# nascom journal

**Zeitschrift für Anwender des NASCOM 1 oder NASCOM 2**

3. Jahrgang | Juli/August 1982 | Ausgabe 7/8

**Herausgeber:**

**MK-SYSTEMTECHNIK** Michael Klein · Pater-Mayer-Straße 6 · 6728 Germersheim/Rhein
Telefon (0 72 74) 20 93 · Telex 453500 mks d

**MK-SYSTEMTECHNIK** Thomas Gräfenecker · Kriegsstraße 164 · 7500 Karlsruhe · Telefon (07 21) 2 92 43
**MK-SYSTEMTECHNIK** Michael von Keitz · Pfaffenberg 4 · 5650 Solingen 1 · Telefon (0 21 22) 4 72 67

Der Heftpreis beträgt DM 4,—. Ein Abonnement erhalten Sie für DM 48,— im Jahr. Dafür bekommen Sie 12 Hefte pro Jahr, bzw. 10 Hefte (zwei dicke Doppelausgaben).
Die Autoren sind für den Inhalt ihrer Beiträge selbst verantwortlich.

INHALT

# nascom journal

## Intern

Liebe Leser,
hier also nun die angekündigte "Jubiläumsausgabe". Sie ist wie versprochen sehr umfangreich geworden, und es ist bei der Vielfalt der Beiträge sicher für jeden Leser etwas dabei.
Gleichzeitig ist dies meine Abschiedsausgabe, denn ich werde mich von der Chefredaktion zurückziehen. Die Aufgaben im Journal haben sich im Laufe der letzten 14 Monate so vermehrt, daß sie mir nun endgültig über den Kopf gewachsen sind. Rein zeitlich ist das einfach nicht mehr zu packen. Deshalb mußten manche Leser in letzter Zeit auch auf eine persönliche Antwort oder die Rücksendung ihrer Cassette warten.
Günter Kreidl wird nun (in Zusammenarbeit mit seiner Frau) die Aufgaben der Chefredaktion übernehmen. Das heißt nicht, daß ich mich nun ganz vom Journal zurückziehen möchte. Ich werde weiterhin in der Redaktion tätig sein und bestimmte Teilbereiche übernehmen. So wird der Folienservice weiterhin über mich laufen, und auch der NASCOMPL wird Ihnen nicht erspart bleiben. Hauptsächlich werde ich mich nun wohl auf eigene Programme und Artikel stürzen.
Mein Rückzug aus der "Chefetage" soll allerdings nicht bedeuten, daß ich nun die Verbindung zu den Lesern abbrechen möchte, mit denen ich durch unsere Zusammenarbeit in engeren Kontakt gekommen bin. Im Gegenteil: ich würde mich freuen, auch weiterhin Gedanken auszutauschen; und wenn es sich um Beiträge für's Journal handelt, werde ich diese gerne weiterleiten.
Die Arbeit für das NASCOM Journal hat mir jedenfalls Spaß gemacht; wahrscheinlich werde ich in Zukunft sogar noch mehr Freude daran haben, wenn es für mich etwas ruhiger zugeht. Letztendlich verdanke ich dies der Mitarbeit der vielen aktiven Leser, wofür ich mich nun nochmals abschließend bedanken möchte. Aus welchem Holz die Journalleser geschnitzt sind, zeigte ja auch die Reaktion auf meine Klagen im letzten Journal, wegen der falsch deklarierten Preise für die Video-Folien. Jeder betroffene Besteller hat anstandslos nachgezahlt, und es blieb (dank mancher "Spenden") sogar noch etwas für die Portokasse übrig.
Mit der Gewißheit, daß die Clubatmosphäre, die sich inzwischen bei uns eingebürgert hat, weiterhin erhalten bleibt, gebe ich nun das Wort an Günter Kreidl.

Weiterhin viel Spaß mit dem Journal
Ihr Günter Böhm

---

*Liebe Leser,*
*wie Ihnen Günter Böhm auf dieser Seite bereits mitgeteilt hat, werde ich von der Septemberausgabe des NASCOM-Journals an seinen Arbeitsbereich übernehmen, während er sich anderen Aufgaben widmen wird. Da meine Freizeit recht knapp bemessen ist, ist dies nur möglich dank der Mithilfe meiner Frau und wenn auch Sie mir helfen, die Arbeit möglichst rationell zu bewältigen. Ich möchte Sie deshalb bitten, ALLE EINSENDUNGEN (auch Leserbriefe etc.) nur noch auf Cassette zu schicken. Gerne würde ich Ihnen versprechen, daß Sie diese Cassette bespielt mit allen Programmen der jeweiligen Ausgabe zurückbekommen, doch ich fürchte, daß ich mich damit übernehmen würde. Ich suche deshalb einen Helfer, der diese Aufgabe übernimmt. Er würde von mir nach Fertigstellung des Layouts alle eingesandten Cassetten erhalten, müßte dann die Programme im gewünschten Format auf die einzelnen Cassetten kopieren und diese verschicken (Portokosten werden ersetzt). Er müßte also beide Cassettenformate lesen und schreiben können. Wer Zeit und Lust hat, diese Arbeit zu übernehmen, soll mich doch bitte einmal anrufen!*
*Für die Zukunft erhoffe ich mir weiter Ihre rege Mitarbeit. Das NASCOM-Journal lebt von den Beiträgen der Leser, und auch der kleinste Beitrag ist willkommen. Ich hoffe, daß ich meine Aufgabe ebenso zu Ihrer Zufriedenheit erfüllen werde, wie dies sicherlich Günter Böhm im vergangenen Jahr getan hat.*
*Ihr Günter Kreidl*

## Ein Jahr Redaktionsarbeit

Als vor mehr als einem Jahr Michael Klein einen Redakteur für das NASCOM-Journal suchte, habe ich mich ebenso spontan gemeldet wie Günter Böhm, Wolfgang Mayer-Günn und Josef Zeller. Seither ist mehr als ein ganzer Jahrgang der "roten" Ausgaben unter unserer gemeinsamen Redaktion erschienen - vielleicht der geeignete Zeitpunkt für einen Rück- und Ausblick auf die Redaktionsarbeit.

*Wie das Journal gemacht wird*

Anfangs haben wir uns eine Arbeitsteilung überlegt: Herr Zeller war für Hardware-Beiträge zuständig, Herr Mayer-Günn für Basic und Floppies, Herr Böhm für T2/T4 und ich selbst für NAS-SYS. Je nach Ressort sollten die von den Lesern eingesandten Beiträge von den zuständigen Redakteuren bearbeitet (geprüft und ins Reine geschrieben) werden. Wegen der räumlichen Nähe zu MKS übernahm Herr Böhm die Zusammenstellung des Layouts. Diese Arbeitsteilung hat aber nicht lange vorgehalten: Fast alle Programme werden heute für NAS-SYS geschrieben, und die Textbeiträge kommen heute (Gott sei Dank!) zum größten Teil auf Cassette an und müssen nur noch formatiert und ausgedruckt werden. Diese Arbeit ruht ganz allein auf den Schultern von Herrn Böhm, der damit so eine Art Schaltstelle der Redaktion geworden ist. Die anderen Mitglieder des Redaktionsteams sind mehr Berater, Ideenlieferanten und vor allem "Pflichtautoren". Wenn man nämlich einmal so eine Aufgabe wie die Redaktion des NASCOM-Journals übernommen hat, beginnt man sich bald mit dem Produkt zu identifizieren und wünscht sich, daß die Zeitschrift eine formal und inhaltlich gute Sache wird. Wenn dann die Beiträge der Leser, von denen das Journal ja eigentlich lebt, nur spärlich eintrudeln, fangen die Sorgen an, ob man auch diesmal wieder ein ordentliches Heft zusammenbekommt. Mir geht es jedenfalls so, daß ich dann in meinen Unterlagen krame, ob ich nicht noch etwas für die Leser Interessantes finden kann, oder ich setze mich hin und schreibe ein Programm, daß ich eigentlich immer schon mal schreiben wollte, und mach dann gleich einen Artikel daraus. Wenn man sich einmal die letzten 10 oder 12 Ausgaben daraufhin ansieht, wie viele Beiträge von den Mitgliedern der Redaktion verfaßt wurden, dann sieht man, daß es ihnen wohl allen so geht wie mir. Kurz vor Redaktionsschluß (und manchmal auch erst später) geht das dann alles zum Günter Böhm, und bei dem ist wohl schon so manche Nacht mit der Zusammenstellung des Journals draufgegangen.

*Warum das NASCOM-Journal gemacht wird*

Warum steckt jemand seine Freizeit in so eine Sache wie das NASCOM-Journal? Da es hier um die persönliche Motivation geht, kann ich auch nur ganz persönlich antworten (vielleicht nehmen auch die anderen Mitarbeiter einmal dazu Stellung). Es ist heute so selbstverständlich geworden, daß man um des Geldes willen arbeitet, daß man sich andere Motive schon gar nicht mehr vorstellen kann. Welchen Sinn sehe ich nun in der Arbeit am NASCOM-Journal, welche Idee verbinde ich damit?

Als vor einigen Jahren die ersten Microcomputer auf dem Markt erschienen, habe ich mich zunächst aus ganz praktischen Gründen dafür interessiert. Wir waren damals mit dem Aufbau unseren Firma (Naturkost-Großhandelsgenossenschaft) beschäftigt, und ich fragte mich, ob wir so ein Ding nicht bei uns einsetzen könnten. Ich war dann sehr schnell von den Möglichkeiten, die sich da auftaten, fasziniert, habe mir Bücher und Zeitschriften beschafft und mich mit Programmiersprachen beschäftigt. Schließlich habe ich mir einen NASCOM-1 gekauft, um das auch praktisch auszuprobieren. Was mich von Anfang an dabei faszinierte, war die Idee des "Volkscomputers", des Computers für Jedermann. Ist es möglich, daß die "Computermacht", die bislang nur der Großindustrie, den Universitäten und dem Staat zur Verfügung stand, nun auch dem Kleinbetrieb und dem Privatmann offensteht? Von der Hardware her gesehen, ist diese Frage heute schon teilweise mit ja zu beantworten. Nur teilweise, weil es zwar unglaublich billige Grundsysteme gibt, aber die für den praktischen Einsatz nötigen Erweiterungen schnell den Betrag von 10000,- DM überschreiten. (Auf diesem Gebiet sind aber noch viele Möglichkeiten offen, z.B. die der "Zweckentfremdung" von billigen Konsumprodukten wie Stereocassettenrekordern und Videorekordern - als billige und schnelle Massenspeicher - und elektronischen Schreibmaschinen als Ein/Ausgabe-Geräten.) Man kann sich also sehr preiswert heute einen Computer kaufen und hat den dann zuhause rumstehen. Was macht man aber dann damit? Und wenn man schon weiß, was die Maschine tun soll, wie kriegt man sie dann dazu, das auch zu tun?

Es wird erst "Volkscomputer" geben, wenn auch

*"Jedermann" die Fähigkeit haben wird, damit umzugehen. Und damit kommen wir wieder zum NASCOM-Journal. Wenn die Benutzer eines Systems (oder ähnlicher Systeme) das Ergebnis ihren Bemühungen, ihre Programme und Hardwareentwicklungen, und ganz allgemein ihre Erfahrungen austauschen, dann potenziert sich der Lerneffekt. Das setzt natürlich die aktive Mitarbeit möglichst vieler Benutzen voraus. Die erste Frage ist damit aber noch nicht beantwortet und ich möchte Sie an die Leser weitergeben - hoffentlich gibt es viele Antworten: LEUTE, WAS MACHT IHR MIT EUREN COMPUTERN?*

*Günter Kreidl*

# LESERBRIEFE

LIEBER HERR BOEHM!
..... BEI DIESER GELEGENHEIT MOECHTE ICH NOCH AUF IHREN ARTIKEL "ROM BASIC V 4.7" IN NJ 1/82 EINGEHEN. UNSERE VERSION UNTERSCHEIDET SICH VON DER IHRIGEN BEI FFFE. BEI UNS STEHT HIER NICHT ØC, SONDERN AE, D.H. BEI WARMSTART WIRD BEI UNS ZUR ADR. EØAE GESPRUNGEN.
Max und Clemens Ballarin, Überlingen

Nachtrag zu FLIPPER (Journal 3/4-82)

1. Berichtigung:
Man kann ganz schön mogeln beim Flippern: Zeile 69Ø muß raus, sonst gibt es jedesmal 5 Punkte wenn man . oder , drückt. Dafür muß die Rechnung "Q(L)=Q(L)+5" ab Zeile 59Ø bis 68Ø hinter die IF-Anweisung geschrieben werden
59Ø IF PEEK(XXXX)=185 THEN D=72:Q(L)=..
etc
6ØØ IF PEEK(XXXX)=185 THEN D=72:Q(L)=Q(..
etc
etc
2. Das Programm läuft schneller (rechnerisch), wenn man für 185 einmal am Anfang "KU" wie Kugel definiert:
LET KU=185
und wenn man Zeile 47Ø und 48Ø in 395 und 396 ändert.
Peter Brendel

...Könnten Sie mal das Thema Störung/Entstörung behandeln? Mir ist nämlich die Post ganz schön aufs Dach gestiegen, weil ich die Kurzwelle vom Zoll gestört habe. Damals hatte ich kaum Abschirmungen montiert, inzwischen ist's besser, aber das schlechte Gewissen ist geblieben und ein bißchen Wut, daß einem niemand etwas darüber gesagt hat; z.B. der Händler!
Eine saubere, totale Entstörung ist auf jeden Fall ein Problem. Falls meine kümmerlichen Erfahrungen aber auch interessieren ( Sie tun's ! Red.), schicke ich gerne einmal einen Beitrag.
Herbert Grasl, 8993 Nonnenhorn

Bei mir läuft das ROM- BASIC 4.7 mit NASBUG T2. Dabei gibt es allerdings einige Schwierigkeiten.
Erstens läßt sich das CSAVE und CLOAD nicht verwenden, weil im ROM auf Adresse FE9F statt dem Loadprogramm das Dumpprogramm aufgerufen wird.
Zweitens funktioniert der Interruot (Software) nicht richtig (Shift+Newline). Das BASIC-Programm unterbricht zwar, aber bei erneutem Drücken wird die nächste Zeile abgearbeitet; das Programm kehrt nicht in den Kommandomodus zurück. Den Fehler habe ich noch nicht gefunden.
Karl Schrödinger, München
Daß sich Cassetten von BASIC aus mit T2 nicht laden oder lesen lassen ist bekannt. Der zweite Fehler ist wohl auch auf dieses Betriebssystem zurückzuführen. Kaufen Sie sich doch NASSYS 3, dann sind Sie alle Sorgen los und haben auch für Maschinenprogramme einen komfortableren Monitor. Red.

Wer hat Erfahrung mit dem Plotten mathematischer Funktionen? Meine Drucknadeln werden im Hex- Code angesteuert, wobei FF z.B. 8 senkrechte Punkte ergibt, also alle Nadeln aktiviert sind. Das LSB steuert dabei die oberen vier Nadeln, das MSB die unteren vier. Noch ein Beispiel: 11 ergäbe 1. und 5. Punkt von oben gesetzt...
Klaus Mombaur, 85ØØ Nürnberg

Ich finde es ja äußerst positiv, daß sich überhaupt jemand die Mühe gemacht hat, für den NASCOM eine Floppy zu entwickeln; aber vielleicht könnte man von Seiten des Journals her Herrn Lampson bitten, den Controller doch auch in Form einer leeren Platine plus Unterlagen und Software zu vertreiben. Mir wurde vor einem Jahr eine entsprechende Anfrage negativ beschieden. Ich bin jedoch überzeugt, daß die NASCOM Hardware Freaks ein solches Angebot sehr begrüßen würden.
Stefan Bürger, 7146 Tamm-Hohenstange

Also, liebe "Hardware Freaks"! Machen Sie Herrn Lampson Dampf, vielleicht läßt er sich erweichen. Was meinen Sie, Herr Lampson?
Red.

Betreffend der Hexdumps im NASCOM-JOURNAL möchte ich Sie bitten, grundsätzlich zu jedem Programm auch die Prüfsummen mit anzugeben. (Der T und L-Befehl von NASSYS 1 haben in dieser Hinsicht unbestreitbare Vorteile).
Da NASSYS aber auch eine eigentümliche Art hat, Prüfsummen zu berechnen (entgegen den Prüfsummen der Zeitschrift MC nicht nur die Summe aller Datenbits), bitte ich Sie fernerhin, zusätzlich noch die Art der Berechnung anzugeben. *
Anbei ein Programm, das beide Prüfsummen ausgeben kann (siehe unten).
Markus Caesar, 5653 Leichlingen

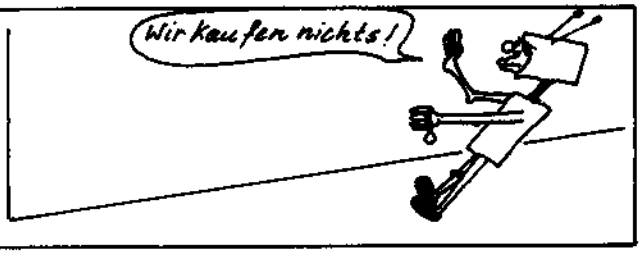
* Um nicht noch mehr Arbeitsaufwand zu treiben, als wir ohnehin haben, geben wir die Prüfsumme immer nur im NASCOM -üblichen Format an (Datenbits plus Adresse). Zum Cassettenlesen scheint diese Art doch vernünftig. Im übrigen geben wir diese Prüfsummen schon seit geraumer Zeit in Jedem Hexdump an. Red.

# PRÜFSUMMEN

## von Markus Caesar

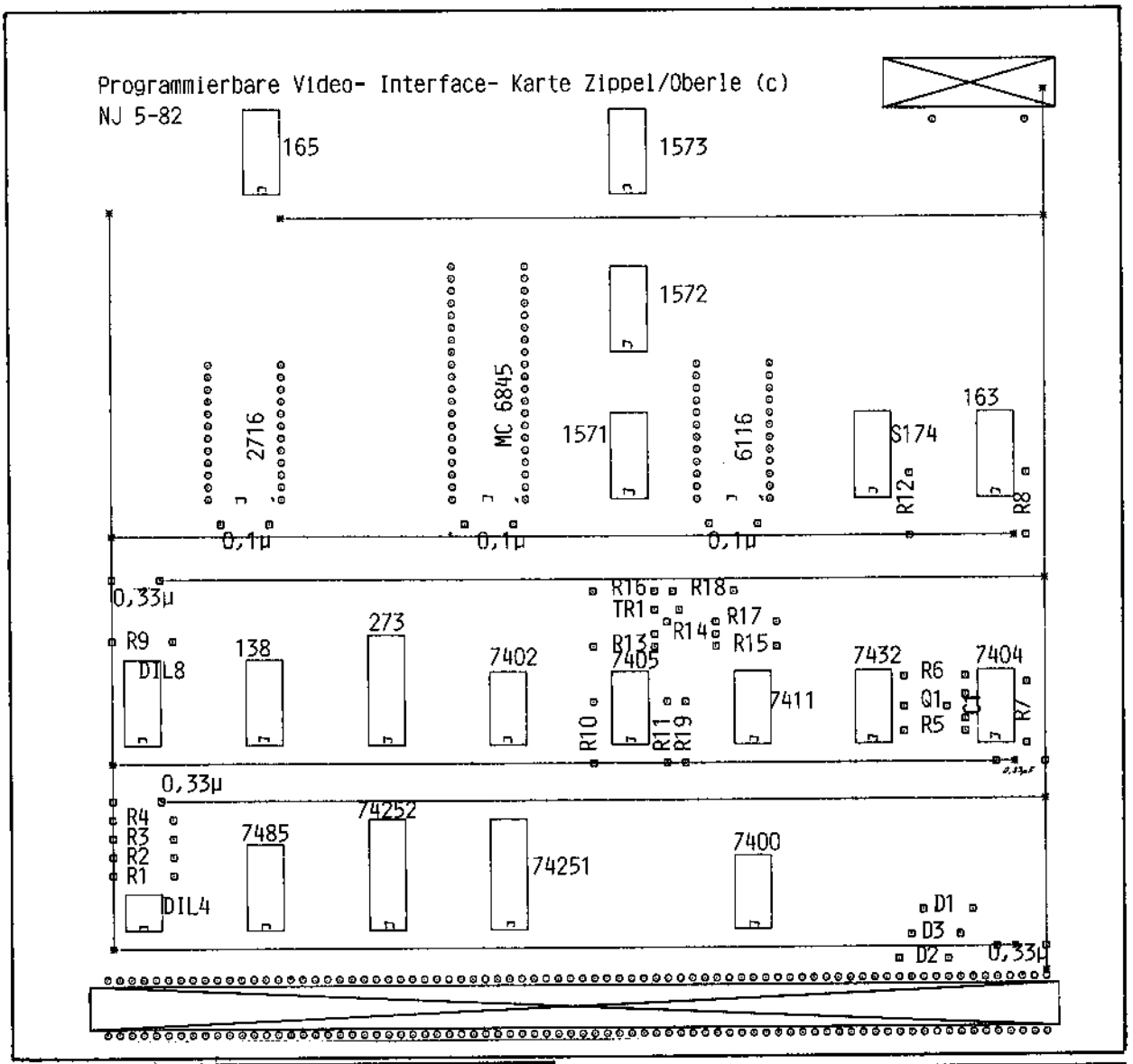
```
90 INPUT"  ";A$:CLS:SCREEN 1,5
91 POKE 4114,128
92 PRINT"      Berechnung von Pruefsummen
93 PRINT"      Maschinenprogramm in RAM laden !
94 INPUT"      Continue - Ø / NAS-SYS - 1 ";B$
95 IF LEFT$(B$,1)="1" THEN MONITOR
96 DIM A$(16)
97 FOR I=Ø TO 15:READ A$(I):NEXT I
98 DATA"Ø","1","2","3","4","5","6","7"
99 DATA"","9","A","B","C","D","E","F"
100 PRINT TAB(6);
101 INPUT"Formate : mc - Ø / nascom - 1 ";F
102 IF F <> 1 THEN PRINT CHR$(19);:GOTO101
103 IF F <> Ø THEN PRINT CHR$(19);:GOTO101
104 INPUT"      Anfangsadresse (hex) : ";W$
105 GOSUB144:A=Y:REM ---- Hex->Dez ----
106 INPUT"      Endadresse     (hex) : ";W$
107 GOSUB144:E=Y:REM ---- Hex->Dez ----
108 FOR I=A TO E STEP 8
109 PRINT" :";
110 :W=I:L=3:GOSUB135:REM -- L=Adresslaenge
111 :PRINT" :";:O=Ø
112 : FOR J=Ø TO 7
113 : O=O+PEEK(I+J)
114 : REM ---- K=Zeilenzaehler ----
115 : NEXT J
116 :O=O+F*(16+K*8+INT(K/32)):K=K+1
117 :IF O>255 THEN O=O-256:GOTO117
118 :PRUEF=O
119 :REM --- Drucke Bytes ----
120 : FOR J=ØTO7
121 : W=PEEK(I+J)
122 : L=1
123 : GOSUB135
124 : NEXT J
125 :REM --- Drucke Pruefsumme ---
126 :PRINT":";:W=PRUEF:GOSUB135:PRINT":";
127 :REM --- Drucke Character ---
128 : FOR J=ØTO7:CHAR=PEEK(I+J)
129 : IF CHAR<32 THEN  PRINT".";:GOTO131
130 : PRINT CHR$(CHAR);
131 : NEXT J:PRINT
132 NEXT I
133 GOTO1ØØ
134 REM ------------- DEZ HEX -------------
135 :IF W<ØTHEN W=W+65535
136 : FOR P=L TO Ø STEP -1
137 : W1=INT(W/(16↑P))
138 : IF W1>15 THEN 141
139 : PRINT A$(W1);
140 : W=W-(16↑P)*W1
141 : NEXT P
142 PRINT" ";:RETURN
143 REM -------------HEX DEZ-----------
144 IF W$="" THEN W$="ØØØØ"
145 Y=Ø:FOR L=Ø TO LEN(W$)-1
146 :Q=ASC(MID$(W$,L+1,1))-48:REM CHAR->HEX
147 :IF Q<Ø THEN RUN
148 :IF Q>23 THEN RUN
149 :IF Q>9 THEN Q=Q-7
150 :Y=Y+Q*(16↑(3-L))
151 :NEXT L
152 IF Y>32767 THEN Y=Y-65535
153 RETURN:=====================================
```





# VIDEOINTERFACE

Auf Wunsch vieler Leser liefern wir hier den Bestückungsplan für die Video- Karte (A.Zippel/D. Oberle) aus dem NASCOM Journal 5-82 nach.




Wer seinen ZEAP 2.Ø Ass. bisher im EPROM hatte und nun (möglicherweise für Floppy- Betrieb) im RAM laufen lassen möchte, erlebt sein blaues Wunder: der Assembler zerstört sich selbst!
Aus wer weiß welchen Gründen ist ins Programm der EPROM- Version ein Test eingebaut, der feststellt, ob der Assembler nicht "verbotenerweise" im RAM abgelegt ist.
Um das Programm auch im RAM lauffähig zu machen, sind nur 8 Bytes ab D5EC zu löschen (mit ØØ zu laden). Sehr einfach, wenn man's weiß! *Red.*

Im letzten Heft brachten wir eine Korrektur zum Video- Interface von Zippel/Oberle. Und die war leider falsch!
Glücklicherweise ist die Schaltung auf der Lay- Out- Folie richtig verdrahtet (wichtig für Besteller).
Der Vollständigkeit halber aber nun die (hoffentlich) endgültige Korrektur zur Schaltung:
IC273, Pin11 muß nicht, wie irrtümlich angegeben, an IC 6845 sondern an 74Ø4, Pin4.

# Einfache ERROM-Erweiterung

## von Günter Brust

Auf einfache Weise läßt sich der PROM-Speicherbereich der Speicherkarte durch Austausch der EPROMs 2708 durch solche vom Typ 2716 vergrößern. Die erforderlichen Änderungen an der Speicherkarte sind minimal und lassen sich leicht durchführen.

Als Vorteil ergeben sich doppelter Speicherplatz - 8 k gegenüber vorher 4 k - und zusätzlich ein geringerer Strombedarf von 2,1 Watt gegenüber 6 Watt.

Folgende funktionellen Änderungen sind durchzuführen:

1. Stromversorgung der EPROMs
2. Zuführung der Adreßleitung A10 an die EPROMs
3. Umschaltung des EPROM- Auswahldekoders auf die Adreßleitungen A11 und A12

EPROM Adressierung:

Wie bisher ist auch jetzt der EPROM- Block nur zusammenhängend adressierbar (8 k-Block). Mit der Verbindung "P5" (ROM-Select) sind jetzt jedoch zwei zusammenliegende 4 k- Ausgänge des Dekoders auszuwerten.

Beispiel:

Gewünschte EPROM- Adressen:

1: EPROM A000-A7FF Dekoderanschluß Nr.15
2. EPROM A800-AFFF

3. EPROM B000-B7FF
4. EPROM B800-BFFF Dekoderanschluß Nr.16

Verbindung also P5 mit Ausgang 15 und 16.

Erforderliche Änderungen an der RAM- Karte im einzelnen:

1. Auf der Lötseite ist am IC27, Pin21 die Verbindung zur -5V Stromversorgung zu trennen und Pin21 mit Pin24 zu verbinden (+5V Stromversorgung).(Bild 1).
2. Auf der Bauteileseite ist am IC27, Pin19 die Verbindung zur +12V Versorgung zu trennen und eine neue Verbindung zwischen Pin19 und der Kartenanschlußleiste Nr. 40 herzustellen (Adreßleitung A10). (Bild 2).
3. Auf der Lötseite sind die Anschlüsse, die zum IC24, Pin2 und Pin3 führen, zu trennen und neu nach Bild 1 zu verschalten. Eine neue Verbindung ist zwischen IC24, Pin3 und IC23, Pin13 herzustellen.

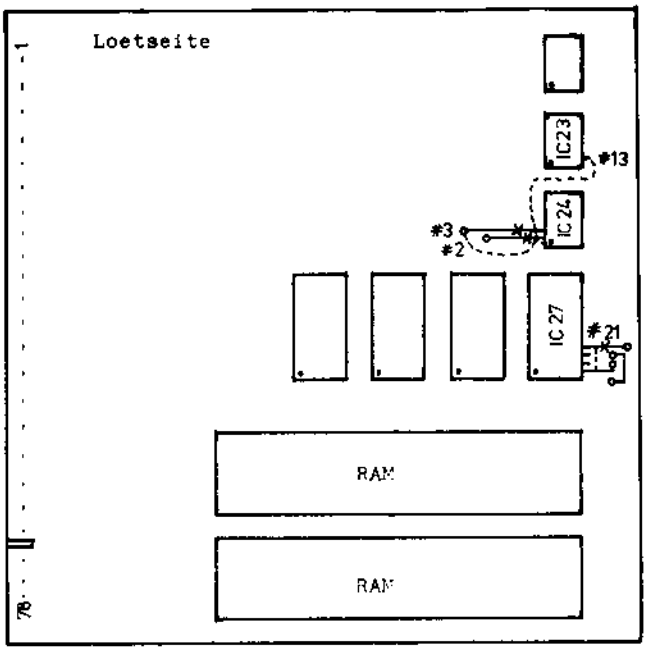
Literatur:'

NASCOM Memory Card Functional Specification PF/003, Issue No.2

Falls es NASCOM Anwender gibt, welche sich nicht trauen, ihre RAM- Karte umzubauen, übernehme ich gerne diese Arbeit einschließlich der Programmierung von 2716 EPROMs nach vorhandenen 2708.

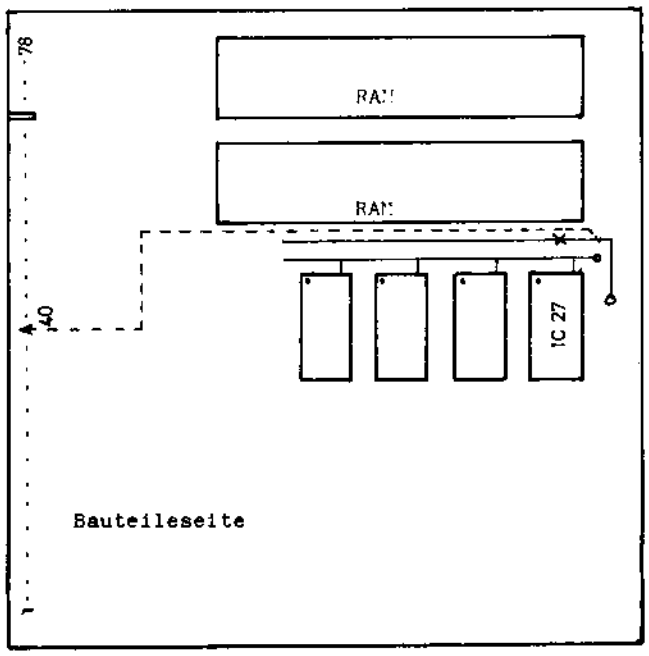
Umbau einschließlich 1x2716 = 55,- DM, mit 2x2716= 85,-DM.

Ich hoffe, damit der NASCOM- Gemeinde auch 'mal etwas Gutes tun zu können.




x Trennstelle

--- Neue Verbindung



# Preiswerte Floppy

## von Christoph Rau

Sicher gibt es noch mehr Leser, die so wie ich schon seit einiger Zeit mit dem Gedanken an eine Floppy liebäugeln. Immerhin besteht bei einem Z 80 - System die Möglichkeit, (später einmal ?) in den großen Benutzerkreis der CP/M - User einzusteigen mit fast 8 Megabyte Software im Hintergrund!

Bis jetzt wurde ich davon abgehalten durch einen ganz simplen aber wirkungsvollen Faktor, der mir schon manche Entscheidung abgenommen hat : zu teuer. Immerhin kostet die MKS - Floppy mit Gehäuse und Netzteil über 2000 Mark, die neue nascom - Floppy sogar fast 3000, und das preiswerteste System aus England von Gemini kostet dort knapp 2000 Mark.

Bei mir läuft nun ein Floppysystem, dessen Hardware - Realisierung mich keine 1000 Mark gekostet hat. Als Controller benutze ich eine Floppy - Controller Karte der Firma Janich & Klass in Wuppertal, die mir schon auf der letzten Hobbytronic aufgefallen waren mit ihren Z 80 Systemkarten. Die Karte ist für den ECB - Bus ausgelegt, für den ja von vielen Herstellern Erweiterungskarten angeboten werden. Sie arbeitet mit dem FD - Controller PD 765 von NEC und wird im Polling bedient, also eine Einfachlösung ohne DMA. Laut Beschreibung können bis zu 4 Mini - Floppies BASF 6106 (single sided) oder 6108 (double sided) mit single oder double density kontrolliert werden, aber meiner Ansicht nach müßte man alle Laufwerke mit Shugart-Bus anschließen können.

Die Karte wird über drei I/O Port - Adressen angesprochen, die durch ein PROM dekodiert werden. Die Dekodierung wird auf Bestellung programmiert, die Standardbelegung benutzt Port 0 bis 2 und ist daher für nascom - Benutzer unbrauchbar. Durch Umstellen des DIL - Schalters LSW2/8 für die Port - Adressen und Bereitstellung des NAS I/O - Signals muß gewährleistet werden, daß die Adressen nicht auf der Hauptplatine dekodiert werden. Bei mir legt eine selbstgebaute I/O - Platine dieses Signal auf high für alle Portadressen größer 0Fhex.

Die Karte kostet als Bausatz DM 330,- + Mwst. und ist in ca. 3 Stunden zusammengelötet. Basissoftware wird als Assemblerlisting mitgeliefert, und nach einem Telefonat kam auch das Datenblatt für den Controller.

Ein einseitiges Laufwerk BASF 6106 für 40 Spuren und double density war für DM 550,- bei Homecomputer in Düsseldorf zu bekommen. Ein Primitiv - Netzteil, was auch noch ein späteres zweites Laufwerk versorgen könnte, war für ca. DM 45,- schnell gebastelt, und nach einigen Laubsägeblättern war auch alles in einem Gehäuse für DM 30,- untergebracht, in dem auch noch Platz für ein zweites Laufwerk ist.

Die Testsoftware ist zwar für 6106 - Laufwerke ausgelegt, aber die Spezifikationen sind falsch angegeben. Daher hatte ich zunächst Problem, aber nun lä uft das System, und ich bin dabei, ein DOS zu schreiben. Das ist zwar nicht ganz einfach, hat aber den Vorteil, daß man wenigstens genau weiß, was passiert.

Mit dem Gedanken an CP/M im Hinterkopf würde ich mich über Gedankenaustausch mit Benutzern freuen, die ähnliches vorhaben oder bei denen vielleicht schon CP/M läuft.

# Hardware-Tips

## von Tom D. Rüdebusch

Einige Probleme, die einem naechtelanges Kopfzerbrechen bereiten koennen und weder bei dem oertlichen noch anderen Computergeschaeften oder Distributoren bekannt zu sein scheinen und deren Loesung, so einfach sie auch ist, fuer immer in einem undurchsichtigen Nebel von Vermutungen verborgen zu bleiben scheint, bis man mehr oder weniger durch Zufall darauf kommt oder sogar jemand kompetenten trifft, verdienen veroeffentlicht zu werden.

1. Das "A$-Problem"

Geben Sie das folgende kurze Basic-Programm ein.

```
10 A$="DIES IST EIN TESTSTRING"
20 FOR X=1 TO LEN(A$)Ü
30 PRINT MID$(A$,X,1);
40 NEXT:PRINT:GOTO20
```

Sollte bei 4 MHz nach einiger Laufzeit der String nur unvollstaendig auf dem Bildschirm ausgegeben werden, so sind Sie nicht der einzige Nascom 2-Besitzer, der sich nach einigem Kopfzerbrechen schon fast damit abgefunden hat, Basic nur bei 2 MHz zu benutzen. Uebrigens kann auch Zeap Schwierigkeiten bereiten.

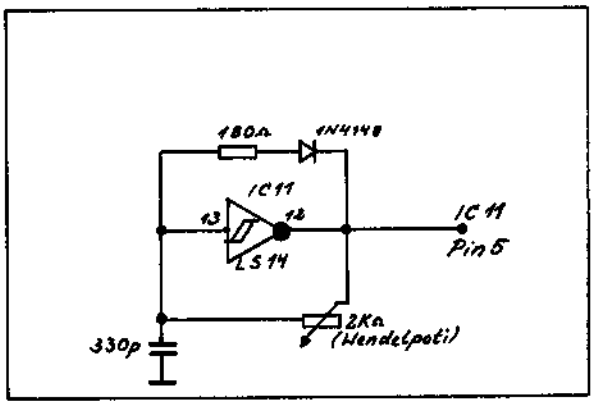
Die Ursache ist beim CPU-Takt zu suchen. Da der Video-Teil eine 8 MHz-Frequenz benoetigt, ist der Taktgenerator mit einem 16 MHz-Quartz aufgebaut. Das ist fuer den Z-80 A eigentlich unnoetig kritisch, da der System-Takt erst durch zweimaliges Teilen erzeugt wird. Und tatsaechlich sind einige Rechner dadurch recht unzuverlaessig.

Abhilfe: Man stellt LSW 1/9 nach unten und schliesst am Bus seinen eigenen Taktgenerator an. Dabei kann es sein, dass man unversehens auf eine ueberraschende Tatsache stoesst.

2. Nascom 2 bei 5 MHz ohne Wait

Als ich mich an den Aufbau des Schwingkreises machen wollte, war ploetzlich in der gesamten Umgebung kein 4 MHz-Quartz aufzutreiben. Mehr versuchsweise schloss ich also einen einfachen, mit Wendelpoti abstimmbaren, RC-Oszillator an. Und siehe da, oben aufgefuehrtes Testprogramm lief nicht nur bei 4 sondern bis gut 5 MHz ohne Wait-Zyklus fehlerfrei. Bei dieser einigermassen beeindruckenden Geschwindigkeit arbeitet mein System mit PIO, 2716-EPROMS und RAM "A" Karte seit einem Monat problemlos. Wer nicht unbedingt Wert auf Quartz-Stabilitaet legt, kann so mit optimaler Einstellung das Letzte aus seinem Rechner herausholen.

Wenn man die NMI-Taste nicht unbedingt benoetigt, kommt man sogar ohne ein zusaetzliches IC aus. Die Beinchen 12 und 13 von IC 11 (74LS14) werden herausgebogen und die entsprechenden Anschluesse der Fassung durch einen Draht ueberbrueckt. Jetzt ist der Interrupt-Taster nicht mehr entprellt, aber damit kann man leben. An den umgebogenen Beinchen des IC wird nun der Oszillator wie folgt aufgebaut (unbedingt Wendelpoti verwenden und LSW 1/9 nach unten schalten):




3. Memory-Plague (schon wieder?!)

Auch bei meiner RAM "A"-Karte versuchte ich alle obligatorischen Mittel, um ihr die "weichen Fehler" abzugewoehnen. Entgegen aller anderslautenden Versprechungen war aber bei 4 MHz kein Programm zum Laufen zu bewegen, bis ich auf der Hobby-Elektronik '81 in Stuttgart die Moeglichkeit hatte, mich mit dem Sales-Manager von Lucas Logic Ltd. zu unterhalten. Der gab mir dann den entscheidenden Tip: Alle zusaetzlichen Kondensatoren und Widerstaende entfernen und stattdessen nur die Drahtbruecken P8-P9 und P12-P13 durch 68 Ohm-Widerstaende ersetzen. Meine Karte laeuft seitdem einwandfrei, und da ich nicht glaube, dass sich diese einfache Massnahme schon allgemein herumgesprochen hat, wollte ich sie nicht unerwaehnt lassen.

