nascom journal

Zeitschrift für Anwender des NASCOM 1 oder NASCOM 2

2. Jahrgang Oktober 1981 · Ausgabe 10

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,
endlich haben wir das Oktoberheft fertigstellen können, und hoffen, daß die recht vielfältigen Artikel für jeden etwas Interessantes bieten. Die versprochenen BASIC Spiele konnten leider nicht mehr aufgenommen werden, da sie so umfangreich sind und unser Heft jetzt schon den geplanten Umfang überschritten hat. Wir werden die Spiele im Nov./Dez. Doppelheft nachliefern. Dazu werden wir einen BASIC Compiler vorstellen, einen Selbstbau-Plotter, 2716 Programmierer mit Software, die (nun vollständige) Software für den MDCR und ein Platinen Lay-Out für ein verbessertes Interface dazu. Weiterhin erfahren Sie Näheres über NASSYS 3 und natürlich über die Entwicklung des FORTH.(Der "FORTH-Rundlauf" hat übrigens regen Zuspruch gefunden, was man von der Cassettentausch Aktion nicht behaupten kann. Wenn wir bis zum nächsten Heft nicht noch einige Interessenten finden (bis jetzt sind es erst 3 !) werden wir die Aktion einstellen müssen.)
Um unser Doppelheft auch wirklich zu einer "Weihnachtsausgabe" machen zu können, bitten wir weiterhin um rege Leserbeteiligung. An dieser Stelle darf ich auch gleich allen Einsendern danken, die mit ihren Beiträgen dem Journal zu einem wirklich guten Niveau verholfen haben.
Die NASSYS3 Aktion hat auch schon einige Interessenten gefunden. Für sie ist weiter unten eine kleine "Einbauanleitung" abgedruckt.
Nun noch ein Aufruf an alle, die an Kontakten mit NASCOM Benutzern aus ihrer Gegend interessiert sind. Senden Sie Namen, Anschrift und benutztes System bzw. besondere Interessen an untenstehende Adresse. Diese Informationen werden wir veröffentlichen und hoffen, damit zur Förderung lokaler Kontakte von NASCOM Benutzern beizutragen.
Genug geflirtet - Viel Spaß mit diesem Heft
Ihr Günter Böhm

Kontaktadresse für Interessenten an lokalen "Computerbekanntschaften":
Werner Öhring
[redacted]
[redacted] München
Tel.[redacted]

Einbauanleitung für NASSYS 3 in NASCOM 1

Die einfachste Lösung wäre, Sie lassen sich das Betriebssystem in 2 2708 schießen. Wenn das zu viel Mühe macht, hier eine kleine Hardwareänderung:
Beim NASSYS3 Eprom werden die Pins 19, 20 und 21 herausgebogen. Pin 21 und 24 werden mit einem kurzen Drahtstück verbunden. Nun muß Pin 19 durch eine Leitung mit A10 der CPU verbunden werden. (rechts neben IC 36 bei Pin 16 ist eine Durchkontaktierung leicht zugänglich).
Nun muß noch der $\overline{CS}$-Eingang Pin 20 über ein Gatter angeschlossen werden, so daß er über den Bereich von 2K aktiv ist. Dazu kann man ein 74LS00 direkt auf IC 44 setzen. Die Versorgungsspannung und Masse (Pin 14 und 7) werden direkt mit IC 44 verlötet. Pin 8, 12 und 13 des 74LS00 werden miteinander verbunden (hochbiegen und mit Draht verlöten) Pin 9 und 10 jeweils mit Pin 8 und 11 von IC 44 verlöten. Alle übrigen Beinchen kann man abzwicken. Pin 11 des Gatters wird nun mit $\overline{CS}$ durch ein Stück Litze verbunden.
Wenn man sich den T4 Monitor in ein 2716 brennt, kann man ihn in den freien Sockel setzen. Durch einen Umschalter, der die Leitung von IC 7400 Pin 11 an an den $\overline{CS}$ Eingängen der Monitoren umschaltet, kann man ohne Umschaltkarte zwischen zwei Betriebssystemen wählen. Ich werde es jedenfalls so machen.

. . .

FORTH für den NASCOM
Teil 3 - von Günter Kreidl

Leider haben sich ins Listing Fehler eingeschlichen: die Zeilen 5480 u. 5495 sind zu ändern:

```
5480  LD  DE, -5
5495  LD  DE, -9
```

Ein weiterer Fehler betrifft die Funktion: ">", Sie erwartet die Argumente genau in der umgekehrten Reihenfolge auf dem Stack als es in FORTH üblich ist, und schließt außerdem den Fall A = B nicht aus.
Hier ist die Korrektur einfach Zeile 6038 wird geändert in:
JR NC,TESTE
Die Funktion gibt nun "Wahr" (= FFFFH) auf den Stack, wenn T-1 größer als T ist, sonst 0.
Die Funktion "ROT" ist zwar richtig beschrieben und programmiert, aber leider genau umgekehrt definiert (Abwärtsrotation statt Aufwärtsrotation) als es sonst in FORTH üblich ist. Um eine Sprachverwirrung auszuschalten, werden wir auch hier eine Änderung vornehmen. Es ist die Reihenfolge der PUSH-Befehle zu verändern in:
6025 PUSH DE
6026 PUSH HL
6027 PUSH BC
Die folgenden Änderungen sollen keine Fehler korrigieren, sondern werden uns später beim Programmieren in FORTH sehr nützlich sein. Sie verbessern die Zusammenarbeit mit dem Betriebssystem und gestalten sie zugleich variabler. Wir nehmen diese Änderungen direkt in Maschinensprache vor. Zunächst verwandeln wir die Constante "MEMORY" in eine Variable. Wir tragen ab 1171H statt 5C 11 ein: 75 11.
Bevor wir die nächste Veränderung vornehmen führen wir zunächst einen Kaltstart des FORTH-Systems aus und kehren dann sofort mit RESET oder dem Befehl "91 NAS-SYS" ins Betriebssystem zurück. Nun fügen wir eine zusätzliche Routine und ihren Namen direkt in Maschinensprache ein mit Hilfe des MODIFY-Kommandos:
16AB AD 16 E1 22 0C 0C DF ,M
16B3 E5 C3 3E 10,
1E4A 06 ,M ,O ,D ,I ,F ,Y AB
1E52 16.
Mit "MODIFY" kann das M-Kommando des Betriebssystems aufgerufen werden. Der Startwert wird auf dem Stack übergeben, bei der Rückkehr wird die nächste freie Speicheradresse ebenfalls auf dem Stack übergeben. (In genau dieser Weise funktioniert das natürlich nur unter NAS-SYS. T2/T4-Anwender müssen sich das etwas umschreiben.) Bevor wir nun das System wieder starten können, müssen wir noch die Werte der beiden Variablen "CODE" und "NAMES" verändern, damit die neue Funktion vom Interpreter erkannt und ausgeführt wird. Wir benutzen wieder das M Kommando und tragen die neuen Werte ein:
1293 B7.
128F 4A.
Wir starten nun den Interpreter mit E1000 und werden dann dieses Tiny-FORTH zunächst dazu benutzen, das System selbst auszubauen und Schritt für Schritt eine komfortable Programmiersprache zu entwickeln.
Wir beginnen mit den bereits bekannten Funktionen CONSTANT und VARIABLE. Es sind definierende Funktionen, die einen Namen in das Dictionary eintragen und ihnen Routinen im Code zuweisen, die einen Wert (CONSTANT) bzw eine Adresse (VARIABLE) auf den Stack geben.

```
: VARIABLE   GETWORD     ENTER
             VARBL CMPLW CMPLW ;
: CONSTANT   GETWORD     ENTER
             CONS  CMPLW CMPLW ;
4755 CONSTANT CODEADR
4158 CONSTANT NEXTADR
```

Diese beiden Konstanten brauchen wir gleich!
Die folgende Funktion erklärt sich selbst:

```
: CR 13 COUT ;
```

Wir brauchen sie bereits bei der nächsten Funktion, ohne die man sich gar nicht erst an größere Programmierversuche herantrauen sollte:

```
: DEL  NAMES PEEKW DUP PRINTS CR
       FIRST + DUP PEEKW CODEADR POKEW
       INC   INC   NAMES   POKEW       ;
```

DEL löscht die jeweils zuletzt compilierte Funktion sowie ihren Namen, der dabei angezeigt wird. CODEADR gibt dabei die Adresse der Variablen CODE, die den nächsten freien Speicherplatz für den FORTH-Code enthält. Mit diesem Trick machen wir uns eine Variable zugänglich, die zwar im Code, nicht aber im Dictionary enthalten ist.
Die folgende Funktion, MCODE, ist wiederum definierend. Sie erlaubt den Einbau von Routinen in Maschinensprache in den FORTH-Code:

```
: MCODE  GETWORD ENTER  CODEADR PEEKW
         DUP INC INC DUP ROT POKEW MODIFY
         DUP 195 SWAP POKEB  INC DUP
         NEXTADR SWAP POKEW INC INC
           CODEADR POKEW
```

Wir wollen MCODE nun gleich benutzen, um eine Standard-FORTH-Routine in Maschinencode zu schreiben:
MCODE OVER
1743 E1 D1 D5 E5 D5.
OVER ist ein Befehl zur Stackmanipulation wie DUP und ROT. Die Funktion kopiert den 2.

Wert auf dem Stack obenauf. Wie man sieht, ist bei der Programmierung mit MCODE kein Rückkehrbefehl erforderlich; der Rücksprung zum Interpreter (NEXT) wird mit Hilfe der Konstante NEXTADR (= 103EH) von MCODE an das Maschinenprogramm angehängt.

Will man von FORTH aus Drucker oder Casset= tenrecorder betreiben, so kann man sich der Ein- und Ausgaberoutinen des Betriebssystems bedienen. Es empfiehlt sich jedoch, auch die Routinen CIN und COUT zu verbessern. Dazu definieren wir zunächst einige Hilfsfunktio= nen:

```
MCODE  NORMIN
XXXX CF 6F 26 00 E5.
MCODE  BLINK
XXXX DF 7B 6F 26 00 E5.
MCODE  NORMOUT
XXXX C1 79 F7.
```

Die Adressen XXXX werden von MCODE angegeben und nicht vom Benutzer. Die folgende Funk= tion verlangt einen Namen als Argument. Wird dieser Namen im Dictionary gefunden, so wird die Startadresse des zugehörigen Codes auf den Stack gegeben, andernfalls wird der Name ausgedruckt:

```
: CADR  GETWORD NAMES PEEKW LOOKUP
        EQZ IF PRINTS THEN          ;
```

Wir definieren folgende Variablen, wobei mit CADR der Anfangswert berechnet wird:

```
CADR NORMOUT VARIABLE OUTVAR
CADR NORMIN  VARIABLE INVAR
```

Wir werden nun neue Ein-/Ausgabefunktionen definieren, die an Stelle von CIN und COUT treten sollen. Sie rufen diejenige Ein- oder Ausgaberoutine auf, deren Startadresse in den Variablen INVAR bzw. OUTVAR gespeichert sind.

```
: VAREX PEEKW EXECUTE ;
: COUT2 OUTVAR VAREX ;
```

Im Interpretermodus geben wir nun folgenden Befehl ein:

```
CADR COUT2 CADR COUT 8 MOVEBYTES DEL
```

COUT2 kann nun unter der Adresse von COUT aufgerufen werden und wird deshalb gelöscht. Ebenso verfahren wir mit CIN; wir compilie= ren wiederum vorläufig CIN2, schieben sie an die Stelle von CIN, und löschen sie dann wieder, da man sie mit CIN aufrufen kann:.

```
:/.IN2  INVAR VAREX ;
CADR CIN2 CADR CIN 8 MOVEBYTES DEL
```

Nachdem wir nun so ausgiebig an dem System herumgebastelt haben, wollen wir nun einige simple FORTH-Funktionen compilieren:

```
MCODE  UNDER
XXXX E1 D1 E5 D5 E5.
```

kopiert den obersten Stackwert an die dritte Stackposition.

```
: <    > EQZ ;
: /    /MOD POP ;
: MOD  /MOD SWAP POP ;
: MAX OVER OVER > IF POP ELSE SWAP POP THEN ;
: MIN OVER OVER < IF POP ELSE SWAP POP THEN ;
```

Diese trivialen Funktionen erwarten jeweils 2 Argumente auf dem Stack.

Die nächsten Funktionen sind etwas komplexer und greifen direkt auf den Speicher zu:

```
: PRINTM  PEEKW CONBXA PRINTS ;
: MEM+   DUP PEEKW ROT + SWAP POKEW ;
: FILL  SWAP ONE FOR  UNDER OVER POKEB
        INC SWAP LOOP  POP POP ;
: ERASE   Ø FILL ;
: BLANKS  32 FILL ;
```

PRINTM druckt den Wert an der Adresse (T) aus (dezimal). MEM+ addiert den Wert von T-1 an die Adresse (T). FILL schreibt T-1 Bytes vom Wert T an die Adresse (T). ERASE und BLANKS schreiben Nullen bzw. Blanks in den Speicher. MESSAGE ist dagegen wieder eine definierende Funktion; sie erlaubt das Ein= fügen von Strings in das Programm:

```
: MESSAGE  GETWORD ENTER  VARBL CMPLW
    REPEAT  CIN DUP COUT 34 EQ UNTIL
            DUP 8 EQ IF POP DEC
                     ELSE OVER POKEB INC
                     THEN LOOP
            POP ZERO OVER POKEB INC
            CODEADR POKEW ;
```

z.B. eine Error-Meldung:

```
MESSAGE ERROR
Error"
```

Die Eingabe wird mit einem Anführungszeichen beendet; sie kann beliebig lang sein und auch Leerzeichen enthalten. Mit TYPE können derartige Strings ausgegeben werden.

```
: TYPE  REPEAT DUP PEEKB DUP EQZ UNTIL
                COUT INC LOOP POP POP ;
```

TYPE erwartet eine Adresse auf dem Stack, der Name des Strings muß ihr vorausgehen:

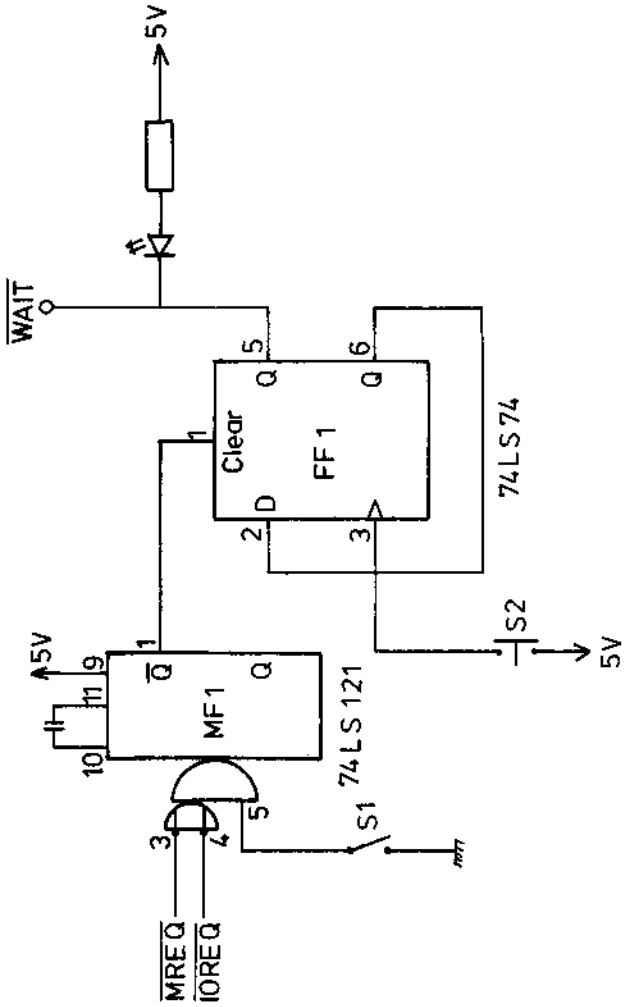
```
ERROR TYPE
Error
```

Da der Platz im NASCOM-Journal beschränkt ist muß ich hier schließen. Im nächsten Heft folgen dann ein Bildschirm-Editor und die Cassettenroutinen, sowie ein erstes kleines Programm in FORTH, eine neue Version eines "alten" Computerspiels: LIFE in einem "ge= "schlossenen Universum".

