nascom journal

Zeitschrift für Anwender des NASCOM 1 oder NASCOM 2

2. Jahrgang · August 1981 · Ausgabe 8

Herausgeber:
MK-SYSTEMTECHNIK Michael Klein · Pater-Mayer-Straße 6 · 6728 Germersheim/Rhein
Telefon (0 72 74) 27 56 · Telex 0453500 mks d

MK-Systemtechnik Thomas Gräfenecker · Kriegsstraße 164 · 7500 Karlsruhe · Tel. 07 21 - 2 92 43
MK-Systemtechnik Michael von Keitz · Pfaffenberg 4 · 5650 Solingen 1 · Tel. 0 21 22 - 4 72 67

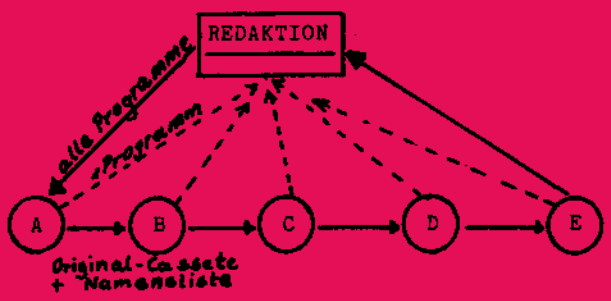
Der Heftpreis beträgt DM 4,—. Ein Abonnement erhalten Sie für DM 48,— im Jahr. Dafür bekommen Sie 12 Hefte pro Jahr, bzw. 10 Hefte (zwei dicke Doppelausgaben).
Die Autoren sind für den Inhalt ihrer Beiträge selbst verantwortlich.

Inhalt:

nascom journal

intern

Liebe Leser,
dies ist nun erst die dritte Ausgabe, die vom neuen Redaktionsteam erstellt wurde, doch schon kann man eine neue Tendenz feststellen: die Entwicklung zu einem Medium, das es den Lesern ermöglicht, sich gegenseitig in ihren Interessen zu unterstützen. Professionelle wissenschaftliche Grundlagen können Sie einer Unmenge anderer Zeitschriften entnehmen. Hard- und Software, die sich bei anderen NASCOM-Benutzern bewährt hat, Tips, die genau Ihr System betreffen, Antwort auf Ihre speziellen Fragen, an denen schon andere Leser geknobelt haben, gegenseitige Unterstützung der NASCOM Anwender, all das sind Bereiche, die Ihr Journal abdecken kann. (Wenn Sie sich daran beteiligen).
Der Floppy-Tausch von Herrn Mayer-Gürr ist angelaufen, der Kassettenservice von Herrn Kreidl hat wie eine Bombe eingeschlagen. Herr Maurer übernimmt vorläufig das Überspielen von NASCOM 2 auf NASCOM 1 Format, Herr Baier zaubert Kassettenprogramme auf Floppy usw. usw. Eine allgemeine Kommunikation unter den Lesern beginnt sich auszubreiten. Nur Herr Zeller wartet einsam auf weitere Hardwarevorschläge.
Inzwischen sind auch viele Themen vorgeschlagen worden, die im Journal erscheinen sollten. Im nächsten Heft werden wir eine Sammlung davon abdrucken. Vielleicht haben Sie schon ein Programm oder Informationen dazu in der Schublade.
Die Redaktionsarbeit ist uns durch die rege Leserbeteiligung etwas über den Kopf gewachsen. Deshalb bitten wir um folgende Änderung: Schicken Sie Programme bitte nur noch auf Kassette ein. (Vergessen Sie nicht, das Format darauf zu vermerken). Jeder Einsender erhält dafür eine Kassette mit einem Programm nach Wunsch aus dem letzten Heft (für solche, die nicht gerne eintippen) oder ein sauberes Listing (für solche, die keinen Drucker besitzen). Auch Texte sollten Sie auf Kassette einschicken. Näheres zum Textformat im Artikel "Formatierprogramm". Für Leute ohne Speichererweiterung steht das Programm "Bildschirm auf Kassette" zur Verfügung, um problemlos Manuskripte auf Kassette zu speichern. (Es ist für NASSYS Benutzer leicht umzuschreiben. Die entsprechenden Monitorroutinen sind unterstrichen. Das Programm ist voll relokatibel).
Inzwischen ist auch der Wunsch nach einer Kassettentauschzentrale laut geworden. Um die Arbeit zu minimieren, bieten wir folgende Möglichkeit an: Jeder Interessent schickt eine Kassette mit seinem Programm an die Redaktion. (Stichwort "Tauschzentrale"). Notwendige Texte, Erklärungen, Spielanleitungen etc. sollten vor dem Programm gespeichert sein. Bitte Ladeformat vermerken! Die Einsender werden in der Reihenfolge ihres Einsendedatums auf einer Liste eingetragen. Der erste Einsender erhält eine Kassette mit sämtlichen eingegangenen Programmen. Er kann sie kopieren und schickt das Original an den nächsten Namen der Liste, nachdem er seinen Namen gestrichen hat. Der letzte Einsender schickt das Original an die Redaktion zurück. Wir sind gespannt, ob dieses System funktioniert! (Jedenfalls müssen die Empfänger die Kassette möglichst schnell weiterschicken, sonst dauert es eine Ewigkeit).



Im letzten Heft wurde ein "Einheitsbus" für PIO angekündigt. Das muß aus zwei Gründen auf die nächste Ausgabe verschoben werden: einmal haben wir keinen Platz mehr in dieser Ausgabe, zweitens soll ein alternativer Bus von Herrn Öhring zur Auswahl gestellt werden. Auch ein Grund, sich schon auf's nächste Heft zu freuen - Viel Spaß mit NASCOM und Journal
Ihr Günter Böhm

FORTH für den NASCOM

von Günter Kreidl

Es soll hier nicht wieder, wie es leider schon häufiger im NASCOM-JOURNAL geschehen ist, auf mehreren Seiten ein Programm ausführlich beschrieben werden, das dann anschließend zum Kauf angeboten wird. Andererseits wird hier auch kein "betriebsfertiges" FORTH-System vorgelegt, das man nur noch einzutippen braucht. Vielmehr verstehe ich diesen und die folgenden Beiträge als Einladung an die Leser, gemeinsam ein solches FORTH-System zu entwickeln. Um eine Grundlage zu haben, wird in diesem Heft zunächst der prinzipielle Aufbau von FORTH erläutert. Im nächsten Heft erscheint dann das Assemblerlisting einer "Schmalspurversion" bzw. eines TINY FORTH, das wir dann gemeinsam ausbauen und verbessern wollen.

"Fädelcode"

Wer öfters größere Programme in Assembler oder Maschinensprache schreibt, hat sich vielleicht eine Bibliothek von Routinen zusammengestellt, die dann universell einsetzbar sind. Große Teile eines solchen Programms sehen dann oft so aus:

CALL Routine1
Registeranpassung
CALL Routine2
Registeranpassung usw.

Unter Registeranpassung wird hier alles verstanden, was mit dem Retten von Registerinhalten über die PUSH-Befehle und der Übergabe von Parametern zwischen den Unterprogrammen zusammenhängt. Wenn man nun statt der individuellen Registeranpassungen ein Standardsystem der Parameterübergabe einführt, vereinfacht sich das Programm erheblich. Nehmen wir einmal an, alle Parameter werden über den Stack übergeben. Da dann auch keine Registerinhalte mehr gerettet werden müssen, besteht das Programm nur noch aus Unterprogrammaufrufen:

CALL Routine1
CALL Routine2 usw.

Jetzt kann man sich aber auch noch den CALL-Befehl selbst sparen, wenn man stattdessen eine Routine einführt, die eine Reihe von aufeinander folgenden Adressen eben als Adressen von Unterprogrammen interpretiert. Wir brauchen also einen kleinen INTERPRETER, der diesen speziellen Code, der nur aus einer Aneinanderreihung von Adressen von Unterprogrammen besteht, abarbeiten kann. Dieser kleine Interpreter, der nicht mehr als ca. 5o Bytes Speicherplatz belegt, und dieser spezielle Code, den man im Englischen THREADED CODE nennt, was ich hier mit Fädelcode übersetzt habe, bilden das Kernstück von FORTH. Der Interpreter ist wesentlich kleiner als bei anderen "höheren" Programmiersprachen und wesentlich schneller (etwa 1o x so schnell wie ein Basic-Interpreter).

Bestandteile von FORTH

Woraus besteht also die Programmiersprache FORTH? Zunächst einmal aus einer Bibliothek von Routinen in Maschinensprache (sogenannten "Primitives"). Aus diesen Routinen sind dann weitere Routinen im Fädelcode zusammengesetzt. Beide Arten von Routinen können von dem oben beschriebenen Interpreter aufgerufen werden. Diesen THREADED-CODE-INTERPRETER wollen wir den internen Interpreter nennen, denn das System verfügt noch über einen anderen, den äußeren oder Benutzerinterpreter, der interaktiv die Eingaben des Anwenders (Text oder Zahlen) in Startadressen für den internen Interpreter umwandelt. Der dritte Bestandteil von FORTH ist ein Compiler, der es ermöglicht, der Sprache weitere Befehle als Threaded-Code -Routinen hinzuzufügen. Ein ausgebautes FORTH-System enthält darüber hinaus noch einen Editor und einen Assembler - und das alles in 5 - 6 K Speicherplatz.

Der interne Interpreter

Der interne Interpreter imitiert gewisse Funktionen der CPU. Dazu braucht er einen Programmzähler und einen eigenen Stack. Programmzähler und Stackpointer befinden sich im Speicher, da die entsprechenden CPU-Register natürlich anderweitig benutzt werden. Seine Funktion ist einfach: er holt die nächste Adresse (aus dem Fädelcode!) aus dem Speicher und führt einen Sprung dorthin aus. Diese Adresse ist entweder die Adresse einer Routine in Maschinensprache - diese wird dann ausgeführt und endet nicht mit einem Return-Befehl sondern mit einem:

JP NEXT (der Name des Interpreters)

- oder die Adresse einer Routine im Fädelcode. Eine solche muß stets mit einer

Routine mit dem Namen TCALL beginnen, die den CALL-Befehl ersetzt. Sie gibt den alten PC (des Interpreters) auf den Stack (des Interpreters!) und ersetzt ihn durch die jeweilige neue Adresse. Ebenso muß jede Routine im Fädelcode mit einem besonderen Rückkehrbefehl (TRET) abgeschlossen werden. Auch die Routinen im Maschinencode weisen noch eine Besonderheit auf: sie tragen ihre eigene Adresse vorweg, damit sie von NEXT aufgerufen werden können. Im Assemblertext sieht das etwa so aus:

```
PEEKB  DEFW   §+2       ;PEEKB holt einen
       POP    HL        ;Wert (Byte) aus
       LD     E,(HL)    ;dem Speicher
       LD     D,o
       PUSH   DE
       JP     NEXT

TSUB   DEFW   TCALL
       DEFW   MINUS
       DEFW   TADD
       DEFW   TRET
```

Die letztere, sehr einfache Routine im Fädelcode subtrahiert den obersten Stackwert vom zweitobersten und gibt das Ergebnis wieder auf den Stack. Es werden dabei die beiden Routinen MINUS (negiert den Stack) und TADD (addiert die beiden obersten Stackwerte) aufgerufen. Wie man sieht, wird der gesamte Fädelcode im Assembler mit dem Pseudo-Opcode DEFW geschrieben.

Der äußere Interpreter

Eingaben des Anwenders werden vom äußeren Interpreter als Zahlen (Umwandlung!) oder als Befehlsworte ausgewertet. Diese Worte sucht er in einem "Wörterbuch" und findet dort auch die Adresse der zugehörigen Routine. Er unterscheidet sich von den Interpretern anderer Programmiersprachen eigentlich nur durch den Eingabemodus; ebenso wie bei HP-Rechnern muß die Eingabe in der umgekehrten polnischen Notation erfolgen, d.h., es müssen zunächst die Argumente eingegeben werden, dann folgt erst die zugehörige Operation, z.B. 5 8 + und der Interpreter antwortet mit dem Ergebnis. Diese Notation ist bei FORTH zwar üblich, könnte aber ebensogut durch eine andere, z.B. die algebraische, ersetzt werden.

Der Compiler

Vielleicht das Erstaunlichste an FORTH ist, daß bereits in der hier vorgestellten Schmalspurversion ein Compiler enthalten ist, der die Sprache um vom Benutzer definierte Befehle erweitert. Diese Befehle werden aus den Befehlswörtern des Interpreters zusammengestellt mit Hilfe einiger zusätzlicher Routinen, die in einem eigenen "Wörterbuch" des Compilers verfügbar gemacht werden. Diese Routinen sind aber gerade besonders interessant, denn es handelt sich dabei um 3 verschiedene Arten der logischen Schleifenbildung:
IF-THEN-ELSE; REPEAT-LOOP (REPEAT-UNTIL); FOR-LOOP (FOR-NEXT)
Aus diesen Schleifenkonstruktionen und den Befehlsworten des Interpreters werden dann neue Befehle erzeugt, denen bei der Compilierung ein Name zugewiesen wird, der in das Befehlsverzeichnis des Interpreters eingetragen wird. Überhaupt besteht die gesamte Programmierung bei FORTH aus der Compilierung neuer Befehle, die dann stufenweise immer komplexer werden, bis zuletzt die gwünschte Funktion (das "Programm") im Interpreter vorhanden ist. So ergibt sich praktisch von selbst die "strukturierte Programmierung", von der heute soviel geredet wird.

Ein einfaches Beispiel einer Compilierung der Funktion SIGMA (= Summe von 1 bis n) soll den Abschluß dieser Einleitung bilden. Es werden 2 Routinen benutzt: + (vergl. oben TADD!) und SWAP (vertauscht auf dem Stack die beiden oberen Werte). Das ":" startet die Compilierung, das ";" beendet sie:

```
:SIGMA
    o SWAP
    1 FOR
       I +
      LOOP;
```

Die Funktion erwartet einen Wert auf dem Stack. Auf die Eingabe: 1o SIGMA, antwortet der Interpreter nun mit dem Wert: 55. Die Funktion "I" gibt übrigens den Wert der innersten Schleifenvariable auf den Stack. Die FOR-LOOP-Schleife wird so lange ausgeführt, bis der Eingabewert (auf dem Stack) erreicht ist.

Literatur:

James, John: FORTH for Microcomputers
Dr.Dobb's Journal, Vol.3,Issue 5,P.21
Fritzson, Richard: Write Your Own FORTH Interpreter, Microcomputing, Februar u. März 1981

FORMATIERPROGRAMM

von Günter Böhm

Zur Verbesserung des äußeren Erscheinungsbildes des NASCOM Journals war es notwendig, für die einzelnen Textblöcke einen rechten Randausgleich zu schaffen. Nun wird schon seit langem das Editierprogramm NASPEN angeboten, das sicher sehr gut arbeitet, und Herr Kreidl hat in der letzten Ausgabe gezeigt, wie man das äußerst leistungsfähige Textverarbeitungsprogramm aus der Zeitschrift MC an den NASCOM anpassen kann. Beide Programme bieten aber viel mehr, als ich benötigte und werden dadurch zu kompliziert in der Anwendung. So habe ich mich entschlossen, ein Programm zu schreiben, das Text speichern kann, ihn nötigenfalls verändert und vor allem einen rechten Randausgleich ermöglicht.

Das Programm sollte erst sehr kurz und einfach werden, hat sich aber von Test zu Test so weit gemausert, daß es nun ca. 750 Byte einnimmt. So kann man es aber immer noch ohne Speichererweiterung anwenden, wenn man sich auf kurze Texte beschränkt.

Der Hauptvorteil des Programmes liegt darin, daß der Text völlig unformatiert eingegeben wird. Zum Einlesen genügt also eine Cassette, auf der ohne Rücksicht auf irgendwelche Konventionen Text in ASCII abgespeichert ist. Man kann also Text verarbeiten, der aus einem bestimmten Speicherbereich kopiert ist, oder auch Text, der mit der unten aufgeführten Eingaberoutine auf Band gespeichert wurde.

