Rolf-Dieter Klein

NDR- und mc-CP/M-Computer: IBM-kompatibel

Der Industriestandard-Computer hat sich durchgesetzt. Wobei die Bezeichnung „Industriestandard" eine bemüht neutrale Umschreibung der Tatsache ist, daß es sich um den von IBM entwickelten PC handelt. Der Softwaremarkt bietet heute eine solche Fülle von Programmen für den IBM-Computer, der Hardwaremarkt eine solche Fülle von Karten, daß es für alle anderen Computerhersteller Pflicht ist, sich einen Weg in die IBM-Welt zu bahnen. Also: auch der NDR-Computer wird kompatibel.

Hier wird eine Buskopplung vorgestellt, die beide Welten so verbindet, daß ein System daraus entsteht, in dem man Original-IBM-Karten verwenden kann. Man kann sogar im System eine NDR- und eine IBM-CPU parallel arbeiten lassen. Der IBM-Teil kann in den NDR-Teil hineinsehen – ohne die Arbeit des NDR-Computers zu stören. Die Baugruppe besitzt zwei Stecker für die mc-NDR-Welt, einen für den NDR-Computer und einen VG-Stecker für den Einsatz im mc-CP/M-Computer. In einer solchen Koppelung müssen die CPU-Baugruppen DMA-fähig sein, sonst funktionieren sie nicht.

Was heißt hier kompatibel?

Welche Bedingungen muß ein Computer erfüllen um IBM-kompatibel zu sein? Zunächst einmal braucht man eine der CPUs, die im IBM verwendet werden, also entweder einen 8088 oder 80286 oder gar einen 80386. Der IBM-Computer arbeitet mit MS-DOS. Dieses Betriebsystem kann man im Prinzip so anpassen, wie man es vom alten CP/M her kennt. Das MS-DOS besitzt eine sauber definierte Schnittstelle, die mit sogenannten Software-Interrupts arbeitet. Damit hätte man ein portables Konzept und man wäre unabhängig von Hardwaredetails, da alle Schnittstellen, wie Floppy, Bildschirm und Tastatur, durch Softwareschnittstellen standartisiert ansprechbar sind. Leider haben sich die Softwareproduzenten beim Programmieren oft nicht daran gehalten. So sind Programme, die zum Beispiel direkt auf den Bildwiederholspeicher des IBM zugreifen, eher der Normalfall. Aber auch der Floppy-Controller wird direkt angesprochen, um zum Beispiel einen Kopierschutz zu bewerkstelligen. Der Tastaturtreiber des IBM-Computers produziert einen Code, der nicht mit ASCII beziehungsweise mit ISO übereinstimmt. So ist man gezwungen, will man IBM-kompatibel sein, wirklich die Hardware des IBM-Computers weitestgehend nachzubauen, anfangen von der Tastatur über den Floppy-Controller, die Bildschirmschaltung bis hin zu innersten Details.

Der IBM-Computer ist darüberhinaus in seinem Aufbau nicht modular konstruiert. Daher ist es kaum möglich, die Modularität des NDR-Systems in eine kompatible Version mit hinüber zu nehmen. Nun wollten wir aber nicht einfach einen IBM-Computer (mit seiner großen Grundplatte) neben einen NDR-Computer stellen. Wir sind daher einen anderen Weg gegangen. Seit kurzem gibt es den IBM-Computer auf einer steckbaren Platine. Dies ist durch Integration möglich geworden. Man benötigt kein großes Motherboard mehr, wenn man einen IBM-Computer aufbauen will. Es genügt eine IBM-Busplatte mit aufgesteckter CPU- sowie Floppy- und Bildschirmkarte. Fertig ist der IBM.

Mit dem NDR-Computer hat das noch nichts zu tun, der IBM-Teil ist zunächst erst einmal halbwegs modular geworden. Wenn man dabei einen AT-Bus verwendet, so kann man später auch durch Wechsel der CPU zum IBM-AT aufrüsten.

Jede der bisherigen IBM-Computerklassen kann über eine Schaltung, die die Busse beider Systeme verkoppelt, mit dem mc-NDR-System verbunden werden. Der IBM-Computer kann dabei in den Speicher des NDR-Computers hineinschauen. Und er kann auf alle IO-Ports des NDR-Computers zugreifen. Auf dem NDR-Computer-Bus darf (muß aber nicht) auch eine CPU sitzen, die dann ihre eigenen Aufgaben erledigt.

Viele verschiedene Gesamtkonfigurationen sind möglich.

Bild 1 zeigt die einfachste aber umfangsreichste Variante. Beide Systeme sind komplett eigenständig. So gibt es zwei Tasturen und zwei Bildschirmkontroller. Dabei ist nur ein Bildschirm notwendig, denn auf der Koppelplatine ist ein Bildschirmumschalter vorgesehen. Weil ein IBM-Kompatibler mit 40-Spur-Laufwerken und der NDR normalerweise mit 80-Spur-Laufwerken arbeitet, sind zwei Floppy-Laufwerke nötig.

Bild 2 zeigt eine Variante, in der nur auf der IBM-Seite Peripherieeinheiten angeschlossen sind. Der NDR-Computer greift auf diese Einheiten mit Hilfe be-

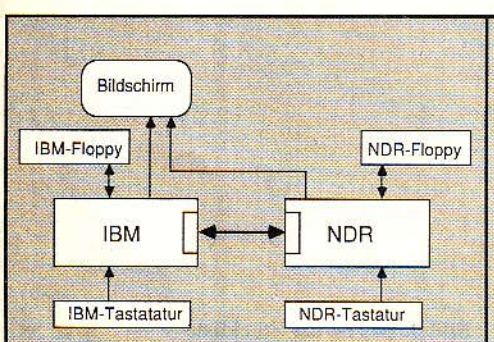


Bild 1. Die Kopplung mit zwei getrennten Einheiten

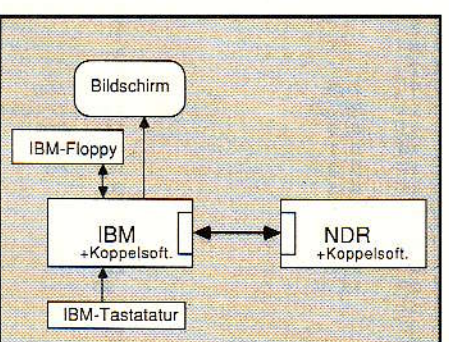


Bild 2. Der NDR-Computer verwendet die IBM-Peripherie

sonderer Koppelsoftware zu. So legt die NDR-CPU in einer bestimmten Speicherzelle einen Zugriffswunsch ab, und ein auf dem IBM laufendes Programm fragt diese Zelle ständig ab. So kann man zum Beispiel Bildschirmausgaben auf den IBM umleiten aber auch Tastatureingaben vom IBM auswerten und sogar das Floppy-Laufwerk (dann aber nur mit 40 Spuren) mitverwenden.

