80-bus journal

Zeitschrift für NASCOM, GEMINI und andere Z80-Anwender

1.JAHRGANG*DEZEMBER 1983*AUSGABE 12

Inhalt

Der Heftpreis beträgt DM 5,-. Ein Abonnement erhalten Sie für DM 60,- im Jahr.

80-bus journal

Intern

Liebe Leser,
hier nun endlich die letzte Ausgabe 83. Eigentlich wollte sich Günter Kreidl an dieser Stelle verabschieden, aber sein Päckchen ist nun schon wieder zwei Wochen überfällig, und so ergreife ich die Gelegenheit, Sie an dieser Stelle zu begrüßen: nämlich zu einem weiteren Jahrgang des Journals. Die vielen positiven Zuschriften und sogar bereits eingegangene Überweisungen haben gegenüber der verschwindend geringen Kündigungen die Oberhand behalten, und so bleibt es nun bei der Herausgabe der Zeitschrift, wie kürzlich dargelegt. Dieser Ausgabe ist eine Überweisung beigefügt, mit der Sie das Abo "besiegeln" können. Sollten Sie schon überwiesen haben, so werfen Sie sie einfach in den Papierkorb. (Leider gingen einige Überweisungen schon an Günter Kreidl und müssen nun umgebucht werden. Bitte beachten Sie, daß keine Zahlungen für's Journal auf das alte Konto gemacht werden sollen. Wir haben es nun auch im Impressum geändert)!
Falls Sie eine Rechnung benötigen, vermerken Sie auf dem Empfängerabschnitt "Rechnung". Diese erhalten Sie dann mit der nächsten Ausgabe. Wir wollen uns auf diese Weise unnötigen Papierkrieg ersparen.
Manche Leser werden vielleicht erschrecken, welchen Umfang die Floppy-Routinen in diesem Heft einnehmen. Das soll auch wieder anders werden, im Augenblick sitzen aber so viele Leser am Nachbau der Karte und an der Anpassung, daß sich dieser große Aufwand entschuldigen läßt. Wenn Sie noch schwanken, ob Sie nicht doch auch in die "Floppy-Bewegung" einsteigen sollten, kann ich Sie nur ermutigen. Jetzt bieten wir schon eine Diskette mit Floppy-Routinen an, in Zukunft ist geplant, sämtliche Programme (vor allem lange Assembler- und BASIC-Listings) einer Journalausgabe auf Diskette anzubieten und dazu noch einiges, was sich wegen der Länge schon garnicht abdrucken läßt. Dies ist eine Erleichterung, die von vielen Lesern schon lange gewünscht wurde.
Die Cassettenbenutzer aus Überzeugung könnten aber ebenfalls eine "Tipphilfe" gebrauchen. Für die Redaktion wäre ein Cassettenangebot zu zeitaufwendig (es gibt ja für's Journal noch einiges mehr zu tun), aber es gäbe zwei Möglichkeiten:
1. Ein Leser (oder mehrere), der Floppy besitzt, könnte von der Redaktion alle Programme erhalten und diese auf Cassette kopieren. In eigener Regie und zu eigenen Preisvorstellungen könnte er diese den Lesern anbieten. Wer hat Interesse?
2. Uwe Fricke hat trotz des verschwundenen letzten Rundlaufs den Mut nicht aufgegeben, einen weiteren Rundlauf ins Leben zu rufen. Er sucht dazu noch gleichfalls Interessierte, die behilflich sind, die alten Journalprogramme und auch die neueren zusammenzutragen und auf Cassette zu sammeln. In Form einer Art "Kettenbrief" sollten diese Cassetten dann an die Teilnehmer weitergeleitet werden. Wer hilft ihm? Die Adresse:
Uwe Fricke
[redacted]
[redacted] Neunkirchen [redacted]
Wichtig:
Die Schublade mit den Beiträgen ist ziemlich leer geworden. Was wir vor allem brauchen sind
1. Anwenderprogramme und Spiele für NASCOM1 und NASCOM2 ohne besondere Hardwareerweiterungen
2. Allgemeine Artikel zur Benutzung von CP/M
3. Erfahrungsberichte über die Anpassung von CP/M
4. Anwenderprogramme, die mit CP/M laufen (Die kommerzielle Software, die jeder kennt, ist ja wohl nicht das einzige Einsatzgebiet dieses Systems)
Ganz leer ist unsere Schublade allerdings noch nicht; sie enthält noch einige Bonbons, die mit der nächsten Ausgabe auf Sie zukommen. Vor allem die Hochauflösende Grafik-Karte (die schon im Platinenservice angeboten wird) mit Software für Grafik von 256x 512 Punkten.
Dann eine komfortable EPROMMER-Karte, die sich per Software von 2708 bis 2764 umschalten läßt; ein SPRITE-Editor, der mit BASIC grafische Figuren definieren kann (mit unse-

rer gewohnten N2 Grafik) und weitere Anwenderprogramme und auch interessante Spiele.
Aber man muß ja auch schon an die übernächste Ausgabe denken, und die Aufforderung zur weiteren regen Mitarbeit ist wirklich eine ernstgemeinte Bitte.
Mit herzlichem Dank schon im Voraus und mit dem Wunsch, daß das Journal auch weiterhin die Arbeit mit unseren Rechnern belebt und vorantreibt.
Ihr Günter Böhm

Service

Im Augenblick können wir drei Platinen anbieten,fertig durchkontaktiert und glanzverzinnt:
80x24 Zeichen Karte DM 60,-
Floppy Controller Karte incl 2 Proms DM 70.-

Hochauflösende Grafik 256x512 incl. 1 Prom DM 65,-
Der Preis versteht sich einschließlich 14% Mehrwertsteuer, Porto und Verpackung. Es entstehen keine weiteren Kosten! (Sollte man beachten, wenn man Preisvergleiche macht. Was da manchmal an Verpackung und Bearbeitungskosten herbeigezaubert wird!)
Die Karten sind momentan (noch) sofort lieferbar.
Bestellung durch Überweisung des Betrages auf folgendes Konto (Bitte gewünschte Karte auf Abschnitt vermerken):
[redacted] PSchA Klrh
Gabi Böhm
In etwa 14 Tagen können wir eine Floppy liefern (Format in diesem Heft beschrieben), Sie enthält folgende Files:

EMDOS.ASSEMBLER	PHEAS.ASSEMBLER
WBOOT.ASSEMBLER	CBOOT.ASSEMBLER
NASGEN.ASSEMBLER	FORMAT.ASSEMBLER
READTRK.ASSEMBLER	STAT.ASSEMBLER
NASGEN.COMMAND	READTRK.COMMAND
FORMAT.COMMAND	STAT.COMMAND

Jeweils die neuesten Versionen. Die Systemspuren enthalten WBOOT und EMDOS, sodaß mithilfe des abgedruckten CBOOT direkt gebootet werden kann. Das Eintippen der übrigen Programme können Sie sich dann sparen.
Die Diskette ist für DM 20,- (incl MWSt, Porto und Verpackung) erhältlich. Bestellung durch Überweisung auf obiges Konto.

Suche 80-Bus News Hefte
Wer kann Sie mir leihweise zur Verfügung stellen?
E.Hecker ; [redacted]
[redacted] Tel. [redacted]
(abends)

Bugs
Formatierprogramm Journal 10/11-83 Seite 17
Folgende Zeilen einfügen bzw. ändern:
1725 LD A, #34
2080 Masch.Code falsch gedruckt:CD B9 81

Impressum

HERAUSGEBER:
Günter Böhm Ludwigshafener Str. 71d
75 Karlsruhe Tel. [redacted]
Redaktion
Gabi Böhm ebendort
Layout u. Versand
Günter Kreidl Bertenweg 18
4172 Straelen Tel. [redacted]
Buchhaltung
KORRESPONDENTEN:

Karl Georg Englmann [redacted]
[redacted] Mutterstadt Tel. [redacted]
Reinzeichnungen
Wolfgang Mayer-Gürr [redacted]
[redacted] Recklinghausen Tel. [redacted]
Clemens u. Max Ballarin [redacted]
[redacted] Ueberlingen Tel. [redacted]
Michael Bach [redacted]
[redacted] Stegen Tel. [redacted]
Peter Brendel [redacted]
[redacted] Mannheim Tel. [redacted]
Hans-Jürgen Plath [redacted]
[redacted] Kiel
Hans Schneider [redacted]
[redacted] Esens
Oesterreich:
Gerhard Klement [redacted]
A-[redacted] Wien Tel. [redacted]
Niederlande:
Eric v.d.Vaart [redacted]
NL-[redacted] Waddixveen
England:
Frank M. Butler [redacted]
Mansfield Woodhouse/Notts
Luxemburg:
Rene Claus [redacted]
L-[redacted] Bonneweg
Schweiz:
Markus Zimmer [redacted]
CH-[redacted] Basel Tel. [redacted]
Jugoslawien:
Gilvazi Istvan [redacted]
YU-[redacted] Becej

VERLAG:
Günter Kreidl 4172 Straelen

Ab 1984 alle Zahlungen für das Journal und damit zusammenhängende Serviceleistungen auf folgendes Konto:
Gabi Böhm
Kto [redacted] PSchA Klrh (Karlsruhe)

BASIC-Step

von KLAUS MOMBAUR

B A S I C - Single Step

```
C by K.Mombaur, Leitenweg 31, 8500 Nuernberg 60
Zeile 0: POKE4100,3448: A=USR(1)
Zeile 1: mindestens REM
Programmstart: E 0C00
Nach Angabe ab welcher Zeilennummer:
     2 x ENTER
Nun ist BASIC im Direct mode:
     Moeglichkeit der Variablenpruefung,
     -veraenderung, Listen etc.
Abarbeitung der naechsten BASIC - Zeile:
     RUN, 3 x ENTER
```

```
0C00   EF     RST   PRS
0C01   0C           CLS
0C02   "BASIC Single Step
0C13    ab welcher Zeile ?"   Gewuenschte
0CA6   0D           CR         Zeile in HEX
0CA7   00                      wandeln, in IX
0CA8   DF 63        INLIN      speichern und
0CAA   11     LD    DE,088F    Stellenzahl
0CAD   1B     DEC   DE         feststellen
0CAE   1A     LD    A,(DE)
0CAF   FE     CP    30
0CB1   38     JR    NC,02(0CB5) DE hat Einerpos
0CB3   18     JR    F8(0CAD)
0CB5   DD21   LD    IX,0000
0CB9   01     LD    BC,0001    =1 Dez
0CBC   CD     CALL  0CE0       >U-Pruefen
0CBF   01     LD    BC,000A    =10 Dez
0CC2   CD     CALL  0CE0
0CC5   01     LD    BC,0064    =100 Dez
0CC8   CD     CALL  0CE0
0CCB   01     LD    BC,03E8    =1000 Dez
0CCE   CD     CALL  0CE0
0CD1   01     LD    BC,2710    =10000 Dez
0CD4   CD     CALL  0CE0
0CD7   31     LD    SP,1000
0CDA   C3     JP    0D00       > weiter

0CDD   xx                      Speicherzellen
0CDE   xx

                               U - Pruefen
0CE0   1A     LD    A,(DE)
0CE1   FE     CP    30
0CE3   DA     JP    C,0CD7     < Dez 0: Ende
0CE6   28     JR    Z,03(0CEB) nur bei <> 0:
0CE8   CD     CALL  0CED       >UU-Addieren
0CEB   1B     DEC   DE
0CEC   C9     RET

                               UU - Addieren
0CED   06     SUB   30         Stellenwert
0CEF   6F     LD    L,A        sooft addieren
0CF0   0D09   ADD   IX,BC      wie (L) angibt.
0CF2   2D     DEC   L          Ergebnis in IX
0CF3   28     JR    NZ,FB(0CF0)

0D00   FD21   LD    IY,10FA    Zeilenzr im
0D04   FD46   LD    B,(IY+01)  Basic-RAM suchen
0D07   FD4E   LD    C,(IY+00)   nae.Blockbeginn
0D0A   ED43   LD    (0CDD),BC  merken
0D0E   DDE5   PUSH  IX
0D10   E1     POP   HL         Zeile erreicht?
0D11   FD56   LD    D,(IY+03)  HOB
0D14   FD5E   LD    E,(IY+02)  LOB
0D17   38     JR    NC,01(0D1A)
0D19   3F     CCF
0D1A   ED52   SBC   HL,DE
0D1C   28     JR    Z,28(0D46) wenn gefunden:
0D1E   FD2A   LD    IY,(0CDD)  nae.Blockbeginn
0D22   FD7E   LD    A,(IY+01)  HOB pruefen
0D25   FE     CP    00         wenn <> 0:
0D27   28     JR    NZ,DB(0D04) weitersuchen
0D29   EF     RST   PRS        Endeerhalg
```

```
0D2A   0C           CLS
0D2B   "Zeile nicht vorhanden"
0D48   00
0D41   DF     RST   18H
0D42   50           TDEL
0D43   C3     JP    0C00
0D46   FD6E   LD    L,(IY+04)  IY=Blockbeginn
0D49   FD66   LD    H,(IY+05)   gesuchte Zeile
0D4C   22     LD    (0DF0),HL  BC=B.beg nae.Zei
0D4F   FD36   LD    (IY+04),3F IX=ZNr in HEX
0D53   FD36   LD    (IY+05),3A 2 Daten retten
0D57   FD22   LD    (0DF2),IY  2 neue Daten:
0D5B   DD22   LD    (0DF4),IX   "STOP :"
0D5F   ED43   LD    (0DF6),BC  Register retten
0D63   EF     RST   PRS
0D64   0C 0D        CLS CR
0D65   "RUN1"
0D6A   0D           CR
0D6B   13           DC3
0D6C   13           DC3
0D6D   00
0D6E   DF     RST   18H        Z-Befehl
0D6F   5A                      > BASIC

                               Ansprung v.BASIC
0D70   FD2A   LD    IY,(0DF2)  akt. Zeile
0D74   DD2A   LD    IX,(0DF6)  nae. Zeile
0D78   2A     LD    HL,(0DF0)  Zeile und Regist
0D7B   FD75   LD    (IY+04),L  restaurieren
0D7E   FD74   LD    (IY+05),H
0D81   DD6E   LD    L,(IX+04)  neue Zeile
0D84   DD66   LD    H,(IX+05)   Daten retten
0D87   22     LD    (0DF0),HL
0D8A   DD36   LD    (IX+04),3F 2 neue Daten
0D8E   DD36   LD    (IX+05),3A
0D92   DD6E   LD    L,(IX+00)  Register fuer
0D95   DD66   LD    H,(IX+01)  nae Zeile retten
0D98   22     LD    (0DF6),HL
0D9B   DD22   LD    (0DF2),IX
0D9F   FD22   LD    (0DF4),IY
0DA3   21     LD    HL,0DB6    Ausg Tabelle
0DA6   11     LD    DE,088A    on Screen
0DA9   01     LD    BC,002F
0DAC   EDB8   LDIR
0DAE   21     LD    HL,384A    Cursorpos
0DB1   22     LD    (0C29),HL
0DB4   DF     RST   18H        Z-Befehl
0DB5   5A                      > BASIC

0DB6-0DE6: A=PEEK(3332):CLS:?:?"GOTO"+STR$(PEEK
           (A+2)):SCREEN1,1:END
0DF8-0DF7: Speicherzellen
```

Mini-Calc

von ISTVAN GILVAZI

In Heft 7/8-83 wurde ein Minikalkulationsprogramm angekündigt. Jetzt können wir es endlich abdrucken. Die Beschreibung finden Sie in Heft 7/8 auf der Seite 9. Das Programm kann für eigene Zwecke leicht abgeändert werden.

```
2 REM MINI-CALC
3 REM von GILVAZI ISTVAN
4 REM [redacted]
5 REM [redacted] Becej/ Jugoslawien
6 :
10 DATA-29747,-4631,-32685,-8436,-389,-13811
15 DATA3220,-15369,3209,201
```

```
20 FOR A=3202 TO 3220 STEP 2
25 READ B:DOKEA,B:NEXTA
40 DOKE4100,3202
50 INPUT"NAME :";Q$
55 CLS
60 SCREEN1,16:PRINTQ$TAB(14)"PREIS";
65 PRINT".....MENGE.....G.PREIS"
70 FORA=1TO14:SCREEN21,A:PRINT".":NEXTA
75 FORA=1TO14:SCREEN32,A:PRINT".":NEXTA
80 B=49
85 FORA=2059TO2635STEP64:POKEA,B
90 B=B+1:NEXTA
95 FORA=0TO44:SET(95,A):NEXTA
100 FORA=1TO95:SET(A,40):NEXTA
105 FORA=1TO13:SCREEN3,A:PRINTCHR$(148):NEXTA
110 FORA=1TO13:SCREEN14,A:PRINTCHR$(148):NEXTA
115 FORA=1TO13:SCREEN24,A:PRINTCHR$(148):NEXTA
120 FORA=1TO13:SCREEN35,A:PRINTCHR$(148):NEXTA
125 FORA=1TO13:SCREEN36,A:PRINTCHR$(148):NEXTA
200 POKE2634,49:POKE2635,48:POKE2698,49
205 POKE2699,49:POKE2762,49:POKE2763,50
210 POKE2826,49:POKE2827,51
215 SCREEN1,15:PRINT"DRUCK :";
220 Z=USR(0)
225 CLEAR1000
230 DIMA$(14),B$(14),C$(14),E(15),F(15),A(15)
235 DIMB(15),C(15),H(15)
240 D=1
245 FORC=2072TO2840STEP64
250 FORA=CTOC+8
255 B=PEEK(A)
260 A$(D)=A$(D)+CHR$(B)
265 NEXTA
270 D=D+1
271 NEXTC
272 D=1
273 FORC=2083TO2851STEP64
274 FORA=CTOC+8
275 B=PEEK(A)
276 B$(D)=B$(D)+CHR$(B)
277 NEXTA
278 D=D+1
279 NEXTC
280 FORD=1TO14
285 A(D)=VAL(A$(D)):B(D)=VAL(B$(D))
290 NEXTD
295 FORD=1TO14
300 C(D)=A(D)*B(D):C(D)=(INT(C(D)*100))/100
305 NEXTD
310 FORD=1TO14
315 C$(D)=STR(C(D))
320 E(D)=LEN(C$(D))
325 NEXTD
330 FORX=1TO13
335 G=2104+((X-1)*64)
340 FORD=1TOE(D)
345 F(X)=ASC(RIGHT$(C$(X),D))
350 POKEG,F(X)
355 G=G-1
360 NEXTD
365 NEXTX
370 FORD=1TO13
375 I=C(D)-INT(C(D))+0.0005:I1=I1+I
380 I2=I2+INT(C(D))
385 NEXTD
390 I2=I2+INT(I1):I1=I1-INT(I1)
395 I1=INT(I1*100)
400 D$=STR$(I2)+"."+RIGHT$(STR$(I1),2)
410 E1=LEN(D$)
420 G=3000
430 FOR D=1TOE1
440 F1=ASC(RIGHT$(D$,D))
450 POKEG,F1:G=G-1
460 NEXTD
470 Z=USR(0)
480 GOTO225
Ok
```

