

LOOP

Zeitung für Computer-Bauer, -Anwender, -Programmierer und -Starter

DM 3,-

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 Kassetten 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 wir 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 bereit erklärt 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 cm²; zum Vergleich hat ein Bügeleisen auf der höchsten Stufe etwa 6W/cm².

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 Unterrubriken (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 5 1/4" 80 Spuren

10769 3 1/2" 80 Spuren

10770 8" SSSD

Musik auf dem 68008

von Manfred Zehner
Schwenckestraße 2, 2000 Hamburg 20

Wer seinem 68008er endlich die „Flöten-töne“ 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 Halbtönschritte 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 besser 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-Com-

puter, z.B. mit der COL256 und „etwas“ Software.

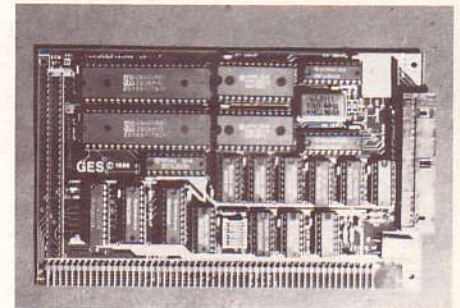
Dieses Programm soll die Befehle des Laser-Druckers interpretieren und auf dem Bildschirm darstellen, um so die Gestaltung auf dem Bildschirm vorzunehmen. Wer hat Lust zu einem solchen Programm?

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 Fertigerät ist die umfang-

reiche 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.



Preise:

- 10741 LOG16F Fertigerät + 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 intern (= manuel), extern, Triggerbyte

Triggermöglichkeit: ca. 0.4 A

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.

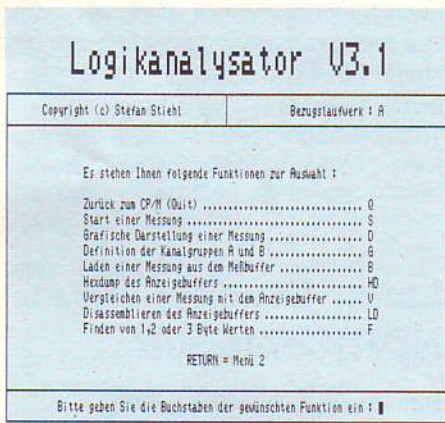


Bild 1

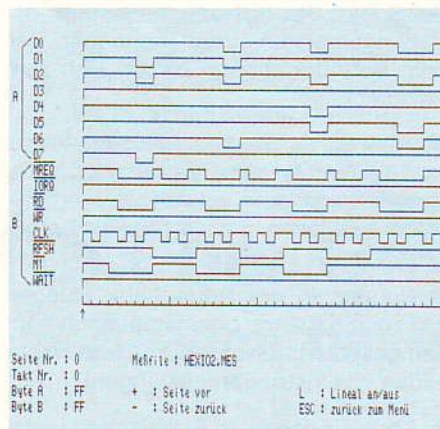


Bild 3



Bild 2

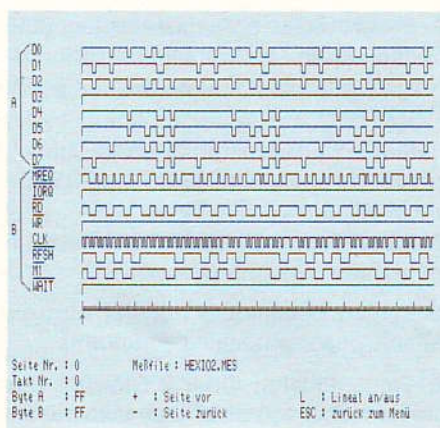
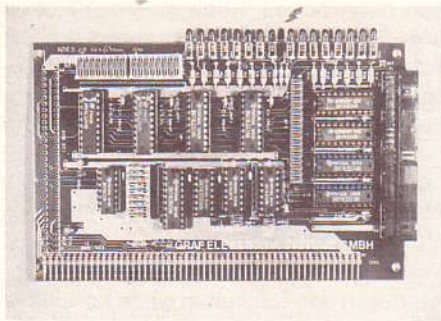


Bild 4



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 Centronicschnittstelle
- Ports voll ausdekodiert und frei einstellbar (mit DIL-Schalter)

Preise:

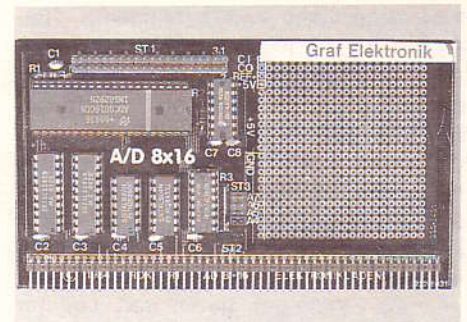
10625 Fertigerät	200,-
10626 Bausatz	150,-
10627 Leiterplatte	30,-
10628 Handbuch	20,-

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
- Analogspannungseingänge: 16
- Eingangsspannung: 0 - + 5 V



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

Preise:

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

CENT2

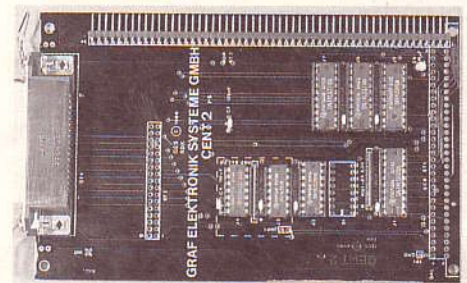
Die CENT2 ist eine eigenständige Baugruppe (nicht wie die CEN eine Aufsteckplatine auf die IOE). An diese Baugruppe kann direkt das Centronicskabel zum Drucker angeschlossen werden. Außerdem ist die CENT2 für Interruptsteuerung vorbereitet.

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 Fertigerät	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 Fertigerät, 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 Elektronikkenntnisse (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 Programmierstellung gewidmet): sie machen klar, daß ohne Problemanalyse 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 prak-

tischen 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 Beispielpogrammen 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 adressierte 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 (Exklusiv- 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 wer-

den 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!)]

Erhellend wird auch die Frage, wie der Rechner durch das Schließen eines Kontaktes (= drücken einer Taste auf der HEXIO-Tastatur) sich eine bestimmte Zahl einverleiht. 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	c3 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              ab 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              05 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.. dec HL       position um 11 Bild-
8906              ...2b      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) Löschttext (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

```

wird, verschoben 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.

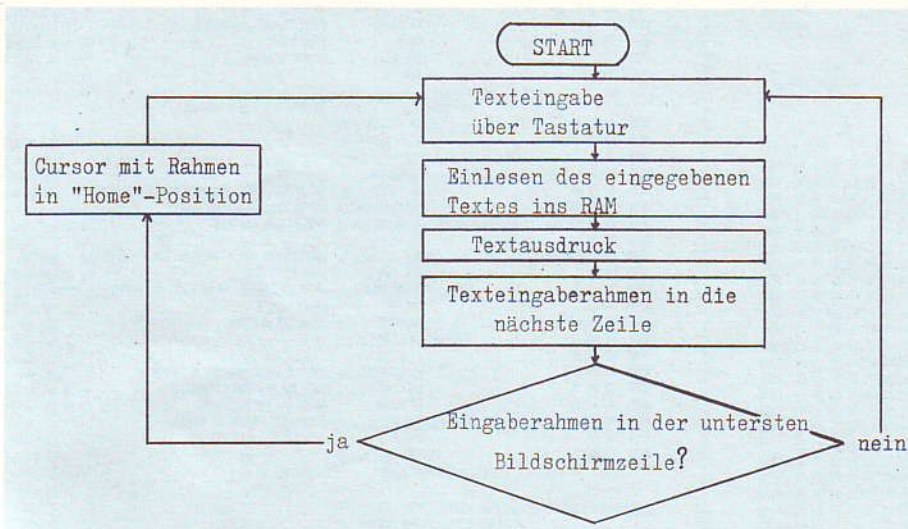
1.2 Die Symbole, die auf Speicherplätze im transportierten Bereich weisen, werden so geändert, daß sie auf den neuen Speicherplatz zeigen.

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 „ändern“ 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 Löschmodul 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 Programmierung gute Dienste geleistet.



PROGRAMM-SCHIEBER

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

Beim Arbeiten mit dem Grundprogramm ist es sehr umständlich, Korrekturen an bereits geschriebenen Programmen an-

zubringen. 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

BLOCK 17.01.1984

PAGE 1

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

TITLE BLOCK 17.01.1984
ORG 8000H

```
WRITE EDU 001EH
READ EDU 0021H
WAIT EDU 00F8H
SEDRNG EDU 0007H ;GRUNDPROGRAMMRoutine
LENGTH EDU 002AH ;
ANDBE1 EDU 8000H ;ANFANGSADRESSE DES ERSTEN DURCHSUCHTEN BEREICHES
ANDBE1 EDU 8000H ;ENDADRESSE DES ERSTEN DURCHSUCHTEN BEREICHES
ANDBE2 EDU 2200H ;ANFANGSADRESSE DES ZWEITEN DURCHSUCHTEN BEREICHES
ANDBE2 EDU 8000H ;ENDADRESSE DES ZWEITEN DURCHSUCHTEN BEREICHES
```

BLOCK: ;HAUPTPROGRAMM

```
8000 21 55 BF LD HL,ÜBSRT ;ÜBERSCHRIFT DRUCKEN
8003 CD 1E 00 CALL WRITE
8006 21 A6 BF LD HL,ERKL ;ERKLÄRUNG DRUCKEN
8009 CD 1E 00 CALL WRITE
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 SYMMD
801B 21 00 88 LD HL,ANDBE1
801E 22 0F BF LD (ANDBEGINN),HL
8021 21 00 80 LD HL,ANDBE1
8024 22 E1 BF LD (ANDBE2),HL
8027 CD CC 8E CALL PROGAND ;BEREICH AUF SBCII
802A 21 00 22 LD HL,ANDBE2
802D 22 0F BF LD (ANDBEGINN),HL
8030 21 00 80 LD HL,ANDBE2
8033 22 E1 BF LD (ANDBE2),HL
8036 CD CC 8E CALL PROGAND ;BEREICH VOR 8000H
8039 C9 RET
```

ANFEING: ;NACH ANFANGSADRESSE FRAGEN

