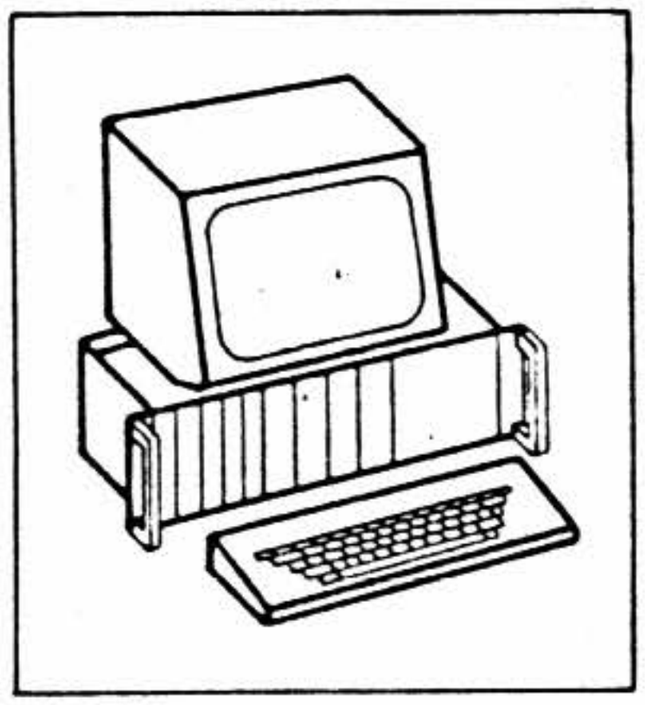
FACHPRAKTISCHE ÜBUNG MIKROCOMPUTER-TECHNIK



NSC-800-Adapter

BfZ MFA

Inhaltsverzeichnis

Funktionsbeschreibung

1. Allgemeines

Mit dem NSC800-Mikroprozessor liegt ein Prozessor vor, der zur Vereinigung der jeweiligen Vorteile der Prozessoren Z80 und 8085 entwickelt worden ist. Er ist hardware-mäßig dem 8085- und software-mäßig dem Z80-Prozessor weitgehend kompatibel. Damit bietet er die Möglichkeit, den 8085-Prozesssor im MFA-Mikrocomputersystem mit geringem Aufwand durch einen Z80-kompatiblen Prozessor zu ersetzen. Da der Befehlssatz des 8085-Prozessors bis auf die Befehle RIM und SIM eine Teilmenge des Z80-Befehlssatzes darstellt, können 8085-Programme in Z80-Systemen ablaufen. Umgekehrt ist das nicht möglich, d.h., Z80-Programme laufen nicht in 8085-Systemen.

Diese Ausbaustufe für den MFA-Mikrocomputer ist für das CP/M-System vorgesehen, weil sich damit die Möglichkeit eröffnet, das gesamte, für CP/M-Systeme verfügbare Software-Angebot nutzen zu können.

Beispielsweise wird es mit diesem Prozessor möglich, Hochsprachen wie Turbo-PASCAL auf dem MFA-Mikrocomputer einzusetzen. Speziell diese Programmiersprache ist im Personal-Computer-Bereich weit verbreitet und bietet auch im MFA-System weitere didaktische Möglichkeiten.

Darüber hinaus kann der Makroassembler M80, der Teil des Microsoft-Entwicklungspaketes (BASIC-Interpreter/-Compiler) ist, durch Ausnutzung der Z80-Mnemonik für die Einführung in die Z80-Assemblersprache verwendet werden.

2. Funktionsbeschreibung

Dem Stromlaufplan der Adapter-Platine (Anhang A1) kann entnommen werden, daß die Signale des NSC800-Prozessors denen des 8085-Prozessors weitgehend entsprechen und sich teilweise lediglich in der Bezeichnung unterscheiden. Allerdings ist die Pin-Belegung zum 8085 verschieden und einige Signale müssen invertiert werden.

In der folgenden Beschreibung wird daher im wesentlichen auf die Besonderheiten dieses Prozessors und auf die Unterschiede zum Z80 oder 8085 eingegangen.

Die Analogie der Anschlüsse zwischen dem 8085- und dem NSC800-Prozessor kann dem Schaltbild direkt entnommen werden. Der 8085-Prozessor unterscheidet sich insbesondere durch die Anschlüsse SID (Serial Input DATA, Pin 5) und SOD (Serial OUTPUT DATA, Pin 4) vom NSC800. Dafür besitzt der NSC800 die Anschlüsse PS (Power Save, Pin 39) und RFSH (Refresh, Pin 28). Auf der Adapterplatine sind diese Signale durch offene Lötbrücken vom Platinenstecker getrennt und der Eingang PS (Pin 39) liegt über einen Pull-Up-Widerstand fest auf H-Pegel. Diese Signale werden hier nicht weiter verwendet.

Funktionsbeschreibung

Darüber hinaus ist der Interrupt-Eingang NMI (Non-Maskable Interrupt, Pin 21) des NSC800 vom Platinenstecker getrennt und ebenfalls fest auf H-Pegel gelegt. Dieser Anschluß entspricht dem Interrupt-Signal TRAP des 8085. Gegenüber dem 8085 ist das NMI-Signal ein aktiv Low-Signal und die zugehörige (Restart-)Adresse ist vom 8085 verschieden.

8085-Anschluß	Restart-Adresse	NSC800-Anschluß	Restart Adresse
TRAP (Pin 6)	0024H	$\overline{\text{NMI}}$ (Pin 21)	0066H

Die Anpassung des NSC800-Prozessors an die 8085-Prozessor-Platine erfolgt über einen PAL-Baustein (GAL16V8). Sie wurde so vorgenommen, daß die sich im aktiven Pegel unterscheidenden Signale über Inverter geführt werden. Darüber hinaus wurde auch eine weitgehende Kompatibilität zwischen dem RST7.5-Anschluß des 8085-Prozessors und dem RSTA-Anschluß des NSC800 hergestellt. Beide Anschlüsse unterscheiden sich dadurch, daß der RST7.5-Eingang des 8085 flankengsteuert arbeitet, während der RSTA-Eingang des NSC800 statisch wirkt. Da der flankengesteuete RST7.5-Eingang Vorteile bei der Anwendung bietet, wurde mittels des PAL-Bausteins diese Anpassung ebenfalls vorgenommen. So lassen sich leichter interrupt-gesteuerte Programme realisieren, die in Verbindung mit der Zähler- und Zeitgeber-Baugruppe (BFZ/MFA 4.6) ausgetestet werden können, indem ein Interrupt über den Handtaster in der Frontplatte der Baugruppe ausgelöst wird. Dabei ist es dann unbedeutend, wie lange der Taster betätgt wird und es brauchen keine programmiertechnischen Abfragen in das Programm eingefügt werden.
Im Anhang ist ein einfaches Beispiel für ein solches interruptgesteuertes Programm angegeben, welches manuell getestet werden kann.

Funktionsbeschreibung

2.1 Die Eingangssignale des NSC800

Im folgenden sind alle NSC800-Bausteinanschlüsse kurz beschrieben. Für eine genaue Funktionsbeschreibung sei auf das zugehörige Datenblatt der Firma National Semiconductor Corporation verwiesen.

RESET INPUT (Pin 33):
Aktiv Low-Signal; schaltet die Leitungen A8 - A15 sowie AD0 - AD7 in den Tri-State und löscht die Register PC, I und R; sperrt Interrupts und erzeugt das Signal RESET OUT.

BUS REQUEST (Pin 36):
Aktiv Low-Signal; wird verwendet, wenn ein anderes Gerät den BUS anfordert; die CPU erkennt das Signal am Ende des aktuellen Maschinenzyklusses; bewirkt, daß die Leitungen A8 - A15, AD0 - AD7, sowie IO/M, RD, and WR in den Tri-State geschaltet werden; die CPU bestätigt die Anforderung durch das Signal BUS ACKNOWLEDGE.

NON-MASKABLE INTERRUPT (Pin 21):
Aktiv Low-Signal; flankengesteuert; nicht-sperrbares Interrupt-Signal mit der höchsten Priorität, d.h. unabhängig vom Interrupt-Enable-Flipflop; wenn aktiv, so sichert die CPU den PC auf dem STACK und verzweigt zur Adresse 0066H.

RESTART INTERRUPTS A,B,C (Pin 22,23,24):
Aktiv Low-Signale; sperrbare Interrupt-Signale; Ausführung wie beim NMI-Interrupt; Priorität in der Folge A, B, C; zugehörige Adressen RSTA -> 003CH, RSTB -> 0034H, RSTC -> 002CH.

8085-Anschluß	Restart-Adresse	NSC800-Anschluß	Restart Adresse
RST7.5 (Pin 7)	003CH	RSTA (Pin 22)	003CH
RST6.5 (Pin 8)	0034H	RSTB (Pin 23)	0034H
RST5.5 (Pin 9)	002CH	RSTC (Pin 24)	002CH

Funktionsbeschreibung

INTERRUPT REQUEST (Pin 25):
Aktiv Low-Signal; Interrupt mit der niedrigsten Priorität; die CPU unterscheidet drei Betriebsarten; in der Betriebsart 0 kann wie beim 8085 über Restart-Vektoren (RST 0 - RST 7) bzw. über einen drei-Byte Call-Befehl zur Interrupt-Service-Routine verzweigt werden; in der Betriebsart 1 verzweigt die CPU fest zur Restart-Adresse 0038H; die Betriebsart 2 erfordert spezielle Ein-/Ausgabe-Bausteine, die nach Anforderung eines Interruptes das Low-Byte einer Adresse liefern können, dieser Teil ergänzt den Inhalt des Interrupt-Vektor-Registers der CPU und bildet eine Speicheradresse, von der die CPU die Adresse der Interrupt-Service-Routine liest (indirekter Sprung zur ISR).

WAIT (Pin 38):
Aktiv Low-Signal; die CPU verlängert die Maschinenzyklen durch Warte-Zyklen, wenn dieses Signal während eines RD-, WR- oder INTA-Maschinenzyklusses aktiviert wird.

POWER SAVE (Pin 39):
Aktiv Low-Signal; wenn aktiv, dann stoppt die CPU am Ende des Befehlszyklusses und geht in einen Low-Power-Mode mit geringem Stromverbrauch.

Xin,Xout (Pin 10,11):
Quarzanschlüsse für den integrierten Taktgenerator.

2.2 Ausgangssignale des NSC800

BUS AKCKNOWLEDGE (Pin 35):
Aktiv Low-Signal; zeigt an, daß die CPU den BUS und die Steuer-Signale in den Tri-State geschaltet hat (siehe BUS REQUEST).

A8 - A15 (Pin 1 - 8):
Aktiv High-Signale; acht höchstwertige Adress-Leitungen.