HARDWARE TIPS
VON Otto Fößel

Der nachfolgende Artikel von Otto Fößel führt den Gedanken, die CPU durch $\overline{WAIT}$ anzuhalten, welchen Josef Zeller im letzten Heft geäußert hat, konsequent weiter. Durch die vorgeschlagene Schaltung ist es möglich, die CPU bei jedem $\overline{MREQ}$ oder $\overline{IOREQ}$ - Zyklus durch den $\overline{WAIT}$ anzuhalten. Man sollte noch anmerken, daß während des $\overline{WAIT}$ - Zyklus von der CPU kein Refresh ausgeführt wird, dynamische RAMs also ihre Information verlieren.

Die nachfolgende Beschreibung soll all denjenigen helfen, die ratlos vor ihrer nicht funktionierenden Platine stehen und nicht wissen, wie und wo man jetzt mit der Fehlersuche beginnen könnte. Es wird gezeigt, wie man alleine mit einem Vielfachmeßgerät, der kleinen Schaltung im Bild und ein bißchen Überlegung sehr viele Fehler finden kann.
Als Erstes wird sich eine optische Kontrolle anbieten: sind alle Bauteile dort, wo sie hingehören, sind alle ICs richtig eingesetzt, alle IC-Beinchen im Sockel, alle notwendigen Durchkontaktierungen und Lötbrücken gemacht, befinden sich irgendwo noch Reste von Lötzinn, die Kurzschlüsse verursachen könnten usw. Dann mit dem Voltmeter: sind die Betriebsspannungen in Ordnung, und kommt die Betriebsspannung auch an allen ICs an? Wenn man bis hierher noch nichts Ungewöhnliches feststellen konnte, hilft erst einmal Nachdenken. Woran kann es liegen, daß sich nach dem Reset nicht das Erwartete tut. 1.) Es könnten Leitungsbahnen unterbrochen oder schlecht durchkontaktiert sein 2.) Kurzschlüsse mit anderen Leitungsbahnen 3.) defekte ICs bzw. Bauteile. Die Möglichkeit von dynamischen Fehlern soll hier nicht in Betracht gezogen werden, da sie erstens viel seltener vorkommen und mit einfachen Mitteln sowieso nicht behoben werden können. Außerdem soll davon ausgegangen werden, daß die Systemsoftware (Betriebsprogramm) in Ordnung ist.
Bei dem Durcheinander von Leitungsbahnen wird man es schnell aufgeben, mit dem Ohmmeter alle Leitungen zu verfolgen bzw. Kurzschlüsse mit allen möglichen Nachbarleitungen zu suchen. Auch das Testen der ICs scheint auf dem ersten Blick mit dem Vielfachinstrument nicht möglich. An diesem Punkt soll deshalb die kleine Schaltung (siehe Bild) zum Einsatz kommen.
Die Schaltung wird folgendes bewirken: der Prozessor, der mit einer Taktfrequenz von 2 MHz arbeitet und ein ruhiges Beobachten der Signale unmöglich macht, soll in den entscheidenden Augenblicken angehalten werden, so daß alle Signale, die irgendwo auf einer Leitung liegen, eingefroren werden und dann in aller Ruhe mit dem Vielfachmeßinstrument überprüft werden können.
Dieses Anhalten geschieht mit dem Anlegen eines Null-Pegels an den Eingang "$\overline{WAIT}$" des Mikroprozessors bei jedem $\overline{MREQ}$ oder $\overline{I/O-REQ}$. Erzeugt wird dieses Signal vom D-FF 1 durch Rücksetzen am CLEAR-Eingang. Dieser CLEAR Eingang bekommt dafür vom Monoflop MF 1 bei jedem $\overline{MREQ}$ oder $\overline{IOREQ}$ einen kurzen Impuls. Durch Schließen von S1 wird dieser Impuls verhindert und folglich auch kein $\overline{WAIT}$ Signal an FF1 erzeugt.
Möchte man jetzt, daß der Mikroprozessor genau einen Schritt weiterarbeitet, d.h. bis zum nächsten Speicher- oder I/O-Zugriff, so wird die Taste S2 kutzgeschlossen. Sie erzeugt ein Taktsignal an FF1 und schaltet somit die log. "1" vom Eingang zum Ausgang Q durch, d.h. $\overline{WAIT}$ = 1. Aber schon beim nächsten $\overline{MREQ}$ wird wieder ein CLEAR-Impuls erzeugt und der Prozessor über FF1 erneut gestoppt.
Der erste Vorteil dieser Schaltung ist ihre Einfachheit und folglich auch der problemlose Nachbau . Zweitens kann sie ebenso einfach an den Mikrocomputer angeschlossen werden, indem man die drei Leitungen $\overline{MREQ}$, $\overline{IOREQ}$ und $\overline{WAIT}$ anzapft. Einfacher geht es nicht!
Aber Jetzt zum Messen. Wie können Leitungsunterbrechungen und Kurzschlüsse mit dem Voltmeter erkannt werden? Kurzschlüsse werden wir immer erkennen, wenn auf der Leitung eine log."1" liegen soll, aber durch eben einen Kurzschluß mit einer Nachbarleitung auf eine Spannung kleiner 2V gezogen wird. Ähnlich ist es bei Leitungsunterbrechungen. Hier wird der Eingang an dem das Signal ankommen soll infolge der Unterbrechung "in der Luft hängen". Da wir es fast ausschließlich mit TTL- Eingängen zu tun haben, wird sich dort eine Spannung zwischen 1 und 2 Volt einstellen. Also auf Jeden Fall kein



sauberer TTL-Pegel. Die Bruchstelle kann dann mit dem Ohmmeter noch genau lokalisiert werden. Sollte die Verbindung dennoch in Ordnung sein, so würde dies auf einen Defekt am Ausgang des treibenden ICs bzw. am Eingang des angeschlossenen ICs hindeuten, was durch wahlweises Herausnehmen eines ICs noch genau bestimmt werden kann.

Ein kurzes Beispiel an einem NASCOM 1 mit dem Betriebssystem NAS-SYS mag diese Fehlersuche noch ein wenig verdeutlichen. Die Hilfsschaltung ist angeschlossen, d.h. der Prozessor wurde inzwischen angehalten und nach dem Reset und einem ersten Tastendruck auf S2 wird genau dort gestoppt, wo der Prozessor sein erstes Befehlsbyte vom Monitor-ROM holt. Folglich wird auf dem Adreßbus ØØØØ anliegen, auf dem Datenbus wird vom ROM her die 31H liegen, die $\overline{\text{READ}}$-Leitung muß log"Ø" aufweisen, die $\overline{\text{WRITE}}$-Leitung log"1", die $\overline{\text{MREQ}}$ auf log"Ø" und die $\overline{\text{I/ORQ}}$-Leitung auf log"1". Diese Pegel sind dorthin zu verfolgen, wo sie evtl. benötigt werden.

Ist soweit alles in Ordnung wird kurz S2 gedrückt. Auf dem Adreßbus muß ØØØ1 erscheinen und auf dem Datenbus, laut Monitorlisting, ØØ. Durch dieses Vorgehen wird also der Mikroprozessor sofort beim nächsten Speicherzugriff wieder angehalten.

Aber auch größere Sprünge, nämlich bis zum nächsten I/O-Befehl, die ebenfalls im Monitorprogramm vorkommen, sind möglich. Dafür wird einfach die $\overline{\text{MREQ}}$-Leitung zur Zusatzschaltung unterbrochen und ein $\overline{\text{WAIT}}$-Signal kann folglich erst beim nächsten $\overline{\text{IORQ}}$ erzeugt werden.

Zusätzliche Überlegungen sind dann erforderlich, wenn die Signale auf ihrem Weg zu den Adressaten noch verschiedene Funktionsgatter durchlaufen (z.B. UND-, ODER-Gatter, Multiplexer für IC-Auswahl in der $\overline{\text{RD}}$- oder $\overline{\text{WR}}$ Leitung usw.). Dies ermöglicht dann gleichzeitig, diese verschiedenen Gatter auf ihre Funktionsfähigkeit zu überprüfen.

Damit sind aber die Möglichkeiten der Fehlersuche mit diesen Hilfsmitteln bei weitem noch nicht erschöpft. Sollte Interesse bestehen, kann ich noch ausführen, wie sich z. B. der Einsatz eines selbstprogrammierten EPROMs an Stelle des Monitor-ROMs lohnen kann. (Mit den drei Befehlen LD A,FF; ST A ØC00; LD A,ØC00 kann die Funktionstüchtigkeit des wichtigen RAM-Speichers für Monitorzwecke überprüft werden). EPROMs dafür könnte ich evtl. zur Verfügung stellen.

Ein weiteres Einsatzgebiet wäre das Testen von selbstentwickelten Zusatzschaltungen, bzw. Interfaceschaltungen bei funktionierendem Grundsystem. Dafür müßte das treibende Programm an den entscheidenden Stellen angehalten werden, um dann die Signale zu der neuen Schaltung im statischen Zustand überprüfen zu können. Dafür eignen sich nicht die Monitorhilfsmittel (Break, Single Step). Auch hier kann mit besagter Zusatzschaltung und einem kleinen Trick das laufende Programm gezielt gestoppt werden.

GRAPHICS-GENERATOR

VON Günter Böhm

Kürzlich habe ich einen Graphics Generator für den NASCOM 2 erworben, um meinen NASCOM 1 um die Graphic Möglichkeiten zu erweitern. Dabei ist mir aufgefallen, daß dieser Generator bis auf die Pins 19 und 21 pinkompatibel mit den Eproms auf meiner Grundplatine ist. So habe ich den Generator einfach einmal in einen freien Epromsockel meiner Erweiterung gesteckt und mit folgendem Programm die Nullen und Einsen durch den Drucker in schwarze und weiße "Punkte" verwandelt. Durch Start bei C8Ø und als zweiten Parameter Eingabe der Startadresse im EPROM wird ein Zeichen von 8 Punkten Breite und 16 Punkten Höhe abgebildet.

Wenn ein Charactergenerator aber nur ein einfaches EPROM ist, warum sollte man sich dann nicht seinen eigenen Charactergenerator programmieren?

Da, wie bereits erwähnt, ein Character aus 8 Spalten und 16 Reihen besteht, sollte man sich für selbstprogrammierte Zeichen zunächst ein "Schnittmuster" aufzeichnen, und darin die Punkte eintragen, die gesetzt werden sollen. Dann wird ein beliebiger Bereich im RAM gewählt und von links oben beginnend Zeile für Zeile des Characters als Byte eingegeben, wobei jeder gesetzte Punkt als 1 und jeder nichgesetzte als Ø betrachtet wird.

Die Zeile . . .. . wird als 1Ø1Ø11Ø1 betrachtet und als AD gespeichert. Entsprechend verfährt man mit den folgenden 15 Zeilen. Daß ein Zeichen exakt 16 Bytes benötigt, macht die Speicherung übersichtlicher. Beginnt Ihr "Character-RAM" z.B. bei ØD ØØ, so folgt das nächste Zeichen in ØD1Ø, dann ØD2Ø etc. Man verliert also nicht den Überblick.

Wenn Sie eine Reihe von Zeichen programmiert haben, müssen sie in ein EPROM 2716 "geschossen" werden. (Maximal sind 128 Zeichen möglich). Nach Einstecken des EPROMS in den Graphics Sockel des NASCOM 2 können Ihre Zeichen abgebildet werden.

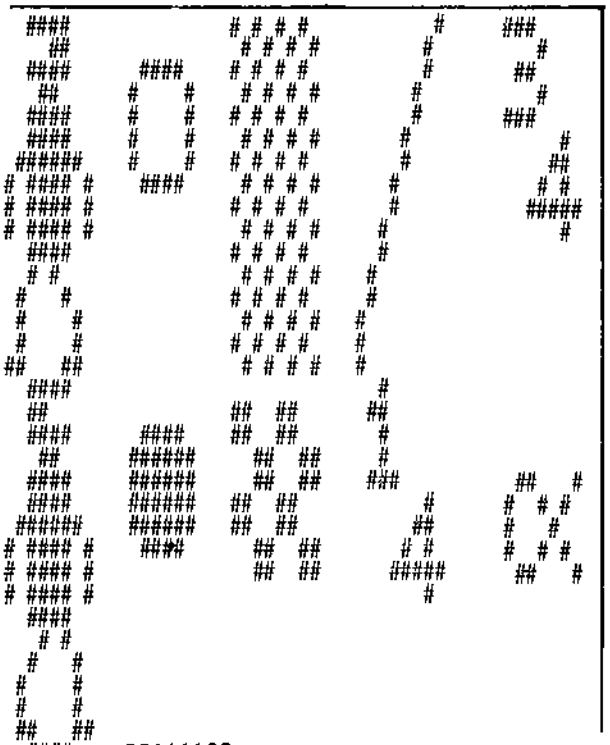
NASCOM 1 Besitzer müssen leider in der Lage sein, ein 2732 EPROM programmieren zu können, um mit Hilfe der Graphic Karte von Herrn Ploß eigene Zeichen herstellen zu können. Die Fassung des NASCOM 1 ist leider für 2716 nicht pinkompatibel.

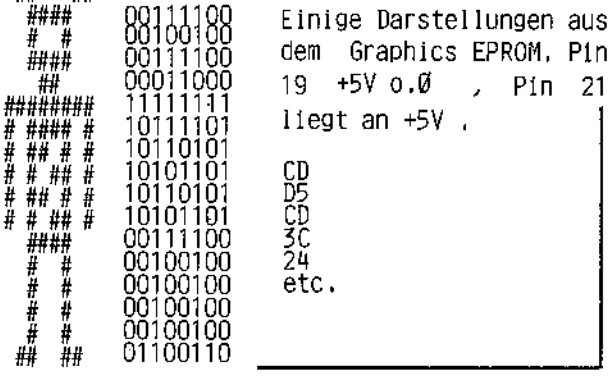
Es wurde aber schon einmal eine Graphic Karte von MKS angeboten, vielleicht kann die ein selbstprogrammiertes EPROM aufnehmen? Ich bin darüber leider nicht informiert.

Vor dem Brennen des EPROMs empfiehlt es sich, mit obigem Druckprogramm die selbsterstellten Zeichen auszudrucken. Man erhält einen Eindruck Ihrer Gestalt und kann Fehler verbessern.

Drücken von NEW LINE bildet übrigens den nächsten Character ab, eine beliebige Taste läßt zum Monitor zurückspringen.

```
ØC8Ø 2A ØE ØC ØE 1Ø Ø6 Ø8 7E 7A
ØC88 CB 27 38 Ø9 F5 3E 2Ø CD E7
ØC9Ø 5D ØØ F1 18 Ø7 F5 3E 23 5F
ØC98 CD 5D ØØ F1 1Ø EA 23 3E 1A
ØCAØ ØA CD 5D ØØ ØD 2Ø DE CD B8
ØCA8 69 ØØ 3Ø FB FE 1F 28 D3 6Ø
ØCBØ CF
```





```
00111100
01100110
00111100
00011000
11111111
10111101
10110101
10101101
10110101
10101101
00111100
00100100
00100100
00100100
00100100
01100110
```

Einige Darstellungen aus dem Graphics EPROM. Pin 19 +5V o.Ø , Pin 21 liegt an +5V .

```
CD
D5
CD
3C
24
etc.
```

TYPENRADDRUCKER OLYMPIA ESW 100

von Günter Böhm

Der Anschluß der ESW 100 KSR von OLYMPIA ist absolut kein Hexenwerk und ohne große Hardwareänderungen möglich. Die einfachste Lösung ist wohl der Anschluß an die TTY (20 mA) Schnittstelle.
Dazu sind folgende Pins zu verbinden:

ESW 100 KSR	NASCOM1 SK2
Pin 9 TTYT-	Pin 8 Masse
Pin 10 TTYT+	Pin 5 KBD+
Pin 21 TTYR+	Pin 12 PTR+
Pin 25 TTYR-	Pin 11 PTR-

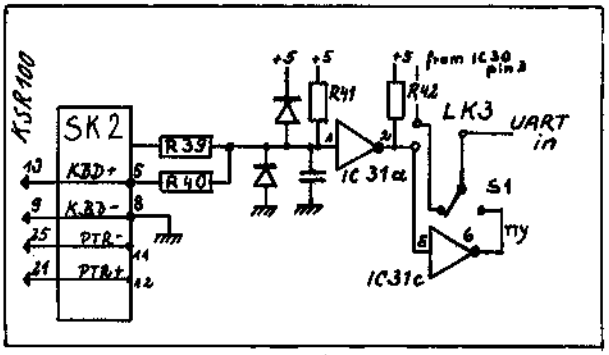
Weiterhin sind am Drucker die Pins 2 und 11 und die Pins 3 und 18 jeweils mit einander zu verbinden.
Um das CTS (Clear to send) Signal zu unterdrücken, muß auf der Platine des KSR ein Widerstand 2,7 K eingesetzt werden. Seine Position ist aus der mitgelieferten Schaltung ersichtlich.
Um den UART auf die 2 Stopbits umzustellen, die der KSR braucht, muß Link 2 auf der NASCOM Platine von Masse getrennt werden.
Man könnte aber ebenso den KSR auf 1 Stopbit umstellen; dazu müßten die Dioden D26 und D27 aus der KSR Platine entfernt werden.
Soweit funktioniert der Typenraddrucker schon als Empfänger. Um aber sein Keyboard auch als Sender benutzen zu können, muß das Signal des Keyboards noch invertiert werden.
Glücklicherweise ist dafür noch ein Gatter auf der NASCOM Platine frei. Biegen Sie einfach die Pins 5 und 6 aus dem IC 31 heraus und legen das Gatter so zwischen Link 3 des NASCOM, daß das Eingangssignal invertiert wird. (siehe Schaltung).
Um den 1760 Hz Takt des NASCOM TTY Clock Generators verwenden zu können, muß man eine Drahtbrücke auf der KSR PLatine verändern. (siehe Schaltplan). Besser wäre es allerdings in Hinblick auf die Übertragungsgeschwindigkeit, den NASCOM Takt durch Verwendung eines kleineren Kondensators am 555 zu verändern und so auf max. 17 Zeichen pro Sekunde zu erhöhen.