Spielothek 1

von D.KASTRUP/W.SAUERBREY

Das Programm enthält Grafikzeichen.
Die Umlaute sind folgendermaßen zu
interpretieren:
ö = |
Ö = \
Beim Eintippen können die Zeichen
durch gleichzeitiges Drücken folgender
Tasten erreicht werden:

shift/contr./<	|
shift/contr./graf/W	Ä
contr. E	Ö
contr. G	Ü
Contr. N	Ø

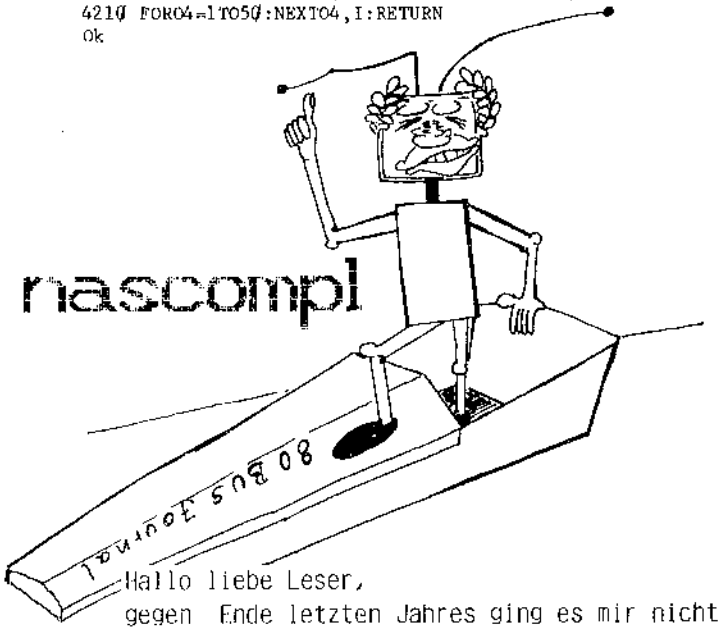
```
1 CLS:PRINT"WAEHLE :":PRINT:PRINT:GOSUB4110
2 PRINT,"ANDROIDEN",1:PRINT,"JAGD",2
3 PRINT,"FLIP",3:PRINT,"4 IN 1 REIHE",4
4 GOSUB4140:X=IN-48:IFX<0GOTO4
5 ONXGOTO10,1510,2460,3230:GOTO4
6 FORX=1TO2000:RUN
10 REM **** A N D R O I D E N ****
20 REM AUTOR: Wolfgang Sauerbrey
30 REM          David Kastrup
40 REM VERSION: 2.12.81
50 CLS
60 CLEAR
70 PRINT
80 PRINT"Wir spielen: RETTE DEN ANDROIDEN."
90 PRINT:PRINT"Aufgabe dieses Spieles ist es,";
100 PRINT" den Androiden   vor den Robotern in"
;
110 PRINT" Sicherheit zu bringen. Die Roboter "
;
```

```
120 PRINT"versuchen, sich dem Androiden auf dem
"
130 PRINT"kuerzesten Weg zu naehern. Sie werde"
;
140 PRINT"n aber     dabei zerstoert, wenn sie"
;
150 PRINT" auf eine Bombe oder  einen anderen "
;
160 PRINT"Roboter laufen. Die Bewegung des And"
;
170 PRINT"roiden wird wie folgt gesteuert:":PRI
NTTAB(38);
180 INPUT"fertig ";X:PRINT:PRINTTAB(20);"Z  1
X"
190 PRINTTAB(21);"ö ö /"
200 PRINTTAB(22);:POKEPEEK(3113)-2,13:PRINT"-+-
                "
210 PRINTTAB(21);"/ ö ö"
220 PRINTTAB(20);".     /":PRINT:PRINT
230 PRINT"Um stehenzubleiben, druecke die Space
-Tas";
240 PRINT"te,":PRINT"um aufzugeben,Shift a)":PRI
NT:PRINT
250 PRINT"Zeichenerlaeuterung:"
260 PRINT"   Ω - Androide"
270 PRINT"   Æ     - Roboter"
280 INPUT"   θ - Bombe            alles klar";A$
290 PRINT:PRINT"Wie gross soll das Spielfeld s"
;
300 PRINT"ein?","(max 13,24)";
310 G=13:H=24:INPUTG,H
320 IFG<13ORG>5GOTO290
330 IFH>24ORH<5GOTO290
340 DIMS$(G,H),A(62,2)
350 R=30
360 PRINT:PRINT"Wieviele Roboter sollen den";
370 PRINT" Androiden jagen?   (max 30)";
380 INPUTR
390 P=2*R+2
400 IFP>62GOTO360
410 CLS
420 S=0:T=0:Z=0
430 FORI=1TOG:FORJ=1TOH
440 S$(I,J)="."
450 SCREEN2*J,I
460 PRINTS$(I,J)
470 NEXTJ,I
480 FORI=1TOP
490 Y=INT(G*RND(1)+1)
500 X=INT(H*RND(1)+1)
510 IFS$(Y,X)<>"."GOTO490
520 IFI=1GOTO560
530 IFI>P/2+1GOTO580
540 S$(Y,X)="θ"
550 GOTO590
560 S$(Y,X)="Ω"
570 GOTO590
580 S$(Y,X)="Æ"
590 SCREEN2*X,Y
600 A(I,1)=Y:A(I,2)=X
610 PRINTS$(Y,X)
620 NEXTI
630 GOSUB1430
640 SCREEN2,G+1:PRINT"                        ";
650 SCREEN2,G+1
660 GOSUB4140
670 IFIN=64GOTO1410
680 IFIN=32GOTO960
690 S$(A(1,1),A(1,2))="."
700 SCREEN2*A(1,2),A(1,1)
710 PRINTS$(A(1,1),A(1,2))
720 IFIN=46GOTO830
730 IFIN=20GOTO840
740 IFIN=47GOTO850
750 IFIN=17GOTO860
760 IFIN=18GOTO870
770 IFIN=90GOTO880
780 IFIN=19GOTO890
790 IFIN=88GOTO900
800 S$(A(1,1),A(1,2))="Ω":SCREEN2*A(1,2),A(1,1)
810 PRINT"Ω"
820 GOTO640
830 A(1,2)=A(1,2)+(A(1,2)<1)
840 A(1,1)=A(1,1)-(A(1,1)<G):GOTO910
850 A(1,2)=A(1,2)-(A(1,2)<H):GOTO840
860 A(1,2)=A(1,2)+(A(1,2)>1):GOTO910
870 A(1,2)=A(1,2)-(A(1,2)<H):GOTO910
880 A(1,2)=A(1,2)+(A(1,2)>1)
890 A(1,1)=A(1,1)+(A(1,1)>1):GOTO910
900 A(1,2)=A(1,2)-(A(1,2)<H):GOTO890
910 IFS$(A(1,1),A(1,2))="θ"GOTO1190
920 IFS$(A(1,1),A(1,2))="Æ"GOTO1190
930 S$(A(1,1),A(1,2))="Ω"
940 SCREEN2*A(1,2),A(1,1)
950 PRINTS$(A(1,1),A(1,2))
960 Z=Z+1
970 C=9
980 IFZ>9THENC=8
990 SCREENC,16
1000 PRINTZ;
1010 FORI=R+3TOP
1020 IFA(I,1)<0GOTO1160
1030 S$(A(I,1),A(I,2))="."
1040 SCREEN2*A(I,2),A(I,1)
1050 PRINTS$(A(I,1),A(I,2))
1060 FORJ=1TO2
1070 IFA(1,J)-A(I,J)=0GOTO1090
1080 A(I,J)=A(I,J)+SGN(A(1,J)-A(I,J))
1090 NEXTJ
1100 IFS$(A(I,1),A(I,2))="θ"GOTO1240
1110 IFS$(A(I,1),A(I,2))="Æ"GOTO1240
1120 IFS$(A(I,1),A(I,2))="Ω"GOTO1360
1130 S$(A(I,1),A(I,2))="INT"
1140 SCREEN2*A(I,2),A(I,1)
1150 PRINTS$(A(I,1),A(I,2))
1160 NEXTI
1170 GOTO640
1180 RUN6
1190 SCREEN2,G+1
1200 PRINT"Der Androide hat sich selbst zersto"
;
1210 PRINT"ert!"
1220 GOTO1410
1230 CLS:GOTO360
1240 A(I,1)=-1
1250 T=T+1
1260 C=33
1270 IFT>9THENC=32
1280 SCREENC,16
1290 PRINTT;
1300 IFT=RGOTO1320
1310 GOTO1160
1320 SCREEN2,G+1
1330 PRINT"Der Androide ist gerettet!"
1340 GOTO1410
1350 CLS:GOTO360
1360 S$(A(I,1),A(I,2))="INT"
1370 SCREEN2*A(I,2),A(I,1)
1380 PRINTS$(A(I,1),A(I,2))
1390 SCREEN2,G+1
1400 PRINT"Der Androide wurde vernichtet! "
1410 PRINT" Noch ein Spiel?";
1411 GOSUB4140:IFIN$="N"THENRUN6
1420 IFIN$="J"THENCLS:GOTO360
1425 GOTO1411
1430 SCREEN1,16
1440 PRINT" Zuege:  0          zerst. Rob.:  0";
1450 RETURN
1510 REM **** J A G D ****
1520 REM AUTOR: Wolfgang Sauerbrey
1530 REM 26.10.80
1540 REM
1550 CLS
1560 PRINT"Ziel dieses Spieles ist es, den hori
z";
1570 PRINT"ontal wan-dernden Strich (-) zu tref
f";
1580 PRINT"en. Dazu laesst sichder auf und ab b
e";
1590 PRINT"wegende Ball mit Hilfe der    Tasten
 ";
1600 PRINT"CTRL und LF/CH auch waagerecht nach"
```

```
161Ø PRINT"links, bzw. nach rechts steuern. Es
";
162Ø PRINT"werden die Punkte von 1Ø Spielen auf
";
163Ø PRINT"addiert.":PRINT
164Ø PRINT"Betaetige ENTER, um das Spiel zu sta
";
165Ø PRINT"rten!"
166Ø INPUTA$
167Ø Z=Ø
168Ø FORK=1TO1Ø
169Ø PRINT
17ØØ CLS
171Ø FORI=1TO46
172Ø PRINT"ß";
173Ø NEXT
174Ø PRINT"ß"
175Ø SCREEN1,14
176Ø FORI=1TO46
177Ø PRINT"_";
178Ø NEXT
179Ø PRINT" "
18ØØ M=INT(RND(1)*12)+2
181Ø SCREEN1,M
182Ø PRINT"-"
183Ø N=1
184Ø S=INT(RND(1)*47)+1
185Ø R=S
186Ø SCREEN5,1
187Ø PRINT"☒"
188Ø FORI=1TO1ØØ
189Ø FORJ=2TO14
19ØØ GOSUB232Ø
191Ø SCREEN5,J-1
192Ø IFJ=2GOTO195Ø
193Ø PRINT" "
194Ø GOTO196Ø
195Ø PRINT"ß"
196Ø S=R
197Ø SCREEN5,J
198Ø PRINT"☒"
199Ø IFS<>NGOTO2Ø1Ø
2ØØØ IFJ=MGOTO218Ø
2Ø1Ø NEXT
2Ø2Ø FORJ=13TO1STEP-1
2Ø3Ø GOSUB232Ø
2Ø4Ø SCREEN5,J+1
2Ø5Ø IFJ=13GOTO2Ø8Ø
2Ø6Ø PRINT" "
2Ø7Ø GOTO2Ø9Ø
2Ø8Ø PRINT"_"
2Ø9Ø S=R
21ØØ SCREEN5,J
211Ø PRINT"☒"
212Ø IFS<>NGOTO214Ø
213Ø IFJ=MGOTO218Ø
214Ø NEXTJ,I
215Ø SCREEN1,15
216Ø PRINT"Nicht getroffen! Ø Punkte.";
217Ø GOTO221Ø
218Ø SCREEN1,15
219Ø PRINT"Getroffen! ";1Ø1-I;"PUNKTE  ";
22ØØ Z=Z+1Ø1-I
221Ø PRINT"  ges.:";Z;" ";K;".Spiel";
222Ø FORI=1TO4ØØØ
223Ø NEXTI,K
224Ø PRINT
225Ø CLS
226Ø PRINT"Du hast insgesamt";Z;"Punkte."
227Ø PRINT:PRINT"Willst du ein neues Spiel?"
228Ø A$="":INPUTA$
229Ø IFA$="N"ORA$="NEIN"THENRUN6
23ØØ IFA$="J"ORA$="JA"GOTO167Ø
231Ø GOTO227Ø
232Ø L=INP(Ø)
233Ø IFL=247THENR=S-1
234Ø IFL=191THENR=S+1
235Ø IFR<1THENR=1
236Ø IFR>47THENR=47
237Ø IFN=1THENF=1
238Ø IFN=47THENF=-1
239Ø SCREEN5,M
24ØØ PRINT" "
241Ø N=N+F
242Ø SCREEN5,M
243Ø PRINT"-"
244Ø RETURN
246Ø CLS:SCREEN1Ø,16:PRINT"F   L   I   P"
247Ø B1=5Ø
248Ø PRINT"Erklaerung (J oder N) ?"
249Ø GOSUB316Ø
25ØØ IFIN$="N"GOTO261Ø
251Ø PRINT"Bei jedem Mal raetst du Ja (J) oder
Nein";
252Ø PRINT" (N).":PRINT"Der Computer hat sich s
chon ";
253Ø PRINT"vorher eine":PRINT"der beiden Moegli
chkeiten ";
254Ø PRINT"ausgesucht.":PRINT"  Zuerst betraeg
t deine ";
255Ø PRINT"Fehlerrate 5Ø%, aber":PRINT"dann ver
sucht der";
256Ø PRINT"Computer, hinter deinen":PRINT"Einga
ben Syst";
257Ø PRINT"eme zu entdecken.":PRINT
258Ø PRINT"Das Spiel endet nach";B1;" Versuchen
;"
259Ø PRINT"eine Punktzahl von";INT(B1/2-1);"ist
 gut."
26ØØ PRINT
261Ø PRINT
262Ø PRINT
263Ø DIMP(16),X(4)
264Ø PRINT"Anfang."
265Ø FORI=1TO16
266Ø P(I)=.5
267Ø NEXTI
268Ø FORI=1TO4
269Ø X(I)=Ø
27ØØ IFRND(1)<.5GOTO272Ø
271Ø X(I)=1
272Ø NEXTI
273Ø F1=.8
274Ø F2=.3
275Ø S1=Ø
276Ø S2=Ø
277Ø A$="Versuch Nr."
278Ø I9=8*X(4)+4*X(3)+2*X(2)+X(1)+1
279Ø Z1=P(I9)
28ØØ Z2=Z1
281Ø IFZ2<>.5GOTO284Ø
282Ø Z2=RND(1)
283Ø GOTO288Ø
284Ø IFZ2<.5GOTO287Ø
285Ø Z2=(Z2+1)*F2-1
286Ø GOTO288Ø
287Ø Z2=(Z2-1)*F2+1
288Ø Z5=Ø
289Ø IFRND(1)<Z2GOTO291Ø
29ØØ Z5=1
291Ø PRINTCHR$(27):PRINTCHR$(I9);A$;S2+1;"?";
292Ø Z3=Ø
293Ø GOSUB316Ø
294Ø IFIN$="N"GOTO296Ø
295Ø Z3=1
296Ø A$="Falsch geraten! Versuch Nr."
297Ø S2=S2+1
298Ø IFZ3<>Z5GOTO3Ø1Ø
299Ø A$="Richtig geraten! Versuch Nr."
3ØØØ S1=S1+1
3Ø1Ø X(1)=X(3)
3Ø2Ø X(2)=X(4)
3Ø3Ø X(3)=Z3
3Ø4Ø X(4)=Z5
3Ø5Ø P(I9)=F1*P(I9)+(1-F1)*X(3)
3Ø6Ø IFS2<B1GOTO278Ø
3Ø7Ø PRINTCHR$(27):PRINTCHR$(19);LEFT$(A$,16)
3Ø8Ø PRINT
3Ø9Ø PRINT"Ende des Spieles."
31ØØ PRINT"Du hast ";S1;" von ";S2;" erraten."
311Ø PRINT:PRINT
312Ø PRINT"Noch ein Spiel ?"
```

```
3130 GOSUB3160
3140 IFIN$="J"GOTO2640
3150 RUN6
3160 GOSUB4140:IFIN$="J"ORIN$="N"THENRETURN
3170 GOTO3160
3230 COSUB3240:QP$=CHR$(27)+CHR$(19):GOTO3250
3240 CLS:SCREEN15,16:PRINT"Vier in einer Reihe"
:RETURN
3250 T$="              ";SCREEN1,5:GOSUB4110
3260 DIML(8),S(4),F(4)
3270 DIMV(16),N(4),B(8,8)
3280 DATA1,100,500,1E20,1,800,4000,1E20
3290 DATA1,75,900,1E18,1,450,3000,1E18
3300 FORZ1=1TO16:READV(Z1):NEXT
3310 PRINT"Das Spiel 'Vier in einer Reihe' :"
3320 PRINT"Brauchst du eine Anleitung ?":GOSUB4
140
3330 PRINT:PRINT:IFIN$="N"GOTO3420
3340 IFIN$="J"GOTO3360
3350 PRINT"Ja oder Nein":GOTO3320
3360 PRINT"Das Spiel  beruht auf dem Stapeln vo
n"
3370 PRINT"X's und O's (mein Zeichen ist O) ,bi
s"
3380 PRINT"einer der Spieler horizontal,vertika
l"
3390 PRINT"oder diagonal  vier seiner Steine  i
n"
3400 PRINT"einer Reihe hat.              Alles kl
ar ?"
3410 GOSUB4140:PRINT:PRINT
3420 X=ASC("X"):O=ASC("O")
3430 GOSUB3240:SCREEN1,4:FORD1=1TO8:PRINTT$;
3440 FORD2=1TO8:PRINT"-  ";:NEXT:PRINT:NEXT
3450 PRINTTAB(12);:FORM5=1TO8:PRINTM5;:NEXT:PRI
NT
3460 PRINT:FORZ1=1TO8:L(Z1)=0:FORL1=1TO8
3470 B(L1,Z1)=0:NEXTL1,Z1
3480 PRINTQP$:PRINTT$;"Willst du anfangen ?";:G
OSUB4140
3490 IFIN$="N"GOTO3640
3500 IFIN$="J"GOTO3530
3510 GOTO3480
3520 PRINTQP$:PRINTT$;"Illegaler Zug !";:FORO4=
1TO2000:NEXT
3530 PRINTQP$:PRINTT$;"Dein Zug (1-8) ?";:GOSUB
4140
3540 M=IN-48
3550 IFM<1ORM>8GOTO3520
3560 L=L(M)
3570 IFL>7GOTO3520
3580 P=X:GOSUB4160
3590 B(L,M)=X
3600 GOSUB3960
3610 FORZ=1TO4
3620 IFS(Z)<4THENNEXT:GOTO3640
3630 PRINTQP$:PRINTT$;"DU HAST GEWONNEN  ! !";:
GOTO4070
3640 N9=0:V1=0
3650 PRINTQP$:PRINTT$;"Ich denke nach  . . .";
3660 N1=1
3670 FORM4=1TO8:L=L(M4)+1:IFL>8THENNEXT:GOTO384
0
3680 V=1:P=O:W=0:M=M4
3690 GOSUB3960:FORZ1=1TO4:N(Z1)=0:NEXT
3700 FORZ=1TO4:S=S(Z):IFS<>3GOTO2880
3710 T=S+F(Z):IFT<4THENNEXT:GOTO3730
3720 V=V+4:N(S)=N(S)+1:NEXT
3730 FORI=1TO4:N=N(I)-1:IFN=-1THENNEXT:GOTO3750
3740 I1=8*W+4*SGN(N)+I:V=V+V(I1)+N*V(8*W+I):NEX
T
3750 IFW=1GOTO3770
3760 W=1:P=X:GOTO3690
3770 L=L+1:IFL>8GOTO3800
3780 GOSUB3960:FORZ=1TO4:IFS(Z)=3THENV=2
3790 NEXT
3800 IFV<V1THENNEXTM4:GOTO3840
3810 IFV>V1THENN1=1:GOTO3830
3820 N1=N1+1:IFRND(1)>1/N1THENNEXTM4:GOTO3840
3830 V1=V:M9=M4:NEXTM4
3840 IFM4<>0GOTO3870
3850 PRINTQP$:PRINTT$;"UNENTSCHIEDEN  ! ! ! !";
3860 GOTO4070
3870 M=M9
3880 PRINTQP$:PRINTT$;"Ich  ziehe  nach ";M;
3890 P=O:GOSUB4160:B(L,M)=O:GOSUB3960
3900 FORZ=1TO4
3910 IFS(Z)<4GOTO3940
3920 PRINTQP$:PRINTT$;"ICH HABE GEWONNEN  ! !";
3930 GOTO4070
3940 NEXT
3950 GOTO3530
3960 Q=X:IFP=XTHENQ=O
3970 D2=1:D1=0:Z=0:GOSUB3990:D1=1:D2=1:GOSUB399
0
3980 D2=0:D1=1:GOSUB3990:D2=-1:D1=1
3990 D=1:S=1:T=0:Z=Z+1
4000 C=1:M5=M:L1=L
4010 FORK=1TO3:M5=M5+D1:L1=L1+D2
4020 IFM5<1ORM5>8ORL1<1ORL1>8GOTO4050
4030 IFCANDB(L1,M5)=PTHENS=S+1:NEXT:GOTO4050
4040 C=0:IFB(L1,M5)<>QTHENT=T+1:NEXT
4050 IFDTHEND=0:D1=-D1:D2=-D2:GOTO4000
4060 S(Z)=S:F(Z)=T:RETURN
4070 FORO4=1TO4000:NEXT:PRINTQP$:PRINTT$;"Noch
ein Spiel ?"
4080 GOSUB4140:IFIN$="J"GOTO3430
4090 IFIN$<>"N"GOTO4080
4100 RUN6
4110 DOKE3300,25311:DOKE3302,312:DOKE3304,18351
4120 DOKE3306,10927:DOKE3308,-8179:POKE3310,233
4130 DOKE4100,3300:RETURN
4140 IN=USR(0):IFIN=0GOTO4140
4150 IN$=CHR$(IN):RETURN
4160 POKE2260+M*3,P
4170 L=L(M)+1:L(M)=L:FORO4=1TO2500:NEXT
4180 IFL=8THENRETURN
4190 FORI=7TOLSTEP-1
4200 POKE2708+M*3-I*64,45:POKE2772+M*3-I*64,P
4210 FORO4=1TO50:NEXTO4,I:RETURN
Ok
```

Hinderhalten-version: -8182



Hallo liebe Leser,

gegen Ende letzten Jahres ging es mir nicht sehr gut, denn die Vorstellung, vielleicht nie mehr dumme Sprüche machen zu dürfen, war mir garnicht angenehm. Dank Ihrer Unterstützung, die das Journal nun zumindest ein weiteres Jahr am Leben erhält, darf ich mich nun wie der Phönix aus der Asche erheben. Auch zu Ihrem Vorteil; denn wer sollte Ihnen in allen Lebenslagen mit Rat und Tat zur Seite stehen, wenn nicht Ihr NASCOMPL?

In diesem Sinne gegenseitig vielen Dank.