AD0 - AD7 (Pin 12 - 19):
Adress- und Daten-Leitungen, die im Betrieb umgeschaltet (gemultiplext) werden; die Leitungen werden beim Datentransport auch als Eingänge verwendet.

RESET OUT (Pin 37):
Aktiv High-Signal; wenn aktiv, so führt die CPU einen RESET aus.

INPUT/OUTPUT/MEMORY (Pin 34):
Das High-Sgnal zeigt einen Schreib- oder Lese-Zugriff der CPU auf Ein-/Ausgabe-Baugruppen an; das Low-Signal entsprechend den Zugriff auf den Speicher.

REFRESH (Pin 28):
Aktiv Low-Signal; zeigt einen Refresh-Zyklus für dynamische RAM-Speicher an.

ADDRESS LATCH ENABLE (Pin 30):
Aktiv High-Signal; ein High-Low-Signalwechsel signalisiert eine gültige Adresse, das untere Adreß-Byte muß mit diesem Signal gespeichert werden.

READ STROBE (Pin32):
Aktiv Low-Signal; die CPU liest Daten mit dem Signalwechsel am Ende des Signals.

WRITE STROBE (Pin 31):
Aktiv Low-Signal; die CPU sendet Daten während des aktiven Signals.

CLOCK (Pin 9):
Systemtakt mit einer Frequenz, die halb so groß ist wie die Frequenz des Taktgenerators.

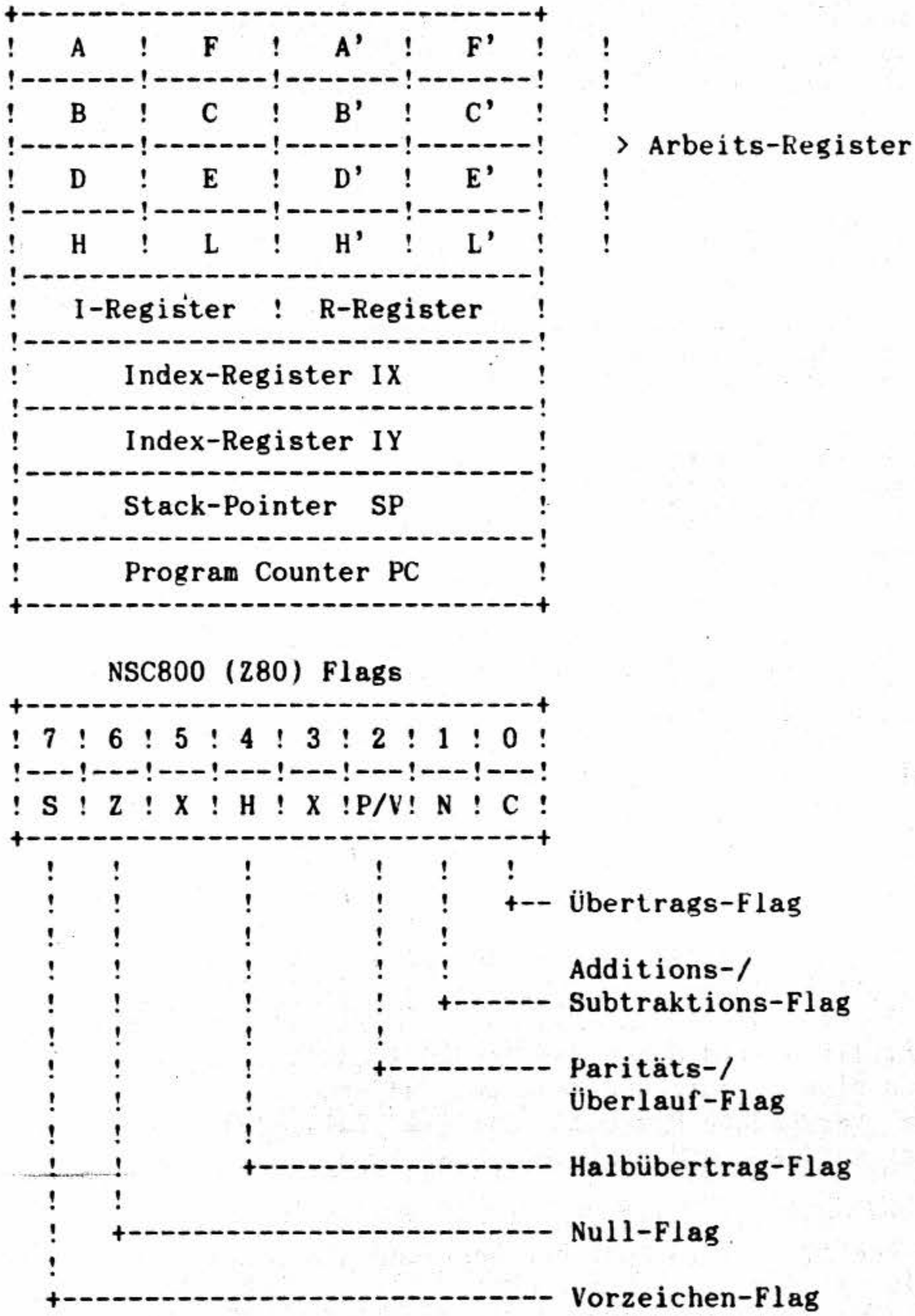
INTERRUPT AKNOWLEDGE (Pin 26):
Aktiv Low-Signal; CPU liest mit diesem Signal den Interrupt-Vektor der Interface-Baugruppe, die den Interrupt ausgelöst hat.

3. Der Register- und Befehlssatz des NSC800

Der Registersatz des NSC800 entspricht dem des Z80, d.h. er besitzt gegenüber dem 8085 den doppelten Arbeitsregistersatz sowie zwei Indexregister (IX und IY). Daneben gibt es zwei weitere Register, das Interrupt-Vektor- (I) und das Refresh-Register (R).
Der NSC800 unterscheidet in der Interrupt-Verarbeitung des Interrupt-Request-Signals (INTR, Pin 25) zwischen drei Betriebsarten. Die Betriebsart 2, auch Mode 2 genannt, verlangt spezielle Ein-/Ausgabe-Bausteine als Interrupt-Steuerbausteine. In dieser Betriebsart wird der Inhalt des I-Registers als oberer Adressteil einer Zeigeradresse für das Auffinden der Interrupt-Service-Routine verwendet. Werden dynamische RAM-Bausteine als Speicher verwendet, so müssen die RAM-Zellen in bestimmten Zeitabständen aufgefrischt werden. Der Z80 sowie der NSC800 verwenden dafür ein spezielles Register, welches als Refresh-Zähler verwendet werden kann. Im Gegensatz zum Z80 ist dieses Refresh-Register auf 8-Bit erweitert worden.

Register- und Befehlssatz

NSC800 - REGISTER

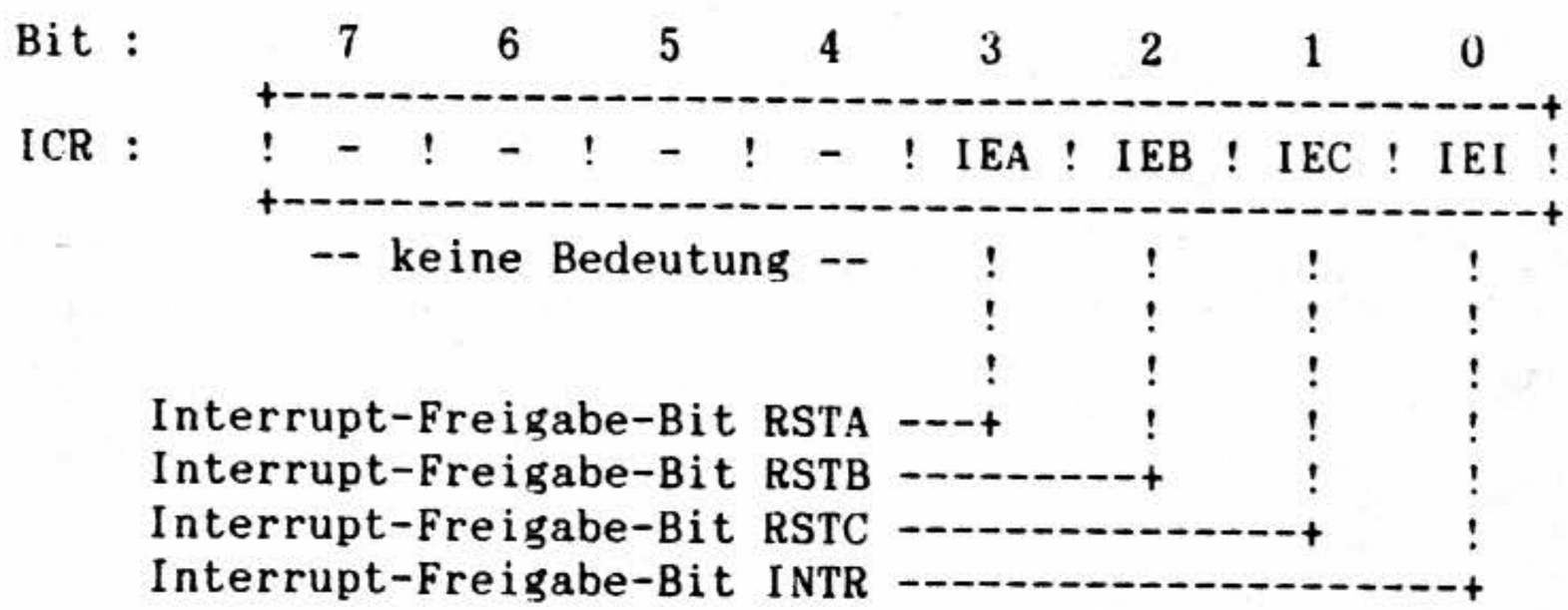


NSC800 (Z80) Flags

3.1 Das Interrupt-Control-Register (ICR)

Wesentlicher Unterschied zum Z80 ist ein weiteres Register, das Interrupt-Control-Register (ICR), welches in der vorstehenden Darstellung nicht aufgeführt ist. Dieses Register wurde notwendig, um die zum 8085 kompatiblen Interruptsignale RSTA, RSTB und RSTC maskieren, d.h. sperren oder freigeben zu können. Diese drei Interruptsignale, die der Z80 nicht besitzt, entsprechen den 8085-Interruptsignalen RST7.5, RST6.5 und RST5.5. Der 8085 besitzt zum Lesen und Schreiben seines Interrupt-Masken-Registers die Befehle RIM und SIM, die beim NSC800-Prozessor nicht vorhanden sind. Sein Befehlssatz ist voll kompatibel zum Z80. Daher

wird beim NSC800 das Interrupt-Control-Register wie eine Ausgabe-Einheit mit einer festen Baugruppen-Nummer (BBH) behandelt und kann mit einem OUT-Befehl verändert werden. Im folgenden Bild ist die Zuordnung der einzelnen Masken-Bits zu den Interrupt-Signalen wiedergegeben.