Diese Ausgabe des Journals ist mit dem Programm gestaltet, was seine Funktionsfähigkeit beweist. Es wird uns ungeheuere Arbeit ersparen, wenn Sie in Zukunft Ihre Textbeiträge auf Cassette einsenden, die wir dann korrigieren und im passenden Format ausdrucken können. Einzige Kriterien für die Speicherung sind:

1. NASCOM 1 Format (Kansas City Standard können wir hoffentlich später verarbeiten)
2. OD als Zeichen für einen Abschnitt im Text.

Wenn Sie das unten aufgeführte Programm für die Textspeicherung auf Cassette verwenden, kann überhaupt nichts schiefgehen. Interessant wäre natürlich die Verwendung von Umlauten; aber aus einem ae können wir im Notfall immer noch ein ä bei der Korrektur machen. Deshalb auch keine Angst vor Rechtschreibfehlern.

Anwendung des Programmes

Das Programm wird bei F67 gestartet. Es meldet sich dann mit "FORMAT" auf dem Bildschirm. Nun kann man durch Eintippen eines Buchstabens und eventuell einer Hexzahl die Funktion wählen, die durch NEW LINE abgerufen wird.

"I" steht für "INPUT" und bedeutet das Eingeben eines Textes in den Textspeicher, der im Programm mit 1000 beginnt aber auch verschoben werden kann. (siehe Assemblerlisting)

"R" steht für "READ", d.h. Einlesen eines Textes von Cassette in den Textspeicher.

"W" bedeutet "WRITE" und speichert den Text aus dem Speicher auf Cassette.

Mit "M" für "MODIFY" kann der TEXT Zeile für Zeile gelesen werden. NL schiebt dabei den Text eine Zeile weiter, Shift und NL eine Zeile zurück. Mit > wird der Text ab der Pfeilstellung nach rechts verschoben, mit< geschieht das gleiche umgekehrt. Man kann dann durch Drücken von Tasten neuen Text einfügen. Mit BACKSPACE bzw. SPACER wird der PFEIL nach links oder rechts verschoben.

"F" bedeutet "FORMATIEREN" und bringt beim Druck die Zeilen auf die Länge, die durch zwei Hexziffern als Argument mit eingegeben werden. Die folgenden zwei Ziffern geben die Anzahl der Zeilen für einen Block an.

Das Format des Journals ist 44 Zeichen pro Zeile und 54 Zeilen pro Spalte. Um einen Text auf dieses Format zu bringen, muß folgendes eingegeben werden:

F2C36 dann NL. Ist eine Spalte fertig gedruckt, meldet sich das Programm mit "Block Ende". Es kann dann das Papier gewechselt werden oder eine neue Spalte eingestellt. Durch das Drücken einer beliebigen Taste wird der Druckvorgang dann fortgesetzt.

Wenn der Text zuende ist, meldet das Programm "Text Ende" und springt zum Monitor zurück, allerdings mit der programmeigenen Befehlstabelle. Zum NASBUG Monitor kann man durch Reset zurückkehren oder durch Anfügen eines zusätzlichen Befehls an die Befehlstabelle am Programmende. Dort können auch leicht noch weitere Routinen angehängt wer-

den. Zum Modify Programm ist noch anzumerken, daß man dieses durch Drücken der " & " Taste verlassen kann.
Das eigentliche Formatierprogramm läuft nun folgendermaßen ab: Nach Start durch F + Parameter lädt das Programm den Beginn des Textspeichers in HL und zählt die eingegebene Zeichenanzahl ab. HL zeigt somit auf den Beginn der nächsten Speicherzeile.
Nun wird die Zeile auf den Bildschirm übertragen, wobei darauf geachtet wird, daß kein Text in den Bildschirmrand gespeichert wird. Es können also beliebig lange Zeilen formatiert werden. DE zeigt dabei auf das letzte Zeichen +1 .
Nun wird das letzte Zeichen untersucht. Wenn es ein Trennungszeichen ist, kann die Zeile gedruckt werden. Wird aber ein Space gefunden, so muß dieser in die Zeile eingefügt werden, damit der rechte Rand ausgeglichen wird. Ist das letzte Zeichen weder Space noch Trennungszeichen, dann nimmt das Programm an, es handelt sich um einen Buchstaben. In HL wird nun getestet, ob ein Space oder Trennzeichen auf das Wort folgt. Ist dies der Fall, so paßt das Wort in die Zeile, und diese wird ausgedruckt.
Falls das Wort aber länger ist, wird zunächst festgestellt, ob sich der Wortteil in der Zeile durch Spaces ersetzen läßt. Sind in der Zeile nicht genügend Spaces zum Einfügen vorhanden, so wird der Bediener zu Hilfe gerufen. Das Zeilenende wird dabei durch einen Pfeil gekennzeichnet, und das letzte Wort wird über die Zeile hinaus dargestellt. Durch Drücken der BACKSPACE Taste schreitet man im Wort nun so weit zurück, bis ein trennbarer Wortteil übrig bleibt. Dann gibt man einen Trennungsstrich ein und startet mit NEW LINE. Sollte jetzt immer noch ein Space am Zeilenende stehen, so kann man diesen durch erneutes NL einfügen.
Nur ein Fall bringt diesen Programmteil zum Zusammenbruch: Wenn in einer Zeile nur ein einziges (langes) Wort steht, findet das Programm keine Spaces zum Einfügen und - bleibt in einer Schleife hängen. Dieser Sonderfall tritt aber normalerweise nur bei sehr kurzen Zeilenformaten auf. Die Erklärungen zur Handhabung klingen komplizierter als sie sind. Wenn Sie das Programm einmal laufen haben, merken Sie sehr schnell, wie das Prinzip funktioniert.

Sehr viel eleganter wäre natürlich eine automatische Worttrennung gewesen, wie sie anscheinend auch schon angeboten wird. Hier konnte ich aber keine Lösung finden. Das liegt nicht an der Programmiertechnik sondern an der Rechtschreibung.Es wäre bestimmt nicht schwierig gewesen, eine Routine zu schreiben, die die grundsätzliche Worttrennung vornimmt.(Bei zwei Konsonanten Trennung der beiden, bei dreien Trennung zwei/eins oder Sonderregelung ck, sp, st etc. Beispiele: Af-fe, Punk-te; Mük-ke etc.) Was mir Schwierigkeiten bereitete, waren Wortpaare wie "Wellen-gang/ lang-atmig", "Gesangs-partie/Breiten-sport" und so weiter. Woher soll nun ein Programm wissen, wann -wie in den Beispielen- ng oder sp getrennt werden und wann nicht? Vielleicht hat da ein Leser eine Idee, wie man das Problem lösen könnte. Bis dahin funktioniert die manuelle Trennung einwandfrei, zumal sie relativ selten in Anspruch genommen wird; meist kommt der Computer alleine mit den Zeilen zurecht.
Der detaillierte Programmablauf kann dem Assemblerlisting entnommen werden. Das Programm ist zunächst für T4 geschrieben, kann aber sehr leicht für T2 geändert werden. Da dieser Monitor kein "Cursor Home" kennt, muß zum "Rückwärtslesen" im Modify Betrieb eine andere Taste verwendet werden. Dazu wird die Zelle 0E66 von 1C zu 25 geändert. Dann kann man die Prozent Taste verwenden.
Auf der Adresse 0F3B ist noch Platz für den Aufruf einer Verzögerung. Diese ist aber nur bei automatischer Steuerung des Cassetten Motors nötig. Falls Sie selbst keine Subroutine dieser Art zur Verfügung haben, können wir eine veröffentlichen. Denken Sie daran, daß Sie beim Einschicken von Beiträgen ans Journal den Text mit dem Eingabeprogramm auf Cassette speichern können. Wir sind sehr dankbar, wenn wir Texte nicht noch einmal abtippen müssen. Achten Sie dann auch bitte darauf, Groß-und Kleinbuchstaben zu verwenden.
Zum Verschieben des Programmes sind wieder die Adressen unterstrichen. Außerdem ist die Relocate Information ab 0F89 angehängt. Achten Sie auch auf den Beginn des Textspeichers in den Eingabeprogrammen bzw. im Formatierprogramm.
Anmerkung zu "INPUT": Das Programm wird mit @ verlassen und zeigt damit im Speicher das Textende an.

Texteingabe für Formatierprogramm

```
0C80            0010        ORG  #C80
013B            0020 CRT    EQU  #13B
0C80 210010     0030        LD   HL,#1000   ;Beginn des Speichers
0C83 CD6900     0040 KEYB   CALL #69        ;Keyboard abfragen
0C86 30FB       0050        JR   NC KEYB
0C88 FE1D       0060        CP   #1D        ; BACKSPACE
0C8A 2006       0070        JR   NZ NEWL
0C8C 2B         0080        DEC  HL         ;Wenn Backspace dann
0C8D CD3B01     0090        CALL CRT        ;zurück im Speicher
0C90 18F1       0100        JR   KEYB       ;und auf Bildschirm
0C92 FE1F       0110 NEWL   CP   #1F        ;Wenn New Line dann
0C94 2009       0120        JR   NZ ENDE    eine Zeile auf Schirm
0C96 CD3B01     0130        CALL CRT        ;nach oben und ASCII
0C99 3E0D       0140        LD   A,#0D      ;für Line Feed in
0C9B 77         0150        LD   (HL),A     ;Speicher
0C9C 23         0160        INC  HL
0C9D 18E4       0170        JR   KEYB
0C9F 77         0180 ENDE   LD   (HL),A
0CA0 23         0190        INC  HL
0CA1 CD3B01     0200        CALL CRT
0CA4 FE40       0210        CP   #40        ;Wenn Endzeichen
0CA6 20DB       0220        JR   NZ KEYB    ;dann zurück zu
0CA8 C35903     0230 ESCAPE JP   #359       ;Monitor (für T2 Sprung
                                            ;nach 0359
```