```
803A 21 6A BF LD HL,BLOCKA
803D CD 1E 00 CALL WRITE
8040 21 96 00 LD HL,009AH
8043 22 C8 BF LD (YPOFE),HL
8046 CD 7C 80 CALL EINGAB
8049 21 03 BF LD HL,ANFADR
804C CD 88 80 CALL UMWAND
804F C9 RET
```

ENDEING: ;NACH ENDADRESSE FRAGEN

```
8050 21 7E BF LD HL,BLOCKE
8053 CD 1E 00 CALL WRITE
8056 21 64 00 LD HL,0064H
8059 22 C8 BF LD (YPOFE),HL
805C CD 7C 80 CALL EINGAB
805F 21 05 BF LD HL,ENDADR
8062 CD 88 80 CALL UMWAND
8065 C9 RET
```

ZIELEING: ;NACH ZIELADRESSE FRAGEN

```
8066 21 92 BF LD HL,BLOCKZ
8069 CD 1E 00 CALL WRITE
806C 21 32 00 LD HL,0032H
```

BLOCK 17.01.1984

PAGE 2

```
806F 22 C8 BF LD (YPOFE),HL
8072 CD 7C 80 CALL EINGAB
8075 21 07 BF LD HL,ZIELADR
8078 CD 88 80 CALL UMWAND
807B C9 RET
```

EINGAB: ;EINGABE UND TEST AUF RICHTIGKEIT

```
807C 21 00 00 LD HL,0
807F 22 CE BF LD (PU),HL
8082 22 00 BF LD (PU+2),HL ;PUFFER LÖSCHEN
8085 21 C6 BF LD HL,PUFFER
8088 0E 01 LD C,01
808A CD 21 00 CALL READ ;EINLESEN DER ADRESSEN
808D 3A CE BF LD A,(PU) ;ERSTES EINGEGEBNES ZEICHEN AUS PUFFER HOLEN
8090 C8 AF RES 5,A
8092 FE 53 CP 'S'
8094 CA EC 80 JP Z,SCHLUSS ;FALLS 'S', ABBRUCH
8097 3A CD BF LD A,(AKT)
809A FE 04 CP 04 ;SIND 4 ZEICHEN EINGEGEBEN WORDEN ?
809C CD 7C 80 JP NZ,EINGAB ;WENN NEIN, NOCHMAL EINGABE
809F FD 21 CE BF LD IY,PU
80A3 3E 00 LD A,00
80A5 CD FB 00 CALL WAIT
80A8 03 70 OUT (70H),A ;AUF POSITIVE SCHRIFT SCHALTEN
80AA FD 7E 00 LD A,(IY+0)
80AD FE 00 CP 00 ;0 IST SCHLUSZZEICHEN
80AF C8 RET Z ;FALLS '00', ABBRUCH
80B0 CD 07 00 CALL SEDRNG ;TEST OB SEDEZIMALES ZEICHEN
80B3 FD 23 INC IY ;NÄCHSTES 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 SEDEZIMALZAHLEN
;HL WIRD VORHER AUF DIE SPEICHERSTELLE EINGESTELLT,
;IN DER DIE SED.ZAHL NÄCHSTER STEHEN SOLL.

```
80BB FD 21 CE BF LD IY,PU
80BE FD 7E 02 LD A,(IY+2)
80C1 FE 40 CP 40H
80C4 04 E9 80 CALL NC,KORR
80C7 ED 4F RLD
80C9 FD 7E 03 LD A,(IY+3)
80CC FE 40 CP 40H
80CE 04 E9 80 CALL NC,KORR
80D1 ED 4F RLD
80D3 23 INC HL
```

```
80D4 FD 7E 00 LD A,(IY+0)
80D7 FE 40 CP 40H
80D9 04 E9 80 CALL NC,KORR
80DC ED 4F RLD
80DE FD 7E 01 LD A,(IY+1)
80E1 FE 40 CP 40H
80E3 04 E9 80 CALL NC,KORR
80E6 ED 4F RLD
80E8 C9 RET
```

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

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

80EE C9 TRANSP: ;TRANSPORT DES BESTIMMTEN BEREICHES

BLOCK 17.01.1984

PAGE 3

```
80EF 2A 05 BF LD HL,(ENDADR)
80F2 ED 5B 03 BF 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 70 LD A,L
80FC C6 01 ADD A,01
80FE 4F LD L,A
80FF 7C LD A,H
8000 CE 00 ADC A,00
8002 67 LD H,A
8003 22 09 BF LD (LANGE),HL ;LANGE+1 IN LANGE GESPEICHERT
8006 2A 07 BF LD HL,(ZIELADR)
8009 7C LD A,H
800A 92 SUB 0
800B FA 25 BE JP M,TRUNT ;FALLS ZIEL ANFANG : TRANSPORT NACH UNTEN
800E C2 1A BE JP NZ,TROB ;FALLS ZIEL ANFANG : TRANSPORT NACH OBEN
8011 70 LD A,L
8012 93 SUB E
8013 FA 25 BE JP M,TRUNT ;NIEDERMERTIGES BYTE DER ADRESSE TESTEN,
8016 C2 1A BE JP NZ,TROB ;WENN OBEN GLEICHHEIT
8019 C9 RET
```

801A C1 TROB: ;TRANSPORT NACH OBEN
801B 09 POP BC ;BC=ENDADR-ANFADR
801C E8 ADD HL,BC ;HL=ZIELADR+(ENDADR-ANFADR)
801D 09 EX DE,HL ;DE=HL-ANFADR
801E ED 4B 09 BF ADD HL,BC ;HL=ANFADR+(ENDADR-ANFADR)
8020 ED 88 LD BC,(LANGE) ;BC=LANGE-(ENDADR-ANFADR+1)
8022 ED 88 LDOR ;TRANSPORT
8024 C9 RET

8025 ED 4B 09 BF TRUNT: ;TRANSPORT NACH UNTEN
8029 EB LD BC,(LANGE)
802A ED 80 EX DE,HL ;DE=ZIELADR, HL=ANFADR
802C C1 LDIR ;TRANSPORT
802D C9 POP BC ;STACK SAUBERN
802E C9 RET

802E 2A 07 BF SYMMD: ;ÄNDERN DER SYMBOLTABELLE
8031 ED 5B 03 BF LD DE,(ANFADR)
8035 A7 AND A
8036 ED 52 SBC HL,DE
8038 22 00 BF LD (OFFSET),HL ;IN OFFSET STEHT DER ABSTAND ZIELADR-ANFADR
803B FD 21 C9 01 LD IY,BICPH ;IY=ZEIGER AUF SYMBOLTABELLE
803F FD 7E 00 LD A,(IY+0) ;ERSTES ZEICHEN HOLEN
8042 FE 00 CP 00
8044 C8 RET Z ;RÜCKSPRUNG FALLS TABELLE LEER
8045 C8 7F BIT 7,A ;7.BIT GESETZT ?
8047 C2 4F BE JP NZ,PRUF ;WENN JA : PRÜFEN OB ADRESSE IN TRANSPORTIERTEM BEREICH
804A FD 23 INC IY ;WENN NEIN :
804C C3 3F BE JP HOLEN ;NÄCHSTES ZEICHEN
804F FD 4E 01 LD L,(IY+1)
8052 FD 44 02 LD H,(IY+2)
8055 22 00 BF LD (ADR),HL ;IN ADR STEHT ADRESSE DES SYMBOLS
8058 CD 78 BE CALL GROSSE ;ADRESSE IN TRANSP. BEREICH ?
805B DA 47 BE JP C,MOD ;WENN JA, MODIFIZIEREN
805E FD 23 INC IY ;WENN NEIN, NÄCHSTES SYMBOL
8060 FD 23 INC IY
8062 FD 23 INC IY
8064 C3 3F BE JP HOLEN

BLOCK 17.01.1984

PAGE 4

8067 2A 00 BF MOD: LD HL,(ADR)
806A ED 5B 03 BF LD DE,(OFFSET)
806E 19 ADD HL,DE ;HL=ALTE ADRESSE+ABSTAND
806F FD 75 01 LD (IY+1),L
8072 FD 74 02 LD (IY+2),H
8075 C3 5E BE JP RÜCK ;NEUE ADRESSE IN TABELLE
;NÄCHSTES SYMBOL

8078 ED 4B 03 BF GROSSE: ;PRÜFUNG OB ADRESSE IN TRANSP. BEREICH
807C A7 LD BC,(ANFADR)
807D A7 AND A
807E ED 42 SBC HL,BC ;HL=ADR-ANFADR
807F DA 97 BE JP C,NC1 ;C-) ADR(ANFADR -) RÜCKSPRUNG OHNE CARRY
8082 2A 00 BF LD HL,(ADR)
8085 ED 4B 05 BF LD BC,(ENDADR)
8089 A7 AND A
808A ED 42 SBC HL,BC ;HL=ADR-ENDADR
808C D2 9E BE JP NC,NUTE ;NC-) ADR(ANFADR -) TEST OB ADR=ENDADR
808F C3 95 BE JP C1 ;ANFADR(ADR-ANFADR)
8092 C2 97 BE JP NZ,NC1 ;ADR(ANFADR -) RÜCKSPRUNG OHNE CARRY
8095 37 NUTE: SCF
8096 C9 RET ;CARRY
8097 A7 AND A
8098 C9 RET ;OHNE CARRY

8098 C9 LOSCHEN: ;LÖSCHEN VON ANFADR BIS ENDADR
8099 A7 AND A
809A ED 42 SBC HL,BC ;HL=ADR-ANFADR
809C D2 9E BE JP NC,NUTE ;NC-) ADR(ANFADR -) TEST OB ADR=ENDADR
809F C3 95 BE JP C1 ;ANFADR(ADR-ANFADR)
8092 C2 97 BE JP NZ,NC1 ;ADR(ANFADR -) RÜCKSPRUNG OHNE CARRY
8095 37 NUTE: SCF
8096 C9 RET ;CARRY
8097 A7 AND A
8098 C9 RET ;OHNE CARRY

