  LD HL,(ANFADR)
8EBB  11 01 00                  LD DE,0001H
8EBE  19                        ADD HL,DE
8EBF  EB                        EX DE,HL        ;DE=ANFADR+1
8EC0  2A D3 8F                  LD HL,(ANFADR)
8EC3  ED 4B D9 8F               LD BC,(LANGE)
8EC7  36 00                     LD (HL),00
8EC9  ED B0                     LDIR            ;BEREICH ANFADR BIS ENDADR WIRD MIT 00 GEFÜLLT
8ECB  C9                        RET

                        PROGAND:        ;ANDERT SPRUNGADRESSEN IM BEREICH AND.BEGINN BIS AND.ENDE
8ECC  2A DF 8F                  LD HL,(ANDERBEGINN)     ;ANFANG DES ZU DURCHSUCHENDEN BEREICHES
8ECF  22 D7 8F          LÄ:     LD (ZIELADR),HL ;ZIELADR IST HIER ZEIGER AUF BEFEHLE !
8ED2  CD 2A 00                  CALL LENGTH     ;HL ZEIGT AUF BEFEHL, IN B NACHHER LANGE DES BEFEHLS
8ED5  48                        LD C,B
8ED6  06 00                     LD B,00
8ED8  ED 43 D9 8F               LD (LANGE),BC   ;IN LANGE STEHT JETZT LANGE DES BEFEHLS
8EDC  79                        LD A,C
8EDD  FE 03                     CP 03
8EDF  CA E5 8E                  JP Z,BEFPR      ;FALLS LANGE=3, PRÜFUNG OB SPRUNGBEFEHL
8EE2  C3 40 8F                  JP NABEF        ;SONST WEITER
8EE5  7E                BEFPR:  LD A,(HL)
8EE6  E6 F0                     AND 0F0H        ;HOHERWERTIGES NIBBLE PRÜFEN
8EE8  FE D0                     CP 0D0H
8EEA  CA 0C 8F                  JP Z,NIZW2
8EED  FE E0                     CP 0E0H
8EEF  CA 0C 8F                  JP Z,NIZW2


BLOCK 17.01.1986                                        PAGE  5


8EF2  FE F0                     CP 0F0H
8EF4  CA 0C 8F                  JP Z,NIZW2
8EF7  FE C0                     CP 0C0H
8EF9  CA FF 8E                  JP Z,NIZW1
8EFC  C3 40 8F                  JP NABEF
8EFF  7E                NIZW1:  LD A,(HL)
8F00  E6 0F                     AND 0FH         ;NIEDRIGERWERTIGES NIBBLE PRÜFEN
8F02  FE 0D                     CP 0DH          ;CD UND C3 BEFEHLE
8F04  CA 26 8F                  JP Z,AND
8F07  FE 03                     CP 03
8F09  CA 26 8F                  JP Z,AND
8F0C  7E                NIZW2:  LD A,(HL)
8F0D  E6 0F                     AND 0FH         ;NIEDERWERTIGES NIBBLE PRÜFEN
8F0F  FE 0C                     CP 0CH          ;DA,DC,D4,D2,EA,EC,E4,E2,FA,FC,F4,F2,CA,CC,C4,C2
8F11  CA 26 8F                  JP Z,AND        ;BEFEHLE
8F14  FE 04                     CP 04H
8F16  CA 26 8F                  JP Z,AND
8F19  FE 0A                     CP 0AH
8F1B  CA 26 8F                  JP Z,AND
8F1E  FE 02                     CP 02H
8F20  CA 26 8F                  JP Z,AND
8F23  C3 40 8F                  JP NABEF        ;FALLS KEINER DIESER BEFEHLE, WEITER
8F26  23                AND:    INC HL
8F27  5E                        LD E,(HL)
8F28  23                        INC HL
8F29  56                        LD D,(HL)
8F2A  EB                        EX DE,HL        ;HL=SPRUNGADRESSE,DE=ADRESSE DES BEFEHLS+2
8F2B  E5                        PUSH HL         ;RETTEN
8F2C  22 DD 8F                  LD (ADR),HL     ;IN ADR SPEICHERN (FÜR GRÖSSE)
8F2F  CD 78 8E                  CALL GRÖSSE     ;SPRUNGADRESSE DES BEF. IN TRANSP. BEREICH ?
8F32  E1                        POP HL          ;RÜCKHOLEN
8F33  D2 40 8F                  JP NC,NABEF     ;WENN NICHT, WEITER
8F36  ED 4B DB 8F               LD BC,(OFFSET)
8F3A  09                        ADD HL,BC       ;HL=SPRUNGADRESSE+OFFSET
8F3B  7C                        LD A,H
8F3C  12                        LD (DE),A       ;HÖHERWERTIGES BYTE DER NEUEN ADRESSE IN PROGRAMM
8F3D  7D                        LD A,L          ;SCHREIBEN
8F3E  1B                        DEC DE
8F3F  12                        LD (DE),A       ;EBENSO NIEDRIGERWERTIGES
8F40  2A D7 8F          NABEF:  LD HL,(ZIELADR)
8F43  ED 4B D9 8F               LD BC,(LANGE)
8F47  09                        ADD HL,BC       ;HL=ADRESSE DES NACHSTEN BEFEHLS
8F48  E5                        PUSH HL
8F49  D1                        POP DE          ;DE= " =HL
8F4A  ED 4B E1 8F               LD BC,(ANDERENDE)       ;ENDE DES ZU ANDERNDEN BEREICHES
8F4E  ED 42                     SBC HL,BC
8F50  D0                        RET NC          ;FALLS ADR. DES NACHSTEN BEFEHLS)ANDERENDE -) SCHLUSS
8F51  EB                        EX DE,HL        ;HL=ADRESSE DES NACHSTEN BEFEHLS
8F52  C3 CF 8E                  JP LÄ

                        ;--------------------TEXTE

                        ÜBSRT:
8F55  50 00 D9 00 44 00 42 6C   DB 50H,00H,0D9H,00H,44H,00H,'Blocktransport',00H
                        BLOCKA:
8F6A  0A 00 96 00 33 00 42 6C   DB 0AH,00H,96H,00H,33H,00H,'Blockanfang :',00H
                        BLOCKE:
8F7E  0A 00 64 00 33 00 42 6C   DB 0AH,00H,64H,00H,33H,00H,'Blockende   :',00H
                        BLOCKZ:
8F92  0A 00 32 00 33 00 5A 69   DB 0AH,00H,32H,00H,33H,00H,'Zieladresse :',00H
                        ERKL:
8FA6  5E 01 14 00 21 00 45 6E   DB 5EH,01H,14H,00H,21H,00H,'Ende = (S)',00H
                        LÖÜB:


BLOCK 17.01.1986                                        PAGE  6


8FB7  80 00 DC 00 44 00 4C 6F   DB 80H,00H,0DCH,00H,44H,00H,'Loeschen',00H
                        PUFFER:
8FC6  0E 01                     DB 0EH,01H
                        YPOFE:
                                DS 2
8FCA  33 00 05                  DB 33H,00H,05H
                        AKT:
                                DS 1
                        PU:
                                DS 5
                        ANFADR:
                                DS 2
                        ENDADR:
                                DS 2
                        ZIELADR:
                                DS 2
                        LANGE:
                                DS 2
                        OFFSET:
                                DS 2
                        ADR:
                                DS 2
                        ANDERBEGINN:
                                DS 2
                        ANDERENDE:
                                DS 2

                                END

BLOCK 17.01.1986                                        PAGE  7


001E WRITE                      0021 READ
00F8 WAIT                       0007 SEDRNG
002A LENGTH                     8800 ANDBEG1
8000 ANDEND1                    2200 ANDBEG2
8000 ANDEND2                    8D00 BLOCK
8D3A ANFEING                    8D50 ENDEING
8D66 ZIELEING                   8D7C EINGAB
8DAA TEST                       8DBB UMWAND
8DE9 KORR                       8DEC SCHLUSS
8DEF TRANSP                     8E1A TROB
8E25 TRUNT                      8E2E SYMBMOD
8E3F HOLEN                      8E4F PRÜF
8E5E RÜCK                       8E67 MOD
8E78 GRÖSSE                     8E92 NUTE
8E95 C1                         8E97 NC1
8E99 LÖSCHEN                    8EAB LÖ
8ECC PROGAND                    8ECF LÄ
8EE5 BEFPR                      8EFF NIZW1
8F0C NIZW2                      8F26 AND
8F40 NABEF                      8F55 ÜBSRT
8F6A BLOCKA                     8F7E BLOCKE
8F92 BLOCKZ                     8FA6 ERKL
8FB7 LÖÜB                       8FC6 PUFFER
8FC8 YPOFE                      8FCD AKT
8FCE PU                         8FD3 ANFADR
8FD5 ENDADR                     8FD7 ZIELADR
8FD9 LANGE                      8FDB OFFSET
8FDD ADR                        8FDF ANDERBEGINN
8FE1 ANDERENDE


no fatal error(s)


SUM=5F99
CRC=430A


; Hexdump des Blocktransport - und Löschprogrammes
*
*
*
*D 8D00,8FFF
8D00  21 55 8F CD 1E 00 21 A6 8F CD 1E 00 CD 3A 8D CD !U.M..!&.M..M:.M
8D10  50 8D CD 66 8D CD EF 8D CD 2E 8E 21 00 88 22 DF P.Mf.Mo.M..!..."_
8D20  8F 21 00 80 22 E1 8F CD CC 8E 21 00 22 22 DF 8F .!.."a.ML.!.""_.
8D30  21 00 80 22 E1 8F CD CC 8E C9 21 6A 8F CD 1E 00 !.."a.ML.I!j.M..
8D40  21 96 00 22 C8 8F CD 7C 8D 21 D3 8F CD BB 8D C9 !.."H.Mö.!S.M;.I
8D50  21 7E 8F CD 1E 00 21 64 00 22 C8 8F CD 7C 8D 21 !ß.M..!d."H.Mö.!
8D60  D5 8F CD BB 8D C9 21 92 8F CD 1E 00 21 32 00 22 U.M;.I!..M..!2."
8D70  C8 8F CD 7C 8D 21 D7 8F CD BB 8D C9 21 00 00 22 H.Mö.!W.M;.I!.."
8D80  CE 8F 22 D0 8F 21 C6 8F 0E 01 CD 21 00 3A CE 8F N."P.!F...M!.:N.
8D90  CB AF FE 53 CA EC 8D 3A CD 8F FE 04 C2 7C 8D FD K/ßSJl.:M.ß.Bö.ü
8DA0  21 CE 8F 3E 00 CD F8 00 D3 70 FD 7E 00 FE 00 CB !N.>.Mx.Spüß.ß.H
8DB0  CD 07 00 FD 23 DA 7C 8D D2 AA 8D FD 21 CE 8F FD M..ü#Zö.R*.ü!N.ü
8DC0  7E 02 FE 40 D4 E9 8D ED 6F FD 7E 03 FE 40 D4 E9 ß.ß§Ti.moüß.ß§Ti
8DD0  8D ED 6F 23 FD 7E 00 FE 40 D4 E9 8D ED 6F FD 7E .mo#üß.ß§Ti.moüß
8DE0  01 FE 40 D4 E9 8D ED 6F C9 C6 09 C9 F1 F1 C9 2A .ß§Ti.moIF.Iqq1*
8DF0  D5 8F ED 5B D3 8F 37 3F ED 52 E5 7D C6 01 6F 7C U.mÄS.7?mReüF.oö
8E00  CE 00 67 22 D9 8F 2A D7 8F 7C 92 FA 25 8E C2 1A N.g"Y.*W.ö.z%.B.
8E10  8E 7D 93 FA 25 8E C2 1A 8E C9 C1 09 EB 09 ED 4B .ü.z%.B..IA.k.mK
8E20  D9 8F ED B8 C9 ED 4B D9 8F EB ED B0 C1 C9 2A D7 Y.m8ImKY.km0AI*W
8E30  8F ED 5B D3 8F A7 ED 52 22 DB 8F FD 21 C9 81 FD .mÄS.'mR"Ä.ü!I.ü
8E40  7E 00 FE 00 C8 CB 7F C2 4F 8E FD 23 C3 3F 8E FD ß.ß.HK.BO.ü#C?.ü
8E50  6E 01 FD 66 02 22 D0 8F CD 78 8E DA 67 8E FD 23 n.üf."ü.Mx.Zg.ü#
8E60  FD 23 FD 23 C3 3F 8E 2A DD 8F ED 5B DB 8F 19 FD ü#ü#C?.*ü.mÄÄ..ü
8E70  75 01 FD 74 02 C3 5E 8E ED 4B D3 8F A7 ED 42 DA u.üt.C^.mKS.'mBZ
8E80  97 8E 2A DD 8F ED 4B D5 8F A7 ED 42 D2 92 8E C3 ..*ü.mKU.'mBR..C
8E90  95 8E C2 97 8E 37 C9 A7 C9 21 B7 8F CD 1E 00 21 ..B..7I'I!7.M..!
8EA0  A6 8F CD 1E 00 CD 3A 8D CD 50 8D 2A D5 8F ED 5B &.M..M:.MP.*U.mÄ
8EB0  D3 8F A7 ED 52 22 D9 8F 2A D3 8F 11 01 00 19 EB S.'mR"Y.*S.....k
8EC0  2A D3 8F ED 4B D9 8F 36 00 ED B0 C9 2A DF 8F 22 *S.mKY.6.m0I*_."
8ED0  D7 8F CD 2A 00 48 06 00 ED 43 D9 8F 79 FE 03 CA W.M*.H..mCY.yß.J
8EE0  E5 8E C3 40 8F 7E E6 F0 FE D0 CA 0C 8F FE E0 CA e.C§.ßfpßPJ..ß`J
8EF0  0C 8F FE F0 CA 0C 8F FE C0 CA FF 8E C3 40 8F 7E ..ßpJ..ß§J..C§.ß
8F00  E6 0F FE 00 CA 26 8F FE 03 CA 26 8F 7E E6 0F FE f.ß.J&.ß.J&.ßf.ß
8F10  0C CA 26 8F FE 04 CA 26 8F FE 0A CA 26 8F FE 02 .J&.ß.J&.ß.J&.ß.
8F20  CA 26 8F C3 40 8F 23 5E 23 56 EB E5 22 DD 8F CD J&.C§.#^#Vke"ü.M
8F30  78 8E E1 D2 40 8F ED 4B DB 8F 09 7C 12 7D 1B 12 x.aR§.mKÄ..ö.ü..
8F40  2A D7 8F ED 4B D9 8F 09 E5 D1 ED 4B E1 8F ED 42 *W.mKY..eQmKa.mB
8F50  D0 EB C3 CF 8E 50 00 D9 00 44 00 42 6C 6F 63 6B PkCO.P.Y.D.Block
8F60  74 72 61 6E 73 70 6F 72 74 00 0A 00 96 00 33 00 transport.....3.
8F70  42 6C 6F 63 6B 61 6E 66 61 6E 67 20 3A 00 0A 00 Blockanfang :...
8F80  64 00 33 00 42 6C 6F 63 6B 65 6E 64 65 20 20 20 d.3.Blockende
8F90  3A 00 0A 00 32 00 33 00 5A 69 65 6C 61 64 72 65 :...2.3.Zieladre
8FA0  73 73 65 20 3A 00 5E 01 14 00 21 00 45 6E 64 65 sse :.^...!.Ende
8FB0  20 3D 20 3C 53 3E 00 80 00 DC 00 44 00 4C 6F 65  = (S)...Ö.D.Loe
8FC0  73 63 68 65 6E 00 0E 01 00 00 33 00 05 00 00 00 schen.....3.....
8FD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
8FE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
8FF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
*
```

Z 80-Vollausbau bis ZEAT

Textverarbeitung mit ZEAT – ZEAT ist mehr als ein Assembler

von Christoph Köhler

Daß man mit dem ZEAT-Assembler-Programm von Fa. Christiani nicht nur Assembler-Programme schreiben kann, sondern daß sich dieser auch hervorragend als Textverarbeitungs-Programm verwenden läßt, beschreibt nachfolgender Beitrag.

Anhand des beigelegten Handbuches ergibt es keine Schwierigkeiten, im EDITOR einen Text zu erstellen (Scroll-Cursor-Blockbefehle . . .). Dieser im EDITOR stehende Text kann von der Kommandoebene (oder A:0) mit dem Befehl TYPE/L ausgedruckt werden.

Jeder Drucker ist in der Lage, verschiedene Schriftarten (Fettdruck, Schmalschrift . . .) auszudrucken. Die Zeichen, die den Drucker dazu veranlassen, nennt man Steuerzeichen.

Der EDITOR (Platz zum Schreiben) hat zwei wichtige Eigenschaften:

1. Man kann den Text auf Kassette/Diskette abspeichern und wieder laden.
2. Hat man bereits Text im EDITOR stehen und möchte einen anderen Text einladen, hängt der EDITOR diesen einfach hinten an.

Diesen Effekt kann man beim Umgang mit den Steuerzeichen verwenden. Z.B.: Man schreibt in den EDITOR nur die Steuerzeichen für Fettdruck und speichert diese unter dem merkbaren Namen „Fett" ab. Wenn man dann einen Brief schreibt, können jederzeit die Steuerzeichen für „Fett"-Schrift eingeladen werden. Der nachfolgende Text wird beim späteren Ausdruck (TYPE/L) als Fettdruck auf dem Papier sein. (Siehe folgendes Beispiel.)

Durch die Kombination der Steuerzeichen (z.B. Fett und gedehnt) können viele verschiedene Schriftarten eingestellt werden.

Für den Umgang mit den Steuerzeichen ist eine kleine Umrechnung notwendig, die zwar kurz erklärt wird, aber durch die Handhabung mit der Tabelle 1 entfällt.

Im Handbuch Ihres Druckers stehen viele Steuerzeichen und ihre Bedeutung, z.B.:
ESC E oder
CHR$(27); "E"
für Fettdruck

Will man nun dem Drucker das Zeichen „ESC" mitteilen, muß man sich zunächst entsprechende ASCII-Zeichen aussuchen. (Siehe Tabelle: ESC = 1b usw. 0001 1011). Die jetzt notwendige Umrechnung besteht darin, das zweithöchste Bit zu setzen. Binär sieht das jetzt so aus: 0101 1011. Dies bedeutet sedezimal 5b und im ASCII-Code: Ä. Durch die Umrechnung werden allen Steuerzeichen zu „großen" Buchstaben zwischen A und Z (Ausnahme: $ ä, ö ü, ^, ...).

Statt der Umrechnung kann der Wert direkt aus der Tabelle entnommen werden, indem ein Lineal an dem Steuerzeichen angelegt und der gesuchte Wert in der Spalte „Großbuchstaben" abgelesen wird.

In Ihrem Handbuch steht der BASIC-Befehl CHR$(27) für „ESC", wobei die dezimal 27 sedezimal 1b und somit in der Tabelle zu finden ist. (Wird später noch benötigt.)

Nun zum ersten Beispiel: Es werden hier Steuerzeichen beschrieben, welche für EPSON-Drucker (MX 80, RX 80, FX 80 . . .) stimmen, aber bei anderen Druckern abweichen können (siehe eigenes Handbuch).

Z.B.: Im Handbuch steht SO für Breitschrift aus der Tabelle entnehmen Sie den Buchstaben N. Für Steuerzeichen ESC entnehmen Sie den Buchstaben Ä.

Da im EDITOR laut ZEAT-Handbuch die Druckersteuerung mit den Tasten ESC und P (nacheinander) eingeleiten werden kann und die Anweisung an den Drucker ESC SO (Ä N) lauten muß, ist folgende Eingabe notwendig: ESC P Ä ESC P N.
Wichtig: ESC = Taste "ESC", RET = Taste "RET" bzw. "CR".

Vorgangsbeschreibung: (Ausgangslage: bzw. A:0)
EDIT RET
normale Schrift RET
ESC P Ä ESC P N Fettdruck RET
wieder normal RET
ESC E
TYPE/L

```
1. Beispiel: Breitschrift (Gilt nur für die jeweilige Zeile)
   Breitschrift (ST=Steuerzeichen=SO)
Eingabe: ESC P Ä   ESC P N   TEXT
  (entspricht "ESC" "SO" (Shift out))

2. Beispiel: Schmalschrift
 Schmalschrift (ST=SI)
Eingabe: ESC P Ä   ESC P O   TEXT
  (entspricht "ESC" "SI" (Shift in))
Diese Schrift muss wieder abgeschaltet werden

3. Beispiel: Schmalschrift abschalten
 Schmalschrift Aus (ST=DC2)
Eingabe: ESC P Ä   ESC P R   TEXT
  (entspricht "ESC" "DC2")

4. Beispiel: Schmalschrift und 7/72 " Zeilenabstand
 Schmal und eng (ST=ESC 1)
        Schmal und eng
        Schmal und eng
Eingabe: ESC P Ä   ESC P O   ESC P Ä   ESC P 1
   (Entspricht "ESC" "O" "ESC" "1"
```

```
5. Beispiel: Wieder normal ( Normieren )
   Damit kann der Drucker aus allen Schriftarten auf
   "normal" umgeschaltet werden.

Eingabe: ESC P Ä §
    (Entspricht "ESC" "§"

Achtung, wenn im Handbuch z.B.   ESC § steht, heißt die
Anweisung nicht ESC P Ä  ESC P § sondern ESC P Ä §.