Formatierprogramm

```
0CAB            0010        ORG  #CAB          Formatierprogramm
0C0C            0020 ARG1   EQU  #C0C
005E            0030 DRUCK  EQU  #005E         Hier: Ausgabe über UART
0CAB EF         0040 FORMAT RST  40            Clear Screen
0CAC 1E00       0050        DEFW #1E
0CAE 210010     0060        LD   HL,#1000      Beginn des Textspeichers
0CB1 ED4B0C0C   0070        LD   BC,(ARG1)     B= Anzahl der Zeichen pro Zeile,C= Zeilen pro
0CB5 114A08     0080        LD   DE,#84A       Block    DE= Beginn des Bildschirmpuffers
0CB8 C5         0090        PUSH BC
0CB9 48         0100        LD   C,B           Zeichenanzahl in BC
0CBA 0600       0110        LD   B,0
0CBC 7E         0120 ERSTZ  LD   A,(HL)        Ist das erste Zeichen einer Zeile ein Zwischen-
0CBD FE20       0130        CP   #20           raum,so wird es nicht ausgedruckt
0CBF 2003       0140        JR   NZ LOOP1
0CC1 23         0150        INC  HL
0CC2 18F8       0160        JR   ERSTZ
0CC4 1A         0170 LOOP1  LD   A,(DE)        Test auf Zeilenende des Bildschirms
0CC5 FE00       0180        CP   0             Wenn ja, dann nächster Zeilenanfang in DE
0CC7 2003       0190        JR   NZ LINEFE
0CC9 118A08     0200        LD   DE #88A
0CCC 7E         0210 LINEFE LD   A,(HL)
0CCD FE0D       0220        CP   #D            TEST auf Zeichen für neue Zeile
0CCF 2006       0230        JR   NZ END1
0CD1 C1         0240        POP  BC
0CD2 12         0250        LD   (DE),A        Line FEED Zeichen abbilden und Zeile zum
0CD3 23         0260        INC  HL            Drucken ausgeben
0CD4 C3650D     0270        JP   AUSGAB
0CD7 FE40       0280 END1   CP   #40           Test auf Endzeichen
0CD9 28F6       0290        JR   Z 5LINEFE     Wenn ja, dann Endzeichen abbilden und Zeile dr.
0CDB EDA0       0300        LDI                Kein Sonderzeichen gefunden: Zeichen aus Spei-
0CDD 78         0310        LD   A,B           cher auf Bildschirm
0CDE B1         0320        OR   C             BC testen: Zeilenende?
0CDF 20E3       0330        JR   NZ LOOP1      nein: weiter
0CE1 C1         0340 TRENN  POP  BC            ja: DE auf letztes Zeichen der Zeile
0CE2 1B         0350        DEC  DE
0CE3 1A         0360        LD   A,(DE)
0CE4 CDEE0C     0370        CALL SUBTRE        Call : ist Trennungszeichen?
0CE7 FE00       0380        CP   0
0CE9 C2650D     0390        JP   NZ AUSGAB     ja: Zeile drucken
0CEC 181D       0400        JR   SPACE         nein: ist Zwischenraum?
0CEE FE21       0410 SUBTRE CP   #21           Diese Subroutine testet, ob das letzte Zei-
0CF0 C8         0420        RET  Z             chen ein Trennungszeichen ist. ( Punkt, Komma,
0CF1 FE2D       0430        CP   #2D           Bindestrich etc.)
0CF3 C8         0440        RET  Z             Trifft das zu, kehrt das Programm mit dem
0CF4 FE2C       0450        CP   #2C           ASCII Code in A zurück.
0CF6 C8         0460        RET  Z             Wenn kein Trennungszeichen, dann steht in A Ø
0CF7 FE2E       0470        CP   #2E
0CF9 C8         0480        RET  Z
0CFA FE2F       0490        CP   #2F
0CFC C8         0500        RET  Z
```

```
0CFD FE3A      0510         CP   #3A
0CFF C8        0520         RET  Z
0D00 FE3B      0530         CP   #3B
0D02 C8        0540         RET  Z
0D03 FE3D      0550         CP   #3D
0D05 C8        0560         RET  Z
0D06 FE3F      0570         CP   #3F
0D08 C8        0580         RET  Z
0D09 AF        0590         XOR  A
0D0A C9        0600         RET
0D0B 1A        0610 SPACE   LD   A,(DE)        ist letztes Zeichen ein Zwischenraum?
0D0C FE20      0620         CP   #20          Wenn nein, dann Sprung zur Buchstabenroutine
0D0E C2B90D    0630         JP   NZ BUCHST    Wenn ja, dann den Zwischenraum in Zeile einfü-
0D11 C5        0640         PUSH BC           gen
0D12 0600      0650         LD   B,0          Test : wieviele "Spaces" sind vorhanden?
0D14 04        0660 ANZAHL  INC  B            Anzahl wird in B gespeichert
0D15 1B        0670         DEC  DE
0D16 1A        0680         LD   A,(DE)
0D17 FE20      0690         CP   #20
0D19 28F9      0700         JR   Z ANZAHL
0D1B 13        0710         INC  DE
0D1C E5        0720 EINFU1  PUSH HL           Einfügen der Spaces
0D1D 214A08    0730         LD   HL,#84A      Pufferzeilen Anfang
0D20 D5        0740         PUSH DE           erster Space am Zeilenende
0D21 23        0750         INC  HL           vom Zeilenanfang suchen, bis Zwischen raum
0D22 7E        0760         LD   A,(HL)       gefunden
0D23 E5        0770         PUSH HL
0D24 A7        0780         AND  A
0D25 ED52      0790         SBC  HL,DE        Test ob Zeile schon durchsucht
0D27 E1        0800         POP  HL
0D28 3008      0810         JR   NC HELP1     wenn ja Sprung zum Hilfsprogramm (manuelles Tren-
0D2A 2806      0820         JR   Z HELP1      nen von Zeilen)
0D2C FE20      0830         CP   #20
0D2E 2807      0840         JR   Z EINFU2     Raum gefunden: Space einfügen
0D30 18EF      0850         JR   5EINFU1     nicht gefunden: weiter suchen
0D32  D1             HELP1  POP DE

0D33 E1        0870         POP  HL
0D34 C3FE0D    0880         JP   HELP
0D37 1A        0890 EINFU2  LD   A,(DE)        vorläufiges Zeilenende in A und retten
0D38 F5        0900         PUSH AF
0D39 13        0910 INCR    INC  DE           eins nach rechts auf dem Schirm
0D3A 1A        0920         LD   A,(DE)       ist dort @ (Bildschirmrand)
0D3B FE00      0930         CP   0
0D3D 28FA      0940         JR   Z INCR       wenn ja: so lange weitersuchen, bis neue
0D3F F1        0950         POP  AF           Zeile erreicht
0D40 12        0960         LD   (DE),A       dort A ablegen.(So wird der Inhalt der Zeile
0D41 1B        0970         DEC  DE           vom ersten Space ab nach rechts verschoben)
0D42 1B        0980         DEC  DE           nächstes Zeichen zum Verschieben suchen
0D43 1A        0990         LD   A,(DE)
0D44 FE00      1000         CP   0            linker Bildschirmrand?
0D46 200C      1010         JR   NZ NONUL
0D48 13        1020         INC  DE           Wenn ja: rückwärts suchen bis vorherige Zeile
0D49 E5        1030         PUSH HL           gefunden und dann über die Nullen im Bild-
0D4A D5        1040         PUSH DE           schirmrand hinweg verschieben. DE zeigt dann
0D4B E1        1050         POP  HL           auf das vorletzte Zeichen der vorherigen Zeile,
0D4C 1B        1060 DECR    DEC  DE           und die Verschiebung kann normal weitergehen.
0D4D 1A        1070         LD   A,(DE)       Die beiden Routinen INCR und DECR sind nur
0D4E A7        1080         AND  A            dann notwendig, wenn Zeilenformate gewählt
0D4F 28FB      1090         JR   Z DECR       werden, die mehr als 48 Zeichen beinhalten.
0D51 77        1100         LD   (HL),A
0D52 1B        1110         DEC  DE
0D53 E1        1120         POP  HL
0D54 A7        1130 NONUL   AND  A            normales Verschieben
0D55 E5        1140         PUSH HL
0D56 ED52      1150         SBC  HL,DE        ist (erster) Zwischenraum schon erreicht?
0D58 E1        1160         POP  HL
0D59 20DC      1170         JR   NZ EINFU2    nein: weiter verschieben
0D5B 1A        1180         LD   A,(DE)       ja: Zwischenraum einfügen
0D5C 13        1190         INC  DE
0D5D 12        1200         LD   (DE),A
0D5E 23        1210         INC  HL           nächstes Wort im Speicher
0D5F D1        1220         POP  DE
0D60 13        1230         INC  DE           nächster Space am Zeilenende
0D61 10BD      1240         DJNZ 4EINFU1      sind schon alle Spaces eingefügt? Nein:weiter
0D63 E1        1250         POP  HL           ja: dann Zeile drucken
0D64 C1        1260         POP  BC
0D65 E5        1270 AUSGAB  PUSH HL           Ausgabe einer Zeile an das Druckprogramm
0D66 C5        1280         PUSH BC           (Der Anfang der nächsten Zeile im Speicher und
0D67 214908    1290         LD   HL,#849      Zeichen-und Zeilenanzahl werden gerettet)
```

```
0D6A 23       1300         INC  HL
0D6B 7E       1310         LD   A,(HL)
0D6C FE40     1320         CP   #40          ist auszugebendes Zeichen das Endzeichen?
0D6E 200E     1330         JR   NZ NEWL2
0D70 EF       1340         RST  40           ja: "Text Ende" auf Schirm schreiben und
0D71 54       1350         DEFM "TEXT ENDE"  Rücksprung zum Monitor
0D7A 00       1360         DEFB 0
0D7B C35903   1370         JP   #359
0D7E FE0D     1380 NEWL2   CP   #D            ist Zeichen ASCII für LINE FEED ?
0D80 2809     1390         JR   Z ENDZ       ja: zur Routine für Ende der Zeile
0D82 FE00     1400         CP   0            ist rechter Bildschirmrand?
0D84 28E4     1410         JR   Z 5AUSGAB    ja: weiter suchen
0D86 CD5E00   1420         CALL DRUCK        kein Sonderzeichen: zur Druckroutine
0D89 10DF     1430         DJNZ 5AUSGAB      Zeile noch nicht zu Ende: weiter ausgeben
0D8B 3E0A     1440 ENDZ    LD   A,#A         Zeile zuende: Carriage Return und LINE FEED
0D8D CD5E00   1450         CALL DRUCK        ausgeben. (Die 5 NOPs sind der Platz, an den
0D90 0000     1460         DEFW 0            man die Ausgabe für Line Feed bei TTY Betrieb
0D92 0000     1470         DEFW 0            einfügen kann)
0D94 00       1480         DEFB 0
0D95 C1       1490         POP  BC
0D96 0D       1500         DEC  C            Zeilenzahl für einen Block herunterzählen
0D97 79       1510         LD   A,C          Block Ende?
0D98 A7       1520         AND  A
0D99 E1       1530         POP  HL
0D9A 2806     1540         JR   Z ENDBL
0D9C EF       1550         RST  40           nein: Clear Screen
0D9D 1E00     1560         DEFW #1E
0D9F C3B50C   1570         JP   10FORMAT     nächste Zeile auf Schirm
0DA2 EF       1580 ENDBL   RST  40           Block Ende:Text auf Schirm
0DA3 42       1590         DEFM "BLOCK ENDE"
0DAD 00       1600         DEFB 0
0DAE CD6900   1610 KEYB2   CALL #69          Call keyboard (Beim Drücken einer beliebigen
0DB1 30FB     1620         JR   NC KEYB2     Taste wird der nächste Block ausgegeben. Bis
0DB3 EF       1630         RST  40           dahin kann man das Papier wechseln oder den
0DB4 1E00     1640         DEFW #1E          Randsteller ändern)
0DB6 C3B10C   1650         JP   6FORMAT
0DB9 7E       1660 BUCHST  LD   A,(HL)       Wie geht das Wort am Zeilenende weiter?
0DBA FE20     1670         CP   #20          mit Space?
0DBC 28A7     1680         JR   Z AUSGAB     dann Zeile drucken
0DBE C5       1690         PUSH BC           nein:
0DBF 0600     1700         LD   B,0
0DC1 04       1710 LETZTZ  INC  B            Wieviele Buchstaben des Wortes stehen in der
0DC2 1A       1720         LD   A,(DE)       Zeile (B zählt)
0DC3 FE00     1730         CP   0
0DC5 2837     1740         JR   Z HELP       Wenn Wort über linken Bildschirmrand reicht:Hilfe
0DC7 1B       1750         DEC  DE           durch manuelles Trennen
0DC8 FE20     1760         CP   #20
0DCA 280A     1770         JR   Z WSPACE     Wortene erreicht: Wieviele Spaces stehen in Zeile
0DCC 13       1780         INC  DE           zur Verfügung?
0DCD CDEE0C   1790         CALL SUBTRE       Trennungszeichen erreicht?
0DD0 FE00     1800         CP   0
0DD2 1B       1810         DEC  DE
0DD3 28EC     1820         JR   Z LETZTZ     kein Space und kein Trennungszeichen: weiter
0DD5 05       1830         DEC  B            nach Wortende suchen
0DD6 E5       1840 WSPACE  PUSH HL
0DD7 214A08   1850         LD HL,#84A
0DDA AF       1860         XOR A                  Akku zählt die
0DDB F5       1870 NEXT    PUSH AF                Spaces, die in der Zeile zum Einfügen zur Ver-
0DDC 23       1880 NOSPAC  INC  HL                fügung stehen
0DDD E5       1890         PUSH HL
0DDE A7       1900         AND  A
0DDF ED52     1910         SBC  HL,DE
0DE1 E1       1920         POP  HL
0DE2 2809     1930         JR   Z TEST
0DE4 7E       1940         LD   A,(HL)
0DE5 FE20     1950         CP   #20
0DE7 20F3     1960         JR   NZ NOSPAC
0DE9 F1       1970         POP  AF
0DEA 3C       1980         INC  A
0DEB 18EE     1990         JR   NEXT
0DED F1       2000 TEST    POP  AF                A wird mit B verglichen (vorhandene und nötige
0DEE E1       2010         POP  HL                Spaces)
0DEF B8       2020         CP   B
0DF0 380C     2030         JR   C HELP            es sind nicht genügend Spaces da: manuell
0DF2 C5       2040         PUSH BC                trennen
0DF3 2B       2050 ENDLIN  DEC  HL                zu löschender Buchstabe im Textspeicher zu-
0DF4 13       2060         INC  DE                rück
0DF5 3E20     2070         LD   A,#20            in Bildschirmzeile Space ans Ende
0DF7 12       2080         LD   (DE),A
0DF8 10F9     2090         DJNZ ENDLIN            wiederholen bis Wortteil durch spaces ersetzt
```

```
0DFA C1        2100         POP   BC
0DFB C31C0D    2110         JP    EINFU1      Spaces am Zeilenende in Zeile einfügen

0DFE C1        0030 HELP    POP   BC
0DFF C5        0040         PUSH  BC
0E00 114908    0050         LD    DE,#849     DE wird auf die Position des letzten Zeichens
0E03 13        0060 INCR    INC   DE          in der Zeile gesetzt
0E04 1A        0061         LD    A,(DE)      Die Nullen des Bildschirmrandes werden
0E05 A7        0062         AND   A           übersprungen
0E06 28FB      0063         JR    Z INCR
0E08 10F9      0070         DJNZ  INCR
0E0A D5        0080         PUSH  DE          Eine Zeile unter "der" Zeile wird das Zeilen-
0E0B E5        0090         PUSH  HL          ende durch einen Pfeil markiert
0E0C EB        0100         EX    DE,HL
0E0D 114000    0110         LD    DE,#40
0E10 19        0120         ADD   HL,DE
0E11 365E      0130         LD    (HL),#5E
0E13 E1        0140         POP   HL
0E14 D1        0150         POP   DE
0E15 D5        0160         PUSH  DE
0E16 C1        0170         POP   BC          BC zeigt nun auf die Zeile
0E17 D5        0180 SAVE    PUSH  DE          Zeilenende und nächstes Zeichen im Speicher
0E18 E5        0190         PUSH  HL          werden gerettet
0E19 7E        0200 WORTDR  LD    A,(HL)      Das Ende des zu trennenden Wortes wird auf
0E1A FE20      0210         CP    #20         dem Schirm angezeigt (Space zeigt das Wortende
0E1C 280B      0220         JR    Z HAND      an)
0E1E 13        0230 NEUZ    INC   DE          Suchvorgang geht weiter
0E1F 1A        0240         LD    A,(DE)
0E20 FE00      0250         CP    0           Bildschirmrand wird übersprungen
0E22 28FA      0260         JR    Z NEUZ
0E24 7E        0270         LD    A,(HL)      Speicherzeichen auf Schirm
0E25 12        0280         LD    (DE),A
0E26 23        0290         INC   HL
0E27 18F0      0300         JR    WORTDR
0E29 E1        0310 HAND    POP   HL
0E2A D1        0320         POP   DE
0E2B CD6900    0330 KEYB3   CALL  #69          Keyboard wartet auf Eingabe
0E2E 30FB      0340         JR    NC KEYB3
0E30 FE1D      0350         CP    #1D
0E32 2007      0360         JR    NZ NEWL3
0E34 3E20      0370         LD    A,#20       Wenn Backspace gedrückt:
0E36 02        0380         LD    (BC),A      in Zeile wird ein Space anstelle eines
0E37 0B        0390         DEC   BC          Buchstabens gesetzt. Zeiger rückt zurück
0E38 2B        0400         DEC   HL          Textspeicher wird zurückgesetzt
0E39 18DC      0410         JR    SAVE        Rücksprung zum keyboard
0E3B FE1F      0420 NEWL3   CP    #1F         ist NEW LINE gedrückt?
0E3D 2004      0430         JR    NZ AENDER
0E3F 13        0440         INC   DE          ja: Zeiger auf Zeilenende +1
0E40 C3E10C    0450         JP    TRENN       Sprung zum Einfügen der Spaces
0E43 03        0460 AENDER  INC   BC          Wenn beliebige Taste gedrückt, ersetzt sie
0E44 02        0470         LD    (BC),A      den Buchstaben ,auf den BC zeigt durch ein
0E45 18E4      0480         JR    KEYB3       entsprechendes Zeichen (z.B.Trennungsstrich)
                                              Rücksprung zum keyboard
```

