

LOOP

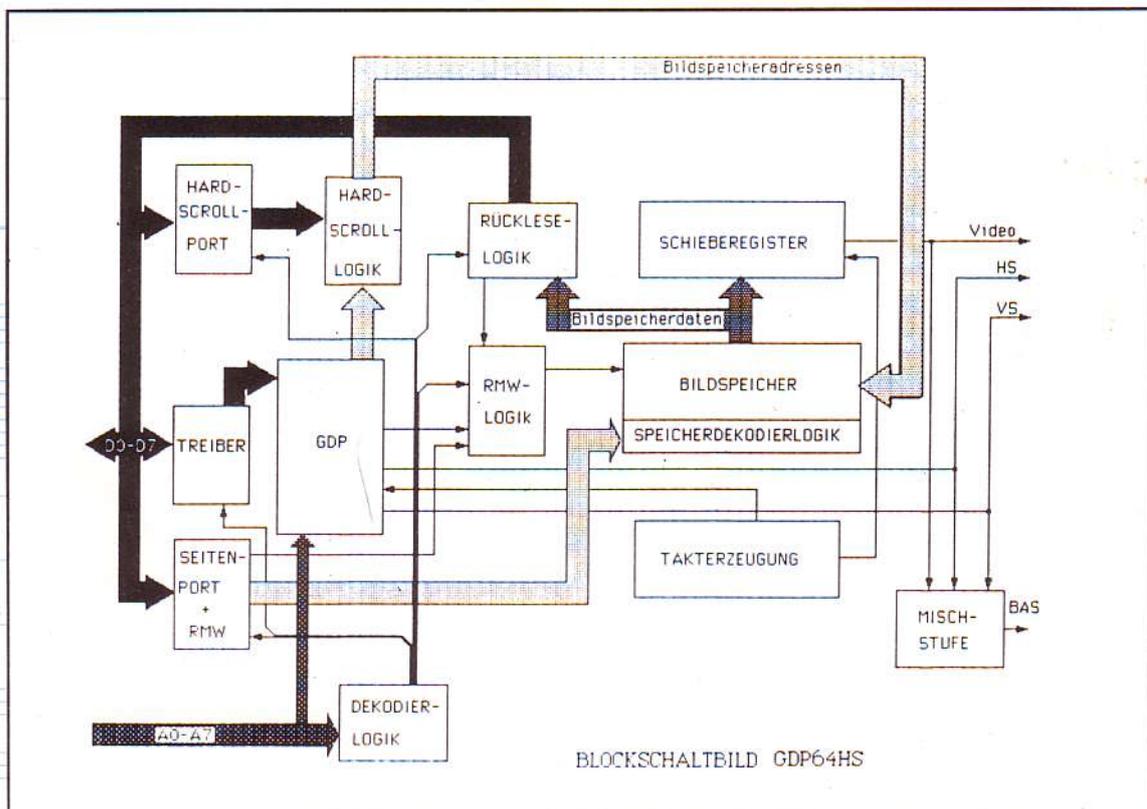
100 Juli 1989

18

3. JAHRGANG

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

DM 3,-



Hardware:

GDP64HS - Schnittstelle zum Monitor (S. 18/4)

Test:

LOG 16 gegen LOGAN (S. 18/21)

CPU 680xx Grundprogramm:

Wie schreibt man relocative Programme? (S. 18/23)

Leitartikel

Die neue GDP 64 HS

Vorstellung der überarbeiteten Grafikbaugruppe 18/4

CPU Z80

Lampensteuerung

Helligkeitssteuerung mit dem Einsteigerpaket 18/12

Tonleiter mit der Baugruppe-Sound

Kleines Orgelprogramm mit dem Einsteigerpaket 18/16

Wunschkonzert

Neue Software für den NDR-Computer 18/18

Z80 - Grundprogramm

Symboltabelle ausgeben 18/19

LOG 16 gegen LOGAN

Der Vergleichstest zwischen den beiden Logik-Analysatoren 18/21

CPU 680XX

Relativitätstheorie

Wie werden relocative Programme geschrieben ? 18/23

Großrechner <---> NDR - Computer

Datenaustausch zwischen den besagten Rechnern 18/27

mc - modular - AT

Das neue TOWER - Gehäuse

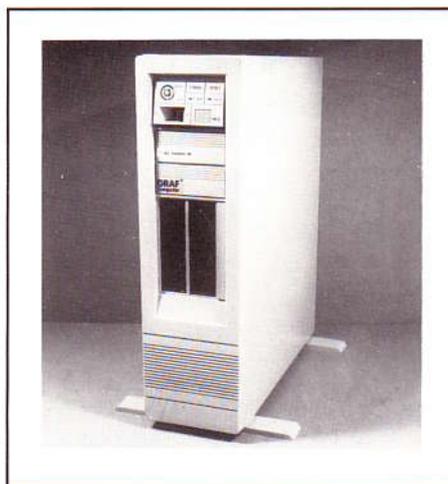
Untertisch - Gehäuse für den mc - modular - AT 18/35

Vorankündigung:

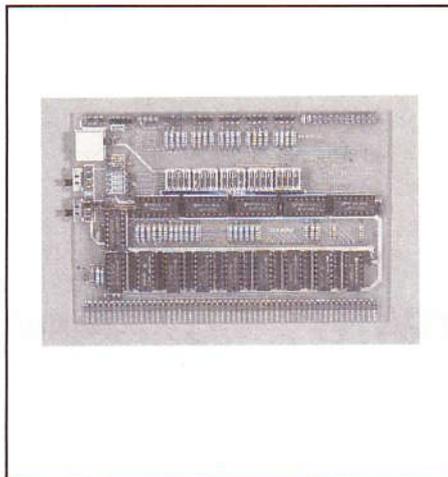
Das mc - modular - AT Sonderheft 18/35



LOOP 18 - Titelbild von GDP64HS



Neu: Tower-Gehäuse für den mc-modular-AT



Neu: Bustest-Baugruppe

Der Vergleich:
LOG16 gegen LOGAN

LOGAN

LOG16

Rubriken

Editorial 18/3

Tips und Tricks

C - Compiler
Probleme bei der Systemgenerierung 18/33

Aus der Technik

Reparaturbericht 18/33

In eigener Sache

Tag der offenen Türe bei GES 18/7
Der neue GRAF - Katalog 18/8

Softwarepartner stellen sich vor

Klaus Janßen 18/8

Jetzt lieferbar

BUSTEST - Baugruppe 18/10
TURBO - PASCAL - Tooldiskette 18/9

Kontakte 18/34

Impressum 18/2

Impressum:

LOOP Zeitung für Computerbauer

Herausgeber: Gerd Graf

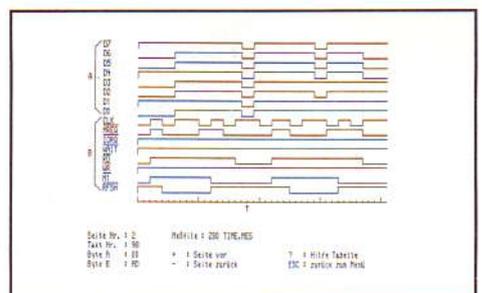
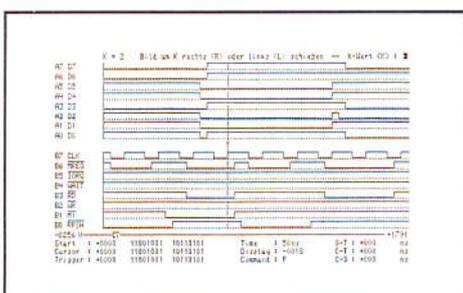
Redaktion: Rolf Dieter Klein, Gerd Graf, Christoph Köhler

Gestaltung: Christoph Köhler

Druck: Karl-Heinz Rieder, Kempten

Herstellung und Anzeigenverwaltung:
GES GmbH
Magnusstraße 13, 8960 Kempten

Anzeigenpreisliste 4/88



Liebe Leser,

Die Zeitschrift LOOP

kennen Sie bereits vermutlich seit 17 Ausgaben. Mit der hier nun vorliegenden 18. Ausgabe haben Sie es sicher schon bemerkt: Wir versuchen, aus der "LOOP" eine "richtige" Zeitung zu machen.

Das neue Layout

Dazu gehört auch etwas, das man gemeinhin mit "Layout" bezeichnet, und das sich, vergleichen Sie einmal die Ausgabe 0 bis 17, doch im Lauf der Zeit entwickelt hat.

Natürlich sind wir alle keine geborenen Journalisten, sondern mussten auch im Laufe der Zeit dazulernen. Uns stand - und steht - immer die Information unserer Kunden im Vordergrund, und nicht die Schönheit.

Viele Briefe aus Ihren Reihen haben uns aber auch erkennen lassen, daß sich eine Zeitschrift mit einem "richtigen" Layout einfach besser und leichter liest als eine Aneinanderreihung von Texten. Natürlich soll durch die so gewonnene Schönheit nicht der Inhalt leiden - dafür stehen wir gerade.

LOOP und der mc-modular-AT

Die letzte Nummer hat's gezeigt: Auch der mc-modular-AT wird nun seinen festen Platz in der LOOP finden. Einige NDR-Computer-Anwender haben sich über diese Abwanderung der LOOP bitter beklagt: Hier soll all denen die Angst genommen werden, daß aus LOOP bald eine PC-, XT-, AT oder sonstige "Kompatible" Zeitung wird. Wir werden auch in LOOP nicht jeden AT oder PC testen. Wir wollen erreichen:

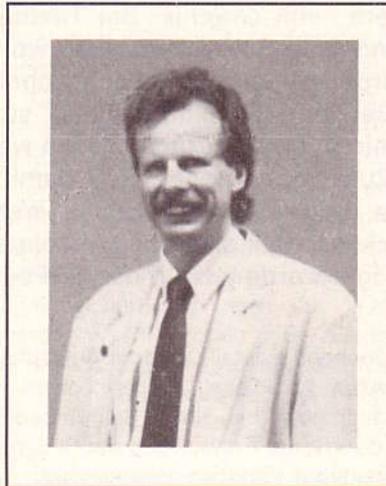
Daß unsere NDR-Kunden, die im Beruf oder auch Zuhause mit IBM zu tun haben, nicht allein gelassen werden

Daß unsere mc-modular-AT Kunden die gleiche Unterstützung auch nach dem Kauf erfahren wie unsere NDR-Kunden.

Daß unsere NDR-Kunden auch etwas über IBM erfahren, speziell auch was die Kopplung der beiden Systeme angeht.

Wir wollen nicht auf "Kosten" des NDR-

Computers arbeiten - der NDR-Computer wird auch in Zukunft in der LOOP breit unterstützt werden. Bedenken Sie bitte, daß der Umfang der LOOP von ehemals 16 Seiten - davon 4 Werbung - auf generell 32 Seiten (diesmal 36) - bei glei-



chem Preis - gestiegen ist. Allerdings werden wir wohl bald gezwungen sein, den Preis der LOOP etwas anzuheben.

OS/9 oder nicht OS/9

Selten hat ein Hinweis in der LOOP so die Gemüter erregt: Der Verzicht auf OS/9 für den NDR-Computer. Man muß erst einmal mitteilen, daß man etwas NICHT macht, um dann festzustellen, wie viele unserer Kunden nun DOCH gerne OS/9 hätten! Einer unser Software-Partner hat sogar im Rahmen einer "Selbsthilfe-Aktion" bereits während der Hannover Messe begonnen, potentielle Interessenten zu sammeln, um die 50 Lizenzen abnehmen zu können!

Nun ging es uns während der Messe fast wie dem Sterntaler-Mädchen: Ein NDR-Anwender kam zu uns auf den Stand, erzählte uns über OS/9, daß er auf eigene Kosten (!) sich einfach 50 Lizenzen gekauft und diese bereits an den NDR-Computer angepasst hätte! Wir waren schlicht stumm.

Mittlerweile hat auch schon die erste Vorführung stattgefunden; einige sehr kleine Änderungen an der Hardware müssen noch durchgeführt werden, und bereits im Spätsommer wollen wir OS/9 zunächst unseren Software-Partnern, dann unseren Kunden anbieten.

LOOP wird Sie informieren!

Dies jedoch nur als unverbindliche Vorab-Information. Wir haben noch einiges mit dem NDR-Computer, speziell auf der 68000-Seite vor. Dies als Hinweis an die NDR-User, die glauben, es "kommt nichts mehr".

Die neue GDP ist da.

Dank der Tätigkeit des Herrn Bulwien, der die Hardcopy-Aufsatzplatte entwickelte, haben wir nun daraus und aus einigen Verbesserungsvorschlägen die neue GD64HS entwickelt. Die alte GP64 Streitfrage: Geht nun der Hardware-Scroll und das Rücklesen oder nicht? ist nunmehr entschieden: Es geht.

Grundsätzlich geht es wahrscheinlich: Es gab tatsächlich eine Maskenversion des GDP9366, bei dem das Timing so verschoben war, daß auch das Rücklesen bei der neuen GDP nicht funktioniert.

Alle neuen Baugruppen werden mit getesteten GDP's geliefert - sollten Sie beim Umbauen Probleme haben, so KANN es am GDP liegen.

Für den Umbau haben wir uns einen Umbausatz einfallen lassen: Er enthält die Leiterplatte sowie alle auf der GDP eingelöteten Bauelemente. Wir haben den Preis sehr knapp kalkuliert, um möglichst allen Anwendern einen Umstieg zu erleichtern.

Ein Missverständnis sei vorab behoben: Die GDP64HS schafft nur die hardware-mässigen Voraussetzungen für Hardscroll und Rücklesen: Natürlich müssen Programme dies auch unterstützen! Ebenso natürlich arbeiten bereits viele Software-Partner daran, ihre Programme anzupassen.

Ihre Meinung ist uns wichtig

Schreiben Sie uns Ihre Meinung über die "neue" LOOP. Wir sind immer an allen Verbesserungsvorschlägen interessiert

Wir warten auf Ihre Beiträge

Die LOOP steht und fällt mit IHREN Beiträgen. Auch wenn Sie glauben, das sei doch gar nicht so toll: Schicken Sie uns Ihre Programme, Ihre Tips + Tricks und Hinweise! Wir danken Ihnen im Voraus.

Gerd Graf
und die LOOP Redaktion

Die neue GDP64 mit Hardscroll, Rücklesen und RMW

GDP64HS

Das Rücklesen wurde vorher von der Baugruppe HCOPY/MAUS erledigt, die dann entfällt. Der Anschluß der Maus wird dann auf der neuen KEY-Baugruppe (KEY3) enthalten sein.

Da die GDP64k weit verbreitet ist, haben wir uns entschlossen, außer dem Fertigergerät, dem Bausatz und der Leiterplatte noch einen Aufrüstsatz anzubieten. Dieser Aufrüstsatz soll es den bisherigen GDP64k Benutzern ermöglichen, die IC's ihrer alten GDP weiter zu benutzen, und nur die Leiterplatte, die passiven Bauelemente und natürlich die zusätzlichen Bauteile neu zu beziehen. Diese neu zu

Die GDP64k gehört seit Beginn der NDR-Fernsehserie zum NDR-Computer. Sie war die erste Standardgraphikkarte für diesen Rechner und war sowohl für Textdarstellung als auch für Graphikdarstellung geeignet. Allerdings hatte sie zwei Nachteile, sowohl in der Textdarstellung als auch beim Abspeichern von Graphik. Bei Textdarstellung war vor allem der langsame Scroll bemängelt worden, der jetzt durch eine Hardscrolllogik erheblich schneller gemacht werden kann. Bei der Graphikdarstellung störte meistens, daß Graphiken nicht direkt zurückgelesen werden konnten (nur mit einer Zusatzbaugruppe) und damit auch nicht direkt auf den Drucker ausgegeben werden konnten. Auch diese Rückleselogik gekoppelt mit einer RMW (Read-Modify Write) Logik wurde jetzt in der GDP64HS mit aufgenommen.

beziehenden Bauteile sind im Aufrüstsatz enthalten. An dieser Stelle möchten wir uns auch noch bei Herrn Bullwin bedanken, der bei der Entwicklung der Baugruppe wertvolle Vorarbeit geleistet hat.

In den folgenden Abschnitten wollen wir etwas näher auf die GDP64HS eingehen und anhand des Blockschaltbildes grob die Funktion der Baugruppe erläutern. Detailliertere Information erhalten Sie natürlich wie immer aus unseren Handbüchern, wobei das neue GDP64HS-Handbuch mit rund 100 Seiten besonders ausführlich geworden ist.

Das Herz der GDP64HS ist der Graphikprozessor 9366 von Thomson.

Auf diesen Prozessor hier näher einzugehen wäre sinnlos und würde den Rahmen dieses Artikels sprengen. Deshalb wollen wir hier nur kurz einige Characteristica des Prozessors erläutern.

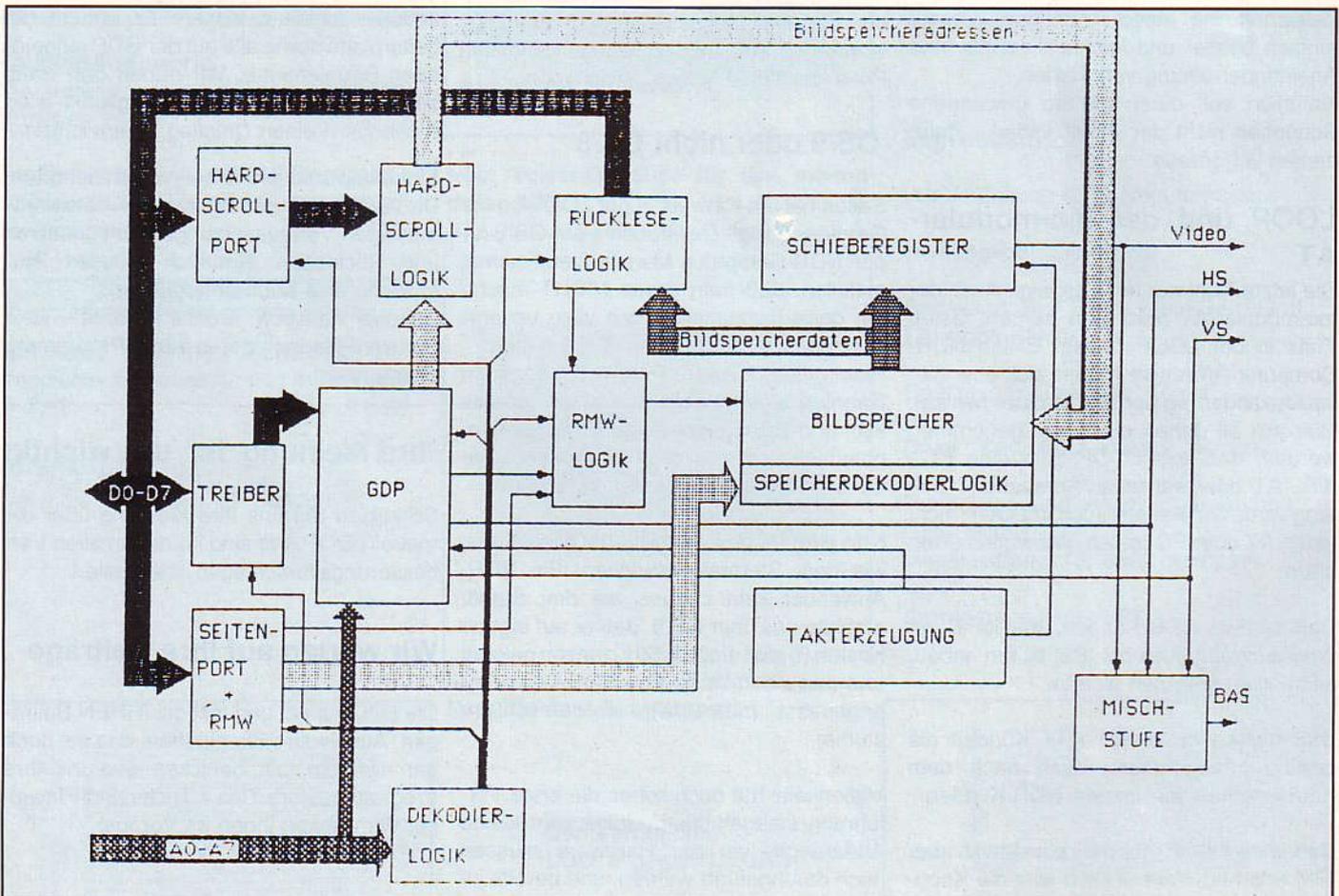


Bild 1: Blockschaltbild der GDP64HS

Der 9366 hat den kompletten ASCII-Satz integriert; diese Zeichen können mit Hilfe des sogenannten "CSIZE" Registers noch in verschiedenen Größen dargestellt werden. Außerdem besteht die Möglichkeit, Zeichen kursiv und senkrecht darzustellen. Andererseits besitzt der 9366 auch einige gute Graphikmöglichkeiten. Hier ist vor allem die Vektorgraphik zu erwähnen. Um extrem schnelle Graphiken zu erzeugen, kann mit sogenannten Kurzvektoren gearbeitet werden, die mit Hilfe nur eines Befehlsbytes an den Graphikprozessor übergeben werden. Es besteht auch die Möglichkeit mit Blockgraphik 8 x 5 oder 4 x 4 zu arbeiten.

Der 9366 verwaltet seinen Bildschirmspeicher selbst. Er kann direkt 16KByte adressieren, was bei einer Auflösung von 512 x 256 Bildpunkten einer Bildschirmseite entspricht. Will man mehr als eine Bildschirmseite verwalten, muß wie hier eine spezielle Logik bereitgestellt werden.

Von der Haupt-CPU (z.B. von der CPU68008) wird über die Adressleitungen A0...A3 eines der 16 Register des GDP (EF 9366) angewählt. Soll z.B. ein Vektor gezeichnet werden, so teilt man dem Graphik-Prozessor lediglich den Anfangs- und Endpunkt mit. Die Zwischenwerte werden von ihm selbst berechnet und dann in den Speicher abgelegt.

Der interne Aufbau des Speichers wird durch den Grafikprozessor und sekundär durch die Speicherdekodierlogik organisiert. Im Speicher steht dann die Information, die später auf dem Bildschirm erscheint. Um den Ablauf des "Displays" aus dem Speicher etwas zu verdeutlichen, haben wir ein kleines Beispiel angeführt.

Beispiel: Wir verfolgen das Auslesen eines Bytes vom Speicher zum Monitor (z.B. 10001110)

Das Byte steht am Ausgang des Speichers und wird beim Aktivieren des Signals SH/L (Shift Load, am Schieberegister) parallel in das Schieberegister eingelesen. Hier wird das Signal mit dem Punktetakt (14 MHz, CLK) verknüpft und seriell (in der Punktfolge 10001110) zum Monitor hinausgeschoben. Da jedes Bit einen Bildpunkt darstellt, werden jetzt 8 Punkte auf dem Bildschirm angezeigt. Ein dunkler Punkt entspricht einer 1 und ein heller einer 0. (ebenfalls in der Reihenfolge 10001110), siehe Bild 2.

Der Speicher ist für 4 Bildschirmseiten aufgebaut, die durch die Seitenumschaltung ausgewählt werden können. Es kann in eine Seite geschrieben und zugleich eine

weitere gelesen werden. Das aus dem Schieberegister kommende Videosignal geht entweder direkt zum Monitor (TTL) oder zum Videomischer. In dieser Mischstufe wird das Video-Signal mit horizontalen und vertikalen Synchronisationssignalen (HS und VS) so aufbereitet, daß es danach als BAS-Signal (Bild Austast Synchron Signal) zur Verfügung steht.