BLS Syntax

von GERHARD KLEMENT

```
10 REM -- BLS SYNTAX FILE R ---
20 REM  16.3.83
30 REM * DEN PASCAL BEGRIFF EINGEBEN
40 REM * DAS PROGRAMM FINDET DIE DEFINITIONEN
50 REM * ZB:
60 REM Suchstring? PROGRAM
70 REM <program heading> <block> .
80 :
90 REM Suchstring? PROGRAM HEADING
100 REM <empty> ö
110 REM PROGRAM ä <character> ü ;
120 :
130 REM  Suchstring? BLOCK
140 REM <declaration part> <statement part>
150 :
160 REM Suchstring? STATEMENT PART
170 REM <compound statement>
180 :
190 :
200 CLS:CLEAR15000:AN=85:DIMA$(AN)
210 D$="B L S  PASCAL SYNTAX"
220 FORI=1TOLEN(D$):POKE3027+I,ASC(MID$(D$,I,1)
)
230 NEXT:GOSUB430
240 :
250 PRINT:D$="":PRINT"Suchstring";:INPUTC$
260 FORI=1TOLEN(C$)
270 IFMID$(C$,I,1)=" "THEND$=D$+" ":GOTO290
280 D$=D$+CHR$(ASC(MID$(C$,I,1))+32)
290 NEXT
300 D$="<"+D$+">":F=0:FORI=0TOAN
310 IFLEFT$(A$(I),LEN(D$))=D$THENF=1
320 IFF=1THENDI=I:I=AN
330 NEXT:IFF=0THENPRINT"Not found":GOTO250
340 I=DI:FORJ=1TOLEN(A$(I))
350 IFMID$(A$(I),J,1)="<"THEND=J:J=LEN(A$(I))
360 NEXT
370 FORJ=D+6TOLEN(A$(I))
380 IFMID$(A$(I),J,1)="ö"THENPRINT"ö":GOTO400
390 PRINTMID$(A$(I),J,1);:GOTO410
400 J=J+1
410 NEXT:PRINT:GOTO250
420 :
430 A$(0)="<final value> ::= <expression>"
440 A$(1)="<initial value> ::= <expression>"
450 A$(2)="<for list> ::= <initial value> TO"
460 A$(2)=A$(2)+" <final value> ö <initial"
470 A$(2)=A$(2)+" value> DOWNTO <final value>"
480 A$(3)="<control variable> ::= <variable>"
490 A$(4)="<for statement> ::= FOR <control "
500 A$(4)=A$(4)+"variable> := <for list> DO "
510 A$(4)=A$(4)+"<statement>"
520 A$(5)="<repeat statement> ::= REPEAT <st"
530 A$(5)=A$(5)+"atement> ä ; <statement> ü"
540 A$(5)=A$(5)+" UNTIL <expression>"
550 A$(6)="<while statement> ::= WHILE <ex"
560 A$(6)=A$(6)+"pression> DO <statement>"
570 A$(7)="<repetitive statement> ::= <wh"
580 A$(7)=A$(7)+"ile statement> ö <repeat "
590 A$(7)=A$(7)+"statement> ö <for state"
600 A$(7)=A$(7)+"ment>"
610 A$(8)="<case list element> ::= <const"
620 A$(8)=A$(8)+"ant list> : <statement>"
630 A$(9)="<case list> ::= <case list elemen"
640 A$(9)=A$(9)+"t> ä ; <case list element> ü"
650 A$(10)="<case statement> ::= CASE <expr"
660 A$(10)=A$(10)+"ession> OF <case list> "
670 A$(10)=A$(10)+"END ö CASE <expression> "
680 A$(10)=A$(10)+"OF <case list> ; OTHERS: "
690 A$(10)=A$(10)+"<statement> END"
700 A$(11)="<if statement> ::= IF <expressi"
710 A$(11)=A$(11)+"on> THEN <statement> ö IF"
720 A$(11)=A$(11)+" <expression> THEN <state"
730 A$(11)=A$(11)+"ment> ELSE <statement>"
740 A$(12)="<conditional statement> ::= <if"
750 A$(12)=A$(12)+" statement> ö <case state"
760 A$(12)=A$(12)+"ment>"
770 A$(13)="<structured statement> ::= <com"
780 A$(13)=A$(13)+"pound statement> ö <cond"
790 A$(13)=A$(13)+"itional statement ö <rep"
800 A$(13)=A$(13)+"etitive statement>"
810 A$(14)="<empty statement> ::= <empty>"
820 A$(15)="<constant list> ::= <constant> ä"
830 A$(15)=A$(15)+" , <constant> ü"
840 A$(16)="<init statement> ::= INIT <array"
850 A$(16)=A$(16)+" identifier> TO <constant"
860 A$(16)=A$(16)+" list> ö INIT MEM ä <exp"
870 A$(16)=A$(16)+"ression> ü TO <constant"
880 A$(16)=A$(16)+" list>"
890 A$(17)="<goto statement> ::= GOTO <label>"
900 A$(18)="<procedure statement> ::= <pro"
910 A$(18)=A$(18)+"cedure identifier> <act"
920 A$(18)=A$(18)+"ual parameter list>"
930 A$(19)="<actual parameter> ::= <expressi"
940 A$(19)=A$(19)+"on> ö <variable> ö <array "
950 A$(19)=A$(19)+"identifier>"
960 A$(20)="<actual parameter list> ::= <empty"
970 A$(20)=A$(20)+"> ö ( <actual parameter> ä"
980 A$(20)=A$(20)+" , <actual parameter> ü )"
990 A$(21)="<function designator> ::= <fun"
1000 A$(21)=A$(21)+"ction identifier> <actual"
1010 A$(21)=A$(21)+" parameter list>"
1020 A$(22)="<unsigned constant> ::= <unsigned"
1030 A$(22)=A$(22)+" number> ö <string> ö <co"
1040 A$(22)=A$(22)+"nstant identifier>"
1050 A$(23)="<unsigned factor> ::= <variable> "
1060 A$(23)=A$(23)+"ö <unsigned constant> ö ("
1070 A$(23)=A$(23)+" <expression> ) ö <functi"
1080 A$(23)=A$(23)+"on designator>"
1090 A$(24)="<uncomplemented factor> ::= <uns"
1100 A$(24)=A$(24)+"igned factor> ö <sign> <"
1110 A$(24)=A$(24)+"unsigned factor>"
1120 A$(25)="<factor> ::= <uncomplemented fac"
1130 A$(25)=A$(25)+"tor> ö NOT <uncomplemented"
1140 A$(25)=A$(25)+" factor>"
1150 A$(26)="<multiplying operator> ::= * ö /"
1160 A$(26)=A$(26)+" ö DIV ö MOD ö AND ö SHIFT"
1170 A$(27)="<term> ::= <factor> ä <multiply"
1180 A$(27)=A$(27)+"ing operator> <factor> ü"
1190 A$(28)="<adding operator> ::= + ö - ö OR"
1200 A$(28)=A$(28)+" ö EXOR"
1210 A$(29)="<simple expression> ::= <term> ä"
1220 A$(29)=A$(29)+" <adding operator> <term> ü
"
1230 A$(30)="<relational operator> ::=  = ö <>"
1240 A$(30)=A$(30)+" ö < ö <= ö >= ö =="
1250 A$(31)="<expression> ::= <simple expressio
n>"
1260 A$(31)=A$(31)+"n> ö <simple expression> <"
1270 A$(31)=A$(31)+"relational operator> <simp"
1280 A$(31)=A$(31)+"le expression>"
1290 A$(32)="<function identifier> ::= <identi"
1300 A$(32)=A$(32)+"fier>"
1310 A$(33)="<array identifier> ::= <identifier
>"
1320 A$(34)="<component variable> ::= <array"
1330 A$(34)=A$(34)+" identifier> Ä <expression"
1340 A$(34)=A$(34)+"> ä , <expression> ü Ü"
1350 A$(35)="<simple variable> ::= <identifier>
"
1360 A$(36)="<variable> ::= <simple variable> ö
"
1370 A$(36)=A$(36)+" <component variable>"
1380 A$(37)="<assignment statement> ::= <va"
1390 A$(37)=A$(37)+"riable> := <expression> ö"
1400 A$(37)=A$(37)+" <function identifier> :="
1410 A$(37)=A$(37)+" <expression>"
1420 A$(38)="<simple statement> ::= <assign"
1430 A$(38)=A$(38)+"ment statement> ö <pro"
1440 A$(38)=A$(38)+"cedure statement> ö <goto"
1450 A$(38)=A$(38)+" statement> ö <init state"
1460 A$(38)=A$(38)+"ment> ö <empty statement>"
1470 A$(39)="<unlabelled statement> ::= <sim"
1480 A$(39)=A$(39)+"ple statement> ö <struct"
```

```
1490 A$(39)=A$(39)+"ured statement>"
1500 A$(40)="<statement> ::= ä <label>"
1510 A$(40)=A$(40)+" : ü <unlabelled state"
1520 A$(40)=A$(40)+"ment>"
1530 A$(41)="<compound statement> ::= BEGIN"
1540 A$(41)=A$(41)+" <statement> ä ; <state"
1550 A$(41)=A$(41)+"ment> ü END"
1560 A$(42)="<statement part> ::= <compound "
1570 A$(42)=A$(42)+"statement>"
1580 A$(43)="<result type> ::= <simple type>"
1590 A$(44)="<function heading> ::= FUNCTION"
1600 A$(44)=A$(44)+" <identifier> <formal pa"
1610 A$(44)=A$(44)+"rameter list> : <result "
1620 A$(44)=A$(44)+"type> ; ö FUNCTION <ident"
1630 A$(44)=A$(44)+"ifier> <formal parameter "
1640 A$(44)=A$(44)+"list> : <result type> ; "
1650 A$(44)=A$(44)+"<external/code specifica"
1660 A$(44)=A$(44)+"tion>"
1670 A$(45)="<function declaration> ::= <func"
1680 A$(45)=A$(45)+"tion heading> <block>"
1690 A$(46)="<code specification> ::= CODE "
1700 A$(46)=A$(46)+"<constant> ä , <constant>"
1710 A$(46)=A$(46)+" ü"
1720 A$(47)="<external specification> ::= EX"
1730 A$(47)=A$(47)+"TERNAL <constant>"
1740 A$(48)="<external/code specification> ::"
1750 A$(48)=A$(48)+"= <external specification"
1760 A$(48)=A$(48)+"> ö <code specification>"
1770 A$(49)="<parameter group> ::= <variable"
1780 A$(49)=A$(49)+" declaration>"
1790 A$(50)="<formal parameter part> ::= <par"
1800 A$(50)=A$(50)+"ameter group> ö VAR <par"
1810 A$(50)=A$(50)+"ameter group>"
1820 A$(51)="<formal parameter list> ::= <em"
1830 A$(51)=A$(51)+"pty> ö ( <formal paramet"
1840 A$(51)=A$(51)+"er part> ä ; <formal par"
1850 A$(51)=A$(51)+"ameter part> ü )"
1860 A$(52)="<procedure heading> ::= PROCED"
1870 A$(52)=A$(52)+"URE <identifier> <formal"
1880 A$(52)=A$(52)+" parameter list> ; ö PRO"
1890 A$(52)=A$(52)+"CEDURE <identifier> <form"
1900 A$(52)=A$(52)+"al parameter list> ; <ex"
1910 A$(52)=A$(52)+"ternal/code specificati"
1920 A$(52)=A$(52)+"on ;"
1930 A$(53)="<procedure declaration> ::= <"
1940 A$(53)=A$(53)+"procedure heading> <block>"
1950 A$(54)="<procedure or function declarati"
1960 A$(54)=A$(54)+"on> ::= <procedure decla"
1970 A$(54)=A$(54)+"ration> ö <function decl"
1980 A$(54)=A$(54)+"aration>"
1990 A$(55)="<procedure and function declara"
2000 A$(55)=A$(55)+"tion part> ::= ä <proce"
2010 A$(55)=A$(55)+"dure or function declarat"
2020 A$(55)=A$(55)+"ion> ; ü"
2030 A$(56)="<index type> ::= <constant> .."
2040 A$(56)=A$(56)+" <constant>"
2050 A$(57)="<structured type> ::= ARRAY Ä "
2060 A$(57)=A$(57)+"<index type> ä , <index"
2070 A$(57)=A$(57)+" type> ü Ü OF <simple "
2080 A$(57)=A$(57)+"type>"
2090 A$(58)="<string type> ::= STRING Ä <co"
2100 A$(58)=A$(58)+"nstant> Ü"
2110 A$(59)="<simple type> ::= INTEGER ö REA"
2120 A$(59)=A$(59)+"L ö BOOLEAN ö <string "
2130 A$(59)=A$(59)+"type>"
2140 A$(60)="<type> ::= <simple type> ö <stru"
2150 A$(60)=A$(60)+"ctured type>"
2160 A$(61)="<variable declaration> ::= <id"
2170 A$(61)=A$(61)+"entifier> ä , <identifie"
2180 A$(61)=A$(61)+"r> ü : <type>"
2190 A$(62)="<variable declaration part> ::="
2200 A$(62)=A$(62)+" <empty> ö VAR <variable"
2210 A$(62)=A$(62)+" declaration> ; ä <vari"
2220 A$(62)=A$(62)+"able declaration> ; ü"
2230 A$(63)="<string> ::= ' ä <character> ü '"
2240 A$(64)="<constant identifier> ::= <ident"
2250 A$(64)=A$(64)+"ifier>"
2260 A$(65)="<unsigned hexinteger> ::= $ <he"
2270 A$(65)=A$(65)+"xdigit> ä <hexdigit> ü"
2280 A$(66)="<sign> ::= + ö -"
2290 A$(67)="<scale factor> ::= <unsigned in"
2300 A$(67)=A$(67)+"teger> ö <sign> <unsign"
2310 A$(67)=A$(67)+"ed integer>"
2320 A$(68)="<unsigned real> ::= <unsigned i"
2330 A$(68)=A$(68)+"nteger> . <digit> ä <dig"
2340 A$(68)=A$(68)+"it> ü ö <unsigned integer"
2350 A$(68)=A$(68)+"> . <digit> ä <digit> ü"
2360 A$(68)=A$(68)+" E <scale factor> ö <uns"
2370 A$(68)=A$(68)+"igned integer> E <scale"
2380 A$(68)=A$(68)+" factor>"
2390 A$(69)="<unsigned number> ::= <unsigned"
2400 A$(69)=A$(69)+" integer> ö <unsigned rea"
2410 A$(69)=A$(69)+"l> ö <unsigned hexinteger>"
2420 A$(70)="<constant> ::= <unsigned number>"
2430 A$(70)=A$(70)+" ö <sign> <unsigned num"
2440 A$(70)=A$(70)+"ber> ö <constant identifi"
2450 A$(70)=A$(70)+"er> ö <sign> <constant "
2460 A$(70)=A$(70)+"identifier> ö <string>"
2470 A$(71)="<constant definition> ::= <iden"
2480 A$(71)=A$(71)+"tifier> = <constant>"
2490 A$(72)="<constant definition part> ::= "
2500 A$(72)=A$(72)+"<empty> ö CONST <constan"
2510 A$(72)=A$(72)+"t definition> ; ä <const"
2520 A$(72)=A$(72)+"ant definition> ; ü"
2530 A$(73)="<letter or digit> ::= <letter>"
2540 A$(73)=A$(73)+" ö <digit> ö ."
2550 A$(74)="<identifier> ::= <letter> ä <let"
2560 A$(74)=A$(74)+"ter or digit> ü"
2570 A$(75)="<unsigned integer> ::= <digit> "
2580 A$(75)=A$(75)+"ä <digit> ü"
2590 A$(76)="<label> ::= <unsigned integer> "
2600 A$(76)=A$(76)+"ö <identifier>"
2610 A$(77)="<label declaration part> ::= <"
2620 A$(77)=A$(77)+"empty> ö LABEL <label> ä"
2630 A$(77)=A$(77)+" , <label> ü"
2640 A$(78)="<declaration part> ::= <label "
2650 A$(78)=A$(78)+"declaration part> <const"
2660 A$(78)=A$(78)+"ant definition part> <va"
2670 A$(78)=A$(78)+"riable declaration part>"
2680 A$(78)=A$(78)+" <procedure and function"
2690 A$(78)=A$(78)+" declaration part>"
2700 A$(79)="<block> ::= <declaration part> "
2710 A$(79)=A$(79)+"<statement part>"
2720 A$(80)="<program heading> ::= <empty> ö"
2730 A$(80)=A$(80)+" PROGRAM ä <character> ü ;"
2740 A$(81)="<program> ::= <program heading> "
2750 A$(81)=A$(81)+"<block> ."
2760 A$(82)="<empty> ::= "
2770 A$(83)="<hexdigit> ::= <digit> ö A-F"
2780 A$(84)="<digit> ::= 0-9"
2790 A$(85)="<letter> ::= A-Z,a-z,Ö,_"
2800 RETURN
```

Im Programm werden verschiedene Klammern benötigt, die vom Drucker leider als Umlaute interpretiert werden. Deshalb hier die Tabelle zur Umsetzung:

ä = {
ö = |
ü = }
Ä = [
Ü =]