6. Beispiel: Fettdruck

Dies ist Fettdruck (ST=ESC E)
Eingabe: ESC P Ä E
 Fettdruck aus (ST=ESC F)
Eingabe: ESC P Ä F

7. Beispiel: gedehnte Schrift

Auch hier ist was neues zu beachten. Als Basic-Befehl
steht im Handbuch: CHR$(27); "W" ;CHR$(n);
Das heißt ESC W mit Faktor (n)
          weiter steht n = 1     gedehnte Schrift
                       n = 0     Normalschrift
Aus der Tabelle ermittelt man für 1= sedezimal 1 = A
und 0 = §. (Siehe oben)
```

```
Gedehnte Schrift (ST=ESC W (n))
Eingabe: ESC P A W ESC P A
Wieder AUS

Eingabe: ESC P A W ESC P §


B. Beispiel: Fett und gedehnt

Fett und gedehnt ist möglich
Eingabe: ESC P A E ESC P A W ESC P A

Wieder normal
Eingabe: ESC P A F  ESC P A ESC P §


Letztes Beispiel: Piepton am Schluss des Druckens


Eingabe: ESC P G
   (entspricht "ESC" "BEL")
```

Technisches Lehrinstitut
Dr.-Ing. P. Christiani

7-Bit-Code ASCII

Steuerzeichen		Schriftzeichen		Schriftzeichen		Schriftzeichen	
Zeichen	Sedez.	Zeichen	Sedez.	Zeichen	Sedez.	Zeichen	Sedez.
NUL	00	SP	20	@	40	\	60
SOH	01	!	21	A	41	a	61
STX	02	"	22	B	42	b	62
ETX	03	#	23	C	43	c	63

EOT	04	$	24	D	44	d	64
ENQ	05	%	25	E	45	e	65
ACK	06	&	26	F	46	f	66
BEL	07	'	27	G	47	g	67
BS	08	(	28	H	48	h	68
HT	09	)	29	I	49	i	69
LF	0A	*	2A	J	4A	j	6A
VT	0B	+	2B	K	4B	k	6B
FF	0C	,	2C	L	4C	l	6C
CR	0D	-	2D	M	4D	m	6D
SO	0E	.	2E	N	4E	n	6E
SI	0F	/	2F	O	4F	o	6F
DLE	10	0	30	P	50	p	70
DC_1	11	1	31	Q	51	q	71
DC_2	12	2	32	R	52	r	72
DC_3	13	3	33	S	53	s	73
DC_4	14	4	34	T	54	t	74
NAK	15	5	35	U	55	u	75
SYN	16	6	36	V	56	v	76
ETB	17	7	37	W	57	w	77
CAN	18	8	38	X	58	x	78
EM	19	9	39	Y	59	y	79
SUB	1A	:	3A	Z	5A	z	7A
ESC	1B	;	3B	[Ä	5B	{ ä	7B
FS	1C	<	3C	\ Ö	5C	| ö	7C
GS	1D	=	3D	] Ü	5D	} ü	7D
RS	1E	>	3E	↑	5E	~ ß	7E
US	1F	?	3F	_	5F	DEL	7F

Es ist zu empfehlen, sich mit dem ZEAT-Programm eine reine Textverarbeitungs-Diskette/Kassette zuzulegen, in der die Druckeranweisungen mit gut merkbaren Namen abgelegt sind (z.B. FETT-EIN für „Fettdruck einschalten"). Dies ermöglicht eine reibungslose Textverarbeitung mit dem Einsatz der Schriftarten, ohne dem lästigen Suchen nach den richtigen Steuerzeichen in der Tabelle.

Z80-CP/M2.2

RAM-Floppy automatisch gefunden

BIOS-Eintrag für CP/M2.2
von Eckhard Zarling, Heeper Straße 362,
4800 Bielefeld 17, Tel.: (0521) 332849

```
.z80
banken  equ     0f284h          ; Systemroutine transportiert 128 Byte
size    equ     0e9c0h          ; Speicherzelle Diskettenkapazitaet
bdos    equ     0005            ; Systemaufruf
cls     equ     1ah             ; Bildschirm loeschen
cr      equ     0dh             ; carriage return
lf      equ     0ah             ; line feed
tab     equ     09h             ; Ttabulator

start:  ld      e,cls
        ld      c,2
        call    bdos            ; Bildschirm loeschen
        ld      de,text1
        ld      c,9
        call    bdos            ; Startmeldung ausgeben
        ld      hl,480d         ; 480 Bloecke mit 128 Byte / 62 k Bank
        ld      b,01            ; Ziel ist Bank 1
        ld      c,00            ; Quelle ist Bank 0
        ld      a,0
        ld      (size),a        ; keine zusaetzliche RDA
        call    format          ; Bank formatieren
        cp      0               ; wurde Bank formatiert ?
        jp      nz,exit         ; --> Bank nicht vorhanden
        ld      a,62d           ; 1. Bank ist formatiert
        ld      (size),a        ; Diskettenkapazitaet eintragen ( 62 k )
        ld      hl,480d         ; 480 Bloecke mit 128 Byte / 62 k Bank
        ld      b,02            ; Ziel ist Bank 2
        ld      c,0             ; Quelle ist Bank 0
        call    format          ; Bank formatieren
        cp      0               ; wurde Bank formatiert
        jp      nz,exit         ; --> Bank nicht vorhanden ?
        ld      a,7eh           ; 2. Bank formatiert
        ld      (size),a        ; Diskettenkapazitaet eintragen ( 124 k )
        ld      hl,480d         ; 480 Bloecke mit 128 Byte / 62 k Bank
        ld      b,03            ; Ziel ist Bank 3
        ld      c,0             ; Quelle ist Bank 0
        call    format          ; Bank formattieren
        cp      0               ; wurde Bank formatiert ?
        jp      nz,exit         ; --> Bank nicht vorhanden
        ld      a,0b3h          ; 3. Bank formatiert
        ld      (size),a        ; Diskettenkapazitaet eintragen ( 178 k )
exit:   ld      a,(size)        ; Diskettenkapazitaet bestimmen
        ld      de,text2
        cp      0
        jp      z,print1        ; Msg. keine zusaetzliche RAM - karte
        ld      de,text3
        cp      62d             ; 1. Bank
        jp      z,print
        ld      de,text4        ; 2. Bank
        cp      7eh
        jp      z,print
        ld      de,text5        ; 3. Bank
print:  ld      c,9
        call    bdos            ; Meldung ausgeben
        ld      de,texte
print1: ld      c,9
        call    bdos            ; Endemeldung ausgeben
        ld      c,0
        call    bdos            ; Init. Warmstart

;-------------------- U n t e r p r o g r a m m e ----------------------

format: ld      de,0000h        ; ab Adr. 0000 auf Bank 1
formal: push    hl              ; Anzahl 128 Byte Bloecke merken
        ld      hl,sektor       ; Adr. Fuellbytes -> HL
        push    de              ; Zieladresse merken
        push    bc              ; akt. Bank merken
        ld      a,0
        call    banken          ; Block formatieren
        pop     bc
        jp      c,error         ; --> Bank nicht vorhanden
        pop     ix              ; copiert DE -> IX
        push    bc              ; Bank merken
        ld      bc,128d         ; naechster 128 Byte Block
        add     ix,bc
        pop     bc
        push    ix
        pop     de              ; copiert IX -> DE
        pop     hl              ; Anzahl Bloecke vom Stack holen
        dec     hl
        ld      a,h
        or      l
        jp      nz,formal       ; naechsten Block formatieren
        ld      a,0             ; Fehlerflag loeschen
        ret                     ; --> Hauptprogramm
error:  ld      a,0ffh          ; Fehlerflag setzen ( keine RDA )
        pop     de
        pop     de              ; Dummy POP's (Stackbereinigung)
        ret                     ; --> Hauptprogramm
```

```
;-------------------------- T e x t e ----------------------------------

text1:  db      cr,lf,lf,lf,tab
        db      'RAM - Floppy wird formatiert ; Bitte warten ...$'

text2:  db      cr,lf,lf,tab
        db      'F e h l e r , keine zusaetzliche RAM - Karte vorhanden !$'

text3:  db      cr,lf,lf,tab
        db      '62 k RAM - Floppy formatiert !  .$'

text4:  db      cr,lf,lf,tab
        db      '126 k RAM - Floppy formatiert !  .$'

text5:  db      cr,lf,lf,tab
        db      '178 k RAM - Floppy formatiert !  .$'

texte:  db      cr,lf,lf,tab
        db      'Laufwerk E: --> RAM - Floppy  .$'

sektor: ds      128,0e5h

        end
```

Z80-CP/M2.2

von
Christoph Köhler

EGRUND, EGOSI, EBASIC, EZASS und ESPS mit Disketten verwalten

Durch die oft gestellte Frage „Wie kann man ein Eprom von der ROA 64 (siehe oben) unter CP/M auf Diskette abspeichern bzw. wieder in den Arbeitsspeicher einlesen?“, ist nachfolgender Beitrag entstanden.

Es wurde darauf geachtet, daß der Vorschlag ohne allzuviel Aufwand und vor allem für alle CP/M-Anwender nachvollziehbar bleibt.

Ziel der folgenden Zeilen ist, Programme wie EBASIC oder EGOSI unter CP/M aufzurufen und die damit selbst erstellten Anwenderprogramme auf Diskette abzuspeichern bzw. wieder einzulesen.

Die Lösung des Problems ist eigentlich recht einfach: Man schafft im Arbeitsspeicher von CP/M (TPA) die gleichen Verhältnisse wie auf der ROA 64 mit den EPROMS.

Bild 1 zeigt die Benützung der TPA am Beispiel der Verwendung des EBASIC2.

```
0000h - 00FFh     Reserviert für CP/M

0100h             Sprung nach 4000h

4000h - 5FFFh     EBASIC im RAM - Bereich

8000h - 9FFFh     Platz für Anwenderprogramme
                  im RAM - Bereich
```

Bild 1:

Vorgangsbeschreibung:

Im Prinzip wird ein EPROM (z.B. EBASIC2) von der ROA 64K in die TPA auf die jeweils festgelegte Adresse (bei EBASIC = 4000h) kopiert, dann von Adresse 0100h ein direkter Sprung zu dem Programm geschrieben und schließlich das ganze (0100h – 5FFFh) unter der Bezeichnung “EBASIC.COM“ abgespeichert.

Das EBASIC2 muß auf Adresse 4000h kopiert werden, da es nur auf diesem Platz laufen kann, um den Arbeitsspeicher muß man sich nicht kümmern, da dieser ohnehin auf Adresse 8000h zur Verfügung steht.

Ja richtig, man kann von Speicherverschwendung reden, aber bei einem Platz von ca. 800 KByte auf einer Diskette spielt die Inanspruchnahme des EBASIC.-COM-Files von ca. 24 KByte nur eine nebensächliche Rolle.

Ruft man das Programm EBASIC auf, meldet es sich wie gewohnt und man kann die alten Programme mit dem Kassettenrecorder einlesen.

Beim Konservieren des jeweiligen Anwenderprogrammes könnte man die TPA von 0100h bis 9FFFh (Sprung zum EPROM + EPROM + Anw. Prog) unter einem Namen abspeichern, aber man wird zu dieser großzügigen Speicherhandhabung (ca. 40 KByte) nur in eiligen Fällen greifen.

Im Normalfall kann man mit dem DDT den Bereich von 8000h bis 9FFFh nach 0100h verschieben und dann abspeichern, um alle Anwenderprogramme auf einer Länge von 8 KByte zu behalten.

Schließlich bietet sich an, ein Programm zum Speichern der eigenen Anwenderprogramme sowie eines zum Laden der selbigen zu schreiben.

Die notwendigen Maßnahmen teilen sich in 3 Gruppen:

1. Kopieren des EPROMS in die TPA
2. Abspeichern des Programmes auf Diskette
3. Laden und Abspeichern von Anwenderprogrammen

1. EPROM-Software in TPA

Es bieten sich verschiedene Möglichkeiten an:

1.1 EPROM mit PROMER in TPA einlesen (z.B. mit ZEAT)

1.2 Bei Verwendung von statischen Speichern kann das EPROM direkt auf den richtigen Platz gesteckt werden.

1.3 Anwender von ZEAT können die Transport-Routine aus der *LOOP 6,* Seite 12 verwenden.

Achtung: Bei BASIC2 ist AQ=4000h, EA=5FFFh AZ=4000h, NQ=0Eh, NZ=00h

1.4 Nur CP/M-Betreiber müssen die nachfolgende Routine im DDT mit dem S-Befehl eingeben (CP/M verwendet leider keine Z80-Befehle)

```
auf Bildschirm    Eingabe          Kommentar
-------------------------------------------------------------
A>                DDT <RET>        Aufruf DDT
-                 S100 <RET>       Andern ab 100h
0100 -            21 <RET>
0101 -            00 <RET>         So die ganze Routine
                                   eingeben
                                   Achtung: Kein Byte darf
                                   mit einem Buchstaben be-
                                   ginnen, d.h. z.B. statt
                                   FF ---> 0FF eigeben.
011E -            00 <RET>
011F -            .                Ende der Anderung
-                 <CTRL> C         gleichzeitig
```

Bild 2:

2. Abspeichern des Programmes auf Diskette

```
A>          EBASIC
?           C                      Kaltstart wie gewohnt

  Nachdem ein Programm mit Recorder eingelesen oder selbst ges

  ist, kann man das BASIC (außer mit einem Reset) folgendermaß
  verlassen:

>           CLRS                   Bildschirm löschen
>           CALL 0                 EXIT
A>


Mit dem DDT wird das Programm abgespeichert:

A>          DDT
-           M8000,9FFF,100         nach 100h verschieben
-           CTRL C <RET>
A>          SAVE 33 TEST1.BAS      Unter einem beliebigen
                                   Namen abspeichern
```

2.1 Ändern und Abspeichern. Wenn das EPROM in die TPA kopiert ist, muß noch der Sprung nach Adresse 4000h eingetragen werden. Dies geschieht ebenfalls mit dem DDT.

3. Abspeichern und Einlesen von eigenen Programmen

Jetzt ist man in der Lage, das Programm mit aufzurufen

```
A>          DDT EBASIC.COM         Beides gleichzeitig aufrufe
-           M100,2100,4000         EBASIC nach 4000h verschieb
-           S100                   Sprung eingeben
0100 -      0C3                    JP 4000h
0101 -      00
0102 -      40
0103 -      .
-           CTRL C


A>          SAVE 95 EBASIC.COM              Warum 95 ?
                                   Letzte Adresse = 5FFFh,
                                   Entscheidend ist das höher-
                                   wertige Byte. Hier 5Fh = 95
```

```
Das Einlesen eines Programmes von der Diskette geschieht
folgendermaßend:

A>          DDT TEST1.BAS          beides aufrufen, TEST1.BAS
                                   liegt auf Adresse 0100h
-           M100,20ff,8000         nach 8000h verschieben
-           CTRL C
A>

Jetzt kann EBASIC aufgerufen werden:

A>          EBASIC
?           W                      Wichtig: Warmstart
>           list                   o.k. ?
```

Ganz ähnlich verläuft der Vorgang bei dem Einsatz der anderen EPROMS. (EGOSI2, EZASS2, EGRUND2000, ESPS2000 ...). Die Verwendung der oben beschriebenen Speich-/Lade-Programme erspart jeglichen Umgang mit dem DDT und wirkt somit wesentlich komfortabler.

```
Speichern von Programmen auf Adresse 8000H                    PAGE 1


                ;*****************************************************************
                ;Hauptprogramm  SPEICHERN
                ;Automatische Abspeicherung von Anwenderprogrammen
                ;der EPROM-Software EGRUND2, EGOSI2, EBASIC2, EIASS2
                ;Speich.asm
                ;Christoph Kähler - 20/05/86
                ;*****************************************************************


                SYSTEM  EQU     0005H           ;Systemfunktionen


                WSTARTF EQU     0               ;Warmstart

                DIRCONF EQU     6               ;Direkte Konsole Ein/Ausgabe

                STRAUSF EQU     9               ;Ausgabe von Strings auf Konsole

                STREINF EQU     10              ;Einlesen von Strings in Puffer

                OPENF   EQU     15              ;Datei öffnen

                CLOSEF  EQU     16              ;Datei schließen

                DELETEF EQU     19              ;Datei löschen

                READF   EQU     20              ;aus Datei lesen

                WRITEF  EQU     21              ;in Datei schreiben

                MAKEF   EQU     22              ;Datei anlegen

                RENAMEF EQU     23              ;Datei umbenennen

                SETDMAF EQU     26              ;DMA-Adresse setzen

                MAKEFCB EQU     250             ;FCB - Block kreieren


                CR      EQU     0DH             ;Cursor an Zeilenanfang

                LF      EQU     0AH             ;Cursor in neue Zeile

                EOF     EQU     1AH             ;Markierung für Textende



                KOM     EQU     0               ;Kommentar: Bit 0
                STR     EQU     1               ;String:    Bit 1

                        TITLE Speichern von Programmen auf Adresse 8000H

                        ORG     100H

                START:
0100  C3 BA 01                  JP      START1

Speichern von Programmen auf Adresse 8000H                    PAGE 2

0103  4E 61 6D 65 20 64 65 72  FRAGE:          DB      'Name der Datei zum Abspeichern ? $'
0125  49 6E 68 61 6C 74 73 76  FEHLER1:        DB      'Inhaltsverzeichniss voll',CR,LF,'$'
0140  44 69 73 6B 65 74 74 65  FEHLER2:        DB      'Diskette voll !',CR,LF,'$'


0152  00 00                    SATZ_ZAHL:      DW      0

0154  10 00                    PUFFER:         DB      16,0
                                              DS      16

0166  00 00 00 00 00 00 00 00  AUSGABE_FCB:     DS      36,0


                START1:
018A  11 03 01                  LD      DE,FRAGE        ;Name ?
018D  0E 09                     LD      C,STRAUSF
018F  CD 05 00                  CALL    SYSTEM

0192  11 54 01                  LD      DE,PUFFER       ;Name einlesen
0195  0E 0A                     LD      C,STREINF
0197  CD 05 00                  CALL    SYSTEM


019A  11 56 01                  LD      DE,PUFFER+2     ;FCB - erzeugen
019D  FD 21 54 01               LD      IY,PUFFER
01A1  DD 21 66 01               LD      IX,AUSGABE_FCB
01A5  CD 26 02                  CALL    MAKE_FCB
01A8  D0                        RET     NC

01A9  11 66 01                  LD      DE,AUSGABE_FCB  ;Datei schon vorhanden ?
01AC  0E 0F                     LD      C,OPENF
01AE  CD 05 00                  CALL    SYSTEM
01B1  FE FF                     CP      0FFH
01B3  28 0B                     JR      Z,MAKE          ;Wenn nicht: ----> Make it
01B5  11 66 01                  LD      DE,AUSGABE_FCB  ;Wenn ja: Delete it
01B8  0E 13                     LD      C,DELETEF
01BA  CD 05 00                  CALL    SYSTEM
01BD  C4 E7 01                  CALL    NZ,WEITER       ;falls Datei vorhanden,
                                                        ; ----> weiter
                MAKE:
01C0  11 66 01                  LD      DE,AUSGABE_FCB  ;Nein, neu kreieren
01C3  0E 16                     LD      C,MAKEF
01C5  CD 05 00                  CALL    SYSTEM
01C8  FE FF                     CP      0FFH
01CA  20 0B                     JR      NZ,WEITER1
01CC  11 25 01                  LD      DE,FEHLER1      ;Inhaltsverz. voll
01CF  0E 09                     LD      C,STRAUSF
01D1  CD 05 00                  CALL    SYSTEM
01D4  C3 00 00                  JP      0000H


                WEITER1:
01D7  11 66 01                  LD      DE,AUSGABE_FCB  ;und dann öffnen
01DA  0E 0F                     LD      C,OPENF
01DC  CD 05 00                  CALL    SYSTEM
01DF  FE FF                     CP      0FFH
01E1  C4 E7 01                  CALL    NZ,WEITER       ;falls Datei vorhanden,
01E4  C3 00 00                  JP      0000H

                WEITER:
01E7  11 00 80                  LD      DE,8000H

Speichern von Programmen auf Adresse 8000H                    PAGE 3

01EA  01 40 00                  LD      BC,64

                SCHREIB_SCHLEIFE:
01ED  78                        LD      A,B
01EE  B1                        OR      C               ;BC = 0 --> Ende
01EF  28 2C                     JR      Z,SCHREIB_ENDE

01F1  21 80 00                  LD      HL,128          ;DE+128 = DE
01F4  19                        ADD     HL,DE
01F5  E5                        PUSH    HL

01F6  0B                        DEC     BC              ;BC retten (Zähler)
01F7  C5                        PUSH    BC

01F8  0E 1A                     LD      C,SETDMAF       ;in DE ist Speicheradresse
01FA  CD 05 00                  CALL    SYSTEM

01FD  11 66 01                  LD      DE,AUSGABE_FCB  ;Speichern
0200  0E 15                     LD      C,WRITEF
0202  CD 05 00                  CALL    SYSTEM

0205  C1                        POP     BC              ;Zähler restaurieren
0206  D1                        POP     DE
0207  A7                        AND     A               ;prüfen ob Speicherung o.k.
0208  28 E3                     JR      Z,SCHREIB_SCHLEIFE ;Weiter bis voll

020A  11 66 01                  LD      DE,AUSGABE_FCB
020D  0E 13                     LD      C,DELETEF
020F  CD 05 00                  CALL    SYSTEM

0212  11 40 01                  LD      DE,FEHLER2
0215  0E 09                     LD      C,STRAUSF
0217  CD 05 00                  CALL    SYSTEM

021A  C3 71 02                  JP      ENDE

                ;*****************************************************************
                ; Unterprogram DATEI_SCHLIEßEN                                   *
                ;*****************************************************************

                SCHREIB_ENDE:
021D  11 66 01                  LD      DE,AUSGABE_FCB  ;Datei schließen
0220  0E 10                     LD      C,CLOSEF
0222  CD 05 00                  CALL    SYSTEM
0225  C9                        RET


                ;*****************************************************************
                ; Unterprogram MAKE_FCB                                          *
                ;*****************************************************************

                MAKE_FCB:
0226  FD 7E 01                  LD      A,(IY+1)        ;Pufferlänge laden
0229  A7                        AND     A
022A  C8                        RET     Z               ;Länge = 0 --> Fehler
022B  4F                        LD      C,A

022C  DD 36 00 00               LD      (IX),0          ;Laufwerk = A:

0230  DD 23                     INC     IX
0232  0C                        INC     C
0233  06 0B                     LD      B,11            ;Länge Dateinahme im FCB

Speichern von Programmen auf Adresse 8000H                    PAGE 4

                MAKEFCB_SCHL:
0235  0D                        DEC     C               ;Wiederholen bis Puffer
0236  28 1D                     JR      Z,MAKEFCB_FUELL ;leer
0238  1A                        LD      A,(DE)
0239  FE 2E                     CP      '.'             ;Punkt ?
023B  28 25                     JR      Z,MAKEFCB_TYP   ;Typ fängt an
023D  FE 61                     CP      'a'             ;Kleinbuchstabe
023F  38 02                     JR      C,MAKEFCB_GROSS ;nein ---> wandeln
0241  D6 20                     SUB     20H
                MAKEFCB_GROSS:
0243  FE 21                     CP      ' '+1           ;Steuerzeichen ?
0245  38 0E                     JR      C,MAKEFCB_FUELL ;Rest auffüllen
0247  FE 3D                     CP      '='             ; = ?
0249  28 0A                     JR      Z,MAKEFCB_FUELL ;ja, füllen
024B  13                        INC     DE              ;Pufferzeiger weiter
024C  DD 77 00                  LD      (IX),A          ;Zeichen in FCB
024F  DD 23                     INC     IX              ;FCB-Zeiger weiter
0251  10 E2                     DJNZ    MAKEFCB_SCHL    ;Schleife bis Zähler=0

0253  37                        SCF                     ;kein Fehler: Carry=1
0254  C9                        RET

                MAKEFCB_FUELL:
0255  78                        LD      A,B             ;Zähler schon 0 ?
0256  B7                        OR      A
0257  37                        SCF
0258  C8                        RET     Z               ;ja: fertig
0259  DD 36 00 20               LD      (IX),' '        ;nein: ZWR in FCB
```

```
025D  DD 23                   INC   IX                ;Zeiger inc.
025F  05                      DEC   B                  ;Zähler dec.
0260  18 F3                   JR    MAKEFCB_FUELL

                      MAKEFCB_TYP:
0262  13                      INC   DE                 ;Pufferzeiger weiter
                      MAKEFCB_TYPSCHL:
0263  78                      LD    A,B
0264  FE 04                   CP    4                  ;Zähler < 4
0266  38 CD                   JR    C,MAKEFCB_SCHL     ;ja: Typ eintragen

0268  DD 36 00 20             LD    (IX),' '           ;nein: auffüllen bis
026C  DD 23                   INC   IX                 ;zu Dateityp
026E  05                      DEC   B
026F  18 F2                   JR    MAKEFCB_TYPSCHL


                      ENDE:
                              END   START
```

Speichern von Programmen auf Adresse B000H PAGE 5

```
0005 SYSTEM                   0000 WSTARTF
0006 DIRCONF                  0009 STRAUSF
000A STREINF                  000F OPENF
0010 CLOSEF                   0013 DELETEF
0014 READF                    0015 WRITEF
0016 MAKEF                    0017 RENAMEF
001A SETDMAF                  00FA MAKEFCB
000D CR                       000A LF
001A EOF                      0000 KDM
0001 STR                      0100 START
0103 FRAGE                    0125 FEHLER1
0140 FEHLER2                  0152 SATZ_ZAHL
0154 PUFFER                   0166 AUSGABE_FCB
018A START1                   01C0 MAKE
01D7 WEITER1                  01E7 WEITER
01ED SCHREIB_SCHLEIFE         021D SCHREIB_ENDE
0226 MAKE_FCB                 0235 MAKEFCB_SCHL
0243 MAKEFCB_GROSS            0255 MAKEFCB_FUELL
0262 MAKEFCB_TYP              0263 MAKEFCB_TYPSCHL
0271 ENDE

no fatal error(s)

SUM=629F
CRC=AA3C
```

LADEN VON B000H PAGE 1

```
                      ;*****************************************************************
                      ;Hauptprogramm  Laden
                      ;Automatische Laden von Anwenderprogrammen für
                      ;EPROM-Software EGRUND2, EBOSI2, EBASIC2, EZASS2
                      ;Lade.ASM
                      ;Christoph Köhler - 21/06/86
                      ;*****************************************************************

                      SYSTEM  EQU   0005H        ;Systemfunktionen


                      WSTARTF EQU   0            ;Warmstart

                      DIRCONF EQU   6            ;Direkte Konsole Ein/Ausgabe

                      STRAUSF EQU   9            ;Ausgabe von Strings auf Konsole

                      STREINF EQU   10           ;Einlesen von Strings in Puffer

                      OPENF   EQU   15           ;Datei öffnen

                      CLOSEF  EQU   16           ;Datei schließen

                      DELETEF EQU   19           ;Datei löschen

                      READF   EQU   20           ;aus Datei lesen

                      WRITEF  EQU   21           ;in Datei schreiben

                      MAKEF   EQU   22           ;Datei anlegen

                      RENAMEF EQU   23           ;Datei umbenennen

                      SETDMAF EQU   26           ;DMA-Adresse setzen

                      MAKEFCB EQU   250          ;FCB - Block kreieren


                      CR      EQU   0DH          ;Cursor an Zeilenanfang

                      LF      EQU   0AH          ;Cursor in neue Zeile

                      EOF     EQU   1AH          ;Markierung für Textende



                              TITLE LADEN VON B000H

                              ORG   100H

                      START:
0100  C3 9B 01                JP    START1
```

```
0103  4E 61 6D 65 20 64 65 72  FRAGEF:      DB    'Name der Datei zum Einlesen ?',CR,LF,'$'
0127  0D 0A 44 61 74 65 69 20  FEHLER1:     DB    CR,LF,'Datei nicht gefunden !',CR,LF,'$'
013E  0D 0A 53 70 65 69 63 68  FEHLER2:     DB    CR,LF,'Speicher voll !',CR,LF,'$'

LADEN VON B000H                                                    PAGE 2

0152  0D 0A 4C 65 73 65 66 65  FEHLER3:     DB    CR,LF,'Lesefehler !',CR,LF,'$'

0167  00 00                   SATZ_ZAHL:    DW    0

0165  10 00                   PUFFER:       DB    16,0
                                            DS    16

0177  00 00 00 00 00 00 00 00  EINGABE_FCB:  DS    36,0


                      START1:
019B  11 03 01                LD    DE,FRAGE           ;Name ?
019E  0E 09                   LD    C,STRAUSF
01A0  CD 05 00                CALL  SYSTEM

01A3  11 65 01                LD    DE,PUFFER          ;Name einlesen
01A6  0E 0A                   LD    C,STREINF
01A8  CD 05 00                CALL  SYSTEM


01AB  11 67 01                LD    DE,PUFFER+2        ;FCB - erzeugen
01AE  FD 21 65 01             LD    IY,PUFFER
01B2  DD 21 77 01             LD    IX,EINGABE_FCB
01B6  CD 19 02                CALL  MAKE_FCB
01B9  D0                      RET   NC

01BA  11 77 01                LD    DE,EINGABE_FCB     ;Datei vorhanden ?
01BD  0E 0F                   LD    C,OPENF
01BF  CD 05 00                CALL  SYSTEM
01C2  FE FF                   CP    0FFH
01C4  20 0A                   JR    NZ,WEITER

01C6  11 23 01                LD    DE,FEHLER1         ;Datei nicht gefunden
01C9  0E 09                   LD    C,STRAUSF
01CB  CD 05 00                CALL  SYSTEM
01CE  37                      SCF
01CF  C9                      RET


                      WEITER:                          ;Ja: also einlesen
01D0  11 00 B0                LD    DE,B000H           ;Adresse zum Einlesen
01D3  01 FF FF                LD    BC,-1              ;Zähler = -1

                      LESE_SCHLEIFE:
01D6  2A 06 00                LD    HL,(SYSTEM+1)      ;Test ob Speicher voll
01D9  2B                      DEC   HL                 ;Anmerk.: Die Adresse
01DA  ED 52                   SBC   HL,DE              ;System+1 RAM-Ende
01DC  30 0A                   JR    NC,LESE_SCHLEIFE1

01DE  11 3E 01                LD    DE,FEHLER2         ;Text: Speicher voll
01E1  0E 09                   LD    C,STRAUSF
01E3  CD 05 00                CALL  SYSTEM
01E6  37                      SCF
01E7  C9                      RET

                      LESE_SCHLEIFE1:
01E8  21 80 00                LD    HL,128             ;128 = Versatz für DMA-Adresse
01EB  19                      ADD   HL,DE
01EC  03                      INC   BC
01ED  C5                      PUSH  BC
01EE  E5                      PUSH  HL

LADEN VON B000H                                                    PAGE 3

01EF  0E 1A                   LD    C,SETDMAF          ;DMA - Adresse setzen
01F1  CD 05 00                CALL  SYSTEM

01F4  11 77 01                LD    DE,EINGABE_FCB     ;Record einlesen
01F7  0E 14                   LD    C,READF
01F9  CD 05 00                CALL  SYSTEM             ;A=0 falls alles o.k.