Damit eine Interrupt-Anforderung von der CPU angenommen werden kann, muß das zugehörige Freigabe-Bit gesetzt (= 1) sein. Darüber hinaus müssen Interrupts über den Befehl Enable Interrupt (EI) freigegeben sein.

3.2 Der Befehlssatz

Im Anhang A6 ist die Befehlsliste des NSC800-/Z80 dargestellt. Um die Liste möglichst kurz und übersichtlich zu gestalten, wird eine Symbolik verwendet, die in solchen Kurzdarstellungen üblich ist. In der Befehlsliste wird neben der Mnemonik und der Wirkung eines Befehls auch eine Erläuterung gegeben, sofern dies erforderlich ist. Die verwendete Sysmbolik soll im folgenden noch einmal kurz erklärt werden, um die Liste lesbar zu machen.

Ein Datum (oder ein Operand), das von einem Befehl beeinflußt wird, ist durch seine Adresse im Speicher oder durch das Register gekennzeichnet, in welchem es sich befindet. Dieser Sachverhalt wird durch folgende Schreibweise ausgedrückt:

< Adresse > = Datum im Speicher unter Adresse
< Register> = Datum im Prozessor-Register

Die spitzen Klammern symbolisieren den Inhalt der Speicherstelle oder des Registers und können wie folgt gelesen werden:

< ... > = "Inhalt von ..."

In der Ausführung der Befehle kommt es vor, daß ein Datun im Speicher beeinflußt wird, dessen Adresse in einem Registerpaar, z.B. dem HL-Registerpaar, steht. Dieser Sachverhalt wird wie folgt dargestellt:

<< Registerpaar >> = Datum im Speicher unter der Adresse, die im Registerpaar steht

Wird nun durch einen Befehl ein Datentransport ausgeführt, so wird dies durch die in der Programmiersprache PASCAL übliche Wertzuweisung zum Ausdruck gebracht:

< Adresse > := Ergebnis
< Register> := Ergebnis

Dies Bedeutet, daß das Ergebnis unter der Adresse im Speicher oder in dem angegebenen Register abgelegt wird.

Mit dieser Schreibweise (Notation) ist es möglich, die Befehlsliste kurz und übersichtlich zu gestalten. Nach einer Eingewöhnungsphase wird es dem Benutzer keine Probleme mehr machen, mit einer solchen Befehlsliste umgehen zu können.

Im Einzelfall kann es notwendig werden, die exakte Beschreibung eines Befehls nachlesen zu können. Dazu sei auf die Darstellungen in den Datenbüchern verwiesen oder auf die umfangreiche Fachliteratur, die es zum Z80-Prozessor gibt.

4. Inbetriebnahmehinweise

Für die Inbetriebnahme der Baugruppe ist lediglich die 8085-CPU auf der Prozessor-Baugruppe BFZ/MFA 2.1 gegen die Adapter-Platine auszutauschen. Dabei ist darauf zu achten, daß alle Steckerstifte korrekt in den 40poligen IC-Stecker auf der Basisplatine eingeführt werden. Im Anhang A3 ist die mit der Adapter-Platine bestückte CPU-Baugruppe abgebildet.

Neben dem Austausch der CPU ist das BOOT-EPROM 2716 mit dem BIOS-Programm für das CP/M-Betriebssystem auf der 64-K-RAM-Baugruppe BFZ/MFA 3.3 bzw. auf der ersten 16-K-RAM/EPROM-Baugruppe BFZ/MFA 3.2 auszutauschen (Anhang A4). Auf den Speicher-Baugruppen bleiben alle Brücken unverändert.

Nach dem Einschalten der Betriebspannung muß sich das System wie in der ursprünglichen 8085-Konfiguration verhalten und das Betriebsprogramm von der Diskette lesen. Dies ist der Fall, wenn am Laufwerk A der Zugriff durch Aufleuchten der roten LED angezeigt wird. Nach dem Laden des Betriebsprogramms muß das MC-System die Bereitschaft durch Ausgabe der Kommandoanforderung " A > " anzeigen.

Programmierbeispiele

Sollten hier Probleme auftreten, so führen Sie die gleichen Funktionskontrollen und Inbetriebnahmeschritte durch, wie sie in den Fachpraktischen Übungen zur Prozessor-Baugruppe, der Speicher-Baugruppe und zum CP/M-System beschrieben sind.

Sowohl die BUS-Signalanzeige wie auch der BUS-Signalgeber können für den Funktionstest wie in gewohnter Weise verwendet werden.

5.1 Z80-Programmbeispiel

Im Anhang A7 ist ein Programmbeispiel abgebildet, welches in der Mnemonik des Z80 geschrieben ist und mit dem Makro-Assembler M80 übersetzt wurde. Der Makro-Assembler M80 ist ein universeller Assembler, der von der Firma Microsoft zusammen mit den Hochsprachen vertrieben wird. Er ist beispielsweise in dem Mikrosoft-Entwicklungspaket BASIC-Interpreter/-Compiler enthalten und für die Einbindung von Assembler-Programmteilen in Hochsprchenprogrammen vorgesehen. Der M80 ermöglicht zusammen mit dem Linker (Binder) L80 die modulare Entwicklung von Programmen. Darüber hinaus bietet er die Möglichkeit, neben der 8080/85-Mnemonik auch Programmteile in der Z80-Mnemonik zu übersetzen, so daß mit dem gleichen Assembler in die Z80-Programmierung eingeführt werden kann.
Das vorliegende Beispiel realisiert einen Zähler an der Ausgabe-Baugruppe (BG-Nr. 00) der über einen Zeit-Interrupt von der Zähler- und Zeitgeber-Baugruppe gesteuert wird. Es wurde deshalb gewählt, um die mit dem 8085-Prozessor vergleichbaren Interrupt-Möglichkeiten durch die Hardware-Anpassung über den PAL-Baustein auf der NSC800-Adapter-Platine zu demonstrieren. Für den Lerner ist es günstig, die Interrupts zunächst manuell über den Handtaster in der Frontplatte der Zähler- und Zeitgeber-Baugruppe auszulösen, um die Funktion und Wirkungsweise des Programms zu testen. Dafür ist es günstig, wenn der Interrupt-Eingang der CPU auf eine Signalflanke reagiert. Das ist beim RSTA-Eingang des NSC800 nicht der Fall, wird aber mit Hilfe eines Schaltwerkes im PAL-Baustein realisiert.
Das Programm ist so dokumentiert, daß es hier nicht weiter beschrieben werden muß. Es wurde wie folgt mit Hilfe des Assemblers M80 und des Linkers L80 übersetzt:

```
A>M80 ZAEHLER,ZAEHLER=ZAEHLER
        !       !       !
        !       !       +--- Name der Quell-Datei
        !       !            z.B. ZAEHLER.MAC
        !       +------------ Name der Object-Datei
        !                     ZAEHLER.REL
        +-------------------- Name der List-Datei
                              ZAEHLER.PRN
```

Programmier-Beispiele

Der Assembler erzeugt zunächst einen Maschinencode für die Start-Adresse 0000H. Diesen Code nennt man den relativen Maschinencode. Er wird vom M80 in der Object-Datei mit dem Datei-Namen ZAEHLER.REL abgelegt. Erst mit Hilfe des Linkers L80, dem man die wirkliche (absolute) Programmstart-Adresse mitteilen muß, wird der ablauffähige Programmcode erzeugt.

```
A>L80 /P:100,ZAEHLER,ZAEHLER/N/E
      !     !      !     ! !
      !     !      !     ! +-Beenden des LINK-Vorgangs
      !     !      !     !
      !     !      !     +---Abspeichern des Programm-
      !     !      !         codes
      !     !      +---------Name der Programm-Datei
      !     !                hier ZAEHLER.COM
      !     +----------------Name der Object-Datei
      !                      hier ZAEHLER.REL
      +-----------------------Start-Adresse für den
                             Programmcode (CP/M-
                             Startadresse 0100H)
```

Das abgebildete Listing stellt den Inhalt der vom Assembler erzeugten List-Datei ZAEHLER.PRN dar.

5.2 Turbo-PASCAL-Programmbeispiel

Auf Personal- und Hobby-Computern ist zwar die Programmiersprache BASIC am weitesten verbreitet, weil sie zweifellos leicht erlernbar ist und seit je her als Interpreter ein interaktives Arbeiten erlaubt. Jedoch bieten die heute gebräuchlichen Compiler-Sprachen ebenfalls diese Möglichkeit. Dies verhalf insbesondere der Programmiersprache Turbo-PASCAL zu einem Durchbruch im Personal-Computerbereich. Leistungsfähigkeit, Einfachheit in der Bedienung sowie der günstige Preis führten dazu, daß Turbo-PASCAL heute ein Standard darstellt. Turbo-PASCAL wurde ursprünglich für CP/M-Systeme entwickelt. Die heute auf den IBM- und kompatiblen Personal-Computer eingesetzten Versionen sind bezüglich der Bedienoberfläche weiter verbessert und im Sprachumfang, angepaßt an die PC-Möglichkeiten, erweitert worden. Welche didaktischen Möglichkeiten bietet diese Programmiersprache?
Ein wesentliches Merkmal dieser wie vergleichbarer höherer Programmiersprachen ist das Prozedur-/Unterprogramm-Konzept. Programme werden vergleichbar mit der Assembler-Programmierung in übersichtliche und funktionsbezogene Einheiten, d.h. in Unterprogramme unterteilt, die später durch Angabe des Namens aufgerufen werden.