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 wer-

den, 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) eingeleitet 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=S)

Eingabe: ESC P Ä ESC P D TEXT

(entspricht "ESC" "S" (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 I)

Schmal und eng

Schmal und eng

Eingabe: ESC P Ä ESC P D ESC P A ESC P I

(entspricht "ESC" "D" "ESC" "I")

5. Beispiel: Wieder normal (Normalieren)

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 A

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 A

Letztes Beispiel: Piepton am Schluss des Druckens

Eingabe: ESC P A B
(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
ENO	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	[5B	{	7B
FS	1C	<	3C	\	5C		7C
GS	1D	=	3D]	5D	}	7D
RS	1E	>	3E	^	5E	~	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 merkbar Namen abgelegt sind (z.B. FETT-EIN für „Fettdruck einschalten“). Dies ermög-

licht 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 Zaring, Heeper Straße 362,
4800 Bielefeld 17, Tel.: (0521) 332849

```

;Z80
banken equ 04204h ; 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 ; tabulator

start: ld e,cls ; Bildschirm loeschen
ld c,2
call bdos ; Bildschirm loeschen
ld de,txt1
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 zusaetzhche RDA
call format ; Bank formatieren
cp 0 ; wurde Bank formatiert ?
jz 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
jz 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 formatieren
cp 0 ; wurde Bank formatiert
jz nz,exit ; --> Bank nicht vorhanden
ld a,0d3h ; 3. Bank formatiert
ld (size),a ; Diskettenkapazitaet eintragen ( 178 k )
ld a,(size) ; Diskettenkapazitaet bestimmen
ld de,txt2
cp 0
jp z,print1 ; Msg. keine zusaetzhche RAM - karte
    
```

```

ld de,txt3 ; 1. Bank
cp 62d
jp z,print1 ; 2. Bank
ld de,txt4
ld c,7eh
jp z,print2 ; 3. Bank
print: ld c,9 ; Meldung ausgeben
call bdos
ld de,txt5
print1: ld c,9 ; Endemeldung ausgeben
call bdos
ld c,0 ; Init. Warmstart
call bdos

;----- Unterprogramme -----

format: ld de,0000h ; ab adr. 0000 auf Bank 1
format: push hl ; Anzahl 128 Byte Bloecke merken
ld hl,sektor ; Adr. Fuehrlbytes -> 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 ; copiert DE -> DE
pop hl ; Anzahl Bloecke vom Stack holen
ld hl ; a,h
or l
jp nz,format ; 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
    
```

```

-----
text1: db cr,1f,1f,1f,tab
      db RAM - Floppy wird formatiert ; Bitte warten , $

text2: db cr,1f,1f,tab
      db F e h l e r , keine zusätzliche RAM - Karte vorhanden , $

text3: db cr,1f,1f,tab
      db 62 k RAM - Floppy formatiert ! , $

text4: db cr,1f,1f,tab
      db 126 k RAM - Floppy formatiert ! , $

text5: db cr,1f,1f,tab
      db 178 k RAM - Floppy formatiert ! , $

text6: db cr,1f,1f,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 Speicherverwendung 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 nach 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
0100 -	5100 <RET>	Ändern ab 100h
0101 -	21 <RET>	So die ganze Routine eingeben
	00 <RET>	Achtung! Kein Byte darf mit einem Buchstaben beginnen, d.h. z.B. statt FF --> OFF eingeben.
011E -	00 <RET>	Ende der Änderung
011F -	CTRL C	gleichzeitig

Bild 2:

2. Abspeichern des Programmes auf Diskette

A)	EBASIC	Kaltstart wie gewohnt
?	C	
		Nachdem ein Programm mit Recorder eingelesen oder selbst geladen ist, kann man das BASIC (läufer mit einem Reset) folgendermaßen verlassen:
>	CLRS	Bildschirm löschen
A)	CALL 0	EXIT
		Mit dem DDT wird das Programm abgespeichert:
A)	DDT	
-	M8000,9FFF,100	nach 100h verschieben
-	CTRL C <RET>	Unter einem beliebigen Namen abspeichern
A)	SAVE 35 TEST1.BAS	

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 aufrufen
-	M100,2100,4000	EBASIC nach 4000h verschiebt
-	5100	Sprung eingeben
0100 -	0C3	JP 4000h
0101 -	00	
0102 -	40	
0103 -	CTRL C	
A)	SAVE 95 EBASIC.COM	März 95 ?
		Letzte Adresse = 5FFFh, Entscheidend ist das höherwertige Byte. Hier 5Fh = 95

Das Einlesen eines Programmes von der Diskette geschieht folgendermaßen:		
A)	DDT TEST1.BAS	beides aufrufen, TEST1.BAS liegt auf Adresse 0100h
-	M100,2044,8000	nach 8000h verschieben
-	CTRL C	
A)		
Jetzt kann EBASIC aufgerufen werden:		
A)	EBASIC	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.


```

0250 DD Z3      INC  IX      ;Zeiger inc.
025F 05        DEC  B       ;Zähler dec.
0260 18 F3      JR    MAKEFCB_FUELL

MAKEFCB_TYP:
INC  DE        ;Pufferzeiger weiter
MAKEFCB_TYPSCHL:
LD   A,B
CP   4         ;Zähler < 4
JR   C,MAKEFCB_SCHL ;ja: Typ eintragen

0268 DD 36 00 20 LD  (IX), ' ' ;nein: auffüllen bis
026C DD Z3      INC  IX      ;zu Dateityp
026E 05        DEC  B       ;
026F 18 F2      JR    MAKEFCB_TYPSCHL

ENDE:
END  START
    
```

Speichern von Programmen auf Adresse 8000H

PAGE 5

```

0005 SYSTEM      0000 WSTARTF
0006 DIRCONF     0009 STRAUSF
0008 STREINF     000F OPENF
0010 CLOSEF     0013 DELETF
0014 READF      0015 WRITEF
0016 MAKEF      0017 RENAMF
001A SETMAF     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
0107 WEITER1    01E7 WEITER
01ED SCHREIB_SCHLEIFE 0210 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=6297
CRC=AA3C

LADEN VON 8000H

PAGE 1

```

;*****
;Hauptprogramm Laden
;Automatische Laden von Anwenderprogrammen für
;EPROM-Software EGRUND1, EGRUND2, EBASIC2, EZASS2
;Lade.ASM
;Christoph Kohler - 21/06/84
;*****
    
```

```

SYSTEM EQU 00005H ;Systemfunktionen

WSTARTF EQU 0 ;Warstart
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
DELETF EQU 19 ;Datei löschen
READF EQU 20 ;aus Datei lesen
WRITEF EQU 21 ;in Datei schreiben
MAKEF EQU 22 ;Datei anlegen
RENAMF EQU 23 ;Datei umbenennen
SETMAF EQU 26 ;DMA-Adresse setzen
MAKEFCB EQU 250 ;FCB - Block kreieren

CR EQU 00H ;Cursor an Zeilenanfang
LF EQU 0AH ;Cursor in neue Zeile
EOF EQU 1AH ;Markierung für Textende
    
```

TITLE LADEN VON 8000H

ORG 100H

```

START:
JP  START1
    
```

0100 C3 98 01

```

0103 4E 61 4D 65 20 64 65 72 FRAGE: DB 'Name der Datei zum Einlesen?',CR,LF,'$'
0123 0D 0A 44 61 74 65 49 20 FEHLER1: DB CR,LF,'Datei nicht gefunden!',CR,LF,'$'
017E 0D 0A 53 70 65 69 43 68 FEHLER2: DB CR,LF,'Speicher voll!',CR,LF,'$'
    
```

LADEN VON 8000H

PAGE 2

```

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

```

0163 00 00          SATZ_ZAHL: DW 0
    
```

```

0165 10 00          PUFFER:   DB 16,0
                                DB 16
    
```

```

0177 00 00 00 00 00 00 00 00 00 EINGABE_FCB: DB 16,0
    
```

START1:

```

0198 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  IX,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   OFFH
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:

```

01D0 11 00 80      LD  DE,8000H      ;Ja: also einlesen
01D3 01 FF FF      LD  BC,-1         ;Adresse zum Einlesen
                                ;Zähler = -1
    
```

LESE_SCHLEIFE:

```

01D6 2A 06 00      LD  HL,(SYSTEM+1) ;Test ob Speicher voll
01D9 2B            DEC  HL            ;Anmerk.: Die Adresse
01DA E0 52          SBC  HL,DE        ;System+1 RAM-Ende
01DC 30 0A          JR   NC,LESE_SCHLEIFE
    
```

```

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:

```

01EB 21 80 00      LD  HL,120        ;120 = Versatz für DMA-Adresse
01ED 19            ADD  HL,DE
01EE 02            INC  BC
01EF C5            PUSH BC
01F0 E5            PUSH HL
    
```

LADEN VON 8000H

PAGE 3

```

01EF 0E 1A          LD  C,SETMAF      ;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 01            POP  DE           ;Adresse restaurieren
01FD A7            AND  A
01FE 2B 08          JR   Z,WEITER1   ;weiter ?
    