Will man eine Hardcopy erstellen, muß man ähnlich der Datenübertragung zum Monitor den Bildspeicher auslesen. Das Auslesen des Bildschirmspeichers läuft parallel zum Display (Ausgeben der Bildschirminformation). Durch einen Befehl an den Graphikprozessor kann jeweils ein Byte des Bildschirmspeichers in die Rücklese Logik gelesen werden und dann über den Datenbus im Arbeitsspeicher abgelegt werden. Durch eine Druckeroutine kann nun das Bild auf einen Drucker ausgegeben werden. Das Auslesendes Bildschirmspeichers ist auch für andere Aufgaben sehr nützlich (z.B. Vergleich von Bildschirmseiten, Abfrage des Mauszeigers für Graphikprogramme, Window-technik, usw.)

Die Hardware-Scroll-Logik dient zum

Scrolladresse, die über Port 61 ausgegeben werden kann, aufaddiert und damit die neue Bildspeicheradresse erzeugt. Dadurch kann der Bildschirm zyklisch gescrollt werden, natürlich mit einer größeren Geschwindigkeit und "sanfter" (smoothscroll).

Mit Hilfe des RMW-Modus kann man einfach bewegte Bilder erzeugen, da bei Schreibvorgängen in den Bildspeicher (=Monitor) alle Punkte komplementiert werden. Wenn z.B. ein Punkt gesetzt war, so wird er gelöscht, wenn er nicht gesetzt war, so wird er eingeschrieben. Der größte Vorteil des RMW-Modus ist, daß zerstörungsfrei gezeichnet werden kann. Dies ist z.B. nötig, um einen Maus-Zeiger oder ein Fadenkreuz auf einer Graphik oder einem Bild darzustellen, ohne das Hintergrundbild zu zerstören.

Die Dekodierlogik mit den Adressen A4...A7, den Signalen IORQ und M1 wird benötigt, um die GDP64HS bei I/O-Zugriff von 70...7F (Graphikprozessor) und 60...63h (Seitenport, RMW-Mode, Hardcopy und Hardscroll) anzusprechen.

Die Takterzeugung stellt alle benötigten Frequenzen (Pixelclock, Clock für Schie-

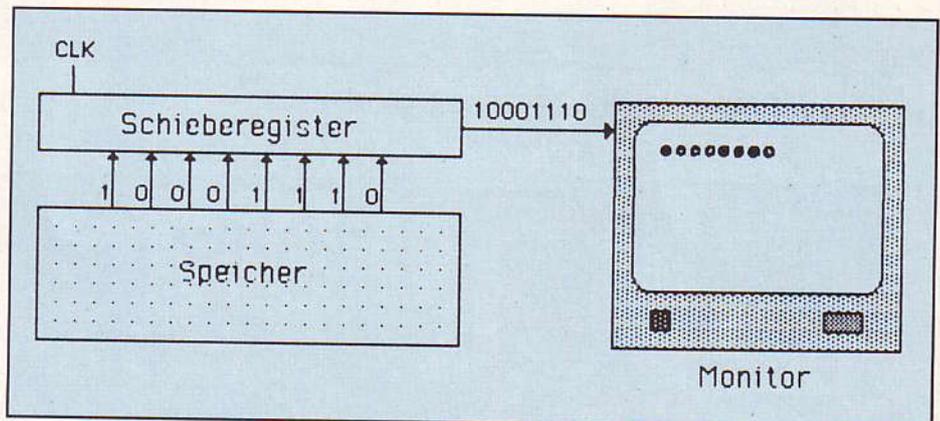


Bild 2: Prinzipielle Datenübertragung vom Speicher zum Monitor

Rollen (Scroll) des Bildschirms. Bei der bisherigen GDP64k wurde der Scroll softwaremäßig erzeugt. Sollte der Bildschirm bei Textdarstellung um eine Textzeile nach oben gescrollt werden, so mußte das gesamte Bild noch einmal aufgebaut werden (um eine Zeile versetzt). Dadurch war der Scroll natürlich sehr langsam und nur zeilenweise (Textzeile = 8 Bildschirmzeilen) möglich. Der Hardware-Scroll behebt diese beiden Mängel. Dabei wird der Bildschirmspeicher beim Scrollen nicht mehr verändert, sondern nur die Adressierung des Bildschirmspeichers. Dies wird durch zwei Addierer erledigt, die lediglich die Bildspeicheradressen, die vom Graphikprozessor kommen, mit einer

beregister, Takte für die Adressierung der Speicher) aus dem Grundtakt von 14 MHz her.

Im neuen Grundprogramm 680xx wird die GDP64HS mit RMW, Hardscroll und Rücklesen bereits unterstützt. Beim Z80 ist ein neues FLOMON in Vorbereitung, das diese Optionen ebenfalls unterstützen wird. Sollten Sie die GDP mit der alten Software verwenden, ist diese natürlich absolut aufwärtskompatibel und verhält sich wie die alte GDP64k. Im Handbuch sind sowohl für Z80 als auch für 680xx Programme enthalten, die die neuen Funktionen der GDP64HS unterstützen.

Technische Daten

Spannungsversorgung	+5V	Speicher	64k RAM (dynamisch) - dadurch 4 Seiten
Stromaufnahme	500 mA		
Busformat	NDR -Bus 54-polig ECB - Bus 64-polig	Sonstige Funktionen	Read Mody Write (zerstörungsfreies Zeichnen auf dem Bildschirm) Hardcopy (Rücklesen des Bildschirminhaltes und Ausgabe auf einen Drucker) Hardscroll (Scrollen des Bildschirmes mit Hilfe der Hardware, d.h. die Adressen des Bildschirmspeichers werden hardwaremäßig aufaddiert)
Leiterplattenformat	160 x 100 mm (Europakarte)		
Ausgang	1: BAS: beinhaltet HS, VS und Video-Signal 2: TTL: beinhaltet HS, VS und Video-Signal (invertierbar)		
Grafik-Controller	EF 9366 (Thomson-EFCIS) - kann 4 Seiten bedienen, wobei in eine geschrieben und zugleich eine weiter gelesen werden kann - integrierter ASCII-Zeichensatz		

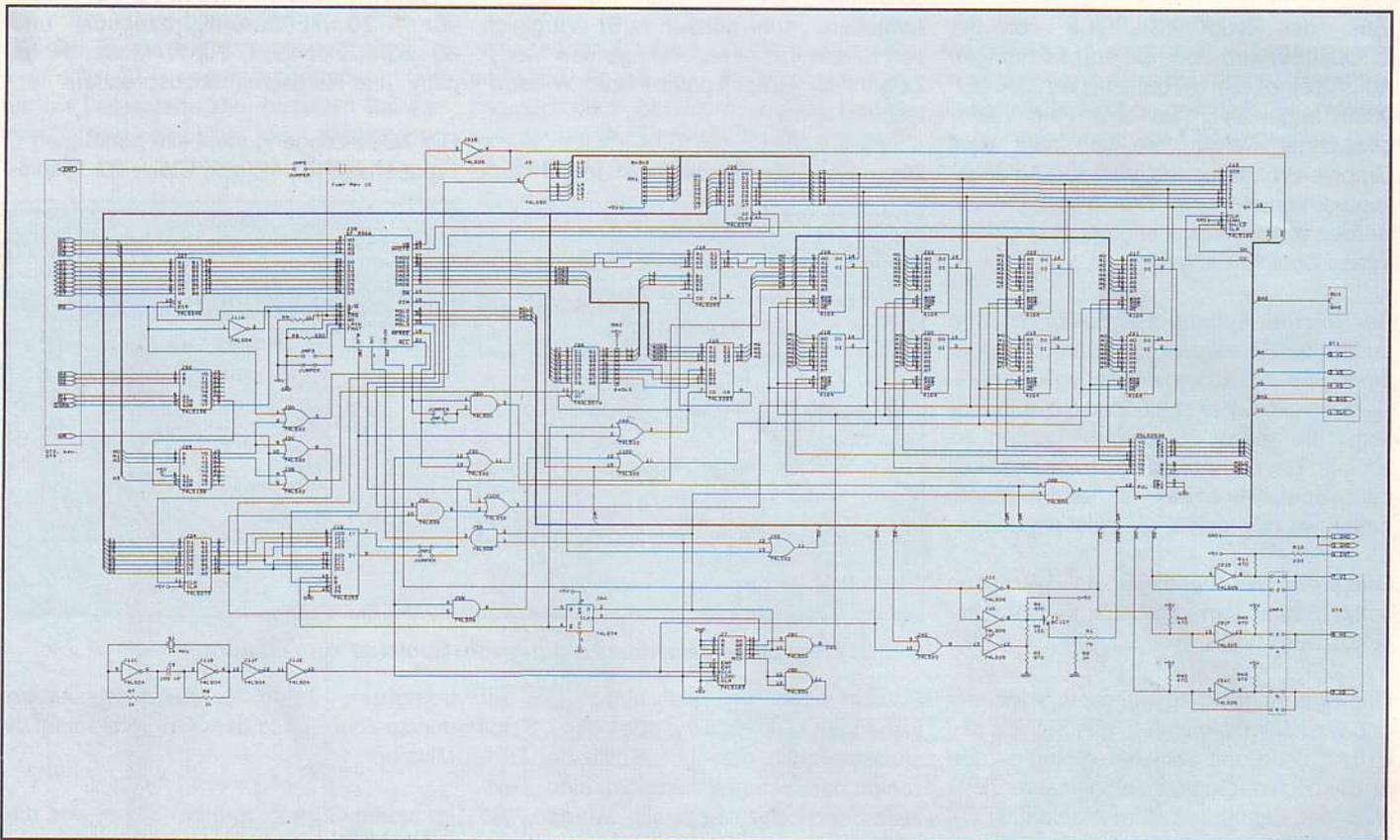


Bild 1: Schaltplan der GDP64HS

Spruch der Woche:

Bildung ist das, was übrig bleibt, wenn man alles vergessen hat,
was man in der Schule gelernt hat

Albert Einstein

10 Jahre Graf - Computer

Geburtsstagsfeier mit Tagen der offenen Türe bei GES

Am 15. und 16. April feierten mehr als 2000 Besucher aus Kempten und Umgebung mit uns Geburtstag und nahmen diese Gelegenheit wahr, sich die neuen Geschäftsräume in unserem Hause an-

zusehen. Ein besonderer Gag war das Torwandschießen, bei dem man einen BMW 320 i als Hauptpreis gewinnen konnte.

Bei strahlendem Sonnenschein und som-

merlichen Temperaturen kam auch der gemütliche Teil nicht zu kurz. Alle Besucher zeigten sich von den neuen Räumen und den technischen Vorführungen beeindruckt.

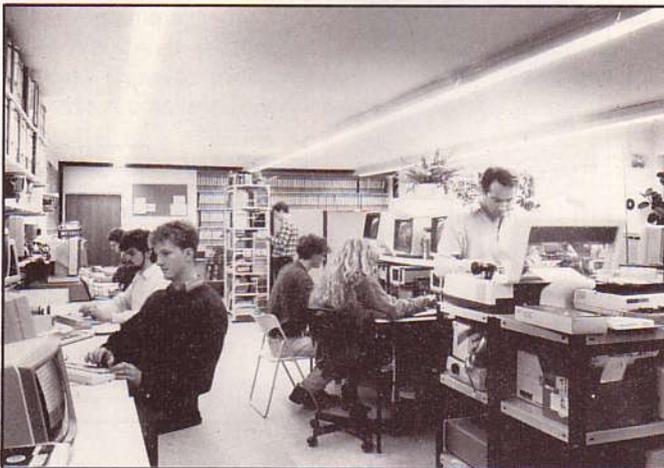


Bild 1: Blick in die Entwicklungsabteilung

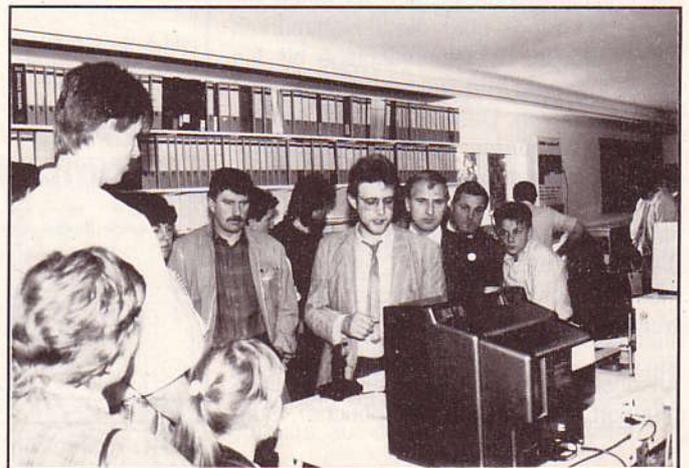


Bild 2: Besonderes starkes Interesse fanden die Führungen durch die Räumlichkeiten



Bild 3: Ein "schlagkräftiges" TEAM: GES zählt heute 40 Mitarbeiter

Neuer Katalog lieferbar

Endlich ist er da, die zweite Ausgabe des Farbkataloges mit noch mehr Information, mehr Produkten und Abbildungen. Er enthält statt bisher 168 jetzt 208 Seiten. Außerdem haben wir natürlich aus der ersten Ausgabe gelernt und auch für die Übersichtlichkeit bzw. Überblickbarkeit dieses "Nachschlagewerkes" etwas getan.

Die einzelnen Kapitel sind nicht mehr nach dem Einstiegsprinzip geordnet, sondern nach den einzelnen Funktionseinheiten eines Computers. So sind z.B. CPU's in einem eigenen Kapitel, dann Speicherbaugruppen, usw. Vor diesen Kapiteln über die Funktionseinheiten gibt es einen erklärenden Block über die Einsetzbarkeit dieser Blöcke, Systemvorschläge bis hin zur Erläuterung von Datenbusbreiten und Taktfrequenzen. Danach wird die Software - wieder geordnet - und die Teachware (Bücher, Literatur) vorgestellt.

Neu in den Katalog aufgenommen wurden die Kapitel "mc-modular AT" und "Logiksimulator". Vor allem dem "mc-modular AT" wurden mehr als 10 Seiten gewidmet.

Ansonsten sind wie bisher sämtliche Baugruppen, Leiterplatten, Handbücher, Soft-

ware und sonstige Dokumentation und Schulungssoftware zum NDR- und mc-Computer enthalten. Auch hier sind natürlich neue Baugruppen und vor allem viel neue Software enthalten.

Für die Anwender des NDR- und mc-Computer, sowie des mc-modular AT ist der neue Katalog das einzig übersichtliche Nachschlagewerk. Da der NDR-Computer und auch der Logiksimulator große Anteile an dem Mikrocomputer Ausbildungsmarkt hat, sollte dieser Katalog in keiner Schule oder Ausbildungsbetrieben, die nur irgendwie mit Mikroelektronik arbeiten, fehlen.



Bild 1: Ausschnitt aus dem neuen GES - Katalog

Softwarepartner stellen sich vor:

Persönliche Daten

Klaus Janßen
Hanninxweg 74
415 Krefeld 1

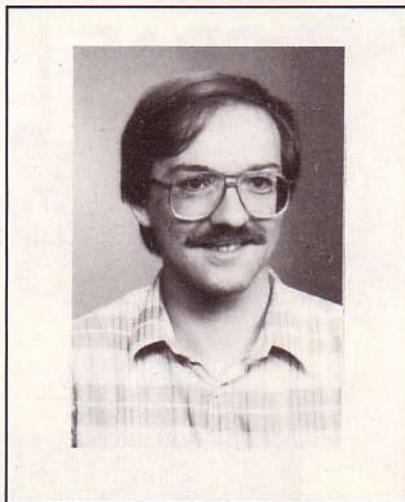
31 Jahre
 verheiratet

Ausbildung und Beruf

1974 Abitur
 1976-1982 Studium Elektrotechnik, Fachrichtung Nachrichtentechnik, an der Universität Duisburg 1982-1984 Wissenschaftlicher Mitarbeiter im Fachgebiet Meß- und Regelungstechnik seit 1984 Software-Entwicklungs-Ingenieur in der Industrie

Programmiere

seit circa 8 Jahren, zunächst in Fortran und Basic, später in Pascal, 8086-Assembler, 68000-Assembler, weniger in Modula 2, am liebsten in Pascal!



Mit dem NDR-Klein Computer (68008-Prozessor) begann ich - in meiner Freizeit - im Frühjahr 1985, weil es sich um ein offenes System mit einem leistungsfähigen Prozessor handelt, ich den 68000-Prozessor kennenlernen wollte und er ein breites Betätigungsfeld für die Softwareentwicklung bietet.

Mit JADOS

glaube ich, ein sehr leistungsstarkes Diskettenbetriebssystem für den NDR-Computer entwickelt zu haben. Meine weiteren Produkte sind der Disassembler DISASS, der Diskettendoktor INSPEC, das Strategiespiel REVERSI und zahlreiche kleinere Utilities.

Zur Zeit (Herbst 1987)

entwickle ich einen komfortablen Editor und nach einer Entscheidung über das Object-File-Format werde ich einen Linker und einen Assembler für das JADOS-System in Angriff nehmen. Ich hoffe, es wird sich jemand finden, der dann gleichzeitig an einem Compiler baut, denn meine Zeit ist leider begrenzt!

MeinTip für Anfänger:

Ein Programm immer erst auf dem Papier entwickeln!

NEU!

Die TURBO-TOOL-Diskette

Zum Betrieb der Diskette wird das Betriebssystem CP/M benötigt. Die Programme können über das jeweilige .PAS - File (=Pascal Quelltext) oder .COM - File gestartet werden.

Die Programme dienen teilweise als Software-Werkzeuge im Umgang mit Turbo-Pascal, teilweise als Anwenderprogramme mit Lerneffekt oder spielerischem Charakter.

Nach unserem "Aufruf an alle Software-Freaks !" in LOOP 15/3 können wir Ihnen jetzt als erstes Ergebnis dieser Aktion die TURBO-PASCAL-TOOL Diskette vorstellen. Die Diskette wurde ausschließlich aus Kundenroutinen erstellt. Für die vielen zugesandten Programme dürfen wir uns noch einmal ganz herzlich bedanken.

Vokabeltrainer

zum Einüben von bis zu 100 Vokabeln. Abfrage der Vokabeln erfolgt per Zufallsgenerator.

Diskettenverwaltung
Druckerausgabe im Diskettenformat
Erfassen und Sortieren der Disketten

Mathematische Funktionen

Berechnung mit 3 Real - Variablen

Puzzle Spiel

unterstützt von Mouse-Tools

Mühlespiel

Jeder kennt dieses Spiel. Die Vorlage zu diesem Spiel wurde CHIP-SPEZIAL entnommen und für die Grafischen Möglichkeiten des NDR-Computers umgeschrieben. (siehe Bild 2)

An dieser Stelle noch einmal vielen Dank für freundliche Genehmigung der CHIP-Redaktion für die Verwendung des Programms auf der Turbo-Tool-Diskette.

Virgula

"Würfelspiel" für 2-6 Personen

Routinen zur Ansteuerung der COL256
Abspeichern und Laden des Bildschirminhaltes

Ansteuerung der CLUT

Ansteueroutine für die Grafik der GDP64K

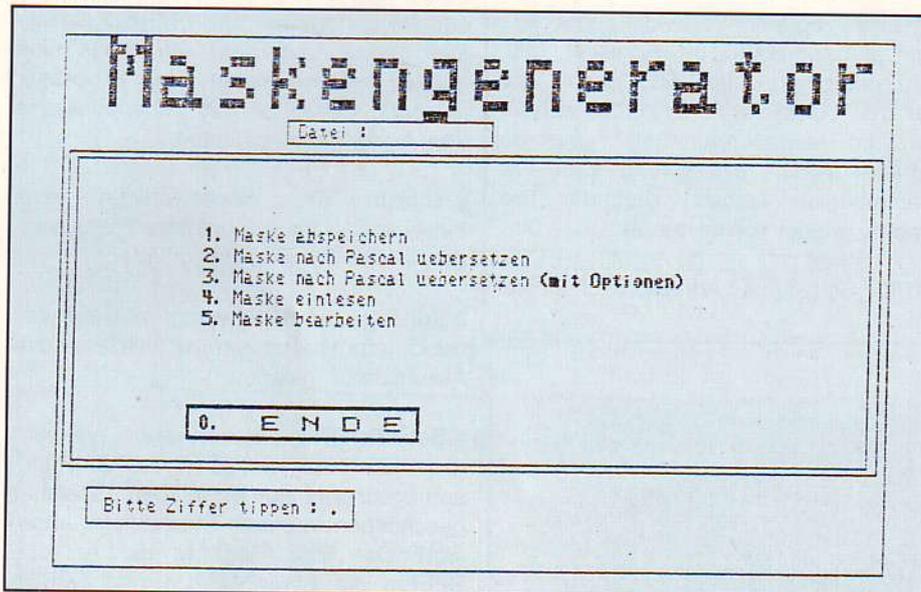


Bild 1: Maskengenerator für Turbo-Pascal-Programme

Eine kurze Beschreibung für einen Teil der Programme:

Maskenroutine

Das Programm Maskengenerator dient zum Erstellen von Bildschirmmasken unter Turbo-Pascal. Es verarbeitet den Text auf dem Bildschirm mit allen angegebenen Ein- und Ausgaben in Pascal Source-Code. Die Graphik des NDR-Computers wird hierbei voll berücksichtigt.

Die Copyright Meldung in Bild 1 wurde mit dieser Maskenroutine erstellt.

Vokabelprogramm

zum schnellen Nachschlagen von beliebigen Vokabel bzw. Übersetzungen als Vokabellexikon universell verwendbar

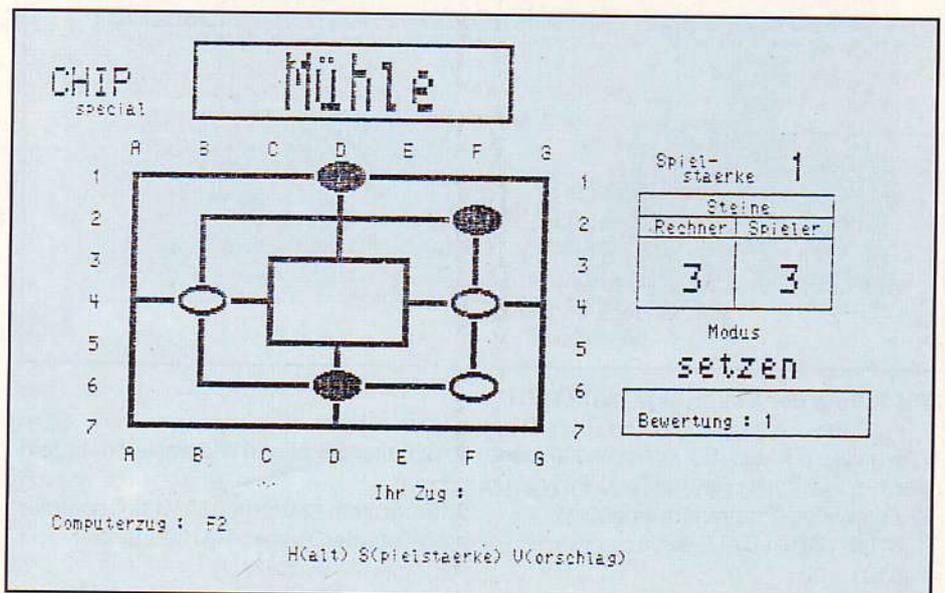


Bild 2: Mühlespiel unter Turbo-Pascal

Klaus Bischof

Verstehen von Rechnersignalen mit der neuen Baugruppe

BUSTEST

Mit der Baugruppe BUSTEST kann man jetzt die Abläufe des Rechners sichtbar machen und somit sehen, was im Rechner wirklich vor sich geht. Diese Baugruppe hält den Rechner nach jedem Befehlszyklus an und stellt die entsprechenden Rechnersignale (die auf dem BUS liegen) mit Leuchtdioden dar. Mit dem folgenden kleinen Programm wollen wir Ihnen zeigen, wie Sie mit Hilfe des Einsteigerpaketes und der BUSTEST Rechnerabläufe darstellen können.

Anwendungsbeispiel mit dem Einsteigerpaket Z80

Mit Hilfe dieses Beispiels sind die Speicherzugriffe (Lese- und Schreib-Befehl) sehr gut zu erkennen. Dies kann durch schrittweises Abarbeiten der nachfolgenden Punkte geschehen.