4. Toolkit und Nas-Sys

Wie der Befehl zur automatischen Erzeugung von Zeilennummern auch unter Nas-Sys 3 zum korrekten Arbeiten zu bringen ist, wurde in der letzten Ausgabe der INMC 80-News beschrieben, wenn auch die Adresse nicht ganz stimmte. Liegt Toolkit bei B000, so muss der Inhalt von Adresse B15B von 08 auf 0C geaendert werden. Mit folgenden Aenderungen ist er uebrigens auch im RAM lauffaehig:

B021 3A statt C9
B23F E1 " D7
B248 C1 " FF

# Typenrad-Terminal

## von Günter Kreidl

*Im Mai-Heft habe ich gezeigt, wie man die Olivetti Praxis 30/35 über 18 I/O-Leitungen als Drucker ansteuern kann. In diesem Heft möchte ich ein einfaches Interface vorstellen, das mit den 8 Leitungen eines Pio-Ports auskommt und zudem mit Interrupt arbeiten kann (nicht muß). Das versprochene Tastatur-Interface ist noch in Arbeit (Es soll ebenfalls mit Interrupt arbeiten!).*

*Zur Druckgeschwindigkeit*

*Ich habe lange herumexperimentiert, um herauszubekommen, warum die Maschine nicht die Geschwindigkeit von 12 Z./Sek. erreicht, die vom Hersteller angegeben wird. Das Ergebnis war verblüffend: Sie erreicht diese Geschwindigkeit wirklich, aber nur gemittelt über die unterschiedliche Druckgeschwindigkeit einzelner Zeichen und über die Häufigkeit, mit der die einzelnen Zeichen in einem gewöhnlichen Text vorkommen. Kleinbuchstaben und Zahlen werden relativ schnell verarbeitet, Großbuchstaben und Sonderzeichen um etwa den Faktor 1,5 langsamer. Besteht ein Text also nur aus Großbuchstaben, kommt die Maschine ins Stolpern, wenn die Datenrate größer als etwa 8 Zeichen pro Sekunde ist. Für die Textverarbeitung kann man also eine höhere Geschwindigkeit wählen als etwa für das Ausdrucken eines Assembler- oder Basic-Listings. Ein 2. Problem ist die Umschaltung zwischen den beiden Zeichensätzen. Von Hand geschieht dies über einen Schiebeschalter an der Tastatur, also relativ langsam. Der Mikrocomputer in der Schreibmaschine fragt diesen Schalter immer nur dann ab, wenn der Zeichenpuffer leergeschrieben ist, weil man von Hand sowieso nicht so schnell umschalten kann. Wechselt man den Zeichensatz nun während des Druckerbetriebs, muß vorher eine Pause eingelegt werden, damit zunächst der Textpuffer leergeschrieben wird. Kommt ein solcher Wechsel zu häufig vor, dann vermindert sich die Druckgeschwindigkeit entsprechend. Man sollte deshalb für jede Aufgabe die Zeichenebene so auswählen, daß man mit möglichst wenig Umschaltungen auskommt. Braucht man z.B. häufig eines der folgenden Zeichen "$,£,*", sollte man die KBII-Belegung benutzen, KBI hingegen, wenn häufig die Zeichen "=,%,&,!" benötigt werden. Das betrifft nicht die Schalterstellung an der Maschine (der Schalter muß für Computerbetrieb immer auf KBII stehen!), sondern die Ansteuerung über das Interface: Bit 7 im Normalfall = 1 (KBII) oder = 0 (KBI). Es empfiehlt sich daher, für verschiedene Aufgaben jeweils geeignete Treiberprogramme zu schreiben, um die optimale Geschwindigkeit zu erreichen. (Ich habe inzwischen 4 verschiedene Druckprogramme für verschiedene Anwendungen in Betrieb und erreiche je nach Anwendung Geschwindigkeiten zwischen 8 und 12 Zeichen pro Sekunde.) Die Geschwindigkeit kann per Hardware (Interruptbetrieb) oder per Software eingestellt werden. Wählt man die Hardwareeinstellung, sollte man die beiden Trimmpotis durch ein Doppelpoti und zwei 10K-Widerstände ersetzen.*

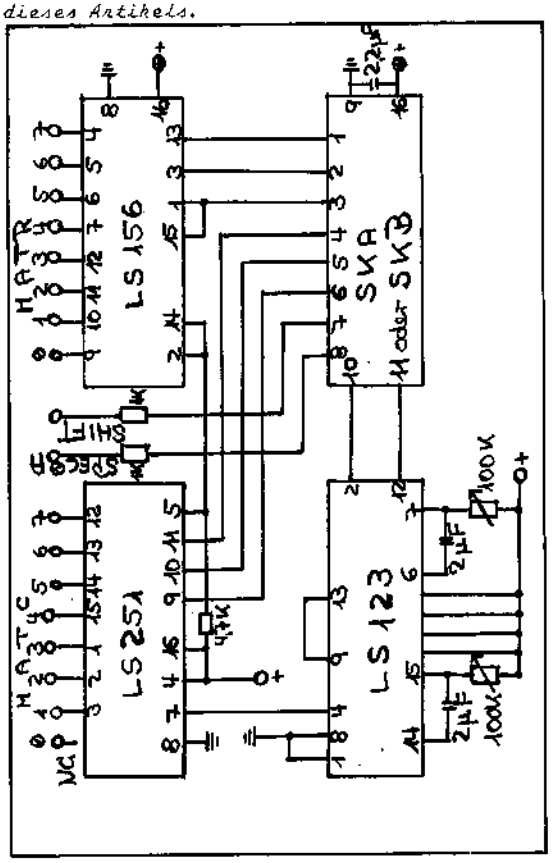
*Das Interface*

*Die Schaltung besteht aus nur 3 ICs: Der 8/1-Multiplexer LS251 hört den Tastatureingang ab und gibt ein Strobe-Signal auf den 3/8-Decoder LS156, wenn die über ein 3-Bit-Wort angewählte Zeile der Tastaturmatrix vom internen Mikroprozessor abgefragt wird. Der 3/8-Decoder gibt dann den ebenfalls mit 3 Bit angewählten Spaltencode aus und imitiert damit das Drücken einer Taste. Dieser Vorgang muß mindestens 3 mal wiederholt werden, danach muß etwa ebenso oft die entsprechende Spaltenleitung wieder auf log. 1 gesetzt werden, damit der interne Mikrocomputer das Signal als "Taste losgelassen" interpretiert. Die Steuerung dieses Ablaufs übernehmen die beiden Monoflops des LS123. Das erste MF wird über den RDY-Ausgang der Pio getriggert und erzeugt das Freigabesignal für das LS251. Dieses Signal wird dann wieder zurückgesetzt und gleichzeitig MF2 getriggert. Der invertierte Ausgang dieses Monoflops ist mit dem STB-Eingang der Pio verbunden und erzeugt im Interruptbetrieb das Unterbrechungssignal, das ein neues Zeichen für den Drucker anfordert. Bit 6 und 7 der Pio sind über einen Schutzwiderstand direkt*

*mit der SHIFT- und KB-Select-Leitung der Schreibmaschine verbunden. MATCO wird nicht ausgewertet. Damit spart man eine zusätzliche Enable-Leitung und kann einen 0-Code ausgeben, der keine Wirkung hat und im Interruptbetrieb für zusätzliche Verzögerungen (z.B. nach CRLF) als Pseudocode benutzt wird. Man verzichtet damit allerdings auf 4 Funktionen der Schreibmaschine, die beim Druckerbetrieb auch in der Regel nicht benötigt werden.*

*Die Software*

*Das hier vorgestellte Programm arbeitet zwar im Interruptbetrieb, benutzt den Interrupt aber nur anstelle einer Warteschleife. Das Programm ist deshalb auch unter NAS-SYS-1 lauffähig. Interessant wäre natürlich eine Version, die einen echten Interruptbetrieb ermöglicht, wobei z.B. ein bestimmter Speicherbereich ausgedruckt wird, während der Computer gleichzeitig andere Aufgaben erfüllt. Eine solche Version wäre allerdings unter NAS-SYS-1 nicht lauffähig. Mehr darüber in einem der nächsten Hefte. Es wurde eine KBI-Belegung gewählt, die sich etwa für den MC-Texteditor oder ähnliche Aufgaben eignet. Eine besondere Funktion wurde den CTRL-Codes 1 und 3 zugeordnet. Sie erzeugen einen halben bzw. anderthalbfachen Zwischenraum mit Hilfe der Halbschritt-Funktion. Damit erfolgte auch die Formatierung dieses Artikels.*




```
                0010 ; PRAXIS-30-TREIBER
                0020 ;Ausgabe über Pio-Port A
                0030 ;mit Interrupt statt
                0040 ;Software-Verzögerung
                0050 ; VERSION 2.1
                0060 ;für Textverarbeitung
                0070 ; SONDERFUNKTIONEN:
                0080 ;CTRL A = Halbschritt
                0090 ;CTRL C = 1 1/2-Schritt
                0100 ;G.K.     19.7.82
0E00            0110        ORG  £E00
0E00 F5         0120 PRINT  PUSH AF
0E01 E5         0130        PUSH HL
0E02 D5         0140        PUSH DE
0E03 C5         0150        PUSH BC
0E04 21D30E     0160        LD   HL,EFLAG
0E07 0E01       0170        LD   C,1
0E09 71         0180        LD   (HL),C
0E0A FB         0190        EI
0E0B CD1B0E     0200 WAIT   CALL PR
0E0E 76         0210        HALT
0E0F 21D30E     0220        LD   HL,EFLAG
0E12 7E         0230        LD   A,(HL)
0E13 B7         0240        OR   A
0E14 20F5       0250        JR   NZ,WAIT
0E16 C1         0260        POP  BC
0E17 D1         0270        POP  DE
0E18 E1         0280        POP  HL
0E19 F1         0290        POP  AF
0E1A C9         0300        RET
0E1B 35         0310 PR     DEC  (HL)
0E1C 2806       0320        JR   Z,PR1
0E1E 3AD20E     0330        LD   A,(REP)
0E21 C3A50E     0340        JP   PR10
0E24 21CE0E     0350 PR1    LD   HL,CRDEL
0E27 46         0360        LD   B,(HL)
0E28 FE08       0370        CP   £8
0E2A 2006       0380        JR   NZ,PR2
0E2C 05         0390        DEC  B
0E2D 05         0400        DEC  B
0E2E 3E75       0410        LD   A,£75
0E30 186F       0420        JR   PR9
0E32 FE0D       0430 PR2    CP   £D
0E34 201A       0440        JR   NZ,PR3
0E36 CB38       0450        SRL  B
0E38 CB38       0460        SRL  B
0E3A CB38       0470        SRL  B
0E3C 78         0480        LD   A,B
0E3D CB38       0490        SRL  B
0E3F 90         0500        SUB  B
0E40 C604       0510        ADD  A,4
0E42 32D30E     0520        LD   (EFLAG),A
0E45 0600       0530        LD   B,0
0E47 3E40       0540        LD   A,£40
0E49 32D20E     0550        LD   (REP),A
0E4C 3E76       0560        LD   A,£76
0E4E 1851       0570        JR   PR9
0E50 FE01       0580 PR3    CP   1
0E52 2005       0590        JR   NZ,PR4
0E54 04         0600        INC  B
0E55 3E7A       0610        LD   A,£7A
0E57 1848       0620        JR   PR9
0E59 FE03       0630 PR4    CP   3
0E5B 200C       0640        JR   NZ,PR6
0E5D 3E02       0650        LD   A,2
0E5F 32D30E     0660        LD   (EFLAG),A
0E62 3E7A       0670        LD   A,£7A
0E64 32D20E     0680        LD   (REP),A
0E67 3E20       0690        LD   A,£20
0E69 FE80       0700 PR6    CP   £80
0E6B 3804       0710        JR   C,PR8
0E6D 3E40       0720 PR7    LD   A,£40
0E6F 1834       0730        JR   PR10
0E71 FE20       0740 PR8    CP   £20
0E73 38F8       0750        JR   C,PR7
0E75 04         0760        INC  B
0E76 04         0770        INC  B
0E77 1600       0780        LD   D,0
0E79 21B40E     0790        LD   HL,PRTAB-£20
0E7C FE60       0800        CP   £60
0E7E 3809       0810        JR   C,DEC1
0E80 D620       0820        SUB  £20
0E82 5F         0830        LD   E,A
0E83 19         0840        ADD  HL,DE
0E84 7E         0850        LD   A,(HL)
```

```
0E85 C640.     0860         ADD  A,£40
0E87 1803      0870         JR   DEC2
0E89 5F        0880 DEC1    LD   E,A
0E8A 19        0890         ADD  HL,DE
0E8B 7E        0900         LD   A,(HL)
0E8C FE80      0910 DEC2    CP   £80
0E8E 3811      0920         JR   C,PR9
0E90 F5        0930         PUSH AF
0E91 3E03      0940         LD   A,3
0E93 21D30E    0950         LD   HL,EFLAG
0E96 77        0960         LD   (HL),A
0E97 3E40      0970 DEC3    LD   A,£40
0E99 D304      0980         OUT  (4),A
0E9B 76        0990         HALT
0E9C 35        1000         DEC  (HL)
0E9D 20F8      1010         JR   NZ,DEC3
0E9F FB        1020         EI
0EA0 F1        1030         POP  AF
0EA1 21CE0E    1040 PR9     LD   HL,CRDEL
0EA4 70        1050         LD   (HL),B
0EA5 D304      1060 PR10    OUT  (4),A
0EA7 C9        1070         RET
0EA8 F5        1080 IPR     PUSH AF
0EA9 3AD30E    1090         LD   A,(EFLAG)
0EAC B7        1100         OR   A
0EAD 2003      1110         JR   NZ,IPR1
0EAF F1        1120         POP  AF
0EB0 1802      1130         JR   IPRE
0EB2 F1        1140 IPR1    POP  AF
0EB3 FB        1150         EI
0EB4 ED4D      1160 IPRE    RETI
               1170 ;Pio-Initialisierung:
               1180 ;(muß vor dem ersten
               1190 ;Aufruf der Druckroutine
               1200 ;ausgeführt werden)
0EB6 3E0F      1210 PINIT   LD   A,£F
0EB8 D306      1220         OUT  (6),A
0EBA 3E87      1230         LD   A,£87
0EBC D306      1240         OUT  (6),A
0EBE 21CC0E    1250         LD   HL,ITABLE
0EC1 7D        1260         LD   A,L
0EC2 D306      1270         OUT  (6),A
0EC4 7C        1280         LD   A,H
0EC5 ED47      1290         LD   I,A
0EC7 ED5E      1300         IM   2
0EC9 DF        1310         RST  £18
0ECA 5B        1320         DEFB £5B
0ECB 00        1330         NOP
0ECC A80E      1340 ITABLE  DEFW IPR
0ECE 00        1350 CRDEL   DEFB 0
0ECF 00        1360 LCOUNT  DEFB 0
0ED0 00        1370 MAXCHR  DEFB 0
0ED1 00        1380 MAXLN   DEFB 0
0ED2 00        1390 REP     DEFB 0
0ED3 00        1400 EFLAG   DEFB 0
0ED4 77        1410 PRTAB   DEFB £77,£12,£10,£98
0ED8 A0        1420         DEFB £A0,£20,£30,£0A
0EDC 28        1430         DEFB £28,£33,£A6,£5E
0EE0 5A        1440         DEFB £5A,£4A,£52,£1E
0EE4 1D        1450         DEFB £1D,£48,£50,£58
0EE8 60        1460         DEFB £60,£70,£68,£73
0EEC 6E        1470         DEFB £6E,£66,£16,£08
0EF0 28        1480         DEFB £28,£18,£33,£1A
0EF4 26        1490         DEFB £26,£0C,£32,£1F
0EF8 1C        1500         DEFB £1C,£1B,£24,£2F
0EFC 31        1510         DEFB £31,£25,£29,£21
0F00 19        1520         DEFB £19,£22,£2A,£1D
0F04 15        1530         DEFB £15,£0B,£23,£14
0F08 2B        1540         DEFB £2B,£2D,£27,£13
0F0C 17        1550         DEFB £17,£0F,£2C,£09
0F10 11        1560         DEFB £11,£0D,£16,£2E
               1570 ;
               1580 ;TESTPROGRAMM
               1590 ;dient zur Einstellung
               1600 ;der Druckgeschwindig=
               1610 ;keit am Interface;
               1620 ;muß alle den Werten
               1630 ; 7FH bis 20H
               1640 ;entsprechenden Zeichen
               1650 ;fehlerfrei ausdrucken.
0F14 0E03      1660         LD   C,3
0F16 3E0D      1670 LPP1    LD   A,£D
0F18 CD000E    1680         CALL PRINT
0F1B 065F      1690         LD   B,£5F
0F1D 78        1700 LOOP    LD   A,B
0F1E C620      1710         ADD  A,£20
0F20 CD000E    1720         CALL PRINT
0F23 10F8      1730         DJNZ LOOP
0F25 0D        1740         DEC  C
0F26 20EE      1750         JR   NZ,LPP1
0F28 DF        1760         RST  £18
0F29 5B        1770         DEFB £5B
```

# STEREO Aussteuerungsmesser

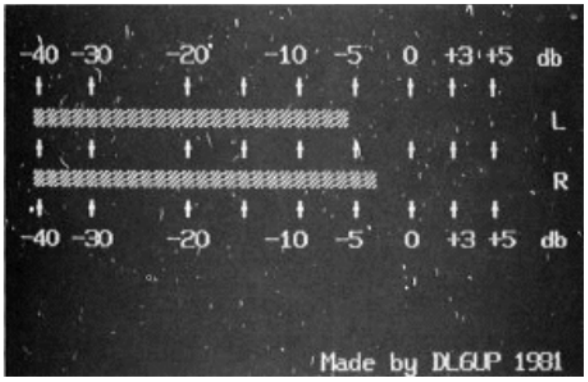
## von Günter Brust

Der eine oder andere Dia-, Film-, Audio- oder Videofreund hatte sicher schon einmal die Gelegenheit, einen Blick in ein Rundfunk- oder Plattenstudio zu werfen. Die dort verwendeten Aussteuerungsmesser (nach DIN 45406) lassen das Herz höher Schlagen, wenn man an seine kleinen VU-Meter(chen) denkt, welche vielfach nur eine schöne Verzierung darstellen. Die Anschaffungspreise für professionelle Aussteuerungsmesser bringen einen aber schnell in die Wirklichkeit zurück. Viel wurde in Fachzeitschriften zwar schon geschrieben über den Bau derartiger Einrichtungen, im Endeffekt ist der Aufwand für den Selbstbau aber ebenfalls erheblich: Meßgleichrichter, logarithmische Gleichspannungsverstärker und eine Anzeigeeinrichtung, welche das Meßsignal unverfälscht wiedergibt, sind erforderlich.

Angeregt durch den CHIP- Artikel "Wandler" (11/79) kam mir die Idee, mein Mikroprozessorsystem als Aussteuerungsmesser zu programmieren. Das Video- Display ergibt eine ideale Anzeigeeinrichtung, die logarithmische Umwandlung kann softwaremäßig erfolgen. Zwei brauchbare A/D- Wandler lassen sich einfach für ca. DM 25.- herstellen. Als Meßgleichrichter wurde, da von früheren Experimenten vorhanden, der in ELEKTOR 4/77 beschriebene Gleichrichter verwendet (Preis ca. DM 50.-). Das verwendete Prozessorsystem ist der NASCOM 1 (ohne Erweiterung) mit NAS-SYS.

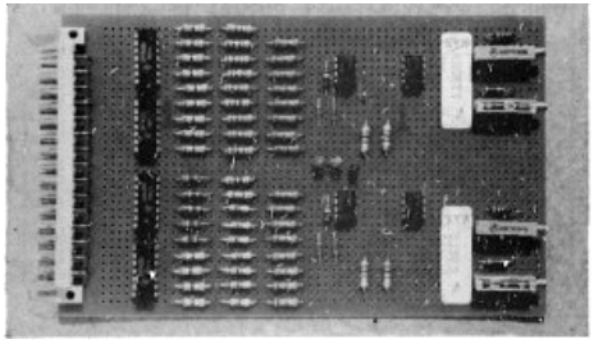
Hardwarebeschreibung:

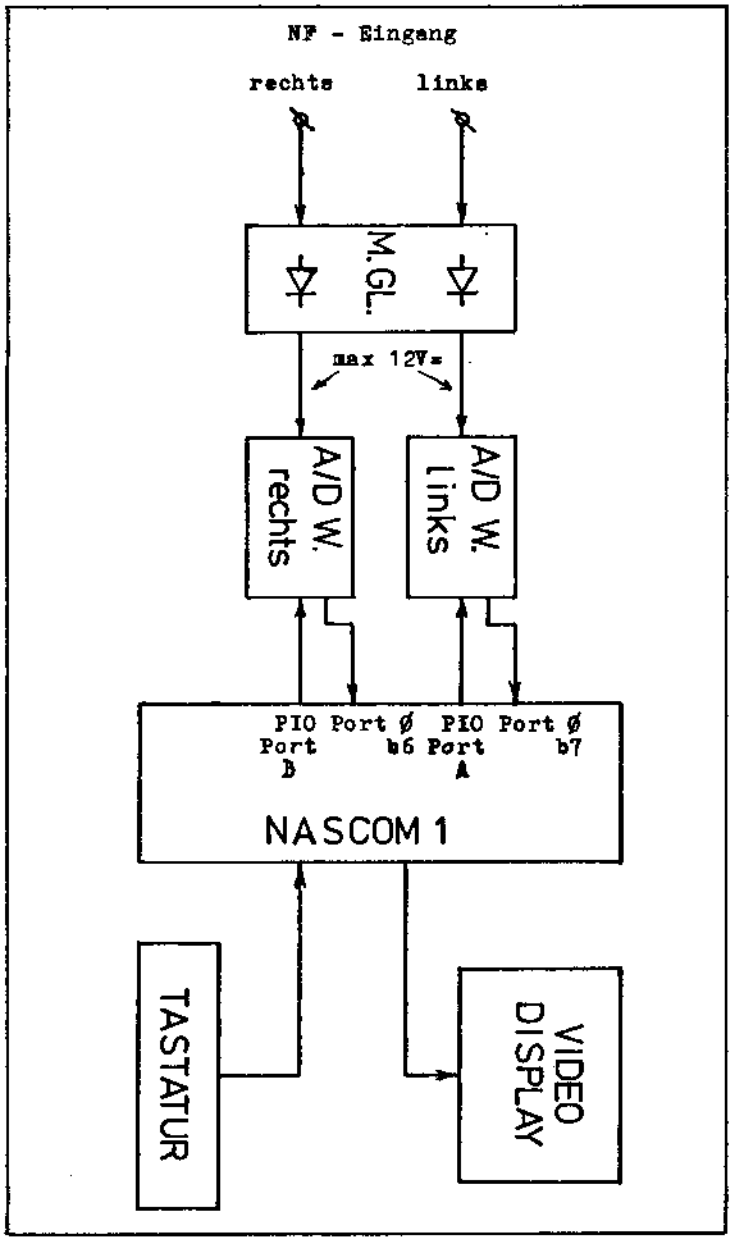
Zwei A/D- Wandler, welche leicht auf einer Europakarte aufzubauen sind, übernehmen die Digitalisierung des vom Meßgleichrichter gelieferten Signals. Es liefert bei Vollaussteuerung (+5 dB Anzeige) eine Ausgangsspannung von 12V=. Einstellregler für Null- und Maximumabgleich sind vorhanden. Die zwei A/D- Wandler werden an Port A und Port B der PIO angeschlossen, (Das Hex-Dump ist für diese Ports geschrieben. Bei Verwendung anderer Ports fordern Sie bitte das Assemblerlisting an, um das Programm anzupassen), für die Komparatorrückmeldung werden die zwei freien Eingänge des Port 0 (Tastaturanschluß) verwendet.




Softwarebeschreibung:
Das Betriebssystem NASSYS wird verwendet, um Skalen, Marken und Bezeichnungen auf das Video- Display zu schreiben. (Den Flußdiagrammen können Sie entnehmen, wie das Programm auf anderen Prozessorsystemen anzuwenden ist).
Die zwei a/D- Wandler werden abwechseln vom Programm abgefragt. (Siehe auch CHIP 11/79 "Wandler"). Die angewendete Methode der schrittweisen Annäherung (succesive approximation) erfordert zur Abfrage nur ca. Ø,5 mS. Die hereingeholten Meßwerte werden mit einer im Speicher abgelegten Tabelle logarithmisch umgeformt und ebenfalls auf den Bildschirm in Form von Leuchtbalken geschrieben.
Mit einer 8- Bit- Digitalisierung kann man zwar eine maximale Auflösung von 47 dB erreichen, da sich jedoch durch die logarithmische Umwandlung die kleinsten Meßwerte nicht mehr umwandeln lassen, mußte die Anzeige auf 45 dB begrenzt werden.
Die Genauigkeit der Anzeige, Anstieg- und Abfallzeiten werden ausschließlich von den verwendeten Meßverstärkern bestimmt.

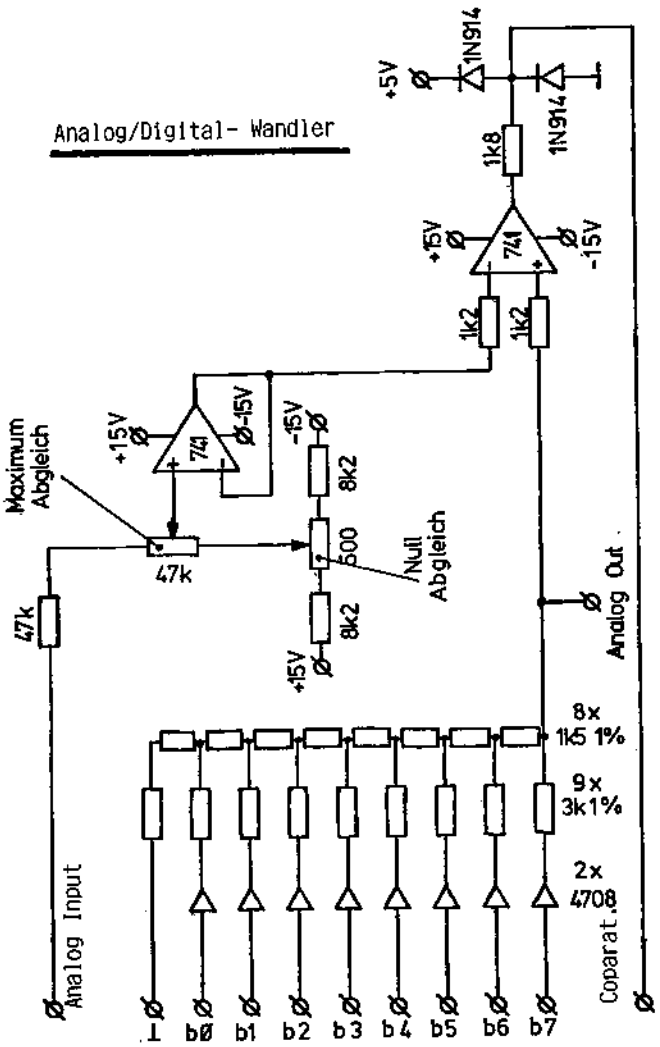
Literatur:
"Wandler" CHIP 11/79
"LED- Aussteuerungsmesser" ELEKTOR 4/77
DIN Blatt 45 4Ø6 11/66






```
0c80 3e 0c f7 21 ce 08 22 29 0c cd 1a 0d 21 4e 0a 22
0c90 29 0c cd 1a 0d 21 0f 09 22 29 0c cd 45 0d 21 8f
0ca0 09 22 29 0c cd 45 0d 21 0f 0a 22 29 0c cd 45 0d
0cb0 3e 4c 32 75 09 3e 52 32 f5 09 21 64 0b 22 29 0c
0cc0 ef 4d 61 64 65 20 62 79 20 44 4c 36 55 50 20 31
0cd0 39 38 31 00 cd e2 0d 00 00 00 00 00 00 00 00 00
0ce0 3e 04 32 be 0d 3e 7f 32 c7 0d cd b8 0d cd 81 0d
0cf0 21 4f 09 cd 6a 0d 21 4f 09 cd 74 0d 3e 05 32 be
0d00 0d 3e 77 32 c7 0d cd b8 0d cd 81 0d 21 cf 09 cd
0d10 6a 0d 21 cf 09 cd 74 0d 18 c6 ef 2d 34 4f 20 2d
0d20 33 4f 20 20 20 20 2d 32 4f 20 20 20 20 2d 31 4f
0d30 20 20 2d 35 20 20 20 4f 20 20 2b 33 20 2b 35 20
0d40 20 64 62 00 c9 ef 19 20 20 20 19 20 20 20 20 20
0d50 20 19 20 20 20 19 20 20 20 19 20 20 20 19 20 20
0d60 20 19 20 20 19 20 20 19 00 c9 0e 23 3e 20 77 23
0d70 0d 20 fb c9 3e 00 b8 28 07 3e 7f 77 23 05 18 f4
0d80 c9 21 90 0d 06 00 be d8 c8 04 23 18 f9 00 00 00
0d90 01 02 03 04 05 06 07 08 09 0a 0c 0e 11 13 16 19
0da0 1d 21 27 2d 33 3b 45 50 5b 6a 7a 8d a0 b4 c8 d7
0db0 eb ff 00 00 00 00 00 00 06 80 0e 40 78 d3 00 1e
0dc0 30 1d 20 fd db 00 cb 00 78 28 05 91 16 01 18 03
0dd0 81 16 00 47 79 fe 00 1f 4f 20 e1 7a 2f 3c 80 c9
0de0 00 00 3e ff d3 06 3e 00 d3 06 3e ff d3 07 3e 00
0df0 d3 07 c9
```

## Analog/Digital- Wandler




## HAUPTPROGRAMM

START

Startadresse: 0C80

Schirm löschen, Skalen, Marken, Bezeichnungen setzen. PIO A+B initialisieren.

ANFANG

Adresse und Maske für linken Kanal ins Unterprogramm ABFRAGE laden.

ABFRAGE links / LOGWAndlung — Messwert in A, Leuchtpunktzahl in B

Anzeigeadr. links laden

Letzte Anzeige LOESCHen

Anzeigeadr. links laden

Neue ANZEIGE → Schirm

Adresse und Maske für rechten Kanal ins Unterprogramm ABFRAGE laden.

ABFRAGE rechts / LOGWAndlung — Messwert in A, Leuchtpunktzahl in B

Anzeigeadr. rechts laden

Letzte Anzeige LOESCHen

Anzeigeadr. rechts laden

Neue ANZEIGE → Schirm

Beendigung durch RESET

## Unterprogramm linear/LOGWAndlung

LOGWA — Meßwert wird in A übergeben

Tabellenadresse → HL
00 → B (Leuchtpunktzähler)

NEXT

Vergleich A mit Tabellenwert

grösser ? — ja

nein

gleich ? — ja

nein

INC B (Punktzahl)
INC HL (nächste Tabellenst.)

Anzahl der Leuchtpunkte steht in B.

Return

## Unterprogramm Bildschirm LOESCHen

Schirmadresse für LOESCHen wird vom Hauptprogramm in HL übergeben.

LOESCH

C ← 23 h
A ← 20 h

Max. Zahl der Leuchtpunkte
ASCII-Zeichen 20

PLOE1

Punkt auf Schirmadresse löschen.

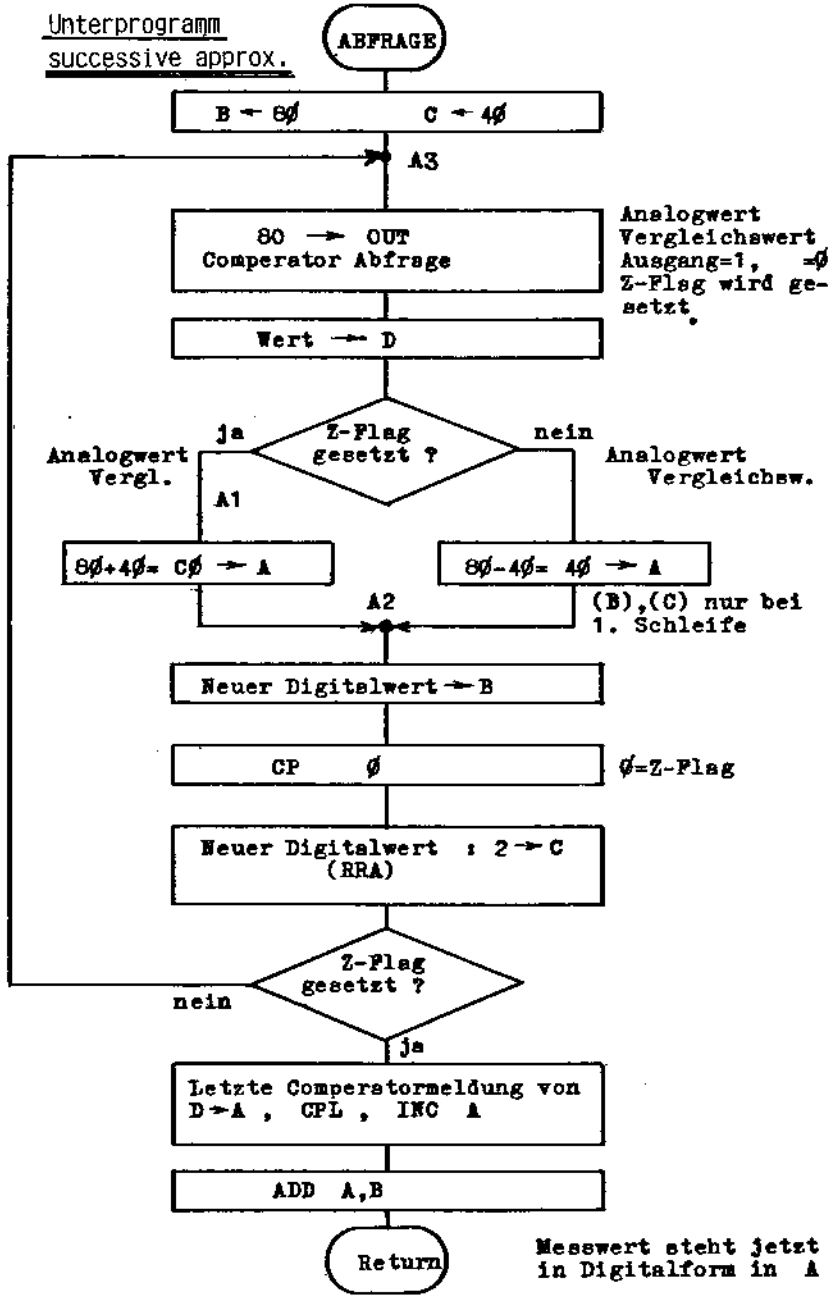
INC HL (Schirmadresse)
DEC C (Leuchtpunktzähler)

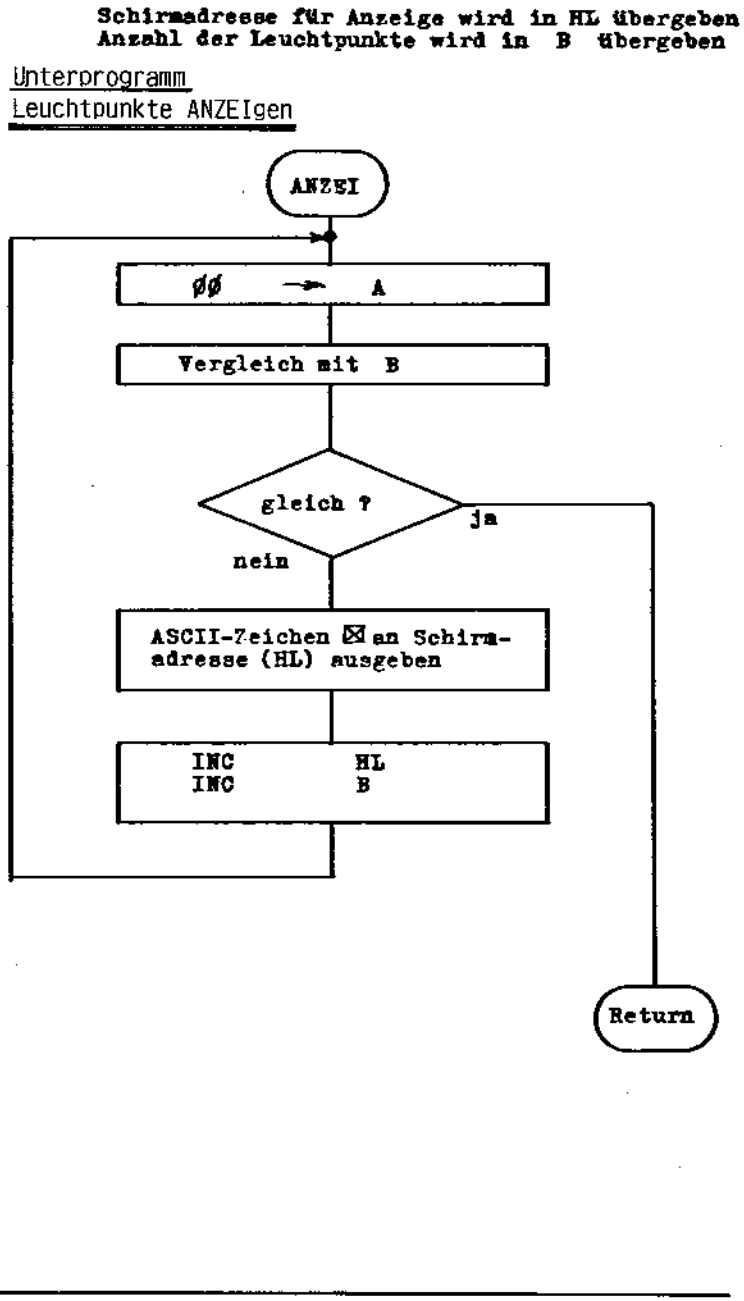
Reg. C=NZ ?

ja

nein

Return


Unterprogramm
successive approx.
ABFRAGE
B ← 8Ø
C ← 4Ø
A3
80 → OUT
Comperator Abfrage
Analogwert
Vergleichswert
Ausgang=1, =Ø
Z-Flag wird gesetzt
Wert → D
Z-Flag gesetzt ?
ja
nein
Analogwert Vergl.
Analogwert Vergleichsw.
A1
8Ø+4Ø= CØ → A
8Ø-4Ø= 4Ø → A
A2
(B),(C) nur bei 1. Schleife
Neuer Digitalwert → B
CP Ø
Ø=Z-Flag
Neuer Digitalwert : 2 → C
(RRA)
Z-Flag gesetzt ?
nein
ja
Letzte Comperatormeldung von
D→A , CPL , INC A
ADD A,B
Return
Messwert steht jetzt in Digitalform in A


Schirmadresse für Anzeige wird in HL übergeben
Anzahl der Leuchtpunkte wird in B übergeben
Unterprogramm
Leuchtpunkte ANZEIgen
ANZEI
ØØ → A
Vergleich mit B
gleich ?
ja
nein
ASCII-Zeichen ☒ an Schirmadresse (HL) ausgeben
INC HL
INC B
Return

# Sprachausgabe

## von Tom D. Rüdebusch

Mit diesem recht einfachen Programm ist es moeglich, einem Nascom deutlich verstaendliche Worte zu entlocken. Auf meinem Rechner laeuft eine Version, die Pruefsummen oder ganze Speicherbereiche "vorliest". Wer schon einmal laengere Maschinenprogramme nach dem Eintippen auf Fehler ueberprueft hat, wird einschaetzen koennen, welche Erleichterung es bedeutet, nicht mit einem Finger auf dem Listing und dem anderen auf dem Bildschirm Hex-Werte vergleichen zu muessen. Als Wortspeicher fuer die Zahlen Ø-F muss man allerdings etwa 12 kByte rechnen. Aber auch andere nicht weniger interessante Anwendungen lassen sich vorstellen.

Wer also seinem Nascom das Sprechen beibringen will, geht wie folgt vor:

Jedes Wort, das der Rechner "koennen" soll, muss ihm zuerst "vorgesagt" werden. Dabei muss die digitalisierte Sprache an Bit Ø/Port B anliegen. Dazu wird der Ausgang des Recorders, der die gewuenschten Laute von Cassette abspielt, oder des Verstaerkers, an den ein Mikrofon angeschlossen ist, mit TP 9 (Cass. In) verbunden. Dann ist noch eine Verbindung von TP 19 (Cass. Schmitt) zu Bit Ø/Port B herzustellen. Auf keinen Fall darf vergessen werden, BSTROBE auf Masse zu legen, sonst funktioniert die Eingabe nicht. Schliesslich muss noch ein Verstaerker an Bit 5/Port Ø angeschlossen werden. Das war dann schon der ganze Hardware-Aufwand.