```

```

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

WEITER1:

```

0208 C1            POP  BC           ;Zähler restaurieren
0209 79            LD  A,C
020A FE 3F          CP   63           ;Prüfen ob 8k in 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,(IX+1)      ;Pufferlänge laden
021C A7            AND  A
021D CB            RET  Z           ;Länge = 0 --> Fehler
021E 4F            LD  C,A
    
```

```

021F 0D 36 00 00   LD  (IX),0        ;Lautwerk = A:
    
```

```

0223 DD Z3        INC  IX
0225 0C          INC  C
    
```

```

0226 06 08      LD      B,11          ;Länge Dateiname in FCB
                MAKEFCB_SCHL:
0228 00          DEC      C          ;Wiederholen bis Puffer
0229 28 10      JR      Z,MAKEFCB_FUELL ;leer
022B 1A          LD      A,(DEI)
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 ;inein ---> wandeln
0234 06 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 04      JR      Z,MAKEFCB_FUELL   ;ja, füllen
023E 13          INC     DE          ;Pufferzeiger weiter
023F 00 77 00  LD      (IX),A      ;Zeichen in FCB
0242 00 23      INC     IX          ;FCB-leiger weiter
0244 10 E2      DJNZ   MAKEFCB_SCHL      ;Schleife bis Zähler=0
                PAGE 4
LADEN VON 8000H
0246 37          SCF      ;kein Fehler: Carry=1
0247 C9          RET
                MAKEFCB_FUELL:
024B 78          LD      A,B
0249 87          OR      A
024A 37          SCF
024B 08          RET     Z          ;ja: fertig
024C 00 36 00 20 LD      (IX),' '          ;inein: TWR in FCB
0250 00 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
0259 38 0D      JR      C,MAKEFCB_SCHL ;ja: Typ eintragen

```

```

025B 00 36 00 20 LD      (IX),' '          ;inein: auffüllen bis
025F 00 23      INC     IX          ;zu Gateityp
0261 05          DEC     B
0262 18 F2      JR      MAKEFCB_TYPSCHL
                WEITERS:
0264 C3 00 00   JP      0000H
                ENDE:
                END     START
                PAGE 5
LADEN VON 8000H
0005 SYSTEM          0000 WSTARTF
0006 DIRCONF         0009 STRAUSF
000A STREINF         000F OPENF
0010 CLOSER          0013 DELETF
0014 READF          0015 WRITF
0016 MAKEF          0017 RENAMF
001A SETDMF         00FA MAKEFCB
000D CR             000A LF
001A EOF            0100 START
0103 FRAGE          0123 FEHLER1
013E FEHLER2       0152 FEHLER3
0143 SATZ_ZAHL     0165 PUFFER
0177 EINGABE_FCB  0198 START1
0180 WEITER        0186 LESE_SCHLEIFE
01E8 LESE_SCHLEIFE 0208 WEITER1
0219 MAKE_FCB     0228 MAKEFCB_SCHL
0236 MAKEFCB_GROSS 0248 MAKEFCB_FUELL
0255 MAKEFCB_TYP  0256 MAKEFCB_TYPSCHL
0264 WEITERS      0267 ENDE
                no fatal error(s)
                SUM=5C05
                CRC=4F32

```

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-Regi-

```

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:";VD: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 = VD TO BD
    190 US="": IF K<100 THEN US=" " z' Formatierung
    200 IF K<10 THEN US=" "
    210 PRINT$DU," ";US;K; " ";HEX$(K); " ";
        230 FOR I = 7 TO 0 STEP -1
            240 PRINT$DU,SGN(K AND 2^I);
            250 NEXT I
    270 PRINT$DU," ";CHR$(K);ZZ=ZZ+1
280 IF ZZ/60=INT(ZZ/60) THEN FOR T=1 TO 12:PRINT$DU;NEXT T
290 NEXT K
300 END

```

Bild 1: BASIC-Zeilen mit WordStar geschrieben

ster, 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).

```

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$(0) : REM Cursor ein
60 PRINT E$
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-Vektoren 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 Vektoren 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:

$$\begin{aligned} /NMI &= \text{Ramstart} + \$0 \\ /INT &= \text{Ramstart} + \$6 \\ /INT + /NMI &= \text{Ramstart} + \$C \end{aligned}$$

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 Programm 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 separate Leitung zur KEY geführt. Hier muß dann lediglich noch eine Leitung von PIN 15 der Stiftheiste 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 RÜDIGER BÄCKER - 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'
000408 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
= 00000008 INTVEC EQU #08 * ADRESSE FUER INTERRUPTVECTOR (A5)
    
```

```

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:
00042E 46E7 FFFF MOVEM.L A0-A7/D0-D7,-(A7). * AUFRUF DES USERPROGRAMMES
000432 4E89 00000000 JSR BLOAD * ERST ALLE REGISTER RETTEN
* HIER ADRESSE DES AUFRUFENDEN PROGRAMMES
-----
Undefiniertes Symbol
00043B
00043B 4C0F 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 Menu. 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 Menu 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 Menu-Programms beschränken kann. Die Initialisierung erfolgt wiederum über die Bibliothek. Bei Auftreten eines Interrupts erscheint dann das Menu 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 Menu 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 Menu
- Aufruf von Funktionen des Grundprogrammes
- Mausgesteuerte Auswahl aus dem Menu
- Verbesserung des Bildschirmhandlings durch retten der GDP-Register etc.

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

Dr 11.05.86 - 10:46:26
Roll-D.Klein: s8000/08 Assembler 4.0 (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 RÜDIGER BÄCKER - 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'
000408 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 MENU
000420 55AA0180 DC.L #55AA0180 * ACHTUNG, BEMERKUNG BEI INTENDE BEACHTEN !!
000424 525542412D4D45 DC.B 'RUBA-MEN'
000428 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
000448 0008
00044C 46FC 2000 MOVE #2000,SR * INTERRUPTS FREIGEBEN
000450 4E75 RTS
000452
000452 INTHAND: * HAUPTPROGRAMMSTART, AUFRUF BEI INTERRUPT
000452 4B77 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 9340 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 * SCHRIFTSRESSE
00046A 123C 0000 MOVE.B #0,D1 * Y-POSITION
00046E 143C 000C MOVE.B #20,D2 * Y-POSITION
000472 3E3C 000A MOVE #WRITE,D7 * UND AUSGEBEN
000476 4E41 TRAP #1
000478 0442 0020 SUB #20,D2 * NEUE Y-POSITION
00047C 103C 0020 MOVE.B #20,D0 * NEUE SCHRIFTSRESSE
000480 41FA 0178 LEA DTX2(PC),A0 * USW. ....
000484 3E3D 000A MOVE #WRITE,D7
000488 4E41 TRAP #1
00048A 0442 0010 SUB #20,D2
00048E 103B 0021 MOVE.B #21,D0
Roll-D.Klein: s8000/08 Assembler 4.0 (C) 1984, Seite 2
000492 41FA 0183 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 018D LEA TXT1(PC),A0
0004A8 3E3C 000A MOVE #WRITE,D7
0004AC 4E41 TRAP #1
0004AE 0442 000A SUB #10,D2
0004B2 41FA 018F 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 TXTB(PC),A0
00050A 3E3C 000A MOVE #WRITE,D7
00050E 4E41 TRAP #1
000510 0442 000A SUB #10,D2
000514 41FA 0194 LEA TXTB(PC),A0
000518 3E3C 000A MOVE #WRITE,D7
00051C 4E41 TRAP #1
00051E 0442 000A SUB #10,D2
000522 41FA 0194 LEA NTAT(PC),A0
000526 3E3C 000A MOVE #WRITE,D7
00052A 4E41 TRAP #1
00052C
00052E 3E3C 000C MOVE #1C1,D7 * AUSWAHLEN
000530 4E41 TRAP #1
000532
000534 3F00 MOVE D0,-(A7) * DANN ZUR SICHERHEIT BILDSCHIRM LOESCHEN
000538 3E3C 0010 MOVE #CLR,D7 * DO WIRD GERETTET, DA IN CLR BENUTZT UND
00053C 4E41 TRAP #1 * DIE NUMMER DES GEWAELHTEN UNTERPROGRAMMES
00053A 301F MOVE (A7)+,D0 * NOCH IN DO STEHT
00053C
00053C 0000 0031 CMP.B #1',D0 * PROGRAMM 1 AUSGEWAHLT ?
000540 6700 004E BEQ PROG1 * JA, DANN DORTHIN
000544 0000 0032 CMP.B #2',D0 * SONST WEITER ...
000548 6700 0050 BEQ PROG2
00054C 0000 0032 CMP.B #3',D0
000550 6700 0052 BEQ PROG3
000554 0000 0034 CMP.B #4',D0
000558 6700 0054 BEQ PROG4
00055C 0000 0035 CMP.B #5',D0
000560 6700 0056 BEQ PROG5
000564 0000 0036 CMP.B #6',D0
    
```

Rolf-D.Klein ©8000/08 Assembler 4.0 (C) 1984, Seite 7

```

000568 6700 0058 BEQ PROG6
00056C 0000 0037 CMP.B #7',D0
000570 6700 005A BEQ PROG7
000574 0000 0038 CMP.B #8',D0
000578 6700 005C BEQ PROG8
00057C 0000 0039 CMP.B #9',D0
000580 6700 005E BEQ PROG9
000584 0000 0060 CMP.B #a',D0
000588 6700 005C BEQ INTENDE
00058C 6000 FED4 BRA START * WENN KEINE GUELTIGE AUSWAHL
000590
000590 PROG1: * NUN DAS GEWAELHTE PROGRAMM AUSFUEHREN
000590 4EB9 00001420 JSR PROG1A * MIT JSR, DA ADRESSDISTANZ NICHT KLAR
000596 6000 004E BRA INTENDE
00059A
00059A PROG2:
00059A 4EB9 00002820 JSR PROG2A * PROGRAMM AUSFUEHREN
0005A0 6000 0044 BRA INTENDE
0005A4
0005A4 PROG3: * PROGRAMM AUSFUEHREN
0005A4 4EB9 00001C20 JSR PROG3A
0005AA 6000 003A BRA INTENDE
0005AE
0005AE PROG4: * PROGRAMM AUSFUEHREN
0005AE 4EB9 00001820 JSR PROG4A
0005B4 6000 0030 BRA INTENDE
0005B8
0005B8 PROG5: * PROGRAMM AUSFUEHREN
0005B8 4EB9 00003000 JSR PROG5A
0005BE 6000 0026 BRA INTENDE
0005C2
0005C2 PROG6: * PROGRAMM AUSFUEHREN
0005C2 4EB9 00001218 JSR PROG6A
0005C8 6000 001C BRA INTENDE
0005CC
0005CC PROG7: * PROGRAMM AUSFUEHREN
0005CC 4EB9 00002028 JSR PROG7A
0005D2 6000 0012 BRA INTENDE
0005D6
0005D6 PROG8: * PROGRAMM AUSFUEHREN
0005D6 4EB9 00006020 JSR PROG8A
0005DC 6000 0008 BRA INTENDE
0005E0
0005E0 PROG9: * PROGRAMM AUSFUEHREN
0005E0 4EB9 00002420 JSR PROG9A
    
```

```

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 GPRPG.
0005EC
0005EC ** PROGRAMMTABELLE : * HIER DIE STARTADRESSEN DER PROGRAMME
0005EC * EINTRAGEN
0005EC
= 00001420 PROG1A EQU #1420
= 00002820 PROG2A EQU #2820
= 00001C20 PROG3A EQU #1C20
= 00001820 PROG4A EQU #1820
= 00003000 PROG5A EQU #3000
= 00001218 PROG6A EQU #1218
= 00002028 PROG7A EQU #2028
= 00006020 PROG8A EQU #6020
= 00002420 PROG9A EQU #2420
0005EC
    
```