01FC  D1                      POP   DE                 ;Adresse restaurieren
01FD  A7                      AND   A
01FE  28 08                   JR    Z,WEITER1          ;weiter ?

0200  11 52 01                LD    DE,FEHLER3         ;Einlesen Fehler
0203  0E 09                   LD    C,STRAUSF          ;Text: Lese - Fehler
0205  CD 05 00                CALL  SYSTEM

                      WEITER1:
0208  C1                      POP   BC                 ;Zähler restaurieren
0209  79                      LD    A,C
020A  FE 3F                   CP    63                 ;Prüfen ob 8k im Speicher

020C  20 C8                   JR    NZ,LESE_SCHLEIFE

020E  11 77 01                LD    DE,EINGABE_FCB     ;Record schließen
0211  0E 10                   LD    C,CLOSEF
0213  CD 05 00                CALL  SYSTEM

0216  C3 64 02                JP    WEITER3

                      ;*****************************************************************
                      ; Unterprogramm MAKE_FCB                                        *
                      ;*****************************************************************

                      MAKE_FCB:
0219  FD 7E 01                LD    A,(IY+1)           ;Pufferlänge laden
021C  A7                      AND   A
021D  C8                      RET   Z                  ;Länge = 0 --> Fehler
021E  4F                      LD    C,A

021F  DD 36 00 00             LD    (IX),0             ;Laufwerk = A:

0223  DD 23                   INC   IX
0225  0C                      INC   C
```

```
0226  06 0B              LD      B,11            ;Länge Dateinahme in FCB
                      MAKEFCB_SCHL:
0228  0D                DEC     C               ;Wiederholen bis Puffer
0229  28 10              JR      Z,MAKEFCB_FUELL ;leer
022B  1A                LD      A,(DE)
022C  FE 2E              CP      '.'             ;Punkt ?
022E  28 25              JR      Z,MAKEFCB_TYP   ;Typ fängt an
0230  FE 61              CP      'a'             ;Kleinbuchstabe
0232  38 02              JR      C,MAKEFCB_GROSS ;nein ---) wandeln
0234  D6 20              SUB     20H
                      MAKEFCB_GROSS:
0236  FE 21              CP      ' '+1           ;Steuerzeichen ?
0238  38 0E              JR      C,MAKEFCB_FUELL ;Rest auffüllen
023A  FE 3D              CP      '='             ; = ?
023C  28 0A              JR      Z,MAKEFCB_FUELL ;ja, füllen
023E  13                 INC     DE              ;Pufferzeiger weiter
023F  DD 77 00           LD      (IX),A          ;Zeichen in FCB
0242  DD 23              INC     IX              ;FCB-Zeiger weiter
0244  10 E2              DJNZ    MAKEFCB_SCHL    ;Schleife bis Zähler=0

LADEN VON B000H                                  PAGE  4

0246  37                 SCF                     ;kein Fehler: Carry=1
0247  C9                 RET

                      MAKEFCB_FUELL:
0248  78                 LD      A,B             ;Zähler schon 0 ?
0249  B7                 OR      A
024A  37                 SCF
024B  C8                 RET     Z               ;ja: fertig
024C  DD 36 00 20        LD      (IX),' '        ;nein: IWR in FCB
0250  DD 23              INC     IX              ;Zeiger inc.
0252  05                 DEC     B               ;Zähler dec.
0253  18 F3              JR      MAKEFCB_FUELL

                      MAKEFCB_TYP:
0255  13                 INC     DE              ;Pufferzeiger weiter
                      MAKEFCB_TYPSCHL:
0256  78                 LD      A,B
0257  FE 04              CP      4               ;Zähler < 4
0259  38 CD              JR      C,MAKEFCB_SCHL  ;ja: Typ eintragen
025B  DD 36 00 20        LD      (IX),' '        ;nein: auffüllen bis
025F  DD 23              INC     IX              ;zu Dateityp
0261  05                 DEC     B
0262  18 F2              JR      MAKEFCB_TYPSCHL

                      WEITER3:
0264  C3 00 00           JP      0000H

                      ENDE:
                         END     START

LADEN VON B000H                                  PAGE  5

0005 SYSTEM              0000 WSTARTF
0006 DIRCONF             0009 STRAUSF
000A STREINF             000F OPENF
0010 CLOSEF              0013 DELETEF
0014 READF               0015 WRITEF
0016 MAKEF               0017 RENAMEF
001A SETDMAF             00FA MAKEFCB
000D CR                  000A LF
001A EOF                 0100 START
0103 FRAGE               0123 FEHLER1
013E FEHLER2             0152 FEHLER3
0163 SATZ_ZAHL           0165 PUFFER
0177 EINGABE_FCB         019B START1
01D0 WEITER              01D6 LESE_SCHLEIFE
01EB LESE_SCHLEIFE1      0208 WEITER1
0219 MAKE_FCB            0228 MAKEFCB_SCHL
0236 MAKEFCB_GROSS       0248 MAKEFCB_FUELL
0255 MAKEFCB_TYP         0256 MAKEFCB_TYPSCHL
0264 WEITER3             0267 ENDE

no fatal error(s)

SUM=5CD5
CRC=4FJ2
```

PASCAL und BASIC

Tips und Tricks bei Hebas, Nr. 5

von Dr. Hans Hehl

1. HEBAS und WordStar . . . der Bildschirmeditor ist da.

Nachdem der Klassiker der Textverarbeitungsprogramme, WordStar von der Fa. Graf zu einem vernünftigen Preis erhältlich ist (siehe *LOOP* 8/9 u. 10), können Basic-Programme sehr bequem damit geschrieben und editiert werden. Dazu nun einige Erläuterungen, die wir gleich am Bildschirm ausprobieren wollen.

Starten Sie HEBAS und laden dann ein damit geschriebenes Basic-Programm. Dieses speichern Sie im ASCII-Code mit dem Befehl RESAVE „Dateiname",a wieder ab. Der Buchstabe a nach dem Komma bewirkt, daß Ihr Programm nicht im internen Tokenformat (siehe *LOOP* 7, S. 5/6) abgelegt wird. Sie können unbesorgt sein, Ihr Programm ist auch im ASCII-Format wieder ladbar. Ganz Vorsichtige können ja einen neuen Programmnamen verwenden (dann genügt der Befehl „SAVE neuer Dateiname",a).

Nun geht es mit dem Befehl BYE zurück zu CP/M. Starten Sie WordStar. Das lästige Warten nach der Copyright-Meldung, auf das Menue kann durch einen beliebigen Tastendruck, z. B. die Leertaste, abgekürzt werden. Verwenden Sie den N-Befehl zum Bearbeiten einer Programm-Datei. Der Start geht sehr schnell, wenn WordStar auf der RAM-Floppy (die eventuell sogar noch gepuffert ist) zur Verfügung steht. Als Dateiname geben Sie den Namen Ihres Basic-Programmes an. Jetzt muß der Dateizusatz „.bas" unbedingt angegeben werden. Fortgeschrittene schalten mit dem Befehl [CTRL-J], [H] und [2] die Anzeige des Menues ab (siehe Handbuch WordStar). Jetzt können Sie beliebig in Ihrem Programm mit den WordStar-Befehlen editieren, einfügen, blättern usw. Allerdings, Zeilennummern müssen Sie nun selbst eingeben, die mächtigen Befehle AUTO und RENUMBER stehen nicht zur Verfügung.

Gute Listings . . . kein Problem

Sie wollen in Ihrem Listing Leerzeilen ohne Zeilennummern als Absatz haben. Geben Sie am Zeilenende oder am Zeilenanfang einer Basic-Zeile [CTRL-N] ein und WordStar erzeugt eine Leerzeile. Wenn Sie mit [CTRL-K] und [D] Ihre Datei abspeichern und WordStar verlassen haben, starten Sie HEBAS und laden Ihr Programm. Der LIST-Befehl zeigt, daß alle Leerzeilen fehlen. Beim Laden werden diese von HEBAS ignoriert.

Sie wollen Leerzeichen für Einschübe verwenden. Auch dies geht. Sie können die ganze Basic-Zeile mit Zeilennummer nach rechts einrücken. Auch zwischen Zeilennummer und Programmzeile sind Leerzeichen möglich. Sehr schöne Listings entstehen durch Verwendung des Tabulators mit [CTRL-I].

WordStar kann mehr . . . z. B. HEBAS starten

Verwenden Sie den R-Befehl und geben Sie als Programm-Name Hebas.com an. Hebas meldet sich mit der Frage nach der Speichergrenze. Wie gewohnt geht es nun unter Basic weiter. Der BYE-Befehl führt zu WordStar zurück.

Der Leser ist aufgerufen, weitere Möglichkeiten zu erforschen.

In Bild 1 ist ein Programm der in *LOOP* 10 im Trick Nr. 4 vorgestellten Grundlagen-Datei HEBASMA.DOC (Abschnitt 6 für Anfänger) enthalten. Es zeigt sehr schön, wie Listings aussehen können, wenn man WordStar verwendet.

2. HEBAS-Fehler

Nach Murphy's Gesetz ist der Fehler dort, wo die Daten ganz offensichtlich richtig sind und nicht überprüft werden müssen.

– verschobenes Directory bei schreibgeschützten Dateien:

So zeigt sich bei HEBAS, daß bei mit STAT *.* $R/O schreibgeschützten BASIC-Programmen ein verschobenes Directory entsteht (besonders unschön beim Ausdruck, da 6-spaltig).

Schreibgeschützte Dateien enthalten beim ersten Buchstaben des Dateisatzes nach dem Punkt ein gesetztes Bit 7, aus 42h wird C2. Dies wurde vom Zeichenzähler in HEBAS nicht berücksichtigt.

Abhilfe: das Byte 7F in Adresse 32E8 wird durch DD ersetzt.

3. Cursor abschalten unter HEBAS

Stört das Blinken des Cursors beim BYTE-Befehl, so hilft das Umschalten in den Graphik-Modus beim NDR-Rechner. Bild 2 zeigt ein Demo-Programm.

4. Jux-Befehl ... mein Computer wird high

Probieren Sie einmal den Befehl CALL 2805. Was dann geschieht, ist hervorragend geeignet, Computerlaien das Fürchten zu lehren. Der Programmablauf kann mit der Leertaste angehalten werden (ab Vers. 1.3, siehe *LOOP* 10), [CTRL-E] bricht ab (Für Insider: es erfolgt ein Sprung in die LVAR-Routine).

5. Planungen:

HEBAS läuft demnächst auch mit dem in *LOOP* 10, S. 27 beschriebenen Z80-Emulator, da alle Befehle mit illegalen Z80-Opcodes (z. B. Halbbytes beim IX-Register, siehe mc 1/82, S.27, Franzis-Verlag) entfernt werden. Dann ist auch die CPU 64180 verwendbar. HEBAS enthält neuerdings den Befehl USER wie unter CP/M für unterteilte Directories. Bis zu 9 User-Ebenen sind möglich. HEBAS wird demnächst auch an CP/M+ (Plus) angepaßt.

Beachten Sie den Up-Date-Service (siehe *LOOP* 8/9, S. 22).

```
100 ' dezhexbi.bas   Dezimal-Hex-Binär-Liste
110 ' siehe mc 12/82  Seite 44
115 ' angepaßt von Dr. Hehl Hans

120 ZZ=0:
130             INPUT"von Dezimalzahl";VB:PRINT
140             INPUT"bis Dezimalzahl";BD:PRINT
    150 IF BD>255 THEN PRINT"zu große Zahl!":GOTO 140
160             INPUT"Bildschirm = 0, Drucker = 2";DV

180 FOR K = VB TO BD

        190 W$="": IF K<100 THEN W$=" " :' Formatierung
        200 IF K<10  THEN W$=" "

        210 PRINT#DV,"            ";W$;K;"   ";HEX$(K);"   ";

                230 FOR I = 7 TO 0 STEP -1
                    240 PRINT#DV,SGN(K AND 2^I);
                250 NEXT I

        270 PRINT#DV,"      ";CHR$(K):ZZ=ZZ+1

  280 IF ZZ/60=INT(ZZ/60) THEN FOR T=1 TO 12:PRINT#DV:NEXT T

290 NEXT K

300 END
```

Bild 1: BASIC-Zeilen mit WordStar geschrieben

```
10 G$ = CHR$(27) + CHR$(27) + "G"        : REM Graphikmodus ein
20 E$ = "A"                              : REM Graphik-Modus aus
30 PRINT"Taste drücken";
40 PRINT G$                              : REM Cursor aus
50 B = BYTE($0)
60 PRINT E$                              : REM Cursor ein
70 PRINT B
80 IF B = 5 THEN STOP ELSE GOTO 30
90 GOTO 50
```

Bild 2: Abschalten des Cursors beim BYTE-Befehl (NDR-Rechner)

FÜR 68000-EINSTEIGER

Hilfsprogramme für den 68008

Teil 6

Rüdiger Bäcker

Nach Erscheinen der Hardcopy-Maus-Baugruppe wurden nun schon verschiedene Möglichkeiten zum Aufruf der Hardcopyroutine vorgestellt. Eine der Lösungen war, die Routine CI umzulenken. Diese an sich schon sehr gute Lösung erlaubt den Aufruf der Hardcopy in jedem Programm mit einer Tastatureingabemöglichkeit.

Es fehlt also nur noch eine Lösung für die Fälle, wo keine Tastaturabfrage erfolgt. Auch das kann man recht einfach lösen. Ich stelle hier nun eine Möglichkeit vor, mit der an jeder Stelle eine Hardcopy ausgelöst werden kann. Realisiert wird dies durch einen Interrupt. Einen Interrupt (engl.: Unterbrechung) kann man beim 68008 hardwaremäßig auf drei Arten erzeugen:

- durch einen negativen Impuls auf der Leitung /INT
- durch einen negativen Impuls auf der Leitung /NMI
- durch einen gleichzeitigen negativen Impuls auf den Leitungen /INT und /NMI

Was passiert nun eigentlich bei einem Interrupt?

Ein Interrupt ist eine Ausnahme (Exception) vom normalen Betrieb des Prozessors. Für solche Ausnahmen hat der 68008 einen Speicherbereich (Adressen $0 – $400), in dem er die sogenannten Exception-Vectoren erwartet. Das heißt, im Falle einer Exception erwartet der Prozessor an einer bestimmten, von der Art der Exception abhängigen Speicherstelle eine Adresse für ein Programm, das beim Auftreten der Exception abgearbeitet werden soll. Für die beim 68008 verwendbaren Interrupts sind dies die Adressen $68 (/NMI), $74 (/INT) und $70 (/NMI + /INT). Da diese Adressen unter Umständen in einem Eprom liegen, können sie normalerweise nicht geändert werden. Um dennoch eigene Programme zur Interruptbehandlung einsetzen zu können, sind in den Vectoren RAM-Adressen im Arbeitsspeicher des Grundprogrammes eingetragen. In diesen Bereich des Grundprogrammes kann dann wiederum die Adresse der Interruptroutine eingetragen werden. Die Adressen im Rambereich sind:

/NMI = Ramstart + $0
/INT = Ramstart + $6
/INT + /NMI. = Ramstart + $C

Das Eintragen der Adressen geschieht wie bereits in *LOOP* 8/9 gezeigt und wird durch das Programm INTINIT erledigt. Dies ist jedoch nicht die einzige Aufgabe der Routine. Die Interrupts /INT und /NMI müssen freigegeben werden. Dazu muß in das Statusregister das entsprechende Steuerwort geschrieben werden. Wir benutzen hier die Leitung /INT, das ist der Interrupt-Level 5. Dazu wird in die entsprechende Adresse im Ram die Programmadresse eingetragen. Da das Grundprogramm den OPCODE für JMP schon dort hinschreibt, können wir uns das sparen. Das Steuerwort # $2000 gibt dann den Interruptlevel 5 frei. Nun wird bis zum nächsten Reset bei jedem negativen Impuls auf der Leitung /INT unser Programm angesprungen. In unserer Interruptroutine werden dann zunächst einmal alle Register gerettet und erst dann das entsprechende Porgramm für die Hardcopy aufgerufen. Die Adresse des gewünschten Programmes kann an der Adresse 432 eingetragen werden. Der Aufruf der Initialisierung kann durch einen Bibliothekseintrag erfolgen. Wichtig ist noch, daß die Leitung /INT von der FLO2 vom Bus getrennt wird!

Zur Erzeugung des Interrupts gibt es nun verschiedene Möglichkeiten. Die einfachste bietet sich den Besitzern der Cherry-Tastatur an. Die Taste BREAK ist als seperate Leitung zur KEY geführt. Hier muß dann lediglich noch eine Leitung von PIN 15 der Stiftleiste zum PIN 32 vom Bus gezogen werden. Natürlich kann man auch eine Taste der Maus oder einen separaten Taster einsetzen. Wichtig ist dabei aber, daß der Taster entprellt wird. Wie man das macht, ist schon mehrfach beschrieben worden.

```
000400                  ORG $400                        * GRUNDPROGRAMM AB $E0000 !
000400
000400                  *       R U B A - I N T
000400                  *
000400                  *       COPYRIGHT (C) 1986 BY RUEDIGER BAECKER - POSTFACH 4111 - 5820 GEVELSBERG
000400                  *
000400                  *       ROUTINE ZUM AUFRUF VON PROGRAMMEN UEBER INTERRUPT /INT
000400
000400                  KOPF:                           * BIBLIOTHEKSEINTRAG FUER INIT
000400  55AA0180        DC.L $55AA0180
000404  494E542D494E49   DC.B 'INT-INIT'
00040B  54
00040C  00000020        DC.L INTINIT-KOPF
000410  0000003E        DC.L ENDEA-KOPF
000414  01              DC.B 1
000415  00 00 00        DC.B 0,0,0
000418  00000000        DC.L 0,0
00041C  00000000
000420
= 0000000B              INTVEC          EQU     $0B     * ADRESSE FUER INTERRUPTVECTOR (A5)
```

```
000420
000420                  INTINIT:                        * SETZT VECTORADRESSE UND GIBT /INT FREI
000420  2B7C 0000042E   MOVE.L #PROGRAMM,INTVEC(A5)     * LEVEL 5 - INT-LEITUNG
000426  0008
000428  46FC 2000       MOVE #$2000,SR                  * INTERRUPTS FREIGEBEN
00042C  4E75            RTS
00042E
00042E                  PROGRAMM:                       * AUFRUF DES USERPROGRAMMES
00042E  4BE7 FFFF       MOVEM.L A0-A7/D0-D7,-(A7)       * ERST ALLE REGISTER RETTEN
000432  4EB9 00000000   JSR GLOAD                       * HIER ADRESSE DES AUFZURUFENDEN PROGRAMMES
                        ------------------------------
Undefiniertes Symbol
00043B
00043B  4CDF FFFF       MOVEM.L (A7)+,A0-A7/D0-D7       * REGISTER ZURUECK
00043C  4E73            RTE                             * RETURN FROM EXCEPTION
00043E
00043E                  ENDEA:
00043E                  END.

0001    Fehler entdeckt
0EBABA  Ende-Symboltabelle
```

Hilfsprogramme für den 68008

Teil 7

von Rüdiger Bäcker

Die Hilfsprogramme für den 68008 werden mit einem Hilfsprogramm zur Verwaltung der Hilfsprogramme fortgesetzt. Die Routine RUBA-MEN gestattet den Aufruf beliebiger Programme über ein Menue. Das ist an sich nichts besonderes, denn die Programme könnten ebensogut über die Bibliothek aufgerufen werden. Das besondere an RUBA-MEN ist die Art, wie die Routine selbst aufgerufen wird! Man kann das Menue an jeder beliebigen Stelle auf den Bildschirm holen und die gewünschten Programme dann ausführen. Wer den Artikel RUBA-INT gelesen hat, wird jetzt schon wissen wie das geht: Richtig, über einen Interrupt. Die Funktion des Interrupts ist im o.g. Artikel schon beschrieben worden, so daß ich mich hier auf die Programmbeschreibung des Menue-Programms beschränken kann. Die Initialisierung erfolgt wiederum über die Bibliothek. Bei Auftreten eines Interrupts erscheint dann das Menue auf dem Bildschirm und man kann ein beliebiges Programm ausführen. Nach Beendigung des Programmes gelangt man wieder an den Ausgangspunkt zurück.

Die Anzahl und Art der im Menue erscheinenden Programme kann durch Anpassen der Tabellen und der Textausgabe natürlich beliebig verändert werden.

Wer will, kann das Programm auch noch verbessern, dazu einige Anregungen:

- Suchen aller Bibliothekseinträge und Aufnahme ins Menue
- Aufruf von Funktionen des Grundprogrammes
- Mausgesteuerte Auswahl aus dem Menue
- Verbesserung des Bildschirmhandlings durch retten der GDP-Register etc.

Will man das Menue aus der Bibliothek aufrufen, so ist der Befehl RTE in RTS zu ändern.