Auf diese Art läßt sich das FLOMON-Programm beim Z80 leicht anpassen. Wenn man beim NDR-Computer nicht auf die schnelle Grafik verzichten will, so kann man die Grafikkarte des NDR-Computers verwenden und nur die Tastatur und die Floppy des IBM-Gerätes nutzen.

Die verwendete NDR-CPU muß wie gesagt DMA-fähig sein, um mit dieser Kopplungsart arbeiten zu können. Derzeit gilt das nur für die Z80-CPU (Vollausbau, oder SBC3), die 64180-CPU und die 68020-CPU (32-Bit Rechner). Nicht jedoch für 68008 und 68000, die man jedoch leicht umrüsten kann., wie noch in einem späteren Artikel beschrieben wird.

Aufbau und Wirkungsweise der Schaltung

Bild 3 zeigt eine Blockschaltbild der gesamten Anordnung. Links ist die IBM-Koppellogik zu sehen und rechts der Schaltungsteil des NDR-Computers. Beide Teile sind über ein 50-poliges Flachbandkabel miteinander verbunden.
Im IBM-Adreßraum belegt der NDR-Computer nur 64 KByte für Speicher und 32 KByte für IO. Um bis zu 1 MByte im NDR-Computer erreichen zu können, wird ein Latch genutzt, das „Bänke" umschaltet. Dorthinein wird die zusätzliche Adreßinformation gespeichert.

Bild 4 zeigt den Schaltungsteil für die IBM-Seite. Die Adreßleitungen A0 bis A15 werden dabei mit Hilfe zweier Bustreiber (74LS244) ständig durchgeschaltet. Die Adreßleitungen A16 bis A19 kommen von dem Latch IC 74LS273. Dieses Latch belegt acht IO-Adressen im IBM-Adreßraum und zwar von 300h bis 307h, da diese nicht eindeutig dekodiert sind. Dieser Adreßraum ist im IBM für Prototypenadapter reserviert. Wenn man auf eine dieser Adressen schreibt, so wird der Wert am Datenbus in das Latch gespeichert. Die IO-Adresse auf der IBM-Seite kann man in der Schaltung durch Ändern der Brücken am IC 74LS688 auch anders

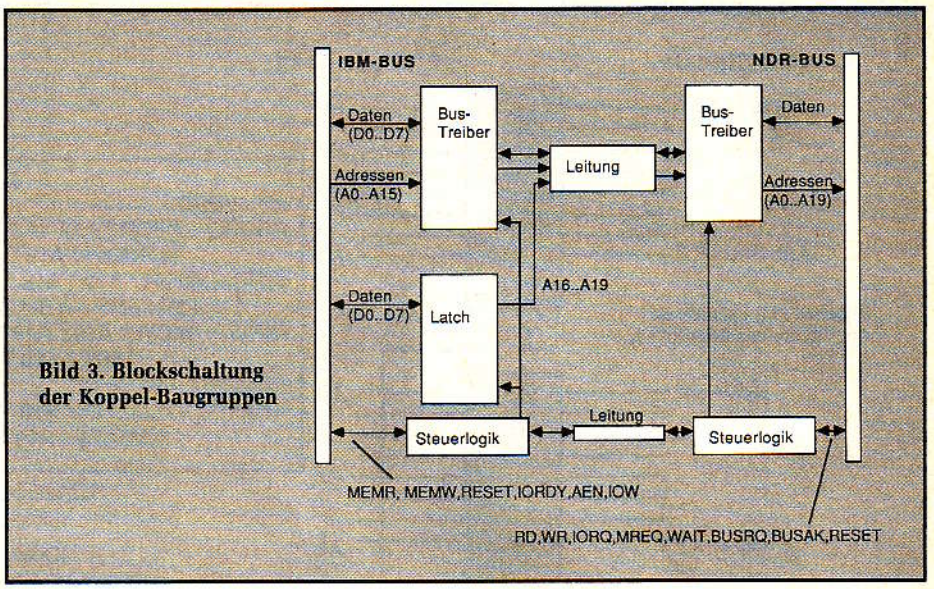


Bild 3. Blockschaltung der Koppel-Baugruppen

einstellen, im Layout sind entsprechende Brücken vorgesehen.

Das IC 74LS245 ist ein bidirektionaler Bustreiber, der D0 bis D7 auf die Leitung schaltet. Dabei schaltet der Bustreiber die Datenrichtung nur dann zum IBM-Bus, wenn der IBM lesend auf einen der gültigen Bereiche zugreift. Die Bereiche werden durch einen PAL-Baustein bestimmt, deren Gleichungen ebenfalls im Bild sichtbar sind.
Zur Verknüpfung werden die Adreßleitungen A14 bis A19, sowie die Steuerleitungen MEMR, MEMW, AEN und RESET DRV vom IBM an das PAL geführt. Es erzeugt damit auch die Signale IORQ, MREQ, RD, WR und RESET, sowie das DIR-Signal. Ein Zugriff auf den Bereich DC000 bis DFFFF wird als IO-Zugriff in den NDR-Computer ausgewertet. Daher erscheint dann das Signal IORQ. Bei einem Lesezugriff werden ferner das Signal RD und das Signal DR auf 0 gelegt, bei einem Schreibzugriff nur das Signal WR. Bei einem Zugriff auf den Bereich E0000 bis EFFFF wird das Signal MREQ anstelle von IORQ aktiviert. Damit gibt es einen Speicherzugriff in den NDR-Computer.
Das Signal RESET DRV ist beim IBM-Computer High-Aktiv und wird daher durch das PAL invertiert, um so z.B. das Latch 74LS273 nach dem Einschalten rückzusetzen. Ferner ist es dann mit dem NDR-Bus verträglich. Das Signal WAIT wird von der Leitung über einen Bustreiber an den IBM-Bus auf das Signal I/O-RDY geschaltet. Der Bustreiber ist dabei so verschaltet, daß er nur dann aktiv ist, wenn ein 0-Signal an seinem Eingang liegt. Damit wird ein offener Kollektor nachgebildet. Das Latch liefert noch vier weitere Signale, die mit 'relais' und 'a1','b1' und 'c1' beschriftet sind. Sie dienen zu späteren Erweiterungen und werden für die Grundschaltung zunächst nicht benötigt.
Die Signale, die auf die Verbindungsleitung zum NDR-Computer führen, entsprechen schon direkt denen des NDR-Busses. Man könnte sie direkt mit dem Bus verbinden. Dann wäre es aber nicht möglich, auf dem NDR-Bus auch gleichzeitig eine eigene CPU zu betreiben. Ferner ist die Leitungslänge bei direkter Verbindung kritisch.