Rolf-D.Klein ©8000/08 Assembler 4.0 (C) 1984, Seite 8

```

0005EC ** TEXTE:
0005EC
0005EC DTXT1:
0005EC 494E5420D2048 DC.B 'INT - Handler',0
0005F3 616E46C657200
0005FA DTXT2:
0005FA 28422920313938 DC.B '(C) 1986 by Ruediger Baecker',0
000601 76206279205275
000608 65646967657220
00060F 4216563686572
000616 00
000617 DTXT3:
000617 426974746E2050 DC.B 'Bitte Programm auswaehlen : ',0
00061E 726F9772516060
000625 20617573776165
00062C 686C656E203A00
000633 DTXT4:
000633 312030303E2052 DC.B '1 ==> RUBATRANS',0
00063A 5542415452414E
000641 5700
000643 DTXT5:
000643 322030303E2052 DC.B '2 ==> RUBAFILL',0
00064A 55424146494C40
000651 00
000652 DTXT6:
000652 332030303E2052 DC.B '3 ==> RUBADUMP',0
000659 55424144554050
000660 00
000661 DTXT7:
000661 342030303E2056 DC.B '4 ==> VERIFY',0
000668 455249465900
00066E DTXT8:
00066E 352030303E2052 DC.B '5 ==> RAM-USR',0
000675 41402D55352000
00067C DTXT9:
00067C 362030303E2051 DC.B '6 ==> QUICKPROM',0
000683 5549424850524F
00068A 4000
00068C DTXT10:
00068C 372030303E2044 DC.B '7 ==> 4SEMPINT',0
000693 53454050494E54
00069A 00
00069E DTXT11:
00069E 382030303E2048 DC.B '8 ==> HARDCOPY',0
0006A2 4152444434F50509
0006A9 00
0006AA DTXT12:
0006AA 392030303E2054 DC.B '9 ==> TEXT 68',0
0006B1 45585420262800
0006B8 DTXT13:
0006B8 402030303E2045 DC.B 'a ==> Ende',0
0006BF 6E446500
0006C3 ENDEA:
0006C3 ENDU.
0006C3
0006C0 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. 0 7144/2 42 46

```
(* Programm LÖTTO H.Schiedek 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[j] := z;
end; (* von for i *)
end;
clrscr;
writeln (chr(27),'z1'); (* deutscher Zeichensatz einsch. *)
writeln (chr(27),chr(27),'G'); (* Grafikmodus einschalten *)
writeln ('M10 230'); (* Cursor setzen *)
writeln ('G3 $22'); (* GDP Port 73 Schrift grösse *)
writeln ('Bich empfehle DIR als Lottozahlen : ');
```

```
(* Cursor 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 (* Textstärkt 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 auszugebendes 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 #!b,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 #!b,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 VDM 03.4.1984
009C00          *PROGRAMM ZUM ERSTELLEN VON ZEICHNUNGEN
009C00          *(C) 1984 HEINRICH ZAREK, 5483 AHRWEILER
010000          DRG $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 -0- D +1 F+ 5
010000          *           I
010000          *           X - 1
010000          *           C - 5
010000          *
010000          *           FADENKREUZPOSITIONSANZEIGE
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          *           ABSPEICHERUNG 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          *           *** VEREINBARUNGSAUSTEIN *****
010000          *
010000          *           KOOR EQU 12  *ANZAHL BYTES PRO EINTRAG
010000          *           X1 EQU 0    *OFFSET ADRESSIERUNG
010000          *           Y1 EQU 4
010000          *           ST EQU 8
010000          *
010000          *           *** VARIABLEN EINGABE ***
010000          *           AX: DC.L 0  *WERT AUSSCHNITTSRAHMEN
010000          *
010000          *           Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 2
010004          *
010004          *           AY: DC.L 0
010008          *           PX: DC.L 0  *WERT ABSOLUT
01000C          *           PY: DC.L 0
010010          *           EX: DC.L 0  *WERT ZEICHENAUSSCHNITT
010014          *           EY: DC.L 0
010018          *           SX: DC.L 0  *WERT SCHILDKRÄTTE
01001C          *           SY: DC.L 0
010020          *           YA: DC.L 0
010024          *           YA: DC.L 0
010028          *           ANZAHL: DC.L 0
01002C          *           KOOP:DC.L 0
010030          *           STIFT:DC.L 1
010034          *
010034          *           *** VARIABLEN AUSGABE ***
010034          *
010034          *           XAP:DC.L 0  *X-ANFANG
010038          *           YAP:DC.L 0  *Y-ANFANG
01003C          *           XE:DC.L 0   *X-ENDE
010040          *           YE:DC.L 0   *Y-ENDE
010044          *           PXS:DC.L 0  *POSITION X-STIFT
    
```

```

01004B          00000000          PYS:DC.L 0          *POSITION Y-STIFT
01004C          00000000          STIFTP:DC.L 0       *STIFTP HOCH ODER GEBENKT
010050          00000000          MAAGR:DC.L 0       *WERT FÜR LINKS ODER RECHTS
010054          00000000          SENKR:DC.L 0       *WERT FÜR HOCH ODER TIEF
01005B          00000000          ISCHRITT:DC.L 0    *WERT FÜR ADDITION POSITIV ODER NEGATIV
01005C          00000000          YSCHRITT:DC.L 0    *WERT FÜR ADDITION POSITIV ODER NEGATIV
010060
010060          103C 00B8          START: MOVE.B #5B,D0 *PROGRAMMBEGINN
010064          323C 007B          MOVE #120,D1
010068          343C 00B0          MOVE #128,D2
01006C          41F9 0010AFE        LEA TEX1,A0          *TITEL
010072          4EB9 0001386        JSR $WRITE
010078          23FC 0000029        MOVE.L #41,AX
01007E          00010000          MOVE.L #11,AY
010082          23FC 0000000B        MOVE.L #11,AY
010088          00010004          LEA KOOR0,A0        *KOORDINATENVERZEICHNISSANFANGSKOORDINATEN LADEN
01008C          41F9 00010BDC        MOVE.L A0,KOOP
010092          23BC 0001002C        JSR SCHLEIFE
010098          4EB9 00010AEA
01009E
01009E          4EB9 00000DA0        MENUE: JSR $CLR      *MENÜ
0100A4          103C 0044          MOVE.B #344,D0
0100AB          323C 007B          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 007B          MOVE #120,D1
0100C4          343C 0096          MOVE #150,D2
0100CB          41F9 00010B07        LEA TEX3,A0
0100CE          4EB9 00001386        JSR $WRITE
0100D4          323C 007B          MOVE #120,D1
0100DB          343C 007B          MOVE #120,D2
0100DC          41F9 00010B16        LEA TEX5,A0
0100E2          4EB9 00001386        JSR $WRITE
0100E8
0100E8          4EB9 00000A00        MAHL: JSR $CI       *AUSWAHL DER PROGRAMMPUNKTE
0100EE          0C00 0031          CMP.B #0',D0
0100F2          6606          BNE.S ST1
0100F4          4EB9 00010108        JSR EINGABE
0100FA          0C00 0032          ST1:  CMP.B #2',D0
0100FE          66EB          BNE.S MAHL
010100          4EB9 000106A8        JSR AUSGABE
010106          60E0          BRA.S MAHL
    