```
Do 11.05.86 - 17:46:26
Rolf-D.Klein b8000/08 Assembler 4.7 (C) 1984, Seite 1


000400                  ORG $400                        * GRUNDPROGRAMM AB $E0000 !
000400
000400                  *       R U B A - M E N
000400                  *
000400                  *       COPYRIGHT (C) 1986 BY RUEDIGER BAECKER - POSTFACH 4111 - 5820 GEVELSBERG
000400                  *
000400                  *       ROUTINE ZUM AUFRUF VON PROGRAMMEN UEBER INTERRUPT /INT ODER BIBLIOTHEK
000400
000400                  KOPF:                           * BIBLIOTHEKSEINTRAG FUER INIT
000400  55AA0180        DC.L $55AA0180
000404  494E542D494E49   DC.B 'INT-INIT'
00040B  54
00040C  00000044        DC.L INTINIT-KOPF
000410  000002C3        DC.L ENDEA-KOPF
000414  01              DC.B 1
000415  00 00 00        DC.B 0,0,0
000418  00000000        DC.L 0,0
00041C  00000000
000420
000420                  KOPF1:                          * BIBLIOTHEKSEINTRAG FUER MENUE
000420  55AA0180        DC.L $55AA0180                  * ACHTUNG, BEMERKUNG BEI INTENDE BEACHTEN !!
000424  5255424120D045   DC.B 'RUBA-MEN'
00042B  4E
00042C  00000032        DC.L INTHAND-KOPF1
000430  000002C3        DC.L ENDEA-KOPF
000434  01              DC.B 1
000435  00 00 00        DC.B 0,0,0
000438  00000000        DC.L 0,0
00043C  00000000
000440  00000000        DC.L 0
000444
= 00000008              INTVEC          EQU     $08     * ADRESSE FUER INTERRUPTVECTOR (A5)
000444
000444                  INTINIT:                        * SETZT VECTORADRESSE UND GIBT /INT FREI
000444  2B7C 00000452   MOVE.L #INTHAND,INTVEC(A5)     * LEVEL 5 - INT-LEITUNG
00044A  0008
00044C  46FC 2000       MOVE #$2000,SR                  * INTERUPTS FREIGEBEN
000450  4E75            RTS
000452
000452                  INTHAND:                        * HAUPTPROGRAMMSTART, AUFRUF BEI INTERRUPT
000452  48E7 FFFF       MOVEM.L A0-A7/D0-D7,-(A7)       * ALLE REGISTER RETTEN
000456
000456  3E3C 0010       MOVE #'CLR,D7                   * BILDSCHIRM LOESCHEN
00045A  4E41            TRAP #1
00045C
00045C  9040            SUB D0,D0                       * REGISTER LOESCHEN
00045E  9241            SUB D1,D1
000460  9442            SUB D2,D2
```

```
000462
000462                  START:
000462  41FA 0188       LEA DTX1(PC),A0                 * TEXTE AUSGEBEN
000466  103C 0043       MOVE.B #$43,D0                  * SCHRIFTGROESSE
00046A  123C 0000       MOVE.B #$0,D1                   * X-POSITION
00046E  143C 00DC       MOVE.B #220,D2                  * Y-POSITION
000472  3E3C 000A       MOVE #'WRITE,D7                 * UND AUSGEBEN
000476  4E41            TRAP #1
000478  0442 002D       SUB #75,D2                      * NEUE Y-POSITION
00047C  103C 0032       MOVE.B #$32,D0                  * NEUE SCHRIFTGROESSE
000480  41FA 0178       LEA DTX2(PC),A0                 * USW. ....
000484  3E3D 000A       MOVE #'WRITE,D7
000488  4E41            TRAP #1
00048A  0442 0019       SUB #25,D2
00048E  1033 0021       MOVE.B #$21,D0

Rolf-D.Klein b8000/08 Assembler 4.7 (C) 1984, Seite 2

000492  41FA 01B3       LEA DTX3(PC),A0
000496  3E3C 000A       MOVE #'WRITE,D7
00049A  4E41            TRAP #1
00049C  303C 0021       MOVE #$21,D0
0004A0  0442 0014       SUB #20,D2
0004A4  41FA 01BD       LEA TXT1(PC),A0
0004A8  3E3C 000A       MOVE #'WRITE,D7
0004AC  4E41            TRAP #1
0004AE  0442 000A       SUB #10,D2
0004B2  41FA 01BF       LEA TXT2(PC),A0
0004B6  3E3C 000A       MOVE #'WRITE,D7
0004BA  4E41            TRAP #1
0004BC  0442 000A       SUB #10,D2
0004C0  41FA 0190       LEA TXT3(PC),A0
0004C4  3E3C 000A       MOVE #'WRITE,D7
0004C8  4E41            TRAP #1
0004CA  0442 000A       SUB #10,D2
0004CE  41FA 0191       LEA TXT4(PC),A0
0004D2  3E3C 000A       MOVE #'WRITE,D7
0004D6  4E41            TRAP #1
0004D8  0442 000A       SUB #10,D2
0004DC  41FA 0190       LEA TXT5(PC),A0
0004E0  3E3C 000A       MOVE #'WRITE,D7
0004E4  4E41            TRAP #1
0004E6  0442 000A       SUB #10,D2
0004EA  41FA 0190       LEA TXT6(PC),A0
0004EE  3E3C 000A       MOVE #'WRITE,D7
0004F2  4E41            TRAP #1
0004F4  0442 000A       SUB #10,D2
0004F8  41FA 0192       LEA TXT7(PC),A0
0004FC  3E3C 000A       MOVE #'WRITE,D7
000500  4E41            TRAP #1
000502  0442 000A       SUB #10,D2
```

```
000506  41FA 0193       LEA TXT8(PC),A0
00050A  3E3C 000A       MOVE #!WRITE,D7
00050E  4E41            TRAP #1
000510  0442 000A       SUB #10,D2
000514  41FA 0194       LEA TXT9(PC),A0
00051B  3E3C 000A       MOVE #!WRITE,D7
00051C  4E41            TRAP #1
00051E  0442 000A       SUB #10,D2
000522  41FA 0194       LEA MTXT(PC),A0
000526  3E3C 000A       MOVE #!WRITE,D7
00052A  4E41            TRAP #1
00052C
00052C  3E3C 000C       MOVE #!CI,D7                 * AUSWAEHLEN
000530  4E41            TRAP #1
000532
000532  3F00            MOVE D0,-(A7)                * DANN ZUR SICHERHEIT BILDSCHIRM LOESCHEN
000534  3E3C 0010       MOVE #!CLR,D7                * D0 WIRD GERETTET, DA IN CLR BENUTZT UND
000538  4E41            TRAP #1                      * DIE NUMMER DES GEWAEHLTEN UNTERPROGRAMMES
00053A  301F            MOVE (A7)+,D0                * NOCH IN D0 STEHT
00053C
00053C  0C00 0031       CMP.B #'1',D0                * PROGRAMM 1 AUSGEWAEHLT ?
000540  6700 004E       BEQ PROG1                    * JA, DANN DORTHIN
000544  0C00 0032       CMP.B #'2',D0                * SONST WEITER ...
000548  6700 0050       BEQ PROG2
00054C  0C00 0033       CMP.B #'3',D0
000550  6700 0052       BEQ PROG3
000554  0C00 0034       CMP.B #'4',D0
000558  6700 0054       BEQ PROG4
00055C  0C00 0035       CMP.B #'5',D0
000560  6700 0056       BEQ PROG5
000564  0C00 0036       CMP.B #'6',D0

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 3

000568  6700 0058       BEQ PROG6
00056C  0C00 0037       CMP.B #'7',D0
000570  6700 005A       BEQ PROG7
000574  0C00 0038       CMP.B #'8',D0
000578  6700 005C       BEQ PROG8
00057C  0C00 0039       CMP.B #'9',D0
000580  6700 005E       BEQ PROG9
000584  0C00 006D       CMP.B #'m',D0
000588  6700 005C       BEQ INTENDE
00058C  6000 FED4       BRA START                    * WENN KEINE GUELTIGE AUSWAHL
000590
000590                  PROG1:                       * NUN DAS GEWAEHLTE PROGRAMM AUSFUEHREN
000590  4EB9 00001420   JSR PROG1A                   * MIT JSR, DA ADRESSDISTANZ NICHT KLAR
000596  6000 004E       BRA INTENDE
00059A
00059A                  PROG2:
00059A  4EB9 00002B20   JSR PROG2A                   * PROGRAMM AUSFUEHREN
0005A0  6000 0044       BRA INTENDE
0005A4
0005A4                  PROG3:
0005A4  4EB9 00001C20   JSR PROG3A                   * PROGRAMM AUSFUEHREN
0005AA  6000 003A       BRA INTENDE
0005AE
0005AE                  PROG4:
0005AE  4EB9 00001B20   JSR PROG4A                   * PROGRAMM AUSFUEHREN
0005B4  6000 0030       BRA INTENDE
0005B8
0005B8                  PROG5:
0005B8  4EB9 00003000   JSR PROG5A                   * PROGRAMM AUSFUEHREN
0005BE  6000 0026       BRA INTENDE
0005C2
0005C2                  PROG6:
0005C2  4EB9 00001218   JSR PROG6A                   * PROGRAMM AUSFUEHREN
0005C8  6000 001C       BRA INTENDE
0005CC
0005CC                  PROG7:
0005CC  4EB9 00002028   JSR PROG7A                   * PROGRAMM AUSFUEHREN
0005D2  6000 0012       BRA INTENDE
0005D6
0005D6                  PROG8:
0005D6  4EB9 000D6020   JSR PROG8A                   * PROGRAMM AUSFUEHREN
0005DC  6000 0008       BRA INTENDE
0005E0
0005E0                  PROG9:
0005E0  4EB9 00002420   JSR PROG9A                   * PROGRAMM AUSFUEHREN
0005E6
0005E6                  INTENDE:
0005E6
0005E6  4CDF FFFF       MOVEM.L (A7)+,A0-A7/D0-D7    * REGISTER RESTAURIEREN
0005EA  4E73            RTE                          * HIER RTS EINTRAGEN, FUER AUFRUF VOM GRPRG.
0005EC
0005EC                  ** PROGRAMMTABELLE :         * HIER DIE STARTADRESSEN DER PROGRAMME
0005EC                                               * EINTRAGEN
0005EC
= 00001420              PROG1A      EQU    $1420
= 00002B20              PROG2A      EQU    $2B20
= 00001C20              PROG3A      EQU    $1C20
= 00001B20              PROG4A      EQU    $1B20
= 00003000              PROG5A      EQU    $3000
= 00001218              PROG6A      EQU    $1218
= 00002028              PROG7A      EQU    $2028
= 000D6020              PROG8A      EQU    $D6020
= 00002420              PROG9A      EQU    $2420
0005EC

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 4

0005EC                  ** TEXTE:
0005EC
0005EC                  DTX1:
0005EC  494E54202D2048  DC.B 'INT - Handler',0
0005F3  616E646C657200
0005FA                  DTX2:
0005FA  28432920313938  DC.B '(C) 1986 by Ruediger Baecker',0
000601  36206279205275
000608  65646967657220
00060F  42616563686572
000616  00
000617                  DTX3:
000617  42697474652050  DC.B 'Bitte Programm auswaehlen :',0
00061E  726F6772616D6D
000625  20617573776165
00062C  686C656E203A00
000633                  TXT1:
000633  31203D3D3E2052  DC.B '1 ==> RUBATRANS',0
00063A  5542415452414E
000641  5300
000643                  TXT2:
000643  32203D3D3E2052  DC.B '2 ==> RUBAFILL',0
00064A  5542414649404C
000651  00
000652                  TXT3:
000652  33203D3D3E2052  DC.B '3 ==> RUBADUMP',0
000659  55424144554D50
000660  00
000661                  TXT4:
000661  34203D3D3E2056  DC.B '4 ==> VERIFY',0
000668  455249465900
00066E                  TXT5:
00066E  35203D3D3E2052  DC.B '5 ==> RAM-USR',0
000675  414D2D55535200
00067C                  TXT6:
00067C  36203D3D3E2051  DC.B '6 ==> QUICKPROM',0
000683  5549434B50524F
00068A  4D00
00068C                  TXT7:
00068C  37203D3D3E2041  DC.B '7 ==> ASEMPINT',0
000693  53454D50494E54
00069A  00
00069B                  TXT8:
00069B  38203D3D3E2048  DC.B '8 ==> HARDCOPY',0
0006A2  415244434F5059
0006A9  00
0006AA                  TXT9:
0006AA  39203D3D3E2054  DC.B '9 ==> TEXT 68',0
0006B1  45585420363800
0006B8                  MTXT:
0006B8  6D203D3D3E2045  DC.B 'm ==> Ende',0
0006BF  6E646500
0006C3                  ENDEA:
0006C3                  END.

0E8D30  Ende-Symboltabelle
```


Impressum:

LOOP, Zeitung für Computerbauer

Herausgeber: Gerd Graf

Redaktion: Rolf-Dieter Klein, Gerd Graf

Gestaltung und Druck:
Karl-Heinz Rieder, Kempten

Herstellung und Anzeigenverwaltung:

GES GmbH
Magnusstraße 13, 8960 Kempten

Anzeigenpreisliste 1/84

TURBOPASCAL-PROGRAMM

Dieses Turbopascal-Programm ist für den NDR 80-CP/M mit dem GDP (EF 9366) geschrieben.

Schriftbefehle wie HÖHE, BREITE und Umschaltung des Zeichensatzes in deutsch.

Zum Programm selbst, es zeichnet einen Lottoblock mit 49 Zahlen im Format 7 x 7 auf den Bildschirm. Die durch Zufall errechneten 6 Lottozahlen werden durch ein Pluszeichen gekennzeichnet.

Heinz Sciedeck
Kälblingstraße 8 · 7141 Steinheim 3
Tel. 07144/24246

```
  (* Programm LOTTO    H.Schiedeck   Steinheim-3 den 24.07.86  *)

program lotto;

var  i,j,z,k  : integer;
var  ch : char;
     L : array  [1..6]  of integer;
     kreuz,neuezahl,ende :boolean;

begin
 repeat
     randomize; (* neue Zufallszahlen *)
     for i:=1 to 6 do begin          (* 6 mal *)
       repeat
         z := random (48) + 1; (* zahl zwischen 1 und 49 *)
         neuezahl := true;
         for j:=1 to i do if z = L[j] then neuezahl := false
           (* ist wahr , wenn z neue zufallszahl ist *)
       until neuezahl;
       L[i] := z
     end; (* von for i *)

clrscr;
  writeln (chr(27),'z1');      (* deutscher Zeichensatz einsch. *)
  writeln (chr(27),chr(27),'G');      (* Grafikmodus einschalten *)
  writeln ('M10 230');                (* Curser setzen *)
  writeln ('G3 $22');     (* GDP Port 73 Schrift grösse *)
  writeln ('Bich empfehle DIR als Lottozahlen : ');
  writeln ('M0 20');                        (* Curser setzen *)
  writeln ('G3 $21');     (* GDP Port 73 Schrift grösse *)
  writeln ('Bwillst DU noch einen Tip haben  j / n   ?');
  writeln ('Y0');           (* GDP nur Bildseite null ausgeben *)
  writeln ('A');             (* Grafikm. aus Alphamodus wieder ein *)
  writeln; writeln;

ende := false;
i := 1;
repeat
  writeln;
for k := 1 to 7 do begin            (* Lotto-Block ausgeben *)
   kreuz := false;
   for j := 1 to 6 do if L[j]=i then kreuz := true;
   if  kreuz  then  write  ('      +')  (* Zufallszahl  durch  kreuz
                                          ersetzen *)
               else write (i:7);
   if i >= 49 then ende := true;       (* Ausgabe bei 49 abbrechen *)
        i := i+1;
                end;
   writeln;
until ende;
    writeln;
    writeln (' Meine Angaben sind leider ohne Gewähr');

    read (kbd,ch)          (* Zeichen von Tast. lesen*)
until  ch <>'j';
   clrscr;
  writeln (chr(27),'z0');
end.
```

Gemischte Ausgabe

Ralph Dombrowski, 2208 Glückstadt

Es wurden schon viele Druckerroutinen in der *LOOP* veröffentlicht, aber kein Programm bietet die Möglichkeit deutsche und amerikanische Zeichen auf einem Drucker auszugeben. Obwohl der Texteditor umschaltbar ist, war der Drucker bisher fest auf einen Zeichensatz eingestellt. Ich stelle hier deshalb eine kleine Routine vor, die Abhilfe schafft.

Die Aufgabe dieses Programmes ist es, zu erkennen, was für ein Zeichen ausgegeben werden soll und den Zeichensatz des Druckers gegebenenfalls umzuschalten. Die Routine hat einen kleinen Merker, der anzeigt, welcher Zeichensatz am Drucker eingeschaltet ist. Bevor das erste Zeichen auf Drucker ausgegeben wird, muß der Drucker initialisiert werden. Dazu wird vor dem ersten Druckeraufruf die Zahl -1 in die Speicherzelle „dflag" geschrieben. Den Rest erledigt das Programm selbständig.

```
************************ Druckerzeichensätze umschalten ******************

dflag: ds.b 1            * 0=amerikanischer Zeichensatz an/1=deutscher Z. an
ds 0

start:
 move.b #-1,dflag        * Zum Drucker initialisieren
 moveq #!ciinit2,d7      * Textstart festlegen
 trap #1
start0:
 moveq #!ci2,d7          * Zeichen holen
 trap #1
 beq.s ende              * Wenn 0, dann Ende
 bsr.s lo                * Zeichen ausgeben
 bra.s start0            * Nächstes Zeichen
ende:
rts

lo:
 tst.b dflag             * Drucker initialisiert ?
 bpl.s lo1               * Ja, dann weiter
 bsr.s amer              * Sonst Zeichensatz auf amerikanisch einstellen
lo1:
 tst.b d0                * Ist auszugebenes Zeichen deutsches Zeichen ?
 bmi.s lo2               * Ja, dann weiter
 tst.b dflag             * Schon amerikanischer Zeichensatz eingestellt ?
 beq.s lo3               * Ja, dann Zeichen ausgeben
 bsr.s amer              * Sonst amerikanischen Zeichensatz einschalten
 bra.s lo3               * Zeichen ausgeben
lo2:
 cmp.b #1,dflag          * Ist deutscher Zeichensatz eingeschaltet ?
 beq.s lo3               * Ja, dann Zeichen ausgeben
 bsr.s deutsch           * Sonst deutschen Zeichensatz einstellen
lo3:
 moveq #!lo,d7           * Zeichen ausgeben
 trap #1
rts

amer:                    * Amerikanischen Zeichensatz einschalten
 move d0,-(a7)           * d0 nicht zerstören
 moveq #$1b,d0
 moveq #!lo,d7
 trap #1
 moveq #'R',d0
 moveq #!lo,d7
 trap #1
 clr d0
 moveq #!lo,d7
 trap #1

 move (a7)+,d0
 clr.b dflag
rts

deutsch:                 * Deutschen Zeichensatz einschalten
 move d0,-(a7)           * d0 nicht zerstören
 moveq #$1b,d0
 moveq #!lo,d7
 trap #1
 moveq #'R',d0
 moveq #!lo,d7
 trap #1
 moveq #2,d0
 moveq #!lo,d7
 trap #1
 move (a7)+,d0
 move.b #1,dflag
rts

end
```

Cad-Programm für den NDR-Computer 68008

von Heinrich Zarch, 5438 Ahrweiler

Das Programm läuft auf dem NDR-Computer mit dem RDK-Grundprogramm 68K.

Mit dem Programm können Zeichnungen auf dem Bildschirm erstellt und über Bildschirm oder einen angeschlossenen Selbstbauplotter ausgegeben werden.

Nach dem Start meldet sich das Menue und es kann zwischen Eingabe und Ausgabe gewählt werden.

Unter Eingabe-Bildausschnitt kann der zu bearbeitende Ausschnitt mit den Tasten E, D, X, S eingestellt werden, und nach Drücken der Taste C erscheint der Bildausschnitt mit Angabe der Schildkrötenposition und der Schildkröte, die als Fadenkreuz dient sowie die eventuell erstellten Zeichnungen.

Die Schildkröte wird mit den Tasten E, D, X, S um eine, und mit den Tasten R, F, C, A um 5 Stellen um ihre x- und y-Achse bewegt.

Mit der Taste Z wird gezeichnet und mit U auf nicht zeichnend umgestellt (siehe Listinganfang).

Die Koordinaten werden am Ende des Programmes abgespeichert. Mit M kehrt man in das Menue zurück.

Unter Ausgabe

a) kann man die Zeichnung auf den Bildschirm flimmerfrei ausgeben, dabei kann der Ausgabemaßstab im Programmlisting geändert werden;

b) kann die Zeichnung über einen Selbstbauplotter ausgegeben werden, der Maßstab ist im Programmlisting festlegbar;

c) der Selbstbauplotter kann justiert werden.

Die Grafikgröße ist nur durch die Ausgabegröße des Bildschirm und Plotters begrenzt. Die Grafik kann zusammen mit dem Programm dann abgespeichert werden.

Hinweis der Redaktion: Dieses Programm soll auch als Grundlage für Erweiterungen dienen. Diese wären: Einbinden der MAUS (Hardcopy-Maus), Symbole abspeichern, . . . Vielleicht erhalten wir einige Ideen?

```
Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 1

009C00                  *CAD 1.12 VOM 03.4.1986
009C00                  *PROGRAMM ZUM ERSTELLEN VON ZEICHNUNGEN
009C00                  *(C) 1986 HEINRICH ZAREK, 5483 AHRWEILER
010000                  ORG $10000
010000                  ******************
010000                  * MENÜ IST ZUSTAND
010000                  *
010000                  *               EINGABE
010000                  *               -------
010000                  *               BILDSCHIRM
010000                  *
010000                  *               EINSTELLUNG DES ZU BEARBEITENDEN BILDSCHIRMAUSSCHNITTES
010000                  *
010000                  *               BILDSCHIRMAUSSCHNITT    256 X 256
010000                  *                       (Y-WERT*2)
010000                  *
010000                  *               FADENKREUZSTEUERUNG             ABSPEICHERUNG
010000                  *
010000                  *                       R + 5                   Z - LINIE ZEICHNEN
010000                  *                       E + 1                   U - LINIE NICHT ZEICHNEN
010000                  *                         I                         (ANFANG LINIE)
010000                  *                A - 5 S -1 -O- D +1  F+ 5
010000                  *                         I
010000                  *                       X - 1
010000                  *                       C - 5
010000                  *
010000                  *               FADENKREUZPOSITIONSANZEIGE
010000                  *
010000                  ************************************************************
010000                  *
010000                  *               AUSGABE
010000                  *               -------
010000                  *               PLOTTER         BILDSCHIRM
010000                  *
010000                  *               POSITIONIEREN   STIFTSTEUERUNG
010000                  *
010000                  ************************************************************
010000                  * GEPLANTE ERWEITERUNGEN
010000                  *
010000                  *       EINGABESTEUERUNG ÜBER DIE MAUS
010000                  *       ------------------------------------------------------
010000                  *       ABSPEICHERUG UND LADEN DER PUNKTE
010000                  *
010000                  *       CASSETTE                DISKETTE
010000                  *       ------------------------------------------------------
010000                  *       ZEICHENSÄTZE
010000                  *
010000                  *       KREISE      KREISTEILE
010000                  *
010000                  *       HARDCOPY
010000                  *
010000                  *       VERSCHIEDENE MASSTÄBE
010000                  *
010000                  ************************************************************
010000                  ***  VEREINBARUNGSBAUSTEIN  ******
010000
= 0000000C              KOOR  EQU 12    *ANZAHL BYTES PRO EINTRAG
= 00000000              X1 EQU 0        *OFFSET ADRESSIERUNG
= 00000004              Y1 EQU 4
= 00000008              ST EQU 8
010000
010000                  *** VARIABLEN EINGABE ***
010000  00000000        AX: DC.L 0      *WERT AUSSCHNITSRAHMEN