Jeder hat einmal klein angefangen. Dies gilt wohl auch in der Mikrocomputerei und natürlich auch für den NDR-Anwender. Dabei steht meistens am Anfang ein riesiges Loch bezüglich der Hardware und der Hardwareabläufe. Will man in die Hardware einsteigen, sieht man sich gezwungen, Stapel von Literatur, speziell Datenblätter, durchzuwühlen und kann das Ganze natürlich nur theoretisch (daher auch meist sehr trocken) bearbeiten.

```
8104 00
8105 81
8200 C9 RET
```

2.1 Das Programm verzweigt zunächst in ein Unterprogramm. In Adresse 8200 wird durch den Befehl "RET" sofort wieder ins Hauptprogramm zurückgesprungen. Im Hauptprogramm steht unter der Adresse 8103h ein Sprung zum Programmbeginn. Danach läuft das Programm wieder von vorne ab.

3. DIL - Schalter auf Adresse 8100 einstellen

7. Durch Taster das Programm Schritt für Schritt ablaufen lassen.

Dabei ergeben sich folgende Bitmuster:

(siehe Bild 2 --->)

Die oben abgebildeten Bitmuster entsprechen den Leuchtdioden auf der Bustest Baugruppe. Im folgenden Absatz werden diese Bitmuster kurz erläutert.

1.Schritt: Auf Adresse 8100 steht der Befehl CD (CALL). Die Steuersignale RD und MREQ werden aktiv. MREQ signalisiert, daß auf dem Adressbus eine gültige Adresse anliegt. Der Prozessor ist bereit, den Inhalt des Datenbusses in den Akkumulator zu laden.

2.Schritt: Die niederwertigen zwei Bytes der Unterprogrammadresse werden in den Akkumulator geladen.

3.Schritt: Die höherwertigen zwei Bytes der Unterprogrammadresse werden in den Akkumulator geladen.

4.Schritt: Das Steuersignal WR wird aktiv. Die höherwertigen Bytes der Rücksprungadresse werden in den Speicher geschrieben (auf dem Stackpointer abgelegt). Der Stack liegt ab der Adresse BFFFh; die Bytes BFFEh und BFFDh werden nun beschrieben.

5.Schritt: Die niederwertigen Bytes werden in den Speicher geschrieben.

6.Schritt: Das Programm ist auf die Adresse 8200 gesprungen. Hier wird der Befehl C9 (RTS) in den Prozessor gelesen.

7.Schritt: Die höherwertigen Bytes der Rücksprungadresse werden vom Stackpointer geholt.

8.Schritt: Die niederwertigen Bytes der Rücksprungadresse werden vom Stackpointer geholt.

9.Schritt: Der Befehl C3 (JP) auf Adresse 8103 wird vom Prozessor gelesen.

10.Schritt: Die niederwertigen Bytes der Sprungadresse werden gelesen.

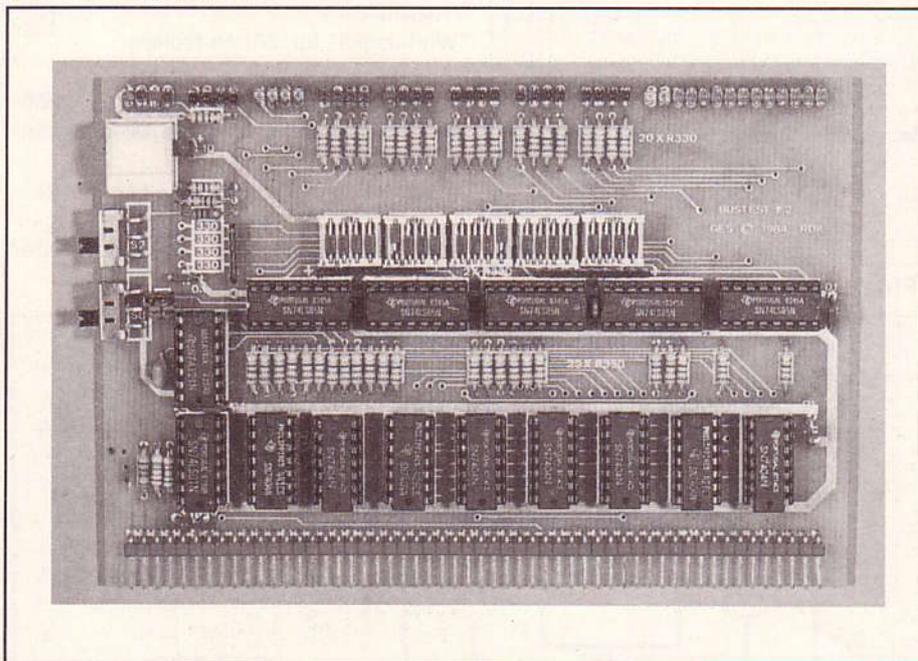


Bild 1: Foto der Baugruppe "BUSTEST"

1. Schalter S7 und S8 schließen (Hebel oben)
2. Folgendes Programm eingeben:
8100 CD CALL 8200
8101 00
8102 82
8103 C3 JP 8100
3. Schalter S7 öffnen (Einzelschritt)
4. Schalter S8 öffnen (Adresse einfangen)
5. Programm starten (Der Computer bleibt bei der Adresse 8100 stehen)

Schritt	Spg.	Daten	Adresse	Steuersignale
1	+5V	C D	0 8 1 0 0	RD MREQ
2	+5V	0 0	0 8 1 0 1	
3	+5V	8 2	0 8 1 0 2	
4	+5V	8 1	0 B F F E	WR MREQ
5	+5V	0 3	0 B F F D	
6	+5V	C 9	0 8 2 0 0	M1
7	+5V	0 3	0 B F F D	
8	+5V	8 1	0 B F F E	
9	+5V	C 3	0 8 1 0 3	
10	+5V	0 0	0 8 1 0 4	
11	+5V	8 1	0 8 1 0 5	

LED leuchtet
 LED leuchtet nicht

Bild 2: Bitmuster

11.Schritt: Die höherwertigen Bytes der Sprungadresse werden gelesen.

12.Schritt: Das Programm beginnt wieder von vorne.

An diesem Beispiel läßt sich der Ablauf

des Rechners schön darstellen. Vor allem kann man erkennen, welche Signale wann aktiv sind.

Die Baugruppe BUSTEST ist vor allem für die Selbstausbildung gedacht.

Sollte sie dann als Ausbildungsbaugruppe ihren Dienst getan haben, kann sie

aber immer noch sinnvoll bei der Fehlersuche - auch in Programmen - eingesetzt werden.

Nikolaus Bischof, GES

Theorie ist

wenn man alles weiß
und nichts funktioniert

Praxis ist

wenn alles funktioniert
und niemand weiß warum

Gunter Mader

Einsteigerpaket steuert Glühbirne

Hierzu benötigen wir nur einige wenige Bauteile, wie ein Fahrradlämpchen 6V mit entsprechender Fassung und ein Leistungstransistor TIP 120 (zwei Transistoren in Darlingtonschaltung). Dieser Baustein wird hardwaremäßig wie ein Transistor behandelt. Sehen Sie sich hierzu auch Bild 1.0 an.

Prinzip der Lampenansteuerung

Bevor wir uns der notwendigen Software zuwenden, lassen Sie mich einiges zur wirklich einfachen Hardware dieses Versuches sagen. Die Schaltung besteht im Prinzip aus dem Transistorbaustein und Lampe mit Fassung. Die Lampe wird im Kollektorzweig des Transistors betrieben. Ein gewisser positiver Spannungspegel an der Basis schaltet den Transistor durch - die Lampe leuchtet.

Damit nun die Lampe aufleuchtet, kann man die Basis des Transistors auch durch den Rechner mit Rechteckimpulsen ansteuern. Diese werden durch ein

Bei einer ersten Betrachtung des Schaltungsaufbaus wird sich der Leser fragen, welchen Sinn es hat, mit einem Computer ein Fahrradlämpchen anzusteuern. Nun, mit diesem Versuch wollen wir aufzeigen, wie man mit dem Rechner auf Tastendruck z.B. einen kleinen Motor mit mehr oder weniger Leistung, zeitlich begrenzt, betreiben kann. Zur Veranschaulichung dessen soll uns die Ansteuerung eines Fahrradlämpchens über Port 31h des NDR-Einsteigerpakets dienen.

Programm erzeugt und bestimmen, abhängig von Impulsdauer und Impulsfolge, wie oft der Transistor durchgeschaltet wird, bzw. wie hell dann die Lampe aufleuchtet. Die Lampe wird also immer durch unser Programm gesteuert - ganz kurz ein- und ausgeschaltet.

Erläuterungen zur Hardware

Sehen Sie sich nun Bild 1.2 an. Es zeigt acht Impulsdigramme mit denen sich aufzeigen läßt, warum die Lampe bei entsprechender Ansteuerung mehr oder weniger hell aufleuchtet.

Die Erklärung dieser Diagramme folgt nun anhand eines Modells. Stellen Sie sich ein ACHT-BIT-REGISTER vor (siehe

auch Einsteigerpaket, Christiani Band 2).

Füllt man dieses REGISTER-MODELL schrittweise von rechts nach links mit 'HIGH-BITS', so ergeben sich acht sedezimale Zahlen (00;01;03;07;0f;1f;3f;7f;ff), die entsprechenden Helligkeits-

werten zugeordnet werden können (siehe Bild 1.2).

Bei '00' bleibt die Lampe dunkel, bei 'ff' leuchtet sie am hellsten.

Die Ansteuerung und damit die Ausgabe des Helligkeitswertes hängt nun in erster Linie von der Impulsdauer und der Impulsfolge an der Basis des Transistorelements ab. Die Impulsdauer können wir mit unseren sedezimalen Zahlenwerten einstellen. Die Impulsfolge hängt von der Länge des 'LAMPEN-TRIGGERPROGRAMMS' ab und damit von der Zeit, die der Rechner für die Abarbeitung dieses Programms benötigt. Mit anderen Worten, je öfter und länger an der Basis Signal HIGH ansteht, umso heller leuchtet die Lampe auf.

Bild 1.0 Bauform Tip 120

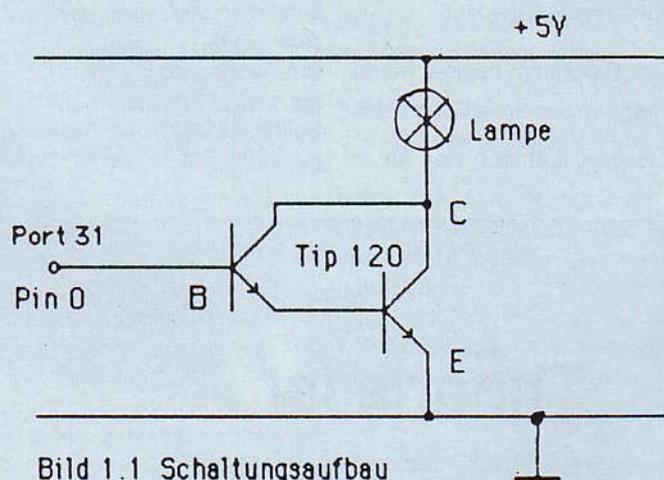
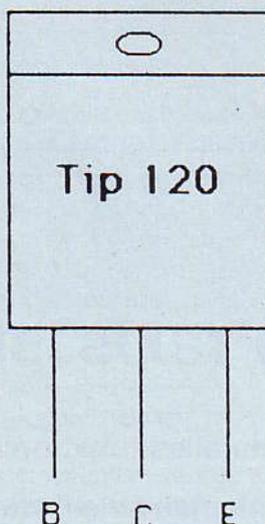
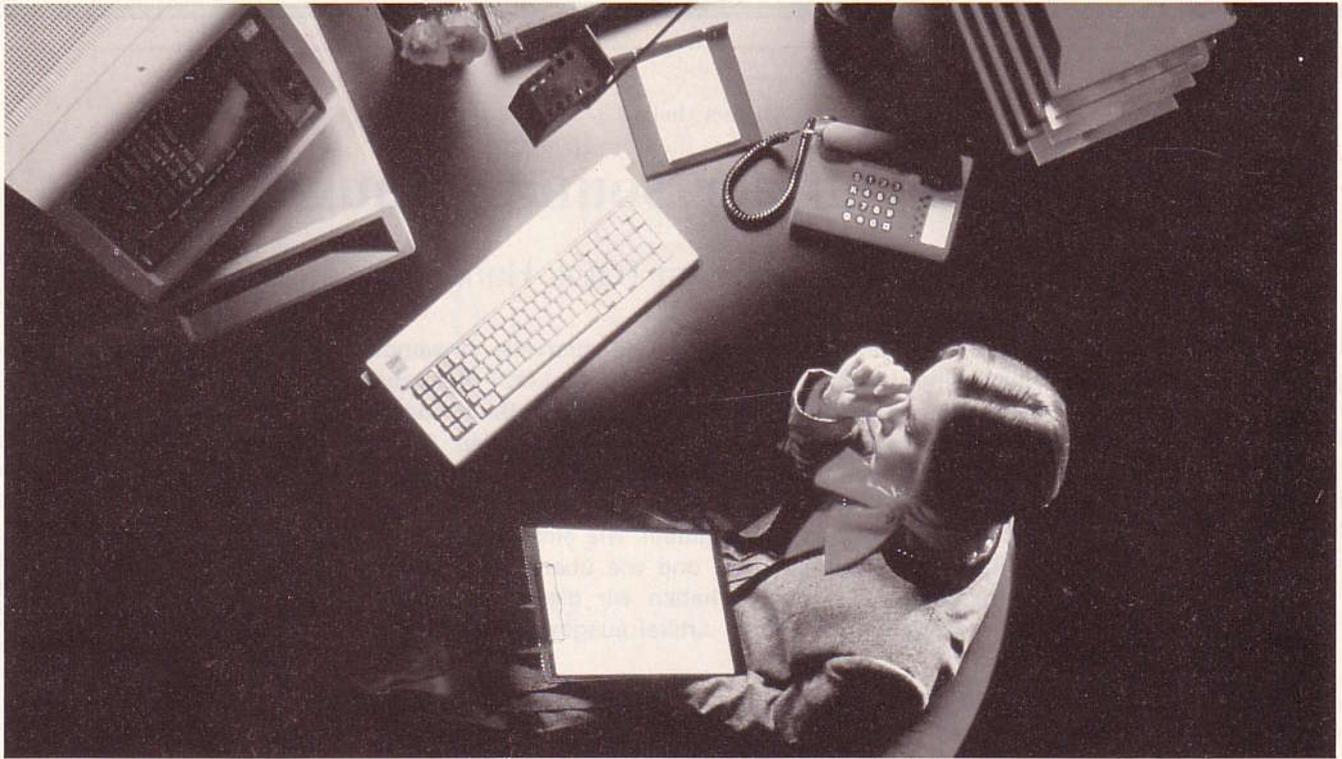


Bild 1.1 Schaltungsaufbau



Durch mc wissen Computer-Profis,
was der Markt zu bieten hat:

In der Juli-Ausgabe zeigt mc
u. a. 100 Matrixdrucker.

Mit mc sind Sie bestens informiert.
Denn mc wird für Profis gemacht, die mit
und durch den Computer Geld verdienen.
Sie lesen u. a. in der Juli-Ausgabe:

Computer-Tuning

Wenn ein PC-XT oder AT zu langsam ist,
kann man ihn mit Einsteckkarten
schneller machen. Höchstgeschwindig-
keiten erreicht man mit dem Inboard-386.

Künstliche Wesen

Wie sich biologische Schaltvorgänge in
Modula 2 simulieren lassen.



Workshop

Im mc-Juli-Heft beginnt die neue Serie
„C-Workshop“. Fundiertes Wissen für
die Praxis. Präprozessor-Kommandos für
Turbo C.

mc-Report

Über Hacker und Computerviren...

mc gibt es jeden Monat neu an jeder
größeren Zeitschriften-Verkaufsstelle zum
Preis von DM 7,-.

Die Juli-Ausgabe ist ab 27. Juni
im Handel



**Die Mikrocomputer-Zeitschrift -
bringt Profis weiter.**

Sollte die neueste mc an Ihrer Zeitschriften-Verkaufsstelle vergriffen sein, senden wir Ihnen einmalig ein kostenloses Probeheft.
Frankierte Postkarte an Franzis-Verlag, Postfach 37 02 80 L18, 8000 München 37, mit Berufsangabe und Vermerk
„Bitte Probeheft mc“ oder Anruf zum Ortstarif (Tel.: 01 30-51 17) genügt.

Michael Giuliani

Tonleiter auf der Sound - Baugruppe

Ein Auszug aus dem TOOL-Handbuch

Das Programm Sound1 belegt die 13 Tasten 0-C mit den Tönen einer Oktave von C-C' (einschließlich der Halbtöne). Zusätzlich kann mit der Plus-Taste die Lautstärke erhöht, mit der Minus-Taste erniedrigt werden.

Die Sound-Baugruppe wird über die Adressen E0h und E1h angesteuert.

Dabei wird über die Adresse E0h ein internes Register der Karte ausgewählt, über die Adresse E1h werden dann die Daten an das angewählte Register ausgegeben.

Daß mit dem Einsteigerpaket bereits sinnvolle Anwendungen möglich sind, wurde bereits in der letzten LOOP-Ausgaben bewiesen. Alle unsere Standard EIN/AUSGABE-Baugruppen sind mit dem Einsteigerpaket programmierbar. Um den Anwendern das Programmieren zu erleichtern, haben wir, wie bereits in der letzten LOOP erwähnt, ein TOOL-EPROM mit Handbuch geschaffen. Um Ihnen ein Bild zu vermitteln, wie einfach die Baugruppe SOUND zu programmieren ist und wie übersichtlich das Handbuch dazu gestaltet wurde, haben wir diesen Ausschnitt aus diesem Handbuch für diesen Artikel ausgewählt.

Programmablauf

Zuerst wird das Register 7 angewählt und der Ton abgeschaltet. Dies ist wichtig, wenn am Programmende wieder zum Anfang gesprungen wird.

Die Betriebssystemroutine HOLETASTE wartet auf einen Tastendruck und speichert den Tastencode dann im Akku ab. Durch den Befehl "cp" wird geprüft, ob entweder die Minus-Taste oder die Plus-Taste gedrückt wurde. Ist dies der Fall, so wird das D-Register (Lautstärke) erhöht (+) bzw. erniedrigt (-).

Anschließend wird der Tastencode durch die Routine TONUM umgewandelt. In das IX-Register wird die Anfangsadresse der Tabelle geladen, in der die Werte für die Noten stehen. Dazu wird der doppelte Wert der gedrückten Taste hinzuaddiert, damit die richtigen Notenwerte aus der Tabelle geholt werden können. Jetzt wird der Kanal 0 der Sound-Karte angewählt und das LSB der Tonhöhe an die Baugruppe ausgegeben. Identisch wird das MSB der Tonfrequenz dem Kanal 1 übermittelt.

Über den Kanal 7 wird das Byte 11111110b gesendet, das den Tongene-

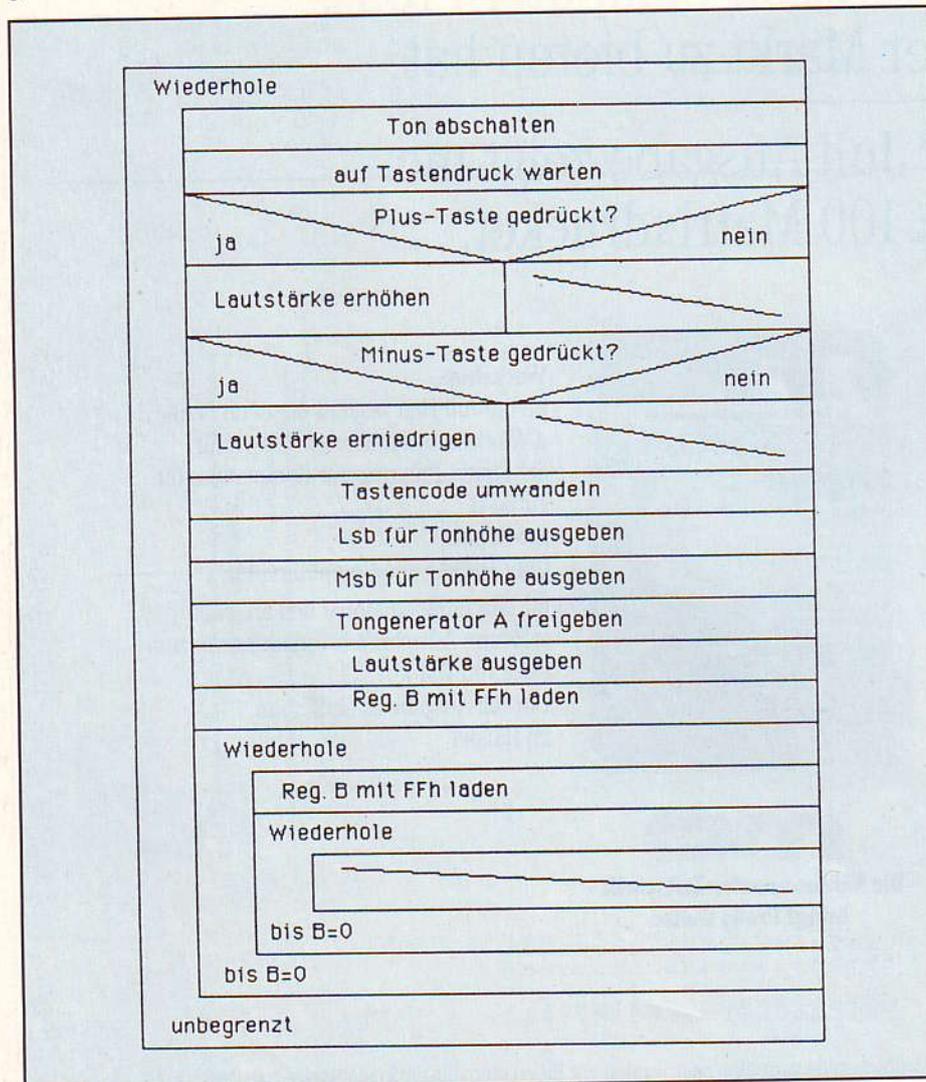


Bild 1: Struktogramm für Soundprogramm

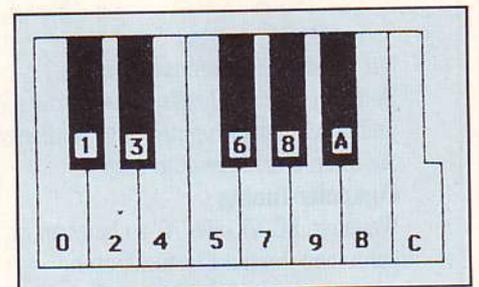


Bild 2: Tastenbelegung

rator A freigibt. Nun muß nur noch über das Register 8 die Lautstärke festgelegt werden, damit der Ton im angeschlossenen Lautsprecher gehört werden kann. Um nicht nur ein kurzes Knacksen im Lautsprecher zu hören, muß durch eine Verzögerungsschleife das Ausschalten des Tones hinaus geschoben werden. Um eine respektable Tonlänge zu erreichen, werden zwei Schleifen, die von FFh bis 0h hinabzählen, ineinandergeschachtelt.

Abschließend wird zum Programmstart gesprungen, wo der Ton ausgeschaltet wird und das Programm auf einen neuen Tastendruck wartet.

Christoph Köhler



Wunschkonzert



Spaßhalber kann man die Aktion statt einer genauen Erklärung auch durch ein Struktogramm erläutern. Versuchen Sie doch ausnahmsweise erst das "Struktogramm" (Bild 1) zu verstehen, bevor Sie das "Pflichtenheft" lesen.