Das Synthese-Programm wird gestartet durch "E 1ØØØ BegAdr EndAdr". Dabei ist BegAdr die erste Speicherstelle, die als Wortspeicher verwendet werden darf und EndAdr die erste, die nicht mehr zu benutzen ist. Nun kann "M" fuer Ruecksprung zum Monitor, "A" fuer Aufnahme und "W" fuer Wiedergabe gedrueckt werden. Solange der Computer digitalisierte Laute abspeichert, leuchtet die Cassrec.-LED auf; ist der gesamte vorgegebene Speicherbereich belegt, erlischt sie, und bevor wieder eine Eingabe moeglich ist, gibt eine Warteschleife 2 sec. Zeit, Cassettenrecorder oder Mikrofon abzuschalten, um den "Schmutzeffekt" zu vermeiden. Wird dann "W" getippt, kann man sich das Ergebnis sofort anhoeren.

Die beiden Unterprogramme "record" und "talk", der "Kern" des Synthese-Programms, kann natuerlich auch in anderen Anwendungen benutzt werden.Das Register HL muss dann BegAdr, und DE EndAdr enthalten, BC und AF werden veraendert. Man sollte uebrigens nicht versuchen, Bit 7/Port Ø fuer die Eingabe zu verwenden; das klingt nicht besonders. Die beiden scheinbar ueberfluessigen Befehle "neg" und "or a" passen die Aufnahmegeschwindigkeit der der Wiedergabe an. Die Verzoegerung "WAIT" stellt einen Kompromiss zwischen Sprachqualitaet und -Speicherbedarf dar. Sie ist fuer 4 MHz ausgelegt und kann versuchsweise veraendert werden.

```
ZEAP Z80 Assembler - Source Listing

1000                    0010          ORG  #1000
                        0020 ;
                        0030 ; MONITOR-ROUTINEN
1000 0008               0040 RIN      EQU  #8
1000 0028               0050 PRS      EQU  #28
1000 0030               0060 ROUT     EQU  #30
1000 0038               0070 RDEL     EQU  #38
1000 005F               0080 MFLP     EQU  #5F
1000 005B               0090 MRET     EQU  #5B
1000 005D               0100 TDEL     EQU  #5D
1000 0060               0110 ARGS     EQU  #60
1000 007B               0120 BLINK    EQU  #7B
                        0130 ;
1000 000D               0140 CR       EQU  #0D
                        0150 ;
                        0160 ; VARIABLEN
1000 0005               0170 WAIT     EQU  5
1000 0000               0180 PORTO    EQU  0
1000 0007               0190 CNTRPB   EQU  7
1000 0005               0200 PORTB    EQU  5
1000 007F               0210 PBIN     EQU  #7F
                        0220 ;
                        0230 ;
1000                    0240          ENT
1000 DF60               0250          SCAL ARGS
1002 EB                 0260          EX   DE,HL
1003 50                 0270          LD   D,B
1004 59                 0280          LD   E,C
1005 E5                 0290 INPUT    PUSH HL
1006 D5                 0300          PUSH DE
1007 EF                 0310          RST  PRS
1008 0D                 0320          DEFB CR
1009 415546             0330          DEFM /AUF/
100C 4E4148             0340          DEFM /NAH/
100F 4D4520             0350          DEFM /ME /
1012 284129             0360          DEFM /(A)/
1015 2C2057             0370          DEFM /, W/
1018 494544             0380          DEFM /IED/
101B 455247             0390          DEFM /ERG/
101E 414245             0400          DEFM /ABE/
1021 202857             0410          DEFM / (W/
1024 29204F             0420          DEFM /) O/
1027 444552             0430          DEFM /DER/
102A 204D4F             0440          DEFM / MO/
102D 4E4954             0450          DEFM /NIT/
1030 4F5220             0460          DEFM /OR /
1033 284D29             0470          DEFM /(M)/
1036 3F20               0480          DEFM /? /
1038 00                 0490          DEFB 0
1039 DF7B               0500          SCAL BLINK
103B F5                 0510          PUSH AF
103C F7                 0520          RST  ROUT
103D F1                 0530          POP  AF
103E D1                 0540          POP  DE
```

```
103F E1       0550        POP  HL
1040 E5       0560        PUSH HL
1041 D5       0570        PUSH DE
1042 FE41     0580        CP   "A
1044 CC5410   0590        CALL Z,RECORD
1047 FE57     0600        CP   "W
1049 CC7D10   0610        CALL Z,TALK
104C FE4D     0620        CP   "M
104E D1       0630        POP  DE
104F E1       0640        POP  HL
1050 20B3     0650        JR   NZ,INPUT
1052 DF5B     0660        SCAL MRET
              0670 ;
              0680 ;
1054 DF5F     0690 RECORD SCAL MFLP
1056 3E7F     0700        LD   A,PBIN
1058 D307     0710        OUT  (CNTRPB),A
105A 0608     0720 NXTBTE LD   B,8
105C 3E05     0730 GETBTE LD   A,WAIT
105E FF       0740        RST  RDEL
105F ED44     0750        NEG  ;VERZ.
1061 B7       0760        OR   A ;VERZ.
1062 DB05     0770        IN   A,(PORTB)
1064 CB47     0780        BIT  0,A
1066 2801     0790        JR   Z,ZERO
1068 37       0800        SCF
1069 CB19     0810 ZERO   RR   C
106B 10EF     0820        DJNZ GETBTE
106D 71       0830        LD   (HL),C
106E 23       0840        INC  HL
106F B7       0850        OR   A
1070 ED52     0860        SBC  HL,DE
1072 19       0870        ADD  HL,DE
1073 20E5     0880        JR   NZ,NXTBTE
1075 DF5F     0890        SCAL MFLP
1077 DF5D     0900        SCAL TDEL
1079 DF5D     0910        SCAL TDEL
107B CF       0920        RST  RIN
107C C9       0930        RET
              0940 ;
              0950 ;
107D 0608     0960 TALK   LD   B,8
107F 4E       0970        LD   C,(HL)
1080 3E05     0980 BTEOUT LD   A,WAIT
1082 FF       0990        RST  RDEL
1083 79       1000        LD   A,C
1084 CB0F     1010        RRC  A
1086 CB0F     1020        RRC  A
1088 CB0F     1030        RRC  A
108A E620     1040        AND  #20
108C D300     1050        OUT  (0),A
108E CB09     1060        RRC  C
1090 10EE     1070        DJNZ BTEOUT
1092 23       1080        INC  HL
1093 B7       1090        OR   A
1094 ED52     1100        SBC  HL,DE
1096 19       1110        ADD  HL,DE
1097 20E4     1120        JR   NZ,TALK
1099 C9       1130        RET
```

# SPEISEPLAN

## von Klaus Mombaur

```
2 REM ---  Woechentlicher Speiseplan  ---
3 REM
4 REM von Klaus Mombaur auf NASCOM 2 / NAS-SYS1
5 REM [redacted] Nuernberg [redacted]
6 REM Telefon [redacted]
7 REM
8 REM  Speisegruppen:
9 REM  0=Fleisch Mo-Do,     1=kl.Fleisch Mo-Do
10 REM 2=Mehlspeisen Mo-Do,3=Gemuese Mo-Do
11 REM 4=Freitagessen,      5=Samstagessen
12 REM 6=Sonntagessen
13 CLS:SCREEN8,7:PRINT"Bitte warten"
14 DOKE3200,-8385:DOKE3202,14434
15 DOKE3204,6146 :DOKE3206,18426
16 DOKE3208,-8385:DOKE3210,14434
17 DOKE3212,6146 :DOKE3214,-12806
18 DOKE3216,-3854:DOKE3218,201
19 GOTO1600
20 CLS:RESTORE
22 PRINT
25 PRINT"       Ich fertige Ihnen einen Speisepl
an"
26 PRINT"       ==========================
=="
27 PRINT:PRINT:PRINT
30 PRINT"Moechten Sie den Plan fuer die ganze W
oche?"
32 PRINT:PRINT"Dann tippen Sie - W - ENTER
34 PRINT:PRINT
36 PRINT"Moechten Sie ein Einzelgericht?"
38 PRINT:PRINT"Dann tippen Sie - E - ENTER
"
40 INPUT"                                    ";E$
42 IF E$="W"THEN 56
44 IF E$="E"THEN 1200
46 PRINT"Falsche Taste. Bitte wiederholen!":GOT
040
56 CLS
58 SCREEN 1,15
60 PRINT"       * * * S p e i s e p l a n * * *"
;
70 FOR C=2954TO3000 STEP2
80 DOKE C+64,DEEK(C)
90 NEXT C
100 PRINT CHR$(27)
160 GOSUB 900
170 SCREEN1,1
175 PRINT"Mo.:";A$    :MO=S
180 S=S+1:IFS>3THENS=0
185 GOSUB920
190 SCREEN1,3
195 PRINT"Di.:";A$    :DI=S
200 S=S+1:IFS>3THENS=0
205 GOSUB920
210 SCREEN1,5
220 PRINT"Mi.:";A$    :MI=S
230 S=S+1:IFS>3THENS=0
240 GOSUB 920
245 SCREEN1,7
250 PRINT"Do.:";A$    :DO=S
260 Z=4
270 GOSUB915
280 SCREEN1,9
290 PRINT"Fr.:";A$ :FR=S
300 Z=5
310 GOSUB915
320 SCREEN1,11
330 PRINT"Sa.:";A$ :SA=S
340 Z=6
350 GOSUB 915
360 SCREEN1,13
370 PRINT"So.:";A$ :SO=S
380 SCREEN 1,14
390 PRINT"Wuenschen Sie Korrektur?    (Tag ange
ben!)"
400 REM Per USR die 2 ersten Buchst.abfragen
410 DOKE4100,3200
420 K=USR(2)
430 IF K=20301THEN 500
440 IF K=18756THEN 540
450 IF K=18765THEN 590
460 IF K=20292THEN 640
470 IF K=21062THEN 690
480 IF K=16723THEN 740
485 IF K=20307THEN 790
491 SCREEN1,14
492 PRINT"Den Tag gibt es nicht! Bitte nochmals
!  "
495 GOTO 410
500 S=MO
510 SCREEN1,1:PRINTCHR$(27)
520 GOSUB920
530 SCREEN1,1:PRINT"Mo.:";A$:GOTO380
540 S=DI
550 SCREEN1,3:PRINTCHR$(27)
560 GOSUB920
570 SCREEN1,3:PRINT"Di.:";A$:GOTO380
590 S=MI
600 SCREEN1,5:PRINTCHR$(27)
610 GOSUB920
620 SCREEN1,5:PRINT"Mi.:";A$:GOTO380
```

```
640 S=DO
650 SCREEN1,7:PRINTCHR$(27)
660 GOSUB920
670 SCREEN1,7:PRINT"Do.:";A$:GOTO380
690 S=FR
700 Z=4
710 SCREEN1,9:PRINTCHR$(27)
720 GOSUB915
730 SCREEN1,9:PRINT"Fr.:";A$:GOTO380
740 S=SA
750 Z=5
760 SCREEN1,11:PRINTCHR$(27)
770 GOSUB915
780 SCREEN1,11:PRINT"Sa.:";A$:GOTO380
790 S=SO
800 Z=6
810 SCREEN1,13:PRINTCHR$(27)
820 GOSUB915
830 SCREEN1,13:PRINT"So.:";A$:GOTO380
900 REM Zufallszahl erzeugen fuer Mo - Do
910 Z=RND(1):Z=INT(Z*10):IF Z>3THEN910
915 S=Z
920 IFS=0THENG=G0
925 IFS=1THENG=G1
930 IFS=2THENG=G2
935 IFS=3THENG=G3
940 IFS=4THENG=G4
945 IFS=5THENG=G5
950 IFS=6THENG=G6
1000 REM Zufallszahl entspr. Menge Gruppe G
1010 Z=RND(1):Z=INT(Z*1000):IFZ=>G THEN1010
1020 RESTORE
1030 FOR X=0 TO Z
1050 READ A,A$
1060 IF A<>S THEN 1050
1070 NEXT X
1080 RETURN
1200 REMZuf.zahl innerhalb Menge aller Gerichte
1210 GG=G0+G1+G2+G3+G4+G5+G6
1215 CLS:PRINT"                Ich denke nach"
1220 Z=RND(1):Z=INT(Z*1000):IFZ>GG THEN1220
1230 RESTORE
1240 FOR Y=0 TO Z
1250 READA,A$
1260 NEXT Y
1270 CLS
1280 PRINT:PRINT:PRINT
1290 PRINT"Mein Vorschlag:"
1300 PRINT
1310 PRINTA$
1320 PRINT:PRINT
1330 PRINT"Moechten Sie noch einen Vorschlag? (
J/N)"
1340 INPUT"                                    ";N
$
1350 IF N$="J"THEN 1200
1360 IF N$<>"N"THEN 1340
1370 SCREEN1,15
1380 PRINT"Herzlichen Dank. Bis zum naechsten M
al!"
1390 FOR X =1 TO 2000
1400 NEXT X
1410 CLS
1500 END
1600 READ G:G=INT(G): READX$
1610 IF G=0 THEN G0=G0+1
1620 IF G=1 THEN G1=G1+1
1630 IF G=2 THEN G2=G2+1
1640 IF G=3 THEN G3=G3+1
1650 IF G=4 THEN G4=G4+1
1660 IF G=5 THEN G5=G5+1
1670 IF G=6 THEN G6=G6+1
1680 IF G=9 THEN 20
1690 GOTO 1600
5000 DATA1,Paprikaschoten mit Reis und Salat
5001 DATA1,Moehreneintopf mit Dosenrindfleisch
5002 DATA1,Pichelsteiner
5003 DATA2,Semmelnudeln mit Apfelmus
5004 DATA0,Gulasch mit Reis oder Nudeln
:

5064 DATA6,Ueberbackene Champignonkotelette
5065 DATA6,Schweinefillet auf Gemuesebett
5066 DATA3,Risotto Tessin
5067 DATA2,Makkaroniauflauf
5068 DATA0,Schmorbraten mit Erbsen und Moehren
5069 DATA4,Schellfisch mit Buttersauce
6000 DATA9,Mein Vorrat ist erschoepft!
```

Die Speisen sind in 7 Gruppen eingeteilt (Zeile 9-12). Sie können natürlich andere oder weitere Speisen einbringen, diese hier sollen nur Beispiel sein. Geben Sie unter DATA erst die Gruppe an, dann Komma, dann das Gericht, solange der Speicherplatz reicht. Der Inhalt der Zeile 6000 muß aber als letzte DATA-Anweisung stehen bleiben. Und nun guten Appetit Ihnen und besonders Ihrer Frau.

# Mini-PILOT

## von W. Mayer-Gürr

Angeregt durch einen Artikel in der Zeitschrift Byte soll hier ein Interpreter für ein Subset der Programmiersprache Pilot vorgestellt werden. Ein wesentliches Element dieser Sprache ist der Vergleich einer Eingabe mit einem vorgegebenen Zeichen und einer Entscheidung über den weiteren Verlauf. Der Interpreter verarbeitet Zeilen, an derem Anfang immer ein Befehl stehen muß. Folgende Befehle sind bekannt:

| | |
|---|---|
| T:text | bring 'text' auf den Schirm |
| A: | erwartet Zeichen |
| M:x | vergleicht x mit dem eingegebenen Zeichen und setzt ein Flag (Y oder N) |
| J:n | bei n=0 Sprung zum letzten A: bei n=1 bis n=9 Sprung zur n. Markierung |
| S: | Beendet Programm, Sprung zum Editor |
| Y N | Bedingung vor einem Befehl, wird nur ausgeführt, wenn Flag gesetzt ist |
| * | Markierung für Sprungbefehl |

Als guter Interpreter hat Mini Pilot auch einen Texteditor. Das Programm wird normal eingeben. Will man zum Programmanfang zurück, reicht das '&' Zeichen, mit Backspace läßt sich ein Fehler korrigieren, '/' zeigt die nächste Zeile. Soll eine Zeile ersetzt werden, werden die vorhergehenden gelistet, dann wird neu eingegeben. Die neue Zeile darf nicht größer als die fehlerhafte sein, sie muß mit '%' statt Newline abgeschlossen werden. Gestartet wird das Programm mit dem '$' Zeichen.

Obwohl die Sprache als Einstieg für Kinder gepriesen wird, kann ich keine großen Vorteile sehen. Deshalb ist der Interpreter 'nur' in Pascal geschrieben, das geht schneller.

```
PROGRAM Mini_Pilot;
 CONST platz=5000; bs=CHR(8); eol=CHR(10);
 VAR   ort,stelle,i : INTEGER;
       ende,flag,zbuf          :CHAR;
       prog:ARRAY[1..platz]OF CHAR;

PROCEDURE lese;
 BEGIN
  READ(zbuf);
  IF EOLN=TRUE THEN zbuf:=eol
 END;

PROCEDURE liste;
 VAR i:INTEGER;
 BEGIN i:=0;
  REPEAT
   zbuf :=prog[ort];ort:=ort+1;i:=i+1;
   WRITE(zbuf)
  UNTIL (i>80) OR (zbuf=eol)
END;
```

```
PROCEDURE ausfuehren;
 VAR fertig:BOOLEAN;
 BEGIN ort:=1;fertig := FALSE;
  REPEAT
   zbuf :=prog[ort];IF zbuf< '*' THEN zbuf :='*';
   IF NOT
    (zbuf IN ['*','Y','N','A','M','J','T','S'])
    THEN liste
    ELSE CASE zbuf OF
     '*'     :ort :=ort+1;
     'Y','N':IF zbuf=flag THEN ort :=ort+1
                          ELSE REPEAT
                                 zbuf:=prog[ort];
                                 ort :=ort+1
                               UNTIL zbuf=eol;
     'A'     :BEGIN
                stelle:=ort;
                lese;
                ende:=zbuf;
                ort:=ort+2
              END;
     'M'     :BEGIN
                IF ende=prog[ort+2]
                   THEN flag :='Y'
                   ELSE flag :='N';
                ort:=ort+3
              END;
     'J'     :IF prog[ort+2] ="0'
                THEN ort:=stelle
                ELSE
                 BEGIN
                  i:=ORD(prog[ort+2])-48;
                  REPEAT ort:=ort+1;
                   IF prog[ort]='*' THEN i:=i-1;
                  UNTIL I=0
                 END;
     'T'     :BEGIN
                ort:=ort+2;
                liste
              END;
     'S'     :BEGIN
                fertig:=TRUE;
                ort    :=1
              END
     END
   UNTIL fertig
END;

BEGIN
 zbuf:='&';
  WHILE TRUE DO BEGIN
   IF zbuf ='&' THEN ort:=1
                ELSE IF zbuf= bs
                        THEN ort:=ort-1
   ELSE IF zbuf='/' THEN liste
   ELSE IF zbuf='$' THEN ausfuehren
   ELSE IF zbuf='%' THEN
    BEGIN i:=0;
     WHILE (i<80) AND (prog[ort] <> eol) DO
      BEGIN
       prog[ort]:=CHR(0);
       ort:=ort+1
      END;
     prog[ort]:= eol;ort:=ort+1
    END
   ELSE BEGIN
         prog[ort]:=zbuf;
         ort:=ort+1;
        END;
   lese
  END
END.
```

Das NIM(M)- Spiel ist ja hinreichend bekannt, und viele haben es schon als Einstieg zum Programmieren benutzt. Wir haben es bereits in Heft 4/5-80 für NASSYS und in Heft 3-81 für T2 veröffentlicht. Nun im folgenden eine Version in Mini- PILOT von Wolfgang Mayer- Gürr und eine BASIC-Version von Christian Peter.
Die Sammlung ist komplett (?). Eine schöne Vergleichsmöglichkeit zu den einzelnen Sprachen für Analytiker!

# PILOT - NIM

## von W. Mayer-Gürr

```
T:Das NIM-Spiel in Mini-Pilot.
T:Wir haben 7 Streichhoelzer. Du darfst 1, 2
T:oder 3 wegnehmen. Der, der das letzte nimmt,
T:hat verloren. Wieviele nimmst du?
A:
M:1
YJ:1
M:2
YJ:2
M:3
YJ:6
T:Du darfst nur 1, 2 oder 3 nehmen!
J:0
*T:Es bleiben 6, ich nehme 1, bleiben 5.
T:Wieviele?
A:
M:1
YJ:5
M:2
YJ:4
M:3
YJ:3
T:Nur 1, 2 oder 3!
J:0
*T:Es bleiben 5, ich nehme 1, bleiben 4.
T:Wieviele?
A:
M:1
YJ:3
M:2
YJ:2
M:3
YJ:1
T:Nun lern es endlich:1,2 oder 3!
J:0
*T:Es bleibt nur eins.Du hast gewonnen!
J:5
*T:Es bleiben 2, ich nehme eins!
J:3
*T:Es bleiben 3, ich nehme zwei!
J:2
*T:Es bleiben 4, ich nehme drei!
*T:Wieviele?
A:
M:1
NT:Du hast keine Wahl, nimm 1!
NT:Wieviele?
NJ:0
T:Ich habe gewonnen!
*T:Noch ein Spiel -> Dollarzeichen!
S:
```

# BASIC - NIMM

## von Christian Peter

```
10 DIMA(3):A(1)=3:A(2)=4:A(3)=5
20 GOSUB 980
30 INPUT"Wollen Sie beginnen";I$
40 IF LEFT$(I$,1)<>"J" THEN 70
50 INPUT"Ihr Zug";I,D
51 I=INT(I):D=INT(D)
52 IF I<1 OR I>3 THEN GOSUB150:GOTO50
```

```
53 IFD<>1THENGOSUB150:GOTO50
55 IF A(I)-D <>0 THEN GOSUB150:GOTO50
60 GOSUB 1040:GOSUB980
70 IFA(1)=0ANDA(2)=0ANDA(3)=1THEN10000
80 IFA(1)=0ANDA(2)=1ANDA(3)=0THEN10000
90 IFA(1)=1ANDA(2)=0ANDA(3)=0THEN10000
100 GOSUB1070:GOSUB3300:GOSUB1040:GOSUB980
110 IFA(1)=0ANDA(2)=0ANDA(3)=1THEN10050
120 IFA(1)=0ANDA(2)=1ANDA(3)=0THEN10050
130 IFA(1)=1ANDA(2)=0ANDA(3)=0THEN10050
140 GOTO50
150 SCREEN1,12:PRINTSPC(96)
160 SCREEN15,12:PRINT"NICHT SCHUMMELN!"
170 FORJ=1TO1000:NEXT:GOSUB980:RETURN
980 CLS
990 SCREEN8,1:PRINT"*******  NIMM-SPIEL  ******
*"
1000 FORI=1TO3:SCREEN21-I,3+2*I
1010 IFA(I)THENFORJ=1TOA(I):PRINT"I ";:NEXTJ
1020 NEXT I
1030 SCREEN1,12:RETURN
1040 SCREEN 21-I+(A(I)-D)*2,3+2*I
1050 FORJ=1TOD:PRINT"* ";:NEXT
1060 A(I)=A(I)-D:FORJ=1TO300:NEXT:RETURN
1070 IFA(1)=0ORA(2)=0ORA(3)=0THEN2000
1080 IFA(1)=A(2)ANDA(1)<>1THENI=3:D=A(3):RETURN
1085 IFA(1)=A(2)ANDA(3)<>1THENI=3:D=A(3)-1:RETUR
N
1090 IFA(1)=A(3)ANDA(1)<>1THENI=2:D=A(2):RETURN
1095 IFA(1)=A(3)ANDA(2)<>1THENI=2:D=A(2)-1:RETUR
N
1100 IFA(2)=A(3)ANDA(2)<>1THENI=1:D=A(1):RETURN
1105 IFA(2)=A(3)ANDA(1)<>1THENI=1:D=A(1)-1:RETUR
N
1110 FORJ=3TO1STEP-1
1120 FORY=A(J)-1TO1STEP-1
1130 D1=A(J)-Y
1140 IFD1=A(1)ANDJ<>1ORD1=A(2)ANDJ<>2ORD1=A(3)A
NDJ<>3ORA(J)=1THEN1160
1150 I=J:D=Y:RETURN
1160 NEXTY,J
1170 I=INT(RND(8)*3+1):D=INT(RND(5)*A(J)+1):RET
URN
2000 IFA(1)=0ANDA(2)=0THENI=3:D=A(3)-1:RETURN
2010 IFA(1)=0ANDA(3)=0THENI=2:D=A(2)-1:RETURN
2020 IFA(2)=0ANDA(3)=0THENI=1:D=A(1)-1:RETURN
2030 IFA(1)<>0THEN2090
2040 IFA(2)<>A(3)ANDA(3)<>1THENI=2:D=A(2)-A(3):RE
TURN
2050 IFA(2)<>A(3)ANDA(2)<>1THENI=3:D=A(3)-A(2):RE
TURN
2060 IFA(2)<>A(3)THENI=2:D=A(2):RETURN
2070 IFA(3)<>A(2)THENI=3:D=A(3):RETURN
2080 IFA(2)=1THENI=3:D=1:RETURN
2090 IFA(2)<>0THEN3050
3000 IFA(1)<>A(3)ANDA(3)<>1THENI=1:D=A(1)-A(3):RE
TURN
3010 IFA(1)<>A(3)ANDA(1)<>1THENI=3:D=A(3)-A(1):RE
TURN
3020 IFA(1)<>A(3)THENI=1:D=A(1):RETURN
3030 IFA(1)<>A(3)THENI=3:D=A(3):RETURN
3040 IFA(1)=1THENI=3:D=1:RETURN
3050 IFA(3)<>0THEN3110
3060 IFA(1)<>A(2)ANDA(2)<>1THENI=1:D=A(1)-A(2):RE
TURN
3070 IFA(1)<>A(2)ANDA(1)<>1THENI=2:D=A(2)-A(1):RE
TURN
3080 IFA(1)<>A(2)THENI=1:D=A(1):RETURN
3090 IFA(1)<>A(2)THENI=2:D=A(2):RETURN
3100 IFA(1)=1THENI=2:D=1:RETURN
3110 R=INT(RND(8)*3+1)
3120 IFA(R)THENI=R:D=1:RETURN
3130 GOTO3110
3200 END
3300 SCREEN1,12:PRINTSPC(96)
3310 SCREEN30,12:PRINT"Mein Zug:";I;",";D
3320 FORJ=1TO700:NEXT:RETURN
10000 PRINT"SIE HABEN GEWONNEN!":GOTO10060
10050 PRINT"SIE HABEN VERLOREN!"
10060 INPUT"Noch ein Spiel";Q$
10070 IFLEFT$(Q$,1)="N"THENCLS:SCREEN20,8:PRINT
"SCHADE!":END
10080 A(1)=3:A(2)=4:A(3)=5:GOSUB980:GOTO40
```

# BCD-ARITHMETIK

**von Günter Kreidl**

## Günter Kreidl

*Im Januarheft habe ich einige Rechenroutinen für das BCD-Format vorgeführt. In einer etwas einfacheren Ausführung laufen diese Programme seit anderthalb Jahren in einem Fakturierprogramm. Für die Darstellung im NASCOM-Journal habe ich einige Routinen "verbessert". Dabei haben sich leider einige kleine Fehler eingeschlichen, die allerdings nur in ganz speziellen Anwendungen auftreten. So weigert sich z.B. die Ausgaberoutine RBAUSG beharrlich, Zahlen ohne Nachkommastellen auszugeben. Die verbesserten Routinen werden hier nochmals abgedruckt, zusammen mit einer BCD-Division, die damals noch nicht enthalten war. Die Speicherbelegung wurde so gewählt, daß sich die Routinen direkt an die im Januarheft veröffentlichten anschließen. Die Kommentare habe ich aus Platzgründen weggelassen. Sie können im Januarheft nachgelesen werden. Noch einige Bemerkungen zu dem Divisionsprogramm. Es benötigt 3 Rechenregister, die soviele Bytes umfassen müssen, wie die Zahlen maximal Stellen aufweisen. R1 und R2 enthalten die Operanden und zwar in der höherwertigen ("rechten") Hälfte. Wo sich das Komma befinden soll, ist für die Routine selbst belanglos, es muß nur bei beiden Zahlen an der gleichen Stelle stehen (das besorgt die Eingaberoutine!). R3 enthält das Ergebnis und zwar in der zweiten "Hälfte" die Nachkomma-, in der ersten die Vorkommastellen. In IX und IY müssen Adressen der Operandenregister ("rechts") übergeben werden, HL muß auf die Mitte des Ergebnisregisters zeigen (bei 16 Stellen auf die 8. Adresse des Registers!) und in B wird die max. Stellenzahl = Registerbreite in Bytes übergeben. Es werden alle, auch die Austauschregister, verändert. Die Programmierung stellt einen Kompromiß zwischen Speicherplatzbedarf, Arbeitsgeschwindigkeit und Durchsichtigkeit des Programms dar. Die Division zweier 128-stelliger Zahlen benötigt ca. 4-5 Sekunden. Auch das Ergebnis hat dann 128 Stellen. Eine 16-stellige Division benötigt etwa 60 ms.*

```
                2440 ;BCD-Routinen
                2450 ;Verbessert:
                2460 ; BCDIN und RBAUSG
                2470 ;Neu: BCD-Division
                2480 ;G.K. - 5.8.82
                2490 ;benutzte Unterprogramme
                2500 ;und Kommentare
                2510 ;siehe NASCOM-Journal 1/82:
0D00            2520 ADDBCD EQU  £D00
0D12            2530 SUBBCD EQU  £D12
0D24            2540 BCDLI  EQU  £D24
0D2B            2550 BCDNR  EQU  £D2B
0D8A            2560 BCDTST EQU  £D8A
0D93            2570        ORG  £D93
0D93 C5         2580 BCDIN  PUSH BC
0D94 E5         2590        PUSH HL
0D95 CB39       2600        SRL  C
0D97 78         2610        LD   A,B
0D98 91         2620        SUB  C
0D99 3D         2630        DEC  A
0D9A 4F         2640        LD   C,A
0D9B 0600       2650        LD   B,0
0D9D 09         2660        ADD  HL,BC
0D9E 0E00       2670        LD   C,0
0DA0 1A         2680 COUNT1 LD   A,(DE)
0DA1 FE2C       2690        CP   ",
0DA3 2811       2700        JR   Z,COUNT2
0DA5 FE2E       2710        CP   ".
0DA7 280D       2720        JR   Z,COUNT2
0DA9 FE20       2730        CP   £20
0DAB 2818       2740        JR   Z,TRANS2
0DAD CD8A0D     2750        CALL BCDTST
0DB0 3845       2760        JR   C,ERR
0DB2 04         2770        INC  B
0DB3 13         2780        INC  DE
0DB4 18EA       2790        JR   COUNT1
0DB6 13         2800 COUNT2 INC  DE
0DB7 1A         2810        LD   A,(DE)
0DB8 FE20       2820        CP   £20
0DBA 2809       2830        JR   Z,TRANS2
0DBC CD8A0D     2840        CALL BCDTST
0DBF 3836       2850        JR   C,ERR
0DC1 04         2860        INC  B
0DC2 0C         2870        INC  C
0DC3 18F1       2880        JR   COUNT2
0DC5 1B         2890 TRANS2 DEC  DE
0DC6 1A         2900        LD   A,(DE)
0DC7 FE2C       2910        CP   ",
0DC9 28FA       2920        JR   Z,TRANS2
0DCB FE2E       2930        CP   ".
0DCD 28F6       2940        JR   Z,TRANS2
0DCF E60F       2950        AND  £F
0DD1 ED67       2960        RRD
0DD3 05         2970        DEC  B
0DD4 2004       2980        JR   NZ,TRANS3
0DD6 04         2990        INC  B
0DD7 AF         3000        XOR  A
0DD8 180C       3010        JR   TRANS1
0DDA 1B         3020 TRANS3 DEC  DE
0DDB 1A         3030        LD   A,(DE)
0DDC FE2C       3040        CP   ",
0DDE 28FA       3050        JR   Z,TRANS3
0DE0 FE2E       3060        CP   ".
0DE2 28F6       3070        JR   Z,TRANS3
0DE4 E60F       3080        AND  £F
0DE6 ED67       3090 TRANS1 RRD
0DE8 2B         3100        DEC  HL
0DE9 10DA       3110        DJNZ TRANS2
0DEB E1         3120        POP  HL
0DEC D1         3130        POP  DE
0DED AF         3140        XOR  A
0DEE B1         3150        OR   C
0DEF 2804       3160        JR   Z,ENDE
0DF1 42         3170        LD   B,D
0DF2 CD2B0D     3180        CALL BCDNR
0DF5 AF         3190 ENDE   XOR  A
0DF6 C9         3200        RET
0DF7 DF         3210 ERR    RST  £18
0DF8 6B         3220        DEFB £6B;ERRMS
0DF9 E1         3230        POP  HL
0DFA C1         3240        POP  BC
0DFB 37         3250        SCF
0DFC C9         3260        RET
0DFD 00         3270        NOP
0DFE CB39       3280 RBAUSG SRL  C
0E00 78         3290        LD   A,B
0E01 91         3300        SUB  C
0E02 47         3310        LD   B,A
0E03 7E         3320 TEST1  LD   A,(HL)
0E04 B7         3330        OR   A
0E05 200D       3340        JR   NZ,TEST2
0E07 23         3350        INC  HL
0E08 DF         3360        RST  £18
0E09 69         3370        DEFB £69;SPACE
0E0A DF         3380        RST  £18
0E0B 69         3390        DEFB £69
0E0C 10F5       3400        DJNZ TEST1 .
0E0E EF         3410        RST  £28
0E0F 08         3420        DEFB £08,£30,0
0E12 1815       3430        JR   KOMMA
0E14 E6F0       3440 TEST2  AND  £F0
0E16 7E         3450        LD   A,(HL)
0E17 200B       3460        JR   NZ,AUSG2
0E19 F5         3470        PUSH AF
0E1A DF         3480        RST  £18
0E1B 69         3490        DEFB £69
0E1C F1         3500        POP  AF
0E1D DF         3510        RST  £18
0E1E 7A         3520        DEFB £7A;B1HEX
0E1F 23         3530        INC  HL
0E20 05         3540        DEC  B
0E21 2806       3550        JR   Z,KOMMA
0E23 7E         3560 AUSG1  LD   A,(HL)
0E24 DF         3570 AUSG2  RST  £18
0E25 68         3580        DEFB £68;B2HEX
0E26 23         3590        INC  HL
0E27 10FA       3600        DJNZ AUSG1
0E29 79         3610 KOMMA  LD   A,C
0E2A B7         3620        OR   A
0E2B C8         3630        RET  Z
0E2C 3E2C       3640        LD   A,",
0E2E F7         3650        RST  £30
0E2F 7E         3660 AUSG3  LD   A,(HL)
0E30 DF         3670        RST  £18
0E31 68         3680        DEFB £68
0E32 23         3690        INC  HL
0E33 0D         3700        DEC  C
0E34 20F9       3710        JR   NZ,AUSG3
0E36 C9         3720        RET
0E37 E5         3730 CLEAR  PUSH HL
0E38 D1         3740        POP  DE
0E39 13         3750        INC  DE
0E3A 0B         3760        DEC  BC
0E3B AF         3770        XOR  A
0E3C 77         3780        LD   (HL),A
0E3D EDB0       3790        LDIR
0E3F C9         3800        RET
0E40 CB39       3810 RNDBCD SRL  C
0E42 0600       3820        LD   B,0
0E44 09         3830        ADD  HL,BC
0E45 7E         3840        LD   A,(HL)
0E46 FE32       3850        CP   50
0E48 D8         3860        RET  C
0E49 2B         3870 RND    DEC  HL
0E4A 7E         3880        LD   A,(HL)
0E4B 3C         3890        INC  A
0E4C 27         3900        DAA
0E4D 77         3910        LD   (HL),A
0E4E D0         3920        RET  NC
0E4F 18F8       3930        JR   RND
                3940 ;Anzahl führ. Leerst.
                3950 ;HL: Reg. links
0E51 AF         3960 LST    XOR  A
0E52 4F         3970        LD   C,A
0E53 ED6F       3980 LST1   RLD
0E55 47         3990        LD   B,A
0E56 ED67       4000        RRD
0E58 78         4010        LD   A,B
0E59 B7         4020        OR   A
0E5A C0         4030        RET  NZ
```

```
0E5B 0C       4040          INC   C
0E5C ED67     4050          RRD
0E5E 47       4060          LD    B,A
0E5F ED6F     4070          RLD
0E61 78       4080          LD    A,B
0E62 B7       4090          OR    A
0E63 C0       4100          RET   NZ
0E64 0C       4110          INC   C
0E65 23       4120          INC   HL
0E66 18EB     4130          JR    LST1
              4140 ;Ergebnis in C
              4150 ;N-Mal Linksschieben:
              4160 ;C: Anzahl
              4170 ;B: Reg.-Länge
              4180 ;HL: Reg. "rechts"
0E68 C5       4190 NBCDLI   PUSH  BC
0E69 E5       4200          PUSH  HL
0E6A CD240D   4210          CALL  BCDLI
0E6D E1       4220          POP   HL
0E6E C1       4230          POP   BC
0E6F 0D       4240          DEC   C
0E70 20F6     4250          JR    NZ,NBCDLI
0E72 C9       4260          RET
              4270 ;DIV: R1 / R2 nach R3
              4280 ;IY:R1,IX:R2,"rechts"
              4290 ;HL: R3 1. VK-Stelle
              4300 ;B: Reg.Länge
0E73 E5       4310 DIVBCD   PUSH  HL
0E74 C5       4320          PUSH  BC
0E75 DDE5     4330          PUSH  IX
0E77 FDE5     4340          PUSH  IY
0E79 D9       4350          EXX
0E7A D1       4360          POP   DE
0E7B E1       4370          POP   HL
0E7C C1       4380          POP   BC
0E7D C5       4390          PUSH  BC
0E7E 48       4400          LD    C,B
0E7F 0600     4410          LD    B,0
0E81 0B       4420          DEC   BC
0E82 AF       4430          XOR   A
0E83 ED42     4440          SBC   HL,BC
0E85 EB       4450          EX    DE,HL
0E86 AF       4460          XOR   A
0E87 ED42     4470          SBC   HL,BC
0E89 C1       4480          POP   BC
0E8A C5       4490          PUSH  BC
0E8B D9       4500          EXX
0E8C FDE5     4510          PUSH  IY
0E8E E1       4520          POP   HL
0E8F 48       4530          LD    C,B
0E90 CD680E   4540          CALL  NBCDLI
0E93 D9       4550          EXX
0E94 E5       4560          PUSH  HL
0E95 C5       4570          PUSH  BC
0E96 CD510E   4580          CALL  LST
0E99 79       4590          LD    A,C
0E9A C1       4600          POP   BC
0E9B E1       4610          POP   HL
0E9C EB       4620          EX    DE,HL
0E9D D9       4630          EXX
0E9E 57       4640          LD    D,A
0E9F C1       4650          POP   BC
0EA0 C5       4660          PUSH  BC
0EA1 DDE5     4670          PUSH  IX
0EA3 E1       4680          POP   HL
0EA4 48       4690          LD    C,B
0EA5 CD680E   4700          CALL  NBCDLI
0EA8 D9       4710          EXX
0EA9 E5       4720          PUSH  HL
0EAA C5       4730          PUSH  BC
0EAB CD510E   4740          CALL  LST
0EAE 79       4750          LD    A,C
0EAF C1       4760          POP   BC
0EB0 E1       4770          POP   HL
0EB1 D9       4780          EXX
0EB2 C1       4790          POP   BC
0EB3 C5       4800          PUSH  BC
0EB4 5F       4810          LD    E,A
0EB5 7A       4820          LD    A,D
0EB6 1600     4830          LD    D,0
0EB8 93       4840          SUB   E
0EB9 2832     4850          JR    Z,DIVO
0EBB 3018     4860          JR    NC,DIVR
0EBD 2F       4870 DIVL     CPL
0EBE 3C       4880          INC   A
0EBF 5F       4890          LD    E,A
0EC0 4F       4900          LD    C,A
0EC1 DDE5     4910          PUSH  IX
0EC3 E1       4920          POP   HL
0EC4 CD680E   4930          CALL  NBCDLI
0EC7 C1       4940          POP   BC
0EC8 48       4950          LD    C,B
0EC9 E1       4960          POP   HL
0ECA CB3B     4970          SRL   E
0ECC F5       4980          PUSH  AF
0ECD AF       4990          XOR   A
0ECE ED52     5000          SBC   HL,DE
0ED0 F1       5010          POP   AF
0ED1 381F     5020          JR    C,DIV1
0ED3 182D     5030          JR    DIV2
0ED5 5F       5040 DIVR     LD    E,A
0ED6 D9       5050          EXX
0ED7 E5       5060          PUSH  HL
0ED8 C5       5070          PUSH  BC
0ED9 4F       5080          LD    C,A
0EDA CD2B0D   5090          CALL  BCDNR
0EDD C1       5100          POP   BC
0EDE E1       5110          POP   HL
0EDF D9       5120          EXX
0EE0 C1       5130          POP   BC
0EE1 48       5140          LD    C,B
0EE2 E1       5150          POP   HL
0EE3 CB3B     5160          SRL   E
0EE5 F5       5170          PUSH  AF
0EE6 ED5A     5180          ADC   HL,DE
0EE8 F1       5190          POP   AF
0EE9 3807     5200          JR    C,DIV1
0EEB 1815     5210          JR    DIV2
0EED C1       5220 DIVO     POP   BC
0EEE 48       5230          LD    C,B
0EEF E1       5240          POP   HL
0EF0 1810     5250          JR    DIV2
0EF2 110010   5260 DIV1     LD    DE,LNIB
0EF5 CD220F   5270          CALL  DILOOP
0EF8 7E       5280          LD    A,(HL)
0EF9 83       5290          ADD   A,E
0EFA 27       5300          DAA
0EFB 77       5310          LD    (HL),A
0EFC 0D       5320          DEC   C
0EFD 2816     5330          JR    Z,DIVEND
0EFF CD160F   5340          CALL  DIVROT
0F02 110001   5350 DIV2     LD    DE,RNIB
0F05 CD220F   5360          CALL  DILOOP
0F08 7E       5370          LD    A,(HL)
0F09 83       5380          ADD   A,E
0F0A 27       5390          DAA
0F0B 77       5400          LD    (HL),A
0F0C 0D       5410          DEC   C
0F0D 2806     5420          JR    Z,DIVEND
0F0F 23       5430          INC   HL
0F10 CD160F   5440          CALL  DIVROT
0F13 18DD     5450          JR    DIV1
0F15 C9       5460 DIVEND   RET
0F16 D9       5470 DIVROT   EXX
0F17 E5       5480          PUSH  HL
0F18 C5       5490          PUSH  BC
0F19 0E01     5500          LD    C,1
0F1B CD2B0D   5510          CALL  BCDNR
0F1E C1       5520          POP   BC
0F1F E1       5530          POP   HL
0F20 D9       5540          EXX
0F21 C9       5550          RET
0F22 DDE5     5560 DILOOP   PUSH  IX
0F24 FDE5     5570          PUSH  IY
0F26 C5       5580          PUSH  BC
0F27 CD120D   5590          CALL  SUBBCD
0F2A C1       5600          POP   BC
0F2B FDE1     5610          POP   IY
0F2D DDE1     5620          POP   IX
0F2F 3806     5630          JR    C,DILEND
```

```
OF31 7B       5640         LD   A,E
OF32 82       5650         ADD  A,D
OF33 27       5660         DAA
OF34 5F       5670         LD   E,A
OF35 18EB     5680         JR   DILOOP
OF37 DDE5     5690 DILEND  PUSH IX
OF39 FDE5     5700         PUSH IY
OF3B C5       5710         PUSH BC
OF3C CD000D   5720         CALL ADDBCD
OF3F C1       5730         POP  BC
OF40 FDE1     5740         POP  IY
OF42 DDE1     5750         POP  IX
OF44 C9       5760         RET
1000          5770 LNIB    EQU  £1000
0100          5780 RNIB    EQU  £0100
```

