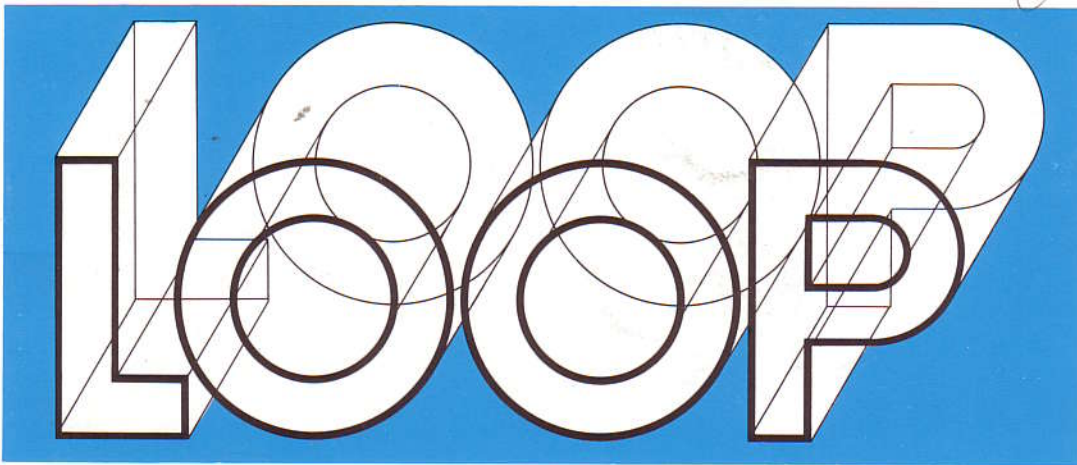


P. 5. SZ

(1)

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



13
a

3. JAHRGANG

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

DM 3,-

Neues JADOS-Update: Version 2.1

Schnelleres Arbeiten durch speicherresidente Programme

von Klaus Janßen

Das inzwischen recht weit verbreitete Betriebssystem JADOS wurde von mir kräftig überarbeitet und steht den Freunden des 680xx-Prozessors im NDR-Klein-Computer jetzt in der Version 2.1 zur Verfügung. Sehr geholfen haben mit dabei die zahlreichen Anregungen der Anwender des Systems.

Die wichtigsten Änderungen zu Version 2.0 sind:

- Der FCB wird nur noch nach Schreibzugriffen zurückgeschrieben. Das Problem mit dem Schreibschutz ist damit behoben (siehe LOOP 11).
- Durch eine geänderte Stackbehandlung arbeitet JADOS nun auch mit anderen Bootprogrammen zusammen; z. B. FLOBOOT von A. Granel.
- JADOS kann nun auch nach einem RESET oder einer Ausnahmebehandlung mit der Bibliotheksfunktion gestartet werden.
- Unterstützt werden jetzt bis zu vier Laufwerke im Standardformat NDR80.
- Mit dem *neuen Kommando INST* können nun bis zu 20 Programme resident in den Hauptspeicher geladen werden. Diese Programme werden im Kommandointerpreter durch Eingabe ihres Namens gestartet.
- Die Behandlung der formalen Parameter in Stapelverarbeitungsdateien wurde verbessert.
- Die Kommandoeingabe erlaubt nun auch das Löschen und Einfügen von Zeichen.
- Für den 68020-Computer mit Grundprogramm 5.0 ab Adresse \$0000 liegt die Ladeadresse jetzt auf \$14000.
- Für den Programmierer stehen 12 neue Unterprogramme zur Verfügung; u. a. schnelle Blocklese- und -schreibrouti-

BIBLIOTHEKSFUNKTION V 2.1 (C) 1986,1987 BY Klaus Janßen

---NAME---	---START---	---LAENGE---
INSPEC	03E820	001F92
DISASS	044020	002A86
BRDRUCK	0474DC	001378
JADOS	048020	0035F2

"+" --> nächster Eintrag ; "-" --> voriger Eintrag "E" --> Tabellenende
"A" --> Tabellenanfang ; "S" --> STARTEN ESC --> Abbrechen

Bild 1: Bildschirmlayout von ROMSTART an einem Beispiel

nen sowie Routinen zum Löschen und Einfügen in Textrings.

Gleichzeitig mit dem Betriebssystem sind von mir auch die Hilfsprogramme überarbeitet worden. Neu hinzugekommen sind ein Kopierprogramm für komplette Disketten - *DISKCOPY* -, mit dem man in weniger als 70 sec eine Diskette kopieren kann und ein komfortables Interface zum Grundprogramm - *GP* -. Da GP direkte Einsprungadressen in das Grundprogramm 4.3 benutzt, steht es den Anwendern zwecks Anpassung auch als *Quellprogramm* zur Verfügung. Das Programm *MORE* gibt jetzt auch deutsche Umlaute aus. *FORMAT* unterstützt bis zu vier Laufwerke im Format NDR80, erledigt den Prüflasevorgang in der halben Zeit und besitzt ein verbessertes Timing bei der Behandlung der Laufwerksmotoren.

Das Hilfsprogramm *ROMSTART* konnte bisher nicht alle Bibliothekseinträge fin-

Aus dem Inhalt:

Neues JADOS-Update: Version 2.1.	13a/1
NDR-Computer im Prüffeld . . .	13a/3
In eigener Sache	13a/3
Jetzt lieferbar.	13a/3
Für Einsteiger Z80, SBC2	13a/6
Z80-Vollausbau bis ZEAT	13a/6
Z80 - CP/M 2.2	13a/7
Für 68000-Einsteiger	13a/9
68000, YOGIDOS und JADOS, RL-Basic.	13a/15
CP/M68k, C und Modula	13a/19
Der mc-CP/M-Computer.	13a/24
Briefe, Kontakte und Kleinanzeigen	13a/24

den. Grafikprogramme hatten Schwierigkeiten mit der Seitenumschaltung. Das neue ROMSTART 2.1 sucht nun alle 32 Bytes nach dem Bibliothekseintrag und berücksichtigt die unterschiedlichen Adressierungskapazität der verschiedenen Mitglieder der 68000-Familie. Es werden nun bis zu 16 Bibliotheksprogramme gleichzeitig angezeigt. Insgesamt verwaltet ROMSTART bis zu 127 Programme. Bild 1 zeigt das neue Erscheinungsbild von ROMSTART.

In völlig neuem Gewand präsentiert sich jetzt auch BRDRUCK 2.1. Ein komfortables Menü erlaubt die Einstellung vieler für den Druckvorgang wichtiger Parameter. Eine Statuszeile gibt Auskunft über den Namen der zu druckenden Datei und die Anzahl der noch zu druckenden Zeichen. Bild 2 zeigt das Menü des Programms.

Das Programm benutzt die SteuerCodes für Epsondrucker. Wer einen Drucker mit anderen Steuersequenzen besitzt, kann das Programm entsprechend anpassen, da es im Quellcode ausgeliefert wird - Module DR-MAIN.ASM und DR-TOOL.ASM -.

Die wichtigste Neuerung im JADOS 2.1 betrifft das Speichermanagement. Die letzten 6 KByte sind für die JADOS-Variablen und Pufferspeicher reserviert. Davor liegt das JADOS-Programm mit 14 KByte. Ein eigener Stack von 2 KByte steht ausschließlich dem Kommandointerpreter zur Verfügung und liegt vor dem JADOS-Programm. Insgesamt „verschlingt“ das JADOS-System 22 KByte. Vor dem JADOS-Stack liegt der freie Benutzerspeicher. Dieser besteht aus drei Komponenten: einem freien Bereich ab der Ladeadresse, dem Benutzerstack und einem Bereich für die residenten Programme. Diese Aufteilung wird von JADOS automatisch verwaltet. Wenn ein Programm mit dem Kommando INST resident geladen wird, dann verschiebt sich der Benutzerstack um die Länge des Programms plus 1 KByte Sicherheitsreserve nach unten in Richtung Ladeadresse. Zu Fehlern mit dem Stack kann es dabei nicht kommen, da der Benutzerstack erst vor der Ausführung eines Programms zugeteilt wird. Es ist natürlich klar, daß der freie Benutzerspeicher mit jedem residenten Programm kleiner wird. Mit dem neuen Kommando TABLE kann man sich anschauen, welche Programme resident geladen wurden. Dabei werden jeweils der Programmname, Programmtyp und die Adresse ausgegeben, ab der das Programm installiert ist. Bild 3 zeigt ein Beispiel für die Ausgabe des Kommandos TABLE.

Die residenten Programme werden einfach durch Eingeben des Programmnamens gestartet. Der Ladeteil im JADOS schaut erst in einer Tabelle nach, ob das Programm installiert ist. Wenn nicht, dann sucht der Ladeteil auf der Diskette; wenn doch, dann hängt die weitere Aktion vom

Druckprogramm für Epson-Drucker V 2.1 (C) 1986,1987 by K. Janßen

>>>> V O R E I N S T E L L U N G E N <<<<<

F-ormularlänge	[65]	B-eginn Seitenzahl	[1]
L-inker Rand	[8]	P-osition Seitenzahl	[40]
K-opienzahl	[0]	C-ontrolzeichen	[#]
S-eitenauswahl	[1 - 999]		
1 = Schriftart	[Normal]	2 = Zeichensatz	[NDR]
3 = Zeilenabstand	[1/6"]	4 = Seitenzahlen	[AUS]
5 = Einzelblatt	[AUS]	6 = Seitenvorschub	[am Ende]

>>>> S T A T U S <<<<

Datei: 2:LOOP14.ART Größe: 10444 Kopien: 0

>>>> A K T I O N E N <<<<

A-bbrechen	N-ormieren des Druckers
D-rucken	U-rschub auf nächste Seite
T-extdatei laden	Z-eilenvorschub

Bild 2: Das Menü von BRDRUCK

VERZEICHNIS DER RESIDENTEN PROGRAMME

--NAME--	ADRESSE	TYP
VERS	\$049C00	68K
1COPY	\$048800	68K
BRDRUCK	\$047000	68K
DISASS	\$044000	68K
DISKCOPY	\$043000	68K
DSAVE	\$042800	68K
FORMAT	\$041800	68K
GP	\$040C00	COM
INSPEC	\$03E800	68K
MORE	\$03D000	68K
ROMSTART	\$03D000	68K
SYS	\$03C400	68K
HACO	\$037C00	COM

1>

JADOS-Statuszeile: 1C = Warmstart , 1A = Kaltstart , Freier Speicher: 220

Bild 3: Beispiel für das Kommando TABLE

Programmtyp ab. COM-Dateien sind so angelegt, daß sie im hinteren Teil des Speichers ablaufen können. Deshalb werden sie auch direkt auf der Adresse gestartet, ab der sie installiert sind. Dabei wird natürlich ein eventuell vorhandener Bibliotheksvorspann berücksichtigt. 68K-Dateien verwenden häufig Speicherplatz hinter dem eigentlichen Programm. Darum dürfen sie nicht auf der Installationsadresse ablaufen, da sie den Speicherbereich hinter sich „zerstören“ können. Aus diesen Gründen werden 68K-Dateien vor dem Start auf die Ladeadresse kopiert. Dieser Kopiervorgang arbeitet mit MOVE.L-Befehlen und verläuft sehr schnell; auf einem 68008 mit 8 MHz und 25 Waitzyklen werden 10 KByte in etwa 25 msec kopiert.

Mit den residenten Programmen ist es nun auch möglich, Treiberprogramme dauerhaft zu installieren, ohne daß man Speicherkonflikte befürchten muß. Eine sehr nützliche Anwendung wäre z. B. ein Hardcopy-Programm, das sich auf bekannte Art und Weise in die Tastaturroutine CI „einhängt“; siehe z.B. „HARD-COPY mit 68008/68000 von Elmer Schnuit“ in LOOP 8/9. Dieses Programm kann allerdings nicht unverändert übernommen werden. So muß der Bildschirmspeicher von 16 KByte als ds.b

\$4000-Anweisung reserviert werden. Man erhält dann ein Programm von etwa 18 KByte Länge. Nach dem Assemblieren muß das Programm - nennen wir es HACO.68K - in HACO.COM umbenannt werden. Es wird anschließend resident geladen mit „INST HACO.COM“ und gestartet mit „HACO“. Beim Start wird nur die Routine CI umgeleitet. Der Hardcopyvorgang wird durch CTRL @ ausgelöst.

In der Datei AUTOEXEC.BAT kann man festlegen, welche Aktionen JADOS nach dem Bootvorgang unternehmen soll. Eine sehr sinnvolle Anwendung für diesen Autostartmechanismus wäre z. B. das residente Installieren der 9 Hilfsprogramme von JADOS und z. B. dem Hardcopytreiber. Man kann dann die Systemdiskette aus dem Laufwerk entfernen und die gerade benötigte Arbeitsdiskette einlegen. Auf diese Weise kann man auch recht gut mit nur einem Laufwerk auskommen. Man sollte dann aber wenigstens über 256 KByte Hauptspeicher verfügen.

Bestell-Nr.	Best.-Bez.	inkl. MWSt. DM
10896	JADOS V2.1 UPDATE 8 1/4" 80 Spuren	50,-
10993	JADOS V2.1 UPDATE 3 1/2" 80 Spuren	50,-

NDR-Computer im Prüffeld

Der Computer wird bei meinem Arbeitgeber als Funktionsgerät eingesetzt. Es werden Phasenanschnittsteuergeräte für Motoren geprüft.

Die Hardware dazu ist: Eine Frequenz-



Bild 1. Hier sieht man das ganze Testgerät mit dem Drucker für die Typenschilder, auf dem mittleren Gehäuse befindet sich der Prüf- und Bremsmotor.

zählerkarte (analog aus MC8/85), AD-Wandler 10 Bit, AD-Wandler 8 x 16, 2 x 4 Digitaleingänge, IOE-Karte im Multiplexbetrieb für 64 Relais. Alle Meßeingänge sind potentialfrei.

Die Software: Das Programm ist in Assembler geschrieben. Graphische Darstellung des Prüflings, Anzeige der Trimpoties, welche gestellt werden. Die Spannungen können digital oder analog angezeigt werden. Es kann ein Prüfproto-



Bild 2. Bei diesem Bild sieht man das Testgerät mit dem Nadeladapter, im linken Gehäuse ist der Computer und die beiden Drives. In der Mitte sind die Schaltungen für die Schnittstellen und Stimulimodule. Im rechten Gehäuse sind die Netzteile.

koll ausgedruckt werden, Anzahl Fehler, geprüfte Stückzahl und Prüfzeit. Die Typenschilder können für Sonderfälle auf dem Bildschirm korrigiert werden. Das Programm für die Typenschilder ist in „C“ geschrieben.

Beiliegend sende ich Ihnen ein Listing von einem Programm. Damit können Zahlen binär, hexadezimal oder dezimal angezeigt werden und auch untereinander gerechnet werden.

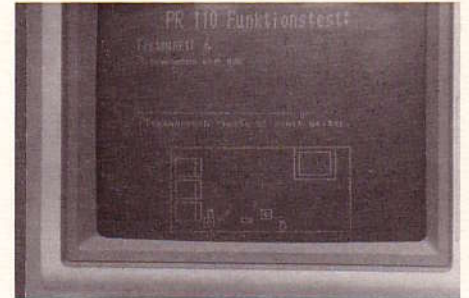


Bild 3. Auf dem Monitor sieht man das Bild für den Testpunkt 6. Unten ist der Prüfprint mit den Trimpoties, in der Mitte ist der Analogbalken mit dem das Poti abgeglichen werden kann.

In eigener Sache

Die CeBit 87 ist vorbei, und mit ihr ein großer Erfolg für den NDR-Computer.

Wir zeigten das einzige offene Computersystem auf der Messe, bei dem man begreifen kann, was ein Computer ist. Gerade diese Tatsache ist bei vielen Besuchern sehr gut angekommen – die steigende Nachfrage nach dem Einsteigerpaket beweist es.

Unsere IBM-Kompatibilität ist gut angekommen. Wir konnten aufgrund der ebenfalls hohen Nachfrage für die AT-CPU-Karte einen wesentlich günsti-

geren Preis machen; damit ist der so aufgebaute kompatible nicht wesentlich teurer als ein entsprechendes importiertes System.

Die nun eingesetzte AT-CPU-Karte kann 1MByte Speicher auf der Baugruppe direkt unterbringen (SIP-Module) und läuft mit 10 oder 12 (!) MHz. Die Preise sind stark nach unten in Bewegung.

Mit Freude stellten wir fest, daß die Qualität der uns angebotenen Software weiter zunimmt. Während der CeBit konnten wir einen ganz hervorragenden Text-Editor

für die 68xxx-Linie testen, der die GDP voll ausnützt – inverse Darstellung und Scroll-Geschwindigkeit sind hier kein Thema mehr. Der Text-Editor wird bald von uns angeboten werden.

Wir freuen uns ebenso über die steigende Qualität der uns zugesandten LOOP-Artikel. Unser Ziel ist es, vielleicht ab nächstem Jahr die LOOP monatlich erscheinen zu lassen. Ebenso werden wir in Zukunft Fremdanzeigen anderer Hersteller und Lieferanten aufnehmen, um Ihnen noch eine breitere Information, verbunden mit einer dickeren LOOP zukommen zu lassen.

„Und das Beste: Ich kann's alleine machen“

Geniale Neuerungen helfen Behinderten, sich selbst zu helfen!

Einen interessanten Artikel über Computer und eine überraschende Sparte seines Einsatzes fanden wir im *TIME* Nr. 38 vom 22. September 1986; hier Auszüge: Mit 19 war D. Yong nach einem Autounfall und einem Genickbruch gelähmt. Er lernte danach mit einem Mundstab eine Schreibmaschine zu bedienen. Mit 27 und als Student bekam er einen IBM PC; er schreibt heute mit Hilfe des Compu-

ters als Biologe mehr, als er es je mit der Hilfe von Armen und Beinen getan hatte. 13 Millionen Behinderte gibt es allein in Nord-Amerika. Bisher versuchte man die Behinderten mit Hilfe der Technik wieder zu mobilisieren; jetzt ändert man die Technik, um sie behindertengerecht zu machen. Alan Brightman von Apple Computers sagt: Schlüsselworte sind: Zugang, Unabhängigkeit und Erlangung neuer Möglichkeiten. Wenn Sie nur Ihre Augenbraue bewegen können, habe ich einen Schalter, um Sie zu befähigen, Daten in einen Computer zu füttern. Und Zugang zum Computer = Zugang zur Welt! Beispiele: W. Garmen (51), der unfähig ist zu sprechen und zu schreiben, kann mit einem Infrarotsensor den Cursor eines

Computers durch die Bewegung eines Augenlides dirigieren und so mit der Umwelt kommunizieren.

G. Griffith (54) die, obwohl blind, als Musiklehrerin graduierte und arbeitete, dann zudem noch ertaubte, sagt von sich: „Klar, ich bin blind und taub, aber ich bin nicht behindert.“ Sie liest Musik in Braille-Schrift, die mit Computerhilfe erzeugt wird.

Ein System zur Verfolgung der Augenbewegungen, das ursprünglich für Jetpiloten entwickelt wurde, hat T. Hutchison für Behinderte weiterentwickelt. Er meint: Mit diesem System befreien wir Seelen, die in funktionsfähigen Körpern stecken!

Einige Dinge müssen für Behinderte noch verbessert werden: Es darf z. B. keine Knöpfe geben, die gleichzeitig mit anderen zu bedienen sind. Auch die für

Normalbenutzer so praktikable „Maus“ nützt Behinderten gar nichts. „Eines jedenfalls ist sicher“, meint K. Dahlke (blinder Software-Ingenieur bei ITT): „eine

Tastatur muß man bedienen können.“ Und in dieser Hinsicht müssen Behinderte und Nichtbehinderte von der gleichen Basis aus starten!

Jetzt lieferbar - Jetzt lieferbar

Roboter steuern mit Einsteigerpaketen (Z 80)

von Ulrich Kracker

Gleich drei neue EPROMs mit Robotersteuerprogrammen sind ab sofort von GES lieferbar. Sie ermöglichen den Teach-In-Betrieb mit den zwei unterschiedlichen FISCHERTECHNIK-Robotern.

Alle drei unterstützen die neuerschienene Baugruppe „ROB2“, die als wesentlichste Neuerung je vier Leistungstransistorendstufen in Brückenschaltung besitzt (kontaktloses Schalten von Links- oder Rechtslauf).

So ist auch im ersten der neu eingeführten EPROMs, dem „E2ROB2“, eine Drehzahlverminderung der Roboter motoren durch schnelles Takten der Versorgungsspannung implementiert worden; sie tritt ein, wenn sich der Arm des Zwei-Achs-Teachroboters seiner Sollposition annähert. Das Listing des Programmes ist im Handbuch der ROB2-Baugruppe zu finden.

Weitere Merkmale des EPROMs „E2ROB2“:

- In jedem Falle erforderlich ist die ROB2-Baugruppe
- Verwendbare CPU-Konfigurationen:
 - * SBC3 (z. B.: in den Einsteigerpaketen HEXIO oder HEXIO2)
 - * CPU Z80 mit ROA 64 oder mit Bank-Boot-Baugruppe
- Speicherbausteintyp: 2764 (8 kB-EPROM)
- Steckplatz/Startadresse: 2000 sedezimal
- Speicherplatz (ROM): 582 Byte
- Speicherplatz (RAM): 2 kB
- Robotermodell: FISCHERTECHNIK 2-Achs-Teachroboter

Die anderen beiden EPROMs unterscheiden sich voneinander im Grunde genommen nur in der Anpassung an die SBCs. Das „ETROB“ ist für den Einsatz auf der SBC2 und das „ETROB2“ für den Einsatz auf der SBC3 vorgesehen. Im Inhalt und in der Funktionsweise sind beide Versionen jedoch völlig identisch. Bei der (mitgelieferten) Beschreibung konnten daher zwei Fliegen mit einer Klappe erledigt werden. Diese beiden Programme sind speziell auf den Einsatz mit dem HEXMON-Grundprogramm ausgelegt und auch nur in Verbindung mit diesem lauffähig, denn es werden etliche Unter-

routinen des HEXMON angesprochen (Zeichenausgabe und Tastaturabfrage). Bei dem damit zu steuernden dreiachsigen Trainingsroboter befinden sich keine Schalter mehr auf der Robotergrundplatte, so dienen hierbei die Tasten des Einsteigerpaketes der Befehlsübermittlung an das steuernde Teach-In-Programm.