Das Pflichtenheft

Das Wunschkonzert beginnt damit, daß Sie (wir) sich Programme oder Routinen wünschen, die Sie als eine Bereicherung für Ihren Rechner betrachten. Dabei kommen alle Ausbaustufen des NDR-Computers in Frage. Anschließend werden wir Ihre Anregung in der LOOP veröffentlichen. Falls sich jemand in der Lage fühlt, einen der Wünsche kurz- oder langfristig zu erfüllen, melde er sich bei der LOOP-Redaktion.

Um Parallelentwicklung zu vermeiden, können wir noch gegebenenfalls Kontakt mit Ihnen aufnehmen. Dann beginnt die "Programmiermühle" zu mahlen.

Selbstverständlich soll das nicht umsonst sein.

Es werden alle Beiträge, die auf TOOL-Disketten übernommen werden, mit Warengutscheinen belohnt und deren Autoren entsprechend auf der Diskette und gegebenenfalls in der LOOP namentlich veröffentlicht. Unvollständige oder fehlerhafte Programme können eventuell in "Teamwork" vervollständigt werden.

Es ist also ein interessanter Versuch !

Neuentwicklungen von Programmen können von Anwender für Anwender mitgestaltet und vor allen Dingen auch mitverfolgt werden.

Mit diesem Artikel beginnt eine Serie, um die Zusammenarbeit zwischen den NDR-Computer-Anwendern auszubauen. Wir nennen die mit diesem Artikel begonnene Aktion schlicht "Wunschkonzert".

Der Ablauf des neuen Wunschkonzertes ist folgendermaßen:

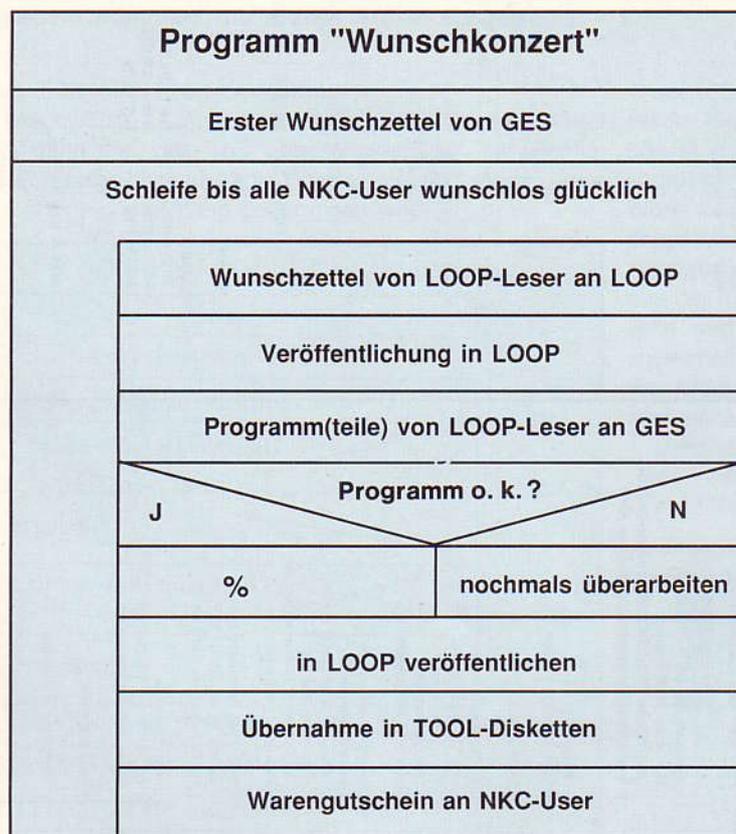


Bild 1. Das Struktogramm der Aktion Wunschkonzert

Als Auftakt stellen wir den ersten Wunschzettel.

An unserem Wunschzettel können Sie ersehen, in welcher Form Ihre Wunschzettel eingegeben werden könnten. Am besten teilen Sie es auf in:

- Hard- und Softwarevoraussetzungen
- Leistungsbeschreibung
- Skizzen
- Tips bzw. eigene Routinen

Nun aber zu unserem Wunschzettel:

Programm: "Ringpuffer"

CPU: Z80 oder 680xx

Betr. Sys.: CP/M xx oder JADOS

Sprache:

unerheblich

Haben Sie sich nicht schon mal geärgert, wenn Sie nach einer Kommando-Eingabe auf Ihrem Betriebssystem wegen einem falschen Buchstaben die ganze Zeile neu eingeben müssen. Logischerweise stellt sich in diesem Fall Ihr Betriebssystem auf "stur".

Abhilfe dazu wäre das Programm "Ringpuffer", das sich am besten gleich die letzten 20 Eingaben merkt. Diese jeweiligen 20 letzten Kommandos könnten mit den Pfeiltasten durchgeblättert werden. Jedes Kommando kann von Ihnen editiert werden (Cursor-Steuerung mit CTRL S,D,A,F,T,Y ..) und dann von jeder Cursor-Stellung aus übernommen werden. Die geblätterten Kommandos bleiben immer in der aktuellen Zeile nach dem Prompt.

(z.B. A>)

Nun aber "RUN" !

Hiermit starten wir das "Programm" Wunschkonzert und werden es betreiben, solange Sie Spaß daran haben. Schreiben Sie Ihre Vorstellungen an die LOOP-Redaktion mit dem Stichwort "Wunschkonzert".

Wir freuen uns auf eine gute Zusammenarbeit mit Ihnen.

Die LOOP - Redaktion

Leo Hepperger

Für Z80-Grundprogramm:

Symbole am Drucker ausgeben

Geben Sie "TESTPRINT := 0613H" ein, sonst kann dieses interne Grundprogramm nicht angesprochen werden. Start mit ST oder 0F333H. Noch ein Tip: Wird zu einem Programm (im Speicher) dazugeladen, werden dessen Symbole gelöscht (bzw. Zeiger

Durch dieses Programm erspart man sich das Abschreiben von Symbolen vom Bildschirm. Auch während dem Programmieren leistet das Programm "Symboltabelle" gute Dienste. Das Programm druckt die Symboltabelle so aus, wie sie am Bildschirm erscheint. Die Symbole werden absichtlich untereinander gedruckt, um Bemerkungen anhängen zu können.

neu gesetzt). Daher SYMBOLTABELLE immer zuerst laden, oder resident im Speicher halten. (Bei mir der Fall). Wird jetzt ein Programm

dazugeladen, werden die Symbole von SYMBOLTABELLE gelöscht, daher mit 0F333H starten.

Leo Hepperger
Am Schlag 7
8031 Eichenau

```

Start-Adresse:0F333H
End-Adresse:0F477H
F333 0D9F3 ST: CALL TEXT6
F334 3E00 LD A,00H
F335 3E00 LD A,00H
F336 0D700 SLOOP: CALL C875
F337 0B7F BIT 7,A
F338 2007 JR NZ,START
F339 00 NOP
F340 1B76 JR SLOOP
F341 00 NOP
F342 00 NOP
F343 0D300 CALL CLR
F346 21C981 START: LD HL,SS
F349 4E NEXT: LD C,(HL)
F34A 3E00 LD A,00H
F34C 89 CP C
F34D 0000 JP Z,0000H
F350 11000 LD DE,0000H
F353 7E ANZHL: LD A,(HL)
F354 13 INC DE
F355 23 INC HL
F356 0B7F BIT 7,A
F358 2B79 JR Z,ANZHL
F35A 0802 LD B,02H
F35C 23 INC HL
F35D 0B0F3 CALL SPACE
F360 7E OUT(0): LD A,(HL)
F361 0B0F3 CALL AUSBYTE2
F364 2B DEC HL
F365 10F9 DJNZ OUTBYTE
F367 0B0F3 CALL SPACE
F36A 0B0F3 CALL SPACE
F36B 23 INC HL
F36E 47 AND A
F36F ED52 SBC HL,DE
F371 7E OUTSYM1: LD A,(HL)
F372 0B7F BIT 7,A
F374 2B0E JR I,OUTSYM2
F376 0B0F3 CALL DRUCKER2
F378 0B0F3 CALL CLR2
F37E 23 INC HL
F37F 23 INC HL
F380 23 INC HL
F381 0B4F3 CALL NEXT
F384 0B0F3 OUTSYM2: CALL DRUCKER2
F387 23 INC HL
F388 1B07 JR OUTSYM1
F38A 3E20 SPACE: LD A,20H
F38C 0B0F3 CALL DRUCKER2
F38F 09 RET
F390 3E00 CLR2: LD A,00H
F392 0B0F3 CALL DRUCKER2
F393 3E0A LD A,0AH
F397 0B0F3 CALL DRUCKER2
F39A 09 RET
F39B 05 DRUCKER2: PUSH BC
F39C 05 PUSH DE
F39D 05 PUSH HL
F39E 0E49 F39E 0E49 LD C,49H
F39F 0E40 0DUMP2: IN B,(C)
F3A0 00 BIT 0,B
F3A1 00 LD B,00H
F3A2 20F4 JR NZ,0DUMP2
F3A3 0348 OUT (4BH),A
F3A4 06FF LD B,0FFH
F3A5 0E41 OUT (C),B
F3A6 0600 LD B,00H
F3A7 0E41 OUT (C),B
F3A8 06FF LD B,0FFH
F3A9 0E41 OUT (C),B
F3AA 0E41 OUT (C),B
F3AB 0E41 OUT (C),B
F3AC 0600 LD B,00H
F3AD 0E41 OUT (C),B
F3AE 06FF LD B,0FFH
F3AF 06FF LD B,0FFH
F3B0 06FF LD B,0FFH
F3B1 0E41 OUT (C),B
F3B2 0E41 OUT (C),B
F3B3 0E41 OUT (C),B
F3B4 0E41 OUT (C),B
F3B5 01 POP DE
F3B6 01 POP BC
F3B7 09 RET
F3B8 05 AUSBYTE2: PUSH AF
F3B9 05 PUSH AF
F3BA 05 RRA
F3BB 05 RRA
F3BC 05 RRA
F3BD 05 RRA
F3BE 05 RRA
F3BF 05 RRA
F3C0 05 RRA
F3C1 05 RRA
F3C2 05 RRA
F3C3 05 RRA
F3C4 05 RRA
F3C5 05 RRA
F3C6 05 RRA
F3C7 05 RRA
F3C8 05 RRA
F3C9 05 RRA
F3CA 05 RRA
F3CB 05 RRA
F3CC 05 RRA
F3CD 05 RRA
F3CE 05 RRA
F3CF 05 RRA
F3D0 05 RRA
F3D1 05 RRA
F3D2 05 RRA
F3D3 05 RRA
F3D4 05 RRA
F3D5 05 RRA
F3D6 05 RRA
F3D7 05 RRA
F3D8 05 RRA
F3D9 05 RRA
F3DA 05 RRA
F3DB 05 RRA
F3DC 05 RRA
F3DD 05 RRA
F3DE 05 RRA
F3DF 05 RRA
F3E0 05 RRA
F3E1 05 RRA
F3E2 05 RRA
F3E3 05 RRA
F3E4 05 RRA
F3E5 05 RRA
F3E6 05 RRA
F3E7 05 RRA
F3E8 05 RRA
F3E9 05 RRA
F3EA 05 RRA
F3EB 05 RRA
F3EC 05 RRA
F3ED 05 RRA
F3EE 05 RRA
F3EF 05 RRA
F3F0 05 RRA
F3F1 05 RRA
F3F2 05 RRA
F3F3 05 RRA
F3F4 05 RRA
F3F5 05 RRA
F3F6 05 RRA
F3F7 05 RRA
F3F8 05 RRA
F3F9 05 RRA
F3FA 05 RRA
F3FB 05 RRA
F3FC 05 RRA
F3FD 05 RRA
F3FE 05 RRA
F3FF 05 RRA
F400 05 RRA
F401 05 RRA
F402 05 RRA
F403 05 RRA
F404 05 RRA
F405 05 RRA
F406 05 RRA
F407 05 RRA
F408 05 RRA
F409 05 RRA
F40A 05 RRA
F40B 05 RRA
F40C 05 RRA
F40D 05 RRA
F40E 05 RRA
F40F 05 RRA
F410 05 RRA
F411 05 RRA
F412 05 RRA
F413 05 RRA
F414 05 RRA
F415 05 RRA
F416 05 RRA
F417 05 RRA
F418 05 RRA
F419 05 RRA
F41A 05 RRA
F41B 05 RRA
F41C 05 RRA
F41D 05 RRA
F41E 05 RRA
F41F 05 RRA
F420 05 RRA
F421 05 RRA
F422 05 RRA
F423 05 RRA
F424 05 RRA
F425 05 RRA
F426 05 RRA
F427 05 RRA
F428 05 RRA
F429 05 RRA
F42A 05 RRA
F42B 05 RRA
F42C 05 RRA
F42D 05 RRA
F42E 05 RRA
F42F 05 RRA
F430 05 RRA
F431 05 RRA
F432 05 RRA
F433 05 RRA
F434 05 RRA
F435 05 RRA
F436 05 RRA
F437 05 RRA
F438 05 RRA
F439 05 RRA
F43A 05 RRA
F43B 05 RRA
F43C 05 RRA
F43D 05 RRA
F43E 05 RRA
F43F 05 RRA
F440 05 RRA
F441 05 RRA
F442 05 RRA
F443 05 RRA
F444 05 RRA
F445 05 RRA
F446 05 RRA
F447 05 RRA
F448 05 RRA
F449 05 RRA
F44A 05 RRA
F44B 05 RRA
F44C 05 RRA
F44D 05 RRA
F44E 05 RRA
F44F 05 RRA
F450 05 RRA
F451 05 RRA
F452 05 RRA
F453 05 RRA
F454 05 RRA
F455 05 RRA
F456 05 RRA
F457 05 RRA
F458 05 RRA
F459 05 RRA
F45A 05 RRA
F45B 05 RRA
F45C 05 RRA
F45D 05 RRA
F45E 05 RRA
F45F 05 RRA
F460 05 RRA
F461 05 RRA
F462 05 RRA
F463 05 RRA
F464 05 RRA
F465 05 RRA
F466 05 RRA
F467 05 RRA
F468 05 RRA
F469 05 RRA
F46A 05 RRA
F46B 05 RRA
F46C 05 RRA
F46D 05 RRA
F46E 05 RRA
F46F 05 RRA
F470 05 RRA
F471 05 RRA
F472 05 RRA
F473 05 RRA
F474 05 RRA
F475 05 RRA
F476 05 RRA
F477 05 RRA
F478 05 RRA
F479 05 RRA
F47A 05 RRA
F47B 05 RRA
F47C 05 RRA
F47D 05 RRA
F47E 05 RRA
F47F 05 RRA
F480 05 RRA
F481 05 RRA
F482 05 RRA
F483 05 RRA
F484 05 RRA
F485 05 RRA
F486 05 RRA
F487 05 RRA
F488 05 RRA
F489 05 RRA
F48A 05 RRA
F48B 05 RRA
F48C 05 RRA
F48D 05 RRA
F48E 05 RRA
F48F 05 RRA
F490 05 RRA
F491 05 RRA
F492 05 RRA
F493 05 RRA
F494 05 RRA
F495 05 RRA
F496 05 RRA
F497 05 RRA
F498 05 RRA
F499 05 RRA
F49A 05 RRA
F49B 05 RRA
F49C 05 RRA
F49D 05 RRA
F49E 05 RRA
F49F 05 RRA
F500 05 RRA
F501 05 RRA
F502 05 RRA
F503 05 RRA
F504 05 RRA
F505 05 RRA
F506 05 RRA
F507 05 RRA
F508 05 RRA
F509 05 RRA
F50A 05 RRA
F50B 05 RRA
F50C 05 RRA
F50D 05 RRA
F50E 05 RRA
F50F 05 RRA
F510 05 RRA
F511 05 RRA
F512 05 RRA
F513 05 RRA
F514 05 RRA
F515 05 RRA
F516 05 RRA
F517 05 RRA
F518 05 RRA
F519 05 RRA
F51A 05 RRA
F51B 05 RRA
F51C 05 RRA
F51D 05 RRA
F51E 05 RRA
F51F 05 RRA
F520 05 RRA
F521 05 RRA
F522 05 RRA
F523 05 RRA
F524 05 RRA
F525 05 RRA
F526 05 RRA
F527 05 RRA
F528 05 RRA
F529 05 RRA
F52A 05 RRA
F52B 05 RRA
F52C 05 RRA
F52D 05 RRA
F52E 05 RRA
F52F 05 RRA
F530 05 RRA
F531 05 RRA
F532 05 RRA
F533 05 RRA
F534 05 RRA
F535 05 RRA
F536 05 RRA
F537 05 RRA
F538 05 RRA
F539 05 RRA
F53A 05 RRA
F53B 05 RRA
F53C 05 RRA
F53D 05 RRA
F53E 05 RRA
F53F 05 RRA
F540 05 RRA
F541 05 RRA
F542 05 RRA
F543 05 RRA
F544 05 RRA
F545 05 RRA
F546 05 RRA
F547 05 RRA
F548 05 RRA
F549 05 RRA
F54A 05 RRA
F54B 05 RRA
F54C 05 RRA
F54D 05 RRA
F54E 05 RRA
F54F 05 RRA
F550 05 RRA
F551 05 RRA
F552 05 RRA
F553 05 RRA
F554 05 RRA
F555 05 RRA
F556 05 RRA
F557 05 RRA
F558 05 RRA
F559 05 RRA
F55A 05 RRA
F55B 05 RRA
F55C 05 RRA
F55D 05 RRA
F55E 05 RRA
F55F 05 RRA
F560 05 RRA
F561 05 RRA
F562 05 RRA
F563 05 RRA
F564 05 RRA
F565 05 RRA
F566 05 RRA
F567 05 RRA
F568 05 RRA
F569 05 RRA
F56A 05 RRA
F56B 05 RRA
F56C 05 RRA
F56D 05 RRA
F56E 05 RRA
F56F 05 RRA
F570 05 RRA
F571 05 RRA
F572 05 RRA
F573 05 RRA
F574 05 RRA
F575 05 RRA
F576 05 RRA
F577 05 RRA
F578 05 RRA
F579 05 RRA
F57A 05 RRA
F57B 05 RRA
F57C 05 RRA
F57D 05 RRA
F57E 05 RRA
F57F 05 RRA
F580 05 RRA
F581 05 RRA
F582 05 RRA
F583 05 RRA
F584 05 RRA
F585 05 RRA
F586 05 RRA
F587 05 RRA
F588 05 RRA
F589 05 RRA
F58A 05 RRA
F58B 05 RRA
F58C 05 RRA
F58D 05 RRA
F58E 05 RRA
F58F 05 RRA
F590 05 RRA
F591 05 RRA
F592 05 RRA
F593 05 RRA
F594 05 RRA
F595 05 RRA
F596 05 RRA
F597 05 RRA
F598 05 RRA
F599 05 RRA
F59A 05 RRA
F59B 05 RRA
F59C 05 RRA
F59D 05 RRA
F59E 05 RRA
F59F 05 RRA
F600 05 RRA
F601 05 RRA
F602 05 RRA
F603 05 RRA
F604 05 RRA
F605 05 RRA
F606 05 RRA
F607 05 RRA
F608 05 RRA
F609 05 RRA
F60A 05 RRA
F60B 05 RRA
F60C 05 RRA
F60D 05 RRA
F60E 05 RRA
F60F 05 RRA
F610 05 RRA
F611 05 RRA
F612 05 RRA
F613 05 RRA
F614 05 RRA
F615 05 RRA
F616 05 RRA
F617 05 RRA
F618 05 RRA
F619 05 RRA
F61A 05 RRA
F61B 05 RRA
F61C 05 RRA
F61D 05 RRA
F61E 05 RRA
F61F 05 RRA
F620 05 RRA
F621 05 RRA
F622 05 RRA
F623 05 RRA
F624 05 RRA
F625 05 RRA
F626 05 RRA
F627 05 RRA
F628 05 RRA
F629 05 RRA
F62A 05 RRA
F62B 05 RRA
F62C 05 RRA
F62D 05 RRA
F62E 05 RRA
F62F 05 RRA
F630 05 RRA
F631 05 RRA
F632 05 RRA
F633 05 RRA
F634 05 RRA
F635 05 RRA
F636 05 RRA
F637 05 RRA
F638 05 RRA
F639 05 RRA
F63A 05 RRA
F63B 05 RRA
F63C 05 RRA
F63D 05 RRA
F63E 05 RRA
F63F 05 RRA
F640 05 RRA
F641 05 RRA
F642 05 RRA
F643 05 RRA
F644 05 RRA
F645 05 RRA
F646 05 RRA
F647 05 RRA
F648 05 RRA
F649 05 RRA
F64A 05 RRA
F64B 05 RRA
F64C 05 RRA
F64D 05 RRA
F64E 05 RRA
F64F 05 RRA
F650 05 RRA
F651 05 RRA
F652 05 RRA
F653 05 RRA
F654 05 RRA
F655 05 RRA
F656 05 RRA
F657 05 RRA
F658 05 RRA
F659 05 RRA
F65A 05 RRA
F65B 05 RRA
F65C 05 RRA
F65D 05 RRA
F65E 05 RRA
F65F 05 RRA
F660 05 RRA
F661 05 RRA
F662 05 RRA
F663 05 RRA
F664 05 RRA
F665 05 RRA
F666 05 RRA
F667 05 RRA
F668 05 RRA
F669 05 RRA
F66A 05 RRA
F66B 05 RRA
F66C 05 RRA
F66D 05 RRA
F66E 05 RRA
F66F 05 RRA
F670 05 RRA
F671 05 RRA
F672 05 RRA
F673 05 RRA
F674 05 RRA
F675 05 RRA
F676 05 RRA
F677 05 RRA
F678 05 RRA
F679 05 RRA
F67A 05 RRA
F67B 05 RRA
F67C 05 RRA
F67D 05 RRA
F67E 05 RRA
F67F 05 RRA
F680 05 RRA
F681 05 RRA
F682 05 RRA
F683 05 RRA
F684 05 RRA
F685 05 RRA
F686 05 RRA
F687 05 RRA
F688 05 RRA
F689 05 RRA
F68A 05 RRA
F68B 05 RRA
F68C 05 RRA
F68D 05 RRA
F68E 05 RRA
F68F 05 RRA
F690 05 RRA
F691 05 RRA
F692 05 RRA
F693 05 RRA
F694 05 RRA
F695 05 RRA
F696 05 RRA
F697 05 RRA
F698 05 RRA
F699 05 RRA
F69A 05 RRA
F69B 05 RRA
F69C 05 RRA
F69D 05 RRA
F69E 05 RRA
F69F 05 RRA
F700 05 RRA
F701 05 RRA
F702 05 RRA
F703 05 RRA
F704 05 RRA
F705 05 RRA
F706 05 RRA
F707 05 RRA
F708 05 RRA
F709 05 RRA
F70A 05 RRA
F70B 05 RRA
F70C 05 RRA
F70D 05 RRA
F70E 05 RRA
F70F 05 RRA
F710 05 RRA
F711 05 RRA
F712 05 RRA
F713 05 RRA
F714 05 RRA
F715 05 RRA
F716 05 RRA
F717 05 RRA
F718 05 RRA
F719 05 RRA
F71A 05 RRA
F71B 05 RRA
F71C 05 RRA
F71D 05 RRA
F71E 05 RRA
F71F 05 RRA
F720 05 RRA
F721 05 RRA
F722 05 RRA
F723 05 RRA
F724 05 RRA
F725 05 RRA
F726 05 RRA
F727 05 RRA
F728 05 RRA
F729 05 RRA
F72A 05 RRA
F72B 05 RRA
F72C 05 RRA
F72D 05 RRA
F72E 05 RRA
F72F 05 RRA
F730 05 RRA
F731 05 RRA
F732 05 RRA
F733 05 RRA
F734 05 RRA
F735 05 RRA
F736 05 RRA
F737 05 RRA
F738 05 RRA
F739 05 RRA
F73A 05 RRA
F73B 05 RRA
F73C 05 RRA
F73D 05 RRA
F73E 05 RRA
F73F 05 RRA
F740 05 RRA
F741 05 RRA
F742 05 RRA
F743 05 RRA
F744 05 RRA
F745 05 RRA
F746 05 RRA
F747 05 RRA
F748 05 RRA
F749 05 RRA
F74A 05 RRA
F74B 05 RRA
F74C 05 RRA
F74D 05 RRA
F74E 05 RRA
F74F 05 RRA
F750 05 RRA
F751 05 RRA
F752 05 RRA
F753 05 RRA
F754 05 RRA
F755 05 RRA
F756 05 RRA
F757 05 RRA
F758 05 RRA
F759 05 RRA
F75A 05 RRA
F75B 05 RRA
F75C 05 RRA
F75D 05 RRA
F75E 05 RRA
F75F 05 RRA
F760 05 RRA
F761 05 RRA
F762 05 RRA
F763 05 RRA
F764 05 RRA
F765 05 RRA
F766 05 RRA
F767 05 RRA
F768 05 RRA
F769 05 RRA
F76A 05 RRA
F76B 05 RRA
F76C 05 RRA
F76D 05 RRA
F76E 05 RRA
F76F 05 RRA
F770 05 RRA
F771 05 RRA
F772 05 RRA
F773 05 RRA
F774 05 RRA
F775 05 RRA
F776 05 RRA
F777 05 RRA
F778 05 RRA
F779 05 RRA
F77A 05 RRA
F77B 05 RRA
F77C 05 RRA
F77D 05 RRA
F77E 05 RRA
F77F 05 RRA
F780 05 RRA
F781 05 RRA
F782 05 RRA
F783 05 RRA
F784 05 RRA
F785 05 RRA
F786 05 RRA
F787 05 RRA
F788 05 RRA
F789 05 RRA
F78A 05 RRA
F78B 05 RRA
F78C 05 RRA
F78D 05 RRA
F78E 05 RRA
F78F 05 RRA
F790 05 RRA
F791 05 RRA
F792 05 RRA
F793 05 RRA
F794 05 RRA
F795 05 RRA
F796 05 RRA
F797 05 RRA
F798 05 RRA
F799 05 RRA
F79A 05 RRA
F79B 05 RRA
F79C 05 RRA
F79D 05 RRA
F79E 05 RRA
F79F 05 RRA
F800 05 RRA
F801 05 RRA
F802 05 RRA
F803 05 RRA
F804 05 RRA
F805 05 RRA
F806 05 RRA
F807 05 RRA
F808 05 RRA
F809 05 RRA
F80A 05 RRA
F80B 05 RRA
F80C 05 RRA
F80D 05 RRA
F80E 05 RRA
F80F 05 RRA
F810 05 RRA
F811 05 RRA
F812 05 RRA
F813 05 RRA
F814 05 RRA
F815 05 RRA
F816 05 RRA
F817 05 RRA
F818 05 RRA
F819 05 RRA
F81A 05 RRA
F81B 05 RRA
F81C 05 RRA
F81D 05 RRA
F81E 05 RRA
F81F 05 RRA
F820 05 RRA
F821 05 RRA
F822 05 RRA
F823 05 RRA
F824 05 RRA
F825 05 RRA
F826 05 RRA
F827 05 RRA
F828 05 RRA
F829 05 RRA
F82A 05 RRA
F82B 05 RRA
F82C 05 RRA
F82D 05 RRA
F82E 05 RRA
F82F 05 RRA
F830 05 RRA
F831 05 RRA
F832 05 RRA
F833 05 RRA
F834 05 RRA
F835 05 RRA
F836 05 RRA
F837 05 RRA
F838 05 RRA
F839 05 RRA
F83A 05 RRA
F83B 05 RRA
F83C 05 RRA
F83D 05 RRA
F83E 05 RRA
F83F 05 RRA
F840 05 RRA
F841 05 RRA
F842 05 RRA
F843 05 RRA
F844 05 RRA
F845 05 RRA
F846 05 RRA
F847 05 RRA
F848 05 RRA
F849 05 RRA
F84A 05 RRA
F84B 05 RRA
F84C 05 RRA
F84D 05 RRA
F84E 05 RRA
F84F 05 RRA
F850 05 RRA
F851 05 RRA
F852 05 RRA
F853 05 RRA
F854 05 RRA
F855 05 RRA
F856 05 RRA
F857 05 RRA
F858 05 RRA
F859 05 RRA
F85A 05 RRA
F85B 05 RRA
F85C 05 RRA
F85D 05 RRA
F85E 05 RRA
F85F 05 RRA
F860 05 RRA
F861 05 RRA
F862 05 RRA
F863 05 RRA
F864 05 RRA
F865 05 RRA
F866 05 RRA
F867 05 RRA
F868 05 RRA
F869 05 RRA
F86A 05 RRA
F86B 05 RRA
F86C 05 RRA
F86D 05 RRA
F86E 05 RRA
F86F 05 RRA
F870 05 RRA
F871 05 RRA
F872 05 RRA
F873 05 RRA
F874 05 RRA
F875 05 RRA
F876 05 RRA
F877 05 RRA
F878 05 RRA
F879 05 RRA
F87A 05 RRA
F87B 05 RRA
F87C 05 RRA
F87D 05 RRA
F87E 05 RRA
F87F 05 RRA
F880 05 RRA
F881 05 RRA
F882 05 RRA
F883 05 RRA
F884 05 RRA
F885 05 RRA
F886 05 RRA
F887 05 RRA
F888 05 RRA
F889 05 RRA
F88A 05 RRA
F88B 05 RRA
F88C 05 RRA
F88D 05 RRA
F88E 05 RRA
F88F 05 RRA
F890 05 RRA
F891 05 RRA
F892 05 RRA
F893 05 RRA
F894 05 RRA
F895 05 RRA
F896 05 RRA
F897 05 RRA
F898 05 RRA
F899 05 RRA
F89A 05 RRA
F89B 05 RRA
F89C 05 RRA
F89D 05 RRA
F89E 05 RRA
F89F 05 RRA
F900 05 RRA
F901 05 RRA
F902 05 RRA
F903 05 RRA
F904 05 RRA
F905 05 RRA
F906 05 RRA
F907 05 RRA
F908 05 RRA
F909 05 RRA
F90A 05 RRA
F90B 05 RRA
F90C 05 RRA
F90D 05 RRA
F90E 05 RRA
F90F 05 RRA
F910 05 RRA
F911 05 RRA
F912 05 RRA
F913 05 RRA
F914 05 RRA
F915 05 RRA
F916 05 RRA
F917 05 RRA
F918 05 RRA
F919 05 RRA
F91A 05 RRA
F91B 05 RRA
F91C 05 RRA
F91D 05 RRA
F91E 05 RRA
F91F 05 RRA
F920 05 RRA
F921 05 RRA
F922 05 RRA
F923 05 RRA
F924 05 RRA
F925 05 RRA
F926 05 RRA
F927 05 RRA
F928 05 RRA
F929 05 RRA
F92A 05 RRA
F92B 05 RRA
F92C 05 RRA
F92D 05 RRA
F92E 05 RRA
F92F 05 RRA
F930 05 RRA
F931 05 RRA
F932 05 RRA
F933 05 RRA
F934 05 RRA
F935 05 RRA
F936 05 RRA
F937 05 RRA
F938 05 RRA
F939 05 RRA
F93A 05 RRA
F93B 05 RRA
F93C 05 RRA
F93D 05 RRA
F93E 05 RRA
F93F 05 RRA
F940 05 RRA
F941 05 RRA
F942 05 RRA
F943 05 RRA
F944 05 RRA
F945 05 RRA
F946 05 RRA
F947 05 RRA
F948 05 RRA
F949 05 RRA
F94A 05 RRA
F94B 05 RRA
F94C 05 RRA
F94D 05 RRA
F94E 05 RRA
F94F 05 RRA
F950 05 RRA
F951 05 RRA
F952 05 RRA
F953 05 RRA
F954 05 RRA
F955 05 RRA
F956 05 RRA
F957 05 RRA
F958 05 RRA
F959 05 RRA
F95A 05 RRA
F95B 05 RRA
F95C 05 RRA
F95D 05 RRA
F95E 05 RRA
F95F 05 RRA
F960 05 RRA
F961 05 RRA
F962 05 RRA
F963 05 RRA
F964 05 RRA
F965 05 RRA
F966 05 RRA
F967 05 RRA
F968 05 RRA
F969 05 RRA
F96A 05 RRA
F96B 05 RRA
F96C 05 RRA
F96D 05 RRA
F96E 05 RRA
F96F 05 RRA
F970 05 RRA
F971 05 RRA
F972 05 RRA
F973 05 RRA
F974 05 RRA
F975 05 RRA
F976 05 RRA
F977 05 RRA
F978 05 RRA
F979 05 RRA
F97A 05 RRA
F97B 05 RRA
F97C 05 RRA
F97D 05 RRA
F97E 05 RRA
F97F 05 RRA
F980 05 RRA
F981 05 RRA
F982 05 RRA
F983 05 RRA
F984 05 RRA
F985 05 RRA
F986 05 RRA
F987 05 RRA
F988 05 RRA
F989 05 RRA
F98A 05 RRA
F98B 05 RRA
F98C 05 RRA
F98D 05 RRA
F98E 05 RRA
F98F 05 RRA
F990 05 RRA
F991 05 RRA
F992 05 RRA
F993 05 RRA
F994 05 RRA
F995 05 RRA
F996 05 RRA
F997 05 RRA
F998 05 RRA
F999 05 RRA
F99A 05 RRA
F99B 05 RRA
F99C 05 RRA
F99D 05 RRA
F99E 05 RRA
F99F 05 RRA