Die Änderungen an Link 2 und 3 und die unterschiedlichen Clocks sollten natürlich durch Schalter oder Relais für Cassette und Drucker umschaltbar gestaltet werden.
Für eventuelle Fragen zu diesem Thema stehe ich gerne zur Verfügung.

Da der KSR nicht alle Zeichen abbilden kann, die(vor allem in BASIC) benötigt werden , wurde folgendes kleine Programm entwickelt, das die Zeichen entsprechend umformt.Sie werden folgendermaßen abgebildet:

↑	↑
Ø	Ø
>	≥
<	≤
@	a)

INIT STARTET das Programm als Ersatz für die CRT Routine, START2 gibt ohne CRT aus; für die Verwendung mit dem ASM Assembler wird bei START3 begonnen (weil sonst zwei Leerzeilen ausgegeben werden). Für BASIC läßt sich das Programm mit DOKE, 3147,3207 zum Listen verwenden; es ist aber bis auf die Startadressen voll verschieblich.




```
                0010 ;OUTPUT FUER KSR
0C80            0020        ORG  #C80
013B            0030 CRT    EQU  #13B
0C4B            0040 $CRT   EQU  #C4B
CA49            0050 DELAY  EQU  #CA49
005E            0060 OUT    EQU  #05E
                0070 ;
0C80 21870C     0080 INIT   LD   HL,START1
0C83 224B0C     0090        LD   ($CRT),HL
0C86 CF         0100        RST  8
                0110 ;
0C87 CD3B01     0120 START1 CALL CRT
0C8A FE1F       0130 START2 CP   #1F
0C8C 2009       0140        JR   NZ BACKS
0C8E 3E0A       0150        LD   A,#0A
0C90 CD5E00     0160        CALL OUT
0C93 CD49CA     0170        CALL DELAY
0C96 C9         0180        RET
0C97 00 00      0190 BACKS
0C99 00 00      0200        NOP
0C9B 00 00      0210
0C9D FE1D       0220        CP   #1D
0C9F 2005       0230        JR   NZ POT
0CA1 3E08       0240        LD   A,8
0CA3 C35E00     0250        JP   OUT
0CA6 FE5E       0260 POT    CP   #5E
0CA8 200D       0270        JR   NZ NULL
0CAA CD5E00     0280        CALL OUT
0CAD 3E60       0290        LD   A,#60
0CAF CD5E00     0300        CALL OUT
0CB2 3E31       0310        LD   A,#31
0CB4 C35E00     0320        JP   OUT
0CB7 FE30       0330 NULL   CP   #30
0CB9 200D       0340        JR   NZ KLEIN
0CBB CD5E00     0350        CALL OUT
0CBE 3E08       0360        LD   A,8
0CC0 CD5E00     0370        CALL OUT
```

```
0CC3 3E2F     0380          LD    A,#2F
0CC5 C35E00   0390          JP    OUT
0CC8 FE3C     0400 KLEIN    CP    #3C
0CCA 200A     0410          JR    NZ GROSS
0CCC 3E5E     0420          LD    A,#5E
0CCE CD5E00   0430          CALL  OUT
0CD1 3E2D     0440          LD    A,#2D
0CD3 C35E00   0450          JP    OUT
0CD6 FE3E     0460 GROSS    CP    #3E
0CD8 200A     0470          JR    NZ SCHN
0CDA 3E60     0480          LD    A,#60
0CDC CD5E00   0490          CALL  OUT
0CDF 3E2D     0500          LD    A,#2D
0CE1 C35E00   0510          JP    OUT
0CE4 FE40     0520 SCHN     CP    #40
0CE6 200C     0530          JR    NZ END
0CE8 3E61     0540          LD    A,#61
0CEA CD5E00   0550          CALL  OUT
0CED 3E5E     0560          LD    A,#5E
0CEF CD5E00   0570          CALL  OUT
0CF2 3E29     0580          LD    A,#29
0CF4 C35E00   0590 END      JP    OUT
              0600 ;
              0610 ; START FUER ASM ASS.
0CF7 FE1F     0620 START3   CP    #1F
0CF9 209C     0630          JR    NZ BACKS
0CFB 3E0D     0640          LD    A,#D
0CFD CD5E00   0650          CALL  OUT
0D00 CD49CA   0660          CALL  DELAY
0D03 C9       0670          RET
```

```
0C80 21 87 0C 22 4B 0C CF CD | 55
0C88 3B 01 FE 1F 20 09 3E 0A | 5E
0C90 CD 5E 00 CD 49 CA C9 FE | 6E
0C98 1D 20 0B 3E 08 FE 1D 20 | 6D
0CA0 05 3E 08 C3 5E 00 FE 5E | 74
0CA8 20 0D CD 5E 00 3E 60 CD | 77
0CB0 5E 00 3E 31 C3 5E 00 FE | A8
0CB8 30 20 0D CD 5E 00 3E 08 | 92
0CC0 CD 5E 00 3E 2F C3 5E 00 | 85
0CC8 FE 3C 20 0A 3E 5E CD 5E | FF
0CD0 00 3E 2D C3 5E 00 FE 3E | A4
0CD8 20 0A 3E 60 CD 5E 00 3E | 15
0CE0 2D C3 5E 00 FE 40 20 0C | A4
0CE8 3E 61 CD 5E 00 3E 5E CD | 27
0CF0 5E 00 3E 29 C3 5E 00 FE | E0
0CF8 1F 20 9C 3E 0D CD 5E 00 | 55
0D00 CD 49 CA C9 E5 18 D5 CD | 55
```

»DOKE« PROGRAMM

VON Reinhard Zickwolf

Das folgende Programm wandelt ein Maschinenprogramm, welches von BASIC durch USR(X) aufgerufen werden soll, in das nötige Doke Format um. Die Speicherung der Startadresse des Maschinencodes in USRLOC wird ebenfalls angezeigt.

```
120 CLS
130 SCREEN 15,1:PRINT"DOKE PROGRAMM
140 SCREEN 15,2:PRINT"--------------
150 PRINT"BITTE, DIE HEXZAHLEN ZUSAMMEN";
160 PRINT"HAENGEND EINGEBEN.":PRINT
170 PRINT:INPUT"ANFANGSADRESSE";A$
180 INPUT"ENDADRESSE    ";E$
190 INPUT"STARTADRESSE";S$
200 H$=A$:GOSUB440:A=DE
210 H$=E$:GOSUB440:E=DE
220 LM=E-A+1
230 IF LM<0 THEN PRINT"FEHLER":GOTO170
240 DIM M$(LM):PRINT
250 PRINT"MASCHINENPROGRAMM BYTE FUER BYTE ";
260 PRINT"EINGEBEN":PRINT
270 FOR B=1 TO LM:PRINT B;":";:INPUT M$(B):NEXT
290 REM UMWANDLUNG DES MASCHINENPROGRAMMS
300 CLS
310 PRINT"MASCHINENPROGRAMM IM DOKE-FORMAT
320 PRINT"--------------------------------
330 PRINT
340 H$=S$:GOSUB440:PRINT"DOKE 4100 ,";DO:PRINT
350 H$=A$:GOSUB440:A=DE
360 LG=INT(LM/2)*2:FOR B=1 TO LG STEP 2
370 H$=M$(B+1)+M$(B):GOSUB440
380 PRINT"DOKE";A;",";DO:A=A+2:NEXT
390 IF LM-LG <.5 THEN PRINT:END
400 H$=M$(LM):GOSUB440
410 PRINT"POKE";A;",";DO:PRINT:END
430 REM UMRECHNUNG HEX/DEZ UND DOKE
440 DE=0:S=1:FOR L=LEN(H$)-1 TO 0 STEP -1
450 C=ASC(MID$(H$,S,1)):S=S+1
460 IF C >= 48 AND C <= 57 THEN C=C-48:GOTO490
470 IF C >=65 AND C <=70 THEN C=C-55:GOTO490
480 PRINT"FALSCHE EINGABE":RETURN
490 IF L=0 THEN DE=DE+C:GOTO510
500 SW=1:FOR I=1 TO L:SW=16*SW:NEXT:DE=DE+C*SW
510 NEXT:DO=DE:IF DO >=32768 THEN DO=DO-65536
520 RETURN
```

CLDDOS Teil 4

CLDDOS-SYSTEM
Programme von Gerhard Baier

In den letzten Ausgaben des Nascom-Journals wurden einige Unterprogramme fuer das CLDDOS-System vorgestellt. In dieser Ausgabe werden nun zwei nuetzliche Programme abgedruckt, die jene Unterprogramme verwenden:
WRFILE und RDFILE