Weitere wichtige Merkmale der Programme „ETROB“ und „ETROB2“:

- Benötigte Konfiguration:
 - * HEXIO-Einsteigerpaket mit SBC2 für das EPROM „TROB“
 - * SBC3 für das EPROM „TROB2“
 - * ROB2-Baugruppe
- Speicherbausteintypen
 - * TROB: 2732 (4 kB)
 - * TROB2: 2764 (8 kB)
- Steckplatz/Startadresse
 - * TROB: 1000 sedez. (SBC2)
 - * TROB2: 2000 sedez. (SBC3)
- Speicherbedarf (ROM): 1986 Byte
- Anzahl der abspeicherbaren Positionen: je 65 in 5 unabhängigen Ebenen
- Speicherbedarf (RAM): 2,8 kB
- Robotermodell: 3-Achs-Trainingsroboter

Bestell-Nr.		DM
10877	E2ROB2	40,00
10923	ETROB	40,00
10924	ETROB2	40,00

Neue Uhr für den NDR-Computer verfügbar Ulrich Kracker

Die neue Hardwareuhr ist keine Baugruppe im herkömmlichen Sinn, sondern vielmehr ein einziger Baustein, der lediglich einem statischen 8k-RAM-Baustein 6264 „untergeschoben“ wird. Wegen dieser eleganten Methode (der „höhergesetzte“ Speicherbaustein bleibt natürlich voll in Funktion), hat die neue NDR-Uhr auch den Namen „SMART-WATCH“ erhalten. Der offizielle Name ist jedoch: DS1216 von Dallas Semiconductor. Die SMART-WATCH wird auf einen ROA64-Steckplatz 2000 sedezimal gesteckt und dort von 68008er Modula-2 sowie von Z80-CP/M-Demoprogrammen angesprochen. Die CP/M-Demos befinden sich auf den neu ausgelieferten CP/M TOOL-Disks.

Weitere interessante Merkmale der SMART-WATCH:

- Integrierte Lithiumbatterie puffert Uhr und CMOS-RAM 6264
- Uhrfunktion unabhängig von der RAM-Funktion
- Aus Registern auslesbar: von hundertstel Sekunde,.....Jahre,Datum
- Ganggenauigkeit besser als +/- 1 min/Monat
- Erkennung von Schaltjahren

Bestell-Nr.		DM
10702	SMART-WATCH	159,00

Neue Produkte - jetzt lieferbar

EMALE V1.1. EPROM-Software zum „Malen“ mit der COL256. Benötigte Hardware zum Betrieb von EMALE ist ein 68008-System mit dem RDK-Grundprogramm.

Vorsicht: EMALE nicht mit CAD-Programm verwechseln. Mit EMALE können Bilder mit Hilfe einiger Funktionen erstellt werden, aber rechnerisch wird das Zeichnen vom Computer nicht unterstützt.

Das Hauptprogramm und die COL256-Treiber sind in Assembler geschrieben. Benötigte Hardware zum EMALE-Betrieb ist ein 68008-Rechner mit dem RDK-Grundprogramm und eine COL256. Die Eingabe erfolgt über ATARI-Maus, Micro-soft-Maus, Preh-Tablett oder Joystick.

Bestell-Nr.	Best.-Bez.	incl. MWSt. DM
10708	EMALE 68008	59,00

MALE-Disk V1.6. Wie das EMALE, dient die MALE-Disk zum Erstellen von farbigen Bildern. Jedoch ist es hiermit möglich, die gezeichneten Bilder auf Diskette zu speichern und zu laden.

Achtung!! Auch die MALE-Disk ist nicht mit einem CAD-Programm zu verwechseln.

Am unteren Bildschirmrand erscheint das bereits von Messen bekannte MALE-Grundmenü (Text, Raster, Spray, Linie usw.). Hierzu kann man mit dem Befehl „EXTRA“ ein Zusatz-Menü über die GDP64K zuschalten. Dieses Menü erscheint dann auf dem im System befindlichen Monochrom-Monitor. Hiermit hat man die Möglichkeit, das Programm zu beenden, umzubenennen, ein Bild zu laden oder zu speichern.

Das Hauptprogramm der MALE-Disk ist in C und die COL256-Unterprogramme in Assembler geschrieben. Da die MALE-Disk mit allen Prozessoren unserer 68XXX-Familie zu verwenden ist, sind auf der Diskette drei Dateien vorhanden. Diese Dateien sind nötig, um jeden einzelnen Prozessor (68000, 68008, 68020) anzusprechen zu können.

Die Eingabe erfolgt entweder über Microsoft-Maus, Atari-Maus und Apple-Maus, Joystick oder über Pit-Pat/Graphiktablett.

Bestell-Nr.	Best.-Bez.	incl. MWSt. DM
10710	MALE 58 5/4", 80 Tr.	59,00
10711	MALE 38 3/2", 80 Tr.	59,00

68K-Spiele. Nun auch Spiele für den NDR-Computer. Auf der 68K-Spielediskette sind zwei Spiele enthalten: Dame und Reversi.

Als Hardware benötigt man eine 680XX-CPU, GDP64K, 80-Zeichen-Monitor, FL02 und ein Disketten-Laufwerk. Dazu noch das Grundprogramm Version 4.3 und als Betriebssystem JADOS V2.0.

Wenn der Rechner gestartet ist, können die Spiele ganz einfach entweder mit DAME oder REVERSI gestartet werden.

Zu dem Spiel Dame ist weiter nichts zu sagen, außer daß es nach den üblichen Regeln abläuft. Aber Reversi, was ist das eigentlich? Reversi oder auch Othello genannt, ist ein ähnliches Spiel wie Tick-Tack-Tow, mit dem einen Unterschied, daß man hier gewinnen kann. Gespielt wird auf 8 x 8, insgesamt 64 Feldern.

Jeder Spieler setzt abwechselnd eine Spielmarke (0 bzw. *) auf das Spielfeld und zwar so, daß mindestens eine „Marke“ des Gegners eingeschlossen wird. Dies kann waagrecht, senkrecht oder diagonal geschehen. Die nun eingeschlossenen Marken des Gegners werden in eigene Spielmarken umgewandelt (daher Reversi) und bleiben auf dem Spielfeld.

Im allgemeinen gibt es zwei Varianten für

Spieltiefen und zwei Graphikdarstellungen.

Bestell-Nr.	Best.-Bez.	incl. MWSt. DM
10894	68K Spiele 38 3/2", 80 Tracks	39,00
10895	68K Spiele 58 5/2", 80 Tracks	39,00

RDVIDEO. Etwas für Leute, die außer ihrem Computer auch noch für andere „Hobbies“ Zeit haben.

RDVIDEO – ein Programm, welches es ermöglicht, 350 Video-Kassetten zu verwalten.

Nach Eingabe der Dateien können Titel, Freiräume und/oder Filmlängen gesucht werden. Eine Liste der Video-Kassetten-Sammlung kann auf einen EPSON (oder kompatiblen) Drucker ausgegeben werden.

Zum Betrieb des Programmes ist ein JADOS-Betriebssystem ab Version 2.0 nötig. RDVIDEO ist relokativ und kann daher von JADOS an jeder beliebigen Stelle im Speicher abgelegt werden. Das Programm selbst benötigt einen Speicher von 15KB. Es ist auf allen Prozessoren der 68000er-Serie lauffähig.

Bestell-Nr.	Best.-Bez.	incl. MWSt. DM
10853	RDVIDEO 3/2"	49,00
10852	RDVIDEO 5/4"	49,00



EPSON LX-800:

Professionelle Druckertechnik für DM 798,00

Mit in dieser Preisklasse unerreichten 180 Zeichen pro Sekunde und einem zweiten Schönschrift-Zeichensatz im ROM, bietet der neue LX-800 auch dem semiprofessionellen Anwender attraktive Merkmale. Der Bedienungskomfort wurde auf das Niveau der Mittelklassendrucker angehoben. Die Schriftarten Pica und Sans-Serif können in allen ihren Variationen an der Tastatur des LX-800 eingestellt und in NLQ ausgedruckt werden. Das gilt sogar für die hoch- und tiefgestellten Kleinschriften Subscript und Superscript. Ein Traktor ist serienmäßig, ebenso der halbautomatische Blatteinzug. Eine vollautomatische Einzelblattzuführung ist als Zubehör erhältlich.

Bestell-Nr.	Best.-Bez.	incl. MWSt. DM
30269	EPSON LX-800798,-	

Für Einsteiger Z80 SBC2

Das unbekannte Unterprogramm – der Programmzerstörer

von Günter Renner
7206 Emmingen-Liptingen

Arbeiten Sie mit dem Z80, EGRUND und EZASS? War ein schwarzer Bildschirm schon einmal das Resultat eines beherzten Programmstarts? Wenn sich dann im gesamten RAM eine Abfolge von zwei Bytes befand – 80h und 0xh – dann sind Sie einem Unterprogramm zum Opfer gefallen, das nicht dokumentiert ist.

Für Zwecke der Interrupt-Verarbeitung enthält das Grundprogramm bei 0038h einen Sprungbefehl nach 8003h. Dort kann wiederum ein Sprungbefehl eingetragen werden – frei vom Anwender, was ja im EPROM-Bereich nicht möglich ist –, der zur eigentlichen Interrupt-Routine führt.

Der Sprungbefehl in 0038h wird jedoch nicht nur dann aktiv, wenn ein Interrupt auftritt, sondern auch dann, wenn ein Restart-7-Befehl abgearbeitet wird. Dieser entspricht dem Wert 0ffh.

Hat man in einem Programm irgendwo irrtümlich einen solchen Befehl stehen, so erfolgt ein Aufruf ab der Adresse 0038h als Unterprogrammaufruf. Es erfolgt ein Sprung nach 8003h. Die

unfreiwillige Unterprogrammroutine wird dort mit den „Befehlen“ fortgesetzt, die dort zufällig stehen. Hat man das Pech, daß dann erneut 0ffh auftritt, so erfolgt ein erneuter Unterprogrammaufruf bei 0038h, und diese Schleife setzt sich unendlich fort.

Die leidige Begleiterscheinung ist nun, daß die Adresse, bei der der erneute UP-Aufruf jeweils erfolgt, auf dem Stack abgelegt wird. Letzterer ist bekanntlich ein dynamisches Gebilde, das die potentielle Fähigkeit hat, sich über den gesamten Speicherbereich auszubreiten, wenn man nur oft genug von ihm Gebrauch macht.

Die beschriebene Kette von Unterprogrammaufrufen tut nun genau das. Bei jedem UP-Aufruf wird der Stackpointer dekrementiert und der nächste Speicherplatz mit der Adresse belegt, bei der er erfolgt. So kommt es, daß der ganze RAM voll ist mit einer Adresse, die bei 8003h oder wenig darüber liegt.

Nebenbei bemerkt ist dieser Programmablauf etwas ziemlich Raffiniertes – nämlich ein rekursives Programm.

Franzis'
Eine Information der Buchabteilung

Rechner modular

Rolf-Dieter Klein

Der NDR-Klein-Computer – selbstgebaut und programmiert. Ca. 432 Seiten, ca. 325 Abbildungen und 21 Tabellen. Ca. DM 68,00. ISBN 3-7723-8721-7

Erscheinungstermin: März/April 1987.

Am 14. 1. 1987 ist im Bayerischen Fernsehen die gleichnamige Fernsehsendung angelaufen. Es handelt sich um 20 Folgen, jeweils am Mittwoch um 17.30 Uhr. Etwa ab Mitte März ist die Sendung so weit fortgeschritten, daß das Buch zum Einsatz kommt.

Der Titel ist aber auch ohne die Fernsehsendung ein eigenständiges Werk.

Wie ist Abhilfe zu schaffen? Man muß bei 8003h irgendetwas hineinschreiben, daß nichts passiert. Das könnte z. B. sein:

```
halt
jp 0000
rst 0
```

Wäre das eine Anregung für die nächste Revision des Grundprogramms? Besonders Leute, die viel mit EPROMs arbeiten, sind mit dieser Erscheinung geplagt, denn jene enthalten im unprogrammierten Zustand nichts anderes als 0ffh!

Z 80-Vollausbau bis ZEAT

Vorsicht Falle – Stack und System-RAM-Bereiche des EZASSØ

Günter Renner, Schloßbühlstraße 11
7206 Emmingen-Liptingen

Wer schon größere Programme mit dem EZASSØ geschrieben hat, hat vielleicht schon Fehler gefunden, für die keine Erklärung zu existieren scheint. Kritisch sind Programme, die ab Adresse 8800h den Bereich vor 88ffh und solche, die ab Adresse 9000h die Adresse 9f00h erreichen. Der Grund liegt darin, daß der Bereich von 9f00h bis 9fffh von EZASSØ als Systemram benutzt und der Bereich ab 8ffffh abwärts als Stack verwendet wird. Es kommt zu Kollisionen, wenn der Assembler Maschinencode in diesen Bereichen ablegt. In einigen Unterlagen zu EZASSØ ist irrtümlich nur angegeben, daß der Bereich 8f00h bis 8ffffh nicht benutzt werden darf.

Der Assembler hat zwei getrennte Stacks (8ffffh und 9ffffh), was beim Testen von Programmen ebenfalls zu Schwierigkeiten führen kann, wenn man z. B. Programme abschnittsweise testet, indem man Rücksprünge auf die Adresse 6000h einbaut. Solche Programme, die push und pop verwenden, können abstürzen, weil der Assembler den Stackpointer verändert.

Auch das Grundprogramm hat einen eigenen Stack-Bereich. Es ist deshalb ratsam, für Rücksprünge genauso wie für das Starten von Programmen zu Testzwecken nur das Grundprogramm, nicht aber den Assembler/Debugger zu verwenden. Man kann sich nur dann darauf verlassen, wenn man mit ein und demselben Stack arbeitet.

Apropos große Programme: man kommt nicht umhin, Programme zu verschieben, wenn sie die oben angegebenen Größen überschreiten. Dazu benötigt man geeignete kleine Routinen. Eine solche zeigt Bild 1. Sie hat den Vorteil, daß man die Länge nicht angeben muß, wie dies z. B. bei dem Idir-Befehl der Fall ist. Es wird auf eine Routine im Z80-Grundprogramm zurückgegriffen, die die Länge von Befehlen berechnet. Wer die Version ab Adresse 2000h verwendet, muß zum Wert des Systems „befl“ 2000h addieren.

Die Startadresse der Quelle steht in hl, die des Ziels in de. Die Routine überträgt solange, bis der Befehl „ret“ auftritt. Danach wird (c) dekrementiert. Das Ganze wird wiederholt, bis (c) auf Null steht. In c

wird die Anzahl der zu übertragenden Programme, die als Unterprogramme geschrieben sein müssen, eingetragen. So entfällt das lästige Eingeben der Programmlänge; Quelle und Ziel können in bekannter Weise symbolisch eingegeben werden.

Zunächst läuft das Programm in einer Schleife, deren Zähler (b) ist. In dieser Schleife werden alle Bytes übertragen, die zu einem kompletten Befehl gehören. Dann wird geprüft, ob der nächste Befehl „ret“ ist. Wenn nicht, wird die Länge des neuen Befehls berechnet und neu in die

erste Schleife eingetreten. Handelt es sich um „ret“, so wird (c) dekrementiert und das Programm nur dann fortgesetzt, solange der Wert Null noch nicht erreicht ist. Zuletzt muß nur noch der letzte Befehl übertragen werden, der nur aus einem einzigen Byte bestehen kann.

```
Bild 1
Transportroutine für Unterprogramme
befl:= 0D2Ah
start: push de
      call befl
      pop de
```

```
st1: ld a,(hl)
     ld (de),a
     inc hl
     inc de
     dec b
     jr nz,st1
     ld a,(hl)
     cp 0c9h
     jr nz,start
     dec c
     jr nz,start
     ld a,(hl)
     ld (de),a
     ret
```

Z80-CP/M2.2

CP/M 2.2-Diskettenkapazität besser nützen

von Christoph Köhler

Die hohe Speicherfähigkeit der von Ihnen verwendeten Disketten mit 80 Spuren, beidseitig verwendbar und doppelter Dichte von ca. 778 kByte, müssen Sie mit einem relativ hohen Preis bezahlen.

Der Übersicht zuliebe wird natürlich für jedes Programm eine eigene Diskette angelegt, was aber auch dann oft den Nutzungsgrad auf ein Zehntel reduziert.

Vom Betriebssystem CP/M 2.2 wird ein wenigverwendeter Befehl Namens USER gestellt, der aber in den meisten Fachbüchern nur kärglich behandelt wird.

Durch diesen Befehl kann die Diskette in maximal 16 verschiedene Directory's unterteilt werden (USER 0-15).

Der Befehl selbst ist resident und somit immer zur Verfügung. Nach einem Kaltstart ist immer USER 0 eingestellt. Wenn dann nur die Benutzerbereiche 1-15 verwendet sind, wird dem Anwender nach dem Hochbooten der Diskette auf die Eingabe DIR nur ein gemogeltes NO FILE eingeblendet.

Somit können Sie Ihre Dateien zumindest vor Nichtprofis verstecken.

Das Beschreiben der jeweiligen Benutzerbereiche ist etwas aufwendiger als die Anwendung selbst:

Das einzige Programm zum Verändern der Benutzerbereiche ist das CP/M 2.2 File PIP.COM. Da das PIP-Programm Files nur aus anderen USER-Bereichen lesen (nicht schreiben) kann, ist man gezwungen, in dem gewünschten USER-Bereich erst mal das PIP.COM File zu errichten.

Von dort aus kann man beliebig entweder von einem anderen Benutzerbereich oder von einer anderen Diskette Programme einkopieren.

Der USER-Bereich ist schnell eingerichtet:

1. PIP-Programm in z.B. USER 5 kopieren

CP/M 2.2 Diskette in Laufwerk A einlegen.

```
A)PIP          PIP aufrufen
* CTRL C      wieder aussteigen
A)USER 5      USER 5 anwählen
A)SAVE 8 PIP.COM Das noch in der TPA
                befindliche PIP-Program
                m in USER 5
                abspeichern
```

```
A)DIR
A: PIP      COM      Directory U 5
A)          anschauen
```

2. Neue Programme in USER 5 kopieren

Beispiel 1:
STAT.COM von Laufwerk A: USER 0 nach USER 5 kopieren.

```
A)USER 5
A)PIP A:= A:STAT.COMÄGOÜ
(Beim PIP wird erst Ziel, dann Quelle
genannt. Die Quelle ist mit einem GO
(USER 0) in eine rechteckige Klammer
angegeben. Klammer auf = Ä, Klammer
zu = ü).
```

Beispiel 2:
TURBO.COM von Laufwerk B: in Laufwerk A: USER 5 kopieren (in Laufwerk B: muß TURBO.COM sein).

```
A)USER 5
A)PIP A:= B:TURBO.COM
```

Da die Befehle wie DIR, REN oder ERA nur jeweils für den angegebenen Benutzerbereich gelten ist ein komfortables Arbeiten gewährleistet.

Man kann sich jetzt die Diskette entweder projektbezogen (z.B. eigene verschiede-

ne Basic-Anwendungen in verschiedene USER-Bereiche) oder nach Programm sortiert (z.B. ZEAT, TURBO, dBASEII, TOOLS . . . in die verschiedenen USER-Bereiche) auf der Diskette anlegen.

Natürlich kann man die Benutzerbereiche auch wörtlich nehmen und das gleiche Programm für Paul, Werner und Berta . . . einrichten.

Das Umschalten zwischen USER-Bereichen geschieht einfach durch die Angabe USER 0-15.

Ein Insider kann sich mit dem STAT-Befehl einen schnellen Überblick über alle verwendeten Benutzerbereiche verschaffen:

```
A)STAT USER:
Active User: 0
Active Files: 0 5
```

Mit Sicherheit aber brauchen Sie dadurch in nächster Zeit weniger Geld für Ihre Disketten auszugeben, ohne die notwendige Übersicht zu verlieren.

Messen von analogen Spannungen . . . mit der AD8/16 Karte unter Turbo Pacal

von R. Pawlowitz

1. Die Maschinenebene

Hardwarenahe und praxisorientierte Programmierung stellen den eigentlichen Anwendungsbereich eines Computers dar. Die Programmiersprache Turbo Pas-

cal eignet sich wegen ihrer hervorragenden Struktur besonders für diesen Zweck. Turbo Pascal vereint Geschwindigkeit und direkten Zugriff mit Komfortabilität. Um noch mehr Geschwindigkeit und Flexibilität in der Programmierung zu erreichen bietet T. Pascal einen schnellen und trickreichen Befehl.

2. Der INLINE-Befehl

Die INLINE-Anweisung bietet die Möglichkeit, auf einfache Art Assembleranweisung in den Programmablauf von Pascal miteinzubinden. Eine INLINE-Anweisung setzt sich wie folgt zusammen:

– Aus dem reservierten Wort **INLINE**.
– Aus einem oder mehreren Codeelementen (vom Assembler erzeugter Hexcode) die durch Schrägstriche voneinander getrennt sind.

INLINE darf als Unterprogramm deklariert werden, kann aber auch Bestandteil des

```
(* Abb 1.1 *)
(* Programm zur Ansteuerung der AD 8/16 Baugruppe *)
(* Vor Inbetriebnahme des Programms aktuellen Messkanal an Spannung *)
(* zwischen 0-5 Volt anschliessen *)
(* Nicht beschaltete Messkanäle müssen auf Masse gelegt werden *)

PROGRAM AD;
VAR
  X,Messwert,T,TTA :Integer;

PROCEDURE MASKE; (* Zeichnen der Anzeigenskala *)
BEGIN
  WriteLn('M 119 165,B . . . . .');
  WriteLn('M 119 155,B 0 . . . . . 5');
  WriteLn('M 10 10,B Eine Taste druecken...');
  WriteLn('M 120 150,R 274 81,R 274 86');
END;

PROCEDURE WANDELN; (* Routine AD 8/16 *)
BEGIN
  Inline ($D3/$E0/ (* OUT A,E0H *)
         $DB/$E0/ (* IN A,E0H *)
         $CB/$7F/ (* BIT 7,A *)
         $28/$FA/ (* JR,Z *)
         $DB/$E1/ (* IN A,EH *)
         $32/Messwert);
END;

PROCEDURE GRAFIK; (* Umschalten in Grafikmodus *)
BEGIN
  WriteLn(chr(27),chr(27),'G','Y0');
END;

PROCEDURE ZEIGER; (* Zeichnen des beweglichen Zeigers *)
BEGIN
```

```
WriteLn(chr(27),chr(27),'G','Y0');
WriteLn('M',TTA,' ',T,'0');
WriteLn('D 260 230');
END;

PROCEDURE Messwertfassung; (* Bringt Messwert auf Bildschirm *)
BEGIN
  REPEAT
  BEGIN
    Wandeln;
    X:=Messwert-10620; (* Messwert an Zeigerkoordinaten anpassen *)
    TTA:=X;
    T:=17;
    WHILE TTA=X DO
    BEGIN
      Wandeln;
      X:=Messwert-10620;
      Grafik;
      Zeiger;
      IF Keypressed THEN EXIT; (* Wenn Taste gedrueckt Programmende *)
    END;
    WriteLn('G 0 1'); (* GDP auf loeschen schalten *)
    Zeiger;
    Write('A');
  END;
  UNTIL Keypressed;
  WriteLn('A');
END;

BEGIN (* Hauptprogramm *)
  Clrscr;
  Grafik;
  Maske;
  Messwertfassung;
  WriteLn('A');
  Clrscr;
END. (* In Alphamodus zurueck *)
```

Hauptprogrammes sein. Für den NDR-Computer gibt es Turbo Pascal unter CP/M 2.2. Wir müssen also unsere **INLINE**-Anweisung mit dem Befehlssatz des Z80 schreiben. Hat man nun den Wunsch, eine **INLINE**-Anweisung mit einzubinden, so übersetzt man sein Assemblerprogramm mit Hilfe eines Assemblers (ZEAT) oder von Hand mit einer Befehlsliste und trägt den erzeugten Hexcode in die **INLINE**-Routine ein. Ein Beispiel:

Der Assemblerbefehl **IN A,(E0)** hat die Bedeutung, Lade den Akku aus dem peripheren Port **E0H**. Dieser Assemblerbefehl besitzt den Hexcode (Objektcode) **DB E0**. Eingesetzt in die **INLINE**-Anweisung besitzt das ganze nun folgendes Format.

```
INLINE ($DB/$E0);
```

3. Die Hardware

Der NDR-Computer hat auf der Hardwareseite einiges zu bieten. Die AD8/16 Karte in Verbindung mit Turbo Pascal

bringt neue Anwendungsbereiche hervor. Sie kann gleichzeitig an 16 Eingangskanälen Spannungen in der Größe von 0 – 5 V mit einer Auflösung von 8 Bit verarbeiten.