```

Bild 1: Listing des Programmes "Symboltabelle ausgeben"

Markus Haberstock

Logikanalyzer: LOG16 gegen LOGAN

Ein Logikanalyzer sollte mindestens 16 Kanäle besitzen, um auch den Adressbus eines 8-Bit Prozessors abbilden zu können. Logikanalyzer mit einem wesentlich höheren Preis (30000 DM und teurer) besitzen in der Regel mehr Kanäle. Von weiterer Bedeutung ist die Abtastrate des Logikanalyzers. Sie sollte deutlich über der des Eingangssignals liegen, damit die Impulse korrekt bewertet wer-

Wer sich schon einmal intensiver mit digitalen Schaltungen beschäftigt hat, wird sicher schon einmal den Wunsch gehabt haben, den Signalverlauf sichtbar vor sich zu haben. Dies ist mit einem Logikanalyzer möglich. Denn damit kann man exakt den zeitlichen Verlauf und das Zusammenspiel von digitalen Signalen darstellen. Während bei Oszilloskopen die Kurvenform möglichst genau wiedergegeben werden soll, ist hier nur der Zeitverlauf entscheidend.

den können.

Dies verdeutlicht Bild 1

Die Triggerung eines Logikanalyzers, d.h. wenn er damit beginnt, die Eingangsdaten in den Speicher zu schreiben, ist

der dritte wichtige Punkt. Was nützt es einem schließlich, wenn die Messung wegen des vollen Speichers beendet ist, noch bevor der interessante Bereich begonnen hat.

Eine preiswerte Alternative zu den professionellen Logikanalyzern bieten die Logikanalyzer-Baugruppen LOG16 und LOGAN, die hier verglichen werden sollen.

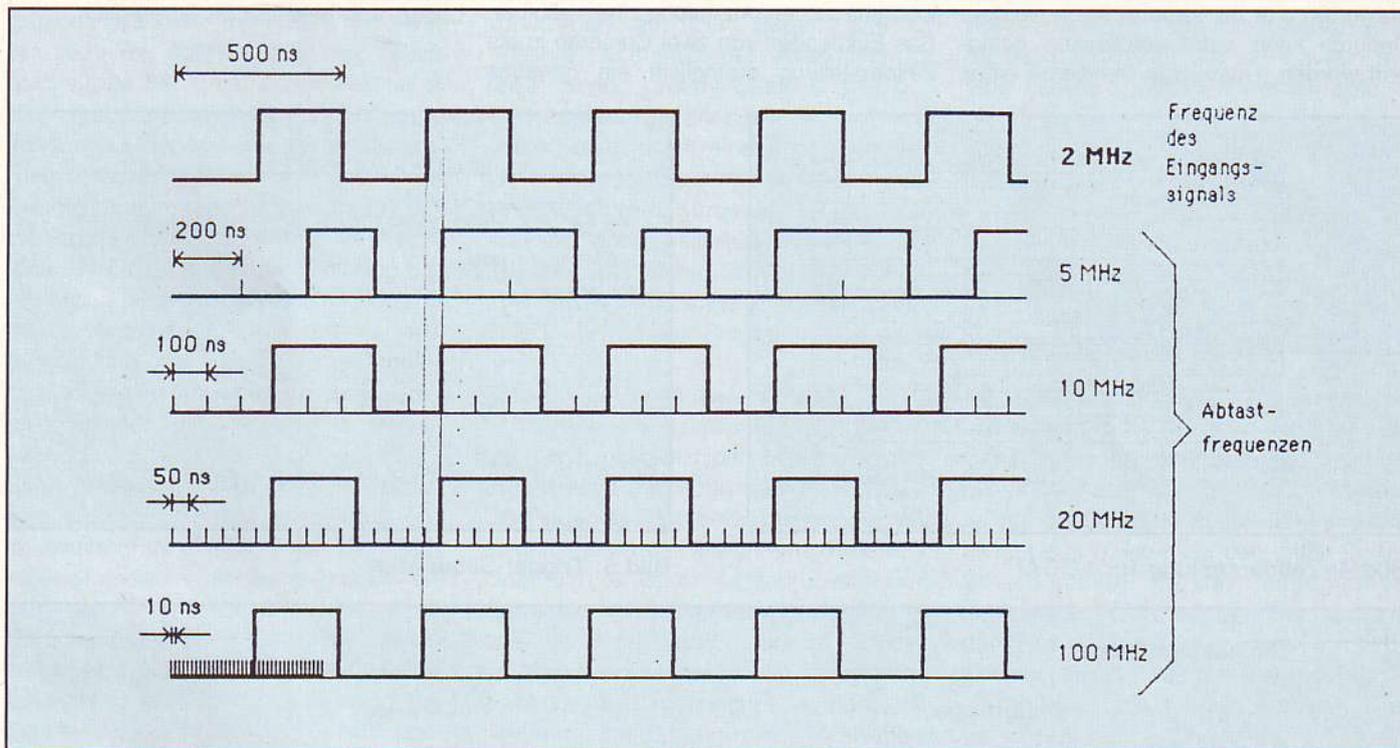


Bild 1: Abtastfrequenz

Die Baugruppe LOG16

Das Menu des Hauptprogrammes enthält eine große Anzahl von Funktionen. Mit Hilfe des Startkommandos werden die Signale in den Meßpuffer eingelesen. Dabei wird entweder intern oder extern getriggert. Bei externer Triggerung kann sowohl durch den Anschluß 'ext. Trigger' als auch durch ein Triggerbyte (Kanalgruppe B) die Messung begonnen werden. Durch Laden der Daten vom Meßpuffer in den Anzeigepuffer ist es möglich, die

Messung binär und zeitlich darzustellen. Im Anzeigemodus können durch 25 Befehle komfortable Funktionen ausgeführt werden. Jede Messung kann auf Diskette abgespeichert und jederzeit wieder geladen werden. Da die Baugruppe sowohl einen Meßpuffer als auch einen Anzeigepuffer besitzt, können weitere interessante Abläufe durchgeführt werden. So ist es möglich, eine alte Messung in den Anzeigepuffer zu laden und diese mit der aktuellen Messung im Meßpuffer zu vergleichen. Danach kann die aktuelle Messung

wieder auf dem Bildschirm dargestellt werden. Zwei weitere Funktionen der Software sind das Erstellen eines Z80-Disassembler und eines Hexdump-Listings. Andere Unterprogramme ermöglichen, das Bezugslaufwerk zu wechseln und das Disketteinhaltsverzeichnis oder eine genaue Beschreibung der einzelnen Menüfunktionen auf dem Bildschirm aufzulisten. Hochinteressant ist die neue Möglichkeit, zwei Baugruppen zu koppeln und damit 32 Kanäle darzustellen (LOOP 17).

Kommandos des Darstellungsmodus

1. Positionierungskommandos		2. Scroll Kommandos:	
Ctrl-D	: ↑ um eine Position nach rechts	< nn	: Scroll um nn Takte nach links
Ctrl-S	: ↑ um eine Position nach links	> nn	: Scroll um nn Takte nach rechts
Ctrl-D-D	: ↑ an den rechten Rand	.	: Scroll um einen Takt nach links
Ctrl-D-S	: ↑ an den linken Rand	.	: Scroll um einen Takt nach rechts
Ctrl-I	: ↑ um 10 Positionen nach rechts	N	: Scrollkommandos rückgängig machen
S + nn	: nn Seiten vor		
S - nn	: nn Seiten zurück		
+	: eine Seite vor		
-	: eine Seite zurück		
T nnnn	: ↑ auf die Takt Nummer nnnn		
3. Auswertungshilfen:		4. Verschiedenes:	
A	: ASCII an/aus	L	: Lineal an/aus
D	: Disassembler an/aus	K	: Kalibrierung der Ordinate
X	: Step Modus an/aus	Ctrl-@	: Hardcopy
R	: Receiver an/aus	B	: Einreue oder Grafische Darstellung
Z	: Zeitmasstab an/aus	I	: Invertieren (Kanal oder Messung)

ESC = Zurück zum Darstellungsmodus

Bild 2: Befehlsübersicht für LOG 16

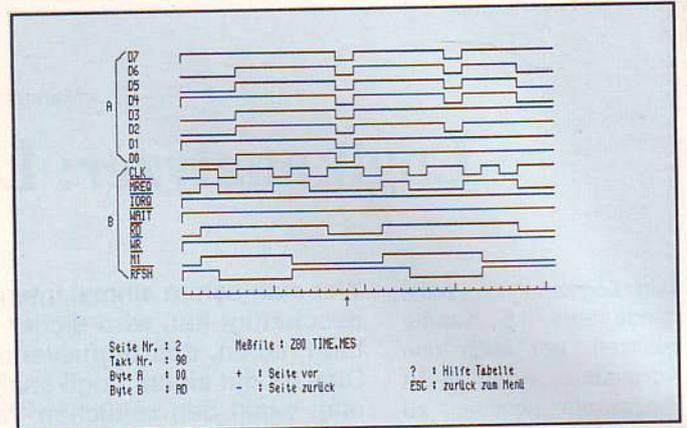


Bild 3: Zeitdarstellung für LOG 16

Die Baugruppe LOGAN

Mit dieser Baugruppe werden die Eingangssignale ebenfalls binär oder zeitlich dargestellt. Die externe Triggerung kann durch zwei Triggerebenen zu je zwei Triggerworten a' 8 Bit vorgenommen werden. Dadurch kann auf Signalfanken getriggert werden. Durch das Definieren eines

Pre-Trigger - Parameters ist es möglich, den Signalverlauf auch vor der Triggerung zu untersuchen. Die Frequenz, mit der die Eingangssignale abgetastet werden, ist in logarithmischer Abstufung frei wählbar. Das Einblenden von zwei Cursors in die Zeitdarstellung ermöglicht ein genaues

Abmessen der Dauer der Signale. Weiterhin kann man den Bildausschnitt um einen beliebigen Wert nach rechts oder links verschieben. Das Abspeichern und Laden von Messungen ist möglich.

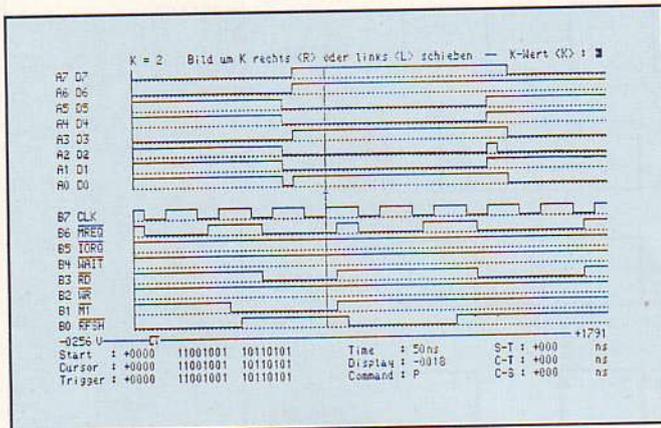


Bild 4: Zeitdarstellung für LOGAN

Trigger - Sequenzen

Die Triggerbedingung wird mit Hilfe zweier Triggerebenen zu je zwei Triggerworten a' 8 Bit eingestellt. Dadurch kann eine IF - THEN Bedingung realisiert werden. Die einzelnen Bits können dabei folgende Zustände annehmen :

1 = Triggerung bei HIGH
 0 = Triggerung bei LOW
 X = Don't care

Triggerbedingung :

```

A7...A0    B7...B0
IF 10000001 AND 10000001 THEN
IF 10000001 AND 10000001 THEN → Triggerung
    
```

Triggerbedingung richtig definiert ? (Y/N) : █

Bild 5: Trigger-Sequenzen

Technische Daten:	LOG16	LOGAN
Baugruppe	LOG16	LOGAN
Anzahl der Kanäle	16	16
Max. Abtastfrequenz	10 MHz	20MHz
Triggermöglichkeit	intern/extern	intern/extern
Stromaufnahme	0.4 A	1.3 A
Rechnertyp	NDR-Computer	NDR- und mc-Computer

Gegenüber dem LOG16 hat die Baugruppe LOGAN zwei wesentliche Vorteile. Durch die doppelte Abtastfrequenz von 20 MHz lassen sich die Signale genauer darstellen (man vergleiche dazu nur das CLK-Signal der beiden Zeitdarstellungen).

Der zweite Punkt ist die Definitionsmöglichkeit der Triggerbedingung. Durch das Einstellen eines jeden einzelnen Bits in zwei Ebenen sind hier keine Grenzen gesetzt. Dagegen sehen die Befehle der Software gegenüber der Fülle der Befehle

des LOG16 eher etwas dürftig aus. Denn genau darin liegt die Stärke der Baugruppe LOG16. Doch eines haben beide gemeinsam: Für relativ wenig Geld erhalten die Käufer einen brauchbaren Logikanalyzer.

Folkert Hallenga

680XX - Programme laufen überall

Die relative Adressierung

Damit die Überschrift zu diesem Artikel nicht völlig zusammenhanglos erscheint, möchte ich an dieser Stelle einmal den Begriff der relativen Adressierung einführen.

Dies sei an einem Beispiel erklärt: Nehmen wir einmal an, Sie wollten jemandem erklären, wo die Familie Hoppenstedt wohnt (Name aus Sicherheitsgründen geändert - der Autor). Sie könnten dazu die Adresse angeben: Schlaglochstraße 88. Da Sie sich absolut sicher sind, daß diese Adresse richtig ist, können wir hier von absoluter Adressierung sprechen.

Sie hätten aber auch sagen können, daß die Familie Hoppenstedt in der übernächsten Straße auf der linken Seite und dort im dritten Haus rechts wohnt. Da mit dieser Antwort (bei geeignetem Ausgangspunkt) die Familie relativ genau ausfindig gemacht werden kann, sprechen wir hier von einer relativen Adressierung.

Spaß beiseite: Im ersten Fall haben wir eine absolute Adresse angegeben und im zweiten Fall eine Adresse, die relativ zu einer anderen (der Ausgangsposition) war.