Modifizieren des Textspeichers

```
0E47           0010         ORG   #E47
0E47 210010    0020         LD    HL,#1000    Anfang Textspeicher
0E4A 114A0B    0030 SCRANF  LD    DE,#B4A     Anfang vorletzte Bildschirmzeile
0E4D 013000    0040         LD    BC,#30      Zeichen pro Zeile
0E50 EF        0050         RST   40          Clear Screen
0E51 1E00      0060         DEFW  #1E
0E53 EDB0      0070         LDIR              Eine Textzeile auf Bildschirm
0E55 0000      0080         NOP NOP
0E57 3E5E      0090         LD    A,#5E       Pfeil als "Cursor" in unterste Bildschirm
0E59 328A0B    0100         LD    (#B8A),A    Zeile ↑
0E5C CD6900    0110 KEYB    CALL  #69         Keyboard wartet auf Eingabe
0E5F 30FB      0120         JR    NC KEYB
0E61 FE1F      0130         CP    #1F
0E63 28E5      0140         JR    Z SCRANF    Wenn NEW LINE gedrückt Sprung:nächste Zei-
0E65 FE1C      0150         CP    #1C         le ausgeben
0E67 2007      0160         JR    NZ RECHTS
0E69 0660      0170         LD    B,#60       Wenn Cursor Home gedrückt:im Textspei-
0E6B 2B        0180 BACK48  DEC   HL          cher um eine Zeile zurück
0E6C 10FD      0190         DJNZ  BACK48
0E6E 18DA      0200         JR    SCRANF
0E70 FE3E      0210 RECHTS  CP    #3E
0E72 2044      0220         JR    NZ BACKSP
0E74 E5        0230         PUSH  HL          Wenn > gedrückt:Anfang nächster Zeile
0E75 CDA30E    0240         CALL  PFEIL1      retten; Position des "Cursors" suchen
```

```
0E78 E1        0250         POP  HL          Anfang nächster Zeile
0E79 E5        0260         PUSH HL
0E7A 2B        0270 SPPOS   DEC  HL          Textspeicher zurück bis "Cursor"
0E7B 10FD      0280         DJNZ SPPOS
0E7D 010100    0290         LD   BC,1
0E80 03        0300 ENDTEX  INC  BC          ab "Cursor" den Textspeicher durchsuchen,
0E81 23        0310         INC  HL          bis Textende gefunden
0E82 7E        0320         LD   A,(HL)
0E83 FE40      0330         CP   #40
0E85 20F9      0340         JR   NZ ENDTEX
0E87 E5        0350         PUSH HL          Textende in DE
0E88 D1        0360         POP  DE
0E89 13        0370         INC  DE          Textende+1
0E8A EDB8      0380         LDDR             Text nach rechts verschieben
0E8C 23        0390         INC  HL          In Textspeicher einen Freiraum einfügen
0E8D 3620      0400         LD   (HL),#20
0E8F CDA30E    0410         CALL PFEIL1      Cursorposition?
0E92 48        0420         LD   C,B         Zeichenzahl ab Cursor nach rechts bis Rand
0E93 0600      0430         LD   B,0         in BC
0E95 117A0B    0440         LD   DE,#B7A     Bildschirmzeile um 1 nach rechts schie-
0E98 21790B    0450         LD   HL,#B79     ben
0E9B EDB8      0460         LDDR
0E9D 23        0470         INC  HL
0E9E 3620      0480         LD   (HL),#20    Space in Bildschirm einfügen
0EA0 E1        0490         POP  HL
0EA1 18B9      0500         JR   KEYB        zurück zur Tastatur
0EA3 11BA0B    0510 PFEIL1  LD   DE,#BBA     Diese Subroutine sucht die Position des
0EA6 0600      0520         LD   B,0         "Cursors" (Pfeil)
0EA8 1B        0530 SUCH    DEC  DE          Sie kehrt zurück mit der Position in
0EA9 04        0540         INC  B           Registerpaar DE und der Anzahl der Zei-
0EAA 1A        0550         LD   A,(DE)      chen von Cursor bis Bildschirmrand in
0EAB FE5E      0560         CP   #5E         Register B
0EAD 20F9      0570         JR   NZ SUCH
0EAF 214000    0580         LD   HL,#40
0EB2 A7        0590         AND  A
0EB3 EB        0600         EX   DE,HL
0EB4 ED52      0610         SBC  HL,DE
0EB6 EB        0620         EX   DE,HL
0EB7 C9        0630         RET
0EB8 FE1D      0640 BACKSP  CP   #1D
0EBA 2018      0650         JR   NZ SPACER
0EBC 11BA0B    0660         LD   DE,#BBA     Wenn BACKSPACE gedrückt:DE auf rechten
0EBF 1B        0670 BACK    DEC  DE          Zeilenrand
0EC0 1A        0680         LD   A,(DE)      zurück bis Cursor gefunden
0EC1 FE5E      0690         CP   #5E
0EC3 20FA      0700         JR   NZ BACK
0EC5 1B        0710         DEC  DE          Cursor -1
0EC6 1A        0720         LD   A,(DE)      Test auf Bildschirmrand
0EC7 FE00      0730         CP   0
0EC9 2891      0740         JR   Z KEYB      wenn Rand: Pfeil stehen lassen
0ECB 3E5E      0750         LD   A,#5E       kein Rand: Cursor auf neue Position
0ECD 12        0760         LD   (DE),A
0ECE 13        0770         INC  DE          alten Cursor löschen
0ECF 3E20      0780         LD   A,#20
0ED1 12        0790         LD   (DE),A
0ED2 1888      0800         JR   KEYB
0ED4 FE20      0810 SPACER  CP   #20
0ED6 202E      0820         JR   NZ LINKS
0ED8 11890B    0830         LD   DE #B89     Wenn Spacer gedrückt: DE auf linken
0EDB 13        0840 FORW    INC  DE          Bildschirmrand
0EDC 1A        0850         LD   A,(DE)      Cursorpos. suchen
0EDD FE5E      0860         CP   #5E
0EDF 20FA      0870         JR   NZ FORW
0EE1 13        0880         INC  DE          Cursor+1
0EE2 1A        0890         LD   A,(DE)
0EE3 FEFF      0900         CP   #FF         Test auf rechten Bildschirmrand
0EE5 CA5C0E    0910         JP   Z KEYB      wenn Rand: Cursor stehen lassen
0EE8 3E5E      0920         LD   A,#5E       kein Rand:neuen Cursor setzen
0EEA 12        0930         LD   (DE),A
0EEB 1B        0940         DEC  DE          alten Cursor löschen
0EEC 3E20      0950         LD   A,#20
0EEE 12        0960         LD   (DE),A
0EEF C35C0E    0970         JP   KEYB
0EF2 FE26      0980 BUCHST  CP   #26         Bei Drücken eines beliebigen Zeichens:
0EF4 283C      0981         JR   Z ESCAPE    ("&" beendet die Routine)
0EF6 E5        0982         PUSH HL
0EF7 F5        0990         PUSH AF
0EF8 E5        1000         PUSH HL
0EF9 CDA30E    1010         CALL PFEIL1      Cursor Position suchen
0EFC E1        1020         POP  HL
```

```
0EFD 2B       1030 ADJUST DEC  HL          Textspeicher um gleiche Anzahl zurück
0EFE 10FD     1040        DJNZ ADJUST
0F00 F1       1050        POP  AF
0F01 77       1060        LD   (HL),A     neues Zeichen in Textspeicher
0F02 12       1070        LD   (DE),A     und auf Bildschirm
0F03 E1       1080        POP  HL
0F04 18D3     1090        JR   -2FORW
0F06 FE3C     1100 LINKS  CP   #3C
0F08 20E8     1110        JR   NZ BUCHST
0F0A E5       1120        PUSH HL          Bei Drücken von < : Verschieben des
0F0B CDA30E   1130        CALL PFEIL1      Textes nach links
0F0E E1       1140        POP  HL          arbeitet analog zu "Rechts", darf aber
0F0F E5       1150        PUSH HL          den Text nicht über den Cursor hinaus
0F10 2B       1160 LEFT   DEC  HL          verschieben
0F11 10FD     1170        DJNZ LEFT
0F13 E5       1180        PUSH HL
0F14 D1       1190        POP  DE
0F15 010000   1200        LD   BC,0
0F18 23       1210 ENDTE1 INC  HL
0F19 03       1220        INC  BC
0F1A 3E40     1230        LD   A,#40
0F1C BE       1240        CP   (HL)
0F1D 20F9     1250        JR   NZ ENDTE1
0F1F D5       1260        PUSH DE
0F20 E1       1270        POP  HL
0F21 E5       1280        PUSH HL
0F22 23       1290        INC  HL
0F23 EDB0     1300        LDIR
0F25 CDA30E   1310        CALL PFEIL1
0F28 48       1320        LD   C,B
0F29 0600     1330        LD   B,0
0F2B E1       1340        POP  HL
0F2C EDB0     1360        LDIR
0F2E E1       1370        POP  HL
0F2F C35C0E   1380        JP   KEYB
0F32 C35903   1390 ESCAPE JP   #359        Nach Drücken von "&" Rücksprung zum
                                           Betriebssystem(mit neuem Comm.Table)
```

Cassetten speichern und lesen

```
0F35          0010        ORG  #F35
0051          0020 MOTOR  EQU  #51
005D          0030 SRLOUT EQU  #5D
003E          0040 SRLIN  EQU  #3E
013B          0050 CRT    EQU  #13B
0F35 210010   0060 WRITE  LD   HL,#1000    Speicherbeginn
0F38 CD5100   0061        CALL MOTOR       Cassettenmotor ein
0F3B 00       0070        NOP
0F3C 00       0080        NOP              (Call Verzögerung;siehe Text)
0F3D 00       0090        NOP
0F3E 7E       0100        LD   A,(HL)      Text auf Cassette
0F3F 23       0110        INC  HL
0F40 F5       0120        PUSH AF
0F41 CD5D00   0130        CALL SRLOUT
0F44 F1       0140        POP  AF
0F45 FE40     0150        CP   #40         bis Textende
0F47 20F5     0160        JR   NZ 9WRITE
0F49 CD5100   0170        CALL MOTOR       Cassettenmotor aus
0F4C C35903   0180        JP   #359        Sprung Monitor
0F4F 21FF0F   0190 READ   LD   HL,#F FF    Speicherbeginn-1
0F52 CD5100   0200        CALL MOTOR       Motor ein
0F55 23       0210        INC  HL
0F56 CD3E00   0220        CALL SRLIN       Zeichen von Cassette
0F59 CD3B01   0230        CALL CRT         auf Bildschirm
0F5C 77       0240        LD   (HL),A      und in Textspeicher laden
0F5D FE40     0250        CP   #40         Test auf Endzeichen
0F5F 20F4     0260        JR   NZ 6READ    Schleife bis Textende
0F61 CD5100   0270        CALL MOTOR       Motor aus
0F64 C35903   0280        JP   #359        Rücksprung Monitor
```

Programmstart

```
0F67          0010        ORG  #F67
0F67 EF       0020 START  RST  40          Clear Screen
0F68 1E       0030        DEFB #1E
0F69 46       0040        DEFM "FORMAT"    Meldung mit Text "Format"
0F6F 1F00     0050        DEFW #1F         neue Zeile
0F71 217A0F   0060        LD   HL,COTAB    neue  Kontrolltabelle
```

```
0F74 22450C    0070          LD   (#C45),HL
0F77 CD8602    0080          CALL #286
0F7A 49        0090 COTAB    DEFB "I
0F7B 800C      0100          DEFW #C80
0F7D 46        0110          DEFB "F
0F7E AB0C      0120          DEFW #CAB
0F80 4D        0130          DEFB "M
0F81 470E      0140          DEFW #E47
0F83 57        0150          DEFB "W
0F84 350F      0160          DEFW #F35
0F86 52        0170          DEFB "R
0F87 4F0F      0180          DEFW #F4F
0F89 00        0190          NOP
```

in entspr. Adresse laden
Sprung zum Monitor (Read a Line and obey)
I ruft Input

F ruft Formatieren

M ruft Modify

W ruft Write (auf Cassette)

R ruft Read (von Cassette)

```
0C80 21 00 10 CD 69 00 30 FB 1E
0C88 FE 1D 20 06 2B CD 3B 01 09
0C90 18 F1 FE 1F 20 09 CD 3B F3
0C98 01 3E 0D 77 23 18 E4 77 FD
0CA0 23 CD 3B 01 FE 40 20 DB 11
0CA8 C3 59 03 EF 1E 00 21 00 01
0CB0 10 ED 4B 0C 0C 11 4A 08 7F
0CB8 C5 48 06 00 7E FE 20 20 93
0CC0 03 23 18 F8 1A FE 00 20 3A
0CC8 03 11 8A 08 7E FE 0D 20 23
0CD0 06 C1 12 23 C3 65 0D FE 0B
0CD8 40 28 F6 ED A0 78 B1 20 18
0CE0 E3 C1 1B 1A CD EE 0C FE 8A
0CE8 00 C2 65 0D 18 1D FE 21 7C
0CF0 C8 FE 2D C8 FE 2C C8 FE A7
0CF8 2E C8 FE 2F C8 FE 3A C8 EF
0D00 FE 3B C8 FE 3D C8 FE 3F 4E
0D08 C8 AF C9 1A FE 20 C2 B9 08
0D10 0D C5 06 00 04 1B 1A FE 2C
0D18 20 28 F9 13 E5 21 4A 08 D1
0D20 D5 23 7E E5 A7 ED 52 E1 4F
0D28 30 08 28 06 FE 20 28 07 E8
0D30 18 EF D1 E1 C3 FE 0D 1A DE
0D38 F5 13 1A FE 00 28 FA F1 78
0D40 12 1B 1B 1A FE 00 20 0C D9
0D48 13 E5 D5 E1 1B 1A A7 28 07
0D50 FB 77 1B E1 A7 E5 ED 52 96
0D58 E1 20 DC 1A 13 12 23 D1 75
0D60 13 10 BD E1 C1 E5 C5 21 BA
0D68 49 08 23 7E FE 40 20 0E D3
0D70 EF 54 45 58 54 20 45 4E 64
0D78 44 45 00 C3 59 03 FE 0D 38
0D80 28 09 FE 00 28 E4 CD 5E F3
0D88 00 10 DF 3E 0A CD 5E 00 F7
0D90 00 00 00 00 00 C1 0D 79 E4
0D98 A7 E1 28 06 EF 1E 00 C3 2B
0DA0 B5 0C EF 42 4C 4F 43 4B C8
0DA8 20 45 4E 44 45 00 CD 69 27
0DB0 00 30 FB EF 1E 00 C3 B1 69
0DB8 0C 7E FE 20 28 A7 C5 06 07
0DC0 00 04 1A FE 00 28 37 1B 63
0DC8 FE 20 28 0A 13 CD EE 0C FF
0DD0 FE 00 1B 28 EC 05 E5 21 15
0DD8 4A 08 AF F5 23 E5 A7 ED 77
0DE0 52 E1 28 09 7E FE 20 20 0D
0DE8 F3 F1 3C 18 EE F1 E1 B8 A5
0DF0 38 0C C5 2B 13 3E 20 12 B4
0DF8 10 F9 C1 C3 1C 0D C1 C5 41
0E00 11 49 08 13 1A A7 28 FB 67
0E08 10 F9 D5 E5 EB 11 40 00 15
0E10 19 36 5E E1 D1 D5 C1 D5 E8
0E18 E5 7E FE 20 28 0B 13 1A 07
0E20 FE 00 28 FA 7E 12 23 18 19
0E28 F0 E1 D1 CD 69 00 30 FB 39
0E30 FE 1D 20 07 3E 20 02 0B EB
0E38 2B 18 DC FE 1F 20 04 13 B9
0E40 C3 E1 0C 03 02 18 E4 21 20
0E48 00 10 11 4A 0B 01 30 00 FD
0E50 EF 1E 00 ED B0 00 00 3E 46
0E58 5E 32 8A 0B CD 69 00 30 F1
0E60 FB FE 1F 28 E5 FE 1C 20 CD
0E68 07 06 60 2B 10 FD 18 DA 0D
0E70 FE 3E 20 44 E5 CD A3 0E 81
0E78 E1 E5 2B 10 FD 01 01 00 86
0E80 03 23 7E FE 40 20 F9 E5 6E
0E88 D1 13 ED B8 23 36 20 CD 65
```

```
0E90 A3 0E 48 06 00 11 7A 0B 33
0E98 21 79 0B ED B8 23 36 20 69
0EA0 E1 18 B9 11 BA 0B 06 00 3C
0EA8 1B 04 1A FE 5E 20 F9 21 85
0EB0 40 00 A7 EB ED 52 EB C9 83
0EB8 FE 1D 20 18 11 BA 0B 1B 0A
0EC0 1A FE 5E 20 FA 1B 1A FE 91
0EC8 00 28 91 3E 5E 12 13 3E 8E
0ED0 20 12 18 88 FE 20 20 2E 1C
0ED8 11 89 0B 13 1A FE 5E 20 34
0EE0 FA 13 1A FE FF CA 5C 0E 46
0EE8 3E 5E 12 1B 3E 20 12 C3 F2
0EF0 5C 0E FE 26 28 3C E5 F5 CA
0EF8 E5 CD A3 0E E1 2B 10 FD 82
0F00 F1 77 12 E1 18 D3 FE 3C 8F
0F08 20 E8 E5 CD A3 0E E1 E5 48
0F10 2B 10 FD E5 D1 01 00 00 0E
0F18 23 03 3E 40 BE 20 F9 D5 77
0F20 E1 E5 23 ED B0 CD A3 0E 33
0F28 48 06 00 E1 ED B0 E1 C3 A7
0F30 5C 0E C3 59 03 21 00 10 F9
0F38 CD 51 00 00 00 00 7E 23 06
0F40 F5 CD 5D 00 F1 FE 40 20 BD
0F48 F5 CD 51 00 C3 59 03 21 AA
0F50 FF 0F CD 51 00 23 CD 3E B9
0F58 00 CD 3B 01 77 FE 40 20 45
0F60 F4 CD 51 00 C3 59 03 EF 8F
0F68 1E 46 4F 52 4D 41 54 1F 7D
0F70 00 21 7A 0F 22 45 0C CD 69
0F78 86 02 49 80 0C 46 AB 0C E1
0F80 4D 47 0E 57 35 0F 52 4F 6D
0F88 0F 52 45 4C 4F D5 0C E5 9E
0F90 0C EA 0C 0F 0D 35 0D A0 9F
0F98 0D B7 0D CE 0D FC 0D 76 D2
0FA0 0E 90 0E FA 0E 0C 0F 26 A4
0FA8 0F 72 0F 7B 0F 7E 0F 81 DF
0FB0 0F 84 0F 87 0F 00 00 00 F7
```

Der Beginn des Textspeichers wird festgelegt in: 0C81, 0CAF, 0E48, 0F36 und 0F50. Die Adresse der Druckroutine ist in folgenden Adressen gespeichert: 0D87 und 0D8E.
Im vorliegenden Listing ist die Druckerroutine SRLOUT (Cassettenrecorder oder Drucker über UART).

SORTIEREN in BASIC Teil 2

von Wolfgang Mayer-Gürr

Wenn man zeitaufwendige Routinen beschleunigen will, sollte man 2 Möglichkeiten untersuchen.

1.Muß der Rechner überflüssige Arbeit verrichten?

2.Gibt es einen effektiveren Algorithmus für das Problem?

Ein Basic-Interpreter arbeitet ein Programm zeilenweise ab. Dabei wird bei jeder Zeile ein Kontrollmechanismus durchlaufen, der die Zeile auf mögliche Fehler (Syntax, Overflow usw.) untersuchen muß. Außerdem erfordern die einzelnen Operationen natürlich einen unterschiedlich hohen Zeitaufwand. Bei dem "Hauruck" Sortierverfahren aus dem letzten Journal stellt man fest, daß 3 Arten von Operationen häufig benutzt werden:

1. Zuweisungen (z.B. H$=N$(I)
2. FOR NEXT Schleifen
3. Vergleiche (IF N$(I)< H$ THEN)

Messen und vergleichen Sie einmal die Rechenzeit für folgende Programme!

1.)
```
100 FOR I=1 TO 1000
110 NEXT I
```

2.)
```
100 FOR I=1 TO 1000
110 A=3.14
120 NEXT I
```

3.)
```
100 FOR I=1 TO 1000
110 A=B
120 NEXT I
```

4.)
```
100 FOR I=1 TO 1000
110 IF A>B THEN PRINT
120 NEXT I
```

5.)
```
100 FOR I=1 TO 1000
110 A=B
120 REM Überflüssige Bemerkung
130 NEXT I
```

6.)
```
100 FOR I=1 TO 1000: A=B: NEXT I
```

7.)
```
100 FOR I=1 TO 1000: A=B: NEXT
```

Merke: Wie mache ich mein Programm schnell, aber unübersichtlich?