Bild 5 zeigt die Schaltung auf der NDR-Bus-Seite. Die Steuerschaltung besteht auch hier wieder aus einem PAL-Baustein. Daten und Adressen werden auf den NDR-Bus genau dann durchgeschaltet, wenn das Signal 'sel1' auf 0 liegt. Die Steuerleitungen, RD, WR, IORQ und MREQ werden mit 'sel2' freigegeben.

Zwei Betriebsarten.

Befindet sich keine CPU auf dem NDR-Bus, dann muß man die Brücke J1 am PAL schließen und damit werden 'sel1' und 'sel2' fest auf 0V gelegt. Der IBM hat dann dauerhafte Verbindung mit dem NDR-Bus.
Ist eine CPU vorhanden, so muß die Brücke offenbleiben und dann geschieht folgendes: Wenn der IBM einen Zugriff auf den NDR-Bus durchführen will, so legt er entweder MREQ oder IORQ auf die Leitung. Das PAL schaltet diese Signale in Form eines BUSRQ (Bus Request) auf den NDR-Bus. Das PAL sorgt ferner auch dafür, daß der IBM nun zunächst einmal wartet, indem es das Signal WAIT auf 0 V legt. *Bild 6* zeigt den

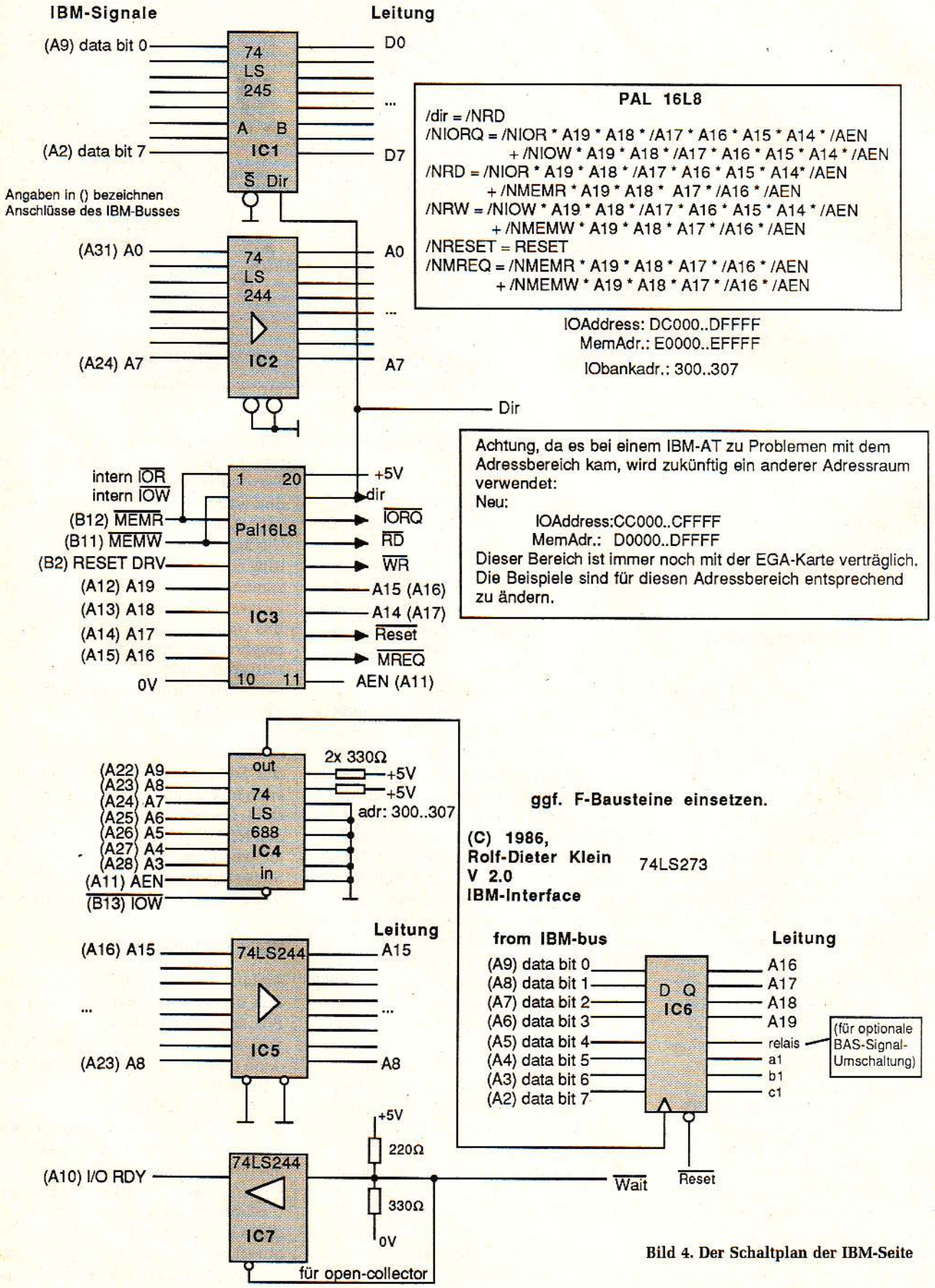
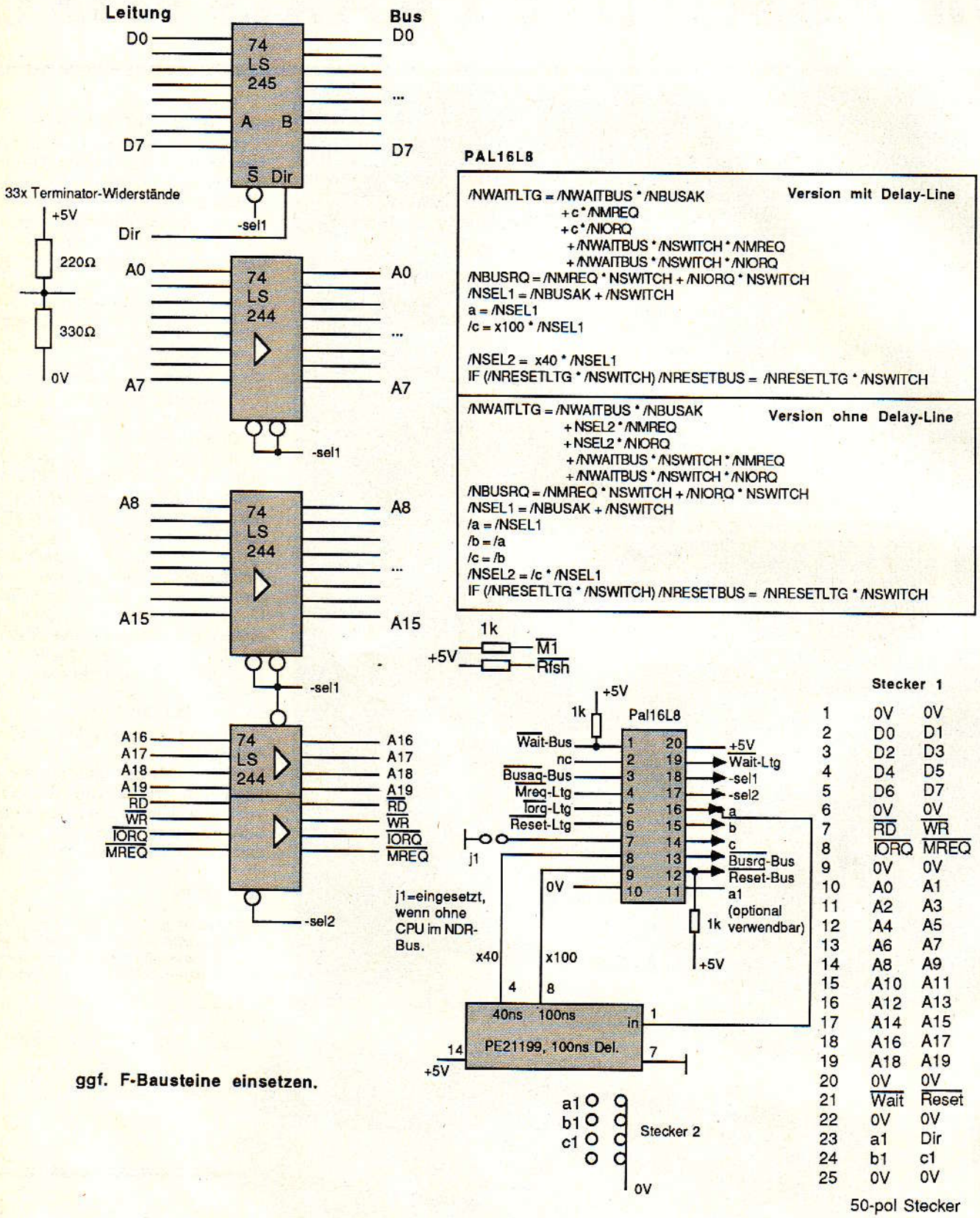