6502-Assembler

von RÜDIGER MAURER

Im Februar haben wir auf die Existenz eines 6502- Assemblers hingewiesen. Inzwischen wurde einiges Interesse aus dem Leserkreis geäußert, sodaß wir das Programm hiermit abdrucken.

```
5 REM ** (C) R.MAURER 15.10.82 **
6 :
10 CLS:SCREEN5,16:PRINT"6502 Assembler"
20 DIMMN$(256),BY(256),CO$(16)
30 FORE=0TO255:READMN$(E),BY(E):NEXT
40 FORE=0TO15:READCO$(E):NEXT
50 PRINT:PRINT"1 Assemblierung"
60 PRINT"2 Disassemblieren"
90 PRINT:INPUT"Bitte eingeben (1,2) ";A$
100 ONVAL(A$)GOSUB910,170
110 GOTO50
158 :
159 REM ** Umrechnung DEZIMAL - HEX fuer CODE
160 SX=INT(DC/16):UN=DC-(SX*16):SX$=CO$(SX)
165 UN$=CO$(UN):HX$=SX$+UN$:RETURN
168 :
169 REM ** Disassemblieren
170 INPUT"Start Adresse :";AD$:AD=VAL(AD$)
171 INPUT"End Adresse   :";ED$:ED=VAL(ED$)
172 INPUT"Druckerausgabe (J/N):";DR$
173 IFLEFT$(AD$,1)="#"THENOP$=AD$:GOSUB1280:AD=
OP
174 IFLEFT$(ED$,1)="#"THENOP$=ED$:GOSUB1280:ED=
OP
175 DR=1:IFLEFT$(DR$,1)="N"THENDR=0:I=0:GOTO190
176 IFDR=1THENGOSUB1500
180 IFI=14ANDDR=0THEN430
182 IFAD>EDANDDR=1THENGOSUB1600:RETURN
183 IFAD>EDTHENRETURN
190 I=I+1:IB=PEEK(AD):IFMN$(IB)<>"NULL"THEN240
200 DC=IB:GOSUB160:GOSUB880
210 PRINTAD;TAB(7);AD$TAB(14)HX$"=:";:AD=AD+1
215 IFDC>32ANDDC<96THENPRINTCHR$(DC);
220 PRINT:GOTO180
240 ONBY(IB)GOTO250,290,350
248 :
249 REM ** Ein - Byte - Befehl **
250 DC=IB:GOSUB160:GOSUB880
260 PRINTAD;TAB(7);AD$TAB(14)HX$TAB(25)MN$(IB)
270 AD=AD+1:GOTO180
288 :
289 REM ** Zwei - Byte - Befehl **
290 DC=IB:GOSUB160
300 B1$=HX$:DC=PEEK(AD+1):GOSUB160
310 B2$=HX$:GOSUB880:P=DC
320 PRINTAD;TAB(7);AD$TAB(14)B1$" "B2$TAB(25);
321 PRINTMN$(IB)TAB(30);:AD=AD+2:XX$=MN$(IB)
322 IFLEFT$(XX$,1)<>"B"THENPRINTP:GOTO180
324 IFLEFT$(XX$,3)="BIT"THENPRINTP:GOTO180
326 IFP>127THENP=P-256
328 P=AD+P:PRINTP
330 GOTO180
348 :
349 REM ** Drei - Byte - Befehl **
350 DC=IB:GOSUB160
360 B1$=HX$:DC=PEEK(AD+1):GOSUB160
370 B2$=HX$:DC=PEEK(AD+2):GOSUB160
380 B3$=HX$:OP=PEEK(AD+1)+(PEEK(AD+2)*256)
385 GOSUB880
390 PRINTAD;TAB(7);AD$TAB(14)B1$" "B2$" "B3$;
395 PRINTTAB(25)MN$(IB)TAB(30)OP:AD=AD+3
400 GOTO180
420 :
430 INPUT"Weiter = NEW LINE  Stop = S";A$
440 IFA$="S"THENI=0:PRINT:PRINT:RETURN
460 I=0:GOTO180
878 :
879 REM ** Umrechnung DEZIMAL - HEX der Adresse
880 A=AD:S3=INT(AD/4096):A=A-S3*4096
882 S2=INT(A/256):A=A-S2*256:S=INT(A/16)
890 U=AD-(S3*4096+S2*256+S*16):S3$=CO$(S3)
892 S2$=CO$(S2):S$=CO$(S):U$=CO$(U)
900 AD$=S3$+S2$+S$+U$:RETURN
908 :
910 PRINT:AD=3328:ZZ=3328:REM ** Default #0D00
911 PRINT"Beachte ORG - Anweisung !"
912 PRINT"Ablageadresse des Programms (Dez.)"
913 PRINT"Default = #0D00 (3328)"
914 PRINT"Hexeingabe mit '#' moeglich!":PRINT
915 GOSUB880
916 PRINT" Dez. Hex.   Op.- Code    Mnemonic"
919 SCREEN25,16:PRINT"ORG="AD"(#"AD$")"
920 GOSUB1160: REM  ** Eingabe
930 F=0
938 :
939 REM ** Bestimmung des Maschinencodes **
940 FORE=0TO255
950 IFMN$=MN$(E)THENBY=BY(E):F=1:CD=E:E=256
960 NEXT
970 IFF=0THEN1060
978 :
980 ONBYGOSUB1000,1010,1030
990 GOTO920
998 :
999 REM ** Ein - Byte - Befehl **
1000 PRINTCHR$(19);AD;:GOSUB880:PRINT"#";AD$;
1002 DC=CD:GOSUB160:PRINT"  ";HX$
1005 POKEAD,CD:AD=AD+1:RETURN
1008 :
1009 REM ** Zwei - Byte - Befehl **
1010 IFOP>255OROP<0THENPRINT"*   max. 255 !":RE
TURN
1020 PRINTCHR$(19);AD;:GOSUB880:PRINT"#";AD$;
1022 DC=CD:GOSUB160:PRINT"  ";HX$;
1023 DC=OP:GOSUB160:PRINT" ";HX$
1025 POKEAD,CD:POKEAD+1,OP:AD=AD+2:RETURN
1028 :
1029 REM ** Drei - Byte - Befehl **
1030 IFOP>65535OROP<0THENPRINT"*  max. 65535";R
ETURN
1040 POKEAD,CD:B2=INT(OP/256):B1=OP-(B2*256)
1041 PRINTCHR$(19);AD;:GOSUB880:PRINT"#";AD$;
1042 DC=CD:GOSUB160:PRINT"  ";HX$;
1043 DC=B1:GOSUB160:PRINT" ";HX$;
1044 DC=B2:GOSUB160:PRINT" ";HX$
1045 POKEAD+1,B1:POKEAD+2,B2:AD=AD+3
1050 RETURN
1058 :
1060 IFMN$="ORG"ORMN$="END"ORMN$="DC"THEN1080
1070 PRINT"Unbekannte Mnemonic !":GOTO920
1079 :
1080 IFMN$="ORG"THEN1100
1090 GOTO1120
1100 AD=OP:ZZ=OP:GOSUB880
1105 SCREEN25,16:FORI=1TO20:PRINT" ";:NEXT
1110 SCREEN25,16:PRINT"ORG ="OP"(#"AD$")"
1115 GOTO920
1120 IFMN$="END"THENEN=AD-1:RETURN
1150 POKEAD,OP:AD=AD+1:GOTO920
1158 :
1159 REM ** Mnemonic - Eingabe **
1160 INPUT"                              ";A$
1170 IFLEN(A$)<3THENPRINT"Zu kurze Eingabe!":GO
TO1160
1180 IFLEN(A$)=3THENMN$=A$:OP=0:RETURN
1188 :
1189 REM ** Abspaltung des Operanden OP **
1190 S=0:FORM=1TOLEN(A$)
1200 IFMID$(A$,M,1)=" "THENS=M:M=LEN(A$)
1210 NEXT
```

```
1220 IFS=0THENMN$=A$:RETURN
1230 MN$=LEFT$(A$,S-1)
1240 OP$=RIGHT$(A$,LEN(A$)-S)
1250 IFLEFT$(OP$,1)<>"#"THENOP=VAL(OP$):RETURN
1260 :
1270 REM ** Umrechnung HEX - DEZ **
1280 L=LEN(OP$)
1290 IFL<2ORL>5THENPRINT"Falscher Hexwert!";:RE
TURN
1300 FORI=1TO4:Z(1)=0:NEXTI:J=2
1310 FORI=6-LTO4:Q$=MID$(OP$,J,1)
1320 IFQ$="A"THENZ(I)=10:GOTO1400
1330 IFQ$="B"THENZ(I)=11:GOTO1400
1340 IFQ$="C"THENZ(I)=12:GOTO1400
1350 IFQ$="D"THENZ(I)=13:GOTO1400
1360 IFQ$="E"THENZ(I)=14:GOTO1400
1370 IFQ$="F"THENZ(I)=15:GOTO1400
1380 IFASC(Q$)<48ORASC(Q$)>57THENPRINT"Kein Hex
wert!":RETURN
1390 Z(I)=VAL(Q$)
1400 J=J+1:NEXTI
1410 OP=4096*Z(1)+256*Z(2)+16*Z(3)+Z(4)
1430 RETURN
1498 :
1499 REM ** Einschalten des Druckers **
1500 RESTORE11000:IR=3216:REM ** #C90 **
1520 READID:DOKEIR,ID:IR=IR+2
1530 IFIR<3256THEN1520
1540 DOKE4100,3216:REM ** Initialisierung #C90
**
1550 A=USR(0)
1560 DOKE4100,-27162:REM ** Druckrout. #95E6 **
1570 A=USR(0)
1580 RESTORE10000:RETURN
1598 :
1599 REM ** Abschalten des Druckers **
1600 DOKE3255,20191:DOKE3257,-55
1620 DOKE4100,3255:REM ** #CB7 **
1630 A=USR(0)
1650 RETURN
9999 :
10000 DATABRK,1,ORAIX,2,NULL,0,NULL,0,NULL,0
10010 DATAORAZ,2,ASL,2,NULL,0,PHP,1,ORAIM,2
10020 DATAASLA,1,NULL,0,NULL,0,ORA,3,ASL,3
10030 DATANULL,0,BPL,2,ORAIY,2,NULL,0,NULL,0
10040 DATANULL,0,ORAZX,2,SASLZX,2,NULL,0,CLC,1
10050 DATAORAY,3,NULL,0,NULL,0,NULL,0,ORAX,3
10060 DATASLX,3,NULL,0,JSR,3,ANDIX,2,NULL,0
10070 DATANULL,0,BITZ,2,ANDZ,2,ROLZ,2,NULL,0
10080 DATAPLP,1,ANDIM,2,ROLA,1,NULL,0,BIT,3
10090 DATAAND,3,ROL,3,NULL,0,BMI,2,ANDIY,2
10100 DATANULL,0,NULL,0,NULL,0,ANDZX,2,ROLZX,2
10110 DATANULL,0,SEC,1,ANDY,3,NULL,0,NULL,0
10120 DATANULL,0,ANDX,3,ROLX,3,NULL,0,RTI,1
10130 DATAEORIX,2,NULL,0,NULL,0,NULL,0,EORZ,2
10140 DATALSRZ,2,NULL,0,PHA,1,EORIM,2,LSRA,1
10150 DATANULL,0,JMP,3,EOR,3,LSR,3,NULL,0
10160 DATABVC,2,EORIY,2,NULL,0,NULL,0,NULL,0
10170 DATAEORZX,2,LSRZX,2,NULL,0,CLI,1,EORY,3
10180 DATANULL,0,NULL,0,NULL,0,EORX,3,LSRX,3
10190 DATA NULL,0,RTS,1,ADCIX,2,NULL,0,NULL,0
10200 DATANULL,0,ADCZ,2,RORZ,2,NULL,0,PLA,1
10210 DATAADCIM,2,RORA,1,NULL,0,JMPI,3,ADC,3
10220 DATAROR,3,NULL,0,BVS,2,ADCIY,2,NULL,0
10230 DATANULL,0,NULL,0,ADCZX,2,RORZX,2,NULL,0
10240 DATASEI,1,ADCY,3,NULL,0,NULL,0,NULL,0
10250 DATA ADCX,3,RORX,3,NULL,0,NULL,0,STAIX,2
10260 DATA NULL,0,NULL,0,STYZ,2,STAZ,2,STXZ,2
10270 DATA NULL,0,DEY,1,NULL,0,TXA,1,NULL,0
10280 DATA STY,3,STA,3,STX,3,NULL,0,BCC,2
10290 DATA STAIY,2,NULL,0,NULL,0,STYZX,2,STAZX,
2
10300 DATA STXZY,2,NULL,0,TYA,1,STAY,3,TXS,1
10310 DATA NULL,0,NULL,0,STAX,3,NULL,0,NULL,0
10320 DATA LDYIM,2,LDAIX,2,LDXIM,2,NULL,0,LDYZ,
2
10330 DATA LDAZ,2,LDXZ,2,NULL,0,TAY,1,LDAIM,2
10340 DATA TAX,1,NULL,0,LDY,3,LDA,3,LDX,3
10350 DATA NULL,0,BCS,2,LDAIY,2,NULL,0,NULL,0
10360 DATA LDYZX,2,LDAZX,2,LDXZY,2,NULL,0,CLV,1
10370 DATA LDAY,3,TSX,1,NULL,0,LDYX,3,LDAX,3
10380 DATA LDXY,3,NULL,0,CPYIM,2,CMPIX,2,NULL,0
10390 DATA NULL,0,CPYZ,2,CMPZ,2,DECZ,2,NULL,0
10400 DATA INY,1,CMPIM,2,DEX,1,NULL,0,CPY,3
10410 DATA CMP,3,DEC,3,NULL,0,BNE,2,CMPIY,2
10420 DATA NULL,0,NULL,0,NULL,0,CMPZX,2,DECZX,2
10430 DATA NULL,0,CLD,1,CMPY,3,NULL,0,NULL,0
10440 DATA NULL,0,CMPX,3,DECX,3,NULL,0,CPXIM,2
10450 DATA SBCIX,2,NULL,0,NULL,0,CPXZ,2,SBCZ,2
10460 DATA INCZ,2,NULL,0,INX,1,SBCIM,2,OP,1
10470 DATA NULL,0,CPX,3,SBC,3,INC,3,NULL,0
10480 DATA BEQ,2,SBCIY,2,NULL,0,NULL,0,NULL,0
10490 DATA SBCZX,2,INCZX,2,NULL,0,SED,1,SBCY,3
10500 DATA NULL,0,NULL,0,NULL,0,SBCX,3,INCX,3
10510 DATA NULL,0,0,1,2,3,4,5,6,7,8,9,A,B,C,D,E
,F
10998 :
10999 REM ** Unterprogrammdaten Druckroutine **
11000 DATA-6667,-194,4819,-11345
11010 DATA15890,-11265,15891,-11518
11020 DATA15891,-11519,8465,-27162
11030 DATA30754,8460,47,31522
11040 DATA-8436,-7851,-13839,-1
Ok
```

Seite(n) für Floppy-Einsteiger

von GÜNTER BÖHM

Nachlese

Nachdem nun schon etwa ein Drittel unserer Leser den Anschluß an unser Floppy- System mit der Zippel/Oberle Controller Karte gefunden haben, scheint es mir gerechtfertigt, in der Aufbauphase diesem Thema einen breiten Raum zu geben. Daß die Ausführungen viel Platz beanspruchen werden, kann ich schon jetzt abschätzen. Deshalb möchte ich mich bei den Lesern entschuldigen, die mit der Floppy nichts im Sinn haben. Ihnen zum Trost: Wenn die Floppy bei allen läuft, bieten die Disketten ein so bequemes Medium, daß wohl eine ganze Menge an Software auf diesem Wege transportiert wird, und nicht-interessierte Leser nicht mehr belästigt.
Wir haben schon jetzt damit begonnen und bieten eine Diskette an, die die folgenden Programme und noch einiges mehr enthält, sodaß man sich viel Tipparbeit ersparen kann. Mehr dazu in einer gesonderten Meldung in diesem Heft.
Zunächst einige wichtige Ergänzungen zu den Floppy- Seiten der letzten Hefte.
EMDOS: Bei der Benutzung mehrerer Laufwerke werden diese folgendermaßen selektiert (eingeklammerte Zeichen können weggelassen werden!):

D (A:) Directory Laufwerk A
D B Directory Laufwerk B (andere entsprechend)
(A:)FORMAT.COM Command Laufwerk A
B:FORMAT.COM Command Laufwerk B und andere
L (A:)BASIC Laden von Laufwerk A
L B:BASIC Laden von Laufwerk B etc.

Bitte beachten Sie, daß zwischen Befehl und Laufwerksnummer ein Space, zwischen Doppelpunkt und Laufwerksnummer kein Space stehen muß.

Verschiedentlich wurde nachgefragt, was denn eigentlich die Funktion des PHEAS sei. Deshalb hier nochmals: EMDOS ist ein Verwaltungsprogramm (ähnlich dem BDOS in CP/M), das rein logisch verschiedene Files nach bestimmten Regeln auf der Diskette "verteilt". Es verwaltet das Inhaltsverzeichnis und kontrolliert, daß die Files an die richtigen Stellen geladen werden. EMDOS arbeitet theoretisch mit jeder Floppy-Controller Karte. Es benötigt aber eine Schnittstelle zur Hardware, und eben diese wird durch PHEAS dargestellt. PHEAS teilt EMDOS jeweils mit, wie die Diskette in Spuren und Sektoren aufgeteilt ist und übernimmt die direkte physikalische Steuerung der Laufwerke (deshalb auch PHysikalisches Ein Ausgabe System).

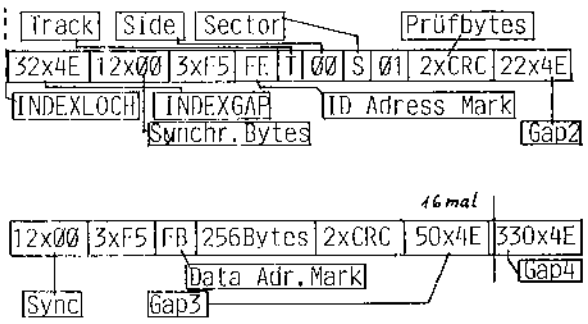
Bei den Testroutinen (Heft 9-83) wurde die Geschichte mit den Statusmeldungen des Controllers entweder mißverständlich formuliert oder aber zum Teil böswillig falsch verstanden. Zur Klärung:
Nach Lese- oder Schreiboperationen muß das Statusregister ØØ melden, dann wurden die Operationen richtig durchgeführt. Wird aber nur ein RESTORE (HOME) durchgeführt, lautet die Meldung Ø4 (=Kopf auf Spur Ø). Nach einem SEEK muß die Meldung 2Ø (=Kopf auf Diskette) gegeben werden. (Siehe Tabelle TypeI Commands Heft 9 Seite 8).
Nun aber endlich zu den aktuellen Meldungen über die Floppy, die mittlerweile mit Double Density problemlos läuft (wurde ja auch Zeit).

Double Density

Inzwischen ist das seit langem angeforderte Assemblerpaket zum mc- Computer angekommen, und ich konnte mich endlich mit den Aufzeichnungsformaten auseinandersetzen. Hier nun zunächst eine Beschreibung, wie sich die mc- Leute das Starten eines Betriebssystems (CP/M) vorstellen.
Hier existiert als erstes ein sogenanntes Urladeprogramm, das es ermöglicht, den ersten Sektor von Spur Ø einzulesen. Die Spur Ø ist dabei in Single Density (16 Sektoren zu 128 Bytes), kann also auch von Rechnern gelesen werden, die Double Density nicht verarbeiten können.
In diesem Sektor ist nun der Booter abgespeichert, der das Betriebssystem (CP/M) von Spur 1 bis 3 einlesen und im Rechner ablegen soll. Er enthält auch die Information, ob der Rest der Diskette in Single oder Double Density formatiert ist (ab Spur 1).

Das bedeutet, daß der Benutzer eines Rechners, der hardwaremäßig nur Single Density verarbeiten kann, zwar das Bootprogramm einlesen kann; wenn das Betriebssystem aber in doppelter Schreibdichte abgespeichert ist, mit diesem Bootprogramm nichts anfangen kann. In der Praxis: anstatt von vorne weg die Diskette nicht lesen zu können, gönnt man ihm das erfolgreiche Laden des Booters und läßt ihn dann um so frustrierter sitzen.

Für diesen minimalen Aufschub der Enttäuschung (die man sich sowieso ersparen könnte, wenn man weiß, daß die Diskette in DD formatiert ist) wird aber ein aufwendiges Formatierprogramm benötigt, das die erste Spur mit 128 Byte- Sektoren SD und den Rest in 256 Byte- Sektoren DD beschreibt. Der gleiche Aufwand muß beim Systemgenerieren (Sysgen), d.h. beim Abspeichern des Betriebssystems auf den ersten 4 Spuren getrieben werden.
Wir wollen aber doch möglicherweise mit mc-Computer Benutzern Programme tauschen. Wie ist das ohne diesen Programmieraufwand möglich?
Nun, die ersten 4 Spuren der Diskette interessieren beide Gruppen eigentlich garnicht, denn sie enthalten ja das Bootprogramm (wobei bei mc 15 Sektoren unbeschrieben und somit verschenkt sind), das von der Hardware abhängig ist (in erster Linie von der Controller-Karte) und das Betriebssystem, das als CP/M durch das BIOS ebenfalls hardwareabhängig ist und auf verschiedenen Rechnern nicht einfach ausgetauscht werden kann. Zudem wollen wir die Möglichkeit haben, auch CLD-DOS und NASSYS zu laden (Jetzt können wir ja "booten" sagen), die wiederum an die Hardware angepaßt sein müssen.
So können wir den "Standard" der ersten 4 Spuren vergessen, und wir legen nur fest, daß das Inhaltsverzeichnis auf Spur 4 Sektor 1 beginnt (wie bei mc) und die gleiche Länge hat. Damit haben wir mit den Hardware-formatierten Diskettenspuren nichts zu tun, können aber Files von der Diskette lesen, und unsere Files können gelesen werden.
Was man mit diesen Files dann anfangen kann, ist ein anderes Problem. Sind es CP/M-Programme, kann man auf alle Fälle damit arbeiten, auch Textfiles können gelesen werden.
Wie soll nun unser vereinfachtes Format aussehen?
Damit wir keine Konfusion mit der Sektorlänge bekommen, legen wir als Standard 256 Bytes pro Sektor fest. Damit bietet sich für Single Density das bereits in Heft 9/83 beschriebene Osborne-Format an. Als Formatierprogramm kann dabei weiter die Version aus Heft 10/11-83 dienen.
Da die Floppy-Karte ja nun auch mit DMA zum Laufen gebracht wurde (mehr dazu hoffentlich auch noch in diesem Heft), könnten aber alle Nachbauer (auch mit 2 MHz Systemtakt) mit Double Density arbeiten. Hier zunächst das mc-Format als Tabelle zum Vergleich mit den in Heft 9-83 Seite 8 veröffentlichten Formaten. Es entspricht weitestgehend dem IBM-Standard.



Ein Formatierprogramm das die mc- Norm benutzt, finden Sie im folgenden teilweise abgedruckt. Als Gerüst dazu dient das Programm aus Journal 10/11 Seite 16. Lediglich das Hauptprogramm "FORMAT" und die Format-Tabelle wurden neu geschrieben.
Ab Zeile 2070 müßte das Listing mit "TSTOK" fortgesetzt werden. In Zeile 2030 (des alten Listings) muß man "CP 16+1 ;16 Sektoren fertig?" eintragen.
Das vollständige Assemblerlisting können auf unserer Sammeldiskette finden.
Ich wäre froh, wenn alle Nachbauer unserer Floppy-Karte das DD For mat benützen würden, sonst muß beim Verschickenvon Software auf Diskette wieder zweispurig gefahren werden. Wir sollten genug Schwierigkeiten mit den NASCOM 1 und 2 Cassettenformaten gehabt haben.