Wenn mit Konstanten gerechnet werden soll, empfiehlt es sich, diese immer einer Variablen am Programmanfang zuzuweisen. Das erleichtert auch spätere Änderungen. Es muß nur einmal die Umwandlung stattfinden, ein späterer Zugriff auf diese Variable beeinflußt intern nur 2 Byte statt der 8 Byte des Fließkomma-Akkus. (siehe Beispiel 2 und 3) . REMs erhöhen zwar die Lesbarkeit eines Programms, kosten aber Laufzeit. Ein Tip: bei der Programmentwicklung ruhig viele REM-Zeilen einbauen. Diese sollten aber nie direkt aufgerufen werden (z.B. 100 GOTO 200; 200 REM; 210 A=B). Wenn das Programm ausgetestet ist, kann bei einer 2.Version eine Laufzeitoptimierung vorgenommen werden. Die Variable nach NEXT weglassen (auch hier mit dem Mangel an Übersichtlichkeit erkauft).

Möglichst viele Befehle in eine Zeile packen.

Wenn dies alles noch nicht genug ist, sollte man den Algorithmus überdenken, also die Zahl der Vergleichsoperationen minimieren. Beim Haurucksortieren ist der Programmablauf starr, also unabhängig vom Aufbau des Feldes. Eine Verbesserung bringt das Sortieren durch Einfügen.

```
100  REM  ********************************
110  REM  *  SORTIEREN DURCH EINFUEGEN *
120  REM  ********************************
130  REM
140 N = 10
150  REM  * N = ANZAHL DER ELEMENTE
160  DIM N$(N)
170  FOR I = 1 TO N
180  PRINT "NR. ";I; TAB( 8);
190  INPUT N$(I)
200  NEXT I
210  GOSUB 280
220  REM  * ZUM UNTERPROGRAMM SORTIEREN
230  REM  * AUSGABE
240  FOR I = 1 TO N
250  PRINT N$(I)
260  NEXT I
270  END
280  REM  * SORTIEREN
290  FOR J = 1 TO N - 1
300 H$ = N$(J + 1)
310  FOR I = J TO 1 STEP  - 1
320  IF H$ >  = N$(I) THEN 360
330 N$(I + 1) = N$(I)
340  NEXT I
350 I = 0
360 N$(I + 1) = H$
370  NEXT J
380  RETURN
```

Hier wird nicht jedesmal das Minimum gesucht, sondern die kleinere "Karte" wird sofort zurückgesteckt. Fügt man als Zeile 315 ein Unterprogramm ein, das den Inhalt der einzelnen Variablen ausdruckt, erhält man für die einzelnen Durchläufe bei der Eingabe von S P I E L (N=5) folgende Werte:

```
I  J  H$  N$( 1  2  3  4  5)

1  1  P        S  P  I  E  L
2  2  I        P  S  I  E  L
1  2  I        P  S  S  E  L
3  3  3        I  P  S  E  L
2  3  E        I  P  S  S  L
1  3  3        I  P  P  S  L
4  4  L        E  I  P  S  L
3  4  L        E  I  P  S  S
2  4  L        E  I  P  P  S    und fertig.
```

KLINGEL

von Michael Bach

Schon mehrfach habe ich es vermißt, daß das ASCII-Zeichen 07 (BEL) beim Nascom keinen Piepston auslöst. Das wäre praktisch für Terminal-Betrieb oder auch z.B. bei Basic Programmen, um den Benutzer zu einer Eingabe aufzufordern. Das vorgestellte Programm füllt diese Lücke. Einen Verstärker oder Lautsprecher an ein Bit in Port 0 anzuschließen, ist ja nichts Neues und in dieser Zeitschrift schon beschrieben worden. Das Interessante daran ist vielleicht die Methode, durch Umdefinieren der Output-Tabelle und Verwendung der "User"-Funktion diesen Piepser von allen Programmen durch einen Output auf den Bildschirm einschalten zu können; in Basic z.B. durch den Befehl "PRINT CHR (7)". In der abgedruckten Version erscheint zusätzlich zum Piepston auch das BEL-Symbol auf dem Bildschirm . Das Programm ist relokatibel, kann also an beliebiger Stelle im Speicher stehen. Es muß einmal bei "SUMMER" gestartet werden. Dann kehrt es zum Monitor zurück und kann sofort durch Drücken der Tasten Crtl-G getestet werden. Es bleibt aktiv, bis die Reset-Taste betätigt wird.

```
0080 ;       **** KLINGEL *****        29.03.81
0090 ;
0100 ;M. Bach
0110 ;[redacted]
0120 ;    Stegen
0130 ;[redacted]
0140 ;
0150 ;Wenn der Kode 07H (crtl-G) in der CRT-
0160 ;Routine kommt, wird ein Summer angestellt
0170 ;Der Summer hängt an Bit 5 von Port 0.
0180 ;   Das Programm wird initialisiert durch
0190 ;Start bei SUMMER. Es kehrt zum Monitor
0200 ;zurück; und solange nicht RESET gedrückt
0210 ;wird, kann der Summer von allen Programm-
0220 ;men benutzt werden (z.B. BASIC).
0230 ;Der User-Output wird verwendet und ist
0240 ;damit nicht anderweitig verfügbar.
0250 ;
0260 ;Betriebssystem NAS-SYS
0270 ;
0280 ;
0290 MRET   EQU  #5B
0300 FFLP   EQU  #5E
0310 CRT    EQU  #65
0320 NOM    EQU  #71
0330 ZUOUT  EQU  #75
0340 RPS    EQU  #28
0350 RREL   EQU  #38
0360 BELL   EQU  "G; Kode 07
0377 $UOUT  EQU  #C77
0380 ;
0390        ORG  #8000
0400        ENT
0410 ;
0420 ;Output-Table umschalten
0430 ;-----------------------
8000 0440 SUMMER LD   HL,OUTAB
DF71 0450        SCAL NOM;neue Output-Table
21F900 0460      LD   HL,OUTN
227800 0470      LD   ($UOUT+1),HL;neu
DF5B 0480        SCAL MRET;fertig
0490 ;
0500 ;Neue Output-Table
0510 ;-----------------
7500 0520 OUTAB  DEFB ZUOUT,0
0530 ;
0540 ;Neue CRT-Routine
0550 ;----------------
DF65 0560 OUTN   SCAL CRT;Normalversion
F5   0570        PUSH AF
FE07 0580        CP   BELL;Klingel?
2008 0590        JR   NZ,ORET
C5   0600        PUSH BC
0632 0610        LD   B,50;LÄnge des Piep
3E20 0620 M1     LD   A,#20;an Bit 5
DF5E 0630        SCAL FFLP;1 X umpolen
3E3C 0640        LD   A,60;LÄnge der Verz.
FF   0650        RST  RREL
10F7 0660        DJNZ M1;das Ganze X-mal
C1   0670        POP  BC
F1   0680 ORET   POP  AF
C9   0690        RET
0700 ;
0710 ;Ende
```

TEXTVERARBEITUNG in BASIC

von Thomas E. Schreiner

Das Programm läuft auf NASCOM 1 mit NASSYS in Verbindung mit einem Ausgabeprogramm, das bei C80 beginnen muß. Das Ausgabeprogramm sollte statt einem Semikolon ein Komma setzen. Dies hat einen bestimmten Grund, denn das 8K-Basic läßt keine Kommata im String zu. Daher sind diese im Bildschirmtext durch Semikolons zu ersetzen. Mit 20 000 freien Speicherzellen lassen sich bis zu 250 Zeilen schreiben. Je nach Einstellung lassen sich Zeilen mit einer Länge von 66 bis 70 Buchstaben auf dem Fernschreiber abarbeiten. Das Unterprogramm zur Ausgabe sowie die Steuerung der Zeilenlänge werden vom Programm selbst übernommen. Mittels zweier Steuerbefehle kommt man sofort aus dem Eingabeprogramm: Pfeil nach oben (shift O) bedeutet Sprung in die Subroutinen, "größer als"-Zeichen Sprung zur Ausgaberoutine. Alles

weitere (Rücksprung in die Eingaberoutine, Änderung einer Zeile, Einschieben von Zeilen, zeilenweises Auflisten des Textes und Sprung zur Ausgaberoutine) können vom Unterprogramm direkt aus abgerufen werden. Das Programm wird mit Run 5 gestartet.

Hinweise der Redaktion:

Zeile 10 sollte auf Ihren Speicherumfang zugeschnitten werden. Zeile 15 gibt die max. Zeilenanzahl an. In Zeile 20 wird die Länge der Eingabezeile festgelegt. Die Zeilen 100 und 110 bewirken, daß ein Maschinenprogramm ab C80 bei jedem Printbefehl aufgerufen wird. Dies gilt für NASSYS.

Für T2/4 sollte man in Zeile 100 "Doke 3147, 3200" eingeben. Als Ausgabeprogramm für Drucker ist möglich:

```
0C80 FE 3B 20 02 3E 2C CD 3B   T4!
0C88 01 CD 5E 00 C9
```

Dann wird der Text auf Bildschirm und UART ausgegeben. Natürlich könnte man das auch einfacher machen mit X-Befehl bei T4 und U-Befehl bei NASSYS; aber die Adresse C80 ist eigentlich für ein Programm vorgesehen, das ASSCII in Baudot-Code umwandelt und ausgibt. Dieses Programm wird noch vervollständigt.

(Ein solches Umwandlungsprogramm wurde schon einmal im "Einfachdisassembler" Heft 6-7/80 benutzt.)

```
5 REM TEXTVERARBEITUNGSPROGRAMM
10 CLEAR (20000)
15 DIMA$(200)
20 WIDTH (65)
25 CLS
30 N=1
40 INPUT A$(N)
50 IF A$(N)="↑" THEN GOSUB 200
55 IF A$(N)="↓" THEN 40
60 IF A$(N)="°" THEN 90
70 N=N+1: IF N=220 THEN PRINT "KAPAZITAETSGR
ENZE ° 10 Z .
80 GOTO 40
90 WIDTH(70)
100 DOKE 3192,3200
110 DOKE 3189,1921 : DOKE 3187,1918
120 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRIN
T:PRINT
130 FOR X=1TO N
140 PRINTA$(X): NEXTX
150 DOKE 3189,1922: DOKE 3187,1919
160 INPUT"PRINT°1,NEU°2";X
170 ON X GOTO 110 , 10
200 CLS:INPUT"WELCHE ZEILEN:VON,BIS";G,H
205 IF G=0 THEN CLS:RETURN
210 FOR X=G TO H
220 PRINT X;A$(X)
240 NEXT X
250 INPUT B$
260 GOTO 300
270 INPUT A$(S)
280 GOTO 210
300 PRINT "RUECKSPRUNG° 0 ,ZEILENAENDERUNG°
1"
305 PRINT"EINSCHIEBEN EINER ODER MEHRERER ZEI
LEN°2"
310 PRINT "WEITERES AUFLISTEN°3,DRUCK°4"
320 INPUT S
325 IF S=0 THEN RETURN
330 ON S GOTO 340,360,200,90
340 INPUT"WELCHE ZEILE";S
350 GOTO 270
360 CLS:INPUT"AB WELCHER ZEILE BITTE";S
370 INPUT "WIEVIELE ZEILEN ";L
380 FOR X=N TO S STEP-1
390 A$(X+L)=A$(X):NEXTX
400 N=N+L
410 FOR X=S TO S+L-1:INPUTA$(X):NEXTX
420 PRINT"ES SIND KEINE ZWISCHENZEILEN MEHR
FREI!"
430 FOR F=1TO 3000 :NEXTF
440 CLS:GOTO 200
Ok
```

° ist statt > gedruckt.

LESERBRIEFE

Ich freue mich, daß Sie das Nascom Journal übernommen haben, der Anfang sah ja schon recht vielversprechend aus. Ich hätte auch einige Vorschläge:

1. Drucken Sie Leserbriefe ab.
2. Drucken Sie Hex-Listings mit Prüfsumme. Der Vorschlag wurde ja schon mehrfach unterbreitet und auch verwirklicht, aber nicht oft genug. Z.B. das Listing von Othello war (im Gegensatz zum Abdruck in INMC-News) ohne. Übrigens heißt das Spiel auf deutsch "Reversi".
3. Man sollte mal eine allgemeine Diskussion über das Schreiben relokatibler Programme in Gang setzen. Bei kurzen Programmen ist das ja einfach, aber bei längeren gibt's vielleicht noch ein paar tolle Tips, die viele (ich z.B.) noch nicht kennen.

Michael Bach, Stegen

Antwort der Redaktion:

1. Hat sich hiermit erledigt.
2. Von jetzt an nur noch mit Prüfsumme.
3. Die Diskussion ist hiermit hoffentlich in Gang gesetzt. Wir freuen uns auf Zuschriften.

Kritik: Zu viele primitive Spiele, zu viel NASCOM 1, bzw. T2/T4. Für mich sind technische Erfahrungen mit dem NASCOM 2 interessant. Z.B. läuft meine 16K Speicherplatine nicht auf 4 MHZ. Die Kochrezepte in der Bauanleitung brachten keine Abhilfe. Es sollte mehr Zubehör für den NASCOM vorgestellt werden, auch neue Programme (NASSYS 3, Toolkit ...). Die englische INMC-News gefällt mir wesentlich besser. Leider weiß ich nicht, wie man an die Hefte rankommt.

Mir fielen nur einige durch Zufall in die Hände. An Hardware interessiert mich: A/D-D/A-Wandler (schnell), EPROM-Programmer (der für 185,-DM ist doch wohl leicht überteuert und scheint auch nicht vernünftig zu arbeiten). Werden solche Geräte nicht vorgestellt, damit der eigene Schrott verkauft werden kannn?
Zum Thema Strichcode: Das ist eine interessante Spielerei, aber ich glaube nicht, daß die Druckqualität des Journals eine gute Arbeitsweise des Code-Lesers erwarten läßt. Das Leseprogramm für 88,-DM soll ja wohl ein Witz sein. In der MC gibt's sowas umsonst. Man sollte nicht für einfache Programme horrende Preise fordern und dann erwarten, daß die Leser Programme zum Nulltarif einschicken. Fazit: kein Strichcode!
Das letzte Heft (6/81) war o.k. Wenn's so weiter geht, bin ich zufrieden.

Reinhard Zickwolff, Hannover

Antwort der Redaktion:

1. Die INMC News haben auch uns gut gefallen, und wir versuchten, einiges davon zu übernehmen. Diese Zeitschrift existiert aber nicht mehr
2. Die Redaktion ist in ihrer Meinung völlig unabhängig von MK Systemtechnik und hat kein Interesse, irgendwelchen "Schrott" zu verkaufen. Wenn wir Programme oder Geräte vorstellen, dann deshalb, weil einige Leser daran interessiert sind.
3. Ein schneller AD/DA Wandler wird gerade von Herrn Peter Bentz entwickelt. Er hat vor einiger Zeit im Journal das ausgezeichnete Plotterprogramm vorgestellt, und wir sind gespannt, wie seine Wandlerplatine aussieht. (Sie lassen doch bald von sich hören, Herr Bentz?)
4. Strichcode: Seit wir das neue Papier benutzen, ist die Druckqualität wesentlich besser. Wenn wir den Strichcode plotten,wird die Lesefähigkeit weit größer als im MC (wurde getestet). Der Eigenbau-Plotter (er wird demnächst vorgestellt) braucht nur noch eine bessere Führung des Tuscheschreibers, dann können wir gut lesbare Strichcodelistings auf Sie loslassen. Hier als Vorgeschmack ein Muster.(übrigens im MC-Format!)

5. Das Srichcodeprogramm, das im Mai-Heft angeboten wurde, ist speziell für MK Systemtechnik entwickelt worden. Manche Leute leben nun einmal vom Verkauf von Software, und die wird dann eben teuer. Andere machen das als Hobby nebenher (wie z.B. die Redaktion und viele Leser) und wollen nicht jedes Ergebnis zu barer Münze machen. Daß eine Zeitschrift wie MC es sich leisten kann, Programme entwickeln zu lassen, um sie zu veröffentlichen, ist bei einer Auflage von 60 000 und einer Unmenge von Werbung nicht verwunderlich. Wir hingegen (unsere Auflage traue ich mir nicht bekanntzugeben) sind auf gegenseitigen Austausch der Programme unserer Leser angewiesen. Deshalb können wir es uns auch nicht leisten, "Profi-Artikel" zu veröffentlichen. Die Druckkosten liegen höher als die Abonnement-Einnahmen. MKS schießt den Rest zu. (zum Glück!)