Die benoetigten Unterprogramme werden einzeln am Ende des Programms durch "ENCLU" dazugelinkt. Die ganzen Unterprogramme koennten natuerlich auch zu einem einzigen File (wie z.B. SYSLIB.ACM) zusammengebunden werden. Allerdings werden in diesem Fall immer alle in dem ACM-File enthaltenen Routinen zum Programm dazugelinkt, egal ob sie von dem Hauptprogramm aufgerufen werden oder nicht.

```
                CLD 8080/Z80 Assembler
                  03-Mai-81    Seite    1

-------------------------------------------

RDFILE
VERSION 1.0
03-MAI-81
GERHARD BAIER

MIT DIESEM PROGRAMM KOENNEN BELIEBIGE FILES
INS MEMORY GELADEN WERDEN

-------------------------------------------
```

```
2280                            ENCLU   CLDDOS
2280                    RDFILE  ORG     2280H
2280    3E 03                   LD      A,03H
2282    21 D6 22                LD      HL,BREAK
2285    FF 21                   SCALL   .CTLC
                        * BILDSCHIRM LOESCHEN UND TITEL AUSGEBEN
2287    21 E1 22                LD      HL,TEXT1
228A    FF 03                   SCALL   .PRINT
                        * FILENAMEN EINLESEN
228C    3E 10                   LD      A,10H
228E    11 14 23                LD      DE,TEXT2
2291    21 5C 23                LD      HL,FILNAM
2294    CD 6D 23                CALL    $INSTR
                        * MEMORY ADRESSE EINLESEN INS STACK
2297    EB                      EX      DE,HL
2298    3E 04                   LD      A,04H
229A    21 24 23                LD      HL,TEXT3
229D    CD 97 23                CALL    $INHEX
22A0    E5                      PUSH    HL
22A1    E5                      PUSH    HL
22A2    EB                      EX      DE,HL
                        * FILE EROEFFNEN
22A3    11 56 23                LD      DE,DEFBLK
22A6    3E 01                   LD      A,01H
22A8    FF 22                   SCALL   .OPENR
22AA    38 2D                   JR      C,ERROR
                        * FILE EINLESEN
22AC    3E 01                   LD      A,01H
22AE    01 00 FF                LD      BC,0FF00H
22B1    D1                      POP     DE
22B2    FF 04                   SCALL   .READ
22B4    30 04                   JR      NC,RDFILE1
22B6    FE 01                   CMP     01H
22B8    20 1F                   JR      NZ,ERROR
                        * FILE-DATEN AUSGEBEN
                        * LETZTES BELEGTES BYTE IM MEMORY AUSGEBEN
22BA    21 35 23        RDFILE1 LD      HL,TEXT4
22BD    FF 03                   SCALL   .PRINT
22BF    62                      LD      H,D
22C0    6B                      LD      L,E
22C1    2B                      DEC     HL
22C2    CD E4 23                CALL    $OUTWRD
                        * LAENGE DES FILES AUSGEBEN
22C5    21 45 23                LD      HL,TEXT5
22C8    FF 03                   SCALL   .PRINT
22CA    EB                      EX      DE,HL
22CB    D1                      POP     DE
22CC    AF                      XOR     A
22CD    ED 52                   SBC     HL,DE
22CF    CD E4 23                CALL    $OUTWRD
                        * KANAL SCHLIESSEN
22D2    3E 01                   LD      A,01H
22D4    FF 26                   SCALL   .CLOSE
                        *RUECKKEHR ZUM SYSTEM
22D6    AF              BREAK   XOR     A
22D7    FF 00                   SCALL   .EXIT
                        * ERROR-BEHANDLUNG
22D9    26 0A           ERROR   LD      H,0AH
22DB    FF 2F                   SCALL   .ERROR
22DD    3E 01                   LD      A,01H
22DF    FF 00                   SCALL   .EXIT
```

```
                                        * DATA-BEREICH
22E1    1B 09 52 45 41 44 20TEXT1       DB      27,09H,'READ - FILE',0AH
        2D 20 46 49 4C 45 0A
22EF    09 3D 3D 3D 3D 3D 3D            DB      09H,'==========',0AH,0AH
        3D 3D 3D 3D 3D 0A 0A
22FD    56 65 72 73 69 6F 6E            DB      'Version 1.0',0AH
        20 31 2E 30 0A
2309    30 33 2D 4D 41 49 2D            DB      '03-MAI-81',0AH,8AH
        38 31 0A 8A
2314    46 69 6C 65 6E 61 6DTEXT2       DB      'Filename      :',0A0H
        65 20 20 20 20 20 20
        3A A0
2324    0A 76 6F 6E 20 41 64TEXT3       DB      0AH,'von Adresse   :',0A0H
        72 65 73 73 65 20 20
        20 3A A0
2335    62 69 73 20 41 64 72TEXT4       DB      'bis Adresse   :',0A0H
        65 73 73 65 20 20 20
        3A A0
2345    0A 46 69 6C 65 2D 4CTEXT5       DB      0AH,'File-Laenge   :',0A0H
        61 65 6E 67 65 20 20
        20 3A A0
2356    53 59 30 41 42 53       DEFBLK  DB      'SY0ABS'
235C                            FILNAM  DS      17
236D                                    ENCLU   INSTR
2397                                    ENCLU   INHEX
23D2                                    ENCLU   CNVAB
23E4                                    ENCLU   OUTWRD
2402                                    ENCLU   CNVBA
2414    00                              END     RDFILE
```

```
2280  3E 03 21 D6 22 FF 21 21 - E1 22 FF 03 3E 10 11 14
2290  23 21 5C 23 CD 6D 23 EB - 3E 04 21 24 23 CD 97 23
22A0  E5 E5 EB 11 56 23 3E 01 - FF 22 38 2D 3E 01 01 00
22B0  FF D1 FF 04 30 04 FE 01 - 20 1F 21 35 23 FF 03 62
22C0  6B 2B CD E4 23 21 45 23 - FF 03 2B D1 AF ED 52 CD
22D0  E4 23 3F 01 FF 26 AF FF - 00 26 0A FF 2F 3E 01 FF
22E0  00 1B 09 52 45 41 44 20 - 2D 20 46 49 4C 45 0A 09
22F0  3D 3D 3D 3D 3D 3D 3D 3D - 3D 3D 3D 0A 0A 56 65 72
2300  73 69 6F 6E 20 31 2E 30 - 0A 30 33 2D 4D 41 49 2D
2310  38 31 0A 8A 46 69 6C 65 - 6E 61 6D 65 20 20 20 3A
2320  20 20 3A A0 0A 76 6F 6E - 20 41 64 72 65 73 73 65
2330  20 20 20 3A A0 62 69 73 - 20 41 64 72 65 73 73 65
2340  20 20 20 3A A0 0A 46 69 - 6C 65 2D 4C 61 65 6E 67
2350  65 20 20 20 3A A0 53 59 - 30 41 42 53 2C 02 0F 50
2360  21 8F 4F CD E1 3D DA E6 - 51 FE 3D C2 04 C5 3C 4F
2370  D5 E5 E5 D5 FF 07 E1 FF - 03 E1 41 FF 01 38 FC 36
2380  00 FE 0A 28 0C 77 23 10 - F2 3E 07 FF 02 E1 D1 18
2390  DF 79 90 E1 D1 C1 C9 C5 - 3C 4F D5 E5 E5 C5 FF 07
23A0  C1 E1 FF 03 21 00 00 41 - FF 01 38 FC FE 0A D8 1D
23B0  CD D2 23 38 11 C5 06 04 - 07 07 07 07 C8 15 C9
23C0  14 10 F9 C1 10 E2 3E 07 - FF 02 E1 18 CF 7D D1 D1
23D0  C1 C9 D6 30 FE 00 D8 FE - 0A 38 07 FE 11 D8 D6 07
23E0  FE 10 3F C9 F5 C5 E5 06 - 04 AF C5 06 04 CB 15 CB
23F0  14 17 10 F9 E6 0F CD 02 - 24 FF 02 C1 10 EB E1 C1
2400  F1 C9 FE 00 D8 FE 10 3F - D8 FE 0A 38 02 C6 07 C6
2410  30 37 3F C9 00 69 6C 65 - 6E 61 6D 65 20 20 20 20
```

```
                *-----------------------------------------
                * WRFILE
                * VERSION 1.0
                * 05-MAI-81
                * GERHARD BAIER
                *
                * MIT DIESEM PROGRAMM KANN EIN BELIEBIGER
                * SPEICHERBEREICH ALS FILE AUF DISKETTE
                * GESCHRIEBEN WERDEN
                *
                *-----------------------------------------
                *
                        ENCLU   CLDDOS
2280                    ORG     2280H
2280 3E 03              LD      A,03H
2282 21 E1 22           LD      HL,BREAK
2285 FF 21              SCALL   .CTLC
                * BILDSCHIRM LOESCHEN UND TITEL AUSGEBEN
2287 21 EC 22           LD      HL,TEXT1
228A FF 03              SCALL   .PRINT
                * FILENAMEN EINLESEN
228C 3E 10              LD      A,10H
228E 11 21 23           LD      DE,TEXT2
2291 21 68 23           LD      HL,FILNAM
2294 CD 79 23           CALL    $INSTR
                * ANFANGS-ADRESSE EINGEBEN
2297 3E 04              LD      A,04H
2299 21 31 23           LD      HL,TEXT3
229C CD A3 23           CALL    $INHEX
229F EB                 EX      DE,HL
                * END-ADRESSE EINGEBEN
22A0 3E 04              LD      A,04H
22A2 21 42 23           LD      HL,TEXT4
22A5 CD A3 23           CALL    $INHEX
                * LAENGE BERECHNEN UND AUSGEBEN
22A8 23                 INC     HL
22A9 E5                 PUSH    HL
22AA E5                 PUSH    HL
22AB 21 52 23           LD      HL,TEXT5
22AE FF 03              SCALL   .PRINT
22B0 E1                 POP     HL
22B1 AF                 XOR     A       ;CLEAR CARRY UND AKKU
22B2 ED 52              SBC     HL,DE
22B4 CD F0 23           CALL    $OUTWRD ;LAENGE AUSGEBEN
                * REST DES LETZTEN BLOCKS MIT 0 BESETZEN
22B7 95                 SUB     L
22B8 E1                 POP     HL
22B9 28 04              JR      Z,WRFIL1
22BB 47                 LD      B,A
22BC CD 8A 19           CALL    $ZERO
                * FILE EROEFFNEN
22BF E5         WRFIL1  PUSH    HL
22C0 D5                 PUSH    DE
22C1 3E 01              LD      A,01H
22C3 21 68 23           LD      HL,FILNAM
22C6 11 62 23           LD      DE,DEFBLK
22C9 FF 23              SCALL   .OPENW
22CB 38 17                      JR      C,ERROR
22CD D1                         POP     DE
22CE E1                         POP     HL
                        * FILE SCHREIBEN
22CF ED 52                      SBC     HL,DE
22D1 44                         LD      B,H
22D2 0E 00                      LD        C,00H
22D4 3E 01                      LD      A,01H
22D6 FF 05                      SCALL   .WRITE
22D8 38 0A                      JR      C,ERROR
                        * KANAL SCHLIESSEN
22DA 3E 01                      LD      A,01H
22DC FF 26                      SCALL   .CLOSE
                        * RUECKKEHR ZUM SYSTEM
22DE AF                         XOR     A
22DF FF 00                      SCALL   .EXIT
                        *RUECKKEHR ZUM SYSTEM
22E1 AF                 BREAK   XOR     A
22E2 FF 00                      SCALL   .EXIT
                        * ERROR-BEHANDLUNG
22E4 26 0A              ERROR   LD      H,0AH
22E6 FF 2F                      SCALL   .ERROR
22E8 3E 01                      LD      A,01H
22EA FF 00                      SCALL   .EXIT
                        *
                        * DATA-BEREICH
22EC 1B 09 57 52 49 54 45TEXT1  DB      27,09H,'WRITE - FILE',0AH
     20 2D 20 46 49 4C 45
     0A
22FB 09 3D 3D 3D 3D 3D 3D       DB      09H,'==========',0AH,0AH
     3D 3D 3D 3D 3D 3D 0A
     0A
230A 56 65 72 73 69 6F 6E       DB      'Version 1.0',0AH
     20 31 2E 30 0A
2316 30 35 2D 4D 41 49 2D       DB      '05-MAI-81',0AH,8AH
     38 31 0A 8A
2321 46 69 6C 65 6E 61 6DTEXT2  DB      'Filename      :',0A0H
     65 20 20 20 20 20 20
     3A A0
2331 0A 76 6F 6E 20 41 64TEXT3  DB      0AH,'von Adresse   :',0A0H
     72 65 73 73 65 20 20
     20 3A A0
2342 62 69 73 20 41 64 72TEXT4  DB      'bis Adresse   :',0A0H
     65 73 73 65 20 20 20
     3A A0
2352 46 69 6C 65 2D 4C 61TEXT5  DB      'File-Laenge   :',0A0H
     65 6E 67 65 20 20 20
     3A A0
2362 53 59 30 41 42 53    DEFBLK      DB      'SY0ABS'
2368                      FILNAM  DS      17
2379                              ENCLU   INSTR
23A3                              ENCLU   INHEX
23DE                              ENCLU   CNVAB
23F0                              ENCLU   OUTWRD
240E                              ENCLU   CNVBA
2420                              ENCLU   ROMSUBS
2420 00                           END     WRFILE
```

```
WRFILE  MEMORY:2280  LAENGE:01A0  START:2280

2280  3E 03 21 E1 22 FF 21 21 - EC 22 FF 03 3E 10 11 21
2290  23 21 68 23 CD 79 23 3E - 04 21 31 23 CD A3 23 EB
22A0  3E 04 21 42 23 CD A3 23 - 23 E5 E5 21 52 23 FF 03
22B0  E1 AF ED 52 CD F0 23 95 - E1 28 04 47 CD 8A 19 E5
22C0  D5 3E 01 21 68 23 11 62 - 23 FF 23 38 17 D1 E1 ED
22D0  52 44 0E 00 3E 01 FF 05 - 38 0A 3E 01 FF 26 AF FF
22E0  00 AF FF 00 26 0A FF 2F - 3E 01 FF 00 1B 09 57 52
22F0  49 54 45 20 2D 20 46 49 - 4C 45 0A 09 3D 3D 3D 3D
2300  3D 3D 3D 3D 3D 3D 3D 3D - 0A 0A 56 65 72 73 69 6F
2310  6E 20 31 2E 30 0A 30 35 - 2D 4D 41 49 2D 38 31 0A
2320  8A 46 69 6C 65 6E 61 6D - 65 20 20 20 20 20 20 3A
2330  A0 0A 76 6F 6E 20 41 64 - 72 65 73 73 65 20 20 20
2340  3A A0 62 69 73 20 41 64 - 72 65 73 73 65 20 20 20
2350  3A A0 46 69 6C 65 2D 4C - 61 65 6E 67 65 20 20 20
2360  3A A0 53 59 30 41 42 53 - 51 FE 3D C2 04 52 CD A3
2370  52 CD 70 52 DA 57 50 21 - FF C5 3C 4F D5 E5 E5 D5
2380  FF 07 E1 FF 03 E1 41 FF - 01 38 FC 36 00 FE 0A 28
2390  0C 77 23 10 F2 3E 07 FF - 02 E1 D1 18 DF 79 90 E1
23A0  D1 C1 C9 C5 3C 4F D5 E5 - E5 C5 FF 07 C1 E1 FF 03
23B0  21 00 00 41 FF 01 38 FC - FE 0A 28 1D CD DE 23 38
23C0  11 C5 06 04 07 07 07 07 - 07 CB 15 CB 14 10 F9 C1
23D0  10 E2 3E 07 FF 02 E1 18 - CE 7D D1 D1 C1 C9 D6 30
23E0  FE 00 D8 FE 0A 38 07 FE - 11 D8 D6 07 FE 10 3F C9
23F0  F5 C5 E5 06 04 AF C5 06 - 04 CB 15 CB 14 17 10 F9
2400  E6 0F CD 0E 24 FF 02 C1 - 10 EB E1 C1 F1 C9 FE 00
2410  D8 FE 10 3F D8 FE 0A 38 - 02 C6 07 C6 30 37 3F C9
```

NACHTRAG zu »FORMAT«

von Günter Böhm

Das Formatierprogramm aus dem Augustheft hat sich bisher gut bewährt. Nur ist manchmal die Notwenigkeit aufgetreten, eine Zeile mit einem Space beginnen zu müssen oder verschiedene Spalten zu tabulieren. Dazu muß das Programm selbst nicht verändert werden; man muß nur Zeichen festlegen, die durch das Druckerprogramm entsprechend interpretiert werden. Deshalb formatiere ich auch nicht mehr direkt über den UART sondern über eine Druckerroutine, die an anderer Stelle beschrieben ist. (Olympia ESW 100 KSR an NASCOM)

Der von mir benutzte Drucker benutzt das Zeichen → (09hex) um eine vorher eingestellte Anzahl von Schritten nach rechts zu rücken. Wenn Sie also Texte auf Cassette einsenden und dieses Zeichen zum Tabulieren verwenden wollen, müßten Sie angeben, wie viele Zeichen der Tabulator überspringen soll.

Eine andere Möglichkeit des Tabulierens bildet das Festlegen eines Zeichens für einen Zwischenraum, der nicht unterdrückt werden darf. Mit T 4 oder NASSYS ist es kein Problem, das Zeichen ↰ für BS zu erzeugen, das ich als Space z.B. am Zeilenanfang abdrucke. Folgendes Programm zum Umwandeln von 08hex in einen Zwischenraum ist durch das Einfügen von ↰ eingerückt:

```
      CP,#08
      JR NZ OUT
      LD A,#20
 OUT JP Druck(z.B.SRLOUT)
```

Damit ist das Formatierprogramm wieder ein Stück vielseitiger geworden.

MINIPROGRAMM

```
                0010 ;ANZEIGE EINER
                0020 ;2-BYTE-PRUEFSUMME
                0030 ;ARGUMENT 2 ENTHAELT
                0040 ;DIE STARTADR.DES
                0050 ;SPEICHERS, ARG.3
                0060 ;ENTHAELT DIE LAENGE
                0070 ;DER ZEILE ODER DES
                0080 ;BLOCKS, BELIEBIGE
                0090 ;TASTE STARTET AUSGABE
                0091 ;DER NAECHSTEN ZEILE
                0092 ;3.10.81-NASBUG T4
0C0E            0100 ARG2   EQU  #C0E
0C10            0110 ARG3   EQU  #C10
0232            0120 HLOUT  EQU  #232
023C            0130 SPACE  EQU  #23C
0F00            0140        ORG  #F00
0F00 2A0E0C     0150 START  LD   HL,(ARG2)
0F03 E5         0160        PUSH HL
0F04 ED5B100C   0170        LD   DE,(ARG3)
0F08 19         0180        ADD  HL,DE
0F09 220E0C     0190        LD   (ARG2),HL
0F0C EB         0200        EX   DE,HL
0F0D E1         0210        POP  HL
0F0E CD3202     0220        CALL HLOUT
0F11 CD3C02     0230        CALL SPACE
0F14 DD210000   0240        LD   IX,0
0F18 0600       0250        LD   B,0
0F1A 4E         0260 LOOP   LD   C,(HL)
0F1B DD09       0270        ADD  IX,BC
0F1D 23         0280        INC  HL
0F1E E5         0290        PUSH HL
0F1F AF         0300        XOR  A
0F20 ED52       0310        SBC  HL,DE
0F22 E1         0320        POP  HL
0F23 38F5       0330        JR   C LOOP
0F25 DDE5       0340        PUSH IX
0F27 E1         0350        POP  HL
0F28 CD3202     0360        CALL HLOUT
0F2B EF         0370        RST  40
0F2C 1F00       0380        DEFW #1F
0F2E CD6900     0390 KEYB   CALL #69
0F31 30FB       0400        JR   NC KEYB
0F33 18CB       0410        JR   START
                0420 ;
                0430 ;ANPASSUNG AN NASSYS:
                0440 ;FOLGENDE PROGRAMMZEILEN
                0450 ; MUESSEN GEAENDERT
                0460 ;WERDEN
                0470 ;
     00         0220        NOP
     DF66       0221        DEFW #66DF
     00         0230        NOP
     DF69       0231        DEFW #69DF
     00         0360        NOP
     DF66       0361        DEFW #66DF
     00         0370        NOP
     DF6A       0380        DEFW #6ADF
     00         0390        NOP
     DF62       0391        DEFW #62DF
```

Das Programm ist voll verschieblich

GRAFIC ZUSATZKARTE
für den NASCOM 1

Mit dieser Zusatzkarte wird der NASCOM-1 auf die Grafikmöglichkeiten des NASCOM-2 erweitert. Auf der Grafikkarte ist ein EPROM 2732, das den gesamten Zeichensatz enthält, d.h. den Zeichensatz des vorhandenen Zeichengenerators und 128 weitere Zusatzzeichen. Für die Zusatzzeichen kann man zwischen zwei Möglichkeiten wählen. Entweder können die 128 Zusatzzeichen untergebracht werden, die das Grafik-ROM für den NASCOM-2 enthält (BASIC-SET-Funktionen & weitere Zeichen), oder aber die BASIC-SET-Funktionen und die Schachgrafik für das Schachprogramm von W. Bregulla. Auf Wunsch können auch spezielle Sonderzeichen eingebaut werden.

Der Aufbau der Grafikkarte ist einfach. Man muß lediglich die beiden 24-poligen Sockel einlöten, sowie den Widerstand R1 und die beiden Lötstützpunkte. Auf der Grundplatine sind drei zusätzliche Drahtbrücken nötig. Eine Drahtbrücke verbindet IC20 Pin 12 mit IC17 Pin 18. Die zweite Drahtbrücke verbindet IC17 Pin 19 mit IC16 Pin 14. Dann muß man den Pin 6 von IC15 aus der Fassung biegen (IC aus der Fassung nehmen und diesen Pin horizontal wegbiegen) und mit IC 16 Pin 10 verbinden. Jetzt kann man den Zeichengenerator MCM 6576 aus seiner Fassung auf der Grundplatine des NASCOM-1 nehmen. Dafür wird der eine Stecker des 24-poligen Kabels in diese Fassung gesteckt, der andere Stecker in die Fassung auf der Grafikkarte. Die Grafikkarte ist jetzt betriebsbereit. Die beiden Lötstützpunkte auf der Grafikkarte kann man entweder mit einer Drahtbrücke verbinden oder aber einen Schalter dazwischenlegen. Dann kann man die Grafik abschalten, wenn z.B. Programme das Bit7 im Bildschirmbereich anders verwenden.