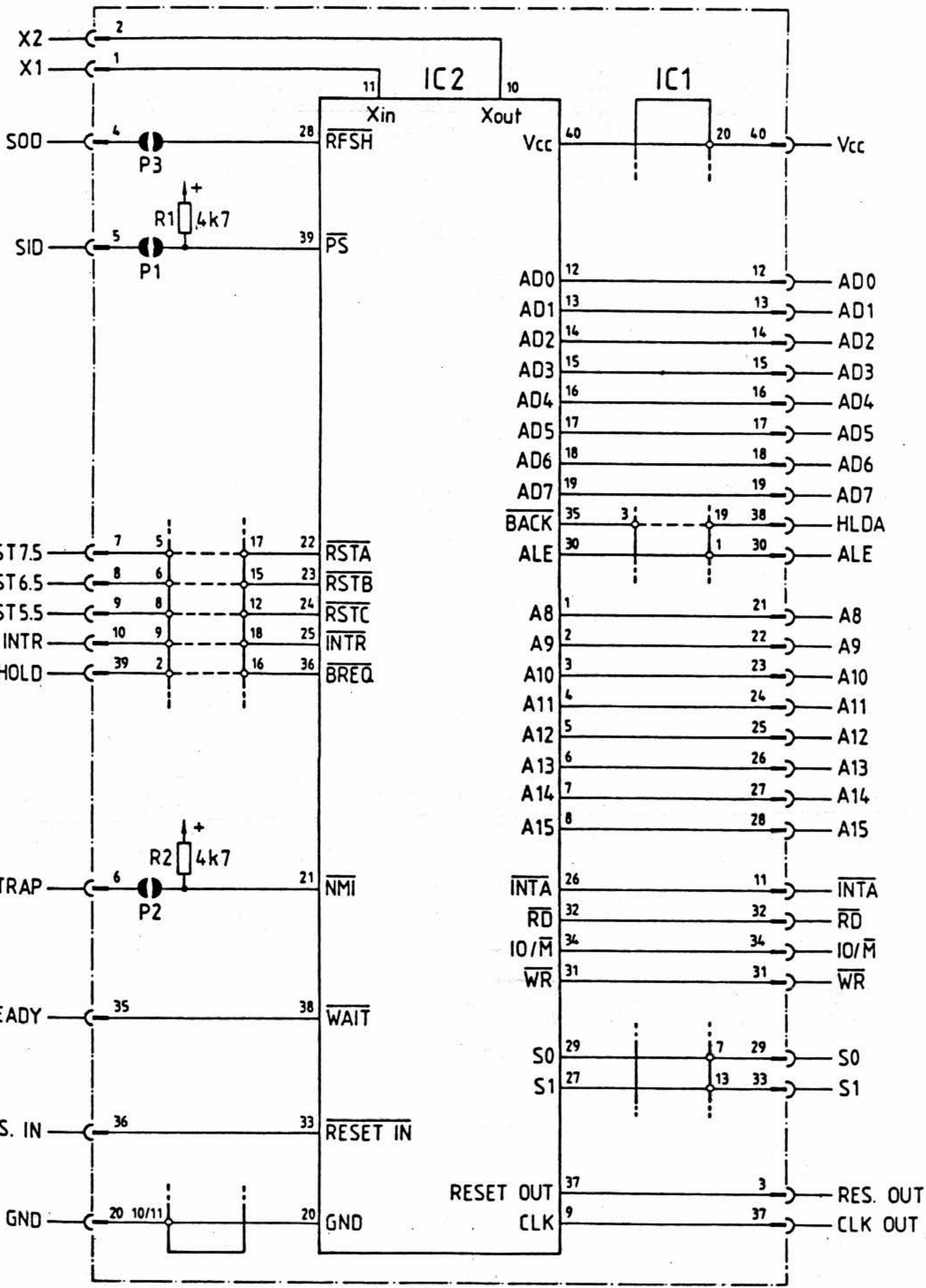
Programmier-Beispiele

Ein PASCAL-Programm könnte beipsielsweise folgende Form haben:

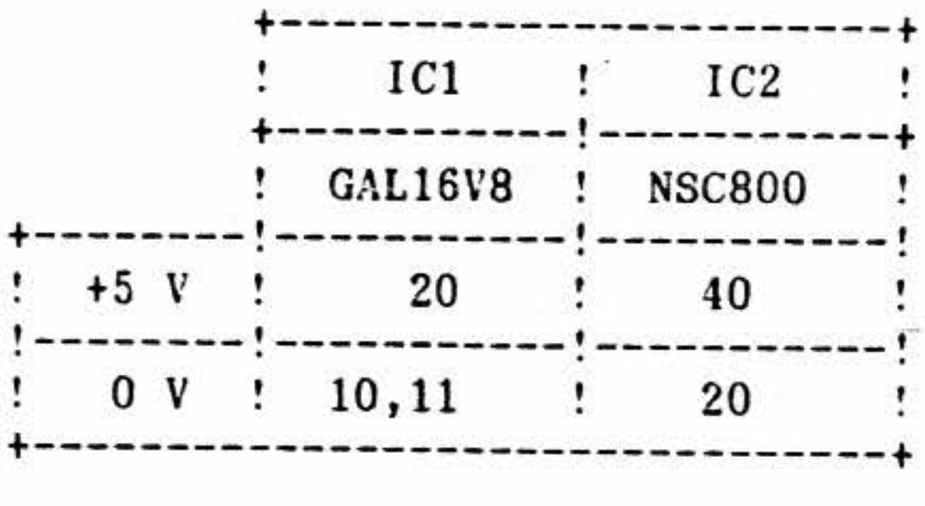
```
program  MOTOR_STEUERUNG;
(*$I UP.PAS *)
begin
   WARTE_AUF_START_TASTE;
   MOTOR_EINSCHALTEN;
   STARTE_LAUFZEIT;
   WARTE_AUF_LAUFZEIT_ENDE;
   MOTOR_AUSSCHALTEN;
end.
```

Durch die Definition von Unterprogrammen, die in einer Datei (UP.PAS) abgelegt sind, wird die Funktion des Programms direkt lesbar. Hiermit bietet sich die Möglichkeit, dem Lernenden eine Reihe von vordefinierten Programmteilen "unsichtbar" zur Verfügung zu stellen, die er aber nutzen kann. Damit lassen sich Aufgabenbeipiele in die Vermittlung einbinden, die programmiertechnisch zu kompliziert, aber für den Auszubildenden von Interesse sind. Verfolgt man dieses Konzept konsequent, so lassen sich auch Lernanfänger über einen anwendungsbezogenen Befehlssatz in Form dieser vordefinierten Unterprogramme in den Einsatz und die Handhabung von Computern einführen.
Turbo-PASCAL hat einen bildschirm-orientierten Editor integriert, der in seinen Funktionen dem Textverarbeitungsprogramm Wordstar entspricht. Das setzt allerdings ein Computer-System voraus, welches mit einem Daten-Terminal ausgestattet ist, das Steuerfunktionen für die Positionierung des Cursors auf dem Bildschirm ermöglicht. Diese Funktionen eines Terminals nennt man "direkte Cursor-Steuerung". Für das MFA-Mikrocomputer-System stehen mit dem Video-Interface BFZ/MFA 8.4 diese Funktionen zur Verfügung. Für den Einsatz von Turbo-PASCAL ist dieses Interface in der CP/M-Ausbaustufe erforderlich.
Bei dem nachfolgend abgebildeten Programmbeispiel, welches in der Programmiersprache Turbo-PASCAL realisiert wurde, handelt es sich wieder um den Zähler. Allerdings wurde hier die Interrupt-Verarbeitung so realisiert, daß die Interrupt-Service-Routine unter der Restart-Adresse 003CH lediglich aus dem RET-Befehl besteht und in dem Unterprogramm "warte_auf_interrupt" der Prozessor bis zu einem eintreffenden Interrupt-Impuls über einen HALT-Befehl gestoppt wird.

Anhang A1: Stromlaufplan Adapter-Platine



Anhang A1: Stromlaufplan Adapter-Platine



Anm.:

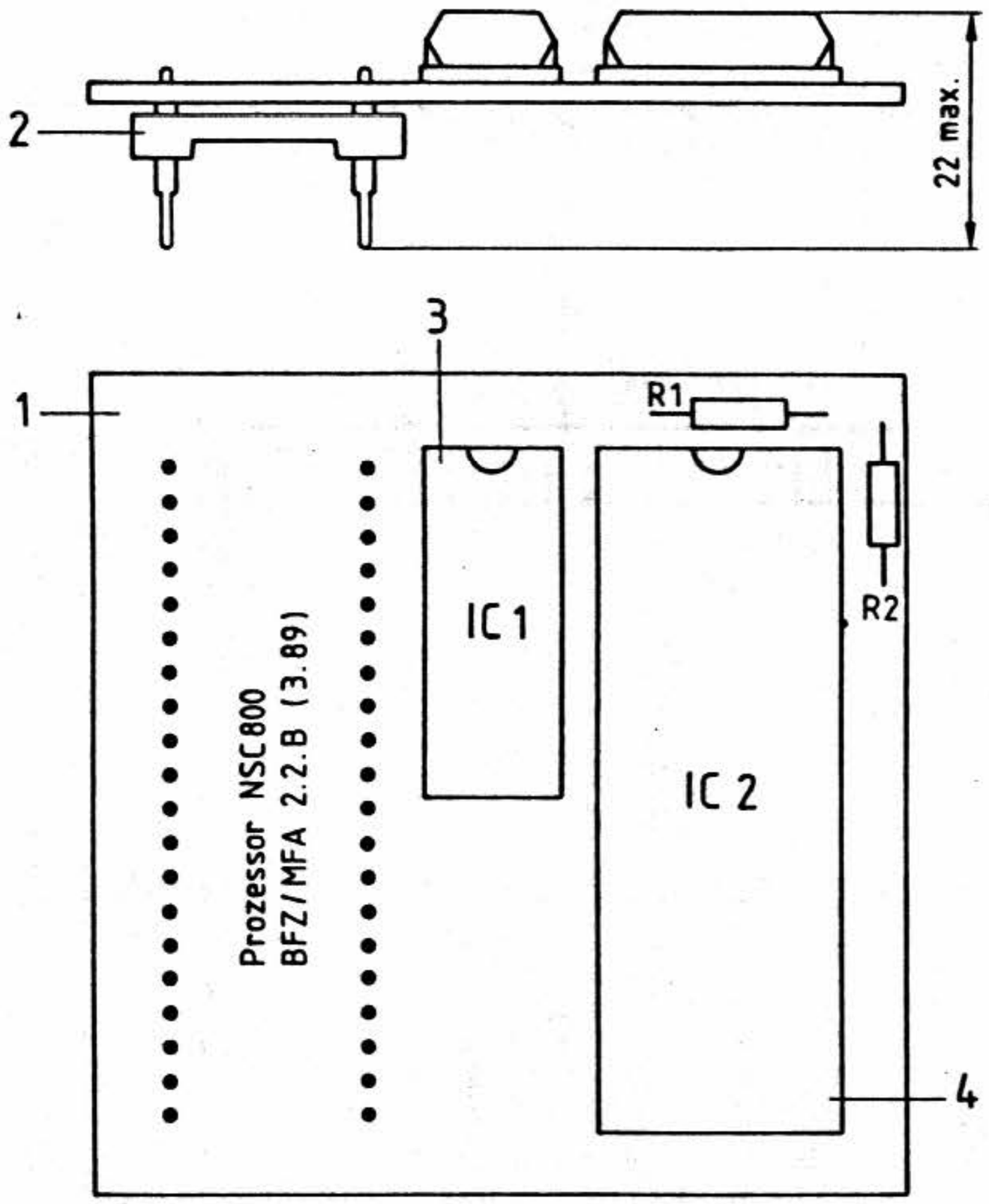
1) Der RST7.5-Eingang wirkt flankengesteuert; die Eingangsschaltung zum RSTA-Eingang ist vereinfacht dargestellt!

2) Der NMI-Eingang NSC800 ist aktiv Low, so daß gegenüber der 8085-CPU hier die Bezeichnung /TRAP angegeben ist!