BILDSCHIRM auf CASSETTE

von Günter Böhm

Dieses Miniprogramm läßt nach Start bei 0C80 den Bildschirm vollschreiben. NEW LINE wird als 0D abgebildet. Durch Drücken von @ wird der Textanfang gesucht und der ganze Text bis @ über UART ausgegeben. Dann springt das Programm in den Monitor. Anstelle des Cassettenrecorders könnte auch ein Drucker angeschlossen werden. (Das wäre mit den Editiermöglichkeiten von NASSYS interessant.)

```
                0C80  EF                  Clear Screen
                0C81  1E 00
Keyboard        0C83  CD 69 00            CALL 0069
                0C86  30 FB               JR   NC,0C83
                0C88  FE 1F               CP   1F
                0C8A  20 02               JR   NZ,0C8E
                0C8C  3E 0D               LD   A,0D
CRT             0C8E  CD 3B 01            CALL 013B
                0C91  FE 40               CP   40
                0C93  20 EE               JR   NZ,0C83
                0C95  21 0A 08            LD   HL,080A
                0C98  7E                  LD   A,(HL)
                0C99  FE 20               CP   20
                0C9B  28 03               JR   Z,0CA0
                0C9D  A7                  AND  A
                0C9E  20 03               JR   NZ,0CA3
                0CA0  23                  INC  HL
                0CA1  18 F5               JR   0C98
Motor           0CA3  CD 51 00            CALL 0051
                0CA6  FF                  RST  38
                0CA7  FF                  Platz für den
                                          Aufruf einer
                0CA8  FF                  Verzögerungsroutine

                0CA9  7E                  LD   A,(HL)
                0CAA  23                  INC  HL
                0CAB  A7                  AND  A
                0CAC  28 FB               JR   Z,0CA9
                0CAE  F5                  PUSH AF
UART OUT        0CAF  CD 5E 00      für T2 CALL 005E
                0CB2  F1            muß   POP  AF
                0CB3  FE 40         "5D"  CP   40
                0CB5  20 F2         einge- JR  NZ,0CA9
Motor           0CB7  CD 51 00      setzt CALL 0051
Monitor         0CBA  C3 59 03      werden JP  0359
```

UNTERPROGRAMME für CLDDOS - Teil 2

von Gerhard Baier

Im Teil 1 wurden Eingabe-Routinen für das NASCOM-CLDDOS-System beschrieben. In diesem Teil sollen nun entsprechende Ausgabe-Routinen abgedruckt werden. Die erste Routine konvertiert HEX-Ziffern vom Binaer- ins ASCII-Format. Dann folgen drei Subroutinen für die Ausgabe von HEX-Ziffern, von Bytes und von Worten an das Terminal.

```
$CNVBA
VERSION 1.0
20-NOV-80

DIE SUBROUTINE $CNVBA KONVERTIERT
EINE HEX-ZIFFER VOM BINAER- INS
ASCII-FORMAT.

EINGABE-PARAMETER:
- A -  ZIFFER IM BINAER-FORMAT

AUSGABE-PARAMETER:
- A -  ZIFFER IM ASCII-FORMAT
- CARRY=0 - GUELTIGE ZIFFER IN A
- CARRY=1 - KEINE HEX-ZIFFER IN A

VERWENDETE REGISTER: AF
```

```
0000    FE 00      $CNVBA  CMP     00H
0002    D8                 RET     C
0003    FE 10              CMP     10H
0005    3F                 CCF
0006    D8                 RET     C
0007    FE 0A              CMP     0AH
0009    38 02              JR      C,$CNVBA1
000B    C6 07              ADD     07H
000D    C6 30      $CNVBA1 ADD     30H
000F    37                 STC
0010    3F                 CCF
0011    C9                 RET
```

```
$OUTHEX
VERSION 1.0
25-APR-81

DIE SUBROUTINE $OUTHEX GIBT EINE HEX-ZIFFER
AN DAS TERMINAL AUS. FALLS IM REGISTER A
KEINE HEX-ZIFFER ENTHALTEN IST, WIRD EIN
'?' AN DAS TERMINAL AUSGEGEBEN.

EINGABE-PARAMETER:
- A -  ZIFFER IM BINAER-FORMAT

AUSGABE-PARAMETER:
- A -  ZIFFER IM BINAER-FORMAT

VERWENDETE REGISTER: -
```

```
0000    F5         $OUTHEX PUSH    AF
0001    CD 00 00           CALL    $CNVBA
0004    30 02              JR      NC,$OUTHE1
0006    3E 3F              LD      A,3FH
0008    FF 02      $OUTHE1 SCALL   .SCOUT
000A    F1                 POP     AF
000B    C9                 RET
```

```
$OUTBYT
VERSION 1.0
25-APR-81

DIE SUBROUTINE $OUTBYT GIBT EIN BYTE IN
HEX-DARSTELLUNG AUS.

EINGABE-PARAMETER:
- A -  BYTE

AUSGABE-PARAMETER:
- A -  BYTE

VERWENDETE REGISTER: AF
```

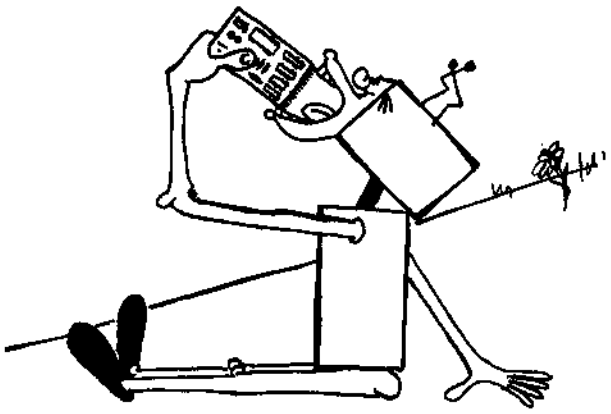
```
0000    CD 03 00   $OUTBYT CALL    $OUTBY1
0003    0F         $OUTBY1 RRCA
0004    0F                 RRCA
0005    0F                 RRCA
0006    0F                 RRCA
0007    F5                 PUSH    AF
0008    E6 0F              AND     0FH
000A    CD 00 00           CALL    $CNVBA
000D    FF 02              SCALL   .SCOUT
000F    F1                 POP     AF
0010    C9                 RET
```

```
$OUTWRD
VERSION 1.0
25-APR-81

DIE SUBROUTINE $OUTWRD GIBT EIN WORD IN
HEX-DARSTELLUNG AN DAS TERMINAL AUS.

EINGABE-PARAMETER:
- HL -  WORD

AUSGABE-PARAMETER:
- HL -  WORD

VERWENDETE REGISTER: -
```

```
0000    F5         $OUTWRD PUSH    AF
0001    C5                 PUSH    BC
0002    E5                 PUSH    HL
0003    06 04              LD      B,04H
0005    AF         $OUTWR1 XOR     A
0006    C5                 PUSH    BC
0007    06 04              LD      B,04H
0009    CB 15      $OUTWR2 RL      L
000B    CB 14              RL      H
000D    17                 RLA
000E    10 F9              DJNZ    $OUTWR2
0010    E6 0F              AND     0FH
0012    CD 00 00           CALL    $CNVBA
0015    FF 02              SCALL   .SCOUT
0017    C1                 POP     BC
0018    10 EB              DJNZ    $OUTWR1
001A    E1                 POP     HL
001B    C1                 POP     BC
001C    F1                 POP     AF
001D    C9                 RET
```

nascompl

PLATINISMUS



Hallo, liebe Leser,
Immer häufiger hört man, daß Besitzer eines NASCOM 1 oder 2 von der Computerseuche No.1 befallen sind: dem Platinismus. Diese Krankheit äußert sich darin, daß der Computer Hobbyist immer häufiger zu Erweiterungs- oder Zusatzplatinen greift, um sein System (wie er meint) zu "ergänzen". Wir müssen dabei zwischen zwei Arten von Platinikern unterscheiden: dem Konfliktplatiniker und dem Gewohnheitsplatiniker. Ersterer Typ bestellt immer dann eine neue Platine, wenn er Schwierigkeiten mit Programmen hat. Dabei schiebt er unbewußt die Schuld auf den Computer, anstatt die Mißerfolge bei seinen Programmierkünsten zu suchen.
Der zweite Typ tritt meistens da auf, wo man sich einen 19" Einschub zur Erweiterung bereitgestellt hat und nun durch die gähnende Leere dazu angeregt wird, immer neue Platinen zu erwerben.
Wie können Sie nun erkennen, ob Sie bereits zu einem dieser Suchttypen gehören? Der Verein AP (Anonyme Platiniker) hat dazu einige Fragen aufgestellt. Wenn Sie mehr als eine mit "ja" beantworten müssen, sind Sie bereits in Suchtgefahr.
1. Wenn Sie eine neue Platine erwerben, denken Sie dann schon an eine neue Erweiterung?
2. Bestellen Sie gleich 2 Platinen, falls eine defekt sein sollte?
3. Verstecken Sie neue Platinen vor Ihrer Frau?
4. Haben Sie eine panische Angst davor, daß MKS wieder in Lieferschwierigkeiten geraten könnte?
Falls Sie befürchten, schon zu tief drin zu stecken, wenden Sie sich vertrauensvoll an mich.
In diesem Sinne Ihr NASCOMPL

REDAKTION: Günter Böhm, Günter Kreidl, Wolfgang Mayer-Gürr, Josef Zeller
RESSORTS:
NASSYS: Günter Kreidl, [redacted] Straelen, Tel.[redacted]
BASIC und FLOPPY: Wolfgang Mayer-Gürr [redacted] Recklinghausen, Tel.[redacted]
HARDWARE: Josef Zeller, [redacted] Bayreuth, Tel.[redacted]
NASBUG T2/T4: Günter Böhm, [redacted], [redacted] Karlsruhe, Tel.[redacted]
Verlag: Verlag NASCOM Journal, c/o MK-Systemtechnik, Pater-Mayer-Str.6, 6728 Germersheim, Tel.07274/2756
Telex 453 500 mks d.
Vertrieb: Direktvertrieb durch den Verlag.
Erscheinungsweise: Monatlich
Bezugspreis: Im Inland 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.
Bezugsmöglichkeiten: Durch Bestellung bei: M K - Systemtechnik.(beigefügte Bestellkarte)
Bankverbindungen: Alle Zahlungen für das NASCOM-JOURNAL unter Angabe der Rechnungsnummer nur (!!) an das folgende Konto:
Fa.Michael Klein, Sonderkonto 29926-674
beim Postscheckamt Ludwigshafen.
Zahlungen: Nach Eingang Ihrer Bestellung erhalten Sie von uns die ausstehenden Hefte bis zur aktuellen Ausgabe sowie eine Rechnung. Bitte, zahlen Sie dann den Rechnungsbetrag auf unser Sonderkonto (s.o.) ein. Bitte keine Vorauszahlungen!

Spielecke

REVERSI von Christoph Rau

Reversi ist - wie Schach, Dame, Mühle etc. - ein strategisches Brettspiel oder, spieltheoretisch ausgedrückt, ein endliches Zwei-Personen-Nullsummen-Spiel. Gespielt wird auf einem Brett mit 8x8 Feldern und mit Spielsteinen, die auf einer Seite weiß und auf der anderen schwarz sind. Der eine Spieler setzt die Steine mit der weißen Seite nach oben, sein Gegner mit der schwarzen Seite nach oben. In der Anfangskonfiguration sind die 4 mittleren Felder mit je 2 schwarzen und 2 weißen Steinen senkrecht untereinander besetzt. Weiß beginnt, und es wird abwechselnd so gesetzt, daß mit einem Stein der eigenen Farbe und dem neu gesetzten Stein eine ununterbrochene Reihe gegnerischer Steine horizontal, vertikal oder diagonal eingeklammert wird. Diese Reihe, also mindestens ein gegnerischer Stein, wird dann umgedreht und damit zu eigenen Steinen. Werden durch den neuen Stein mehrere Reihen eingeklammert, so werden sie alle umgedreht. Kann ein Spieler durch das Setzen eines Steines keine gegnerische Reihe einklammern, so ist sein Gegner noch einmal an der Reihe. Das Spiel ist beendet, wenn keiner der beiden Spieler mehr setzen kann. Gewonnen hat der, von dessen Farbe die meisten Steine auf dem Brett sind.

Das Programm läuft unter NAS-SYS 1 und belegt Speicherplatz von 1oooH bis 16B2H. Gestartet wird das Spiel mit "E 1ooo". Das Programm fragt dann nach der gewünschten Analysetiefe, die Einfluß auf die Qualität der Antwortzüge des Computers hat und ebenso natürlich auf die Antwortzeiten. Es sind Eingaben von 1 bis 6 möglich, gut spielen läßt sich z.B. mit Tiefe 4, was bedeutet, daß der Computer 2 eigene und 2 Gegenzüge tief analysiert.

Nun zeigt das Programm das Spielfeld mit der Anfangskonfiguration und fordert den Spieler (Weiß) zum Setzen auf. Dabei kann der Cursor mit den 4 Kontrolltasten über das Brett bewegt werden, aber nur auf Felder, auf die Weiß auch setzen darf. Durch Drücken der Enter-Taste setzt Weiß auf das Feld, auf dem der Cursor steht, und sämtliche eingeklammerten Reihen werden umgedreht. Danach setzt der Computer. Nach jedem Zug werden Anzahl der weißen und der schwarzen Steine angezeigt. Weiß kann zu jeder Zeit mit "B" sich die Situation vor dem Antwortzug des Computers noch einmal ansehen und mit "T" die Analysetiefe verändern. Bei Spielende zeigt das Programm an, wer wie hoch gewonnen hat und fragt, ob ein neues Spiel gewünscht wird. Reversi ist ein Spiel, das sich relativ leicht und kompakt programmieren läßt, da es mit sehr einfachen Datenstrukturen darzustellen ist. Zuggenerierung und -bewertung geschehen mit einem rekursiven Minimax-Algorithmus mit Alpha-Beta-Verkürzung. Das Minimax-Theorem von Shannon besagt, daß bei einem gegebenen Bewertungsschema das Maximum der für einen Spieler mindestens erreichbaren Werte und das Minimum der für seinen Gegner maximal erreichbaren Werte gleich sind. Der Algorithmus erzeugt also für einen möglichen Zug bis in die gegebene Analysetiefe alle möglichen Gegenzüge und deren Gegenzüge und erzeugt so einen Spielbaum. An jedem Knoten des Spielbaums wird über die Minima der möglichen Züge maximiert und somit die Bewertung für die nächsthöhere Ebene erzeugt. Der Alpha-Beta-Algorithmus beschleunigt die Zuggenerierung dadurch, daß die Analyse von Unterbäumen, die eine schlechtere Bewertung liefern, als bereits erzeugt wurde, so früh wie möglich abgebrochen wird. Es wird also nicht mehr der gesamte Spielbaum untersucht, sondern nur noch die relevanten Teile.

Wem dies alles furchtbar theoretisch und kompliziert vorkommt: im Grunde ist alles "common sense", es ist nur schwer möglich, hier in wenigen Zeilen mehr als die Grundidee zu skizzieren. Wer sich mehr dafür interessiert, den verweise ich auf BYTE November 1979 sowie die angegebene Literatur.