Der Zeichengenerator MCM 6576 des NASCOM-1 wird nicht mehr benötigt. Damit ist gleichzeitig das Bauteil aus dem NASCOM entfernt, das mit Abstand die häufigsten Fehler verursachte. Der Zeichengenerator auf der Grafikkarte benötigt nur noch +5 Volt, so daß auch keine Probleme mit dem Netzteil mehr auftreten können. Wenn man statt der EPROMs 2708 auf der NASCOM-Platine 5-Volt-Typen einsetzt, kann man aus dem NASCOM-1 ein reines 5-Volt-System machen.

Bauteileliste:
Platine, fertig gebohrt
Zeichengenerator 2732
24-poliges Flachbandkabel mit 2 Steckern, komplett montiert
Widerstand 4,7 k Ohm
2 Lötstützpunkte
2 Fassungen 24 polig

Der Bausatz kostet 140.- DM incl. MWSt

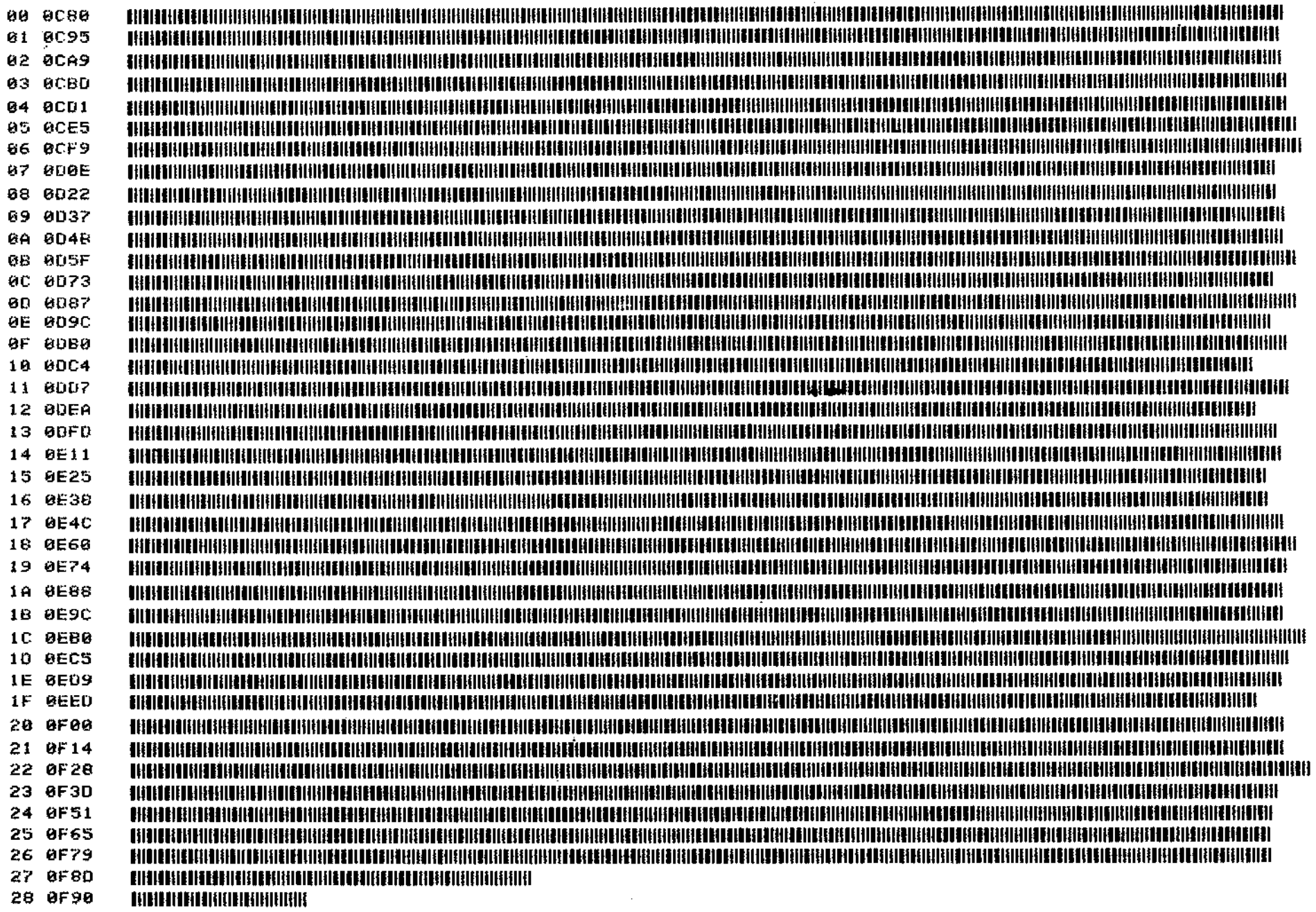
Bernd Ploss
Datenelektronik - Systementwicklung
[redacted]
[redacted] Rastatt
Tel. [redacted]

STRICHCODES
von Bernd Ploss

Nachdem ich den Strichcodeleser von Ing.-Büro Kanis erhielt, zeigte es sich, daß damit das im NASCOM-Journal 4/81 abgedruckte Programm nicht gelesen werden konnte. Eine genauere Untersuchung zeigte, daß der Kanis-Leser etwas größere Abstände zwischen den Strichen benötigt, als der Hewlett-Packkard-Lesestift (der aber auch mehr als doppelt so teuer ist), zumindest wenn der Strichcode mit einem Nadeldrucker gedruckt wird. Deshalb hier das Programm

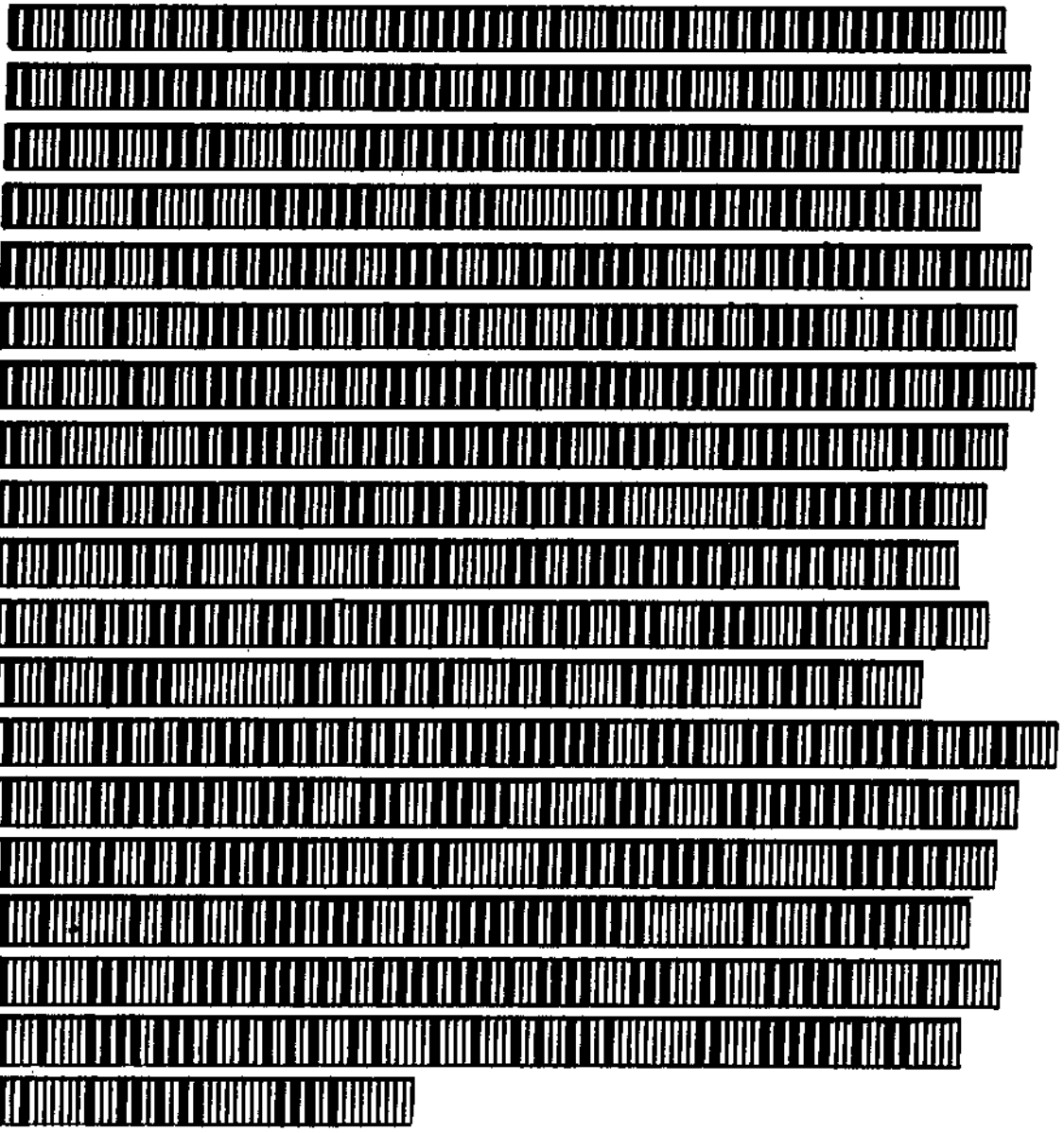
"Darts", das von Günther Brust für NAS-SYS umgeschrieben wurde.

Die Versuche mit dem Kanis-Leser zeigten auch, daß die Justierung dieses Stiftes kritisch sein kann, wenn die Vorlage nicht optimal ist, d.h. die Balken tiefschwarz und die Zwischenräu-



DARTS Start bei 0D30

Relocator Programm aus Heft 6/81



Für Leser, die bereits einen Barcode-Reader anschließen können: Probieren Sie zunächst aus, ob Sie das gedruckte Programm von Herrn Ploss (im BYTE Format) einwandfrei lesen können. Wenn ja, erübrigt sich das etwas aufwendigere Plotten der Programme(, das hier noch etwas schief geraten ist). Falls Sie den Plot einfacher zu lesen finden, werden wir in Zukunft weitere Programme im MC Format abdrucken.

Obige Zeilen beginnen mit zwei Synchronisationsbalken, dann folgt die Anzahl der Bytes pro Zeile (durchweg 8 wie im NASCOM Tabulate Befehl) und dann eine Prüfsumme. Bei Eingabe der Startadresse (hier 0C80) ins Leseprogramm wird der Maschinencode automatisch richtig in den Speicher gelesen

me rein weiß. Eventuell muß von der angegebenen Einstellung (Mitte des Bereichs, in dem die LED Control leuchtet) abgewichen werden. Viele Vorlagen ließen sich besser einlesen, wenn der Trimmer im Gegenuhrzeigersinn so weit gedreht wurde, bis die LED Control ausging. Der Hewlett-Packard Stift zeigte sich in dieser Hinsicht überlegen. Er verkraftet auch kontrastarme Vorlagen gut.

Von Programmen, die im NASCOM-Journal erscheinen sollen, kann ich Strichcodes ausdrucken. In diesem Fall bitte ich die Autoren, eine Kassette mit dem Programm direkt an mich zu schicken. Im Augenblick kann ich allerdings nur Kassetten im NASCOM-1 Format lesen.

Strichcodeleseprogramm für NAS-SYS

Das Leseprogramm für die Strichcodes ist jetzt bei MK-Systemtechnik und bei mir ab Lager lieferbar (DM 88.- incl. MWSt). Das Programm ist für NAS-SYS und für NASBUG lieferbar. Standardadresse ist AC00. Das Programm benutzt Bit 0 vom Port A der PIO. Eine andere Adressbelegung bzw. Benutzung eines anderen Ports muß bei der Bestellung besonders angegeben werden (DM 20.- Aufpreis). Wird der Lesestifft HEDS 3000 von Hewlett-Packard verwendet, benötigt man noch einen Pull-up Widerstand von 2,2 k Ohm nach &5 V, der Strichcodeleser vom Ing.-Büro Kanis kann direkt angeschlossen werden. Auf Anfrage kann das Programm auch in anderen Datenträgern, bei anderen Startadressen und für andere Ports geliefert werden.

Da inzwischen drei verschiedene Strichcodes zur Darstellung von Programmen eingeführt sind, besteht das Programm auch aus drei Teilen. Die Darstellung für absoluten und für verschieblichen Code wurde von der Zeitschrift Byte vorgeschlagen, die dritte Darstellung wird von der Zeitschrift mc verwendet.

Will man absoluten Code einlesen, wird das Programm mit E AC00 gestartet. Absoluten Code erkennt man daran, daß am Anfang jeder Zeile eine zweistellige Zeilennummer und die vierstellige Anfangsadresse stehen. Dann verschwindet der Cursor und der Lesestift ist bereit. Jetzt kann man Zeile für Zeile einlesen, wobei die Reihenfolge einzuhalten ist. Ist eine Zeile richtig eingelesen, erscheint die Zeilennummer und die Anfangsadresse auf dem Bildschirm. Erscheint keine Anzeige, dann trat ein Lesefehler auf und die Zeile muß nochmals gelesen werden. Will man ein absolutes Programm an einen anderen als den vorgesehenen Speicherbereich laden, startet man das Leseprogramm mit E AC00 nnnn. nnnn ist der Offset zu der Originaladresse.

Verschieblichen Code erkennt man daran, daß am Zeilenanfang nur die zweistellige Zeilennummer steht. Um solchen Code einzulesen, startet man das Leseprogramm mit E AC03 nnnn. Der Code wird dann ab der Adresse nnnn in den Speicher geladen. Ansonsten funktioniert das Einlesen von verschieblichem Code wie das Einlesen von absolutem Code.

Code in der mc-Darstellung wird etwas anders gehandhabt. Hierbei funktioniert der Leser als Eingabegerät wie z.B. auch die Tastatur. Mit E AC05 wird der Leser aktiviert und die Eingaberoutine bei $UIN (0C77) eingetragen, anschliessend meldet sich NAS-SYS wieder. UIN dient jetzt als Eingaberoutine. Dann wird das Programm gestartet, das die Eingabedaten aufnehmen soll, z.B. Basic oder Assembler. Dann muß auf die Eingaberoutine UIN umgeschaltet werden. In NAS-SYS und in ASM/EPROM wird dazu einfach der Befehl U eingegeben, im

Basic führt man DOKE 3189,1921 aus. Jetzt ist die Tastatur abgeschaltet und man kann mit dem Lesestift eingeben. Der Inhalt richtig gelesener Zeilen erscheint auf dem Bildschirm, erscheint keine Anzeige muß die Zeile nochmals gelesen werden. Ist die letzte Zeile gelesen, wird wieder auf die Tastatur als Eingabegerät umgeschaltet. Dieses dritte Leseprogramm verwendet den RAM-Bereich 0CB0 bis 0D00 als Textpuffer.

NAS-SYS 3
von Günter Böhm

Das neue Betriebssystem NASSYS3 ist eine verbesserte Version von NASSYS1 und dazu voll kompatibel. Alle Befehle, ausgenommen die unten beschriebenen Änderungen, werden wie gewohnt ausgeführt. Allerdings gibt es keinen L-Befehl mehr (d.h. Laden im T2 Format) und der Tabulate-Befehl gibt keine Prüfsumme mehr aus. Für manche Benutzer ist dies wohl ein Mangel, der aber durch folgende Verbesserungen sicher ausgeglichen wird.

1. Alle Tasten außer der @ - Taste führen die Repeat-Funktion aus, wenn sie niedergedrückt bleiben. Die Zeitdauer bis zur ersten Wiederholung und die Wiederholgeschwindigkeit können eingestellt werden.
2. Alle Unterprogramme sind interruptfähig.
3. Die CRT Routine kann Daten in jeden gewünschten Speicherbereich ausgeben. So können Überschriften auch in die Zeile 16 geschrieben werden.
4. Der Read-Befehl läßt durch eine zusätzliche Eingabe Programme oder Daten von Cassette in einen beliebigen Speicherbereich laden.
5. Das Tabulate Kommando wurde folgendermaßen erweitert:

a) Die ASCII Zeichen der Bytes werden wie die bisherigen Hex-Werte ausgegeben.

b) Ein vierter Parameter läßt die Ausgabe Zusätzlicher Zeichen pro Zeile zu, um Drucker mit verschiedener Zeilenbreite zu berücksichtigen.

c) Ein fünfter Parameter unterdrückt die Ausgabe der Hex - oder ASCII Werte.