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 2

010004  00000000        AY: DC.L 0
010008  00000000        PX: DC.L 0      *WERT ABSOLUT
01000C  00000000        PY: DC.L 0
010010  00000000        EX: DC.L 0      *WERT ZEICHENAUSSCHNITT
010014  00000000        EY: DC.L 0
010018  00000000        SX: DC.L 0      *WERT SCHILDKRÖTE
01001C  00000000        SY: DC.L 0
010020  00000000        XA: DC.L 0
010024  00000000        YA: DC.L 0
010028  00000000        ANZAHL: DC.L 0
01002C  00000000        KOOP:DC.L 0
010030  00000001        STIFT:DC.L 1
010034
010034                  *** VARIABLEN AUSGABE ***
010034
010034  00000000        XAP:DC.L 0      *X-ANFANG
010038  00000000        YAP:DC.L 0      *Y-ANFANG
01003C  00000000        XE:DC.L 0       *X-ENDE
010040  00000000        YE:DC.L 0       *Y-ENDE
010044  00000000        PXS:DC.L 0      *POSITION X-STIFTP
010048  00000000        PYS:DC.L 0      *POSITION Y-STIFTP
01004C  00000000        STIFTP:DC.L 0   *STIFTP HOCH ODER GESENKT
010050  00000000        WAAGR:DC.L 0    *WERT FÜR LINKS ODER RECHTS
010054  00000000        SENKR:DC.L 0    *WERT FÜR HOCH ODER TIEF
010058  00000000        XSCHRITT:DC.L 0 *WERT FÜR ADDITION POSITIV ONER NEGATIV
01005C  00000000        YSCHRITT:DC.L 0 *WERT FÜR ADDITION POSITIV ODER NEGATIV
010060
010060  103C 00BB       START:  MOVE.B #$BB,D0  *PROGRAMMBEGINN
010064  323C 0078               MOVE #120,D1
010068  343C 00B0               MOVE #128,D2
01006C  41F9 00010AFE           LEA TEX1,A0     *TITEL
010072  4EB9 00001386           JSR §WRITE
010078  23FC 00000029           MOVE.L #41,AX
01007E  00010000
010082  23FC 0000000B           MOVE.L #11,AY
010088  00010004
01008C  41F9 00010BDC           LEA KOORD,A0    *KOORDINATENVERZEICHNISSANFANGSKOORDINATEN LADEN
010092  23CB 0001002C           MOVE.L A0,KOOP
01009B  4EB9 00010AEA           JSR SCHLEIFE
01009E
01009E  4EB9 00000DA0   MENUE:  JSR §CLR        *MENÜ
0100A4  103C 0044               MOVE.B #$44,D0
0100A8  323C 0078               MOVE #120,D1
0100AC  343C 00C8               MOVE #200,D2
0100B0  41F9 00010B02           LEA TEX2,A0
0100B6  4EB9 00001386           JSR §WRITE
0100BC  103C 0033               MOVE.B #$33,D0
0100C0  323C 0078               MOVE #120,D1
0100C4  343C 0096               MOVE #150,D2
0100C8  41F9 00010B07           LEA TEX3,A0
0100CE  4EB9 00001386           JSR §WRITE
0100D4  323C 0078               MOVE #120,D1
0100D8  343C 0078               MOVE #120,D2
0100DC  41F9 00010B16           LEA TEX5,A0
0100E2  4EB9 000013B6           JSR §WRITE
0100E8
0100E8  4EB9 00000A00   WAHL:   JSR §CI         *AUSWAHL DER PROGRAMMPUNKTE
0100EE  0C00 0031               CMP.B #'1',D0
0100F2  6606                    BNE.S ST1
0100F4  4EB9 00010108           JSR EINGABE
0100FA  0C00 0032       ST1:    CMP.B #'2',D0
0100FE  66EB                    BNE.S WAHL
010100  4EB9 000106A8           JSR AUSGABE
010106  60E0                    BRA.S WAHL

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 3

010108
010108  4EB9 00000DA0   EINGABE:JSR §CLR        *EINSTELLEN DES BILDAUSSCHNITTES 256 * 256
01010E  103C 0033               MOVE.B #$33,D0
010112  323C 003C               MOVE #60,D1
010116  343C 00E4               MOVE#228,D2
01011A  41F9 00010B25           LEA TEX11,A0    *KOORDINATENEINGABE
010120  4EB9 00001386           JSR §WRITE
010126  103C 0022               MOVE.B #$22,D0
01012A  323C 005A               MOVE #90,D1
01012E  343C 00C8               MOVE #200,D2
010132  41F9 00010B3D           LEA TEX12,A0    * X= UND Y=  WERT
010138  4EB9 000013B6           JSR §WRITE
01013E  323C 0028               MOVE #40,D1     *BILDRAHMEN
010142  343C 000A               MOVE #10,D2
010146  4EB9 00000EE2           JSR §MOVETO
01014C  323C 0028               MOVE #40,D1
010150  343C 00BE               MOVE #190,D2
010154  4EB9 00000F4C           JSR §DRAWTO
01015A  323C 01E0               MOVE #480,D1
01015E  343C 00BE               MOVE #190,D2
010162  4EB9 00000F4C           JSR §DRAWTO
010168  323C 01E0               MOVE #480,D1
01016C  343C 000A               MOVE #10,D2
010170  4EB9 00000F4C           JSR §DRAWTO
010176  323C 0028               MOVE #40,D1
01017A  343C 000A               MOVE #10,D2
01017E  4EB9 00000F4C           JSR §DRAWTO
010184  2239 00010000   FIG:    MOVE.L AX,D1
01018A  2439 00010004           MOVE.L AY,D2
010190  41F9 00010BC8           LEA TABELLE,A0
010196  303C 000F               MOVE #15,D0     *AUSSCHNITTGRÖßE
01019A  4EB9 000032BC           JSR §FIGUR
0101A0  41F9 00010BD3           LEA BUFX,A0     *TEXT AUF BILDSCHIRM
0101A6  2039 00010010           MOVE.L EX,D0
0101AC  4EB9 000015FC           JSR §PRINT4D
0101B2  41F9 00010BD3           LEA BUFX,A0
0101B8  303C 0022               MOVE #$22,D0
0101BC  323C 0078               MOVE #120,D1
0101C0  343C 00C8               MOVE #200,D2
0101C4  4EB9 00001386           JSR §WRITE
0101CA  41F9 00010BD7           LEA BUFY,A0
0101D0  2039 00010014           MOVE.L EY,D0
```

```
0101D6  4EB9 000015FC          JSR $PRINT4D
0101DC  41F9 00010BD7          LEA BUFY,A0
0101E2  303C 0022              MOVE #$22,D0
0101E6  323C 0118              MOVE #280,D1
0101EA  343C 00C8              MOVE #200,D2
0101EE  4EB9 000013B6          JSR $WRITE
0101F4
0101F4  4EB9 00000A00  JUST:   JSR $CI         *BILDAUSSCHNITT EINSTELLEN
0101FA  0C00 0045              CMP.B #'E',D0
0101FE  6628                   BNE.S ST11
010200  06B9 00000008         ADD.L #8,AY
010206  00010004
01020A  04B9 00000064         SUB.L #100,YA
010210  00010024
010214  06B9 00000064         ADD.L #100,EY
01021A  00010014
01021E  4EB9 000101B4         JSR FIG
010224  6000 FFCE              BRA JUST
010228  0C00 0044      ST11:  CMP.B #'D',D0
01022C  6628                   BNE.S ST12
01022E  06B9 0000000E         ADD.L #14,AX
010234  00010000

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 4

010238  04B9 000000C8         SUB.L #200,XA
01023E  00010020
010242  06B9 00000064         ADD.L #100,EX
010248  00010010
01024C  4EB9 000101B4         JSR FIG
010252  6000 FFA0              BRA JUST
010256  0C00 0058      ST12:  CMP.B #'X',D0
01025A  6628                   BNE.S ST13
01025C  04B9 00000008         SUB.L #8,AY
010262  00010004
010266  06B9 00000064         ADD.L #100,YA
01026C  00010024
010270  04B9 00000064         SUB.L #100,EY
010276  00010014
01027A  4EB9 000101B4         JSR FIG
010280  6000 FF72              BRA JUST
010284  0C00 0053      ST13:  CMP.B #'S',D0
010288  6628                   BNE.S ST14
01028A  04B9 0000000E         SUB.L #14,AX
010290  00010000
010294  06B9 000000C8         ADD.L #200,XA
01029A  00010020
01029E  04B9 00000064         SUB.L #100,EX
0102A4  00010010
0102A8  4EB9 000101B4         JSR FIG
0102AE  6000 FF44              BRA JUST
0102B2  0C00 0043      ST14:  CMP.B #'C',D0
0102B6  6600 FF3C              BNE JUST
0102BA  4EF9 000102C4         JMP AUSSCHNITT
0102C0  6000 FF32              BRA JUST
0102C4
0102C4                 AUSSCHNITT:          *BILDSCHIRMAUSSCHNITT
0102C4  4EB9 00000DA0         JSR $CLR
0102CA  2A39 00010028         MOVE.L ANZAHL,D5*VORHANDENE GRAFIK AUS VERZEICHNIS LESEN UND
0102D0  0C85 00000000         CMP.L #0,D5
0102D6  6700 0074              BEQ AUS
0102DA  223C 00000000         MOVE.L #0,D1
0102E0  92B9 00010010         SUB.L EX,D1
0102E6  243C 00000000         MOVE.L #0,D2
0102EC  94B9 00010014         SUB.L EY,D2
0102F2  4EB9 00000EE2         JSR $MOVETO
0102F8  0485 00000001         SUB.L #1,D5
0102FE  41F9 00010BDC         LEA KOORD,A0
010304  222B 0000      SCH:   MOVE.L X1(A0),D1
010308  92B9 00010010         SUB.L EX,D1
01030E  C3FC 0002              MULS #2,D1
010312  242B 0004              MOVE.L Y1(A0),D2
010316  94B9 00010014         SUB.L EY,D2
01031C  2028 0008              MOVE.L ST(A0),D0
010320  0C80 00000001         CMP.L #1,D0
010326  6700 000E              BEQ MARKE1
01032A  4EB9 00000D6E         JSR $ERAPEN
010330  4EB9 0001033C         JSR MARKE2
010336  4EB9 00000D7C  MARKE1: JSR $SETPEN
01033C  4EB9 00000F4C  MARKE2: JSR $DRAWTO
010342  D1FC 0000000C         ADDA.L #KOOR,A0
010348  51CD FFBA              DBRA D5,SCH
01034C  4EB9 0001063A  AUS:    JSR ANZEI       *SICHTBAREN TEIL AUS BILDSCHIRM DARSTELLEN
010352  2239 00010008         MOVE.L PX,D1    *SCHILDKRÖTE POSITIONIEREN
010358  92B9 00010010         SUB.L EX,D1
01035E  C3FC 0002              MULS #2,D1
010362  23C1 0001001B         MOVE.L D1,SX
010368  2439 0001000C         MOVE.L PY,D2
01036E  94B9 00010014         SUB.L EY,D2

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 5

010374  C5FC 0002              MULS #2,D2
010378  23C2 0001001C         MOVE.L D2,SY
01037E  263C 0000005A         MOVE.L #90,D3
010384  4EB9 0000131A         JSR $SET
01038A  4EB9 00000A00  TAS:    JSR $CI         *SCHILDKRÖTENSTEUERUNG
010390  0C00 0045              CMP.B #'E',D0
010394  6630                   BNE.S ST20
010396  06B9 00000001         ADD.L #1,PY     *ABSOLUTPOSITION
01039C  0001000C
0103A0  06B9 00000002         ADD.L #2,SY
0103A6  0001001C
0103AA  2239 00010018         MOVE.L SX,D1
0103B0  2439 0001001C         MOVE.L SY,D2
0103B6  4EB9 0000131A         JSR $SET
0103BC  4EB9 0001063A         JSR ANZEI
0103C2  6000 FFC6              BRA TAS
0103C6  0C40 0052      ST20:  CMP #'R',D0
0103CA  6600 0032              BNE ST21
0103CE  06B9 00000005         ADD.L #5,PY
0103D4  0001000C
0103D8  06B9 0000000A         ADD.L #10,SY
0103DE  0001001C
0103E2  2239 00010018         MOVE.L SX,D1
0103E8  2439 0001001C         MOVE.L SY,D2
0103EE  4EB9 0000131A         JSR $SET
0103F4  4EB9 0001063A         JSR ANZEI
0103FA  6000 FFBE              BRA TAS
0103FE  0C00 0058      ST21:  CMP.B #'X',D0
010402  6600 0032              BNE ST22
010406  04B9 00000001         SUB.L #1,PY
01040C  0001000C
010410  04B9 00000002         SUB.L #2,SY
010416  0001001C
01041A  2239 00010018         MOVE.L SX,D1
010420  2439 0001001C         MOVE.L SY,D2
010426  4EB9 0000131A         JSR $SET
01042C  4EB9 0001063A         JSR ANZEI
010432  6000 FF56              BRA TAS
010436  0C00 0043      ST22:  CMP.B #'C',D0
01043A  6600 0032              BNE ST23
01043E  04B9 00000005         SUB.L #5,PY
010444  0001000C
01044B  04B9 0000000A         SUB.L #10,SY
01044E  0001001C
010452  2239 00010018         MOVE.L SX,D1
010458  2439 0001001C         MOVE.L SY,D2
01045E  4EB9 0000131A         JSR $SET
010464  4EB9 0001063A         JSR ANZEI
01046A  6000 FF1E              BRA TAS
01046E  0C00 0044      ST23:  CMP.B #'D',D0
010472  6600 0032              BNE ST24
010476  06B9 00000001         ADD.L #1,PX
01047C  00010008
010480  06B9 00000002         ADD.L #2,SX
010486  00010018
01048A  2239 00010018         MOVE.L SX,D1
010490  2439 0001001C         MOVE.L SY,D2
010496  4EB9 0000131A         JSR $SET
01049C  4EB9 0001063A         JSR ANZEI
0104A2  6000 FEE6              BRA TAS
0104A6  0C00 0046      ST24:  CMP.B #'F',D0
0104AA  6600 0032              BNE ST25
0104AE  06B9 00000005         ADD.L #5,PX
0104B4  00010008

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 6

0104B8  06B9 0000000A         ADD.L #10,SX
0104BE  00010018
0104C2  2239 00010018         MOVE.L SX,D1
0104C8  2439 0001001C         MOVE.L SY,D2
0104CE  4EB9 0000131A         JSR $SET
0104D4  4EB9 0001063A         JSR ANZEI
0104DA  6000 FEAE              BRA TAS
0104DE  0C00 0053      ST25:  CMP.B #'S',D0
0104E2  6600 0032              BNE ST26
0104E6  04B9 00000001         SUB.L #1,PX
0104EC  00010008
0104F0  04B9 00000002         SUB.L #2,SX
0104F6  00010018
0104FA  2239 00010018         MOVE.L SX,D1
010500  2439 0001001C         MOVE.L SY,D2
010506  4EB9 0000131A         JSR $SET
01050C  4EB9 0001063A         JSR ANZEI
010512  6000 FE76              BRA TAS
010516  0C00 0041      ST26:  CMP.B #'A',D0
01051A  6600 0034              BNE ST27
01051E  04B9 00000005         SUB.L #5,PX
010524  00010008
010528  04B9 0000000A         SUB.L #10,SX
01052E  00010018
010532  2239 00010018         MOVE.L SX,D1
010538  2439 0001001C         MOVE.L SY,D2
01053E  4EB9 0000131A         JSR $SET
010544  4EB9 0001063A         JSR ANZEI
01054A  4EB9 0001038A         JSR TAS
010550  0C00 005A      ST27:  CMP.B #'Z',D0   *ZICHNEN UND KOORDINATEN SPEICHERN
010554  6600 0062              BNE ST28
010558  4282                   CLR.L D2
01055A  2239 00010020         MOVE.L XA,D1
010560  2439 00010024         MOVE.L YA,D2
010566  4EB9 00000EE2         JSR $MOVETO
01056C  4282                   CLR.L D2
01056E  2239 00010008         MOVE.L PX,D1
010574  92B9 00010010         SUB.L EX,D1
01057A  C3FC 0002              MULS.L #2,D1
```

```
01057E  2439 0001000C          MOVE.L PY,D2
010584  94B9 00010014          SUB.L EY,D2
01058A  4EB9 00000F4C          JSR $DRAWTO
010590  23C1 00010020          MOVE.L D1,XA
010596  23C2 00010024          MOVE.L D2,YA
01059C  2239 00010018          MOVE.L SX,D1
0105A2  2439 0001001C          MOVE.L SY,D2
0105A8  4EB9 0000131A          JSR $SET
0105AE  4EB9 000105F2          JSR SPEICHERN
0105B4  6000 FDD4              BRA TAS
0105B8  0C00 0055       ST28:  CMP.B #'U',D0
0105BC  6600 0016              BNE ST29
0105C0  4EB9 00000D6E          JSR $ERAPEN
0105C6  23FC 00000000          MOVE.L #0,STIFT
0105CC  00010030
0105D0  6000 FDB8              BRA TAS
0105D4  0C00 004D       ST29:  CMP.B #'M',D0
0105D8  6600 FDB0              BNE TAS
0105DC  4EB9 0000110A          JSR $GRAPOFF
0105E2  4EB9 00001112          JSR $HIDE
0105E8  4EB9 0001009E          JSR MENUE
0105EE  6000 FD9A              BRA TAS
0105F2                 SPEICHERN:              *ZEICHNEN UND KOORDINATEN ABSPEICHERN
0105F2  42B1                   CLR.L D1
0105F4  2079 0001002C          MOVEA.L KOOP,A0 *WERT KOORDINATENVERZEICHNIS-ANFANG

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 7

0105FA  06B9 00000001          ADD.L #1,ANZAHL *ZÄHLEN DER GESPEICHERTEN KOORDINATEN
010600  00010028
010604  2239 00010008          MOVE.L PX,D1    *PX NACH D1
01060A  2141 0000              MOVE.L D1,X1(A0)*D1 NACH KOORDINATENVERZEICHNIS
01060E  2239 0001000C          MOVE.L PY,D1    *PY NACH D1
010614  2141 0004              MOVE.L D1,Y1(A0)*D1 NACH KOORDINATENVERZEICHNIS
010618  2239 00010030          MOVE.L STIFT,D1 *STIFT NACH D1
01061E  2141 0008              MOVE.L D1,ST(A0)*D1 NACH ST
010622  23FC 00000001          MOVE.L #1,STIFT *STIFT AUF ZEICHNEN SETZEN
010628  00010030
01062C  D1FC 0000000C          ADDA.L #KOOR,A0 *WERT KOORDINATENVERZEICHNIS ERHÖHEN
010632  23C8 0001002C          MOVE.L A0,KOOP
010638  4E75                  RTS
01063A
01063A  103C 0011       ANZEI: MOVE.B #$11,D0  *KOORDINATENANZEIGE
01063E  323C 005A              MOVE #90,D1
010642  343C 00F5              MOVE #245,D2
010646  41F9 00010B3D          LEA TEXI2,A0    * X= UND Y=  WERT
01064C  4EB9 000013B6          JSR $WRITE
010652  41F9 00010BD3          LEA BUFX,A0     *TEXT AUF BILDSCHIRM
010658  2039 00010008          MOVE.L PX,D0
01065E  4EB9 000015FC          JSR $PRINT4D
010664  41F9 00010BD3          LEA BUFX,A0
01066A  103C 0011              MOVE.B #$11,D0
01066E  323C 006E              MOVE #110,D1
010672  343C 00F5              MOVE #245,D2
010676  4EB9 000013B6          JSR $WRITE
01067C  41F9 00010BD7          LEA BUFY,A0
010682  2039 0001000C          MOVE.L PY,D0
010688  4EB9 000015FC          JSR $PRINT4D
01068E  41F9 00010BD7          LEA BUFY,A0
010694  103C 0011              MOVE.B #$11,D0
010698  323C 00BE              MOVE #190,D1
01069C  343C 00F5              MOVE #245,D2
0106A0  4EB9 000013B6          JSR $WRITE
0106A6  4E75                   RTS
0106A8  4EB9 00000DA0  AUSGABE:JSR $CLR        *AUSGABE
0106AE  103C 0044              MOVE.B #$44,D0
0106B2  323C 0078              MOVE #120,D1
0106B6  343C 00C8              MOVE #200,D2
0106BA  41F9 00010B5B          LEA TEX30,A0
0106C0  4EB9 000013B6          JSR $WRITE
0106C6  103C 0033              MOVE.B #$33,D0
0106CA  323C 0078              MOVE #120,D1
0106CE  343C 0096              MOVE #150,D2
0106D2  41F9 00010B60          LEA TEX31,A0
0106D8  4EB9 000013B6          JSR $WRITE
0106DE  323C 0078              MOVE #120,D1
0106E2  343C 0078              MOVE #120,D2
0106E6  41F9 00010B6D          LEA TEX32,A0
0106EC  4EB9 000013B6          JSR $WRITE
0106F2  323C 0078              MOVE #120,D1
0106F6  343C 005A              MOVE #90,D2
0106FA  41F9 00010B7A          LEA TEX33,A0
010700  4EB9 000013B6          JSR $WRITE
010706
010706  4EB9 00000A00   ZUR:  JSR $CI
01070C  0C00 0031              CMP.B #'1',D0
010710  6606                   BNE.S ST30
010712  4EB9 00010732          JSR MONITOR     *ZUR BILDSCHIRMAUSGABE
010718  0C00 0032       ST30:  CMP.B #'2',D0
01071C  6606                   BNE.S ST31
01071E  4EB9 00010BF4          JSR LINIE       *ZUR PLOTTERAUSGABE
010724  0C00 0033       ST31:  CMP.B #'3',D0

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 8

010728  66DC                   BNE.S ZUR
01072A  4EB9 000107AE          JSR JUSTIEREN   *ZUM JUSTEIREN
010730  60D4                   BRA.S ZUR
010732
010732  4EB9 00000DA0  MONITOR:JSR $CLR
010738  223C 00000000          MOVE.L #0,D1
01073E  243C 00000000          MOVE.L #0,D2
010744  4EB9 00000EE2          JSR $MOVETO
01074A  41F9 00010BDC          LEA KOORD,A0
010750  2A39 00010028          MOVE.L ANZAHL,D5
010756  0CB5 00000000          CMP.L #0,D5     *FESTSTELLEN OB KOORDINATEN VORHANDEN
01075C  6700 F940              BEQ MENUE
010760  0485 00000001          SUB.L #1,D5
010766  2228 0000       WID:   MOVE.L X1(A0),D1
01076A                 *       DIVS #0,D1      *GRÖßE DER BILDSCHIRMAUSGABE
01076A  2428 0004              MOVE.L Y1(A0),D2
01076E  85FC 0002              DIVS #2,D2      *GRÖßE DER BILDSCHIRMAUSGABE
010772  202B 0008              MOVE.L ST(A0),D0
010776  0C80 00000001          CMP.L #1,D0
01077C  6700 000E              BEQ MARKE4
010780  4EB9 00000D6E          JSR $ERAPEN
010786  4EB9 00010792          JSR MARKE5
01078C  4EB9 00000D7C  MARKE4: JSR $SETPEN
010792  4EB9 00000F4C  MARKE5: JSR $DRAWTO
010798  D1FC 0000000C          ADDA.L #KOOR,A0
01079E  51CD FFC6              DBRA D5,WID
0107A2  4EB9 00010AEA          JSR SCHLEIFE
0107A8  4EB9 0001009E          JSR MENUE
0107AE
0107AE                 JUSTIEREN:              *PLOTTERSTIFT POSITIONIEREN
0107AE  4EB9 00000DA0          JSR $CLR
0107B4  303C 0022              MOVE #$22,D0
0107B8  323C 0078              MOVE #120,D1
0107BC  343C 0096              MOVE #150,D2
0107C0  41F9 00010B87          LEA TEX34,A0
0107C6  4EB9 00001386          JSR $WRITE
0107CC  323C 0078              MOVE #120,D1
0107D0  343C 0078              MOVE #120,D2
0107D4  41F9 00010B94          LEA TEX35,A0
0107DA  4EB9 00001386          JSR $WRITE
0107E0  323C 0078              MOVE #120,D1
0107E4  343C 005A              MOVE #90,D2
0107E8  41F9 00010BA1          LEA TEX36,A0
0107EE  4EB9 00001386          JSR $WRITE
0107F4  323C 0078              MOVE #120,D1
0107F8  343C 003C              MOVE #60,D2
0107FC  41F9 00010BAE          LEA TEX37,A0
010802  4EB9 00001386          JSR $WRITE
010808  323C 0078              MOVE #120,D1
01080C  343C 001E              MOVE #30,D2
010810  41F9 00010BBB          LEA TEX38,A0
010816  4EB9 00001386          JSR $WRITE
01081C  4EB9 00000A00  JUS:    JSR $CI
010822  0C00 0045              CMP.B #'E',D0
010826  662A                   BNE.S ST40
010828  3A3C 0009              MOVE #10-1,D5   *STIFTSTEUERUNG HOCH
01082C  303C 0020       NO1:   MOVE #32,D0
010830  4EB9 00000D28          JSR $LO
010836  303C 0000              MOVE #0,D0
01083A  4EB9 00000D28          JSR $LO
010840  303C 0001              MOVE #1,D0
010844  4EB9 00000D28          JSR $LO
01084A  51CD FFE0              DBRA D5,NO1
01084E  6000 FFCC              BRA JUS