Bibliographie:

Shannon, C.E. "Programming a Computer to Play Chess", Philosophy Magazine, Serie 7 Vol.41, März 195o, 256 ff

Levy, David,"Chess and Computers, Computer Science Press, Woodland Hills CA, 1976

Newborn,Monroe,"Computer Chess", Academic Press, New York, 1975

REVERSI

NASSYS 1

Christoph Rau

```
1000 1700
1000 CD 14 12 CD 17 14 CD 2D F5
1008 16 28 38 CD 4B 16 38 06 FA
1010 F5 DF 5D F1 18 09 CD 90 C0
1018 14 CD 85 11 CD 17 14 ED 84
1020 44 CD 2D 16 CD 4B 16 38 EA
1028 04 ED 44 18 D9 B7 CD 70 52
1030 13 2A DD 19 CD 85 11 CD A3
1038 17 14 ED 44 08 CD B7 10 40
1040 08 18 C3 21 8D 08 D9 7C 3E
1048 95 38 08 27 08 D9 CD E6 E8
1050 12 18 08 7D 94 27 08 D9 AB
1058 CD F3 12 EF 20 67 65 77 8C
1060 69 6E 6E 74 20 6D 69 74 93
1068 20 00 08 DF 68 EF 20 53 49
1070 74 65 69 6E 65 6E 20 56 79
1078 6F 72 73 70 72 75 6E 67 08
1080 2E 00 CD B7 10 ED 53 29 BB
1088 0C EF 6E 6F 63 68 20 65 C0
1090 69 6E 20 53 70 69 65 6C 94
1098 20 3F 00 DF 7B FE 4A CA 73
10A0 00 10 3E 0C F7 DF 5B 01 3C
10A8 00 0A 00 09 00 08 00 FF D2
10B0 FF F6 FF F7 FF F8 FF 11 B2
10B8 5A 0B ED 53 29 0C 3E 1B FB
10C0 F7 C9 D9 08 11 48 00 21 EB
10C8 55 17 78 19 10 FD 42 4B 6F
10D0 11 4C 17 08 20 07 08 47 D2
10D8 08 E5 D9 E1 C9 30 01 EB 74
10E0 08 ED B0 47 11 18 00 21 26
10E8 35 19 19 10 FD 4F 06 02 C3
10F0 DD 21 B3 16 08 38 45 56 A2
10F8 DD 72 02 23 56 DD 72 07 28
1100 23 56 DD 72 0A 23 56 DD 39
1108 72 0B 23 56 DD 72 10 23 91
1110 56 DD 72 11 23 56 DD 72 9F
1118 37 23 56 DD 72 38 23 56 D9
1120 DD 72 3D 23 56 DD 72 3E C3
1128 23 56 DD 72 41 23 56 DD 98
1130 72 46 23 DD 21 FB 16 10 3B
1138 BE 41 D9 C9 DD 56 02 72 91
1140 23 DD 56 07 72 23 DD 56 76
1148 0A 72 23 DD 56 0B 72 23 CB
1150 DD 56 10 72 23 DD 56 11 7D
1158 72 23 DD 56 37 72 23 DD DA
1160 56 38 72 23 DD 56 3D 72 76
1168 23 DD 56 3E 72 23 DD 56 D5
1170 41 72 23 DD 56 46 72 23 65
1178 DD 21 FB 16 10 BE 41 D9 80
1180 C9 77 E5 18 12 06 08 DD CB
1188 21 A7 10 77 E5 DD 5E 00 08
1190 DD 23 DD 56 00 DD 23 ED C1
1198 44 0E 00 19 0C BE 28 FB 01
11A0 ED 44 BE 20 09 0D 28 06 04
11A8 B7 ED 52 77 18 F7 E1 10 26
11B0 DB CD 2B 15 C9 47 7E B7 EE
11B8 78 C0 06 08 DD 21 A7 10 C4
11C0 ED 44 E5 DD 5E 00 DD 23 22
11C8 DD 56 00 DD 23 19 BE 20 03
11D0 0B 19 BE 28 FC ED 44 BE D6
11D8 28 09 ED 44 E1 10 E3 ED 0C
11E0 44 A7 C9 37 E1 C9 00 40 C6
11E8 E0 0A 05 05 0A E0 40 00 17
11F0 E0 D0 02 01 01 02 D0 E0 67
11F8 00 0A 02 05 01 01 05 02 23
1200 0A 00 05 01 01 01 01 01 26
1208 01 05 00 05 01 01 01 01 29
1210 01 01 05 00 3E 0C F7 21 8B
1218 9E 16 11 D7 0B 01 15 00 E7
1220 ED B0 3E 5F 21 0A 08 01 A0
1228 02 30 77 23 10 FC 21 8A BD
1230 0B 06 30 0D 20 F4 01 24 C9
1238 00 C5 11 B3 16 21 E6 11 01
1240 ED B0 06 24 7E 12 2B 13 E7
1248 10 FA C1 ED B0 06 24 7E 6A
1250 12 2B 13 10 FA 21 43 17 37
1258 E5 3E 02 06 0A 77 23 10 49
1260 FC AF 06 47 77 23 10 FC 10
1268 EB E1 0E 09 ED B0 3E 02 3A
1270 06 07 11 09 00 19 77 10 49
1278 FC 47 3D 1D 21 6B 17 77 41
1280 ED 44 23 77 ED 44 19 10 B7
1288 F6 FD 21 DD 19 21 60 F0 15
1290 22 E9 19 21 D5 09 CD 00 92
1298 13 21 8A 09 CD E6 12 21 57
12A0 92 09 22 29 0C EF 28 2A E5
12A8 29 00 06 02 21 8E 0A 22 C6
12B0 29 0C EF 53 74 65 69 6E E9
12B8 65 00 21 AF 0A 10 F0 21 2A
12C0 AE 09 CD F3 12 21 B6 09 3B
12C8 22 29 0C EF 28 4F 29 00 C0
12D0 21 9F 08 22 29 0C EF 69 59
12D8 73 74 20 61 6D 20 5A 75 AE
12E0 67 2E 00 3E 01 C9 22 29 DA
12E8 0C EF 20 20 57 65 69 73 CD
12F0 73 00 C9 22 29 0C EF 53 D7
12F8 63 68 77 61 72 7A 00 C9 62
1300 08 22 29 0C EF 41 6E 61 71
1308 6C 79 73 65 74 69 65 66 80
1310 65 20 28 31 20 2D 20 36 A4
1318 29 20 3F 20 00 E5 DF 7B 12
1320 E1 FE 31 38 DC FE 37 30 BC
1328 D8 06 30 90 D9 47 D9 3E 10
1330 1B F7 08 C9 06 47 21 00 94
1338 00 DD 21 FC 16 08 DD 7E BE
1340 51 16 00 B7 28 1C 3D 28 1A
1348 10 3D 28 16 DD 5E B8 7B 54
1350 17 30 02 15 B7 ED 52 18 CF
1358 09 DD 5E 00 7B 17 30 01 72
1360 15 19 DD 23 10 D8 08 B7 48
1368 F0 EB 21 00 00 ED 52 C9 7F
1370 37 CD C2 10 11 60 F0 FD B7
1378 73 0E FD 72 0F 21 4D 17 0F
1380 06 47 C5 E5 D5 CD B5 11 F2
1388 30 36 CD 81 11 D9 48 0D 8E
1390 20 5E D9 CD 34 13 EB FD F6
1398 6E 0C FD 66 0D 19 CB 14 8D
13A0 30 6B EB D1 D5 EB B7 ED 6E
13A8 52 F2 BC 13 E1 E1 E5 FD 72
13B0 75 00 FD 74 01 D5 FD 73 EF
13B8 0E FD 72 0F B7 CD C2 10 AD
13C0 D1 E1 C1 23 10 BC 21 60 B6
13C8 F0 B7 ED 52 28 02 EB C9 9F
13D0 CD 2D 16 21 48 F4 C0 CD DD
13D8 81 16 07 0F 30 06 EB 21 DA
13E0 00 00 ED 52 06 06 CB 24 2D
13E8 CB 25 30 01 24 10 F7 C9 10
13F0 05 D9 FD 23 FD 23 ED 44 52
13F8 CD 70 13 EB 21 00 00 B7 1E
1400 ED 52 FD 2B FD 2B ED 44 D4
1408 D9 04 D9 18 89 E1 D5 B7 E0
1410 CD C2 10 E1 D1 C1 C9 21 20
1418 4D 17 08 06 08 D9 21 00 A0
1420 00 D9 11 1A 09 C5 06 08 14
1428 7E FE 00 28 1A FE 01 28 21
1430 0B 3E 01 D9 85 27 6F D9 5B
1438 3E 4F 18 0D 3E 01 D9 84 9A
1440 27 67 D9 3E 2A 18 02 3E 7B
1448 2E 12 13 13 23 10 D9 E5 B3
1450 21 30 00 19 EB E1 23 C1 7E
1458 10 CB 21 10 0A 22 29 0C D9
1460 D9 7C DF 68 D9 21 31 0A 45
1468 22 29 0C D9 7D DF 68 D9 49
1470 08 C9 08 11 4C 17 D5 2B D1
1478 D1 D5 E5 B7 ED 52 E1 30 1E
1480 03 21 93 17 CD B5 11 30 25
1488 EE D1 CD 08 15 08 18 0D 72
1490 21 4C 17 23 CD B5 11 30 0E
1498 FA 08 CD 08 15 E5 DF 7B D7
14A0 E1 FE 42 20 13 08 F5 AF B4
14A8 CD C2 10 F1 23 CD 1A 14 6A
14B0 08 CF 08 CD 17 14 18 D8 8B
14B8 FE 54 20 09 08 21 D5 08 4D
14C0 CD 00 13 18 CB FE 0D 28 CA
14C8 3A FE 11 28 A5 FE 12 28 2A
14D0 14 FE 13 20 08 11 08 00 4A
14D8 B7 ED 52 18 95 FE 14 20 C1
14E0 BC 11 08 00 19 08 11 94 8F
14E8 17 D5 23 D1 D5 E5 B7 ED 3A
14F0 52 E1 38 03 21 4D 17 CD C4
14F8 B5 11 30 EE D1 CD 08 15 AB
1500 08 18 9A CD B7 10 08 C9 34
1508 E5 11 4C 17 B7 ED 52 11 7D
1510 09 00 06 00 04 ED 52 30 A7
1518 FB 19 CB 25 11 D8 08 19 3B
1520 11 40 00 19 10 FD 22 29 F7
1528 0C E1 C9 FD E5 DD 21 B3 86
1530 16 FD 21 FB 16 B7 F2 41 74
1538 15 DD E5 FD E5 DD E1 FD C1
1540 E1 47 E5 11 4D 17 ED 52 16
1548 20 30 FD 36 02 05 FD 36 1A
1550 0A 05 FD 36 0B 05 DD 36 CA
1558 0B 05 3A 4F 17 B8 28 06 03
1560 DD 36 02 FB 18 04 DD 36 B4
1568 02 0A 3A 5F 17 B8 28 06 1F
1570 DD 36 0A FB 18 04 DD 36 CC
1578 0A 0A E1 E5 11 54 17 B7 9A
1580 ED 52 20 30 FD 36 07 05 63
1588 FD 36 10 05 FD 36 11 05 2E
1590 DD 36 10 05 3A 52 17 B8 28
1598 28 06 DD 36 07 FB 18 04 0C
15A0 DD 36 07 0A 3A 66 17 B8 48
15A8 28 06 DD 36 11 FB 18 04 26
15B0 DD 36 11 0A E1 E5 B7 11 81
15B8 8C 17 ED 52 20 30 FD 36 32
15C0 37 05 FD 36 38 05 FD 36 B4
15C8 41 05 DD 36 38 05 3A 7A 27
15D0 17 B8 28 06 DD 36 37 FB 27
15D8 18 04 DD 36 37 0A 3A 8E 25
15E0 17 B8 28 06 DD 36 41 FB 41
15E8 18 04 DD 36 41 0A E1 E5 3D
15F0 11 93 17 B7 ED 52 20 30 06
15F8 FD 36 3D 05 FD 36 3E 05 F8
1600 FD 36 46 05 DD 36 3D 05 E9
1608 3A 91 17 B8 28 06 DD 36 F9
1610 46 FB 18 04 DD 36 46 0A E6
1618 3A 81 17 B8 28 06 DD 36 F9
1620 3E FB 18 04 DD 36 3E 0A E6
1628 E1 FD E1 78 C9 0E 02 21 6F
1630 4D 17 06 47 C5 CD B5 11 4F
1638 C1 38 0A 23 10 F6 ED 44 AB
1640 B7 0D 20 EB C9 0D C0 ED A8
1648 44 B7 C9 38 28 21 5D 0B 0B
1650 22 29 0C 08 EF 6B 61 6E EE
1658 6E 20 6E 69 63 68 74 20 32
1660 73 65 74 7A 65 6E 2E 00 3D
1668 08 F5 07 21 55 0B 38 0C 47
1670 CD E6 12 F1 C9 21 97 08 C5
1678 F5 07 30 F4 CD F3 12 F1 71
1680 C9 21 00 00 11 4D 17 06 FB
1688 47 08 1A B7 28 0B FE 02 F1
1690 28 07 23 FE 01 28 02 2B 4C
1698 2B 13 10 EE 08 C9 2A 20 05
16A0 2A 20 52 20 45 20 56 20 4D
16A8 45 20 52 20 53 20 49 20 71
16B0 2A 20 2A 00 40 E0 0A 05 69
16B8 05 0A E0 40 00 E0 D0 02 AF
16C0 01 01 02 D0 E0 00 0A 02 96
16C8 05 01 01 05 02 0A 00 05 FB
16D0 01 01 01 01 01 01 05 00 F1
16D8 05 01 01 01 01 01 01 05 FE
16E0 00 0A 02 05 01 01 05 02 10
16E8 0A 00 E0 D0 02 01 01 02 BE
16F0 D0 E0 00 40 E0 0A 05 05 EA
16F8 0A E0 40 00 40 E0 0A 05 67
```

GEWINNER DES PREISAUSSCHREIBENS

Eine so rege Beteiligung am Spiele-Preisausschreiben hatten wir nicht erwartet. Um so schwerer fiel uns natürlich auch die Auswahl des Gewinners. Nachdem wir uns zwischen zwei Favoriten überhaupt nicht mehr entscheiden konnten, half MKS- Systemtechnik durch das Aussetzen eines weiteren Preises. Alle Einsendungen werden im Laufe der Zeit veröffentlicht, und jeder Einsender erhält als "Trostpflaster" eine Cassette mit den Spielen, die auf seinem System laufen.

Und hier sind nun die beiden Gewinner:

Herr Markus Caesar
Eichenweg 11
5653 Leichlingen

Herr Caesar hat uns ein äußerst umfangreiches BASIC Programm geschickt, das er "STAR TREK" nennt. Es ist ein sehr unterhaltsames Weltraumspiel. Wir werden es im nächsten Heft vorstellen und möglicherweise ein Listing abdrucken. (Das ist eine Platzfrage). Jedenfalls werden Sie die Möglichkeit haben, das Programm auf Cassette zu beziehen. Herr Caesar hat sich bereit erklärt, gegen Einsendung eines Freiumschlags und einer Cassette das Programm zu kopieren. Allerdings lädt Herr Caesar im NASCOM2 Format; wir werden es uns aber auch ins NASCOM1 Format kopieren lassen.

Mit vielleicht nicht soviel Action, dafür aber in einem ausgeklügelten Maschinenprogramm hat uns

Herr Christoph Rau
[redacted]
[redacted] Bonn

sein Spiel "Reversi" eingesandt. Das Spiel selbst ist wohl nicht mehr ganz neu, aber er hat es sehr interessant aufbereitet. Wir haben es in diesem Heft in der "Spielecke" abgedruckt, die von "Anti-Spielern" aus dem Heft entfernt werden kann. Herr Rau gibt Ihnen dort selbst eine Einführung zu seinem Spiel.

Beide Gewinner erhalten demnächst ihr Schachprogramm nach Wahl. Die Redaktion ist schon dabei, das nächste Preisausschreiben auszuknobeln. Vielleicht hätten einige Leser hierzu interessante Vorschläge.