# KURVEN-DARSTELLUNG

**eingesandt v. Markus Caesar**

Das vorliegende Programm ermöglicht die grafische Darstellung beliebiger Funktionen. Erforderlich ist lediglich die 48x96 Klötzchengrafik des NASCOM 2.

```
7 PRINT"AUTOR:     Martin Pape,
8 PRINT"                 Solingen
9 INPUT A$
10 DIMY(190)
11 CLS
12 SCREEN 13,8 : PRINT "***  Die Kurve  ***"
13 SCREEN 1,1
14 PRINT"Ergaenzen Sie bitte den Ausdruck der"
15 INPUT"Funktion: F(X)=";A$
16 IF LEN(A$)<1 THEN25
17 PRINT"25 DEF FNF(X)=";A$
18 PRINT"A$=";CHR$(34);A$;CHR$(34)
19 PRINT
20 PRINT"RUN 24"
21 PRINTCHR$(19);CHR$(19);CHR$(19);
22 PRINTCHR$(19);CHR$(19);CHR$(19)
23 END
24 DIMY(190)
25 DEF FNF(X)=2*X
26 A$="F(X)="+A$
27 INPUT "von ";A
28 INPUT "bis ";B
29 IF A=B THEN CLEAR : GOTO11
30 DX=(B-A)/190
31 FOR I=0 TO 190
32 C=A+DX*I
33 Y(I)=FNF(C)
34 NEXT
35 M=Y(0) : N=Y(0)
36 FOR I=1 TO 190
37 IF Y(I)>M THEN M=Y(I)
38 IF Y(I)<N THEN N=Y(I)
39 NEXT
40 DY=(M-N)/43
41 IFDY=0THENDY=.000001
42 NU=44-INT(-N/DY+.5)
43 IFDX=0THENDX=.000001
44 X0=INT(-A/DX+1.1)
45 CLS
46 IF NU>41 THEN56
47 IF NU< 0 THEN56
48 NU=INT(NU/3+1.1)
49 FOR I=1 TO 45
50 SCREEN I,NU : PRINT CHR$(175);
51 NEXT
52 PRINT "x"
53 FORI=1TOLEN(A$)
54 POKE3019+I,ASC(MID$(A$,I,1))
55 NEXTI
56 FOR I=0 TO 190
57 J=44-INT((Y(I)-N)/DY+.5)
58 IF X0=I THEN62
59 SET(I/2,J)
60 NEXT
61 SCREEN 1,1 : GOTO27
62 II=INT(I/4+1.5): SCREEN II,1 : PRINT "↑"
63 SCREEN II,2 : PRINT "Y"
64 FOR K=3 TO 14
65 SCREEN II,K : PRINT CHR$(174);
66 NEXT K
67 GOTO59:====================================
```

Wer keine Klötzchengrafik besitzt, kann folgende Version für Normalausrüstung verwenden.

```
154 DIM Y(46)
155 CLS
156 SCREEN 13,8 : PRINT "***  Die Kurve  ***"
157 SCREEN 1,1
158 PRINT"Ergaenzen Sie bitte den Ausdruck der"
159 INPUT"Funktion: F(X)=";A$
160 IF LEN(A$)<1 THEN166
161 PRINT"167 DEF FNF(X)=";A$
162 PRINT"RUN 165"
163 PRINTCHR$(19);CHR$(19);CHR$(19);CHR$(19)
164 END
165 DIM Y(47)
166 CLS
167 DEF FNF(X)=SIN(X)*COS(X)
168 INPUT "von ";A
169 INPUT "bis ";B
170 IF A=B THEN CLEAR : GOTO154
171 DX=(B-A)/46
172 FOR I=0 TO 46
173 C=A+DX*I
174 Y(I)=FNF(C)
175 NEXT
176 M=Y(0) : N=Y(0)
177 FOR I=1 TO 46
178 IF Y(I)>M THEN M=Y(I)
179 IF Y(I)<N THEN N=Y(I)
180 NEXT
181 DY=(M-N)/13
182 NU=14-INT(-N/DY+.5)
183 X0=INT(-A/DX+1.9)
184 CLS
185 IF NU>14 THEN191
186 IF NU< 1 THEN191
187 FOR I=1 TO 45
188 SCREEN I,NU : PRINT "-";
189 NEXT
190 PRINT "X"
191 FOR I=1 TO 47
192 J=14-INT((Y(I-1)-N)/DY+.5)
193 IF X0=I THEN197
194 SCREEN I,J : PRINT "*";
195 NEXT
196 SCREEN 1,1 : GOTO168
197 SCREEN I,1 : PRINT "↑"
198 SCREEN I,2 : PRINT "Y"
199 FOR K=3 TO 14
200 SCREEN I,K : PRINT "I"
201 NEXT K
202 GOTO194:==========================
```

# DATALINE
## von Gerhard Klement

Im Programm DATALINE ist folgende Philosophie enthalten: BASIC Programme sollen sich selbst während des Laufes verändern können; z.B Anfügen von DATA Lines, die Rechenergebnisse,Strings oder PEEK (DEEK) Inhalte beinhalten. Da während des Laufes eine Änderung des Programmes die Variablen-Felder löscht, werden die Änderungen in einen Buffer geladen, der sich am Ende der Exekution selbsttätig entleert. Für BASIC ist es so, als ob eine KBD Eingabe erfolgt. Es ist daher nötig, MEMSIZE zu begrenzen, um für den Buffer ausreichend Platz zu schaffen. Vielleicht fällt Ihnen auf, daß die Aktivierung und die Disaktivierung über SET und RESET erfolgt anstelle der sonst üblichen USR Funktion (ist nicht weiter bedeutungsvoll, war nur ein Versuch.

```
1 REM -- DATALINE V2 GENERATOR FILE F --
2 REM 31.12.1981 8h44
3 CLS:PRINT"** D A T A   L I N E   G E ";
4 PRINT"N E R A T O R **"
5 PRINT"To run this program make sure ";
6 PRINT"that BASIC MEM":PRINT"does not exeed";
7 PRINT" HEX A000 ! ":A$=""
8 INPUT"If this is OK then answer 'Y'";A$
9 IFA$="Y"ORA$="YES"THEN11
10 MONITOR
11 PRINT"TRANSFERRING MACHINE PROGRAM TO RAM"
12 PRINT"SET and RESET is now active !!!"
13 PRINT"and may be deactivatet by USR (0)
14 PRINT"Insert start and stop of DEEKING in";
15 PRINT" line 24 and":PRINT"starting line ";
16 PRINT"number in line 22 and restart
17 PRINT"PGM with 'RUN 21'"
18 GOSUB36
19 DOKE4100,-22520:A=USR(0):END
20 REM === START DEEKING HERE ===
21 DOKE4100,-22469:WIDTH49
22 LI=5000:REM *** STARTLINE ***
23 SET
24 FL=0:FORX=-22034TO-20480STEP2
25 Y$=STR$(DEEK(X))
26 IFLEFT$(Y$,1)=" "THENY$=RIGHT$(Y$,LEN(Y$)-1)
27 DA=DA+LEN(Y$):IFFL=0THENFL=1:GOTO30
28 IFDA>=38THENPRINTCHR$(8):GOTO30
29 PRINTY$",";:DA=DA+1:GOTO32
30 DA=LEN(Y$):PRINTCHR$(17)LI"DATA"Y$",";
31 LI=LI+2:REM *** LINE INCREMENT ***
32 NEXT:PRINTCHR$(8)
33 RESET:A=USR(0):END
34 REM  LINE GENERATOR
35 REM ========================
36 RESTORE39
37 FORX= -22520TO-22350STEP2
38 READI:DOKEX,I:NEXT:RETURN
39 DATA18721,8872,4181,23329,8872,4184
40 DATA17135,26229,27750,28261,15648,32
41 DATA33,-4696,-20645,4880,21229,-21043
42 DATA8697,-22379,31522,8460,-22415,30754
43 DATA-8436,8538,-192,21794,8464,-171
44 DATA22562,-8432,-6822,-20959,-8280,8817
45 DATA-22522,-20694,8976,546,-7768,-6711
46 DATA1578,-8280,8561,-22352,29407,1570
47 DATA10920,-22526,1058,6312,-2591,26079
48 DATA10993,-22526,9079,546,4520,-22528
49 DATA21229,10968,-22522,29151,3567,30018
50 DATA26214,20256,3414,-8448,10843,-22526
51 DATA-4635,1115,-4696,-7854,14462,10758
52 DATA-22522,29407,9161,546,-13912,117
53 DATA118,30208
54 REM ========================
```

# Sortier-algorithmen
## von Gerhard Klement

Folgendes Programm faßt die häufigsten Sortieralgorithmen zusammen und stellt sie gegenüber.

```
1 REM -- SORTS FILE D --
2 REM  17.6.81       20h26
3 REM
100 CLS
110 PRINT"Das Programm demonstriert:
120 PRINTTAB(17)"RIPPLE
130 PRINTTAB(17)"BUBBLE
140 PRINTTAB(17)"SHELL
150 PRINTTAB(17)"INSERTION
160 PRINTTAB(17)"HEAP
170 PRINTTAB(17)"QUICK
180 PRINT"Sorts fuer N Elemente, wobei N ";
190 PRINT" im PGM de -":PRINT"finiert ist.
200 REM -- UPSET --
210 PRINT"GENERIERUNG DES ARRAYS
220 N=30
230 DIMA(N),B(N+1)
240 FORI=1TON
250 A(I)=INT(100*RND(3))
260 NEXTI
```

```
270 REM -- PRINT --
280 FORI=1TON
290 T=INT(40*((I-1)*.1-INT((I-1)*.1))+.5)
300 PRINTTAB(T)A(I);:IFT=36THENPRINT
310 NEXT:INPUTX$
320 REM -- RIPPLE --
1000 GOSUB6180
1010 CLS:PRINT"START RIPPLE SORT";:INPUTX$
1020 M=N
1030 V=0
1040 FORI=1TOM-1
1050 IFB(I)<=B(I+1)THEN1080
1060 H=B(I):B(I)=B(I+1):B(I+1)=H
1070 V=1
1080 NEXTI
1090 IFV=1THEN1030
1100 PRINT"FERTIG"
1110 GOSUB7500
1120 REM ********************************
1130 REM -- BUBBLE SORT --
2000 GOSUB6180
2010 CLS:PRINT"START BUBBLE SORT";:INPUTX$
2020 M=N
2030 FORI=1TOM-1
2040 FORJ=I+1TOM
2050 IFB(I)<=B(J)THEN2070
2060 H=B(I):B(I)=B(J):B(J)=H
2070 NEXTJ,I
2080 PRINT"FERTIG"
2090 GOSUB7500
2100 REM ********************************
2110 REM -- SHELL SORT --
3000 GOSUB6180
3010 CLS:PRINT"START SHELL SORT";:INPUTX$
3020 M=N
3030 M=INT(M/2)
3040 IFM=0THEN3160
3050 J=1:K=N-M
3060 I=J
3070 L=I+M
3080 IFB(I)<=B(L)THEN3130
3090 H=B(I):B(I)=B(L):B(L)=H
3100 I=I-M
3110 IFI<1THEN3130
3120 GOTO3070
3130 J=J+1
3140 IFJ<=KTHEN3030
3150 GOTO3060
3160 PRINT"FERTIG"
3170 GOSUB7500
3180 REM ********************************
3190 REM -- INSERTION --
4000 GOSUB6180
4010 CLS:PRINT"START INSERTION SORT";:INPUTX$
4020 FORJ=1TON-1
4030 C=B(J+1)
4040 FORI=JTO1STEP-1
4050 IFC>=B(I)THEN4090
4060 B(I+1)=B(I)
4070 NEXTI
4080 I=0
4090 B(I+1)=C
4100 NEXTJ
4110 PRINT"FERTIG"
4120 GOSUB7500
4130 REM ********************************
4140 REM -- HEAP SORT --
5000 GOSUB6180
5010 CLS:PRINT"START HEAP SORT";:INPUTX$
5020 M=N
5030 FORL=INT(N/2)TO1STEP-1
5040 C=B(L)
5050 GOSUB5160
5060 NEXTL
5070 L=1
5080 FORM=N-1TO1STEP-1
5090 C=B(M+1)
5100 B(M+1)=B(1)
5110 GOSUB5160
5120 NEXTM
5130 PRINT"FERTIG"
5140 GOSUB7500
5150 GOTO5320
5160 REM ==SBR HEAP==
5200 I=L
5210 J=I+I
5220 IFJ>MTHEN5290
5230 IFJ=MTHEN5250
5240 IFB(J+1)>B(J)THENJ=J+1
5250 IFC>=B(J)THEN5290
5260 B(I)=B(J)
5270 I=J
5280 GOTO5210
5290 B(I)=C
5300 RETURN
5310 REM ********************************
5320 REM -- QUICK SORT --
6000 GOSUB6180
6010 CLS:PRINT"START QUICK SORT";:INPUTX$
6020 M=2*N:DIMST(M,1)
6030 S=1:ST(1,0)=1:ST(1,1)=N
6040 LI=ST(S,0):RE=ST(S,1):S=S-1
6050 I=LI:J=RE
6060 X=B(INT((LI+RE)/2))
6070 IFB(I)<XTHENI=I+1:GOTO6070
6080 IFB(J)>XTHENJ=J-1:GOTO6080
6090 IFI<=JTHENH=B(I):B(I)=B(J):B(J)=H:I=I+1:J=
J-1
```

```
6100 IFI<=JTHEN6070
6110 IFI<RETHENS=S+1:ST(S,0)=I:ST(S,1)=RE
6120 RE=J
6130 IFLI<RETHEN6050
6140 IFS>0THEN6040
6150 PRINT"FERTIG"
6160 GOSUB7500
6170 RUN:REM ******************************
6180 REM -- UMLADEN --
7000 FORI=1TON
7010 B(I)=A(I)
7020 NEXT:RETURN
7030 REM -- DISP SORT --
7500 FORI=1TON
7510 T=INT(40*((I-1)*.1-INT((I-1)*.1))+.5)
7520 PRINTTAB(T)B(I);:IFT=36THENPRINT
7530 NEXT:INPUTX$:RETURN
```

# Partieller Bildschirmscroll

## von Gerhard Klement

SCROLL erlaubt einen partiellen Bildschirm-Scroll (benoetigt aber NASSYS 3).

```
10 REM -- SCROLL  FILE U --
20 REM 5.3.81 22h55m /MODS SYS3 26.3.82 23h31
30 CLS
40 PRINTTAB(240)
50 PRINT" H U R R A ! EIN NEUES TEDDYSOFT JUWEL
60 PRINT
70 FOR I=1TO 2000: NEXT
80 CLS
90 PRINTTAB(12)" ******** DEMO ********
100 DEFFNX(X)=X*94/6.28
110 DEFFNY(Y)=-Y*44/4+20
120 SCREEN 1,15
130 REM
140 REM -- LOAD OBJECT PGM --
150 GOSUB620
160 REM -- DEMO UTILITY --
170 FOR X=0 TO 6.28 STEP.01
180 Y=FNY(SIN(X))
190 SET(FNX(X),INT(Y+.5))
200 PRINTX,SIN(X)
210 NEXT
220 CLS
230 PRINT "BERECHNUNG"
240 PRINT"AUSSTIEG MIT -1"
250 INPUT"X Wert (Radians)";X
260 IF X=-1 THEN GOSUB770:END
270 IF X<0 OR X>6.28 THEN240
280 Y=FNY(SIN(X))
290 SET(FNX(X),Y)
300 GOTO250
310 REM ****************************************
320 DATA 254,13    :REM D00  FE 0D    CP CR
330 DATA 194,144,1:REM D02  C2 90 01  JP NZ,CRT
340 DATA 245       :REM D05  F5       PUSH AF
350 DATA 42,41,12  :REM D06  2A 29 0C LD HL,(CU)
360 DATA 54,32     :REM D09  36 20    LD (HL),"
370 DATA 17,74,11  :REM D0B  11 CA 0A LD DE,TO
380 DATA 33,138,11:REM D0E  21 0A 0B LD HL,FROM
390 DATA 1,48,0    :REM D11  01 B0 00 LD BC,LEN
400 DATA 195,110,2:REM D14  C3 6E 02 JP CRT1
410 DATA 117       :REM D17  75       DEFB UOUT
420 DATA 0         :REM D18  00       DEFB ENDTA
430 REM ****************************************
440 REM
450 REM SCROLL EIGENSCHAFTEN IN HEXLINES :
460 REM         D0B  17,t,T
470 REM         D0E  33,f,F
480 REM         D11   1,l,L
490 REM
500 REM       Andere Scrolls:
510 REM ---------------------------------
520 REM ö LINES ö   t,T  ö   f,F  ö   l,L ö
530 REM ö-------ö--------ö--------ö-------ö
540 REM ö 14/15 ö  74,11 ö 138,11 ö  48,0 ö
550 REM ö 13/14 ö  10,11 ö  74,11 ö 112,0 ö
560 REM ö 12/13 ö 202,10 ö  10,11 ö 176,0 ö
570 REM ö 11/12 ö 138,10 ö 202,10 ö 240,0 ö
580 REM ö  7/8  ö 138,9  ö 202,9  ö 240,1 ö
590 REM ---------------------------------
600 REM
610 REM -- POKE ROUTINE --
620 RESTORE320
630 FOR I=3328 TO 3352
640 READ J
650 POKE I,J
660 NEXT
670 REM -- SAVE OLD ADRESS --
680 U9= DEEK(3192)
690 U8= DEEK(3187)
700 REM -- ACTIVATE NEW OUTPUT --
710 DOKE 3192,3328:REM $UOUT
720 DOKE 3187,3351:REM $OUT
730 RETURN
740 REM
750 REM ---------------------------------
760 REM -- SET OUTPUT TO NORMAL --
770 DOKE 3192,U9
780 DOKE 3187,U8
790 RETURN
```

Das "ö" in obigem Listing ist die Darstellung des Druckers für das ASCII- Zeichen ¦ (7C hex).

# Seite für Kinder

## von Hans-Dieter Schneider

Auch ich habe in ELCOMP diese beiden Artikel für die Kinder gelesen und sie für sehr gut gehalten. Deshalb habe ich sie bearbeitet und sie dabei meinem ABC-80 angepasst. Einen Abdruck davon finden Sie in der Anlage.

Die Entteuschung war jedoch sehr groß. Von etwa 15 Kindern (aus der Nachbarschaft) hat nicht eines auch nur einen einzigen Programmierversuch unternommen. Dabei kommen sie fast nur, um bei einigen Spielchen auf der Tastatur herumhacken zu können.
Neben den Spielen habe ich auch je ein einfaches Programm zum Üben der Grundrechenarten und des Alphabetes geschrieben (Abdrucke in der Anlage) Diese Programme werden allerdings oft genug freiwillig angewendet. Die Freiwilligkeit ist dadurch gesichert, daß die meisten Kinder selbstständig dazu in der Lage sind, Programme von der Cassette zu laden und zu starten. Auf die Auswahl der Cassette nehme ich keinen Einfluß.

Es ist also nicht so, daß die Kinder Angst vor dem Computer haben. Es ist wohl auch nicht die Angst vor dem Nachdenken, denn es werden auch durchaus 'schwierigere' Spiele ausgewählt. Auch unterbinde ich kein 'unsinniges' Herumtippen auf der Tastatur. Auch sonst bin ich nicht 'lehrerhaft' oder 'streng' oder 'kleinlich'. Ich bekomme einfach nicht heraus, warum die Kinder nicht programmieren wollen.
Vielleicht würde das (wenigstens vorrübergehend) anders, wenn man das Interesse durch einen Wettkampf beleben würde.



Kinder am Computer
---

Liebe Kinder!
Wollen wir mal den Computer ausprobieren? Es ist ganz bestimmt nicht so schwer wie Ihr glaubt.

Als erstes schreiben wir jetzt einmal unseren Namen in den Computer und sehen einmal nach, was der Computer dazu meint. Wenn Du Deinen Namen geschrieben hast, dann mußt Du die Taste ENTER drücken. Diese Taste ist rechts im Tastenfeld zu finden.

Wenn der Computer jetzt irgendetwas ausdruckt (zum Beispiel ?NERROR)so ist dies eine Fehlermeldung und bedeutet, daß der Computer jetzt nicht verstanden hat, was Du gerade eingetippt hast. Denn er kann selbst nicht denken und darum auch nicht wissen, was Du gedacht hast.

Jetzt wollen wir ihm einmal etwas eingeben, was er sicher versteht.

```
PRINT"MARIANNE"
```

Hier muß Dein Name hin

Nachdem Du diese Buchstaben eingegeben hast, drückst Du wieder die Taste ENTER. Was macht der Computer jetzt?
Jetzt versteht er Dich und schreibt Deinen Namen. Und dann schreibt er OK und der blinkende Punkt (Cursor genannt) zeigt an, daß er auf eine neue Aufgabe wartet. Versuche das Gleiche mit den Namen deiner Freunde und Bekannten.

Unser nächstes Spiel soll folgendes sein. Gib mal den folgenden Text ein: 10 PRINT "MARIANNE"
20 GOTO 10
RUN
Nach jeder Zeile die Taste RETURN drücken!!

Das war Dein erstes Programm. Es ist so klein und einfach. Aber siehe einmal, was es nicht schon alles macht. Es füllt die linke Hälfte des Bildschirmes mit Deinem Namen. Natürlich aber nur dann, wenn Du statt Marianne Deinen Namen eingesetzt hast.

Der Computer ist jetzt beschäftigt und es sieht so aus, als wolle er nie wieder damit aufhören. Aber das haben wir gleich.

Suche rechts die Taste SHIFT und drücke Sie gleichzeitig mit der ENTER- Taste. Nun meldet sich der Computer mit "Break", das heißt Unterbrechung, und der Nummer der Zeile, in der er das Programm beendet hat.

Gebe LIST ein und drücke wieder die Taste ENTER Jetzt kannst Du Dein Werk wieder einmal ansehen. Probiere jetzt auch andere Namen aus. Ändern kannst Du das Programm auf zwei Arten. Entweder schreibt man die ganze Zeile neu (schreibt also 10 PRINT "HANS" oder man tippt LIST10 ein und drückt RETURN. Der Computer schreibt nun die Zeile, die wir ändern wollen. Mit der Taste 'Pfeil nach rechts' gehen wir nach recht, bis in der neuen Zeile 10 PRINT" steht. Sind wir aus Versehen zu weit gegangen, so können wir mit der anderen Pfeiltaste auch wieder zurückgehen.

Nun also nur 10 PRINT" dort steht, kann ein neuer Name geschrieben werden. Jetzt wieder die Taste ENTER, dann RUN und wieder ENTER.
Weißt Du noch, wie man den Computer stoppt?
( SHIFT+ENTER )

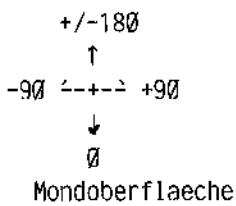
Nun [illegible] ENTER drücken, schreibe hinter das letzte Anführungszeichen noch das Zeichen ; und nun erst ENTER und dann Starten.
Was ist denn jetzt los? Er schreibt Deinen Namen immer hintereinander bis die Bildschirmzeile voll ist und setzt dann auf der nächsten Zeile weiter. Und das ohne Punkt immer weiter. Versuche das Gleiche auch mit anderen Namen. Achte aber darauf, daß der Strichpunkt ; stets am Ende der Zeile 10 steht.

wird fortgesetzt.

# Mondlandung

## von Wolfgang Schröder

Aus BASIC COMPUTER GAMES von D. Ahl
Sie befinden sich auf einer Mondexpedition und muessen die Kapsel sicher auf dem Mond aufsetzen. Als Kommandant der Mondfaehre muessen Sie bestimmte Kommandos an das Navigations-System geben. Der Bord-Computer wird Ihnen dabei laufend die wichtigsten Daten anzeigen, die Sie fuer die Navigation benoetigen.
Der Anflugwinkel ist folgendermassen zu interpretieren:
+/-180 Grad ist direkt vom Mond weg
+/-90 Grad ist parallel zur Mondoberflaeche
0 Grad ist direkt auf den Mond zu



Mondoberflaeche

Es werden alle Winkel zwischen +/-180 Grad angenommen.
Brennstoffeinheit = 1 sec max. Schub
Moegliche Werte hier 0 und zwischen 10 und 100 Prozent.
Negativer Schub oder Zeit ist verboten.
Um die Mission abzubrechen, geben Sie ueberall 0 ein.

```
1 CLS
2 PRINTTAB(10)"MONDLANDUNG"
10 Z$="GO"
15 B1=1
20 M=17.95
25 F1=5.25
30 N=7.5
35 R0=926
40 V0=1.29
45 T=0
50 H0=60
55 R=R0+H0
60 A=-3.425
65 R1=0
70 A1=8.84361E-04
75 R3=0
80 A3=0
85 M1=7.45
90 M0=M1
95 B=750
100 T1=0
105 F=0
110 P=0
115 N=1
120 M2=0
125 S=0
130 C=0
135 IFZ$="JA"THEN1150
140 PRINT
145 PRINT" MONDLANDE-SIMULATION!!"
150 PRINT
155 PRINT
156 PRINT
210 PRINT"WELCHES EINHEITENSYSTEM MOECHTEN SIE:
"
215 PRINT" 1. METRISCHES    0=ENGLISCHES"
220 PRINT"GEBE BITTE DIE GEWUENSCHTE ZAHL EIN";
225 INPUTK
230 PRINT
235 IFK=0THEN280
240 IFK=1THEN250
245 GOTO220
250 Z=1852.8
255 M$="METER"
260 G3=3.6
265 N$=" KILOMETER"
270 G5=1000
275 GOTO305
280 Z=6080
285 M$="FUSS"
290 G3=.592
295 N$=" NAUT. MEILEN"
300 G5=Z
305 IFB1=3THEN485
565 INPUT"KANN'S LOS GEHEN";Y$
570 GOTO670
575 PRINT
580 SCREEN1,8:PRINT"DAUER/SEC
581 SCREEN 10,8:INPUTT1
583 PRINT"% VOM SCHUB        ";
584 SCREEN13,9:INPUTF
585 PRINT"WINKEL/GRAD        ";
586 SCREEN13,10:INPUTP
590 F=F/100
595 IFT1<0THEN905
600 IFT1=0THEN1090
605 IFABS(F-.05)<1THEN945
610 IFABS(F-.05)<.05THEN945
615 IFABS(P)<180THEN925
616 SCREEN1,12:PRINTCHR$(27):PRINTCHR$(27)
620 N=20
625 IFT1<400THEN635
630 N=T1/20
635 T1=T1/N
```

```
640 P=P*3.14159/180
645 S=SIN(P)
650 C=COS(P)
655 M2=M0*T1*F/B
660 R3=-.5*R0*((V0/R)↑2)+R*A1*A1
665 A3=-2*R1*A1/R
670 FORI=1TON
675 IFM1=0THEN715
680 M1=M1-M2
685 IFM1<0THEN725
690 F=F*(1+M1/M2)
695 M2=M1+M2
700 PRINT"DU HAST KEINEN TREIBSTOFF MEHR."
705 M1=0
710 GOTO725
715 F=0
720 M2=0
725 M=M-.5*M2
730 R4=R3
735 R3=-.5*R0*((V0/R)↑2)+R*A1*A1
740 R2=(3*R3-R4)/2+.00526*F1*F*C/M
745 A4=A3
750 A3=-2*R1*A1/R
755 A2=(3*A3-A4)/2+.0056*F1*F*S/(M*R)
760 X=R1*T1+.5*R2*T1*T1
765 R=R+X
770 H0=H0+X
771 HX=H0*Z:SCREEN13,3:PRINTHX"    "
775 R1=R1+R2*T1
780 A=A+A1*T1+.5*A2*T1*T1
785 A1=A1+A2*T1
790 M=M-.5*M2
795 T=T+T1
796 SCREEN10,2:PRINTT
797 DX=R*A1*Z:SCREEN29,6:PRINTDX"    "
798 QX=R0*A*Z:SCREEN27,4:PRINTQX"    "
799 QX=R1*Z:SCREEN29,5:PRINTQX
800 TX=M1*B/M0:SCREEN19,7:PRINTTX
803 IFH0<-3.287828E-04THEN810
805 NEXTI
810 H=H0*Z
815 H1=R1*Z
820 D=R0*A*Z
825 D1=R*A1*Z
830 T2=M1*B/M0
835 GOSUB2000
845 IFH0<-3.287828E-04THEN880
850 IFR0*A>164.4736THEN1050
855 IFM1<0THEN580
860 T1=20
865 F=0
870 P=0
875 GOTO620
880 IFR1<-8.21957E-04THEN1020
885 IFABS(R*A1)>4.941742E-04THEN1020
890 IFH0<-3.287828E-04THEN1020
895 IFABS(D)>10*ZTHEN1065
900 GOTO995
905 PRINT:PRINTCHR$(27);
910 PRINT"DIESES RAUMSCHIFF IST NICHT IN DER LA
GE"
912 PRINT"DAS RAUM-ZEIT CONTINUUM ZU VERLASSEN.
"
920 GOTO575
925 PRINT:PRINTCHR$(27);
930 PRINT"WOHL KLEINER SCHERZ-KEKS"
940 GOTO 575
945 PRINT:PRINTCHR$(27);
950 PRINT"UMMOEGLICHE SCHUBKRAFT.";
955 IFF<0THEN985
960 IFF-.05<.05THEN975
965 PRINT"ZUVIEL!!"
970 GOTO575
975 PRINT"ZU WENIG!!"
980 GOTO575
985 PRINT"NEGATIV!!"
990 GOTO575
995 PRINT
1000 PRINT"HIER MEER DER RUHE"
1002 PRINT"DER ADLER IST GELANDET"
1005 PRINT"GLUECKWUNSCH--ES GAB KEINEN SCHADEN.
"
1010 PRINT"DU KANNST NUN MIT DEINER MONDOBER-"
1015 PRINT"FLAECHEN ERFORSCHUNG BEGINNEN."
1017 GOTO1100
1020 PRINT
1025 PRINT"CRASH !!!!!!!!!!!!!!!!!!!"
1030 PRINT"DU HAST EINEN NACH DIR BENANNTEN KRA
TER"
1031 PRINT"VON";ABS(H)*100;M$;" TIEFE ERZEUGT."
1035 X1=SQR(D1*D1+H1*H1)*G3
1040 PRINT"BEIM AUFSCHLAG BETRUG DEINE GESCHW."
1042 PRINTX1;N$;"/STD."
1045 GOTO1100
1050 PRINT
1055 PRINT"DU BIST IM WELTRAUM VERSCHOLLEN OHNE
"
1057 PRINT"HOFFNUNG AUF HILFE."
1060 GOTO1100
1065 PRINT"DU BIST SICHER GELANDET ...."
1070 PRINT
1080 PRINT"ABER VERFEHLTEST DEN LANDEPLATZ UM";
1082 PRINTABS(D/G5);N$
1085 GOTO1100
1090 PRINT
1095 PRINT"MISSION ABGEBROCHEN"
1100 PRINT
```

```
1105 PRINT"MOECHTEST DU ES NOCHEINMAL VERSUCHEN
?"
1107 PRINT"ANTWORTE MIT 'JA' ODER 'NEIN'";
1110 INPUTZ$
1115 IFZ$="JA"THEN20
1120 IFZ$="NEIN"THEN1130
1125 GOTO1105
1130 PRINT
1135 PRINT"ZU SCHADE. DIE NASA VERLIERT NICHT G
ERN"
1140 PRINT"IHREN EINZIGEN WELTRAUM-AFFEN!"
1145 END
1150 PRINT
1155 Q$="JA"
1160 GOTO190
2000 PRINTCHR$(12):PRINT"ZEIT/SEC:";T
2010 PRINT"HOEHE/";M$;":";H
2020 PRINT"ENTF. ZUM LANDPLATZ/";M$;":";D
2030 PRINT"VERTIKALE GESCHW/(";M$;"/SEC)";H1
2040 PRINT"HORIZONT. GESCHW/(";M$;"/SEC)";D1
2050 PRINT"BRENNSTOFF-VORRAT:";T2
2060 RETURN
```

Z80 DISASSEMBLER auf Cassette abzugeben, evtl. auch im Tausch gegen andere Programme.
K.Körner [redacted]

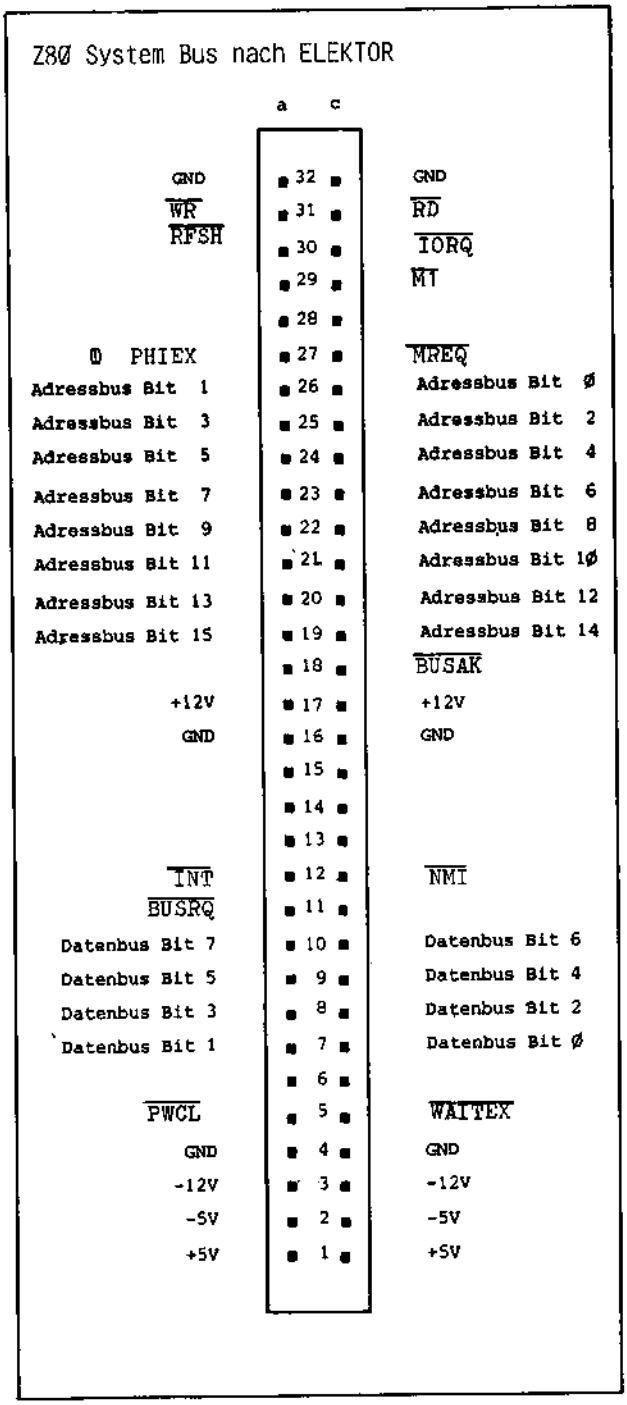
# Preisgünstiges RAM/ROM

## von Stefan Bürger

Im NASCOM Journal 4/5-81 habe ich über die Möglichkeit berichtet, die ELEKTOR 8k RAM/ROM Karte am NASCOM zu betreiben. Da die ELEKTOR Karte für ein 6502 System konzipiert wurde, ergeben sich dabei jedoch einige Schwierigkeiten. Der 6502 signalisiert die Speicherzugriffe nämlich etwas anders als der Z80. Da ich mehrfach wegen dieser Karten angesprochen worden bin, hier zwei Lösungsmöglichkeiten:

1. Möglichkeit eins ist zwar etwas unschön, aber sie funktioniert!
Taktfrequenz auf 1 MHZ reduzieren und WR an 31a legen. MREQ negieren und an 29c legen.

2. Die zweite Möglichkeit wurde in ELEKTOR 5/82 beschrieben. Mit relativ geringem Aufwand kann die Karte so an Z80- Systeme angepaßt werden. Dieser Methode sollte man auch den Vorzug geben, zumal in Heft 4/82 eine äußerst preisgünstige 16k dy. RAM Karte vorgestellt wurde, welche bereits Anpassungsmöglichkeiten für Z80 enthält. ( Print ca. 35.- DM ). Diese Karte läuft in meinem System seit einer Woche bei 2 MHz.