Bild 4. Der Schaltplan der IBM-Seite



ggf. F-Bausteine einsetzen.

Belegung des Leitungssteckers

Bild 5. Der Schaltplan der NDR-Seite

kompletten Zeitablauf. Nun bestätigt die CPU auf dem NDR-Bus nach einiger Zeit mit dem Signal BUSAK, daß der NDR-Bus frei ist.
Jetzt wird vom PAL das Signal 'sel1' auf 0 geschaltet und damit werden zunächst Daten- und Adreßbus auf den NDR-Bus geschaltet. Dies muß zuerst geschehen, sonst gibt es Probleme bei der Adreßerkennung. Nach ca. 50ns wird auch das Signal 'sel2' auf 0V gelegt. Damit werden die Signale IORQ, MREQ, RD und WR gültig. Bis zu dieser Zeit wurde das WAIT-Signal von dem PAL auf 0 gelegt und die IBM-CPU gebremst. Nun wird das WAIT-Signal ggf. von der NDR-Baugruppen bedient. Eine kleine Verzögerung tritt nun dadurch auf, daß die NDR-Baugruppen natürlich auch eine Weile brauchen, um zu erkennen, daß sie adressiert sind. Dadurch entsteht ein kleiner Puls auf der Wait-Leitung. Er ist ca. 40 ns breit. Er stört solange nicht, als er schmäler als die halbe Periodendauer des Betriebstaktes der IBM-CPU ist, da dort das WAIT-Signal (bzw. I/O-RDY) mit den Taktflanken des Prozessortaktes synchronisiert wird.
Will man den Puls vermeiden, so muß man die in die Schaltung eingezeichnete Verzögerungsleitung verwenden und das PAL mit anderen Gleichungen (ebenfalls eingezeichnet) belegen. Für die vorgeschlagene CPU mit dem 8088 reicht aber die einfache Version ohne Verzögerungsleitung aus. Die Leitung 'a1' ist ebenfalls auf den PAL-Baustein gelegt, um ggf. spätere Erweiterungen zu ermöglichen.

Bild 7 zeigt die Schaltung des BAS-Mischers und der Umschaltlogik. Der BAS-Mischer ermöglicht es, einen normalen Monitor als Ausgabegerät für den IBM-Teil zu verwenden. Der Umschalter erlaubt es wahlweise, den IBM- oder den NDR-Computer-Bildschirmausgang an den Monitor zu schalten. Dabei gibt es zwei Möglichkeiten. Einmal kann man dies mit Hilfe des Schalters S1 tun, oder der Computer erledigt das über die Leitung 'relais', die vom 74LS273 kommt. Dabei sind die beiden Signale Exclusiv-Oder verknüpft, so daß man wie bei einer Wechselschaltung mit beiden Signalen eine Umschaltung erreichen kann.