6. Alle NASSYS Routinen können mit Single Step durchlaufen werden. Durch die Verwendung der Repeat Funktion können die Einzelschritte in hoher Geschwindigkeit abgearbeitet werden.
7. Die Registeranzeige wurde erweitert.
8. Um die Lesbarkeit zu verbessern,wurde die Anzeige beim Modify Befehl um zwei Spalten nach rechts verschoben.
9. Der External (X) Befehl hat zusätzliche Optionen. Ebenso wird das NULL Kommando in BASIC nun korrekt ausgeführt (es gibt nun auch wirklich Nullen aus).
10. Es gibt drei neue Befehle: P zeigt die gespeicherten User Register an, D führt ein Programm bei D000 aus und Y bei B000.
11. Die Blinkgeschwindigkeit des Cursors ist veränderbar.
12. Es gibt drei neue Unterprogramme: Wiederholung der Keyboard Abfrage, Ausgabe von zwei Leerzeichen und ein Unterprogramm, das eine beliebige Subroutine ausführen kann.
13. Sowohl bei Breakpoint als auch bei NMI führt das Programm einen Sprung aus, der in $NMI gespeichert ist, bevor die Register angezeigt werden. Durch Änderung des Sprunges in $NMI können Programme auch anders beendet werden.
14. Der BØ Befehl schaltet den Breakpoint vollständig ab, so daß mit der entsprechenden Hardware NASSYS im RAM ausgeführt werden kann.(Kapiere ich selbst nicht!)

Mit manchen der genannten Veränderungen (die übrigens aus INMC80 zitiert sind) kann man ohne Listing oder Manual nichts anfangen. Diese sollen aber in nächster Zeit zur Verfügung stehen, und dann können wir weitere Informationen veröffentlichen.

****

UMBAUANLEITUNG für NASCOM 1 Keyboard
von D. Oberle

NASCOM Keyboard Funktionsbeschreibung:

Das NASCOM-Keyboard enthält magnetisch-dynamische Tasten die völlig verschleißfrei arbeiten. Eine Taste (Bild 1) besteht aus einem Ringkern (1), den zwei Leiterschleifen (2,3) durch den Ringkern, und zwei am beweglichen Tastenkopf (4) befestigte Dauermagnetplättchen (5) die beim Niederdrücken der Taste die magnetische Induktion für die Erzeugung eines Stromimpulses in der Leseleitung des Ringkernübertragers liefern.

Die Leiterschleifen (2,3) der einzelnen Tasten sind, wie aus dem Schaltbild (Bild 4) ersichtlich, zu einer Matrix aus Lese- und Schreibleitungen miteinander verbunden. Über einen BCD-Zähler (IC 5)der von einem über die Software erzeugten Takt gesteuert wird, wird die Tastenmatrix über die Schreibleitung ständig nach gedrückten Tasten abgefragt. Eine gedrückte Taste

NASCOM -MAGNETTASTE

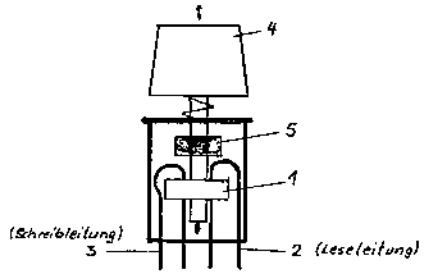


BILD 1

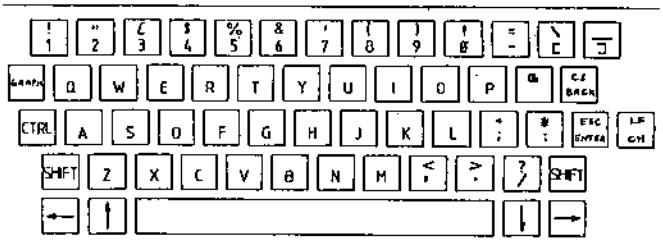
erzeugt in ihrer Leseleitung (Bild 4,S0-S6) durch das Differenzierglied aus C1,R8,R9 einen kurzen Nadelimpuls (Bild 5,IC2(5) F), der von einem Leseverstärker (Transistorarrays IC1,IC9) in den Pufferspeicher (RS-Flip-Flop IC7,IC2,IC8) eingeschrieben wird.
Aus der Anzahl der von der Software ausgegebenen Taktimpulse und der Nummer des zu diesem Zeitpunkt gesetzten Datenbits (aktiv low) am "Sense-Output", ermittelt dann das Monitorprogramm über Codetabellen den zur gedrückten Taste gehörenden ASCII Charakter.
Nach jeweils einem Zählerdurchlauf für die acht Schreibleitungen werden die Pufferspeicher zurückgesetzt (high) und der Abfragezyklus beginnt von neuem.
Falls dieselbe Taste beim nächsten Zählzyklus immer noch gedrückt ist, wird dies von der Software festgestellt und nicht weiterbearbeitet (Entprellung durch Software).

Einbau der neuen Magnettasten:

Die Tastaturbelegung des NASCOM II ist aus Bild 2 ersichtlich Die Tasten, es sollten Originaltasten sein, können einfach nach dem Bohren der vier Löcher für die Anschlußpins in der Epoxyplatine in die Halterung eingedrückt werden. Sie verriegeln sich im Halteblech von selbst und sind damit mechanisch fest eingebaut.
Das Monitorprogramm NASSYS I unterstützt den vollen ASCII-Code und benutzt verschiedene CNTRL-Codes zur Cursorsteuerung. Um die erforderlichen Codes direkt zu erzeugen, d.h. ohne die Nutzung der CNTRL-Taste, muß eine weitere Leseleitung (S6) eingefügt werden, denn es wird ein Datenbit mehr benötigt.

Dazu werden acht der neu eingebauten Tasten verwendet. Zum Einbau der Leseleitung wird einfach ein dünner isolierter Draht, am besten Wire-Wrap-Draht, an dem aus C1,R8,R9 bestehenden Differenzierglied angelötet und, wie in Bild 3 dargestellt, über die neu eingebauten Tasten in beliebiger Reihenfolge durchgeschaltet.

NASCOM II -TASTATUR



Für die acht Schreibleitungen (D0-D7) muß jeweils mit einem geeigneten Werkzeug die Leiterbahn zu einer ebenfalls in dieser Schreibleitung liegenden Taste unterbrochen und über die neu einzufügende Taste wieder verbunden werden. Die Anschlußpins der Tasten dürfen dabei nicht vertauscht werden, da sich sonst die Stromrichtung im Übertrager umkehrt. Mit dem Einfügen der Tasten in die jeweiligen Schreibleitungen ist die Tasten-Code-Zuordnung festgelegt (siehe Bild 2,Bild 3 und Bild 4).
Die neue Leseleitung (S6) wird dann in der aus Bild 4 ersichtlichen Weise über einen Leseverstärker (freier Transistor in IC1) auf die als Pufferspeicher benutzten freien NAND-Gates in IC3 geschaltet. Die dazu notwendigen Bauelemente R40,R41 und R42 müssen an einer leiterbahnfreien Stellen auf der Platine montiert werden. Der "Sense-Output" des S6-Puffers wird dann mit Pin 7 des Sockels (SK 1) für das Keyboardverbindungskabel verbunden.
Weiter empfehlenswert ist es, eine der neu eingebauten Tasten als SHIFT-Taste auf der linken Seite vorzusehen. Dazu müssen nur die Lese- und Schreibleitung der rechten Taste aufgetrennt und über die neue linke SHIFT-Taste wieder verbunden werden.

Bei Inbetriebnahme unter NASSYS I müssen die neu eingebauten Tasten, falls keine Fehler gemacht wurden, die aus Bild 4 zu entnehmenden Funktionen auslösen. Die GRAFIK-Taste kann natürlich nur dann sinnvoll sein, wenn eine GRAFIK-Erweiterung vorhanden ist.

NASCOM I KEYBOARD ERWEITERUNG AUF NASCOM II-STANDARD EINBAU EINER REPEAT-TASTE

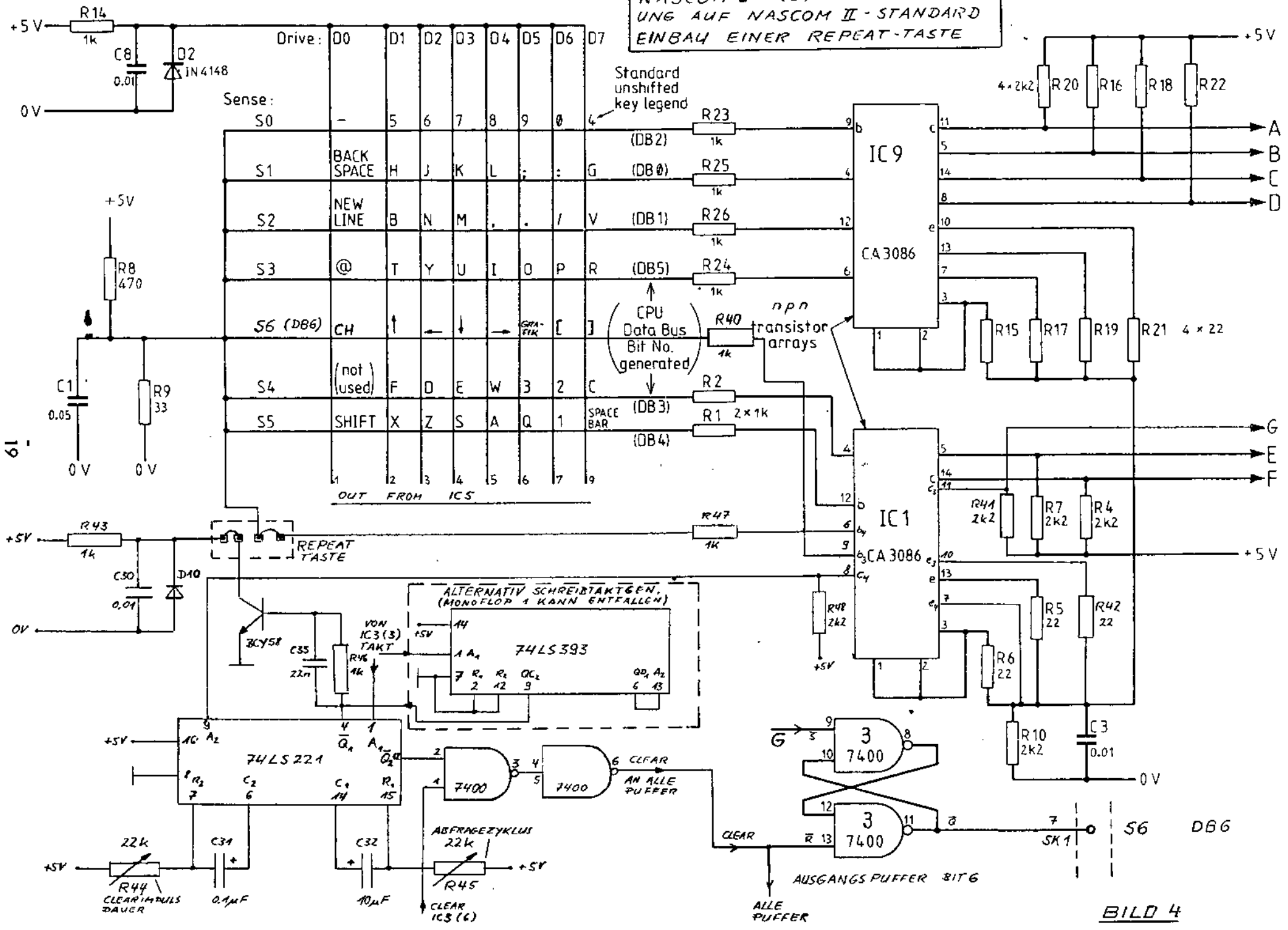


BILD 4

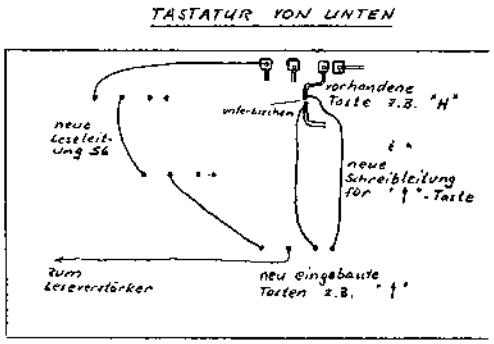


BILD 3

Einbau einer REPEAT-Taste:

Verzichtet man auf eine extra CNTRL-Taste oder auf die GRAFIK-Taste, so läßt sich mit geringem zusätzlichem Hardwareaufwand eine dieser Tasten als REPEAT-Taste verwenden. Man kann auch eine zusätzliche Taste, die allerdings schlecht befestigt werden kann, dazu benutzen.

Funktionsweise:

Da die Entprellung der Tasten von der Software vorgenommen wird, kann man durch einen entsprechend der gewünschten Repeatfrequenz von der Hardware erzeugten CLEAR-Impuls für die Pufferspeicher, die Software dazu bringen bei gleichzeitig gedrückter REPEAT-Taste und einer beliebigen anderen Taste einen Charakter wiederholt einzulesen und darzustellen.
Bei dem hier beschriebenen Beispiel wird davon ausgegangen, daß eine Originaltaste verwendet wird. Prinzipiell kann jedoch auch eine andere Tastenart verwendet werden. Die Schaltung ist dann dieser Taste anzupassen.
Für die Magnettaste muß zunächst ein seperater, mit dem Softwaretakt synchroner Schreibimpulsgenerator eingebaut werden. Außerdem wird noch ein zusätzlicher Leseverstärker notwendig. Dazu wird der Doppelmonoflopchip 74LS221, der leicht an einer freien Stelle auf der Leiterplatte montiert werden kann, eingesetzt. Der Takt für den Schreibimpulsgenerator wird an Pin 3 von IC3 abgegriffen und damit das 1. Monoflop getriggert. Der invertierte Ausgang Q1 dieses Monoflops steuert einen als Schreibimpulstreiber eingebauten Transistor BCY58 (oder ähnlicher Typ) an. Ist die REPEAT-Taste gedrückt, wird über den Leseverstärker (2. freier Transistor in IC1) und das zusätzlich eingebaute Differenzierglied D10,R43,C30 ein Triggerimpuls an das zweite Monoflop gegeben, welches entsprechend der eingestellten Impulsdauer einen CLEAR-Impuls am invertierten Ausgang Q2 abgibt. Dieser wird über zwei zusätzliche NAND-Gates zusammen mit dem von der Software kommenden CLEAR-Impuls auf die vorher aufgetrennte CLEAR-Leitung geschaltet.
Um die einwandfreie Funktion zu gewährleisten müssen die Monoflops auf den Softwaretakt abgeglichen werden. Es empfiehlt sich deshalb Trimmer einzubauen.
Eine bessere und unkritischere aber aufwendigere Lösung ergibt sich wenn als Schreibtaktgenerator für die REPEAT-Taste ein 8-Bit Zähler 74LS393 verwendet wird. Damit ist ein voll zum Softwaretakt synchroner Tastenabfragezyklus möglich was mit einem Monoflop schwierig abzugleichen ist. Die Abgleichprobleme bei der Verwendung von zwei Monoflops sind damit weitgehend ausgeschaltet. Es muß lediglich noch die Impulsdauer des CLEAR-Signals an einem Monoflop, wie in Bild 5 angegeben, eingestellt werden.

Abgleich:

Nach dem Einbau der REPEAT-Taste müssen die Impulszeiten der beiden Monoflops wie im Impulsdiagramm angegeben abgeglichen werden. Den Feinabgleich nimmt man am besten mit gedrückter SHIFT-Taste und einer beliebigen anderen Taste vor. Es dürfen bei gedrückter REPEAT-Taste keine "Unshiftet-Characters" mehr auftreten, besonders kritisch sind die Zeichen %,=,* die als Testzeichen benutzt werden sollten.
Bei gedrückter SHIFT- oder GRAFIK-Taste ist die Repeatfrequenz teilweise sehr ungleichmäßig, was aber wegen unterschiedlicher Softwarebearbeitungszeiten im Monitorprogramm nicht zu vermeiden ist.

Die hier beschriebenen Schaltungsvorschläge sind in meinem NASCOM I schon längere Zeit im Einsatz und arbeiten störungsfrei.
Besonders die CURSORPOSITIONING-Tasten sowie die REPEAT-Taste haben sich beim Editieren

als sehr zeitsparend und nützlich erwiesen.