Da uns die Familie Hoppenstedt weniger interessiert, transferieren wir dieses Problem in ein zu schreibendes Programm. Um mir jetzt die Arbeit nicht unnötig schwer zu machen, gehe ich davon aus, daß Sie sich schon ein wenig mit dem 68000/8 Assembler vertraut gemacht haben. Ein Programm mit absoluten Adressen könnte z.B. folgendermaßen aussehen:

```
Start: * irgendwelche Kommandos
      * irgendwelche Kommandos
      JMP Adresse
Adresse: * irgendwelche Kommandos
      RTS
```

Wenn wir statt "irgendwelche Kommandos" tatsächlich ausführbare Befehle eingesetzt hätten, könnten wir das Programm übersetzen. Der Assembler er-

setzt die symbolischen Befehle und Sprungadressen in eine computergerechte Form und legt sie ab einer bestimmten Adresse (eventuell mit ORG vorwählbar) ab. Den Befehl "JMP - Adresse" übersetzt der Assembler in etwa: "Springe auf die Speicherzelle, die hier Adresse genannt wurde". Im Programm steht somit eine feste Adresse. Wenn dieses Programm nun im Speicher verschoben wird, würde der Sprung nach Adresse aber weiterhin zu der alten Speicherzelle führen, in der inzwischen aber etwas völlig anderes stehen kann. Folglich würde das Programm nicht laufen.

Ersetzen wir den Befehl "JMP" durch "BRA" haben wir unser Problem gelöst, denn nun übersetzt der Assembler den Befehl etwa mit: "Überspringe von hier bis zu der Speicherzelle, die hier Adresse genannt wurde, alle anderen Befehle". Oder um es klarer auszudrücken: Der Assembler berechnet die Sprungdistanz von der momentanen Adresse zu der angegebenen Adresse. Im Programm steht also nicht mehr eine feste Adresse, sondern eine Sprungweite. Wenn nun dieses Programm verschoben wird, bleibt die Sprungweite die gleiche und das Programm würde auch auf der neuen Adresse laufen.

Eng mit solchen Programmen verbunden tauchen auch Begriffe wie "frei verschieblich" und "relokativ" auf. Was es damit auf sich hat und welche Möglichkeiten wir dazu haben, soll in diesem Artikel erklärt werden.

Ein ähnliches Problem taucht bei den Unterprogrammaufrufen mit JSR auf. Beim Übersetzen von Programmtexten, die ein "JSR Unterprogramm" enthalten, schreibt der Assembler wieder feste Speicherzellen als Startadresse für den Unterprogrammaufruf in das Programm. Auch hier heißt die Lösung; Ersetze "JSR" durch "BSR".

Wenn die Sprünge nur sehr kurz sind (max: +/- 128 Byte), dann kann hinter das "BRA" oder "BSR" noch ein ".S" gesetzt werden. Der Assembler benutzt dann

eine Kurzform dieser Sprünge, was jeweils 2 Byte spart und somit in der Ausführung auch schneller ist. Der CP/M 68K Assembler generiert auch ohne das ".S" jeweils den optimalen Code.

Ein weiteres Problem taucht auf, wenn über Register auf bestimmte Adressen innerhalb des Programms zugegriffen werden soll. Solche Fälle treten auf, wenn das Programm zum Beispiel eine Meldung ausgeben soll oder wenn auf eine Tabelle zugegriffen werden soll.

```
Start: * irgendwelche Kommandos
      LEA Text(PC),A0
      BSR Print
      * irgendeine Text-
      * ausgaberoutine
      * irgendwelche Kommandos
Text: DC.B 'Meldung',0
```

Der Befehl "LEA" lädt - wie schon der Name sagt - die effektive Adresse von "Text" in das Adressregister A0. Dazu ermittelt der Assembler die Speicheradresse, auf der die Marke "Text" steht und lädt diesen Wert (also nicht den Inhalt dieser Speicherzelle) in das angegebene Adressregister. Nach der Übersetzung durch den Assembler steht im Programmcode also wieder eine feste Speicherzelle als zu ladende Adresse, was unserer Absicht widerspricht.

Raffinierte Adressierungsarten

Der 6800X Prozessor besitzt einige recht raffinierte Adressierungsarten. Im Rahmen dieses Artikels sollen hier aber nur zwei dieser besonderen Adressierungsarten angesprochen werden: Die Adressierung über Indexregister und die PC relative Adressierung. Wer allerdings viel mit Tabellen oder Feldern arbeitet, dem sei dringend ein gutes Fachbuch empfohlen, da durch geschickte Ausnutzung der enormen Adressierungsmöglichkeiten die Programme erheblich kürzer und somit übersichtlicher und auch schneller werden.

PC (program counter) ist die Bezeich-

nung für das Register, wo der Prozessor seine momentane Programmadresse ablegt. Bei der PC relativen Adressierung ermittelt der Assembler eine Sprungweite von dem momentanen Stand des Programmzählers (PC) bis zu der angegebenen Adresse. Im eigentlichen Programmcode steht also wieder nur eine Sprungweite (die aber nun auf den PC bezogen ist) statt einer festen Adresse. Wird das Programm später auf einer anderen Adresse als ursprünglich gestartet, steht auch der Programmzähler auf einer entsprechend anderen Adresse beim Erreichen des Registerladebefehls. Da die Sprungweite aber beim Verschieben des Programms nicht verändert wurde, wird auch die gewünschte Adresse wieder korrekt geladen. Für das obige Beispiel sähe das Programm dann wie folgt aus:

```
Start: *irgendwelche Kommandos
      LEA Text(PC),A0
      BSR Print
      * irgendeine Text-
      * ausgaberroutine
Text: DC.B 'Meldung',0
```

Verarbeiten von Variablen

Werden innerhalb eines Programms Variablen benötigt, dann gibt es erneut Probleme. Hier müssen wir aber unterscheiden, ob die Variablen noch innerhalb des frei verschieblichen Programmteils angelegt werden sollen oder außerhalb, wie es bei Programmen im Eprom sogar unumgänglich ist. Können die Variablen noch innerhalb des Programmcodes angelegt werden, ist prinzipiell das gleiche Verfahren wie im vorherigen Beispiel möglich.

```
Start: * irgendwelche Kommandos
      LEA Variable(PC),A0
      * arbeiten mit der Variablen
      * irgendwelche Kommandos
Variable: DC.L 0
```

Nachdem die Adresse der Variablen auf diese Weise ermittelt wurde, kann auf die Variable über die einfache indirekte Adressierung "(A0)" zugegriffen werden. Bei der einfachsten Möglichkeit, Variablen außerhalb des eigentlichen Programmkörpers einzurichten, wird lediglich mit ORG eine neue Adresse definiert, ab der die Variablen dann abgelegt werden. Da dieser Bereich unabhängig vom eigentlichen Programmkörper ist, kann auf diese Variablen nicht über relative, sondern nur über absolute Adressie-

rungsarten zugegriffen werden, wodurch das Programm selbst aber frei verschieblich bleibt.

```
Start: * irgendwelche Kommandos
      MOVE.L D0,Variable
      * arbeiten mit der Variablen
      MOVE.L Variable,D0
      * irgendwelche Kommandos
      RTS

ORG    A000
      *neu definierter Variablenbereich

OFFSET 0
Variable: DC.L 0
```

Diese Methode hat leider den Nachteil, daß die Variablen auf eine bestimmte Stelle im RAM festgelegt sind. Andere Programme dürfen nicht auf diesen Bereich zugreifen, wenn der Inhalt der Variablen nicht zerstört werden soll.

Index - Register

Eine Möglichkeit, bei der die Variablen nicht an einen festen (absolut adressierten) Bereich im RAM des Rechners gebunden sind, kann unter Zuhilfenahme eines Indexregisters realisiert werden. Ein solches Indexregister ist nun nicht etwa ein besonderes Register im Prozessor, sondern irgend ein beliebiges Adressregister, das für diesen Zweck mißbraucht wird. So benutzt z.B. das Grundprogramm bei einigen Aufrufen das Adressregister A5 als Indexregister, um bei beliebiger Lage des Grundprogramms auf den RAM-Bereich direkt hinter dem Grundprogramm zuzugreifen. Dazu wird irgendwann vorher einmal die Startadresse des RAMs ermittelt und in diesem Adressregister abgelegt. Aufrufe an eine bestimmte Adresse in diesem Bereich werden dann praktisch nur noch "als die soundsovielte Adresse ab der Adresse in A5" angegeben. Etwas ähnliches fanden wir auch schon bei der PC relativen Adressierung, nur wird normalerweise der Wert im Indexregister konstant gehalten und braucht sich nicht unbedingt in Abhängigkeit der Lage des Programms mitzuändern, wenn der angesprochene Variablenbereich sich nicht mitändern soll.

Um die Komplexität dieser Adressierungsform auch richtig hervorzuheben hat sie einen besonders komplizierten Namen: "Address register indirect with displacement". Auf Deutsch heißt das etwa, daß zu dem Inhalt des Adressregisters ein sogenannter Offset dazugaddiert

wird, ohne daß der Inhalt des Registers selbst dabei verändert wird. Mit dem so erhaltenen Wert wird dann weitergearbeitet. Der Offset ist dabei auf -32768 und +32767 begrenzt. Für unser Problem können wir diese Adressierungsart etwa so einsetzen:

```
Start: * definiere (berechne)
      * Startadresse für
      * die Variablen und
      * lege sie in A4 ab
      * irgendwelche Kommandos
      MOVE.L D0,Var1(A4)
      * arbeiten mit der Variablen
      MOVE.L Var1(A4),D0
      * irgendwelche Kommandos
      RTS

ORG 0

Var1 DS.L 1 * nur DS, kein DC !!!
Var2 DS.W 1
Var3 DS.W 1
```

Mit der Definition der Variablen hinter dem ORG 0 erreichen wir auf eine recht bequeme Art einen sogenannte Adressoffset. Prinzipiell wären die Variablen (oder genauer deren Adressoffsets) auch folgendermaßen definierbar:

```
Var1 EQU 0
Var2 EQU Var1+4
      * da Var2 Langwortgröße
Var3 EQU Var3+2
      * da Var2 Wortgröße
```

Werden die Variablen wie im oberen Programmbeispiel definiert, dürfen die Variablen nicht mit "DC", sondern müssen mit "DS" definiert werden, weil der Assembler beim Befehl "DC" feste Werte an die Stelle schreibt, wobei die Vektortabelle des Prozessors kaputtgeschrieben würde. Beim Befehl "DS" wird lediglich ein Bereich an dieser Stelle reserviert. Im Programm selbst wird dieser Bereich in einen anderen Bereich transferiert (durch das Indexregister angegeben), so daß die Vektortabelle dann unberührt bleibt. Sollen bei Programmstart definierte Werte in den Variablen stehen, müssen sie bei Programmbeginn erst einmal initialisiert werden, indem ihnen ein Wert zugewiesen wird.

Das oben aufgeführte Beispiel läßt sich jetzt noch weiter verfeinern: Mit einer kleinen Abänderung dieses Programms können wir erreichen, daß die Variablen auf dem Stack abgelegt werden. Der

Vorteil ist offensichtlich: Speicherkonflikte mit anderen Programmen können nicht auftreten und der zur Verfügung stehende Speicherraum wird optimal genutzt. Dazu reservieren wir uns gleich als erstes im Programm einen Variablenbereich auf dem Stack, der vor Verlassen des Programms wieder freigegeben wird. Die Adresse, auf der dieser so geschaffene Freiraum beginnt, merken wir uns in einem Adressregister. Wir haben dann praktisch die gleiche Situation wie im Beispiel zuvor.

```

Start: SUBA.L VarEnd-VarAnf,A7
      MOVEA A7,A5
      * irgendwelche Kommandos
      MOVE.L D0,Var1(A5)
      MOVE.W D1,Var2(A5)
      * arbeiten mit den Variablen
      MOVE.L Var1(A5),D0
      MOVE.W Var2(A5),D1
      * irgendwelche Kommandos
      ADDA.L VarEnd-VarAnf,A7
      RTS

ORG 0
VarAnf:
Var1:  DS,L 1    * nur DS, kein DC !
Var2:  DS,W 1

      DS 0      * damit Stack auf
              Wortbreite

VarEnd:

```

Werden im Programm auch bytegroße Variablen definiert, müssen Sie dafür sorgen, daß der Stack nach der Platzreservierung wieder auf einer geraden Adresse zu stehen kommt, da es sonst zu Adressfehlern kommen kann, wenn der Prozessor auf den Stack zugreifen möchte. Die oben vorgestellte Möglichkeit der Platzreservierung mit den symbolischen Namen erfüllt die Bedingung automatisch, da mit dem Kommando "DS 0" die Marke (Label) "VarEnd" auf eine gerade Adresse gezwungen wird. Zu guter Letzt wollen wir das Problem noch von einer anderen Seite betrachten. Wir wollen mit unserem Programm ein anderes Programm oder Teile daraus aufrufen, die sich in ihrer Adresslage verändern können.

Wenn wir in unserem Programm die Möglichkeit haben, die Startadresse zu ermitteln, können wir mit Hilfe des indirekten Sprungs diese Adresse anspringen. Beim NDR- Computer durchsucht z.B. das Bibliotheksmenü den Speicher nach einer festen Kennung und springt

auf Wunsch ein solcherart gefundenes Programm mittels der indirekten Adressierung an. Folgendes Beispiel zeigt das Prinzip:

```

Start: * berechne Startadresse für
      das(den)
      * Programm(teil) und lege sie
      in A0 ab
      * irgendwelche Kommandos
      JSR (A0)
      * irgendwelche Kommandos
      RTS

```

Der Trap

Nun wäre es sicher angenehmer, wenn es irgendwo eine feste Speicherzelle gäbe, in der jeweils steht, wo dieses Programm oder diese Routine zu finden ist. Allerdings müßte immer gewährleistet sein, daß dieses Programm seine jeweilige Lage der Speicherzelle mitteilt. Solche festen Speicherzellen gibt es sogar und außerdem auch gleich ein Kommando, das uns die Sache (Arbeit) damit erleichtert - der TRAP. Und da insgesamt 16 derartige Speicherzellen für solche Fälle vorhanden sind, gibt es 16 TRAPs (TRAP 0 .. TRAP15). Trifft der Prozessor auf einen Befehl "TRAP", springt er auf die dem jeweiligen TRAP zugeordnete Adresse in seiner Vektortabelle. Dort steht ein Sprung zu der Adresse auf der dann das eigentliche Programm steht. Um den Sinn solcher TRAPs zu verstehen, betrachten wir einmal unser Grundprogramm mit den insgesamt 120 Unterprogrammen, die für den User aufrufbar sind, genauer. Wird das Grundprogramm verschoben, verschieben sich sämtliche 120 Einsprungadressen für die Unterprogramme ebenfalls. Das Problem läßt sich lösen, indem ein Programm geschrieben wird, das in einem bestimmten Register einen Kennwert für das aufzurufende Unterprogramm erwartet und in Abhängigkeit von diesem Kennwert einen relativen Sprung zu dem entsprechenden Unterprogramm durchführt. Wenn dieses Programm durch einen TRAP aufgerufen werden kann, muß beim Verschieben des Gesamtprogramms nur noch in der Vektortabelle die Einsprungadresse zu dem Programm geändert werden. In unserem Grundprogramm gibt es auch so ein Programm, das abhängig vom Wert im Datenregister D7 eines der 120 Unterprogramme relativ anspringt. Beim Einschalten des Rechners ermittelt das Grundprogramm die richtige Einsprungadresse für dieses Programm und trägt in

der Vektortabelle für den TRAP 1 einen entsprechenden Sprung dorthin ein.

Als Beispiel soll hier einmal die Routine "SCHREITE" herhalten. Wollen wir diese Routine von einem Programm aus aufrufen, könnte dies z.B. so aussehen:

```

Start: * irgendwelche Kommandos
      MOVE #100,D0
      JSR $SCHREITE
      * irgendwelche Kommandos
      RTS

```

Wenn hier wieder sinnvolle Befehle für "irgendwelche Kommandos" eingesetzt werden, können wir das Programm mit dem im Grundprogramm eingebundenen Assembler übersetzen. Dieser ermittelt für "\$SCHREITE" die Adresse im Grundprogramm, wo das Unterprogramm SCHREITE beginnt und setzt diese Adresse in den Programmcode ein. Es steht somit wieder eine feste (absolute) Adresse im Programmcode. Wird das Grundprogramm nun aber verschoben oder stimmen bei einer anderen Grundprogrammversion die Einsprungadressen nicht mit der Version überein, mit der das Programm übersetzt wurde, stürzt dieses Programm ab.

Benutzen wir dagegen den TRAP 1 des Grundprogramms, so sieht das Programm dann so aus:

```

Start: * irgendwelche Kommandos
      MOVE #100,D0
      MOVE #!SCHREITE, D7
      TRAP #1
      * irgendwelche Kommandos
      RTS

```

Beim Übersetzen dieses Programms mit dem Assembler des Grundprogramms ermittelt dieser nur die Nummer des Unterprogramms und trägt sie in das Datenregister D7 ein. Danach erfolgt ein TRAP 1 Aufruf, der unabhängig von der Lage des Grundprogramms das Unterprogramm startet, dessen Nummer in D7 steht. Somit erfolgt praktisch wieder nur ein relativer Sprung zu dem Unterprogramm und wir haben unseren Willen bekommen.

Ich möchte doch unbedingt alle Assemblerprogrammierer bitten, diese Möglichkeit auch zu nutzen. Es ist traurig, immer wieder in der LOOP zu lesen, daß bei der und der Rechnerkonfiguration dieses oder jenes Programm nicht mehr läuft. Da es im Grundprogramm sogar eine feste Speicherzelle (bei \$414) gibt, auf der eingetragen ist, für welchen Prozes-

sor es geschrieben ist, ist es prinzipiell sogar möglich, auch bei direkten Hardwarezugriffen Programme zu schreiben, die sowohl mit dem 68008 als auch mit dem 68000 laufen. Beim Zugriff auf eine feste IO Adresse muß dann dieser Wert mit dem Wert in der Speicherzelle multipliziert werden. Als Beispiel sei die Abfrage des Tastaturports hier einmal aufgeführt:

```
Tast EQU $FF68

Start: MOVE #!GETBASIS,D7
      TRAP #1
      * in A0 Grundprogrammstart
      MOVE.W $416(A0),D0
      * dort steht CPU Größe/laden
      MULS #TAST,D0
      * mit Tastaturport multiplizieren
      MOVEA.L DO,A0
      * für Abfrage in Adressregister
      MOVE.B (A0),D0
      * in D0.B steht Tastaturwert
      * weitere Kommandos
      RTS
```

Dem Symbol "Tast" wurde hier nur eine Wortgröße zugeordnet, aber das reicht hier vollkommen aus, da wir bei der Multiplikation mit dem Befehl MULS eine automatische Vorzeichenerweiterung erhalten, die dafür sorgt, daß die obere Hälfte des Datenregisters mit \$FFFF vollgeschrieben wird.

Zu guter Letzt noch eine Bemerkung für diejenigen unter Euch, die relativ (um beim Thema zu bleiben) unabhängig vom Assembler (z.B. CP/M Assembler) programmieren wollen. Der CP/M Assemb-

ler kennt keine Ausdrücke wie z.B. !SCHREITE oder \$SCHREITE. Sollen die Programme aber mit Assembler übersetzbar sein, empfiehlt sich eine universellere Schreibweise, die sowohl auf dem RDK Assembler als auch auf dem CP/M Assembler lauffähig ist:

```
SCHREITE EQU 1

START: * irgendwelche Kommandos
      MOVEQ #100,D0
      MOVEQ #SCHREITE, D7
      TRAP #1
      * irgendwelche Kommandos
      RTS
```

Die Nummern der jeweiligen Programme stehen vorne im Buch zum Grundprogramm aufgeführt. Es ist jetzt auch eine Kurzform des Befehls (MOVEQ) möglich, die vorher - wohl durch einen Fehler im RDK Assembler - nicht möglich war.

Leider sind Programme mit Grundprogrammaufrufen unter CP/M 68K nicht so ohne Weiteres lauffähig, da dort der TRAP 1 nicht initialisiert wird. Es ist allerdings möglich, mittels einer kleinen Routine die Adresse des TRAP-Handlers im Grundprogramm zu ermitteln und diese dann in die für den TRAP 1 reservierte Speicherzelle zu laden. Beim Elektronikladen in Detmold gibt es auch eine Utilitydiskette mit einer Reihe von Hilfsprogrammen zum CP/M 68K, wo unter anderem auch ein Programm mit darauf ist, welches nach einmaligem Aufruf bis zum Abschalten des Rechners den TRAP 1 wieder initialisiert. Auf dieser Diskette befinden sich außerdem noch Programme zum Laden und Ab-

speichern von Daten, Programme zum Laden und Abspeichern von Texten auf beliebige Adressen, wobei beim Laden Tabulatorzeichen expandiert und Steuerzeichen gelöscht werden (können sonst zum Absturz des RDK Editors führen). Beim Abspeichern werden überflüssige Leerzeichen am Zeilenende gelöscht. Im Editoraufruf unter CP/M sind diese Verbesserungen ebenfalls eingebaut, zusätzlich zum Anlegen einer Sicherungsdatei (.BAK Datei) und der Möglichkeit einen Text zu verwerfen (nicht abzuspeichern). Des Weiteren gibt es auf dieser Diskette Programme zum Aufruf des Grundprogramms mit definierter Rücksprungmöglichkeit und ein Programm zum Starten von Programmen mit Bibliothekseintrag (z.B. PASCAL). Eine umfassende Dokumentation der Programme und der sich daraus ergebenden Möglichkeiten ist ebenfalls vorhanden.

Kleiner Test zur Lernkontrolle:

Wenn Sie jemandem erzählen, daß die Familie Hoppenstedt in der 3. Straße hinter der Hubertusgasse wohnt, welche Form der Adressierung war das dann? Indirekte Adressierung mit Offset - im Indexregister steht "Hubertusgasse" und der Adressoffset lautet "3 Straßen weiter".

Und wenn Sie nun zum Einwohnermeldeamt gelaufen wären und sich erkundigt hätten, weil die Familie nun mal so oft umzieht?...

Folkert Hallenga
Hufellandstr.16
3000 Hannover 91
Tel.:(0511) 2133239

Merktafel für 680xx - Programmierer

Grundsätzlich: Nur TRAP - Befehle verwenden

Feststellen, welche CPU im System steckt::

Dies kann dadurch geschehen, indem man die Adresse Grundprogrammstart + \$00000414 abfragt

Auswertung:

00000001	---> CPU 68008
00000002	---> CPU 68000
00000004	---> CPU 68020

Tyko Strassen

NDR - Computer <-----> Großrechner

Lochstreifenleser selbstgebaut

Als gemeinsames Übertragungsmedium zu einem anderen Rechnersystem gibt es prinzipiell mehrere Möglichkeiten. Die wohl am häufigsten verwendete Methode ist die Übertragung mit einem Magnetband. Allerdings kommt diese für Hobby-Benutzer nicht in Frage, weil eine Magnetbandstation für die meisten viel zu teuer ist.

Eine Übertragung via Telephon mit einem Modem ist auch ziemlich gebräuchlich. Dafür habe ich zwei sehr nützliche Programme entwickelt, für die man allerdings auch erst einmal tief in die Tasche greifen muß. Denn ein solches Telephonmodem darf man nicht selber bauen, sondern es muß von der Post gekauft werden.

Weiter werden Daten auch via Lochkarten übertragen. Diese Form nützt uns aber auch nicht viel.

Zuletzt bleibt noch das Übertragungsmedium "Papierlochstreifen", das zwar nicht gerade modern ist (die neuen Großrechner haben meistens keinen Lochstreifenstanzer mehr), aber den großen Vorteil hat, daß man sich mit geringem Aufwand an Material und Meßgeräten leicht einen eigenen Lochstreifenleser bauen kann. Dies möchte ich, natürlich mit der zugehörigen Software, ebenfalls beschreiben.

Papierlochstreifenleser

Technische Daten:

- Liest 8-Bit Papierlochstreifen
- Optische Abtastung. Das Papierband wird von Hand mit max. 2m/sec durch gezogen.
- Keine beweglichen Teile.
- RS-232 Ausgang.
Das Gerät läßt sich also auch problemlos an einen beliebigen anderen Rechner anschließen, der über eine serielle Schnittstelle verfügt.
- Übertragungsformat: 9600 Baud, 8 Bit, 1 Stop, keine Parität. (Das Bit 7 des Lochstreifens hat normalerweise die Bedeutung einer geraden Parität.)
- Stromversorgung: +5V, -12V oder -5V (AY-5-1013: -12V notwendig!)
- Zum Aufbau und Abgleich ist nur ein Voltmeter nötig.