Bauteilliste

Bez.	Art	Bemerkung
IC1	GAL 16V8	
IC2	Mikroprozessor NSC800	fmin 2MHz
R1,R2	Widerstand 4,7KOhm	0,25W

Anhang A2: Bestückungsplan Adapter-Platine



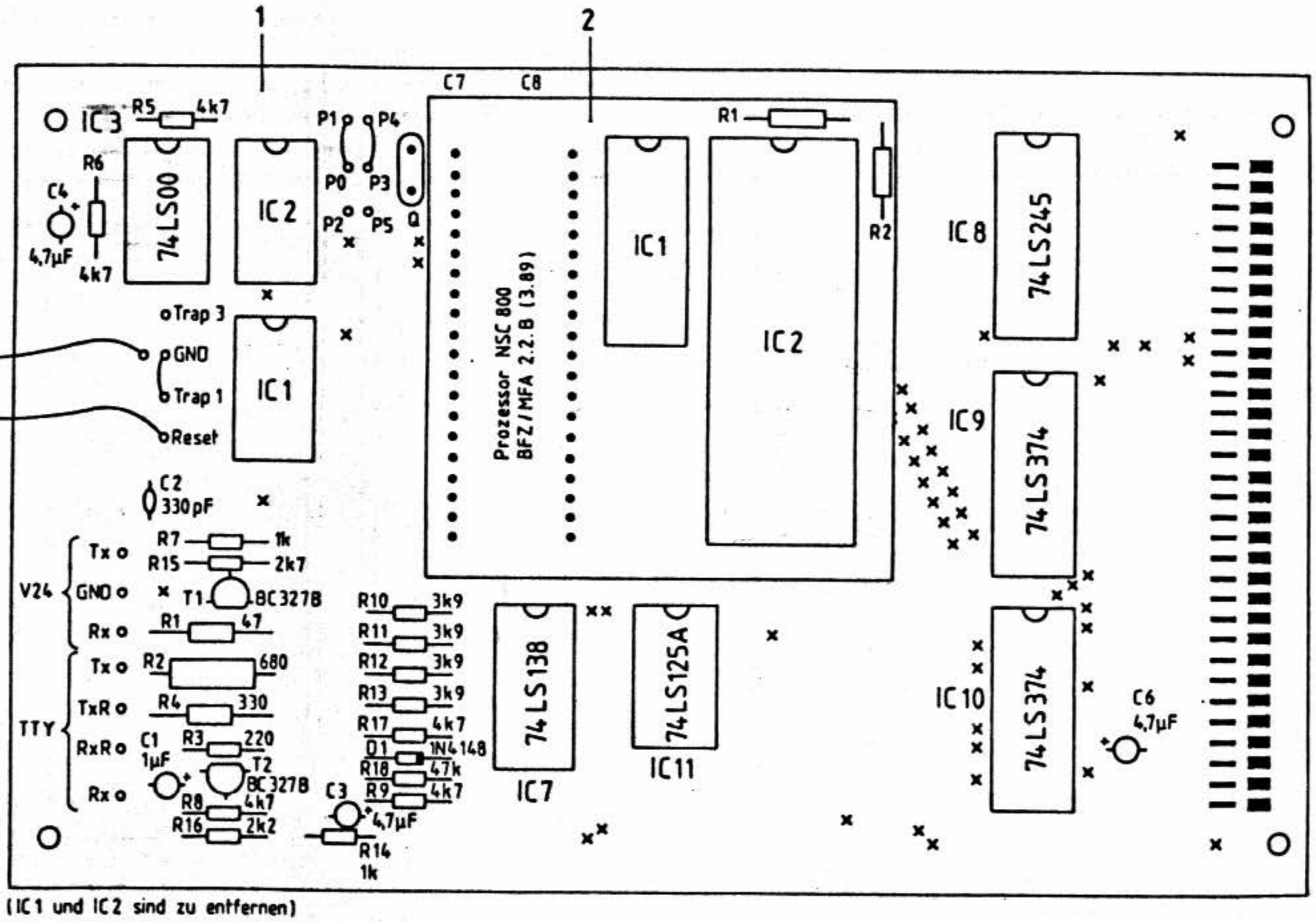
Stückliste Leiterplatte

Pos.	Stckz.	Benennung/Daten	Bemerkung
1	1	Leiterplatte BFZ/MFA 2.2	
2	1	Leiterplatte Dual-in-line-Verbindungsstecker 40polig	z.B. DILV4006ZX1 Fa. Fischer Metroplast
3	1	IC-Fassung 20polig	flache Bauform oder
4	1	IC-Fassung 40polig	unbestückt

Anhang A3: Bestückungsplan CPU-Baugruppe

Siehe auch: Fachpraktische Übung Band 4
CP/M-Ausbaustufe 1 BFZ/MFA 7.3.1
Aufbau und Inbetriebnahme

Seite 20: A5 - Sichtkontrolle/Modifikation der
Baugruppe Prozessor 8085



(IC1 und IC2 sind zu entfernen)

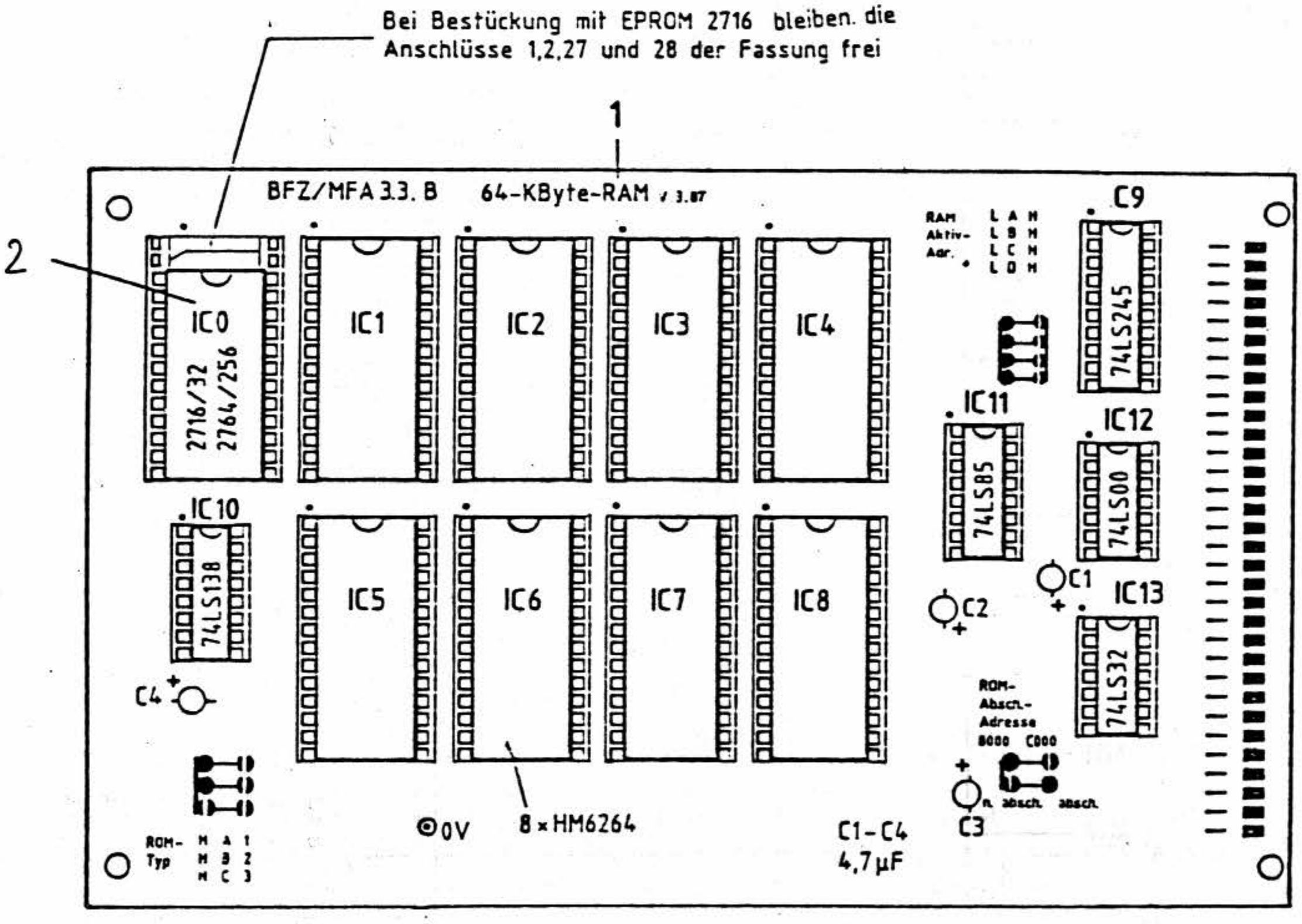
Stückliste

Pos.	Stckz.	Benennung/Daten	Bemerkung
1	1	Prozessor-Baugruppe BFZ/MFA 2.1	bestückt und geprüft
2	1	NSC800 Aapater-Platine BFZ/MFA 2.2	bestückt und geprüft, alle Lötbrücken geöffnet

Anhang A4: Bestückungsplan 64-K-RAM-Baugruppe

Siehe auch: Fachpraktische Übung Band 4
CP/M-Ausbaustufe 1 BFZ/MFA 7.3.1
Aufbau und Inbetriebnahme