# Komprimiertes BASIC

## von E. Moser u. G.T.Klement

BASIC TOTAL COMPRESS komprimiert Basic Programme zu Superzeilen. (Memory Gewinn, aber nicht mehr editierbar). Ein Doppelpunkt am Anfang einer Zeile verhindert die Komprimierung dieser Zeile (notwendig, wenn sie editierbar bleiben soll).

```
10 REM DEMONSTRATION VON BASIC COMPRESS
20 INPUT "EINGABE "; A$
30 INPUT "EINGABE "; B$
40 C$= A$ + B$
50 PRINT C$
60 GOTO 20
Ok
```

```
10 REM DEMONSTRATION VON BASIC COMPRESS
20 INPUT"EINGABE";A$:INPUT"EINGABE";B$:C$=A$+B$
:PRINTC$:GOTO20
Ok
```

ZEAP Z80 Assembler - Source Listing

```
                0010 ; ****************
                0020 ; * BLANKS REMOVE  *
                0030 ; * BASIC COMPRESS *
                0040 ; * 12.7.81 00h18  *
                0050 ; *   CCO  -  E41  *
                0060 ; ****************
                0070 ; MOSER/KLEMENT
OCCO            0080        ORG   #CCO
OCCO 0020       0090 BLANK  EQU   #20
OCCO 0022       0100 QUOTE  EQU   #22
OCCO 0088       0110 GOTO   EQU   #88
OCCO 0089       0120 RUN    EQU   #89
OCCO 008A       0130 IF     EQU   #8A
OCCO 008B       0140 RESTOR EQU   #8B
OCCO 008C       0150 GOSUB  EQU   #8C
OCCO 008E       0160 REM    EQU   #8E
OCCO 00A9       0170 THEN   EQU   #A9
OCCO 0C1E       0180 ACLN   EQU   #C1E
OCCO 0C1C       0190 NEWTOP EQU   #C1C
OCCO 105E       0200 BEGPGM EQU   #105E
OCCO 10D6       0210 BEGSIN EQU   #10D6
OCCO 10D8       0220 BEGAR  EQU   #10D8
OCCO 10DA       0230 BEGFRE EQU   #10DA
OCCO E68A       0240 CPHLDE EQU   #E68A;BASIC
OCCO E838       0250 CHKSYM EQU   #E838;BASIC
OCCO E9A5       0260 ATOH   EQU   #E9A5;BASIC
                0270 ;
OCCO 2A5E10     0280 START  LD    HL,(BEGPGM)
OCC3 E5         0290 NXLN   PUSH  HL
OCC4 DDE1       0300        POP   IX
OCC6 010400     0310        LD    BC,#4
OCC9 09         0320        ADD   HL,BC
OCCA 7E         0330        LD    A,(HL)
OCCB FE8E       0340 NXCHR  CP    REM
OCCD 2826       0350        JR    Z,REMAR
OCCF FE22       0360        CP    QUOTE
OCD1 2829       0370        JR    Z,STRING
OCD3 FE20       0380        CP    BLANK
OCD5 2830       0390        JR    Z,DELBLK
OCD7 23         0400 STREX  INC   HL
OCD8 7E         0410 SHFDON LD    A,(HL)
OCD9 B7         0420        OR    A
OCDA 20EF       0430        JR    NZ,NXCHR
OCDC 23         0440 REMEX  INC   HL
OCDD DD7500     0450        LD    (IX),L
OCE0 DD7401     0460        LD    (IX+1),H
```

CP HL:DE next symbol ASCII TO HEX

```
OCE3 46         0470        LD    B,(HL)
OCE4 23         0480        INC   HL
OCE5 7E         0490        LD    A,(HL)
OCE6 2B         0500        DEC   HL
OCE7 B0         0510        OR    B
OCE8 20D9       0520        JR    NZ,NXLN
OCEA 2AD610     0530        LD    HL,(BEGSIN)
OCED 22D810     0540        LD    (BEGAR),HL
OCF0 22DA10     0550        LD    (BEGFRE),HL
OCF3 182A       0560        JR    LINCPR
                0570 ;
OCF5 23         0580 REMAR  INC   HL
OCF6 7E         0590        LD    A,(HL)
OCF7 B7         0600        OR    A
OCF8 20FB       0610        JR    NZ,REMAR
OCFA 18E0       0620        JR    REMEX
                0630 ;
OCFC 23         0640 STRING INC   HL
OCFD 7E         0650        LD    A,(HL)
OCFE FE22       0660        CP    QUOTE
OD00 28D5       0670        JR    Z,STREX
OD02 B7         0680        OR    A
OD03 20F7       0690        JR    NZ,STRING
OD05 18D5       0700        JR    REMEX
                0710 ;
OD07 E5         0720 DELBLK PUSH  HL
OD08 E5         0730        PUSH  HL
OD09 D1         0740        POP   DE
OD0A 23         0750        INC   HL
OD0B E5         0760        PUSH  HL
OD0C E5         0770        PUSH  HL
OD0D C1         0780        POP   BC
OD0E 2AD610     0790        LD    HL,(BEGSIN)
OD11 ED42       0800        SBC   HL,BC
OD13 E5         0810        PUSH  HL
OD14 C1         0820        POP   BC
OD15 E1         0830        POP   HL
OD16 EDB0       0840        LDIR
OD18 21D610     0850        LD    HL,BEGSIN
OD1B 35         0860        DEC   (HL)
OD1C E1         0870        POP   HL
OD1D 18B9       0880        JR    SHFDON
                0890 ;
OD1F 2AD610     0900 LINCPR LD    HL,(BEGSIN);END PGM TO
OD22 221C0C     0910        LD    (NEWTOP),HL; BUFF
OD25 2A5E10     0920        LD    HL,(BEGPGM);SAVE START PG.
OD28 E5         0930        PUSH  HL        ;
OD29 2B         0940 NXTTRM DEC   HL        ;SET :
OD2A 363A       0950        LD    (HL),":   ;
OD2C 23         0960        INC   HL        ;NEXT LINE S
TART
OD2D CD3A0E     0970        CALL  TSTEND    ; TO DE
OD30 EB         0980        EX    DE,HL     ; TO HL
OD31 20F6       0990        JR    NZ,NXTTRM ; TO HL
OD33 E1         1000        POP   HL        ;
OD34 221E0C     1010 PRGLN  LD    (ACLN),HL ;SAVE START
OD37 CD3A0E     1020        CALL  TSTEND    ;POI NEXTLN
DE
OD3A CAC90D     1030        JP    Z,DONE    ;END IF
OD3D 23         1040        INC   HL        ;POINT TO
OD3E 23         1050 FOLCHR INC   HL        ; TOKEN
OD3F 7E         1060        LD    A,(HL)    ;GET IT
OD40 FE3A       1070        CP    ":        ;TERMINATOR
?
OD42 2878       1080        JR    Z,REMARK  ; THEN FORGE
T IT
OD44 4E         1090 LODCHR LD    C,(HL)    ;TOKEN TO C
OD45 23         1100        INC   HL        ;POI NEXT
OD46 CD8AE6     1110        CALL  CPHLDE    ;EOL ?
OD49 28E9       1120        JR    Z,PRGLN   ;FETCH NEXT
LINE
OD4B 0622       1130        LD    B,#22     ;B="
OD4D 79         1140        LD    A,C       ;TO ACCU
OD4E B8         1150        CP    B         ;IS IT PRINT
START?
OD4F 2013       1160        JR    NZ,NOT22  ;NO, GO AHEA
D
OD51 7E         1170 PRINT  LD    A,(HL)    ;GET CHR
OD52 B8         1180        CP    B         ;PRINT END
OD53 23         1190        INC   HL        ;NEXT CHR
```

```
0D54 28EE     1200         JR   Z,LODCHR  ;IT WAS PRIN
T END
0D56 CD8AE6   1210         CALL CPHLDE    ;EOL ?
0D59 20F6     1220         JR   NZ,PRINT  ;
0D5B 2B       1230         DEC  HL        ;NEXT TOKEN
0D5C 7E       1240         LD   A,(HL)    ;
0D5D B7       1250         OR   A         ;EOL ?
0D5E 2801     1260         JR   Z,NOCHNG  ;POI TERMINA
T.
0D60 70       1270         LD   (HL),B    ; SET "
0D61 EB       1280 NOCHNG  EX   DE,HL     ;POI NEXTL H
L
0D62 18D0     1290         JR   PRGLN     ; AND DO AGA
IN
              1300 ;
0D64 D5       1310 NOT22   PUSH DE        ;
0D65 11B30D   1320         LD   DE,CHK#   ;POI SBR FOR
 DECCHK
0D68 D5       1330         PUSH DE        ;ONTO STACK
0D69 FE88     1340         CP   GOTO      ;
0D6B C8       1350         RET  Z         ;
0D6C FE8C     1360         CP   GOSUB     ;
0D6E C8       1370         RET  Z         ;
0D6F FEA9     1380         CP   THEN      ;
0D71 C8       1390         RET  Z         ;
0D72 FE89     1400         CP   RUN       ;
0D74 C8       1410         RET  Z         ;
0D75 FE8B     1420         CP   RESTOR    ;
0D77 C8       1430         RET  Z         ;
0D78 D1       1440         POP  DE        ;
0D79 FE8E     1450         CP   REM       ;
0D7B D1       1460         POP  DE        ;
0D7C 283E     1470         JR   Z,REMARK  ;IT WAS REM
0D7E FE8A     1480         CP   IF        ;
0D80 20C2     1490         JR   NZ,LODCHR ;NO IF,NEXT
CHR
0D82 1B       1500         DEC  DE        ;POI BACK
0D83 AF       1510         XOR  A         ;
0D84 12       1520         LD   (DE),A    ;PUT TERMINA
TOR
0D85 13       1530         INC  DE        ;POI NEXTL
0D86 18BC     1540         JR   LODCHR    ;
              1550 ;
0D88 CDA5E9   1560 LINCOM  CALL ATOH      ;A  ;ASCII T
O HEX IN
0D8B E5       1570         PUSH HL        ;POI CHR AFT
ER #
0D8C 4B       1580         LD   C,E       ;HEX # TO BC

0D8D 42       1590         LD   B,D       ;
0D8E 2A5E10   1600         LD   HL,(BEGPGM);START PGM
0D91 E5       1610 NXTNUM  PUSH HL        ;SAV POI
0D92 CD3A0E   1620         CALL TSTEND    ;NEXTLN TO D
E
0D95 2003     1630         JR   NZ,FIND   ;EOL END SEA
RCH
0D97 E1       1640         POP  HL        ;SAV START N
EXL
0D98 1811     1650         JR   NOTFND    ;
              1660 ;
0D9A D5       1670 FIND    PUSH DE        ;GET LINUM
0D9B 59       1680         LD   E,C       ; TO DE
0D9C 50       1690         LD   D,B       ;
0D9D 7E       1700         LD   A,(HL)    ;PGM LINUM
```

```
0D9E 23       1710         INC  HL        ; TO HL
0D9F 66       1720         LD   H,(HL)    ;
0DA0 6F       1730         LD   L,A       ;
0DA1 CD8AE6   1740         CALL CPHLDE    ;EQUAL ?
0DA4 E1       1750         POP  HL        ;START NEXL
0DA5 D1       1760         POP  DE        ;START OLDL
0DA6 20E9     1770         JR   NZ,NXTNUM ;UNEQUAL SEARCH
0DA8 1B       1780         DEC  DE        ;-1 START OLD
0DA9 AF       1790         XOR  A         ;
0DAA 12       1800         LD   (DE),A    ;RESET TERMIN.
0DAB E1       1810 NOTFND  POP  HL        ;POI AFTER LINNUM
0DAC 7E       1820         LD   A,(HL)    ;
0DAD FE2C     1830         CP   ",        ;IF NO , THEN
0DAF C0       1840         RET  NZ        ; END ROUTINE
0DB0 23       1850         INC  HL        ;NEXT CHR
0DB1 18D5     1860         JR   LINCOM    ;NEXT LIN
              1870 ;
0DB3 CD38E8   1880 CHK#    CALL CHKSYM    ;IS CHR ASCII
0DB6 D4880D   1890         CALL NC,LINCOM  ;IF YES SEARCH/REPL
0DB9 D1       1900         POP  DE        ;START NEXL
0DBA 1888     1910         JR   LODCHR    ;
              1920 ;
0DBC 2A1E0C   1930 REMARK  LD   HL,(ACLN) ;START CURL
0DBF AF       1940         XOR  A         ;
0DC0 2B       1950         DEC  HL        ;-1
0DC1 77       1960         LD   (HL),A    ;SET ZERO
0DC2 EB       1970         EX   DE,HL     ;NEXT LINE START
0DC3 2B       1980         DEC  HL        ;POI TERMIN.
0DC4 77       1990         LD   (HL),A    ;SET ZERO
0DC5 23       2000         INC  HL        ;NEXTL START
0DC6 C3340D   2010         JP   PRGLN     ;DO NEX
              2020 ;
0DC9 2A5E10   2030 DONE    LD   HL,(BEGPGM);PGMSTART+1
0DCC 2B       2040         DEC  HL        ;POI 1st BYTE
0DCD 73       2050         LD   (HL),E    ;SET ZERO
0DCE 23       2060         INC  HL        ;POI NEXL START
0DCF E5       2070 LIN+    PUSH HL        ;SAVE POI
0DD0 2A1C0C   2080         LD   HL,(NEWTOP);EOP
0DD3 EB       2090         EX   DE,HL     ; TO DE
0DD4 2AD610   2100         LD   HL,(BEGSIN);OLD EOP
0DD7 ED52     2110         SBC  HL,DE     ;DIFFERENCE
0DD9 EB       2120         EX   DE,HL     ; TO DE
0DDA E1       2130         POP  HL        ;POI
0DDB ED52     2140         SBC  HL,DE     ;TO ACTUAL VAL
0DDD E5       2150         PUSH HL        ;SAV
0DDE CD3A0E   2160         CALL TSTEND    ;POI NEXL TO DE
0DE1 2826     2170         JR   Z,TOPFND  ;IF 0 END REACHED
0DE3 EB       2180         EX   DE,HL     ;POI LINUM = DE
0DE4 C1       2190         POP  BC        ;START NEXL
0DE5 0B       2200         DEC  BC        ;TO TERMIN.
0DE6 0A       2210         LD   A,(BC)    ;GET TERM
0DE7 B7       2220         OR   A         ;
0DE8 28E5     2230         JR   Z,LIN+    ;IF ZERO NEXL
0DEA E5       2240         PUSH HL        ;SAVE ASSUMED NEXLST
0DEB 03       2250         INC  BC        ;POI LINST
0DEC FE22     2260         CP   #22       ;IF " EQ PRINT
0DEE 2004     2270         JR   NZ,NOPRNT ;NO CHANGES
0DF0 3E3A     2280         LD   A,":      ;TERMIN
0DF2 02       2290         LD   (BC),A    ; AFTER "
0DF3 03       2300         INC  BC        ;NEXT TOKEN
0DF4 C5       2310 NOPRNT  PUSH BC        ;SAVE POI TOKEN
0DF5 13       2320         INC  DE        ;POI START OF TOKENS
0DF6 13       2330         INC  DE        ;
0DF7 2A1C0C   2340         LD   HL,(NEWTOP);END ADR OF PGM
```

```
0DFA ED52    2350        SBC  HL,DE     ;CALC # BYTES TO TOP
0DFC E5      2360        PUSH HL        ; TO BC
0DFD C1      2370        POP  BC        ;
0DFE E1      2380        POP  HL        ;POI 1st FREE PLACE
0DFF EB      2390        EX   DE,HL     ;MOVEON SHIFT
0E00 EDB0    2400        LDIR ; FROM (DE) TO
0E02 EB      2410        EX   DE,HL     ; (HL)
0E03 221C0C  2420        LD   (NEWTOP),HL;
0E06 E1      2430        POP  HL        ;ASSUMED NEXL START
0E07 18C6    2440        JR   LIN+      ;NEXT LIN
             2450 ;
0E09 22D610  2460 TOPFND LD   (BEGSIN),HL;HL =N
0E0C E1      2470        POP  HL        ;POI 00
0E0D 2B      2480        DEC  HL        ;POI LAST TERM
0E0E 73      2490        LD   (HL),E    ;SET ZERO
0E0F 2A5E10  2500        LD   HL,(BEGPGM)
0E12 E5      2510 NXTAD  PUSH HL        ;CALCULATE ADDR POI
0E13 CD3A0E  2520        CALL TSTEND    ; AND INSERT AT
0E16 D1      2530        POP  DE        ; START OF LINES
0E17 280E    2540        JR   Z,EXIT    ;
0E19 23      2550        INC  HL        ;
0E1A 23      2560        INC  HL        ;
0E1B 7E      2570 FNDT   LD   A,(HL)    ;
0E1C 23      2580        INC  HL        ;
0E1D B7      2590        OR   A         ;
0E1E 20FB    2600        JR   NZ,FNDT   ;
0E20 7D      2610        LD   A,L       ;
0E21 12      2620        LD   (DE),A    ;
0E22 13      2630        INC  DE        ;
0E23 7C      2640        LD   A,H       ;
0E24 12      2650        LD   (DE),A
0E25 18EB    2660        JR   NXTAD
             2670 ;
```

```
2F0F 2A5E10  2680 EXIT   LD   HL,(BEGPGM)
             2690        EX   DE,HL     ;
0E2A EB      2700 FNDEND PUSH DE        ;
0E2B D5      2710        POP  HL        ;
0E2C E1      2720        CALL TSTEND    ;
0E2D CD3A0E  2730        JR   NZ,FNDEND ;
0E30 20F9    2740        LD   (#10D8),HL;BEGARR
0E32 22D810  2750        LD   (#10DA),HL;BEGFRE
0E35 22DA10  2760        SCAL "Z
0E38 DF5A    2770 ;
             2780 TSTEND LD   E,(HL)    ;
0E3A 5E      2790        INC  HL
0E3B 23      2800        LD   D,(HL)
0E3C 56      2810        INC  HL
0E3D 23      2820        LD   A,E
0E3E 7B      2830        OR   D
0E3F B2      2840        RET  ;
0E40 C9
```

SUCHE Pascal-Compiler in EPROM
GEBE AB Basic-Rom MK 36271P
TAUSCHE Software aller Art für NASCOM 2

G. Steuerwald Tel. [redacted]
[redacted] ; [redacted]

# MITARBEITER DIESER AUSGABE

Günter Böhm
[redacted]
[redacted] Karlsruhe
[redacted]

Günter Kreidl
[redacted]
[redacted] Straelen
[redacted]

Wolfgang Mayer-Gürr
[redacted]
[redacted]
[redacted] Recklinghausen
[redacted]

Josef Zeller
[redacted]
[redacted] Neu-Ulm
[redacted] (Vermieter)

Michael Bach
[redacted]
[redacted] Stegen
[redacted]

Klaus Mombaur
[redacted]
[redacted] Nürnberg [redacted]
[redacted]

Clemens Ballarin
[redacted]
[redacted] Überlingen
[redacted]

Christian Peter
[redacted]
A [redacted] Wien
Österreich
[redacted]

Wolfgang von Jan
[redacted]
[redacted] Langenhagen
[redacted]

Wolfgang Schröder
[redacted]
[redacted] Reutlingen
Tel.?

Erich Mehnert
[redacted]
[redacted] Mandelbachtal
[redacted]

Peter Brendel
[redacted]
[redacted] Mannheim
Tel.?

Stefan Bürger
[redacted]
[redacted] Tamm-Hohenstange
[redacted]

Markus Caesar
[redacted]
[redacted] Leichlingen
Tel.?

Christoph Rau
[redacted]
[redacted] Bonn
Tel.?

Hans Dieter Schneider
[redacted]
[redacted] Esens
Tel.?

Tom D. Rüdebusch
[redacted]
[redacted] Giessen
[redacted]

Constantin Olbrich
[redacted]
[redacted] Berlin [redacted]

Günter Brust
[redacted]
[redacted] Baden-Baden
Tel. [redacted]

Gerhard Klement
[redacted]
[redacted] Wien
Österreich
Tel. [redacted]

und letztmals Gabi
souverän am
Typenrad

# Millionärspiel

## von Clemens Ballarin

Spielidee:
Der Spieler muss versuchen, durch den An- und Verkauf von Aktien sich ein moeglichst grosses Vermoegen zu erwerben. Als Startkapital stehen ihm 10000 DM zur Verfuegung. Die schwankenden Aktienkurse und auch andere Ereignisse sorgen, wie im normalen Leben dafuer, dass die Baeume nicht zu schnell in den Himmel wachsen. Die Bewegung der Aktienkurse wird auf der linken Bildschirmhaelfte graphisch dargestellt. Das Spiel belegt ausser dem Basicbereich auch noch Speicherplatz von HEX D00 bis D53 und laeuft nur auf NASSYS.

```
1 REM MILLIONAER-SPIEL
2 REM VON CLEMENS BALLARIN
3 REM GESCHRIEBEN IM JUNI 1982
4 REM
5 REM VERKAUF VON AKTIEN DURCH
6 REM EINKAUF MIT NEGATIVEM VORZEICHEN
10 CLS:DOKE4175,-6649:DOKE4100,3328
20 DEFFNR(H)=5*INT(RND(1)*(H*19000-94000))
25 DEFFNS(G)=USR(2058+G)
30 SCREEN15,2:PRINT"W I L L K O M M E N"
40 SCREEN22,3:PRINT"Z U M"
50 SCREEN9,4:PRINT"M I L L I O N A E R - S P I
E L"
60 PRINT:PRINT:PRINT:PRINT"Wieviele Tage moecht
en Sie spielen";
70 RESTORE:INPUTT
80 FORI=3328TO3410STEP2:READA:DOKEI,A:NEXT
90 FORI=1TOT:E=RND(1):NEXT
100 CLS:D=0:GB=10000
110 FORI=0TO3::A(I)=100:READA$(I):B(I)=0:NEXT
120 FORI=0TO8:S(I)=0:NEXT
130 A$="200":Z=FNS(0):A$="150":Z=FNS(256)
135 A$="100":Z=FNS(448):A$="50":Z=FNS(641)
140 A$="0":Z=FNS(898)
150 FORI=2TO42STEP10:SET(7,I):NEXT
160 A$="Guthaben in":Z=FNS(156)
170 A$="bar    :          DM":Z=FNS(220)
180 A$="Aktien:           DM":Z=FNS(284)
190 A$="gesamt:           DM";Z=FNS(348)
200 A$=", Tag von       Tagen":Z=FNS(90)
210 D=D+1:A=D:L=4:G=86:GOSUB860
220 G=99:A=T:GOSUB860:GOSUB620:GOSUB560
230 GOSUB400:GOSUB490:GOSUB560
240 IFD=TTHEN210
250 CLS
260 IFGB+GA>=10000THEN300
270 PRINT"So wird man bestimmt nicht Millionaer
!"
275 PRINT"Dagobert Duck wuerde sich schief lach
en!"
280 PRINT"Sie haben in";D;"Tagen";10000-GB-GA;
290 PRINT"DM Verlust gemacht!":GOTO360
300 IFGB+GA-10000>T*100THEN340
310 PRINT"Was? Nur";GB+GA-10000;" DM Gewinn?"
320 PRINT"Sie tun mir leid!"
330 GOTO360
340 PRINT"Das war sehr gut!"
350 PRINT"Sie haben";GB+GA-10000;" DM kassiert"
360 INPUT"Noch ein Spiel";A$
370 IFA$="J"THENCLS:GOTO60
380 PRINT"Auf Wiedersehen bis zum naechsten mal
."
390 END
400 FORI=0TO3
410 X=I:GOSUB1050
420 B(I)=B(I)+K:GB=GB-A(I)*K
430 FORJ=1TOK:E=RND(1):NEXTJ
440 GOSUB560
450 L=4:G=(16-3*C(I))*64+21:A=B(I):GOSUB860
460 PRINT")":SCREEN22,17-3*C(I):PRINT"("
470 NEXT
480 RETURN
490 IFD=30ORRND(1)*20>DTHENRETURN
500 X=INT(RND(1)*9)
510 IFS(X)=1THENRETURN
520 S(X)=1:X=X+4
530 GOSUB1050
540 GB=GB+A
550 RETURN
560 GA=0
570 FORJ=0TO3:GA=GA+A(J)*B(J):NEXTJ
580 L=9:A=GA:G=291:GOSUB860
590 A=GB:G=227:GOSUB860
600 A=GB+GA:G=355:GOSUB860
610 RETURN
620 G=D:IFG>27THENG=27
625 FORI=37-GTO37:FORJ=2TO42:A=POINT(I,J)
630 IFA=0THENRESET(I-1,J):GOTO650
640 SET(I-1,J):RESET(I,J)
650 NEXTJ,I
660 FORI=0TO3
670 A=5*INT(RND(1)*7-3)
680 IFA(I)+A>50RA(I)+A>200THEN710
700 A(I)=A(I)+A
710 SET(37,42-A(I)/5)
720 NEXT
730 FORI=0TO3:C(I)=1:NEXT
740 FORI=0TO3
```

```
750 FORJ=1TO3
760 IF1=JTHEN790
770 IFA(I)<>A(J)THENC(J)=C(J)+1:GOTO790
780 C(I)=C(I)+1
790 NEXTJ,I:L=4:FORI=0TO3
800 G=(16-3*C(I))*64+21:A=B(I):GOSUB860
810 PRINT")":SCREEN22,17-3*C(I):PRINT"("
820 A$=A$(I):Z=FNS((15-3*C(I))*64+20)
830 G=(17-3*C(I))*64+20:A=A(I):GOSUB860:PRINT"D
M";
840 NEXT
850 RETURN
860 A$=STR$(A)
870 F=LEN(A$)
880 IFF<LTHEN960
890 C=L-F
900 IFC=0THEN940
910 FORB=1TOC
920 A$=" "+A$
930 NEXTB
940 Z=FNS(G)
950 RETURN
960 FORB=1TOF
970 IFMID$(A$,B,1)="E"THENE=B
980 NEXTB
990 FORB=ETOE-F+L+1STEP-1
1000 B$=MID$(A$,B-2)
1010 C$=MID$(A$,B)
1020 A$=B$+C$
1030 NEXTB
1040 GOTO940
1050 A$="                    "
1060 FORJ=476TO860STEP64:Z=FNS(J):NEXTJ
1070 ONX+1GOTO1310,1310,1310,1310,1320,1330,134
0
1080 ONX-6GOTO1350,1360,1370,1330,1380,1390
1090 Z=FNS(476)
1100 ONX+1GOTO1400,1400,1400,1400,1410,1420,143
0
1110 ONX-6GOTO1440,1450,1460,1470,1480,1490
1120 Z=FNS(540)
1130 ONX+1GOTO1500,1500,1500,1500,1160,1510,152
0
1140 ONX-6GOTO1530,1540,1550,1570,1580,1600
1150 Z=FNS(604)
1160 ONX+1GOTO1610,1610,1610,1610,1630,1640,166
0
1170 ONX-6GOTO1680,1690,1190,1700,1710,1720
1180 Z=FNS(668)
1190 ONX+1GOTO1730,1730,1730,1730,1740,1760,177
0
1200 ONX-6GOTO1780,1790,1220,1820,1830,1840
1205 GOTO1220
1210 Z=FNS(732)
1220 ONX+1GOTO1850,1850,1850,1850,1250,1860,187
0
1230 ONX-6GOTO1880,1890,1250,1900,1920,1930
1235 GOTO1250
1240 Z=FNS(796)
1250 ONX+1GOTO1940,1940,1940,1940,1280,1960,199
0
1260 ONX-6GOTO2000,2020,1280,1280,2030,2060
1265 GOTO1280
1270 Z=FNS(860)
1280 IFY=1THENY=0:GOTO1160
1290 IFX=17THENX=11
1300 RETURN
1310 A$="Wieviele "+A$(X)+"-":GOTO1090
1320 A$="Sie haben":GOTO1090
1330 A$="Sie haben einen":GOTO1090
1340 A$="Moechten Sie eine":GOTO1090
1350 A$="Ihr Haus ist abge-":GOTO1090
1360 A$="In Ihrem Haus ist":GOTO1090
1370 A$="Sie haben bei einem":GOTO1090
1380 A$="Ihr Grossvater ist":GOTO1090
1390 A$="Sie haben eine":GOTO1090
1400 A$="Aktien moechten":GOTO1120
1410 A$="Computeritis.":GOTO1120
1420 A$="Verkehrsunfall":GOTO1120
1430 A$="Autoversicherung":GOTO1120
1440 A$="brannt. Die Ursache":GOTO1120
1450 A$="ein Wasserrohr ge-":GOTO1120
1460 A$="Preisausschreiben":GOTO1120
1470 A$="Bankueberfall ver-":GOTO1120
1480 A$="gestorben.":GOTO1120
1490 A$="sensationelle":GOTO1120
1500 A$="Sie kaufen ":GOTO1150
1510 A$="'gebaut'":GOTO1150
1520 A$="abschliessen ":GOTO1150
1530 A$="ist leider noch":GOTO1150
1540 A$="brochen. Dabei ent-":GOTO1150
1550 L=5:A=FNR(L):G=603:GOSUB860
1560 A$="DM gewonnen.":Z=FNS(609):GOTO1160
1570 A$="hindert.":GOTO1150
1580 A$="Sie erben":Z=FNS(604):L=6:A=FNR(L)
1590 G=613:GOSUB860:A$=" DM.":Z=FNS(619):GOTO11
60
1600 A$="Erfindung gemacht.":GOTO1150
1610 INPUTK:IFK+B(X)<1000THENX=X+13
1620 GOTO1190
1630 A$="Die Arzt-Kosten":GOTO1180
1640 A$="Da Sie ":Z=FNS(668)
1645 IFS(2)=0THENA$="nicht":Z=FNS(675):Y=6
1650 A$=" ver-":Z=FNS(674+Y):Y=0:GOTO1190
1660 INPUTA$:IFA$<>"J"THENX=17:S(2)=0
1670 GOTO1190
1680 A$="nicht geklaert.":GOTO1180
1690 A$="stand ein Schaden":GOTO1180
```

```
1700 A$="Zur Belohnung be-":GOTO1180
1710 A$="Da Sie aber die":GOTO1180
1720 A$="Durch geschickten":GOTO1180
1730 A$="Seien Sie beschei-":GOTO1210
1740 A$="betragen":Z=FNS(732):L=5:A=FNR(L)
1745 G=740:GOSUB860:A$="DM,":Z=FNS(746):A=-A
1750 GOTO1220
1760 A$="sichert sind,":GOTO1210
1770 A$="Wie hoch moechten":GOTO1210
1780 A$="Fuer eine neue Woh-":GOTO1210
1790 A$="von":Z=FNS(732):L=5:A=FNR(L):G=735
1800 GOSUB860:A$="DM, fuer":Z=FNS(741):A=-A
1810 GOTO1220
1820 A$="kommen Sie von der":GOTO1210
1830 A$="Beerdigung bezahlen":GOTO1210
1840 A$="Verkauf des Paten-":GOTO1210
1850 A$="den. Es gibt nur":GOTO1240
1860 A$="kostet Sie das":GOTO1240
1870 A$="Sie sich versichern":GOTO1240
1880 A$="nung mussen Sie":GOTO1240
1890 A$="den Sie aufkommen":GOTO1240
1900 A$="Bank":Z=FNS(796):L=5:A=FNR(L):G=800
1905 GOSUB860:A$="DM.":Z=FNS(806)
1910 GOSUB860:A$="DM.":Z=FNS(806):GOTO1250
1920 A$="muessen, bekommen":GOTO1240
1930 A$="tes erhalten Sie":GOTO1240
1940 A$="noch":Z=FNS(860):A=999-B(X):L=4:G=864
1945 GOSUB860:A$=".":Z=FNS(870):Y=1
1950 GOSUB860:A$=".":Z=FNS(870):Y=1:GOTO1280
1960 L=6:A=FNR(L):G=859
1970 IFS(2)=1THENA=A-V*10:IFA<>0THENA=0
1980 GOSUB860:A$="DM.":Z=FNS(866):A=-A:GOTO1280
1990 A$="lassen":Z=FNS(860):INPUTV:A=-V:GOTO1280
2000 L=6:A=FNR(L):G=859:GOSUB860
2010 A$="DM bezahlen.":Z=FNS(866):A=-A:GOTO1280
2020 A$="muessen.":GOTO1270
2030 A$="Sie nur":Z=FNS(860):L=6:G=867
2040 B=A-2*FNR(5):IFA<>0THENB=A-100
2050 A=B:GOSUB860:A$="DM.":Z=FNS(874):GOTO1280
2060 G=859:L=6:A=FNR(L):GOSUB860:A$="DM."
2070 Z=FNS(866):GOTO1280
2080 DATA-29747,-4631,10579,-4852,-10661,272,6
2090 DATA-10198,-18672,21229,8744,16107,-13568
2100 DATA-16641,1064,-5367,-5096,15907,-16831
2110 DATA808,6187,9203,9030,24099,22051,-2278
2120 DATA4115,-13829,22255,21057,16713,19522
2130 DATA8261,18766,18499,8276,17735,21830
2140 DATA17486,20037,13,201
2150 DATASTAHL  ,ELEKTRO,KERAMIK,DIAMANT
```

# Musik-Synthese

## von Constantin Olbrich

```
0 REM fuer RESET "E000" faehiges BASIC
1 REM E000: C3 03 E0 = LD PC,#E003
2 REM andere Version kann nicht mit RESET starten
3 REM *******************************
4 REM *** MUSIKSYNTHESE UEBER PORT0 ***
5 REM *******************************
6 REM
7 CLS
8 REM ev. 9015 aendern, falls Bit2 benutzt ist.
9 DIM W(96)
10 PRINTTAB(10)"FUER ELISE (L.v.Beethoven)"
15 GOSUB 8500: REM MASCHINEN PROGRAMM ABLEGEN
20 DOKE 4100,RAM :REM USER LOC EINTRAGEN
30 U=94:FOR T=1 TO U
50 READ  W(T):X=0
60 IF W(T) >1000 THEN X=1:W(T)=W(T)-1000
70 GOSUB 1000
80 B=USR(W(T))
90 NEXT
100 CLS:RESTORE
110 END
1000 TN=3334
1001 X=X+1
1010 ON X GOSUB 1030,1050,1070,1090
1020 RETURN
1030 POKE TN,16:RETURN
1050 POKE TN,32:RETURN
1070 POKE TN,32:RETURN
1090 POKE TN,40:RETURN
8000 DATA43,46,43,46,43,57,48,54,1065,107,86
8010 DATA65,1057,86,68,57,1054,86,43,46,43,46
8020 DATA43,57,48,54,1065,107,86,65,1057,86,54
8030 DATA57,1065,57,54,48,1043,72,41,43,1048
8040 DATA81,43,48,1054,86,48,54,1057,86,86,43
8045 DATA 86,43,43,1021
8050 DATA46,43,46,43,46,43,57,48,54
8060 DATA1065,107,86,65,1057,86,68,57,1054,86,43
8070 DATA46,43,46,43,57,48,54,1065,107,86,65,1057
8080 DATA86,54,57,1065,1,1,1,1
8500 RESTORE 9000:RAM=3328
8510 FOR I=0 TO 29
8520 READ BYTE:POKE RAM+I,BYTE
8530 NEXT:RESTORE:RETURN
9000 DATA 205,139,233,75,17,0,16,65,62,0,0,0,0
9010 DATA 205,83,0,27,122,179,200,16,242,65,62
9015 DATA 32 :REM 32=20H => BIT 5 von Port0
9020 DATA 205,83,0,24,242
```

# Seite(n) für Einsteiger von Peter Brendel

Als Fortsetzung zur letzten "Seite für Einsteiger" nun ein Programm von Peter Brendel, das Ihnen die Möglichkeit gibt, die Poke Befehle nochmals praktisch zu erproben. Er hat es speziell Hans-Dieter Schneider gewidmet und dazu noch einen Bildschirm-Atlas des NASCOM beigelegt, um Benutzern von anderen Systemen den NASCOM Bildschirm-Aufbau durchsichtiger zu machen. (NASCOM-Benutzer werden sicher auch dankbar für diese Programmierhilfe sein).

```
10 REM * HANDHABUNG DES POKE - BEFEHLS
20 REM *  ZUR GRAFIK - DARSTELLUNG
30 REM * (Mit kleinem demo-Anhang)
40 N=9
50 CLS
60 FOR I=1 TO N
70 REM
80 REM   * WIEVIELE ELEMENTE ? (DATA-BLOECKE)
90 REM
100 READ A,B,C,D,E,F
110 REM                    * LESEN 1 ELEMENT
120 FOR J=1 TO E
130 REM
140 REM   * LINIE WIRD E-1 MAL WIEDERHOLT
150 REM   * D.H. E=2 EINMALIGE WIEDERHOLG.
160 REM   * BZW. E=1 NICHT WIEDERHOLT
170 REM
180 FOR K=A TO B STEP C:POKE K,D
190 REM
200 REM   * LINIE BESTEHEND AUS DEM ZEICHEN
210 REM   * D WIRD VOM PUNKT(SPEICHERPL.) A
220 REM   * NACH B MIT DER SCHRITTWEITE C
230 REM   * GEZOGEN.
240 REM
250 NEXT K
260 B=B+F:A=A+F
270 REM         * WENN DIE LINIE WIEDERHOLT
280 REM         * WIRD,DANN MIT DEM ABSTAND F
290 REM         * WENN NICHT WIEDERHOLT WIRD,
300 REM         * WERDEN A UND B ZWAR UM F
310 REM         * ERHOEHT,ABER SONST IGNORIERT,
320 REM         * WEIL AB NEXT J DER NAECHSTE
330 REM         * DATA-BLOCK GELESEN WIRD
340 NEXT J:NEXT I
350 DATA2954,3065,1,255,1,0
360 DATA2058,2890,64,255,2,47
370 DATA2190,2894,64,149,2,30
380 DATA2191,2895,64,150,9,1
390 DATA2200,2904,64,151,2,21
400 DATA2126,2136,1,246,1,0
410 DATA2713,2740,1,128,1,0
420 DATA2777,2804,5,199,3,64
430 DATA2138,2157,4,173,2,1
440 REM  A    B   C  D  E  F
450 REM ANF  END STEP DAT ANZ ABSTAND
460 REM
470 REM * MIR IST KEINE RATIONELLERE MOEGLICH -
480 REM * KEIT IN BASIC BEKANNT * P.B.*
```

Bildschirm-Atlas auf der nächsten Seite!