Die Abbildung 1.1 zeigt eine Routine unter Pascal, wo die **INLINE**-Anweisung eingesetzt wird. Mit Hilfe der kleinen Procedure „**WANDELN**“ ist es möglich, die AD-Karte softwaremäßig anzusprechen.

Der Befehl (**OUT A,E0**) bestimmt:

- die Portadresse **E0H**
- und den Messkanal **01 – 0F**.

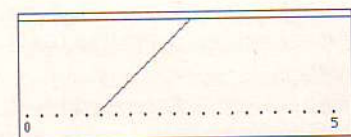
Der darauffolgende Programmablauf steuert den AD-Wandler jetzt mit den von uns voreingestellten Werten an und übergibt uns am Ende der **INLINE**-Routine den am Meßeingang anliegenden analogen Wert. Unsere Aufgabe ist es nun lediglich, den übergebenen Wert, der nun in Form einer Dezimalzahl bereitliegt, an unser weiteres Pascalprogramm anzupassen. Der entscheidende Vorteil dieser Prozedur ist die reibungslose

Variablenübergabe aus der **INLINE**-Procedure heraus zurück zum Pascalprogramm. So ist es spielend möglich, analoge Spannungen an den Eingängen der AD Karte zu messen, und sie innerhalb von Pascal weiter zu verarbeiten.

Anwendungsbeispiele sind:

- Aufnahme von elektrischen Meßwerten an mehreren Meßkanälen
- Windstärke und Temperaturmessung an einer kleinen Wetterstation.

Hier sind der Phantasie keine Grenzen gesetzt. Die Abbildung 1.1 zeigt beispielsweise, wie man den nun bereitliegenden Dezimalwert in Verbindung mit der Flomongrafik innerhalb von Pascal einsetzen kann.



Eine Taste druecken...

Gebrauchsgrafik für den NDR-Computer in BASIC / FLOMON

von Carsten Denneburg, 4460 Nordhorn

Hardware: Grundausbau mit SBC II oder SBC III Ausbau mit CPU Z80, ROA 64 CP/M Ausbau mit CPU Z80, ROA 64 oder DYN-RAM
möglich: Hardcopy

Software: RDK-BASIC, HEBAS

Bereits in vorhergehenden Heften der **LOOP** habe ich versucht, gerade auch für den Anfänger, die Sprache BASIC im Zusammenhang mit dem NDR-Computer

und seinen wirklich guten Grafikfähigkeiten vorzustellen.

Das nachfolgend abgedruckte BASIC-Programm ermöglicht die Erstellung einer „universellen“ Balkengrafik, die mit wenigen Befehlsänderungen auch in eine Kurvengrafik und mit nur wenigen (!) Änderungen an sehr viele Bedürfnisse angepaßt werden kann.

Das Programm ist Bestandteil eines

umfangreicheren Grafikprogramms. Ich habe versucht, mit möglichst einfachen „Standardbefehlen“ zu arbeiten. Daher ist die Übertragung in andere Dialekte sehr einfach.

In der vorgestellten Form läuft das Programm unter CP/M, HEBAS und FLOMON. Um es mit dem „normalen“ RDK-BASIC laufen zu lassen, müssen nur die

typischen „Flomon-Befehle“ umgeschrieben werden. Dies ist im Programm-Kommentar erläutert.

Ein Beispiel: Unter FLOMON lautet der Befehl zum Zeichnen einer Linie: Print „D 100 200“ = Draw x=100 y=200 oder mit Variablen Print „D „x,y Unter RDK-BASIC der gleiche Befehl: Drawto 100,200 oder Drawto x,y

Das Programm wurde übrigens mit ZEAT in der Form beschrieben, wie es Dr. HEHL in einer der letzten LOOP beschrieben hat. Eine hervorragende Arbeitshilfe.

Das Grafik-Programm ist in der vorliegenden Form natürlich nur als Einstieg zu verstehen. Ich denke aber, daß der weitere Ausbau recht einfach sein dürfte? Viel Spaß!

```

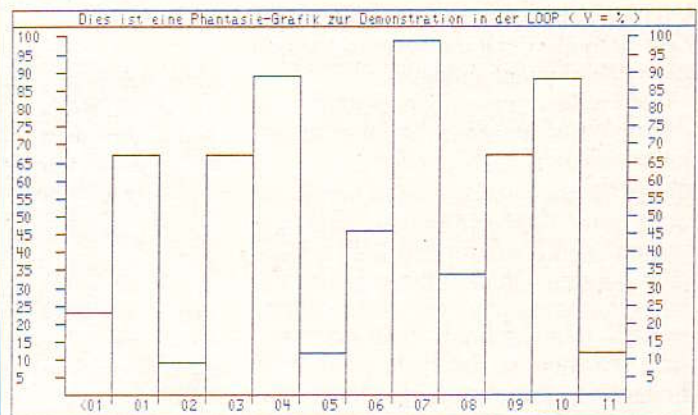
490 ***** BALKENGRAFIK FÜR DEN NDR-COMPUTER UNTER CP/M 2.2 UND FLOMON *****
500 ***** BY CARSTEN DENNEBURG (C) 4468 NORDHORN / FEBRUAR 1987 *****
530 *****
540 * läuft auch ohne Flomon mit dem RDK-BASIC ( z.B. SBC-2 ) ; dazu *
550 * bitte die Flomon-Befehle umschreiben (Moveto, Drawto pp.) s. Text ! *
560 *****
1000 DIM W$(20)                'Platz schaffen für X-Beschriftung
1050 W$(1)="<01"              'Beschriftung der X-Achse in diesem
1100 W$(2)=" 01"              'Fall mit Zahlen 01 - 11
1150 W$(3)=" 02"
1200 W$(4)=" 03"
1250 W$(5)=" 04"
1300 W$(6)=" 05"
1350 W$(7)=" 06"
1400 W$(8)=" 07"
1450 W$(9)=" 08"
1500 W$(10)=" 09"
1550 W$(11)=" 10"
1600 W$(12)=" 11"
1650 CLOSE #0
1700 DIM Y(500)
1750 GRAF$=CHR$(27)+CHR$(27)+"G"
1800 NX=40
1850 NY=15
1900 Z=0
2000 INPUT "UEBERSCHRIFT J/N :";UE$
2050 IF UE$="N" THEN GOTO 2150
2100 INPUT "TEXT:";T$
2150 INPUT "MAXIMALER Y-WERT :";H
2200 INPUT "ANZAHL DER X-WERTE :";A
2250 IF A>450-NX THEN A=450-NX
2300 UX=A
2350 INPUT "UNTERTEILUNG Y-ACHSE :";UY
2400 Z=H/UY
2450 FOR I=1 TO A
2500 INPUT "Y-WERT :";Y
2550 N=N+1
2600 Y(N)=Y
2650 Y(N)=(238-NY)/H*UY
2700 NEXT
2750 N=0
2800 YU=UY
2900 CLOSE#0
2950 PRINT GRAF$
3000 PRINT "M ",NX,NY
3050 Y=(238-NY)/H*UY
3100 Y1=Y
3150 FOR I=1 TO Z
3200 N=N+1
3250 IF N<2 THEN Y=Y+NY
3300 PRINT "D ",NX,Y
3350 PRINT "D ",NX-7,Y
3400 PRINT "M ",NX-120,Y-4
3450 PRINT "B ";YU
3500 PRINT "M ",NX,Y
3550 Y=Y+Y1
3600 YU=YU+UY
3650 NEXT
3750 N=0
3800 X=INT((470-NX)/A)
3850 YR=(NX+X)+(X*(A-1))
3900 PRINT "M ",YR,NY
3950 Y=(238-NY)/H*UY
4000 Y1=Y
4050 YU=UY
4100 FOR I=1 TO Z
4150 N=N+1
4200 IF N<2 THEN Y=Y+NY
4250 PRINT "D ",YR,Y
4300 PRINT "D ",YR+7,Y

```

```

4350 PRINT "M ",YR+10,Y-4
4400 PRINT "B";YU
4450 PRINT "M ",YR,Y
4500 Y=Y+Y1
4550 YU=YU+UY
4600 NEXT
4700 PRINT "M 0 244"
4750 PRINT "R 510 10"
4800 PRINT "M 100 245"
4850 PRINT "B";T$
4900 PRINT "M 0 0"
4970 PRINT "R 510 244"
4990 N=0
5000 YU=UY
5100 PRINT "M ",NX,NY
5150 X=INT((470-NX)/A)
5200 X1=X
5250 FOR I=1 TO A
5300 N=N+1
5350 IF N<2 THEN X=X+NX
5400 PRINT "D ",X,NY
5450 PRINT "D ",X,NY-5
5500 PRINT "M ",X-(X1/2+5),NY-10
5550 PRINT "B";W$(N)
5600 PRINT "M ",X,NY
5650 X=X+X1
5750 NEXT
5800 N=0
5850 PRINT "M ",NX,NY
5900 X=X1
5950 FOR I=1 TO A
6000 N=N+1
6050 IF N<2 THEN X=X+NX
6100 PRINT "D ",X,NY
6150 PRINT "D ",X,NY+Y(N)
6200 PRINT "D ",X-X1,NY+Y(N)
6250 PRINT "D ",X-X1,NY
6300 X=X+X1
6350 NEXT
6400 INPUT W$
6450 IF W$="W" THEN PRINT "A"
6500 GOTO 6400

```



FÜR 68 000-EINSTEIGER

GALACTICA

ein bekanntes Geschicklichkeitsspiel für 2 Personen

von Gerhard Kallmeyer, im Hesse 25, 3008 Garbsen

Programm in BASIC 1.5 für CPU Z80. Das Programm läuft auch mit der SBC2; dann muß die Zeile 300 GOTO 320 lauten; die Zeilen 20,40,50,310,330,1100 bis 1130 entfallen, damit der Speicherplatz reicht.

Der Kapitän des Raumschiffes Galactica gerät in einen Meteorsturm. Er muß ver-

suchen, eine Kollision zu vermeiden. Bei einem Zusammenstoß explodiert und verglüht das Raumschiff. Mit RUN wird das Programm gestartet. Das Spiel wird je nach Schwierigkeit mit 1, 2 oder 3 gestartet. Die Taste V bewegt das Raumschiff nach links, die Taste M nach rechts.

```

$ = Dollarzeichen
10 CLRS
20 F$="RAUMSCHIFF": Z=2: U=1: X=15: Y=211: GOSUB 1000
30 F$="GALACTICA": Z=4: U=2: X=11: Y=166: GOSUB 1000
35 Z=2: U=1
40 F$="IM": X=19: Y=131: GOSUB 1000
50 F$="NEEBOR-STURM": X=14: Y=96: GOSUB 1000
60 F$="STEUERUNG RAUMSCHIFF << = V M = >>": X=0: Y=20: GOSUB 1000
70 F$="START 1 = leicht 2 = mittel 3 = schwer": Y=0: GOSUB 1000
80 A$=GET$: IF A$="1" THEN 91
81 IF A$="2" THEN 92
82 IF A$="3" THEN 93
90 GOTO 80
91 C=3: GOTO 100
92 C=5: GOTO 100
93 C=7: GOTO 100
100 DIM S(2): DIM A(15): DIM B(15): R=.5
110 CLRS: Z=6: U=2: X=18: Y=164: F$="*"
120 FOR E=1 TO 15: A(E)=0: B(E)=0: NEXT
130 F=0: E=0
140 FOR H=1 TO 23: PRINT: NEXT
150 PAGE 0,0: P=P+1: E=E+1
160 IF E>15 THEN E=1
170 IF A(E)>X-1 AND A(E)<X+3 THEN 300
180 IF B(E)>X-1 AND B(E)<X+3 THEN 300
200 OUT 112,1: GOSUB 1000
210 OUT 112,0: A$=GET$
220 IF A$="m" AND X<36-C THEN X=X+1
225 IF A$="v" AND X>1+C THEN X=X-1

```

```

230 GOSUB 1000
240 A=INT((19-C)*RND(1)+1+C): B=INT((19-C)*RND(1)+20)
250 PRINT TAB(A);"*";TAB(B);"*"
260 A(B)=A: B(B)=B: GOTO 150
300 GOSUB 1100
310 FOR W=1 TO 800: NEXT
320 OUT 112,1: GOSUB 1000
330 GOSUB 1100
340 OUT 112,0: FOR W=1 TO 3000: NEXT
400 CLRS: R=R+.5: S=S+1: IF S>2 THEN S=1
410 PRINT "DURCHGANG: ";INT(R)
420 PRINT "SPIELER : ";S
430 PRINT "PUNKTE : ";P-15
440 S(S)=S(S)+P-15: PRINT: PRINT
450 PRINT "PUNKTZAHL SPIELER 1: ";S(1): PRINT
460 PRINT "PUNKTZAHL SPIELER 2: ";S(2): PRINT: PRINT: PRINT
470 PRINT "W = WIEDERHOLUNG E = ENDE"
500 A$=GET$
502 IF A$="w" THEN 110
504 IF A$="e" THEN 520
510 GOTO 500
520 CLRS: OUT 115,17: END
1000 OUT 115,16*2+U: MOVETO X*12,Y
1010 FOR L=1 TO LEN(F$): OUT 112,ASC(MID$(F$,L,1)): NEXT
1020 OUT 115,16*2+1: RETURN
1100 MOVETO X*12+15,130: DRAWTO X*12+15,205
1110 MOVETO X*12-45,140: DRAWTO X*12+75,200
1120 MOVETO X*12+85,135: DRAWTO X*12-35,195
1130 RETURN

```

Text (-bearbeitung)

Dieses Programm ist eine einfache Version einer Textbearbeitung, wie sie in ähnlicher Form schon mehrfach veröffentlicht worden ist. Dieses Programm soll aber – soweit es geht – durchschaubar ge- und beschrieben werden, so daß jeder, der daran interessiert ist, sich dieses Programm für seine eigenen Bedürfnisse anpassen kann.

Darüber hinaus aber soll es im Laufe der Zeit erweitert und verbessert werden, etwa durch die Möglichkeit, im Zusammenhang mit einem Betriebssystem mit Disketten zu arbeiten, Grafiken zu drucken und Zeichensätze zu verändern usw. Zwei Aufgaben soll diese erste Version dieses Programms erfüllen:

- Texte sollen eingegeben werden
- Texte sollen auf einem Drucker ausgegeben werden.

Für die Eingabe von Texten ist schon ein recht guter Editor vorhanden, der hier einfach benutzt werden soll. Die Textausgabe auf dem Drucker hingegen ist mit dem Grundprogramm nur recht einfach möglich (Menuepunkt: Text drucken). Ganz toll wäre es natürlich, wenn etwa Breitschrift, Schrägschrift oder Unterstreichen auf dem Druckbild und auf dem Bildschirm gleich erscheinen würden. Im Prinzip ist es durchaus möglich, auch diese Schriftarten auf dem Bildschirm darzustellen, doch müßte dazu der Editor erheblich umgeschrieben werden. Vielleicht findet Rolf-Dieter Klein hierzu irgendwann einmal eine Gelegenheit.

Weiter wäre denkbar, die Steuersequenz für den Drucker mit dem Editor einzugeben. Viele Steuerzeichen sind aber kleiner als \$20, also sog. „nicht druckbare Zeichen“. Solche Zeichen von der Tastatur benutzt der Editor aber zur Bildschirmsteuerung (die „CTRL + Taste“

Eingaben) oder ignoriert sie (außer \$A (LF) und \$D (CR)).

Die hier genutzte Möglichkeit besteht darin, bestimmte Zeichenfolgen des Textes im Druckprogramm durch Steuersequenzen zu ersetzen. Das Nummernzeichen soll hier bedeuten, daß eine Druckersteuerung folgen soll und das nachfolgende Zeichen bestimmt, um welche Steuerung es sich handelt. Das Druckprogramm ersetzt z. B. „=B“ durch „\$E“ (= Breitschrift). Die verwendeten Zeichen sind ähnlich gewählt wie beim Druckprogramm von JADOS.

```

#b = Breitschrift aus
#s = Schmalschrift aus
#u = Unterstreichen aus
#k = Kursivschrift aus
#h = hochgestellt aus
#t = tiefgestellt aus
#f = Fettdruck aus
#d = Doppeldruck aus
#p = Proportionalschrift aus
#q = Qualitätsschrift (NLQ)
#< = BS (Schritt nach links)
#B = Breitschrift(eine Zeile)
#S = Schmalschrift
#U = Unterstreichen
#K = Kursivschrift
#H = hochgestellt
#T = tiefgestellt
#F = Fettdruck
#D = Doppeldruck
#P = Proportionalschrift
#E = Elite - Schrift
#N = Normalschrift (Pica)
#^ = Neue Seite

```

Voreingestellt wird der Drucker auf Normalschrift, 8 Zeichen linker Rand, 12 dot Zeilenvorschub, Din A4 (12 Zoll Blattlänge).

Zum Programm

Damit ein solches Programm übersichtlich bleibt, auch wenn es im Laufe der Zeit noch wächst, sollte es systematisch und übersichtlich aufgebaut und ausreichend kommentiert sein. Deswegen sind hier Kreuz- und Quersprünge (bra . . .) möglichst vermieden worden und die meisten Aufgaben als Unterprogramme (bsr . . ., jsr . . .) ausgeführt worden. Das scheint zwar manchmal aufwendig und zum Teil überflüssig zu sein, aber jeder, der schon einmal an einem längeren eigenen Programm geschrieben hat, weiß, wie leicht es vorkommt, daß man durch sein eigenes Programm nicht mehr durchsteigt.

Das Programm „TEXT“ ruft systematisch nur die drei Programme „ANFANG“, „HAUPT“ und „SCHLUSS“ auf. Daneben rettet es den alten Editor-Bereich.

Das Unterprogramm „ANFANG“ soll alle Voreinstellungen für dieses Textprogramm vornehmen und das Unterprogramm „SCHLUSS“ soll alle Einstellungen wieder zurücksetzen. Dies wird vor allen Dingen bei späteren Ausbaustufen wichtig. Hier wird in ANFANG nur die Bildschirmausgabe über die Grundprogrammfunktion CO2 vorbereitet, der Editor auf den in der variablen „Textspeicher“ festgelegten Speicherbereich gelegt und das Programm „Vorspann“ aufgerufen, das den Vorspann eine Sekunde zeigt.

Wichtig ist hier das Hilfsprogramm „atrans“. Es gibt eine Zeichenserie an die Schnittstelle CO2 weiter, also je nach vorheriger Einstellung an den Bildschirm (CRT), an den Drucker (LST) oder an ein eigenes Programm (USR). Dabei muß in A1 die Anfangsadresse der Zeichenliste stehen. Damit man jedes beliebige Zeichen weitergeben kann, ist die übliche Enderkennung durch ein „Null“-Byte nicht möglich, denn sonst könnte man ja die Null (\$0) z. B. nicht an den Drucker weitergeben, was aber bei manchen

Steuersequenzen notwendig wird. Bei „atrans“ muß deswegen das erste Byte der Liste die Anzahl der nachfolgenden Bytes (Zeichen) enthalten. Durch die beiden „movem.1 . . .“- Befehle werden alle benutzten Register auf den Stack gerettet und nachher wiederhergestellt, wie dies die meisten Routinen des Grundprogramms auch vornehmen.

Die eigentliche Textbearbeitung beginnt im Programm „HAUPT“. Hier wird in einem Menue abgefragt, ob der Editor gewünscht wird, ausgedruckt werden soll, oder ins Grundprogramm zurückgekehrt werden soll. In Wirklichkeit wird eigentlich nur geprüft, ob „O“ (drucken) oder „M“ (zurück) eingegeben worden ist. Wenn irgendein anderes Zeichen eingegeben wurde, wird in das Unterprogramm „Editor“ gesprungen. Man kann sich so einen bestimmten Menüpunkt voreinstellen. Bei der Eingabe von Buchstaben in Menues ist es günstig, Groß- und Kleinbuchstaben gleich zu behandeln, wie es auch im Grundprogramm geschieht. Sowohl „M“ als auch „m“ beenden hier das Programm.

Die Routine „Editor“ macht (vorläufig) nichts anderes, als den Editor aus dem Grundprogramm aufzurufen und nach Rückkehr den Bildschirm wieder neu für CO2 zu löschen.

Interessanter ist da das Druckprogramm. Zunächst wird der Drucker zurückgesetzt und auf ein bestimmtes Format eingestellt (s. o.). Mit dem Aufruf „jsr @ciinnt2“ wird die Funktion CI2 (einlesen von RAM) auf den Textstart eingestellt und mit LST die Ausgabe (CO2) auf den Drucker eingelenkt. Danach folgt die Textschleife. CI2 liest ein Zeichen ein. Dann wird geprüft, ob es sich um die Null (Textende) handelt. Wenn ja, dann Seitenvorschub und Rückkehr.