Bild 8 zeigt die beiden Baugruppen. Die Flachbandleitung zur Verbindung ist gemäß Stecker 1 wie in *Bild 5* belegt. *Bild 9* zeigt den Einbau in das NDR-Gehäuse. Das Gehäuse wurde dabei schon vor einiger Zeit für die IBM-Slots vorbereitet und auch die meisten NDR-Baugruppen wurden auf ein neues Format gebracht,

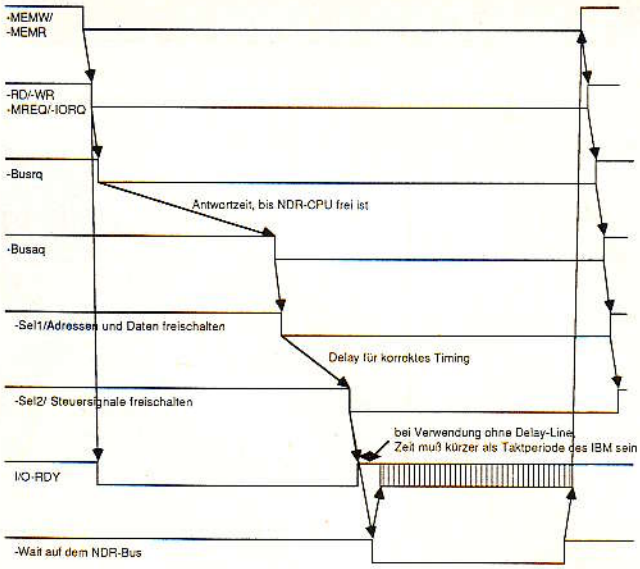


Bild 6. Zeitlicher Ablauf bei einem Zugriff

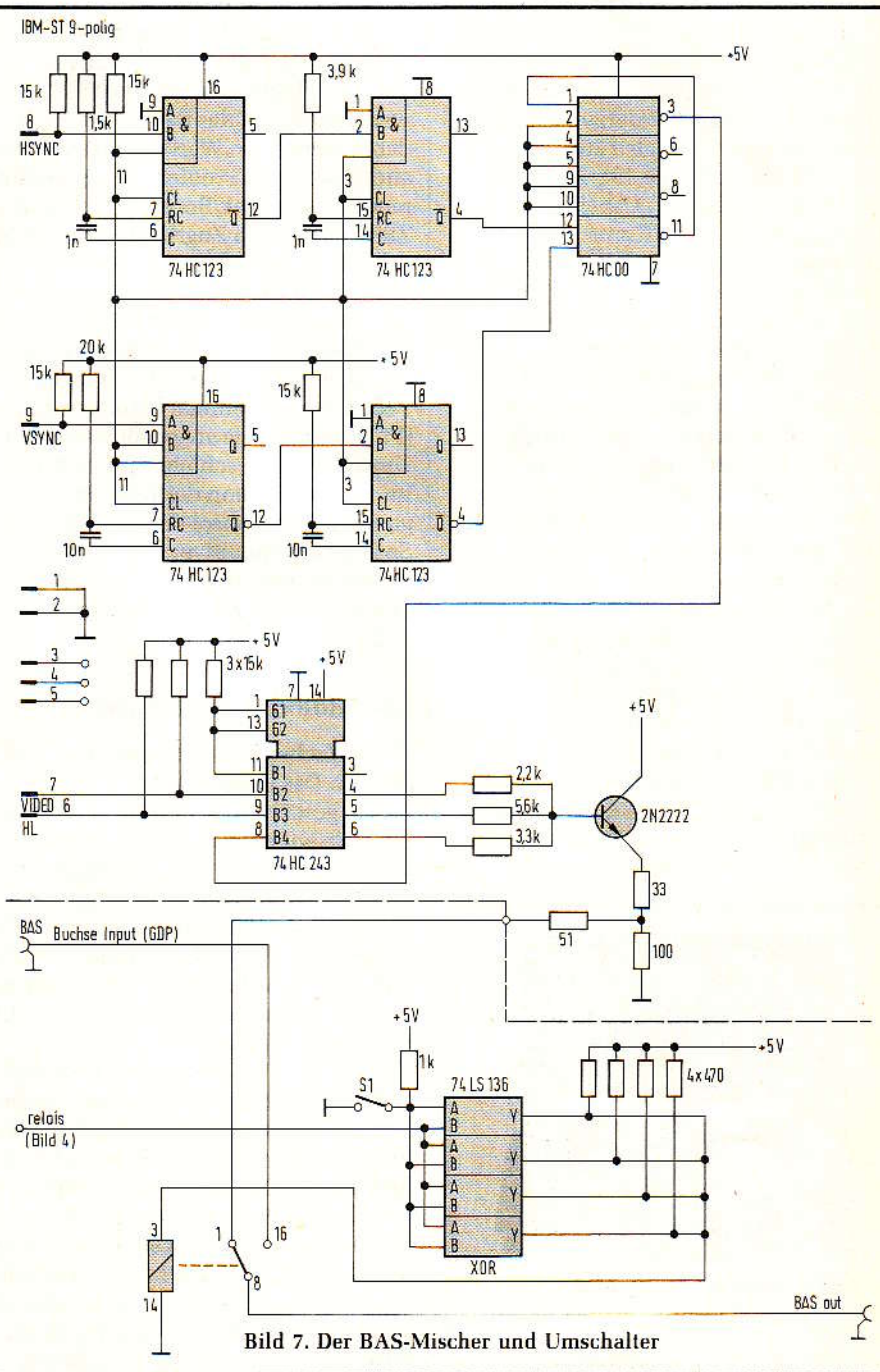


Bild 7. Der BAS-Mischer und Umschalter

so daß alle Verbindungen einheitlich an der Gehäuseseite herausgeführt sind.

Selbst bauen?

Für alle diejenigen, die den Selbstbau wagen wollen, zeigt *Bild 10* die Belegung des IBM-Busses. Wenn man nicht die fertige Leiterplatte verwenden will, so kommt man am ehesten mit einem IBM-Prototypen-Adapter zurecht. Dieser ist allerdings sicher teurer als der ganze Adapter-Bausatz.