```

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 00001386        JSR $WRITE
01013E          323C 0028          MOVE #40,D1        *BILDRAHMEN
010142          343C 000A          MOVE #10,D2
010146          4EB9 00000EE2        JSR $MOVEV
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 00010B08        LEA TABELLE,A0
010196          303C 000F          MOVE #15,D0
01019A          4EB9 000032BC        JSR $FIGUR          *AUSSCHNITTGRÖßE
0101A0          41F9 00010B03        LEA BUF0,A0
0101A6          2039 00010010        MOVE.L EX,D0
0101AC          4EB9 000015FC        JSR $PRINT4
0101B2          41F9 00010B03        LEA BUF0,A0
0101B8          303C 0022          MOVE #22,D0
0101BC          323C 007B          MOVE #120,D1
0101C0          343C 00C8          MOVE #200,D2
0101C4          4EB9 00001386        JSR $WRITE
0101CA          41F9 00010B07        LEA BUF0,A0
0101D0          2039 00010014        MOVE.L EY,D0
    
```

```

010106 4EB9 000015FC JSR $PRINT4D
01010C 41F9 00010B07 LEA BUFY,A0
0101E2 303C 0022 MOVE #22,D0
0101E6 323C 0118 MOVE #280,D1
0101EA 343C 00C8 MOVE #200,D2
0101EE 4EB9 00001386 JSR $WRITE
0101F4
0101FA 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 00010184 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 00010184 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 00010184 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 00010184 JSR FIG
0102AE 6000 FF44 BRA JUST
0102B2 0C00 0043 ST14: CMP.B #'C',D0
0102B6 6600 FF3C BNE JUST
0102BA 4EB9 000102C4 JMP AUSSCHNITT
0102C0 6000 FF32 BRA JUST
0102C4
AUSSCHNITT: *BILDSCHIRMAUSSCHNITT
0102C4 4EB9 00000DA0 JSR $CLR
0102CA 2A39 00010028 MOVE.L ANZAHL_DS+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
0102DE 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 $KODOR,A0
010304 2228 0000 SCH: MOVE.L X1(A0),D1
010308 92B9 00010010 SUB.L EX,D1
01030E C3FC 0002 MULS #2,D1
010312 2428 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 ABDA.L #KODOR,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 *SCHILDKRÄTE POSITIONIEREN
010358 92B9 00010010 SUB.L EX,D1
01035E C3FC 0002 MULS #2,D1
010362 23C1 00010018 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 *SCHILDKRÄTENSTEUERUNG
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
0103DB 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 #'Y',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 FFS6 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
010554 6600 0062 BNE ST28
010558 4282 CLR.L D2
01055A 2239 00010020 MOVE.L YA,D1
010560 2439 00010024 MOVE.L YD,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 9489 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 SSET
0105AE 4EB9 000105F2 JSR SPEICHERN
0105B4 6000 FDB4 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,ST1FT
0105CC 00010030 BRA TAS
0105D0 6000 FDB8 ST29: CMP.B # 'M',D0
0105D4 0C00 004D BNE TAS
0105DB 6600 FDB0 JSR $GRAPOFF
0105DC 4EB9 0000110A JSR $HIDE
0105E2 4EB9 00001112 JSR $MENUE
0105E8 4EB9 0001009E BRA TAS
0105EE 6000 FDBA
0105F2 SPEICHERN: #ZEICHNEN UND KOORDINATEN ABSPEICHERN
0105F2 4281 CLR.L D1
0105F4 2079 0001002C MOVEA.L KOOP,AO #WERT KOORDINATENVERZEICHNIS-ANFANG
    
```

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

```

0105FA 06B9 00000001 ADD.L #1,ANZAHL #ZAHLEN DER GESPEICHERTEN KOORDINATEN
010600 0001002B
010604 2239 00010008 MOVE.L PX,D1 #PX NACH D1
01060A 2141 0000 MOVE.L D1,I(A0)+D1 NACH KOORDINATENVERZEICHNIS
01060E 2239 0001000C MOVE.L PY,D1 #PY NACH D1
010614 2141 0004 MOVE.L D1,Y(A0)+D1 NACH KOORDINATENVERZEICHNIS
010618 2239 00010030 MOVE.L ST1FT,D1 #ST1FT NACH D1
01061E 2141 0008 MOVE.L D1,ST(A0)+D1 NACH ST
010622 23FC 00000001 MOVE.L #1,ST1FT #ST1FT AUF ZEICHNEN SETZEN
010628 00010030
01062C D1FC 0000000C ADDA.L #KOOP,AO #WERT KOORDINATENVERZEICHNIS ERHÖHEN
010632 23CB 0001002C MOVE.L AO,KOOP
010638 4E75 RTS
01063A
01063A 103C 0011 ANZEIG: MOVE.B #11,D0 #KOORDINATENANZEIGE
01063E 323C 005A MOVE #90,D1
010642 343C 00F5 MOVE #245,D2
010646 41F9 00010B3D LEA TEX12,AO # X= UND Y= WERT
01064C 4EB9 00001386 JSR $WRITE
010652 41F9 00010B33 LEA BUF,X,AO #TEXT AUF BILDSCHIRM
010658 2039 00010008 MOVE.L PX,D0
01065E 4EB9 000015FC JSR $PRINT4D
010664 41F9 00010B33 LEA BUF,X,AO
01066A 103C 0011 MOVE.B #11,D0
01066E 323C 004E MOVE #110,D1
010672 343C 00F5 MOVE #245,D2
010676 4EB9 00001386 JSR $WRITE
01067C 41F9 00010B37 LEA BUF,X,AO
010682 2039 0001000C MOVE.L PY,D0
010688 4EB9 000015FC JSR $PRINT4D
01068E 41F9 00010B37 LEA BUF,X,AO
010694 103C 0011 MOVE.B #11,D0
010698 323C 008E MOVE #190,D1
01069C 343C 00F5 MOVE #245,D2
0106A0 4EB9 00001386 JSR $WRITE
0106A6 4E75 RTS
0106AB 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 00010B58 LEA TEX30,AO
0106C0 4EB9 00001386 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,AO
0106D8 4EB9 00001386 JSR $WRITE
0106DE 323C 0078 MOVE #120,D1
0106E2 343C 0078 MOVE #120,D2
0106E6 41F9 00010B6D LEA TEX32,AO
0106EC 4EB9 00001386 JSR $WRITE
0106F2 323C 0078 MOVE #120,D1
0106F6 343C 005A MOVE #90,D2
0106FA 41F9 00010B7A LEA TEX33,AO
010700 4EB9 00001386 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 JUSTIEREN
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 $MOVEVO
01074A 41F9 00010BDC LEA KOOP,AO
010750 2A39 00010028 MOVE.L ANZAHL,D5
010756 0C85 00000000 CMP.L #0,D5 #FESTSTELLEN OB KOORDINATEN VORHANDEN
01075C 6700 F940 BEQ $MENUE
010760 0485 00000001 SUB.L #1,D5
010766 2228 0000 MID: MOVE.L X1(A0),D1
01076A # DIVS #0,D1 #GRÖÖE DER BILDSCHIRMAUSGABE
01076A 2428 0004 MOVE.L Y1(A0),D2
01076E 85FC 0002 DIVS #0,D2 #GRÖÖE DER BILDSCHIRMAUSGABE
010772 2028 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 #KOOP,AO
01079E 51CD FFC6 DBRA D5,MID
0107A2 4EB9 00010AEA JSR $SCHLEIFE
0107AB 4EB9 0001009E JSR $MENUE
0107AE
0107AE JUSTIEREN: #PLOTTERSTIFT POSITIONIEREN
0107AE 4EB9 00000DA0 JSR $CLR
0107BA 303C 0022 MOVE #422,D0
0107BB 323C 0078 MOVE #120,D1
0107BC 343C 0096 MOVE #150,D2
0107C0 41F9 00010B87 LEA TEX34,AO
0107C6 4EB9 00001386 JSR $WRITE
0107CC 323C 0078 MOVE #120,D1
0107D0 343C 0078 MOVE #120,D2
0107D4 41F9 00010B94 LEA TEX35,AO
0107DA 4EB9 00001386 JSR $WRITE
0107E0 323C 0078 MOVE #120,D1
0107E4 343C 005A MOVE #90,D2
0107EB 41F9 00010BA1 LEA TEX36,AO
0107EE 4EB9 00001386 JSR $WRITE
0107F4 323C 0078 MOVE #120,D1
0107FB 343C 003C MOVE #60,D2
0107FC 41F9 00010BAE LEA TEX37,AO
010802 4EB9 00001386 JSR $WRITE
010808 323C 0078 MOVE #120,D1
01080C 343C 001E MOVE #30,D2
010810 41F9 00010BBB LEA TEX38,AO
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
01088B 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
0108BB 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 00010A48 JSR AUSGABE #ZUR AUSGABE
0108F0 6000 FF2A BRA JUS
    
```

```

0108F4 4279 00010034 LINIE: CLR XAP
0108FA 4279 00010038 CLR XAP
010900 4279 00010044 CLR PXS
010906 4279 00010048 CLR PYS
01090C 4279 00010058 CLR XSCHRITT
010912 4279 0001005C CLR XSCHRITT
010918 4286 CLR.L D6
01091A 41F9 000108DC LEA KCOORD,A0 #KORDINATEN LADEN
010920 2A39 00010028 MOVE.L ANZAHL,D5
010926 0485 00000001 SUB.L #1,D5
01092C 2628 0000 MDH: 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 #DELTA X UND Y BERECHNEN
010956 96B9 00010034 SUB.L XAP,D3 #XE-XAP=DELTA X
01095C 0CB3 00000000 CMP.L #0,D3 #FESTSTELLEN OB DELTA X NEGATIV
010962 6800 0134 BMI LINKS #NEGATIV
010966 6A00 0116 BPL RECHTS #POSITIV
01096A
01096A 98B9 00010038 DELY: SUB.L YAP,D4 #YE-YAP=DELTA Y
010970 0CB4 00000000 CMP.L #0,D4 #FESTSTELLEN OB DELTA Y POSITIV
    
```

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 10

```

010976 6800 0156 BMI TIEF #NEGATIV
01097A 6A00 0138 BPL HOCH #POSITIV
01097E
01097E 4280 STAR2: CLR.L D0
010980 8A84 CMP.L D4,D3 #PRÜFEN OB DELTA X ODER DELTA Y GRÖßER
010982 6E00 0054 BGT DMI
010986
010986 9CB3 DPL: SUB.L D3,D6 #KLEINER
010988 2039 0001005C MOVE.L XSCHRITT,D0
01098E D1B9 00010048 ADD.L D0,PYS
010994 2039 0001004C MOVE.L STIFTP,D0
01099A 4EB9 00000028 JSR $LD
0109A0 2039 00010054 MOVE.L SENKR,D0
0109A6 4EB9 00000028 JSR $LD
0109AC 203C 00000000 MOVE.L #0,D0
0109B2 4EB9 00000028 JSR $LD
0109B8 203C 00000001 MOVE.L #1,D0
    
```