Als Zweites wird geprüft, ob das Zeichen in D0 größer als \$80 ist, Der Editor setzt nämlich bei deutschen Umlauten und dem „ß“ jeweils das Bit 7. Die eckige Klammer-zu „]“ (amerikanischer Zeichensatz) hat z. B. der ASCII- Wert \$5D. Das auf der gleichen Taste liegende große „Ü“ (eigentlich auch \$5D, aber deutscher Zeichensatz) legt der Editor als \$DD in den Speicher (er addiert \$80). Also kann man deutsche Sonderzeichen daran erkennen, daß der Zahlenwert >\$80 ist. Drucker auf deutschen Zeichensatz umstellen, \$80 subtrahieren, Zeichen ausgeben und Drucker wieder auf amerikanischen Zeichensatz. Nichts anderes macht das Unterprogramm „umlaut“.

Als Drittes wird geprüft, ob es sich bei dem Zeichen um das Doppelkreuz „#“

handelt. Wenn ja, ist die Druckersteuerung gemeint und es erfolgt ein Sprung in „steuer“. Hier wird das folgende Zeichen eingelesen, was die Art der Druckersteuerung festlegt. Die jeweilige Steuersequenz wird in „buffer“ abgelegt und durch „atrans“ an den Drucker geschickt (am Ende von „steuer“). „#D“ soll z. B. Doppeldruck bewirken. Mit dem Befehl „move.1#\$021B4700,buffer“ werden vier Bytes in buffer abgelegt: \$02, \$1B, \$47, \$00. \$02 oder 2 ist die Anzahl der Bytes (für „atrans“). Die beiden folgenden Bytes werden also an den Drucker geschickt: \$1B und \$47 oder ESC ‚G‘. Das ist genau die Steuersequenz, mit der der Drucker auf Doppeldruck umgeschaltet wird (Handbuch!).

Ein Studium des eigenen Druckerhandbuches gibt jedem Benutzer die Möglichkeit, seine eigene Druckersteuerung auszugestalten. Vorsicht bei Befehlen, die den Seitenvorschub durcheinander bringen könnten!

Soweit dieses Programm. In der nächsten Erweiterung sollen die Steuerbefehle erweitert werden, das Format über Menue frei gewählt werden und einige Editoreinstellungen vorgenommen werden können.

* Alle so: „*;*“ gekennzeichnete Zeilen müssen überprüft und ev. der eigenen Rechnerkonstellation angepaßt werden !!!!
 * Alle so: „;;;“ gekennzeichneten Zeilen enthalten Steuerzeichen für den Drucker und müssen überprüft und ev. an den eigenen Drucker angepaßt werden!!!!

```

textspeicher equ $10000      ;* Hier den Anfang des
                             * Textspeichers festlegen

TEXT:      ***** Oberstes Programm
jsr @getstx      ; alte Textstartadresse ermitteln
move.l d0,-(a7)  ; und retten
jsr ANFANG      ; nur die drei
jsr HAUPT       ; Aufrufe (wegen der
jsr SCHLUSS     ; Systematik)
move.l (a7)+,d0 ; alte Textstartadresse vom Stack
jsr @putstx     ; und wieder einsetzen
rts

-----
ANFANG:      ***** Alle Voreinstellungen
jsr @clrscreen  ; Vorb. CO2-Ausgabe
jsr Vorspann   ; Vorspann darstellen
move.l #Textspeicher,d0 ; Editor- Adresse neu
jsr @putstx    ; einstellen
rts

tvorspann: dc.b 212,10,13
dc.b '*****',10,13
dc.b ' * T E X T - Bearbeitung ',10,13
dc.b ' * Vers. 1.1 ',10,13
dc.b ' * (C) Elmer Schnuit 1986 ',10,13
dc.b '*****',10,13

ds 0

Vorspann:      ***** Stellt Vorspann dar
jsr @crt       ; Ausgabe auf Bildschirm
lea tvorspann,a1 ; Adr. Vorspanntext;
jsr atrans     ; darstellen lassen;
move.l #10,d0  ; 1 Sekunde (.1 !!)
jsr @delay    ; darstellen.
jsr @clrscreen ; und wieder löschen
rts

atrans:      ***** Wichtiges Hilfsprogramm !
* Gibt Zeichenserie (max 255!)
* an CO2 weiter, Anfangsadresse
* in A1, erstes Zeichen = Anzahl
* der folgenden Zeichen
* benutzte Register retten.
* D7 ist Zähler

movem.l d0/d1/d7,-(a7)
clr d7
    
```

```

move.b (a1)+,d7      ; Anzahl in D7
sub.b #1,d7         ; -1 wegen 'dbra'
atrans1:            ; Schleife
move.b (a1)+,d0     ; Zeichen in D0
jsr @co2            ; und weitergeben
dbra d7,atrans1     ; Schleifenende
movem.l (a7)+,d0/d1/d7 ; Register zurück
rts

-----
HAUPT:      ***** Hauptprogramm mit Abfragen
lea tfrage1,a1      ; Menueabfrage
jsr Abfrage1
cmp.b #'O',d0       ; Drucken ?
bne haupt1
jsr Drucken
bra haupt           ; wenn fertig, zurück
haupt1:
cmp.b #'M',d0       ; beendet?
beq haupte         ; dann ans Ende
cmp.b #'m',d0       ; auch Kleinschreibung
beq haupte
haupt2:
jsr Editor          ; Editor ist voreingestellt
bra haupt          ; Endlosschleife
haupte:            ; Ende nur mit 'M'
rts

tfrage1: dc.b 68,10,13
dc.b 'O = Text drucken ',10,13
dc.b 'I = EDITOR ',10,13
dc.b 'M = Menue ',10,13

ds 0

Abfrage1:      ***** Menuedarstellung und Abfrage
jsr @crt        ; Ausgabe auf Bildschirm
jsr atrans     ; s.o.
jsr @ci        ; Tastatur abfragen, in D0
rts

Editor:      ***** Schreiben
jsr @edit     ; nur Editoraufruf
jsr @clrscreen ; neu für CO2
rts

Treset:      ; Druckerreset
dc.b 2,27,64

tvorein: dc.b 16 ; Druckervoreinstellung
dc.b 27,67,0,12 ;;; 12 Zoll Blattlänge
dc.b 27,65,12 ;;; 12 Dot Vorschub
dc.b 27,78,10 ;;; 6 oben und 4 Zeilen unten frei
dc.b 27,108,8 ;;; 8 Zeichnen Rand
dc.b 27,82,0 ;;; amerikanischer Zeichensatz
    
```

```

ds 0

reset:          ***** Drucker rücksetzen
                ; auf Drucker
                ; Reset- Anweisung
jsr @l1st
lea treset,a1
jsr atrans
rts

vorein:        ***** Drucker voreinstellen
                ; auf Drucker
                ; Voreinstellungen
jsr @l1st
lea tvorein,a1
jsr atrans
rts

Drucken:      ***** Text ausdrucken
                ; Drucker rücksetzen
                ; Drucker voreinstellen
                ; CI2 einstellen
                ; auf Drucker
                ; Druckschleife
jsr reset
jsr vorein
jsr @ciinit2
jsr @l1st
drucken1:
clr d0
jsr @ci2          ; Zeichen in D0
cmp.b #0,d0      ; Ende-Test
beq dende        ; dann zurück
cmp #80,d0       ; Umlaut?? (= größer 80)
blt drucken3
jsr umlaut       ; dann umlaut
drucken3:
cmp.b #' ',d0    ; Steuerzeichen?
bne drucken2     ; ja, dann Steuer
jsr steuer
drucken2:
jsr @co2         ; Zeichen an CO2
bra drucken1     ; Schleifenende
dende:
move #12,d0      ; danach LF
jsr @co2
rts

buffer: ds.b 20

umlaut:        ***** Umsteuerung deutscher Z.satz
                ;;; Umschaltung deutscher
                ; Zeichensatz
move.l #031b5202,buffer
lea buffer,a1
jsr atrans
sub.b #80,d0     ; normalisieren
jsr @co2         ; an co2
move.l #031b5200,buffer
lea buffer,a1
jsr atrans
clr d0           ; und D0 löschen
rts

steuer:        ***** Steuerzeichen für Drucker
                ; einlesen
                ; '#' ?, dann wirklich
                ; #
                ; 1 Zeichen: 23 (= #)
jsr @ci2
cmp.b #' ',d0
bne steu1
move #0123,buffer
bra steuend

steu1:
cmp.b #'B',d0   ; B = Breit (Zeile!)
bne steu2
move #010E,buffer
bra steuend

steu2:
cmp.b #'b',d0   ; b = breit aus
bne steu3
move #0114,buffer
bra steuend

steu3:
cmp.b #'S',d0   ; S = schmal
bne steu4
move #010f,buffer
bra steuend

steu4:
cmp.b #'s',d0   ; s = schmal aus
bne steu5
move #0112,buffer
bra steuend

steu5:
cmp.b #'U',d0   ; U = unterstrichen
bne steu6
move.l #031B2D31,buffer
bra steuend

steu6:
cmp.b #'u',d0   ; u = unterstrichen aus
bne steu7
move.l #031B2D30,buffer
bra steuend

steu7:
cmp.b #'K',d0   ; K = Kursiv
bne steu8
move.l #021B3400,buffer

```

```

bra steuend
steu8:
cmp.b #'k',d0   ; k = kursiv aus
bne steu9
move.l #021B3500,buffer
bra steuend

steu9:
cmp.b #'N',d0   ; N = Normalschrift (Pica)
bne steu10
move.l #021b5000,buffer
bra steuend

steu10:
cmp.b #'E',d0   ; E = Elite
bne steu11
move.l #021B4D00,buffer
bra steuend

steu11:
cmp.b #'Q',d0   ; Q = Qualitätsschrift (NLD)
bne steu12
move.l #021B2800,buffer
bra steuend

steu12:
cmp.b #'H',d0   ; H = hochgestellt
bne steu13
move.l #031B5330,buffer
bra steuend

steu13:
cmp.b #'h',d0   ; h = hochgestellt aus
bne steu14
move.l #021B5400,buffer
bra steuend

steu14:
cmp.b #'T',d0   ; T = tiefgestellt
bne steu15
move.l #031B5331,buffer
bra steuend

steu15:
cmp.b #'t',d0   ; t = tiefgestellt aus
bne steu16
move.l #021B5400,buffer
bra steuend

steu16:
cmp.b #'F',d0   ; F = Fettdruck
bne steu17
move.l #031B4500,buffer
bra steuend

steu17:
cmp.b #'f',d0   ; f = Fettdruck aus
bne steu18
move.l #021B4600,buffer
bra steuend

steu18:
cmp.b #'D',d0   ; D = Doppelt
bne steu19
move.l #021B4700,buffer
bra steuend

steu19:
cmp.b #'d',d0   ; d = doppelt aus
bne steu20
move.l #021B4800,buffer
bra steuend

steu20:
cmp.b #'^',d0   ; ^ = FF
bne steu21
move #010C,buffer
bra steuend

steu21:
cmp.b #'P',d0   ; P = proportional
bne steu22
move.l #031B7031,buffer
bra steuend

steu22:
cmp.b #'p',d0   ; p = proportional aus
bne steu23
move.l #031B7030,buffer
bra steuend

steu23:
cmp.b #'<',d0   ; < = BS
bne steu24
move #010B,buffer
bra steuend

steu24:
****           ; !!! eigene Erweiterungen
rts
steuend:
lea buffer,a1
jsr atrans
clr d0
rts

```

SCHLUSS:
rts

Nutzen der freien Sockel der BANKBOOT beim 68000

von Rüdiger Bäcker

Die Bank-Boot dient beim 68000 System des NDR-Computers lediglich zum Booten. Das hierzu benötigte Programm ist relativ kurz. Es bleibt also noch einiges an Platz auf der Karte. Wie man diesen Platz nutzen kann, soll hier einmal beschrieben werden.

Insgesamt sind auf der Bank-Boot Sok-

kel für 32k vorhanden. Der erste Sockel muß dabei mit einem Eprom bestückt sein, da das Bootprogramm dort steht. In diesem Eprom ist jedoch noch Platz für weitere Routinen. Die drei restlichen Sockel können nun wahlweise mit EPROMS, EEPROMS oder RAM's bestückt werden. Ich habe dabei die folgende Kombination gewählt:

1. Sockel EPROM mit Bootroutine und Utilitys
2. Sockel EPROM mit Utilitys
3. Sockel EEPROM für veränderbare die jedoch bei Abschalten des Rechners erhalten bleiben sollen.
4. Sockel RAM als geschützter Zwischenspeicher

Das Auslesen, bzw. Einschreiben der Daten erfolgt mit dem Programm BANKTRANS, welches Routinen für die einzelnen Aufgaben

- Lesen aus EPROM
 - Lesen & Schreiben EEPROM
 - Lesen & Schreiben RAM
- enthält.

Dabei wird im Register A0 die Quelle, im Register A1 das Ziel und im Register A2 die Endadresse des zu übertragenden Bereiches angegeben. Das Register D1 enthält noch einen Code für Lesen und Schreiben (D1 = 1 für Schreiben, D1 = 0 für Lesen). Beim Schreiben wird in Abhängigkeit von der Adresse, in die geschrieben werden soll, für RAM und EEPROM in unterschiedliche Routinen verzweigt. Schreibzugriffe auf die EPROMs oder auf eine Adresse über \$7FFF werden verweigert. Im Register D1 wird nach dem Rücksprung ein Fehlercode zurückgeliefert. Es steht dabei in D1.W :

- \$FF00 für fehlerfreien Transfer
- \$FFF1 für einen Transferfehler (z. B. von/auf Eprombereich oder über \$7FFF)
- \$FFF2 für einen Schreibfehler ins Ram
- \$FFF3 für einen Schreibfehler ins EEPROM

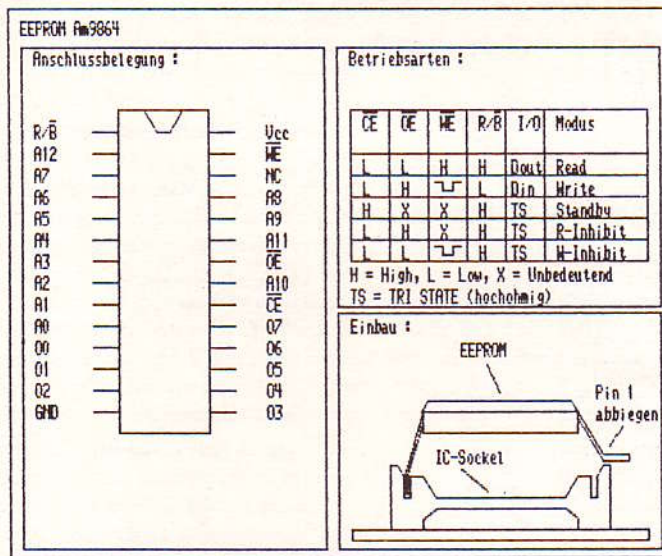
Die jeweiligen Quell- bzw. Zeilenadressen, bei denen der Fehler auftrat, werden dann noch in A0 und A1 zurückgeliefert. Beim Schreiben in das EEPROM ist eine besondere Routine erforderlich, da diese Speicherbausteine einige Besonderheiten aufweisen. Daher zunächst ein paar Worte zum grundsätzlichen Aufbau von EEPROMS.

Ein EEPROM ist ein „elektrical erasable programable read only memory“, oder auf gut deutsch ein elektrisch löschbarer, programmierbarer Nurlesespeicher. Im Prinzip also so etwas wie ein Eprom, mit dem Unterschied, das hier das Löschen nicht mit UV-Licht, sondern eben elektrisch geschieht. Genau das ist aber der große Vorteil gegenüber den EPROMS, denn das elektrische Löschen erfolgt bei den meisten EEPROMS, indem einfach ein neuer Wert in die entsprechende Speicherstelle geschrieben wird. Da die Speicherzellen in einem EEPROM jedoch nicht so aufgebaut sind

wie in einem Ram, dauert das Einschreiben etwas länger. Neuere EEPROM's haben daher einen internen Puffer, der einige Bytes mit einer von Rams gewohnten Geschwindigkeit aufnehmen kann. Um zu signalisieren, daß die Daten übernommen wurden, ist dann ein Ready/-Busy Anschluß vorhanden. Andere Eproms legen die eingeschriebenen Daten solange invertiert auf die Datenleitungen, bis sie komplett übernommen wurden. Um mit einer Software beide Typen programmieren zu können, ist die Software so ausgelegt, daß die eingeschriebenen Daten nach einem kurzen Wait wieder ausgelesen werden und solange neu eingeschrieben werden, bis sie vom EEPROM übernommen werden.

In Bild 1 ist das Timing und die Anschlußbelegung des Types AM9864 abgebildet; hier ist zu erkennen, daß die Anschlußbelegung bis auf den Pin 1 mit der eines RAM 6264 oder eines EPROM 2764 übereinstimmt. Daher braucht man nur den Pin 1 abbiegen und kann das EEPROM so problemlos einsetzen.

Auf diese Art ist also eine sinnvolle Nutzung des Speichers auf der Bank-Boot möglich. Anstelle der EEPROM's könnte man evtl. auch die Ram's mit einer Akkupufferung versehen.



So 16.11.86 - 16:18:53
Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 1

```

008000      ORG $B000          * WICHTIG !! BEREICH $0-$7FFF WIRD JEWEILS AUSGEBLENDET
008000      *
008000      *   B A N K T R A N S
008000      *
008000      *   UTILITY ZUR BENUTZUNG DER BANK-BOOT ALS SPEICHERKARTE
008000      *
008000      *   COPYRIGHT (C) 1984 BY RUEDIGER BAECKER - POSTFACH 4111 - 5820 BEVELSBERG
008000      *
008000      TEST:
008000      41F9 000C6000    LEA $C6000,A0      * QUELLE
008006      43F9 00004000    LEA $04000,A1      * ZIEL
00800C      45F9 000C7FFF    LEA $C7FFF,A2      * BIS
008012      123C 0001        MOVE.B #1,D1      * KENNUNG FUER WRITE
008016      6000 0019        BRA TRANS
00801A
00801A      TEST1:
00801A      41F9 00004000    LEA $4000,A0
    
```

```

008020      43F9 00012000    LEA $12000,A1
008026      45F9 00005FFF    LEA $5FFF,A2
00802C      123C 0000        MOVE.B #0,D1
008030
008030      TRANS:
008030      0C01 0001        CMP.B #1,D1
008034      6700 0006        BEQ NACHBANK
008038      6600 00AC        BNE VONBANK
00803C
00803C      NACHBANK:
00803C      B3FC 00004000    CMPA.L #4000,A1
008042      6D00 0090        BLT TRANSERR
008046      B3FC 00006000    CMPA.L #6000,A1
00804C      6D00 0044        BLT TOEEPROM
008050      B3FC 00007FFF    CMPA.L #7FFF,A1
008056      6C00 007C        BGE TRANSERR
00805A      95CB          SUBA.L A0,A2
00805C      220A          MOVE.L A2,D1
00805E      5341          SUBQ #1,D1
008060
008060      TORAM:
008060      1010          MOVE.B (A0),D0
008062      13FC 0000          MOVE.B #0,$FFFFFFCB
008066      FFFFFFFCB
00806A      12B0          MOVE.B D0,(A1)
00806C      1011          MOVE.B (A1),D0
00806E      13FC 0080          MOVE.B #80,$FFFFFFCB
008072      FFFFFFFCB
008076      B010          CMP.B (A0),D0
008078      6600 0060        BNE RAMERR
00807C      D1FC 00000001    ADDA.L #1,A0
008082      B3FC 00000001    ADDA.L #1,A1
008088      51C9 FF06        DBRA D1,TORAM
00808C      323C FF00        MOVE #FF00,D1
008090      4E75          RTS
008092
008092      TOEEPROM:
008092      95CB          SUBA.L A0,A2
008094      220A          MOVE.L A2,D1
008096      5341          SUBQ #1,D1
008098
008098      TOELP:
008098      1010          MOVE.B (A0),D0
00809A      13FC 0000          MOVE.B #0,$FFFFFFCB
00809E      FFFFFFFCB
0080A2      12B0          MOVE.B D0,(A1)
    
```

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

```

0080A4      203C 00000032    MOVE.L #50,D0
0080AA
0080AA      WAIT:
0080AA      51C8 FFFE        DBRA D0,WAIT
0080AE      1011          MOVE.B (A1),D0
0080B0      13FC 0080          MOVE.B #80,$FFFFFFCB
    
```

```

0080B4 FFFFFFFB
0080B8 B010      CMP.B (A0),D0      * QUELLE UND ZIEL VERGLEICHEN,
0080BA 6600 FFDC      BNE TOEELP        * WENN GLEICH, DANN WEITER MIT NAECHSTEM BYTE
0080BE
0080BE TOEEL:
0080BE D1FC 00000001    ADDA.L #1,A0      * NAECHSTES BYTE TRANSFERIEREN
0080C4 D3FC 00000001    ADDA.L #1,A1      * DAZU ADRESSZAEHLER ERHOEHEN
0080CA 51C9 FFCC      DBRA D1,TOEELP   * WIEDERHOLEN, BIS ALLES TRANSFERIERT
0080CE 323C FF00      MOVE #0,FF00,D1  * OK - MELDUNG IN D1
0080D2 4E75          RTS              * UND RETURN
0080D4
0080D4 TRANSERR:
0080D4 323C FFF1      MOVE #0,FF1,D1   * FEHLERMELDUNG IN D1
0080D8 4E75          RTS
0080DA
0080DA RAMERR:
0080DA 323C FFF2      MOVE #0,FF2,D1
0080DE 4E75          RTS
0080E0
0080E0 EEPERROR:
0080E0 323C FFF3      MOVE #0,FF3,D1
0080E4 4E75          RTS
    
```

```

0080E6
0080E6 VONBANK:
0080E6 B1FC 00007FFF    CMPA.L #0,7FFF,A0
0080EC 6C00 FFE6      BGE TRANSERR
0080F0 95CB          SUBA.L A0,A2
0080F2 220A          MOVE.L A2,D1
0080F4 5341          SUBB #1,D1
0080F6
0080F6 VONLP:
0080F6 13FC 0000      MOVE.B #0,0,FFFFFCB
0080FA FFFFFFFB
0080FE 101B          MOVE.B (A0)+,D0
008100 13FC 00B0      MOVE.B #B0,0,FFFFFCB
008104 FFFFFFFB
008108 12C0          MOVE.B D0,(A1)+
00810A 51C9 FFEA      DBRA D1,VONLP
00810E 323C FF00      MOVE #0,FF0,D1
008112 4E75          RTS
008114
008114 END.
0E958E Ende-Syaboltabelle
    
```

Hilfsprogramme für den 68000 – Teil 8

von Rüdiger Bäcker

Heute nochmals eine kurze Routine zum Ausdruck von Assemblerlistings. Das gab es zwar schon einmal, jedoch wurde die Routine zwischenzeitlich optimiert und um eine nützliche Zusatzfunktion erweitert.