FORMATIERPROGRAMM DD

```
80B3 CD0580     1430 FORMAT CALL INIT
80B6 EF         1440        RST  #28
80B7 0D0D       1450        DEFW #D0D
80B9 464F524D 1460          DEFM "FORMAT DOUBLE DENSITY 16 SEKT.
```

```
     41542044
     4F55424C
     45204445
     4E534954
     59203136
     2053454B
     542E
80D7 0D0D      1470        DEFW #D0D
80D9 4C415546 1480        DEFM "LAUFWERK A oder B ?
     5745524B
     2041206F
     64657220
     42203F
80EC 0D00      1490        DEFW #D
80EE DF7B      1500        DEFW #7BDF ;BLINK/ TASTATUREINGABE
80F0 F5        1510        PUSH AF
80F1 EF        1520        RST  #28 ;PRINT
80F2 0D        1530        DEFB #D
80F3 4C415546 1540        DEFM "LAUFWERK "
     5745524B
     20
80FC 00        1550        DEFB 0
80FD F1        1560        POP  AF
80FE F5        1570        PUSH AF
80FF F7        1580        RST  #30  ;PRINT A ASCII
8100 EF        1590        RST  #28
8101 203F2028 1600        DEFM " ? (Y/N)
     592F4E29
8109 0D00      1610        DEFW #D
810B DF7B      1620        DEFW #7BDF ;BLINK
810D FE59      1630        CP   "Y
810F 2802      1640        JR   Z SURE
8111 F1        1650        POP  AF
8112 C9        1660        RET  ;zu EMDOS ohne Ausführung
8113 F1        1670 SURE   POP  AF
8114 FE41      1680        CP   "A
8116 2004      1690        JR   NZ LFW2
8118 3E21      1700        LD   A,#21 ;BEACHTE LAUFWERKTAB.9/83
811A 1824      1710        JR   DRIVE
811C FE42      1720 LFW2   CP   "B
811E 2004      1730        JR   NZ LFW3
8120 3E22      1740        LD   A,#22
8122 181C      1750        JR   DRIVE
8124 EF        1760 LFW3   RST  #28
8125 0D        1770        DEFB #D
8126 4C415546 1780        DEFM "LAUFWERK NICHT VORHANDEN
     5745524B
     204E4943
     48542056
     4F524841
     4E44454E
813E 00        1790        DEFB 0
813F C9        1800        RET  ;EMDOS
8140 D310      1810 DRIVE  OUT  (PIOAD),A
               1820 ;
8142 3E03      1830        LD   A,FHOME
8144 D30F      1840        OUT  (FDCSTA),A ;HOME
8146 DF5D      1850        DEFW #5DDF ;NASSYS DELAY 1 SEC TDEL
8148 DB0F      1860        IN   A,(FDCSTA)
814A CB57      1870        BIT  2,A  ;KEIN LAUFWERK DA
814C 2004      1880        JR   NZ LFWOK
814E 3E1F      1890        LD   A,#1F
8150 180E      1900        JR   RETERR
8152 CB4F      1910 LFWOK  BIT  1,A
8154 2804      1920        JR   Z DISKOK
8156 3E10      1930        LD   A,#10 ;KEINE DISKETTE DA
8158 1806      1940        JR   RETERR
815A CB77      1950 DISKOK BIT  6,A
815C 281A      1960        JR   Z TSTOK
815E 3E15      1970        LD   A,#15 ;WRITE PROTECT
8160 F5        1980 RETERR PUSH AF
8161 EF        1990        RST  #28
8162 0D        2000        DEFB #D
8163 464F524D 2010        DEFM "FORMATIERFEHLER "
     41544945
     52464548
     4C455220
8173 00        2020        DEFB 0
8174 F1        2030        POP  AF
8175 DF68      2040        DEFW #68DF ;PRINT A
8177 C9        2050        RET  ;ZU EMDOS
```

```
33AD 074E      2570 TRKTAB DEFB 7,#4E ;HEADER
33AF 010E      2580        DEFB 1,#0E
33B1 074E      2590        DEFB 7,#4E
33B3 010E      2600        DEFB 1,#0E
33B5 074E      2610        DEFB 7,#4E
33B7 010E      2620        DEFB 1,#0E
33B9 074E      2630        DEFB 7,#4E
33BB 010E      2640        DEFB 1,#0E
33BD 00        2650        DEFB 0     ;RET
33BE 0C00      2660 TRKSTB DEFB 12,0
33C0 03F5      2670        DEFB 3,#F5
33C2 01FE      2680        DEFB 1,#FE
33C4 01        2690        DEFB 1
33C5 00        2700 FTRK   DEFB 0  ;TRACK
33C6 01        2710        DEFB 1
33C7 00        2720 FSIDE  DEFB 0 ;SIDE 0
33C8 01        2730        DEFB 1
33C9 00        2740 SECNR  DEFB 0 ;SEKTOR
33CA 0101      2750        DEFB 1,1    ;SEKTORLÄNGE 256
33CC 01F7      2760        DEFB 1,#F7  ;CRC
33CE 164E      2770        DEFB 22,#4E ;GAP
33D0 0C00      2780        DEFB 12,0   ;SYNCH
33D2 03F5      2790        DEFB 3,#F5
33D4 01FB      2800        DEFB 1,#FB  ;DA MARK
33D6 FFE5      2810        DEFB 255,#E5 ;DATA
33D8 01E5      2820        DEFB 1,#E5   ;DATA
33DA 01F7      2830        DEFB 1,#F7   ;CRC
33DC 324E      2840        DEFB 50,#4E  ;GAP
33DE 00        2850        DEFB 0       ;RET
33DF FF4E      2860        DEFB 255,#4E  ;PRE INDEX ca 330
33E1 914E      2870        DEFB 145,#4E
33E3 00        2880        DEFB 0       ;RET
               2890 ;
               2900 ;
33E4 0105090D 2910 SEKTAF DEFB 1,5,9,13,2,6,10,14,3,7,11,15
     02060A0E
     03070B0F
33F0 04080C10 2920        DEFB 4,8,12,16 ;SKEW FAKT. 3
               2930 ;
               2940 ; GANZE SPUR ZUM FORMATIEREN SCHREIBEN
```

COLD-BOOTER

Wir haben nun eine formatierte Diskette, und die Controller-Karte ist getestet. Wie bekommen wir nun aber das Betriebssystem und die Floppy-Verwaltung (EMDOS) ins RAM?
Eine Möglichkeit wäre der EPROM-Port von Jürgen Weiermann (Heft 10/11-83), wobei die Programme alle vom EPROM ins RAM kopiert werden könnten. Eine andere Möglichkeit wäre, alle Programme von der Diskette zu laden; aber wo bekommen wir das Ladeprogramm her? Es bleibt wohl nichts anderes übrig, als zumindest einen kleinen Teil des Speicherplatzes (wenn auch nur zeitweise) mit EPROM zu belegen, sodaß nach Einschalten des Gerätes eine Routine zur Verfügung steht, die eine wenn auch primitive Steuerung der Controller-Karte ermöglicht. (Man könnte selbstverständlich auch über Cassettenrecorder die Floppyroutinen einlesen, aber dafür würde ja wieder eine Cassettenroutine im EPROM gebraucht, und das ergäbe keinerlei Vorteil.

Wir wollen also eine Routine im EPROM haben, mit der wir ohne "fremde Hilfe" von der Diskette lesen können. Diese recht kleine Routine sei hier "Cold-Booter" genannt. Sie erkennt automatisch, ob es sich bei einer eingelegten Diskette um Single oder Double Density handelt, und liest (immer nur von Laufwerk A) die ersten beiden Sektoren der Spur ØØ. Die Routine kann auf jedem Z8Ø-Rechner laufen und ist von keinem Betriebssystem abhängig (wenn man die Fehlermeldung, die die NASSYS-Breakpointroutine benutzt, durch HALT ersetzt).

Das Cold-Boot Programm (wie erwähnt für beide Formate), ist im EPROM abgelegt oder wurde (wie in meinem mc-System) nach Einschalten oder Reset ins Ram kopiert. Es gäbe die Möglichkeit, ohne residenten Monitor nach dem Einschalten des Rechners das Betriebssystem direkt durch den Cold-Booter von der Diskette zu laden. Wenn nun aber irgend ein Fehler auftreten sollte, sind Sie ohne Monitor völlig hilflos bei der Fehlersuche. Deshalb empfiehlt es sich, auch einen (wenn auch kleinen) Monitor zur Verfügung zu haben. In meinem Fall (siehe Heft 6/83) stehen 4K EPROM zur Verfügung, in die NASSYS und der Cold-Booter bequem hineinpassen. Bei der NASCOM Grundplatine, die ja nur 2K EPROM ab ØØØØH faßt, sollte man vielleicht auf den alten Monitor T2 zurückgreifen und in den zweiten Sockel den Cold-Booter stecken. Für Besitzer der verschiedenen RAM/EPROM- Karten ergibt sich vielleicht eine andere Möglichkeit, neben NASSYS noch einen EPROM- Platz freizuhalten, der später ausgeblendet werden kann.

Ich für meinen Fall habe NASSYS3 so geändert, daß mit dem Drücken von "L" der Cold-Booter angesprungen wird. Im folgenden nochmals die Routine, die NASSYS3 und den Cold-Booter aus dem EPROM ins RAM lädt, NASSYS auf ØØ zurückkopiert und NASSYS initiiert.

```
                ØØ1Ø ;BOOTROM FUER mc-CP/M-COMPUTER
                ØØ2Ø ;KOPIERT NASSYS + FLOPPY-COLDBOOT INS
                ØØ3Ø ;RAM UND STARTET NASSYS
                ØØ4Ø ;DIE NASSYS MODIFIKATION TASTET NICHT
                ØØ5Ø ;DAS CASS.INTERFACE AB UND SPRINGT MIT
                ØØ6Ø ;Y ZU EMDOS, MIT L ZUM COLDBOOT
                ØØ7Ø ;
                ØØ8Ø ;VER 1.1 (MOD. JOURNAL 6-83)
                ØØ9Ø ;GUENTER BOEHM 31.12.83
                Ø1ØØ ;
22A7 Ø8ØØ       Ø11Ø NASLEN EQU  #8ØØ
22A7 Ø113       Ø12Ø CLDLEN EQU  #113
22A7 8ØØØ       Ø13Ø DEST   EQU  #8ØØØ
                Ø14Ø ;
ØØØØ            Ø15Ø        ORG  #ØØØØ
ØØØØ 21ØE ØØ    Ø16Ø RESET  LD   HL,NASSYS
ØØØ3 11ØØ8Ø     Ø17Ø        LD   DE,DEST
ØØØ6 Ø1ØØ1Ø     Ø18Ø        LD   BC,#1ØØØ ;ZUVIEL SCHADET NICHT
ØØØ9 EDBØ       Ø19Ø        LDIR ;COPY SYS RAM
ØØØB C3 1389    Ø2ØØ        JP   DEST+NASLEN+CLDLEN
                Ø21Ø ;
ØØØE Ø8ØØ       Ø22Ø NASSYS DEFS NASLEN
Ø8ØE Ø113       Ø23Ø        DEFS CLDLEN
                Ø24Ø ;
Ø921 3AØØ7Ø     Ø25Ø        LD   A,(#7ØØØ) ;EPROM ABSCHALTEN
Ø924 11ØØØØ     Ø26Ø        LD   DE,Ø ;NASSYS ADR.
Ø927 21ØØ8Ø     Ø27Ø        LD   HL,DEST
Ø92A Ø1ØØØ8     Ø28Ø        LD   BC,NASLEN
Ø92D EDBØ       Ø29Ø        LDIR
Ø92F C7         Ø3ØØ        RST  Ø
                Ø31Ø ;----------------------------------------

ZEAP Z8Ø Assembler - Source Listing

                Ø32Ø ;
                Ø33Ø ;
                Ø34Ø ;NASSYS MODIFIKATION
                Ø35Ø ;
                Ø36Ø ;#Ø77D ØØ  SERIELLER PORT NICHT ABGEFRAGT
                Ø37Ø ;#Ø798 ØØ 88  L=JP COLDBOOT
                Ø38Ø ;#Ø7B2 ØØ Al  Y=JP EMDOS
                Ø39Ø ;
                Ø4ØØ ;NASSYS wird ab ØØØE abgespeichert
                Ø41Ø ;CBOOT wird ab Ø8ØE abgespeichert
                Ø42Ø ;(lauffähig ab 88ØØH)
                Ø43Ø ;----------------------------------------

ZEAP Z8Ø Assembler - Source Listing

                ØØ1Ø ;----------------------------------------
                ØØ2Ø ;COLD-BOOT
                ØØ3Ø ;Urlader für Floppy-Karte 8Ø-Bus J.7/8-83
                ØØ4Ø ;im EPROM - Günter Böhm Vers.1.3 18.12.83
                ØØ5Ø ;
                ØØ6Ø ;Start durch NASSYS oder RESET je nach
                ØØ7Ø ;Hardware.
                ØØ8Ø ;Mit diesem Programm werden die ersten
                ØØ9Ø ;beiden Sektoren der Spur Ø geladen. Diese
                Ø1ØØ ;enthalten das Warm-Boot-Programm, das je
                Ø11Ø ;nach Version NASSYS/EMDOS, CLD-DOS oder
                Ø12Ø ;CP/M von der Diskette lädt.
                Ø13Ø ;----------------------------------------
8ØØØ            Ø14Ø        ORG  #8ØØØ
8ØØØ C3Ø688     Ø15Ø START  JP   START1
8ØØ3 ØØ         Ø16Ø        NOP
8ØØ4 F288       Ø17Ø INTVEK DEFW FLPINT ;Interrupt Tabelle
8ØØ6 31ØØ1Ø     Ø18Ø START1 LD   SP,#1ØØØ ;falls kein NASSYS da
8ØØ9 CD5688     Ø19Ø        CALL INIT ;Ports initiieren
8ØØC DBØC       Ø2ØØ        IN   A,(FDCSTA)   ;Motor ein
8ØØE CDE688     Ø21Ø        CALL DELAY3      ;1 sec Verzögerung
8Ø11 3EØ3       Ø22Ø        LD   A,FHOME
8Ø13 D3ØC       Ø23Ø        OUT  (FDCCMD),A  ;Restore Head (SpurØ)
8Ø15 CDE688     Ø24Ø        CALL DELAY3      ;1 sec Verzögerung
8Ø18 DBØC       Ø25Ø        IN   A,(FDCSTA)   ;Controller Status
  1A CB57       Ø26Ø        BIT  2 ,A         ;SpurØ ?
  1C 2821       Ø27Ø        JR   Z ERROR      ;nein
                Ø28Ø ;
  1E 3EØ1       Ø29Ø        LD   A,1          ;Sektor1
  2Ø D3ØE       Ø3ØØ        OUT  (FDCSEK),A
  22 21ØØ8C     Ø31Ø        LD   HL WBOOT     ;Sart des WBOOT-Speichers
  25 Ø1Ø1Ø5     Ø32Ø        LD   BC,#Ø5Ø1     ;5 Retries C=Flag Doub.D.
8828 CD9488     Ø33Ø RETRY1 CALL READ
882B B7         Ø34Ø        OR   A            ;gelesen?
882C 2812       Ø35Ø        JR   Z OKREAD     ;ja
882E 1ØF8       Ø36Ø        DJNZ RETRY1
8830 3E31       Ø37Ø        LD   A,#31        ;Single Density versuchen
8832 D31Ø       Ø38Ø        OUT  (PIOAD),A   ;für Laufwerk A
8834 Ø1ØØØ5     Ø39Ø        LD   BC,#Ø5ØØ      ;5 Retries C=Flag SD
8837 CD9488     Ø4ØØ RETRY2 CALL READ
883A B7         Ø41Ø        OR   A
```

```
883B 2803      0420        JR   Z OKREAD
883D 10F8      0430        DJNZ RETRY2
883F E7        0440 ERROR  RST  #20    ;BREAKPOINT (nur möglich,
               0450 ;                  wenn NASSYS vorhanden.
               0460 ;                  Sonst HALT
               0470 ;
8840 3E02      0480 OKREAD LD   A,2     ;Sektor2
8842 D30E      0490        OUT  (FDCSEK),A
8844 21008D    0500        LD   HL,WBOOT+256
8847 0605      0510        LD   B,5           ;Retries
8849 CD9488    0520 RETRY3 CALL READ
884C B7        0530        OR   A
884D 2804      0540        JR   Z END
884F 10F8      0550        DJNZ RETRY3
8851 18EC      0560        JR   ERROR
8853 C3008C    0570 END    JP   WBOOT ;Betriebssystem laden
               0580 ;-----------------------------------------
8856 000C      0590 FDCCMD EQU  0CH
8856 000E      0600 FDCSEK EQU  0EH
8856 000C      0610 FDCSTA EQU  0CH
8856 000F      0620 FDCDAT EQU  0FH.
               0630 ;
8856 0010      0640 PIOAD  EQU  10H
8856 0011      0650 PIOAC  EQU  11H
8856 0012      0660 PIOBD  EQU  12H
8856 0013      0670 PIOBC  EQU  13H
               0680 ;
8856 0003      0690 FHOME  EQU  03 ;RESTORE HEAD
8856 008C      0700 FREAD  EQU  8CH ;SEKTOR LESEN
8856 00D0      0710 FINT0  EQU  0D0H ;Reset FDC
               0720 ;-----------------------------------------
               0730 ;
8856 F3        0740 INIT   DI   ;PIOs u. FDC initiieren
8857 3ECF      0750        LD   A,0CFH
8859 D311      0760        OUT  (PIOAC),A ;CONTROLMODE
885B 3EC0      0770        LD   A,0C0H
885D D311      0780        OUT  (PIOAC),A ;I/O Maske
               0790 ;
885F 3ECF      0800        LD   A,0CFH
8861 D313      0810        OUT  (PIOBC),A ;CONTROLMODE
8863 3EF0      0820        LD   A,0F0H
8865 D313      0830        OUT  (PIOBC),A ;I/O Maske
               0840 ;
8867 3EB7      0850        LD   A,0B7H ;INTERRUPT CONTROL MODE
8869 D311      0860        OUT  (PIOAC),A
886B 3E7F      0870        LD   A,7FH
886D D311      0880        OUT  (PIOAC),A ;MASKE:BIT7 macht Interr.
               0890 ;
886F 3E08      0900        LD   A,8  ;FDC RÜCKSETZEN
8871 D310      0910        OUT  (PIOAD),A
8873 0E01      0920        LD   C,1
8875 CDD288    0930        CALL DELAY
8878 3E28      0940        LD   A,28H
887A D310      0950        OUT  (PIOAD),A
887C 3ED0      0960        LD   A,FINT0
887E D30C      0970        OUT  (FDCCMD),A
8880 E3        0980        EX   (SP),HL
8881 E3        0990        EX   (SP),HL
8882 DB0C      1000        IN   A,(FDCSTA)
8884 3E88      1010        LD   A,#88 ;INTERR.VECTOR MSB
8886 ED47      1020        LD   I,A
8888 3E04      1030        LD   A,04  ;LSB
888A D311      1040        OUT  (PIOAC),A
888C 3E21      1050        LD   A,21H ;LAUFWERK A zunächst Double D.
888E D310      1060        OUT  (PIOAD),A
8890 FB        1070        EI
8891 ED5E      1080        IM   2;INTERRUPT MODE
8893 C9        1090        RET
               1100 ;-----------------------------------------
8894 C5        1110 READ   PUSH BC
8895 E5        1120        PUSH HL
8896 CDB288    1130        CALL SAV66
8899 21CE88    1140        LD   HL,NMIR
889C CDC188    1150        CALL RE66HL
889F E1        1160        POP  HL
88A0 0E0F      1170        LD   C,FDCDAT
88A2 3E8C      1180        LD   A,FREAD
88A4 FB        1190        EI
88A5 E5        1200        PUSH HL
88A6 D30C      1210        OUT  (FDCCMD),A
88A8 18FE      1220 READW  JR   READW
88AA DB0C      1230        IN   A,(FDCSTA)
88AC CDBE88    1240        CALL RE66
88AF E1        1250        POP  HL
88B0 C1        1260        POP  BC
88B1 C9        1270        RET
               1280 ;-----------------------------------------
88B2 216600    1290 SAV66  LD   HL,66H ;RAMBEREICH 66H RETTEN
88B5 11CA88    1300        LD   DE,SAVE
88B8 010400    1310        LD   BC,4
88BB EDB0      1320        LDIR
88BD C9        1330        RET
               1340 ;
88BE 21CA88    1350 RE66   LD   HL,SAVE ;UND WIEDER HERSTELLEN
88C1 116600    1360 RE66HL LD   DE,66H
88C4 010400    1370        LD   BC,4
88C7 EDB0      1380        LDIR
88C9 C9        1390        RET
               1400 ;-----------------------------------------
88CA 0004      1410 SAVE   DEFS 4
               1420 ;-----------------------------------------
88CE ED45      1430 NMIR   RETN
               1440 ;-----------------------------------------
88D0 F5        1460 DELAY  PUSH HL  ;C=Anzahl msec
88D1 D5        1470        PUSH DE
88D2 C5        1480        PUSH BC
88D3 0664      1490 DELAY1 LD   B,100
88D5 17        1500 DELAY2 RLA
88D6 29        1510        ADD  HL,HL
88D7 29        1520        ADD  HL,HL
88D8 05        1530        DEC  B
88D9 C2D788    1540        JP   NZ,DELAY2
88DC 0D        1550        DEC  C
88DD C2D588    1560        JP   NZ,DELAY1
88E0 C1        1570        POP  BC
88E1 D1        1580        POP  DE
88E2 E1        1590        POP  HL
88E3 C9        1600        RET
               1610 ;
88E4 C5        1620 DELAY3 PUSH BC ;1 sec Verz.
88E5 0604      1630        LD   B,4         ;bei 4 MHz
88E7 0EFA      1640 DELAY4 LD   C,250
88E9 CDD288    1650        CALL DELAY
88EC 10F9      1660        DJNZ DELAY4
88EE C1        1670        POP  BC
88EF C9        1680        RET
               1690 ;-----------------------------------------
               1700 ; INTERRUPTROUTINE  AUSGELOEST VOM  FDC
               1710 ;
88F0 F5        1720 FLPINT PUSH AF
88F1 C5        1730        PUSH BC
88F2 E5        1740        PUSH HL
88F3 210600    1750        LD   HL,6
88F6 39        1760        ADD  HL,SP
88F7 4E        1770        LD   C,(HL)
88F8 23        1780        INC  HL
88F9 46        1790        LD   B,(HL)
88FA 0A        1800        LD   A,(BC)
88FB FE18      1810        CP   18H
88FD 2004      1820        JR   NZ,NOLOOP
88FF 03        1830        INC  BC
8900 0A        1840        LD   A,(BC)
8901 FEFE      1850        CP   0FEH
8903 2004      1860        JR   NZ,NOLOOP
8905 03        1870        INC  BC
8906 70        1880        LD   (HL),B
8907 2B        1890        DEC  HL
8908 71        1900        LD   (HL),C
8909 E1        1910 NOLOOP POP  HL
890A C1        1920        POP  BC
890B F1        1930        POP  AF
890C FB        1940        EI
890D ED4D      1950        RETI
               1960 ;-----------------------------------------
               1970 ;
               1980 ;BEACHTE: INTERRUPTVEKTOR MSB ZEILE 1010
               1990 ;BEI VERSCHIEBEN ÄNDERN
               2000 ;-----------------------------------------
               2010 ;
8911 8C00      2020 WBOOT  EQU  #8C00
```

WARM-BOOTER

Wird der Cold-Booter gestartet, so lädt er ein Programm von der Diskette, das das Betriebssystem wiederum von der Diskette liest und an geeigneter Stelle im Speicher ablegt. Das abgedruckte Beispiel WBOOT lädt NASSYS und EMDOS, wir werden aber noch Modifikationen veröffentlichen, die auch CLD-DOS und CP/M dorthin laden, wo sie hingehören.