Programmbeispiele für die Kopplung

Bild 11 zeigt eine Übersicht des IBM-Adreßraums und der Zuordung zum NDR-Adreßraum. Der IO-Adreßraum belegt vom IBM aus gesehen 32 KByte, der NDR-Computer wertet derzeit aber nur die unteren 8 Bit aus, also 256 verschiedene Adressen. Die IOs liegen im IBM von DC000 bis DFFFF. Dabei ist der Bereich DC000 bis DC0FF zum Beispiel ein sinnvoller Adreßbereich. Beim Speicher ist das Problem eher umgekehrt. Im IBM wird der Adreßraum von E0000 bis EFFFF, also 64 KByte belegt, im NDR muß aber der Bereich 00000 bis FFFFF angesprochen werden können. Dazu werden die unteren Adressen direkt vom IBM übernommen, also 0000 bis FFFF und die oberen vier Bit von einem Latch hinzugefügt. Das Latch ist auf dem IBM im IO-Adreßraum bei Adresse 300..307 ansprechbar. Will man auf eine beliebige Speicherzelle im NDR-Computer zugreifen, so muß man zunächst das Latch belegen und kann danach im Bereich E0000 bis EFFFF auf den entsprechenden Bereich zugreifen. Wenn man die Bank-Boot-Baugruppe verwendet oder die SBC3, so kann das derzeit zu einem Buskonflikt führen, da in dieser Baugruppe bisher keine Tristate-Treiber

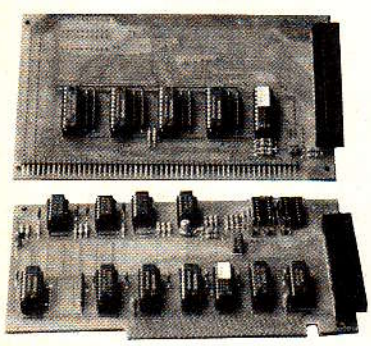


Bild 8. Diese beiden Platinen verbinden Welten

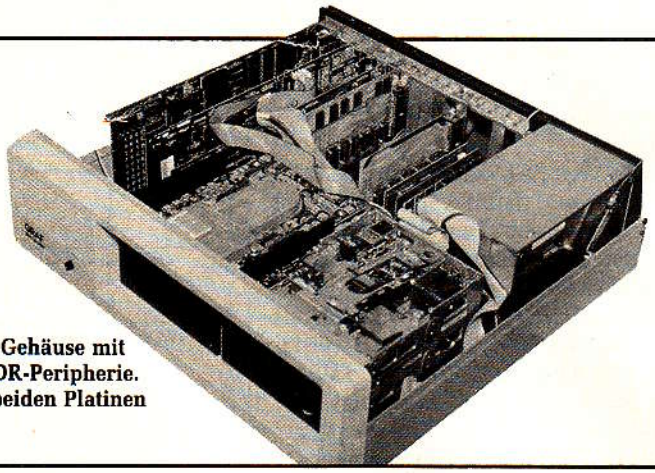


Bild 9. Das neue Gehäuse mit IBM-CPU und NDR-Peripherie. Hinten sind die beiden Platinen eingesteckt

für die Adreßleitungen A16..A19 verwendet wurden. Im normalen CP/M-Betrieb jedoch passiert nichts, da hier nur der Adreßbereich 00000 bis 0FFFF verwendet wird. In das Latch wird man dabei immer den Wert 0 schreiben. Ein anderes kleines Problem gibt es zusammen mit der Bank/Boot-Logik. Wenn der Z80 durch einen Zugriff auf Port C8 den unteren Speicherbereich x0000 bis x7FFF ausblendet und durch den Inhalt der Boot-EPROMs ersetzt, so kann auch der IBM nicht auf diesen Bereich zugreifen. Dies müßte man durch Semaphor-Technik später einmal berücksichtigen (FLOMON müßte angepaßt werden). Die Umschaltung geschieht bei FLOMON immer dann, wenn der Z80 Dienstprogramme (wie Consol-Ausgabe) aufruft. So ist es für ein universelles Kommunikationsprogramm zunächst einfacher im oberen Bereich zuzugreifen, also von 08000 bis 0FFFF.

Leuchtdioden an und aus:

Ein einfaches Beispielprogramm zeigt *Bild 12*. Das Programm ist in Turbo-Pascal geschrieben, und läßt sich daher leicht in andere Sprachen übertragen. Das Programm hat die Aufgabe, acht Leuchtdioden binär durchzuzählen. Die Leuchtdioden befinden sich auf der IOE-Baugruppe des NDR-Computers, an Port 30h. Die Prozedur LEDSAN gibt einen beliebigen Datenwert an dieses Port aus. Dazu wird mit dem MEM-Befehl von Turbo-Pascal gearbeitet. In Klammern stehen bei MEM zwei Angaben. Der erste Wert ist der sogenannte Segment-Offset. Er ist bei uns für den IO-Bereich fest und liegt bei DC00h. Der Segementoffset wird bei der Adressierung durch den 8088 zu einem Adreßwert addiert, nachdem er um vier Bit nach links geschoben wurde. Dieser Adreßwert ist bei uns 30h, denn dort liegt die IOE-Baugruppe im NDR-Adreßraum. Durch eine Zuweisung erhält dieses Port den Wert der Variablen i, die als Parameter bei Aufruf der Prozedur LEDSAN übergeben wurde.

Im Hauptprogramm wird nun eine weitere Variable i von 0 bis 255 durchgezählt und in diesem Block auch LEDSAN mit i als Parameter aufgerufen. Der Aufruf von DELAY(50) sorgt außerdem dafür, daß nach jedem Hochzählen ein klein wenig gewartet wird, so daß das Auge den Zählvorgang an den LEDs verfolgen kann. Die Zählschleife ist schließlich noch durch REPEAT UNTIL geklammert und wird solange durchlaufen, bis man irgendeine Taste (KEYPRESSED) drückt.