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 11

```

010A98 23FC 00000005 LINKS: MOVE.L #5,WAAGR
010A9E 00010050
010AA2 23FC FFFFFFFF MOVE.L #-1,XSCHRITT
010AAB 00010058
010AAC 44B3 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
010ACB 4EF9 0001097E JMP STAR2
010ACE 23FC 00000028 TIEF: MOVE.L #40,SENKR
010AD4 00010054
010ADB 23FC FFFFFFFF MOVE.L #-1,YSCHRITT
010ADE 0001005C
010AE2 44B4 NEG.L D4
010AE4 4EF9 0001097E JMP STAR2
010AEA SCHLEIFE:
010AEA 363C 0063 MOVE #100,-1,D3
010AEE 4EB9 00000E38 WARTE: JSR $SYNCR
010AF4 6700 FFFB BEQ WARTE
010AFB 51CB FFF4 DBRA D3,WARTE
010AFC 4E75 RTS
010AFA
    
```

```

010AFE 43414400 TEX1:DC.B 'CAD',0
010B02 4045EDD00 TEX2:DC.B 'MENÜ',0
010B07 45A9AE47414245 TEX3:DC.B 'EINGABE' 1',0
010B0E 20202020202031
010B15 00
010B16 41555347414245 TEX5:DC.B 'AUSGABE' 2',0
010B1D 20202020202032
010B24 00
010B25
010B25 45A9AE47414245 TEX11: DC.B 'EINGABE BILDAUSSCHNITT',0
010B2C 204249AC444155
010B33 53534348AE4954
010B3A 542000
010B3D 583B2020202020 TEX12: DC.B 'X=' UND Y=' WERT',0
010B44 2020554E442059
010B4B 30202020202020
010B52 205745525400
010B58 41555347414245 TEX30: DC.B 'AUSGABE',0
010B5F 00
010B60 4249AC44534348 TEX31: DC.B 'BILDSCHIRM 1',0
010B67 49524D203100
010B6D 504CF54544552 TEX32: DC.B 'PLOTTER' 2',0
010B74 202020203200
010B7A 4A55354494552 TEX33: DC.B 'JUSTIERUNG 3',0
010B81 554E47203300
    
```

```

0109BE 4EB9 00000028 JSR $LD
0109C4 2039 0001004C MOVE.L STIFTP,D0
0109CA 4EB9 00000028 JSR $LD
0109D0 4280 CLR.L D0
0109D2 4EF9 00010A24 JMP TEST
0109DB 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 00000028 JSR $LD
0109F2 2039 00010050 MOVE.L WAAGR,D0
0109F8 4EB9 00000028 JSR $LD
0109FE 203C 00000000 MOVE.L #0,D0
010A04 4EB9 00000028 JSR $LD
010A0A 203C 00000001 MOVE.L #1,D0
010A10 4EB9 00000028 JSR $LD
010A16 2039 0001004C MOVE.L STIFTP,D0
010A1C 4EB9 00000028 JSR $LD
010A22 4280 CLR.L D0
010A24
010A24 0CB6 00000000 TEST: CMP.L #0,D6
010A2A 6700 000A BEQ ZWEIG #D=0
010A2E 6E00 FFF5 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 B4B1 CMP.L D1,D2 #PXS*YE STAR2
010A44 6600 FFB8 BNE STAR2
010A4B 2239 00010048 MOVE.L PYS,D1
010A4E 2439 00010040 MOVE.L YE,D2
010A54 B4B1 CMP.L D1,D2
010A56 6600 FFB8 BNE STAR2 #PYS*YE STAR2
010A5A 23F9 0001003C MOVE.L XE,YAP #X-E UND YE NACH X-A UND Y-A VERSCHIEDEN
010A60 00010034
010A64 23F9 00010040 MOVE.L YE,YAP
010A6A 00010038
010A6E D1FC 0000000C ADDA.L #KORD,A0
010A74 51CD FEBA DBRA D5,MDH
010A7B 4EB9 0001009E JSR MENUE
010A7E
010A7E ***** UNTERPROGRAMM STIFTSTEUERUNG*****
010A7E 23FC 00000004 RECHTS: MOVE.L #4,WAAGR
010A84 00010050 MOVE.L #1,XSCHRITT
010A88 23FC 00000001
010A8E 00010058
010A92 4EF9 0001096A JMP DELY
    
```

**** UNTERPROGRAMM STIFTSTEUERUNG*****

RECHTS: MOVE.L #1,XSCHRITT

JMP DELY

```

010B87 52454348545320 TEX34: DC.B 'RECHTS' D',0
010B8E 202020204400
010B94 4C49AE48532020 TEX35: DC.B 'LINKS' S',0
010B9B 202020205300
010BA1 484F4348202020 TEX36: DC.B 'HOCH' E',0
010BAB 202020204500
010BAE 54494546202020 TEX37: DC.B 'TIEF' X',0
010BB5 202020205800
010BBB 41555347414245 TEX38: DC.B 'AUSGABE' A',0
010BC2 202020204100
010BCB
010BCB 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
    
```

Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 12

```

010BD3 BUFI: DS.B 4
010BD7 BUFI: DS.B 4
010BDB 00 DS.W 0
010BDC KORD:
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-Baugruppe 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 ausgegeben.

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 Edi-

tor 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
ettxt equ $3e
eottxt equ $42
txtspal equ $fff
letztzei equ $ffd
leerzei equ $ffe
zeichnein equ $ffc
rest equ $ffb
zustaby equ $ffa
buffer equ $f7
count equ $ff6
leer equ $ff5
schirm equ $f000

start:
  clr.b txtspal(a5)
  clr.b letztzei(a5)
  clr.b leerzei(a5)
  clr.b zustaby(a5)
  clr.b count(a5)
  movea.l stxtxt(a5),a0
  lea schirm(a5),a1
  ;löschen aller Parameter
  ;Aktuelle Text Adr. laden
  ;Zwischenspeicher laden

formcopy:
  move.b (a0)+,d0
  beq.s fertig
  move.b d0,(a1)+
  bra.s formcopy
  ;Text in Zwischenspeicher
  ;weiter bis Null

fertig:
  clr.b (a1)
  lea buffer(a5),a0
  move #21,d0
  move #100,d1
  move #100,d2
  move #2,d5
  jsr @read
  lea buffer(a5),a0
  jsr @wert
  ;lese Anzahl der Zeichen auf die
  ;formatiert werden soll
  ;umrechnen

  move.b d0,txtspal(a5)
  movea.l stxtxt(a5),a2
  lea schirm(a5),a0
  ;Wert abspeichern
  ;Formatierter Text in Zieladr.ablegen

format1:
  movea.l a0,a1
  clr d4
  move.b txtspal(a5),d4
  move d4,d5
  subq #1,d5
  ;Adresse merken
  ;Schleife bestimmen

forleer:
  addq.b #1,count(a5)

forleer1:
  move.b (a0)+,d0
  beq format2
  cmp.b #' ',d0
  beq forleer
  cmp.b #j,d0
  beq ctrlr
  cmp.b #'e',d0
  beq dummy1
  bclr #2,zustaby(a5)
  dbra d5,forleer
  ;Zeichen = 0
  ;Leerzeichen gefunden?
  ;Zeilenumbruch?
  ;Dummy?
  ;Nein,dann nächstes Zeichen

format2:
  bclr #2,zustaby(a5)
  movea.l a1,a0
  clr.b zeichnein(a5)
  clr d2
  clr.l d1
  move.b txtspal(a5),d1
  sub.b letztzei(a5),d1
  move.b leer(a5),d2
  beq keinnull
  ;Adresse zurück
  ;Anzahl Zeichen pro Zeile
  ;Minus dem letzten gefundenen Zeichen
  ;Kein Leerzeichen vorhanden?

format21:
  divs d2,d1
  move.b d1,zeichnein(a5)
  swap d1
  tst.b d1
  beq keinrest
  bset #1,zustaby(a5)
  move.b d1,rest(a5)
  swap d1
  ;dividiere Anzahl Leerzeichen durch Anzahl
  ;des übriggebliebenen Zeichen u. speichern
  ;des ganzzahligen Teils
  ;wenn Rest vorhanden auch abspeichern

format3:
  clr d4
  move.b txtspal(a5),d4
  move d4,d5
  subq #1,d5
  ;hier beginnt zurückspeichern einer for-
  ;matierten Zeile an alte Adresse

format4:
  move.b (a0)+,d0
  beq fin
  cmp.b #' ',d0
  beq einleer
  cmp.b #f,d0
  beq ctrlleer
  bclr #2,zustaby(a5)
  ;bis Null gefunden,dann neue Zeile
  ;Leerzeichen? dann evtl. Leerzeichen einf.
  ;CR? dann auch Leerzeichen einf.

format41:
  move.b d0,(a2)+
  dbra d5,format4
  ;und zurückkopieren
  ;neues Zeichen

format42:
  move.b #f,d1,(a2)+
  move.b #f,a,(a2)+
  ;am Ende CR
  ;und line feed

  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
  beq fin
  cmp.b #' ',d0
  beq format12
  cmp.b #f,d0
  beq format12
  cmp.b #f,a,d0
  beq format12
  bra format1
  ;Null? dann Ende
  ;Leerzeichen? dann neues Zeichen
  ;CR? dann neues Zeichen
  ;Line feed? dann neues Zeichen

format12:
  addq #1,a0
  bra format11
  ;einfach nächstes Zeichen
  ;nächstes Zeichen testen

fin:
  clr.b (a2)
  movea.l stxtxt(a5),a0
  movea.l a0,akttxt(a5)

null:
  tst.b (a0)+
  bne null
  subq.l #1,a0
  movea.l a0,ettxt(a5)
  movea.l a0,eottxt(a5)
  jsr @edit
  rts
  ;sämtlich Text Adressen neu definieren
  ;und Editor aufrufen
  ;Programm Ende

ctrlr:
  addq #1,a0
  ;Bei CR einfach nächstes Zeichen
  ;und dafür als Leerzeichen verarbeiten

forleer:
  btst #2,zustaby(a5)
  bne forleer1
  move.b leerzei(a5),leer(a5)
  addq.b #1,leerzei(a5)
  move.b count(a5),d1
  subq.b #1,d1
  move.b d1,leerzei(a5)
  bset #2,zustaby(a5)
  dbra d5,forleer
  ;Anzahl der Leerzeichen +1
  ;letztes Zeichen in einer Zeile

keinrest:
  bclr #1,zustaby(a5)
  swap d1
  bra format3
  ;aus Division kein Rest, daher nicht speichern

ctrlleer:
  addq #1,a0
  move.b #' ',d0
  ;für CR einfach ' ' einsetzen

einleer:
  btst #2,zustaby(a5)
  bne format4
  clr d1
  move.b zeichnein(a5),d1
  beq leer3
  move d1,d4
  subq #1,d4

leer2:
  move.b #' ',(a2)+
  subq #1,d5
  dbra d4,leer2
  ;bei jedem Leerzeichen evtl. weiteres Leer-
  ;zeichen einsetzen

leer3:
  bset #2,zustaby(a5)
  btst #1,zustaby(a5)
  beq leerend
  move.b rest(a5),d1
  beq restende
  move.b #' ',(a2)+
  subq.b #1,rest(a5)
  subq #1,d5
  ;bei Rest aus Division auch weitere Leerzeich
  ;einfügen
  ;reduziert Rest um -1

leerend:
  bra format41

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

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

dummy1:
  movea.l a1,a0
  ;kein Formatieren dieser Zeile

dummy2:
  move.b (a0)+,d0
  cmp.b #'e',d0
  beq format42
  move.b d0,(a2)+
  bra.s dummy2
  ;einfach Zeile unformatiert zurückschreiben

end

```


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 Bildschirmediator.