Natürlich hätte man sich den Umweg über die Diskette sparen können und WBOOT (so soll das geladene Programm heißen) direkt vom Monitor starten können. Dann bräuchte man aber für jedes Betriebssystem ein eigenes Ladeprogramm im EPROM, und so kommen wir mit einem einzigen "Urlader" (CBOOT) aus; die speziellen Daten wie Sektorlänge und Speicherbereiche müssen nur auf der Diskette sein und sind somit flexibel und auf jedes Betriebssystem anwendbar.

```
ZEAP Z80 Assembler - Source Listing

                0010 ; W B O O T  DOUBLE DENSITY  V 1.4
                0020 ;abgespeichert in den ersten beiden
                0030 ;Sektoren der Spur 0
                0040 ;WBOOT lädt das Betriebssystem von der
                0050 ;Diskette. In diesem Beispiel wird Nassys
                0060 ;und EMDOS geladen. WBOOT muß für jedes
                0070 ;Betriebssystem angepaßt werden.
                0080 ;Sektoranzahl und Speicherbereiche müssen
                0090 ;dabei beachtet werden.
                0100 ;
                0110 ;VERS. 1.4 GUENTER BOEHM 27.12.83
                0120 ;Double Dense Vers. 2.2.84
8C00            0130        ORG  #8C00
                0140 ;
8C00 C3 0D 8C   0150 WBOOT  JP   START
8C03 00         0160        NOP
                0170 ;-------------------------------------
8C04 1E 8D      0180 INTVEK DEFW FLPINT ;INTERRUPT TABELLE
                0190 ;-------------------------------------
8C06 CD 80 8C   0200 START  CALL INIT
8C09 DB 0C      0210        IN   A,(FDCSTA); MOT. ON
8C0B CD 12 8D   0220        CALL DELAY3
8C0E CD 6E 8C   0230        CALL HOME
8C11 21 00 10   0240        LD   HL,#1000 ;        NASSYS darf nicht
                0250 ;nach 0000 geschrieben werden, da sonst
                0260 ;die NMI- Routinen überschrieben würden.
8C14 16 03      0270        LD   D,3 ;Sektornummer
8C16 1E 08      0280        LD   E,8 ;Sektoranzahl
8C18 AF         0290        XOR  A  ;Spur0
8C19 CD 40 8C   0300        CALL WREAD ;NASSYS laden
                0310 ;
8C1C 16 0B      0340        LD   D,11;AB SEKT. 11
8C1E 1E 06      0350        LD   E,6 ;NOCH 6 SEKTOREN
8C20 21 00 A1   0360        LD   HL,#A100 ;EMDOS Start
8C23 CD 40 8C   0370        CALL WREAD  ;EMDOS Part1 laden
                0380 ;
8C26 3E 01      0390        LD   A,1  ;next track
8C28 CD 76 8C   0400        CALL SEEK
8C2B 16 01      0410        LD   D,1  ;ab Sektor1
8C2D 1E 0A      0420        LD   E,10 ;noch 10 Sektoren
8C2F CD 40 8C   0430        CALL WREAD ;EMDOS Part2 laden
                0440 ;

8C32 21 00 10   0450        LD   HL,#1000 ;NASSYS Buffer
8C35 11 00 00   0460        LD   DE,0     ;NASSYS Destination
8C38 01 00 08   0470        LD   BC,#800  ;NASSYS Länge
8C3B EDB0       0480        LDIR ;NASSYS nach 0000 laden
8C3D C3 00 00   0490        JP   0  ;NASSYS STARTEN
                0500 ;
8C40 F5         0510 WREAD  PUSH AF ;TRACK retten
8C41 01 02 05   0520        LD   BC #0502 ;2X5 Retries
8C44 7A         0530        LD   A,D ;Sektor
8C45 D3 0E      0540        OUT  (FDCSEK),A ;programmieren
8C47 CD BE 8C   0550 WRETRY CALL READ
8C4A B7         0560        OR   A
8C4B 28 11      0570        JR   Z NXTRD ;Sektor gelesen
8C4D 10 F8      0580        DJNZ WRETRY
8C4F 0D         0590        DEC  C
8C50 28 0C      0600        JR   Z NXTRD ;keine Versuche mehr
8C52 CD 6E 8C   0610        CALL HOME  ;Restore Head
8C55 F1         0620        POP  AF
8C56 F5         0630        PUSH AF
8C57 CD 76 8C   0640        CALL SEEK
8C5A 06 05      0650        LD   B,5 ;nochmals 5 Versuche
8C5C 18 E9      0660        JR   WRETRY
                0670 ;
8C5E B7         0680 NXTRD  OR   A ;FDC STATUS
8C5F C2 3D 8D   0690        JP   NZ ERROR
8C62 F1         0700        POP  AF
8C63 D5         0710        PUSH DE
8C64 11 00 01   0720        LD   DE,256  ;Sektorlänge
8C67 19         0730        ADD  HL,DE
8C68 D1         0740        POP  DE
8C69 14         0750        INC  D ;next sector
8C6A 1D         0760        DEC  E
8C6B C8         0770        RET  Z ;alle Sekt. fertig
8C6C 18 D2      0780        JR   WREAD ;noch weiter laden
                0790 ;
                0800 ;---------------------------------------
8C6E 3E 03      0810 HOME   LD   A,FHOME
8C70 D3 0C      0820        OUT  (FDCCMD),A
8C72 FB         0830        EI
8C73 18 FE      0840 HOMEW  JR   HOMEW
8C75 C9         0850        RET
                0860 ;
8C76 D3 0F      0870 SEEK   OUT  (FDCDAT),A
8C78 3E 1B      0880        LD   A,FSEEK
8C7A FB         0890        EI
8C7B D3 0C      0900        OUT  (FDCCMD),A
8C7D 18 FE      0910 SEEKW  JR   SEEKW
8C7F C9         0920        RET
                0930 ;
                0940 ;---------------------------------------
8C80 000C       0950 FDCCMD EQU  0CH
8C80 000E       0960 FDCSEK EQU  0EH
8C80 000C       0970 FDCSTA EQU  0CH
8C80 000F       0980 FDCDAT EQU  0FH
                0990 ;
8C80 0010       1000 PIOAD  EQU  10H
8C80 0011       1010 PIOAC  EQU  11H
8C80 0012       1020 PIOBD  EQU  12H
8C80 0013       1030 PIOBC  EQU  13H
                1040 ;
8C80 0003       1050 FHOME  EQU  03 ;RESTORE HEAD
8C80 008C       1060 FREAD  EQU  8CH ;SEKTOR LESEN
8C80 00D0       1070 FINT0  EQU  0D0H ;Reset FDC
8C80 001B       1080 FSEEK  EQU  #1B ;SPUR SUCHEN
                1090 ;---------------------------------------
                1100 ;
8C80 F3         1110 INIT   DI   ;PIOs u. FDC initiieren
8C81 3E CF      1120        LD   A,0CFH
8C83 D311       1130        OUT  (PIOAC),A ;CONTROLMODE
8C85 3E C0      1140        LD   A,0C0H
8C87 D311       1150        OUT  (PIOAC),A ;I/O Maske
                1160 ;
8C89 3E CF      1170        LD   A,0CFH
8C8B D313       1180        OUT  (PIOBC),A ;CONTROLMODE
8C8D 3E F0      1190        LD   A,0F0H
8C8F D313       1200        OUT  (PIOBC),A ;I/O Maske
                1210 ;
8C91 3E B7      1220        LD   A,0B7H ;INTERRUPT CONTROL MODE
8C93 D311       1230        OUT  (PIOAC),A
8C95 3E 7F      1240        LD   A,7FH
8C97 D311       1250        OUT  (PIOAC),A ;MASKE:BIT7 macht Int
```

```
                1260 ;
8C99 3E08       1270        LD   A,8  ;FDC RÜCKSETZEN
8C9B D310       1280        OUT  (PIOAD),A
8C9D 0E01       1290        LD   C,1
8C9F CDFE8C     1300        CALL DELAY
8CA2 3E28       1310        LD   A,28H
8CA4 D310       1320        OUT  (PIOAD),A
8CA6 3ED0       1330        LD   A,FINT0
8CA8 D30C       1340        OUT  (FDCCMD),A
8CAA E3         1350        EX   (SP),HL
8CAB E3         1360        EX   (SP),HL
8CAC DB0C       1370        IN   A,(FDCSTA)
                1380 ;------------------------------
8CAE 3E8C       1390        LD   A,#8C ;INTERR.VECTOR MSB
8CB0 ED47       1400        LD   I,A
8CB2 3E04       1410        LD   A,04  ;LSB
8CB4 D311       1420        OUT  (PIOAC),A
                1430 ;------------------------------
8CB6 3F31       1440        LD   A,31H ;LAUFWERK A VERS. SINGLE DENSE
8CB8 D310       1450        OUT  (PIOAD),A
8CBA FB         1460        EI
8CBB ED5E       1470        IM   2;INTERRUPT MODE
8CBD C9         1480        RET
                1490 ;----------------------------------------
8CBE C5         1500 READ   PUSH BC
8CBF D5         1510        PUSH DE
8CC0 E5         1520        PUSH HL
8CC1 CDDE8C     1530        CALL SAV66
8CC4 21FA8C     1540        LD   HL,NMIR
8CC7 CDED8C     1550        CALL RE66HL
8CCA E1         1560        POP  HL
8CCB 0E0F       1570        LD   C,FDCDAT
8CCD 3E8C       1580        LD   A,FREAD
8CCF FB         1590        EI
8CD0 E5         1600        PUSH HL
8CD1 D30C       1610        OUT  (FDCCMD),A
8CD3 18FE       1620 READW  JR   READW
8CD5 DB0C       1630        IN   A,(FDCSTA)
8CD7 CDEA8C     1640        CALL RE66
8CDA E1         1650        POP  HL
8CDB D1         1660        POP  DE
8CDC C1         1670        POP  BC
8CDD C9         1680        RET
                1690 ;----------------------------------------
8CDE 216600     1700 SAV66  LD   HL,66H ;RAMBEREICH 66H RETTEN
8CE1 11F68C     1710        LD   DE,SAVE
8CE4 010400     1720        LD   BC,4
8CE7 EDB0       1730        LDIR
8CE9 C9         1740        RET
                1750 ;
8CEA 21F68C     1760 RE66   LD   HL,SAVE ;UND WIEDER HERSTELLEN
8CED 116600     1770 RE66HL LD   DE,66H
8CF0 010400     1780        LD   BC,4
8CF3 EDB0       1790        LDIR
8CF5 C9         1800        RET
                1810 ;----------------------------------------
8CF6 0004       1820 SAVE   DEFS 4
                1830 ;----------------------------------------
8CFA EDA2       1840 NMIR   INI
8CFC ED45       1850        RETN
                1860 ;----------------------------------------
8CFE E5         1870 DELAY  PUSH HL  ;C=Anzahl msec
8CFF D5         1880        PUSH DE
8D00 C5         1890        PUSH BC
8D01 0664       1900 DELAY1 LD   B,100
8D03 17         1910 DELAY2 RLA
8D04 29         1920        ADD  HL,HL
8D05 29         1930        ADD  HL,HL
8D06 05         1940        DEC  B
8D07 C2038D     1950        JP   NZ,DELAY2
8D0A 0D         1960        DEC  C
8D0B C2018D     1970        JP   NZ,DELAY1
8D0E C1         1980        POP  BC
8D0F D1         1990        POP  DE
8D10 E1         2000        POP  HL
8D11 C9         2010        RET
                2020 ;
8D12 C5         2030 DELAY3 PUSH BC ;1 sec Verz.
8D13 0604       2040        LD   B,4        ;bei 4 Miz
8D15 01EA       2050 DELAY4 LD   C,250
8D17 CDFE8C     2060        CALL DELAY
8D1A 10F9       2070        DJNZ DELAY4
8D1C C1         2080        POP  BC
8D1D C9         2090        RET
                2100 ;----------------------------------------
                2110 ; INTERRUPTROUTINE  AUSGELOEST VOM  FDC
                2120 ;
8D1E F5         2130 FLPINT PUSH AF
8D1F C5         2140        PUSH BC
8D20 E5         2150        PUSH HL
8D21 210600     2160        LD   HL,6
8D24 39         2170        ADD  HL,SP
8D25 4E         2180        LD   C,(HL)
8D26 23         2190        INC  HL
8D27 46         2200        LD   B,(HL)
8D28 0A         2210        LD   A,(BC)
8D29 FE18       2220        CP   18H
8D2B 200A       2230        JR   NZ,NOLOOP
8D2D 03         2240        INC  BC
8D2E 0A         2250        LD   A,(BC)
8D2F FEFE       2260        CP   0FEH
8D31 2004       2270        JR   NZ,NOLOOP
8D33 03         2280        INC  BC
8D34 70         2290        LD   (HL),B
8D35 2B         2300        DEC  HL
8D36 71         2310        LD   (HL),C
8D37 E1         2320 NOLOOP POP  HL
8D38 C1         2330        POP  BC
8D39 F1         2340        POP  AF
8D3A FB         2350        EI
8D3B ED4D       2360        RETI
                2370 ;----------------------------------------
8D3D E7         2380 ERROR  RST  #20 ;BREAKPOINT bei NASSYS
```

```
                0010 ; W B O O T  SINGLE DENSITY  V 1.4
                0020 ;abgespeichert in den ersten beiden
                0030 ;Sektoren der Spur 0

                0100 ;
                0110 ;VERS. 1.4 GUENTER BOEHM 27.12.83
                0120 ;
                0130        ORG  #8C00
                0140 ;
8C00 C30E8C     0150 WBOOT  JP   START
                0160        NOP
                0170 ;-------------------------------------
                0180 INTVEK DEFW FLPINT ;INTERRUPT TABELLE
                0190 ;-------------------------------------
8C0E CD1E8C     0200 START  CALL INIT
                0210        IN   A,(FDCSTA); MOT. ON
                0220        CALL DELAY3
                0230        CALL HOME
                0240        LD   HL,#1000 ;        NASSYS darf nicht
                0250 ;nach 0000 geschrieben werden, da sonst
                0260 ;die NMI- Routinen überschrieben würden.
                0270        LD   D,3 ;Sektornummer
                0280        LD   E,8 ;Sektoranzahl
                0290        XOR  A  ;Spur0
                0300        CALL WREAD ;NASSYS laden
                0310 ;
                0320        LD   A,1
                0330        CALL SEEK ;TRACK 1
                0340        LD   D,1 ;AB SEKT. 1
                0350        LD   E,10 ;GANZE SPUR
                0360        LD   HL,#A100 ;EMDOS Start
                0370        CALL WREAD  ;EMDOS Part1 laden
                0380 ;
                0390        LD   A,2  ;next track
                0400        CALL SEEK
                0410        LD   D,1  ;ab Sektor1
                0420        LD   E,6  ;noch 6 Sektoren
                0430        CALL WREAD ;EMDOS Part2 laden
                0440 ;
                0450        LD   HL,#1000 ;NASSYS Buffer
                0460        LD   DE,0     ;NASSYS Destination
                0470        LD   BC,#800  ;NASSYS Länge
                0480        LDIR ;NASSYS nach 0000 laden
                0490        JP   0  ;NASSYS STARTEN
```

NASGEN

Wenn Sie von uns die Systemdiskette beziehen, kann sie mit dem CBOOT direkt gebootet werden (allerdings müssen wir aus Copyrightgründen das NASSYS weglassen; das müssen Sie sich selbst irgendwie in den Speicher mogeln). Nun wollen Sie ja nicht nur von dieser Diskette abhängig sein, sondern auch nach Einschalten Ihres Rechners die Floppyroutinen und (falls nicht im EPROM vorhanden) das NASSYS von Ihren eigenen Disketten ins RAM "booten" können. Wie bekommen Sie aber die entsprechenden Systeme auf die äußersten Diskettenspuren (Spur ØØ bis Ø3 sind ja dafür vorgesehen)?
Zu diesem Zweck gibt es "NASGEN". NASSYS und EMDOS müssen sich an ihrem angestammten Speicherplatz befinden. Mit Aufruf von NASGEN.COM werden dann beide inclusive WBOOT auf die entsprechenden Systemspuren kopiert, und im Handumdrehen haben Sie Ihre eigene Systemdiskette generiert. Dabei können natürlich jede erdenklichen NASSYS Modifikationen gemacht worden sein, sodaß eine spezielle Diskette auch ein spezielles NASSYS bereithält. Für Tauschzwecke haben wir ja immer noch eine Diskette mit unserem original NASSYS zur Verfügung, mit dem auch fremde Programme laufen.

```
ZEAP Z80 Assembler - Source Listing

                ØØ1Ø ; N A S G E N
                ØØ2Ø ;PROGRAMM ZUM LADEN VON NASSYS, EMDOS
                ØØ3Ø ;UND WARMBOOT AUF DIE SYSTEMSPUREN
                ØØ4Ø ;
                ØØ5Ø ;VERS. 1.1  GUENTER BOEHM 3Ø.12.83
                ØØ6Ø ;-------------------------------------
                ØØ7Ø ;        SINGLE +  DOUBLE DENSITY
1ØØØ            ØØ8Ø        ORG  #1ØØØ
                ØØ9Ø ;
1ØØØ EF         Ø1ØØ        RST  #28
1ØØ1 ØD ØD      Ø11Ø        DEFW #DØD
1ØØ3 57454C43   Ø12Ø        DEFM "WELCHES LAUFWERK ?
     48455320
     4C415546
     5745524B
     203F
1Ø15 ØDØØ       Ø13Ø        DEFW #D
1Ø17 DF7B       Ø14Ø        DEFW #7BDF
1Ø19 F5         Ø15Ø        PUSH AF
1Ø1A EF         Ø16Ø        RST  #28
1Ø1B 4C415546   Ø17Ø        DEFM "LAUFWERK "
     5745524B
     20
1Ø24 ØØ         Ø18Ø        DEFB Ø
1Ø25 F1         Ø19Ø        POP  AF
1Ø26 F5         Ø2ØØ        PUSH AF
1Ø27 F7         Ø21Ø        RST  #3Ø ;PRINT A
1Ø28 EF         Ø22Ø        RST  #28
1Ø29 2Ø3F2Ø28   Ø23Ø        DEFM " ? (Y)
     5929
1Ø2F ØDØØ       Ø24Ø        DEFW #D
1Ø31 DF7B       Ø25Ø        DEFW #7BDF ;BLINK
1Ø33 FE59       Ø26Ø        CP   "Y
1Ø35 28Ø2       Ø27Ø        JR   Z SEL
1Ø37 F1         Ø28Ø        POP  AF
1Ø38 C9         Ø29Ø        RET
1Ø39 F1         Ø3ØØ SEL    POP  AF
1Ø3A 3D         Ø31Ø        DEC  A
1Ø3B E6ØF       Ø32Ø        AND  #F ;ASCII in HEX bis max. ØF
1Ø3D 4F         Ø33Ø        LD   C,A
1Ø3E 3EØ2       Ø34Ø        LD   A,2 ;PSEL CODE PHEAS
1Ø4Ø CD69A9     Ø35Ø        CALL PHEAS
                Ø36Ø ;-------------------------
1Ø43 118E1Ø     Ø37Ø        LD   DE,WBOOT
1Ø46 21ØØØØ     Ø38Ø        LD   HL,Ø ;PHYS. SEKTOR -1
1Ø49 3EØ4       Ø39Ø        LD   A,4 ;WRITE CODE
1Ø4B CD69A9     Ø4ØØ        CALL PHEAS
1Ø4E CD861Ø     Ø41Ø        CALL NXTSEK
1Ø51 23         Ø42Ø        INC  HL
1Ø52 3EØ4       Ø43Ø        LD   A,4
1Ø54 CD69A9     Ø44Ø        CALL PHEAS
                Ø45Ø ;-----------------
1Ø57 11ØØ2Ø     Ø46Ø        LD   DE,#2ØØØ ;BUFFER
1Ø5A 21ØØØØ     Ø47Ø        LD   HL,Ø     ;NASSYS
1Ø5D Ø1ØØØ8     Ø48Ø        LD   BC #8ØØ  ;NASLEN
1Ø6Ø EDBØ       Ø49Ø        LDIR ;NASSYS IN BUFFER
                Ø5ØØ ;
1Ø62 11ØØ2Ø     Ø51Ø        LD   DE,#2ØØØ
1Ø65 Ø6Ø8       Ø52Ø        LD   B,8      ;8 SEKTOREN
1Ø67 21Ø2ØØ     Ø53Ø        LD   HL,2     ;PHYS SEKTOR -1
1Ø6A 3EØ4       Ø54Ø WRNAS  LD   A,4 ;WRITE CODE
1Ø6C CD69A9     Ø55Ø        CALL PHEAS
1Ø6F CD861Ø     Ø56Ø        CALL NXTSEK
1Ø72 23         Ø57Ø        INC  HL
1Ø73 1ØF5       Ø58Ø        DJNZ WRNAS
                Ø59Ø ;
                Ø6ØØ ;--------------------
1Ø75 11ØØA1     Ø61Ø        LD   DE,#A1ØØ
1Ø78 Ø61Ø       Ø62Ø        LD   B,16  ;SEKTOREN
1Ø7A 3EØ4       Ø63Ø WRDOS  LD   A,4
1Ø7C CD69A9     Ø64Ø        CALL PHEAS
1Ø7F CD861Ø     Ø65Ø        CALL NXTSEK
1Ø82 23         Ø66Ø        INC  HL
1Ø83 1ØF5       Ø67Ø        DJNZ WRDOS
1Ø85 C9         Ø68Ø        RET
                Ø69Ø ;
                Ø7ØØ ;---------------------------------------
1Ø86 E5         Ø71Ø NXTSEK PUSH HL ;BUFFER DE WIRD UM EINEN
1Ø87 21ØØØ1     Ø72Ø        LD   HL,256     ;SEKTOR  WEITER GESETZT
1Ø8A 19         Ø73Ø        ADD  HL,DE
1Ø8B EB         Ø74Ø        EX   DE,HL
1Ø8C E1         Ø75Ø        POP  HL
1Ø8D C9         Ø76Ø        RET
                Ø77Ø ;---------------------------------------
1Ø8E A969       Ø78Ø PHEAS  EQU  #A969
1Ø9F Ø143       Ø79Ø WBOOT  DEFS #143 ;HIER MUSS WBOOT LIEGEN
                Ø8ØØ ; LAUFFAEHIG BEI #8CØØ
```

PHEAS MODIFIKATION

Wie Sie sehen, hat sich in der Zeit "zwischen den Jahren" einiges getan. Diese Entwicklung hat allerdings das PHEAS aus Heft 1Ø/11-83 nicht unbeschadet überstanden. Durch die Anpassung an Double Density haben sich einige Notwendigkeiten zur Änderung ergeben, und die häufige Benutzung hat einige Mängel aufgedeckt, die nun behoben wurden.
Bevor ich nun versuche, Ihnen die Änderungen rein theoretisch zu erklären, möchte ich lieber die geänderten Programmteile nochmals abdrucken, die Änderungen werden dann wohl besser verstanden. Das vollständige Assemb-

lerlisting ist ebenfalls auf unserer Systemdiskette enthalten.