Da man im Laufe der Zeit sicher so einige Listings gedruckt hat, ist es sinnvoll, die Listings mit dem Datum des Ausdrucks zu versehen. Das kann manuell von Hand geschehen oder mit dem Computer. Wer

die Uhrenkarte besitzt, kann die 1. Möglichkeit zu den Akten legen, das hier vorgestellte Programm erledigt diese Aufgabe.

Die Funktion in kurzen Worten: Zunächst wird der Drucker wieder auf Schmalschrift und amerikanischen Zeichensatz umgeschaltet. Dann wird die Routine ZEIT aufgerufen, die ihrerseits über die \$GETUHR-Routine des Grundprogrammes, das Datum und die Zeit in einen Puffer schreibt. Da die Daten dort im BCD-Format abgelegt werden, müssen wir für eine Umwandlung ins druckbare ASCII-Format sorgen. Dies ge-

schieht in zwei Stufen: Zunächst wird der gewünschte Wert mittels eines über Tabelle verwalteten Adressoffsets ins Register D0 geholt und in der Routine BCDHEX gewandelt. Dabei stehen die Einer anschließend im Register D1, die Zehner im Register D0. Diese Routine kann sicher auch mal in eigene Programme übernommen werden. Nach der Wandlung in HEX wird durch Addieren von \$30 ein ASCII-Zeichen erzeugt und in einen Ausgabepuffer geschrieben. Der Ausdruck erfolgt dann mit der Routine PRINT. Nun wird die C02-Ausgabe auf den Drucker gelenkt und der Assembler gestartet – fertig.

So 16.11.86 - 16:41:18
Rolf-D.Klein 68000/08 Assembler 4.3 (C) 1984, Seite 1

```

010000      ORG #10000
010000
010000      *   A S S P R I N T   & B
010000      *
010000      *   PROGRAMM ZUM AUSDRUCK VON ASSEMBLERLISTINGS MIT DATUM UND UHRZEIT
010000      *
010000      *   COPYRIGHT (C) 1986 BY RUEDIGER BAECKER - POSTFACH 4111 - 5820 BEVELSBERG
010000
010000      KOPF:
010000      DC.L $55AA01B0
010004      4153535052494E DC.B 'ASSPRINT'
010008      54
01000C      00000024      DC.L START-KOPF
010010      000000FB      DC.L ENDE-KOPF
010014      01
010015      00 00 00      DC.B 0,0,0
010018      00000000      DC.L 0,0,0
01001C      00000000
010020      00000000
010024
= 000000F4      PUFFER1 EDU #F4
= 000000FC      PUFFER2 EDU PUFFER1+8
010024
010024      START:
010024      BSR INIT          * HIER HAUPTPROGRAMM
010028      6100 002B      BSR INIT          * DRUCKER INITIALISIEREN
01002B      6100 003C      BSR ZEIT          * DATUM UND ZEIT DRUCKEN
01002C      3E3C 0055      MOVE #SYMCLR,D7
010030      4E41          TRAP #1
010032      3E3C 0032      MOVE #1,ST,D7      * AUSGABE AUF DRUCKER GEBEN
010036      4E41          TRAP #1
010038      3E3C 0041      MOVE #ASSEMBLE,D7 * UND ASSEMBLER AUFRUFEN
01003C      4E41          TRAP #1
01003E      3E3C 0034      MOVE #NIL,D7       * UND WIEDER AUF NUR FEHLER
010042      4E41          TRAP #1
010044      41FA 004C      LEA INIT(PC),A0
010048      6100 0072      BSR PRINT
01004C      4E75          RTS
01004E
01004E      INIT:
01004E      41FA 0096      LEA INITAB(PC),A0
010052      6100 006B      BSR PRINT          * UND DANN AN DRUCKER
010056      4E75          RTS
010058
010058      BCDHEX:
010058      3200          MOVE D0,D1          * WANDELT BCD-ZAHL AUS D0 IN HEXIAHL
01005A      0240 000F      AND #0F,D0          * WERT AUCH IN D1
01005E      0241 00F0      AND #0F,D1          * EINER IN D0
01005E      0241 00F0      AND #0F,D1          * ZEHNER IN D1
    
```

```

010062      E859          ROR #4,D1          * UND NOCH ROTIEREN
010064      4E75          RTS
010066
010066      ZEIT:
010066      41ED 00F4      LEA PUFFER1(A5),A0
01006A      3E3C 0073      MOVE #GETUHR,D7
01006E      4E41          TRAP #1
010070      43ED 00FC      LEA PUFFER2(A5),A1
010074      45FA 00B0      LEA TAB(PC),A2
010078      7604          MOVEQ.L #5-1,D3
01007A
01007A      LOOP:
01007A      141A          MOVE.B (A2)+,D2
01007C      1030 2000      MOVE.B #0(A0,D2),D0
010080      6100 FFD6      BSR BCDHEX
010084      0600 0030      ADDI.B #30,D0
    
```

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

```

01008B      0601 0030      ADDI.B #30,D1
01008C      12C1          MOVE.B D1,(A1)+
01008E      12C0          MOVE.B D0,(A1)+
010090      0C02 0004      CMP.B #4,D2
010094      6600 000A      BNE LOOP1
010098      12FC 002D      MOVE.B #'-',(A1)+
01009C      6000 0006      BRA LOOP2
0100A0
0100A0      LOOP1:
0100A0      12FC 002E      MOVE.B #'.',(A1)+
0100A4
0100A4      LOOP2:
0100A4      51CB FFD4      DBRA D3,LOOP
0100AB      93FC 00000001    SUBA.L #1,A1
0100AE      12BC 00FF      MOVE.B #FF,(A1)
0100B2      41ED 00FC      LEA PUFFER2(A5),A0
0100B6      6100 0004      BSR PRINT
0100BA      4E75          RTS
0100BC
0100BC      PRINT:
0100BC      101B          MOVE.B (A0)+,D0
0100BE      0C00 00FF      CMP.B #FF,D0
0100C2      6700 000C      BEQ ENDE
0100C6      3E3C 0016      MOVE #10,D7
0100CA      4E41          TRAP #1
0100CC      6000 FFE6      BRA PRINT
0100D0
0100D0      ENDE:
0100D0      103C 000D      MOVE.B #D,D0
0100D4      3E3C 0016      MOVE #10,D7
0100D8      4E41          TRAP #1
0100DA      103C 000A      MOVE.B #A,D0
    
```

```

0100DE 3E3C 0016 MOVE #10,D7
0100E2 4E41 TRAP #1
0100E4 4E75 RTS
0100E6
0100E6 1B 0F 1B 5200 INITTAB: DC.B #1B,15,#1B,'R',0,#1B,'1',B,#7,#D,#A,$FF
0100EB 1B 6C08 07 0D
0100F0 0A FF
0100F2 1B 4007 FF INIT1: DC.B #1B,'0',#7,$FF
0100F6 02 03 04 00 01 TAB: DC.B 2,3,4,0,1
0100FB
0100FB ENDEA:
0100FB
0100FB END.
    
```

0E96BA Ende-Symboltabelle

---GES-GES-GES---
07/09/86

```

*****
* LAUF SCHRIFT *
* (c) 1986 by *
* Johannes Hunter *
* 14.5.1986 *
*****
* fuer GES, zur Unterhaltung!
org #15000 * beliebig
s:
warten:
    
```

```

jsr $wait
move.b #511,$#ffff73
move #511,d1
clr d2
schleife:
jsr $moveto
move #0,d0
bsr schreibe
syl:
jsr $sync
beq.s syl
jsr $moveto
move #1,d0
bsr schreibe
jsr $csts
bne ende
subq #2,d1
cmp #-400,d1
bgt schleife
bra.s
schreibe:
jsr $cmd
lea text,a0
schl2:
move.b (a0)+,d0
beq.s ende
jsr $cmd
bra.s schl2
ende:
rts
Text:
dc.b "Es war einmal ein Mann, der hatte sieben Soehne. Die sieben Soeh
dc.b " sprachen: "Vater, erzaehle uns eine Geschichte!". Da fing
dc.b " der Vater an:",0
    
```

68000 — YOGIDOS UND JADOS, RL-BASIC

Komfortables Grundprogramm

von Klaus Janßen

Mit seinem Grundprogramm hat Rolf-Dieter Klein eine hervorragende Grundlage für das Arbeiten mit dem NDR-Klein-Computer geschaffen. Nur leider ist die Benutzerführung mit den vielen Menüs für den erfahrenen Anwender recht unständig. Dies hat einige findige Köpfe dazu veranlaßt, Änderungen am Grundprogramm vorzunehmen. Die meisten Vorschläge dieser Art – auch die LOOP hat einige Artikel dazu veröffentlicht – beschränken sich jedoch darauf, einige „Patches“ (engl. für Flicker) vorzunehmen, um z. B. den Editor auf 80 Zeichen/Zeile einzustellen oder einen schnelleren Programmieralgorithmus für Eproms zu haben. Für eine Änderung der Benutzerführung ist dieses Patchen allerdings weniger gut geeignet.

Im folgenden Beitrag möchte ich ein sehr einfaches Verfahren vorstellen, mit dem jeder sein Grundprogramm nach „Herzenslust“ verändern kann, ohne auch nur ein einziges Byte im Grundprogramm zu ändern.

Da Herr Klein lobenswerterweise sehr modular programmiert hat, stehen die

meisten Funktionen der Menüs als Unterprogramme zur Verfügung. Die Einsprungsadressen erhält man dann durch intensives Studieren des Grundprogrammlistings. Man muß nun lediglich noch die gewünschte Menüführung „drumherum“ programmieren.

Das nachfolgend abgedruckte Programm stellt einen solchen Vorschlag dar. Das Grundmenü enthält alle wichtigen Funktionen:

GRUNDPROGRAMM-MENUE

- A = Aendern
- B = Starten
- C = Ansehen
- D = Symboltabelle
- E = Editor
- F = Assembler
- G = Bibliothek
- H = Optionen
- I = EPROM progr.
- J = EPROM lesen
- K = Speicherbereiche
- L = Text drucken
- M = IO lesen
- N = IO setzen
- O = Einzelschritt
- Z = BEENDEN

Bild 1: Grundmenü

```

*****
* Grundprogramm-Interface *
* fuer Original-Grundprogramm V 4.3 *
* *
* 20.11.1986 von Klaus Janssen *
*****
-----
* Hier Tabelle der Einsprungsadressen
* relativ zum Grundprogramm anfang
* !! Bei geaenderten Grundprogrammen
* hier die geaenderten Adressen
* eintragen !!
-----
    
```

aendere	EQU \$28A4	* Aendern
asstext	EQU \$281A	* Assembler
ausspber	EQU \$1DE0	* Speicherbereiche
bibo	EQU \$27CA	* Bibliothek
edittext	EQU \$1F1E	* Editor
einzel	EQU \$1F24	* Einzelschritt
getadr	EQU \$2E18	* Adresse erfragen
initdebug	EQU \$366E	* Debug initialisieren
ioread	EQU \$2410	* IO lesen
iowrite	EQU \$24C4	* IO setzen
promread	EQU \$253E	* EPROM lesen
promwrite	EQU \$2592	* EPROM programmieren
speicheraus	EQU \$2F20	* Speicherdump

Die Optionen sind aus Gründen der Übersichtlichkeit in einem eigenen Menü zusammengefaßt:

- ```

--- OPTIONEN ---
A = Nur Fehlerausg.
B = Ausgabe auf CRT
C = Ausgabe auf LST
D = wie 3 ohne LF
E = Textstart alt
F = Textstart neu
G = Symbole loeschen
H = 40 Zeichen/Zeile
I = 80 Zeichen/Zeile
J = Debug-Info AN
K = Debug-Info AUS
Z = Grundmenue

```

Bild 2: Optionen

Wer bereits über geänderte Grundprogramme verfügt, muß die entsprechenden Einsprungsadressen anpassen. Diese sind relativ zum Grundprogramm anfang. Das Programm ist daher von der absoluten Lage des Grundprogramms im Speicher unabhängig.

Wer will, kann das Programm in ein Eprom brennen. Dabei sollte aber noch der Bibliotheksvorspann dazugesetzt werden. Ich selber benutze es unter JADOS als externes Kommando.

Viel Spaß beim Programmieren und Arbeiten mit dem Grundprogramm!

```

startex EQU $2D7A * Starten
symbolaus EQU $2F96 * Symboltabelle
syaloesche EQU $30EA * Symbole loeschen

```

-----  
 \* Hier Variablen des Grundprogramms  
 -----

```

akttxt EQU $003A * Startzeile Bildschirm
debug EQU $0220 * Debug-Flag
eottxt EQU $0042 * Ende aller Texte
etxtxt EQU $003E * Textende
groesse EQU $021B * Textgroesse
iostat EQU $002A * IO-Unschalter
stxtxt EQU $0036 * Textstart

```

-----  
 \* Hier Programmstart

```

start:
 bsr seta5 * A5 setzen
 bsr getbasis * Grundprogrammumfang
 movea.l d0,a4 * in A4 speichern
 move.b #2,d5 * IO-Status in D5
grumaus:
 move.b groesse(a5),d6
 move.b #21,groesse(a5) * Textgroesse
 bsr clrscreen * Schirm loeschen
 lea tgrumenu(pc),a0 * Menu ausgeben
 bsr schreibe
 move.b d6,groesse(a5)
grualoop: * Grundmenu-Schleife
 bsr ci * Zeichen lesen
 cmp.b #'a',d0 * Klein- in Grossbuchstaben
 blt.s grA
 cmp.b #'z',d0
 bgt.s grA
 sub.b #32,d0
grA: * Auswahl
 cmp.b #'A',d0
 bne.s grB
 bsr clrscreen
 bsr flipoff
 jsr aendere(a4) * Aendern
 bra grumaus
grB:
 cmp.b #'B',d0
 bne.s grC
 bsr clrscreen
 bsr flipoff
 jsr startex(a4) * Starten
 bra grumaus
grC:
 cmp.b #'C',d0
 bne.s grD
 bsr clrscreen
 bsr flipoff
 jsr speicheraus(a4) * Speicherdump
 bra grumaus
grD:
 cmp.b #'D',d0
 bne.s grE
 bsr clrscreen
 bsr flipoff
 jsr symbolaus(a4) * Symboltabelle
 bra grumaus
grE:
 cmp.b #'E',d0
 bne.s grF
 bsr clrscreen
 bsr flipoff
 jsr edittext(a4) * Editor
 jsr initdebug(a4)
 bra grumaus
grF:
 cmp.b #'F',d0
 bne.s grG
 bsr clrscreen
 bsr flipoff
 jsr initdebug(a4)
 move.b d5,iostat(a5) * Assembler
 jsr asstext(a4)
 bsr crt
 bra grumaus
grG:
 cmp.b #'G',d0
 bne.s grH
 bsr clrscreen
 bsr flipoff
 jsr bibo(a4) * Bibliothek
 bra grumaus
grH:
 cmp.b #'H',d0
 bne.s grI

```

```

 bsr optionen * Optionen
 bra grumaus
grI:
 cmp.b #'I',d0
 bne.s grJ
 bsr clrscreen
 bsr flipoff
 jsr promwrit(a4) * Eprom programmieren
 bra grumaus
grJ:
 cmp.b #'J',d0
 bne.s grK
 bsr clrscreen
 bsr flipoff
 jsr promread(a4) * Eprom lesen
 bra grumaus
grK:
 cmp.b #'K',d0
 bne.s grL
 bsr clrscreen
 bsr flipoff
 jsr ausspber(a4) * Speicherbereiche
 bra grumaus
grL:
 cmp.b #'L',d0
 bne.s grM
 movea.l stxtxt(a5),a0
 bsr lst
 bsr schreibe * Drucken
 bsr crt
 bra grumaus
grM:
 cmp.b #'M',d0
 bne.s grN
 bsr clrscreen
 bsr flipoff
 jsr ioread(a4) * IO lesen
 bra grumaus
grN:
 cmp.b #'N',d0
 bne.s grO
 bsr clrscreen
 bsr flipoff
 jsr iowrite(a4) * IO setzen
 bra grumaus
grO:
 cmp.b #'O',d0
 bne.s grZ
 bsr clrscreen
 lea twarnung(pc),a0
 bsr schreibe
 bsr ci
 cmp.b #'J',d0
 beq.s grO1
 cmp.b #'j',d0
 beq.s grO1
 bra grumaus
grO1:
 bsr clrscreen
 bsr flipoff
 jsr einzel(a4) * Einzelschritt
 bra grumaus
grZ:
 cmp.b #'Z',d0
 beq.s ende * Beenden
 bra grumloop * Nicht implementierter Befehl
ende:
 bsr clrscreen
 move.b #11,groesse(a5) * Rueckkehr
 rts

```

-----  
 \* Unterprogramm OPTIONEN

```

optionen:
 move.b groesse(a5),d6
 move.b #21,groesse(a5) * Textgroesse
 bsr clrscreen * Schirm loeschen
 lea toptionen(pc),a0 * Menu ausgeben
 bsr schreibe
 move.b d6,groesse(a5)
optiloop: * Optionen-Schleife
 bsr ci * Zeichen lesen
 cmp.b #'a',d0 * Klein- in Grossbuchstaben
 blt.s opA
 cmp.b #'z',d0
 bgt.s opA
 sub.b #32,d0
opA: * Auswahl
 cmp.b #'A',d0
 bne.s opB
 move.b #1,d5 * Nur Fehlerausgabe
 bra optionen
opB:

```



```

cmp.b #'B',d0
bne.s opC
move.b #2,d5 * Ausgabe auf CRT
bra optionen
opC:
cmp.b #'C',d0
bne.s opD
move.b #3,d5 * Ausgabe auf LST
bra optionen
opD:
cmp.b #'D',d0
bne.s opE
move.b #4,d5 * wie 3 ohne LF
bra optionen
opE:
cmp.b #'E',d0
bne.s opF
bsr clrscreen * Textstart alt
bsr flipoff
jsr getadr(a4)
tst d1
beq optionen
move.l d0,stxtxt(a5)
move.l d0,akttxt(a5)
move.l d0,a0
opEloop:
tst.b (a0)+
bne.s opEloop
subq.l #1,a0 * Auf Textende zeigen
move.l a0,etxtxt(a5)
move.l a0,eottxt(a5)
bra optionen
opF:
cmp.b #'F',d0
bne.s opG
bsr clrscreen * Textstart neu
bsr flipoff
jsr getadr(a4)
tst d1
beq optionen
move.l d0,stxtxt(a5)
move.l d0,akttxt(a5)
move.l d0,etxtxt(a5)
move.l d0,eottxt(a5)
move.l d0,a0
clr.b (a0)
bra optionen
opG:
cmp.b #'G',d0
bne.s opH
jsr symloesche(a4) * Symbole loeschen
bra optionen
opH:
cmp.b #'H',d0
bne.s opI
move.b #21,groesse(a5) * 40 Zeichen/Zeile
bra optionen
opI:
cmp.b #'I',d0
bne.s opJ
move.b #11,groesse(a5) * 80 Zeichen/Zeile
bra optionen
opJ:
cmp.b #'J',d0
bne.s opK
move.b #1,debug(a5) * Debug-Info AN
jsr initdebug(a4)
bra optionen
opK:
cmp.b #'K',d0
bne.s opZ
move.b #0,debug(a5) * Debug-Info AUS
bra optionen
opZ:
cmp.b #'Z',d0
beq.s opende * Beenden
bra optiloop * Nicht implementierter Befehl
opende:
rts * Rueckkehr

```

```

*-----
* Hier kleine Unterprogramme

```

```

ci:
move.l d7/a6,-(a7)
move #!ci,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

clrscreen:
move.l d7/a6,-(a7)
move #!clrscreen,d7
trap #1
move.l (a7)+,d7/a6

```

```

rts
co2:
move.l d7/a6,-(a7)
move #!co2,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

crt:
move.l d7/a6,-(a7)
move #!crt,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

flipoff:
move.l d0/d1/d7/a6,-(a7)
clr d0
clr d1
move #!setflip,d7
trap #1
move.l (a7)+,d0/d1/d7/a6
rts

```

```

getbasis:
move.l d7/a6,-(a7)
move #!getbasis,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

lst:
move.l d7/a6,-(a7)
move #!lst,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

schreibe: * Startadresse in A0
move.l d0/a0,-(a7)

```

```

schrloop:
move.b (a0)+,d0
beq.s schrende
bsr co2
bra.s schrloop

```

```

schrende:
move.l (a7)+,d0/a0
rts

```

```

seta5:
move.l d7/a6,-(a7)
move #!seta5,d7
trap #1
move.l (a7)+,d7/a6
rts

```

```

*-----
* Hier Texte
*-----

```

```

tgrumenu:
dc.b 13,10,10
dc.b '----- GRUNDPROGRAMM - MENUE -----',13,10
dc.b 13,10
dc.b ' A = Aendern I = EPROM progr. ',13,10
dc.b ' B = Starten J = EPROM lesen ',13,10
dc.b ' C = Ansehen K = Speicherbereiche',13,10
dc.b ' D = Symbolabelle L = Text drucken ',13,10
dc.b ' E = Editor M = IO lesen ',13,10
dc.b ' F = Assembler N = IO setzen ',13,10
dc.b ' G = Bibliothek O = Einzelschritt',13,10
dc.b ' H = Optionen Z = BEENDEN ',13,10,0

```

```

toptionen:
dc.b 13,10,10
dc.b '----- OPTIONEN -----',13,10
dc.b 13,10
dc.b ' A = Nur Fehlerausg. G = Symbole loeschen',13,10
dc.b ' B = Ausgabe auf CRT H = 40 Zeichen/Zeile',13,10
dc.b ' C = Ausgabe auf LST I = 80 Zeichen/Zeile',13,10
dc.b ' D = wie 3 ohne LF J = Debug-Info AN ',13,10
dc.b ' E = Textstart alt K = Debug-Info AUS ',13,10
dc.b ' F = Textstart neu Z = Grundmenue ',13,10,0

```

```

twarnung:
dc.b 'ACHTUNG !! ACHTUNG !! ACHTUNG !! ACHTUNG !!',13,10,10
dc.b 'Nach Einzelschrittbearbeitung erfolgt Neustart ',13,10
dc.b 'in das Original-Grundprogramm',13,10,10
dc.b 'Einzelschritt ausfuehren (J/N) ? ',0

```

# Disketten löschen in 3,5 Sekunden

von Olaf Leinhold, Butterberg, 3300 Braunschweig

Um bisher eine Diskette zu löschen, gab es eigentlich nur zwei Methoden:

1. jeden Bibliothekseingang mit dem ,ERA' Befehl löschen oder
2. die Disk neu formatieren

Beide Verfahren sind jedoch recht umständlich um die Floppy leer zu putzen, ganz zu schweigen von der Zeit, die dazu aufgebracht werden muß. Braucht

der Formatierer noch ca. 2,15 Minuten, steigt die Zeit ins qualvolle, wollte man jeden Eintrag einzeln löschen.