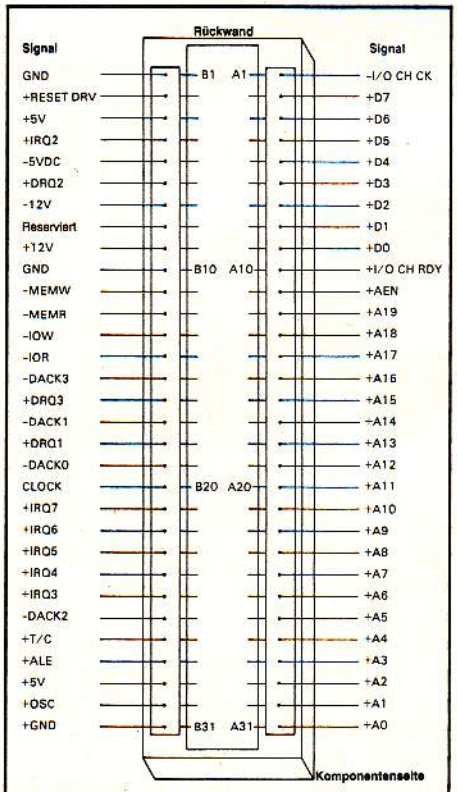


Bild 10. Die Belegung des IBM-Busses

Schalter einlesen

So einfach wie mit dem Schreiben verhält es sich auch mit dem Lesen von Daten. *Bild 13* zeigt ein Beispielprogramm. Hier wird von der IOE-Baugruppe gelesen.

Wenn man dort zum Beispiel eine Schalterreihe mit den Eingängen von Port 30h verbindet, so kann man den Schalterzustand direkt am Bildschirm in binärer Schreibweise sehen. Dabei ist Bit 0 links auf dem Bildschirm angeordnet. Die Schalterstellungen werden durch das Programm solange ausgegeben bis man eine Taste drückt.

Speicher testen

Bild 14 zeigt ein kleines Programm, das bei der Inbetriebnahme der Baugruppen ganz nützlich sein kann. Es setzt zunächst Port 300 auf 0, und wählt damit den Adreßraum 00000 bis 0FFFF im NDR-Computer aus. Dann wird der Inhalt der Speicherzelle 00000 gelesen und gleich wieder geschrieben. Nun kann man z.B. mit einem Oszilloskop den Zugriff und die Wait-Signal-Erzeugung kontrollieren. Wenn man den Test mit eingesteckter NDR-CPU fährt, sollte man allerdings besser eine der Adressen im Bereich 08000 bis 0FFFF verwenden, indem man bei MEM nach dem Doppelpunkt z.B. $8000 angibt.

Speicher anzeigen

Interessant ist es, den Speicherinhalt auf den Bildschirm des IBM zu bringen. Dazu dient das Programm in *Bild 15*. Es gibt den Speicherbereich 0F000 bis 0FFFF in sedezimaler Schreibweise auf dem IBM-Bildschirm aus. Andere Bereiche kann man leicht durch Änderung der FOR-Schleife ausgeben. Wer will, kann das Programm zu einem richtigen kleinen Monitorprogramm ausbauen, mit Funktionen wie Speicher-Lesen, Speicher-Ändern usw.

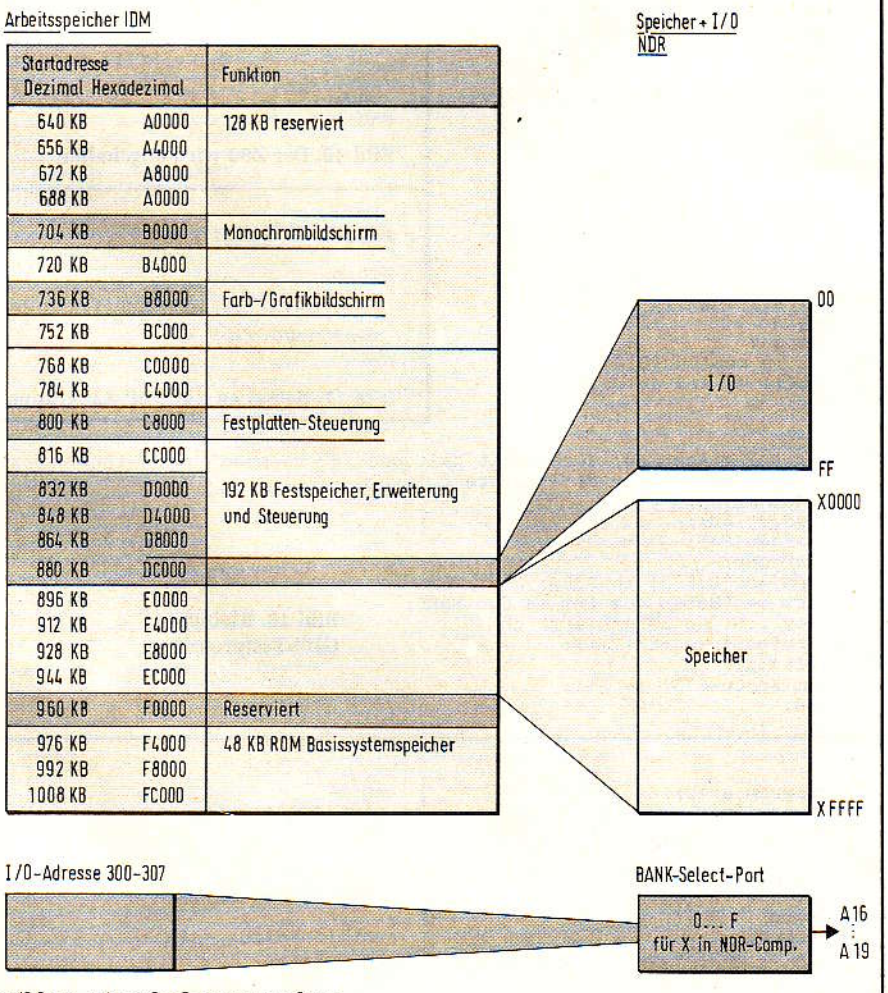


Bild 11. Die Adreßraumzuordung

Stoppt den Z80

Manchmal stört es, wenn die Z80-CPU arbeitet. In *Bild 16* ist dargestellt, wie man sie anhalten kann. Man überschreibt den Speicherbereich, in dem die CPU gerade arbeitet, mit Halt-Befehlen. Das Programm arbeite z.B. mit FLOMON. Der Z80 muß dann ständig im Bereich 0F000 bis 0FFFF arbeiten. Er springt von dort aus in die Bank/Boot-EPROMs und kehrt dann wieder zurück. Wenn man den Bereich 0F000 bis 0FFFF daher mit Halt-Befehlen überschreibt, muß der Z80 irgendwann stehen bleiben. Anschließend kann man sich den Bereich einmal mit dem Programm aus Bild 15 ansehen. Man wird feststellen, daß unter Umständen nicht alle Zellen der Wert 76h (Halt-Befehlscode) enthalten, da der Z80 auch seinen Stack im Bereich vor FFFF betreibt, und er vor Erreichen eines Halt-Befehls dort noch Werte ablegen kann. Wenn der Z80 einmal steht, kann man zum Beispiel auch mit der GDP direkt vom IBM aus arbeiten, sonst wird sie vom FLOMON verwendet.