Ein großes Problem für Rechner unter CP/M 68K ist, daß kaum Software verfügbar ist. Es gibt sicher viele Hobby-Programmierer, die versuchen, von anderen Rechnern die Software zu kopieren und dann, wenn nötig, dem NDR-Computer anzupassen. Zum Beispiel gibt es gerade auf UNIX-Systemen viele interessante C-Programme, die man im Prinzip direkt übernehmen könnte.

- Kostengünstig: Bauteile ca. 60.-DM

Dieser Papierlochstreifenleser funktioniert nach dem optischen Abtastungsprinzip. Auf der einen Seite des Streifens strahlen Leuchtdioden (aus Kostengründen vorzugsweise Infrarot) und auf der anderen Seite registrieren dann (Infrarot)-Photodioden, ob Löcher vorhanden sind. Diese Signale werden durch Schmitt-Trigger sauber aufgearbeitet und man erhält die acht parallelen Datenbits und den Strobe.

In dieser Form könnte man die Daten schon zum Tastatureingang (auch paral-

es ist wirklich

(Für die Photodiode muß man einen Typ nehmen, den man im 2,45mm-(1/10")-Raster nebeneinander setzen kann, weil die Löcher im Lochstreifen diesen Abstand haben. Der große Vorteil dieses Rasters ist außerdem, daß zur Lochstreifenführung und für die "Optik" eine Standard-Lochrasterplatine

5mm-Leuchtdioden, die jeweils vom Widerstand R' im Strom begrenzt werden. R' wird mit der Formel $U=R'I$ berechnet. Für I nimmt man den maximal zulässigen Leuchtdiodenstrom. Man kann R' ruhig etwas höher wählen, denn alles völlig unkritisch.

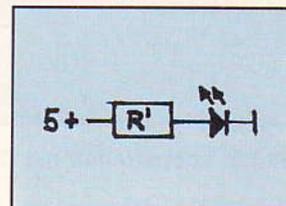


Bild 2: Leuchtdiode

es ist wirklich (Für die Photodiode muß man einen Typ nehmen, den man im 2,45mm-(1/10")-Raster nebeneinander setzen kann, weil die Löcher im Lochstreifen diesen Abstand haben. Der große Vorteil dieses Rasters ist außerdem, daß zur Lochstreifenführung und für die "Optik" eine Standard-Lochrasterplatine

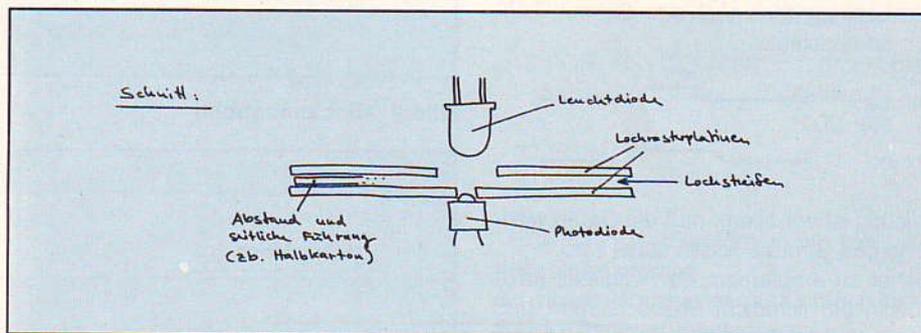


Bild 1: Anbringen der Dioden

lel) des NDR-Rechners führen. Sie werden jedoch durch einen parallel-seriell-Wandler (UART), der seinen Takt von einem 555-Oszillator erhält, in ein serielles Signal umgewandelt. Das serielle TTL-Signal wird dann anschließend durch einen Treiber auf den RS-232-Stecker geführt (25-polig).

Zur Schaltung:

Bei den Photo- und Leuchtdioden kann man entweder im sichtbaren oder im infraroten Spektrum arbeiten. Infrarot ist zu empfehlen, weil die Bauteile dann billiger werden. Probleme, weil man das Licht nicht sieht, wird es kaum geben, da das optische System in der Art völlig unkritisch ist.

Als Sender nimmt man 4 handelsübliche

verwendet werden kann.

Die Leselöcher, hinter denen die Photodioden liegen, werden auf ca. 1.8mm aufgebohrt, damit sie so groß sind wie die des Lochstreifens. Nur das Loch für den Strobe bleibt so klein wie es ist, damit dieser beim Lesen erst aktiviert wird, wenn die Daten schon stabil sind.

Die Widerstände R der Photodioden müssen so bemessen sein, daß der nachfolgende Schmitt-Trigger klar schalten kann.

Das heißt, bei einem Loch muß die Spannung unter 0.5V, bei den anderen über 1.9V liegen. Als Richtwert für R sei hier 1kOhm genannt. Der genaue Wert muß aber unter Berücksichtigung des

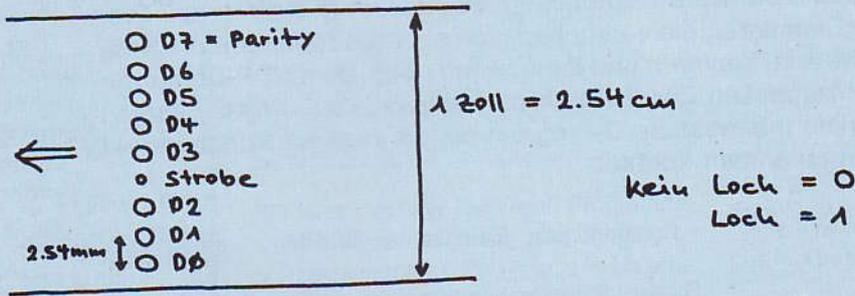
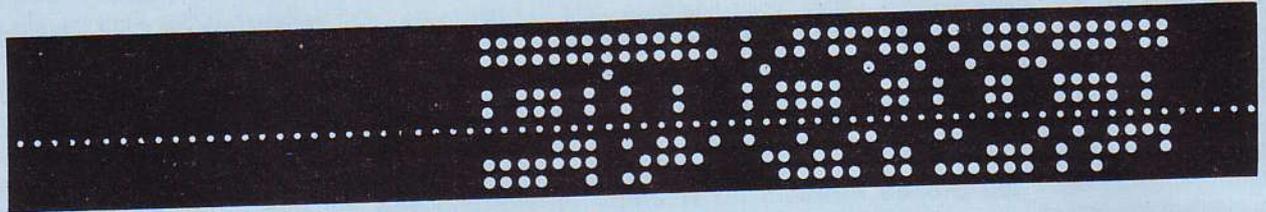
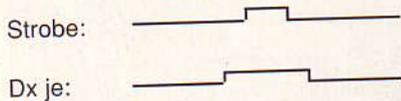


Bild 3: Lochstreifen mit Codierung

max. zulässigen Diodenstroms experimentell ermittelt werden. Er ist auch bei der Photodiode für den Strobe muß man etwas genauer sein. Deshalb wird hier ein Trimpoti mit ca. $R'' = 5 \cdot R$ verwendet. Der Trimmer wird so eingestellt, daß das Zeitdiagramm beim Durchziehen eines Lochstreifens nach dem Schmitt-Trigger so aussieht:



Wichtig ist vor allem, daß die Daten während des Strobes schon stabil sind. Es ist zu empfehlen, den Optikteil nach außen hin lichtdicht abzuschließen und innen alle Metallteile mit einem schwarzen Filzstift anzumalen, um Reflektionen zu vermeiden.

Es gibt zwei Typen UARTs (AY-5-1013, AY-3-1015), von denen einer (AY-5-1013) eine Versorgungsspannung von -12V benötigt. Sonst sind beide identisch. Hier wird nur ein Teil des UARTs benötigt, deshalb sind viele Pins nicht belegt. Das Datenformat kann anhand der Liste leicht abgeändert werden. (Siehe Bild 8)

Zur Takterzeugung wird ein einfacher 555 verwendet, der mit der 16-fachen Baudrate, also mit 153.6kHz schwingt. Man muß nur darauf achten, daß der Kondensator (3900pF) einigermaßen temperaturstabil ist. Vor dem Frequenzabgleich wird mit dem

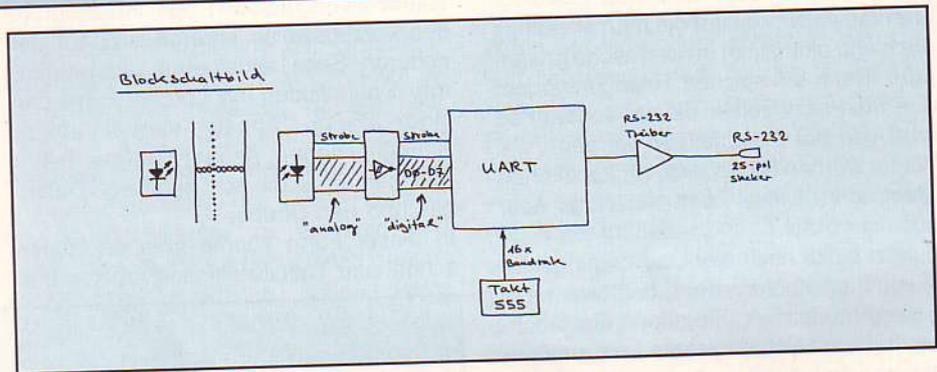


Bild 4: Blockschaltbild

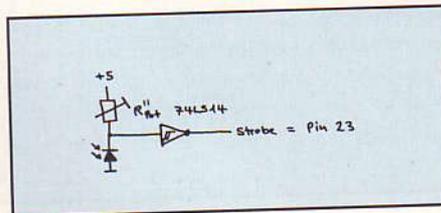
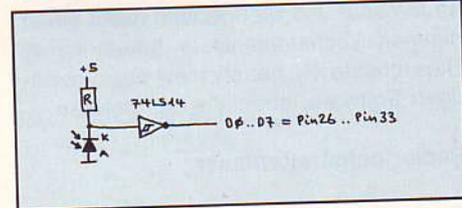


Bild 5 und 6: Photodioden

Trimmer ausprobiert, in welchem Bereich der Rechner noch alle Zeichen korrekt empfangen kann. Dann stellt man den Trimmer einfach in die Mitte dieses Bereiches. Beim Treiber (Typ 1489) kann man als negative Betriebsspannung wahlweise -12V oder -5V nehmen. Die Steckerbelegung ist angegeben. Das zugehörige Einleseprogramm ist aufgelistet.

Nun also zur Software zum Übertragen von Daten mit Telefonmodem und Papierlochstreifenleser via serielle Schnittstelle:



Hierfür ist sicher das C-Programm <siinit.c>, mit dem man die serielle Schnittstelle initialisieren kann, nützlich. Man hat die Möglichkeit, die Baudrate, die Anzahl der Daten und Stopbits, normalen oder Echo-Modus und das Paritätsbit einstellen.

Das Programm demonstriert auch, wie man direkt in C, ohne Assemblerteil, abhängig von der CPU I/O-Adressen setzen kann. |

m Prinzip lassen sich mit diesem Programm und dem Dienstprogramm <pip> alle angesprochenen Übertragungsprobleme lösen.

Hierzu Beispiele:

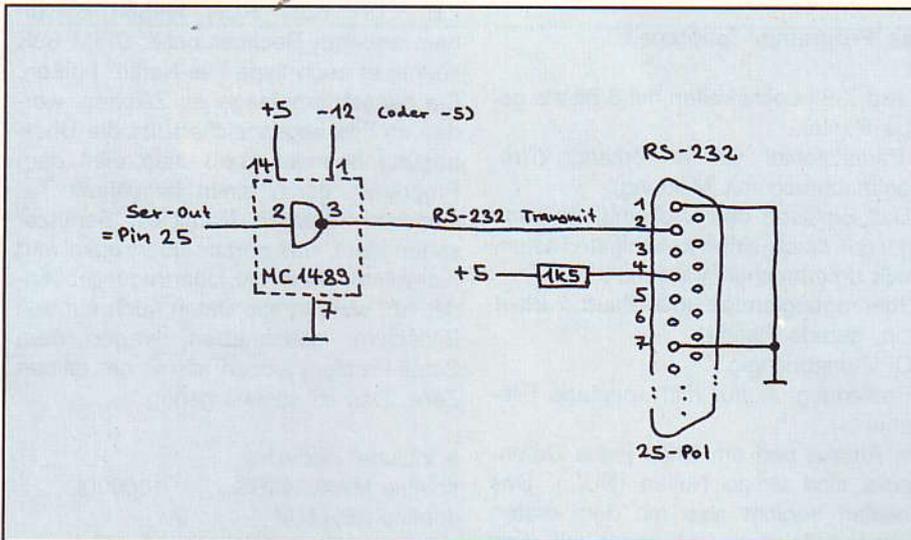


Bild 7: Steckerbelegung des AY-5-1013

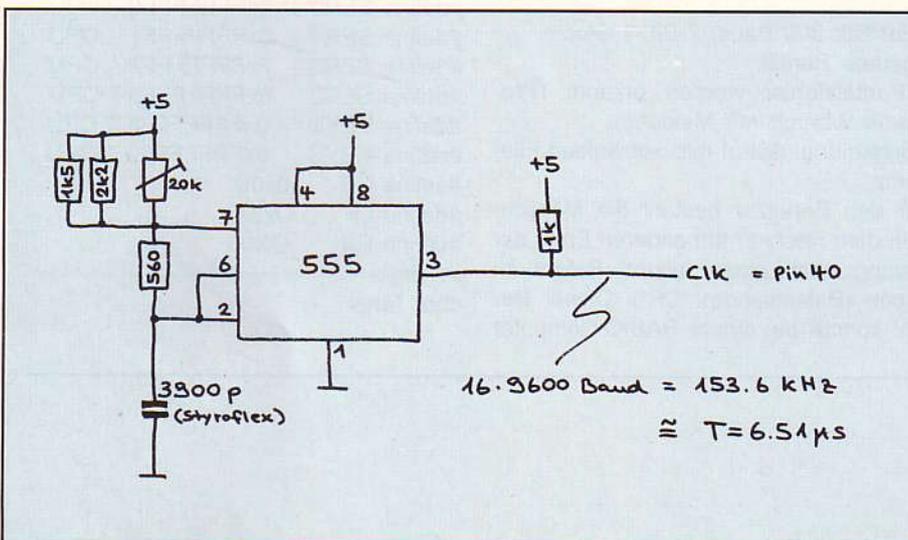


Bild 8: Takterzeugung

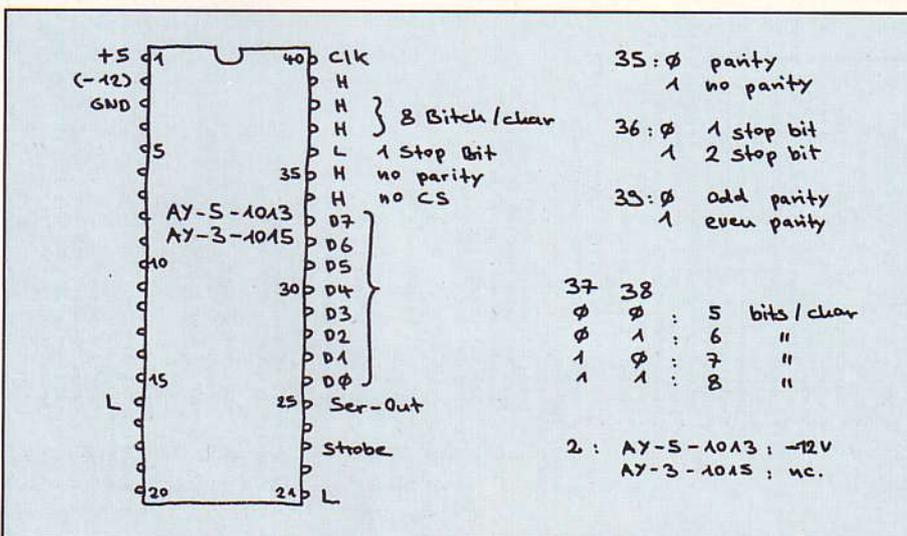


Bild 9: Serieller Treiber

1. Allgemeine Erklärung des <pip>-Befehls:

PIP <Ziel> := <Quelle>

Für Ziel:

- CON: Bildschirm
- AXO: Serieller Ausgang
- File-Name: File

Für Quelle:

- CON: Tastatur
- AXI: Serieller Eingang
- File-Name: File

2. Übertragung eines Files vom NDR-Computer auf einen UNIX-Rechner:

PIP AXO:= CON:

cat >UnixFile CCR ;Zeichen in Unix

PIP AXO:= CON

CTRL Z

PIP AXO:= NDRFILE

PIP AXO: CON

CTRL D

CTRL Z ; Ende der Übertragung usw.

3. Basicprogramm

Am Seriellen Eingang ist ein BASIC-Einplatinenrechner angeschlossen und auf dem NDR-Rechner ist folgendes kleines Programm unter <BASPROG> abgespeichert:

```
10 FOR N=1 TO 1000 ; druckt die Zahlen 1-1000 aus.
```

```
20 PRINT N
```

```
30 NEXT N
```

RUN ← Das Programm soll direkt ausgeführt werden.

So wird das Programm ganz einfach in den BASIC-Rechner eingelesen:

```
>PIP AXO: <- BASPROG
```

4. Wenn man den Output des Programms gleich danach auf dem Bildschirm sehen will:

```
>PIP CON: <- AXI
```

Hier treten zwei Probleme auf:

Einmal sind natürlich die ersten Zahlen schon vorbei und außerdem gehen teilweise Zeichen verloren, weil der NDR-Rechner beim Scrollen sehr langsam ist.

5. Die empfangenen Zeichen sollen nicht auf dem Bildschirm erscheinen, sondern in einem File abgespeichert werden:

```
>PIP OUTFILE <- AXI:
```



```

)
printf("Ansetze Adresse #21x auf %2x",SER2,SER2 & 0xFF);
printf("Ansetze Adresse #21x auf %2x\n\n",SER3,SER3 & 0xFF);
*((char *) SER2) = ser2;
*((char *) SER3) = ser3;

#include <stdio.h>
#define MAXCHARS 10000L
#define dummy 4
BOOLEAN ParityCheck;

int CPU;
/* 68008E1 68000E2 68020E4 */
#define SER0 0xFFFFFFFF * CPU
#define SER1 0xFFFFFFFF * CPU
#define SER2 0xFFFFFFFF * CPU
#define SER3 0xFFFFFFFF * CPU

#define KEY0 0xFFFFF68 * CPU
#define KEYS 0xFFFFF69 * CPU

main(argc,argv)
int argc;
char *argv[];
{
FILE *fopen();*fip;
static char text[MAXCHARS];c;
int n;
long anz_chr,pos;

if (argc != 2) {
printf(stderr,"Usage: paptape filename.ext\n");
exit(1);
}

printf("\032P A P E R T A P E - R E A D E R\n");
printf("Jan\88 T-Strassen");
printf("\nFormat : 9600 Baud 8 Bit 1 Stop No Parity");
printf("\nInterner Speicherplatz (BYTES) : %ld",MAXCHARS);
printf("\n-----");
printf("\n");

if ((fp = fopen(argv[1],"r")) == NULL) {
printf(stderr,"Cannot create %s\n",argv[1]);
exit(1);
}

init();
printf(" Lochstreifen einlesen \n\n");

/* Erst den Anfangs-Muell einlesen (Dummy) */
ParityCheck = FALSE;
for (n=0;n<dummy;n++) s(i);
ParityCheck = TRUE;

/* Zero's einlesen */
do c = s(i); while (c == 0);

/* Tape einlesen */
anz_chr = 0L;
while (c != 0) {
if (anz_chr == MAXCHARS) {
printf("\nmaximale Speicherfaehigkeit erreicht !!!\n");
break;
}
text[anz_chr] = c;
++anz_chr;
c = s(i);
}

printf("\n%ld Zeichen eingelesen (max %ld)\n",anz_chr,MAXCHARS);
for (pos = 0;pos < anz_chr;pos++)
putc(text[pos],fpi);
fclose(fpi);
printf("\nDaten im File <S> abgespeichert.\n",argv[1]);
}
}

VOID
init() /* Initialisiert Variable <CPU> und serielle Schnittstelle */
{
asm("move #23,d0");
asm("trap #3");
asm("move.w #616(a4),_CPU");
*((char *) SER3) = 0x3E; /* 200111110 : 7 Bit 1 Stop 9600 Baud */
*((char *) SER2) = 0x69; /* 201101001 : even Parity, no Echo */
}

BOOLEAN
s(i_empty) /* Wurde ein Zeichen empfangen ? */
{
register char c;
c = *((char *) SER1);
return( (c & 0x08) != 0x00 );
}

int
s(i) /* Liest ein Zeichen von serieller Schnittstelle ein */
{
while (!s(i_empty)) if (keypressed()) exit(0);
if (((char *) SER1) & 0x01) == 0) /* ParityCheck */
printf("\n*** Parity-Fehler\n");
exit(1);
}
return( *((char *) SER0) & 0x7F );
}

BOOLEAN
keypressed() /* Wurde eine Taste gedrueckt ? */
{
register char d;
d = *((char *) KEY0);
if ((d & 0x80) == 0x00) {
if ((char *) KEYS) {
return(TRUE);
}
return(FALSE);
}
}

```

Bild 11: Listing "Paptape - Reader"

```

anz_chr == ANHACK;
for (pos = 0; pos < anz_chr; pos++)
    putc(text[pos], fp);
fclose(fp);
printf("Daten im File <Z> abgespeichert.\n\n", argv[1]);
void
wt_c(c) /* Schmelze Zeichenausgabe: kein Scroll # */
char c;
{
    static int cur_pos;
    if (c > 0x7F) return;
    if (c == CR)
        putchar(CR); putchar(CR); putchar(ESC);
        cur_pos = 1;
    if (c == 0x20) return;
    if (cur_pos > 7) return; /* kein Platz mehr # */
    /* normalisiere # */
    putchar(CR); putchar(CR); putchar(ESC);
    cur_pos++;
}

void
init() /* Initialisiert Variable CPU und serielle Schnittstelle # */
{
    asm("move #25, d0");
    asm("trap #3");
    asm("move.w #416(a4), CPU");
    *((char *) SER3) = 0x36; /* 20011010 : 7 Bit 1 Stop 300 Baud # */
    *((char *) SER2) = 0x69; /* 201101001 : even Parity, no Echo # */
    ParityError = FALSE;
}

BOOLEAN
SI_empf() /* Wurde ein Zeichen empfangen ? # */
{
    register char c;
    c = *((char *) SER1);
    return( (c & 0x08) != 0x00 );
}

int
SI() /* Liest ein Zeichen von serieller Schnittstelle ein # */
{
    while (!SI_empf()) if (keypressed()) exit(0);
    if (((char *) SER1) & 0x01) ParityError = TRUE;
    return(*((char *) SER0));
}

VOID
sp(c) /* Gibt ein Zeichen auf serielle Schnittstelle aus # */
char c;
{
    register char d;
    do d = *((char *) SER1); while (d & 0x40) != 0x00;
    do d = *((char *) SER1); while (d & 0x10) == 0x00;
    *((char *) SER0) = c;
}

main() /* Wurde eine Taste gedrueckt ? # */
{
    keypressed();
    register char d;
    d = *((char *) KEYD);
    if (d & 0x80) == 0x00 {
        if (*((char *) KEYS) & 0x01)
            return(TRUE);
        return(FALSE);
    }
}

```

```

#include <stdio.h>
#define MAXCHRS 100000
#define ANHACK 2
BOOLEAN ParityError;
int CPU;
/* 68008:1 68008:2 68008:4 # */
#define SER0 0xFFFFF0 + CPU
#define SER1 0xFFFFF4 + CPU
#define SER2 0xFFFFF8 + CPU
#define SER3 0xFFFFFC + CPU
#define KEYD 0xFFFFE8 + CPU
#define KEYS 0xFFFFF0 + CPU
#define CR 0x0D
#define LF 0x0A
#define BS 0x08
main(argc, argv)
int argc;
char *argv[];
{
    FILE *fpopen();
    static char text[MAXCHRS];
    char c;
    long anz_chr;
    if (argc != 2) {
        printf(stderr, "Usage: download filename.ext\n");
        exit(1);
    }
    printf("\n020 0 W N L O A U HOST -> CP/M 68K");
    printf("Jawab: T. Strauss");
    printf("Format: 300 Baud / Bit 1 Stop Even Parity");
    printf("Minirener Speicherplatz (BYTES) : 21d, MAXCHRS");
    printf("-----\n");
    printf("-----\n");
    if ( (fp = fopen(argv[1], "w")) == NULL ) {
        printf(stderr, "Cannot create %s\n", argv[1]);
        exit(1);
    }
    init();
    printf("Letzter Befehl an HOST : #");
    while ( (c = SI()) & (c != LF) ) {
        sp(c);
        c = getchar();
    }
    sp(CR); sp(LF);
    printf("Ankommando ist eingegeben.\n");
    do c = SI(); while ( (c != CR) & (c != LF) );
    printf("Abertragung beginnt ... \n");
    anz_chr = 0;
    while ( !keypressed() ) {
        if (SI_empf()) continue;
        if (anz_chr == MAXCHRS) {
            printf("\n\n\nmaximale Zeichertiefe erreicht.\n");
            break;
        }
        text[anz_chr] = c;
        ++anz_chr;
    }
    printf("\n\n21d Zeichen eingelesen (max 21d).\n", anz_chr, MAXCHRS);
    if (ParityError) printf("\n\nParity - Fehler\n");
}

```

Bild 12: Listing "Download HOST -> CP/ M 68K"

Erfahrungsbericht aus der NDR/mc- Reparaturabteilung

von Ulrich Kracker

FRUST? Sie haben mühevoll eine NDR- bzw. eine mc- Baugruppe aufgebaut- Ins System gesteckt- Einschalten- Keine Reaktion- Frust! ABER NICHT DOCH! Jetzt heißt es kühlen Kopf bewahren. Nachdem die erste Enttäuschung überwunden ist, sollte man (möglichst) systematisch den Fehler suchen gehen. Hier möchte ich versuchen ein paar Tips zu geben, wie Sie in dieser Situation weiterkommen könnten, ohne die Baugruppe gleich an GES zurückschicken zu müssen...