Das unten abgedruckte Programm dient zum Löschen einer bereits formatierten Diskette. Dazu werden lediglich das Betriebssystem, die Spurtabelle sowie das Inhaltsverzeichnis, also nur die ersten 4 Tracks einer Floppy, überschrieben. Somit ist es möglich, eine Diskette innerhalb kürzester Zeit zu löschen: in etwa 3,5 Sekunden!

Zweckmäßigerweise wählt man für das Löschen bzw. das Überschreiben denselben Wert, der auch zum Formatieren frischer Disketten verwendet wird. In unserem Fall ist dies der Wert E5h. Werden die ersten 4 Spuren mit E5h über-

schrieben, so erkennt JADOS diese Diskette als leer und behandelt sie dementsprechend.

Man sollte jedoch äußerst vorsichtig mit dem Aufrufen des Programms sein, denn sollte einmal eine Diskette gelöscht worden sein, sind damit auch sämtliche Dateien und Programme zu 99 Prozent verloren. Ausgenommen, Sie wissen genau auf welcher Spur und Sektor sich die gesuchte Datei aufhält, was jedoch in den seltensten Fällen der Fall ist. Sollten Sie die genaue Stelle doch kennen, so können Sie die Datei noch retten, vorausgesetzt, daß ein Disketten-Monitor nicht weit ist, da nur das System, die Spurtabelle und das Direktory vor dem Überschreiben bedroht sind.

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

```

00C000 *****
00C001 * PROGRAMM ZUM FORMATIEREN VON BEREITS *
00C002 * FORMATIERTEN DISKETTEN. ES WERDEN NUR *
00C003 * DAS BETRIEBSSYSTEM, DIE SPURTABLELLE *
00C004 * UND DAS INHALTSVERZEICHNIS MIT DEM *
00C005 * WERT #E5 ÜBERSCHRIBEN. *
00C006 * *
00C007 * (C) DLAF REINHOLD , 24.11.86 *
00C008 * VERSION 1.0 *
00C009 *****
00C010
00C011 MOVE #1,CLSCREEN,D7
00C012 TRAP #1
00C013 LEA PRGKOFF(PC),A0
00C014 MOVE #7,D7
00C015 TRAP #6 ;PROGRAMMKOPF
00C016 LEA LAUFWERK(PC),A0
00C017 MOVE #7,D7
00C018 TRAP #6 ;AUSWAHLMENÜ
00C019
00C020 AUSWAHL:
00C021 MOVE #1,C1,D7
00C022 TRAP #1 ;AUSWAHLEN
00C023 CMP.B #1,'0'
00C024 BEQ.S AUSWOK ;LAUFWERK 1
00C025 CMP.B #2,'0'
00C026 BEQ.S AUSWOK ;LAUFWERK 2
00C027 CMP.B #9,'0'
00C028 BEQ ERROR ;BEENDEN
00C029 BRA.S AUSWAHL ;ALLE ANDEREN EINGABEN SIND UNGÜLTIG
00C030
00C031 AUSWOK:
00C032 MOVE #1,C02,D7
00C033 TRAP #1 ;NUMMER AUSGEBEN
00C034 SUBI #32,00
00C035 MOVE.B 00,04 ;IN D4 LAUFWERKSCODE
00C036 MOVE #1,CRLF,D7
00C037 TRAP #1
00C038 MOVE #1,CRLF,D7
00C039 TRAP #1
00C040 MOVE #1,GETFLOP,D7
00C041 TRAP #1 ;DICHTER ERMITTELN
00C042 BCS ERROR ;FEHLER, VIELLEICHT UNFORMATIERTE DISKETTE
00C043 CLR D1
00C044 CLR D3
00C045 MOVE #1,FLOPPY,D7
00C046 TRAP #1 ;SCHNELLSTE STEPRATE
00C047 LEA DATA(PC),A0
00C048 MOVE #1024-1,D5
00C049
00C050 DATAES:
00C051 MOVE.B #E5,(A0)+ ;BUFFER MIT WERT #E5 AUFGEFÜLLT
00C052 DBRA.S D5,DATAES ;SCHREIBEN
00C053 MOVE #2,D1 ;BEGINNEN BEI SEKTOR 1
00C054 MOVE.B #0,D3 ;SPUR 0
00C055 MOVE #20-1,D6 ;ERSTEN 20K DER DISK LÖSCHEN MIT #E5
00C056
00C057 LOOP:
00C058 LEA DATA(PC),A0
00C059 MOVE #1,FLOPPY,D7
00C060 TRAP #1
00C061 CMP.B #5,D2 ;BEI SEKTOR 5
00C062 BEQ.S SEKNEU ;NEUE SPUR UND SEKTOR = 1
00C063 ADDG.B #1,D2 ;SEKTOR = SEKTOR + 1
00C064 DBRA.S D6,LOOP

```

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

```

00C098 600A BRA.S FORMDK
00C099
00C09A 5202 ADDG.B #1,D3 ;NEUE SPUR
00C09B 143C 0001 MOVE.B #1,D2 ;SEKTOR BEI 1 BEGINNEN
00C09C 51CE FFEC DBRA.S D6,LOOP

```

```

00C0A4 41FA 0119 FORMDK: LEA TEXT(PC),A0
00C0A5 3E3C 0907 MOVE #7,D7
00C0A6 4E46 TRAP #6 ;Diskette ist fertig formatiert.
00C0A7 4E75 RTS
00C0B0
00C0B1 ERROR:
00C0B2 3E3C 0063 MOVE #1,CRLF,D7
00C0B3 4E41 TRAP #1 ;BEENDEN WENN '9' GEWÄHLT ODER FEHLER
00C0B4 4E75 RTS
00C0B5
00C0B6 PRGKUFF:
00C0B7 202020202020 DC.B '-----',#D,#A
00C0B8 202020202020
00C0B9 202020202020
00C0C0 202020202020
00C0C1 2A2A2A2A2A2A
00C0C2 202020202020
00C0C3 202020202020
00C0C4 2A2A2A2A2A2A
00C0C5 202020202020
00C0C6 2000 0A
00C0C7 5367686E6E6C6C DC.B 'Schnellformatierer für erneutes Formatieren der Disketten',#D,#A
00C0C8 666F726B617469
00C0C9 657265722066F0
00C0CA 722065726E6575
00C0CB 74657202466F72
00C0CC 6D6174659657265
00C0CD 6E206465722044
00C0CE 69736865747465
00C0CF 6E00 0A
00C0D0 284329204FAC61 DC.B '(C) Olaf Reinhold , 24.11.86',#D,#A
00C0D1 66205265696E68
00C0D2 6F4C6420202020
00C0D3 2032342E31312E
00C0D4 383600 0A
00C0D5 202020202020 DC.B '-----',#D,#A,0
00C0D6 202020202020
00C0D7 202020202020
00C0D8 202020202020
00C0D9 2A2A2A2A2A2A
00C0DA 202020202020
00C0DB 202020202020
00C0DC 202020202020
00C0DD 2000 0A 00
00C0DE
00C0DF LAUFWERK: DC.B #0,#A,#A
00C0E0 3120204C6175 DC.B '1 = Laufwerk 1',#D,#A
00C0E1 667765726E2031
00C0E2 00 0A
00C0E3 3220204C6175 DC.B '2 = Laufwerk 2',#D,#A
00C0E4 667765726E2032
00C0E5 00 0A
00C0E6 392020426565 DC.B '9 = Beenden ',0
00C0E7 6E64656E202020
00C0E8 20202000
00C0E9
00C0EA 44977268657474 TEXT: DC.B 'Diskette ist fertig formatiert.',#D,#A,0
00C0EB 65206973742066

```

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

```

00C1C0 65727469672066
00C1C1 6F726B61746965
00C1C2 72742E00 0A 00
00C1C3 00 DS 0
00C1C4
00C1C5 DATA: DS.B 1024 ;BUFFER FÜR FORMATIERWERT #E5
00C1C6
00B014 Ende-Symboltabelle

```

# CP/M 68 K, C UND MODULA

## Antialiasing Linienroutine in MODULA2

von Willi Sicking und Dieter Blommel

In der „LOOP“ Nr. 7 hatte Rolf-Dieter Klein ein Linienprogramm mit einem Antialiasingeffekt vorgestellt. Mit diesem Verfahren ist es möglich, Linien ohne störende Treppenstufen auf der COL256 darzustellen. Für den realen Einsatz war das Programm jedoch nicht gedacht, da es, wie auch der Originalalgorithmus, nur im Bereich von 0 bis 45 Grad funktioniert.

Im folgenden ist nun ein funktionsfähiges Modul für Modula2 abgebildet. Für Monitore, bei denen eine Gammaentzerrung nötig ist, kann diese durch eine Tabelle oder wie in ANTI1 durch eine Sinusfunk-

tion von 0 bis 90 Grad erreicht werden. Die Multiplikation mit dieser Funktion bewirkt eine Anhebung der dunklen Bildpunkte. Der Parameter Color gibt bei Schwarz-Weiss-Monitoren die maximale Helligkeit der Linie an. Farbdarstellung ist wohl erst mit der CLUT möglich, die ja bald lieferbar sein soll. Dann kann die Gammaentzerrung mit der Farbtabelle erfolgen. Das Programm wurde in der Gleitkommaform belassen, da auch Änderungen, die kleiner als ein Koordinatenpunkt sind, zu einer Linienveränderung führen. Da der Koprozessor immer mit 32 Bit Genauigkeit arbeitet, können die Parameter für die Unterprogramme in der Mathlib zur Zeit nur vom Typ LONG-FLOAT übergeben werden.

Die Punktsetzroutine wird in einem C-Programm mit setfdot(col,x,y) aufgerufen. Unter Modula2 ist die Parameterübergabe vertauscht, so daß der Aufruf hier setfdot(y,x,col) lauten muß.

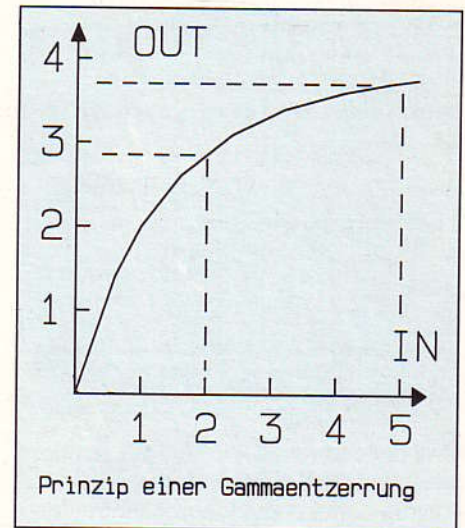


Bild 1)  
Prinzipieller Verlauf einer Gammaentzerrung

```
G>type test.mod
MODULE Test;

FROM Aline IMPORT ALine;
FROM SimpleIO IMPORT ReadChar;
FROM col256 IMPORT colinit;

VAR m : INTEGER;
BEGIN
 colinit;
 FOR m:= 0 TO 255 BY 10 DO
 ALine (128.0,128.0,FLOAT(m),0.0,100.0);
 ALine (128.0,128.0,FLOAT(m),255.0,150.0);
 ALine (128.0,128.0,0.0,FLOAT(m),200.0);
 ALine (128.0,128.0,255.0,FLOAT(m),250.0);
 END;
END Test.
```

```
G>type anline.def
DEFINITION MODULE Aline;

FROM col256 IMPORT setfdot;
FROM MathLib IMPORT Sin,Arctan;

EXPORT QUALIFIED ALine;

PROCEDURE ALine (xs,ys,xe,ye,color: REAL);

END Aline.
```

```
G>type anline.mod
IMPLEMENTATION MODULE Aline;

(* ANTIALIASING-LINE *)
(* (C) Dieter Blommel *)
(* u. Willi Sicking *)
(* 6.1.1987 *)

FROM col256 IMPORT setfdot,colinit;
FROM MathLib IMPORT Sin,Arctan;

CONST pi = 3.141592654;
 pi2= 1.57;

PROCEDURE Anti1(xs,ys,xe,ye,color :REAL);

 VAR ix1,iy1 :CARDINAL;
 c2,cp,c,a,q :REAL;
```

```
BEGIN
 c := color;
 ix1 := TRUNC(xs+0.5);
 iy1 := TRUNC(ys+0.5);
 q := (ye-ys)/(xe-xs);
 a := q*(FLOAT(ix1)-xs)+ys;
 IF a < FLOAT(iy1) THEN
 DEC(iy1);
 ELSE
 cp := q*c;
 c2 := (a-FLOAT(iy1))*c;
 END;
 REPEAT
 (* Demo fuer Gammaentzerrung *)
 setfdot(iy1,ix1,TRUNC(color*FLOAT
 (Sin(LONGFLOAT((c-c2)/color*pi2)))));
 setfdot(iy1+1,ix1,TRUNC(color*FLOAT
 (Sin(LONGFLOAT((c2)/color*pi2)))));
 c2 := c2+cp;
 IF c2 >= c THEN
 INC (iy1);
 c2 := c2-c;
 END;
 INC (ix1);
 UNTIL FLOAT(ix1) >= xe;
END Anti1;
```

```
PROCEDURE Anti2(ys,xs,ye,xe,color :REAL);
```

```
VAR ix1,iy1 :CARDINAL;
 c2,cp,c,a,q :REAL;

BEGIN
 c := color;
 ix1 := TRUNC(xs+0.5);
 iy1 := TRUNC(ys+0.5);
 q := (ye-ys)/(xe-xs);
 a := q*(FLOAT(ix1)-xs)+ys;
 IF a < FLOAT(iy1) THEN
 DEC(iy1);
 ELSE
 cp := q*c;
 c2 := (a-FLOAT(iy1))*c;
 END;
 REPEAT
 setfdot (ix1,iy1,TRUNC(c-c2));
 setfdot (ix1,iy1+1,TRUNC(c2));
 c2 := c2+cp;
 IF c2 >= c THEN
```

```

 INC (iy1);
 c2 := c2-c;
 END;
 INC (ix1);
 UNTIL FLOAT(ix1) >= xe;
END Anti2;

PROCEDURE Anti3(xs,ys,xe,ye,color :REAL);

VAR ix1,iy1 :CARDINAL;
 c2,cp,c,a,q :REAL;

BEGIN
 c := color;
 ix1 := TRUNC(xs+0.5);
 iy1 := TRUNC(ys+0.5);
 q := (ye-ys)/(xs-xe);
 a := q*(xs-FLOAT(ix1))+ys;
 IF a < FLOAT(iy1) THEN
 DEC(iy1);
 ELSE
 cp := q*c;
 c2 := (a-FLOAT(iy1))*c;
 END;
 REPEAT
 setfdot (iy1,ix1,TRUNC(c-c2));
 setfdot (iy1+1,ix1,TRUNC(c2));
 c2 := c2+cp;
 IF c2 >= c THEN
 INC (iy1);
 c2 := c2-c;
 END;
 DEC (ix1);
 UNTIL FLOAT(ix1) <= xe;
END Anti3;

PROCEDURE Anti4(ys,xs,ye,xe,color :REAL);

VAR ix1,iy1 :CARDINAL;
 c2,cp,c,a,q :REAL;

BEGIN
 c := color;
 ix1 := TRUNC(xs+0.5);
 iy1 := TRUNC(ys+0.5);
 q := (ye-ys)/(xs-xe);
 a := q*(xs-FLOAT(ix1))+ys;
 IF a < FLOAT(iy1) THEN
 DEC(iy1);
 ELSE
 cp := q*c;
 c2 := (a-FLOAT(iy1))*c;
 END;
 REPEAT
 setfdot (ix1,iy1,TRUNC(c-c2));
 setfdot (ix1,iy1+1,TRUNC(c2));
 c2 := c2+cp;
 IF c2 >= c THEN
 INC (iy1);
 c2 := c2-c;
 END;
 END;
END;

```

```

 DEC (ix1);
 UNTIL FLOAT(ix1) <= xe;
END Anti4;

PROCEDURE absWinkel (k1x,k1y,k2x,k2y :REAL):REAL;

VAR deltax,deltay :REAL;
 winkel :REAL;

BEGIN
 deltax := k2x - k1x;
 deltax := k2y - k1y;
 IF deltax = 0.0 THEN
 IF deltax >= 0.0 THEN
 RETURN (pi/2.0)
 END;
 RETURN (-pi/2.0)
 END;
 winkel := FLOAT(Arctan(LONGFLOAT
 (ABS(deltay/deltax))));
 IF (deltax < 0.0) AND (deltay < 0.0) THEN
 RETURN (-pi + winkel)
 ELSIF (deltax < 0.0) AND (deltay >= 0.0) THEN
 RETURN (pi - winkel)
 ELSIF (deltax >= 0.0) AND (deltay >= 0.0) THEN
 RETURN (winkel)
 ELSE
 RETURN (-winkel)
 END
END absWinkel;

PROCEDURE ALine (xs,ys,xe,ye,color :REAL);

VAR Winkel :REAL;

BEGIN
 Winkel := absWinkel (xs,ys,xe,ye);
 IF Winkel < (-3.0/4.0*pi) THEN
 Anti1(xe,ye,xs,ys,color)
 ELSIF Winkel < (-pi/2.0) THEN
 Anti2(xe,ye,xs,ys,color)
 ELSIF Winkel < (-pi/4.0) THEN
 Anti4(xs,ys,xe,ye,color)
 ELSIF Winkel < 0.0 THEN
 Anti3(xe,ye,xs,ys,color)
 ELSIF Winkel < (pi/4.0) THEN
 Anti1(xs,ys,xe,ye,color)
 ELSIF Winkel < (pi/2.0) THEN
 Anti2(xs,ys,xe,ye,color)
 ELSIF Winkel < (3.0/4.0*pi) THEN
 Anti4(xe,ye,xs,ys,color)
 ELSE
 Anti3(xs,ys,xe,ye,color)
 END
END ALine
END Aline.

```

Bild 2)

Testprogramm, Definitionmodule, Implementationsmodule

## ⊗ Tips für NKC-User

von Uwe Koch, Lüdenscheid

Heute habe ich ein paar kleine Tips für alle NKC-User, die demnächst eine Hardcopy/Maus-Karte bauen wollen, aber keinen Epson(-kompatiblen)-Drucker besitzen, sondern einen grafikfähigen billigeren Matrixdrucker. Ich selbst habe den GP-550A von Seikosha. Dieser, wie auch viele andere, einfache Matrixdrucker kann zwar Grafiken erzeugen, jedoch sind die Steuerbefehle nicht die gleichen wie die, die im Handbuch der Karte ver-

wendet werden. Um die Programme an andere Drucker anpassen zu können, muß man allerdings erst einmal wissen, was die Epson-Befehle bewirken. So setzt der Befehl ,ESC C' (1Bh,,C' bzw. \$1B,,C') den Drucker in den Normalzustand wie nach dem Einschalten. Dieser Befehl kann bei anderen Druckern auch entfallen oder man kann beim GP-550A z. B. mit ,ESC Z072' (1Bh,,Z072' bzw. \$1B,,Z072') dem Gerät die Papierseitenlänge mitteilen (nach dem Einschalten sind es 66 Zeilen).

Der nächste Epson-Befehl ist ,ESC 3 24' (1Bh,,3',24 bzw. \$1B,,3',24). Hiermit wird

der Zeilenabstand auf 24/216 Zoll eingestellt, um eine kontinuierliche Grafik zu erzeugen. Bei anderen Druckern gibt es entsprechende Befehle, die im Handbuch speziell für lückenlose Grafik empfohlen werden (beim GP-550A ist es ,ESC 9'). Schließlich muß am Beginn jeder Zeile dem Drucker mitgeteilt werden, daß keine ASCII-Zeichen, sondern Grafikzeichen folgen. Beim Epson macht dies der Befehl ,ESC K n m'. Dabei bedeutet das ,K' Grafik mit einfacher Dichte, und die beiden Zahlen n und m geben an, wieviele Graphikdaten folgen (m\*256 + n). Beim GP-550A und wahrscheinlich

auch einigen anderen einfachen Druckern gibt es nur einen 8-Bit-Graphik-Modus. Dieser entspricht der doppelten Dichte beim Epson. Daher müssen die Graphikprogramme hier entsprechend abgeändert werden. Dazu müssen alle Graphikdaten doppelt ausgegeben werden, um einfache Dichte zu imitieren. Es muß also der Graphikmode für 512 Daten gewählt werden (GP-550A: ,ESC G512'). Dann muß auch die Datenausgabe zweimal erfolgen, d. h. in der Routine PrtLine muß die Druckerausgabe (JSR @LO, bzw. CALL 0F00Fh, bzw. CALL OUT) zweimal direkt hintereinander aufgerufen werden. Mit diesen Anpassungen wird man allerdings immer noch keine formatgetreuen Hardcopies erzeugen können, da einerseits die Drucknadeln bei einfachen Druckern weiter auseinanderstehen als bei den teureren, und damit das Druckbild breiter ist, während die Höhe durch die Druckdichte gewählt werden kann. Andererseits ist das Monitorbild bekanntlich aus 512 x 256 Bildpunkten aufgebaut. Ein Quadrat ist also z. B. 200 Punkte breit und 100 Punkte hoch, da die Punkte in X-Richtung doppelt so dicht liegen wie in Y-Richtung. Auf dem Drucker läßt sich aber ein Dichteverhältnis von 1:2 wahrscheinlich nicht einstellen, so daß Quadrate zu Rechtecken und

Kreise zu Ellipsen werden (siehe Bilder). Bei einigen Druckern, so auch beim GP-550A, taucht außerdem noch ein weiteres Problem auf. Diese Geräte erzeugen mit den so abgeänderten Programmen nämlich zwar ein akzeptables Bildformat, aber leider kein vernünftiges Bild. Das liegt daran, daß beim Epson das Bit 7 (MSB) den obersten Punkt des Graphik-elementes darstellt und Bit 0 (LSB) den untersten Punkt. Bei einigen Druckern ist dies jedoch genau umgekehrt, so daß jede Zeile der Hardcopy in sich auf dem Kopf steht. Um dieses zu ändern, muß jedes einzelne Byte umgedreht werden. So wird z. B. aus 10110001b (= B1h) das Byte 10001101b (= 8Dh). Für dieses Umdrehen kann man das folgende Unterprogramm ,WENDEN' benutzen.