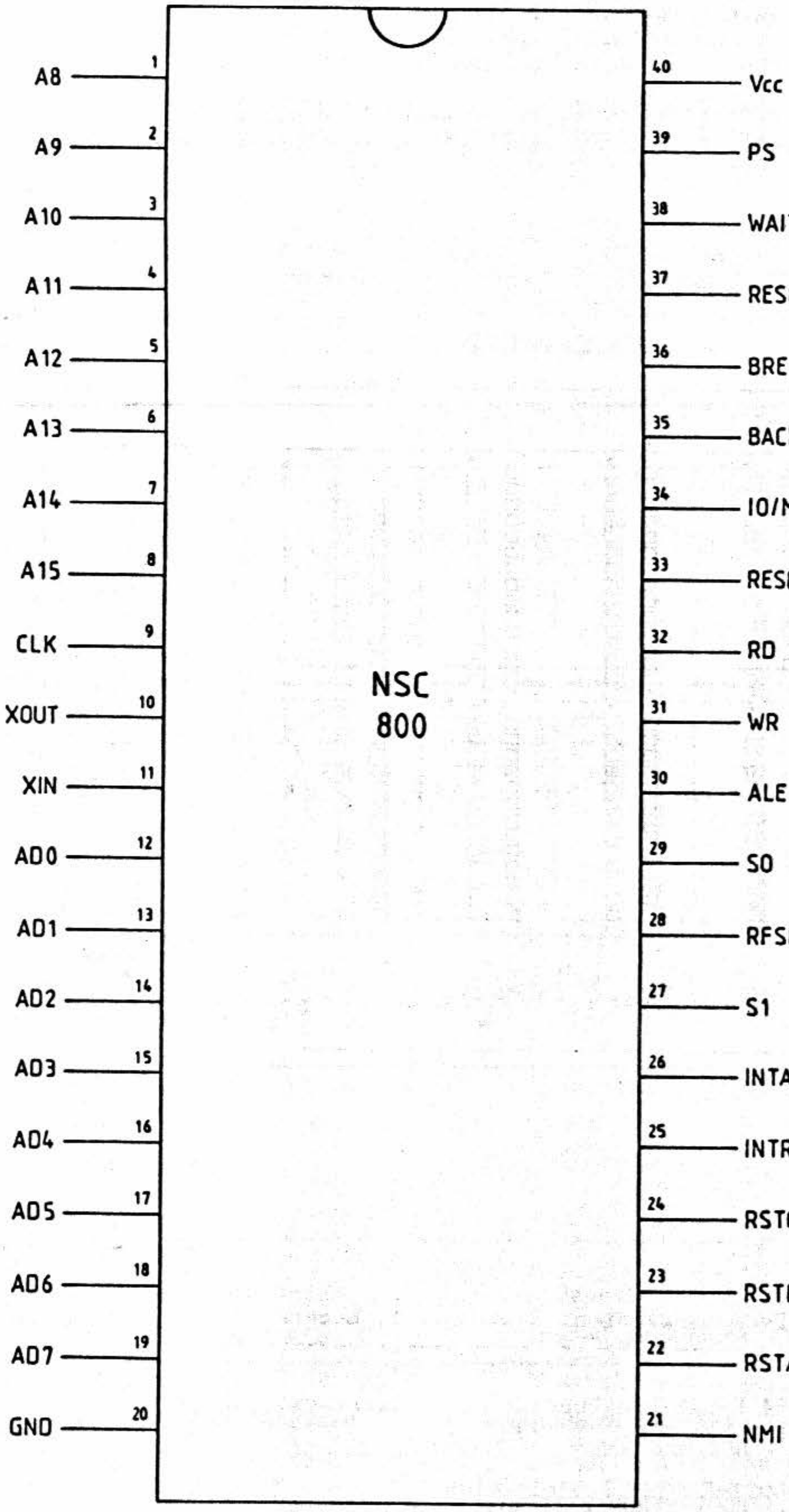
Seite 34: A10 - Sichtkontrolle 64-K-RAM-Baugruppe
Seite 30: A 9 - Sichtkontrolle 16-K-RAM/EPROM-Baugruppe



Stückliste

Pos.	Stckz.	Benennung/Daten	Bemerkung
1	1	64-K-RAM-Baugruppe BFZ/MFA 3.3	bestückt und geprüft
2	1	BOOT-EPROM 2716 mit BIOS-Programm, angepaßt an NSC800-Prozessor	

Anhang A5: Anschlußplan NSC800



Anhang A6: Befehlsliste NSC800/Z80

MNEMOCODE	WIRKUNG	ERLÄUTERUNGEN
8-Bit-Ladebefehle		
LD r,s	<r>:=<s>	<s> = <r>,d8,<<HL>>,<<xy+offs>>
LD d,r	<d>:=<r>	d = r,<HL>,<xy+offs>
LD d,d8	<d>:=d8	d = <HL>,<xy+offs>
LD A,s	<A>:=<s>	s = <BC>,<DE>,I,R,adr
LD d,A	<d>:=A	d = <BC>,<DE>,I,R,adr
16-Bit-Ladebefehle		
LD dd,d16	<dd>:=d16	dd = BC,DE,HL,SP,xy
LD dd,(adr)	<dd>:=<adr+1><adr>	
LD (adr),ss	<adr+1><adr>:=<ss>	ss = BC,DE,HL,SP,xy
LD SP,ss	<SP>:=<ss>	ss = HL,xy
PUSH ss	<<SP>-1>:=hi(<ss>)	ss = BC,DE,HL,AF,xy
	<<SP>-2>:=lo(<ss>)	
POP dd	lo(<dd>):=<<SP>>	dd = BC,DE,HL,AF,xy
	hi(<dd>):=<<SP>+1>	
Austauschbefehle		
EX DE,HL	<DE> <--> <HL>	
EX AF,AF'	<AF> <--> <AF'>	
EXX	<BCDEHL> <--> <BCDEHL'>	
EX (SP),ss	<<SP>> <--> lo(<ss>)	ss = HL,xy
	<<SP>+1> <--> hi(<ss>)	
Blocktransferbefehle		
LDI	<<DE>>:=<<HL>>,<ss>:=<ss>+1	Byte ss = HL,DE
	<BC>:=<BC>-1	
LDIR	<<DE>>:=<<HL>>,<ss>:=<ss>+1	Block
	<BC>:=<BC>-1 bis <BC>=0	
LDD	<<DE>>:=<<HL>>,<ss>:=<ss>-1	Byte
	<BC>:=<BC>-1	
LDDR	<<DE>>:=<<HL>>,<ss>:=<ss>-1	Block
	<BC>:=<BC>-1 bis <BC>=0	
Vergleichsbefehle		
CPI	<HL>:=<HL>+1,<BC>:=<BC>-1	Vergleich von <A> mit <<HL>>
CPIR	<HL>:=<HL>+1,<BC>:=<BC>-1	
	bis <BC>=0 oder <A>=<<HL>>	
CPD	<HL>:=<HL>-1,<BC>:=<BC>-1	<A>-<<HL>> beeinflußt Flags
CPDR	<HL>:=<HL>-1,<BC>:=<BC>-1	<A> bleibt unverändert
	bis <BC>=0 oder <A>=<<HL>>	

Anhang A6: Befehlsliste NSC800/Z80

8-Bit-ALU-Befehle

ADD	s	<A>:=<A>+<s>	<s>=<r>,d8,<<HL>>, auch ADD A,s
ADC	s	<A>:=<A>+<s>+<CY>	<<xy+offs>> auch ADC A,s
SUB	s	<A>:=<A>-<s>	auch SUB A,s
SBC	s	<A>:=<A>-<s>-<CY>	auch SBC A,s
AND	s	<A>:=<A> and <s>	bitweise Operation
OR	s	<A>:=<A> or <s>	
XOR	s	<A>:=<A> xor <s>	
CP	s	<A>:=<A>	<A>-<s> beeinflußt Flags
INC	d	<d>:=<d>+1	d = r,<HL>,<xy+offs>
DEC	d	<d>:=<d>-1	

16-Bit-ALU-Befehle

ADD	dd,ss	<dd>:=<dd>+<ss>	dd = xy, ss = BC,DE,SP
ADC	HL,ss	<HL>:=<HL>+<ss>+<CY>	ss = HL,BC,DE,SP
SBC	HL,ss	<HL>:=<HL>-<ss>-<CY>	
INC	ss	<ss>:=<ss>+1	ss = BC,DE,SP,xy
DEC	ss	<ss>:=<ss>-1	

Konvertierbefehle

DAA	BCD-Korrektur von A	
CPL	<A>:=</A>	</A> bitweise Negation
NEG	<A>:=</A>+1	
CCF	<CY>:=</CY>	
SCF	<CY>:=1	

Bitbefehle

BIT	b,s	<Z>:=b-tes Bit von s	s = r,<HL>,<xy+offs>
SET	b,s	b-tes Bit von s := 1	
RES	b,s	b-tes Bit von s := 0	

Steuerbefehle

NOP	keine Operation	<P>:=<PC>+1
HALT	HALT-Zustand	
DI	Interrupts sperren	
EI	Interrupts freigeben	
IM0	Interruptmode 0	8080-Modus
IM1	Interruptmode 1	INT --> CALL 0038H
IM2	Interruptmode 2	INT --> CALL <I><d8>
		<d8> vom IO-Baustein

Anhang A6: Befehlsliste NSC800/Z80

Verschiebebefehle

```
                        +---------------+          s = r,<HL>,<xy+offs>
RLC   s           CY<-+<-/...s..../<-+

                  +---------------------+
RL    s           +-CY--<--/...s..../<-+

                        +---------------+
RRC   s             CY<-+->/...s..../->+

                  +---------------------+
RR    s           +-CY-->--/...s..../->+

SLA   s             CY<----/...s..../<-0

                  +---------------------+
SRA   s           +->CY +->/...s..../->+
                        +---+
                  +---------------------+
SRL   s           +->CY 0->/...s..../->+
RLD                                          Rotieren von Tetraden
RRD
```

Ein- und Ausgabebefehle

IN	A,(p)	<A>:=<p>	
IN	r,(C)	<r>:=<<C>>	<C> = Port-Adresse
INI		<<HL>>:=<<C>>, <HL>:=<HL>+1	Byte
		<B>:=<B>-1	
INIR		<<HL>>:=<<C>>, <HL>:=<HL>+1	Block
		<B>:=<B>-1 bis <B>=0	
IND		<<HL>>:=<<C>>, <HL>:=<HL>-1	Byte
		<B>:=<B>-1	
INDR		<<HL>>:=<<C>>, <HL>:=<HL>-1	Block
		<B>:=<B>-1 bis <B>=0	
OUT	(p),A	<p>:=<A>	
OUT	(C),A	<<C>>:=<A>	
OUTI		<<C>>:=<<HL>>, <HL>:=<HL>+1	Byte
		<B>:=<B>-1	
OUTIR		<<C>>:=<<HL>>, <HL>:=<HL>+1	Block
		<B>:=<B>-1 bis <B>=0	
OUTD		<<C>>:=<<HL>>, <HL>:=<HL>-1	Byte
		<B>:=<B>-1	
OUTDR		<<C>>:=<<HL>>, <HL>:=<HL>-1	Block
		<B>:=<B>-1 bis <B>=0	

Anhang A6: Befehlsliste NSC800/Z80

Sprungbefehle

JP	adr	<PC>:=adr	
JP	cc,adr	wenn Bedingung erfüllt	cc = NZ,Z,NC,C,PO,PE, P,M
JR	offs	<PC>:=<PC>+offs	
JR	kk,offs	wenn Bedingung erfüllt	kk = NZ,NC,Z,C
JP	(ss)	<PC>:=<ss>	ss = HL,xy
DJNZ	offs	<B>:=<B>-1 und <PC>:=<PC>+offs wenn <B> ungleich 0	
CALL	adr	<<SP>-1><<SP>-2>:=<PC> und <PC>:=adr	
CALL	cc,adr	wenn Bedingung erfüllt	
RST	n	<<SP>-1><<SP>-2>:=<PC> und <PC>:=n*8	n = 0,1,2,3,4,5,6,7
RET		<PC>:=<<SP>><<SP>+1>	
RET	cc	wenn Bedingung erfüllt	
RETI		Rückkehr von Interrupt Int.-Flipflop stellen	
RETN		Rückkehr vom NMI	