---

Im Journal 2-82 wurden die Ladebefehle erklärt. Hier nun endlich die Fortsetzung, um dieses Thema abzuschließen.
In der ersten Folge wurden die Register I und R ausgespart. Das I Register ist ein "Interrupt (=Unterbrechung) Adressen Register". Es enthält das höherwertige Byte einer Adresse, die nach einem (Hardware-) Interrupt angesprungen wird. Dies ist eine besondere Anwendung, die auch in einer gesonderten Fortsetzung behandelt werden sollte. (Wäre doch etwas für Sie, Herr Föße!?)
Das R (Refresh=Auffrischen)- Register wird von der CPU verwendet, um die Speicherinhalte der dynamischen RAM- Bausteine vor dem "Verschwinden" zu bewahren. Normalerweise müssen wir uns beim Programmieren überhaupt nicht damit beschäftigen, denn die CPU macht das automatisch bei jedem Einholen eines Befehls.
Eine Möglichkeit, das R-Register auch in einem Programm zu benutzen, ist das Erzeugen einer Zufallszahl zwischen 0 und 255 (0 und FF hex). Der Befehl ED 4F (=LD A, R) lädt dann eine zufällige Zahl in den Akku. Allerdings wird die Zahl nach einer bestimmten Anzahl von Maschinenzyklen wiederholt; bei den meisten Anwendungen (z.B. Spielen) genügt aber diese "Zufälligkeit".
Nun aber zurück zu unseren (inzwischen hoffentlich) vertrauten Registerpaaren AF, HL

```
16  3018 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5
 1  2058 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7 8 9100 1 2 3 4 5
 2  2122 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9
 3  2186 7 8 9 9 1 2 3 4 5 6 7 8 9200 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5 6 7 8 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3
 4  2250 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7
 5  2314 5 6 7 8 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1
 6  2378 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7 8 9400 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5 6 7 8 9 2 1 2 3 4 5
 7  2442 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9
 8  2506 7 8 9 1 1 2 3 4 5 6 7 8 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3
 9  2570 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7 8 9600 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5 6 7
10  2634 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1
11  2698 9700 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5 6 7 8 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5
12  2762 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7 8 9800 1 2 3 4 5 6 7 8 9
13  2826 7 8 9 3 1 2 3 4 5 6 7 8 9 4 1 2 3 4 5 6 7 8 9 5 1 2 3 4 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3
14  2890 1 2 3 4 5 6 7 8 9900 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5 6 7 8 9 2 1 2 3 4 5 6 7 8 9 3 1 2 3 4 5 6 7
15  2954 5 6 7 8 9 6 1 2 3 4 5 6 7 8 9 7 1 2 3 4 5 6 7 8 9 8 1 2 3 4 5 6 7 8 9 9 1 2 3 4 5 6 7 8 9300 1

    1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48
       11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47
```

BILDSCHIRM - ATLAS für NASCOM 1/2 (dez.)

etc.
Durch einfache 1-Byte- Befehle kann man die ganz schön zum Rotieren bringen (allerdings auch den Programmierer)! Gemeint sind die PUSH- und POP- Befehle. Dazu muß man wissen daß im RAM- Speicher ein sogenannter STACK angelegt ist; d.h. ein Stapelspeicher, dessen Beginn durch den STACKPOINTER (=SP= Stapelzeiger - siehe Registeranzeige Heft 2-82) im NASCOM auf 1000 festgelegt ist. Der Programmierer kann ihn allerdings durch den Befehl LD SP,nn (Lade Stapelzeiger mit nn) auf eine x-beliebige Adresse legen (und dadurch z.B. verschiedene Stacks im Speicher anlegen). Diese Anwendung scheint mir aber schon etwas "fortgeschritten". Lassen wir unseren Stack zunächst einmal bei 1000.
Durch den PUSH- Befehl wird nun ein Register auf diesen Stack geladen; d.h. der Stackpointer wird jeweils um 1 erniedrigt, und die beiden Komponenten des Registerpaares werden in die entsprechenden Adressen geladen.
Beispiel:
Stackpointer zeigt auf 1000 (Top of Stack= Obergrenze oder Spitze des Stapels)
Programm: LD HL,1122
PUSH HL
Nach Ausführung mit Single-Step enthalten
0FFF - 22 und
0FFE - 11
Wird nun ein anderes Register "gepusht", dann wächst der Stapel in die Tiefe ( erniedrigt den Stackpointer weiter),und das andere Register liegt obenauf. Daraus folgt, daß das zuletzt "gepushte" Register als erstes wieder "gepoppt" werden muß. (Man spricht von Last In/First Out= LIFO = als letztes hinein, als erstes heraus).
Wenn man diese Reihenfolge umgeht, kann man auch leicht Register austauschen, die normalerweise nicht getauscht werden könnten. Z. B. kann HL mit DE durch den Code EB (EX HL, DE) getauscht werden. Für einen Austausch von DE und BC existiert kein solcher Befehl, (es sei denn, man tauscht alle Register gleichzeitig, was ja wohl nicht immer wünschenswert ist). Um auch hier austauschen zu können, legen wir erst DE und dann BC auf den Stack und holen sie in gleicher Reihenfolge wieder herein:

```
PUSH DE   first in
PUSH BC
POP  DE
POP  BC   last out
```

Die Register sind nun vertauscht.
Die häufigste Anwendung dürfte es allerdings sein, bestimmte Register, deren Inhalt später nochmals gebraucht wird, in "Sicherheit" zu bringen, wenn sie für andere Zwecke kurzzeitig benötigt werden (z.B. in Unterprogrammen). So sieht man häufig vor einem Call- Befehl (Aufruf eines Unterprogramms) ein ganzes Sammelsurium von PUSH- Befehlen:

```
PUSH AF
PUSH HL
PUSH BC
CALL XYZ
POP  BC
POP  HL
POP  AF
```

Beachten Sie die Reihenfolge (FILO!), damit keine Register vertauscht werden!
Vor dem Aufruf des Unterprogramms wird übrigens die Rücksprungadresse automatisch auf dem Stack abgelegt, damit das Programm weiß, wo es nach dem Abarbeiten des Unterprogramms weitermachen muß. Hier muß man höllisch aufpassen, damit man in einem Unterprogramm nicht versehentlich (z.B. nach einer Verzweigung) vergißt, ein Register zu "poppen", das vorher "gepusht" wurde. Dann Springt das Programm nämlich zur zuletzt gepushten (ich spare mir jetzt die Anführungszeichen bei diesem seltsamen neudeutschen Ausdruck. Herr Bach sei mir gnädig!) Adresse, weil seine Rücksprungadresse ja weiter unten im Stack liegt. Wer weiß, wo es dann ankommt! Sehr häufig sind Programme beim Test abgestürzt, weil man einfach den Überblick über die Adressen auf dem Stack verloren hatte. Sie können in einem Unterprogramm auch keine Werte vom Stack nehmen, weil Sie ja sonst Ihre Rücksprungadresse zerstören. Ich schreibe mir jeden PUSH- Befehl gesondert auf und hake dann jeden POP- Befehl ab, um einen Überblick zu behalten. Dennoch kann ich gelegentliche Abstürze nicht vermeiden.
Grundsätzlich sollte nach Abarbeiten eines Programms der Stackpointer wieder auf 1000 zeigen. Tut er das nicht, (was ein Programm bei bestimmten Bedingungen durchaus einmal verkraften könnte), liegt ein Programmierfehler vor, der früher oder später (wenn der Stack dann systematisch in ein Maschinenprogramm hineinwächst) dann doch zum "Absturz" führt.
Der Umfang des Stacks, der je nach Anzahl

Fortsetzung Seite 43

# STOCKCAR

## von Wolfgang Schröder

```
10 CLS
20 A$="STOCK CAR"
30 FORA=1TOLEN(A$)
40 SCREENA+14,16
50 PRINTMID$(A$,A,1):NEXT
60 PRINT:PRINT"SIE STEUERN IHR AUTO UEBER DEN B
ILD-"
70 PRINT"SCHIRM MIT HILFE DER CURSOR-TASTEN."
80 PRINT"JEDESMAL WENN SIE EIN AUTO RAMMEN"
90 PRINT"WIRD IHRE PUNKTZAHL UM 1 ERHOEHT."
100 PRINT"WENN SIE VERSUCHEN, DAS SPIELFELD"
110 PRINT"ZU VERLASSEN, WIRD IHR ACCIDENT-"
120 PRINT"COUNT INKREMENTIERT. JEDESMAL WENN"
130 PRINT"SIE EIN AUTO RAMMEN WIRD EIN KREUZ"
140 PRINT"ERSCHEINEN. RAMMEN SIE DIESES KREUZ"
150 PRINT"WIEDER, WIRD IHR ACCIDENT-COUNT UM"
160 PRINT"EINS ERHOEHT. JEDER UNFALL KOSTET"
170 PRINT"5 PUNKTE MAL DIE RUNDENZAHL. VON"
180 PRINT"RUNDE ZU RUNDE WIRD IHR AUTO IMMER"
190 PRINT"SCHNELLER."
200 FORA=1TO29999:NEXT:PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT:PRINT
210 PRINT"IHRE ENDPUNKTZAHL ERRECHNET SICH AUS"
220 PRINT"DEN ERREICHTEN PUNKTEN MAL RUNDEN-"
230 PRINT"ZAHL. DER MAXIMALE FAKTOR IST HIER"
240 PRINT"9. AM ANFANG HABEN SIE NEUN ZEITEIN-"
250 PRINT"HEITEN ZEIT IN DER SICH DIE AUTOS"
260 PRINT"NICHT BEWEGEN. DANACH STEHEN IHNEN"
270 PRINT"NOCH 50 ZEITEINHEITEN BIS ZUM ENDE"
280 PRINT"DER RUNDE ZU. DANACH WIRD IHNEN IHR"
290 PRINT"ENDPUNKTSTAND AUSGEDRUCKT MIT IHREM"
300 PRINT"JEWEILIGEN REKORD."
310 PRINT:PRINT"SO, UND NUN VIEL SPASS!!"
320 FORA=1TO15000:NEXT
330 A1=0:DIMH(20):IC=1
340 CLS:CC=34-IC*4:IFCC<1THENCC=1
350 IFIC>9THENIC=9
360 A$="TIME    ACCIDENTS    SCORE"
370 FORD=1TOLEN(A$)
380 SCREEND+5,16
390 PRINTMID$(A$,D,1)
400 NEXT
410 B=2954:C=2122
420 FORA=2058TO2105
430 POKEA,5:POKEB,5
440 IFC>3001THEN470
450 POKEC,5:C=C+47
460 POKEC,5:C=C+17
470 B=B+1
480 NEXT
490 E=INT(10*RND(1)+6)
500 FORF=1TOE
510 G=INT(900*RND(1)+2090)
520 IFPEEK(G)<>32THEN510
530 POKEG,14
540 H(F)=G
550 NEXT:RESTORE
560 FORI=20480TO20498STEP2
570 READA:DOKEI,A:NEXT
580 POKE20500,233:DOKE4100,20480
590 I=2595
600 POKEI,7
610 K=45:J=57:L=0:X=0:W=0:Y=0:S=49:U=S:T=48:V=T
620 IFK<>46THEN670
630 POKE3029,K:POKE3030,J
640 IFJ=49THENK=48
650 J=J-1
660 GOTO770
670 IFJ<>48ORK<>48THEN710
680 POKE3029,K:POKE3030,J
690 J=J+1
700 GOTO770
710 X=X+1:IFX=50THEN1110
720 J=J+1:IFJ=58THEN750
730 POKE3030,J
740 GOTO770
750 K=K+1:POKE3029,K
760 J=48:POKE3030,J
770 L=L+1:IFL=FTHENL=0:GOTO620
780 IFK=45THEN880
790 M=INT(RND(1 )*4+1)
800 IFM=1THENN=64
810 IFM=2THENN=-1
820 IFM=3THENN=1
830 IFM=4THENN=-64
840 IFPEEK(H(L)+N)=5THEN790
850 POKEH(L),32
860 H(L)=H(L)+N
870 POKEH(L),14
880 FORO=1TOCC
890 P=USR(0)
900 IFP=0THEN920
910 Q=P
920 NEXT
930 R=I
940 IFQ=17THENI=I-1
950 IFQ=18THENI=I+1
960 IFQ=19THENI=I-64
970 IFQ=20THENI=I+64
980 IFPEEK(I)=50RPEEK(I)=25THEN1020
990 IFPEEK(I)=14THEN1070
1000 POKER,32:POKEI,7
1010 GOTO770
1020 W=W+1:POKER,32:IFS<>57THENT=T+1:S=48
```

```
1030 POKE3041,T:POKE3042,S
1040 S=S+1
1050 I=2595:POKEI,7:Q=0
1060 GOTO770
1070 Y=Y+1:POKER,25:IFU=57THENV=V+1:U=48
1080 POKE3059,V:POKE3061,U
1090 U=U+1
1100 GOTO770
1110 CLS
1120 PRINTTAB(10)"GAME OVER        END"
1130 PRINT
1140 PRINTTAB(4)"SCORE= (*"IC")";TAB(37);Y
1150 PRINTTAB(4)"ACCIDENTS= (TIMES FIVE)";TAB(3
6);
1160 PRINT"-";W
1170 PRINTTAB(36)"-----":Y=Y*IC:W=W*IC
1180 PRINTTAB(4)"TOTAL =";TAB(37);:Z=W*5:PRINTY
-Z
1190 PRINTTAB(35)"- - - - -"
1200 IFY-Z=A1THEN1230
1210 PRINT:PRINTTAB(10)"RECORD SCORE=";A1
1220 GOTO1250
1230 A1=Y-Z
1240 PRINTTAB(10)"YOU'VE JUST SET A NEW RECORD"
1250 PRINT:PRINT:PRINT:IC=IC+1
1260 INPUT"DO YOU WANT ANOTHER TRY";C$
1270 IFLEFT$(C$,1)="Y"THENQ=0:GOTO340
1280 PRINT"PROGRAM END"
1290 DATA289,1548,13833,8960,-12,25311
1300 DATA312,18351,10927,-8179 *
```

* für die seltsame zweite BASIC Version (siehe Heft 1/82) muß hier der Wert-8182 eingesetzt werden Red.

# ZAUBERWÜRFEL

## von Erich Mehnert

Der verdrehbare Zauberwürfel besitzt statt der herkömmlichen neun Felder 25 Felder pro Würfelfläche. Alle sechs Flächen sind auf dem Bildschirm sichtbar. Oben rechts läuft eine Uhr, die mit ":" auf 0 gestellt werden kann. Start des Programms ist bei FD0.
Die Tasten 1,2,3,4,5; Q,W,R,T und A,S,D,F,G werden zur Verdrehung des Würfels betätigt. Jede Drehung wird angezeigt. Mit der Sterntaste wird zu NASSYS zurückgesprungen, mit dem Schnecken-A kann der Würfel wieder in die Ausgangslage gebracht werden.

Das besondere an dem Programm ist, daß man das "Würfelknacken" in mehreren Etappen durchführen kann, indem man die augenblickliche Lage des Würfels auf Kassette abspeichert und das Spiel am nächsten Tag fortsetzt.

```
0C80 21 12 12 11 6A 0B 01 0A 62
0C88 00 ED B0 C9 3E 0C F7 C9 04
0C90 21 36 00 00 00 00 19 22 2E
0C98 84 0C C9 2A 81 0C 11 28 ED
0CA0 00 00 00 19 22 81 0C C9 3D
0CA8 CD 80 0C CD 90 0C CD 9B DE
0CB0 0C C9 CD A8 0C CD A8 0C 93
0CB8 CD A8 0C CD A8 0C CD A8 3B
0CC0 0C C9 21 00 10 22 81 0C 81
0CC8 21 8C 08 22 84 0C CD B2 BA
0CD0 0C 21 0A 10 22 81 0C 21 F3
0CD8 9B 08 22 84 0C CD B2 0C C4
0CE0 21 14 10 22 81 0C 21 AA AB
0CE8 08 22 84 0C CD B2 0C 21 5A
0CF0 1E 10 22 81 0C 21 0C 0A 10
0CF8 22 84 0C CD B2 0C 21 40 A2
0D00 11 22 81 0C 21 1B 0A 22 35
0D08 84 0C CD B2 0C 21 4A 11 AC
0D10 22 81 0C 21 2A 0A 22 84 C7
0D18 0C CD B2 0C C9 21 6E 10 24
0D20 22 81 0C 21 00 14 22 84 B7
0D28 0C CD 80 0C 21 50 10 22 3D
0D30 81 0C 21 0A 14 22 84 0C BB
0D38 21 1D 00 22 87 0C CD 80 85
0D40 0C 21 00 14 22 81 0C 21 5E
0D48 50 10 22 84 0C 21 28 00 B0
0D50 22 87 0C CD 80 0C 21 0A 96
0D58 00 22 87 0C CD C2 0C C9 7E
0D60 21 1E 10 22 1E 0D 21 00 2A
0D68 10 22 2D 0D 22 48 0D CD 25
0D70 1D 0D C9 21 46 10 22 1E 27
0D78 0D 21 28 10 22 2D 0D 22 69
0D80 48 0D CD 1D 0D C9 21 6E 31
0D88 10 22 1E 0D 21 50 10 22 95
0D90 2D 0D 22 48 0D CD 1D 0D 45
0D98 C9 21 96 10 22 1E 0D 21 A3
0DA0 78 10 22 2D 0D 22 48 0D 08
0DA8 CD 1D 0D C9 21 BE 10 22 86
0DB0 1E 0D 21 A0 10 22 2D 0D 15
0DB8 22 48 0D CD 1D 0D C9 3A 36
0DC0 26 14 32 B4 10 C9 2A C0 B0
0DC8 0D 11 02 00 19 22 C0 0D FD
0DD0 2A C3 0D 11 28 00 19 22 4B
0DD8 C3 0D C9 3E 28 32 CA 0D ED
0DE0 3E 02 32 D4 0D C9 3E 02 49
0DE8 32 CA 0D 3E 28 32 D4 0D 77
0DF0 C9 21 14 10 22 C0 0D 21 1B
0DF8 00 14 22 C3 0D CD DB 0D C0
0E00 CD BF 0D CD 10 0E CD 10 6F
0E08 0E CD 10 0E CD 10 0E C9 C3
0E10 CD C6 0D CD BF 0D C9 21 41
0E18 1E 14 22 C0 0D 21 14 10 8C
0E20 22 C3 0D CD E6 0D CD BF 6C
0E28 0D CD 10 0E CD 10 0E CD E6
0E30 10 0E CD 10 0E C9 21 00 31
0E38 00 22 81 0C 21 00 00 22 38
0E40 84 0C CD 80 0C C9 21 08 29
0E48 10 22 F2 0D 21 14 14 22 F2
0E50 F8 0D CD F1 0D 21 14 10 73
0E58 22 F2 0D 21 00 14 22 F8 D6
0E60 0D CD F1 0D 21 EA 11 22 84
0E68 81 0C 21 1E 14 22 84 0C 08
0E70 21 0A 00 22 87 0C CD 80 AB
0E78 0C 21 40 11 22 81 0C 21 D4
0E80 0A 14 22 84 0C CD 80 0C B7
0E88 21 00 14 22 81 0C 21 40 DB
0E90 11 22 84 0C CD 80 0C 21 DB
0E98 14 14 22 81 0C 21 EA 11 99
0EA0 22 84 0C CD 80 0C 21 0A E4
```

```
0EA8 14 22 18 0E 21 08 10 22 6D
0EB0 1E 0E CD 17 0E 21 1E 14 2F
0EB8 22 18 0E 21 14 10 22 1E 93
0EC0 0E CD 17 0E CD C2 0C C9 32
0EC8 21 08 10 22 47 0E 22 AD 55
0ED0 0E 21 14 10 22 56 0E 22 D9
0ED8 BC 0E 21 40 11 22 7A 0E CC
0EE0 22 8F 0E 21 EA 11 22 65 50
0EE8 0E 22 9E 0E CD 46 0E C9 BC
0EF0 21 08 10 22 C9 0E 21 14 65
0EF8 10 22 D2 0E 21 40 11 22 AC
0F00 DB 0E 21 EA 11 22 E4 0E 28
0F08 CD C8 0E C9 21 06 10 22 DC
0F10 C9 0E 21 16 10 22 D2 0E 3F
0F18 21 68 11 22 DB 0E 21 C2 AF
0F20 11 22 E4 0E CD C8 0E C9 C0
0F28 21 04 10 22 C9 0E 21 18 9E
0F30 10 22 D2 0E 21 90 11 22 35
0F38 DB 0E 21 9A 11 22 E4 0E 10
0F40 CD C8 0E C9 21 02 10 22 10
0F48 C9 0E 21 1A 10 22 D2 0E 7B
0F50 21 B8 11 22 DB 0E 21 72 E7
0F58 11 22 E4 0E CD C8 0E C9 F8
0F60 21 00 10 22 C9 0E 21 1C D6
0F68 10 22 D2 0E 21 E0 11 22 BD
0F70 DB 0E 21 4A 11 22 E4 0E F8
0F78 CD C8 0E C9 3A C4 0F 3C 3C
0F80 27 32 C4 0F 00 21 1F 08 03
0F88 22 29 0C 00 DF 68 3A C4 33
0F90 0F FE 00 CA 97 0F C9 3A 1F
0F98 C5 0F 3C 27 32 C5 0F 21 05
0FA0 1D 08 22 29 0C DF 68 3A AC
0FA8 C5 0F FE 00 CA B0 0F C9 DB
0FB0 3A C6 0F 00 3C 00 27 00 31
0FB8 32 C6 0F 21 1B 08 22 29 5D
0FC0 0C DF 68 C9 57 00 00 CD 0F
0FC8 7C 0F 3E 00 C9 00 00 00 69
0FD0 3E 0C F7 CD C2 0C CD 50 D8
0FD8 15 C3 BD 13 80 07 80 07 9D
0FE0 80 07 65 07 24 35 FA 0B 40
0FE8 48 0C 79 02 B8 01 0C 0C 97
0FF0 6B 07 0C 0C 00 00 34 0C C9
0FF8 85 04 00 00 00 00 01 30 C1
1000 00 20 00 20 00 20 00 20 90
1008 0E 20 05 20 05 20 05 20 B5
1010 05 20 05 20 1B 20 10 20 D5
1018 10 20 10 20 10 20 4F 20 27
1020 4F 20 4F 20 4F 20 4F 20 EC
1028 00 20 00 20 00 20 00 20 B8
1030 0E 20 05 20 05 20 05 20 DD
1038 05 20 05 20 1B 20 10 20 FD
1040 10 20 10 20 10 20 4F 20 4F
1048 4F 20 4F 20 4F 20 4F 20 14
1050 00 20 00 20 00 20 00 20 E0
1058 0E 20 05 20 05 20 05 20 05
1060 05 20 05 20 1B 20 10 20 25
1068 10 20 10 20 10 20 4F 20 77
1070 4F 20 4F 20 4F 20 4F 20 3C
1078 00 20 00 20 00 20 00 20 08
1080 0E 20 05 20 05 20 05 20 2D
1088 05 20 05 20 1B 20 10 20 4D
1090 10 20 10 20 10 20 4F 20 9F
1098 4F 20 4F 20 4F 20 4F 20 64
10A0 00 20 00 20 00 20 00 20 30
10A8 0E 20 05 20 05 20 05 20 55
10B0 05 20 05 20 1B 20 10 20 75
10B8 10 20 10 20 10 20 4F 20 C7
10C0 4F 20 4F 20 4F 20 4F 20 8C
10C8 0E 20 0E 20 0E 20 0E 20 90
10D0 0E 20 1B 20 1B 20 1B 20 BF
10D8 1B 20 1B 20 10 20 10 20 BE
10E0 10 20 10 20 10 20 4F 20 EF
10E8 4F 20 4F 20 4F 20 4F 20 B4
10F0 0E 20 0E 20 0E 20 0E 20 B8
10F8 0E 20 1B 20 1B 20 1B 20 E7
1100 1B 20 1B 20 10 20 10 20 E7
1108 10 20 10 20 10 20 4F 20 18
1110 4F 20 4F 20 4F 20 4F 20 DD
1118 0E 20 0E 20 0E 20 0E 20 E1
1120 0E 20 1B 20 1B 20 1B 20 10
1128 1B 20 1B 20 10 20 10 20 0F
1130 10 20 10 20 10 20 4F 20 40
1138 4F 20 4F 20 4F 20 4F 20 05
1140 10 20 10 20 10 20 10 20 11
1148 10 20 1B 20 1B 20 1B 20 3A
1150 1B 20 1B 20 10 20 10 20 37
1158 10 20 10 20 10 20 4F 20 68
1160 4F 20 4F 20 4F 20 4F 20 2D
1168 0E 20 0E 20 0E 20 0E 20 31
1170 0E 20 1B 20 1B 20 1B 20 60
1178 1B 20 1B 20 10 20 10 20 5F
1180 10 20 10 20 10 20 4F 20 90
1188 4F 20 4F 20 4F 20 4F 20 55
1190 0E 20 0E 20 0E 20 0E 20 59
1198 0E 20 1B 20 1B 20 1B 20 88
11A0 1B 20 1B 20 10 20 10 20 87
11A8 10 20 10 20 10 20 4F 20 B8
11B0 4F 20 4F 20 4F 20 4F 20 7D
11B8 0E 20 0E 20 0E 20 0E 20 81
11C0 0E 20 1B 20 1B 20 1B 20 B0
11C8 1B 20 1B 20 10 20 10 20 AF
11D0 10 20 10 20 10 20 4F 20 E0
11D8 4F 20 4F 20 4F 20 4F 20 A5
11E0 0E 20 0E 20 0E 20 0E 20 A9
11E8 0E 20 00 20 00 20 00 20 87
11F0 00 20 00 20 10 20 10 20 A1
11F8 10 20 10 20 10 20 4F 20 08
1200 21 0A 10 22 F2 0D 21 00 8F
1208 14 22 F8 0D CD F1 0D 21 41
1210 40 11 22 F2 0D 21 0A 14 D3
1218 22 F8 0D CD F1 0D 21 1E 5B
1220 10 22 F2 0D 21 14 14 22 CE
1228 F8 0D CD F1 0D 21 4A 11 86
1230 22 F2 0D 21 1E 14 22 F8 D0
1238 0D CD F1 0D 21 00 14 22 79
1240 18 0E 21 40 11 22 1E 0E 38
1248 CD 17 0E 21 0A 14 22 18 C5
1250 0E 21 1E 10 22 1E 0E CD DA
1258 17 0E 21 14 14 22 18 0E 20
1260 21 4A 11 22 1E 0E CD 17 20
1268 0E 21 1E 14 22 18 0E 21 44
1270 0A 10 22 1E 0E CD 17 0E DC
1278 CD C2 0C C9 21 0A 10 22 4B
1280 01 12 22 70 12 21 40 11 BB
1288 22 10 12 22 43 12 21 1E 94
1290 10 22 1F 12 22 52 12 21 AC
1298 4A 11 22 2E 12 22 61 12 FC
12A0 CD 00 12 C9 21 0A 10 22 B7
12A8 7D 12 21 40 11 22 86 12 75
12B0 21 1E 10 22 8F 12 21 4A 3F
12B8 11 22 98 12 CD 7C 12 C9 CB
12C0 21 0C 10 22 7D 12 21 42 23
12C8 11 22 86 12 21 20 10 22 18
12D0 8F 12 21 4C 11 22 98 12 CD
12D8 CD 7C 12 C9 21 0E 10 22 6F
12E0 7D 12 21 44 11 22 86 12 B1
12E8 21 22 10 22 8F 12 21 4E 7F
12F0 11 22 98 12 CD 7C 12 C9 03
12F8 21 10 10 22 7D 12 21 46 63
1300 11 22 86 12 21 24 10 22 55
1308 8F 12 21 50 11 22 98 12 0A
1310 CD 7C 12 C9 21 12 10 22 AC
1318 7D 12 21 48 11 22 86 12 EE
1320 21 26 10 22 8F 12 21 52 C0
1328 11 22 98 12 CD 7C 12 C9 3C
1330 21 00 16 22 81 0C 21 00 4A
1338 10 22 84 0C 21 FF 01 22 50
1340 87 0C CD 80 0C 21 0A 00 6A
1348 22 87 0C CD 8C 0C CD C2 04
1350 0C C3 BD 13 CD A4 12 CD 52
1358 C7 0F C9 CD C0 12 CD C7 3D
1360 0F C9 CD DC 12 CD C7 0F A9
1368 C9 CD F8 12 CD C7 0F C9 87
1370 CD 14 13 CD C7 0F C9 CD B0
1378 C7 0F CD 60 0D C9 CD C7 F8
1380 0F CD 73 0D C9 CD C7 0F 5B
1388 CD 86 0D C9 CD C7 0F CD 34
1390 99 0D C9 CD C7 0F CD AC 2E
1398 0D C9 CD F0 0E CD C7 0F EF
13A0 C9 CD 0C 0F CD C7 0F C9 D0
13A8 CD 28 0F CD C7 0F C9 CD F8
13B0 44 0F CD C7 0F C9 CD 60 AF
13B8 0F CD C7 0F C9 21 99 99 99
13C0 22 C4 0F 3E 99 32 C6 0F A6
13C8 CD C7 0F C3 29 14 00 00 7E
13D0 00 00 00 00 00 00 00 00 E3
13D8 00 00 00 00 00 00 00 00 EB
13E0 00 00 00 00 00 00 00 00 F3
13E8 00 00 00 00 00 00 00 00 FB
13F0 00 00 00 00 00 00 00 00 03
13F8 00 00 00 00 00 00 00 20 2B
1400 10 20 10 20 10 20 10 20 D4
1408 10 20 0E 20 0E 20 0E 20 D6
1410 0E 20 0E 20 00 20 00 20 C0
1418 00 20 00 20 00 20 1B 20 C7
1420 1B 20 1B 20 1B 20 1B 20 20
1428 00 DF 62 FE 31 CC 77 13 02
1430 FE 32 CC 7E 13 FE 33 CC CE
1438 85 13 FE 34 CC 8C 13 FE 7F
1440 35 CC 93 13 FE 51 CC 9A B0
1448 13 FE 57 CC A1 13 FE 45 87
1450 CC A8 13 FE 52 CC AF 13 C9
1458 FE 54 CC B6 13 FE 41 CC 5E
1460 54 13 FE 53 CC 5B 13 FE 64
1468 44 CC 62 13 FE 46 CC 69 7A
1470 13 FE 47 CC 70 13 FE 2A 53
1478 CA 00 00 FE 40 CA 60 15 D3
1480 FE 3A CC 50 15 CD 14 15 F3
1488 CD 8E 14 C3 29 14 3A 10 55
1490 15 00 3C 27 37 32 10 15 AA
1498 00 00 00 00 00 00 00 00 AC
14A0 3A 10 15 FE 99 CA A9 14 31
14A8 C9 3A 11 15 00 3C 27 37 7F
14B0 FE 60 CA B8 14 C3 BA 14 49
14B8 3E 00 32 11 15 21 2D 08 B8
14C0 22 29 0C DF 68 3A 11 15 D2
14C8 FE 00 CA CE 14 C9 3A 12 9B
14D0 15 00 3C 27 37 FE 60 CA BB
14D8 DD 14 C3 DF 14 3E 00 32 03
14E0 12 15 21 2A 08 22 29 0C C5
14E8 DF 68 3A 12 15 FE 00 CA 6C
14F0 F3 14 C9 3A 13 15 00 3C 72
14F8 27 37 FE 24 CA 02 15 C3 30
1500 04 15 3E 00 32 13 15 21 E7
1508 27 08 22 29 0C DF 68 C9 B3
1510 00 33 01 00 21 A5 00 3E 5D
1518 01 F5 AF 00 00 00 F1 C3
1520 3D C2 19 15 2B 7D B4 C2 80
1528 17 15 00 00 00 00 00 00 69
1530 00 00 00 00 00 00 00 00 45
1538 C9 00 00 00 00 00 00 00 16
1540 00 00 00 00 00 00 00 00 55
1548 00 00 00 00 00 00 00 00 5D
1550 21 98 59 22 10 15 21 59 38
1558 23 22 12 15 C9 00 00 00 A2
1560 CD 50 15 C3 30 13 27 37 0B
1568 32 10 15 CD 7C 0F 3E 00 6A
1570 C9 14 CD C7 0F C9 4F 20 3D
1578 00 20 00 20 00 20 00 20 0D
1580 00 20 05 20 05 20 05 20 24
1588 05 20 05 20 10 20 10 20 47
1590 10 20 10 20 10 20 4F 20 A4
1598 4F 20 4F 20 4F 20 4F 20 69
15A0 00 20 00 20 00 20 00 20 35
15A8 00 20 05 20 05 20 05 20 4C
15B0 05 20 05 20 10 20 10 20 6F
15B8 10 20 10 20 10 20 4F 20 CC
15C0 4F 20 4F 20 4F 20 4F 20 91
15C8 0E 20 0E 20 0E 20 0E 20 95
15D0 0E 20 1B 20 1B 20 1B 20 C4
15D8 1B 20 1B 20 10 20 10 20 C3
15E0 10 20 10 20 10 20 4F 20 F4
15E8 4F 20 4F 20 4F 20 4F 20 B9
15F0 0E 20 0E 20 0E 20 0E 20 BD
15F8 0E 20 1B 20 1B 20 1B 20 EC
1600 00 20 00 20 00 20 00 20 96
1608 00 20 05 20 05 20 05 20 AD
1610 05 20 05 20 10 20 10 20 D0
1618 10 20 10 20 10 20 4F 20 2D
1620 4F 20 4F 20 4F 20 4F 20 F2
```

```
1628 00 20 00 20 00 20 00 20 BE
1630 00 20 05 20 05 20 05 20 D5
1638 05 20 05 20 10 20 10 20 F8
1640 10 20 10 20 10 20 4F 20 55
1648 4F 20 4F 20 4F 20 4F 20 1A
1650 00 20 00 20 00 20 00 20 E6
1658 00 20 05 20 05 20 05 20 FD
1660 05 20 05 20 10 20 10 20 20
1668 10 20 10 20 10 20 4F 20 7D
1670 4F 20 4F 20 4F 20 4F 20 42
1678 00 20 00 20 00 20 00 20 0E
1680 00 20 05 20 05 20 05 20 25
1688 05 20 05 20 10 20 10 20 48
1690 10 20 10 20 10 20 4F 20 A5
1698 4F 20 4F 20 4F 20 4F 20 6A
16A0 00 20 00 20 00 20 00 20 36
16A8 00 20 05 20 05 20 05 20 4D
16B0 05 20 05 20 10 20 10 20 70
16B8 10 20 10 20 10 20 4F 20 CD
16C0 4F 20 4F 20 4F 20 4F 20 92
16C8 0E 20 0E 20 0E 20 0E 20 96
16D0 0E 20 1B 20 1B 20 1B 20 C5
16D8 1B 20 1B 20 10 20 10 20 C4
16E0 10 20 10 20 10 20 4F 20 F5
16E8 4F 20 4F 20 4F 20 4F 20 BA
16F0 0E 20 0E 20 0E 20 0E 20 BE
16F8 0E 20 1B 20 1B 20 1B 20 ED
1700 1B 20 1B 20 10 20 10 20 ED
1708 10 20 10 20 10 20 4F 20 1E
1710 4F 20 4F 20 4F 20 4F 20 E3
1718 0E 20 0E 20 0E 20 0E 20 E7
1720 0E 20 1B 20 1B 20 1B 20 16
1728 1B 20 1B 20 10 20 10 20 15
1730 10 20 10 20 10 20 4F 20 46
1738 4F 20 4F 20 4F 20 4F 20 0B
1740 0E 20 0E 20 0E 20 0E 20 0F
1748 0E 20 1B 20 1B 20 1B 20 3E
1750 1B 20 1B 20 10 20 10 20 3D
1758 10 20 10 20 10 20 4F 20 6E
1760 4F 20 4F 20 4F 20 4F 20 33
1768 0E 20 0E 20 0E 20 0E 20 37
1770 0E 20 1B 20 1B 20 1B 20 66
1778 1B 20 1B 20 10 20 10 20 65
1780 10 20 10 20 10 20 4F 20 96
1788 4F 20 4F 20 4F 20 4F 20 5B
1790 0E 20 0E 20 0E 20 0E 20 5F
1798 0E 20 1B 20 1B 20 1B 20 8E
17A0 1B 20 1B 20 10 20 10 20 8D
17A8 10 20 10 20 10 20 4F 20 BE
17B0 4F 20 4F 20 4F 20 4F 20 83
17B8 0E 20 0E 20 0E 20 0E 20 87
17C0 0E 20 1B 20 1B 20 1B 20 B6
17C8 1B 20 1B 20 10 20 10 20 B5
17D0 10 20 10 20 10 20 4F 20 E6
17D8 4F 20 4F 20 4F 20 4F 20 AB
17E0 0E 20 0E 20 0E 20 0E 20 AF
17E8 0E 20 1B 20 1B 20 1B 20 DE
17F0 1B 20 1B 20 10 20 10 20 DD
17F8 10 20 10 20 10 20 4F 00 EE
```

Fortsetzung von Seite 39 (Einsteiger)

der PUSH- Befehle beträchtlichen Umfang erreichen kann, ist auch der Grund, weshalb man Programme im Speicher ab C8Ø nicht beliebig nach oben ausdehnen kann. Irgendwo im Bereich ab FØØ (meist FEØ oder ähnlich) wird das Programm dann vom herabwachsenden Stack überschrieben und spielt verrückt.

Allgemein wäre zu sagen: PUSH und POP - sehr brauchbare Befehle, die aber genau beobachtet sein wollen.

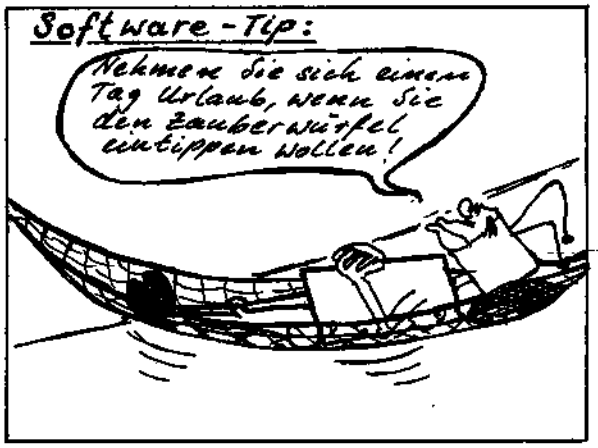
Als Abschluß vielleicht noch eine interessante Anwendung. Unter bestimmten Bedingungen wird ein gepushter Wert nicht mehr gebraucht; dann poppt man ihn einfach in ein nicht benötigtes Register. (Im Quelltext liest man dann oft als Kommentar "Throw away HL", was bedeutet, daß man den Wert von HL auf dem Stack "wegwirft". Als Beispiel:

Sie rufen ein Unterprogramm auf, das wiederum ein Unterprogramm aufruft. Normalerweise würde das zweite Unterprogramm am Ende wieder zum ersten zurückkehren (seine Rücksprungadresse steht ja auf dem Stack). Nehmen wir an, eine Bedingung sei erfüllt, die einen Rücksprung ins Hauptprogramm notwendig macht. Dann löschen Sie einfach die aktuelle Adresse durch z.B. POP BC (Throw away Return Address). Beim Return des zweiten Unterprogramms erfolgt nun der Sprung ins Hauptprogramm, da nun die Rücksprungadresse des ersten Unterprogramms oben auf dem Stack liegt.

Dies klingt beim ersten Durchlesen alles ein wenig kompliziert. Wenn Sie Fragen haben, wenden Sie sich vertrauensvoll ans Journal.

Als nächstes Thema müßten nun die Block-Transfer- Befehle folgen. Wer wagt es?

G.Böhm




-Verkaufe ZEAP 2.Ø auf original EPROMs
4x 27Ø8 DM 13Ø .-
TINI BASIC ebenfalls origin.
2x 27Ø8 DM 4Ø .-
-Suche CLD- oder MKS- Floppy
-Suche PASCAL Interessenten zum Kauf der EPROM-Version des BLS- PASCALs
-Kontakt: Jürgen Weiermann [redacted]
[redacted] (ab 18 h)
[redacted]

# DISASSEMBLER

## von Christoph Rau

In der mc 4/82 war ein Z80-Betriebssystem für den TRS-80 veröffentlicht. Ein Teil davon ist ein Disassembler, der in Maschinensprache programmiert ist und auch im stand-alone betrieben werden kann.
Dazu muß zuerst das Listing von 7207H bis 7739H eingetippt werden. Die Output-Routine des TRS-80 wird in 7571H aufgerufen, dort muß also statt CD 33 00 F7 00 00 (oder entsprechendes) eingetragen werden.

Der Disassembler wird als Unterprogramm mit der Einsprungadresse 7601H aufgerufen. Dabei muß die Adresse des Befehls, der disassembliert werden soll, in HL stehen. Nach einem <CR> werden die Adresse und der disassemblierte Befehl ausgegeben. Die Routine kehrt mit der Adresse des folgenden Befehls in HL zurück. Stand an der Adresse kein gültiger Z80-Befehl, dann bleibt HL unverändert.

Das gesamte Programmpaket benutzt einige Bytes ab 7000H für Flags etc., was man am besten in 7740H umändert. Dazu ersetzt man an den Adresse 7612H, 7644H und 76A5H 70 durch 72, bei 7611H 00 durch 40 und bei 7643H und 76A4H 01 durch 41.
Folgende Bytes sind High-Bytes von absoluten Adresse und müssen beim Relozieren geändert werden (z.B. 62H abziehen, wenn das Programm ab 1000H stehen soll) :
755AH, 7563H, 757BH, 7586H, 7593H, 759FH, 7606H, 7609H, 760EH, 7612H, 7640H, 7644H, 7654H, 7664H, 7678H, 7689H, 76A5H, 76ADH, 76B6H, 76BAH, 76C3H, 76C7H, 76D1H, 76D7H, 76FCH, 7704H, 7713H, 772CH, 7733H.

Mit folgender Änderung schließlich wird erreicht, daß ein <CR> nicht zu Beginn sondern am Ende einer Zeile ausgegeben wird :
Bei 7602H 5 mal NOP (00) eintragen, bei 76D9H einen JP 773AH (C3 3A 77), und bei 773AH LD A,0D (3E 0D), CALL 756FH (CD 6F 75) und RET (C9) anhängen. Diese absoluten Adresse müssen natürlich beim Relozieren mit beachtet werden.