Beide Programme drehen das Byte im C-Register bzw. D0-Register und müssen daher direkt vor der Druckerausgabe in der PrtLine-Routine mit CALL WENDEN bzw. BSR WENDEN aufgerufen werden.

Die Bilder zeigen die verschiedenen Formate des Druckes.

Zum Schluß noch ein Tip für den Monitoranschluß an die Hardcopy-Karte. Wenn man auf die Lötseite der Karte direkt auf den Winkelstecker für die Ver-

```
Für 260 :

Wenden: ld b,b ; 8 Durchläufe für 8 Bits
 rr c ; letztes Bit des Originalbytes
 rl a ; ins Carry schieben
 dec b ; Bit aus Carry als erstes in
 jp nz,wend ; Akku A schieben
 ld c,a ; Zähler dekrementieren
 dec b ; noch nicht 0, dann nochmal
 ld c,a ; fertig, dann gewendetes Byte
 ret ; wieder nach C
 und zurückspringen
```

```
Für 68000/68008 :

Wenden: movem d1-d2,-(a7) ; Register auf Stack
 clr d1 ; D1 löschen
 move.b #8,d2 ; D2 als Zähler auf 8
wendel: roxl.b #1,d0 ; letztes Bit ins X- Bit
 roxr.b #1,d1 ; X-Bit als erstes in D1
 subq #1,d2 ; Zähler dekrementieren
 bne wendel ; noch nicht 0, nochmal
 move.b d1,d0 ; gewendetes Byte nach D0
 movem (a7)+,d1-d2 ; Register restaurieren
 rts ; fertig, zurück
```

bindung zur GDP64k einen zweiten Winkelstecker mit 2 x 7 Polen lötet, kann man den Monitoranschluß sehr einfach entweder in die innere Reihe (ohne Fadenkreuz) oder die äußere Reihe (mit Fadenkreuz) stecken. Die Verbindung zur GDP läßt sich sehr einfach mit einem Stückchen 16poligem Flachkabel und zwei 2 x 8-poligen Quetschbuchsen herstellen. Auf die GDP wird dann nur die innere Kontaktreihe gesteckt. Beim Betrieb mit Fadenkreuz muß man den Monitor wahrscheinlich heller einstellen, um ein Bild zu erhalten.

```

 bsbq.b #1,d5 ; alle Punkte abgetastet ?
 bne Loop5 ; nein, dann nochmal

 bsbq.b #1,d4 ; alle Spalten abgetastet ?
 bne Loop4 ; nein, dann nochmal

 * Ausgabe des Zeilenpuffers

PrtLine: lea Buffer,a0 ; Adresse des Zeilenpuffers
Loop3: move.w #256,d1 ; Zahl der Speicherstellen
 move.b (a0)+,d0 ; Speicherstelle laden
 not.b d0 ; Byte invertieren
 bsr wenden ; Byte umdrehen
 jsr @10 ; Byte ausgeben
 jsr @10 ; / zweimal wegen Format
 subq.w #1,d1 ; alle Bytes ausgeben
 bne Loop3 ; noch nicht, dann nochmal
 move.b #$0D,d0 ; <CR> ins Register
 jsr @10 ; und ausgeben
 move.b #$0A,d0 ; <LF> ins Register
 jsr @10 ; und ausgeben
 rts

```

Bild 1: Programm ohne Doppelausgabe mit WENDEN

Textstart=019000 Fenster=01A2E9 Tor=01AAB5 amr Ctrl-J = Rechner

```

 subq.w #1,d5 ; alle Punkte abgetastet ?
 bne Loop5 ; nein, dann nochmal

 subq.b #1,d4 ; alle Spalten abgetastet ?
 bne Loop4 ; nein, dann nochmal

 * Ausgabe des Zeilenpuffers

PrtLine: lea Buffer,a0 ; Adresse des Zeilenpuffers
Loop3: move.w #256,d1 ; Zahl der Speicherstellen
 move.b (a0)+,d0 ; Speicherstelle laden
 not.b d0 ; Byte invertieren
 bsr wenden ; Byte umdrehen
 jsr @10 ; Byte ausgeben
 jsr @10 ; / zweimal wegen Format
 subq.w #1,d1 ; alle Bytes ausgeben
 bne Loop3 ; noch nicht, dann nochmal
 move.b #$0D,d0 ; <CR> ins Register
 jsr @10 ; und ausgeben
 move.b #$0A,d0 ; <LF> ins Register
 jsr @10 ; und ausgeben
 rts

```

Bild 2: Doppelausgabe der Grafikbytes mit WENDEN

Textstart=019000 Fenster=01A827 Tor=01A9E3 deut Ctrl-J = Rechner

```

bne Loop5 * nein, dann nochmal

subq.b #1,d4 * alle Spalten abgetastet ?
bne Loop4 * nein, dann nochmal
rts

 * Ausgabe des Zeilenpuffers

PrtLine: lea Buffer,a0 * Adresse des Zeilenpuffers
 move.w #256,d1 * Zahl der Speicherstellen
Loop3: move.b (a0)+,d0 * Speicherstelle laden
 not.b d0 * Byte invertieren
 bsr wenden * / Byte umdrehen für GP-550A
 jsr @10 * Byte ausgeben
 jsr @10 * / dreimal wegen
 jsr @10 * / doppelter Dichte
 subq.w #1,d1 * alle Bytes ausgeben
 bne Loop3 * nochmal, dann nochmal
 move.b #0D,d0 * <CR> ins Register
 jsr @10 * und ausgeben
 move.b #0A,d0 * <LF> ins Register
 jsr @10 * und ausgeben
 rts

```

```

 jsr @10 * / Output an display
 rts * alle abtastet

 jsr @10 * / Output an display
 rts * alle abtastet

 * Output an display
 jsr @10 * / Output an display
 rts * alle abtastet

```

Textstart=009000 Fenster=00A86B Tor=00A827 deut Ctrl-J = Rechner

Textstart=009100 Fenster=00A86B Tor=00A82E deut Ctrl-J = Rechner

Bild 3: Dreifachausgabe der Grafikbytes mit WENDEN

Bild 4: Dreifachausgabe ohne WENDEN

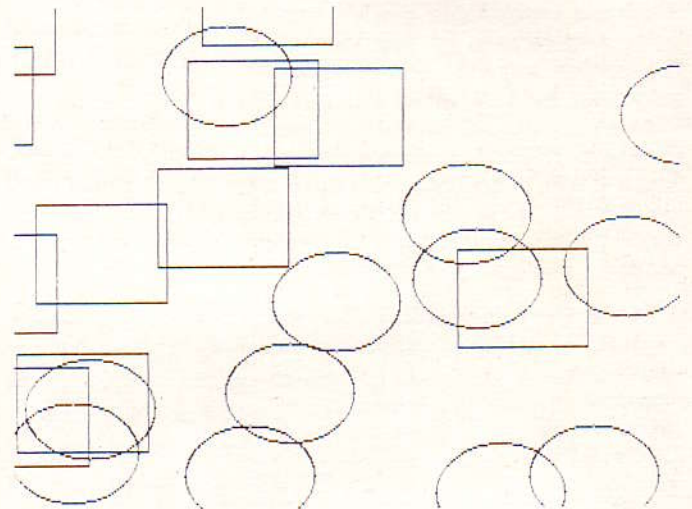
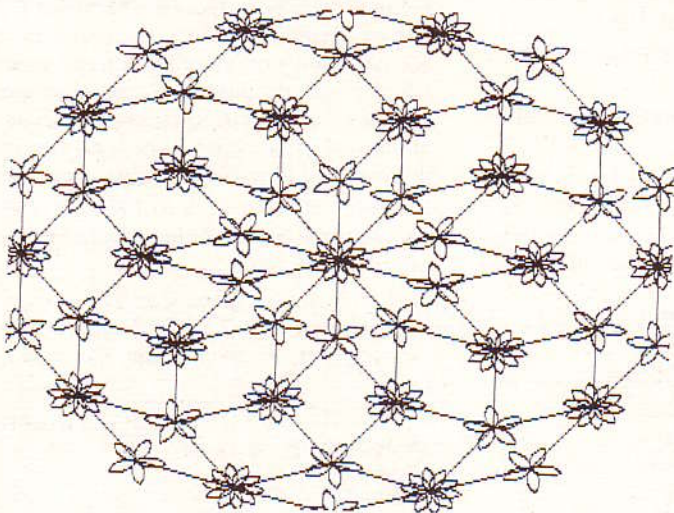


Bild 5: Blumen-Demo aus Schritt für Schritt 2

Bild 6: Quadrate und Kreise in RL-Basic

# Neue Version des Z80 Emulators für CP/M-68k

Seit drei Jahren ist nun der CP/M und Z80 Emulator der Firma Softdesign in München erhältlich. In der Zwischenzeit wurde der Emulator auch auf den Betriebssystemen ATARI TOS und OS/9-68K implementiert. Durch die ständige Weiterentwicklung des Produkts kann auch die neueste Version für CP/M-68K mit einigen erfreulichen Verbesserungen aufwarten.

Bevor jedoch Einzelheiten der neuen Version besprochen werden, kurz noch einmal das wichtigste über Zweck und Funktion des CP/M-Z80 Emulators.

## Der CP/M-Z80 Emulator

Trotz der Fülle an neuen Systemen, die auf dem leistungsfähigen 68000 Prozessor basieren, läßt die Entwicklung auf dem Softwaremarkt noch so manche Wünsche offen. Ein ganzes Jahrzehnt

war notwendig, bis sich eine so umfassende Software-Bibliothek wie die für CP/M-Z80 entwickeln konnte. All diese Programme sind nur auf den Z80- und 8080/8085-Mikroprozessoren lauffähig. Viele Besitzer von Z80-Rechnern würden gern von ihrem alten System zur neuen 68000-Klasse aufsteigen, möchten aber nicht ihre komplette Softwaresammlung wegwerfen. Aber auch mancher Neueinsteiger, der gleich mit einem 68000-System angefangen hat, möchte gern wissen, was es mit CP/M-80 und so berühmten Programmen wie TurboPascal, Multiplan und dBase II eigentlich auf sich hat.

Für all jene hat die Firma Softdesign in München den CPMZ80 Emulator entwickelt. Rechner mit den Betriebssystemen CP/M-68K, ATARI TOS oder OS-9/68K können mit Hilfe des Emulators ohne

Hardware-Zusatz in einen Z80-Rechner mit CP/M-80 verwandelt werden. Weitere Vorteile, die sich durch die Benutzung des Emulators ergeben sind:

- Der Übergang von der Z80-Generation zur neuen 68000-Generation wird leicht gemacht, auch auf einem neuen Computer kann mit der gewohnten Software-Umgebung gearbeitet werden.
- Für 68000-Rechner noch nicht verfügbare Software kann durch Zugriff auf die CP/M-80-Programm-Bibliothek ersetzt werden.
- Selbst entwickelte CP/M-80-Programme können ohne Änderung auf 68000-Systeme übernommen werden.
- Ein 68000-Computer wird zum Entwicklungssystem für Z80 Software. Im Gegensatz zu sogenannten Cross-Assembler/linker-Softwarepaketen kann

man selbst entwickelte Z80-Programme unter dem Emulator auch ausführen und debuggen.

### Was ist Emulation?

Die Grundidee dazu geht zurück bis auf Alan M. Turing. Dieser englische Mathematiker bewies 1947, daß man mit Hilfe eines geeigneten Programms jeden Computer dazu bringen kann, sich so zu verhalten wie ein beliebig anderer. Läuft auf einem Computer ein Programm, das einen anderen Computer nachbildet, nennt man diesen Simulationsvorgang Emulation. Seit vielen Jahren werden neue Rechner schon in der Entwurfsphase auf Großrechnern durch Software emuliert, noch bevor sie wirklich gebaut werden.

Schwierig ist die tatsächliche Implementierung eines solchen Emulators, wenn eine realistische Geschwindigkeit des emulierten Computers gefordert wird. Ein Emulator arbeitet nämlich ähnlich wie ein Interpreter, d.h., jeder emulierte Maschinenbefehl wird durch eine eigene Befehlssequenz nachgebildet. Das Lesen eines Z80-Maschinenbefehls und das Verzweigen auf die zugehörige 68000-Routine steht im Mittelpunkt des Emulationsvorgangs und ist ein zeitkritischer Teil der Emulation. Trotz der hohen Rechenleistung des 68000-Prozessors wurde die Emulation des Z80-Prozessors, bei gleichzeitig realistischem Zeitverhalten, erst durch sehr aufwendige, zeitoptimierte Assemblerprogrammierung möglich.

Das Zeitverhalten der Emulation ist natürlich stark von dem Z80-Programm, das emuliert werden soll, abhängig. Enthält das Programm viele Ein-/Ausgabe-Sy-

stemaufrufe, ist weniger die Geschwindigkeit der Emulation, als die Geschwindigkeit des Betriebssystems und der Hardware (Floppy, Harddisk, Konsole) ausschlaggebend. Bei solchen Programmen (Assembler, Compiler, Linker) kann der CPMZ80 Emulator im Extremfall Z80-Software auf modernen 68000-Systemen sogar schneller als auf Z80-Hardware ausführen. Auf der anderen Seite können sehr rechenintensive Programme durch den Emulator bis zum Faktor 5 verlangsamt werden. Im Mittel, über sehr viele Programme, die getestet wurden, hat sich eine Geschwindigkeit entsprechend einer 2 MHz Z80 CPU ergeben.

### Die neue Version des Emulators

Doch nun zu den Eigenschaften der aktuellen Version des CPMZ80 Emulators. Die Verbesserungen fallen in zwei verschiedene Bereiche:

#### 1. Die Emulation der Z80 CPU

Nach nunmehr dreijähriger Entwicklungszeit konnte durch eine Überarbeitung der Z80-Emulationsroutinen die Geschwindigkeit des Emulators nochmals um bis zu 20% gesteigert werden. Erzielt wurde diese Geschwindigkeitssteigerung hauptsächlich durch eine neue Z80-Opcode-Fetch-Routine.

Der Emulator erkennt nun automatisch, ob er auf einer 68000/68008/68010 oder auf einer 68020 CPU läuft und nutzt optimal die spezifischen Eigenschaften der jeweiligen CPU.

#### 2. Das integrierte CP/M-80 kompatible Betriebssystem

Das große Ärgernis beim Umgang mit CP/M-80 wurde durch die neue automati-

sche Disk-Change-Erkennung beseitigt – der Zwang zu „Control-C“ bei Diskettenwechsel und der dazugehörige „BDOS error read/only“ gehören der Vergangenheit an.

Wie unter CP/M-68K kann jetzt bei Fehlern in Disk-Schreib-/Lese-Operationen der Zugriff wahlweise wiederholt, ignoriert oder abgebrochen werden.

Durch konsequente Weiterentwicklung ist auch der Betriebssystemkern schneller geworden.

Geschachtelte „Submit“-Prozesse auf beliebigen Laufwerken werden nun unterstützt. Submit-Dateien können zur besseren Verstehbarkeit mit Kommentaren versehen werden.

Ein eingebautes Directory-Kommando, mit sortierter Ausgabe und ausführlichen Angaben über Dateigrößen, die Anzahl der Dateien und den noch verfügbaren Speicherplatz, erübrigt die Verwendung des START-Kommandos.

Ein eingebautes COPY-Kommando ermöglicht das Kopieren von Dateien, genau wie mit PIP. Dadurch entfällt die Notwendigkeit, zum Kopieren immer erst die PIP.COM-Datei suchen und laden zu müssen.

Insgesamt ist die Emulation eines CP/M- und Z80-Systems in Software schneller und komfortabler geworden. Die Benutzung eines „Soft-Z80“ stellt eine echte Alternative zu Z80-Hardware-Systemen dar.

**Hinweis: Der neue Emulator ist entschieden billiger geworden!**

| Bestell-Nr. |                   | DM    |
|-------------|-------------------|-------|
| 10609       | Z80-Emulator-Disc | 298,- |

## Gleichzeitiger Betrieb von Z 80 und 68000/8 auf einem Bus

Leider lief bei mir das mit zwei Prozessor-karten (Z80 und 68008) auf einem Bus nach Herrn Bäckers Schaltungsvorschlag aufgebaute System nicht. Herr Bäcker bestätigte mir am Telefon, daß es bei ihm zwar laufe, er habe schon von vielen anderen Leuten mit gleichen Problemen gehört. Für die Leute, die den Versuch noch wagen wollen bzw. bei denen keine Probleme auftreten, kurz eine Beschreibung der Schwierigkeiten:

Im ‚Z80-Modus‘ (Z80 an/68008 aus) meldete sich zwar der Monitor, jedoch streikte die Floppy beim Laden des Systems. Beim Umschalten auf den ‚68008-Modus‘ gab es einen regelrechten Systemzusammenbruch: nichts lief mehr. Offen-

bar sind gewisse Steuersignale trotz Abschalten der Betriebsspannung auf den jeweils anderen Karten und trotz Tri-state- bzw. open-collector-Treibern vom Bus nicht richtig getrennt und beeinflussen dadurch die jeweils in Betrieb befindlichen Karten. Nach viel Herumprobieren (ich habe nicht mal ein Skop) habe ich jetzt eine Lösung gefunden, die jedenfalls bei mir zu einem einwandfreien Ergebnis geführt hat:

Die RESET-Leitung beim Z80 (Pin26) muß mit einem 1K Widerstand auf 5 V gezogen werden; dadurch wird der Floppy-Betrieb unter Z80 nicht mehr gestört. Um den 68008 bei der Arbeit nicht zu stören, habe ich den Widerstand R11 (4,7k) an Pin 33 des 68008 (BR quer) gegen einen von 1K ersetzt. Damit lief's!

Übrigens müssen nicht unbedingt zwei Reset-Schalter vorhanden sein, wenn man über einen Umschalter (der auch in den 5V- und INT-Umschalter nach Schaltung Bäcker integriert sein kann) das Si-

gnal von Masse zum jeweiligen RESET-Eingang der jeweiligen Prozessorkarte führt. Dies ist bei beiden Karten auf der Bestückungsseite jeweils der linke Abschluß.

Ich glaube, daß mit dieser Lösung vielen Mitcomputeristen, die dieselben Probleme haben wie ich sie hatte, geholfen werden kann und schlage deshalb eine Veröffentlichung in der nächsten LOOP vor.

### Impressum:

LOOP, Zeitung für Computerbauer

Herausgeber: Gerd Graf

Redaktion: Rolf-Dieter Klein, Gerd Graf

Gesamtherstellung:

Buch- und Offsetdruck A. Rieder, Kempten

Herstellung und Anzeigenverwaltung:

GES GmbH

Magnusstraße 13, 8960 Kempten

# Der mc-CP/M-COMPUTER

## EPF2.

**Die Eprom-Floppy für den mc-/CP/M-Computer – endlich technisch frei und lieferbar!**

Wozu dient die Eprom-Floppy? Selbst bei Verwendung einer RAM-Floppy müssen die Programme und Daten zunächst einmal von Floppy-Disk geladen werden. Auch jeder Warmstart erfordert einen neuen Floppy-Zugriff, mit entsprechenden Wartezeiten. Daher liegt es nahe, oft benötigte Programme, wie Editor, Compiler, Assembler und Utilities in Form von Firm-Software bereitzuhalten. Hinzu kommt, daß Eproms heute in Kapazitäts- und Preisregionen vorgerückt sind, die einen Einsatz mittels Eprom-Floppy sinnvoll erscheinen lassen.

Hier sind besonders die neuen, billigen Einmal-Eproms zu erwähnen, die keine Löschenfenster mehr besitzen. Besonders in Verbindung mit einer RAM-Floppy und

unter Ausnutzung des DMA-Controllers ist ein Arbeiten fast ohne IO-Wartezeiten möglich.

So wird zum Beispiel der Macro-Assembler M80 in 70 ms in den Arbeitsspeicher geladen und eine Kopie der gesamten 320K-Eprom-Floppy in die RAM-Floppy erfolgt in ca. 7 sec. Die hier vorgestellte ECB-Karte ist in erster Linie für den mc-/CP-Computer gedacht, kann aber auch in andere ECB-Systeme eingesetzt werden.

Falls nur Festprogramme geplant sind oder der Rechner als Workstation im Rechnernetz eingesetzt wird, läßt sich mit Hilfe der Eprom-Floppy ein Computer-System ohne Floppy-Laufwerke realisieren. In den meisten Fällen empfiehlt es sich, wenigstens ein Floppy-Laufwerk anzuschließen, um das zu bearbeitende Programm in die RAM-Floppy zu laden. Anschließend erfolgt die Bearbeitung mit Hilfe der in der Eprom-Floppy

gespeicherten Dienstprogramme, unter voller Ausnutzung der Transferrgeschwindigkeit.

Hierbei wäre aber noch ein wichtiger Hinweis anzubringen: CP/M2.2 darf nur von lizenzierten Benutzern auf Eproms gebrannt werden.

| Bestell-Nr. | Bestell-Bez. incl. MWSt | DM    |
|-------------|-------------------------|-------|
| 10649       | EPF-Bausatz             | 150,- |
| 10648       | EPF-Fertiggerät         | 230,- |
| 10651       | EPF-Handbuch            | 10,-  |
| 10650       | EPF-Platine             | 50,-  |

## Hardcopy-Tip

Das Programm HCOPYLD.COM ist auf EPSON-Drucker abgestimmt. Besitzer des STAR-Druckers SG 10 erhalten dann einen einwandfreien Hardcopy-Ausdruck, wenn der IBM-Befehlssatz mit DIP-Schalter eingeschaltet wird (Schalter 2-2).

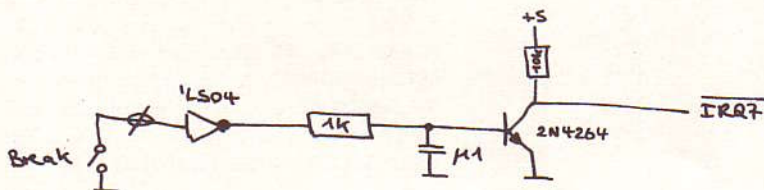
## Briefe - Kontakte - Kleinanzeigen

**Ich habe hier ein paar Fragen, deren Antwort eventuell auch andere NDR-Klein Computerbenutzer interessieren könnten.**

1. Wie kann ich mit CP/M 68K einen Speicherteil disassemblieren, so daß ich das Listing nachher auf einem File habe, um z. B. dieses dann abzuändern und dann wieder assemblieren zu können?
2. Ich besitze, wie viele andere sicher auch, moderne Floppy-Laufwerke mit 5.25 oder 3.5 Zoll, welche einen 3ms-Step zulassen würden, zur Zeit leider

den rechten Rand hinaus, der Text überschreibt sich dann nämlich selbst wieder von links. Wie also muß ich den EF9367 korrekt anschließen?