LITERATURVERZEICHNIS:

NASCOM I Construction articel
NASCOM I Software notes
NASCOM II Documentation

PARALLELE ERGÄNZUNG
zu V. 24 von Werner Öhring

Ausgegangen wurde von den Erfahrungen mit der seriellen V.24-Schnittstelle und der Druckerschnittstelle von Centronics. Von der Druckerschnittstelle wurden die Handshakesignale Stb und Full (Busy + Ack) übernommen. Da mit der V.24-Schnittstelle die Erfahrung gemacht wurde, daß Datenübertragungen häufig nur in einer der beiden möglichen Richtungen erfolgen, werden die Signale beim PCC-Interface so über die Leitungen geführt, daß jeweils die freien Leitungen als Schirm für die gerade aktive Richtung wirken. Der Aufbau der Steckerbelegung ermöglicht den Einsatz eines Flachbandkabels zur Verbindung zwischen zwei Rechnern. Das 25polige Flachbandkabel kann daneben auch als einfaches Verlängerungskabel eingesetzt werden; falls am Flachbandkabel noch ein zusätzlicher Stecker montiert wird, kann das Kabel sogar als IEC-Bus-Kabel Verwendung finden.
Die PCC-Schnittstellen sind zur parallelen Übertragung zwischen zwei Geräten konzipiert. Durch einfaches Umschalten der Betriebsart des eingesetzten parallelen Ein-Ausgabebausteins kann von einer 8-Bit Datenübertragung in zwei Richtungen auf eine 16-Bit-Datenübertragung in zwei Richtungen umgestellt werden (nur vom Rechner zur Peripherie gedacht).
Über die PCC-Schnittstellenfamilie können nicht nur Kopplungen zwischen "intelligenten" Geräten, sondern auch zwischen Rechnern und einfachen Zusätzen verwirklicht werden (z.B. Zusatztastur, Lesestift, Zusatzanzeige, einfache Testgeräte oder Steuerungen). Auch lassen sich einfache Adapter zu anderen vorhandenen Schnittstellen ohne eigene Spannungsversorgung einsetzen.
Der Ablauf der Datenübertragung ist für beide Richtungen gleich: Der Sender prüft das Signal BFullI. Solange BFullI gleich "1" (TTL-Pegel) ist, darf der Sender keine Datengültigkeitsmeldung (BStbo) auslösen (die Datenleitungen Bxo sind ohne Bedeutung für den Empfänger). Sobald BFullI gleich "0" ist, werden die Daten vom Ausgangspuffer des Senders an die Datenleitungen Bxo angelegt. Die Gültigkeit der Daten wird dem Empfänger mit der fallenden Flanke des Signals BStbo vom Sender mitgeteilt. Der Empfänger signalisiert mit der steigenden Flanke des Signals BFullI an den Sender, daß die Daten in den Eingangspuffer übernommen worden sind. Spätestens jetzt muß der Sender das Signal

Pin-Nr.	PCC-8BM1.1 Buchse	PCC-8BM1.1 Stecker	PCC-8CM1.1 Buchse	PCC-8CM1.1 Stecker	PCC-16UM1.1 Buchse	PCC-16UM1.1 Stecker
1	Gnd	Gnd	Gnd	Gnd	Gnd	Gnd
2	A0I	B0o	SeI	SeI	B8o	A8I
3	AFullo	BFullI	(AFullo)	(BFullI)	(AFullo)	(BFullI)
4	A1I	B1o	FlI	Flo	B9o	A9I
5	AStbI*	BStbo*	(AStbI*)	(CH*)	(AStbI*)	(BStbo*)
6	A2I	B2o	MEr	MEo	B10o	A10I
7	A3I	B3o	AlI	Alo	B11o	A11I
8	A4I	B4o	RsI	Rso	B12o	A12I
9	A5I	B5o	(A5I)	TTL-0	B13o	A13I
10	A6I	B6o	(A6I)	TTL-1	B14o	A14I
11	A7I	B7o	(A7I)	TTL-0	B15o	A15I
12	API	BPo	(API)	(BPo)	(AHPI)	(BHPo)
13	+5Vout	+5Vin	+5Vout	+5Vin	+5Vout	+5Vin
14	BPo	API	CAckI*	CAcko*	BPo	API
15	B7o	A7I	B7o	D7I	B7o	A7I
16	B6o	A6I	B6o	D6I	B6o	A6I
17	B5o	A5I	B5o	D5I	B5o	A5I
18	B4o	A4I	B4o	D4I	B4o	A4I
19	B3o	A3I	B3o	D3I	B3o	A3I
20	B2o	A2I	B2o	D2I	B2o	A2I
21	BStbo*	AStbI*	BStbo*	DStbI*	BStbo*	AStbI*
22	B1o	A1I	B1o	D1I	B1o	A1I
23	BFullI	AFullo	CBusyI	CBusyo	BFullI	AFullI
24	B0o	A0I	B0o	D0I	B0o	A0I
25	Gnd	Gnd	Gnd	Gnd	Gnd	Gnd

Dxl: Daten für den Drucker
DStbI*: Datengültigkeitsmeldung

Tabelle der Buchsen- und Steckerbelegungen

BStbo auf "1" (TTL) wechseln. Die Datenleitungen sind dabei für den Empfänger ohne Bedeutung. Sobald der Eingangspuffer vom Empfänger leer und bereit für neue Daten ist, wechselt das Signal BFullI zum Sender von logisch "1" auf "0".
Ein nicht angeschlossener Empfänger entspricht "dauernd belegt" (BFull = "1"). Rückmeldungen (zum Beispiel Drucker wartet, weil Papier fehlt) werden nicht über spezielle Zusatzleitungen, sondern als Folge von ASCII-Zeichen realisiert. Der Inhalt der Rückmeldungen kann damit beliebig vielfältig gestaltet werden.
Die Signalpegel entsprechen in der Normalausführung der PCC-Schnittstelle der TTL Spezifikation. Die Bezeichnung der Schnittstellen ergibt sich aus der Zahl der Datenleitungen gefolgt von der Art der Datenübertragung und dem mechanischen Aufbau. Ein "B"

weist auf eine Datenübertragung in zwei Richtungen (bidirektional auf getrennten Leitungen) hin, ein "C" auf nur eine Datenrichtung mit Rückmeldung vom Drucker auf der zweiten Richtung (Signalverlauf analog der Spezifikationen von Centronics), ein "U" bedeutet, daß die Schnittstelle für unidirektionalen Datenverkehr ausgelegt ist. Die Abkürzung M1.1 steht für "mechanisch einfache Schnittstelle", Version 1.1. Die Schnittstelle 8 CM 1.1 (TTL-Pegel) ist kompatibel mit dem Centronics-Interface, die Schnittstelle 8BM 1.1 (TTL-Pegel) paßt zur V.24/RS232-Definition (Parity). Die beiden Schnittstellen können ineinander umgewandelt werden

Neben Datenleitungen (8 oder 16) verfügen die PCC-Schnittstellen über eine Anzahl von Signalleitungen: AI oder SS (Hardware Alarm), CAck (Acknowledge bestätigt die Übernahme der Daten in den Drucker), CBusy (Busy meldet, daß der Drucker beschäftigt ist und keine Daten in den Eingangspuffer übernehmen kann), CH oder Change (Hilfssignal zur Anpassung der Schnittstelle PCC-8CM an PCC-8BM, FI oder Fault (allgemeine Fehlermeldung), ME oder Papier End (Material zu Ende, zum Beispiel Papier, Farbband, Tinte), RS (Drucker wurde rückgesetzt), Sel oder Select (Drucker ist betriebsbereit und angewählt), TTL-0, TTL-1 (die sogenannten Signale ermöglichen zusammen mit dem Signal CH eine Kompatibilität zwischen hard- (Einzelleitung) beziehungsweise softwaremäßig (ASCII-Zeichen) erzeugten Rückmeldungen vom intelligenten Peripheriegerät)

HOCHAUFLÖSENDE GRAPHIK

von Hans-Martin Pohl

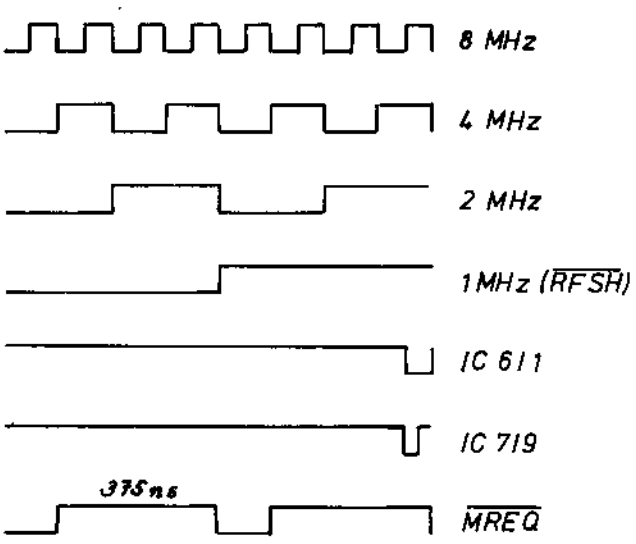
Der folgende Beitrag wurde vom Verfasser noch etwas ausführlicher überarbeitet und mit Anmerkungen zur Anwendung auf NASCOM1 versehen. Leider ist die Cassette auf dem Postweg verlorengegangen. Falls Sie Fragen haben oder Hinweise zur Anpassung an NASCOM1 benötigen, schreiben Sie uns. Herr Pohl ist gerne bereit, noch weitere Informationen zu liefern. (Red.).

Man kann mit Hilfe von 8 TTL-ICs und zwei Potis zum vorhandenen Videoteil eine hochauflösende Graphik mit 256 x 256 Bildpunkten bauen, wenn man mindestens 8K RAM besitzt.

Funktionsweise:

Durch das VBLANK-Signal wird der Bus vom Prozessor angefordert. Wenn der Prozessor den Bus mit BUSAK freigibt, werden die Zeilen und Spaltenzähler des Videoteils auch als Adresszähler verwendet. Dadurch werden mit Hilfe des RD- und MREQ-Signals die Graphik-Bytes auf den Datenbus gelesen. Mit dem Schieberegister (IC 6) wird daraus das Bildsignal erzeugt. Diese wird mit dem Bild- und Synchronisationssignal vom Videoteil gemischt und kann so direkt in meinen Fernseher eingespeist werden. Damit sich die Inhalte der dynamischen RAMs nicht verändern, wird bei jeder Adresse noch ein Refresh-Lesen durchgeführt. Außerdem wird das Ein- und Ausschalten durch zwei Flip-Flops gesteuert. (74LS74 hat bei mir nicht funktioniert).

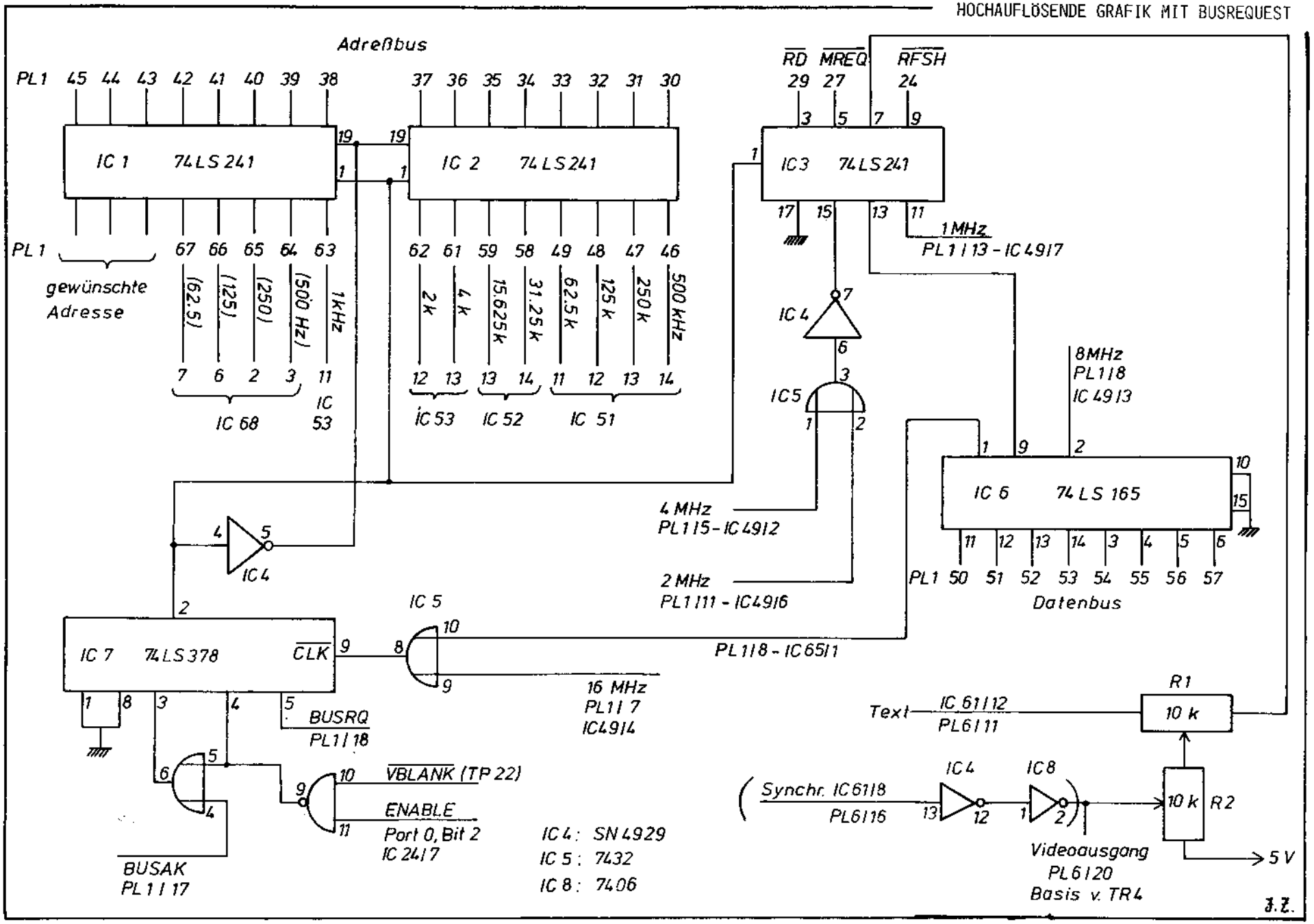
Zeitablauf vereinfacht



Um alle 65536 Bildpunkte abbilden zu können, muß im Videoteil bei IC 44 Pin 8 herausgebogen werden. Dadurch wird jedes Zeichen mit seiner vollen gespeicherten Höhe von 16 Punkten abgebildet, und nicht nur mit 12 oder 14. Außerdem verringert sich die Bildfrequenz und das Bild wird etwas größer.

Da die Graphik den Prozessor dauernd unterbricht, ist es nicht möglich, bei eingeschaltener Graphik Programme laufen zu lassen, die nicht unterbrochen werden dürfen (z.B. Cassette lesen). Andere Programme werden nur langsamer abgearbeitet.

HOCHAUFLÖSENDE GRAFIK MIT BUSREQUEST



Es ist auch möglich, die Graphik durch den Modulator abzubilden. Dazu muß auf der Hauptplatine R 68 herausgenommen werden. Statt diesem wird der Videoausgang der Graphikplatine an die Basis von Tr 4 angeschlossen. Der 5V-Anschluß an R2 und das Synchronisationssignal entfallen dann. Da das Text- und Graphik-Signal gemischt wird, kann man auch bei eingeschaltener Graphik den normalen Text auf dem Bildschirm lesen. Mit R1 wird das Helligkeitsverhältnis zwischen Graphik und Text geregelt.
Die Hälfte der gespeicherten Graphik wird doppelt abgebildet, weil der Video-Teil in horizontaler Richtung 384 Bildpunkte abbildet, die Graphik aber nur 256 gespeichert hat.
Mit folgendem Programm kann eine Sinuslinie gezeichnet werden, wenn die Graphik vorher mit einem Copy-Befehl gelöscht wird.

```
10 DOKE 4100,3200: POKE 3072,4
20 FORX=GOTO 255:Y=SIN(X/40.7)*127+127
30 POKE 3201,Y: POKE 3203,X: X=USR(X): NEXT
```

Das Einschalten der Graphik geschieht über Port0 Bit 2 in Zeile 10.

Maschinensprache - Unterprogramm zum Punktsetzen-rücksetzen

```
0C80 26 70 3E FB C6 80 6F 17
0C88 17 17 2F E6 38 F6 86 32
0C90 A4 0C 7C D6 10 1F CB 1D
0C98 1F CB 1D 1F CB 1D E6 3F
0CA0 F6 60 67 CB A6 C9
```

Das Byte in Zelle C8E bestimmt, ob das adressierte Bit gesetzt (C6) oder rückgesetzt werden soll (86).