GDP löschen

Bild 17 zeigt ein einfaches Programm, das einen Lösch-Befehl an die GDP-Baugruppe schickt. Sicherheitshalber sollte man allerdings auch Port 60h vor Einsatz des Befehls auf 0 stellen, sonst wird nur die aktuelle Bildseite gelöscht.

GDP Textausgabe

Wie man einen Text auf den Bildschirm der GDP bringt zeigt *Bild 18*. Hierzu sollte man den Z80 vorher lahmlegen. Doch probieren Sie es auch einmal mit laufendem FLOMON aus, es können jedoch dabei Störungen auftreten, wenn der Z80 zum Beispiel die Bildseite gerade umschaltet, bevor der IBM seine Ausgabe gemacht hat.

Der Z80 als intelligentes Terminal

Eine interesante Anwendung ergibt sich bei FLOMON: echter Multiprozessor-Betrieb. Der Z80 dient als intelligentes Terminal mit Grafik-Ausgabe. Der IBM kann dann dort Bilder zeichnen. Die Anordnung ähnelt dem Verhalten der TERM1. Dazu wird ein Programm auf dem NDR gestartet, das wie folgt aussieht:

```
START:
XOR A
LD (8000h),A
;löschen der Kommunikationszelle
```

```
program ledstest;

 var i:integer;

procedure ledsan(i:integer);
begin
 mem[$dc00:$30] := i;
end;


begin
 writeln('Rolf-Dieter Klein, IBM-NDR Kopplung');
 writeln('LEDs auf Adresse 30h, IOE werden angesteuert');
 repeat
  for i:=0 to 255 do begin
   delay(50);
   ledsan(i);
  end;
 until keypressed;
end.
```

Bild 12. Leuchtdioden auf der IOE schalten

```
program switchtest;


procedure schalteraus;
 var val,i : integer;
begin
 val := mem[$dc00:$30];
 for i:=0 to 7 do begin
  write(val mod 2);
  val := val div 2;
 end;
 writeln;
end;

begin
 writeln('IBM - NDR, Rolf-Dieter Klein');
 writeln('Schalter-Test Port 30h');
 repeat
  schalteraus;
 until keypressed;
end.
```

Bild 13. Schalter auf der IOE ablesen

```
program memtest;

var i : integer;

begin
 port[$300] := 0; (* Bank 0 anwaehlen *)
 repeat
  i:=mem[$e000:0];
  mem[$e000:0] := i;
 until keypressed;

end.
```

Bild 14. Testprogramm für den Speicherzugriff

```
program spdump;

var i : integer;

procedure hexaus( n : integer);
procedure nibble( i:integer);
begin
 if (i >= 10) then
  write(chr(i -10 + ord('A')))
 else
  write(chr(i + ord('0')));
end;

begin
 nibble(n div 16);
 nibble(n mod 16);
end;


begin
 port[$300] := 0;
 for i:=$f000 to $ffff do begin
   if (i mod 16) = 0 then writeln;
   hexaus(mem[$e000:i]);
   write(' ');
 end;
end.
```

Bild 15. Programm zum Speicherdump

```
program gdptest;
var i,x,y: integer;
 str: string[100];

procedure befgdp(i : integer);
var k:integer;
begin
 repeat
  k := mem[$dc00:$70];
 until (k and 4) <> 0;
 mem[$dc00:$70] := i;
end;

procedure waitgdp;
var k:integer;
begin
 repeat
  k := mem[$dc00:$70];
 until (k and 4) <> 0;
end;

begin
 writeln('Achtung, vorher mit halt.pas z80 stoppen');
 mem[$dc00:$60] := 0; (* Seite 0 *)
 befgdp($4);
 befgdp($d);
 befgdp($e);
 waitgdp;
 mem[$dc00:$73] := $33;
 str := 'Daten vom IBM an den NDR';
 for i:=1 to length(str) do
  befgdp(ord(str[i]));
waitgdp;
mem[$dc00:$73] := $11;
end.
```

Bild 18. Kleines GDP-Testprogramm

```
program haltz80;
var i : integer;
begin
 port[$300] := 0;
 for i := $f000 to $ffff do begin
  mem[$e000:i] := $76;
 end;
end.
```

Bild 16. Der Z80 wird angehalten

```
program clrgdp;

begin


 mem[$dc00:$70]:= 7;
end.
```

Bild 17. Befehl an die GDP-Baugruppe

```
program exec;

var ch : char;

procedure cmdout(i : integer);
var n : integer;
begin
 repeat
  n := mem[$e000:$8000];
 until n=0;
 mem[$e000:$8000] := i;
end;

begin
 port[$300] := 0;
 cmdout($1a);
 repeat
  read(kbd,ch);
  cmdout(ord(ch));

 until ord(ch) = 1;
 cmdout(2);
end.
```

Bild 19. Der NDR-Computer als TERM1

```
SCHLEIFE:
LD A,(8000h)
;warten bis Befehl da
OR A
;0= kein Befehl da
JR Z,SCHLEIFE
LD C,A
;Befehl ausführen
CALL 0F009h
;CO-Routine im FLOMON
JR START
;Befehl löschen und dann warten
```

Bild 19 zeigt den Programmteil auf dem IBM. Hier als Test programmiert, kann man nun auf der IBM-Tastatur Text oder Befehle eintippen und erhält eine Ausgabe auf dem NDR.
Wenn man z.B. die Sequenz:

```
ESC ESC G CR
M0 0 CR
R100 100 CR
```

eingibt, erhält man ein Rechteck auf dem Bildschirm. Die verwendeten Grafik-Befehle sind übrigends im Sonderheft Mikrocomputer Schritt für Schritt 2 beschrieben, bzw. im Buch „Rechner modular", von R. D. Klein, Franzis-Verlag.