Ich habe die BREAK-Taste meiner Tastatur mit der IRQ7-Leitung des 68000 verbunden, um in CP/M 68K alle Programme abbrechen zu können. Durch die unten beschriebene Entprellschaltung bleibt das Signal dann ziemlich lange low. Dies führt dann jedoch dazu, (das vermute ich wenigstens) daß der RTE-Befehl am Ende der Ausnahmeroutine gerade wieder einen Interrupt auslöst, da das IRQ7-



aber mind. 6ms erhalten. Wie läßt sich eine solche Geschwindigkeitssteigerung in CP/M 68K am elegantesten realisieren?

3. Welches sind die Unterschiede der beiden in CP/M 68K mitgelieferten Linker L068 und LINK68?
4. Um die Bildbreite zu vergrößern, habe ich auf der GDP-Karte den EF9366 durch den EF9367 und den 14MHz-Quarz durch einen von 12MHz ersetzt. Das funktioniert gut, bis auf das Schreiben über

Signal immer noch auf low ist. Läßt sich dieses Problem irgendwie softwaremäßig lösen?.

von Tyko Strassen

**Antwort LOOP:**

Sehr geehrter Herr Strassen, vielen Dank für Ihr Schreiben vom 13. 9. 86. Ich habe Ihren Brief an die Loop-Re-

daktion weitergeleitet. Wir werden prüfen, ob wir ihn als Leserbrief mit Beantwortung veröffentlichen können.

Nun aber zu Ihren Fragen:

1. Leider ist uns nicht bekannt, wie man mit den Standardprogrammen von CP/M 68K einen Speicherteil disassemblieren und das Listing später in einem File abspeichern kann. Hierzu wird ein spezielles Programm erforderlich sein, über das wir aber im Moment noch nicht verfügen.
2. Der von Ihnen gewünschte 3ms-Step für Ihre Floppy-Laufwerke kann vom Floppy-Controller nur erzeugt werden, wenn dieser vorher auf 8 Zoll Betrieb umgeschaltet wird. Hierzu ist eine Bios Änderung erforderlich, die vor jedem nötigen Step auf 8 Zoll und nach jedem Step wieder auf Minilauferwerk umschaltet. Eine elegantere Realisierungsmöglichkeit ist uns nicht bekannt.
3. Der Unterschied der beiden in CP/M 68K mitgelieferten Linker L068 und LINK68 ist in der beigelegten Dokumentation ersichtlich. LINK68 hat wesentlich mehr Möglichkeiten, ist komfortabler zu bedienen und funktionssicherer. LINK68 kann hervorragend mit AR68 zusammenarbeiten. Darum wurde er extra mitgeliefert.
4. Aus Ihrer Beschreibung können wir entnehmen, daß Sie den EF9367 korrekt angeschlossen haben. Das Schreiben



über den rechten Rand hinaus ist beim EF9367 anders realisiert als beim EF9366.

5. Ein Interrupt-Request der Prioritätsebene 7 ist flankengeträgert, daher wird bei einer entprellten Schaltung nur ein Interrupt von der CPU erkannt, wenn Sie die Taste betätigen. Wie lange die Interrupt-Request-Anforderung danach auf Low Pegel ist, ist für die Hardware völlig belanglos. Falls Ihre Programme nicht wie gewünscht funktionieren, muß die Ursache hierfür an Ihrer Software liegen. Ich hoffe, Ihre Fragen zu Ihrer vollsten Zufriedenheit beantwortet zu haben und verbleibe

mit freundlichen Grüßen

GES GmbH

Axel Granel

## Zum Thema Anfänger:

Wie lange bleibt man eigentlich ein solcher?? Nachdem ich mich seit etwa zwei Jahren mit dem Bau des NDR-Klein-Computers befasse, aber immer noch sowohl in den Bauanleitungen als auch in vielen Beschreibungen in der Loop Schwierigkeiten hatte und noch habe, die zu vielen schriftlichen und telefonischen Rückfragen sowie Einsenden von Platinen führten, hatte ich schon an meiner Auffassungsgabe gezweifelt! Durch die Loop fanden sich zunächst 7 Gleichgesinnte zusammen, denen es ähnlich wie mir erging. Ich schöpfte wieder neuen Mut, schließlich braucht man hin und wieder ein Erfolgserlebnis! Wir vereinbarten, unsere Probleme vorzutragen und Erfahrungen auszutauschen. Hierfür stellte uns in dankenswerter Weise Herr Peters von der Firma SYMIC in Mönchengladbach einen geeigneten Raum zur Verfügung. Das Angenehme daran ist, daß die Firma SYMIC über einen Akustik-Koppler verfügt und mit Kempfen sofort in Verbindung treten kann und daß alle Bausätze dort erworben werden können.

Das Angenehme in dieser Gemeinschaft ist, daß keiner sich scheut zuzugeben, daß er das eine oder andere nicht kapiert hat. Solche gemeinsame Treffen sollten vielmehr gefördert werden. Alle NDR-Klein-Computer-Bauer seien aufgerufen, sich zusammenzufinden, auch auf die Gefahr hin, daß einzelne Teilnehmer nicht immer oder nur selten davon profitieren. Sie können aber des Dankes derjenigen sicher sein, die von ihnen etwas lernen. Schließlich betreiben wir die Computerei ja nicht kommerziell sondern weil es uns Spaß macht!

Anmerkung der Redaktion:

Kontaktadresse:

H. Peters, Fa. SYMIC  
Lessingstraße 1 a  
4950 Mönchengladbach  
Telefon (02 11) 8 38 53 77

## KONTAKTE

Suche Kontakt mit NKC-Anwendern, die Erfahrung mit Terminalprogrammen wie z. B. MOVE-IT haben.

Olaf Hempelmann  
Gellertstraße 57 A · 3000 Hannover 1

Betr.: Kontakt zu anderen NDR-Klein-Computer-Nutzern.

Sehr geehrte Damen und Herren,  
Seit einiger Zeit arbeite ich an und mit meinem NDR-Computer-System. Mit den damit zusammenhängenden Problemen habe ich mich bisher allein herumgeschlagen. Das soll nun anders werden. Da ich bei einer von Ihnen durchgeführten Fragebogenaktion mein Einverständnis zur Weitergabe meiner Anschrift an andere NDR-Computer-Nutzer geben sollte und auch gegeben habe, denke ich, daß Sie auch andere Adressen weitergeben können.

Insbesondere bin ich daran interessiert, andere NDR-Computer-Besitzer, Arbeitsgruppen oder Clubs aus Norddeutschland und hier insbesondere (aber nicht ausschließlich) aus den Postleitzahlbereich 28 (Bremen) und 29 (Oldenburg) kennenzulernen. Für Ihre Bemühungen bedanke ich mich im voraus.  
Reinhard Kück  
Haareneschstraße 41 · 2900 Oldenburg

Suche Kontakt Z80 CPM2,2  
Ernst van Aaken  
Sperberweg 18 · 3258 Aerzen 1

Bitte teilen Sie mit, ob es in meiner Nähe einen Anwender des NDR-Computers gibt, der seine Adresse freigegeben hat für Kontaktaufnahme zwecks Erfahrungsaustausch.

Meine Adresse können Sie selbstverständlich auch veröffentlichen.  
Winfried Helfmeier  
Hauptstr. 32 · 4796 Salzkotten-Verne

## Kleinanzeigen

Zu verkaufen:

SCHACH für 680xx-Systeme: DM 29,-, 3 Spielst., speichert 300 gespielte oder eingegebene Partien auf Disk., löst Matt

bis zu 9 Zügen, etc. (Opt. Quelltext m. Beschreibung); MÜHLE DM 19,-, GOMUKO DM 9,- u.v.m. Preise + Porto per NN auf Disk. 5 1/4" unter JADOS. Auf Wunsch für andere Systeme oder Eporm. Info (10 S.) DM 1,50 i. Briefm. Uwe Steinhaus-Peers, 2300 Kiel 1, Schauenburgerstr. 52

**NDR 68000/8:** JADOS-Entwickler verk.

- REVERSI (starkes Strategiespiel) 29,-
- DISASS (komfort. Disassembler) 49,-
- INSPEC (Disketten-Editor/Doktor) 44,- zzgl. Porto, per NN, auf Disk 3 1/2 o. 5 1/4" unter JADOS! Reversi auch auf EPROM.

K. Janßen, Hanninxweg 74, 4150 Krefeld 1

**Zu verkaufen:**

CPU 68K, GDP, Key, Tast, 2 \* ROA 64, DRAM 64, CAS, BUS III, 2 \* IOE, CENT, Gehäuse, Monitor, Tastatur, Drucker, Netzgerät, Bücher, Listings, Loop, Software auf EPROMS: Pascal, MON 68, EDATED, EDEMO, PAC-BOY, REVERSI, EBIO, nur komplett abzugeben. Verkaufspreis: DM 2200,-, K. Schertel, Arndtstraße 9, 8480 Weiden

**Verkaufe** für NDR-680xx: Kopierprogramm „UNICOPY“ (Single-drive kopieren, Speicherausbau unabhängig, Bibliotheksaufruf, im Speicher verschiebbar, alle Formate, Fehleranzeige, Menue „Uform“-ähnlich, schneller Algorithmus), Disk 5 1/4" m. Anleitung, nur DM 35,- per NN oder Vorkasse. K. Ziegler, Poststraße 6, 8481 Eslarn, Tel. (09653) 330

**Verkaufe** NDR-Computer fertig aufgebaut und funktionsfähig: CPUZ80, BUS2, GDP64K, KEY, TAST1 komplett mit Originalgehäuse und -kabel, EBASIC, EGRUND, EGRUND 2000, ESKOP, Handbücher einschl. FLO3, Sonderheft 1 + 2, Loop 0 - 3. VB: DM 850,-, auch einzeln abzugeben. Josef Eisele, Alex.-Küster-Weg 4, 8918 Dießen

**Verkaufe:** Z80 Grundsystem: SBC2, MONI1, IOE, KEY, TAST1, Gehäuse, GDP64K, CAS, BUS, POW5V, als Bausatzpaket.  
A. Trendelkamp, (0231) 573694, nach 18.00 Uhr.

**Verkaufe:** CPU Z80, FLO2, GDP64, BANKBOOT, RAM64/256, HD64180 usw. Platinen ca. 1/2 Bausatzpreis; komplett VHB. W. Weigelt, Mühlenstr. 6, 2913 Augustfehn, Tel. (04489) 2909

**Verkaufe:** Paket4 + Promer + Gehäuse1 funktionsfähig DM 700,-, Harlacher, Hainbuchenweg 24, 8900 Augsburg, Tel. (0821) 705367

**Verkaufe** wegen Hobbyaufgabe: CPU68K, GDP64, BUS3, IOE + CENT, 2 x ROA64, PROMER, CAS, Datenrecorder,

TAST1, KEY, NE2; Monitor, Typenrad-drucker. Einzeln zum halben Neupreis – komplett viel billiger. Tel. (040) 3905428

#### Zu verkaufen:

1TEAC-Laufwerk FD-55FV-13 nur zu Test-zwecken wenig gebraucht DM 275,-. Erich Rohweder, Weberstr. 7, 4005 Meerbusch 1, Tel. (02105) 73362

**Verkaufe:** CP/M-NDR-Computer bestehend aus IBM-Gehäuse, 2 ROA64K mit RAM, 2 TEAC 5¼", GDP, gr. Tastatur, KEY, CENT, BUS3, SBC3, ZEAT + DISC, Turbo-pascal, HEBAS, FLO2, Netzteil, einzeln oder kompl. für 50% vom Bausatzpreis. R. Weber, Tel. (089) 3114962

**Verkaufe:** NDR-CP/M2.2-Computer mit HEBAS und Extras, betriebsbereit. Außerdem SBC2 mit RAMs und EBASIC, CAS, BUS1, POW5V. Martin Jäckering, Kolberger Str. 8, 4460 Nordhorn

**Verkaufe:** CPU-Z80, ROA64, 2 x R8, BUS3, IOE, HEXMON in Gehäuse, EHEX, ROA16, NE2, Literatur, günstig, VB. Martin Mayländer, Saarweiderstraße 4, 6600 Sarbrücken 6

**Verkaufe:** Großer CPM2.2-Ausbau, z. B. RAM256, voll IOE2, Prommer, Floppy, BUS3, Gehäuse, VB DM 1500,-. Tel. (0821) 529434

**Verkaufe:** NDR 68008 DL4NI-MAUS-GRAFIK-Programm DM 50,-. Info: Hahn, Kreuzlach 19, 8806 Neuendettelsau

**NDR-Klein-Comp.** Z80, 2 x 5¼" Laufwerke, Zubehör, Preis: VHS. R. Hargens, 2371 Bredenbek, Tel. (04334) 835

**Verkaufe:** SBC2 50,-; Prommer 50,-; BUS1 30,-; GDP 100,-; Z80 40,-; Bank-Boot 40,-; CPU68K + Grundprogr. 100,-; TAST1 + KEY 50,-; FLO2 100,-; Monitor 100,-; Gehäuse1 80,- komplett + Literatur 700,-, Tel. (08323) 3597

**Verkaufe** komplett aufgeb. NDR-Klein-Comp. mit Geh., Monitor, Tast., Prommer, Floppy, Druckeranschluß. VB für Gesamtpaket DM 3000,- (50% v. Neupreis auch auf Teile). Tel. (06301) 4327

**Verkaufe:** Schroff-Gehäuse ohne Front- und Rückenabdeckung für nur DM 75,-. Farbmonitor TAXAN RGB-VISION II, Auflösung 560 x 210, Superpreis nur DM 290,-. Gerold Hilpert, Kampenwandstraße 35, 8225 Traunreut

## Sonderangebote aus unserem Lager:

Art.-Nr. **30236** EPSON-Drucker RX 100, komplett mit Traktor, 96 ASCII-Zeichen, 100 Z/Sec., Centronic-Schnittstelle, neu, mit Garantie **DM 698,-**

### Fertigeräte NDR-Computer

| Anzahl | Artikel-Nr. | Bezeichnung | VK inkl. MWSt. DM |
|--------|-------------|-------------|-------------------|
| 8      | 10108       | BUS1F r3    | 49,-              |
| 21     | 10110       | BUS2F r3    | 65,-              |
| 15     | 10116       | BUS4F r1    | 169,-             |
| 10     | 10140       | CENF r2     | 38,-              |
| 4      | 10994       | CPU68KF r2  | 109,-             |
| 19     | 10184       | CPUZ80F r4  | 66,-              |
| 13     | 10277       | HCOPYNDR r2 | 145,-             |
| 2      | 10995       | KEYF r2     | 55,-              |
| 13     | 10362       | RAM64F r2   | 175,-             |
| 11     | 10141       | CENP r2     | 8,-               |
| 20     | 10139       | CENB r2     | 29,-              |

### Fertigeräte mc-CP/M-Computer

|     |       |           |       |
|-----|-------|-----------|-------|
| 7   | 10512 | RAM16F r1 | 120,- |
| 3   | 10504 | PROMF r2  | 75,-  |
| 7   | 10472 | FLO1F r3  | 250,- |
| 2   | 10503 | PROMB r2  | 55,-  |
| 21  | 10471 | FLO1B r3  | 155,- |
| 255 | 10473 | FLO1P r3  | 16,-  |

### Sonstiges

|   |       |                                          |       |
|---|-------|------------------------------------------|-------|
| 7 | 10547 | HAMEG 103 Oszilloskop                    | 600,- |
| 6 | 10548 | HAMEG 203-5 Oszilloskop                  | 899,- |
| 4 | 10614 | SA850 Shugart Laufwerk 8"                | 550,- |
| 4 | 60203 | SA200 Shugart Laufwerk 5¼",<br>40 Tracks | 350,- |

### IBM-kompatible Karten wie in LOOP 13 beschrieben, jetzt noch billiger.

|       |              |        |
|-------|--------------|--------|
| 10961 | AT-CPU-Karte | 1498,- |
| 10955 | AT-BUS-Karte | 98,-   |

# Lehrgänge zum NDR-Computer

Wenn Sie sich intensiv mit der Hard- und Software des NDR-Computers beschäftigen wollen, helfen Ihnen diese Kurse weiter:



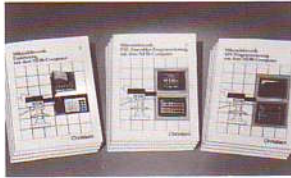
## Kompakt-Kurs Elektronik

Ca. 200 Seiten im Format A4 mit Experimentiermaterial, Tonbandkassette und Flip-chart zur Einführung in die Grundlagen der Elektrotechnik und Halbleiterphysik. Der Kompakt-Kurs ist ideal für alle, die sich nicht nur mit dem Nachbauen von Schaltungen zufrieden geben, sondern auch selbst Schaltungen ändern, ergänzen oder entwerfen wollen.



## Kompakt-Kurs BASIC

Ca. 200 Seiten im Format A4 mit Abschlußtest. Der Lehrgang behandelt alle gängigen BASIC-Anweisungen und führt Sie anhand von zahlreichen Beispielprogrammen in die ersten Schritte der BASIC-Programmierung ein.



## Kompakt-Kurs Mikroelektronik – Einführung

Ca. 240 Seiten Lehrmaterial im Format A4 mit Abschlußtest. Der Lehrgang zeigt anhand des NDR-Einsteigerpakets, wie Sie Ihren Computer in Maschinensprache programmieren. Jeder Befehl wird erläutert und anhand von Beispielen lernen Sie den Einsatz der Maschinensprache des Z80 lernen.



## Kompakt-Kurs Z80-Assembler-Programmierung

Ca. 240 Seiten Lehrmaterial im Format A4 mit Abschlußtest. Wenn Sie sich in die Assemblersprache einarbeiten wollen, ist dies der richtige Kurs für Sie. Als Hardware-Grundlage dient das ZEAT-Betriebssystem.

# Christiani

Bitte Bestellschein abtrennen und einsenden an:  
Dr.-Ing. P. Christiani GmbH, Postfach 35000, 7750 Konstanz



### Bestellung · Information

Senden Sie mir gleich über die Christiani Kurse ausführliches, kostenloses Informationsmaterial wie Lehrpläne, Probeseiten und Preisliste.

Name, Vorname

Straße, Nummer

PLZ, Ort

35519

## BESTELLKARTE

Ich / Wir bestelle(n) unter Anerkennung Ihrer Geschäfts- und Lieferungsbedingungen folgende Artikel:

| Stück | Bestell-Nr. | Bezeichnung | Einzelpreis |
|-------|-------------|-------------|-------------|
|       | 10 834      | GES-Katalog | 10,-        |
|       | 10061       | LOOP-Abo    | 20,-        |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |

Adresse (umseitig) nicht vergessen!

Datum

Unterschrift  
Bei Minderjährigen die des gesetzl. Vertreters

## BESTELLKARTE

Ich / Wir bestelle(n) unter Anerkennung Ihrer Geschäfts- und Lieferungsbedingungen folgende Artikel:

| Stück | Bestell-Nr. | Bezeichnung | Einzelpreis |
|-------|-------------|-------------|-------------|
|       | 10 834      | GES-Katalog | 10,-        |
|       | 10061       | LOOP-Abo    | 20,-        |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |
|       |             |             |             |

Adresse (umseitig) nicht vergessen!

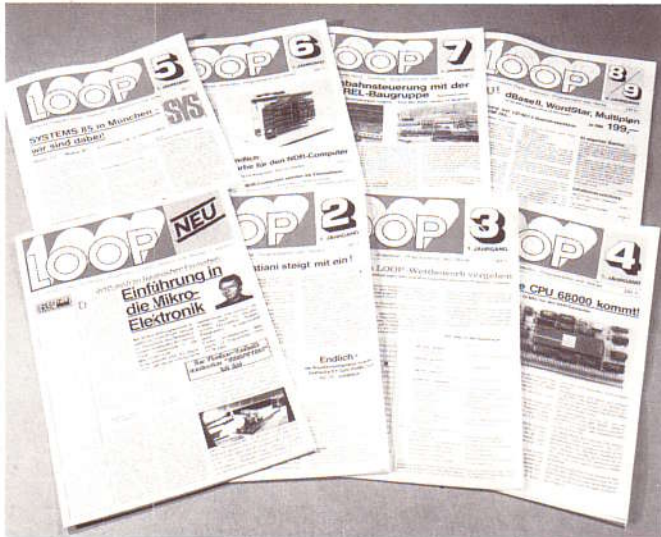
Datum

Unterschrift  
Bei Minderjährigen die des gesetzl. Vertreters

## Auch nach dem Kauf:

Die Computerzeitschrift *LOOP* ist die Brücke zum Kunden – Programme, Infos, Tips + Tricks!

Jahres-Abo DM 20,-, Probeexemplar kostenlos!



Umfassend informiert Sie unser Katalog.

Schutzgebühr: DM 10,- incl. MWSt.



Bitte bestellen Sie mit anhängender Postkarte!

Bitte  
Porto  
nicht  
vergessen

**BESTELL-  
POSTKARTE**

**GRAF  
computer**

Graf Elektronik Systeme GmbH  
Postfach 1610

**8960 Kempten**

Bitte  
Porto  
nicht  
vergessen

**BESTELL-  
POSTKARTE**

**GRAF  
computer**

Graf Elektronik Systeme GmbH  
Postfach 1610

**8960 Kempten**

Anschrift:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Lieferform:  Nachnahme  Vorkasse  
 Bankeinzug

Bankeinzug: Hiermit ermächtige ich die Firma GES GmbH, den Rechnungsbetrag für die auf dieser Karte angegebenen Bestellungen von meinem Konto:

BLZ \_\_\_\_\_ Konto-Nr. \_\_\_\_\_

Bank: \_\_\_\_\_  
abzubuchen. Falls mein Konto die erforderliche Deckung nicht aufweist, besteht seitens des kontoführenden Kreditinstitutes keine Verpflichtung zur Einlösung.

Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_

Anschrift:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Lieferform:  Nachnahme  Vorkasse  
 Bankeinzug

Bankeinzug: Hiermit ermächtige ich die Firma GES GmbH, den Rechnungsbetrag für die auf dieser Karte angegebenen Bestellungen von meinem Konto:

BLZ \_\_\_\_\_ Konto-Nr. \_\_\_\_\_

Bank: \_\_\_\_\_  
abzubuchen. Falls mein Konto die erforderliche Deckung nicht aufweist, besteht seitens des kontoführenden Kreditinstitutes keine Verpflichtung zur Einlösung.

Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_