Verwendete Kurzzeichen:

< >	= Inhalt von	d8	= 8-Bit-Wert
:=	= Wertzuweisung	d16	= 16-Bit-Wert
r	= A,B,C,D,E,H,L	adr	= Adresse
xy	= IX oder IY	hi( )	= High-Byte
offs	= Offset	lo( )	= Low-Byte
p	= Port-Adresse		

Anhang A7: Z80-Assemblerprogramm-Listing

```
                        ;-----------------------------------------
                        ;       NSC800-/Z80 PROGRAMM-BEISPIEL
                        ;            Z A E H L E R
                        ;
                        ; ASSEMBLIERT MIT DEM M80-ASSEMBLER
                        ;
                        ; BENOETIGTE BAUGRUPPEN/KONFIGURATION:
                        ;
                        ; CP/M-AUSBAUSTUFE DES MFA-SYSTEMS
                        ; MIT NSC800-ADAPTER-PLATINE
                        ; SOWIE:
                        ; ZAEHLER UND ZEITGEBER    BFZ/MFA 4.6
                        ; 8-BIT-AUSGABE           BFZ/MFA 4.1
                        ;
                        ; FUNKTION:
                        ; IM RHYTHMUS VON 1 SEKUNDE WIRD DAS
                        ; AUSGABE-BIT-MUSTER AN DER AUSGABE-
                        ; BAUGRUPPE INTERRUPT-GESTEUERT HOCH-
                        ; GEZAEHLT
                        ;
                        ; AUTOR: F. DERRIKS 5/1989
                        ;-----------------------------------------

                        .Z80                        ;Z80-MNEMONIK

003C                    INT75   EQU     003CH       ;RSTA-/RST7.5-EINSPRUNG-
                                                    ;ADRESSE

0063                    TIMSTWR EQU     063H        ;TIMER-BAUGRUPPEN-NUMMERN
0060                    TIM0    EQU     060H
0061                    TIM1    EQU     061H

0036                    TSTW0   EQU     036H        ;TIMER-STEUERWORTE
0076                    TSTW1   EQU     076H
                                                    ;TEILER-FAKTOR TIMER 0
00D0                    TL0LB   EQU     0D0H        ;07D0H -> 2000:1
0007                    TL0HB   EQU     007H        ;1 MSEC -  IMPULSE
                                                    ;TEILER-FAKTOR TIMER 1
00E8                    TL1LB   EQU     0E8H        ;03E8H -> 1000:1
0003                    TL1HB   EQU     003H        ; 1 SEC -  IMPULSE


                        ;-----------------------------------------
                        ; DAS HAUPT-PROGRAMM BESTEHT LEDIGLICH AUS DEM
                        ; AUFRUF DES INITIALISIERUNGS-PROGRAMMS, DER
                        ; INTERRUPT-FREIGABE SOWIE EINER PROGRAMM-SCHLEIFE,
                        ; BEI DER DER PROZESSOR BIS ZUM EINTREFFENDEN
                        ; INTERRUPT-IMPULS VERHARRT,
                        ; NACH AUSFUEHRUNG DER INTERRUPT-SERVIVE-
                        ; ROUTINE WIRD DIE SCHLEIFE ERNEUT GESTARTET

0000'   CD 0012         START:  CALL    INIT        ;INITIALISIERUNG
0003'   FB                      EI
0004'   C3 0004         LOOP:   JP      LOOP        ;WARTE BIS INTERRUPT
```

Anhang A7: Z80-Assemblerprogramm-Listing

```
                        ;-----------------------------------------
                        ; INTERRUPT-SERVIVE-ROUTINE, DIE IM RHYTHMUS DER
                        ; INTERRUPT-IMPULSE VOM TIMER ANGESPRUNGEN UND
                        ; AUSGEFUEHRT WIRD

0007'   3A 004A     ISR:    LD      A,(ZAEHLER)     ;ALTEN ZAEHLWERT
000A'   3C                  INC     A               ;INKREMENTIEREN
000B'   32 004A             LD      (ZAEHLER),A     ;UND SICHERN
000E'   D3 00               OUT     (0),A           ;SOWIE ANZEIGEN
0010'   FB                 EI                      ;INTERRUPTS FREIGEBEN
0011'   C9                  RET                     ;UND ZURUECK ZUM HAUPT-
                                                    ;PROGRAMM

                        ;-----------------------------------------
                        ; INITIALISIERUNGS-PROGRAMM FUER INTERRUPT, TIMER-
                        ; BAUGRUPPE UND RAM-ZAEHLER

0012'   CD 0021     INIT:   CALL    INTIN   ;INTERRUPT-INITIALISIERUNG
0015'   CD 0031            CALL    TIMIN   ;TIMER-INITIALISIERUNG
0018'   CD 001C            CALL    INITZ   ;ZAEHLER-INITIALISIERUNG
001B'   C9                  RET

                        ;-----------------------------------------
                        ; INITIALISIERUNG DES RAM-ZAEHLERS, DER DAS
                        ; AUSGANGS-BITMUSTER SPEICHERT

001C'   AF          INITZ:  XOR     A
001D'   32 004A             LD      (ZAEHLER),A
0020'   C9                  RET

                        ;-----------------------------------------
                        ; INITIALISIERUNG DES RSTA-/RST7.5-INTERRUPTS
                        ; DURCH SETZEN DES MASKEN-BITS SOWIE DURCH EIN-
                        ; TRAGEN DES SPRUNGBEFEHLS ZUR INTERRUPT-SERVIVE-
                        ; ROUTINE IM SPEICHER UNTER DER ZUM RSTA GEHOEREN-
                        ; DEN RESTART-ADRESSE 003CH

0021'   3E 08       INTIN:  LD      A,08H           ;INT.-MASKE FUER
0023'   D3 BB              OUT     (0BBH),A        ;RSTA/RST7.5
0025'   3E C3              LD      A,0C3H          ;JMP-BEFEHL NACH
0027'   32 003C            LD      (INT75),A       ;003CH
002A'   21 0007            LD      HL,ISR
002D'   22 003D            LD      (INT75+1),HL
0030'   C9                 RET
```

Anhang A7: Z80-Assemblerprogramm-Listing

```
                        ;-----------------------------------------------
                        ; INITIALISIERUNG DER ZAEHLER- UND ZEITGEBER-BAU-
                        ; GRUPPE, VERWENDET WERDEN DER TIMER 0 UND 1, UM
                        ; DEN 2 MHZ - SYSTEMTAKT AUF 1 HZ HERUNTERZUTEILEN,
                        ; DAZU MUESSEN BEIDE TEILER HINTEREINANDER GESCHAL-
                        ; TET WERDEN, INDEM DER AUSGANG DES TIMERS 0 (OUT0)
                        ; MIT DEM EINGANG DES TIMER 1 (CLK 1) VERBUNDEN
                        ; WIRD
                        ; DARUEBER HINAUS MUSS DER AUSGANG DES TIMERS 1 UEBER
                        ; STECKER St1 UND St2 AUF DER BAUGRUPPE MIT DER
                        ; INTERRUPT-LEITUNG RST7.5 (STIFT 25a) DES BUS-
                        ; SYSTEMS VERBUNDEN WERDEN
                        ; IM TESTFALL KANN DER TIMER-IMPULS VON HAND
                        ; AUSGELOEST WERDEN, INDEM UEBER DEN STECKER St3
                        ; DER HAND-INTERRUPT SELEKTIERT WIRD

0031'  3E 36    TIMIN:  LD      A,TSTW0         ;TIMER 0 AUF 1 MSEC-
0033'  D3 63            OUT     (TIMSTWR),A     ;IMPULSE
0035'  3E D0            LD      A,TL0LB
0037'  D3 60            OUT     (TIM0),A
0039'  3E 07            LD      A,TL0HB
003B'  D3 60            OUT     (TIM0),A
003D'  3E 76            LD      A,TSTW1         ;TIMER 1 AUF 1 SEC-
003F'  D3 63            OUT     (TIMSTWR),A     ;IMPULSE
0041'  3E E8            LD      A,TL1LB
0043'  D3 61            OUT     (TIM1),A
0045'  3E 03            LD      A,TL1HB
0047'  D3 61            OUT     (TIM1),A
0049'  C9               RET

                        ;-----------------------------------------------
                        ; RAM-SPEICHERSTELLE, IN DER DIE ANZAHL DER EINGEGAN-
                        ; GENEN INTERRUPT-IMPULSE FESTGEHALTEN WIRD

004A'  00       ZAEHLER: DEFB   0

                        END

Symbols:
INIT    0012'   INITZ   001C'   INT75   003C    INTIN   0021'
ISR     0007'   LOOP    0004'   START   0000'   TIM0    0060
TIM1    0061    TIMIN   0031'   TIMSTW  0063    TL0HB   0007
TL0LB   00D0    TL1HB   0003    TL1LB   00E8    TSTW0   0036
TSTW1   0076    ZAEHLE  004A'

No  Fatal error(s)
```

Anhang A8: Turbo-PASCAL-Programmlisting

```
(*-------------------------------------------------------*)
(*   Turbo-PASCAL  Programm-Beispiel: Interrupt-gesteuerter  *)
(*   Zaehler, im 1 Sekunden-Rhythmus ueber Timer-Interrupt-  *)
(*   Impuls am RSTA-(RST7.5) Eingang des Prozessors          *)
(*   Autor: F.Derriks 5/1989                                 *)
(*-------------------------------------------------------*)
  program zaehler(input,output);
(*-------------------------------------------------------*)
        const
           output_0      = 0;
        var
           zaehler_stand : byte;

(*------------ U n t e r p r o g r a m m e ----------------*)
  procedure interrupt_init;
(*-------------------------------------------------------*)
        const
           int_contl_reg   = $BB;
           int_maske       = $08;
           restart_adr     = $003C;
           ret_instr_code  = $C9;
        begin
           mem[restart_adr]    := ret_instr_code;
           port[int_contl_reg] := int_maske;
        end;

(*-------------------------------------------------------*)
  procedure timer_init;
(*-------------------------------------------------------*)
        const
           timer_contl_reg = $63;
           timer_0         = $60;
           timer_1         = $61;
           timer_stw_0     = $36;
           timer_stw_1     = $76;
           teiler_0        = 2000;
           teiler_1        = 1000;
        begin
           port[timer_contl_reg] := timer_stw_0;
           port[timer_0]         := lo(teiler_0);
           port[timer_0]         := hi(teiler_0);
           port[timer_contl_reg] := timer_stw_1;
           port[timer_1]         := lo(teiler_1);
           port[timer_1]         := hi(teiler_1);
        end;

(*-------------------------------------------------------*)
  procedure warte_auf_interrupt;
(*-------------------------------------------------------*)
        begin
           inline ($FB/        (* EI   *)
                   $76);       (* HALT *)
        end;
```