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 9

010852  0C00 0058       ST40:  CMP.B #'X',D0
010856  662A                   BNE.B ST41
010858  3A3C 0009              MOVE #10-1,D5
01085C  303C 0028       NO2:   MOVE #40,D0     *STIFTSTEUERUNG TIEF
010860  4EB9 00000D28          JSR $LO
010866  303C 0000              MOVE #0,D0
01086A  4EB9 00000D28          JSR $LO
010870  303C 0001              MOVE #1,D0
010874  4EB9 00000D28          JSR $LO
01087A  51CD FFE0              DBRA D5,NO2
01087E  6000 FF9C              BRA JUS
010882  0C00 0044       ST41:  CMP.B #'D',D0
010886  662A                   BNE.B ST42
010888  3A3C 0009              MOVE #10-1,D5   *STIFTSTEUERUNG RECHTS
01088C  303C 0004       NO3:   MOVE #4,D0
010890  4EB9 00000D28          JSR $LO
010896  303C 0000              MOVE #0,D0
01089A  4EB9 00000D28          JSR $LO
0108A0  303C 0001              MOVE #1,D0
0108A4  4EB9 00000D28          JSR $LO
0108AA  51CD FFE0              DBRA D5,NO3
0108AE  6000 FF6C              BRA JUS
0108B2  0C00 0053       ST42:  CMP.B #'S',D0
0108B6  662A                   BNE.B ST43
0108B8  3A3C 0009              MOVE #10-1,D5   *STIFTSTEUERUNG LINKS
0108BC  303C 0005       NO4:   MOVE #5,D0
0108C0  4EB9 00000D28          JSR $LO
0108C6  303C 0000              MOVE #0,D0
0108CA  4EB9 00000D28          JSR $LO
0108D0  303C 0001              MOVE #1,D0
0108D4  4EB9 00000D28          JSR $LO
0108DA  51CD FFE0              DBRA D5,NO4
0108DE  6000 FF3C              BRA JUS
0108E2  0C00 0041       ST43:  CMP.B #'A',D0
0108E6  6600 FF34              BNE JUS
0108EA  4EB9 000106A8          JSR AUSGABE     *ZUR AUSGABE
0108F0  6000 FF2A              BRA JUS
```

```
0108F4  4279 00010034   LINIE:  CLR XAP
0108FA  4279 00010038            CLR YAP
010900  4279 00010044            CLR PXS
010906  4279 00010048            CLR PYS
01090C  4279 00010058            CLR XSCHRITT
010912  4279 0001005C            CLR YSCHRITT
010918  4286                     CLR.L D6
01091A  41F9 00010BDC            LEA KOORD,A0     *KOODINATEN LADEN
010920  2A39 00010028            MOVE.L ANZAHL,D5
010926  0485 00000001            SUB.L #1,D5
01092C  2628 0000        WDH:    MOVE.L X1(A0),D3*XE-KOORDINATE
010930  C7FC 0003                MULS #3,D3       *VERGRößERUNGSFAKTOR FüR DIE PLOTTERAUSGABE
010934  23C3 0001003C            MOVE.L D3,XE
01093A  2828 0004                MOVE.L Y1(A0),D4*YE-KOORDINATE
01093E  C9FC 0003                MULS #3,D4       *VERGRößERUNGSFAKTOR FüR DIE PLOTTERAUSGABE
010942  23C4 00010040            MOVE.L D4,YE
010948  2028 0008                MOVE.L ST(A0),D0*ST-WERT
01094C  C1FC 0080                MULS #128,D0
010950  23C0 0001004C            MOVE.L D0,STIFTP
010956                           *DELDA X UND Y BERECHNEN
010956  96B9 00010034            SUB.L XAP,D3     *XE-XAP=DELTA X
01095C  0C83 00000000            CMP.L #0,D3      *FESTSTELLEN OB DELTA X NEGATIV
010962  6B00 0134                BMI LINKS        *NEGATIV
010966  6A00 0116                BPL RECHTS       *POSITIF
01096A
01096A  98B9 0001003B   DELY:    SUB.L YAP,D4     *YE-YAP=DELTA Y
010970  0C84 00000000            CMP.L #0,D4      *FESTSTELLEN OB DELTA Y POSITIV
```



```
010976  6B00 0156                BMI TIEF         *NEGATIV
01097A  6A00 0138                BPL HOCH         POSITIV
01097E
01097E  4280             STAR2:  CLR.L D0
010980  B684                     CMP.L D4,D3      *PRüFEN OB DELTA X ODER DELTA Y GRößER
010982  6E00 0054                BGT DMI
010986
010986  9C83             DPL:    SUB.L D3,D6      *KLEINER
010988  2039 0001005C            MOVE.L YSCHRITT,D0
01098E  D1B9 00010048            ADD.L D0,PYS
010994  2039 0001004C            MOVE.L STIFTP,D0
01099A  4EB9 00000D28            JSR §LO
0109A0  2039 00010054            MOVE.L SENKR,D0
0109A6  4EB9 00000D28            JSR §LO
0109AC  203C 00000000            MOVE.L #0,D0
0109B2  4EB9 00000D28            JSR §LO
0109B8  203C 00000001            MOVE.L #1,D0
0109BE  4EB9 00000D28            JSR §LO
0109C4  2039 0001004C            MOVE.L STIFTP,D0
0109CA  4EB9 00000D28            JSR §LO
0109D0  4280                     CLR.L D0
0109D2  4EF9 00010A24            JMP TEST
0109D8  DC84             DMI:    ADD.L D4,D6      *GRößER
0109DA  2039 00010058            MOVE.L XSCHRITT,D0
0109E0  D1B9 00010044            ADD.L D0,PXS
0109E6  2039 0001004C            MOVE.L STIFTP,D0
0109EC  4EB9 00000D28            JSR §LO
0109F2  2039 00010050            MOVE.L WAAGR,D0
0109F8  4EB9 00000D28            JSR §LO
0109FE  203C 00000000            MOVE.L #0,D0
010A04  4EB9 00000D28            JSR §LO
010A0A  203C 00000001            MOVE.L #1,D0
010A10  4EB9 00000D28            JSR §LO
010A16  2039 0001004C            MOVE.L STIFTP,D0
010A1C  4EB9 00000D28            JSR §LO
010A22  4280                     CLR.L D0
010A24
010A24  0C86 00000000   TEST:    CMP.L #0,D6
010A2A  6700 000A                BEQ ZWEIG        *D=0
010A2E  6E00 FF56                BGT DPL          *D>0 NACH DPL
010A32  6D00 FFA4                BLT DMI          *D<0 NACH DMI
010A36
010A36  2239 00010044   ZWEIG:   MOVE.L PXS,D1
010A3C  2439 0001003C            MOVE.L XE,D2
010A42  B481                     CMP.L D1,D2      *PXS^XE STAR2
010A44  6600 FF38                BNE STAR2
010A48  2239 00010048            MOVE.L PYS,D1
010A4E  2439 00010040            MOVE.L YE,D2
010A54  B481                     CMP.L D1,D2
010A56  6600 FF26                BNE STAR2        *PYS^YE STAR2
010A5A  23F9 0001003C            MOVE.L XE,XAP    *X-E UND YE NACH X-A UND Y-A VERSCHIEBEN
010A60  00010034
010A64  23F9 00010040            MOVE.L YE,YAP
010A6A  00010038
010A6E  D1FC 0000000C            ADDA.L #KOOR,A0
010A74  51CD FEB6                DBRA D5,WDH
010A78  4EB9 0001009E            JSR MENUE
010A7E
010A7E                   **** UNTERPROGRAMM STIFTSTEUERUNG*******
010A7E  23FC 00000004   RECHTS:  MOVE.L #4,WAAGR
010A84  00010050
010A88  23FC 00000001            MOVE.L #1,XSCHRITT
010A8E  00010058
010A92  4EF9 0001096A            JMP DELY
```



```
010A98  23FC 00000005   LINKS:   MOVE.L #5,WAAGR
010A9E  00010050
010AA2  23FC FFFFFFFF            MOVE.L #-1,XSCHRITT
010AA8  00010058
010AAC  4483                     NEG.L D3
010AAE  4EF9 0001096A            JMP DELY
010AB4  23FC 00000020   HOCH:    MOVE.L #32,SENKR
010ABA  00010054
010ABE  23FC 00000001            MOVE.L #1,YSCHRITT
010AC4  0001005C
010AC8  4EF9 0001097E            JMP STAR2
010ACE  23FC 00000028   TIEF:    MOVE.L #40,SENKR
010AD4  00010054
010AD8  23FC FFFFFFFF            MOVE.L #-1,YSCHRITT
010ADE  0001005C
010AE2  4484                     NEG.L D4
010AE4  4EF9 0001097E            JMP STAR2
010AEA                   SCHLEIFE:
010AEA  363C 0063                MOVE #100-1,D3
010AEE  4EB9 00000E38   WARTE:   JSR §SYNC
010AF4  6700 FFF8                BEQ WARTE
010AF8  51CB FFF4                DBRA D3,WARTE
010AFC  4E75                     RTS
010AFE
```

```
010AFE  43414400         TEX1:DC.B 'CAD',0
010B02  4D454EDD00       TEX2:DC.B 'MENü',0
010B07  45494E47414245   TEX3:DC.B 'EINGABE      1',0
010B0E  20202020202031
010B15  00
010B16  41555347414245   TEX5:DC.B 'AUSGABE      2',0
010B1D  20202020202032
010B24  00
010B25
010B25  45494E47414245   TEX11:   DC.B 'EINGABE BILDAUSSCHNITT ',0
010B2C  2042494C444155
010B33  535343484E4954
010B3A  542000
010B3D  583D2020202020   TEX12:   DC.B 'X=       UND Y=       WERT',0
010B44  2020554E442059
010B4B  3D2020202020
010B52  205745525400
010B58  41555347414245   TEX30:   DC.B 'AUSGABE',0
010B5F  00
010B60  42494C44534348   TEX31:   DC.B 'BILDSCHIRM 1',0
010B67  49524D203100
010B6D  504C4F54544552   TEX32:   DC.B 'PLOTTER    2',0
010B74  202020203200
010B7A  4A5553544945 52  TEX33:   DC.B 'JUSTIERUNG 3',0
010B81  554E47203300
```

```
010B87  52454348545320   TEX34:   DC.B 'RECHTS     D',0
010B8E  202020204400
010B94  4C494E4B532020   TEX35:   DC.B 'LINKS      S',0
010B9B  202020205300
010BA1  484F4348202020   TEX36:   DC.B 'HOCH       E',0
010BA8  202020204500
010BAE  54494546202020   TEX37:   DC.B 'TIEF       X',0
010BB5  202020205800
010BBB  41555347414245   TEX38:   DC.B 'AUSGABE    A',0
010BC2  202020204100
010BC8
010BC8  02 02 00 00 00   TABELLE: DC.B 2,2,0,0,0,6,6,4,4,4,10
010BCD  06 06 04 04 04
010BD2  0A
010BD3
```



```
010BD3                   BUFX: DS.B 4
010BD7                   BUFY: DS.B 4
010BDB  00               DS.W 0
010BDC                   KOORD:
010BDC
010BDC                   END

009558  Ende-Symboltabelle
```

- ges aktuell - ges aktuell - ges aktuell -

ges aktuell

Seit 1. 9.: Wesentliche Preisreduzierungen bei GES-Produkten!

Teilweise erhebliche Preisreduzierungen wurden beim Elektronikladen Detmold und bei uns vorgenommen. So kostet die ACRTC-Bau-Gruppe nun DM 1.300,– (früher DM 2.052,–), COL256B DM 448,– (598,–), CPU68KB DM 139,– (199,–), FL02B DM 258,– (389,–), POW26/22N DM 99,–(159,–), SBC2B DM 59,– (89,–). Wesentlich günstiger wurde auch der mc-CP/M-Computer: SYS1B DM 198,– (398,–), TERM166B DM 398,– (498,–). Dies nur einige Beispiele aus über 100 Preiskorrekturen. Neuen Katalog und Preisliste anfordern – es lohnt sich!

ges aktuell

Texte formatieren mit dem 68008

von Olaf Reinhold, 3300 Braunschweig

Das folgende Programm formatiert Texte, die mit dem Texteditor erstellt wurden. Die Adresse des zu formatierenden Textes ist dieselbe, die für den Editor eingestellt ist.

Durch setzen eines Dummy (ASCII Code 40h) in einer Zeile, wird diese Zeile nicht formatiert. Steht als erstes Zeichen in einer Zeile ein Dummy, wird diese Zeile als Leerzeile ausgeben.

Nach dem Start des Programmes (gestartet wird mit „Start"), muß die Anzahl der Zeichen pro Zeile eingegeben werden. Nach dem Formatieren, was relativ schnell vor sich geht, wird gleich der Editor aufgerufen und das Ergebnis ist jetzt auf dem Bildschirm sichtbar.

Es muß jedoch auf einen genügend großen Speicher geachtet werden. Ist auf Adresse $A000 bzw. $EA000 kein RAM-Speicher mehr vorhanden, darf der zu formatierende Text max. 4KByte groß sein.

```
org $20000
stxtxt equ $36
akttxt equ $3a
etxtxt equ $3e
eottxt equ $42
txtspal equ $fff
letztzei equ $ffe
leerzei equ $ffd
zeichein equ $ffc
rest equ $ffb
zustaby equ $ffa
buffer equ $ff7
count equ $ff6
leer equ $ff5
schirm equ $1000

start:
        clr.b txtspal(a5)
        clr.b letztzei(a5)
        clr.b leerzei(a5)
        clr.b zustaby(a5)
        clr.b count(a5)          ;löschen aller Parameter
        movea.l stxtxt(a5),a0    ;Aktuelle Text Adr. laden
        lea schirm(a5),a1        ;Zwischenspeicher laden
formcopy:
        move.b (a0)+,d0
        beq.s fertig
        move.b d0,(a1)+          ;Text in Zwischenspeicher
        bra.s formcopy           ;weiter bis Null
fertig:
        clr.b (a1)
        lea buffer(a5),a0
        move #$21,d0
        move #100,d1
        move #100,d2
        move #2,d3
        jsr @read                ;lese Anzahl der Zeichen auf die
        lea buffer(a5),a0        ;formatiert werden soll
        jsr @wert                ;umrechnen

        move.b d0,txtspal(a5)    ;Wert abspeichern
        movea.l stxtxt(a5),a2    ;Formatierter Text in Zieladr.ablegen
        lea schirm(a5),a0
format1:
        movea.l a0,a1            ;Adresse merken
        clr d4
        move.b txtspal(a5),d4
        move d4,d5
        subq #1,d5               ;Schleife bestimmen
forleer:
        addq.b #1,count(a5)
forleer1:
        move.b (a0)+,d0          ;Zeichen = 0?
        beq format2
        cmp.b #' ',d0            ;Leerzeichen gefunden?
        beq formleer
        cmp.b #$d,d0             ;Zeilenumbruch?
        beq ctrld
        cmp.b #'@',d0            ;Dummy?
        beq dummy1
        bclr #2,zustaby(a5)
        dbra d5,forleer          ;Nein,dann nächstes Zeichen
format2:
        bclr #2,zustaby(a5)
        movea.l a1,a0            ;Adresse zurück
        clr.b zeichein(a5)
        clr d2
        clr.l d1
        move.b txtspal(a5),d1    ;Anzahl Zeichen pro Zeile
        sub.b letztzei(a5),d1    ;Minus dem letzten gefundenen Zeichen
        move.b leer(a5),d2
        beq keinnull             ;Kein Leerzeichen vorhanden?
format21:
        divs d2,d1               ;dividiere Anzahl Leerzeichen durch Anzahl
        move.b d1,zeichein(a5)   ;der übriggebliebenen Zeichen u. speichern
        swap d1                  ;des ganzzahligen Teils
        tst.b d1
        beq keinrest
        bset #1,zustaby(a5)      ;wenn Rest vorhanden auch abspeichern
        move.b d1,rest(a5)
        swap d1
format3:                         ;hier beginnt zurückspeichern einer for-
        clr d4                   ;matierten Zeile an alte Adresse
        move.b txtspal(a5),d4
        move d4,d5
        subq #1,d5
format4:
        move.b (a0)+,d0          ;bis Null gefunden,dann neue Zeile
        beq fin
        cmp.b #' ',d0            ;Leerzeichen? dann evtl. Leerzeichen einf.
        beq einleer
        cmp.b #$d,d0             ;CR? dann auch Leerzeichen einf.
        beq ctrlleer
        bclr #2,zustaby(a5)
format41:
        move.b d0,(a2)+          ;und zurückkopieren
        dbra d5,format4          ;neues Zeichen

format42:
        move.b #$d,(a2)+         ;am Ende CR
        move.b #$a,(a2)+         ;und line feet

        clr.b letztzei(a5)
        clr.b leerzei(a5)
        clr.b zustaby(a5)
        clr.b count(a5)          ;löschen der Parameter für neue Zeile
format11:
        move.b (a0),d0           ;Null? dann Ende
        beq fin
        cmp.b #' ',d0            ;Leerzeichen? dann neues Zeichen
        beq format12
        cmp.b #$d,d0             ;CR? dann neues Zeichen
        beq format12
        cmp.b #$a,d0             ;Line feet? dann neues Zeichen
        beq format12
        bra format1
format12:
        addq# 1,a0               ;einfach nächstes Zeichen
        bra format11             ;nächstes Zeichen testen
fin:
        clr.b (a2)
        movea.l stxtxt(a5),a0
        move.l a0,akttxt(a5)
null:
        tst.b (a0)+
        bne null
        subq.l #1,a0
        move.l a0,etxtxt(a5)
        move.l a0,eottxt(a5)     ;sämtlich Text Adressen neu definieren
        jsr @edit                ;und Editor aufrufen
        rts                      ;Programm Ende
ctrld:
        addq #1,a0               ;Bei CR einfach näschtes Zeichen
formleer:                        ;und dafür als Leerzeichen verarbeiten
        btst #2,zustaby(a5)
        bne forleer1
        move.b leerzei(a5),leer(a5)
        addq.b #1,leerzei(a5)    ;Anzahl der Leerzeichen +1
        move.b count(a5),d1
        subq.b #1,d1
        move.b d1,letztzei(a5)   ;letztes Zeichen in einer Zeile
        bset #2,zustaby(a5)
        dbra d5,forleer
        bra format2

keinrest:
        bclr #1,zustaby(a5)      ;aus Division kein Rest, daher nicht speicher
        swap d1
        bra format3
ctrlleer:
        addq #1,a0
        move.b #' ',d0           ;für CR einfach ' 'einsetzen
einleer:
        btst #2,zustaby(a5)
        bne format4
        clr d1
        move.b zeichein(a5),d1
        beq leer3
        move d1,d4
        subq #1,d4
leer2:
        move.b #' ',(a2)+        ;bei jedem Leerzeichen evtl. weiteres Leer-
        subq #1,d5               ;zeichen einsetzen
        dbra d4,leer2
leer3:
        bset #2,zustaby(a5)
        btst #1,zustaby(a5)
        beq leerend
        move.b rest(a5),d1
        beq restende             ;bei Rest aus Division auch weitere Leerzeich
        move.b #' ',(a2)+        ;einfügen
        subq.b #1,rest(a5)       ;reduziert Rest um -1
        subq #1,d5
leerend:
        bra format41

restende:
        bclr #1,zustaby(a5)      ;kein Rest mehr da
        bra format41

keinnull:
        move.b #1,d2             ;für Division, da evtl. Null sein kann
        bra format21

dummy1:
        movea.l a1,a0            ;kein Formatieren dieser Zeile
dummy2:
        move.b (a0)+,d0
        cmp.b #'@',d0
        beq format42
        move.b d0,(a2)+          ;einfach Zeile unformatiert zurückschreiben
        bra.s dummy2

        end
```

68000 — YOGIDOS UND JADOS, RL-BASIC

Leistungszuwachs bei Jados

von Klaus Janßen

Seit einem halben Jahr steht für die Freunde des NDR-Klein-Computers ein leistungsfähiges Diskettenbetriebssystem für den 680xx-Ausbau zur Verfügung, das zudem äußerst preisgünstig angeboten wird: JADOS in der bisher aktuellen Version 1.2a. In der Bedienung ähnelt es CP/M 68k und eignet sich besonders gut für die Entwicklung kleinerer und mittlerer Programme in Assembler sowie zur Textverarbeitung mit dem eingebauten Bildschirmeditor.

Der recht gute Markterfolg und die zahlreichen Verbesserungsvorschläge kompetenter Entwickler bewogen mich, die aktuelle JADOS-Version 1.2a hinsichtlich Schnelligkeit und Bedienungskomfort gründlich zu überarbeiten. Oberstes Gebot war dabei, vollständig kompatibel zu den vorigen Versionen zu bleiben. Mit der neuen Version 2.0 steht nun den Freunden des NDR-Klein-Computers ein professionelles Betriebssystem zur Verfügung, das mit allen Prozessoren der 68000-Familie arbeitet.

Der Startvorgang des Systems ist wesentlich bedienungsfreundlicher geworden. Mehrfachinstallationen von JADOS oder des Variablenspeichers können nicht mehr auftreten. Neu ist ein sogenannter Autostartmechanismus. Bei jedem kompletten Ladevorgang des Betriebssystems startet JADOS die Datei „AUTOEXEC.BAT“, sofern sie existiert. Dies ermöglicht es, nach dem Bootvorgang Systeminitialisierungen oder ein Anwendungsprogramm automatisch zu starten.

Beeindruckend an der neuen JADOS-Version ist die Schnelligkeit, mit der Dateien gelesen und geschrieben werden. Bei der Bearbeitung kompletter Dateien wird die Spurtabelle jetzt im Hauptspeicher verwaltet. Dadurch muß das System nicht ständig auf die Directoryspur positionieren, was zum einen der Diskettenmechanik zugute kommt und zum anderen die Lade- und Speicherzeit kompletter Dateien wesentlich reduziert. Lade- und Speichervorgänge benötigen jetzt nur noch etwa die Hälfte der Zeit, während das Speichern neuer Dateien sogar 4 - 6 mal schneller erfolgt.

Für die Eingabe von Kommandos und Dateinamen stehen nun Puffer zur Verfügung, die jederzeit mit der Tastenkombination Ctrl J bzw. Linefeed abgerufen werden können. Das bisher nötige wiederholte Eingeben desselben Dateinamens entfällt somit.

An jeder beliebigen Stelle, an der eine Tastatureingabe möglich ist, kann nun mit Ctrl P eine Hardcopy des Textinhalts des Bildschirms auf den Drucker veranlaßt werden. Dazu ist keine Hardcopy-Baugruppe notwendig. Graphik kann damit allerdings nicht ausgedruckt werden.

Alle internen Kommandos können nun wahlweise mit Parametern aufgerufen oder wie bisher menügesteuert bedient werden. Fehlende Parameter erfragt JADOS automatisch. Viele neue Kommandos sind hinzugekommen:

- BELL zur Erzeugung eines Tonsignals mit Hilfe der Baugruppe SOUND
- CLS zum Löschen des Bildschirms
- FF zum Erzeugen eines Seitenvorschubs auf dem Drucker
- PRINT zum Drucken von Textdateien mit automatischem Seitenvorschub.

Die Anzeige des Inhaltsverzeichnisses erfolgt nun in drei Spalten, so daß fast 70 Dateien auf einer Bildschirmseite dargestellt werden können. Zudem können jetzt Dateigruppen selektiert werden, z.B.:

- DIR *.68K selektiert alle Dateien vom Typ 68K
- DIR 2:T*.TXT selektiert alle Dateien auf Laufwerk 2, deren Name mit T beginnt und die vom Typ TXT sind.

```
                INHALTSVERZEICHNIS VON LAUFWERK 2
--NAME-- TYP  LAENGE     --NAME-- TYP  LAENGE     --NAME-- TYP  LAENGE
FORMAT00 ASM  10         BRDRUCK  ASM  15         BRDRUCK  68K  3
DRTEST        1          FCOPY    ASM  7          FCOPY    68K  2
ROMSTART ASM  6          ROMSTART 68K  1          APFELMAN ASM  2
PRTELITE 68K  1          APFELMAN 68K  1          GDEMO1   ASM  1
GDEMO1   68K  1          REVERSI  ASM  22         PRTELITE ASM  1
REVERSI  68K  5          REVTOOL  ASM  14         BEDIENH  DOC  13
DSAVE    ASM  4          DSAVE    68K  1          REVERSI  DOC  13
FORMAT20 ASM  10         RDKGRUND 43   32         FLOPTEST ASM  1
FLOPTEST 68K  1          SOUND    ASM  1          SOUND    68K  1
RHYTHMUS ASM  1          RHYTHMUS 68K  1          KLAVIER  ASM  3
KLAVIER  68K  1          UFORM68  68K  8          FORMAT00 68K  2
UTILSAVE BAT  1          FORMAT20 68K  2          FORMAT08 ASM  10
FORMAT08 68K  2          ROMCAL08 ASM  2          ROMCAL08 68K  1
LIESMICH      7          MORE     A68  9          HC       ASM  7
FORMAT   ASM  12         FORMAT   68K  2          1COPY    ASM  6
1COPY    68K  2          AUSGABE  ASM  6          AUSGABE  68K  2
MORE     ASM  9          MORE     68K  2          HC       68K  2

  ANZAHL DATEIEN:  51      ANZAHL SEKTOREN: 270
  FREIE SEKTOREN:  510     FREIE SPUREN:    72

2>
Kommandos: ↑A ↑C ASS COPY DIR EDIT ERA REN SYS TLOAD TSAVE TYPE 1: 2:
```

Bild 1: Inhaltsverzeichnis (Beispiel)

Das eingebaute COPY-Kommando war bisher sehr langsam, so daß fast nur mit der Utility FCOPY gearbeitet wurde. Mit Version 2.0 ist COPY nun sogar fast zweimal schneller als das alte FCOPY.

Entwickler, die ein größeres Programm in mehrere Quellmodule unterteilt haben, mußten bisher beim ASS-Kommando alle Dateinamen von der Tastatur eingeben. Mit der neuen Version gibt es jetzt die Möglichkeit, die Eingabe zu automatisieren. Dazu erstellt der Programmierer eine normale Textdatei und schreibt alle Modulnamen hinein, die zum Programm gehören. Das ASS-Kommando bietet die Möglichkeit, statt der Modulnamen den Namen der Antwortdatei einzugeben, worauf alle Quellmodule in der Reihenfolge geladen und assembliert werden, in der sie in der Antwortdatei stehen. Bild 2 zeigt einen solchen automatischen Ladevorgang.