Der folgende Disassembler benutzt den (relozierten) Disassembler aus mc 4/82 ab 1000H bis 1540H. Es können Anfangs- und Endadresse für die Disassemblierung eingegeben werden, und das Programm kann durch ESCAPE angehalten, durch nochmaliges ESCAPE verlassen werden. Einsprungadresse ist 1000H.



```
0010 ; Disassembler-Hauptprogramm
0020 ; (c) Christoph Rau, Bonn
0030 ; Ver 1.2
0040 ; 26.6.82
0050 ARG1    EQU  #0C0C
0060 ARG2    EQU  #0C0E
0070 DISAS   EQU  #1401
0080 ROUT    EQU  #30
0090 RIN     EQU  #08
0100 SCAL    EQU  #18
0110 PRS     EQU  #28
0120 INLIN   EQU  #63
0130 RLIN    EQU  #79
0140 MRET    EQU  #5B
0150 IN      EQU  #62
0160 ;
0170         ORG  #1000
0180         JP   MAIN
0190         ORG  #1371   ; Output
0200         CALL OUTPUT ; Disassembler
0210 MAIN    ORG  #1548
0220         LD   HL,FLAG
0230         RES  0,(HL)
0240         RST  PRS
0250         DEFM "Anfang Ende "
0260         DEFB #0D,#00
0270         RST  SCAL
0280         DEFB INLIN
0290         RST  SCAL
0300         DEFB RLIN
0310         LD   HL,(ARG1)
0320 LAB     PUSH HL
0330         CALL DISAS
0340         POP  DE
0350         OR   A
0360         PUSH HL
0370         SBC  HL,DE ; gueltiger Befehl?
0380         POP  HL
0390         JR   NZ,OK
0400         INC  HL     ; kein Z80-Befehl
0410 OK      LD   DE,(ARG2)
0420         PUSH HL
0430         OR   A
0440         SBC  HL,DE ; fertig?
0450         POP  HL
0460         JR   C,LAB
0470         RST  SCAL
0480         DEFB MRET
0490 ;
0500 FLAG    DEFB 0
0510 ;
0520 OUTPUT CP    #0D
0530         JR   Z,EOLINE
0540 OUTBR1 RST   ROUT ; Zeichen ausgeben
0550         RET
0560 EOLINE RST   SCAL ; Taste gedrueckt am
0570         DEFB IN   ; Ende einer Zeile?
0580         JR   C,KEYPR
0590 OUTBR2 LD    A,#0D
0600         JR   OUTBR1
0610 KEYPR  PUSH  HL
0620         LD   HL,FLAG
0630         CP   #1B       ; Escape?
0640         JR   Z,OUTBR3
0650         RES  0,(HL)
0660         POP  HL
0670         JR   OUTBR2
0680 OUTBR3 BIT   0,(HL) ; 2. Escape?
0690         SET  0,(HL) ; Flag setzen
0700         POP  HL
0710         JR   Z,WAIT
0720         LD   A,#0D
0730         RST  ROUT
0740         RST  SCAL    ; abbrechen
0750         DEFB MRET
0760 WAIT    RST  RIN     ; 1.Escape
0770         JR   KEYPR
  1549
```

# BLS-PASCAL
## von Michael Bach

-Neues vom BLS-Pascal (3. Teil)

Wie ich dem neuesten INMC-News (das jetzt 80-Bus News heißt) entnommen habe, ist das BLS-Pascal, über das ich in den letzten Heften hier schon einiges geschrieben habe (wenn auch vom Nascompl-Formatierer heftig bekämpft), jetzt von Lucas Logic übernommen und in Nascom-Pascal umgetauft worden. Dazu kann ich nur sagen, daß dies eine gute Wahl war. Anbei ist ein weiteres Pascal-Programm, diesmal eine Uhr, d.h. es wird ein Zifferblatt mit sich bewegenden Zeigern dargestellt. Der Anblick ist zugegebenermaßen etwas kläglich, aber das kann jeder ja selbst verbessern (und hier veröffentlichen). Der eigentliche Leckerbissen dabei ist m.E. der "endgültige" Vektor-Algorithmus: Die Prozedur "LINE" zeichnet im Grafik-Mode eine bestmögliche Anpassung an die Verbindungslinie zweier Punkte, und zwar ohne daß eine Multiplikation oder Division nötig wäre! Eine einmalige Division durch 2 läßt sich mit dem SHIFT-Befehl beschleunigen. Da sonst nur Integer Addition- und Subtraktion vorkommt, ist dieser Programmteil sehr schnell; die Linie wird im Nu hingezaubert. Wegen der genannten Eigenschaften eignet sich dieser Algorithmus auch zur Progammierung in Assembler (in Verbindung mit den im Journal 2/82 p.27 veröffentlichten Assembler-Grafik- Unterprogrammen). Wer macht's? Die Anregung zu diesem Algorithmus fand ich in BYTE Aug.81,p414 (M. Higgins). Für Basic-Anhänger, die dies auch ausprobieren wollen: PLOT(X,Y,Z) entspricht bei Z=1: SET(X,Y), bei Z=0: RESET(X,Y) und bei Z=2 wird der Helligkeitswert invertiert. Außerdem ist (0,0) in der Titelzeile, man braucht mit PLOT das lästige "Rumwickeln" also nicht zu berücksichtigen.

Nun noch etwas anderes: Ich finde es im höchsten Maße beschämend, daß die überwältigende Mehrheit der Rechner-Spiele militaristisch ist. Fast immer muß irgendetwas abgeschossen werden. Das finde ich nicht nur langweilig, sondern auch äußerst primitiv! Es gibt natürlich auch positive Ausnahmen, gerade in diesem Journal (!). Ich zähle sie aber lieber nicht auf, sonst vergesse ich noch jemanden, außerdem ist mein Geschmack ja nicht der entscheidende Maßstab.

```
PROGRAM UHR;                                    (*24.7.82*)
(*M.Bach, [redacted], [redacted] Stegen, [redacted]*)

LABEL 1;
CONST L=21; DEL=8350; (*DEL hier fuer 2Mhz*)

VAR I,SEC,MIN,STUN: INTEGER;
    SINT,COST: ARRAY(.0..59.) OF INTEGER;

PROCEDURE LINE(X0,Y0,X1,Y1,Z: INTEGER);
(*Vektor von (X0,Y0) nach (X1,Y1), Z wie in PLOT(X,Y,Z
  nach M.Higgins, BYTE Aug. 81, p414.
  Ohne Multiplikation und Division! *)
VAR I,DX,DY,D,AX,AY,BX,BY: INTEGER;
BEGIN
  DX:=X1-X0; DY:=Y1-Y0; BX:=0; AY:=0;
  IF DX<0 THEN BEGIN AX:=-1; DX:=-DX END ELSE AX:=1;
  IF DY<0 THEN BEGIN BY:=-1; DY:=-DY END ELSE BY:=1;
  IF DX<DY THEN BEGIN
   I:=DX; DX:=DY; DY:=I; BX:=AX; AX:=0; AY:=BY; BY:=0;
  END;
  I:=1; D:=DX SHIFT -1 (*=DIV 2*);
  REPEAT
    PLOT(X0,Y0,Z);
    X0:=X0+AX; Y0:=Y0+AY; D:=D+DY; I:=I+1;
    IF D>DX THEN BEGIN
      D:=D-DX; X0:=X0+BX; Y0:=Y0+BY;
    END;
  UNTIL I>DX
END; (*LINE*)

FUNCTION XWERT(T: INTEGER): INTEGER;
BEGIN XWERT:=((SINT(.T.)*750) DIV 512)+48 END;

FUNCTION YWERT(T: INTEGER): INTEGER;
BEGIN YWERT:=24-COST(.T.) END;

PROCEDURE ZIFBLATT;
VAR I,X0,Y0,X1,Y1,XA,YA: INTEGER;
    R,R1: REAL;
BEGIN
  R:=2*PI/60.0;
  FOR I:=0 TO 59 DO BEGIN
    R1:=R*I;
    SINT(.I.):=ROUND(L*SIN(R1));
    COST(.I.):=ROUND(L*COS(R1));
    X0:=X1; Y0:=Y1;
    X1:=ROUND((SINT(.I.)*1.42)*1.11)+48;
    Y1:=ROUND(COST(.I.)*1.11)+24;
    IF I=0 THEN BEGIN XA:=X1; YA:=Y1 END
```

```
    ELSE LINE(X0,Y0,X1,Y1,1);
  END;
  LINE(X1,Y1,XA,YA,1);
END;

PROCEDURE ZEIGER1(T: INTEGER);
BEGIN
  LINE(48,24,XWERT(T),YWERT(T),2);
END;

PROCEDURE ZEIGER2(T: INTEGER);
PROCEDURE Z2(T,L2: INTEGER);
BEGIN LINE(48,24,((SINT(.T.)*25*L2) DIV 70)+48,
          24-(COST(.T.)*L2 DIV 4),2);
END;
BEGIN
  T:=T*5; Z2(T,3);
  IF T<>0 THEN Z2(T-1,2) ELSE Z2(59,2);
  IF T<>59 THEN Z2(T+1,2) ELSE Z2(0,2);
END;

BEGIN (*HP*)
  STUN:=4; MIN:=0; SEC:=0;
  WRITE(CHR(12),'Uhrzeit (Stunden)? '); READLN(STUN);
  STUN:=STUN MOD 12;
  WRITE('Minuten? '); READ(MIN);
  WRITE(CHR(12));
  ZIFBLATT;
  REPEAT
    ZEIGER2(STUN);
    REPEAT
      ZEIGER1(MIN);
      REPEAT
        IF SEC<>MIN THEN ZEIGER1(SEC);
        FOR I:=0 TO DEL DO;
        IF KEYBOARD<>0 THEN GOTO 1;
        IF SEC<>MIN THEN ZEIGER1(SEC);
        SEC:=SEC+1;
      UNTIL SEC=59; SEC:=0;
      ZEIGER1(MIN);
      MIN:=MIN+1;
    UNTIL MIN=59; MIN:=0;
    ZEIGER2(STUN); STUN:=STUN+1;
  UNTIL FALSE;
END.
```



# Römische Ziffern

## von W. Mayer-Gürr

Bei einem System zur Textverarbeitung sollte als kleiner Pfiff die Ausgabe von Seitenzahlen auch in römischen Ziffern möglich sein. Das Programm ist als .ACM File für CLD-DOS gedacht, kann aber leicht für NAS-SYS geändert werden. Bei dem DOS wird das letzte Zeichen eines Strings mit einem gesetzten 8. Bit gekennzeichnet. Der Nachteil dieser Methode ist, daß Graphik nach dem herkömmlichen Verfahren nicht ausgegeben werden kann. Das Vorbild Heath HDOS verarbeitet Graphik über eine Escapesequenz an das Terminal.

Bei dem Programm wird ein Hexwert im Register A in einen String mit römischen Ziffern gewandelt. Soll direkt ausgegeben werden, muß in den Abschnitten ROEM1 und ROEM2 jeweils der Wert von Register E nach A übertragen werden. Dann folgt beim NAS-SYS ein ROUT (RST 30H). Es können nur Werte bis 255 verarbeitet werden, ein so dickes Buch will erst einmal geschrieben sein!

```
*-----------------------------------------------
* ROEMER.ACM
*
* Aufruf mit CALL $ROEMER
* Der Wert in Register A wird in einen String
* mit römischen Zahlen umgewandelt.
* Der String steht in ROEM , beim letzten
* Zeichen ist Bit 8 gesetzt (Ausgabe mit .PRINT).
*
* Es werden lediglich die Register AF zerstört.
*
*-----------------------------------------------
$ROEME  PUSH    BC
        PUSH    HL
        PUSH    DE
        LD      B,A             Wert in A
        LD      HL,ROEM         ab hier String
        LD      DE,6443H        100 in D,'C'in E
        CALL    ROEM1
        LD      DE,5A43H        90 in D, 'C' in E
        LD      A,58H           und 'X' in A
        CALL    ROEM2
        LD      DE,324CH        50 in D, 'L' in E
        CALL    ROEM1
        LD      DE,284CH        40 in D, 'L' in E
        LD      A,58H           und 'X' in A
        CALL    ROEM2
        LD      DE,0A58H        10 in D, 'X' in E
        CALL    ROEM1
        LD      DE,0958H        9 in D, 'X' in E
        LD      A,49H           und 'I' in A
        CALL    ROEM2
        LD      DE,055          5 in D, 'V' in E
        CALL    ROEM1
        LD      DE,0456H        4 in D, 'V' in E
        LD      A,49H           und 'I' in A
        CALL    ROEM2
        LD      DE,0149H        1 in D, 'I' in E
        CALL    ROEM1
*
        DEC     HL
        SET     7,(HL)          Stringende mark.
        POP     DE              fertig
        POP     HL
        POP     BC
        RET
*
ROEM1   LD      A,B             Wert wieder in A
        SUB     D               stimmt Ziffer?
        JR      C,ROEM11        wenn klein.-> ROEM11
        LD      B,A
        LD      (HL),E          röm.Zeichen in ROEM
```

```
        INC     HL
        JR      ROEM1
ROEM11  CCF                     komplem. Carry
        RET
*
ROEM2   PUSH    AF              rette röm. Zeichen
ROEM21  LD      A,B             hole Wert
        SUB     D               vergleiche
        JR      C,ROEM22        wenn klein.-> ROEM22
        LD      B,A
        POP     AF              hole 1.röm. Zeichen
        LD      (HL),A          in ROEM
        INC     HL
        LD      (HL),E          2. röm. Zeichen in ROEM
        INC     HL
        RET
ROEM22  POP     AF
        RET
*
ROEM    DS      8               hiert steht Ergebnis
```

# FORTH für den NASCOM

## Teil 7 von Günter Kreidl

FORTH für den NASCOM - Abschluß?

Das Fragezeichen in der Überschrift soll die Hoffnung ausdrücken, daß die anderen NASCOM-Anwender, die an dem in dieser Artikelserie beschriebenen Fädelcode-Interpreter arbeiten, (z.B. an einer Anpassung an das Standard-FORTH) oder gar eine eigene Implementation entwickeln, das Ergebnis ihrer Arbeit den Lesern des Journals zur Verfügung stellen. Die Artikelserie wird aber mit diesem Beitrag beendet.

Allen Versprechungen zum Trotz werde ich nochmals eine Erweiterung des Systems beschreiben, die erst in den neueren FORTH-Versionen eingeführt wurde und die Fähigkeiten der Sprache erheblich verbessert. Die Funktionen "(BUILDS" und "DOES)" ermöglichen die Erzeugung definierender Funktionen. (Diesen Satz bitte zweimal lesen!) Es lassen sich damit Klassen von beliebigen Datenstrukturen erzeugen. Einige Beispiele sollen dies erläutern:

```
: ARRAY   (BUILDS   ONE FOR ZERO CMPLW LOOP
          DOES)     SWAP DEC 2 * +      ;
```

Damit wird die Funktion "ARRAY" definiert, mit der wiederum "Arrays" definiert werden können:

```
20 ARRAY A1
```

erzeugt die eindimensionale Matrix A1 mit 20 Elementen. Mit

```
5 A1
```

erhalten wir die Adresse des 5. Wertes der Matrix A1 (mit PEEKW erhalten wir den Wert selbst). Mit

```
20000 7 A1 POKEW
```

Erteilen wir dem 7. Element der Matrix A1 den Wert 20000.

Wir können die Funktion "ARRAY" auch mit einer Überlaufkontrolle versehen. Dann muß sie folgendermaßen definiert werden:

```
: ARRAY   (BUILDS    DUP CMPLW ONE FOR
                     ZERO CMPLW LOOP
           DOES)     OVER OVER PEEKW )
                  IF  ERROR TYPE CLEAR
                ELSE  SWAP 2 * +     THEN ;
```

Wird eine Matrix z.B. mit 20 Elementen definiert, dann erfolgt bei Eingabe eines größeren Index eine Fehlermeldung.

Auf ähnliche Weise lassen sich mehrdimensionale Matrizen, BCD- oder Stringvariablen oder "Records" (gemischte Datentypen) definieren. Allgemein hat jede solche Definition die Form:

```
: (Name)   (BUILDS (Code1)  DOES) (Code2) ;
```

Erklären kann man das vielleicht am besten anhand der Beispielfunktion "ARRAY": Wird die Matrix A1 definiert, so wird (Code1) ausgeführt, wird die Matrix A1 selbst aufgerufen, wird (Code2) ausgeführt. Dabei muß berücksichtigt werden, daß auch die Funktion "DOES)" einen Beitrag leistet: sie gibt stets die Startadresse des Datensatzes auf den Stack. Konkret bedeutet dies: der auf "(BUILDS" folgende Code in der (ersten) Definition von "ARRAY" erzeugt in der Definition von A1 die 20 freien Speicherplätze; "DOES)" und der darauf folgende Code berechnen während des Aufrufs von A1 die Adresse des (im Beispiel) 5. Elementes. (Ich hoffe, daß jetzt keiner das NASCOM-Journal frustriert in die Ecke pfeffert - Ich hab das auch erst nach etlichen Anläufen kapiert!) Die Erklärung der Programmierung von "(BUILDS" und "DOES)" erspare ich mir; das ist wirklich nur etwas für Tüftler. Es folgt hier einfach der Code:

```
MCODE R)
1C78 7A 9E 0E 2B 56 2B 5E 22
1C80 9E 0E D5.
: SCODE   R) INC NAMES PEEKW FIRST +
          PEEKWPOKEW ;
: SYSTEM  CODEADR PEEKW DEC DEC
          MODIFY CODEADR POKEW  ;
```

```
: (BUILDS  ZERO CONSTANT ;                ,
: DOES)  R) INC NAMES PEEKW FIRST + PEEKW
        INC INC POKEW SCODE ;             ,
SYSTEM
1CCC EB 23 5E 23 56 23 E5 2A
1CD4 9C OE 1B EB 22 9C OE 2A
1CDC 9E OE 73 23 72 23 22 9E
1CE4 OE C3 3E 10.
```

Die Eingabe muß genau in der hier angegebenen Form erfolgen; nur die bei den Maschinencode-Routinen angegebenen Adressen werden je nach Ausbau des Systems unterschiedlich sein; sie werden jedoch von den Funktionen "MCODE" und "SYSTEM" erzeugt.

Programmbeispiele

Zunächst zwei Beispiele für rekursive Programmierung: (Argument1) (Argument2) GGT gibt den größten gemeinsamen Teiler der beiden Argumente auf den Stack.

```
: GGT       OVER OVER SWAP )
      IF    SWAP GGT
      ELSE  DUP EQZ
            IF    POP
            ELSE  SWAP OVER MOD GGT
            THEN
      THEN                                ;
```

Das zweite Beispiel ist die berühmte Funktion von Ackermann. Die beiden Argumente dürfen maximal die Werte (0 60), (1 60), (2 30), (3 3) und (4 0) annehmen, sonst gibt es einen Stacküberlauf.

```
: ACKERMANN  OVER EQZ
             IF     SWAP POP INC
             ELSE   DUP EQZ
              IF     POP DEC ONE ACKERMANN
              ELSE   OVER DEC ROT ROT DEC
                     ACKERMANN ACKERMANN
              THEN
             THEN                         ;
```

Als letztes Beispiel das 8-Damen-Problem, das ich eigentlich in dieser Form gar nicht veröffentlichen dürfte, denn es ist schauderhaft herunterprogrammiert. Interessant ist aber der Zeitvergleich mit anderen Sprachen.

```
: ABS DUP 32767 ) IF MINUS THEN ;
: CMPLB CODEADR PEEKW SWAP OVER POKEB
        INC CODEADR POKEW ;     ,
: NBYTES  (BUILDS ONE FOR ZERO CMPLB LOOP
          DOES) + DEC ;    ,
ZERO VARIABLE INDEX ,
8 NBYTES SPALTEN ,
0 VARIABLE BFLAG ,
: BEDROHT?  ONE BFLAG POKEW INDEX PEEKW
            DUP DEC ONE FOR
                DUP SPALTEN PEEKB I SPALTEN PEEKB
                OVER OVER EQ IF ZERO BFLAG POKEW THEN
                - ABS OVER I - EQ IF ZERO BFLAG POKEW THEN
            LOOP   POP BFLAG PEEKW  ;     ,
: LOESUNG   1 SPALTEN 8 ONE FOR
            DUP PEEKB = SPACE INC LOOP CR POP ;
: 8DAMEN  ONE INDEX POKEW
          ZERO ONE SPALTEN POKEB
    REPEAT INDEX PEEKW EQZ UNTIL
     REPEAT INDEX PEEKW SPALTEN DUP PEEKB
            INC DUP ROT POKEB BEDROHT?
            SWAP 8 ) OR UNTIL LOOP
     INDEX PEEKW SPALTEN PEEKB 8 (=
     IF  INDEX PEEKW 7 (=
  IF INDEX PEEKW INC DUP INDEX POKEW SPALTEN
   ZERO SWAP POKEB ELSE LOESUNG NEGONE INDEX
      MEM+ THEN ELSE NEGONE INDEX MEM+ THEN
    LOOP ;
```

Implementierungshinweise

Das 8-Damen-Programm läuft ca. 6 Minuten - immer noch erheblich schneller als Basic, aber doch langsamer, als ich es erwartet hatte. Um den Grund herauszufinden, habe ich die Funktion ABS in Maschinensprache programmiert. Das Programm läuft dann bereits 20-30 Sekunden schneller. Damit ist auch klar, weshalb das System relativ langsam ist: ca. 80% des Interpreters (die Erweiterungen eingerechnet) ist im Fädelcode geschrieben. Würde man den gesamten "Kern" des Systems in Maschinencode schreiben, würde sich die Arbeitsgeschwindigkeit vervielfachen. Den gleichen Hinweis gibt auch R. Loeliger in seinem Buch "THREADED INTERPRETIVE LANGUAGES", auf das ich zum Abschluß hinweisen möchte. In einer sehr klaren Sprache und Gedankenführung beschreibt der Autor den Aufbau eines Fädelcodeinterpreters für Z-80-Systeme. Sämtliche Routinen werden in Maschinencode und in Fädelcode gezeigt. Das ganze ist eine Art Baukastensystem, mit dem man sich seinen eigenen Interpreter zusammenstellen kann. Auch für Leute, die keinen Assembler besitzen, ist das Buch interessant, den es enthält auch alle Bausteine für einen konditionellen Assembler. Ärgerlich an dem schönen Buch ist eigentlich nur der Preis in Deutschland (DM 65,-)!

Wer sich für die Entstehungsgeschichte von FORTH interessiert, der sei auf den sehr schönen Artikel von Charles H. Moore, dem "Erfinder" der Sprache, in BYTE, August 1980, S.76 ff. verwiesen.

---

# DMA

**von Josef Zeller**

```
0080                    DMA     EQU     80H             ;I/O-ADRESSE DMA
000B                    INPUT   EQU     0BH             ;I/O-ADRESSE DES PERIPHERIE-BAUSTEINS, VON
                                                        ;DEM DATEN GELESEN WERDEN

                        ;
                        ;
                        ;*****************************************************************
                        ;*                                                               *
                        ;*                PROGRAMM 1                                     *
                        ;*                                                               *
                        ;*              PROGRAMM 'COPY' KOPIERT SPEICHERBEREICH          *
                        ;*              (MEMORY-MEMORY-TRANSFER)                         *
                        ;*****************************************************************
                        ;
0000'                   COPY:
0000'   E5                      PUSH    HL
0001'   C5                      PUSH    BC
0002'   0E 80                   LD      C,DMA           ;C <- DMA-ADRESSE
0004'   21 0011'               LD      HL,TAB1         ;HL <- ANFANG KOMMANDOBYTES
0007'   06 11                  LD      B,TAB1E-TAB1    ;B <- ANZAHL DER KOMMANDOBYTES
0009'   ED B3                  OTIR                    ;DMA INITIALISIEREN
000B'   CD 013D'               CALL    GLOBAL
000E'   C1                      POP     BC
000F'   E1                      POP     HL
0010'   C9                      RET
                        ;
                        ;
                        ;       KOMMANDOTABELLE 1
                        ;
0011'                   TAB1:
0011'   C3                      DEFB     0C3H           ;DMA-RESET
0012'   7D                      DEFB      7DH           ;WR0

0013'   D000            PORTA1: DEFW    0D000H          ;STARTADRESSE PORT A
0015'   01F3            BYTEC1: DEFW    01F3H           ;BLOCKLAENGE
0017'   AD                      DEFB     0ADH           ;WR4
0018'   4000            PORTB1: DEFW    4000H           ;STARTADRESSE PORT B
001A'   54                      DEFB      54H           ;WR1
001B'   CE                      DEFB     0CEH           ;PORT A TIMING
001C'   50                      DEFB      50H           ;WR2
001D'   CE                      DEFB     0CEH           ;PORT B TIMING
001E'   82                      DEFB      82H           ;WR5
001F'   CF                      DEFB     0CFH           ;LADE STARTADRESSEN
0020'   B3                      DEFB     0B3H           ;SETZE READY-SIGNAL
0021'   87                      DEFB      87H           ;ENABLE DMA
0022'                   TAB1E:
                        ;
                        ;
                        ;
                        ;*****************************************************************
                        ;*                    PROGRAMM 2                                 *
                        ;*                                                               *
                        ;*                    PROGRAMM 'SEARCH' SUCHT IM SPEICHER NACH   *
                        ;*                    EINEM BYTE. DIE ERGEBNISADRESSE WIRD DEM   *
                        ;*                    AUFRUFENDEN PROGRAMM IN HL UEBERGEBEN      *
                        ;*****************************************************************
                        ;
0022'                   SEARCH:
0022'   C5                      PUSH    BC
0023'   0E 80                   LD      C,DMA           ;C <- DMA-ADRESSE
0025'   21 0036'               LD      HL,TAB2         ;HL <- ANFANG KOMMANDOBYTES
0028'   06 0F                  LD      B,TAB2E-TAB2    ;B <- ANZAHL BYTES
002A'   ED B3                  OTIR                    ;PROGRAMMIERE DMA
002C'   CD 013D'               CALL    GLOBAL
002F'   2A 0294'               LD      HL,(RR3RR4)     ;HL <- ERGEBNISADRESSE
0032'   2B                      DEC     HL              ;DA PORT A COUNTER AUF UEBERNAECHSTE
0033'   2B                      DEC     HL              ;ADRESSE ZEIGT, MUSS ERGEBNISADRESSE UM
                                                        ; 2 VERMINDERT WERDEN
0034'   C1                      POP     BC
0035'   C9                      RET
                        ;
                        ;
                        ;       KOMMANDOTABELLE 2
                        ;
0036'                   TAB2:
0036'   C3                      DEFB     0C3H           ;DMA-RESET
0037'   7E                      DEFB      7EH           ;WR0
0038'   D000            PORTA2: DEFW    0D000H          ;STARTADRESSE
```

```
100:
101:
102:  003A'  FFFF              BYTEC2: DEFW   OFFFFH          ;SUCHLAENGE
103:  003C'  54                      DEFB     54H          ;WR1
104:  003D'  CE                      DEFB    0CEH          ;PORT A TIMING
105:  003E'  A1                      DEFB    0A1H          ;WR4
106:  003F'  BA                      DEFB     8AH          ;WR5
107:  0040'  CF                      DEFB    0CFH          ;LADE STARTADRESSEN
108:  0041'  83                      DEFB    0B3H          ;SETZE READYSIGNAL
109:  0042'  94                      DEFB     94H
110:  0043'  D8              MATCH2: DEFB    0D8H          ;MATCH-BYTE
111:  0044'  87                      DEFB     87H          ;ENABLE DMA
112:  0045'                  TAB2E:
113:                         ;
114:                         ;
115:                         ;
116:                         ;*************************************************************************
117:                         ;*              PROGRAMM 3                                               *
118:                         ;*                                                                       *
119:                         ;*              PROGRAMM 'TRASEA' TRNASPORTIERT DATEN                    *
120:                         ;*              BLOCKWEISE VON EINEM I/O-PORT IN DEN                     *
121:                         ;*              SPEICHER, BIS EOT(04H) EINGELESEN WIRD                   *
122:                         ;*              (MEMORY-I/O-TRANSFER)                                    *
123:                         ;*************************************************************************
124:                         ;
125:  0045'                  TRASEA:
126:  0045'  F5                      PUSH    AF
127:  0046'  E5                      PUSH    HL
128:  0047'  C5                      PUSH    BC
129:  0048'  F3                      DI
130:  0049'  21 029B'                LD      HL,INTVECT      ;HL <- ANFANG DER VEKTOR-TABELLE
131:  004C'  7C                      LD      A,H
132:  004D'  ED 47                   LD      I,A             ;I <- HIGH BYTE INTERRUPTVEKTOR
133:  004F'  7D                      LD      A,L
134:  0050'  32 0073'                LD      (LOWVEC),A      ;(LOWVEC) <- LOW BYTE INTERRUPTVEKTOR
135:  0053'  ED 5E                   IM      2
136:  0055'  0E B0                   LD      C,DMA           ;C <- DMA-ADRESSE
137:  0057'  21 0063'                LD      HL,TAB3         ;HL <- ANFANG KOMMANDOTABELLE
138:  005A'  06 17                   LD      B,TAB3E-TAB3    ; B <- ANZAHL BYTES
139:  005C'  ED B3                   OTIR                    ;PROGRAMMIERE DMA
140:  005E'  FB                      EI
141:  005F'  C1                      POP     BC
142:  0060'  E1                      POP     HL
143:  0061'  F1                      POP     AF
144:  0062'  C9                      RET
145:                         ;
146:                         ;
147:                         ;       KOMMANDOTABELLE 3
149:
150:
151:                         ;
152:  0063'                  TAB3:
153:  0063'  C3                      DEFB     0C3H          ;DMA-RESET
154:  0064'  7F                      DEFB      7FH          ;WR0
155:  0065'  000B            PORTA3: DEFW    INPUT          ;I/O-PORT-ADRESSE
156:  0067'  00FF            BYTEC3: DEFW    00FFH          ;BLOCKLAENGE
157:  0069'  3C                      DEFB      3CH          ;WR1
158:  006A'  50                      DEFB      50H          ;WR2
159:  006B'  FE                      DEFB     0FEH          ;PORT B TIMING
160:  006C'  94                      DEFB      94H          ;WR3
161:  006D'  04              MATCH3: DEFB      04H          ;MATCH BYTE (EOT=04H)
162:  006E'  9D                      DEFB      9DH          ;WR4
163:  006F'  4000            PORTB3: DEFW    4000H          ;STARTADRESSE PORT B
164:  0071'  3F                      DEFB      3FH          ;INTERRUPT-CONTROL-BYTE
165:  0072'  FF                      DEFB     0FFH          ;PULSE-CONTROL-BYTE
166:  0073'                  LOWVEC: DEFS      1            ;LOW BYTE INTERRUPT-VEKTOR
167:  0074'  9A                      DEFB      9AH          ;WR5
168:  0075'  C7                      DEFB     0C7H          ;PORT A TIMING (WIE CPU)
169:  0076'  CF                      DEFB     0CFH          ;LADE STARTADRESSEN
170:  0077'  BB                      DEFB      BBH          ;LOESCHE STATUS-BITS
171:  0078'  AB                      DEFB     0ABH          ;ENABLE INTERRUPTS
172:  0079'  87                      DEFB      87H          ;ENABLE DMA
173:  007A'                  TAB3E:
174:                         ;
175:                         ;               KOMMANDOTABELLE 4
176:                         ;
177:  007A'                  TAB4:
178:  007A'  BB                      DEFB      BBH          ;STATUS-BITS LOESCHEN
179:  007B'  D3                      DEFB     0D3H          ;LADE BLOCKLAENGE
180:  007C'  A3                      DEFB     0A3H          ;INTERRUPT ZURUECKSETZEN
181:  007D'  AB                      DEFB     0ABH          ;INTERRUPT-FREIGABE
182:  007E'  B7                      DEFB     0B7H          ;DMA-FREIGABE ERST NACH 'RETI
183:  007F'  87                      DEFB      87H          ;DMA-FREIGABE
184:  0080'                  TAB4E:
185:                         ;
186:                         ;
187:                         ;       INTERRUPT-ROUTINE BEI 'READY'=AKTIV
188:                         ;
189:  0080'                  IRDY:
190:  0080'  F3                      DI
191:  0081'  CD 0282'               CALL    STROUT
192:  0084'  20 49 4E 54            DEFM    ' INTERRUPT : READY AKTIV'
193:  0088'  45 52 52 55
194:  008C'  50 54 20 3A
195:  0090'  20 52 45 41
196:  0094'  44 59 20 41
```

```
                                                
0098'  4B 54 49 56                                      DEFB    0
009C'  00                                               CALL    CRLF
009D'  CD 0000*                                         EI
00A0'  FB                                               RETI
00A1'  ED 4D                                    ;
                                                ;
                                                ;
                                                ;       INTERRUPT-ROUTINE, WENN MARKE GEFUNDEN WURDE
                                                ;
00A3'                                           IMATCH:
00A3'  F3                                               DI
00A4'  CD 0282'                                         CALL    STROUT
00A7'  20 49 4E 54                                      DEFM    ' INTERRUPT : MATCH FOUND'
00AB'  45 52 52 55
00AF'  50 54 20 3A
00B3'  20 4D 41 54
00B7'  43 48 20 46
00BB'  4F 55 4E 44
00BF'  00                                               DEFB    0
00C0'  CD 0000*                                         CALL    CRLF
00C3'  CD 013D'                                         CALL    GLOBAL
00C6'  FB                                               EI
00C7'  ED 4D                                            RETI
                                                ;
                                                ;
                                                ;
                                                ;       INTERRUPT-ROUTINE, WENN BLOCK GANZ UEBERTRAGEN WURDE
                                                ;
00C9'                                           IENDBL:
00C9'  F3                                               DI
00CA'  E5                                               PUSH    HL
00CB'  C5                                               PUSH    BC
00CC'  CD 0282'                                         CALL    STROUT
00CF'  20 49 4E 54                                      DEFM    ' INTERRUPT : END OF BLOCK'
00D3'  45 52 52 55
00D7'  50 54 20 3A
00DB'  20 45 4E 44
00DF'  20 4F 46 20
00E3'  42 4C 4F 43
00E7'  4B
00EB'  00                                               DEFB    0
00E9'  CD 0000*                                         CALL    CRLF
00EC'  0E 80                                            LD      C,DMA
00EE'  06 06                                            LD      B,TAB4E-TAB4
00F0'  21 007A'                                         LD      HL,TAB4
                                                
00F3'  ED B3                                            OTIR
00F5'  C1                                               POP     BC
00F6'  E1                                               POP     HL
00F7'  FB                                               EI
00F8'  ED 4D                                            RETI
                                                ;
                                                ;
                                                ;
                                                ;       INTERRUPT-ROUTINE, WENN LETZTES UEBERTRAGENES BYTE
                                                ;       GLEICHZEITIG MATCH-BYTE WAR
                                                ;
00FA'                                           MATEND:
00FA'  F3                                               DI
00FB'  CD 0282'                                         CALL    STROUT
00FE'  20 49 4E 54                                      DEFM    ' INTERRUPT : END OF BLOCK AND MATCH FOUND'
0102'  45 52 52 55
0106'  50 54 20 3A
010A'  20 45 4E 44
010E'  20 4F 46 20
0112'  42 4C 4F 43
0116'  4B 20 41 4E
011A'  44 20 4D 41
011E'  54 43 48 20
0122'  46 4F 55 4E
0126'  44
0127'  00                                               DEFB    0
0128'  CD 0000*                                         CALL    CRLF
012B'  FB                                               EI
012C'  ED 4D                                            RETI
                                                ;
                                                ;
                                                ;
                                                ;***************************************************************************
                                                ;*                      PROGRAMM 4                                         *
                                                ;*                                                                         *
                                                ;*                      PROGRAMM 'STATUS' LIEST DMA-STATUS                 *
                                                ;***************************************************************************
                                                ;
012E'                                           STATUS:
012E'  F5                                               PUSH    AF
012F'  3E BF                                            LD      A,0BFH          ;READ-STATUS-KOMMANDO
0131'  D3 80                                            OUT     (DMA),A         ;AN DMA AUSGEBEN
0133'  DB 80                                            IN      A,(DMA)         ;STATUS-BYTE EINLESEN
0135'  32 0291'                                         LD      (RRO),A         ;STATUS ABSPEICHERN
0138'  CD 015A'                                         CALL    PRINT           ;BILDSCHIRMAUSGABE DES STATUS IM KLARTEXT
```

```
297:
298:  013B'  F1                        POP     AF
299:  013C'  C9                        RET
300:                                ;
301:                                ;
302:                                ;
303:                                ;
304:                                ;
305:                                ;*************************************************************************
306:                                ;*              PROGRAMM 5                                               *
307:                                ;*                                                                       *
308:                                ;*              UNTERPROGRAMM 'GLOBAL' LIEST VOM DMA                     *
309:                                ;*              STATUSBYTE, BLOCKLAENGEN- UND ADRESSZAEHLER              *
310:                                ;*              DER DMA-PORTS IN DEN PUFFERSPEICHER                      *
311:                                ;*************************************************************************
312:                                ;
313:  013D'                         GLOBAL:
314:  013D'  E5                             PUSH    HL
315:  013E'  C5                             PUSH    BC
316:  013F'  0E 80                          LD      C,DMA           ;C <- DMA-ADRESSE
317:  0141'  06 A7                          LD      B,0A7H          ;KOMMANDO LEITET READ-SEQUENZ EIN
318:  0143'  ED 41                          OUT     (C),B
319:  0145'  06 BB                          LD      B,0BBH          ;KOMMANDO 'READ-MASKE FOLGT'
320:  0147'  ED 41                          OUT     (C),B
321:  0149'  06 7F                          LD      B,7FH           ;READ-MASKE: ALLE LESBAREN REGISTER
322:  014B'  ED 41                          OUT     (C),B           ;WERDEN AUSGELESEN
323:  014D'  06 07                          LD      B,7             ;B <- REGISTER-ZAEHLER
324:  014F'  21 0291'                       LD      HL,RR0          ;HL <- ANFANG PUFFER-SPEICHER
325:  0152'  ED B2                          INIR                    ;REGISTER LESEN
326:  0154'  CD 020A'                       CALL    LIST
327:  0157'  C1                            POP     BC
328:  0158'  E1                            POP     HL
329:  0159'  C9                            RET
330:                                ;
331:                                ;
332:                                ;
333:                                ;
334:                                ;
335:                                ;
336:                                ;*************************************************************************
337:                                ;*              PROGRAMM 6                                               *
338:                                ;*                                                                       *
339:                                ;*              UNTERPROGRAMM 'PRINT' GIBT DEN DMA-STATUS                *
340:                                ;*              IM KLARTEXT AN DEN BILDSCHIRM AUS                        *
341:                                ;*************************************************************************
342:                                ;
343:  015A'                         PRINT:
345:
346:
347:  015A'  F5                             PUSH    AF
348:  015B'  CD 0000*                       CALL    CRLF
349:  015E'  CD 0282'                       CALL    STROUT
350:  0161'  20 2A 20 2A                    DEFM    ' * * *   DMA-STATUS   * * *'
351:  0165'  20 2A 20 20
352:  0169'  20 44 4D 41
353:  016D'  2D 53 54 41
354:  0171'  54 55 53 20
355:  0175'  20 20 2A 20
356:  0179'  2A 20 2A
357:  017C'  00                            DEFB    0
358:  017D'  CD 0000*                       CALL    CRLF
359:  0180'  3A 0291'                      LD      A,(RR0)
360:  0183'  1F                            RRA                     ;CARRY <- BIT0
361:  0184'  30 21                         JR      NC,PRINT1       ;DMA NICHT AKTIV
362:  0186'  CD 0282'                      CALL    STROUT
363:  0189'  20 44 4D 41                   DEFM    ' DMA OPERATION HAS OCCURED'
364:  018D'  20 4F 50 45
365:  0191'  52 41 54 49
366:  0195'  4F 4E 20 48
367:  0199'  41 53 20 4F
368:  019D'  43 43 55 52
369:  01A1'  45 44
370:  01A3'  00                            DEFB    0
371:  01A4'  CD 0000*                      CALL    CRLF
372:  01A7'                         PRINT1:
373:  01A7'  1F                            RRA
374:  01A8'  38 14                         JR      C,PRINT2        ;READY NICHT AKTIV
375:  01AA'  CD 0282'                      CALL    STROUT
376:  01AD'  20 52 45 41                   DEFM    ' READY ACTIVE'
377:  01B1'  44 59 20 41
378:  01B5'  43 54 49 56
379:  01B9'  45
380:  01BA'  00                            DEFB    0
381:  01BB'  CD 0000*                      CALL    CRLF
382:  01BE'                         PRINT2:
383:  01BE'  1F                            RRA
384:  01BF'  1F                            RRA
385:  01C0'  38 19                         JR      C,PRINT3        ;KEIN INTERRUPT ANGEFORDERT
386:  01C2'  CD 0282'                      CALL    STROUT
387:  01C5'  20 49 4E 54                   DEFM    ' INTERRUPT PENDING'
388:  01C9'  45 52 52 55
389:  01CD'  50 54 20 50
390:  01D1'  45 4E 44 49
391:  01D5'  4E 47
392:  01D7'  00                            DEFB    0
```

```
                        CALL    CRLF
                PRINT3:
                        RRA
                        JR      C,PRINT4        ;SUCHE ERFOLGLOS
                        CALL    STROUT
                        DEFM    ' MATCH FOUND'
                        DEFB    0
                        CALL    CRLF
                PRINT4:
                        RRA
                        JR      C,PRINT5        ;BLOCKENDE NOCH NICHT ERREICHT
                        CALL    STROUT
                        DEFM    ' END OF BLOCK'
                        DEFB    0
                        CALL    CRLF
                PRINT5:
                        POP     AF
                        RET
                ;
                ;
                ;
                ;
                ;*****************************************************************************
                ;*              PROGRAMM 7                                                   *
                ;*                                                                           *
                ;*              PROGRAMM 'LIST' GIBT STATUS UND DIE AKTUELLEN                *
                ;*              WERTE VON BYTE- UND ADRESS-COUNTER ZUM BILDSCHIRM AUS        *
                ;*****************************************************************************
                ;
020A'           LIST:
020A'   E5              PUSH    HL
020B'   CD 015A'        CALL    PRINT           ;STATUS AUSGEBEN
020E'   CD 0282'        CALL    STROUT
0211'   20 42 59 54     DEFM    ' BYTE-COUNTER VALUE    : '
0215'   45 2D 43 4F
0219'   55 4E 54 45
021D'   52 20 56 41
0221'   4C 55 45 20
0225'   20 20 20 3A
0229'   20
022A'   00              DEFB    0
022B'   2A 0292'        LD      HL,(RR1RR2)     ;HL <- BYTE-COUNTER
022E'   CD 0000*        CALL    ADROUT          ;WERT AUSGEBEN
0231'   CD 0000*        CALL    CRLF
0234'   CD 0282'        CALL    STROUT
0237'   20 50 4F 52     DEFM    ' PORT A ADRESS-COUNTER : '
023B'   54 20 41 20
023F'   41 44 52 45
0243'   53 53 2D 43
0247'   4F 55 4E 54
024B'   45 52 20 3A
024F'   20
0250'   00              DEFB    0
0251'   2A 0294'        LD      HL,(RR3RR4)     ;HL <- PORT A ADRESS-COUNTER
0254'   CD 0000*        CALL    ADROUT
0257'   CD 0000*        CALL    CRLF
025A'   CD 0282'        CALL    STROUT
025D'   20 50 4F 52     DEFM    ' PORT B ADRESS-COUNTER : '
0261'   54 20 42 20
0265'   41 44 52 45
0269'   53 53 2D 43
026D'   4F 55 4E 54
0271'   45 52 20 3A
0275'   20
0276'   00              DEFB    0
0277'   2A 0296'        LD      HL,(RR5RR6)     ;HL <- PORT B ADRESS-COUNTER
027A'   CD 0000*        CALL    ADROUT
027D'   CD 0000*        CALL    CRLF
0280'   E1              POP     HL
0281'   C9              RET
                ;
                ;
                ;
                ;
                ;*****************************************************************************
                ;*              PROGRAMM 8                                                   *
                ;*                                                                           *
                ;*              PROGRAMM 'STROUT' GIBT DIE NACH CALL STROUT FOLGENDEN        *
                ;*              STRINGS AN DEN BILDSCHIRM AUS, WOBEI EINE AUF DEN            *
                ;*              STRING FOLGENDE 0 ALS STRINGENDE INTERPRETIERT WIRD          *
                ;*****************************************************************************
                ;
0282'           STROUT:
0282'   E3              EX      (SP),HL         ;HL <- STRINGANFANG
0283'   F5              PUSH    AF
0284'           STR1:
```

```
493:
494:    0284'   7E                              LD      A,(HL)
495:    0285'   23                              INC     HL              ;INCREMENT POINTER
496:    0286'   B7                              OR      A               ;STRINGENDE ? (A=0?)
497:    0287'   28 05                           JR      Z,STR2          ;BEENDE AUSGABE
498:    0289'   CD 0000*                        CALL    CHROUT          ;INHALT VON A AN BILDSCHIRM AUSGEBEN
499:    028C'   18 F6                           JR      STR1            ;CONTINUE
500:    028E'                           STR2:
501:    028E'   F1                              POP     AF
502:    028F'   E3                              EX      (SP),HL         ;UPDATE STACK
503:    0290'   C9                              RET
504:                                    ;
505:                                    ;
506:                                    ;
507:                                    ;
508:                                    ;       PUFFERSPEICHER, IN DEM DIE READ-REGISTER
509:                                    ;       ZWISCHENGESPEICHERT WERDEN
510:                                    ;
511:    0291'                           RR0:    DEFS    1               ;RR0
512:    0292'                           RR1RR2: DEFS    2               ;RR1, RR2
513:    0294'                           RR3RR4: DEFS    2               ;RR3, RR4
514:    0296'                           RR5RR6: DEFS    2               ;RR5, RR6
515:                                    ;
516:                                    ;
517:                                    ;
518:                                    ;       INTERRUPT-TABELLE
519:                                    ;
520:    0298'                           INTVEC:
521:    0298'   0080'                           DEFW    IRDY
522:    029A'   00A3'                           DEFW    IMATCH
523:    029C'   00C9'                           DEFW    IENDBL
524:    029E'   00FA'                           DEFW    MATEND
525:
526:    02A0'   00                              NOP
527:                                            END
528:            MACRO-80 3.36   17-Mar-80       PAGE    8
529:
530:
531:  Macros:
532:
533:  Symbols:
534:  ADROUT  0278*   BYTEC1  0015'   BYTEC2  003A'   BYTEC3  0067'
535:  CHROUT  028A*   COPY    0000'   CRLF    027E*   DMA     0080
536:  GLOBAL  013D'   IENDBL  00C9'   IMATCH  00A3'   INPUT   000B
537:  INTVEC  0298'   IRDY    0080'   LIST    020A'   LOWVEC  0073'
538:  MATCH2  0043'   MATCH3  006D'   MATEND  00FA'   PORTA1  0013'
539:  PORTA2  003B'   PORTA3  0065'   PORTB1  001B'   PORTB3  006F'
540:  PRINT   015A'   PRINT1  01A7'   PRINT2  01BE'   PRINT3  01DB'
541:  PRINT4  01F1'   PRINT5  0208'   RR0     0291'   RR1RR2  0292'
542:  RR3RR4  0294'   RR5RR6  0296'   SEARCH  0022'   STATUS  012E'
543:  STR1    0284'   STR2    028E'   STROUT  0282'   TAB1    0011'
544:  TAB1E   0022'   TAB2    0036'   TAB2E   0045'   TAB3    0063'
545:  TAB3E   007A'   TAB4    007A'   TAB4E   0080'   TRABEA  0045'
546:
547:
548:
549:  No  Fatal error(s)
550:
551:
```