Zunächst wurde der unbenutzte Akkuinhalt "1" beim Aufruf von PHEAS in einen Ansprung der Testroutine umfunktioniert, (Zeile 100). So kann diese auch von Hilfsprogrammen und nicht nur von PHEAS selbst benutzt werden. (siehe READTRK)

In die Select-Routinen zur Auswahl der Laufwerke wurden noch zwei Abspeicherungen für die Anzahl der Sektoren pro Spur und für die Sektoren pro Sustemspuren aufgenommen, damit die Umrechnung von logischen zu physikalischen Sektoren (ab Zeile 1490 ebenfalls geändert) für jedes Format durchgeführt werden kann. Bisher setzte die Umrechnung 30 Sektoren für das System voraus. Beim neuen Format mußte das geändert werden:

SD 4 Spuren a 10 Sektoren = 40

DD 4 Spuren a 16 Sektoren = 64

Die einzige Routine, die ein NASSYS-Unterprogramm verwendete (CTDEL Verzögerungsroutine) wurde durch ein Unterprogramm ersetzt, das die Routine DELAY von PHEAS benutzt. Dadurch ist PHEAS von jedem Betriebssystem unabhängig geworden und kann auf jedem Rechner laufen.(Zeile 1220).

Schließlich wurde in PINIT (Zeile 3750) die Initiierung der PIO vor dem Zugriff auf die Karte vorgenommen, damit sich nicht beide Laufwerke angesprochen fühlen. Auch hier wurde das NASSYS Delay entfernt.

ZEAP Z80 Assembler - Source Listing

```
                0010 ;-----------------------------------------
                0020 ;PHEAS-ANPASSUNG VER. 2.2  / 30.12.83
                0030 ;GUENTER BOEHM KARLSRUHE
                0040 ;nach Vorgaben v. H.Emmelmann 10/83
                0050 ;-----------------------------------------
A969            0060        ORG  #A969
A969 B7         0070 PHEAS  OR   A
A96A CA80AB     0080        JP   Z PINIT ;KARTE INITIIEREN
A96D 3D         0090        DEC  A
A96E CAE6A9     0100        JP   Z TSTHOM  ;LAUFWERKTEST
A971 3D         0110        DEC  A
A972 280C       0120        JR   Z, PSEL ;LAUFWERK SELEKTIEREN
A974 3D         0130        DEC  A
A975 CAFFA9     0140        JP   Z PREAD ;SEKTOR LESEN
A978 3D         0150        DEC  A
A979 CA6BAA     0160        JP   Z PWRITE;SEKTOR SCHREIBEN
A97C 37         0170        SCF
A97D C9         0180        RET
                0190 ;-----------------------------------------
A97E 5BAB       0200 INTVEK DEFW FLPINT ;INTERRUPT TABELLE
                0210 ;-----------------------------------------
A980 79         0220 PSEL   LD   A,C
A981 B7         0230        OR   A
A982 2810       0240        JR   Z PSELA
A984 FE01       0250        CP   1
A986 281E       0260        JR   Z PSELB
A988 FE02       0270        CP   2
A98A 282C       0280        JR   Z PSELC
A98C FE03       0290        CP   3
A98E 283A       0300        JR   Z PSELD
A990 3E17       0310        LD   A,#17
A992 37         0320        SCF
A993 C9         0330        RET
                0340 ;
A994 3E21       0350 PSELA  LD   A ,#21 ; LAUFW. A DD
A996 D310       0360        OUT  (10H),A
A998 21DCA9     0370        LD   HL,DSBA
A99B 3E10       0380        LD   A,16 ;Sekt.
A99D 32E2A9     0390        LD   (SEKT),A
A9A0 3E40       0400        LD   A,64
A9A2 32E4A9     0410        LD   (SYS),A
A9A5 C9         0420        RET
                0430 ;
                0440 ;
A9A6 3E22       0450 PSELB  LD   A,#22 ; LAUFWERK B DD
A9A8 D310       0460        OUT  (10H),A
A9AA 21DCA9     0470        LD   HL,DSBA ;gleiches Format wie Lw A
A9AD 3E10       0480        LD   A,16 ;Sekt.
A9AF 32E2A9     0490        LD   (SEKT),A
A9B2 3E40       0500        LD   A,64
A9B4 32E4A9     0510        LD   (SYS),A
A9B7 C9         0520        RET
                0530 ;
A9B8 3E31       0540 PSELC  LD   A,#31 ;Laufwerk A SD
A9BA D310       0550        OUT  (#10),A
A9BC 21DFA9     0560        LD   HL,DSBB
A9BF 3E0A       0570        LD   A,10 ;Sekt.
A9C1 32E2A9     0580        LD   (SEKT),A
A9C4 3E28       0590        LD   A,40
A9C6 32E4A9     0600        LD   (SYS),A
A9C9 C9         0610        RET
                0620 ;
A9CA 3E32       0630 PSELD  LD   A,#32 ;Laufwerk B SD
A9CC D310       0640        OUT  (#10),A
A9CE 21DFA9     0650        LD   HL,DSBB
A9D1 3E0A       0660        LD   A,10 ;Sekt.
A9D3 32E2A9     0670        LD   (SEKT),A
A9D6 3E28       0680        LD   A,40
A9D8 32E4A9     0690        LD   (SYS),A
A9DB C9         0700        RET
                0710 ;
A9DC 40         0720 DSBA   DEFB 64;4 X SEKTORANZAHL (res.System)
A9DD 08         0730        DEFB 8 ;8 Sekt. DIRECTORY
A9DE 8F         0740        DEFB 143 ;max.Gruppenanzahl-1
                0750 ;
A9DF 280857     0760 DSBB   DEFB 40,8,87 ;NEUES FORMAT mc !
A9E2 0000       0770 SEKT   DEFW 0       ;SEKTOREN PRO SPUR
A9E4 0000       0780 SYS    DEFW 0 ;SEKT. FUER SYSTEM
                0790 ;
A9E6 3E03       0800 TSTHOM LD   A,FHOME ;WARUM IST LAUFWERK
A9E8 F3         0810        DI   ;        NICHT BEREIT?
A9E9 D30C       0820        OUT  (FDCCMD),A
A9EB CD28AA     0830        CALL CTDEL
A9EE DB0C       0840        IN   A,(FDCSTA)
A9F0 CB57       0850        BIT  2,A
A9F2 2004       0860        JR   NZ NODISK
A9F4 3E1F       0870        LD   A,#1F ;KEIN LAUFWERK
A9F6 1805       0880        JR   TSTERR
A9F8 CB4F       0890 NODISK BIT  1,A
A9FA C8         0900        RET  Z
A9FB 3E10       0910        LD   A,#10 ;KEINE DISKETTE
A9FD 37         0920 TSTERR SCF
A9FE C9         0930        RET
                0940 ;
                0950 ;-----------------------------------------
A9FF C5         0960 PREAD  PUSH BC
AA00 E5         0970        PUSH HL
AA01 CD3AAA     0980        CALL PSEEK
AA04 D5         0990        PUSH DE
AA05 E1         1000        POP  HL ;BUFFER
AA06 010205     1010        LD   BC,#0502 ;2 X 5 RETRIES
AA09 CDE3AA     1020 RRETRY CALL READ
AA0C B7         1030        OR   A
AA0D 2813       1040        JR   Z OKREAD
AA0F 10F8       1050        DJNZ RRETRY
AA11 0D         1060        DEC  C
AA12 280E       1070        JR   Z OKREAD
AA14 CDCAAA     1080        CALL HOME
```

```
AA17 E1       1090        POP  HL ;SEKTORNUMMER
AA18 E5       1100        PUSH HL
AA19 CD3AAA   1110        CALL PSEEK
AA1C D5       1120        PUSH DE
AA1D E1       1130        POP  HL ;BUFFR
AA1E 0605     1140        LD   B,5 ;NOCH 5 RETRIES
AA20 18E7     1150        JR   RRETRY
AA22 E1       1160 OKREAD POP  HL
AA23 C1       1170        POP  BC
AA24 B7       1180        OR   A
AA25 200F     1190        JR   NZ RDERR
AA27 C9       1200        RET
              1210 ;---------------------------------------
AA28 F5       1220 CTDEL  PUSH AF
AA29 C5       1230        PUSH BC
AA2A 0602     1240        LD   B,2 ;1/2 sec Verz. bei 4 MHz
AA2C 0EFA     1250 DELAY4 LD   C,250
AA2E CD47AB   1260        CALL DELAY
AA31 10F9     1270        DJNZ DELAY4
AA33 C1       1280        POP  BC
AA34 F1       1290        POP  AF
AA35 C9       1300        RET
              1310 ;
              1320 ;---------------------------------------
AA36 3E11     1330 RDERR  LD   A,11H;Lesefehler
AA38 37       1340        SCF
AA39 C9       1350        RET
              1360 ;---------------------------------------
AA3A DB0C     1370 PSEEK  IN   A,(FDCSTA)
AA3C CB7F     1380        BIT  7,A ;READY?
AA3E C4E6A9   1390        CALL NZ TSTHOM
AA41 3004     1400        JR   NC PSE050
AA43 E1       1410        POP  HL ;RETURN ADDRESS
AA44 E1       1420        POP  HL ; "
AA45 E1       1430        POP  HL ; "
AA46 C9       1440        RET
AA47 C5       1450 PSE050 PUSH BC
AA48 CB7C     1460        BIT  7,H
AA4A 2807     1470        JR   Z PSE100 ;phys.=log. Sektornummer
AA4C CBBC     1480        RES  7,H      ;Bit 7 rücksetzen
AA4E ED4BE4A9 1490        LD   BC,(SYS) ;reserviert fuer System
AA52 09       1500        ADD  HL,BC ;phys.=relat.+Anzahl Syst.Sekt.
AA53 D5       1510 PSE100 PUSH DE
AA54 ED5BE2A9 1520        LD   DE,(SEKT);Sekt.pro Spur
AA58 3EFF     1530        LD   A,0FFH
AA5A B7       1540        OR   A ;Reset Carry
AA5B 45       1550 PSE200 LD   B,L ;Rest
AA5C ED52     1560        SBC  HL,DE ;DIVISION
AA5E 3C       1570        INC  A ;Ergebnis=Spur
AA5F 30FA     1580        JR   NC PSE200
AA61 D1       1590        POP  DE
AA62 CDD4AA   1600        CALL SEEK
AA65 78       1610        LD   A,B
AA66 3C       1620        INC  A ;Sekt.0-9=Sekt.1-10
AA67 C1       1630        POP  BC
AA68 C3E0AA   1640        JP   SETSEK
              1650 ;---------------------------------------
AA6B C5       1660 PWRITE PUSH BC
AA6C E5       1670        PUSH HL
AA6D CD3AAA   1680        CALL PSEEK
AA70 D5       1690        PUSH DE
AA71 E1       1700        POP  HL ;BUFFER
AA72 060A     1710        LD   B,10
AA74 CD03AB   1720 WRETRY CALL WRITE
AA77 B7       1730        OR   A
AA78 2802     1740        JR   Z OKWRIT
AA7A 10F8     1750        DJNZ WRETRY
AA7C E1       1760 OKWRIT POP  HL
AA7D C1       1770        POP  BC
AA7E B7       1780        OR   A
AA7F 2001     1790        JR   NZ WTERR
```

ZEAP Z80 Assembler - Source Listing

```
3B52 AA8C     3731 INIT   EQU  #AA8C
3B52 A9E6     3732 TSTHOM EQU  #A9E6
AB80          3733        ORG  #AB80
              3740 ;---------------------------------------
AB80 CD8CAA   3750 PINIT  CALL INIT
AB83 CDE6A9   3760        CALL TSTHOM
AB86 C9       3770        RET
              3780 ;---------------------------------
```

ZEAP Z80 Assembler - Symbol Table

```
AA28H 1220 CTDEL        AB47H 3280 DELAY
AB4AH 3310 DELAY1       AB4CH 3320 DELAY2
AA2CH 1250 DELAY4       A9DCH 0720 DSBA
A9DFH 0760 DSBB         0D0CH 1920 FDCCMD
0D0FH 1950 FDCDAT       0D0EH 1940 FDCSEK
0D0CH 1960 FDCSTA       0D0DH 1930 FDCTRK
0D03H 2030 FHOME        0D0DH 2080 FINT0
AB5BH 3460 FLPINT       AB7FH 3730 FLPSTA
0DE4H 2100 FRDTRK       0D8CH 2060 FREAD
0D1BH 2040 FSEEK        0D1FH 2050 FSEEKV
0DACH 2070 FWRITE       0DF4H 2090 FWRTRK
AACAH 2500 HOME         AACFH 2530 HOMEW
AA8CH 2130 INIT         A97EH 0200 INTVEK
AB3FH 3230 NMIR         AB43H 3250 NMIW
A9F8H 0890 NODISK       AB79H 3670 NOLOOP
AA22H 1160 OKREAD       AA7CH 1760 OKWRIT
A969H 0070 PHEAS        AB80H 3750 PINIT
0D11H 1990 PIOAC        0D10H 1980 PIOAD
0D13H 2010 PIOBC        0D12H 2000 PIOBD
A9FFH 0960 PREAD        AA47H 1450 PSE050
AA53H 1510 PSE100       AA5BH 1550 PSE200
AA3AH 1370 PSEEK        A980H 0220 PSEL
A994H 0350 PSELA        A9A6H 0450 PSELB
A9B8H 0540 PSELC        A9CAH 0630 PSELD
AA6BH 1660 PWRITE       AA36H 1330 RDERR
AB2FH 3150 RE66         AB32H 3160 RE66HL
AAE3H 2690 READ         AAF8H 2810 READW
AA09H 1020 RRETRY       AB23H 3090 SAV66
AB3BH 3210 SAVE         AAD4H 2580 SEEK
AADBH 2620 SEEKW        A9E2H 0770 SEKT
AAE0H 2660 SETSEK       A9E4H 0780 SYS
A9FDH 0920 TSTERR       A9E6H 0800 TSTHOM
AA74H 1720 WRETRY       AB03H 2890 WRITE
AB18H 3010 WRITEW       AA82H 1820 WTERR
AA8AH 1860 WTERR1
```

SKEW FACTOR

Beim Durchschauen des Double Density Formatierprogrammes werden Sie feststellen, daß die Sektortabelle in Zeile 2910 geändert wurde.

Da EMDOS nach dem Einlesen eines Sektors einige Rechenzeit benötigt, bevor der nächste Sektor gelesen werden kann, ist dieser Sektor in der Zwischenzeit schon am Lesekopf vorbeigerauscht, und es muß eine ganze Diskettenumdrehung gewartet werden, bis der entsprechende Sektor wieder am Kopf erscheint. Deshalb formatiert man die Sektoren am besten nicht in der Reihenfolge 1,2,3,4 etc. , sondern läßt in der Reihe eine bestimmte Anzahl von Sektoren aus, die der Rechenzeit angepaßt ist.

Ich habe diesen Skew-Faktor (Interleaving Factor) nun experimentell ermittelt. Als Grundlage diente ein Speicherbereich von 31 KByte (Beachten Sie, daß EMDOS maximal 32K

auf einmal verwalten kann, 31K + Directory-Eintrag ergeben dieses Maximum. Die neue Version soll 48K verwalten können) ,dessen Schreib- und Lesezeit abgestoppt wurde.
Ohne Skewfaktor und beim Auslassen von 1 und 2 Sektoren betrug die Zeit ca 31 Sekunden. Bei einem Faktor 3 wurde sie drastisch auf die Hälfte gesenkt. Größere Faktoren ließen die Zeit wieder ansteigen. Der im Formatierprogramm benutzte Faktor scheint also für EMDOS optimal zu sein.
Wenn wir die Tabelle betrachten, stellen wir fest, daß bei einer Diskumdrehung statt eines Sektors nun 4 Sektoren gelesen werden können.
Das Lesen eines Speicherbereichs von 1K dauert 8 Sekunden (vom Drücken der Taste bis zur Meldung auf dem Schirm), 31K benötigen 15 Sekunden. Man sieht, daß das Suchen im Directory und damit verbundene Verwaltungsarbeit, vor allem aber das Anlaufen des Motors und die Bereitschaft des Laufwerks eine Menge Zeit schlucken; das Lesen der Sektoren selbst geschieht recht schnell.
Das mc-Format sieht keinen Skew-Faktor vor. Dort werden hintereinander jeweils 4 Sektoren gelesen, ohne durch Rechenzeit zu unterbrechen. Die Geschwindigkeit bleibt so die gleiche, wenn vom mc-Rechner eine unserer Disketten gelesen wird. Umgekehrt müssen wir etwa die doppelte Lesezeit in Kauf nehmen. Das scheint mir kein großes Handikap.
Bei Single Density müßte folgende Tabelle mit dem Faktor 3 die günstigsten Ergebnisse bringen (ausprobiert habe ich sie nicht):
1 9 4 7 2 10 5 8 3 6

TIPS, TRICKS und KÄFER

Tape- Operationen (W+R), die zumeist einen wirren Screen- Salat verursachen, arbeiten sauber, wenn man bei NASSYS1 zuerst MC75 eingibt und anschließend 86. Bei NASSYS3 anstatt der 86 eine 80.
Die CLD-Banked- EPROMkarte von Lampson ist teilweise verkehrt beschriftet. Es fehlt die Kennzeichnung der Position der Tantal-Elkos, außerdem sind die DIL- Schalter genau entgegengesetzt bezeichnet (16=1 und 1=16).
Rolf Kottke, 1000 Berlin

IO-Karte

von KARL SCHULMEISTER

IO-Karte mit zwei Z80 PIO und einer Z80 SIO:

In MC 10/82 ist bei der Beschreibung des CP/M-Monitors eine Initialisierungs, sowie eine Testroutine für SIO und PIO abgedruckt. Möglicherweise (ich kann es mir aber nicht vorstellen) gibt es noch eine andere Ausgabe von SIO- und PIO- Bausteinen, da diese Routinen die Portadressen anders verwenden. So wird jeweils Port 1+3 als Data und 2+4 als Control programmiert (richtig 1+2 und 3+4), und es hat mich viel Zeit gekostet, den Fehler nicht auf meiner IO-Karte, sondern im Programm aus MC zu finden.