\+ + +

Suche BASF Mini-Floppy-Laufwerk (Typ 6106 o.ä.)
Ulrich Wallis [redacted] (nach 17°°)

Wer hat Interesse an einer Mitarbeit beim Bau kleiner (preiswerter!) Elektronikgeräte für Behinderte oder ähnliche Anwendungen? Wer hat eventuell schon Erfahrungen auf diesem Gebiet?
H.Grasl [redacted] (ab 18°°)
[redacted]

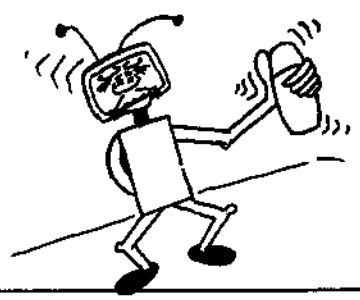
APPLE II Europlus 48K mit leerem Hobbyboard, Grafikdefinitionshilfe und Spielprogrammen DM 2 800.-
L.Bayer [redacted]

SORTIEREN IN BASIC

TEIL 4 von Wolfgang Mayer-Gürr

Das Bubble-Sortierverfahren ist zwar vom Programmaufbau sehr kurz, in der Ausführung aber sehr langsam. Dies ist besonders bei einer ungünstigen Mischung der Elemente der Fall, wenn nur ein einziges Element im Feld aufsteigen muß. Das Feld muß dann n-1 mal durchlaufen werden.
Eine kleine Verbesserung bietet der Shaker Algorithmus. Der Zeiger wandert abwechselnd von oben nach unten und dann umgekehrt über den Arrayinhalt.

```
100  REM  ******************************
110  REM  *  SHAKER SORTIERVERFAHREN   *
120  REM  ******************************
130 N = 10
140  REM   N= ANZAHL DER EINGABEN
150  DIM N$(N)
160  REM  EINGABE
170  FOR I = 1 TO N
180  PRINT "NR. ";I;
190  INPUT N$(I)
200  NEXT I
210  REM   ZUM UNTERPROGRAMM SORTIEREN
220  GOSUB 285
230  REM   AUSGABE
240  FOR I = 1 TO N
250  PRINT N$(I)
260  NEXT I
270  END
280  REM   UNTERPROGRAMM SORTIEREN
285 L = 2
290 R = N
300 I = N - 1
310  REM   BEGINN DER REPEAT-SCHLEIFE
320  FOR J = R TO L STEP  - 1
330  IF N$(J - 1) <  = N$(J) THEN 360
340  GOSUB 470: REM   VERTAUSCHEN
350 I = J
360  NEXT J
370 L = I + 1
380  FOR J = L TO R
390  IF N$(J - 1) <  = N$(J) THEN 420
400  GOSUB 470
410 I = J
420  NEXT J
430 R = I - 1
440  IF L <  = R THEN 310
450  RETURN : REM  ENDE UP SORTIEREN
460  REM  UP AUSTAUSCH
470 H$ = N$(J)
480 N$(J) = N$(J - 1)
490 N$(J - 1) = H$
500  RETURN
```



BASIC-TOKEN NAS 2

von Wolfgang von Jan

Die folgende Liste mit BASIC-Token für den NASCOM2 wurde bewußt gegenüber der Auflistung in CHIP 3/80 geändert, da ich einige Fehler mit jener Auflistung machte. Die Begriffe werden ja nur dann gebraucht, wenn Zeilen über den "Rand" hinausgehen würden. Dann ist es besonders ärgerlich, wenn ein Begriff falsch eingegeben wurde.
Die Liste wurde alphabetisch sortiert, um die Tastenkombinationen schneller zu finden. Die Hexzahlen geben die Werte an, unter denen die Befehle im Arbeitsspeicher stehen. Die Werte habe ich wie folgt erhalten:
RESET - Kaltstart BASIC - Zeilennummer - Befehl - gefolgt aus Kontrollgründen von ABCD - Monitor - T1060 -
Bei meinem NASCOM2 mit Toolkit und NASPEN stand dann in 1061 jeweils der HEX-Wert, gefolgt von 20 und den vier Kontrollbuchstaben.

BASIC Token NASCOM 2
ROM BASIC VER. 4.7

Befehl	Graphic Taste +	HEX
ABS	8	B8
AND	1	B1
ASC	K	CB
ATN	D	C4
CHR$	L	CC
CLEAR	shift1	A1
CLOAD	shift2	A2
CLS	CTRL Y	99
CONT	CTRL shift	9F
COS	A	C1
CSAVE	shift3	A3
DATA	CTRL C	83
DEEK	F	C6
DEF	Cursor	94
DIM	CTRL E	85
DOKE	CTRL V	96
END	shift CTRL a	80
EXP	shift a	C0
FN	shift7	A7
FOR	CTRL A	81
FRE	:	BA
GOSUB	CTRL L	8C

Befehl	Graphic Taste+	HEX
GOTO	BACK	88
IF	CTRL J	8A
INP	;	BB
INPUT	CTRL D	84
INT	7	B7
LEFT$	M	CD
LEN	H	C8
LET	CTRL G	87
LINES	CTRL X	98
LIST	Leertaste	A0
LOG	shift /	BF
MID$	0	CF
MONITOR	CTRL	9B
NEW	shift 4	A4
NEXT	CTRL B	82
NOT	shift :	AA
NULL	Cursor	92
ON	Cursor	91
OR	2	B2
OUT	CTRL P	90
PEEK	E	C5
POINT	G	C7
POKE	CTRL U	95
POS	shift ,	BC
PRINT	? (ohne Graph.)	9E
READ	CTRL F	86
REM	CTRL N	8E
RESET	CTRL	9D
RESTORE	CTRL K	8B
RETURN	ENTER	8D
RIGHT$	N	CE
RND	shift .	BE
RUN	CTRL I	89
SCREEN	CH	97
SET	shift CTRL	9C
SGN	6	B6
SIN	B	C2
SPC (	shift 8	A8
SQR	shift_	BD
STEP	shift ;	AB
STOP	CTRL O	8F
STR$	I	C9
TAB (	shift 5	A5
TAN	C	C3
THEN	shift CTRL I	A9
TO	shift 6	A6
USR	9	B9
VAL	J	CA
WAIT	Cursor	93
WIDTH	CTRL Z	9A

KLEINANZEIGEN

Jeder Abonnent kann kostenlose Kleinanzeigen bis 40 Wörter aufgeben!

Verkaufe Zeap 2, NASDIS, DEBUG auf original 8 X 1 K EPROM DM 250.-
Siemens T 100, Tischgehäuse 75 Bd, Großbuchstaben DM 200.-
Tel. [redacted]

Wer hat Interesse an Software-/Hardware-/ Informationsaustausch??
Uwe Fricke Tel. [redacted]

Suche gegen Gebot Assemblerlisting +Manual-Kopie von NASSYS(1).- Wer hat in Hamburg Interesse an einem NASCOM-User-Treffen?
F.-L. Bruhns [redacted]

Verkaufe NASBUG T2, T4, TINY-BASIC, GRAPHIC-ROM je EPROM (2708) DM 20.-
incl. original Dokum.
Programmiere Sonderzeichen (64 max) für NASCOM 2 -ROM (2716) DM 40.-
Formular bei Manfred Segelke E.S.G. [redacted]

Elektrotechnikstudent sucht NASCOM 1 - Benutzer im Raum Kiel zwecks Erfahrungsaustausch.
Hans-Jürgen Plath [redacted]

Suche ZEAP 2.0 ASSEMBLER mit :=Command-Funktion, NASSYS 3 incl. Listing
Rüdiger Maurer Tel. [redacted]

Verkaufe NASCOM 2, 32 K, mit NASPEN und TOOL-KIT in orig. Pultgehäuse, betriebsbereit DM 1200.-
P.-M. Hax Tel. [redacted] (Mo-Do ab 17°°) [redacted]

Verkaufe NASCOM 1 , 32 K RAM, Graphic, Tastaturerweiterung; NASBUG-NASSYS umschaltbar; 20 K ROM (8K BASIC,2K TOOLKIT,3K Assembler ZEAP 2 , 2K Disassembler NASDIS, 1K Debugger) Preis VB DM 1200.-
Nach 18°° Uhr Tel. [redacted]

Verkaufe NASCOM 1 mit NASSYS 1, 3 Amp. Netzteil, Buffer-Board, Cass.recorder, Schreibm.-Interface ("Hofer-Drucker") und div. Ass.-Programmen: DM 1000.-
Martin Riedel, [redacted]

Verkaufe NASSYS 5 voll interruptfähig, mit 4 neuen Befehlen
in 2716 (5V) DM 50.-
in 2X2708 DM 65.-
J.C. Lotter [redacted]

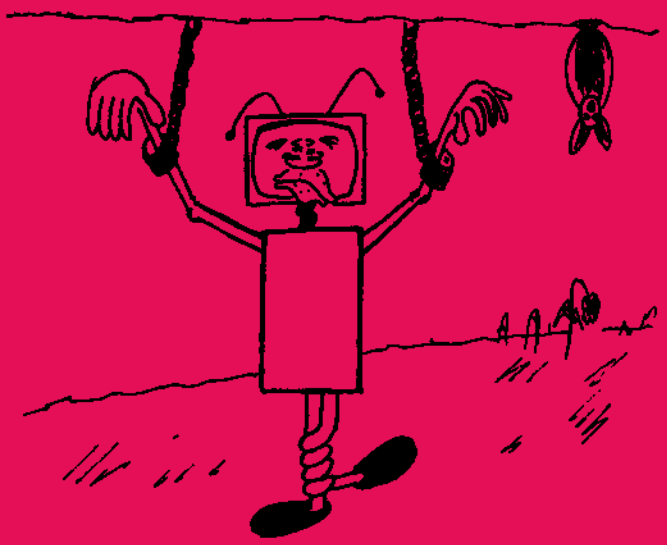
PUSH/POP REGISTER

Unterprogramme sollten die Register, die sie benutzen und die nicht zur Übergabe von Parametern dienen, unverändert lassen. Deshalb besteht der Anfang der meisten Unterprogramme aus etlichen PUSHs und das Ende aus POPs. Um in größeren Programmen Platz zu sparen - und zwar sowohl im Objektkode als auch im Quellkode, wobei letzteres die "Schallgrenze" für Programme über ca. 4KB darstellt - habe ich die folgenden beiden Programme benutzt. Das erste "PUA" rettet alle Register (IX & IY können leicht noch dazu) auf den Stack. Dies ist nicht so trivial, wie man zunächst denkt, denn auf dem Stack steht ja zunächst die Rückkehradresse von "PUA". Die vorgestelle Version speichert den Inhalt von HL in einem Speicherplatz "HLS" ab, der natürlich nicht im Eprom sein darf. Durch schauderhafte Stackmanipulationen könnte man das zwar auch umgehen, aber es wäre deutlich länger & langsamer.
Am Anfang eines Unterprogrammes steht dann "CALL PUA". Das Unterprogramm wird nicht mit RET abgeschlossen, sondern mit "JP POARET". In dem simplen "POARET" werden die Register zurückgeladen und der "RET"-Befehl ausgeführt, im Unterprogramm braucht man also kein "RET".
Überhaupt möchte ich in diesem Zusammenhang darauf hinweisen, daß man beim Aufruf eines zweiten Unterprogrammes am Ende des ersten Unterprogrammes mit "JP UP2" statt "CALL UP2, RET" den gleichen Effekt erzielt.

```
ASSEMBLER - Source Listing
PUSH/POP (fast) alle Register.

M. Bach, Am Schloßpark 12, 7801 Stegen
07661/61457

2198 22A921    0070 PUA     LD   (HLS),HL;zwischensp
219B E3        0080         EX   (SP),HL;Push HL
219C D5        0090         PUSH DE
219D C5        0100         PUSH BC
219E F5        0110         PUSH AF
219F E5        0120         PUSH HL;Rückkehradresse
21A0 2AA921    0130         LD   HL,(HLS);HL wiederh
21A3 C9        0140         RET

Pseudo-UP: POP alle Register & Return
21A4 F1        0170 POARET  POP  AF
21A5 C1        0180         POP  BC
21A6 D1        0190         POP  DE
21A7 E1        0200         POP  HL
21A8 C9        0210         RET

21A9 0002      0212 HLS     DEFS 2
```

nascompl

COMPUTERMISSHANDLUNG



Hallo liebe Leser,
Heute ein ernstes Thema.Immer wieder werden NASCOMs zur Reparatur eingeschickt, die sehr seltsame Beschädigungen aufweisen. Der Techniker kommt dann betroffen häufig zu dem Schluß: Hier liegt ein Fall von Computermißhandlung vor!
Natürlich gibt niemand, der seinen NASCOM mißhandelt, das zu; viele Fälle werden als Unglücksfälle deklariert. Es wundert daher nicht, daß die Techniker, die sich mit den kleinen Patienten und ihren Kurzschlüssen und Tastenprellungen zu befassen haben, von einer "heimlichen Epidemie" sprechen.
Was sind das für Leute, die ihren NASCOM mißhandeln?
Sie kommen aus allen Schichten. Es könnten Ihre Nachbarn, Verwandten, Freunde und Arbeitskollegen sein.
Meist handelt es sich um unreife Menschen, die den Anforderungen des Programmierens nicht gewachsen sind. Der falsche relative Sprung, der verlorene Speicherinhalt und die andauernde ERROR - Meldung können auch dem geduldigsten und liebevollsten Programmierer zusetzen. Bei denen aber, die selbst noch unfertig und labil sind, können solch alltägliche Ärgernisse unkontrollierbare Gewaltausbrüche auslösen.
Was kann man gegen das bittere Los der armen NASCOMs tun? Das einzige Mittel ist eine verstärkte öffentliche Wachsamkeit.
Wenn Sie also in der Nachbarschaft wütendes Hämmern auf eine Tastatur vernehmen, wenn es verdächtig nach Elektronen riecht oder wenn gar eine Platine aus dem Fenster fliegt, dann wenden Sie sich sofort an MKS - Systemtechnik. In besonders schwerwiegenden Fällen steht Ihnen Herr Klein auch mitten in der Nacht zur Verfügung.
Also: wachsam sein!
In diesem Sinne Ihr NASCOMPL

IMPRESSUM

REDAKTION: Günter Böhm, Günter Kreidl
Wolfgang Mayer-Gürr, Josef Zeller
RESSORTS:
MASCHINENPROGRAMME:
Günter Böhm, [redacted] Karlsruhe, Tel. [redacted]
Günter Kreidl, [redacted], [redacted] Straelen
Tel. [redacted]
BASIC und FLOPPY:
Wolfgang Mayer-Gürr, [redacted], [redacted], [redacted] Recklinghausen, Tel. [redacted]
HARDWARE:
Josef Zeller, [redacted], [redacted] Ulm
VERLAG:NASCOM JOURNAL, c/o MK-Systemtechnik,
Pater-Mayer-Str.6, 6728 Germersheim
Tel.07274/2756
Telex 453500 mks d
VERTRIEB: Direktvertrieb durch den Verlag
Erscheinungsweise: monatlich
Bezugspreis: 1M In- und Ausland 48.- für ein Jahresabonnement. Abonnements können aus technischen Gründen immer nur für die Dauer eines Kalenderjahres, d.h. vom 1.1. bis 31.12. laufen. Bei Bestellung nach dem 1.1. werden die fehlenden Hefte mit der ersten Lieferung bis zum Bestellzeitpunkt automatisch mitgeliefert.
Bezugsmöglichkeiten: Durch Bestellung bei MK-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.
Zahlung: Nach Eingang Ihrer Bestellung erhalten Sie von uns die ausstehenden Hefte bis zur aktuellen Ausgabe sowie eine Rechnung. Bitte, zahlen Sie dann den Rechnungsbetrag auf unser Sonderkonto (s.o.) ein. Bitte keine Vorauszahlungen!
Bitte, Anfragen wegen Abonnements oder Lieferung nicht an die Redaktion sondern nur an den Verlag.

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 **2975,-**

mit 1 Mini DCR	3495,-
mit 2 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 (0 72 74) 27 56
Telex 0453500 mks d

MK-Systemtechnik
Kriegsstraße 164
7500 Karlsruhe
Telefon (07 21) 2 92 43

MK-Systemtechnik
Pfaffenberg 4
5650 Solingen 1
Telefon (0 21 22) 4 72 67