**ZUR BEACHTUNG**
**Aus drucktechnischen Gründen kann der Begleitartikel zu Josef Zellers Programmbeispielen erst im nächsten Heft abgedruckt werden.**

# Formulierungsautomat

## von Christian Peter

```
10 DIMA$(2,9),R(2)
30 A$(0,0)="konzertierte "
40 A$(0,1)="integrierte "
50 A$(0,2)="permanente "
60 A$(0,3)="systematisierte "
70 A$(0,4)="progressive "
80 A$(0,5)="funktionelle "
90 A$(0,6)="orientierte "
100 A$(0,7)="synchrone "
110 A$(0,8)="qualifizierte "
120 A$(0,9)="ambivalente "
130 A$(1,0)="Fuehrungs"
140 A$(1,1)="Organisations"
150 A$(1,2)="Identifikations"
160 A$(1,3)="Drittgenerations"
170 A$(1,4)="Koalitions"
180 A$(1,5)="Fluktuations"
190 A$(1,6)="Uebergangs"
200 A$(1,7)="Wachstums"
210 A$(1,8)="Aktions"
220 A$(1,9)="Interpretations"
230 A$(2,0)="struktur"
240 A$(2,1)="flexibilitaet"
250 A$(2,2)="ebene"
260 A$(2,3)="tendenz"
270 A$(2,4)="programmierung"
280 A$(2,5)="konzeption"
290 A$(2,6)="phase"
300 A$(2,7)="potenz"
310 A$(2,8)="problematik"
320 A$(2,9)="kontingenz"
324 GOSUB 380
325 CLS:FORJ=1TO10
330 FORI=0TO2:R(I)=INT(RND(I+1)*10):NEXT
340 PRINTTAB(3);A$(0,R(0));A$(1,R(1));A$(2,R(2)
)
350 NEXTJ
360 PRINT:PRINT:PRINT:INPUT"Weiter";N$:IFN$="JA
"THEN 325
370 END
380 PRINTTAB(5);"**** FORMULIERUNGSAUTOMAT ****
390 PRINT:PRINTTAB(3);"Dieser super-hyper Formu
lierungs-
400 PRINTTAB(3);"automat erspart Ihnen das laes
tige
410 PRINTTAB(3);"Formulieren von nichtssagenden
,
420 PRINTTAB(3);"aber toll klingenden (eindruck
-
430 PRINTTAB(3);"schindenden) Redewendungen. Si
e
440 PRINTTAB(3);"brauchen nur mehr den Knopf zu
450 PRINTTAB(3);"druecken - und haben noch dazu
 die
460 PRINTTAB(3);"Ausrede: DER COMPUTER WAR'S!
470 PRINT:PRINT:INPUTN$:RETURN
```

# Zwei praktische BASIC-Programme

## von Wolfgang v. Jan

```
10 CLS:REM 20.5.82 v.Jan, [redacted] Langenhagen
20 REM [redacted], Tel [redacted]
30 PRINT"Entfernungs- und Winkelprogramm"
40 PRINT"zwischen 2 Orten, z.B. Empfangsort zu
50 PRINT"den Sendern. Die geograf. Koordinaten
60 PRINT"muessen bekannt sein!
70 PRINT
80 PRINT"Koordinaten in Grad, Minuten eingeben!
90 PRINT"                    = "
100 PRINT"Dabei NULLEN hinter Komma miteingeben!"
110 PI=3.141593:B=PI/180:PRINT
120 A$="Langenhagen "
130 H=9:I=44.47:REM GEOGR. LAENGE F. LANGENHAGEN
140 J=52:K=26.16:REM GEOGR. BREITE F.LANGENHAGEN
150 PRINT"Wenn Emfaengerort "A$" ,"
160 PRINT"jewells nur ENTER eingeben!"
170 PRINT
180 PRINT"Eingabe Empfangsort";
190 INPUT A$
200 PRINT"Geogr. Laenge ";
210 INPUT H,I
220 C=H+I/60
230 LE=C*B
240 PRINT"Geogr. Breite ";
250 INPUT J,K:PRINT
260 C=J+K/60
270 PE=C*B
280 PRINT:PRINT "Sendeort";
290 RESTORE
300 B$=""
310 INPUT B$
320 IF B$="" THEN280
330 PRINT"Geogr. Breite  ";
340 INPUT L,M
350 C=L+M/60
360 LS=C*B
370 PRINT"Geogr. Laenge ";
380 INPUT N,O
390 C=N+O/60
400 PS=C*B
410 REM G-BERECHNUNG - - - - - -
420 DP=LE-LS
430 CG=SIN(PS)*SIN(PE)+COS(PS)*COS(PE)*COS(DP)
440 GOSUB840
450 WB=G*57.2958
460 REM ENTFERNUNG EMPFAENGER-SENDER   - - - -
470 E=2*PI*6371*WB/360
480 E=INT(E*10+.5)/10
490 REM WINKEL SENDER GEGEN NORD       - - - -
500 AA=SIN(90*B-PE)*SIN(DP)/SIN(G)
510 REM ARC SIN AA=WS
520 IF AA=(-1) THEN AA=1
530 IF ABS(AA)=1 THEN560
540 WS=ABS(ATN(AA/SQR(1-AA↑2))/B)
550 GOTO580
560 WS=AA*1.570796/B
570 GOTO580
580 REM DATENAUSGABE - - - - - -
590 WS=INT(WS*10+.5)/10
600 CLS
610 PRINTA$" "H;I" ; "J;K
620 PRINTB$" "L;M" ; "N;O"
630 PRINT"Entfernung: "E" km"
640 GOSUB1030:REM 4 QU-WINKEL
650 PRINT"Richtung von "A$" : ";
660 GOSUB940:REM HIMMELSRICHTUNG
670 PRINT"Nord zu "B$ " : "WS" Grad
680 PRINT"Sendeleistung in kW / Feldstaerke in mV/m
690 ZE=7:SP=1:WE=15:REM FUER TABELLENAUSGABE
700 FOR Z=1 TO 24
710 READ SL
720 GOSUB1230:REM UPRO TABELLENAUSGABE - - -
730 REM EE=SENDERFELDSTAERKE AM EMPFANGSORT
740 EE=222*SQR(SL)/E
750 EE=INT(EE+.5) :REM GANZE ZAHL!
760 PRINT SL"/"EE"   "
770 NEXT Z
780 REM SENDERLEISTUNGSTABELLE
790 DATA .1,.2,.5,1,2,5,8,10,15,20,30,40,50,60
800 DATA 70,80,90,100,150,200,300,500,800,1000
810 INPUT "neuer Sendeort: ENTER ";Z
820 CLS
830 GOTO280
840 REM ARC COS G=G
850 IF ABS(CG)=1E-10 THEN910
860 IF CG=0 THEN890
870 G=ATN(SQR(1-CG↑2)/CG)
880 RETURN
890 G=PI+G
900 RETURN
910 G=PI/2
920 RETURN
930 REM HIMMELSRICHTUNG IN KLARTEXT
940 IF WS=22.5 THEN1140
950 IF WS=67.5 THEN1150
960 IF WS=112.5 THEN1160
970 IF WS=157.5 THEN1170
980 IF WS=202.5 THEN1180
990 IF WS=247.5 THEN1190
1000 IF WS=292.5 THEN1200
1010 IF WS=337.5 THEN1210
1020 IF WS=360 THEN1220
1030 REM WINKELAUFTEILUNG IN 4 QUADRANTEN
1040 IF LS>=LE GOTO1060:REM 1.+2.QU
1050 IF LS=LE GOTO1080:REM 3.+4.QU
1060 IF PS>=PE GOTO1100:REM 1.QU
1070 IF PS=PE GOTO1110:REM 2.QU
1080 IF PS=PE GOTO1120:REM 3.QU
```

Obiges Programm dient der Bestimmung der Entfernung des Winkels zwischen zwei Orten. Es ist bei mir im Einsatz zur Bestimmung der Empfangswürdigkeitg verschiedener Sender. Hierzu benötigt man die geografischen Koordinaten beider Orte(ggf. Generalkarte) und z.B. das Verzeichnis der Ton-und Fernseh-Rundfunksendestellen der Bundespost.

Viele Foto-Objektive weisen eine Tiefenschaerfe-Skala auf. Bei Zoomobjektiven hat man meist darauf zu verzichten. Das folgende Programm erlaubt das Berechnen der Entfernung, die man einstellen muss, um von moeglichst nahe bis unendlich alles scharf eingestellt zu haben. Die am Objektiv moeglichen Blendengrenzen sind anzugeben, zusaetzlich der zugelassene Zerstreuungskreis (bei Kleinbild 1/30) und die Objektivbrennweite.

```
10 CLS
20 PRINT"Objektiv - Nah - Unendlichkeiseinstellung
30 PRINT
40 REM Lt. K.D.Solf, FOTOGRAFIE,Fischer Handb.
50 REM S. 166; 24.9.81, Vers. 20.5.81
60 REM v.Jan, [redacted], [redacted] Langenhagen
70 Z=30:REM FUER KLEINBILD
80 PRINT"Zerstreuungskreis-Durchmesser in 1/mm;
90 INPUT"wenn 1/30, nur ENTER   ";Z
100 Z=1/Z
110 PRINT
120 INPUT"Blende von, bis  ";AB,EB
130 PRINT
140 INPUT"Brennweite in mm  ";F
150 PRINT TAB(32)" bis unendlich
160 K=1
170 FOR I=1 TO 12
180 READ N:REM BLENDEN LESEN
190 IF N<AB THEN NEXT I
200 IF N>EB THEN240
210 N(K)=N
220 K=K+1
230 NEXT I
240 PRINT"Blende    einstellen auf(m)  scharf ab(m)
250 FOR I=1 TO K-1
260 EA=F*(F/N(I)/Z+1)/1000:REM EINSTELLEN AUF
270 ER=INT(EA*10+.5)/10
280 SA=EA/2:REM SCHARF AB
290 SA=INT(SA*10+.5)/10
300 PRINT TAB(1)N(I);TAB(9)ER;TAB(28)SA
310 NEXT I
320 INPUT" neue Brennweite, dann B ";I$
330 IF I$="B" THEN RESTORE:CLS:GOTO140
340 END
350 DATA 1,1.2,1.4,2,2.8,4,5.6,8,11,16,22,32,64
```

Im "Tiefenschärfe-Programm" ist ein Fehler in Zeile 90. Durch Drücken der Enter-Taste alleine wird Z=0, und dadurch ergibt sich ein /0 Fehler(Divide by Zero). Man müßte folgende Zeile einfügen:
99 IF Z=0 THEN Z=30
Dafür könnte natürlich Zeile 70 wegfallen.
Red.

# Softwaretracer

## von Christoph Rau

Der nascom bietet die Möglichkeit, zur Fehlersuche oder Analyse von Programmen die Befehle im Single Step abzuarbeiten. Dazu wird über Bit 3 von Port 0 softwaremäßig ein NMI erzeugt, der grundsätzlich ein Verzweigen nach 0066H bewirkt. Dort springt NAS-SYS nach 0C7DH in die Workspace, und an der Stelle ist ein Sprung auf die Single Step Routine eingetragen, die die Registerinhalte ausgibt und auf Eingabe wartet.
Oft reicht aber der Bildschirm für eine übersichtliche Analyse nicht aus, oder man weiß gar nicht, was für ein Befehl ausgeführt wurde. Vielleicht stört auch die Ausgabe der Register bei geforderten Eingaben, oder es sollen nur einzelne Befehlsfolgen analysiert werden.

Das folgende Programm generiert nach Aufruf zu jedem Befehl des Programms, das getraced werden soll, einen NMI, rettet den Programmstatus und ruft ein Unterprogramm auf, von dem abhängt, was beim Trace passieren soll. Bei mir wird die Ausgabe auf den Drucker umgelegt, die Registerinhalte des getraceden Programms werden ausgegeben, gefolgt von dem nächsten auszuführenden Befehl. Dazu benötigt das Programm den (reloziierten) Disassembler aus der mc 4/82 auf 1000H bis 153FH. Diese Ausgabe erfolgt aber nur für Befehle ab 1000H, damit der Aufruf von Monitor-Routinen zur Ein- und Ausgabe nicht getraced wird. Beliebige Verfeinerungen sind hier denkbar.

Das Programm hat die Einsprungadresse 1000H. Es kann als Hauptprogramm angesprungen werden, dann muß eine Hexadezimale als Adresse eingegeben werden, ab der getraced werden soll. Wird das Programm als Unterprogramm aufgerufen, so muß ein Zeichen >F eingegeben werden und das Programm beginnt zu tracen, wenn es ins rufende Programm zurückkehrt.
Um ROM-BASIC zu tracen, muß das ganze natürlich verschoben werden (z.B. nach 5000H). Dann ruft man das Programm am besten mit der USR-Funktion auf. ROM-BASIC legt übrigens bei FEC9H die NMI-Adresse in der Workspace um auf seine Break-Routine. Beim Tracen sollte dieser Befehl also umgangen werden.

```
0010 ; Trace
0020 ; (c) Christoph Rau, Bonn
0030 ; Ver 1.1
0040 ; 26.6.82
0050 OUTTAB EQU  #0C73
0060 NASTAB EQU  #077F
0070 UOUTJP EQU  #0C7B
0080 PRINT  EQU  #0E2C ; meine Routine
0090 NMIADR EQU  #0C7E
0100 OUTADR EQU  #1371
0110 ARG1   EQU  #0C0C
0120 DISASM EQU  #1401
0130 RBC    EQU  #0C61
0140 RHL    EQU  #0C65
0150 RPC    EQU  #0C69
0160 RSP    EQU  #0C6B
0170 SCAL   EQU  #18
0180 PRS    EQU  #28
0190 ROUT   EQU  #30
0200 INLIN  EQU  #63
0210 RLIN   EQU  #79
0220 B2HEX  EQU  #68
0230 SPACE  EQU  #69
0240 TBCD3  EQU  #66
0250 ;
0260        ORG  #1000
0270        JP   MAIN
0280        ORG  #1371
0290        RST  ROUT  ; Output Disassembler
0300        DEFW #0000 ; ueber nascom
0310 ;
0320 ; MAIN aktiviert den Trace.
0330 ; Man kann angeben, von welcher
0340 ; Adresse der
0350 ; Bei Eingabe eines ungueltigen Wertes
0360 ; wird die Return-Adresse vom Stack
0370 ; genommen (Aufruf von MAIN als Routine
0380 ;
0390 MAIN   ORG  #154B
0400        LD   HL,FLAG
0410        RES  0,(HL)
0420        LD   HL,PRINT
0430        LD   (UOUTJP),HL
0440        LD   HL,USTAB
0450        LD   (OUTTAB),HL ; nur User
0460        RST  PRS         ; Tabulierung
0470        DEFB #1B,"D,52,0 ; Drucker
0480        LD   HL,NASTAB
0490        LD   (OUTTAB),HL ; nur Video
0500        RST  PRS
0510        DEFM "Anfang"
0520        DEFB #0D,#00
0530        RST  SCAL
0540        DEFB INLIN
0550        RST  SCAL
0560        DEFB RLIN
0570        JR   C,RETURN
0580        LD   HL,(ARG1) ; neue Return-
0590        PUSH HL        ; Adresse
0600 RETURN LD   HL,NMI
0610        LD   (NMIADR),HL
0620 ACTNMI PUSH AF
0630        LD   A,8
0640        OUT  (0),A ; NMI aktivieren
0650        POP  AF
0660        RETN
0670 ;
0680 USTAB  DEFB #75,0
0690 FLAG   DEFB 0
0700 ;
0710 ; NMI wird bei jedem Step des Traces
0720 ; angesprungen. Die User-Register, der
0730 ; User-Stackpointer und die Return-
0740 ; Adresse werden gerettet und der
0750 ; Stackpointer umgesetzt.
0760 ;
0770 NMI    PUSH AF
0780        LD   A,0   ; NMI-FlipFlop
0790        OUT  (0),A ; zuruecksetzen
0800        PUSH HL
0810        PUSH DE
0820        PUSH BC
0830        LD   HL,00
0840        ADD  HL,SP
0850        LD   SP,RBC
0860        LD   DE,RBC
0870        LD   BC,8 ; retten der
0880        LDIR      ; User-Register
0890        LD   (RSP),HL
0900        LD   E,(HL)
0910        INC  HL
0920        LD   D,(HL)
0930        LD   (RPC),DE
0940        CALL NMIDO
0950        POP  BC  ; Stackpointer
0960        POP  DE  ; und
0970        POP  HL  ; User-Register
0980        POP  AF  ; restoren
0990        LD   HL,(RSP)
1000        LD   SP,HL
1010        LD   HL,(RHL)
1020        JR   ACTNMI
1030 ;
1040 ; NMIDO bestimmt, was bei jedem Schritt
1050 ; gemacht werden soll. Hier werden fuer
1060 ; alle Programmschritte, die ueberhalb
1070 ; 1000H liegen, die Register ausgegeben
1080 ; und der Disassembler aufgerufen.
1090 ; Die Ausgabe kommt nur auf die von
1100 ; der Benutzerfunktion bestimmten
1110 ; Geraete.
1120 ;
1130 NMIDO  LD   A,(RPC+1)
1140        AND  #80 ; kleiner 1000H?
1150        RET  Z
1160        LD   HL,USTAB
1170        LD   (OUTTAB),HL ; nur User
1180        LD   HL,#0C6D
1190        LD   B,6
1200 ER2    DEC  HL      ; Register ausgeben
1210        LD   A,(HL)
1220        RST  SCAL
1230        DEFB B2HEX
1240        DEC  HL
1250        LD   A,(HL)
1260        RST  SCAL
1270        DEFB B2HEX
1280        RST  SCAL
1290        DEFB SPACE
1300        DJNZ ER2
1310        LD   A,I
1320        RST  SCAL
1330        DEFB B2HEX
1340        RST  SCAL
1350        DEFB SPACE
1360        PUSH IX
1370        POP  HL
1380        RST  SCAL
1390        DEFB TBCD3
1400        PUSH IY
1410        POP  HL
1420        RST  SCAL
1430        DEFB TBCD3
1440        LD   A,(#0C67)
1450        LD   DE,#048B
1460        LD   B,8
1470 ER4    INC  DE    ; Flags ausgeben
1480        RLA
1490        PUSH AF
1500        LD   A,(DE)
1510        JR   NC,ER6
1520        RST  #30
1530 ER6    POP  AF
1540        DJNZ ER4
1550        LD   A,#09
1560        RST  #30
1570        LD   HL,(RPC)
1580        CALL DISASM
1590        LD   HL,NASTAB
1600        LD   (OUTTAB),HL
1610        RET
```

# UMLAUTE

## von Christian Peter

UMLAUTE UND ANDERE SONDERZEICHEN FÜR DEN NASCOM

Nachdem ich mir vor kurzem selbst die Mühe gemacht habe, herauszufinden, wie man ein 2716 EPROM als Zeichengenerator verwendet, möchte ich den anderen Lesern des Nascom Journals diese Mühe sparen und es ihnen in einem kleinen Artikel erklären:

DIE HARDWARE

Der Zeichengenerator des Nascom ist auch nichts weiter als ein read-only-memory (ROM). Nachdem dieses ROM auch noch Pin-kompatibel mit dem 2716 EPROM ist, stellt die Hardware kein Problem dar: Zeichengenerator-ROM gegen entsprechend programmiertes 2716 EPROM austauschen - fertig.

FUNKTION DES ZEICHENGENERATORS

Der Viedeoteil der Nascom Platine besteht unter anderem aus einer Anzahl von Zählern, die ständig vom CPU-Quarz getaktet werden. Die Ausgänge dieser Zähler sind so verschaltet, daß sie einerseits ständig die Synchron signale für das Videobild erzeugen und andererseits ständig den Bildschirmwiederholspeicher adressieren ('800 bis 'BFF für die CPU). Mittels Multiplexern haben sowohl die CPU als auch diese Zähler Zugriff auf diesen Speicherbereich (die CPU hat dabei Priorität).

Der Datenausgang der jeweils von den Zählern aktivierten Speicherzelle wird an die Adressleitungen A4 bis A10 des Generator-ROMs angelegt und bildet so eine "Grund-adresse" des adressierten Characters.

Nachdem das Fernsehbild aber zeilenweise abgetastet wird, muß der Buchstabe auch zeilenweise abgespeichert sein. Diese Zeilenabtastung des ROM-Inhalts erfolgt auch durch einen der oben erwähnten Zähler. Der Ausgang dieses Zählers ist die sogenannte "Row"-Adresse, die an die Adressleitungen A0 bis A3 des ROM angeschlossen wird.

Durch diese Art der Adressierung wird klar, daß der Nascom grundsätzlich 16 Bildschirm-zeilen braucht, um einen Character darzu-stellen, daß es also 16 Rows pro Buchstabe gibt. Nur beim Nascom 2 wird ein vorzeitiger Reset-Impuls für den Row-Zähler erzeugt, der bewirkt, daß nicht alle Rows abgefragt und auf dem Bildschirm dargestellt werden.

Die Bit-Information, die in der so adres-sierten Speicherzelle steht, wird in ein Schieberegister eingelesen und auf den Vi-deoausgang geschiftet, wobei für jedes "1" Bit ein heller Punkt entsteht.

DIE PROGRAMMIERUNG DES 2716

Durch die beschriebene Art der Adressierung des Charactergenerator-ROMs kommen also die niedrigsten 4 Bits der Adresse vom Row-Counter, die höherwertigen 7 Bits vom Bild-schirmspeicher (das 8. Bit dient ja bekanntlich zur Umschaltung auf Graohik).

Das bedeutet, daß die Adresse eines Charac-ters im ROM leicht zu finden ist: Die Row 1 des Buchstaben "A" liegt z.B. auf Adresse '410, die Row 16 auf '41F. Man braucht also den ASCII-Code nur um eine Hex-Stelle erwei-tern und hat die Adresse des Buchstaben im ROM. Außerdem wissen wir schon, daß jedes "1" Bit einen hellen Punkt erzeugt.

Will ich also selber einen Character ändern oder selbst erfinden, dann brauche ich nur ein 8 x 16 Raster aufzuzeichnen, für jeden Punkt, der dann am Bildschirm hell sein soll, ein X zu machen und dann jede Zeile in eine Bit-Information umzusetzen.

Bei allen Zeichen ist zu beachten, daß normalerweise ein Abstand zwischen zwei Zei-chen ist. Dieser Abstand wird einfach da-durch erzeugt, daß eine senkrechte Punktrei-he leergelassen wird. Beim Standard-Zeichen-generator ist diese leere Punktreihe immer links vom Zeichen.

ALS BEISPIEL: UMLAUTE

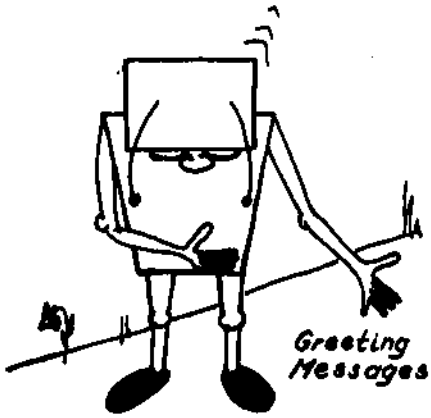
```
Ä              Bits    Hex  Adresse

12345678
1 X   X    01000001   41   5B0
2   XXX    00011100   1C   5B1
3  X   X   00100010   22   5B2
4 x     x  01000001   41   5B3
5 x     x  01000001   41   5B4
6 XXXXXXX  01111111   7F   5B5
7 X     X  01000001   41   5B6
8 X     X  01000001   41   5B7
9 X     X  01000001   41   5B8
0          00000000   00   5B9
1          00000000   00   5BA
2          00000000   00   5BB
3          00000000   00   5BC
4          00000000   00   5BD
5          00000000   00   5BE
6          00000000   00   5BF

Ö: Adresse 5C0
41,1C,22,41,41,41,41,22,1C,00 ... 00
Ü: Adresse 5D0
41,00,41,41,41,41,41,41,3E,00 ... 00
ä: Adresse 7B0
00,24,00,3C,02,3E,42,42,3D,00 ... 00
ö: Adresse 7C0
00,24,00,3C,42,42,42,42,3C,00 ... 00
Ü: Adresse 7D0
00,24,00,42,42,42,42,46,3A,00 ... 00
ß: Adresse 7E0
1C,22,22,24,28,24,22,22,24,20,00 ... 00
```

Weitere Artikel zu diesem Thema finden Sie im NASCOM Journal 10-81 S.7, 1-82 S.19 und 3/4-82 S.20 ff.

# nascompl




Hallo liebe Leser,

Ärgern Sie sich manchmal über die Abkürzungen der ERROR Messages bei BASIC oder gar die Zahlenkodierung bei den Fehlermeldungen im ZEAP Assembler? Sie sollten es nicht tun, denn dahinter steckt die Philosophie "so viel Information wie möglich auf äußerst geringem Speicherplatz". Diese Anschauung sollten wir uns vielmehr auch auf anderen Gebieten zu eigen machen.
Wer kennt nicht das Problem, daß man beim Briefschreiben ein neues Blatt hervorholen muß, nur um die Grußformel noch unterzubringen. Deshalb hier meine Idee der Greeting Message, deren Standard jeder ernsthafte Journal- Leser übernehmen sollte. Die Tabelle sollte noch erweitert werden. Schicken Sie bitte Ihre Vorschläge. Bisher sind folgende Codes festgelegt:
ØØ Hochachtungsvoll Ihr
Ø1 Mit untertänigster Hochachtung Ihr
Ø2 Dein(e) Dich liebende(r)
Ø3 In diesem Sinne Ihr
Ø4 Mit freundlichen Grüßen Ihr
Ø5 Mit freundlichen Grüßen Dein
Ø6 Rutsch mir den Buckel runter Dein
Ø7 Herzlichst Ihr
Ø8 Mit aufrichtiger Nichtswürdigung Euer
Ø9 Bis bald Dein
Eine Code- Tabelle für Anreden ist gerade in Arbeit und wird beizeiten veröffentlicht.
Ø3 NASCOMPL

# IMPRESSUM

REDAKTION: Günter Böhm, Günter Kreidl
Wolfgang Mayer-Gürr, Josef Zeller
RESSORTS:
MASCHINENPROGRAMME:
Günter Böhm, [redacted]
[redacted] Karlsruhe, Tel. [redacted]
Günter Kreidl, [redacted], [redacted] Straelen
Tel. [redacted]
BASIC und FLOPPY:
Wolfgang Mayer-Gürr, [redacted]
[redacted], [redacted] Recklinghausen
Tel. [redacted]
HARDWARE:
Josef Zeller, [redacted], [redacted] Neu-Ulm
VERLAG:NASCOM JOURNAL, c/o MK-Systemtechnik
Pater-Mayer-Str.6, 6728 Germersheim
Tel.Ø7274/2756 Telex 453500 mksd
VERTRIEB: Direktvertrieb durch den Verlag
Erscheinungsweise: monatlich
Bezugspreis: Im In- und Ausland 48.- für ein Jahresabonnement. Abonnements können aus technischen Gründen immer nur für die Dauer eines Kalenderjahres, d.h. vom 1.1. bis 31.12. laufen. Bei Bestellung nach dem 1.1. werden die fehlenden Hefte mit der ersten Lieferung bis zum Bestellzeitpunkt automatisch mitgeliefert. Bei nicht fristgerechter Kündigung verlängert sich das Abonnement automatisch um ein Jahr.Die Kündigung für das Folgejahr muß bis spätestens sechs Wochen vor Jahresende erfolgen.
Bezugsmöglichkeiten:Durch Bestellung bei MK Systemtechnik .
Bankverbindungen: Alle Zahlungen für das NASCOM JOURNAL unter Angabe der Rechnungsnummer an MK - Systemtechnik, Germersheim.
Zahlung: Nach Eingang Ihrer Bestellung erhalten Sie von uns die ausstehenden Hefte bis zur aktuellen Ausgabe sowie eine Rechnung. Bitte, zahlen Sie dann den Rechnungsbetrag .
Bitte keine Vorauszahlungen!
Bitte, Anfragen wegen Abonnements oder Lieferung nicht an die Redaktion sondern nur an den Verlag. Die Autoren tragen die Verantwortung für ihre Beiträge selbst. Für Fehler in Text, Bildern und sonstigen Angaben kann keine Haftung übernommen werden.

# nascom

## Die Alternative!

Kein »langweiliger Computer«
NASCOM 1 und NASCOM 2 sind Computer für Selbermacher, Tüftler, erfolgreiche Do-it-yourself-Freunde. NASCOM-Computer werden niemals langweilig! Die Systeme 1 und 2 sind keine fertigen »Kästen« ohne Erweiterungsmöglichkeit. Der hochwertige Platinensatz Computer und Keyboard kann so aufgebaut, erweitert und »verpackt« werden, wie **Sie** es wünschen.
Für Vollpreis gibt die NASCOMs auch als Bausatz. Aber aufgepaßt: Das ist eine Sache nur für wirkliche Könner! Und damit es auch nach dem Aufbauen nicht langweilig wird, gibt es das monatlich erscheinende NASCOM-JOURNAL. Eine Zeitschrift speziell für NASCOM-Freaks vollgestopft mit Hardware- und Software-Ideen, Kleinanzeigen, den neuesten Infos, und, und, und. . .
Die NASCOMs sind keine »Spielcomputer«. Mehr als 60 % aller NASCOM-Systeme werden als sogenannte »OEM-Baugruppen« von professionellen Anwendern in eigene Systeme eingebaut. Ingenieurbüros verwenden den NASCOM als Entwicklungssystem. Die Anwendungsmöglichkeiten sind mehr durch Ihre Phantasie begrenzt. Ein NASCOM-System kann fast alle gängigen Probleme lösen.

## Mit NASCOM wachsen!

NASCOM-Systeme sind aufwärtskompatibel. Das kleinste, preisgünstigste NASCOM 1-System kann bis auf NASCOM 3-Level mit Floppy-Laufwerken und CP/M* ausgebaut werden. Bildschirm-Ansteuerung, Tastatur Betriebsystem und Systemsoftware sind durchweg kompatibel. Ohne faule Kompromisse!
Mit NASCOM-Systemen gehen Sie kein Risiko ein. Ihr NASCOM wächst mit!

**NASCOM 1**
Spezifikationen:
- QWERTY-Tastatur, aufgebaut mit hochwertigen Magnettasten
- NAS-SYS Betriebsystem (2k Byte)
- 16 I/0-Leitungen
- Video (BAS) und TV-Ausgang
- 1k RAM, ausbaubar auf 192k RAM
- Display 48 Zeichen in 16 Zeilen

**ab DM 935,—**

**NASCOM 2**
Spezifikationen:
Wie NASCOM 1, jedoch zusätzl.:
- 8k Mikrosoft-BASIC u. 8k Stat. RAM
- Z80A-Mikroprozessor, 4 MHz
- Erweiterte Tastatur 57 Tasten
- Integrierte Bus-Pufferung
- Bis 192k Byte RAM
- Grafik-Möglichkeiten: 48 x 96 Punkte
- Serielle Schnittstelle; Baudrate wählbar, R5232C/20mA
- 16 parallele Ein/Ausgabeleitungen (Z80APIO)

**ab DM 1950,—**

## nascom 3 — der Profi

Spezifikationen: Wie NASCOM 2, jedoch zusätzl.: **ab DM 2735,—**
- 0.35 Megabyte pro 5,25-Zoll Laufwerk
- Betriebsystem CP/M*2.2 oder NAS-DOS
- Bildschirmausgabe erweiterbar auf 80 x 25 Zeichen

## Die dritte NASCOM-Generation

NASCOM 1 und 2 haben OEM-Board, Schulungscomputer, Kompaktrechner etc. ca. 20 000 mal ihren Partner gefunden. Der NASCOM 3 möchte Ihr persönlicher Computer werden! Er möchte Ihnen helfen, sich selbst fortzubilden, im Beruf weiter zu kommen, auch mal in die Computertechnik »rein zu riechen«. Ingenieurbüros und Softwareingenieuren dient der NASCOM 3 als preisgünstiges Entwicklungssystem.

## Universelle Betriebssoftware

Der NASCOM 3 kennt zwei Betriebssysteme: Das CP/M* (Version 2.2) — inzwischen Standard — und sein eigenes NAS-DOS. Die 5-Zoll Floppys bieten eine Speicherkapazität von 0.35 Megabyte pro Laufwerk (single sided, double density, double tracked). Damit wird das Spektrum universeller CP/M*-Software verfügbar!

**Wir informieren Sie unverbindlich:**
**Fordern Sie Ihr NASCOM-INFO-PAKET an!****

**Unsere Händler:**

Heinz Vogel Verlag GmbH & Co.
Lehrmittelzentrum, Herr Seloff
Innsbrucker Straße 96
2800 Bremen-Findorff
☎ (0421) 35 10 69

MK-SYSTEMTECHNIK
Michael von Keitz
Pfaffenberg 4
5650 Solingen
☎ (02122) 47267

Radio Zinburg
Herr Zinburg, Jr.
Röhrstraße 10
5760 Arnsberg
☎ (02932) 3 15 10

Christian Lampson
W.-Leuschner-Straße 4
6085 Nauheim
☎ (06152) 56730

MK-SYSTEMTECHNIK
Kriegsstraße 164
7500 Karlsruhe
☎ (0721) 29243

Graf Elektronik Systeme GmbH
Postfach 1610
8791 Kempten
☎ (0831) 61930

**Autorisierter Distributor:**

**MK-SYSTEMTECHNIK**
**Pater-Mayer-Straße 6**
**6728 Germersheim**
**☎ (07274) 2093**
**Telex 453500 mks d**

* CP/M* ist ein eingetragenes Warenzeichen der DIGITAL RESEARCH
** NASCOM-INFO-PAKET gegen DM 2,— in Briefmarken (wird bei Kauf angerechnet)