SORGFÄLTIG GEARBEITET?

Natürlich haben Sie bisher die Schritt-Für-Schritt- Aufbauanleitung, die in jedem Hardwarehandbuch erläutert wird, beachtet. So können wir also z.B. davon ausgehen, daß die Spannungsversorgung aller Bausteine in Ordnung ist und keine Fehlbestückung vorliegt.

SICHER DOCH!

A'propos Fehlbestückung: Halten Sie sich beim Einsetzen der Bauteile genau an den Bestückungsplan bzw. den Bestückungsaufdruck. Achten Sie darauf, daß Sie keine Bauteile versehentlich in eventuell in der Nähe liegende Durchkontaktierungen eingesetzt haben. Wenn Sie lediglich eine Platine von GES bezogen haben und die Bausteine selbst beschafft haben, achten Sie darauf, daß die TTL- ICs auch vom Typ her mit den Angaben in der Stückliste übereinstimmen! Häufig sieht man nämlich LS- Typen anstatt TTL- Standardtypen eingesetzt. Timingfehler können dann damit erklärt werden, daß LS- Typen nicht die Schaltgeschwindigkeit besitzen, wie sie von Standardtypen beherrscht wird.

ERST 'MAL PUTZEN

Als nächsten Schritt in Richtung gezielter Fehlersuche waschen Sie bitte auf der Lötseite die vom Lötlen zurückgebliebenen Flussmittelrückstände ab. Bei frischen Lötresten genügt dazu Spiritus, bei alten Rückständen werden Sie zu 'härteren' Mitteln wie Waschbenzin oder Universalverdünner greifen müssen. **Aber Achtung:** Beachten Sie dabei die Warnhinweise auf der Fläche! Diese Maßnahme empfiehlt sich aus

Tips

Tip der Redaktion

Ein komfortabler C Compiler wird von der Firma Rose unter CP/M z80 angeboten und ist somit auch für den NDR Computer interessant.

Features des Compilers:

- Vollständige Version mit 13 stelliger BCD Arithmetik.
- Ausgabe in Z80,8080,8086 Assemblercode.
- Kompatibel zu M80/L80. (siehe MS-Paket).
- Fehlerverfolgung mittels Trace möglich.
- Deutsche oder englische Version lieferbar.
- 8", 5 1/4, 3 1/2, 3" Disk + dt. Handbuch

Vertriebsanschrift

.Herbert Rose
Bogenstraße 32
4390 Gladbeck

Korrekturen u. Ergänzungen

Probleme bei der Systemgenerierung

Häufige Probleme traten bei der Systemgenerierung des Betriebssystems CP/M68K auf. Dieser Umstand ist auf das unvollständige Submitfile der jeweiligen Speichergröße zurückzuführen. Irrtümlicherweise bestehen die Submitfiles auf der Systemdiskette 0 nur aus der Arbeitsanweisung die das Einfügen einer entsprechenden Basisadresse ausführt.

Dieser Vorgang ist jedoch nicht ausreichend um die Datei CPM.SYS neu zu übersetzen.

Das Bios des NDR Computers muß aus diesem Grund integriert und neu gebunden (gelinkt) werden.

Vollständig würde eine solche Übersetzung lauten:

```
L068 -r -UCPM -o CPMREL CPMLIB
NDRBIOS.0
RELOC -BXXXXX CPM.REL
CPM.SYS
```

zweierlei Gründen, auch wenn kein Fehler vorliegt:-Zum einen verhindern Lötmitlerückstände die 'ungetrübte' Sicht auf Leiterbahnen und Lötstellen.-Zum anderen können im Flussmittel enthaltene Säureanteile zu Langzeitkorrosionen führen. Zwar behaupten viele Hersteller von Elektronik-Löten nur säurefreies Kollophonium zu verwenden, aber sicher ist sicher.

Darüberhinaus bilden gealterte Lötmitlerückstände beim Wiedererhitzen (Schaltungsänderung, Behebung eines später auftretenden Fehlers, usw.) schlechtriachende Dämpfe.

Nachdem die Platine gereinigt und getrocknet ist, kann bei einer funktionierenden Baugruppe die Lötseite mit durchlötbarem Schutzlack beschichtet werden. Dazu eignet sich z.B. Plastik 70 von Kontakt-Chemie. Bei defekter Baugruppe warten wir mit dem Schutzlack, bis sie endgültig funktioniert, da ja noch die eine oder andere Nachlötung erforderlich sein kann (auch diese Lötungen werden selbstverständlich nocheinmal abgewaschen).

SCHAU, SCHAU!

Jetzt beginnt die Suche nach sichtbaren Lötbrücken oder Ätzbrücken. Besonders anfällig für Lötbrücken sind jene Stellen auf der Lötseite, wo eine Leiterbahn der-

art dicht am Lötauge vorbeifährt, daß sie teilweise in die löstopplackfreie Insel um das Lötauge herum hineinreicht.

Auch ungünstig abgewinkelte oder zu lange Drahtenden können Kurzschlüsse mit benachbarten Lötungen oder Durchkontaktierungen bilden.Mühsam zu suchen sind (zum Teil feinste) Ätzbrücken, da sie in der Regel mit 'tarnendem' Lötstopplack überzogen sind und man in diesem Stadium noch nicht konkret weiß, in welchem Schaltungsblock man besonders suchen müßte.

GEHT'S AUCH OHNE?

Wenn es sich um eine 'Erweiterungskarte' in dem Sinne handelt, daß Sie Ihr System auch ohne die betreffende Karte betreiben können, haben Sie noch eine weitere Chance, den Fehler genauer zu lokalisieren:

Nutzen Sie die in allen NDR- Rechnergrundprogrammen vorhandenen Funktionen zum Setzen und Lesen von I/O-Ports! Viele Baugruppen werden über I/O-Adressen vom Prozessor angesprochen und dahinter stehen in der Regel Datenregister, die Werte aufnehmen und ausgeben können.

Unter welcher Adresse die Register zu erreichen sind, wird in den Handbüchern angegeben.

SCHREIBEN UND LESEN...

Geben Sie also mit den Grundprogramm-funktionen einen Wert in eines der Register und lesen Sie diesen Wert zurück. Bei Baugruppen, wo dies systembedingt nicht funktioniert, lesen Sie ein Register zurück, von dem Sie wissen, was aufgrund der vorhergehenden Ausgabe darinstehen müsste. Wenn nun eine Unkorrektheit auftritt, gilt es diese möglichst intelligent zu interpretieren:

...UND INTERPRETIEREN

-Beim Wert 'FF' könnte vorliegen:
Ist Adressdekodierung arbeit nicht richtig?!
Ist die Adressierung der Baugruppe korrekt? Überprüfen Sie die Jumperstellungen auch mit Prüfstift, Voltmeter oder Ohmmeter/ Durchgangsprüfer (vorher Spannung abschalten).
Ist die Chip- Select- Leitung von der Adressdekodierungslogik zum betreffenden Register- Baustein in Ordnung? Überprüfung mit Ohmmeter, bzw. Durchgangsprüfer.
Ist die Adressdekodierlogik selbst in Ordnung (Auswechseln der Bausteine)
Ist der Bustreiber (meist ein 74LS245) in Ordnung?
Ist immer nur ein Bit gesetzt oder gelöscht? Schreiben Sie z.B. die Bitkombinationen 55h und AAh in die Register. Der Fehler liegt dann in den Datenwegen. Untersuchen Sie den betreffenden Daten-signalpfad auf Unterbrechungen, wenn

das Bit auf konstant 1 liegt und auf Kurzschlüsse wenn es auf 0 liegt.
 -Bei nicht reproduzierbaren Ergebnissen liegt der Fehler vermutlich 'hinter' den Registern, soll sagen, eine eventuell vorhandene Zusatzschaltung gibt unerlaubte Signalpegel an die Register (fehlender oder unwirksamer Pull- Up- Widerstand etc.) weiter.

NUR SCHREIBEN

Wenn ein Rücklesen von Registerinhalten nicht möglich ist, hilft oft nur ein Oszilloskop, in manchen Fällen auch ein Logik-Prüfstift weiter.
 Dazu wird sinnvollerweise eine kurze Endlosschleife für den jeweils eingesetzten Prozessor geschrieben, die nur einen Wert an die Baugruppe ausgibt. Damit können auch Speichererweiterungen untersucht werden. Mit dem Messgerät werden nun Adressdekodierung, Daten- und Steuerleitungen überprüft.
 Dabei ist oft nur wichtig zu sehen, daß 'Aktivitäten' mit TTL- Pegeln vorliegen. Um diese Tests durchzuführen ist es aber leider unumgänglich, den Schaltplan und die Schaltungsbeschreibung zu studieren.
 Sollte es zum Auslöten von Bauteilen kommen, sollte immer die Methode des 'zerstörten Bauteiles' wählen: Mit einem Seitenschneider wird das Bauelement derart bearbeitet, daß jeder Drahtanschluß unabhängig von den anderen herausgezogen werden kann. Die Platine wird günstigerweise senkrecht positioniert

(kleiner Schraubstock oder eine 'Dritte Hand') und die Lötstelle von der Lötseite her erwärmt.
 Der freiliegende Drahtanschluss wird auf der Bestückungsseite mit einer Pinzette gefaßt und beim Schmelzen der Lotes herausgezogen. Wenn alle Drahtanschlüsse derart entfernt worden sind, werden die Lötäugen mittels Löt-saugpumpe oder Entlötlitze vom übrigen Zinn befreit.

WENN GAR NICHTS MEHR GEHT

Wenn das System beim Einstecken der defekten Baugruppe nicht mehr korrekt arbeitet, obwohl es müßte, ist ebenso die Addressierung (Jumper) zu überprüfen, oder (und) der Bustreiber 74 LS 245 ist defekt.
 Handelt es sich um eine Baugruppe, die bereits zum Mindestumfang eines Systemes notwendig ist, sind die Möglichkeiten zur 'Selbstdiagnose' sehr beschränkt: Da der 'Wurm' nicht unbedingt auf der gerade bearbeiteten Baugruppe konzentriert sein muß, wird zuerst der Bus mit der Spannungsversorgung nochmals überprüft. Wenn ausser sichtbaren Fehlern nichts zu finden ist, könnte das systematische Auswechseln gesamter Baugruppen innerhalb von zwei Sytemen zur Lokalisierung des Fehlers beitragen.

Ulrich Kracker , GES

Kontakte **Kontakte** **Kontakte**

Suche: NKC - CP/M 2.2 User im Raum Hamburg oder Hannover/Hameln. Wer hat unter CP/M 2.2 schon eine Festplatte installiert und gibt Tips? Rolf Ahrens, Lütt Kollau 1, 2000 Hamburg 61, Tel.:040/588285

Verkaufe: NDR-Computer POW5V, GDP64K, KEY, CPUZ80, CPU68K, BUS3, TAST m.G, 5*ROA64, 2*IOE, SBC2, 2*CAS, 2*DRAM128, DRAM64/256, MON1, ZASS, GOSI, BASIC, MON68K, PASCAL68008, DEMO;Gesamtbausatzwert: ca. DM2800.- aufgebaut, für nur DM1200.- VB.Hans May, Wöllsteiner Str. 8, 6551 Siefersheim, Tel.:06703/3189

Verkaufe: NE2, POW5V, POW22/26V, BUS2, BUS3, SBC2, CPUZ80, KEY, Preh Tastatur, IOE, CENT, GDP64K, BANKBOOT, FLO2, RAM256, 2 Teac 5

1/4" Floppy Disk, PROMER, Flomon 1.5, CP/M 2.2, Nevada Fortran, Z80 Tools usw. nur komplett für DM2500.- VB.Uwe Sänger, Hagenring 11, 2819 Thedinghausen, Tel.:04204/7977

Zu verkaufen weit unter Neupreis, NDR-Klein-Computer mit: GEH3, BUS3, Netzteil, CPUZ80, KEY, CENT, BANKBOOT, CENT2, FLO2, IOE, RAM256, GDP64K, SER, ROA64, große Tastatur, 2 Floppy 5 1/4", alles in Betrieb, mit CP/M und Zeat, Preisvorstellung DM2000.-Anton Lanzl, Dieberg 69, 8411 Walderbach, Tel.:09464/346 (ab 18 Uhr)

68008: **Verkaufe** ROA64 komplett,geprüft, mit Original EASS0-3 V4.3 und Original EPASCAL V3.1 alles zusammen für DM130.-T. Strassen, Zürich, Tel.:(0041 1)555/246

In Berlin Schöneberg, Crellestraße 21, Verein Crelle 21 e.V., findet sich ab Anfang Mai regelmäßig ein **Computer-club** zusammen.Wir arbeiten mit dem NDR-Klein-Computer oder auch mit anderen Rechnern (wenn vorhanden) und werden uns mit Steuern, Messen und Regeln befassen.
 Kontakte bei Interesse: Peter Ulrich 65 73 23 Crelle 21 e.V. 782 32 64

Druckausgabe ohne Verlassen des Editors !
 Sparen Sie Papier und Zeit, indem Sie Dateien nur teilweise ausdrucken! RSX druckt markierten Block aus dem Editor von Turbo-Pascal unter CP/M-Plus. Programm mit ausf. Dokumentation geg. 10 DM (Scheck oder Schein) von K.P.Hasch, Postfach 250364, 4630 Bochum 25

Ab Juni 1988

PC-AT Sonderheft

Nachdem die Artikelserie der Zeitschrift "mc" zum mc-modular AT "voll eingeschlagen" hat, hat sich die Redaktion der "mc" entschieden, ein Sonderheft zum mc-modular-AT herauszubringen.

Dieses Sonderheft wurde bereits im Januar 1988 angekündigt und wird jetzt im Juni 1988 erscheinen.

Für die Anwender des mc-modular AT bildet dieses Sonderheft die ideale Ergänzung zum mc-modular AT Handbuch, das in erster Linie nur den Zusammenbau und die Inbetriebnahme der AT's beschreibt.

Wer tiefer in die Hard- und Software einsteigen will, sollte auf die detaillierten technischen Beschreibungen im PC-AT Sonderheft nicht verzichten.

Eine grobe Inhaltübersicht:

- Vorstellung eines modernen Konzepts für Rechner der AT-Klasse am Beispiel des mc-modular ATs

- Detaillierte Schilderung des in den meisten ATs verwendeten Chipsatzes von Chips & Technology

- Testberichte (Rein 300 LLC, VEGA VGA, MASM 5.0, Sanyo MBC-17-plus

- Aufbau und Funktion der weitverbreiteten Herculeskarte

- Filemanager für MS-DOS

- PC-/XT-/AT-Referenzliste

- Arbeitshilfen für Assembler-Programmierer

- Arbeiten mit dem 80287 (Coprozessor)

Tower-Gehäuse für den mc-modular-AT

Mit der Veröffentlichung dieses Artikels wollen wir (Graf Elektronik Systeme GmbH) unser erstes Unter-tisch-Gehäuse für den mc-modular-AT vorstellen.

Nicht nur den Vorteil der

Plazierung des Gehäuses unter Ihrem Arbeitstisch beinhaltet das Tower-Gehäuse, auch die Geräumigkeit ist wesentlich

Sie haben sich schon immer über Platzmangel auf Ihrem Arbeitstisch geärgert? Und jetzt soll auch noch ein Computer darauf stehen. Auf den Boden stellen wollen Sie den Rechner auch nicht, weil das kostbare Gehäuse zu leicht verkratzt wird. Dann stellt das neue Tower-Gehäuse eine willkommene Alternative zum Standardgehäuse für Sie dar.

vergrößert. In diesem Gehäuse finden Sie Laufwerkschächte für zwei 3 1/2"-Laufwerke, sowie zwei 5 1/4"-Laufwerken oder

Festplatten. Möchten Sie zwei 5 1/4"-Laufwerke und zusätzlich noch eine Festplatte, so ist diese Möglichkeit ebenfalls berücksichtigt, denn im Gehäuse finden Sie für diesen Fall noch eine Installationsvorrichtung (hinter der Frontanzeige).

Die Frontanzeige beinhaltet den Netzschalter, den Schlüsselschalter, Turbo- und Resetschalter und eine Taktfrequenzanzeige, die die jeweils gewählte Arbeitsfrequenz der CPU-Baugruppe mit zwei Siebensegmentanzeigen darstellt. Selbstverständlich ist diese Anzeige mit dem Turboschalter gekoppelt, sodaß bei einer Betätigung auch die entsprechende Taktfrequenz angezeigt wird.

Die Außenmaße:

Breite: 165 mm
Höhe: 605 mm
Tiefe: 490 mm.



Günther Waschilewski, GES

Neue Produkte - Neue Preise!

Das neue Grundprogramm für 680xx-Systeme

Best.-Nr.	Beschreibung	Preis DM
11218	68008 (ab Juli '88)	95,-
11219	68000 (ab August '88)	95,-
11220	Quelle auf Diskette, 5 1/4" 80 Spuren	60,-
11221	3 1/2" 80 Spuren	60,-

11204	Bausatz	148,-
11203	Fertiggerät	198,-

Restposten

10422	TEAC 3 1/2" Laufwerk für NDR-Computer 320,-
-------	---

LOG 16 - der Logik-Analysator, nun auch für 32 Kanäle

Best.-Nr.	Beschreibung	Preis DM
10906	Fertiggerät mit Software	598,-
10902	Bausatz mit SW	498,-
10898	Leiterplatte mit SW	298,-

Die mitgelieferte Software beinhaltet bereits die 32-Kanal-Lösung. Es werden dazu zwei Baugruppen benötigt.

GDPHS - Die neue GDP64K

Best.-Nr.	Beschreibung	Preis DM
11229	GDPHS Platine	30,-
11230	GDPHS Bausatz	269,-
11231	GDPHS Fertiggerät	348,-
11232	GDPHS Handbuch	20,-
11238	Aufbausatz GDP64K - GDPHS	70,-

LOGAN

Best.-Nr.	Beschreibung	Preis DM
10886	Platinen	159,60
10884	Bausatz	939,-
10885	Fertiggerät	1039,-
10887	Handbuch	20,-

mc-modular-AT

Best.-Nr.	Beschreibung	Preis DM
11247	Tower-Gehäuse mit 200W-Netzteil	799,-
11118	VEGA-VGA-Karte, kann VEGA und VGA 800x400	1198,-
11120	NEC MultiSync II	1798,-
11227	mc-modular-AT Sonderheft	30,-

BUSTEST

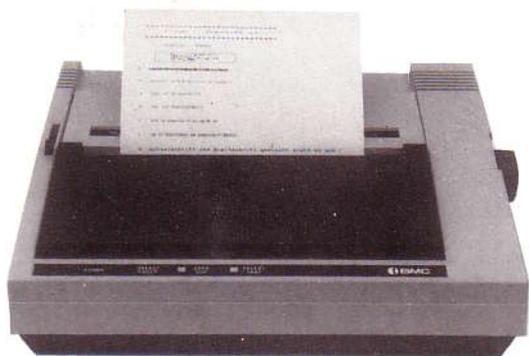
Best.-Nr.	Beschreibung	Preis DM
11206	Handbuch	20,-
11205	Platine	48,-

neu

61150	/286 CPU-Baugruppe ohne RAMs	848,-
61190	/386 CPU-Baugruppe ohne RAMs	3698,-

Speziell für LOOP-LESER

Matrix-Printer BX 100



Technische Daten:

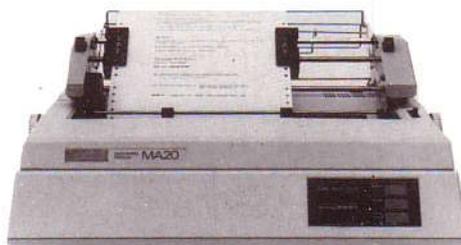
Punktmatrix-Drucker mit 100 Zeichen/sec. · Friktions- u. Traktorantrieb
 Bidirektionaldruck mit Druckwegoptimierung · Inkremental-Druck
 Proportional-Schrift · Unidirektionaldruck bei Graphik
 Papiertransport vorwärts/rückwärts · Kursiv-Schrift
 8 Nationale Zeichensätze · Programmierbarer Zeilenabstand
 Text-Matrix: 9 x 9 Dots · Bit Image Mode (1 : 1 Graphik!)
 Graphic-Matrix: 8 x 480 bzw. 576 Dots · 8 x 640 bzw. 720 Dots
 8 x 960 bzw. 1920 Dots · 9 x 480 / 960 Dots
 Programmierbare Seitenlänge · Skip over perforation
 Programmierbarer linker Rand / rechter Rand
 Vertikaler Tabulator · Horizontaler Tabulator
 Papierbreite: 4" bis 10" (114,3 mm - 254 mm)
 1 Original + 2 Durchschläge · Leistung ca. 80 Watt
 Lebensdauer Druckkopf ca. 100 000 000 Zeichen
 Lebensdauer Farbband ca. 2 500 000 Zeichen

LOOP-Leser-Aktion

inkl. 14 % MWSt.
Versand per UPS-NN - frei Haus DM 698,-

Typenraddrucker PETAL-MA 20

Korrespondenzqualität!
 Jetzt zum erschwinglichen Preis.
 Anschließbar an alle Mikrocomputer!



LOOP-Leser-Aktion:

inkl. 14 % MWSt.
Versand per UPS-NN - Frei Haus DM 698,-
 Optional: Traktor DM 349,- inkl. MWSt.

mirwald
electronic

Computer-Peripherie

Fasanenstraße 8b · 8025 Unterhaching/München
 Tel. (089) 6111224 und 6112040 · Telex 5213476

Büro Frankfurt:
 Adalbertstr. 15 · 6000 Frankfurt 90 · Tel. (069) 703538