Anhang A8: Turbo-PASCAL-Programmlisting

```
(*------------ H a u p t p r o g r a m m --------------------*)
          begin;
             timer_init;
             interrupt_init;
             zaehler_stand := 0;
             repeat
                warte_auf_interrupt;
                port[output_0] := zaehler_stand;
                zaehler_stand  := zaehler_stand + 1;
          until keypressed;
end.
```

Anhang A9: Anpassen des Programms BFZFORM.COM an den NSC800 An

Das Formatierungs-Programm BFZFORM.COM zum MFA-CP/M-System muß an den Prozessor NSC800 angepaßt werden. Diese Anpassung kann mit Hilfe des CP/M-Debuggers DDT durchgeführt werden. Die auszuführenden Schritte sind im folgenden protokolliert, wobei zur Änderung des Programms das S-Kommando verwendet wurde.

```
A>DDT BFZFORM.COM          <---- Aufruf des DDT und Laden des Programms
DDT VERS 2.2                     BFZFORM.COM in den TPA-Bereich
NEXT  PC
0D80 0100
-S066A                     <---- Änderungen bei Adresse 066A beginnen
066A 3E CD                 <---- der jeweilige Inhalt der Speicherstelle
066B 0E 5D                       wird vom DDT angezeigt, die Eingabe-
066C 30 0C                       Daten überschreiben diesen
066D E1 .                  <---- Abbruch der Änderung
-S0C4C                     <---- erneute Änderung ab Adresse 0C4C
0C4C 46 4E
0C4D 4F 49
0C4E 52 43
0C4F 4D 48
0C50 41 54
0C51 54 20
0C52 49 46
0C53 45 4F
0C54 52 52
0C55 45 4D
0C56 4E 41
0C57 20 54
0C58 4E 49
0C59 49 45
0C5A 43 52
0C5B 48 54
0C5C 54 24
0C5D 20 3E
0C5E 4D 02
0C5F 4F D3
0C60 45 BB
0C61 47 C9
0C62 4C 00
0C63 49 00
0C64 43 00
0C65 48 00
0C66 24 00
0C67 5A .                  <---- Abbruch der Änderung
-^C                        <---- Rückkehr zum Betriebssystem
A>SAVE 13 NSCFORM.COM      <---- Speichern des geänderten Programms
A>                               unter dem Namen NSCFORM.COM
```

Anhang A10: Literaturhinweise

NSC800 Microprocessor Family
Databook
National Semiconductor Corporation

MSC-80/85 Family User's Manual
Intel Corporation

Mikrocomputer Bausteine
Mikroprozessor-System SAB 8085
Siemens Datenbuch

Microsoft Utility Software Package
for CP/M 80
Assembler, Linker, Cross Referenz
Facility, Library Manager

Z80 Assembly Language Subroutines
L.A. Leventhal, W. Saville
Osborne/McGraw-Hill

Turbo-PASCAL Reference Manual
Borland International
Scotts Valley, California

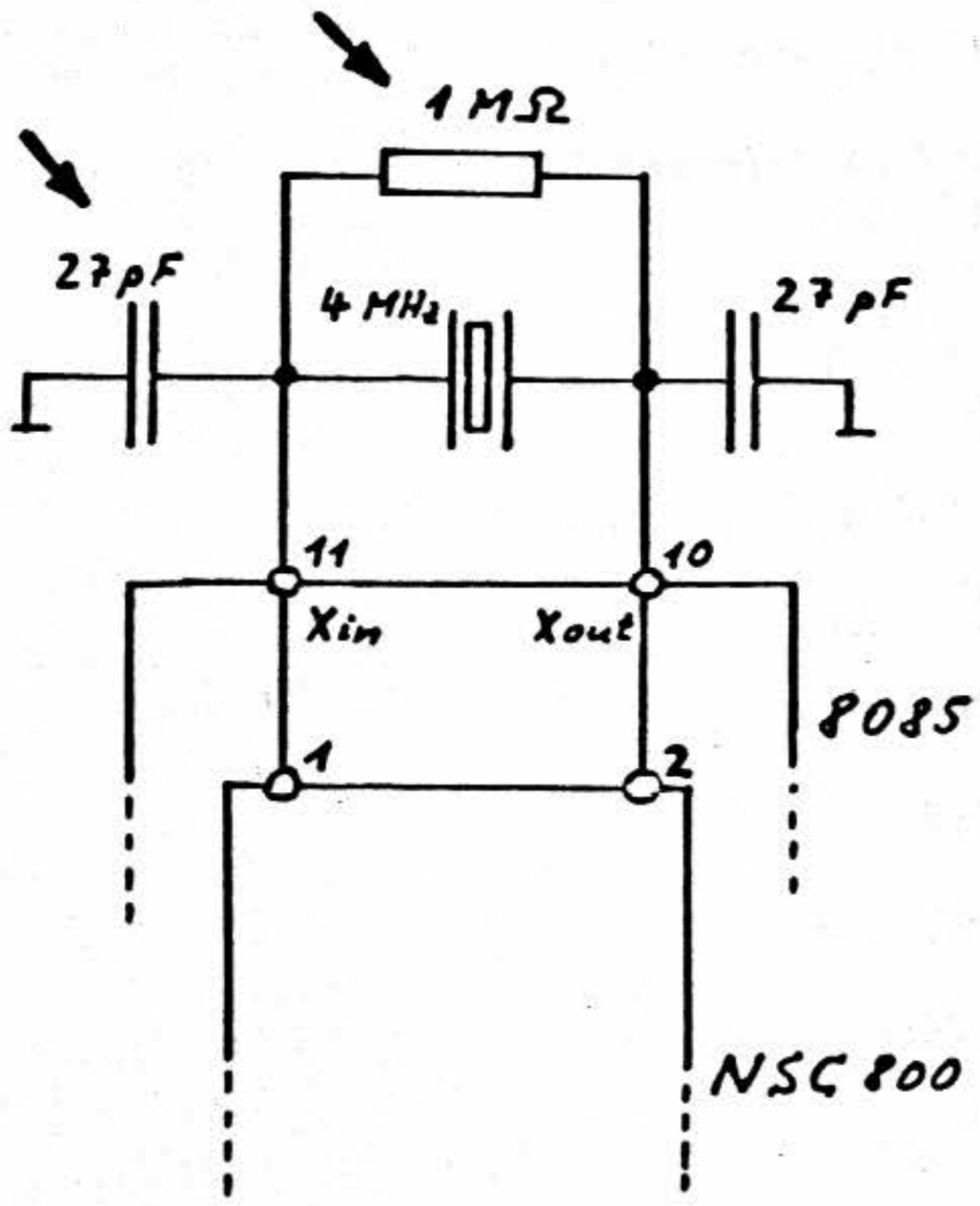
Programmierung des Z80
Rodnay Zaks
Sybex-Verlag, Düsseldorf

Turbo-PASCAL Top Training
Jürgen Handke
Klett-Verlag, Stuttgart

Abhilfe bei Startproblemen

In Einzelfällen hat es sich gezeigt, daß der Oszillator des NSC800-Prozessors nach dem Einschalten der Betriebsspannung Anschwingprobleme hat, so daß zum Start nochmals die Reset-Taste betätigt werden muß.

In hartnäckigen Fällen ist Abhilfe auf folgende Art möglich: Die beiden mitgelieferten Teile (ein Kondensator mit 27 pF und ein Widerstand mit 1 MOhm) werden direkt am 2-MHz-Quarz auf die Prozessor-Hauptplatine gelötet, wie in der Skizze dargestellt:



Tefa Electronic Vorm.
Fenstor 7
064 7791 2020 | 35753 Greifenstein
Wendroth

– Ihr Partner für Ausbildungssysteme

Neue Produkte im Bereich Mediensysteme

Mit Medien, Aus- und Weiterbildung begann sich die vgs bereits vor 20 Jahren mit dem Begleitbuch zur Fernsehreihe „Einführung in die Elektronik" von Jean Pütz zu beschäftigen. Über die Jahre hinweg entstanden in Zusammenarbeit mit ARD und ZDF weitere Bücher und Bausätze aus den Bereichen Elektronik, Chemie und Biotechnologie. Inzwischen ist die vgs der Spezialist für Medienverbund auf allen Gebieten. Die Zusammenarbeit mit dem Berufsförderungszentrum Essen e.V. begann 1983, als die vgs den Zuschlag für die Produktion und Verbreitung des dort entwickelten Mikrocomputer-Ausbildungssystems MFA erhielt. Inzwischen umfaßt das Programm der vgs für den Bereich der Aus- und Weiterbildung folgende Produkte:

- **MFA** – ein Mikrocomputer-Ausbildungssystem, das in den industriellen Metall- und Elektro-Berufen sowie in den handwerklichen Elektro-Berufen im In- und Ausland eine zentrale Rolle spielt. Das vom BFZ in Essen entwickelte Grundsystem wird von der vgs ständig zukunftsorientiert weiterentwickelt, produziert und vertrieben, so daß inzwischen ca. 90 Baugruppen zur Verfügung stehen.

- **40 900 NORMCOMPUTER** – ein Computer-Lehrsystem für die Schulung im Bereich Digitaltechnik, bei dem Wert auf die Einhaltung der gültigen internationalen Digitalnorm nach DIN 40 900 gelegt wurde (in Zusammenarbeit mit dem BBZ Köln). Mit diesem, aus 8 Lehrplatten bestehendem Lehrsystem, kann die immer noch bestehende Lücke zwischen Digitaltechnik und Mikrocomputertechnik geschlossen werden. Basis des NORM-COMPUTERS ist der in der Aus- und Weiterbildung nach wie vor optimal einzusetzende Mikrocomputer 8085.

- **PC-Modelle** für die Schulung – hier bietet die vgs Anwendungsmodelle aus den Bereichen Logikanalyse, Meßtechnik und Robotik an.

- **PTQ** (steht für Produktionstechnische Qualifikation im Lernverbund). An dieser neuen Entwicklung vom BFZ Essen ist die vgs als Werkvertragsnehmer über die Konstruktion und Dokumentation sowie Fertigungsarbeiten für „Portallader-Komponenten" beteiligt.

Die vgs liefert in das Inland (inzwischen mit Schwerpunkt neue Bundesländer) und über Vertragspartner in das Ausland an Industrie, Handwerk, Ausbildungszentren und Schulen.

Bitte fordern Sie weitere Informationen an bei:

vgs verlagsgesellschaft mbH & Co. KG
Postfach 18 02 69
Breite Straße 118-120, 5000 Köln 1

Telefon 02 21/2 08 11-12
Telefax 02 21/24 57 99
Telex 888 2202 vgs d

ISBN 3-8025-1260-X