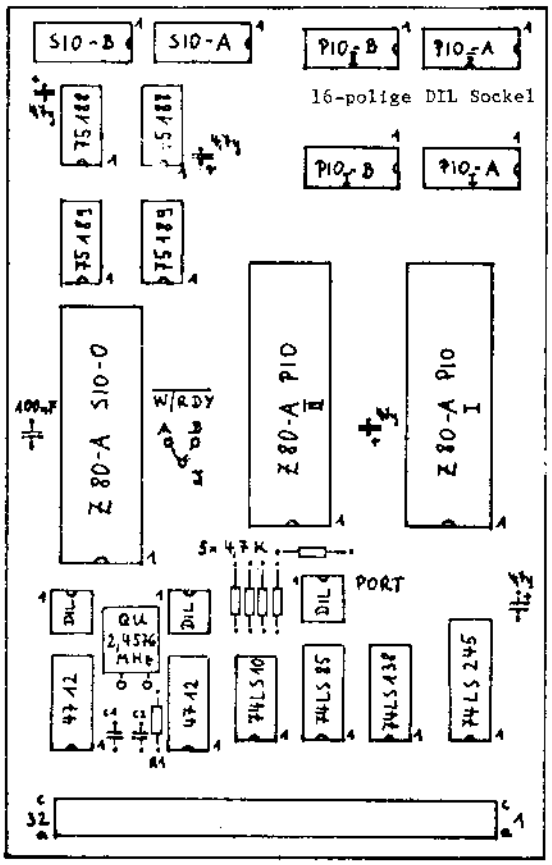
Quellenangabe: Nl- Schaltplan; Z80- Pio- und Z80- SIO Technical Manual; TI- Pocket- Guide Nr.1 und 2; mc Nr.10/82

Informationen zum Bestückungsplan:

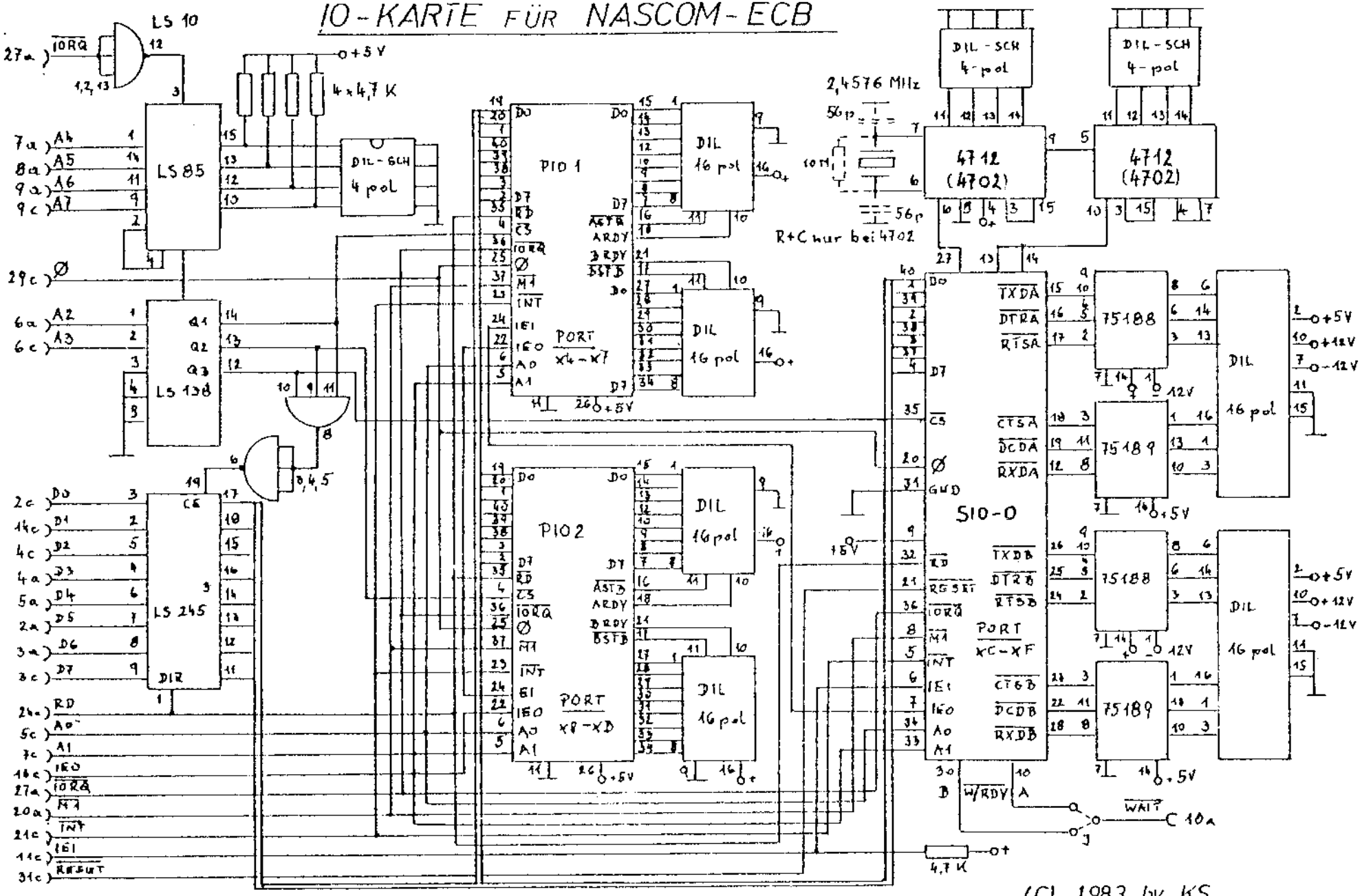
Portbelegung
PIO I Data: x4, x5
Cont: x6, x7
PIOII Data: x8, x9
Cont: xA, xB
SIO Data: xC, xD
Cont: xE, xF

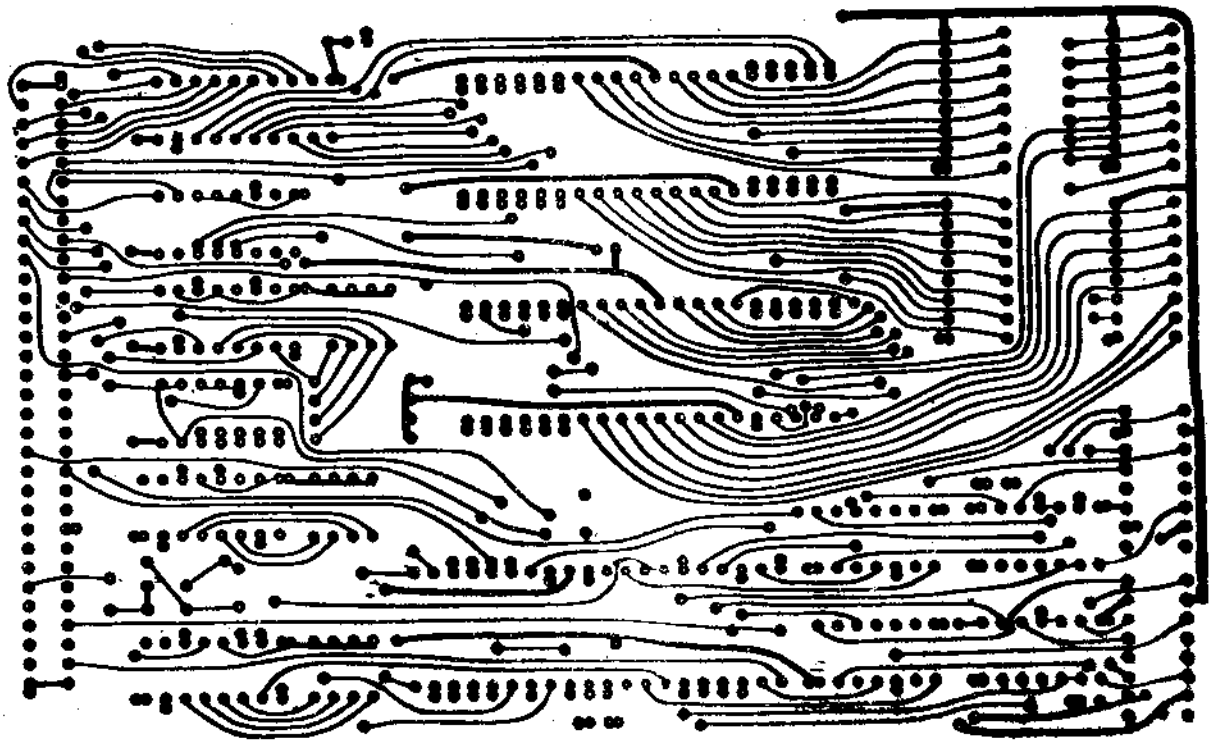
INT- Daisy Chain
SIO- PIOI- PIOII

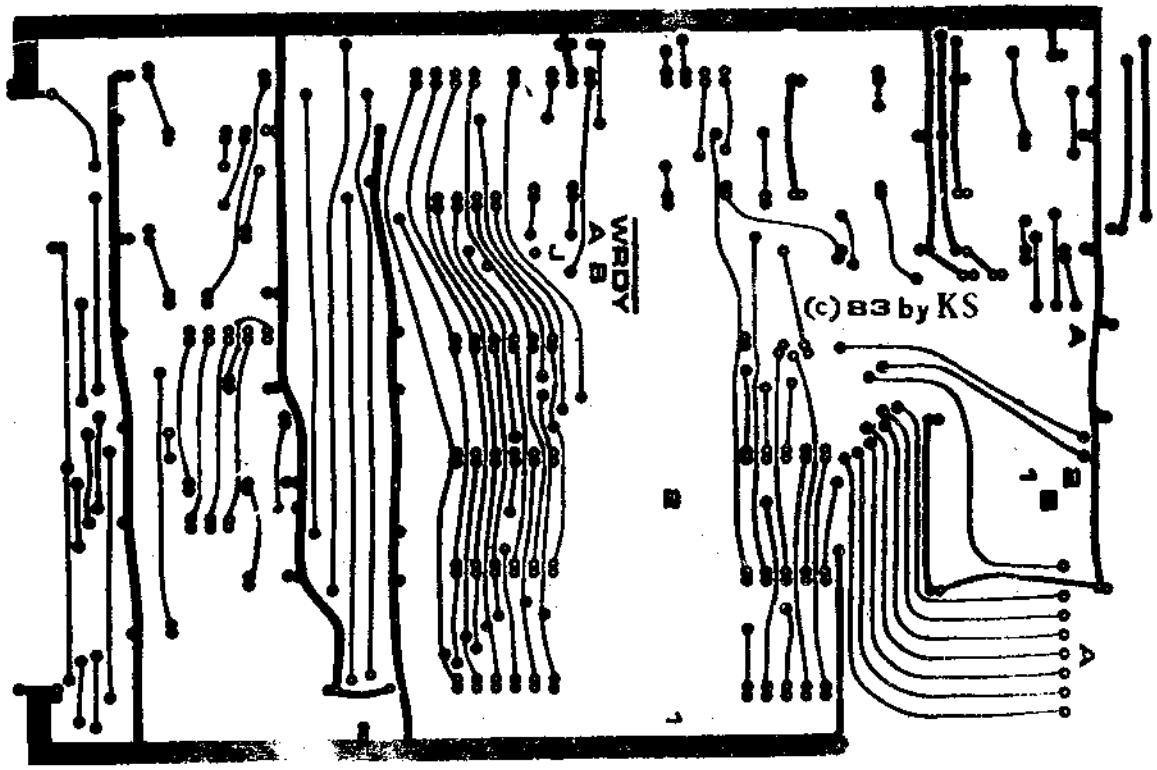
R1 (10 Mo) und C1, C2 (56 pF) nur einsetzen, wenn 4702 statt 4712 verwendet wird.



IO-KARTE FÜR NASCOM-ECB







IN/OUT- Karte NASCOM/ECB- Bus
(c) Karl Schulmeister, Klagenfurt

Grauwerte

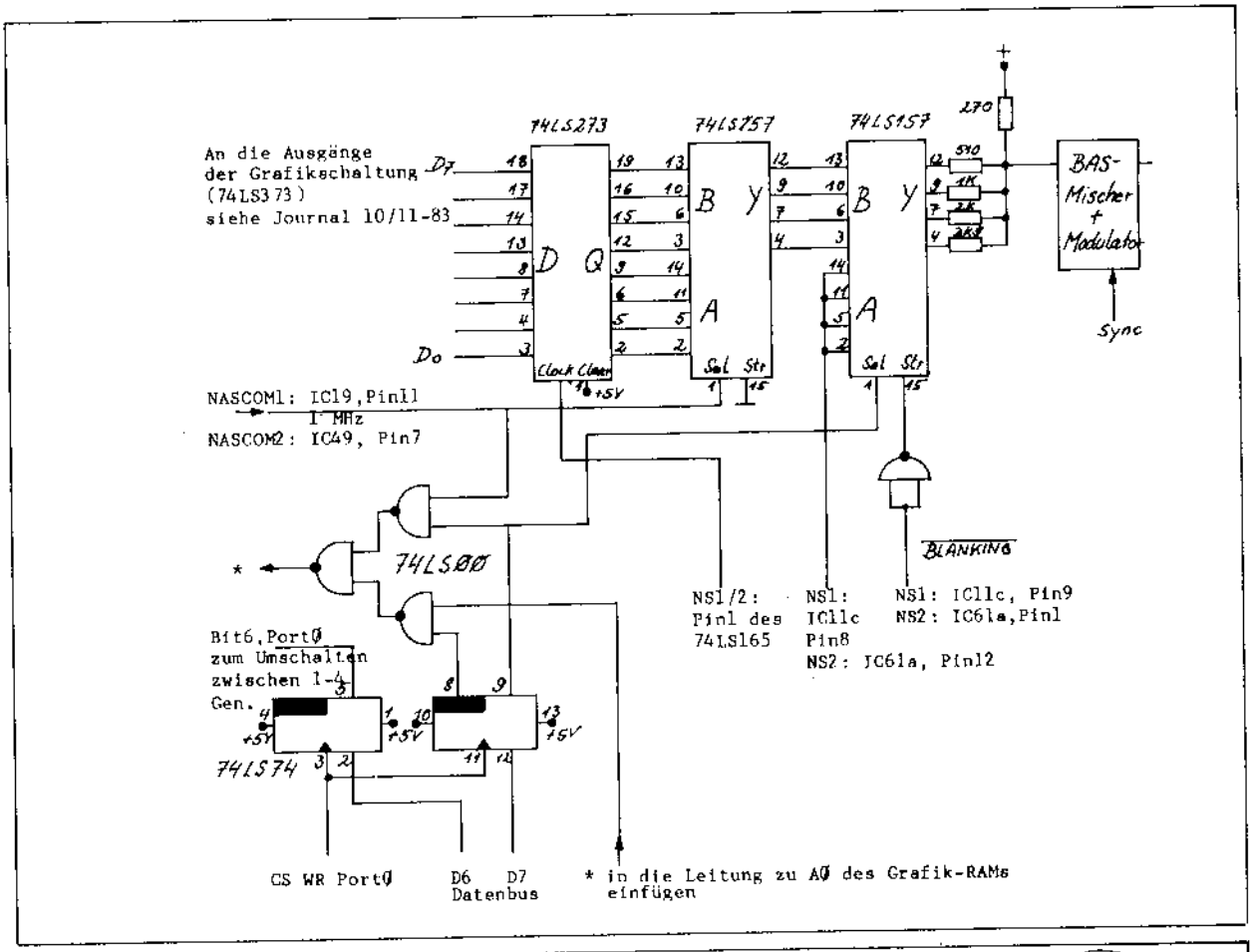
von JÖRG WITTICH

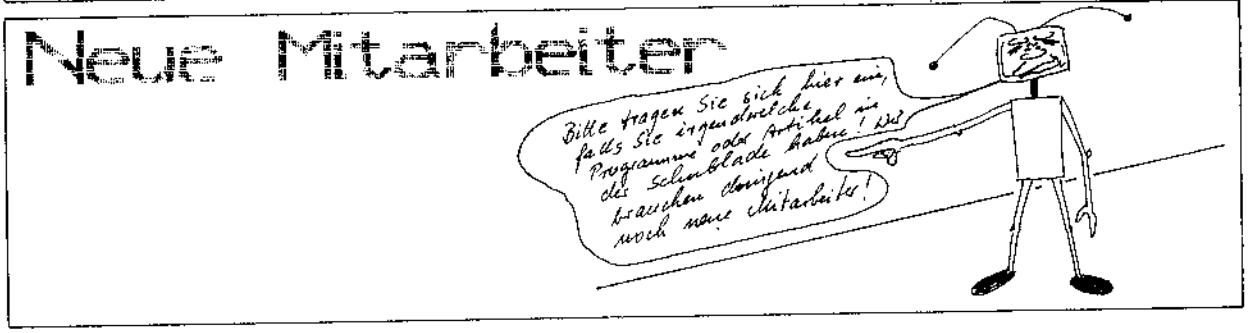
In Heft 10/11-83 stellten wir die preisgünstige hochauflösende Grafik von Jörg Wittich vor. Hier folgt nun die Erweiterung auf 16 Grauwerte, die wohl auch sehr gering im Hardwareaufwand ist. Die Schaltung bringt in einem SSTV- Programm hervorragende Ergebnisse.

Da diese Schaltung nicht an einen NASCOM sondern an eine Eigenkonstruktion angeschlossen ist, bleibt der BAS- Mischer und Modulator offen. Können wir (erprobte) Schaltungen aus dem Leserkreis erwarten?

Nachtrag zur Grafik Heft 10/11 Seite 22

Der Zeichengenerator des NASCOM1 hat keinen CS- Eingang. Man müßte deshalb, damit das ROM abgeschaltet werden kann, ein 2716 mit gleichem Inhalt benutzen (leider nicht Pin-kompatibel; siehe Grafikerweiterung 1982) oder dem Original-Generator ein 27244 nachschalten, was leicht auf der Grafikkarte geschehen kann. Beim NASCOM2 entfällt dieses Problem, da er ja einen 2716 benutzt.





Gemini Microcomputer

Vertriebs – GmbH

S O N D E R A N G E B O T E

solange der Vorrat reicht

RAM 'C' - Platine mit 64 KBytes, Bausatz	DM 450,--
EPROM 'B' - Platine, Bausatz	DM 330,--
SUPERMUM Erweiterung für NASCOM 1, Bausatz ohne Netzt.	DM 299,--
Paketpreis für NASCOM 1 Erweiterung, bestehend aus RAM 'C', EPROM 'B' und SUPERMUM	DM 998,--
EPROM - Programmiergerät für NASCOM oder GEMINI, programmiert 2708 und 2716 (5V), Bausatz einschl. Software (Betriebssystem angeben !)	DM 149,--
RTC Real Time Clock Bausatz, stellt über die PIO Urzeit und Datum zur Verfügung, Quarzgesteuert mit Akku einschl. Software	DM 149,--
BASIC ROM V 4.7 für NASCOM	DM 99,--
GRAFIK ROM für NASCOM	DM 47,--

Ersatzteile für NASCOM und GEMINI MULTIBOARD

Z80 CPU		DM 9,--
Z80 A CPU		DM 9,50
Z80 PIO		DM 6,50
Z80 A PIO		DM 9,--
Z80 A CTC		DM 12,--
UART 6402		DM 25,--
DIL Platform 16-pol.		DM 2,--
Tasten für Nascom 1/2 - Tastatur	10 Stück	DM 60,--
Centronics - Stecker (Weibchen)		DM 19,--

Achtung! Jetzt besonders günstig!

Original Gemini Floppydiskstation mit 2 Laufwerken (jeweils 350 KBytes form.), anschlußfertig im Gehäuse, mit Netzteil und Kabel einschl. Original Gemini FDC, fertig aufgebaut und getestet	DM 2800,--
Vero- Frame Einschubrahmen für 80-Bus Platinen	DM 189,--

Alle Preise einschl. ges. Mehrwertsteuer, zuzüglich Porto/Verpackung, Lieferung nur gegen Nachnahme.

Bitte fordern Sie unser neuestes 80 - Bus Info an !

Schluderstr. 10 • 8000 München 19

Tel. 089 / 168595

nascom

NASCOM - Sonderangebote

NASCOM-C, der neue Maßstab für CP/M -und 80-BUS-Systeme !
NASSYS-kompatibel und derzeit stärkstes CP/M-System !

* NASCOM-C mit Z80A CPU, 64KB RAM, MMU, 2xV24 und eine CENTRONCS-Schnittstelle, Videoteil, NUCLEOSYS.....DM 1.298,-
* NASCOM-C wie oben als Bausatz.....DM 998,-
* NASCOM-C Leerplatine mit Firmware & Dokumentation.DM 298,-
* NASCOM-C DMA-Floppy-Controller Option 5" oder 8"..DM 298,-
* Floppycontroller-Option für 5"+8" gemischt.....DM 348,-
* NASCOM-C 64KB-Erweiterung und Paritylogik.....DM 198,-
* Deutsche Tastatur für NASCOM-C (Cherry).....DM 198,-
* NASCOM-AVC Farbgrafik mit BASIC,-ASSEMBLER-und CP/M Softwareschnittstellen.....DM 798,-
* Hi-Res Farbmonitor für AVC (Zenith).....DM 1.998,-
* NASCOM-2a, NASCOM-2 mit 8KB CMOS-RAM ,ZEAP-Editor/Assembler und Microsoft-Basic in ROM als Bausatz.....DM 1.098,-
* 80-BUS, 4 Steckplätze, Busrahmen, Führungsleisten.DM 148,-
* CLD-Hardcontroller für hardsekt. Minidisketten..DM 498,-
* CLD-Softcontroller mit DMA, Echtzeituhr und Interface für Festplattenlaufwerke.....DM 998,-
* Softcontroller ohne DMA und CTC, Bausatz.....DM 698,-
* Softcontroller als Leerplatine + Firmware.....DM 198,-
* Minidiskettenlaufwerk BASF-6106, 200KB.....DM 498,-
* CLD-BANKED-Epromkarte für 16 Stück 2708 /16 /32, 2532 sowie 8KB ROMs in vier Banks, Bausatz.....DM 248,-
* Leerplatine Epromkarte mit Dok.....DM 148,-
* CLD-256KB-Ramkarte, Bausatz ohne DMA und Paritylogik mit 64KB RAM.....DM 698,-
* CLD-256KB-Ramkarte als Leerplatine mit Dok.....DM 148,-
* BLS-Pascal auf EPROM.....DM 298,-
* Page-Mode-Kit für LUCAS RAM-B Karte.....DM 79,-
* Grafik-Erweiterung für NASCOM-1 (ohne Grafgen.)...DM 98,-
* Grafik-Erweiterung für NASCOM-1 (mit Grafgen.)....DM 119,-
* CP/M 2.2 Betriebssystem mit ADM-31 Terminalemulator fuer AVC-Board, Screen-Editing auf CP/M Kommandoebene, Interface für Centronics-Drucker..DM 498,-
* BIOS-Anpasung bei Zusendung eines liz. CP/M.....DM 99,-

Alle Platinen mit Lötstopplack, vergoldeten Kontakten und Bestückungsdruck gefertigt und für alle NASCOM und GEMINI-Systeme verwendbar, alle Bausätze und neuen CLD-Platinen mit (gedrehten) Präzisionssockeln. FORTH -und BASIC EPROMS für NASCOM-C in Vorbereitung, desgleichen GSX für AVC & CP/M+

Preise inklusiv MwSt., exklusiv Versandkosten

LAMPSON Digitaltechnik
Odenwaldstrasse 21-23 Tel.: 06152/56730
6087 Büttelborn