```
2>ass
A S S E M B L E R

!!  KEINE ORG- ODER OFFSETANWEISUNG BENUTZEN  !!
!!  STARTADRESSE IST KONFIGURATIONSABHANGIG   !!
!!  NORMAL: $C000     -  MIT BOOTKARTE: $0400  !!


QUELLPROGRAMME LADEN
WELCHE STARTADRESSE ? (DEFAULT = $001400 )$4000
 1 - TEXT LADEN
 2 - automatisch
 9 - beenden
NAME DER ANTWORTDATEI ? doslink.ans
DOSMAIN.ASM              00AEF9
DOSTABS.ASM              00CD56
DOSTOOL.ASM              012286
DOSFILE.ASM              018DC2
DOSBATCH.ASM             01AFF0
DOSGRUND.ASM             01BDF8
 1 - TEXT LADEN
 2 - automatisch
 9 - beenden
Kommandos: ↑A ↑C ASS COPY DIR EDIT ERA REN SYS TLOAD TSAVE TYPE 1: 2:
```

Bild 2: Automatischer Ladevorgang beim ASS-Kommando

Programmentwickler können nun auf eine Bibliothek von über 50 Unterprogrammen zugreifen. Neu hinzugekommen sind unter anderem:

- Unterprogramme für die Baugruppe SOUND
- Unterprogramme zum Verschieben von Speicherbereichen
- Unterprogramme zum Laden und Speichern binärer Dateien
- Unterprogramme zur Verarbeitung von
- Programmparametern
- Unterprogramme zum Setzen und Lesen eines Benutzercodes
- Unterprogramm zur Verkettung von Programmen

Das herausragende Leistungsmerkmal der neuen Version ist die sogenannte Stapelverarbeitung. Bisher konnten Kommandos nur über die Tastatur eingegeben werden. Mit der neuen Version ist nun ein Mechanismus verfügbar, der eine Reihe Kommandos, die in einer speziellen Textdatei abgelegt sind, automatisch abarbeitet. Immer wiederkehrende Kommandofolgen, z. B. bei einer Datensicherung, können auf diese Weise bequem und sicher ablaufen.

Die Kommandodateien sind ganz normale Textdateien, die mit dem Editor erstellt werden. Damit JADOS sie als Stapelverarbeitungsdateien erkennen kann, müssen sie die Extension .BAT besitzen.

Jede Textzeile darf ein JADOS-Kommando enthalten. Dabei dürfen sämtliche interne und externe Kommandos (externe Kommandos sind Namen von Programmen) verwendet werden. Lediglich beim Aufruf von Kommandodateien bestehen Einschränkungen. So darf nur das letzte Kommando einer Stapelverarbeitungsdatei wiederum eine Kommandodatei sein! Auf diese Weise können Stapelverarbeitungsdateien verkettet werden.

Die Kommandos werden der Reihe nach abgearbeitet, bis das letzte Kommando bearbeitet worden ist. Die Stapelverarbeitung kann durch einen beliebigen Tastendruck unterbrochen werden. JADOS fragt dann den Benutzer:

STAPELVERARBEITUNG ABBRECHEN (J/N) ?

Durch Eingabe von „J" wird die Stapelverarbeitung komplett beendet, ansonsten wird sie korrekt fortgeführt.

Einige Kommandos sind speziell für die Stapelverarbeitung hinzugekommen:

- CLS – Bildchirm löschen
- ECHO – Kommandos wegblenden oder anzeigen
- PAUSE – Stapelverarbeitung anhalten
- REM – Kommentar in Kommandodatei

Die in Bild 3 angegebene Kommandodatei, mit deren Hilfe eine Datensicherung aller Dateien, die zur Entwicklung von JADOS dienten, ausgeführt wird, möge abschließend als Beispiel einer nützlichen Anwendung der Stapelverarbeitung dienen.

```
bell
rem Quelle-in-1: und-Ziel-in-2:
pause
copy dosmain.asm 2:dosmain.200
copy dostabs.asm 2:dostabs.200
copy dostool.asm 2:dostool.200
copy dosfile.asm 2:dosfile.200
copy dosbatch.asm 2:dosbatch.200
copy dosgrund.asm 2:dosgrund.200
copy doslink.ans 2:doslink.ans
copy dossave.bat 2:dossave.bat
copy dosinit.asm 2:dosinit.200
copy dosinit.68k 2:dosinit.68k
copy jados200.68k 2:jados200.68k
bell
rem Ende des Backups
Textstart=000400 Fenster=000400 Tor=0004
```

Bild 3: Beipiel für eine Stapelverarbeitungsdatei

Die zusammen mit dem Betriebssystem gelieferten Hilfsprogramme wurden ebenfalls überarbeitet. Das Formatierprogramm gilt nun gemeinsam für alle CPUs. Neu ist das Programm MORE.68K, das eine sehr schnelle und bequeme Anzeige von Textdateien ermöglicht.

Die neue Version 2.0 von JADOS ist ab sofort erhältlich.

Schreibschutz und Jados

von Klaus Janßen

Einige Anwender berichteten mir, daß JADOS bei gesetztem Schreibschutz die Meldung „Fehler oder Schreibschutz gesetzt" und einen Fehlercode wie z.B. „2 2 3 33" meldet, obwohl nur lesend auf die Diskette zugegriffen wurde. Es handelt sich dabei um keinen besorgniserregenden Fehler, sondern um eine Unschönheit im JADOS. JADOS führt zu jeder Datei, auf die zugegriffen wird, einen Dateisteuerblock. Darin stehen Daten aus dem Disketteninhaltsverzeichnis (Director) sowie weitere aktuelle Werte. Wird die Datei wieder geschlossen, so schreibt JADOS die aktuellen Werte in das Inhaltsverzeichnis zurück, und zwar unabhängig davon, ob Schreib- oder Leseoperationen ausgeführt wurden. Bei gesetztem Schreibschutz kann dieses Zurückschreiben jedoch nicht ausgeführt werden, und es kommt zu obiger Fehlermeldung. Bei einer Versionsänderung wird dieses Problem von mir berücksichtigt. In der Zwischenzeit kann sich der Anwender damit behelfen, daß er entweder auf den Schreibschutz verzichtet – ich habe ihn noch nie wirklich benötigt – oder aber einfach die Fehlermeldung ignoriert.

Bei einem wirklichen Diskettenfehler ist der oben erwähnte Fehlercode recht interessant. JADOS benutzt die „Floppy"-Routine aus dem Grundprogramm. Bei einem Fehler meldet die Routine einen Wert ungleich Null im Register D0 zurück, worauf JADOS einen Wiederholungsversuch unternimmt. Schlägt auch dieser fehl, so wird der Fehlercode ausgegeben. Dabei wird der Inhalt der Register D1 bis D4 dezimal ausgegeben. Es bedeuten:

D1: 1 = lesender, 2 = schreibender Zugriff
D2: Aktueller physikalischer Sektor
D3: Aktuelle physikalische Spur
D4: Laufwerk und Dichtecode

Im obigen Beispiel „2 2 3 33" wurde also versucht, auf den 2. Sektor der Spur 3 des Minilaufwerks 1 zu schreiben. Auf Spur 3, Sektor 2 bis 5 befindet sich das Directory!

JADOS kostet

Bestell-Nr.	Format	DM
10292	5¼" 80 Sp.	140,–
10690	3½" 80 Sp.	140,–

Betrifft: Reparaturservice

Bei Rücksendungen von Laufwerken entsteht durch unsachgemäßes Verpacken oft ein größerer Schaden.

Achten Sie deshalb beim Verpacken nicht nur auf ausreichende Sicherung gegen Stoß und Druck. Das Laufwerk sollte unbedingt staubgeschützt in Plastiktüte und Karton (denken Sie an die Originalverpackung) verpackt sein, bevor Sie es zwischen Styropor-Chips oder angestaubtem Schaumstoff verstauen.

Zum Schutz der Schreib-/Lese-Köpfe muß entweder der Originalkarton oder eine Diskette in den Diskettenschlitz gesteckt sein und dieser verschlossen werden.

Sie helfen uns und vermeiden erhöhte Reparaturkosten, wenn Sie Ihre Laufwerke sorgfältig verpacken.

Farbhardcopy

von Willi Sicking

Schon seit längerer Zeit beschäftige ich mich in Gedanken mit dem Problem der Farbhardcopy. Daher habe ich mir auf verschiedenen Messen Farbdrucker angesehen. Bei den Vergleichen wurde mir schnell klar, daß in Bezug auf Druckqualität und Farbbrillianz für meine Anwendung nur ein Thermotransferdrukker in Frage kommt. So fiel die Wahl auf den ITHO TPX 80. Bei diesem Drucker wird, wie in dieser Kathegorie üblich, die Farbe vom Farbband mit 24 einzeln ansteuerbaren Temperaturelementen auf das Papier aufgeschmolzen. Eine Farbbandcassette reicht für 330 Druckzeilen, so daß man bei dem im folgenden beschriebenen Farbhardcopyprogramm pro Bild ca. 3,– DM an Farbbandkosten einkalkulieren muß. Ein großer Vorteil gegenüber Nadeldruckern liegt in dem geringen Betriebsgeräusch und dem Near-Letter-Quality-Druck (siehe Listing).

Als Programmiersprache wurde Modula2 der Firma p1 gewählt, da ein derart änderungs- und anpassungsintensives Programm in einer Hochsprache wesentlich besser zu handhaben ist als in Assembler. Die Routine PRINT war erforderlich, da in meiner M2-Version der Befehl WriteCar (CHR[0]) keine Ausgabe macht, alle übrigen M2UTIL-Routinen sind aus dem Programm CUTIL.S von Rolf-Dieter Klein entstanden.

Da der Drucker unter anderem den Epson JX80 emuliert, habe ich zunächst ein JX80 kompatibles Programm geschrieben. In dieser Betriebsart werden immer 2 bzw. am Rand 3 Thermowiderstände zusammengefaßt und simulieren so die 8 Nadeln des JX80. In dieser Betriebsart wird jedoch 50 % mehr Farbband verbraucht als wenn man die 24 Elemente einzeln ansteuert. Interessenten erhalten von mir natürlich die JX80-Version. Es ist aber anzunehmen, daß auf diesem Drucker die Mischfarbenverteilung neu angepaßt werden muß.

Farbhardcopy ist ein sehr komplexes Thema. So stellt dieses Programm nur eine Zwischenstufe dar, da ich ständig durch Versuche und neue Überlegungen eine bessere Farbverteilung zu erreichen suche.

Jeder Punkt der COL256 wird mit einer 4 x 4 Matrix auf dem Drucker dargestellt. Bei 3 Grundfarben ergeben sich so 4096 Kombinationsmöglichkeiten. Viele Farben fallen dabei aber gleich aus, so daß man sich durch Probieren eine Farbtabelle erstellen muß.

Das Hauptprogramm liest nun 6 x 256 Punkte ein und ordnet jedem Bildpunkt eine Farbkombination zu, die in einem dreidimensionalen Feld zwischengespeichert wird. Hier sind die Farben so verteilt, daß die Farbintensitäten ungefähr den Grauwerten entsprechen. Durch eine etwas höhere Auflösung im unteren Wertebereich bewirkt man eine sogenannte Gammaentzerrung, um z.B. bei Satellitenbildern die Erboberfläche deutlicher darzustellen.

Am Ende des Einlesevorganges wird das Feld auf den Drucker ausgegeben. Dabei wird jedem Zahlenwert des dreidimensionalen Feldes eine Matrix von 4 x 4 Punkten zugeordnet. In drei Durchläufen pro Zeile werden nun sechs untereinander liegende Matrizen spaltenweise ausgedruckt.

```
MODULE Copy;
(*******************************************)
(* Farbhardcopy fuer ITHO TPX 80 und COL256 *)
(* (C) Willi Sicking 25.8.86 Vers. 1.2      *)
(*******************************************)


FROM Files       IMPORT File,Open,Close,BinTextMode,ReadWriteMode,FileState;
FROM StandardIO  IMPORT SetOutput;
FROM SimpleIO    IMPORT WriteString, WriteChar, ReadString, WriteLn;
FROM m2util      IMPORT init,getcol,setdot,clear,zeichen,print;
IMPORT Terminal;

TYPE feld = ARRAY[1..3],[1..256],[1..6] OF CARDINAL;

VAR n,i,j,x,y,yoff,wert:CARDINAL;
    Farbe :feld;
    Logo:  ARRAY [1..41] OF CHAR;
    Drucker:File;
    art:FileState;

PROCEDURE druck(VAR Farbe:feld);
VAR out,r,s,ss  :INTEGER;
            a : ARRAY [1..4],[1..6] OF INTEGER;

BEGIN
  FOR n:=1 TO 3 DO       (*  1=gelb, 2=rot, 3=blau  *)
    WriteChar(CHR(27));
    WriteChar('r');      (* Farbband initialisieren *)
    IF n = 1 THEN WriteChar(CHR(4)) END;
    IF n = 2 THEN WriteChar(CHR(1)) END;
    IF n = 3 THEN WriteChar(CHR(2)) END;
    WriteChar(CHR(27));  (* 1024 Punkte Bitimage *)
    WriteChar('*');
    WriteChar(CHR(32));  (* Einzelelementmodus *)
    print(0);            (* WriteChar(CHR(0)); *)
    WriteChar(CHR(4));   (* 4*256+0 Punkte     *)

    FOR x:=1 TO 256 DO
      FOR y:=1 TO 6 DO
        FOR r:=1 TO 4 DO
          a[r,y]:=0;
        END;
        CASE Farbe[n,x,y] OF
        1 :
          a[1,y]:=5;
          a[2,y]:=0;
          a[3,y]:=5;
          a[4,y]:=0;
        2 :
          a[1,y]:=15;
          a[2,y]:=15;
          a[3,y]:=15;
          a[4,y]:=15;
        3:
          a[1,y]:=1;
          a[2,y]:=4;
          a[3,y]:=2;
          a[4,y]:=8;
        END;
      END; (*y*)
      FOR r:=1 TO 4 DO
        FOR s:=1 TO 3 DO
          ss:=s*2;
          out:=a[r,ss-1]*16+a[r,ss];
          print(out);
        END; (*s*)
      END; (*r*)
    END; (*x*)
    WriteChar(CHR(13)); (* CR *)
  END;  (*n*)
  WriteChar(CHR(10));   (* LF *)
END druck ;

BEGIN
(* Bildunterschrift und Testbalken *)
(*init;
  FOR x:=0 TO 255 DO
    FOR y:=0 TO 10 DO
      setdot(x,y,x);
    END;
  END;
  ReadString(Logo);
  FOR i:= 1 TO 41 DO
    zeichen(i*6,1,0,(ORD(Logo[i])-32)*8);
  END; *)

  Open (Drucker,"LST:",textMode,appendOnly,art);
  SetOutput(Drucker);

  Terminal.WriteString(" Start");

  WriteLn;
  WriteChar(CHR(27));       (* Zeilenabstand *)
  WriteChar('A');
  WriteChar(CHR(11));

  yoff:=0;
  REPEAT
    FOR x:=1 TO 256 DO
      FOR y:= 1 TO 6 DO
        Farbe[1,x,y]:=0;
        Farbe[2,x,y]:=0;
        Farbe[3,x,y]:=0;
        wert:=getcol(256-x,y-1+yoff);
        CASE  wert OF
          0..9      : Farbe[1,x,y]:=2;
                      Farbe[2,x,y]:=2;
                      Farbe[3,x,y]:=2;     (* schwarz          *)
          10..19,   : Farbe[2,x,y]:=2;
                      Farbe[3,x,y]:=2;     (* dunkelblau       *)
          20..29    : Farbe[1,x,y]:=2;
                      Farbe[2,x,y]:=1;
                      Farbe[3,x,y]:=2;     (* dunkelgruen      *)
          30..39    : Farbe[1,x,y]:=2;
                      Farbe[3,x,y]:=2;     (* hellgruen        *)
          40..49    : Farbe[2,x,y]:=1;
                      Farbe[3,x,y]:=2;     (* blau             *)
          50..59    : Farbe[1,x,y]:=1;
```

```
                Farbe[3,x+y]:=2;    (* gruenblau      *)
    60..69    : Farbe[3,x,y]:=2;    (* hellblau       *)
    70..79    : Farbe[1,x,y]:=2;
                Farbe[2,x,y]:=2;
                Farbe[3,x,y]:=3;    (* braun          *)
    80..89    : Farbe[1,x,y]:=2;
                Farbe[2,x,y]:=2;
                Farbe[3,x,y]:=1;    (* hellbraun      *)
    90..99    : Farbe[1,x,y]:=2;
                Farbe[2,x,y]:=2;    (* rot            *)
    100..114  : Farbe[2,x,y]:=2;    (* rosa           *)
    115..129  : Farbe[1,x,y]:=1;
                Farbe[2,x,y]:=2;    (* hellrosa       *)
    130..144  : Farbe[1,x,y]:=2;
                Farbe[2,x,y]:=3;    (* rotgelb 1      *)
    145..164  : Farbe[1,x,y]:=2;
                Farbe[2,x,y]:=1;    (* rotgelb 2      *)
    165..184  : Farbe[1,x,y]:=3;
                Farbe[2,x,y]:=1;    (* rotgelb 3      *)
    185..209  : Farbe[1,x,y]:=2;    (* gelb           *)
    210..240  : Farbe[1,x,y]:=1;    (* hellgelb       *)
                                       weiss          *)
  ('241..256
  END;  (* CASE *)
 END;     (* Y *)
END;      (* X *)
yoff:=yoff+6;
druck(Farbe);
```

```
  UNTIL yoff >= 256;
END Copy.


%FOREIGN DEFINITION MODULE m2util;

EXPORT QUALIFIED init,clear,setdot,draw,
                 flaeche,zeichen,getcol,print;

PROCEDURE init;

PROCEDURE clear;

PROCEDURE setdot(x,y,color : INTEGER);

PROCEDURE draw(x1,y1,x2,y2,color,xormode : INTEGER);

PROCEDURE flaeche(x,y,dx,dy,color : INTEGER);

PROCEDURE zeichen(x,y,color,adresse : INTEGER);

PROCEDURE getcol(x,y :INTEGER):INTEGER;

PROCEDURE print(a:INTEGER);

END  m2util.
```

BIOS-Änderungen beim CP/M68K

von Rüdiger Bäcker

Das CP/M68k besteht aus zwei festen (CCP & BDOS) und einem variablen (BIOS) Teil

Das BIOS ist der Teil des Systems, der an die vorhandene Hardware angepaßt werden muß. CCP und BDOS sind in dem auf einer CP/M-Diskette befindlichen File CPMLIB als relokative Programme untergebracht. Das Systemspezifische BIOS muß mit Hilfe des Linkers LO68 hinzugebunden werden.

Das BIOS des NDR-Klein-Computers steht lobenswerter Weise als Source in der Datei NDRBIOS.s zur Verfügung. Will man in diesem BIOS Änderungen vornehmen oder neuere Versionen des Bios einbinden (z.B. die mit Winchester-Routinen), so werden zweckmäßiger Weise die nachstehenden Files auf eine Diskette kopiert:
NDRBIOS.S / AS68.68K / LO68.68K / CPMLIB / EDITRDK.68K / RELOC.68K

Der AS68-Assembler benötigt zum Arbeiten die Datei AS68SYM.DAT, ist diese Datei nicht vorhanden, so muß sie aus der Datei AS68INIT erzeugt werden. Dazu wird der Assembler wie folgt aufgerufen: AS68 -I AS68INIT
Hierzu müssen sich natürlich die Dateien AS68.68K und ! AS68INIT auf der Diskette befinden.

Um aus einem geänderten Source-BIOS wieder eine Datei CPM.SYS zu erhalten, geht man wie folgt vor:

1. Assemblieren mit AS68 -p NDRBIOS.S)NDRBIOS.LST
Es werden die Objektdatei NDRBIOS.O und das Listingfile NDRBIOS.LST erzeugt. Das Listing kann mit der Befehlsfolge PIP LST:=NDRBIOS.LST auf dem Drucker ausgegeben werden. Dazu sollte der Drucker jedoch zunächst auf 132 Zeichen pro Zeile umgeschaltet werden.

2. Linken durch LO68 -R -UCPM -O CPM.REL CPMLIB NDRBIOS.O
Es wird die Datei CPM.REL erzeugt.

3. Größe von CPM.REL durch SIZE CPM.REL ermitteln.
Es wird die Größe u.a. in HEX ausgegeben (letzte Spalte). Dieser Wert wird von der letzten, zur Verfügung stehenden Speicherstelle abgezogen und dazu 1 addiert. Dieser Wert XYZ ist die Laufadresse von CPM.SYS.

4. Vorhandene DATEICPM.SYS umbenennen, z.B. mit REN CPM.SYS = CPM.-BAK

5. Dabei CPM.REL relozieren durch RELOC -BXYZ CPM.REL CPM.SYS (XYZ ist die unter 3. ermittelte Adresse).

6. Reset drücken und System neu booten.

Wird durch die Änderung der gewünschte Effekt erreicht, so kann das File CPM.-SYS auf die anderen Disketten kopiert werden.

Das Übersetzen des Source kann durch das nachfolgende Submit-File erleichtert werden:

```
as68 $1.s
1o68 -r -ucpm -o cpm.rel cpmlib $1.o
ren cpm.sys = spm.bak
reloc -b39000 cpm.rel cpm.sys
era $1.o
era cpm.rel
```

Der Aufruf erfolgt durch "putbios ndrbios", wenn das Sourcefile ndrbios.s heißt.

Zur Erzeugung des Listings dient dann ein weiteres Submit-File:

```
as68 -p $1.s )$1.1st
prninit
pip lst:=$1.1st
```

Der Aufruf erfolgt durch ASS NDRBIOS, es können jedoch beliebige andere .LST Files ausgegeben werden. Das Programm PRNINIT schaltet den Drucker um, hier das Source des Programmes:

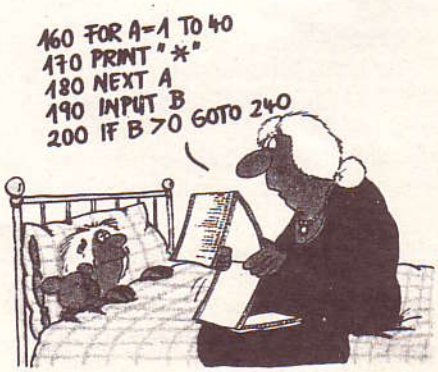
```
*   Printerinitialisierung für AS68 =
    Listingdruck
)   (C) 1968 bei R. Bäcker

lst     equ     5       * BDOS Funktionen
reboot  equ     0

start:
move #lst,d0            * BDOS-Code fuer Druckerausgabe in d0
move #$1b,d1            * auszugebendes Zeichen (ESC) in d1
trap #2                 * und BDOS aufrufen
move #lst,d0            * usw.
move #$0f,d1
trap #2
move #lst,d0
move #$1b,d1
trap #2
move #lst,d0
move #$6c,d1
trap #2
move #lst,d0
move #15,d1
trap #2
move #reboot,d0
trap #2
end
```

Diese Source kann wiederum mit Hilfe eines Submit-Files in ein .68K Programm übertragen werden:

```
as68 $1.s
1o68 -r -o $1.rel $1.o
reloc $1.rel $1.68K
era $1.o
era $1.rel
```

Der Aufruf : ASS PRNINIT. Es können damit beliebige .S Files übersetzt werden.

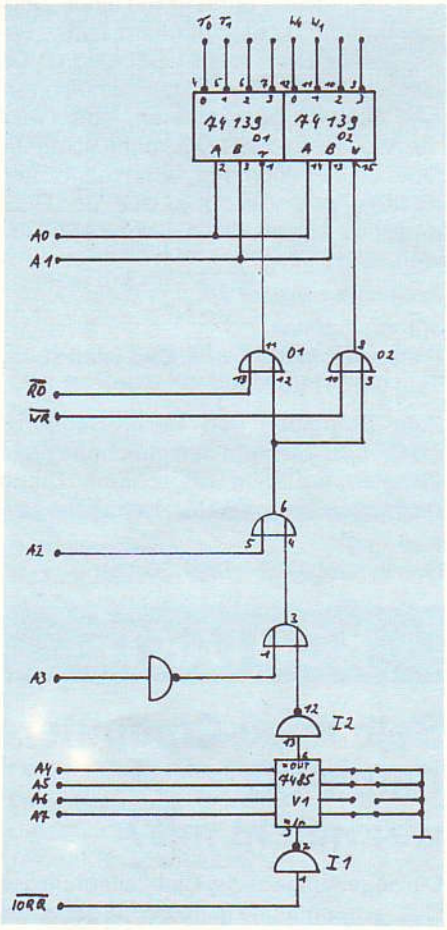
Quellennachweis: Zeitschrift »Freundin«, Heft 9/86



Sound und Cent – Die Ausdecodierung der IOE

von Uwe Koch, Lüdenscheid

Für meinen Drucker verwende ich als Anschluß an den NKC eine normale IOE-Karte. Um Platz zu sparen, habe ich sogar das Lochrasterfeld abgesägt. Bis vor kurzem funktionierte dies auch alles sehr gut. Als ich aber den 68008 in Betrieb nahm, trat ein kleines Problem auf. Ich habe nämlich auch eine SOUND-Karte in meinem Rechner. Diese ist zunächst laut Buch auf Adresse E0h,E1h eingestellt gewesen. Da ich kaum Software für die SOUND-Karte unter dem Z80 hatte, habe ich die Adresse für den 68008 auf 40h,41h umgestellt. (Warum ist eigentlich die SOUND-Karte beim 68008 als einzige auf einer anderen Adresse als beim Z80?) Die SOUND-Routine des Grundprogrammes funktionierte auch richtig, ebenso wie der SOUND-Befehl des RL-Basic. Nun habe ich jedoch inzwischen ein recht gutes Musikprogramm für den Z80 von einem Bekannten bekommen. Ich habe in dem Programm die Adresse vonE0h,E1h auf 40h,41h geändert und das Programm dann gestartet.



Es spielte korrekt die gewünschten Lieder, allerdings machte der zufällig eingeschaltete Drucker zwischendurch von Zeit zu Zeit einen Zeilenvorschub. Nach kurzem überlegen fiel mir ein, daß die Centronics-IO-Karte die Adresse nicht voll decodiert. Da die IO-Karte ausschließlich für die Centronics-Schnittstelle benötigt wird und die Decodierung für A4 – A7 auch festverdrahtet ist (nicht mit Jumpern), habe ich für die feste Adresse 48h, 49h eine Decodierung hinzugefügt. Auf der IO-Karte sind noch einige Inverter und OR-Gatter in den vorhandenen ICs frei. Daher kommt man ohne zusätzliche Bauteile aus (was bei abgesägtem Lochrasterfeld auch schwierig wäre).

Es muß lediglich der Ausgang des Inverters I2 (7404 Pin 12) von der weiteren Schaltung getrennt und die zusätzliche Decodierung dazwischen gefügt werden. Die OR-Gatter verhalten sich für LOW-Pegel wie AND-Gatter und können bei den Low-aktiven Signalen direkt verwendet werden. Die Adresse 48h sieht in der Binärdarstellung so aus: 0 1 0 0 1 0 0 0. Die ersten vier Bits werden vom 7485 decodiert, die letzten beiden von den beiden 74139. Es bleiben also die beiden Bits 1 0 auf den Leitungen A3 und A2. Für die Decodierung werden die OR-Gatter benutzt, daß heißt, es müssen LOW-Pegel "AND-verknüpft" werden. A2 ist schon als LOW-Pegel vorhanden und A3 kann mit einem freien Inverter invertiert werden. Daraus ergibt sich die im Schaltplan dargestellte Schaltung, die jetzt einwandfrei mit den anderen Ports zusammenarbeitet. Da zur Erweiterung nur eine Leiterbahn aufgetrennt werden muß und fünf neue Drahtverbindungen gelegt werden brauchen, ist die Änderung schnell erledigt.

Hinweis der Redaktion: In Vorbereitung befindet sich die IOE2, die die Ports **voll** ausdecodiert. CENT2 decodiert 48h und 49h bereits voll aus.

Speicherinhalt kopieren (Menü 3)

von Helmut Holitzner

```
Kopie:  JSR $2F20      : Speicherausgabe
        JMP $C000      : Adresse der gespeicherten Hardcopy-Routine
        end
```

Starten mit „Kopie". Menü läßt sich normal benutzen. Zum Ausdrucken M = Menü drücken! Aktueller Bildschirminhalt wird als Hardcopy ausgedruckt.

```
Speicher ansehen

+=weiter -=rueckw R=Adr M=Menue

-------  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00009040 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 3B
00009040                                              ;
00009050 20 41 65 6E 64 65 72 75 6E 67 20 66 75 65 72 20
00009050    A  e  n  d  e  r  u  n  g    f  u  e  r
00009060 53 74 61 72 20 44 50 20 35 31 30 2C 20 32 38 2E
00009060 S  t  a  r    D  P    5  1  0  ,    2  8  .
00009070 31 32 2E 38 35 20 48 6F 6C 69 74 7A 6E 65 72 0D
00009070 1  2  .  8  5    H  o  l  i  t  z  n  e  r
00009080 0A 6F 72 67 20 24 62 30 30 30 0D 0A 6C 6F 78 20
00009080    o  r  g    $  b  0  0  0       l  o  x
00009090 20 20 20 20 20 20 20 20 20 20 65 71 75 20 24 66
00009090                         e  q  u    $  f
000090A0 66 66 66 38 39 20 20 20 20 20 20 20 20 20 20 20
000090A0 f  f  f  8  9
000090B0 20 3B 20 46 61 64 65 6E 6B 72 65 75 7A 20 78 2D
000090B0    ;    F  a  d  e  n  k  r  e  u  z    x  -
```

Von CP/M68k zu xx-DOS

von Michael Milzsch, Hamburg

Es gibt viele Mini-DOS-Programme für die 68xxx-Prozessoren, aber es gibt noch kein Programm, welches CP/M68k kompatibel ist.

Für denjenigen, der nun ein CP/M und ein solches DOS-System hat, stellt sich die Frage: Wie bekomme ich mein Text- oder Daten-File nach CP/M? Nun die erste Lösung wäre, ein Textfile auf Adresse $9000 zu legen, dann das CP/M booten und mit „SAVE NAME.XXX" auf die Diskette zu schreiben.

Die zweite Lösung: Das Textfile in einen freien Speicherbereich legen, aber aufpassen, daß CP/M dort nicht hinkommt, CP/M booten und mit EDITRDK einen Namen mit Extension angeben und mit CTRL-KR dann den Text nach $9000 runter kopieren.

Bei Datenfiles ist es nicht ganz so einfach. Man muß das Programm wieder in einen freien Speicher bringen, der von CP/M nicht berührt wird. Wenn das Programm dort einwandfrei abläuft, kann man dann das CP/M booten. Nun kommt man nicht umhin, den DDT oder beim 68020 den DDT10 einzusetzen. Achtung, merken Sie sich, wo das Programm liegt, sonst muß man den ganzen Speicherbereich durchsuchen. Sie sollten sich das Programm mit dem „D"-Befehl ansehen, z.B. D10000. Wenn Sie das Programm dort lassen wollen, können Sie mit „WNAME.???,1000,1?????", das Programm auf die Diskette speichern. Wenn Sie aber das Programm nach $400 haben wollen (dort fängt der Arbeitsbereich von CP/M an), so müssen Sie das Programm mit „M10000,1????,400" runter kopieren. Das Abspeichern erfolgt wie oben genannt.

Selbstverständlich können Sie Programme von CP/M in Ihr DOS einlesen, der Vorgang ist dann nur entgegengesetzt.

Briefe - Kontakte - Kleinanzeigen

NEU:

GES NORM

Was ist die „GES-Norm?" Vereinfacht dargestellt eine interne Norm, nach der alle neuen Layouts erstellt werden. Die Produkte, die noch nicht der Norm entsprechen, werden Schritt für Schritt revidiert, soweit sinnvoll und nötig.

Was beinhaltet diese Norm?

Mechanik:

Einheitliches Europakartenformat (100 x 160 mm). Befestigungslöcher für Metall-Abdeckplatte an der hinteren Schmalseite. Alle Verbindungen und Stecker auf der dem ECB-Stecker gegenüberliegenden Schmalseite.

Soweit möglich:

NDR-BUS auf der Längsseite und ECB-BUS auf der Schmalseite.

Elektrik:

Wait-Erzeugung, soweit nötig, auf der Baugruppe; Interrupt-Signal-Erzeugung, soweit sinnvoll.

Hardware-Entwickler können gerne die gesamte Norm (kostenlos) bei uns anfordern!

Ich bin eigentlich NDR-Computer-Fan weil ich das System gut finde und recht interessiert an der ganzen Sache bin.

Aus verschiedenen Gründen fehlt mir jedoch die Zeit, um mich intensiv mit meinem Computer zu befassen. Vielleicht haben andere *LOOP*-Leser ähnliche Probleme wie ich und deshalb zwei Anregungen an dieser Stelle:

1. Da in der *LOOP* schon eine gewisse Aufteilung verschiedener Computersysteme, Ausbaustufen, etc. vorgenommen wird, wäre es dann nicht möglich, beispielsweise verschiedene Artikel über den Z80 mit einer Seite zu beginnen und auch mit dem Ende einer Seite abzuschließen. Wenn jetzt noch die *LOOP* in Ringbuchformat aufgelegt würde, könnte sich jeder Leser das für ihn Interessanteste heraussuchen, sortieren und abheften.

2. Es gibt viele spezielle Computerausdrücke mit denen ich sehr wenig anfangen kann. Man könnte in einem Anhang alle möglichen Begriffe nach irgendeinem System auflisten. Mit einer guten Erklärung versehen wären sie bestimmt vielen Anfängern hilfreich.

Gerd Seipel
Bezirksstraße 21, 8755 Alzenau 2

Antwort LOOP:

Wir müssen bei jeder *LOOP* um jeden Millimeter Platz kämpfen. Deshalb geht es leider nicht! Den Vorschlag für das Lexikon stellen wir gerne zur Diskussion. Wer ist dafür? Welche Begriffe sollten erscheinen?

Sehr geehrte Redaktion,

ich beziehe erst seit der letzten Ausgabe die *LOOP*, weil ich mir gedacht hatte, daß man durch nur kopieren oder mitlesen letztendlich sich selber schadet. Die letzte Ausgabe erfüllt für mich zum ersten Mal voll die Anforderungen, die man als User einem 3,— DM teuren, 36 Seiten umfassenden und als Werbemagazin funktionierendem Blättchen abverlangen kann. Die Zeitschrift hat sich als Alternative zu den Franzis Zeitschriften gemausert. Es wird versucht, jedem gerecht zu werden, der den Rechner in einer von vielen Möglichkeiten gebaut hat und benutzt. Ich hoffe daher sehr, daß dieses Vertrauensverhältnis von *LOOP* und *LOOP*-Lesern durch so banale Sachen wie einer „Doppelnummer" nicht stören wird. Lange Rede, kurzer Sinn: mein Vorschlag ist es, diesen kleinen Ausrutscher (in Worten: 8/9) dadurch zu versöhnen, indem man die nächste (diese) *LOOP* dann 9b nennt. Oder eine Freinummer. Denn merke: *Keine* NDR-Anwender ohne *LOOP* und GES; *und keine LOOP* und GES ohne NDR-Anwender!!!

Ansonsten: Weiter so.

Jürgen Waffner,
Frintroperstraße 378, 4300 Essen 11,
Tel.: (0201) 607343

Zum Schreiben von Herrn Sabouret, *LOOP* 8/9: Ich hatte den gleichen Fehler (Streifen auf dem Bildschirm). Durch Austausch zweier RAM's behoben!

Karl Hudler
Schillerstraße 13 · 7321 Dürnau

KONTAKTE

Selbstbau-Computer-Club SCC – wer macht mit?

Die neue Ausgabe der Clubzeitschrift des SCC erreichte uns in diesen Tagen. Dem SCC ging es wie uns: Nach einigen Ausgaben wurde eine richtig ernstzunehmende Zeitung daraus!

Obwohl oft vermutet, liegt uns Konkurrenzdenken fern. Im Gegenteil: Wir begrüßen Clubs, Clubgründungen und

Zeitungen, da nur so alle „Register" des NDR-Computers gezogen werden können!

Der SCC bringt für DM 18,– im Jahr sechs Ausgaben der Benutzerzeitschrift, die interessante Programme, Tips und Tricks und Mitteilungen aus dem Clubleben enthält.

Kontaktadresse:

Horst G. Eysel
Klostergut Riechenberg · 3380 Goslar
Telefon (0 53 21) 4 08 06
PSA Hannover 5490 92-307 Sonderkonto

Hinweis für weitere Clubs: Bitte melden!

Oberösterreich/Linz

Suche Kontakt zu NDR-Usern (Z80/68000)

Markus Krippner
Landstraße 70/7 · A-4020 Linz
Telefon 2 70 79 44

Wer hat Interesse an Kommunikation über Akustikkoppler? Tel.-Nr. bekannt geben! Eventuell über Mailbox.

Wendelin Egger
Postweg 3 · 7919 Ritzisried

Wer baut den NDR-Klein-Computer mit 68'er CPU (evtl. Z80)? Ich programmiere in Pascal und 68'er Assembler. Wer im Großraum Aachen, Düren, Köln, Bonn an einem Kontaktaustausch interessiert ist möge doch bitte an folgende Adresse schreiben bzw. telefonieren:

Achim Ladwig
Im Johannistal 39 · 5100 Aachen
Telefon (02 41) 7 85 59

Suche Kontakt zu anderen NKC-Besitzern im Raum Karlsruhe.

Oliver Rettig
Saillerstraße 26 · 7514 Eggenstein-Leo 1

Die erste Wilhelmshavener Mailbox

Betriebszeiten:
Täglich 20.00 Uhr – 7.00 Uhr
Telefon (0 44 21) 2 72 04
300 BAUD · 8 BIT · 1STOP-BIT
Ruf doch mal an!

Suche Kontakt zu anderen NDR-Anwendern im PLZ-Raum 4800, 4900.

Peter Schwarck
Dreyener Straße 67 · 4905 Spenge
Telefon (0 52 25) 17 69

„In unserer Computergruppe hat es nicht nur NDR-Anwender, sondern auch einige Commodore-128-Besitzer. Wir sind nun sehr interessiert zu erfahren, ob sich jemand schon einmal damit befaßt hat, wie der NDR-Computer das vom Commodore-Laufwerk 1571 beschriebene Diskettenformat lesen kann. Viele 80-Spur-Laufwerke lassen sich nämlich einfach auf 40-Spur-Betrieb umschalten, so daß sich vom Laufwerk her keine Probleme stellen. Dagegen ist das exotische Commodore-Format schon eine rechte Knacknuß. Wer hat sich schon einmal hinter eine solche BIOS-Änderung gewagt? Zum Zweiten suchen wir Möglichkeiten, ASCII-Dateien auszutauschen. Wie steuert man unter CP/M die serielle Schnittstelle des C-128 an? Gibt es Terminalprogramme? Für Erfahrungen und Hinweise sind wir sehr dankbar."

Heinz Amgwerd
Rebbergstraße 13a · CH-5610 Wohlen
Telefon (0 57) 22 76 42

Kleinanzeigen

Zu verkaufen:

1 Tast 1 DM 90,–
1 CAS B aufgebaut und nicht geprüft DM 30,–

Herbert Schmidt
Telefon (0 91 61) 97 52

Verkaufe:

NDR-Computer Z80 mit Baisc und Gossi, CPU68K mit EASSO 4.3/MON2G, cen0pow22/26/SPC2 und Literatur

Max Rogler
Bergstraße 14 · Reichertshausen
Telefon (0 81 37) 3 35

Verkaufe:

NDR-Klein-Computer, betriebsbereit, wegen Zeitmangel zu verkaufen, CPU68K/SBC2 + Proms, ROA64, GDP64K, KEY, CAS, IOE, POW5V, BUS2, TAST1, TGEH, Bausatz PROMER, MODU, LOOP 1 – 6, Sonderheft 1 + 2, Listing 68 Grund- und Aufbauprog. RDK-Buch, R. Zaks: Programmierung des Z80, VB DM 800,–.

[redacted]

Verkaufe:

POW5V mit Trafo, CAS mit Datenrecorder MC 3810, kl. Tastatur (Elektronikladen DT). Alle Teile betriebsbereit. Preis VHS.

Peter Schwarck
Dreyener Straße 67 · 4905 Spenge
Telefon (0 52 25) 17 69

Verkaufe: Z80 CPU Vollausbau mit MONI, BASIC und GOSI KEY, IOE, POW5, GDB64K, CAS, BUS1 und kleine Tastatur (Elektronikladen).

Hans Federowsky
U.-H.-Reuth 41 · 8671 Feilitzsch
Telefon (0 92 95) 3 61

Verkaufe:

TAST1 Cherry Tastatur komplett neu mit Original-Gehäuse und Kabel DM 170,–.

R. Cudok
Feldstraße 12 · 3200 Hildesheim
Telefon (0 51 21) 8 38 22

Verkaufe: Kleine Tastatur (Elektronikladen), GDP64K, IOE, KEY, POW5V, BUS1, BUS2, BUS3, CAS, SBC2, Z80 CPU Vollausbau, Hexio, Hexmon, Eprom, Hexmon-Buch, MONI, BASIC, GOSI und HF-Modulator.

Hans Federowsky
U.-H.-Reuth 41 · 8671 Feilitzsch
Telefon (0 92 95) 3 61

Uhrenkarte mit Motorola-Echtzeit-Prozessor (gute Interruptsteuerung!). Dazu komfortables Schaltuhrenprogramm mit programmierbarer Steuerung. Außerdem weitere kleine Programme. Info DM 1,60 in Briefmarken an
M. Zehner
Schwenckestraße 2 · 2000 Hamburg 20

Grafik + Schaltpläne für NDR 68008 – Zeichnen – drucken – speichern DM 68,–

K. Hahn
Kreuzlach 19 · 8806 Neuendettelsau

Verkaufe: NDR-Klein-68K-System alle Platinen weit unter Bausatzpreis.

Elmar Ohst
Carl-Zöllig-Straße 4 · 4030 Ratingen
Telefon (0 21 02) 8 17 36

NDR-68008: Verkaufe Strategiespiel REVERSI, sehr spielstark! Auf Eprom oder 5¼"-Disketten (JADOS). DM 29,– + Porto per NN.

K. Janßen
Hanninxweg 74 · 4150 Krefeld 1

NDR 68008: Verk. hochkomfort. Vokabelprogr. inkl. 4000 (!) englische Vok. u. Redewendungen (ca. 160K). Ideal f. Anfänger u. Fortgeschr. 64K RAM erforderl. Disk. u. ausführl. Anleitung für. nur DM 90,– per NN + Porto u. Verp. Info gegen Rückumschlag.

Kai Ruppert
Kiebitzweg 8 · 3171 Osloß
Telefon (0 53 62) 74 40

Verkaufe: NDR-Computer mit SBC2, IOE, POW5, BUS2, KEY, MON1, GDP, CAS, CERRY-Tastatur im Gehäuse. Preis: VB.

Gerhard Gatena
Friesenstraße 21 · 2912 Uplengen

68008: Verkaufe ROA64 komplett, geprüft, mit Original EASSO-3 V4.3 mit Handbuch und Original EPASCAL V3.1, alles zusammen für DM 220,–.

T. Strassen
Zürich · Tel. (01) 555 246.

Theorie und Praxis rund um den NDR-Computer

Mikroelektronik
Einführung

NEU

4 Kursteile (je ca. 70 Seiten im Format A4), DM 38,- je Kursteil

Der Kurs ist auf die HEXIO abgestimmt und ist für alle geeignet, die ihre ersten Schritte in Z 80-Maschinenprogrammierung machen.

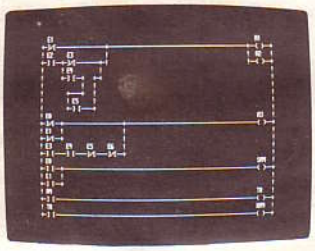
Nach diesem Kurs sind Sie in der Lage, eigene Programme zu schreiben und die Arbeitsweise des Z 80 zu verstehen.

Der Kurs ist in verschiedene Fachgebiete aufgeteilt und bringt eine Menge Aufgaben, Beispielprogramme und Übungen.

Aus dem Inhalt: Was ist ein Mikroprozessor? * Inbetriebnahme des Computers * Planung von Programmen * Aufbau der CPU * Speicher und Adressen * Datentransfer * Lauflicht * Breakpoints * Hilfsfunktionen * Logo-Elemente * Strukturiertes Programmieren * Label & Call.

SPS-Programmierung

NEU



4 Kursteile (je ca. 70 Seiten im Format A4), DM 38,- je Kursteil

Dieser Kurs zeigt Ihnen, wie SPS programmiert wird, die Normung, die Anwendungsmöglichkeiten und die verschiedenen Darstellungsarten.

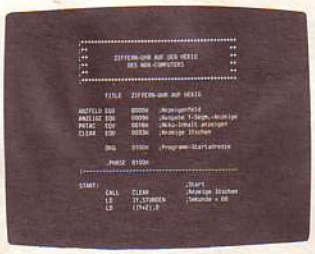
Sie lernen spielend leicht, Relais- und Schützensteuerungen in SPS-Programme umzusetzen.

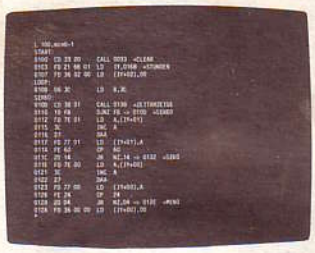
Beispielprogramme, Aufgaben und Übungen geben Ihnen die praktischen Erfahrungen und zeigen, wie SPS professionell eingesetzt wird. Nutzen Sie Ihren NDR-Computer für diese moderne Technik voll aus.

Der Kurs ist in folgende Fachgebiete gegliedert: Steuerungstechnik * Digitaltechnik * Methoden zur Beschreibung von Steuerungsaufgaben * Programmierung * Übungen und Tafeln.

ZEAT-Betriebssystem

NEU





Das Betriebssystem beeinhaltet in drei EPROMs: Z 80-2-Pass-Assembler, Disassembler, Editor, Debugger, Telefonmodem-Programm, FLOMON 1.5, ausserdem eine ausführliche Dokumentation zum Preis von DM 198,– . .

Das Betriebssystem ZEAT benötigt 64-K-RAM (dynamische RAM-Karte). Die EPROMs werden in die BANKBOOT-Karte eingesteckt und sind sofort betriebsbereit. Programmieren Sie Ihren NDR-Computer mit einem Profi-Assembler.

Das Textverarbeitungsprogramm hat volle Bildschirmeditierung und kann neben der Programmeditierung auch zum Textschreiben eingesetzt werden.

Z 80-Assembler-Programmierung

4 Kursteile (je ca. 70 Seiten im Format A4), DM 38,– je Kursteil

Der Kurs ist auf das ZEAT-Betriebssystem abgestimmt und zeigt Ihnen in leicht verständlicher Art, wie der NDR-Computer in Z 80-Assembler programmiert wird, bringt reichhaltig Übungsbeispiele und Anwendungen. Sie werden erstaunt sein, wie leicht diese Art der Programmerstellung ist. Und Sie lernen, wie man die serielle Schnittstelle bedient und Daten über Telefon übertragen kann.

Die Fachgebiete dieses Lehrgangs sind: Systembeschreibung * Betriebssystem * Programmierung * Testen * Modemprogramm * Listings, Tafeln und Tabellen.

Christiani

Hier abtrennen und im Umschlag einsenden an: Dr.-Ing. P. Christiani GmbH, Techn. Lehrinstitut und Verlag, Postfach 35 69189, 7750 Konstanz

Bestellcoupon

	Preis je Teil	Gesamtpreis
☐ Einführung mit dem NDR-Computer (4 Kursteile)	DM 38,–	DM 152,–
☐ Z 80-Assembler-Programmierung (4 Kursteile)	DM 38,–	DM 152,–
☐ SPS-Programmierung mit dem NDR-Computer (4 Kursteile)	DM 38,–	DM 152,–
☐ Kompakt-Kurs BASIC (angepasst an das RDK-BASIC)	DM 198,–	DM 198,–
☐ ZEAT-Betriebssystem (3 EPROMs mit Dokumentation)	DM 198,–	DM 198,–

Name, Vorname

Straße

PLZ, Ort

Datum Unterschrift 86189