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:*: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      GDEM01  ASM 1
GDEM01   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     KLAUIER ASM 3
KLAUIER  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     ICOPY    ASM 6
ICOPY    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 automati-

sieren. 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 KONFIGURATIONSABHÄNGIG !!
!! 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 00AEE9
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

CP/M 68 K, C UND MODULA

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 Farbbrillanz für meine Anwendung nur ein Thermotransferdrucker in Frage kommt. So fiel die Wahl auf den ITHO TPX 80. Bei diesem Drucker wird, wie in dieser Kategorie 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)); (* Einzelementmodus *)
    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
          atr,y:=0;
        END;
        CASE Farbe[n,x,y] OF
          1 :
            af1,y:=5;
            af2,y:=0;
            af3,y:=5;
            af4,y:=0;
          2 :
            af1,y:=15;
            af2,y:=15;
            af3,y:=15;
            af4,y:=15;
          3 :
            af1,y:=1;
            af2,y:=4;
            af3,y:=2;
            af4,y:=8;
        END;
      END;
    END;
  END;
END;
```

```
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
      Farbe1,x,y:=0;
      Farbe2,x,y:=0;
      Farbe3,x,y:=0;
      Farbe3,x,y:=0;
      wert:=getcol(256-x,y-1+yoff);
      CASE wert OF
        0..9 : Farbe1,x,y:=2;
              Farbe2,x,y:=2;
              Farbe3,x,y:=2; (* schwarz *)
        10..19 : Farbe2,x,y:=2;
                  Farbe3,x,y:=2; (* dunkelblau *)
        20..29 : Farbe1,x,y:=2;
                  Farbe2,x,y:=1;
                  Farbe3,x,y:=2; (* dunkelgruen *)
        30..39 : Farbe1,x,y:=2;
                  Farbe3,x,y:=2; (* hellgruen *)
        40..49 : Farbe2,x,y:=1;
                  Farbe3,x,y:=2; (* blau *)
        50..59 : Farbe1,x,y:=1;

```

```

60..69 : Farbe(3,x,y):=2; (* gruенblau *)
70..79 : Farbe(3,x,y):=2; (* hellblau *)
      : 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 *)
(*241..256 : Farbe(1,x,y):=1; (* weiss *)
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 relocative Programme untergebracht. Das System-spezifische 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 relocieren 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 #flb,d1 * auszugebendes Zeichen (ESC) in d1
trap #2 * und BDOS aufrufen
move #lst,d0 * usw.
move #flb,d1
trap #2
move #lst,d0
move #flc,d1
trap #2
move #lst,d0
move #fl5,d1
trap #2
move #reboot,d0
trap #2
end

```

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

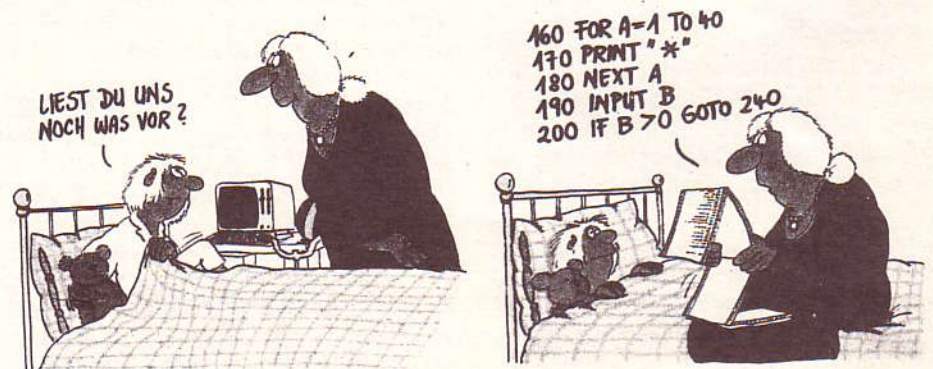
```

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

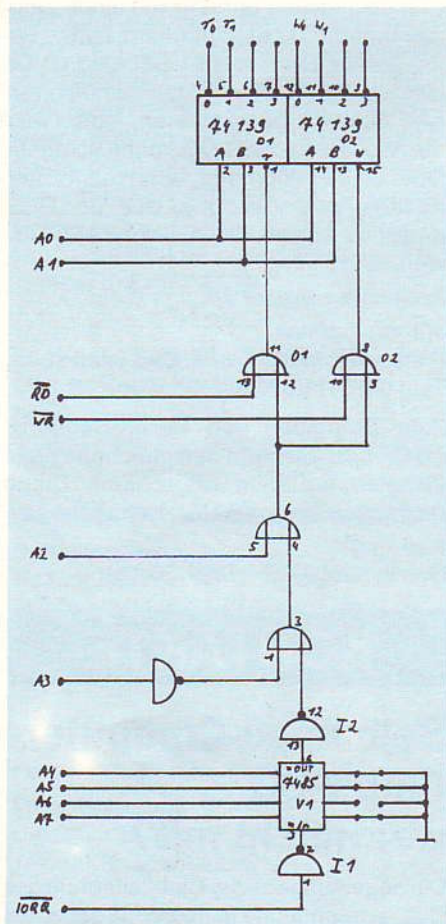
Für meinen Drucker verwende ich als Anschluß an den NKC eine normale IOE-Karte. Um Platz zu sparen, habe ich sogar das Lochrasterfeld abgesägt. Bis vor kurzem funktionierte dies auch alles sehr gut. Als ich aber den 68008 in Betrieb nahm, trat ein kleines Problem auf. Ich habe nämlich auch eine SOUND-Karte 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 inzwi-

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 Jumpfern), 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 10 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.



schen ein recht gutes Musikprogramm für den Z80 von einem Bekannten bekommen. Ich habe in dem Programm die Adresse von E0h,E1h auf 40h,41h geändert und das Programm dann gestartet.

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	20
00009040																3B
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	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	20	65	71	75	20	24
00009090												e	q	u		\$
000090A0	66	66	66	38	39	20	20	20	20	20	20	20	20	20	20	20
000090A0	f	f	f	8	9											
000090E0	20	3B	20	46	61	64	65	6E	6B	72	65	75	7A	20	78	2D
000090E0		;		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 anpassen, 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 Pro-

gramm 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:



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 umfas-

senden und als Werbemagazin funktionierendem Blättchen abverlangen kann. Die Zeitschrift hat sich als Alternative zu den Franzis Zeitschriften gemauert. 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 · 1 STOP-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 Commo-

dore-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,-.
Kay Baumgartel
Hegestieg 20 · 2000 Hamburg 20
Telefon (0 40) 48 19 16 abends

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 5 1/4"-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



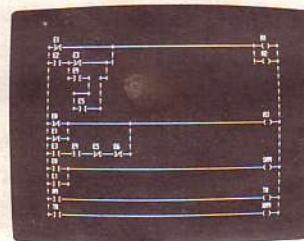
4 Kursteile (je ca. 70 Seiten im Format A4), DM 38,- je Kursteil
Der Kurs ist auf die HEXIO abgestimmt und ist für alle geeignet, die ihre ersten Schritte in Z 80-Maschinenprogrammierung machen.

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



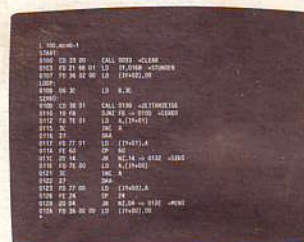
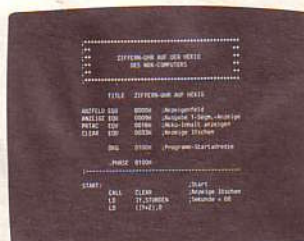
4 Kursteile (je ca. 70 Seiten im Format A4), DM 38,- je Kursteil
Dieser Kurs zeigt Ihnen, wie SPS programmiert wird, die Normung, die Anwendungsmöglichkeiten und die verschiedenen Darstellungsarten.

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



Das Betriebssystem beinhaltet 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 Programmredaktion 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
<input type="checkbox"/> Einführung mit dem NDR-Computer (4 Kursteile)	DM 38,-	DM 152,-
<input type="checkbox"/> Z 80-Assembler-Programmierung (4 Kursteile)	DM 38,-	DM 152,-
<input type="checkbox"/> SPS-Programmierung mit dem NDR-Computer (4 Kursteile)	DM 38,-	DM 152,-
<input type="checkbox"/> Kompakt-Kurs BASIC (angepasst an das RDK-BASIC)	DM 198,-	DM 198,-
<input type="checkbox"/> ZEAT-Betriebssystem (3 EPROMs mit Dokumentation)	DM 198,-	DM 198,-

Name, Vorname _____

Straße _____

PLZ, Ort _____

Datum _____ Unterschrift _____ 86189