ASCII - BAUDOT

Codeumwandlung
von Thomas E. Schreiner

Gerade noch rechtzeitig für diese Ausgabe ist das angekündigte Maschinen-Programm für die Umwandlung von ASCII in Baudot-Code von Herrn Schreiner angekommen. Für die Verwendung mit dem Programm "Textverarbeitung in BASIC" in diesem Heft sind folgende Änderungen vorzunehmen: 0D1B muß von 0D (Semikolon) nach 0C (Komma) geändert werden. Im BASIC Programm muß Zeile 100 lauten:

100 Doke 3192,3225

Zur Verschiebung des Programmes müssen die Werte in den Adressen 0C9C, 0CB2/3, 0CBF/CC0 und 0CE2/3 angepaßt werden. Die Ausgaberoutine DF 6F läuft nur auf NASSYS.

```
0C98 00 F5 C5 06 0D 4F FE 20 DE
0CA0 20 07 3E 04 DF 6F C1 F1 15
0CA8 C9 00 38 3E FE 40 38 24 8D
0CB0 F5 3A 3F 0D FE 00 28 09 66
0CB8 3E 1F DF 6F 3E 00 32 3F 1E
0CC0 0D F1 FE 5E 20 02 3E 56 DC
0CC8 D6 20 FE 40 38 02 D6 20 38
0CD0 4F 00 18 26 3A 3F 0D FE ED
0CD8 5A 28 09 3E 1B DF 6F 3E 54
0CE0 5A 32 3F 0D 79 D6 20 4F 82
0CE8 18 10 FE 08 20 04 3E 04 88
0CF0 18 09 3E 08 DF 6F 3E 02 F1
0CF8 18 01 0A DF 6F C1 F1 C9 F0
0D00 04 1A 05 0B 14 04 11 05 69
0D08 0F 12 09 11 0C 03 1C 1D 98
0D10 16 17 13 01 0A 10 15 07 94
0D18 06 18 0E 0D 0F 1E 12 19 B6
0D20 08 03 19 0E 09 01 0D 1A 90
0D28 14 06 0B 0F 12 1C 0C 18 BB
0D30 16 17 0A 05 10 07 1E 13 C1
0D38 1D 15 11 00 00 00 1E 00 A6
```


Die Autoren dieser Ausgabe:
Günter Böhm, Günter Kreidl, Wolfgang Mayer-Gürr (siehe Impressum), Gerhard Baier, [redacted], Tel. [redacted]; Thomas E. Schreiner, [redacted], [redacted] Hannover, Tel. [redacted]; Michael Bach, [redacted], [redacted] Stegen, Tel. [redacted].
Die Autoren sind für ihre Texte selbst verantwortlich.

kleinanzeigen

Jeder Abonnent kann kostenlose Kleinanzeigen bis 40 Wörter aufgeben!

SUCHE NAS-SYS 3 Manual + Ass.Listing +
DOUBLE PRECISION BASIC für CLD
Tel. [redacted]

VERKAUFE
NASCOM-CLD-FLOPPY (Controller-Karte, ein Laufwerk)
mit CLDDOS, Editor, Assembler, Debugger
CLD BASIC + MICROSOFT-DOUBLE PREC.BASIC
+ MICROSOFT-FORTRAN (beides für Floppy)
zusammen 1700.- VB
Gerhard Baier, [redacted], Tel. [redacted] (tagsüb.)
[redacted] (abends)

VERKAUFE wegen Umstellung

1 Assembler ZEAP II	4X2708	150.-
1 NASDIS mit DEBUG 3+1	2708	120.-
1 GRAPHIK ROM	2716	40.-
1 BASIC TOOLKIT	2X2708	75.-

E.Horch, Tel. [redacted]
[redacted]

Wer hat Erfahrung mit dem MERSEYSIDE TB ?
Wer besitzt PROGRAMMSAMMLUNG NASCOM SOFTWARE in Buchform (MK-N-647) und könnte sie mir kurzfristig ausleihen??
(bei MK leider vergriffen)
Franz-L. Bruhns , [redacted]
[redacted] Tel. [redacted]

VERKAUFE

ZEAP 2.0	4K Eproms	130.-
NASDIS	3K Eproms	100.-
DEBUG	1K Eprom	40.-
oder alles zusammen		260.-

Hans-Martin Pohl , [redacted]
[redacted] Tel. [redacted]

VERKAUFE FERNSCHREIBER LO 15c mit Lochstreifenleser u. Sender, sowie Interface für NASCOM. VB 150.-
NAS-SYS in EPROMs VB 60.-
Achim Kaufmann , [redacted]
[redacted]

VERKAUFE
8 K BASIC in 8 EPROMs 2708 160.-
ZEAP 1.1 3K Assembler/Editor für T2/T4 in 3 EPROMs 2708 60.-
Hans Schneider , [redacted]
Tel. [redacted] ab 18°° Uhr

VERKAUFE NASCOM 1 SUPERSYSTEM
32 K RAM Board, EPROM Board, Netzteil, Tastatur Up Grade, NASSYS 1+3, 8K BASIC (ROM + Tape!!) Tool Kit, ZEAP 2.0 , NASDIS, DEBUG, Menueprogramm, EPROM-Programmiergerät, alles verpackt im prof. NASCOM-Pultgehäuse, Software und Beschreibung dabei.
VHB 1750.- (Neupr. über 4000.-)
GÜnter Mink , [redacted]
[redacted] Tel. [redacted]

SUCHE SCHNELLEN 8Bit A/D - WANDLER mit Sample And Hold (Hardwarebeschreibung) für Spracheingabe
Gerhard Baier , [redacted]
[redacted]

VERKAUFE
1. EPROM - Assembler für NASBUG in drei EPROMS mit Adresse D000 u. deutschem Handbuch 90.-
2. NAS-SYS Monitor 3 (Nachfolger Mon.1) mit Handbuch 75.-
3. 6 stat. RAMs 2114 (2k) 48.-
Heinz Oligmüller , [redacted]
[redacted]

VERKAUFE
ZEAP 2 , NASDIS/NASBUG , Toolkit, Grafik ROM
Pos. 1+2 je 140.- , Pos. 3+4 je 50.-
alles in EPROM 2708
Tel. [redacted]

LEERKASSETTEN

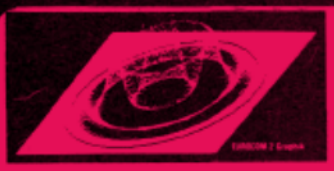
Speziell geeignet für Datenaufzeichnung. Hochwertiges BASF-Band. Cassette 5-fach verschraubt. Cassette C10,d.h. 10 Minuten spieldauer, daher besonders geeignet für Mikrorechnerprogramme.

10 Stk	19.80	Jede Kassette mit selbst-
20 Stk	36.00	klebenden Aufkleber zum
50 Stk	87.50	Beschriften.
100 Stk	160.00	

Bei: M K - Systemtechnik

MK-SYSTEMTECHNIK

Jetzt auch in Solingen!



EUROCOM-2 1670,-

Einplatinencomputer im Doppel-Europaformat. Sehr leistungsfähige Graphik.

- 48k RAM, 4k Betriebssystem
- 6809 CPU (interne 16 Bit Struktur!)
- 128 Zeichen, Groß/Kleinschreibung
- Graphik und Text beliebig mischbar
- Kansas-City-Standard-Interface
- 40 E/A-Leitungen, RS 232 C-Anschluß
- Ausbaubar auf Farbgraphik
- auf 240k RAM erweiterbar

Zubehör für EUROCOM-2

Floppy-Controller Single-Density ohne DMA 1127,-
5"-Laufwerk 847,-
BUS-Karte 84,-
RAM-Karte 32k 779,-
RAM-Karte 96k 1977,-
5A EUROCOM II-Netzteil 384,-
ASCII-Tastatur 279,-
Joystick 110,-

Software f. EUROCOM-2

BASIC 220,-
Assembler/Editor 220,-
DEBUG 179,-
Disassembler 113,-
PASCAL 350,-
FORTH 220,-

wahlweise auf Audiocassette oder Mini-Digitalcassette

MIKOS 1, «Steckdosenfertiges» Komplettgerät auf Basis d. EUROCOM II

inkl. Tastatur und Betriebssystem mit 1 Mini DCR 2975,-
mit 2 Mini DCR 3495,-
mit 3 Mini DCR 3975,-
mit 3 Mini DCR 4425,-

apple computer

ITT 2020

Unser PASCAL-System:

Enthält: Wahlweise Apple II oder ITT 2020, 64k RAM (Hardware für PASCAL-Language System), 12" 18 Mhz-Monitor grün, 2 Stück 5.25" Floppy-Disk-Laufwerke mit Controller, **plotfähiger Drucker EPSON MX 82 FT,** UCSD-PASCAL-Software.
Komplett mit allen Handbüchern und Verbindungskabeln **9985,-**

Unser Farb-System:

Enthält: ITT 2020 mit PAL-Ausgang, dadurch besonders gutes Farbbild, 48k RAM, 14" SANYO echter Farbmonitor, mit Grünschalter für Computertextdarstellung, auch als vollwertiger 8-Kanal-Farbfernseher zu verwenden, 2 Stück 5.25" Floppy-Disk-Laufwerke mit Controller, plotfähiger Drucker EPSON MX 82 F/T, BASIC-Lehrgang auf Diskette.
Komplett mit allen Handbüchern und Verbindungskabeln **8885,-**

Unser Grafik-System:

Enthält: Apple II 48k RAM, 5.25" Floppy-Laufwerk mit Controller, Apple-Grafics-Tablet, plotfähiger Silentype-Drucker, passend zum Grafics-Tablet.
Komplett mit allen Handbüchern und Verbindungskabeln **7985,-**

Für Einzelkomponenten oder andere Konfigurationen übersenden wir Ihnen gerne ein individuelles Angebot!

Erweitern Sie Ihren Apple II / ITT 2020:

Timer Modul	295,-	Asynchron Interface	425,-
3 3/4 BCD A/D Wandler	295,-	Synchron Interface	495,-
IEEE Bus Interface	785,-	Parallel Interface	325,-
Arithmetic Processor	995,-	Kalender/Uhr Modul	335,-

Fordern Sie unseren kostenlosen Zubehör- und Software-Katalog sowie unsere CP/M-Sonderliste an!

WATANABE Plotter 3365,-

An alle Microcomputer mit Parallelschnittstelle anschließbar. DIN A/3.

Zubehör:
Interface u. Kabel IEEE 488 449,-
Interface u. Kabel für Apple und ITT 2020 395,-
Interface u. Kabel RS 232 C 849,-

Neu!
Plotbibliothek in UCSD-PASCAL 895,-
Plotsoftware, dialogorientiert 499,-
WATANABE WX 4675 4945,-
Intelligenter 6-Farben-Plotter DIN A/3

Endlich lieferbar!

MX 82 F/T

mit einem Interface n. Wahl* 2595,-
oh. Int.face (8 Bit Parall.-Eing.) 2325,-

Der neue MX 82 F/T besitzt neben allen bewährten Eigenschaften des MX 80 F/T die Fähigkeit, hochauflösende Grafik zu plotten.

MX 80 F/T

o. Interf. (8 Bit Parall.-Eing.) 1625,-
m. einem Interface n. Wahl 1895,-

*Interfacekarten für alle gängigen Rechnersysteme lieferbar: PET/CBM, TRS 80, MZ 80 K, IEE 488 (HP), HEATH-Computer, NASCOM, COMPUCORP oder RS 232 C (V24).

ATARI 400 1698,-

16k RAM, BASIC-ROM, deutsche Handbüch., PAL-Ausg. m. 16 x 8 Farb.

ATARI 800 2998,-

16k RAM, BASIC-ROM, deutsche Handbüch., PAL-Ausg. m. 16 x 8 Farb.
16k RAM-Erweiterungsmodul 285,-
5.25" Floppy incl. dt. Handb. 1749,-

VIDEO-GENIE 3003

(neue Ausführung, mit Cassetten-Laufwerk) 1395,-

VIDEO-GENIE 3008

(mit Kleinschreibmodul, 10er Tastatur u. Cassett.-Anschl.) 1595,-

Zubehör:

Expansion Interface mit 32k Speichererweiterung 1275,-
5.25" Floppy-Laufwerk mit Gehäuse und Netzteil, 40 Track-Ausführung 995,-
Zweites Laufwerk 40 Track 775,-
Verbindungskabel für 2 Laufwerke 90,-
Kleinschreibmodul für 3003 145,-
RS 232 C Schnittstelle 175,-
S 100 Adapter 295,-

MZ 80 K (48k RAM) 2195,-

Interface Box 525,-
DIN-Tastatur 375,-

SANYO 12" Monitor

grün, 18 Mhz für augenschonende Dauerarbeit, blendfrei 698,-

SANYO 12" Monitor

grün, 25 Mhz, angeätzte Bildröhre, für höchste Ansprüche 898,-

BMC 12" Monitor

grün, 18 Mhz, reflexionsarmer Bildschirm 575,-

Sonderposten!

Original BASF 5.25" Laufwerk 6106, fabrikneu, originalverpackt, ideal als Zweitlaufwerk 595,-

16k dyn. RAM 4116, 200ns, orig. MITSUBISHI, allererste Wahl, stückgeprüft!
8 Stück 59,60
16 Stück 115,30

nascom

Lucas Logic

Neu!

Liebe Leser,

Anfang 1981 wurde die englische Firma NASCOM-Microcomputers übernommen. In Wedgnock werden seit Frühjahr in modernsten Produktionsstätten des Konzerns LUCAS Ltd. alle NASCOM-Systeme und Peripheriegeräte in großen Stückzahlen produziert und weiterentwickelt.

MK-Systemtechnik ist der autorisierte Generalvertreter in Deutschland, Österreich und der Schweiz. Nur bei uns bekommen Sie Original NASCOM-Teile aus der neuen Produktion. Nur wir können für Sie Garantieleistungen an den neuen Geräten durchführen!

NASCOM 1

Einplatinencomputer für »stand-alone« und OEM-Anwendungen · Z 80 CPU · 2k statisches RAM · Neues 2k NAS-SYS 3 Betriebssystem · Hochwertiges Keyboard mit 58 Magnettasten · Video-Interface mit 16 x 48 Zeichen · Groß/Kleinschreibung mit Unterlängen · 128 ASCII-Zeichen · BAS-Ausgang · UHF-Ausgang · RS 232 C und 20mA Serien-Schnittstelle · Cassetten-Interface · Mini-DCRs anschließbar · 16 Ein/Ausgabeleitungen (PIO) · 2 Vektorinterrupts · Erweiterbar auf 256k RAM/ROM

Bausatz 855,- Fertiggerät **985,-** Netzteil hierzu (Fertiggerät) **210,-**
Für OEMs auch ohne Keyboard und in Sonderbestückungen lieferbar.

NASCOM II

Dieses Gerät wird häufig als Entwicklungssystem eingesetzt, z. B. um Software für den NASCOM I als OEM-Modul zu erstellen. Es ist voll softwarekompatibel mit dem NASCOM I, hat hardwareseitig jedoch folgende zusätzliche Vorzüge:

· Z 80 CPU 4 Mhz Taktfrequenz · Bis zu 8k RAM (4118) oder EPROM auf der Grundplatine · 8k BASIC in einem 8k x 8 organis. ROM Typ 36000 (MOSTEK) · Voll gepufferter BUS

NASCOM II Bausatz, 8k stat. RAM, BASIC, 2k Monitor 1665,-
NASCOM II Bausatz, 16k dyn. RAM, BASIC, 2k Monitor 1995,-

Fertiggeräte NASCOM II bitte anfragen.

Floppy Disk 1749,-

5" Floppy, Fertiggerät mit DOS, BASIC, Macroassembler, Debug, Texteditor, für NASCOM 1 oder NASCOM 2 — 1 Jahr Software-Pflege kostenlos.

5" Floppy (s.o.) mit Gehäuse und Netzteil für 2 Laufwerke 2144,-
NAS-SYS-Assembler 4k 299,-
NAS-SYS-Disassembler 199,-
NAS-SYS-Debug 75,-

Schach für NASCOM 1/2 128,-

Schachgrafik ROM für NASCOM 2 98,-
Grafik-Zusatzkarte f. NASCOM 1 198,-

NASCOM-JOURNAL

Zeitschrift für den ernsthaften NASCOM-Anwender. Zahlreiche Programmier-Beispiele, Anregungen zur Hardware etc.

Im NASCOM-JOURNAL stellen wir auch laufend neue Produkte vor!

Jahresabonnement 1981 48,-
Jahrgang 1980 komplett (nur solange Vorrat reicht) 39,-

NASCOM 1

Alle Preise sind in DM und schließen die Mehrwertsteuer ein. Versand per Nachnahme oder nach Vorausrechnung. Preisänderung, Irrtum und Zwischenverkauf vorbehalten

MK-Systemtechnik
Pater-Mayer-Straße 6
6728 Germersheim/Rhein
Telefon (O 72 74) 27 56
Telex 0453500 mks d

MK-Systemtechnik
Kriegsstraße 164
7500 Karlsruhe
Telefon (O 72 1) 2 92 43

MK-Systemtechnik
Platfenberg 4
5650 Solingen 1
Telefon (O 21 22) 4 72 67